

**โรงเพาะเห็ดอัจฉริยะ**

**Smart Mushroom Farm**

**นายนรินทร สอนชัยภูมิ**

**นายวรรรณณรงค์ สถิตวิทยกูล**

**ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต**

**สาขาวิชาวิศวกรรมคอมพิวเตอร์**

**คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลอีสาน วิทยาเขตขอนแก่น**

**พ.ศ. 2563**

**ลิขสิทธิ์คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลอีสาน**

โรงเพาะเห็ดอัจฉริยะ

นายนรินทร สอนชัยภูมิ

นายวรรรณณรงค์ สถิตวิทยกูล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลอีสาน วิทยาเขตขอนแก่น

พ.ศ. 2563

ลิขสิทธิ์คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลอีสาน

Smart Mushroom Farm

Narinthon Sonchaiyaphum

Wannarong Satitwittayakul

A Report Submitted in Partial Fulfillment of the Requirements for

the Degree of Bachelor of Engineering

Department of Computer Engineering

Faculty of Engineering

Rajamangala University of Technology Isan, Khon Kaen Campus

2020

© Faculty of Engineering, Rajamangala University of Technology Isan

หัวข้อปริญญานิพนธ์ โรงเพาะเห็ดอัจฉริยะ

จัดทำโดย นรินทร สอนชัยภูมิ และวรรรณณรงค์ สถิตวิทยกูล

ปีที่สำเร็จการศึกษา พุทธศักราช 2563

สาขาวิชา วิศวกรรมคอมพิวเตอร์

ที่ปรึกษา อาจารย์ ดร.เศวษ หงษ์ประสิทธิ์

**บทคัดย่อ**

ปริญญานิพนธ์นี้วัตถุประสงค์เพื่อออกแบบและสร้างโรงเพาะเห็ด ที่สามารถควบคุมอุณหภูมิความชื้นและแสงสว่างภายในโรงเรือนให้เหมาะสมต่อการออกดอกของเห็ด ออกแบบและสร้างเว็บแอปพลิเคชันและวินโดว์แอปพลิเคชัน ที่สามารถมอนิเตอร์ค่าสภาพแวดล้อมและควบคุมระบบภายในโรงเรือนได้ และได้ประยุกต์ใช้งาน Lora Communication เพื่อเป็นตัวกลางการสื่อสารระหว่างโรงเรือนเพาะเห็ด (Mushroom Node) และพื้นที่ที่มีสัญญาณอินเทอร์เน็ต (STA Node) เพื่อแก้ปัญหาที่ตั้งของโรงเรือนไม่มีสัญญาณอินเทอร์เน็ต และผู้จัดทำได้ประยุกต์ใช้งานตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) ตรวจว่าอุปกรณ์ทำงานจริงตามที่ควบคุมหรือไม่ เพื่อให้การแสดงค่าสถานการณ์ทำงานของอุปกรณ์ต่างๆที่ถูกต้อง

การทดลองเพาะเห็ดเป็นการทดสอบประสิทธิภาพของโรงเรือนเพาะเห็ดอัจฉริยะในการทดลองเพาะเห็ดจริง โดยใช้เห็ดนางฟ้าในการทำการทดลอง จากการทดลองเพาะเห็ดนางฟ้าระหว่างวันที่ 8 ตุลาคม พ.ศ. 2563 ถึงวันที่ 21 ตุลาคม พ.ศ. 2563 รวมทั้งหมด 14 วัน สามารถสรุปผลการทดลองได้ดังนี้ การเพาะเห็ดในโรงเรือนที่มีการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ด มีความกว้างของดอกเฉลี่ยเท่ากับ 7.66 เซนติเมตร มีน้ำหนักทั้งหมดเท่ากับ 1.06 กิโลกรัม การเพาะเห็ดในโรงเรือนที่ไม่มีกาการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยเท่ากับ 6.45 เซนติเมตร มีน้ำหนักทั้งหมดเท่ากับ 0.6 กิโลกรัม

Project Title Smart Mushroom Farm

Proposed by Narinthon Sonchaiyaphum and Wannarong Satitwittayakul

Year 2020

Department Computer Engineering

Project Advisor Dr. Saweth Hongprasit

**Abstract**

The thesis aims to designs and construct a mushroom hatchery, That can control the temperature, humidity and light within the hatchery to suit the flowering of mushroom. Design and build web applications and window applications that can monitor the environment and control the hatchery system and have applied for LoRa communication to serve as a communication medium between Mushroom Mode and internet area (STA Node) to solve the problem of house location without internet signal, And organizer can apply Current Sense Resistors to check whether the device is actually working as it is controlled or not. To provide the correct display of the working situation of various devices.

The mushroom cultivation experiment is a test of the effectiveness of intelligent mushroom hatchery in real mushroom cultivation experiments. Using fairy mushrooms to conduct experiments. From the mushroom cultivation experiment between October 8, 2020 and October 21, 2020, a total of 14 days can be summarized as follows. Cultivation of mushroom in a hatchery that is environmentally controlled to suit mushroom flowering. The average flower width was 7.66 cm with the total weight of 1.06 kg. Mushroom cultivation in the hatchery without environmental control to suit mushroom flowering with the average flower width of 6.45 cm with total weight of 0.6 kg.

**กิตติกรรมประกาศ**

ปริญญานิพนธ์ฉบับนี้ได้รับความช่วยเหลืออย่างดียิ่งจาก ดร.เศวษ หงส์ประสิทธิ์ อาจารย์ที่ปรึกษาปริญญานิพนธ์ ซึ่งกรุณาให้คำแนะนำและถ่ายทอดความรู้ ตลอดจนควบคุมการทำโครงงานจนประสบความสำเร็จ ผู้ทำโครงงานขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ขอกราบขอบพระคุณ คณาจารย์มหาวิทยาลัยเทคโนโลยีราชมงคลอีสาน วิทยาเขตขอนแก่น ที่ได้ประสิทธิประสาทความรู้และคณะตลอดเวลาที่ศึกษาเล่าเรียน

ขอขอบพระคุณคุณพ่อ คุณแม่และครอบครัวที่ข้าพเจ้ารักที่คอยเป็นกำลังใจให้ข้าพเจ้าและสนับสนุนส่งเสริมการศึกษามาโดยตลอด

ขอขอบพระคุณ เพื่อนๆ ทุกท่านที่ได้ให้การช่วยเหลือในการทำโครงงาน การเรียน การทำงานและมิตรภาพที่ดีแก่กันเสมอมา

ขอขอบพระคุณ ทุกท่านที่มิได้เอ่ยนามเป็นอย่างยิ่ง ในความกรุณาและความช่วยเหลือ ในด้านต่าง ๆ ที่ทำให้โครงงานนี้สำเร็จได้

ประโยชน์และคุณค่าอันพึงมีจากปริญญานิพนธ์ฉบับนี้ ผู้ทำโครงงานขอมอบเป็นกตัญญูตาบูชาแด่ บิดามารดา ครูอาจารย์ตลอดผู้มีพระคุณทุกท่าน

นรินทร สอนชัยภูมิ

วรรรณณรงค์ สถิตวิทยกูล

**สารบัญ**

|  |  |  |
| --- | --- | --- |
|  |  | **หน้า** |
| **บทคัดย่อ** | | ค |
| **กิตติกรรมประกาศ** | | ง |
| **สารบัญตาราง** | | ซ |
| **สารบัญรูป** | | ฌ |
|  | |  |
| **บทที่** | บทนำ | 1 |
| 1. | 1.1 ความเป็นมาและความสำคัญของปัญหา | 1 |
|  | 1.2 วัตถุประสงค์ | 2 |
|  | 1.3 ขอบเขต | 2 |
|  | 1.4 ประโยชน์ที่คาดว่าจะได้รับ | 3 |
| 2. | ทฤษฎีและงานวิจัยที่เกี่ยวข้อง | 4 |
|  | 2.1 ทฤษฎีที่เกี่ยวข้อง | 4 |
|  | 2.1.1 ไมโครคอนโทรลเลอร์ ESP32 LoRa | 4 |
|  | 2.1.2 Raspberry Pi 3 Model B | 6 |
|  | 2.1.3 AM2315 เซ็นเซอร์วัดอุณหภูมิและความชื้น | 7 |
|  | 2.1.4 BH1750FVI เซนเซอร์วัดความเข้มแสง | 18 |
|  | 2.1.5 4-Channel Relay Module | 19 |
|  | 2.1.6 จอแอลซีดี (I2C LCD) | 22 |
|  | 2.1.7 เครื่องปั๊มน้ำ (Pump) | 23 |
|  | 2.1.8 พัดลม (Fan) | 25 |
|  | 2.1.9 LoRa (Long Range) | 26 |
|  | 2.1.10 การเชื่อมต่ออุปกรณ์แบบ I2C (I2C) | 30 |
|  | 2.1.11 วงจรแปลงสัญญาณอนาล็อกเป็นดิจิตอล (Analog to Digital Converter) | 33 |
|  | 2.1.12 Pulse Width Modulation (PWM) | 37 |
|  | 2.1.13 อุณหภูมิ (Temperature) | 39 |
|  | 2.1.14 ความชื้น (Humidity) | 40 |
|  | 2.1.15 แสงสว่าง (Light) | 42 |
| **สารบัญ (ต่อ)** | | |
|  | | |
|  | 2.1.16 ตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) | 43 |
|  | 2.1.17 วงจรออปแอมป์ (Op-Amp Circuit) | 45 |
|  | 2.1.18 เว็บแอปพลิเคชัน (Web Application) | 49 |
|  | 2.1.19 วินโดว์แอปพลิเคชัน (Windows Application) | 51 |
|  | 2.1.20 ซอฟต์แวร์ที่เกี่ยวข้อง | 52 |
|  | 2.2 งานวิจัยที่เกี่ยวข้อง | 64 |
|  | 2.2.1 ไอโอทีแพลทฟอร์มโรงเพาะเห็ด | 64 |
|  | 2.2.2 IOT for Smart Farm: A case study of the Lingzhi Mushroom Farm at Maejo University | 64 |
|  | 2.2.3 การประยุกต์ใช้เทคโนโลยีไอโอทีควบคุมฟาร์มอัจฉริยะในโรงเรือนเพาะเห็ดนางฟ้า | 64 |
|  | 2.2.4 IOT BASED DESIGN IMPLEMENTATION OF MUSHROOM FARM MONITORING USING ARDUINO MICROCONTROLLERS & SENSORS | 65 |
| 3. | วิธีการดำเนินงาน | 66 |
|  | 3.1 แผนการดำเนินงาน | 66 |
|  | 3.2 โครงสร้างของระบบ | 67 |
|  | 3.3 การออกแบบวงจรตัวต้านทานตรวจสอบกระแส | 69 |
|  | 3.3.1 การคำนวณแรงดันเอาท์พุทของวงจรเมื่อใช้งานกับพัดลมระบายอากาศ | 70 |
|  | 3.3.2 การคำนวณแรงดันเอาท์พุทของวงจรเมื่อใช้งานกับปั๊มพ่นหมอก | 73 |
|  | 3.4 การออกแบบฮาร์ดแวร์ระบบควบคุมสภาพแวดล้อมภายในโรงเรือนเพาะเห็ด | 75 |
|  | 3.4.1 การอ่านค่าอุณหภูมิและความชื้นจากเซ็นเซอร์ AM2315 | 76 |
|  | 3.4.2 การอ่านค่าความสว่างจากเซ็นเซอร์ BH1750FVI | 78 |
|  | 3.4.3 การควบคุมพัดลมและปั๊มพ่นหมอก | 80 |
|  | 3.4.4 การควบคุมความสว่างภายในโรงเรือนเพาะเห็ด | 84 |
|  | 3.4.5 การใช้จอแอลซี (LCD) แสดงผล | 86 |
|  | 3.4.6 การอ่านค่าแรงดันเอาท์พุทจากวงจรตัวต้านทานตรวจสอบกระแส | 88 |
|  | 3.5 การออกแบบและสร้างโรงเรือนสำหรับเพาะเห็ด | 89 |
|  | 3.6 การออกแบบการสื่อสารผ่าน Lora | 95 |
|  | 3.6.1 การออกแบบการสื่อสาร Lora ที่ Mushroom Node | 96 |
| **สารบัญ (ต่อ)** | | |
|  | | |
|  | 3.6.2 การออกแบบการสื่อสาร Lora ที่ STA Node | 107 |
|  | 3.7 การออกแบบและสร้างเว็บแอปพลิเคชัน | 116 |
|  | 3.7.1 การสื่อสารระหว่าง Server และ Client | 117 |
|  | 3.7.2 การส่ง url ของ Web Application และ Database Server ไปยัง LINE Application | 122 |
|  | 3.7.3 การใช้โปรแกรม Ngrok ในการทำ Port forwarding | 126 |
|  | 3.8 การออกแบบและสร้างวินโดว์แอปพลิเคชัน | 130 |
|  | 3.8.1 การทำงานของวินโดว์แอปพลิเคชัน (Windows Application) | 131 |
|  | 3.9 การออกแบบฐานข้อมูลและการบันทึกข้อมูล | 139 |
|  | 3.9.1 การออกแบบฐานข้อมูล | 139 |
|  | 3.9.2 การบันทึกข้อมูลลงในฐานข้อมูล | 141 |
|  | 3.10 การออกแบบ Use Case Diagram | 145 |
| 4. | ผลการดำเนินงาน | 150 |
|  | 4.1 การทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือนและปุ่มควบคุมที่กล่อง STA | 150 |
|  | 4.2 การทดลองควบคุมระบบภายในโรงเรือนจากวินโดว์แอปพลิเคชัน (Windows Application) | 156 |
|  | 4.3 การทดลองควบคุมระบบภายในโรงเรือนจากเว็บแอปพลิเคชัน (Web Application) | 166 |
|  | 4.4 การทดลองเพาะเห็ด | 178 |
| 5. | สรุปผลการดำเนินงาน | 188 |
|  | 5.1 สรุปผล | 188 |
|  | 5.2 ปัญหาและแนวทางแก้ไข | 189 |
|  | 5.3 ข้อเสนอแนะและแนวทางการพัฒนา | 190 |
| บรรณานุกรม | | 191 |
| ภาคผนวก | | 195 |
|  | ภาคผนวก ก คู่มือการใช้งาน |  |
| ประวัติผู้เขียน | |  |

**สารบัญตาราง**

|  |  |  |
| --- | --- | --- |
| **ตารางที่** |  | หน้า |
| 2.1 | แสดงความถี่ที่ใช้สำหรับ LoRa (หน่วยเป็น MHz) | 27 |
| 2.2 | แสดงสถิติความชื้นสัมพัทธ์เฉลี่ย (%) ของประเทศไทยในช่วงฤดูกาลต่างๆ | 42 |
| 2.3 | แสดงความยาวคลื่นของแสงแต่ละสี | 42 |
| 3.1 | แสดงแผนการดำเนินงาน | 67 |
| 3.2 | แสดงการต่อขาระหว่าง ESP32LoRa และ AM2315 | 76 |
| 3.3 | แสดงการต่อขาระหว่าง ESP32LoRa และ BH1750FVI | 79 |
| 3.4 | แสดงการต่อขาระหว่าง ESP32LoRa และ 4-Channel Relay Module | 81 |
| 3.5 | แสดงการทำงานของปั๊มพ่นหมอกและพัดลมในโหมดการทำงานแบบอัตโนมัติ (AUTO) | 82 |
| 3.6 | แสดงการต่อขาระหว่าง ESP32LoRa และ L298N | 85 |
| 3.7 | แสดงการต่อขาระหว่าง ESP32LoRa และ จอแอลซีดี (LCD) | 86 |
| 3.8 | แสดงการต่อขาระหว่าง ESP32LoRa และวงจรตัวต้านทานตรวจสอบกระแส | 88 |
| 3.9 | แสดง Data Dictionary SensorData | 140 |
| 3.10 | แสดง Login Use Case | 146 |
| 3.11 | แสดง Logout Use Case | 146 |
| 3.12 | แสดง View Dashboard Use Case | 147 |
| 3.13 | แสดง Control Use Case | 147 |
| 3.14 | แสดง Setting Use Case | 148 |
| 3.15 | แสดง View Data Use Case | 148 |
| 3.16 | แสดง View Status Use Case | 149 |
| 4.1 | แสดงค่าอุณหภูมิและความชื้นที่มีผลต่อการออกดอกของเห็ดนางฟ้า | 178 |
| 4.2 | แสดงการทำงานของปั๊มพ่นหมอกและพัดลมในโหมดการทำงานแบบอัตโนมัติ (AUTO) | 179 |
|  |  |  |
|  |  |  |

**สารบัญรูป**

|  |  |  |
| --- | --- | --- |
| **รูปที่** |  | หน้า |
| 2.1 | แสดงภาพของไมโครคอนโทรลเลอร์ ESP32 LoRa | 4 |
| 2.2 | แสดงภาพ Raspberry Pi 3 Model B | 6 |
| 2.3 | แสดงภาพ AM2315 เซ็นเซอร์วัดอุณหภูมิและความชื้น | 7 |
| 2.4 | แสดงภาพขนาดของ AM2315 ในหน่วยมิลลิเมตร (mm) | 8 |
| 2.5 | แสดงภาพ AM2315 pin assignment | 9 |
| 2.6 | แสดงภาพ AM2315 wiring diagram | 9 |
| 2.7 | แสดงภาพ AM2315 relative humidity performance table | 9 |
| 2.8 | แสดงภาพ AM2315 relative temperature performance table | 10 |
| 2.9 | แสดงภาพการต่อวงจรกับ I2C bus ของ AM2315 | 11 |
| 2.10 | แสดงภาพรูปแบบการสื่อสารข้อมูลของ AM2315 | 11 |
| 2.11 | แสดงภาพตัวอย่างของ function code | 12 |
| 2.12 | แสดงภาพ AM2315 Data register | 13 |
| 2.13 | แสดงภาพสาถานะของ Register | 13 |
| 2.14 | แสดงภาพตัวอย่างการอ่านค่าจากเซ็นเซอร์ AM2315 | 14 |
| 2.15 | แสดงภาพการปลุกการทำงานของเซ็นเซอร์ | 15 |
| 2.16 | แสดงภาพตัวอย่างการส่งคำสั่งเพื่ออ่านค่าอุณหภูมิและความชื้น | 15 |
| 2.17 | แสดงภาพตัวอย่างการค่าอ่าอุณหภูมิและความชื้น | 16 |
| 2.18 | แสดงภาพ Flow chart การทำงานของเซ็นเซอร์ | 17 |
| 2.19 | แสดงภาพ BH1750FVI เซนเซอร์วัดความเข้มแสง | 18 |
| 2.20 | แสดงภาพชื่อขาสัญญาณของบอร์ดและรายละเอียด | 18 |
| 2.21 | แสดงภาพของ 4-Channel Relay Module | 19 |
| 2.22 | แสดงภาพของ 4-channel relay schematic | 21 |
| 2.23 | แสดงภาพ 4-channel relay PCB | 22 |
| 2.24 | แสดงภาพของจอแอลซีดี (I2C LCD) | 22 |
| 2.25 | แสดงภาพปั๊มพ่นหมอก DC12V 4.8 บาร์ | 24 |
| 2.26 | แสดงภาพพัดลม 12 นิ้ว 10 ใบ 12V | 26 |
| 2.27 | แสดงภาพตัวอย่างค่า Data Rate (DR) | 27 |
|  |  |  |
| **สารบัญรูป (ต่อ)** | | |
|  |  |  |
| 2.28 | แสดงภาพความสัมพันธ์ระหว่าง BITRATE และระยะห่างของอุปกรณ์กับ Gateway | 28 |
| 2.29 | แสดงภาพคุณสมบัติหลักของเทคโนโลยี LoRa | 29 |
| 2.30 | แสดงลักษณะการการเชื่อมต่ออุปกรณ์แบบ I2C BUS | 30 |
| 2.31 | แสดงรูปแบบการเขียนและอ่านข้อมูลแบบ I2C BUS | 31 |
| 2.32 | แสดง I2C BUS START and STOP Conditions | 31 |
| 2.33 | แสดงรหัสควบคุมของ I2C BUS (Control Byte) | 32 |
| 2.34 | แสดงช่วงเวลารับส่งบิตข้อมูลของ I2C BUS | 33 |
| 2.35 | แสดงบล็อกไดอะแกรมของระบบควบคุมที่ประกอบด้วยวงจร ADC ตัวประมวลผลและวงจร DAC | 34 |
| 2.36 | แสดงภาพความละเอียดและแรงดันที่ ESP32 สามารถอ่านได้ | 35 |
| 2.37 | แสดงภาพค่าอนาล็อกที่อ่านได้เมื่อใช้ ESP32 | 35 |
| 2.38 | แสดงภาพ ESP32 Pinout Reference | 36 |
| 2.39 | แสดงภาพสัญญาณเปิด(HIGH) และปิด(LOW) | 38 |
| 2.40 | แสดงภาพตัวอย่างสัญญาณ PWM | 38 |
| 2.41 | แสดงภาพการวางตัวต้านทานตรวจสอบกระแสแบบการตรวจจับด้านต่ำ (Low-side Sensing) | 44 |
| 2.42 | แสดงภาพการวางตัวต้านทานตรวจสอบกระแสแบบการตรวจจับด้านสูง  (High-side Sensing) | 44 |
| 2.43 | แสดงภาพวงจรออปแอมป์ (Op-Amp Circuit) | 45 |
| 2.44 | แสดงแผนภาพบล็อกการทำงานของออปแอมป์ | 46 |
| 2.45 | แสดงภาพของวงจรขยายสัญญาณแบบกลับเฟส | 47 |
| 2.46 | แสดงภาพของวงจรขยายแบบไม่กลับเฟส | 48 |
| 2.47 | แสดงวงจรขยายสัญญาณแบบตามแรงดัน | 48 |
| 2.48 | แสดงวงจรเปรียบเทียบแรงดัน | 49 |
| 2.49 | แสดงภาพตัวอย่างเว็บแอปพลิเคชัน | 49 |
| 2.50 | แสดงภาพตัวอย่างวินโดว์แอปพลิเคชัน | 51 |
| 2.51 | แสดงภาพตัวอย่างโลโก้ภาษา C++ | 52 |
| 2.52 | แสดงภาพโลโก้ภาษา Visual Basic.NET | 54 |
| **สารบัญรูป (ต่อ)** | | |
|  | | |
| 2.53 | แสดงภาพตัวอย่างของหน้าออกแบบฟอร์มที่ว่างปล่าวใน Visual Studio 2019 | 55 |
| 2.54 | แสดงภาพโลโก้ของภาษา PHP | 56 |
| 2.55 | แสดงภาพโลโก้ของ JavaScript | 57 |
| 2.56 | แสดงภาพสัญลักษณ์ของ Ajax | 58 |
| 2.57 | แสดงภาพโลโก้ของ HTML | 59 |
| 2.58 | แสดงภาพโลโก้ของ CSS | 60 |
| 2.59 | แสดงสัญลักษณ์ของโปรแกรม Ngrok | 61 |
| 2.60 | แสดงสัญลักษณ์ของระบบฐานข้อมูล | 62 |
| 3.1 | แสดงภาพโครงสร้างของระบบ | 69 |
| 3.2 | แสดงภาพวงจรตัวต้านทานตรวจสอบกระแส | 70 |
| 3.3 | แสดงภาพวงจรตัวต้านทานตรวจสอบกระแสเมื่อใช้งานกับพัดลมระบายอากาศ | 71 |
| 3.4 | แสดงภาพวงจรตัวต้านทานตรวจสอบกระแสเมื่อใช้งานกับปั๊มพ่นหมอก | 73 |
| 3.5 | แสดงภาพส่วนประกอบของฮาร์ดแวร์ระบบควบคุมสภาพแวดล้อมภายในโรงเรือน | 75 |
| 3.6 | แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และ AM2315 | 76 |
| 3.7 | แสดงภาพโปรแกรมที่ใช้อ่านค่าจากเซ็นเซอร์ AM2315 | 77 |
| 3.8 | แสดงภาพผลลัพธ์ที่อ่านได้จากเซ็นเซอร์ AM2315 | 78 |
| 3.9 | แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และ BH1750FVI | 78 |
| 3.10 | แสดงภาพโปรแกรมที่ใช้อ่านค่าจากเซ็นเซอร์ BH1750FVI | 79 |
| 3.11 | แสดงภาพผลลัพธ์ที่อ่านได้จากเซ็นเซอร์ BH1750FVI | 80 |
| 3.12 | แสดงภาพ Schematic ของการควบคุมพัดลมและปั๊มพ่นหมอก | 81 |
| 3.13 | แสดงภาพโปรแกรมส่วนที่ใช้ควบคุมการทำงานของ Relay | 81 |
| 3.14 | แสดงภาพโปรแกรมที่ใช้ควบคุมการทำงานของปั๊มพ่นหมอกและพัดลม | 83 |
| 3.15 | แสดงภาพวงจร LED Driver | 84 |
| 3.16 | แสดงภาพ Schematic ของการควบคุมความสว่างภายในโรงเรือนเพาะเห็ด | 85 |
| 3.17 | แสดงภาพโปรแกรมที่ใช้ในการควบคุมความสว่างของหลอดไฟ | 85 |
| 3.18 | แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และ จอแอลซีดี (LCD) | 86 |
| **สารบัญรูป (ต่อ)** | | |
|  | | |
| 3.19 | แสดงภาพโปรแกรมทดสอบการทำงานของจอแอลซีดี (LCD) | 87 |
| 3.20 | แสดงภาพผลลัพธ์ที่ได้จากโปรแกรมทดสอบการทำงานของจอแอลซีดี (LCD) | 87 |
| 3.21 | แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และวงจรตัวต้านทานตรวจสอบกระแส | 88 |
| 3.22 | แสดงภาพโปรแกรมทดสอบการอ่านค่าแรงดันเอาท์พุทจากวงจรตัวต้านทานตรวจสอบกระแส | 89 |
| 3.23 | แสดงภาพการออกแบบโรงเรือนที่ใช้เพาะเห็ด | 90 |
| 3.24 | แสดงภาพโครงสร้างของโรงเรือนเพาะเห็ด | 90 |
| 3.25 | แสดงภาพขั้นตอนการพ่นสีโรงเรือนเพาะเห็ด | 91 |
| 3.26 | แสดงภาพขั้นตอนการติดลวดตาข่ายเพื่อป้องกันก้อนเห็ดหล่น | 91 |
| 3.27 | แสดงภาพขั้นตอนการคลุมโรงเรือนเพาะเห็ดเพื่อเก็บรักษาความชื้น | 92 |
| 3.28 | แสดงภาพขั้นตอนการติดตั้งพัดลมระบายอากาศ | 92 |
| 3.29 | แสดงภาพขั้นตอนการติดตั้งหัวพ่นหมอก | 93 |
| 3.30 | แสดงภาพขั้นตอนการคลุมโรงเรือนโดยใช้สแลนเพื่อป้องกันแสงแดดและลดความร้อนจากแสงแดด | 93 |
| 3.31 | แสดงภาพขั้นตอนการติดตั้งกล่องควบคุมระบบ | 94 |
| 3.32 | แสดงภาพขั้นตอนการติดตั้งหลอด LED สีน้ำเงินเพื่อให้แสงสว่าง | 94 |
| 3.33 | แสดงภาพขั้นตอนการติดตั้งเซ็นเซอร์ BH1750FVI | 95 |
| 3.34 | แสดงภาพการสื่อสารแบบ Node-to-Node | 95 |
| 3.35 | แสดงภาพ Flowchart การทำงานของ Mushroom Node (1) | 96 |
| 3.36 | แสดงภาพรูปแบบของข้อมูล (Data Format) | 97 |
| 3.37 | แสดงภาพฟังก์ชัน sendUpdateData() | 99 |
| 3.38 | แสดงภาพโปรแกรมส่วนที่ตรวจสอบข้อมูลที่ส่งมายัง Mushroom Node | 100 |
| 3.39 | แสดงภาพ Flowchart การทำงานของ Mushroom Node (2) | 101 |
| 3.40 | แสดงภาพ Flowchart การทำงานของ Mushroom Node (3) | 101 |
| 3.41 | แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (1) | 102 |
| 3.42 | แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (2) | 102 |
| 3.43 | แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (3) | 103 |
| 3.44 | แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (4) | 103 |
| **สารบัญรูป (ต่อ)** | | |
|  |  |  |
| 3.45 | แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (5) | 104 |
| 3.46 | แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (6) | 104 |
| 3.47 | แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (7) | 105 |
| 3.48 | แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (8) | 106 |
| 3.49 | แสดงภาพ Flowchart การทำงานของ STA Node (1) | 107 |
| 3.50 | แสดงภาพโปรแกรมส่วนที่รับข้อมูลจาก Serial Port | 108 |
| 3.51 | แสดงภาพโปรแกรมส่วนที่ตรวจสอบข้อมูลที่ส่งมายัง STA Node | 109 |
| 3.52 | แสดงภาพ Flowchart การทำงานของ STA Node (2) | 109 |
| 3.53 | แสดงภาพรูปแบบของข้อมูล (Data Format) ที่ส่งมาจาก Mushroom Node | 110 |
| 3.54 | แสดงภาพการใช้ฟังก์ชัน substring() กับข้อมูลที่ได้รับจาก Mushroom Node | 111 |
| 3.55 | แสดงภาพโปรแกรมที่ใช้ในการส่ง Keyword ทุก 60 วินาที | 112 |
| 3.56 | แสดงภาพ Flowchart การทำงานของ STA Node (3) | 113 |
| 3.57 | แสดงภาพ Flowchart การทำงานของ STA Node (4) | 114 |
| 3.58 | แสดงภาพโครงสร้างการทำงานของ Web Application | 116 |
| 3.59 | แสดงภาพ Flowchart การสื่อสารระหว่าง Server และ Client | 117 |
| 3.60 | แสดงภาพตัวอย่างโปรแกรมส่วนที่ใช้ในการขออัปเดตข้อมูลทุกๆ 5 วินาที | 118 |
| 3.61 | แสดงภาพข้อมูลต่างๆบนหน้าเว็บที่อัปเดตทุกๆ 5 วินาที | 119 |
| 3.62 | แสดงภาพฟังก์ที่ถูกเรียกใช้เมื่อกดปุ่มเปลี่ยนโหมดการทำงาน | 119 |
| 3.63 | แสดงภาพการแจ้งเตือนเมื่อกดปุ่มเปลี่ยนโหมดการทำงาน | 120 |
| 3.64 | แสดงภาพฟังก์ที่ถูกเรียกใช้เมื่อกดปุ่มขออัปเดตข้อมูล | 120 |
| 3.65 | แสดงภาพฟังก์ที่ถูกเรียกใช้เมื่อกดปุ่ม เปิด/ปิด ปั๊ม | 121 |
| 3.66 | แสดงภาพฟังก์ที่ถูกเรียกใช้เมื่อกดปุ่ม เปิด/ปิด พัดลม | 121 |
| 3.67 | แสดงภาพ Flowchart ขั้นตอนการส่ง url ไปยัง LINE Application | 122 |
| 3.68 | แสดงภาพไฟล์ public-url.php | 123 |
| 3.69 | แสดงภาพข้อมูล JSON ที่ได้จาก http://localhost:4040/api/tunnels | 124 |
| 3.70 | แสดงภาพผลลัพธ์ที่ได้จากไฟล์ public-url.php | 124 |
| 3.71 | แสดงภาพโปรแกรมส่วนที่จัดการจัด url ที่ได้จากโปรแกรม Ngrok | 125 |
|  |  |  |
| **สารบัญรูป (ต่อ)** | | |
|  |  |  |
| 3.72 | แสดงภาพตัวอย่าง url ของ Web Server และ Database Server ที่ส่งไปยัง  LINE Application | 125 |
| 3.73 | แสดงภาพ Authtoken ที่ได้จากการสมัครสมาชิก | 126 |
| 3.74 | แสดงภาพ ไฟล์ ngrok configuration file | 127 |
| 3.75 | แสดงภาพไฟล์ autostart.sh | 128 |
| 3.76 | แสดงภาพการใช้ Crontab ในการ run คำสั่ง | 128 |
| 3.77 | แสดงภาพผลลัพธ์ที่ได้จากการทำ Port forwarding | 129 |
| 3.78 | แสดงภาพโครงสร้างของระบบ | 130 |
| 3.79 | แสดงภาพ Flowchart การทำงานของวินโดว์แอปพลิเคชัน (1) | 131 |
| 3.80 | แสดงภาพ Flowchart การทำงานของวินโดว์แอปพลิเคชัน (2) | 132 |
| 3.81 | แสดงภาพหน้า Login ของวินโดว์แอปพลิเคชัน | 133 |
| 3.82 | แสดงภาพวินโดว์แอปพลิเคชันเมื่อยังไม่ได้เชื่อมต่อกับ Serial Port | 133 |
| 3.83 | แสดงภาพวินโดว์แอปพลิเคชันเมื่อเชื่อมต่อกับ Serial Port แล้ว | 134 |
| 3.84 | แสดงภาพส่วนของโปรแกรม Visual Basic ที่จัดการกับข้อมูลที่ส่งมาจาก Serial Port | 135 |
| 3.85 | แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกดปุ่มเปลี่ยนโหมด | 135 |
| 3.86 | แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกดปุ่ม เปิด/ปิด ปั๊ม | 136 |
| 3.87 | แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกดปุ่ม เปิด/ปิด พัดลม | 136 |
| 3.88 | แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าอุณหภูมิต่ำสุด | 136 |
| 3.89 | แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าอุณหภูมิสูงสุด | 137 |
| 3.90 | แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าความชื้นต่ำสุด | 137 |
| 3.91 | แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าความชื้นสูงสุด | 138 |
| **สารบัญรูป (ต่อ)** | | |
|  |  |  |
| 3.92 | แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าความสว่าง | 138 |
| 3.93 | แสดงภาพโครงสร้างของฐานข้อมูล | 139 |
| 3.94 | แสดงภาพตัวอย่างข้อมูลเก็บอยู่ภายในฐานข้อมูล | 140 |
| 3.95 | แสดงภาพ Flowchart ขั้นตอนการบันทึกข้อมูลลงในฐานข้อมูล | 141 |
| 3.96 | แสดงภาพการบันทึกข้อมูลลงในฐานข้อมูลทุกชั่วโมง | 142 |
| 3.97 | แสดงภาพฟังก์ชันที่ใช้บันทึกข้อมูลลงในฐานข้อมูล | 143 |
| 3.98 | แสดงภาพไฟล์ post-esp-data.php บางส่วน | 144 |
| 3.99 | แสดงภาพ Use Case Diagram | 145 |
|  |  |  |

**บทที่ 1**

**บทนำ**

**1.1 ความเป็นมาและความสำคัญของปัญหา**

ในปัจุบันประเทศไทยมีการใช้งานอินเทอร์เน็ตเพิ่มขึ้นอย่างมาก เมื่อเทียบกับในอดีตที่ผ่านมาโดยจะเห็นได้จาก ผลการสำรวจพฤติกรรมผู้ใช้อินเทอร์เน็ตในประเทศไทย ของสำนักงานพัฒนาธุรกรรมทางอิเล็กทรอนิกส์ (สพธอ.) หรือ ETDA (เอ็ตด้า) กระทรวงดิจิทัลเพื่อเศรษฐกิจและสังคม เผยผลการสำรวจพฤติกรรมผู้ใช้อินเทอร์เน็ตในประเทศไทย ปี 2562 หรือ Thailand Internet User Behavior 2019 ชี้ ทศวรรษที่ผ่านมา คนไทยใช้อินเทอร์เน็ตเพิ่มขึ้นอย่างก้าวกระโดดกว่า 150% ส่งผลให้ปัจจุบันไทยมีผู้ใช้อินเทอร์เน็ต 47.5 ล้านคน หรือราว 70% ของจำนวนประชาชนทั้งหมด จากการสำรวจข้อมูลของประชาชนเกี่ยวกับพฤติกรรมการใช้อินเทอร์เน็ต ประจำปี 2562 ผ่านทางออนไลน์ ช่วงเดือน ส.ค.- ต.ค. 2562 โดยมีคนไทยเข้ามาตอบแบบสอบถามกว่า 17,242 คน ซึ่งจากการวิเคราะห์ข้อมูล พบว่า ปี 2562 คนไทยใช้อินเทอร์เน็ตเฉลี่ยวันละ 10 ชั่วโมง 22 นาที เพิ่มขึ้น 17 นาทีจากปี 2561 [5] จากข้อมูลดังกล่าวจะเห็นได้ว่าในปัจุบันอินเทอร์เน็ตมีความสำคัญต่อการดำเนินชีวิตประจำวันของคนไทย ทั้งด้านการศึกษา ด้านธุรกิจและพาณิชย์ ด้านการบันเทิง รวมถึงการประยุกต์ใช้งานอินเทอร์เน็ตในภาคการเกษตร เช่น การควบคุมการให้น้ำ การควบคุมอุณหภูมิในโรงเรือน การติดตามสภาพดิน เป็นต้น

การเพาะเห็ดในปัจุบันนิยมเพาะในโรงเรือน โรงเรือนที่เหมาะสำหรับการเพาะเห็ด ควรเป็นโรงเรือนที่สามารถเก็บความชื้นได้ดี มีระบบถ่ายเทอากาศดี และสะดวกต่อการทำความสะอาด เห็ดจะออกดอกได้ดีเมื่อมีสภาพแวดล้อมเหมาะสม ดังนั้นการควบคุมสภาพแวดล้อมภายในโรงเรือนให้เหมาะสมต่อการออกดอกของเห็ดโดยมนุษย์อาจทำให้สภาพแวดล้อมภายในโรงเรือนไม่เหมาะสำหรับการออกดอกของเห็ดเท่าที่ควร อาจส่งผลให้ขนาดของดอกเห็ดมีขนาดเล็กและมีผลผลิตต่ำ จึงได้มีการนำไมโครคอนโทรลเลอร์มาควบคุมสภาพแวดล้อมภายในโรงเรือน โดยนำค่าที่อ่านได้จากเซ็นเซอร์วัดสภาพแวดล้อมมาประมวลผล แล้วนำค่าที่ได้มาควบคุมการทำงานของอุปกรณ์ปรับสภาพแวดล้อมต่างๆ เช่น พัดลมระบายอากาศ ปั๊มพ่นหมอก หลอดไฟ เพื่อปรับสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ด และเมื่อสามารถควบคุมสภาพแวดล้อมภายในโรงเรือนได้แล้ว จึงนำเทคโนโลยีอินเทอร์เน็ตของสรรพสิ่ง (Internet of Things) หรือ ไอโอที (IoT) มาประยุกต์ใช้งานในการเพาะเห็ด ทำให้สามารถมอนิเตอร์ (Monitor) สภาพแวดล้อมภายในโรงเรือน และความคุมระบบภายในโรงเรือน จากที่ไหนก็ได้ที่สามารถใช้อินเทอร์เน็ตได้ [1]-[4] จากการศึกษางานวิจัยที่เกี่ยวข้องพบว่ายังมีข้อจำกัดคือ ที่ตั้งของโรงเรือนต้องมีสัญญาณอินเทอร์เน็ตหากไม่สัญญาณอินเทอร์เน็ตก็จะไม่สามารถใช้งานได้ และในการแสดงค่าสถานะการทำงานของอุปกรณ์ต่างๆภายในโรงเรือนเป็นค่าที่ได้จากซอฟแวร์ ซึ่งหากอุปกรณ์เกิดขัดข้องหรือมีปัญหาเกิดขึ้นจะไม่สามารถทราบได้ว่าอุปกรณ์ทำงานหรือไม่

ดังนั้นจากปัญหาที่กล่าวมาผู้จัดทำจึงได้ออกแบบและสร้างโรงเพาะเห็ด ที่สามารถควบคุมอุณหภูมิความชื้นและแสงสว่างภายในโรงเรือนให้เหมาะสมต่อการออกดอกของเห็ด ออกแบบและสร้างเว็บแอปพลิเคชันและวินโดว์แอปพลิเคชัน ที่สามารถมอนิเตอร์ค่าสภาพแวดล้อมและควบคุมระบบภายในโรงเรือนได้ และได้ประยุกต์ใช้งาน Lora Communication เพื่อเป็นตัวกลางการสื่อสารระหว่างโรงเรือนเพาะเห็ด (Mushroom Node) และพื้นที่ที่มีสัญญาณอินเทอร์เน็ต (STA Node) เพื่อแก้ปัญหาที่ตั้งของโรงเรือนไม่มีสัญญาณอินเทอร์เน็ต และผู้จัดทำได้ประยุกต์ใช้งานตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) ตรวจว่าอุปกรณ์ทำงานจริงตามที่ควบคุมหรือไม่ เพื่อให้การแสดงค่าสถานะการทำงานของอุปกรณ์ต่างๆที่ถูกต้อง

**1.2. วัตถุประสงค์**

1.2.1 เพื่อออกแบบและสร้างเว็บแอปพลิเคชันสำหรับมอนิเตอร์และควบคุมระบบโรงเรือนเพาะเห็ด

1.2.2 เพื่อออกแบบและสร้างวินโดว์แอปพลิเคชันสำหรับมอนิเตอร์และควบคุมระบบโรงเรือนเพาะเห็ด

1.2.3 เพื่อออกแบบและสร้างโรงเรือนที่ใช้สำหรับเพาะเห็ด

1.2.4 เพื่อออกแบบระบบควบคุมอุณหภูมิความชื้นและแสงสว่างภายในโรงเรือนเพาะเห็ด

1.2.5 เพื่อศึกษาและประยุกต์ใช้งาน Lora Communication สำหรับการควบคุมโรงเรือนเพาะเห็ด

1.2.6 เพื่อศึกษาการพัฒนาเว็บแอปพลิเคชันบนไมโครคอนโทรลเลอร์ (ESP32LoRa)

**1.3 ขอบเขต**

1.3.1 เว็บแอปพลิเคชัน

1) สามารถดูค่าสถานะและควบคุมระบบภายในโรงเรือนได้จากทุกที่ที่สามารถใช้งานอินเทอร์เน็ตได้

2) ใช้แอปพลิเคชัน ngrok ที่ติดไว้บน Raspberry Pi ทำ Port forwarding ของ Web Application เพื่อให้สามารถเข้าถึงเว็บแอปพลิเคชันได้จากทุกที่ที่สามารถใช้งานอินเทอร์เน็ตได้

3) ใช้ Raspberry Pi 3 Model B เป็น Database Server และทำ Port forwarding

4) ใช้ไมโครคอนโทรลเลอร์ (ESP32LoRa) เป็น Web Server

1.3.2 วินโดว์แอปพลิเคชัน

1) สามารถดูค่าสถานะและควบคุมระบบภายในโรงเรือนได้ผ่านทาง Serial Port ระหว่างคอมพิวเตอร์และไมโครคอนโทรลเลอร์

2) วินโดว์แอปพลิเคชันสามารถใช้งานได้กับไมโครคอนโทรลเลอร์ที่อยู่โรงเรือนเพาะเห็ด (Mushroom Node) และไมโครคอนโทรลเลอร์ที่ทำหน้าที่เป็น Web Server (STA Node)

3) สามารถนำวินโดว์แอปพลิเคชันไปติดตั้งที่คอมพิวเตอร์เครื่องอื่นที่เป็นระบบปฏิบัติการวินโดว์แล้วนำแอปพลิเคชันมาใช้งานได้ทันที

1.3.3 โรงเรือนที่ใช้สำหรับเพาะเห็ด

1) ออกแบบโรงเรือนสำหรับเพาะเห็ดขนาด 200x150x210 เซนติเมตรเป็นโรงเรือนแบบปิดสามารถควบคุมอุณหภูมิความชื้นและแสงสว่างได้

2) ใช้ก้อนเห็ดนางฟ้าที่เชื้อเดินเต็มก้อนมาทำการทดลอง

3) ใช้ ESP32LoRa เป็นไมโครคอนโทรลเลอร์ในการควบคุมระบบภายในโรงเรือนเพาะเห็ด

**1.4 ประโยชน์ที่คาดว่าจะได้รับ**

1.4.1 เว็บแอปพลิเคชันสำหรับมอนิเตอร์และควบคุมระบบโรงเรือนเพาะเห็ด

1.4.2 วินโดว์แอปพลิเคชันสำหรับมอนิเตอร์และควบคุมระบบโรงเรือนเพาะเห็ด

1.4.3 โรงเรือนที่ใช้สำหรับเพาะเห็ด

1.4.4 ระบบควบคุมอุณหภูมิความชื้นและแสงสว่างภายในโรงเรือนเพาะเห็ด

1.4.5 ความรู้และการประยุกต์ใช้งาน Lora Communication

1.4.6 ความรู้การพัฒนาเว็บแอปพลิเคชันบน ESP32LoRa

**บทที่ 2**

**ทฤษฎีและงานวิจัยที่เกี่ยวข้อง**

ในการจัดทำโครงงานโรงเพาะเห็ดอัจฉริยะผู้จัดทำโครงงานได้ศึกษาทฤษฎีและงานวิจัยที่เกี่ยวข้องเพื่อใช้ในการดำเนินโครงงานให้เป็นไปตามวัตถุประสงค์และขอบเขตของโครงงาน

**2.1 ทฤษฎีที่เกี่ยวข้อง**

**2.1.1** **ไมโครคอนโทรลเลอร์ ESP32 LoRa**

ESP32 LoRa เป็นบอร์ดที่รวมเอา ESP32 และโมดูล LoRa SX1278 ไว้ในบอร์ดเดียวทำให้สามารถสื่อสารแบบ Wi-Fi Bluetooth และสามารถส่งข้อมูลระยะไกลแบบ LoRa

|  |
| --- |
| **รูปที่ 2.1** แสดงภาพของไมโครคอนโทรลเลอร์ ESP32 LoRa |

รายละเอียด ESP32 เป็นชื่อของไอซีไมโครคอนโทรลเลอร์ที่รองรับการเชื่อมต่อ WiFi และ Bluetooth 4.2 BLE ในตัว ผลิตโดยบริษัท Espressif จากประเทศจีน โดยไอซี ESP32 มีสเปคโดยละเอียด ดังนี้

* ซีพียูใช้สถาปัตยกรรม Tensilica LX6 แบบ 2 แกนสมอง สัญญาณนาฬิกา 240MHz
* มีแรมในตัว 512KB
* รองรับการเชื่อมต่อรอมภายนอกสูงสุด 16MB
* มาพร้อมกับ WiFi มาตรฐาน 802.11 b/g/n รองรับการใช้งานทั้งในโหมด Station softAP และ Wi-Fi direct
* มีบลูทูธในตัว รองรับการใช้งานในโหมด 2.0 และโหมด 4.0 BLE
* ใช้แรงดันไฟฟ้าในการทำงาน 2.6V ถึง 3V
* ทำงานได้ที่อุณหภูมิ -40◦C ถึง 125◦C
* วงจรกรองสัญญาณรบกวนในวงจรขยายสัญญาณ
* เซ็นเซอร์แม่เหล็ก
* เซ็นเซอร์สัมผัส (Capacitive touch) รองรับ 10 ช่อง
* รองรับการเชื่อมต่อคลิสตอล 32.768kHz สำหรับใช้กับส่วนวงจรนับเวลาโดยเฉพาะขาใช้งานต่าง ๆ ของ ESP32 รองรับการเชื่อมต่อบัสต่าง ๆ
* มี GPIO จำนวน 32 ช่อง
* รองรับ UART จำนวน 3 ช่อง
* รองรับ I2C จำนวน 2 ช่อง
* รองรับ ADC จำนวน 12 ช่อง
* รองรับ DAC จำนวน 2 ช่อง
* รองรับ I2S จำนวน 2 ช่อง
* รองรับ PWM / Timer ทุกช่อง
* รองรับการเชื่อมต่อกับ SD-Card
* รองรับการเข้ารหัส WiFi แบบ WEP และ WPA/WPA2 PSK/Enterprise
* มีวงจรเข้ารหัส AES / SHA2 / Elliptical Curve Cryptography / RSA-4096 ในตัว
* รับ – ส่ง ข้อมูลได้ความเร็วสูงสุดที่ 150Mbps เมื่อเชื่อมต่อแบบ 11n HT40 ได้ความเร็วสูงสุด 72Mbps เมื่อเชื่อมต่อแบบ 11n HT20 ได้ความเร็วสูงสุดที่ 54Mbps เมื่อเชื่อมต่อแบบ 11g และได้ความเร็วสูงสุดที่ 11Mbps เมื่อเชื่อมต่อแบบ 11b
* เมื่อใช้การเชื่อมต่อผ่านโปรโตคอล UDP จะสามารถรับ – ส่งข้อมูลได้ที่ความเร็ว 135Mbps
* ในโหมด Sleep ใช้กระแสไฟฟ้าเพียง 2.5uA

รายละเอียด LoRa SX1278 ทำงานร่วมกับโปรโตคอลการสื่อสาร SPI ดังนั้นจึงสามารถใช้กับไมโครคอนโทรลเลอร์ที่รองรับ SPI ได้ จำเป็นต้องใช้ Ariel (เสาอากาศ) ร่วมกับโมดูลมิฉะนั้นอาจทำให้โมดูลเสียหายอย่างถาวร โมดูลควรใช้พลังงาน 3.3V เท่านั้นโดยมีคุณสมบัติต่างดังนี้

* ใช้แรงดันไฟฟ้าในการทำงาน 3.3V
* ความถี่ในการทำงาน 433MHz
* การสื่อสาร SPI แบบ Half Duplex
* เทคนิคการมอดูเลต FSK, GFSK, MSK, GMSK, LoRa
* Packet size: 256 bytes
* ความไว: -148db

**2.1.2 Raspberry Pi 3 Model B**

|  |
| --- |
| **รูปที่ 2.2** แสดงภาพ Raspberry Pi 3 Model B |

Raspberry Pi 3 Model B เปรียบเสมือนคอมพิวเตอร์ขนาดเล็กที่สามารถเชื่อมต่อกับ จอ เมาส์ และคีย์บอร์ดได้ สามารถทำงานได้เหมือนคอมพิวเตอร์ทั่วไป เช่น การทำงานเอกสาร ดูหนัง ฟังเพลง การใช้งานอินเทอร์เน็ต การเขียนโปรแกรม เป็นต้น

Raspberry Pi สามารถรองรับระบบปฏิบัติการ Linux ได้หลาย Distros เช่น Respbian Ubuntu Pidora และ Arch Linux โดยติดตั้งระบบปฏิบัติการลงใน SD Card โดย Raspberry Pi สามารถนำไปประยุกต์ใช้ทำสิ่งต่างๆได้มากมาย เช่น Database Server Robot Microcontroller

รายละเอียดทางเทคนิคของ Raspberry Pi 3 Model B มีดังนี้

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU

- 1GB RAM

- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board

- 100 Base Ethernet

- 40-pin extended GPIO

- 4 USB 2 ports

- 4 Pole stereo output and composite video port

- Full size HDMI

- CSI camera port for connecting a Raspberry Pi camera

- DSI display port for connecting a Raspberry Pi touchscreen display

- Micro SD port for loading your operating system and storing data

- Upgraded switched Micro USB power source up to 2.5A

**2.1.3 AM2315 เซ็นเซอร์วัดอุณหภูมิและความชื้น**

|  |
| --- |
| **รูปที่ 2.3** แสดงภาพ AM2315 เซ็นเซอร์วัดอุณหภูมิและความชื้น |

AM2315 คือเซ็นเซอร์วัดค่าอุณหภูมิและความชื้นคุณภาพสูงที่มีเอาท์พุทเป็นสัญญาณดิจิตอลโดยมีการสอบเทียบสัญญาณดิจิตอลเอาท์พุทแล้ว ใช้โมดูลตรวจจับอุณหภูมิละความชื้นคุณภาพสูงเพื่อให้แน่ใจว่ามีการวัดค่าที่แม่นยำและมีความทนทาน เซ็นเซอร์ประกอบด้วยตัววัดความชื้นประเภทความจุไฟฟ้า อุปกรณ์วัดอุณหภูมิที่มีความแม่นยำสูง และเชื่อมต่อด้วยไมโครคอนโทรลเลอร์ 8 บิตประสิทธิภาพสูง AM2315 สื่อสารด้วย I2C ผู้ใช้สามารถเชื่อต่อ I2C bus ได้โดยตรง

|  |
| --- |
| **รูปที่ 2.4** แสดงภาพขนาดของ AM2315 ในหน่วยมิลลิเมตร (mm) |

รายละเอียดทางเทคนิคของ AM2315 มีดังต่อไปนี้

- 3.5 to 5.5V power and I/O

- 10 mA max current use during conversion (while requesting data)

- Good for 0-100% humidity readings with 2% accuracy

- Good for -20 to 80°C temperature readings ±0.1°C typical accuracy

- 0.5 Hz update rate

- Body size 98mm x 16mm diameter (4" x 0.9" )

- 4 wires 20 inches long

- Weight: 82.64g

- This board/chip uses I2C 7-bit address 05C.

|  |
| --- |
| **รูปที่ 2.5** แสดงภาพ AM2315 pin assignment |
| **รูปที่ 2.6** แสดงภาพ AM2315 wiring diagram |

|  |
| --- |
| **รูปที่ 2.7** แสดงภาพ AM2315 relative humidity performance table |

|  |
| --- |
| **รูปที่ 2.8** แสดงภาพ AM2315 relative temperature performance table |

**2.1.3.1 โปโตคอลการสื่อสาร I2C ของเซ็นเซอร์ AM2315**

AM2315 I2c bus มีจุดเชื่อมต่อ (Interface) ที่ได้มาตรฐาน สามารถต่อกับ I2C bus ของไมโครคอนโทรลเลอร์ได้โดยตรง เซ็นเซอร์ AM2315 มี I2C address (SLAVE ADDRESS) คือ 0xB8

**2.1.3.1.1 คำอธิบายอิ****นเทอร์เฟซ I2C ของเซ็นเซอร์ AM2315**

การวัดค่าอุณหภูมิและความชื้นที่แม่นยำและลดการทำงานของเซ็นเซอร์ เมื่อเซ็นเซอร์ไม่มีการทำงานเซ็นเซอร์จะเปลี่ยนเป็นโหมด Sleep โดยอัตโนมัติ เพื่อลดการทำงานของเซ็นเซอร์ ลดอุณหภูมิภายในของเซ็นเซอร์ เมื่อต้องการจะอ่านค่าจากเซ็นเซอร์ที่อยู่ในโหมด Sleep ไมโครคอนโทรลเลอร์จะต้องจะคำสั่งเพื่อปลุกการทำงานของเซ็นเซอร์ก่อน ดังนั้นถ้าไม่มีการอ่านค่าจากเซ็นเซอร์เป็นเวลานาน เมื่อจะอ่านค่าจากเซ็นเซอร์ต้องอ่านค่าอย่างน้อย 2 วินาที เพราะค่าที่อ่านได้วินาทีแรกจะเป็นค่าล่าสุดก่อนเซ็นเซอร์จะเข้าโหมด Sleep

**2.1.3.1.2 คุณสมบัตินเทอร์เฟซของ I2C**

ลักษณะเฉพาะ I2C ของ AM2315 ถ้าต้องการค่าอุณหภูมิและความชื้นที่ดีที่สุด การต่อสาย I2C bus จะต้องต่อดังรูปที่ 2.9

|  |
| --- |
| **รูปที่ 2.9** แสดงภาพการต่อวงจรกับ I2C bus ของ AM2315 |

**2.1.3.1.3 โปรโตคอลการสื่อสาร**

เซ็นเซอร์ AM2315 มีโปรโตคอลการสื่อสาร I2C bus ที่ได้มาตรฐานโดยอ้างอิงจาก โปรโตคอล ModBus เซ็นเซอร์ AM2315 ตามมาตรฐานโรงงานได้ร่วมเอา I2C\_ModBus โดยมีรูปแบบดังต่อไปนี้

1) รูปแบบการสื่อสารข้อมูล (Communication data format)

|  |
| --- |
| **รูปที่ 2.10** แสดงภาพรูปแบบการสื่อสารข้อมูลของ AM2315 |

2) การสื่อสารและกระบวนการถ่ายโอนข้อมูล

เมื่อมีการส่งคำสั่งการสื่อสารจากไมโครคอนโทรลเลอร์ (host) ไปยัง เซ็นเซอร์ และเซ็นเซอร์ได้รับคำสั่งแล้วจะประมวลผลคำสั่งและอ่านค่าจากนั้นจะส่งข้อมูลที่ได้กลับไปยังไมโครคอนโทรลเลอร์ (host) ข้อมูลที่ส่งกลับไปจะประกอบไปด้วย function code, ข้อมูล (data) และ CRC code

3) I2C slave address ของเซ็นเซอร์ AM2315

เซ็นเซอร์ AM2315 ทุกตัวจะมี I2C address เหมือนกันคือ 0xB8 ดังนั้น ใน I2C bus จะสามารถมีเซ็นเซอร์ AM2315 ได้เพียงแค่ 1 เซ็นเซอร์เท่านั้น

4) การสื่อสาร I2C function code

ในการสื่อสาร I2C function code คือ byte แรกของเฟรมข้อมูล (frame transmission) กฎของการสื่อสาร I2C\_ModBus กำหนด function code ระหว่าง 1 ถึง 127 เช่น ไมโครคอนโทรลเลอร์ (host) ส่งความต้องการที่จะดำเนินงานผ่านทาง function code เพื่อบอกเซ็นเซอร์ดำเนินการตามที่ต้องการ การตอบกลับของเซ็นเซอร์ก็จะส่ง function code เดียวกันไปยังไมโครคอนโทรลเลอร์ (host) เช่นกัน ตัวอย่างของ function code แสดงดังรูปที่ 2.11

|  |
| --- |
| **รูปที่ 2.11** แสดงภาพตัวอย่างของ function code |

5) พื้นที่ข้อมูลการสื่อสาร I2C

พื้นที่ข้อมูล (data area) ประกอบด้วยข้อมูลที่ต้องการจากเซ็นเซอร์ ตัวอย่างของข้อมูล เช่น อุณหภูมิ ความชื้น ข้อมูลต่างๆของเซ็นเซอร์ ข้อมูลที่เขียน เป็นต้น ที่อยู่อ้างอิง (reference address) ตัวอย่างเช่น ไมโครคอนโทรลเลอร์ (host) บอกเซ็นเซอร์ว่า 03 ผ่านทาง function code ข้อมูลที่ได้จะประกอบด้วย ความยาวของข้อมูล และ ข้อมูลภายใน register

I2C\_Modbus เซ็นเซอร์ใช้โปรโตคอลการสื่อสารแบบกำหนดเอง (custom) ไมโครคอนโทรลเลอร์ (host) ใช้คำสั่งการสื่อสาร (function code 03) สามารถอ่านข้อมูลใน register ที่แสดงในรูปที่ 2.12 ได้ทั้งหมด แต่ละ register จะมีขนาดเพียง 1 byte ถ้ามีการอ่านข้อมูลจากซนเซอร์มากกว่า 10 register ขึ้นไป จะได้ค่า error จากเซ็นเซอร์

|  |
| --- |
| **รูปที่ 2.12** แสดงภาพ AM2315 Data register |

6) รูปแบบเอ้าท์พุตของอุณหภูมิ (Temperature output format)

ค่าอุณหภูมิมีความละเอียด 16bit ตำแหน่งค่าอุณหภูมิมากสุด (Bit15) เท่ากับ 1 สำหรับค่าอุณหภูมิที่ติดลบ มีค่าเท่ากับ 0 สำหรับค่าอุณหภูมิที่เป็นบวก ค่าอุณหภูมิจะอยู่ที่ (Bit14-Bit0) เป็นตัวอักษร (string)

7) สาถานะของ Register

สาถานะของ Register Bit7-Bit0 bit แสดงดังรูปที่ 2.13

|  |
| --- |
| **รูปที่ 2.13** แสดงภาพสาถานะของ Register |

**2.1.3.1.4 Timing การสื่อสาร I2C**

การสื่อสาร I2C ของเซ็นเซอร์ AM2315 แม้ว่าจะอ้างอิงมาตรฐาน I2C Timing แต่จำเป็นสำหรับโปรโตคอลของเซ็นเซอร์ AM2315 และ ข้อกำหนดด้านเวลา (timing) ในการสื่อสาร สำหรับการอ่านค่าจากเซ็นเซอร์ที่ถูกต้อง ต้องทำตามข้อกำหนดและ timing

1) ตัวอย่างลำดับการอ่านค่าจากเซ็นเซอร์

รุปที่ 2.14 ตัวอย่างการอ่านค่าจากเซ็นเซอร์ AM2315 การอ่านค่าที่ถูกต้องต้องปฏิบัติตามข้อตกลงอย่างเข้มงวด จากรูปการสื่อสารระหว่างไมโครคอนโทรลเลอร์ (host) กับเซ็นเซอร์ตั้งแต่เริ่มต้นจนสิ้นสุดจะใช้เวลามากที่สุดคือ 3 วินาที

|  |
| --- |
| **รูปที่ 2.14** แสดงภาพตัวอย่างการอ่านค่าจากเซ็นเซอร์ AM2315 |

2) I2C Read and write timing decomposition

การอ่านหรือเขียนเซ็นเซอร์ ต้องเป็นไปตาม 3 ขั้นตอนต่อไปนี้ มิฉะนั้นจะไม่สามารถสื่อสารกับเซ็นเซอร์ได้หรือไม่สามารถอ่านค่าที่ถูกต้อง

ขั้นตอนที่ 1 ปลุกการทำงานของเซ็นเซอร์ ในการลดความร้อนภายในเซ็นเซอร์ เซ็นเซอร์จะอยู่ในสถานะไม่ทำงาน ต้องปลุกการทำงานของเซ็นเซอร์ก่อนส่งคำสั่งอ่านและเขียน มิฉะนั้นเซ็นเซอร์จะไม่ตอบสนอง (respond) โดยการตอบกลับด้วยการส่ง ACK การปลุกการทำงานของเซ็นเซอร์ทำได้ตามคำสั่งต่อไปนี้

ส่งสัญญาณเริ่มต้น (start signal) ใช้กับโฮสต์ที่อยู่เริ่มต้น รอช่วงเวลาหนึ่ง แล้วส่งสัญญาณสิ้นสุด (stop signal) กล่าวคือ initial signal + 0xB8 + wait (800us-3s) + stop signal ดังรูปที่ 2.15

|  |
| --- |
| **รูปที่ 2.15** แสดงภาพการปลุกการทำงานของเซ็นเซอร์ |

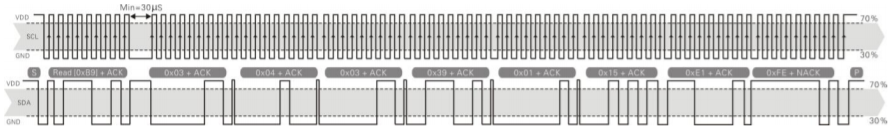
ขั้นตอนที่ 2 อ่านคำสั่งเพื่อส่งหรือส่งคำสั่งเพื่อเขียน

เมื่อปลุกการทำงานของเซ็นเซอร์แล้วจะสามารถอ่านข้อมูลได้อย่างครบถ้วนด้วยมาตรฐาน I2C timing รองรับความเร็วสูงสุด 100Kb/s การอ่านค่าอุณหภูมิและความชื้นเบื่องต้นแสดงดังรูปที่ 2.16 โดยไมโครคอนโทรลเลอร์ (host) จะส่งคำสั่งไปดังนี้ START + 0xB8 (SLA) + 0x03 (function code) + 0x00 (start address) + 0x04 (register length) + STOP

|  |
| --- |
| **รูปที่ 2.16** แสดงภาพตัวอย่างการส่งคำสั่งเพื่ออ่านค่าอุณหภูมิและความชื้น |

ขั้นตอนที่ 3 ข้อมูลอ่านกลับหรือสัญญาณยืนยัน

เมื่อไมโครคอนโทรลเลอร์ (host) ส่งคำสั่งเพื่ออ่านหรือเขียนไปแล้วต้องรออย่างน้อย 1.5ms แล้วขั้นตอนต่อไปก็ส่งลำดับการอ่าน (read sequence) เพื่ออ่านค่าจากเซ็นเซอร์ดังรูปที่ 2.17 เมื่ออ่านค่าเสร็จแล้ว I2C address จะต้องการรอเป็นเวลาอย่างน้อย 30us ก่อนที่จะส่งสัญญาณเพื่ออ่านค่าต่อไปหรือส่งค่า error



**รูปที่ 2.17** แสดงภาพตัวอย่างการค่าอ่าอุณหภูมิและความชื้น

ตัวอย่างของข้อมูลที่อ่านได้และส่งกลับไปยังไมโครคอนโทรลเลอร์ (host)

0x03(Function code) + 0x04(Data length) + 0x03(High humidity) + 0x39(Low humidity) + 0x01(High temperature) + 0x15(Low temperature) + 0xE1(CRC Low byte checksum) + 0xFE(CRC High byte checksum);

ดังนั้น 0339H = 3x256+3x16+9 = 825 => Humidity = 825/10=82.5%RH;

0115H = 1x256+1x16+5 = 277 => Temperature = 277/10=27.7\*C;

จากขั้นตอนทั้งหมดที่ผ่านมาเป็นกระบวนการอ่านค่าจากเซ็นเซอร์ ถ้าสามารถปฏิบัติตามขั้นตอนทั้งหมดได้ก็จะสามารถใช้งานเซ็นเซอร์ได้อย่างมีประสิทธิภาพ การอ่านค่าจากเซ็นเซอร์อย่างต่อเนื่องค่าได้จะอัฟเดททุก 2 วินาทีเป็นอย่างน้อย (minimum interval of continuous reading 2S)

**2.1.3.1.5 Flow chart การทำงานของเซ็นเซอร์**

|  |
| --- |
| **รูปที่ 2.18** แสดงภาพ Flow chart การทำงานของเซ็นเซอร์ |

**2.1.4 BH1750FVI เซนเซอร์วัดความเข้มแสง**

|  |
| --- |
| **รูปที่ 2.19** แสดงภาพ BH1750FVI เซนเซอร์วัดความเข้มแสง |

รายละเอียดทางเทคนิคของ BH1750FVI เซนเซอร์วัดความเข้มแสง มีดังนี้

* ใช้ไอซี BH1750FVI ของบริษัท ROHM SEMICONDUCTOR
* สามารถวัดความเข้มแสงได้ตั ้งแต่ 1-65535 lx ความละเอียด 16 บิต
* ใช้ไฟเลี ้ยง 3V-5V
* ใช้การเชื่อมต่อแบบ I2C ความถี่สูงสุด 400 kHz
* มีขา ADD สำาหรับเลือก Address ของบอร์ดได้ 2 ค่า
* เชื่อมต่อสัญญาณโดยตรงกับไมโครคอนโทรลเลอร์ระบบไฟ 3.3V และระบบไฟ 5V ได้ทันทีโดยไม่ต้องผ่านวงจรปรับระดับแรงดัน

|  |
| --- |
| **รูปที่ 2.20** แสดงภาพชื่อขาสัญญาณของบอร์ดและรายละเอียด |

ในกรณีที่ใช้บอร์ดเซ็นเซอร์ BH1750 กับระบบไฟที่ใช้แรงดันมากกว่า 3.3V และต้องการเลือก Address ของบอร์ดเป็น 0x5C ( โดยปกติถ้าปล่อยขา ADD ลอยไว้หรือต่อลง GND จะได้ค่า Address เป็น 0x23 ) ให้ทำาการต่อตัวต้านทานค่า 1k ที่ขา ADD ก่อนที่จะไปต่อกับขา VCC เพื่อให้ได้ลอจิก 1 เนื่องจากวงจรของบอร์ดที่ขา ADD ไม่มีวงจรปรับระดับแรงดัน ถ้าต่อตรงๆ จะทำาให้ไอซี BH1750 เสียหายได

**2.1.5 4-Channel Relay Module**

|  |
| --- |
| **รูปที่ 2.21** แสดงภาพของ 4-Channel Relay Module |

รีเลย์เป็นสวิตช์ที่ทำงานด้วยไฟฟ้าและเช่นเดียวกับสวิตช์อื่น ๆ ที่สามารถเปิดหรือปิดได้โดยปล่อยให้กระแสไหลผ่านหรือไม่ สามารถควบคุมด้วยแรงดันไฟฟ้าต่ำเช่น 3.3V ที่ให้มาโดย Microcontroller GPIO และช่วยให้สามารถควบคุมแรงดันไฟฟ้าสูงเช่น 12V, 24V หรือแรงดันไฟหลัก (220V)

รายละเอียดทางเทคนิคของ 4-Channel Relay Module มีดังนี้

* 4-Channel Relay breakout
* Power supply range from 5V~7.5V
* Onboard Photocoupler isolation
* Equiped with high-current relay, AC250V 10A ; DC30V 10A.
* Relay Output Indicator LED
* VCC is system power source, and JD\_VCC is relay power source. Ship 5V relay by default. Plug jumper cap to use
* TTL logic interface can be directly connected to microcontroller (Arduino, 8051, AVR, PIC, DSP, ARM, ARM, MSP430, etc )

จากรูปที่ 2.21 ประกอบด้วย relay จำนวน 4 ตัว แต่ละตัวมีสามช่อง common(COM), Normally Closed(NC), Normally Open(NO)

* COM เชื่อมต่อกับแหล่งจ่ายแรงดัน (220V, 12V, 24V)
* NC (Normally Closed) ใช้เมื่อต้องการให้รีเลย์ทำงานแบบ Normally Closed หมายความว่าวงจรปิด(กระแสไหล) เว้นแต่จะส่งสัญญาณจาก Microcontroller ไปยังโมดูลรีเลย์เพื่อเปิดวงจรและหยุดการไหลของกระแส
* NO (Normally Open) ใช้เมื่อต้องการให้รีเลย์ทำงานแบบ Normally Open หมายความว่าวงจรเปิด(กระแสไม่ไหล) เว้นแต่จะส่งสัญญาณจาก Microcontroller เพื่อปิดวงจร

ด้านแรงดันไฟต่ำ(ต่อกับ Microcontroller) ประกอบด้วย 6 พินดังนี้

* VCC แรงดันไฟฟ้าบวก
* GND Ground
* IN1-IN4 พอร์ตสำหรับควบคุมรีเลย์

|  |
| --- |
| **รูปที่ 2.22** แสดงภาพของ 4-channel relay schematic |

จากรูปที่ 2.22 เมื่อพอร์ตสัญญาณ(IN) อยู่ในระดับต่ำ(LOW) ไฟสัญญาณจะสว่างขึ้นและออปโตคัปเปลอร์ 817c จะแปลงแสงเป็นสัญญาณทางไฟฟ้า ออปโตคัปเปลอร์ 817c สามารถแยกสัญญาณไฟฟ้าอินพุตและเอาต์พุตออกจากกัน จากนั้นทรานซิสเตอร์จะทำงาน ขดลวดรีเลย์จะถูกกระแสไฟฟ้า และหน้าสัมผัส(Contact) ที่เปิดตามปกติ(Normally Open) ของรีเลย์จะปิดลง เมื่อพอร์ตสัญญาณ(IN) อยู่ในระดับสูง(HIGH) หน้าสัมผัส(Contact) ที่ปิดตามปกติ(Normally Close) ของรีเลย์จะปิด

|  |
| --- |
| **รูปที่ 2.23** แสดงภาพ 4-channel relay PCB |

**2.1.6** **จอแอลซีดี (I2C LCD)**

|  |
| --- |
| **รูปที่ 2.24** แสดงภาพของจอแอลซีดี (I2C LCD) |

I2C LCD ประกอบด้วยวงจรเสริมขนาดเล็กที่ติดตั้งที่ด้านหลังของโมดูล คือชิป PCF8574 ใช้สำหรับการสื่อสาร I2C และโพเทนชิออมิเตอร์เพื่อปรับแสงจอ (backlight) ข้อดีของ I2C LCD คือการต่อสายทำได้ง่ายใช้เพียงสองพินในการควบคุม LCD โดยทั่วไปแล้ว LCD มาตรฐานต้องการการเชื่อมต่อประมาณ 12 พินซึ่งอาจเป็นปัญหาได้หากคุณไม่มีพิน GPIO เหลือพอ

บัส I2C เป็นบัสอนุกรมที่คิดค้นโดย PHLIPS เป็นบัสอนุกรมประสิทธิภาพสูงซึ่งมีการกำหนดเวลาบัสและฟังก์ชันซิงโครไนซ์อุปกรณ์ความเร็วสูงหรือต่ำที่ระบบโฮสต์หลายระบบต้องการ บัส I2C มีเพียงสองสายสัญญาณสองทิศทาง Serial Data Line (SDA) และ Serial Clock Line (SCL)

ข้อมูลจำเพาะของ 16 × 2, 20 × 4 และ LCD ขนาดอื่น ๆ ส่วนใหญ่จะเหมือนกัน เพาะทั้งหมดใช้ตัวควบคุม HD44780 Hitachi LCD เหมือนกัน

รายละเอียดทางเทคนิคของ จอแอลซีดี (I2C LCD) มีดังต่อไปนี้

* Operating voltage 5 V
* Controller Hitachi HD44780 LCD controller
* Default address 0x27
* Screen resolution 2-lines × 16 characters
* Character resolution 5 × 8 pixels
* Module dimensions 80 × 36 × 12 mm
* Viewing area dimensions 64.5 × 16.4 mm

**2.1.7 เครื่องปั๊มน้ำ (Pump)**

เครื่องปั๊มน้ำ (Pump) เป็นอุปกรณ์ที่ช่วยส่งผ่านพลังงานจากแหล่งต้นกำเนิดไปยังของเหลว เพื่อทำให้ของเหลวเคลื่อนที่จาก ตำแหน่งหนึ่งไป ยังอีก ตำแหน่งหนึ่งที่อยู่สูงกว่า หรือในระยะทางที่ไกลออกไป โดยจุดเริ่มต้นของเครื่องปั๊มน้ำนี้มีประวัติศาสตร์ที่ยาวนานกว่า 2,000 ปีก่อนคริสตศักราช ซึ่งในช่วงเริ่มแรกมีการใช้พลังงาน ที่ได้จากมนุษย์ สัตว์ ต่อมาจึงได้ใช้พลังงานจากธรรมชาติ เช่น พลังงานจากลม และน้ำเป็นแหล่งต้นกำเนิด ซึ่งในช่วงแรกเพียง เพื่อการอุปโภคบริโภคและทำการเกษตรเท่านั้น

ในปัจจุบันเครื่องปั๊มน้ำจัดเป็นอุปกรณ์เครื่องมืออีกชนิดหนึ่งที่มีความเกี่ยวข้องกับชีวิตความเป็นอยู่ของมนุษย์อย่างมาก เป็นอุปกรณ์ ที่ช่วยจัดส่งน้ำเพื่อการอุปโภค บริโภค การเกษตร คมนาคม อุตสาหกรรม ตลอดจนการบำบัดน้ำเสีย เพื่อรักษา สภาวะแวดล้อม ที่ดีให้กับมนุษย์ ซึ่งวิวัฒนาการของเครื่องปั๊มน้ำในปัจจุบันได้เปลี่ยนไปจากเดิม ที่ใช้พลังงานจาก แหล่งธรรมชาติมาเป็น การใช้พลังงานจากไอน้ำ จากเครื่องยนต์ และที่นิยมกันมากคือ การใช้พลังงานไฟฟ้า เนื่องจากความสะดวกและง่ายต่อการใช้งาน

**2.1.7.1** **ปั๊มพ่นหมอก DC12V 4.8 บาร์**

|  |
| --- |
| **รูปที่ 2.25** แสดงภาพปั๊มพ่นหมอก DC12V 4.8 บาร์ |

รายละเอียดทางเทคนิคของ ปั๊มพ่นหมอก DC12V 4.8 บาร์ มีดังต่อไปนี้

* แรงดันไฟฟ้า : DC 12V
* กำลังไฟฟ้า: 24 วัตต์
* กินกระแส : 2 แอมป์
* แรงดัน : 4.8 บาร์
* อัตราการไหลสูงสุด : 3.5ลิตร / นาที หรือ 210 ลิตร / ชั่วโมง
* ท่อน้ำเข้า - ออก : 3 หุน ( 3/8 นิ้ว) สามารถใช้สาย 3 หุน สวมเข้ากับท่อเข้า-ออก ปั๊มน้ำได้เลย
* ระยะดูดลึก : ไม่เกิน 1.8 เมตร
* ระบบการทำงาน : ไม่มีสวิตช์แรงดันอัตโนมัติ ( NO Pressure switch ) เหมือนปั๊มน้ำอัตโนมัติ
* สามารถใช้งานกับแบตเตอรี่ 12V หรือแผงโซล่าเซลล์ไม่ควรต่ำกว่า 30 วัตต์ 12V หรือ 18V
* ขนาดโดยประมาณ 15\*8\*7ซม.
* น้ำหนัก 0.5กก.
* ควรใช้งานต่อเนื่องนานประมาณ 2 ชั่วโมง พัก 15 นาที เพื่อป้องกันการไหม้ของมอเตอร์
* ปั๊มน้ำไม่ควรโดนน้ำ เพราะจะทำให้น้ำซึมผ่านเข้าไปในมอเตอร์ ทำให้มอเตอร์ไหม้ได้
* สามารถใช้เป็นปั๊มพ่นยา พ่นปุ๋ย สามารถใช้สายพ่นยาได้ 50 เมตร
* สามารถพ่นได้ไกลสูงประมาณ 4เมตร

**2.1.8 พัดลม (Fan)**

พัดลม (Fan) เป็นอุปกรณ์ที่ช่วยให้เกิดการไหลของอากาศ ปัจจุบันพัดลมเป็นอุปกรณ์ที่มีใช้โดยทั่วไปในโรงงานอุตสาหกรรมและอาคารบ้านเรือน

**2.1.8.1 การทำงานของพัดลม**

การทำงาน และกลไกที่ทำให้พัดลมกับที่หรือหมุนส่ายไปมาพัดลมจะทำงานได้เมื่อกระแสไฟฟ้า เข้าสู่ระบบ และเมื่อกดปุ่มเลือกให้ลมแรงหรือเร็วตามที่ผู้ใช้ต้องการ กระแสไฟฟ้าจึงไหลเข้าสู่ ตัวมอเตอร์ ทำให้แกนมอเตอร์หมุนใบพัดที่ติดอยู่กับแกนก็หมุนตามไปด้วยจึงเกิดเป็น ลมพัดออกมา พัดลมตั้งโต๊ะจะมีราคาต่ำกว่าพัดลมตั้งพื้นและใช้พลังงานไฟฟ้าต่ำกว่า ทั้งนี้เพราะมีขนาดมอเตอร์และกำลังไฟฟ้าต่ำกว่าและพัดลมตั้งพื้นจะให้ลมแรงมากกว่า

**2.1.8.2 พัดลม 12 นิ้ว 10 ใบ 12V**

|  |
| --- |
| **รูปที่ 2.26** แสดงภาพพัดลม 12 นิ้ว 10 ใบ 12V |

รายละเอียดทางเทคนิคของ พัดลม 12 นิ้ว 10 ใบ 12V มีดังต่อไปนี้

* แรงดันไฟฟ้า : DC 12V
* กำลังไฟฟ้า: 80 วัตต์
* กินกระแส : 6.6 แอมป์
* ขนาด 12 นิ้ว
* 10 ใบพัด
* ใบพัดบาง น้ำหนักเบา มีตะแกรงกันการกระเด็น
* สามารถใช้ระบายความร้อนในรถยนต์ หรืองานดัดแปลงอื่นๆ

**2.1.9 LoRa (Long Range)**

LoRa มากจากคำว่า Long Range เป็นหนึ่งในเทคโนโลยีด้าน LPWAN (Low-Power Wide-Area Network) เครือข่ายสื่อสารแบบกว้างที่เน้นใช้พลังงานต่ำ เป็นเทคโนโลยีที่ถือว่าเป็นหัวใจสำคัญสำหรับงานทางด้าน IoT (Internet of Things) จุดเด่นของ LPWAN เทคโนโลยีได้แก่

* Secure — Bidirectional Communication
* Simple Star Network Topology ( ไม่จำเป็นต้องมี network ที่ซับซ้อนเช่น Mesh หรือ Repeater )
* Low data rate
* Low cost
* Long battery life

LoRa เป็นเทคโนโลยีการส่งข้อมูลแบบไร้สายในระยะไกล โดยใช้เทคนิค Proprietary Spread Spectrum technology ซึ่งรูปแบบถูกพัฒนาโดย Semtech Corporation ซึ่งมีย่านความถี่ในแต่ละภูมิภาคที่แตกต่างกันดังตารางที่ 2.1

**ตารางที่ 2.1** แสดงความถี่ที่ใช้สำหรับ LoRa (หน่วยเป็น MHz)

|  |  |
| --- | --- |
| ไทย | 920-925 |
| ยุโรป | 867-869 |
| อเมริกาเหนือ | 902-928 |
| จีน | 470-510 |
| เกาหลี | 920-925 |
| ญี่ปุ่น | 920-925 |
| อินเดีย | 865-867 |

โดยประกาศจาก กสทช. ที่เกี่ยวข้องกับเทคโนโลยี LPWAN ว่าถ้ามีกำลังส่งไม่เกิน 500 มิลลิวัตต์ ได้รับยกเว้นไม่ต้องได้รับใบอนุญาตให้ มี ใช้ และนำออก ซึ่งเครื่องวิทยุคมนาคมและใบอนุญาตให้ตั้งสถานีวิทยุคมนาคม แต่ไม่ได้รับยกเว้น ใบอนุญาตให้ทำนำเข้า และค้าซึ่งเครื่องวิทยุ คมนาคม

ถ้ามีกำลังส่งสูงกว่า 500 มิลลิวัตต์จะต้องได้รับใบอนุญาตวิทยุคมนาคมที่ เกี่ยวข้อง และการใช้คลื่นความถี่ดังกล่าว จะต้องได้รับอนุญาตให้ใช้คลื่นความถี่ ตามมาตรา ๔๕ แห่งพระราชบัญญัติองค์กรจัดสรรคลื่นความถี่ และ กำกับการประกอบกิจการวิทยุกระจายเสียง วิทยุโทรทัศน์ และกิจการโทรคมนาคม พ.ศ. ๒๕๕๓ และจะต้องได้รับ ใบอนุญาตประกอบกิจการโทรคมนาคมแบบที่ สาม

|  |
| --- |
| **รูปที่ 2.27** แสดงภาพตัวอย่างค่า Data Rate (DR) |

จากรูปที่ 2.27 ตัวอย่างค่า Data Rate ( DR ) สังเกตจากรูป จะเห็นว่า DR เป็น 0 อุปกรณ์จะสามารถส่งข้อมูลได้ไกลที่สุด โดยสามารถส่งด้วย Bitrate ที่ต่ำที่สุดโดยการกำหนด Data Rate จะถูกกำหนดจาก Spreading Factor ( SF ) ตั้งแต่ 7– 12 โดยที่แบนวิดท์ ช่องสัญญาณ และค่า SF ที่ปรับได้อาจจะเปลี่ยนแปลงตาม Frequency plan ของแต่ละโซน

|  |
| --- |
| **รูปที่ 2.28** แสดงภาพความสัมพันธ์ระหว่าง BITRATE และระยะห่างของอุปกรณ์กับ Gateway |

จากรูปที่ 2.28 จะเห็นว่าเมื่ออุปกรณ์เข้าใกล้ gateway มากก็จะสามารถที่จะส่งข้อมูลด้วย BITRATE ที่สูงขึ้นได้ และการส่งข้อมูลจะเร็วขึ้นอีกด้วย รวมถึงพลังงานที่ใช้ในการส่งถ้าเทียบต่อขนาดของแพกเกตก็จะน้อยกว่าอุปกรณ์ที่อยู่ไกล Gateway ซึ่งในระดับ LoRaWAN จะมีโหมด ADR ( Adaptive Data Rate ) ที่เซตในแพกเกตการส่งข้อมูลเพื่อให้การเชื่อมต่อระหว่าง Gateway และ Device สามารถปรับ Spreading Factor แบบอัตโนมัติเพื่อประสิทธิภาพในการส่งโดยดูจากระยะการเชื่อมต่อระหว่าง Gateway และ Device โดยสามารถเลือกได้ว่าต้องที่จะ ปรับเพื่อส่งข้อมูลได้เร็วที่สุด หรือ ปรับเพื่ออายุการใช้งานแบตเตอรี่ที่ยาวนานที่สุดเป็นต้น

**2.1.9.1 คุณสมบัติหลักของเทคโนโลยี LoRa**

|  |
| --- |
| **รูปที่ 2.29** แสดงภาพคุณสมบัติหลักของเทคโนโลยี LoRa |

จากรูปที่ 2.29 เทคโนโลยี LoRa มีคุณสมบัติต่างๆ ดังต่อไปนี้

* Long Range สามารถเชื่อมต่ออุปกรณ์ที่ห่างกันไม่เกิน 30 ไมล์ในพื้นที่ชนบทและเจาะเข้าไปในสภาพแวดล้อมในเมืองที่หนาแน่นหรือในที่ลึก
* Geolocation มีแอปพลิเคชันการระบุตำแหน่งที่ใช้พลังงานต่ำ
* Low Power ใช้พลังงานน้อยที่สุดพร้อมอายุการใช้งานแบตเตอรี่ที่ยาวนานถึง 10 ปีลดค่าใช้จ่ายในการเปลี่ยนแบตเตอรี่
* Mobile รักษาการสื่อสารกับอุปกรณ์ที่เคลื่อนไหวโดยไม่ต้องใช้พลังงานมากเกินไป
* Secure มีการเข้ารหัส AES128 แบบ end-to-end การรับรองความถูกต้องซึ่งกันและกัน
* High Capacity รองรับข้อความนับล้านต่อสถานีฐานตอบสนองความต้องการของผู้ให้บริการเครือข่ายสาธารณะที่ให้บริการในตลาดขนาดใหญ่
* Standardized ความสามารถในการทำงานร่วมกันของอุปกรณ์และความพร้อมใช้งานทั่วโลกของเครือข่าย LoRaWAN สำหรับการปรับใช้แอปพลิเคชัน IoT ได้อย่างรวดเร็วทุกที่
* Low Cost ลดการลงทุนโครงสร้างพื้นฐานค่าใช้จ่ายในการเปลี่ยนแบตเตอรี่และค่าใช้จ่ายในการดำเนินงาน

**2.1.9.2 ความแตกต่างระหว่าง LoRa และ LoRaWan**

LoRa ใช้งานได้เฉพาะโปรโตคอลระดับ Link layer เหมาะอย่างยิ่งที่จะใช้ในการสื่อสารแบบ P2P (point to point) ระหว่างโหนด

LoRaWan สามารถใช้งานโปรโตคอลระดับ Network layer ทำให้สามารถสื่อสารกับอุปกรณ์อื่นที่เชื่อมต่อกันผ่าน Cloud platform

**2.1.10 การเชื่อมต่ออุปกรณ์แบบ I2C (I2C)**

I2C หรือ I2C Bus ย่อมาจาก Inter Integrate Circuit Bus (IIC) นิยมเรียกสั้นๆว่า BUS (ไอ-แสคว-ซี-บัส) เป็นการสื่อสารอนุกรม แบบซิงโครนัส (Synchronous) เพื่อใช้ ติดต่อสื่อสาร ระหว่าง ไมโครคอนโทรลเลอร์ (MCU) กับอุปกรณ์ภายนอก ซึ่งถูกพัฒนาขึ้นโดยบริษัท Philips Semiconductors โดยใช้สายสัญญาณเพียง 2 เส้นเท่านั้น คือ serial data (SDA) และสาย serial clock (SCL) ซึ่งสามารถ เชื่อมต่ออุปกรณ์ จำนวนหลายๆ ตัว เข้าด้วยกันได้ ทำให้ไมโครคอนโทรลเลอร์ใช้พอร์ตเพียง 2 พอร์ตเท่านั้น

I2C BUS ใช้สายสัญญาณ 2 เส้น คือ SCL ,SDA สำหรับติดกับอุปกร์แบบ 2 ทิศทาง โดยที่ขาสัญญาณทั้ง 2 จะต้องต่อกับตัวต้านทานแบบ pull up 2-10K เนื่องจากเอาต์พุตมีลักษณะเป็น แบบ Open Darin หรือเป็นแบบ Open Collector เพื่อให้เอาต์พุตเชื่อมต่อกันได้หลายตัว

|  |
| --- |
| **รูปที่ 2.30** แสดงลักษณะการการเชื่อมต่ออุปกรณ์แบบ I2C BUS |

**2.1.10.1 การเขียน-อ่านข้อมูลกับอุปกรณ์แบบ I2C BUS**

|  |
| --- |
| **รูปที่ 2.31** แสดงรูปแบบการเขียนและอ่านข้อมูลแบบ I2C BUS |

จากรูปที่ 2.31 การรับ-ส่งข้อมูลแบบ I2C BUS ไมโครคอนโทรลเลอร์จะเริ่มต้นการส่งข้อมูลด้วยการส่ง สถานะเริ่มต้น (START Conditions) เพื่อแสดงการขอใช้บัส แล้วตามด้วย รหัสควบคุม (Control Byte) ซึ่งประกอบ ด้วยรหัส ประจำตัวอุปกรณ์ Device ID ,Device Address ,และ Mode ในการเขียนหรืออ่านข้อมูล เมื่ออุปกรณ์ รับทราบว่าไมโครคอนโทรลเลอร์ต้องการ จะติดต่อด้วยก็ต้องส่งสถานะรับรู้ (Acknowledge) หรือแจ้งให้ไมโครคอนโทรลเลอร์รับรู้ว่าข้อมูลที่ได้ส่งมามีความถูกต้อง และเมื่อสิ้นสุดการส่งข้อมูล MCU จะต้องส่ง สถานะสิ้นสุด ( STOP Conditions) เพื่อบอกกับอุปกรณ์ว่า สิ้นสุดการใช้บัส สถานะบัสว่าง คือเมื่อบัสไม่ได้ถูกใช้งาน ทั้ง SCL และ SDA จะเป็น 1 ทั้งคู่

**2.1.10.2 การกำหนดสถานะเริ่มต้นและสถานะสิ้นสุดของ I2C BUS (START and STOP Conditions)**

|  |
| --- |
| **รูปที่ 2.32** แสดง I2C BUS START and STOP Conditions |

จากรูปที่ 2.32 ลักษณะการกำหนดสถานะเริ่มต้นและสถานะสิ้นสุดของ I2C BUS เมื่อต้องการส่งข้อมูล ไมโครคอนโทรลเลอร์จะต้องส่งสถานะเริ่มต้น (START Conditions) คือให้ SDA เปลี่ยนจาก 1 มาเป็น 0 ในขณะที่ SCL มีค่าเป็น 1 และเมื่อสิ้นสุดการการใช้บัส ไมโครคอนโทรลเลอร์จะต้องส่งสถานะสิ้นสุด ( STOP Conditions) คือให้ SDA เปลี่ยนจาก 0 มาเป็น 1ในขณะที่ SCL มีค่าเป็น 1

**2.1.10.3 รหัสควบคุมของ I2C BUS (Control Byte)**

|  |
| --- |
| **รูปที่ 2.33** แสดงรหัสควบคุมของ I2C BUS (Control Byte) |

จากรูปที่ 2.33 รหัสควบคุมของ I2C BUS ประกอบด้วยรหัสประจำตัวของอุปกรณ์ (Device ID) ประกอบด้วยบิต 1-7 และบิต 0 เป็นบิตควบคุมการเขียนอ่าน รหัสประจำตัวของอุปกรณ์ ประกอบด้วยรหัสประจำตัวจากผู้ผลิต Product ID 4 บิต (บิต 4-7) ที่เปลี่ยนแปลงแก้ไขไม่ได้ และ Device Address 3 บิต (บิต 1-3) ซึ่งผู้ใช้ สามารถ กำหนด เองได้ รวมแลัวเป็นรหัส 7 บิท ใช้ระบุตัวอุปกรณ์ ที่ต่ออยู่บนบัส จะมีค่าซ้ำกันไม่ได้ บิตควบคุมการเขียนอ่าน (Mode) บิต 0 เมื่อ ไมโครคอนโทรลเลอร์ต้องการเขียนข้อมูลไปยังอุปกรณ์ก็กำหนดให้บิตนี้เป็น 0 และเมื่อต้องการ อ่านข้อมูล จากอุปกรณ์ ก็กำหนดให้บิตนี้เป็น 1

**2.1.10.4 ช่วงเวลารับส่งบิตข้อมูลของ I2C BUS**

|  |
| --- |
| **รูปที่ 2.34** แสดงช่วงเวลารับส่งบิตข้อมูลของ I2C BUS |

จากรูปที่ 2.34 สะภาวะการรับ-ส่งข้อมูล จะกระทำในขณะที่ขา SCL เป็น 1 และสะภาวะการเปลี่ยนแปลงข้อมูล จะกระทำในขณะที่ขา SCL เป็น 0

**2.1.11 วงจรแปลงสัญญาณอนาล็อกเป็นดิจิตอล (Analog to Digital Converter)**

การรับสัญญาณจากตัวตรวจจับ (Sensor) ที่เป็นสัญญาณอนาล็อกในรูปของแรงดันหรือกระแสที่เปลี่ยนแปลง เช่น ตัวตรวจจับอุณหภูมิ ความดัน แสง ฯลฯ มาประมวลผลด้วยตัวประมวลผลที่เป็นวงจร ดิจิตอล ไมโครโปรเซสเซอร์ หรือไมโครคอมพิวเตอร์ จะต้องมีการเปลี่ยนสัญญาณอนาล็อกจากตัวตรวจจับเหล่านั้นให้เป็นสัญญาณดิจิตอล โดยใช้วงจรที่เรียกว่า วงจร Analog to Digital Converter (ADC) ซึ่งวิธีการในการแปลงสัญญาณจากอนาล็อกเป็นดิจิตอลมีหลายวิธีด้วยกันตั้งแต่ใช้วงจรแปลงสัญญาณจากอนาล็อกเป็น Counter Ramp ADC, แบบ Linear Ramp ADC, แบบ Dual Slope ADC หรือแบบ Successive Approximation ADC ซึ่งวังจรดังกล่าวนี้อาจอยู่ในรูปของวงจรที่ประกอบจากวงจรเปรียบเทียบแรงดัน และวงจรเข้ารหัส หรือวงจรที่ประกอบจากวงจรเปลี่ยนเทียบแรงดันวงจรนับ วงจรเปลี่ยนสัญญาณจากดิจิตอลเป็นอนาล็อก รวมทั้งที่สร้างเป็นไอซีสําเร็จรูปที่ใช้สําหรับแปลงสัญญาณอนาล็อกเป็นดิจิตอลโดยเฉพาะ

วงจรแปลงสัญญาณอนาล็อกเป็นดิจิตอลเป็นวงจรที่ทําหน้าที่แปลงสัญญาณอนาล็อกที่อยู่ในรูปของแรงดันหรือกระแสที่เปลี่ยนแปลง ให้เป็นสัญญาณดิจิตอลเพื่อส่งไปยังส่วนประมวลผลที่ใช้วงจรดิจิตอลไมโครโปรเซสเซอร์ หรือไมโครคอมพิวเตอร์

|  |
| --- |
| **รูปที่ 2.35** แสดงบล็อกไดอะแกรมของระบบควบคุมที่ประกอบด้วยวงจร ADC ตัวประมวลผลและ  วงจร DAC |

จากรูปที่ 2.35 ทรานสดิวเซอร์จะตรวจจับการเปลี่ยนแปลงทางกายภาพ (Physical Variable) เช่นการเปลี่ยนแปลงของอุณหภูมิ, ความเข้มของแสง, อัตราการไหล, ความดัน และความเร็วแล้วทําการเปลี่ยนการเปลี่ยนแปลงทางกายภาพเหล่านั้นให้เป็นการเปลี่ยนแปลงของสัญญาณไฟฟ้าในรูปของแรงดันหรือกระแสที่ เป็นสัดส่วนโดยตรงกับการเปลี่ยนแปลงทางกายภาพนั้น ๆ อุปกรณ์ทรานสดิวเซอร์ที่สามารถตรวจจับ สัญญาณดังกล่าวได้แก่ เทอร์มิสเตอร์ (Thermister) ,โฟโต้ไดโอด (Photo Diode) , โฟโต้เซลล์ (Photo cell) , Flow Meter , Pressure Transducer หรือ Tachometer ตามลําดับ หลังจากนั้นจะส่งสัญญาณไฟฟ้าที่ได้ไปยัง วงจรแปลงสัญญาณอนาล็อกเป็นสัญญาณดิจิตอลเพื่อทําการแปลงสัญญาณไฟฟ้าดังกล่าวให้เป็นสัญญาณ ดิจิตอล แล้วส่งต่อไปยังส่วนควบคุมหรือส่วนประมวลผลที่เป็นวงจรดิจิตอลไมโครโปรเซอสเซอร์ หรือ ไมโครคอมพิวเตอร์ ซึ่งจะทําหน้าที่ประมวลผลสัญญาณดิจิตอลที่อินพุต ตามเงื่อนไขของโปรแกรมที่กําหนดไว้ จากนั้นส่งผลลัพธ์ที่ได้จากการประมวลผลในรูปของสัญญาณดิจิตอลไปที่เอาท์พุตซึ่งมีวงจรแปลงสัญญาณดิจิตอลเป็นอนาล็อกที่ทําการแปลงสัญญาณดิจิตอลให้เป็นอนาล็อกส่งไปยัง Actuator ที่ใช้ควบคุมการเปลี่ยนแปลงทางกายภาพ เช่น ควบคุมการเปิดปิดของวาล์ว ควบคุมการทํางานของ Heaterหรือควบคุมทิศทางความเร็ว หรือตําแหน่งของมอเตอร์ เป็นต้น

**2.1.11.1 การใช้งาน ADC กับไมโครคอนโทรลเลอร์ (ESP32)**

การอ่านค่าอนาล็อกด้วยไมโครคอนโทรลเลอร์ (ESP32) หมายความว่าจะสามารถอ่านค่าแรงดันระหว่าง 0V ถึง 3.3V ซึ่ง ESP32 สามารถอ่านค่าอนาล็อกด้วยความละเอียดสูงสุดถึง 12-bit

จากนั้นแรงดันไฟฟ้าที่วัดได้จะถูกกำหนดให้เป็นค่าระหว่าง 0 ถึง 4095 โดยที่ 0 V สอดคล้องกับ 0 และ 3.3 V สอดคล้องกับ 4095 แรงดันไฟฟ้าระหว่าง 0 V ถึง 3.3 V จะได้รับค่าที่สอดคล้องกันระหว่าง

|  |
| --- |
| **รูปที่ 2.36** แสดงภาพความละเอียดและแรงดันที่ ESP32 สามารถอ่านได้ |

ค่า ADC ไม่ใช่เชิงเส้น (Non-linear) ตามหลักการแล้วเมื่อใช้พิน ESP32 ADC อ่านค่าคาดว่าจะเป็นแบบเชิงเส้น (Linear) แต่ในการใช้งานจริงค่าที่ได้จะเป็นดังรูปด้านล่าง

|  |
| --- |
| **รูปที่ 2.37** แสดงภาพค่าอนาล็อกที่อ่านได้เมื่อใช้ ESP32 |

จากรูปที่ 2.37 จะเห็นว่า ESP32 จะไม่สามารถแยกความแตกต่างจาก 3.2V ถึง 3.3V กล่าวคือค่าความละเอียดที่ได้จะเป็นค่าเดียวกันคือ 4095 เช่นเดียวกันในระดับแรงดันที่ต่ำมากๆ จาก 0V ถึง 0.1V ค่าความละเอียดที่ได้จะเป็นค่าเดียวกันคือ 0

ฟังก์ชัน analogRead() การอ่านอินพุตอะนาล็อกด้วย ESP32 โดยใช้ Arduino IDE ทำได้ง่ายเพียงแค่ใช้ฟังก์ชัน analogRead () โดยการกำหนดขา GPIO ที่จะใช้อ่านค่าอนาล็อก analogRead(GPIO); ESP32 รองรับการวัดใน 18 ช่องสัญญาณ บอร์ด DEVKIT V1 DOIT มีเพียง 15 ตัวเท่านั้น (เวอร์ชันที่มี 30 GPIOs)

|  |
| --- |
| **รูปที่ 2.38** แสดงภาพ ESP32 Pinout Reference |

ไม่สามารถใช้พิน ADC2 เมื่อใช้ Wi-Fi ดังนั้นหากใช้ Wi-Fi และมีปัญหาในการรับค่าจาก ADC2 GPIO อาจพิจารณาใช้ ADC1 GPIO แทนซึ่งจะช่วยแก้ปัญหาได้

ฟังก์ชั่นที่มีประโยชน์อื่นๆ ฟังก์ชั่นขั้นสูงสำหรับใช้กับพิน ADC ที่สามารถเป็นประโยชน์ในการใช้งาน ESP32 ADC

* analogReadResolution(resolution) การกำหนดความละเอียดในการอ่านค่าอนาล็อก สามารถกำหนดความละเอียดได้ตั้งแต่ 9-bit (0-511) ถึง 12-bit ค่าความละเอียดเริ่มต้นคือ 12-bit
* analogSetWidth(width) มีวัตถุประสงค์การใช้งานเหมือน ฟังก์ชั่น analogReadResolution(resolution)
* analogSetCycles(cycles) กำหนดจำนวนรอบต่อตัวอย่าง (sample) ค่าเริ่มต้นคือ 8 (0-255)
* analogSetSamples(samples) กำหนดจำนวนตัวอย่าง (samples) ในช่วง ค่าเริ่มต้นคือ 1 ตัวอย่าง (sample) มีผลในการเพิ่มความไว (sensitivity)
* analogSetClockDiv(attenuation) ตั้งค่าตัวแบ่งสำหรับสัญญาณนาฬิกา ADC ค่าเริ่มต้นคือ 1 ช่วง (1-255)
* analogSetAttenuation(attenuation) ตั้งค่าการลดทอนอินพุตสำหรับพิน ADC ทั้งหมด ค่าเริ่มต้นคือ ADC\_11db การตั้งค่า ADC\_0db คือไม่มีการลดทอนอินพุต(1V input = ADC reading of 1088) การตั้งค่า ADC\_2\_5db ตั้งค่าการลดทอนเป็น 1.34 (1V input = ADC reading of 2086) การตั้งค่า ADC\_6db ตั้งค่าการลดทอน 1.5 (1V input = ADC reading of 2975) การตั้งค่า ADC\_11db sets an attenuation of 3.6 (1V input = ADC reading of 3959)
* analogSetPinAttenuation(pin, attenuation) ตั้งค่าการลดทอนอินพุตสำหรับพินที่ระบุ ค่าเริ่มต้นคือ ADC\_11db ค่าการลดทอนจะเหมือนกันจากฟังก์ชันก่อนหน้า
* adcAttachPin(pin) แนบ (Attach) พินเข้ากับ ADC (ล้างโหมดอะนาล็อกอื่น ๆ ที่อาจเปิดอยู่) ส่งคืนผลลัพธ์เป็นค่า TRUE หรือ FALSE
* adcStart(pin), adcBusy(pin) and resultadcEnd(pin) เริ่มการแปลง ADC บนบัส (bus) ของพินที่แนบ (Attach) มาตรวจสอบว่าการแปลงบนบัส ADC ของพินกำลังทำงานอยู่หรือไม่ (ส่งกลับค่า TRUE หรือ FALSE) รับผลลัพธ์ของการแปลง: ส่งกลับจำนวนเต็ม 16 บิต

**2.1.12 Pulse Width Modulation (PWM)**

Pulse Width Modulation(PWM) เป็นเทคนิคที่ทำให้สามารถอ่าน/เขียนข้อมูลแบบ analog ด้วยสัญญาณ digital ได้ โดยตัวควบคุมการสร้างสัญญาณดิจิตอล (Digital control) จะสร้างสัญญาณคลื่นสี่เหลี่ยมออกมาดังรูปที่ 2.10 โดยสัญญาณที่สร้างออกมาจะสลับกันระหว่าง เปิด(HIGH) กับ ปิด(LOW) รูปแบบสัญญาณเปิด-ปิดนี้สามารถจำลองเป็นแรงดันไฟฟ้าระหว่าง เปิด (5 Volts) กับ ปิด (0 Volts)

|  |
| --- |
| **รูปที่ 2.39** แสดงภาพสัญญาณเปิด(HIGH) และปิด(LOW) |

เรียกช่วงที่เป็นปิดหรือเปิดตามการทำงาน เพราะช่วงที่เป็นปิดแรงดันไฟฟ้าจะเป็น 0 ทำให้ไม่มีการทำงาน ส่วนช่วงที่เป็นเปิดคือช่วงที่มีแรงดันไฟฟ้า ทำให้อุปกรณ์ทำงานได้ซึ่งช่วงของเวลาที่สัญญาณเป็นเปิด เมื่อเทียบกับช่วงเวลาที่สัญญาณออกมาทั้งหมด จะเรียกช่วงเวลาตรงนี้ว่า ความกว้างของสัญญาณ (Pulse width) ดังรูปที่ 2.39 ดังนั้นเพื่อให้ได้ข้อมูลแบบ analog ที่ต่างกัน จะใช้ความกว้างของสัญญาณที่แตกต่างกันในการบ่งบอกค่าของข้อมูล analog

|  |
| --- |
| **รูปที่ 3.40** แสดงภาพตัวอย่างสัญญาณ PWM |

ความกว้างของเส้นสีเขียวจะแสดงถึงช่วงเวลาปกติ ระยะเวลาหรือช่วงเวลานี้เรียกว่า period เป็นค่าผกผันของความถี่อุปกรณ์ หากความถี่ของอุปกรณ์มีค่าประมาณ 50 Hz (ความถี่ขึ้นอยู่กับอุปกรณ์) ตวามกว้างของเส้นสีเขียวจะเป็นช่วงเวลา 20 ms (หาได้จาก T = 1/f) ถ้าอุปกรณ์ที่จะใช้มีการทำงานที่ความกว้างพัลซ์ 1000–2000 us การทำงาน 100% มีความหมายว่าเจอสัญญาณเปิด 2000 us ถ้ามีการทำงานเพียง 50% จะเจอสัญญาณเปิดเป็นเวลา 1500 us ควรรู้เกี่ยวกับเทคนิคนี้เพราะว่า การสั่งการด้วยสัญญาณ digital โดยตรงไม่สามารถที่จะควบคุมปริมาณได้เช่น ควบคุมความเร็วของมอเตอร์ ความสว่างของหลอดไฟ หรือสิ่งอื่น ๆ ที่ต้องควบคุมด้วยสัญญาณ analog แต่อย่างไรสัญญาณ PWM ก็ยังต้องใช้ 1 พอร์ต ต่อการส่งข้อมูลตัวเดียวอยู่ ซึ่งอาจทำให้เรามีช่องสัญญาณไม่พอหากต้องการส่งข้อมูลจำนวนมาก

**2.1.13 อุณหภูมิ (Temperature)**

คือการวัดค่าเฉลี่ยของพลังงานจลน์ของอนุภาคในสสารใด ๆ ซึ่งสอดคล้องกับความร้อนหรือเย็นของสสารนั้น ในอดีตมีแนวคิดเกี่ยวกับอุณหภูมิเกิดขึ้นเป็น 2 แนวทาง คือตามแนวทางของหลักอุณหพลศาสตร์ และตามการอธิบายเชิงจุลภาคทางฟิสิกส์เชิงสถิติ แนวคิดทางอุณหพลศาสตร์นั้น ถูกพัฒนาขึ้นโดยลอร์ดเคลวิน โดยเกี่ยวข้องกับการวัดในเชิงมหภาค ดังนั้นคำจำกัดความอุณหภูมิในเชิงอุณหพลศาสตร์ในเบื้องแรก จึงระบุเกี่ยวกับค่าตัวแปรต่าง ๆ ที่สามารถตรวจวัดได้จากการสังเกต ส่วนแนวทางของฟิสิกส์เชิงสถิติจะให้ความเข้าใจในเชิงลึกยิ่งกว่าอุณหพลศาสตร์ โดยอธิบายถึงการสะสมจำนวนอนุภาคขนาดใหญ่ และตีความพารามิเตอร์ต่าง ๆ ในอุณหพลศาสตร์ (เชิงมหภาค) ในฐานะค่าเฉลี่ยทางสถิติของพารามิเตอร์ของอนุภาคในเชิงจุลภาค

ในการศึกษาฟิสิกส์เชิงสถิติ สามารถตีความคำนิยามอุณหภูมิในอุณหพลศาสตร์ว่า เป็นการวัดพลังงานเฉลี่ยของอนุภาคในแต่ละองศาอิสระในระบบอุณหพลศาสตร์ โดยที่อุณหภูมินั้นสามารถมองเป็นคุณสมบัติเชิงสถิติ ดังนั้นระบบจึงต้องประกอบด้วยปริมาณอนุภาคจำนวนมากเพื่อจะสามารถบ่งบอกค่าอุณหภูมิอันมีความหมายที่นำไปใช้ประโยชน์ได้ ในของแข็ง พลังงานนี้พบในการสั่นไหวของอะตอมของสสารในสภาวะสมดุล ในแก๊สอุดมคติ พลังงานนี้พบในการเคลื่อนไหวไปมาของอนุภาคโมเลกุลของแก๊ส

**2.1.13.1 ความร้อนและอุณหภูมิ**

สสารทั้งหลายประกอบด้วย อะตอมรวมตัวกันเป็นโมเลกุล การเคลื่อนที่ของอะตอม หรือการสั่นของโมเลกุล ทำให้เกิดรูปแบบของพลังงานจลน์ ซึ่งเรียกว่า “ความร้อน” (Heat) เราพิจารณาพลังงานความร้อน (Heat energy) จากพลังงานทั้งหมดที่เกิดขึ้นจากการเคลื่อนที่ของอะตอมหรือโมเลกุลทั้งหมดของสสาร

อุณหภูมิ (Temperature) หมายถึง การวัดค่าเฉลี่ยของพลังงานจลน์ซึ่งเกิดขึ้นจากอะตอมแต่ละตัว หรือแต่ละโมเลกุลของสสาร เมื่อเราใส่พลังงานความร้อนให้กับสสาร อะตอมของมันจะเคลื่อนที่เร็วขึ้น ทำให้อุณหภูมิสูงขึ้น แต่เมื่อเราลดพลังงานความร้อน อะตอมของสสารจะเคลื่อนที่ช้าลง ทำให้อุณหภูมิลดต่ำลง

หากเราต้มน้ำด้วยถ้วยและหม้อบนเตาเดียวกัน จะเห็นได้ว่าน้ำในถ้วยจะมีอุณหภูมิสูงกว่า แต่จะมีพลังงานความร้อนน้อยกว่าในหม้อ เนื่องจากปริมาณความร้อนขึ้นอยู่กับมวลทั้งหมดของสสาร แต่อุณหภูมิเป็นเพียงค่าเฉลี่ยของพลังงานในแต่ละอะตอม ดังนั้นบรรยากาศชั้นบนของโลก (ชั้นเทอร์โมสเฟียร์) จึงมีอุณหภูมิสูง แต่มีพลังงานความร้อนน้อย เนื่องจากมีมวลอากาศอยู่อย่างเบาบาง

**2.1.14 ความชื้น (Humidity)**

ความชื้น (Humidity) หมายถึง ปริมาณไอน้ำที่มีอยู่ในอากาศ เมื่อน้ำได้รับความร้อนจะเปลี่ยนสถานะจากของเหลวกลายเป็นไอลอยขึ้นในอากาศ เรียกว่า การระเหย ซึ่งความร้อนที่ใช้ในการทำให้น้ำระเหยกลายเป็นไอ เรียกว่า ความร้อนแฝง (Latent Heat) เมื่ออากาศเย็นลงไอน้ำจะเริ่มกลั่นตัวเป็นละอองและคายความร้อนแฝงออกมาด้วย กาศจะได้รับไอน้ำมากหรือน้อยขึ้นอยู่กับอุณหภูมิ ดังนั้นอุณหภูมิจึงเป็นตัวกำหนดปริมาณไอน้ำในอากาศ อากาศที่มีอุณหภูมิสูงจะรับไอน้ำได้มากกว่าอากาศที่มีอุณหภูมิต่ำ ถ้าอากาศไม่สามารถรับไอน้ำได้เรียกว่า ไอน้ำอิ่มตัว (Saturate)

**2.1.14.1 ความชื้นในอากาศ**

ความชื้นในอากาสสามารถแบ่งออกเป็น 4 ประเภท คือ

1) ความชื้นสัมบูรณ์ (Absolute Humidity)เป็นน้ำหนักของไอน้ำที่มีอยู่จริงในปริมาตรของอากาศจำนวนหนึ่งคำนวณได้จากน้ำหนักของไอน้ำต่อหนึ่งหน่วยปริมาตรของอากาศ หรือกล่าวอีกนัยหนึ่งว่าความชื้นสัมบูรณ์คือความหนาแน่นของไอน้ำในอากาศ หน่วยที่ใช้มักเป็นกรัมต่อลูกบาศก์เมตรความชื้นสัมบูรณ์ไม่นิยมใช้ในทางอุตุนิยมวิทยาเพราะเมื่ออากาศลอยตัวขึ้นหรือจมตัวลงจะทำให้ปริมาตรของอากาศเปลี่ยนแปลงเนื่องจากบริเวณรอบๆ ความกดอากาศจะเปลี่ยนแปลงแม้ว่าไอน้ำที่มีอยู่ในอากาศจะคงที่

2) ความชื้นจำเพาะ (Specific Humidity) คือน้ำหนักหรือความชื้นที่มีอยู่ในอากาศ (Q) เป็นอัตราส่วนระหว่างน้ำหนักไอน้ำ (Mv) ต่อน้ำหนักของอากาศชื้น น้ำหนักของไอน้ำ (Mv) รวมกับน้ำหนักของอากาศ (Ma) มักใช้เป็นกรัมของน้ำต่อ 1 กิโลกรัมของอากาศชื้น ดังสมการ

Q = Mv/(Mv+Ma)

ความชื้นจำเพาะของอากาศจะมีค่าคงที่ เมื่ออากาศขยายตัวหรือหดตัว โดยที่ความชื้นจะไม่เปลี่ยนแปลงแม้ว่าปริมาตรของอากาศจะขยายตัวหรือหดตัวก็ตาม

3) ความชื้นสัมพัทธ์ (Relative Humidity) คืออัตราส่วนระหว่างน้ำหนักของไอน้ำที่มีอยู่จริงที่อุณหภูมิและความกดดันหนึ่งต่อน้ำหนักของไอน้ำอิ่มตัวที่อุณหภูมิและความกดดันนั้น คิดเป็นค่าร้อยละตัวอย่างเช่น อากาศ 1 ลูกบาศก์เมตร เมื่ออุณหภูมิ 30 องศาเซลเซียสมีไอน้ำอยู่ 9 กรัม และในอุณหภูมินั้นอากาศอิ่มตัวมีไอน้ำอยู่ 30 กรัม ความชื้นสัมพัทธ์เท่ากับ (100\*9)/30 เท่ากับ 30% ความชื้นสัมพัทธ์เป็นวิธีวัดความชื้นในอากาศที่ใช้มากที่สุด การเปลี่ยนแปลงความชื้นสัมพัทธ์จะไม่ทำให้ปริมาณไอน้ำที่มีอยู่ในอากาศเปลี่ยนแปลง แต่อณุหภูมิจะเปลี่ยนแปลง และถ้าอุณหภูมิเปลี่ยนแปลงความชื้นสัมพัทธ์จะเปลี่ยนแปลงด้วย

4) อุณหภูมิจุดน้ำค้าง (Dew Point Temperature) คืออุณหภูมิซึ่งอากาศถูกทำให้เย็นลง (ความกดอากาศคงที่) ถึงอุณหภูมิหนึ่งที่ที่ไอน้ำจุดอิ่มตัวพอดี อุณหภูมิของจุดน้ำค้างจะเป็นเท่าใดก็ได้ขึ้นอยู่กับจำนวนไอน้ำที่มีอยู่จริงในอากาศ ถ้าอากาศมีไอน้ำมากอุณหภูมิของจุดน้ำค้างจะสูง แต่ถ้าไอน้ำมีน้อยอุณหภูมิของจุดน้ำค้างจะต่ำ ถ้าอุณหภูมิของอากาศลดต่ำกว่าจุดน้ำค้างจะมีการกลั่นตัวในรูปของหยดน้ำ เช่น ในฤดูร้อนแก้วน้ำที่ใส่น้ำแข็งตั้งทิ้งไว้ความชื้นของอากาศจะรวมกันเป็นหยดน้ำเกาะอยู่รอบนอกแก้วน้ำ เนื่องจากอุณหภูมิของแก้วน้ำที่ใส่น้ำแข็งจะต่ำกว่าอุณหภูมิจุดน้ำค้างที่อยู่โดยรอบ อุณหภูมิของจุดน้ำค้างจะบอกถึงความไม่สะดวกสบายของมนุษย์ในช่วงที่มีอากาศอุ่นและชื้นได้ดีกว่าความชื้นสัมพัทธ์ ยกเว้นผู้ที่เคยชินกับอากาศร้อนชื้น คนส่วนใหญ่รู้สึกว่าอากาศชื้นไม่สะดวกสบายเมื่ออุณหภูมิจุดน้ำค้างสูงกว่า 17 องศาเซลเซียส ในขนะที่บางคนอาจไม่รู้สึกสบายตัว เมื่ออุณหภูมิจุดน้ำค้างสูงกว่า 20 องศาเซลเซียส

**2.1.14.2 ความชื้นสัมพัทธ์ของประเทศไทยในแต่ละฤดู**

ประเทศไทยตั้งอยู่ในเขตร้อนใกล้เส้นศูนย์สูตร จึงมีอากาศร้อนชื้นปกคลุมเกือบตลอดปี เว้นแต่บริเวณที่ลึกเข้าไปในแผ่นดิน ตั้งแต่ภาคกลางขึ้นไปความชื้นสัมพัทธ์จะลดลงชันเจน ในช่วงฤดูหนาวและฤดูร้อน โดยเฉพาะฤดูร้อนจะเป็นช่วงที่มีความชื้นสัมพัทธ์ลดลงต่ำที่สุดในรอบปี ในบริเวณดังกล่าวมีความชื้นสัมพัทธ์เฉลี่ยตลอดปีที่ 72-74 เปอร์เซ็นต์และจะลดลงเหลือ 62-69 เปอร์เซ็นต์ ในช่วงฤดูร้อน ดังแสดงข้อมูลตามตารางที่ 2.2

**ตารางที่ 2.2** แสดงสถิติความชื้นสัมพัทธ์เฉลี่ย (%) ของประเทศไทยในช่วงฤดูกาลต่างๆ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ภาค** | **ฤดูหนาว** | **ฤดูร้อน** | **ฤดูฝน** | **ตลอดปี** |
| เหนือ | 73 | 62 | 81 | 74 |
| ตะวันออกเฉียงเหนือ | 69 | 65 | 80 | 72 |
| กลาง | 71 | 69 | 79 | 73 |
| ตะวันออก | 71 | 74 | 81 | 76 |
| ใต้ฝังตะวันออก | 81 | 77 | 78 | 79 |
| ใต้ฝังตะวันตก | 77 | 76 | 84 | 80 |

**2.1.15 แสงสว่าง (Light)**

แสงสว่าง เป็นพลังงานรูปหนึ่ง ซึ่งประกอบด้วยคลื่นแม่เหล็กไฟฟ้า ที่มีความยาวคลื่นประมาณ 380 -780 นาโมเมตร ซึ่งเป็นระยะความยาวคลื่นที่มองเห็นได้ (Visible Light) การเปลี่ยนแปลงของความยาวคลื่นของแสงสว่าง จะทำให้ตารู้สึกเห็นเป็นสีต่างๆ ตามความยาวคลื่นนั้น

**ตารางที่ 2.3** แสดงความยาวคลื่นของแสงแต่ละสี

|  |  |
| --- | --- |
| **ความยาวคลื่น (นาโนเมตร)** | **สี** |
| 380-420 | ม่วงเข้ม |
| 420-440 | ม่วง |
| 440-470 | น้ำเงิน |
| 470-500 | ฟ้า |
| 500-550 | เขียว |
| 550-580 | เหลือง |
| 580-620 | ส้ม |
| 620-780 | แดง |

ความเข้มของแสงสว่างหรือปริมาณการส่องสว่าง (Illuminance) หมายถึง ปริมาณแสงสว่างที่ตกกระทบลงบนหนึ่งหน่วยพื้นที่ที่กำหนด

แหล่งกำเนิดของแสงสว่างมี 2 แหล่ง คือ

1. แสงสว่างจากธรรมชาติ (Natural Lighting) แหล่งกำเนิดของแสงสว่างในธรรมชาติที่สำคัญ คือ ดวงอาทิตย์ การใช้ประโยชน์จากแสงอาทิตย์อย่างเหมาะสม จะเป็นการประหยัดค่าใช้จ่ายได้มาก

2. แสงสว่างจากการประดิษฐ์ (Artificial Lighting) เป็นแหล่งกำเนิดแสงสว่างที่มนุษย์ได้ประดิษฐ์คิดค้นโดยอาศัยธรรมชาติและเทคโนโลยี ได้แก่ หลอดไฟฟ้าชนิดต่างๆ เช่น หลอดไฟฟ้าชนิดไส้หลอด, หลอดฟลูออเรสเซนต์,หลอดเมอคิวรี ,หลอดโซเดียม เป็นต้น

**2.1.16 ตัวต้านทานตรวจสอบกระแส (Current Sense Resistors)**

การตรวจสอบกระแสโดยการวัดแรงดันตกคร่อมตัวต้านทานเป็นวิธีทำสามารถทำได้ง่าย แต่ต้องพิจารณาปัจจัยต่างๆก่อนนำมาใช้งาน เช่น การเลือกค่าความต้านทาน ขนาดของตัวต้านทาน ความร้อน การวัดการไหลของกระแสมีพารามิเตอร์ที่สำคัญที่ต้องจัดการเพื่อให้ระบบทำงานอย่างมีประสิทธิภาพ วิธีที่ใช้กันทั่วไปในการตรวจสอบการไหลของกระแสทำได้โดยการนำตัวต้านทานที่มีค่าความต้านทานต่ำ ต่ออนุกรมกับโหลดแล้ววัดแรงดันตกคร่อมตัวต้านทานโดยใช้กฎของโอห์ม

V = IR

การใช้ตัวต้านทานในการตรวจสอบกระแสเป็นวิธีพื้นฐานที่มีประสิทธิภาพ แต่ยังมีปัญหาด้านการออกแบบรายละเอียดปลีกย่อยอีกมากมายที่ต้องคำนึงถึง เช่น ตำแหน่งของตัวต้านทานตรวจสอบกระแส ค่าความต้านทาน ขนาดของตัวต้านทาน และข้อพิจารณาเชิงกลอีกหลายประการ

**2.1.16.1 ตำแหน่งการวางตัวต้านทานตรวจสอบกระแส**

ตำแหน่งการวางตัวต้านทานตรวจสอบกระแสมีการวางอยู่ 2 แบบคือ การตรวจจับด้านต่ำ (Low-side Sensing) และ การตรวจับด้านสูง (High-side Sensing) ในการตรวจจับด้านต่ำจะวางตัวต้านทานตรวจสอบกระแสระหว่างโหลด (Load) และ กราวด์ (Ground) ดังรูปที่ 2.41 ซึ่งช่วยให้สามารถใช้วงจรขยายสัณญาณวัดแรงดันตกคร่อมตัวต้านทานตรวบสอบกระแส (Current Sense Resistors) และกราวด์ (Ground) ได้ง่าย

|  |
| --- |
| **รูปที่ 2.41** แสดงภาพการวางตัวต้านทานตรวจสอบกระแสแบบการตรวจจับด้านต่ำ  (Low-side Sensing) |

แบบที่ 2 เป็นการตรวจจับด้านสูง (High-side Sensing) โดยการวงตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) ระหว่างแหล่งจ่าย (Power) และ โหลด (Load) ดังรูปที่ 2.42 การตรวจจับแบบนี้จะใช้วงจรขยายสัญญาณแบบ differential amplifier หรือ instrumentation amplifier ในการวัดแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแส (Current Sense Resistors)

|  |
| --- |
| **รูปที่ 2.42** แสดงภาพการวางตัวต้านทานตรวจสอบกระแสแบบการตรวจจับด้านสูง  (High-side Sensing) |

**2.1.16.2 การเลือกตัวต้านทานตรวจสอบกระแส**

ในทางปฏิบัติโดยทั่วไปการเลือกตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) เมื่อนำตัวต้านทานไปต่อกับโหลด (Load) แรงดันตกคร่อมตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) ไม่ควรเกิน 100mV หรือต่ำกว่า เพื่อไม่ให้แบ่งแรงดันจากโหลดมากไป ค่าความต้านทานควรมีค่าประมาณ 10-mΩ หรือต่ำกว่า

**2.1.17 วงจรออปแอมป์ (Op-Amp Circuit)**

|  |  |
| --- | --- |
|  |  |

**รูปที่ 2.43** แสดงภาพวงจรออปแอมป์ (Op-Amp Circuit)

**2.1.17.1 คุณสมบัติของออปแอมป์**

ออปแอมป์ (Op-Amp) เป็นชื่อย่อสำหรับเรียกของวงจรขยายที่มาจากคำว่า Operational Amplifier เป็นวงจรขยายแบบต่อตรง (Direct coupled amplifier) ที่มีอัตราการขยายสูงมากใช้การป้อนกลับแบบลบไปควบคุมลักษณะการทำงาน ทำให้ผลการทำงานของวงจรไม่ขึ้นกับพารามิเตอร์ภายในของออปแอมป์ วงจรภายในประกอบด้วยวงจรขยายที่ต่ออนุกรมกันคือวงจรขยายดิฟเฟอเรนเชียลด้านทางเข้า วงจรขยายดิฟเฟอเรนเชียลภาคที่สอง วงจรเลื่อนระดับและวงจรขยายกำลังด้านทางออก สัญลักษณ์ที่ใช้แทนออปแอมป์จะเป็นรูปสามเหลี่ยม ไอซีออปแอมป์เป็นไอซีที่แตกต่างไปจากลิเนียร์ไอซีทั่วๆ ไปคือไอซีออปแอมป์มีขาอินพุท 2 ขาเรียกว่าขาเข้าไม่กลับเฟส(Non-Inverting Input) หรือขา (+) และขาเข้ากลับเฟส (Inverting Input) หรือขา (-) ส่วน ทางด้านออกมีเพียงขาเดียว เมื่อสัญญาณป้อนเข้าขาไม่กลับเฟสสัญญาณทางด้านออกจะมีเฟสตรงกับทางด้านเข้า แต่ถ้าป้อนสัญญาณเข้าที่ขาเข้ากลับเฟส สัญญาณทางออกจะมีเฟสต่างไป 180 องศา จากสัญญาณทางด้านเข้า

|  |
| --- |
| **รูปที่ 2.44** แสดงแผนภาพบล็อกการทำงานของออปแอมป์ |

คุณลักษณะของ OP-AMP ในอุดมคติ (Ideal Op-Amp)

* อัตราขยายของ Op-Amp แบบวงจรรอบเปิด (Open Loop Gain) มีค่าสูงมากจนเป็นอนันต์ AVOL = infinity
* Zin มีค่าสูงมากจนถือได้ว่าเป็น infinity
* Zout มีค่าต่ำมากจนถือได้ว่าเป็น 0
* อัตราขยายของ Op-Amp ไม่ขึ้นกับความถี่
* เมื่อ Vin = 0 จะได้ Vout เป็น 0 ด้วย

**2.1.17.2 วงจรขยายสัญญาณแบบกลับเฟส (Inverting Amplifier)**

ในวงจรขยายออปแอมป์นั้นสามารถที่จะกำหนดอัตราการขยายของวงจรได้โดยการใช้วงจรเนกาทีฟฟีดแบ็ค (Negative Feedback) เมื่อเราป้อนสัญญาณเข้าทางขากลับเฟส (ขา-)แรงดันด้านทางออกจะมีมุมเฟสต่างไปจากแรงดันทางเข้า 180 องศา ซึ่งมีลักษณะตรงข้าม สัญญาณตรงกันข้ามนี้จะถูกป้อนกลับผ่าน Rf เข้ามายังขาอินเวอร์ติงอีกครั้งหนึ่ง ตรงจุดนี้จะทำให้สัญญาณเกิดการหักล้างกันอัตราการขยายก็จะลดลง ถ้าตัวต้านทานที่เป็นตัวป้อนกลับมีค่ามาก จะทำให้สัญญาณป้อนกลับมีขนาดเล็กอัตราการขยายออกจึงสูง ถ้าตัวต้านทานที่ป้อนกลับมีค่าน้อยสัญญาณป้อนกลับไปได้มากอัตราการขยายก็จะลดลง ฉะนั้นอัตราส่วนของความต้านทาน Rin และ Rf จะเป็นตัวกำหนดอัตราการขยายของวงจรโดยไม่ขึ้นกับอัตราการขยายของออปแอมป์ ซึ่งสามารถหาอัตราการขยายแรงดันได้จากสูตร

=

=

|  |
| --- |
| **รูปที่ 2.45** แสดงภาพของวงจรขยายสัญญาณแบบกลับเฟส |

**2.1.17.3 วงจรขยายแบบไม่กลับเฟส (Non-inverting amplifier)**

วงจรขยายนี้เป็นวงจรขยายอีกแบบหนึ่งที่ต้องการเฟสในการขยายเป็นเฟสเดียวกัน ดังนั้นการป้อนสัญญาณอินพุทจึงต้องป้อนเข้าที่ขาอินพุทไม่กลับเฟส (+) ซึ่งเมื่อขยายออกที่เอาท์พุทแล้วจะได้สัญญาณเอาท์พุทที่มีเฟสเหมือนเดิม ดังนั้นในวงจรขยายแบบไม่กลับเฟสนี้การป้อนกลับเพื่อลดอัตราการขยายจึงยังคง ต้องป้อนไปยังขาอินเวอร์ติง (-) เพื่อให้เกิดการหักล้างของสัญญาณกันภายในตัวไอซีออปแอมป์ โดยสามารถหาอัตราการขยายของวงจรได้

|  |
| --- |
| **รูปที่ 2.46** แสดงภาพของวงจรขยายแบบไม่กลับเฟส |

**2.1.17.4 วงจรขยายสัญญาณแบบตามแรงดัน (Voltage Follower (Buffer)**

วงจรขยายสัญญาณแบบตามแรงดัน จะมีแรงดันทางด้านเอาท์พุตเท่ากับแรงดันทางด้านอินพุต

|  |
| --- |
| **รูปที่ 2.47** แสดงวงจรขยายสัญญาณแบบตามแรงดัน |

**2.1.17.5 วงจรเปรียบเทียบแรงดัน (Comparator)**

วงจรเปรียบเทียบแรงดันแสดงตามรูปที 2.ป เป็นวงจรแบบไม่เป็นเชิงเส้น ซึ่งจะให้เอาต์พุตอิ่มตัวออกมาที่ประมาณ 90% ของแหล่งจ่ายไฟทั้งด้านบวก และด้านลบ ขั้วของเอาต์พุตขึ้นอยู่กับเครื่องหมายของผลต่างอินพุต

|  |
| --- |
| **รูปที่ 2.48** แสดงวงจรเปรียบเทียบแรงดัน |

**2.1.18 เว็บแอปพลิเคชัน (Web Application)**

|  |
| --- |
| **รูปที่ 2.49** แสดงภาพตัวอย่างเว็บแอปพลิเคชัน |

เว็บแอปพลิเคชัน (Web Application) คือเว็บที่นำเอา โปรแกรมประยุกต์ (Application) มารวมไว้ในเว็บโดยการใช้ภาษาคอมพิวเตอร์เชิงโปรแกรมมิ่งเช่น PHP, NodeJS, Python, .NET, Java ฯลฯ เช่น การพัฒนาเว็บแอปพลิเคชันให้สามารถคำนวณได้เหมือนเครื่องคิดเลข

เว็บไชต์ (Website) — เป็นเว็บไซต์ที่ไว้นำเสนอข้อมูลความรู้ แต่ไม่สามารถคำนวณ หรือทำงานอย่างโปรแกรมคอมพิวเตอร์ได้

เว็บแอปพลิเคชัน (Web Application) — เป็นเว็บที่นำเอาโปรแกรมคอมพิวเตอร์มาใช้ในเว็บซึ่งเชื่อมต่อฐานข้อมูลเรียบร้อยแล้ว สามารถเปลี่ยนข้อมูลที่แสดงในเว็บได้อย่าง่ายดายโดยไม่ต้องแก้ไข HTML, CSS และ Javascript จะเห็นตัวอย่างได้ในเว็บต่าง ๆ เว็บขายของ ฯลฯ

จุดเด่นของเว็บแอปพลิเคชัน (Web Application)

* Web Application เหมาะกับองค์กรขนาดเล็กเพราะมีค่าใช้จ่ายต่ำ และคิดค่าใช้จ่ายตามจำนวนการใช้งานจริง
* การใช้งานในองค์กรทำได้ง่าย เพียงแค่มีเว็บบราวเซอร์ซึ่งเป็นสิ่งพื้นฐานในคอมพิวเตอร์ปัจจุบันแทบทุกเครื่องก็ใช้งานได้
* ข้อมูลจัดเก็บที่เดียว ง่ายต่อการจัดการ และไม่เกิดความซ้ำซ้อน
* ไม่ต้องการเครื่องคอมพิวเตอร์ประสิทธิภาพสูงซึ่งมีราคาแพง
* อยู่ที่ไหนก็ทำงานได้เพราะสามารถล๊อกอินเข้าใช้งานได้เลยไม่ต้องติดตั้งโปรแกรม
* ไม่ต้องมีบุคคลากรด้านเทคนิคเป็นของตัวเอง เพราะผู้ให้บริการดูแลเซิฟเวอร์และการบำรุงรักษาเองทั้งหมด
* ส่วน มากใช้ได้หลากหลายแพลทฟอร์มทั้ง Windows, Linux และ Mac ทำให้องค์กรสามารถเลือกใช้บางเครื่องเป็น Linux ได้เพื่อลดค่าใช้จ่ายด้านลิขสิทธ์
* เชื่อมต่อกับเว็บแอพหรือบริการออนไลน์อื่นๆได้ง่าย

**2.1.19 วินโดว์แอปพลิเคชัน (Windows Application)**

|  |
| --- |
| **รูปที่ 2.50** แสดงภาพตัวอย่างวินโดว์แอปพลิเคชัน |

Windows Application หรือ WinAPP เป็นโปรแกรมประเภท GUI (Graphic User Interface) เรียกง่ายๆก็คือ โปรแกรมประเภทนี้เป็นโปรแกรมที่จะติดต่อหรือตอบสนองกับผู้ใช้ (User) ผ่านทางระบบภาพ Graphic เพื่อให้มีการใช้งานที่ง่ายและเข้าใจได้ง่ายยิ่งขึ้นนั่นเอง ในสมัยก่อนนั้น โปรแกรมประเภทนี้ จะต้องผ่านการเขียนโปรแกรมขึ้นมาด้วยข้อความหรือ Text จากโปรแกรมอย่างเช่น Notepad หรือ Editplus แต่หากเป็นในปัจจุบัน มีโปรแกรมที่สามารถใช้วิธีคลิ๊กวาง ลากวางให้สามารถจัดการสร้างสรรค์โปรแกรมได้ง่ายยิ่งขึ้น แต่ถึงอย่างไรก็ควรที่จะมีการศึกษาในเรื่องของหลักการพัฒนาโปรแกรมไว้บ้าง เพื่อความสะดวกในการแก้ไขและจัดการโปรแกรมในภายหลัง

**2.1.20 ซอฟต์แวร์ที่เกี่ยวข้อง**

**2.1.20.1 ภาษา C++**

|  |
| --- |
| **รูปที่ 2.51** แสดงภาพตัวอย่างโลโก้ภาษา C++ |

C++ เป็นภาษาคอมพิวเตอร์เพื่อวัตถุประสงค์ทั่วไป ซึ่งสามารถเขียนโปรแกรมได้ทั้งแบบออบเจ็ค และการเขียนแบบปกติทั่วไป และยังมีเครื่องมืออำนวยความสะดวกในการจัดการและเข้าถึงระดับหน่วยความจำนอกจากนี้มันยังถูกนำไปใช้ในการเขียนโปรแกรมแบบต่างๆ มากมาย เช่น โปรแกรมคอมพิวเตอร์ ระบบฝังตัว (Embedded) ไมโครคอนโทรลเลอร์ 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

คอมไพเลอร์คือโปรแกรมคอมพิวเตอร์หรือกลุ่มของโปรแกรมที่แปลงซอสโค้ดที่เขียนขึ้นในภาษา C++ ไปเป็นภาษาคอมพิวเตอร์ (Target language) หลังจากที่ทำการแปลงแล้วจะได้ข้อมูลในรูปแบบของฐานสอง (Binary) ที่เรียกกันว่า Object code เหตุผลที่ต้องแปลงโปรแกรมจากภาษาเขียนโปรแกรมไปเป็นภาษาเครื่องโดยคอมไพเลอร์ก็เพื่อสร้างโปรแกรมที่สามารทำงานได้ (Executable program) คอมพิวเตอร์สามารถเข้าใจแค่ภาษาเครื่อง ภาษาที่ประกอบไปด้วยตัวเลข 1 และ 0 เราจำเป็นต้องใช้คอมไพเลอร์เพื่อแปลงโปรแกรมที่เราเขียนไปเป็นภาษาเครื่องที่ให้คอมพิวเตอร์สามารถทำงานได้คอมไพเลอร์ช่วยให้โปรแกรมเมอร์พัฒนาโปรแกรมของพวกเขาได้อย่างง่ายดายในการเขียนโปรแกรมด้วยภาษาระดับสูง อย่างเช่น ภาษา C++

**2.1.20.2 ภาษา Visual Basic.NET**

|  |
| --- |
| **รูปที่ 2.52** แสดงภาพโลโก้ภาษา Visual Basic.NET |

Visual Basic .NET (VB.NET) เป็นภาษารุ่นที่สามในการเขียนโปรแกรมแบบ event-driven programming (การเขียนโปรแกรมที่ขึ้นกับเหตุการณ์) ซึ่งมาพร้อมกับเครื่องมือพัฒนาจาก Microsoft เปิดตัวครั้งแรกในปี 1991 และได้รับการพัฒนาให้ดีมากขึ้นจนถึงปี 2008 โดย ภาษา Visual Basic นั้นถูกออกแบบมาเพื่อให้ง่ายต่อการเรียนรู้และง่ายต่อการใช้งาน ภาษา Visual Basic นั้นถูกพัฒนามาจากภาษา Basic ภาษาเขียนโปรแกรมที่เข้าใจง่ายสำหรับผู้เริ่มต้น

ภาษา Visual Basic สนับสนุนการพัฒนาโปรแกรมแบบ rapid application development (RAD) และ graphical user interface (GUI) การเข้าถึงฐานข้อมูล และอื่นๆ ที่ทำงานภายใต้ .NET Framework เวอร์ชันล่าสุดของ Visual Basic นั้นสนับสนุนการเขียนโปรแกรมเชิงวัตถุอย่างเต็มรูปแบบ และยังถูกออกแบบมาเพื่อช่วยให้ผู้เรียนสามารถเรียนรู้ได้อย่างรวดเร็ว โปรแกรมเมอร์สามารถสร้างโปรแกรม GUI อย่างง่ายไปจนถึงซับซ้อน การเขียนโปรแกรมใน VB นั้นสามารถออกแบบได้โดยการลากและวางส่วนประกอบต่างๆ ของตัวควบคุมหรือฟอร์ม การระบุแอตทริบิวต์เพิ่มเติมสำหรับส่วนประกอบเหล่านั้น และเขียนโค้ดเพิ่มเติมสำหรับการทำงาน เพราะว่า VB ได้มีการกำหนดค่าแอตทริบิวต์และฟังก์ชันพื้นฐานในการทำงานของโค้ด โปรแกรมเมอร์จึงไม่จำเป็นที่จะเขียนโค้ดเป็นจำนวนมากในการพัฒนาโปรแกรม ข้างล่างนี้เป็นตัวอย่างของหน้าออกแบบฟอร์มที่ว่างปล่าวใน Visual Studio 2019 ในภาษา Visual Basic

|  |
| --- |
| **รูปที่ 2.53** แสดงภาพตัวอย่างของหน้าออกแบบฟอร์มที่ว่างปล่าวใน Visual Studio 2019 |

VB 1.0 นั้นถูกแนะนำครั้งแรกในปี 1991 การออกแบบหน้าตาโดยการลากและวางนั้นได้รับการสืบทอดมาจากตัวสร้าง Prototype ที่พัฒนาโดย Alan Cooper และบริษัทของเขาชื่อว่า Tripod หละงจากนั้น Microsoft ได้ทำการติดต่อกับ Cooper เพื่อพัฒนา Tripod สำหรับระบบฟอร์มที่สามารถเขียนโปรแกรมได้ ภายใต้โค้ดที่มีชื่อว่า Ruby (ซึ่งไม่มีความสัมพันธ์กับภาษา Ruby ในปัจจุบัน) ซึ่ง Tripod ไม่ได้รวมเกี่ยวกับภาษาเขียนโปรแกรมทั้งหมด Microsoft จึงตัดสินใจที่จะรวม Ruby กับ Basic เพื่อสร้างภาษา Visual Basic โดย Interfaces ของ Ruby เป็นตัวสร้างส่วนที่มองเห็นได้ด้วยสายตา (Visual) ของ Visual Basic และได้นำไปรวมกับ "EB" Embedded BASIC engine ที่ออกแบบโดย Microsoft Ruby ยังให้ความสามารถในการเชื่อมโยงกับการโหลดไลบรารี่แบบไดนามิกส์ ที่สนับสนุนการควมคุมเพื่มเติมที่ภายหลังกลายมาเป็น VBX interface

**2.1.20.3 ภาษา PHP**

|  |
| --- |
| **รูปที่ 2.54** แสดงภาพโลโก้ของภาษา PHP |

PHP (พีเอชพี) ย่อมาจากคำว่า PHP Hypertext Preprocessor ภาษา PHP เป็นโปรแกรมภาษาคอมพิวเตอร์ระดับสูง ที่สามารถใช้งานในระบบเครือข่ายอินเทอร์เน็ตได้ มีความสามารถสูง และมีผู้นิยมใช้เป็นจำนวนมาก อีกทั้งยังสามารถดาวน์โหลดมาใช้ได้ฟรี จากเว็บไซต์ <http://php.net/>

PHP เป็นภาษาสคริปต์ ( Scripting Language ) คำสั่งต่างๆ จะเก็บในรูปของข้อความ (Text) สามารถใช้งานอิสระหรืออาจเขียนแทรกอยู่ในภาษา HTML ก็ได้ แต่ในการใช้งานจริงมักใช้งานร่วมกับภาษา HTML ดังนั้นการเขียนโปรแกรมนี้ต้องมีความรู้ด้านภาษา HTML เป็นอย่างดี อย่างไรก็ตามเราสามารถใช้โปรแกรมประยุกต์มาช่วยอำนวยความสะดวกในการสร้างงานได้ เช่น Macromedia Dreamweaver หรือโปรแกรมประเภท Editor เช่น EditPlus ฯลฯ โปรแกรมเหล่านี้จะช่วยจำแนกคำ เช่น คำสั่ง คำทั่วไป ตัวแปร ฯลฯ ให้มีสีต่างกันเพื่อสะดวกในการสังเกต และมีตัวเลขบอกบรรทัดทำให้สะดวกในการแก้ไข

PHP คือ ภาษาคอมพิวเตอร์ Server-Side Script ซึ่งใช้ในการจัดทำเว็บไซต์และสามารถประมวลผลออกมาในรูปแบบ HTML โดยมีรากฐานโครงสร้างคำสั่งมาจากภาษา ภาษาซี ภาษาจาวา และ ภาษาเพิร์ล เป้าหมายหลักของภาษาPHP คือให้นักพัฒนาเว็บไซต์สามารถเขียนเว็บเพจ ที่มีความตอบโต้ได้อย่างรวดเร็ว

ผู้ให้กำเนิด PHP มีชื่อว่า รัสมัส เลอร์ดอร์ฟ (Rasmus Lerdorf) โดยเริ่มต้นเขียนสคริปต์ Perl CGI ใส่ไว้ในโฮมเพจประวัติส่วนตัว และเห็นว่าการเขียน CGI ด้วย Perl มีความยุ่งยาก จึงได้เขียนโปรแกรมขึ้นใหม่ด้วยภาษา C ที่สามารถแยกส่วนที่เป็นภาษาHTML ออกจากส่วนที่เป็นภาษา C เพื่อแยกประมวลผล แล้วทำการสร้างโค้ด HTML ขึ้นใหม่ โดยตั้งชื่อโปรแกรมนี้ว่า Personal Home Page Tools (PHP-Tools) และได้เริ่มแจกจ่ายโค้ดออกไปในลักษณะฟรีแวร์ ต่อมาจึงได้เริ่มเปิดให้ผู้สนใจเข้าร่วมปรับปรุงและพัฒนา จนกลายป็นภาษา PHP ในปัจจุบัน

**2.1.20.4 ภาษา JavaScript**

|  |
| --- |
| **รูปที่ 2.55** แสดงภาพโลโก้ของ JavaScript |

JavaScript คือ ภาษาคอมพิวเตอร์สำหรับการเขียนโปรแกรมบนระบบอินเทอร์เน็ต ที่กำลังได้รับความนิยมอย่างสูง Java JavaScript เป็น ภาษาสคริปต์เชิงวัตถุ (ที่เรียกกันว่า "สคริปต์" (script) ซึ่งในการสร้างและพัฒนาเว็บไซต์ (ใช่ร่วมกับ HTML) เพื่อให้เว็บไซต์ของเราดูมีการเคลื่อนไหว สามารถตอบสนองผู้ใช้งานได้มากขึ้น ซึ่งมีวิธีการทำงานในลักษณะ "แปลความและดำเนินงานไปทีละคำสั่ง" (interpret) หรือเรียกว่า อ็อบเจ็กโอเรียลเต็ด (Object Oriented Programming) ที่มีเป้าหมายในการ ออกแบบและพัฒนาโปรแกรมในระบบอินเทอร์เน็ต สำหรับผู้เขียนด้วยภาษา HTML สามารถทำงานข้ามแพลตฟอร์มได้ โดยทำงานร่วมกับ ภาษา HTML และภาษา Java ได้ทั้งทางฝั่งไคลเอนต์ (Client) และ ทางฝั่งเซิร์ฟเวอร์ (Server)

JavaScript ถูกพัฒนาขึ้นโดย เน็ตสเคปคอมมิวนิเคชันส์ (Netscape Communications Corporation) โดยใช้ชื่อว่า Live Script ออกมาพร้อมกับ Netscape Navigator2.0 เพื่อใช้สร้างเว็บเพจโดยติดต่อกับเซิร์ฟเวอร์แบบ Live Wire ต่อมาเน็ตสเคปจึงได้ร่วมมือกับ บริษัทซันไมโครซิสเต็มส์ปรับปรุงระบบของบราวเซอร์เพื่อให้สามารถติดต่อใช้งานกับภาษาจาวาได้ และได้ปรับปรุง LiveScript ใหม่เมื่อ ปี 2538 แล้วตั้งชื่อใหม่ว่า JavaScript JavaScript สามารถทำให้ การสร้างเว็บเพจ มีลูกเล่น ต่าง ๆ มากมาย และยังสามารถโต้ตอบกับผู้ใช้ได้อย่างทันที เช่น การใช้เมาส์คลิก หรือ การกรอกข้อความในฟอร์ม เป็นต้น

เนื่องจาก JavaScript ช่วยให้ผู้พัฒนา สามารถสร้างเว็บเพจได้ตรงกับความต้องการ และมีความน่าสนใจมากขึ้น ประกอบกับเป็นภาษาเปิด ที่ใครก็สามารถนำไปใช้ได้ ดังนั้นจึงได้รับความนิยมเป็นอย่างสูง มีการใช้งานอย่างกว้างขวาง รวมทั้งได้ถูกกำหนดให้เป็นมาตรฐานโดย ECMA การทำงานของ JavaScript จะต้องมีการแปลความคำสั่ง ซึ่งขั้นตอนนี้จะถูกจัดการโดยบราวเซอร์ (เรียกว่าเป็น client-side script) ดังนั้น JavaScript จึงสามารถทำงานได้ เฉพาะบนบราวเซอร์ที่สนับสนุน ซึ่งปัจจุบันบราวเซอร์เกือบทั้งหมดก็สนับสนุน JavaScript แล้ว อย่างไรก็ดี สิ่งที่ต้องระวังคือ JavaScript มีการพัฒนาเป็นเวอร์ชั่นใหม่ๆออกมาด้วยดังนั้น ถ้านำโค้ดของเวอร์ชั่นใหม่ ไปรันบนบราวเซอร์รุ่นเก่าที่ยังไม่สนับสนุน ก็อาจจะทำให้เกิด error ได้

การทำงานของ JavaScript เกิดขึ้นบนบราวเซอร์ (เรียกว่าเป็น client-side script) ดังนั้นไม่ว่าคุณจะใช้เซิร์ฟเวอร์อะไร หรือที่ไหน ก็ยังคงสามารถใช้ JavaScript ในเว็บเพจได้ ต่างกับภาษาสคริปต์อื่น เช่น Perl, PHP หรือ ASP ซึ่งต้องแปลความและทำงานที่ตัวเครื่องเซิร์ฟเวอร์ (เรียกว่า server-side script) ดังนั้นจึงต้องใช้บนเซิร์ฟเวอร์ ที่สนับสนุนภาษาเหล่านี้เท่านั้น อย่างไรก็ดี จากลักษณะดังกล่าวก็ทำให้ JavaScript มีข้อจำกัด คือไม่สามารถรับและส่งข้อมูลต่างๆ กับเซิร์ฟเวอร์โดยตรง เช่น การอ่านไฟล์จากเซิร์ฟเวอร์ เพื่อนำมาแสดงบนเว็บเพจ หรือรับข้อมูลจากผู้ชม เพื่อนำไปเก็บบนเซิร์ฟเวอร์ เป็นต้น ดังนั้นงานลักษณะนี้ จึงยังคงต้องอาศัยภาษา server-side script อยู่ (ความจริง JavaScript ที่ทำงานบนเซิร์ฟเวอร์เวอร์ก็มี ซึ่งต้องอาศัยเซิร์ฟเวอร์ที่สนับสนุนโดยเฉพาะเช่นกัน แต่ไม่เป็นที่นิยมนัก)

**2.1.20.4.1 Ajax (Asynchronous JavaScript and XML)**

|  |
| --- |
| **รูปที่ 2.56** แสดงภาพสัญลักษณ์ของ Ajax |

Ajax (ย่อมาจาก Asynchronous JavaScript and XML) เป็นเทคนิคการเขียนโปรแกรมแบบนึง ไม่ใช่ภาษาโปรแกรมใหม่อย่างที่บางคนเข้าใจจากชื่อเต็มของ Ajax จะสังเกตว่ามีคีย์เวิร์ด 2 ตัวคือ (Asynchronous) JavaScript และ XML ก็คือ การใช้ JavaScript แบบ Asynchronous (แบบคำสั่งไม่เป็นลำดับขั้นหรือไม่ sync. กัน) และใช้ XML โดยมีหลักการทํางาน 2 ประเด็น คือ การ update หน้าจอแบบบางส่วน และการติดต่อสื่อสารกับ Server โดยใช้หลักการ Asynchronous ทําให้ผู้ใช้ไม่ต้องหยุดการทํางาน เพื่อรอการประมวลผลจาก Server รวมถึงการโหลดและการรีเฟรชหน้าจอ ของบราวเซอร์ทางฝั่ง Client มีการใช้ Ajax โดยการเพิ่มเลเยอร์ระหว่าง user browser กับ server ทําให้ผู้ใช้สามารถทํางานได้โดยไม่ต้องรอให้ Client ติดต่อไปยัง Server รวมถึงการโหลดและการรีเฟรชหน้าจอทั้งหมดด้วย ดังนั้นผู้ใช้สามารถใช้งาน application ได้อย่างมีประสิทธิภาพมากขึ้น

**2.1.20.5 HTML (Hyper Text Markup Language)**

|  |
| --- |
| **รูปที่ 2.57** แสดงภาพโลโก้ของ HTML |

HTML ย่อมาจาก Hyper Text Markup Language คือภาษาคอมพิวเตอร์ที่ใช้ในการแสดงผลของเอกสารบน website หรือที่เราเรียกกันว่าเว็บเพจ ถูกพัฒนาและกำหนดมาตรฐานโดยองค์กร World Wide Web Consortium (W3C) และจากการพัฒนาทางด้าน Software ของ Microsoft ทำให้ภาษา HTML เป็นอีกภาษาหนึ่งที่ใช้เขียนโปรแกรมได้ หรือที่เรียกว่า HTML Application

HTML เป็นภาษาประเภท Markup สำหรับการการสร้างเว็บเพจ โดยใช้ภาษา HTML สามารถทำโดยใช้โปรแกรม Text Editor ต่างๆ เช่น Notepad, Editplus หรือจะอาศัยโปรแกรมที่เป็นเครื่องมือช่วยสร้างเว็บเพจ เช่น Microsoft FrontPage, Dream Weaver ซึ่งอํานวยความสะดวกในการสร้างหน้า HTML ส่วนการเรียกใช้งานหรือทดสอบการทำงานของเอกสาร HTML จะใช้โปรแกรม web browser เช่น Microsoft Internet Explorer (IE), Mozilla Firefox, Safari, Opera, และ Netscape Navigator Google Chrome เป็นต้น

**2.1.20.6 CSS (Cascading Style Sheet)**

|  |
| --- |
| **รูปที่ 2.58** แสดงภาพโลโก้ของ CSS |

CSS ย่อมาจาก Cascading Style Sheet มักเรียกโดยย่อว่า "สไตล์ชีต" คือภาษาที่ใช้เป็นส่วนของการจัดรูปแบบการแสดงผลเอกสาร HTML โดยที่ CSS กำหนดกฏเกณฑ์ในการระบุรูปแบบ (หรือ "Style") ของเนื้อหาในเอกสาร อันได้แก่ สีของข้อความ สีพื้นหลัง ประเภทตัวอักษร และการจัดวางข้อความ ซึ่งการกำหนดรูปแบบ หรือ Style นี้ใช้หลักการของการแยกเนื้อหาเอกสาร HTML ออกจากคำสั่งที่ใช้ในการจัดรูปแบบการแสดงผล กำหนดให้รูปแบบของการแสดงผลเอกสาร ไม่ขึ้นอยู่กับเนื้อหาของเอกสาร เพื่อให้ง่ายต่อการจัดรูปแบบการแสดงผลลัพธ์ของเอกสาร HTML โดยเฉพาะในกรณีที่มีการเปลี่ยนแปลงเนื้อหาเอกสารบ่อยครั้ง หรือต้องการควบคุมให้รูปแบบการแสดงผลเอกสาร HTML มีลักษณะของความสม่ำเสมอทั่วกันทุกหน้าเอกสารภายในเว็บไซต์เดียวกัน โดยกฏเกณฑ์ในการกำหนดรูปแบบ (Style) เอกสาร HTML ถูกเพิ่มเข้ามาครั้งแรกใน HTML 4.0 เมื่อปีพ.ศ. 2539 ในรูปแบบของ CSS level 1 Recommendations ที่กำหนดโดย องค์กร World Wide Web Consortium หรือ W3C

ประโยชน์ของ CSS มีดังต่อไปนี้

* CSS มีคุณสมบัติมากกว่า tag ของ html เช่น การกำหนดกรอบให้ข้อความ รวมทั้งสี รูปแบบของข้อความที่กล่าวมาแล้ว
* CSS นั้นกำหนดที่ต้นของไฟล์ html หรือตำแหน่งอื่น ๆ ก็ได้ และสามารถมีผล กับเอกสารทั้งหมด หมายถึงกำหนด ครั้งเดียวจุดเดียวก็มีผลกับการแสดงผลทั้งหมด ทำให้เวลาแก้ไขหรือปรับปรุงทำได้สะดวก ไม่ต้องไล่ตามแก้ tag ต่างๆ ทั่วทั้งเอกสาร
* CSS สามารถกำหนดแยกไว้ต่างหากจาก ไฟล์เอกสาร html และสามารถนำมาใช้ร่วม กับเอกสารหลายไฟล์ได้ การแก้ไขก็แก้เพียง จุดเดียวก็มีผลกับเอกสารทั้งหมด

CSS กับ HTML / XHTML นั้นทำหน้าที่คนละอย่างกัน โดย HTML / XHTML จะทำหน้าที่ในการวางโครงร่างเอกสารอย่างเป็นรูปแบบ ถูกต้อง เข้าใจง่าย ไม่เกี่ยวข้องกับการแสดงผล ส่วน CSS จะทำหน้าที่ในการตกแต่งเอกสารให้สวยงาม เรียกได้ว่า HTML /XHTML คือส่วน coding ส่วน CSS คือส่วน design

**2.1.20.7 โปรแกรม Ngrok**

|  |
| --- |
| **รูปที่ 2.59** แสดงสัญลักษณ์ของโปรแกรม Ngrok |

Ngrok เป็น Tool Open Source พัฒนาโดย GitHub ซึ่งอำนวยความสะดวกให้บุคคลอื่นสามารถเข้าใช้งาน Website หรือ Application ที่กำลังทำงานอยู่บนเครื่อง Localhost นั่นเอง อีกทั้ง Ngrok ยังมีหลากหลายช่องสัญญาณที่มีความปลอดภัยในการรับและส่งข้อมูลจากเครื่องผู้ใช้ไปจนถึงเครื่อง Localhost

โดยบุคคลอื่นสามารถเข้าใช้งาน Website หรือ Application กำลังทำงานอยู่บนเครื่อง Localhost ผ่านทาง URL ของทาง Ngrok โดยที่ทาง Ngrok จะทำการสุ่มสร้าง URL ขึ้นมา และ URL ที่ได้มานั้น จะทำการเปลี่ยนไปทุกครั้งเมื่อมีการปิดหรือเปิดใช้งาน Ngrok

อีกทั้ง Ngrok ยังมีเว็บอินเตอร์เฟส ซึ่งสามารถใช้ตรวจสอบหรือ Monitor ผ่านทาง URL http://127.0.0.1:4040 ซึ่งสามารถใช้ตรวจสอบการรับส่งข้อมูล Http ทั้งหมด ไม่ว่าจะเป็นการรับส่งข้อมูลแบบ Get หรือแบบ Post, การขอ Request การส่ง Response, Traffic รวมถึง Webhook ที่เข้ามาใช้งานบนเครื่อง Localhost

ระบบปฏิบัติการ (Operating System) ที่ Ngrok รองรับมีดังต่อไปนี้

* Mac OS X
* Linux
* Mac (32-bit)
* Windows (32-bit)
* Linux (ARM)
* Linux (ARM64)
* Linux (32-bit)
* FreeBSD (64-bit)
* FreeBSD (32-bit)

**2.1.20.8 ระบบฐานข้อมูล (Database)**

|  |
| --- |
| **รูปที่ 2.60** แสดงสัญลักษณ์ของระบบฐานข้อมูล |

Database หรือ ฐานข้อมูล คือ กลุ่มของข้อมูลที่ถูกเก็บรวบรวมไว้ โดยมีความสัมพันธ์ซึ่งกันและกัน โดยไม่ได้บังคับว่าข้อมูลทั้งหมดนี้จะต้องเก็บไว้ในแฟ้มข้อมูลเดียวกันหรือแยกเก็บหลาย ๆ แฟ้มข้อมูล

ระบบฐานข้อมูล (Database System) คือ ระบบที่รวบรวมข้อมูลต่าง ๆ ที่เกี่ยวข้องกันเข้าไว้ด้วยกันอย่างมีระบบมีความสัมพันธ์ระหว่างข้อมูลต่าง ๆ ที่ชัดเจน ในระบบฐานข้อมูลจะประกอบด้วยแฟ้มข้อมูลหลายแฟ้มที่มีข้อมูล เกี่ยวข้องสัมพันธ์กันเข้าไว้ด้วยกันอย่างเป็นระบบและเปิดโอกาสให้ผู้ใช้สามารถใช้งานและดูแลรักษาป้องกันข้อมูลเหล่านี้ ได้อย่างมีประสิทธิภาพ โดยมีซอฟต์แวร์ที่เปรียบเสมือนสื่อกลางระหว่างผู้ใช้และโปรแกรมต่าง ๆ ที่เกี่ยวข้องกับการใช้ฐานข้อมูล เรียกว่า ระบบจัดการฐานข้อมูล หรือ DBMS (data base management system)มีหน้าที่ช่วยให้ผู้ใช้เข้าถึงข้อมูลได้ง่ายสะดวกและมีประสิทธิภาพ การเข้าถึงข้อมูลของผู้ใช้อาจเป็นการสร้างฐานข้อมูล การแก้ไขฐานข้อมูล หรือการตั้งคำถามเพื่อให้ได้ข้อมูลมา โดยผู้ใช้ไม่จำเป็นต้องรับรู้เกี่ยวกับรายละเอียดภายในโครงสร้างของฐานข้อมูล

ประโยชน์ของ ฐานข้อมูล มีดังต่อไปนี้

* ลดการเก็บข้อมูลที่ซ้ำซ้อน ข้อมูลบางชุดที่อยู่ในรูปของแฟ้มข้อมูลอาจมีปรากฏอยู่หลาย ๆ แห่ง เพราะมีผู้ใช้ข้อมูลชุดนี้หลายคน เมื่อใช้ระบบฐานข้อมูลแล้วจะช่วยให้ความซ้ำซ้อนของข้อมูลลดน้อยลง
* รักษาความถูกต้องของข้อมูล เนื่องจากฐานข้อมูลมีเพียงฐานข้อมูลเดียว ในกรณีที่มีข้อมูลชุดเดียวกันปรากฏอยู่หลายแห่งในฐานข้อมูล ข้อมูลเหล่านี้จะต้องตรงกัน ถ้ามีการแก้ไขข้อมูลนี้ทุก ๆ แห่งที่ข้อมูลปรากฏอยู่จะแก้ไขให้ถูกต้องตามกันหมดโดยอัตโนมัติด้วยระบบจัดการฐานข้อมูล
* การป้องกันและรักษาความปลอดภัยให้กับข้อมูลทำได้อย่างสะดวก การป้องกันและรักษาความปลอดภัยกับข้อมูลระบบฐานข้อมูลจะให้เฉพาะผู้ที่เกี่ยวข้องเท่านั้นซึ่งก่อให้เกิดความปลอดภัย(security) ของข้อมูลด้วย

**2.2 งานวิจัยที่เกี่ยวข้อง**

**2.2.1 ไอโอทีแพลทฟอร์มโรงเพาะเห็ด**

ขันติชัย, ชยุต (2560) ได้ศึกษา ออกแบบ และสร้างไอโอทีแพลทฟอร์มสำหรับใช้ในโรงเพาะเห็ด โดยการใช้ Raspberry Pi 3 Model B ซึ่งเป็น Microcontroller มาใช้วัดค่าอุณหภูมิและ ความชื้นจากเซ็นเซอร์ภายในโรงเรือน จากนั้นจึงส่งข้อมูลอุณหภูมิและความชื้นที่วัดได้ไปยัง Platform และเก็บข้อมูลอุณหภูมิและความชื้นลงบนฐานข้อมูลโดยใช้ MySQL เป็นทั้งฐานข้อมูล และตัวจัดการฐานข้อมูล นอกจากนั้นยังมีระบบ Web Application เพื่อใช้แสดงผลค่าอุณหภูมิและ ความชื้นที่ถูกส่งมาจาก Raspberry Pi หรือประวัติของค่าอุณหภูมิและความชื้นที่เก็บไว้ในฐานข้อมูล โดย Protocol ที่เป็นตัวกลางในการรับส่งข้อมูลระหว่าง Raspberry Pi, Platform และ Web Application คือ MQTT Protocol และใช้ร่วมกับ NodeJS ซึ่งเป็น Service หลักที่ใช้ภายใน Platform

**2.2.2 IOT for Smart Farm: A case study of the Lingzhi Mushroom Farm at Maejo University**

Oran, Anukit (2560) ได้นำเทคโนโลยีไอโอทีมาใช้ในการเพาะเห็ดหลินจือ โดยวัดค่าความชื้นในโรงเพาะเห็ดหลินจือ และแสดงผลบนโทรศัพท์และคอมพิวเตอร์ผ่าน NETPIE การควบคุมการทำงานของสปริงเกอร์และปั๊มหมอกเป็นแบบอัตโนมัติและแจ้งเตือนสถานะการทำงานของสปริงเกอร์และปั๊มหมอกผ่าน แอปพลิเคชัน LINE

**2.2.3 การประยุกต์ใช้เทคโนโลยีไอโอทีควบคุมฟาร์มอัจฉริยะในโรงเรือนเพาะเห็ดนางฟ้า**

วีรศักดิ์, สุรพงษ์ม, รัฐสิทธิ์ (2561) ได้ออกแบบระบบควบคุมอุณหภูมิและความชื้นสำหรับโรงเรือนเพาะเห็ดด้วย ไมโครคอนโทรลเลอร์โดยมีการศึกษาค้นคว้าข้อมูลที่เกี่ยวข้องต่อการเพาะเห็ดตลอดจนการออกแบบโครงสร้างโรงเรือนที่เหมาะสมโดยแบ่งการทดสอบออก 2 ส่วนคือการทดสอบในส่วนของระบบควบคุมและการทดสอบผลผลิต ของดอกเห็ดในโรงเรือนที่มีการควบคุมอุณหภูมิและความชื้นโดยนำก้อนเห็ดนางรมและเห็ดนางฟ้ามาทดสอบจำนวน 300 ก้อนและเปรียบเทียบประสิทธิภาพโรงเรือนเพาะเห็ดที่มีการควบคุมอุณหภูมิและความชื้นที่สร้างขึ้นกับโรงเรือน โดยประยุกต์ใช้ระบบไอโอทีที่ใช้เซ็นเซอร์วัดประกอบด้วย วัดอุณหภูมิ ความชื้นสัมพัทธ์ในโรงเรือนเห็ดนางฟ้าและควบคุมการเปิดปิดปั๊มน้ำให้สปริงเกอร์และพ่นหมอกแบบอัตโนมัติ และเซอร์วิสที่ใช้ในการ ส่งข้อมูลขึ้นระบบอินเตอร์เน็ตคือ NETPIE และเซอร์วิสย่อยคือ NETPIE freeboard ในการแสดงสถาณะความชื้น และเวลาแบบเรลไทม์(real time) และ NETPIE FEED ในการบันทึกข้อมูลความชื้นและเวลาและการดึงข้อมูลมาใช้งานคือ Node.JS ผ่านเซอร์วิส NETPIE REST API มาเป็นไฟล์ CSV ในส่วนแสดงสถาณะการทำงานของการให้น้ำ แบบสปริงเกอร์และพ่นหมอกแบบอัตโนมัติผ่านมือถือ

**2.2.4 IOT BASED DESIGN IMPLEMENTATION OF MUSHROOM FARM MONITORING USING ARDUINO MICROCONTROLLERS & SENSORS**

Parvati, Megha (2561) ในงานวิจัยฉบับนี้นำเสนอระบบการตรวจสอบและควบคุมสิ่งแวดล้อมเพื่อตรวจสอบและควบคุมสภาพแวดล้อมในฟาร์มเห็ด ช่วยให้ผู้ใช้สามารถตรวจสอบอุณหภูมิความชื้นความเข้มข้นของก๊าซคาร์บอนไดออกไซด์และความเข้มของแสงในฟาร์มเห็ดบนอุปกรณ์ Android โดยใช้แพลตฟอร์มออนไลน์ thing Speak ส่วนของตัวควบคุมสภาพแวดล้อมในฟาร์มเห็ดจะควบคุมสภาพแวดล้อมให้เป็นไปตามที่ผู้ใช้กำหนด ข้อมูลสถานะของสภาพแวดล้อมในฟาร์มเห็ดจะถูกส่งไปแสดงผลผ่าน ESP8266 WiFi modem

**บทที่ 3**

**วิธีการดำเนินงาน**

ในบทนี้จะกล่าวถึงขั้นตอนและวิธีการดำเนินงานจัดทำโครงงานตลอดจนการออกแบบโครงสร้างและส่วนต่างๆ ของโครงงาน ซึ่งได้รับการศึกษาข้อมูลและทฤษฎีที่เกี่ยวข้องในบทที่ 2 จึงได้นำข้อมูลที่ได้มาศึกษาและวิเคราะห์ออกแบบเพื่อเลือกใช้วัสดุอุปกรณ์ที่จะนำมาทำโครงงาน ให้มีความถูกต้องเหมาะสมกับลักษณะงานและดำเนินการสร้างโครงงานให้มีประสิทธิภาพและเป็นไปตามวัตถุประสงค์ของโครงงาน

**3.1 แผนการดำเนินงาน**

ในการดำเนินงานจัดทำโครงงานโรงเพาะเห็ดอัจฉริยะ (Smart Mushroom Farm) ผู้จัดทำได้วางแผนการดำเนินงานในการจัดทำโครงงานดังต่อไปนี้

1. ศึกษาทฤษฎีและงานวิจัยที่เกี่ยวข้อง
2. ศึกษา ออกแบบ โครงสร้างของระบบ
3. ศึกษาและออกแบบ เว็บแอปพลิเคชัน วินโดว์แอปพลิเคชัน และ ฐานข้อมูล
4. ศึกษาและออกแบบโรงเรือนสำหรับเพาะเห็ด
5. ทำการทดลองใช้งาน ไมโครคอนโทรลเลอร์ ร่วมกับอุปกรณ์ต่างๆ
6. ทำการสร้าง เว็บแอปพลิเคชัน และ วินโดว์แอปพลิเคชัน
7. ทำการสร้าง โรงเรือนสำหรับเพาะเห็ด และ ติดตั้งอุปกรณ์อุปกรณ์ต่างๆ
8. ทดสอบการทำงาน และ แก้ไขข้องบกพร่อง
9. จัดทำรูปเล่มรายงาน
10. นำเสนอผลงาน

**ตารางที่ 3.1** แสดงแผนการดำเนินงาน

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **กิจกรรม** | **ก.ค. 2563** | | | | **ส.ค. 2563** | | | | **ก.ย. 2563** | | | | **ต.ค. 2563** | | | |
| 1. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 4. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 5. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 6. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 7. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 8. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 9. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 10. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**3.2 โครงสร้างของระบบ**

โครงสร้างระบบของโครงงานนี้ประกอบด้วยส่วนต่างๆดังรูปที่ 3.1 และ รูปที่ 3.2 ซึ่งจะอธิบายโครงสร้างของระบบตามหัวข้อดังต่อไปนี้ 1. โรงเรือนเพาะเห็ด 2. Lora Communication 3.Windows Application 4. Web Application 5. Line Notify

ส่วนที่ 1 โรงเรือเพาะเห็ดส่วนนี้จะประกอบไปด้วยอุปกรณ์ฮาร์ดแวร์ที่ใช้ในระบบโดยแต่ละอุปกรณ์มีวัตถุประสงค์การนำมาใช้งานดังต่อไปนี้

* AM2315 คือเซ็นเซอร์ที่ใช้วัดอุณหภูมิและความชื้นภายในโรงเรือน สามารถใช้งานกับไมโครคอนโทรลเลอร์ ESP32LoRa ผ่านทาง I2C bus
* BH1750FVI คือเซ็นเซอร์ที่ใช้วัดความสว่างภายในโรงเรือน สามารถใช้งานกับไมโครคอนโทรลเลอร์ ESP32LoRa ผ่านทาง I2C bus
* ESP32LoRa คือไมโครคอนโทรลเลอร์ที่ใช้ควบคุมภายในระบบ เช่น การอ่านค่าจากเซ็นเซอร์ การควบคุมการทำงานของอุปณ์ เป็นต้น
* Current Sense Resistors คือตัวต้านทานที่มีค่าความต้านทานต่ำมากใช้สำหรับการสอบการไหลของกระแส เพื่อตรวจสอบสถานะการทำงานของอุปกรณ์
* I2C LCD คือจอที่ใช้แสดงผลข้อมูลต่างๆที่หน้าโรงเรือนเพาะเห็ด เช่น ค่าอุณหภูมิ ค่าความชื้น โหมดการทำงานของระบบ สถานะการทำงานของอุปกรณ์ เป็นต้นสามารถใช้งานกับไมโครคอนโทรลเลอร์ ESP32LoRa ผ่านทาง I2C bus
* Relay คืออุปกรณ์ที่ใช้ในการควบคุมการทำงานของพัดลมและปั๊ม
* Fan พัดลมระบายอากาศใช้ในการระบายอากาศภายในโรงเรือน
* Pump ปั๊มใช้ในการพ่นหมอกเพื่อเพิ่มความชื้นภายในโรงเรือน
* L298N คือโมดูลที่ใช้ในการควบคุมความสว่างของหลอด LED
* LED คือหลอดไฟที่ใช้ในการให้ความสว่างภายในโรงเรือนเพาะเห็ด

ส่วนที่ 2 Lora Communication ในโครงงานนี้ผู้จัดทำได้นำเทคโนโลยีการสื่อสาร Lora มาประยุกต์ใช้เพื่อเป็นตัวกลางการสื่อสารระหว่างโรงเรือนเพาะเห็ด (Mushroom Node) และพื้นที่ที่มีสัญญาณอินเทอร์เน็ต (STA Node) เพื่อแก้ปัญหาที่ตั้งของโรงเรือนไม่มีสัญญาณอินเทอร์เน็ต โดยเป็นการสื่อสารแบบ Point to Point หรือ Node-to-Node Communication ผ่านคลื่นความถี่ 433MHz

ส่วนที่ 3.Windows Application สามารถดูค่าสถานะและควบคุมระบบภายในโรงเรือนได้ผ่านทาง Serial Port ระหว่างคอมพิวเตอร์และไมโครคอนโทรลเลอร์ ใช้ภาษา Visual Basic ของ .NET Core ในการสร้าง Windows Application สามารถใช้งานได้กับไมโครคอนโทรลเลอร์ที่อยู่โรงเรือนเพาะเห็ด (Mushroom Node) และไมโครคอนโทรลเลอร์ที่ทำหน้าที่เป็น Web Application (STA Node) สามารถนำวินโดว์แอปพลิเคชันไปติดตั้งที่คอมพิวเตอร์เครื่องอื่นที่เป็น ระบบปฏิบัติการวินโดว์แล้วนำแอปพลิเคชันมาใช้งานได้ทันที

ส่วนที่ 4 Web Application ใช้ภาษา HTML CSS Javascript และ PHP ในการสร้าง Web Application และใช้แอปพลิเคชัน ngrok ที่ติดตั้งไว้บน Raspberry Pi ทำ Port forwarding ของ Web Application เพื่อให้สามารถใช้งานเว็บแอปพลิเคชันได้จากทุกที่ที่สามารถใช้งานอินเทอร์เน็ตได้ Web Application ส่วนที่แสดงค่าสถานะต่างๆและควบคุมการทำงานของระบบจะถูกเก็บอยู่ที่ ESP32LoRa และ Web Application ส่วนที่เกี่ยวกับฐานข้อมูล เช่น บันทึกข้อมูลลงในฐานข้อมูล นำข้อมูลในฐานข้อมูลมาแสดงบน Web Application เป็นต้น จะถูกเก็บอยู่ที่ Raspberry Pi 3 Model B เพราะใช้ Raspberry Pi เป็นฐานข้อมูล

ส่วนที่ 5 Line Notify เป็นบริการของ LINE ที่ให้สามารถส่งข้อความ การแจ้งเตือนต่างๆ ไปยังบัญชีหรือกลุ่มต่างๆ ผ่านทาง API ในโครงงานนี้ผู้จัดทำได้นำ Line Notify มาใช้ในการแจ้งเตือนสถานะต่างๆไปยัง Line Application เช่น การแจ้งเตือนเมื่อ Web Application เริ่มทำงานระบบจะส่ง URL ที่ได้จากโปรแกรม Ngrok เพื่อให้สะดวกต่อการเข้าใช้งาน Web Application การแจ้งเตือนเมื่ออุปกรณ์เปลี่ยนสถานะการทำงาน การแจ้งเมื่อมีการบันทึกข้อมูลลงในฐานข้อมูล การแจ้งเมื่อเปลี่ยนโหมดการทำงาน เป็นต้น



**รูปที่ 3.1** แสดงภาพโครงสร้างของระบบ

**3.3 การออกแบบวงจรตัวต้านทานตรวจสอบกระแส**

วงจรตัวต้านทานตรวจสอบกระแส มีวัตถุประสงค์เพื่อใช้การตรวจสอบสถานะการทำงาน ของ พัดลมระบายอากาศ ปั๊มพ่นหมอก ว่าทำงานจริงตามที่ควบคุมหรือไม่ โดยแสดงการออกแบบวงดังรูปด้านล่าง โดยมีหลักการทำงานคือนำตัวต้านทานที่มีค่าความต้านทาน 0.05 โอห์ม มาต่ออนุกรมกับ LOAD จากนั้นนำค่าแรงดันตกคร่อมตัวต้านทานมาขยายแรงดันโดยใช้วงจรขยายวงจรขยายแบบไม่กลับเฟส (Non-inverting Amplifier) เพื่อให้ไมโครคอนโทรลเลอร์สามารถอ่านค่าแรงดันได้

|  |
| --- |
| **รูปที่ 3.2** แสดงภาพวงจรตัวต้านทานตรวจสอบกระแส |

จากรูปที่ 3.2 แสดงภาพของแสดงภาพวงจรตัวต้านทานตรวจสอบกระแส โดยจะประกอบด้วย LOAD คือ พัดลมระบายอากาศ หรือ ปั๊มพ่นหมอก RS คือ ตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) และส่วนของวงจรขยายแบบไม่กลับเฟส (Non-inverting amplifier) ใช้เพื่อขยายแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแส

**3.3.1 การ****คำนวณแรงดันเอาท์พุทของวงจรเมื่อใช้งานกับพัดลมระบายอากาศ**

ในโครงงานนี้ผู้จัดทำใช้พัดลมระบายอากาศขนาด 12 นิ้ว 12V 6.6A ในการควบคุมสภาพแวดล้อมภายในโรงเรือนเพาะเห็ด ดังนั้นสามารถคำนวณแรงดันเอาท์พุทของวงจรตัวต้านทานตรวจสอบกระแส ได้จากสมการ

จากสมการด้านบน คือแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) ดังนั้นจึงต้องทราบค่า ก่อน

|  |
| --- |
| **รูปที่ 3.3** แสดงภาพวงจรตัวต้านทานตรวจสอบกระแสเมื่อใช้งานกับพัดลมระบายอากาศ |

จารรูปที่ 3.3 เพื่อให้ไมโครคอนโทรลเลอร์สมารถแรงดันเอาท์พุทของวงจรได้ โดยสามารถหาอัตราการขยายของวงจรได้จากสมการ

จากสูตร

จะได้

อัตราการขยาย () =

ดังนั้น

อัตราการขยาย () = 10.1 เท่า

สามารหาแรงดันตกคร่อมต้านทานตรวจสอบกระแส หรือ ได้จากกฎของโอห์มดังสมการ

I =

*ดังนั้นหาแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแสจากรูปที่ 3.33 กำหนดให้ กระแส (*I*)* = 6.6A *ความต้านทานของ*ต้านทานตรวจสอบกระแส (Rs) = 0.05 Ω

จากสูตร

I =

*จะได้*

Vin = IRs

*แทนค่า*

Vin = (6.6)(0.05)

*ดังนั้นแรงดันตกคร่องตัวต้านทานตรวจสอบกระแส (*Vin*) มีค่าเท่ากับ* 0.33V *เท่ากับ ดังนั้นสามารถคำนวณแรงดันเอาท์พุทของวงจรเมื่อใช้งานกับพัดลมระบายอากาศได้จากสมการ*

*แทนค่า*

*ดังนั้นแรงดันเอาท์พุทของวงจรเมื่อใช้งานกับพัดลมระบายอากาศ* (Vout) *เท่ากับ* 3.33V *สามารถคำนวณ**กำลัง*วัตต์ของตัวต้านทานตรวจสอบกระแส (Prs)

*จากสูตร*

P = VI

*จะได้*

Prs = Vin I

*แทนค่า*

Prs = (0.33)(6.6)

*ดังนั้นกำลัง*วัตต์ของตัวต้านทานตรวจสอบกระแส (Prs) เท่ากับ 2.178W

**3.3.2 การคำนวณแรงดันเอาท์พุทของวงจรเมื่อใช้งานกับปั๊มพ่นหมอก**

ในโครงงานนี้ผู้จัดทำใช้ปั๊มพ่นหมอกขนาดแรงดัน 4.8 bar / 70 PSI 12V 2A ในการควบคุมสภาพแวดล้อมภายในโรงเรือนเพาะเห็ด ดังนั้นสามารถคำนวณแรงดันเอาท์พุทของวงจรตัวต้านทานตรวจสอบกระแส ได้จากสมการ

จากสมการด้านบน คือแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) ดังนั้นจึงต้องทราบค่า ก่อน

|  |
| --- |
| **รูปที่ 3.4** แสดงภาพวงจรตัวต้านทานตรวจสอบกระแสเมื่อใช้งานกับปั๊มพ่นหมอก |

จารรูปที่ 3.4 เพื่อให้ไมโครคอนโทรลเลอร์สมารถแรงดันเอาท์พุทของวงจรได้ โดยสามารถหาอัตราการขยายของวงจรได้จากสมการ

จากสูตร

จะได้

อัตราการขยาย () =

ดังนั้น

อัตราการขยาย () = 31 เท่า

สามารหาแรงดันตกคร่อมต้านทานตรวจสอบกระแส หรือ ได้จากกฎของโอห์มดังสมการ

I =

*ดังนั้นหาแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแสจากรูปที่ 3.4 กำหนดให้ กระแส (*I*)* = 2A *ความต้านทานของ*ต้านทานตรวจสอบกระแส (Rs) = 0.05 Ω

จากสูตร

I =

*จะได้*

Vin = IRs

*แทนค่า*

Vin = (2)(0.05)

*ดังนั้นแรงดันตกคร่อง**ตัวต้านทานตรวจสอบกระแส (*Vin*) มีค่าเท่ากับ* 0.1V *เท่ากับ ดังนั้นสามารถคำนวณแรงดันเอาท์พุทของวงจรเมื่อใช้งานกับปั๊มพ่นหมอกได้จากสมการ*

*แทนค่า*

*ดังนั้นแรงดันเอาท์พุทของวงจรเมื่อใช้งานกับปั๊มพ่นหมอก (*Vout*) เท่ากับ* 3.1V *สามารถคำนวณ**กำลัง*วัตต์ของตัวต้านทานตรวจสอบกระแส (Prs)

*จากสูตร*

P = VI

*จะได้*

Prs = Vrs I

*แทนค่า*

Prs = (0.1)(2)

*ดังนั้นกำลัง*วัตต์ของตัวต้านทานตรวจสอบกระแส (Prs) เท่ากับ 0.2W

**3.4 การออกแบบฮาร์ดแวร์ระบบควบคุมสภาพแวดล้อมภายในโรงเรือนเพาะเห็ด**

ในการออกแบบฮาร์ดแวร์ระบบควบคุมสภาพแวดล้อมภายในโรงเรือนเพาะเห็ด ผู้จัดทำได้ใช้ AM2315 เป็นเซ็นเซอร์วัดอุณหภูมิและความชื้นภายในโรงเรือน ใช้ BH1750FVI เป็นเซ็นเซอร์วัดความสว่างของแสงภายในโรงเรือน ใช้พัดลมในการระบายอากาศ และปั๊มพ่นหมอกในการควบคุมอุณหภูมิและความชื้นภายในโรงเรือน ใช้หลอด LED ในการให้แสงสว่างภายในโรงเรือน ใช้ตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) สำหรับตรวจสอบสถานะการทำงานของพัดลมและปั๊มพ่นหมอก ใช้จอแอลซีดี (LCD) ในการแสดงผลค่าต่างๆ ภายในระบบ

|  |
| --- |
| **รูปที่ 3.5** แสดงภาพส่วนประกอบของฮาร์ดแวร์ระบบควบคุมสภาพแวดล้อมภายในโรงเรือน |

**3.4.1 การอ่านค่าอุณหภูมิและความชื้นจากเซ็นเซอร์ AM2315**

AM2315 คือเซ็นเซอร์วัดค่าอุณหภูมิและความชื้นคุณภาพสูงที่มีเอาท์พุทเป็นสัญญาณดิจิตอลโดยมีการสอบเทียบสัญญาณดิจิตอลเอาท์พุทแล้ว ใช้โมดูลตรวจจับอุณหภูมิละความชื้นคุณภาพสูงเพื่อให้แน่ใจว่ามีการวัดค่าที่แม่นยำและมีความทนทาน เซ็นเซอร์ประกอบด้วยตัววัดความชื้นประเภทความจุไฟฟ้า อุปกรณ์วัดอุณหภูมิที่มีความแม่นยำสูง และเชื่อมต่อด้วยไมโครคอนโทรลเลอร์ 8 บิตประสิทธิภาพสูง AM2315 สื่อสารด้วย I2C ผู้ใช้สามารถเชื่อต่อ I2C bus ได้โดยตรง

|  |
| --- |
| **รูปที่ 3.6** แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และ AM2315 |

**ตารางที่ 3.2** แสดงการต่อขาระหว่าง ESP32LoRa และ AM2315

|  |  |
| --- | --- |
| **ESP32LoRa** | **AM2315** |
| 5V | VCC |
| GND | GND |
| SDA(GPIO21) | SDA |
| SCL(GPIO22) | SCL |
| **รูปที่ 3.7** แสดงภาพโปรแกรมที่ใช้อ่านค่าจากเซ็นเซอร์ AM2315 | |

จากรูปที่ 3.7 เป็นโปรแกรมภาษา C++ ที่ใช้ในการอ่านค่าจากเซ็นเซอร์ AM2315 โดยมีการเรียกใช้ไลบรารี่ (Library) Wire.h และ Adafruit\_AM2315.h ต่อมาส่วนที่สำคัญที่สุดคือส่วนของการปลุกการทำงานของเซ็นเซอร์ดังรูปที่แสดงด้านบน หากไม่ปลุกการทำงานของเซ็นเซอร์เมื่อไม่มีการใช้งานเซ็นเซอร์เป็นเวลานานจะส่งผลให้ไม่สามารถใช้งานได้ ผลลัพธ์ที่อ่านค่าได้จากเซ็นเซอร์แสดงดังรูปที่ 3.8

|  |
| --- |
| **รูปที่ 3.8** แสดงภาพผลลัพธ์ที่อ่านได้จากเซ็นเซอร์ AM2315 |

**3.4.2 การอ่านค่าความสว่างจากเซ็นเซอร์ BH1750FVI**

เซ็นเซอร์ BH1750FVI เป็นเซ็นเซอร์ที่มีค่าความละเอียดในการอ่านสูงถึง 16 บิตเชื่อมต่อแบบ I2C ทำให้ประหยัดขอ GPIO ของไมโครคอนโทรลเลอร์ที่เชื่อมต่อ ให้ค่าในการวัดเป็นหน่วย lux (ลักซ์) ซึ่งเป็นหน่วยการวัดแบบ SI (The International System of Unit) สำหรับความสว่าง (illuminance) = 1 lumen per square meter

|  |
| --- |
| **รูปที่ 3.9** แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และ BH1750FVI |

**ตารางที่ 3.3** แสดงการต่อขาระหว่าง ESP32LoRa และ BH1750FVI

|  |  |
| --- | --- |
| **ESP32LoRa** | **BH1750FVI** |
| 5V | VCC |
| GND | GND |
| SDA(GPIO21) | SDA |
| SCL(GPIO22) | SCL |

|  |
| --- |
| **รูปที่ 3.10** แสดงภาพโปรแกรมที่ใช้อ่านค่าจากเซ็นเซอร์ BH1750FVI |

จากรูปที่ 3.10 เป็นโปรแกรมภาษา C++ ที่ใช้ในการอ่านค่าจากเซ็นเซอร์ BH1750FVI โดยมีการเรียกใช้ไลบรารี่ (Library) Wire.h และ BH1750.h ก่อนอ่านค่าจากเซ็นเซอร์ เริ่มต้นการทำงานของเซ็นเซอร์โดยการเรียกใช้ฟังก์ชัน Wire.begin(); และ lightMeter.begin(); ภายในฟังก์ชัน setup() จากนั้นทำการอ่านค่าโดยใช้ฟังก์ชัน lightMeter.readLightLevel(); มาเก็บในตัวแปล lux ที่มีประเภทเป็น float และแสดงค่าที่ได้ทาง Serial Monitor ในหน่วย lux (ลักซ์) ดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 3.11** แสดงภาพผลลัพธ์ที่อ่านได้จากเซ็นเซอร์ BH1750FVI |

**3.4.3 การควบคุมพัดลมและปั๊มพ่นหมอก**

การควบคุมพัดลมและปั๊มพ่นหมอกผู้จัดทำได้ใช้ 4-Channel Relay Module ในการควบคุมการทำงานเพราะพัดลมและปั๊มพ่นหมอกใช้แรงดันไฟเลี้ยง 12V 4-Channel Relay Module เป็น Relay แบบ Active LOW จะทำงานเมื่อสั่งขา GPIO ของ ไมโครคอนโทรลเลอร์ ESP32LoRa เป็น LOW

|  |
| --- |
| **รูปที่ 3.12** แสดงภาพ Schematic ของการควบคุมพัดลมและปั๊มพ่นหมอก |

**ตารางที่ 3.4** แสดงการต่อขาระหว่าง ESP32LoRa และ 4-Channel Relay Module

|  |  |
| --- | --- |
| **ESP32LoRa** | **4-Channel Relay Module** |
| 5V | DC + |
| GND | DC - |
| GPIO33 | IN4 |
| GPIO4 | IN3 |

|  |
| --- |
| **รูปที่ 3.13** แสดงภาพโปรแกรมส่วนที่ใช้ควบคุมการทำงานของ Relay |

จากรูปที่ 3.13 แสดงภาพโปรแกรมส่วนที่ใช้ควบคุมการทำงานของ Relay มีรายละเอียดคือ ตัวแปล pumpState และ ตัวแปล fanState เป็นตัวแปลประเภท Boolean ค่าสามารถมีค่าที่เป็นไปได้คือ 0 และ 1 หรือ true และ false โดยผู้จัดทำได้ออกแบบระบบให้พัดลมและปั๊มพ่นหมอกทำงานเมื่อตัวแปล pumpState และ ตัวแปล fanState มีค่าเท่ากับ true แต่ Relay ที่นำมาใช้จะทำงานเมื่อตัวแปล pumpState และ ตัวแปล fanState มีค่าเท่ากับ false ผู้จัดทำจึงได้เขียนโปรแกรมเพื่อให้ Relay และระบบสามารถทำงานร่วมกันได้โดยไม่ต้องแก้ไขทั้งระบบ

**3.4.3.1 การทำงานของปั๊มพ่นหมอกและพัดลมในโหมดการทำงานแบบอัตโนมัติ (AUTO)**

การทำงานของปั๊มพ่นหมอกและพัดลมในโหมดการทำงานแบบอัตโนมัติ (AUTO) จะพิจารณาตัวแปล 4 ตัวแปลคือ ค่าอุณหภูมิต่ำสุด ค่าอุณหภูมิสูงสุด ค่าความชื้นต่ำสุด ค่าความชื้นสูงสุด เพื่อรักษาระดับอุณหภูมิและความชื้นให้อยู่ในช่วงที่กำหนด สมมุติกำหนดให้ ค่าอุณหภูมิต่ำสุดเท่ากับ 25°C ค่าอุณหภูมิสูงสุดเท่ากับ 32°C ค่าความชื้นต่ำสุดเท่ากับ 80% ค่าความชื้นสูงสุดเท่ากับ 85%

**ตารางที่ 3.5** แสดงการทำงานของปั๊มพ่นหมอกและพัดลมในโหมดการทำงานแบบอัตโนมัติ (AUTO)

|  |  |  |  |
| --- | --- | --- | --- |
| **อุณหภูมิ(°C)** | **ความชื้น(%)** | **ปั๊มพ่นหมอก** | **พัดลม** |
| < 25 | < 80 | ON | OFF |
| < 25 | 80-85 | OFF | OFF |
| < 25 | > 85 | OFF | OFF |
| 25-32 | < 80 | ON | OFF |
| 25-32 | 80-85 | OFF | OFF |
| 25-32 | > 85 | OFF | OFF |
| > 32 | < 80 | ON | ON |
| > 32 | 80-85 | ON | ON |
| > 32 | > 85 | ON | ON |

|  |
| --- |
| **รูปที่ 3.14** แสดงภาพโปรแกรมที่ใช้ควบคุมการทำงานของปั๊มพ่นหมอกและพัดลม |

จากรุปที่ 3.14 แสดงภาพโปรแกรมภาษา C++ ที่ใช้ควบคุมการทำงานของปั๊มพ่นหมอกและพัดลมในโหมดการทำงานแบบอัตโนมัติ (AUTO) ตัวแปล ctrlMode คือตัวแปลโหมดการทำงานของระบบ ตัวแปล temp คือตัวแปลค่าอุณหภูมิ และตัวแปล humi คือตัวแปลค่าความชื้นที่อ่านได้จากเซ็นเซอร์ AM2315 ตัวแปล set\_temp\_min และ set\_temp\_max คือค่าอุณหภูมิต่ำสุดและสูงสุด ตัวแปล set\_humi\_min และ set\_humi\_max คือค่าความชื้นต่ำสุดและสูงสุด ตัวแปล pumpState และ fanState ค่าตัวแปลที่ใช้ในการควบคุมการทำงานของปั๊มพ่นหมอกและพัดลม

**3.4.4 การควบคุมความสว่างภายในโรงเรือนเพาะเห็ด**

ในการควบคุมความสว่างภายในโรงเรือนผู้จัดทำได้ใช้ LED ขนาด 5mm สีน้ำเงินจำนวน 112 หลอดเพื่อให้ความสว่าง จากนั้นใช้ L298N ในการควบคุมความสว่างโดยสัญญาณ PWM จากไมโครคอนโทรลเลอร์

|  |
| --- |
| **รูปที่ 3.15** แสดงภาพวงจร LED Driver |

จากรูปที่ 3.15 จะต้องคำนวณหาค่าของตัวต้านทานที่จะนำมาต่ออนุกรมกับหลอด LED เพื่อจำกัดกระแสตามความต้องการของหลอด LED โดยสามารถคำนวณได้จากสมการ โดยกำหนดให้ VLED = 3.2V, ILED = 20mA

V = VRLED + VLED

จะได้ V = ILEDRLED + VLED

RLED = (V - VLED) / ILED

แทนค่า RLED = (12V – 3.2V) / 20mA

RLED = 440 Ω

ดังนั้นค่าของตัวต้านทานที่จะนำมาต่ออนุกรมกับหลอด LED มีค่าเท่ากับ 440 โอห์ม

|  |
| --- |
| **รูปที่ 3.16** แสดงภาพ Schematic ของการควบคุมความสว่างภายในโรงเรือนเพาะเห็ด |

**ตารางที่ 3.6** แสดงการต่อขาระหว่าง ESP32LoRa และ L298N

|  |  |
| --- | --- |
| **ESP32LoRa** | **4-Channel Relay Module** |
| GPIO25 | IN1 |
| GND | GND |
| GND | IN2 |

|  |
| --- |
| **รูปที่ 3.17** แสดงภาพโปรแกรมที่ใช้ในการควบคุมความสว่างของหลอดไฟ |

จากรูปที่ 3.17 โปรแกรมจะทำการตรวจสอบว่าความสว่างต่ำกว่าหรือสูงกว่าค่าที่กำหนดหรือไม่ หากต่ำกว่าค่าที่กำหนดจะทำการเพิ่มค่า pwmvalue เพื่อเพิ่มความสว่างของหลอด LED หากสูงกว่าค่าที่กำหนดจะทำการลดค่า pwmvalue เพื่อลดความสว่างของหลอด LED

**3.4.5 การใช้จอแอลซี (LCD) แสดงผล**

ในโครงงานนี้ผู้จัดทำได้ใช้จอแอลซีดี (I2C LCD) ขนาด 16 x 2 ในการแสดงผลข้อมูลต่างๆในระบบ เช่น ค่าอุณหภูมิ ค่าความชื้น ค่าสถานะการทำงานของอุปกรณ์ โหมดการทำงานของระบบ เป็นต้น

|  |
| --- |
| **รูปที่ 3.18** แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และ จอแอลซีดี (LCD) |

**ตารางที่ 3.7** แสดงการต่อขาระหว่าง ESP32LoRa และ จอแอลซีดี (LCD)

|  |  |
| --- | --- |
| **ESP32LoRa** | **LCD** |
| 5V | VCC |
| GND | GND |
| SDA (GPIO21) | SDA |
| SCL (GPIO22) | SCL |

|  |
| --- |
| **รูปที่ 3.19** แสดงภาพโปรแกรมทดสอบการทำงานของจอแอลซีดี (LCD) |

จากรูปที่ 3.19 เป็นโปรแกรมภาษา C++ ที่ใช้ในการทดสอบการแสดงผลของจอแอลซีดีขนาด 16 x 2 การเชื่อมต่อแบบ I2C โดยกำหนด I2C Address ของจอแอลซีดี (LCD) เป็น Address 0x27 จากนั้นใช้ฟังก์ชัน lcd.begin() เพื่อเริ่มต้นการทำงานของ LCD ใช้ฟังก์ชัน lcd.serCursor() ในการกำหนดต่ำแหน่งเริ่มต้นของตัวอักษร และใช้ฟังก์ lcd.print() ในการแสดงตัวอักษร รูปที่แสดงด้านล่างแสดงผลลัพธ์ที่ได้จากโปรแกรมทดสอบการทำงานของจอแอลซีดี (LCD) แบบ I2C

|  |
| --- |
| **รูปที่ 3.20** แสดงภาพผลลัพธ์ที่ได้จากโปรแกรมทดสอบการทำงานของจอแอลซีดี (LCD) |

**3.4.6 การอ่านค่าแรงดันเอาท์พุทจากวงจรตัวต้านทานตรวจสอบกระแส**

ในโครงงานนี้ผู้จัดทำได้นำตัวต้านทานตรวจสอบกระแสมาประยุกต์ใช้ เพื่อใช้ในการตรวจสอบสถานะการทำงานของอุปกรณ์ เพื่อให้แน่ใจอุปกรณ์ทำงานตามที่ควบคุมหรือไม่ จากนั้นผู้จัดทำได้ใช้ ADC (Analog to Digital Converter) ของ ESP32LoRa ในการอ่านค่าแรงดันเอาท์พุทจากวงจรตัวต้านทานตรวจสอบกระแสดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 3.21** แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และวงจรตัวต้านทาน  ตรวจสอบกระแส |

**ตารางที่ 3.8** แสดงการต่อขาระหว่าง ESP32LoRa และวงจรตัวต้านทานตรวจสอบกระแส

|  |  |
| --- | --- |
| **ESP32LoRa** | **วงจรตัวต้านทานตรวจสอบกระแส** |
| GPIO25 | OUT (LM358P) PUMP |
| GPIO35 | OUT (LM358P) FAN |
| **รูปที่ 3.22** แสดงภาพโปรแกรมทดสอบการอ่านค่าแรงดันเอาท์พุทจากวงจรตัวต้านทานตรวจสอบ  กระแส | |

จากรูปที่ 3.22 แสดงภาพโปรแกรมที่ใช้ในการอ่านค่าแรงดันเอาท์พุทจากวงจรตัวต้านทานตรวจสอบกระแส โดยใช้ GPIO25 และ GPIO35 ของ ESP32LoRa ในการอ่านค่าแรงดันที่ได้จากเอาท์พุทของวงจร ใช้ ADC ความละในการอ่าน 12-bit จากนั้นนำค่าที่ได้มาคำนวลเป็นแรงดันที่สามารถอ่านได้จากเอาท์พุทของวงจร

**3.5 การออกแบบและสร้างโรงเรือนสำหรับเพาะเห็ด**

ในโครงงานนี้ผู้จัดทำได้ใช้โปรแกรม Google Sketchup ในการออกแบบโรงเรือนสำหรับเพาะเห็ดขนาด 200x150x200 เซนติเมตร ตามรูปที่ 3.23 โดยได้ออกแบบให้เป็นโรงเรือนแบบปิด สามารถควบคุมสภาพแวดล้อมต่างๆภายในโรงเรือนได้เช่น อุณหภูมิ ความชื้น และแสง โครงสร้างของโรงเรือนเพาะเห็ดเป็นเหล็กกล่องขนาด 1x1 นิ้ว ดังรูปที่ 3.24 เพื่อโรงเรือนมีความแข็งแรงและสามารถใช้งานได้ในระยะยาว เมื่อทำการเชื่อมโครงสร้างโรงเรือนเสร็จแล้ว ขั้นตอนต่อไปคือการพ่นสีเพื่อป้องกันการเกิดสนิม ดังรูปที่ 3.25

|  |
| --- |
| **รูปที่ 3.23** แสดงภาพการออกแบบโรงเรือนที่ใช้เพาะเห็ด |

|  |
| --- |
| **รูปที่ 3.24** แสดงภาพโครงสร้างของโรงเรือนเพาะเห็ด |
| **รูปที่ 3.25** แสดงภาพขั้นตอนการพ่นสีโรงเรือนเพาะเห็ด |

|  |
| --- |
| **รูปที่ 3.26** แสดงภาพขั้นตอนการติดลวดตาข่ายเพื่อป้องกันก้อนเห็ดหล่น |
| **รูปที่ 3.27** แสดงภาพขั้นตอนการคลุมโรงเรือนเพาะเห็ดเพื่อเก็บรักษาความชื้น |

|  |
| --- |
| **รูปที่ 3.28** แสดงภาพขั้นตอนการติดตั้งพัดลมระบายอากาศ |

|  |  |
| --- | --- |
| **รูปที่ 3.29** แสดงภาพขั้นตอนการติดตั้งหัวพ่นหมอก | |
| **รูปที่ 3.30** แสดงภาพขั้นตอนการคลุมโรงเรือนโดยใช้สแลนเพื่อป้องกันแสงแดดและลดความร้อน  จากแสงแดด | |
|  |  |
| **รูปที่ 3.31** แสดงภาพขั้นตอนการติดตั้งกล่องควบคุมระบบ | |
|  |  |
| **รูปที่ 3.32** แสดงภาพขั้นตอนการติดตั้งหลอด LED สีน้ำเงินเพื่อให้แสงสว่าง | |

|  |
| --- |
| **รูปที่ 3.33** แสดงภาพขั้นตอนการติดตั้งเซ็นเซอร์ BH1750FVI |

**3.6 การออกแบบการสื่อสารผ่าน Lora**

ในโครงงานนี้ผู้จัดทำได้นำเทคโนโลยีการสื่อสาร Lora มาประยุกต์ใช้เพื่อเป็นตัวกลางการสื่อสารระหว่างโรงเรือนเพาะเห็ด (Mushroom Node) และพื้นที่ที่มีสัญญาณอินเทอร์เน็ต (STA Node) เพื่อแก้ปัญหาที่ตั้งของโรงเรือนไม่มีสัญญาณอินเทอร์เน็ต โดยเป็นการสื่อสารแบบ Point to Point หรือ Node-to-Node Communication ดังรูปด้านล่าง

|  |
| --- |
| **รูปที่ 3.34** แสดงภาพการสื่อสารแบบ Node-to-Node |

จากรูปที่ 3.34 Node แต่ละ Node จะมี IP Address เป็นของตัวเองเพื่อใช้ในการระบุตัวตน เช่น Mushroom Node มี IP Address คือ X1 และ STA Node มี IP Address คือ X0

**3.6.1 การออกแบบการสื่อสาร Lora ที่ Mushroom Node**

ในการออกแบบการสื่อสารผ่าน Lora ที่ Mushroom Node ผู้จัดทำจะขออธิบายรายละเอียดต่างๆ โดยใช้ Flowchart ประกอบการอธิบาย

|  |
| --- |
| **รูปที่ 3.35** แสดงภาพ Flowchart การทำงานของ Mushroom Node (1) |

จากรูปที่ 3.35 แสดงการทำงานของ Mushroom Node เริ่มต้นระบบจะตรวจสอบว่าถ้ามีการเปลี่ยนแปลงของข้อมูลก็จะส่งข้อมูลการเปลี่ยนแปลง (C3) ไปยัง STA Node เช่น เมื่อผู้ใช้กดปุ่มเปลี่ยนโหมดการทำงานที่ Mushroom Node จากนั้น Mushroom Node ก็จะส่งข้อมูลการเปลี่ยนแปลงโหมดการทำงานไปยัง STA Node เพื่อให้ STA Node ทราบถึงการเปลี่ยนแปลงของข้อมูลโดยมีรูปแบบของข้อมูล (Data Format) ดังรูปที่ 3.36

|  |
| --- |
| **รูปที่ 3.36** แสดงภาพรูปแบบของข้อมูล (Data Format) |

จากรูปที่ 3.36 แสดงรูปแบบของข้อมูล (Data Format) ที่ส่งไปยัง STA Node เมื่อข้อมูลมีการเปลี่ยนแปลงโดยมีรายละเอียดดังต่อไปนี้

* X0 คือ IP Address ของ STA Node หรือ Node ปลายทางที่ต้องการส่งไปเพื่อให้ปลายทางหรือ STA Node ทราบว่าข้อมูลที่ส่งไปนั้นเป็นข้อมูลที่ต้องการส่งไปยังปลายทางหรือ STA Node
* X1 คือ IP Address ของ Mushroom Node หรือ Node ต้นทางโดยมีวัตถุประสงค์คือเพื่อให้ Node ปลายทางหรือ STA Node ทราบว่าข้อมูลที่ส่งมาเป็นของ Node ไหน ผู้จัดทำได้ออกแบบในลักษณะนี้เพื่อกรณีมี Node ต้นทางมากกว่า 1 Node
* T32.30 คือค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์ AM2315 โดย T คือ Keyword ที่ใช้บอกว่าค่าตัวเลขที่ต่อท้ายมาด้วยจำนวน 4 หลักเป็นค่าอุณหภูมิ
* H71.40 คือค่าความชื้นที่วัดได้จากเซ็นเซอร์ AM2315 โดย H คือ Keyword ที่ใช้บอกว่าค่าตัวเลขที่ต่อท้ายมาด้วยจำนวน 4 หลักเป็นค่าความชื้น
* 30 คือค่าอุณหภูมิสูงสุดที่ใช้ในการควบคุมอุณหภูมิ
* 80 คือค่าอุณหภูมิต่ำสุดที่ใช้ในการควบคุมความชื้น
* M0 คือโหมด (Mode) การทำงานของระบบควบคุมอุณหภูมิและความชื้นภายในโรงเรือนโดย M คือ Keyword ที่ใช้บอกว่าค่าตัวเลขที่ต่อท้ายมาด้วยจำนวน 1 หลักเป็นค่าโหมดการทำงานของระบบ 0 คือเป็นโหมดการทำงานแบบอัตโนมัติ (AUTO Mode) และถ้าเป็น 1 คือเป็นโหมดการทำงานแบบควบคุมเอง (MANUAL Mode)
* P1 คือค่าสถานะการทำงานของปั๊ม (Pump) ที่อ่านได้จากวงจรตัวต้านทานตรวจสอบกระแสโดย P คือ Keyword ที่ใช้บอกว่าค่าตัวเลขที่ต่อท้ายมาด้วยจำนวน 1 หลักเป็นค่าสถานะการทำงานของปั๊ม (Pump) 1 คือค่าที่บอกว่าปั๊ม (Pump) กำลังทำงานถ้าเป็น 0 คือค่าที่บอกว่าปั๊ม (Pump) ไม่มีการทำงาน
* F1 คือค่าสถานะการทำงานของพัดลม (Fan) ที่อ่านได้จากวงจรตัวต้านทานตรวจสอบกระแสโดย F คือ Keyword ที่ใช้บอกว่าค่าตัวเลขที่ต่อท้ายมาด้วยจำนวน 1 หลักเป็นค่าสถานะการทำงานของพัดลม (Fan) 1 คือค่าที่บอกว่าพัดลม (Fan) กำลังทำงานถ้าเป็น 0 คือค่าที่บอกว่าพัดลม (Fan) ไม่มีการทำงาน
* 20 คือค่าอุณหภูมิต่ำสุดที่ใช้ในการควบคุมอุณหภูมิ
* 85 คือค่าอุณหภูมิสูงสุดที่ใช้ในการควบคุมความชื้น
* L0200.33 คือค่าความสว่างที่วัดได้จากเซ็นเซอร์ BH1750FVI โดย L คือ Keyword ที่ใช้บอกว่าค่าตัวเลขที่ต่อท้ายมาด้วยจำนวน 8 หลักเป็นค่าความสว่าง
* 200 คือค่าความสว่างที่กำหนดไว้

การส่งข้อมูลการเปลี่ยนแปลงไปยัง Node ปลายทางหรือ STA Node จะเรียกใช้ฟังก์ชัน sendUpdateData() ในการส่งข้อมูล ฟังก์ชัน sendUpdateData() เป็นฟังก์ชันที่เขียนด้วยภาษา C++ ดังรูปที่ 3.37

|  |  |
| --- | --- |
|  |  |
| **รูปที่ 3.37** แสดงภาพฟังก์ชัน sendUpdateData() | |

จากรูปที่ 3.37 แสดงฟังก์ sendUpdateData() ที่ใช้ส่งข้อมูลเมื่อมีการเปลี่ยนแปลงโดยรายละเอียดมีการทำงานดังต่อไปนี้ ก่อนส่งข้อมูลจะตรวจสอบค่าแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแสก่อนเพื่อให้ได้ค่าสถานะการทำงานของอุปกรณ์ที่ตรงกับค่าจริง จากนั้นจะส่งข้อมูลที่มีรูปแบบ (Data Format) เหมือนรูปที่ 3.36 ตัวแปลต่างๆที่ถูกใช้ในฟังก์ชันมีรายละเอียดดังต่อไปนี้

* Rs\_pump\_voltage แรงดันที่อ่านได้จากวงจรตัวต้านทานตรวจสอบกระแสที่ใช้กับปั๊ม (Pump) ผ่าน ADC Pin
* Rs\_fan\_voltage แรงดันที่อ่านได้จากวงจรตัวต้านทานตรวจสอบกระแสที่ใช้กับพัดลม (Fan) ผ่าน ADC Pin
* pump\_check ค่าที่บอกสถานะการทำงานของปั๊ม (Pump)
* fan\_check ค่าที่บอกสถานการณ์ทำงานของพัดลม (Fan)
* des IP Address ของ STA Node หรือ Node ปลายทาง
* ipAddr IP Address ของ Mushroom Node หรือ Node ต้นทาง
* temp ค่าอุณหภูมิที่วัดได้จากเซ็นเซอร์ AM2315
* tempStr ค่าที่ใช้ส่งแทนค่า temp ในกรณีที่อุณหภูมิต่ำกว่า 10.00 เพื่อป้องกันความยาวของข้อมูลคลาดเคลื่อน
* humi ค่าความชื้นที่วัดได้จากเซ็นเซอร์ AM2315
* humiStr ค่าที่ใช้ส่งแทนค่า humi ในกรณีที่ความชื้นต่ำกว่า 10.00 เพื่อป้องกันความยาวของข้อมูลคลาดเคลื่อน
* set\_temp\_max คือค่าอุณหภูมิสูงสุดที่ใช้ในการควบคุมอุณหภูมิ
* set\_humi\_min คือค่าอุณหภูมิต่ำสุดที่ใช้ในการควบคุมความชื้น
* ctrlMode คือโหมด (Mode) การทำงานของระบบควบคุมอุณหภูมิและความชื้นภายในโรงเรือน
* set\_temp\_min คือค่าอุณหภูมิต่ำสุดที่ใช้ในการควบคุมอุณหภูมิ
* set\_humi\_max คือค่าอุณหภูมิสูงสุดที่ใช้ในการควบคุมความชื้น
* lux คือค่าความสว่างที่วัดได้จากเซ็นเซอร์ BH1750FVI
* set\_lux คือค่าความสว่างที่กำหนด

จากรายละเอียดด้านบนเป็นการส่งข้อมูลการเปลี่ยนแปลงไปยัง STA Node ในกรณีที่ข้อมูลมีการเปลี่ยนแปลง แต่ถ้าหากข้อมูลไม่มีการเปลี่ยนแปลงก็จะทำการตรวจสอบเงื่อนไขข้อต่อไปคือการตรวจสอบว่าข้อมูลส่งมาจาก STA Node หรือไม่ ดังรูปที่ 3.35 ถ้าไม่มีข้อมูลที่ส่งมาก็จะกลับไปตรวจสอบการเปลี่ยนแปลงของข้อมูล

หากมีข้อมูลส่งมาจาก STA Node ก็จะทำการตรวจสอบข้อมูลโดยหา IP Address -ของตัวเอง (Mushroom Node IP Address) ว่าข้อมูลที่ส่งมาเป็นข้อมูลของตัวเองหรือไม่ ถ้าใช่ก็จะนำข้อมูลที่ได้มาประมวลผลต่อไปดังรูปที่ 3.38 เป็นโปรแกรมส่วนตรวจสอบข้อมูล

|  |
| --- |
| **รูปที่ 3.38** แสดงภาพโปรแกรมส่วนที่ตรวจสอบข้อมูลที่ส่งมายัง Mushroom Node |
| **รูปที่ 3.39** แสดงภาพ Flowchart การทำงานของ Mushroom Node (2) |
| **รูปที่ 3.40** แสดงภาพ Flowchart การทำงานของ Mushroom Node (3) |

จากรูปที่ 3.39 และ รูปที่ 3.40 เมื่อได้รับข้อมูลที่ส่งมาจาก STA Node แล้วขั้นตอนต่อไปคือการนำข้อมูลมาประมวลผลโดยตัวอย่างของข้อมูลที่ถูกส่งมาจาก STA Node มีดังนี้ X1M, X1Y20, X1T30, X1H80, X1J85, X1P, X1F, หรือ X1R เป็นต้น มีรายละเอียดการทำงานในแต่ละข้อมูลดังนี้

|  |
| --- |
| **รูปที่ 3.41** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (1) |

จากรูปที่ 3.41 ถ้าข้อมูลที่ได้รับจาก STA Node มี Keyword ตัวแรกเท่ากับ “M” หมายความว่าต้องการเปลี่ยนโหมดการทำงานของระบบควบคุมอุณหภูมิและความชื้นโดยการทำงานคือจะตรวจสอบว่าถ้าโหมดการทำงานเป็นโหมด AUTO (ctrlMode = false) จะเปลี่ยนเป็นโหมด MANUAL (ctrlMode = true) และถ้าโหมดการทำงานเป็นโหมด MANUAL (ctrlMode = true) ก็จะเปลี่ยนโหมดการทำงานเป็นโหมด AUTO (ctrlMode = false) กล่าวคือเป็นการทำงานแบบสลับ (toggle) โหมดการทำงาน เมื่อเปลี่ยนโหมดการทำงานสำเร็จแล้วก็จะส่งข้อมูลการเปลี่ยนแปลงกลับไปยัง STA Node โดยใช้ฟังก์ชัน sendUpdateData()

|  |
| --- |
| **รูปที่ 3.42** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (2) |

จากรูปที่ 3.42 ถ้าข้อมูลที่ได้รับจาก STA Node มี Keyword ตัวแรกเท่ากับ “Y” หมายความว่าต้องการเปลี่ยนแปลงค่าอุณหภูมิต่ำสุดที่ใช้ในการควบคุมอุณหูมิ เช่น ปัจจุบันค่าอุณหภูมิต่ำสุดคือ 20°C ถ้าได้รับ Keyword “Y” ส่งมาจาก STA Node แสดงว่าต้องการเปลี่ยนค่าอุณหภูมิต่ำสุด โดยข้อมูลที่ส่งมาอาจจะเป็น Y21 ค่าตัวเลข 2 หลักต่อจาก Keyword เป็นค่าอุณหภูมิต่ำสุดที่ต้องการเปลี่ยน เมื่อได้รับค่าที่ต้องการเปลี่ยนแล้วก็จะเปลี่ยนค่าแล้วบันที่ค่าที่เปลี่ยนลงใน EEPROM ตำแหน่งที่ 1 จากนั้นก็จะส่งข้อมูลการเปลี่ยนแปลงกลับไปยัง STA Node โดยใช้ฟังก์ชัน sendUpdateData()

|  |
| --- |
| **รูปที่ 3.43** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (3) |

จากรูปที่ 3.43 ถ้าข้อมูลที่ได้รับจาก STA Node มี Keyword ตัวแรกเท่ากับ “T” หมายความว่าต้องการเปลี่ยนแปลงค่าอุณหภูมิสูงสุดที่ใช้ในการควบคุมอุณหูมิ เช่น ปัจจุบันค่าอุณหภูมิสูงสุดคือ 32°C ถ้าได้รับ Keyword “T” ส่งมาจาก STA Node แสดงว่าต้องการเปลี่ยนค่าอุณหภูมิสูงสุด โดยข้อมูลที่ส่งมาอาจจะเป็น T30 ค่าตัวเลข 2 หลักต่อจาก Keyword เป็นค่าอุณหภูมิสูงสุดที่ต้องการเปลี่ยน เมื่อได้รับค่าที่ต้องการเปลี่ยนแล้วก็จะเปลี่ยนค่าแล้วบันที่ค่าที่เปลี่ยนลงใน EEPROM ตำแหน่งที่ 0 จากนั้นก็จะส่งข้อมูลการเปลี่ยนแปลงกลับไปยัง STA Node โดยใช้ฟังก์ชัน sendUpdateData()

|  |
| --- |
| **รูปที่ 3.44** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (4) |

จากรูปที่ 3.44 ถ้าข้อมูลที่ได้รับจาก STA Node มี Keyword ตัวแรกเท่ากับ “H” หมายความว่าต้องการเปลี่ยนแปลงค่าความชื้นต่ำสุดที่ใช้ในการควบคุมความชื้น เช่น ปัจจุบันค่าความชื้นต่ำสุดคือ 80% ถ้าได้รับ Keyword “H” ส่งมาจาก STA Node แสดงว่าต้องการเปลี่ยนค่าความชื้นต่ำสุด โดยข้อมูลที่ส่งมาอาจจะเป็น H79 ค่าตัวเลข 2 หลักต่อจาก Keyword เป็นค่าความชื้นต่ำสุดที่ต้องการเปลี่ยน เมื่อได้รับค่าที่ต้องการเปลี่ยนแล้วก็จะเปลี่ยนค่าแล้วบันที่ค่าที่เปลี่ยนลงใน EEPROM ตำแหน่งที่ 10 จากนั้นก็จะส่งข้อมูลการเปลี่ยนแปลงกลับไปยัง STA Node โดยใช้ฟังก์ชัน sendUpdateData()

|  |
| --- |
| **รูปที่ 3.45** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (5) |

จากรูปที่ 3.45 ถ้าข้อมูลที่ได้รับจาก STA Node มี Keyword ตัวแรกเท่ากับ “J” หมายความว่าต้องการเปลี่ยนแปลงค่าความชื้นสูงสุดที่ใช้ในการควบคุมความชื้น เช่น ปัจจุบันค่าความชื้นสูงสุดคือ 90% ถ้าได้รับ Keyword “H” ส่งมาจาก STA Node แสดงว่าต้องการเปลี่ยนค่าความชื้นสูงสุด โดยข้อมูลที่ส่งมาอาจจะเป็น H85 ค่าตัวเลข 2 หลักต่อจาก Keyword เป็นค่าความชื้นสูงสุดที่ต้องการเปลี่ยน เมื่อได้รับค่าที่ต้องการเปลี่ยนแล้วก็จะเปลี่ยนค่าแล้วบันที่ค่าที่เปลี่ยนลงใน EEPROM ตำแหน่งที่ 9 จากนั้นก็จะส่งข้อมูลการเปลี่ยนแปลงกลับไปยัง STA Node โดยใช้ฟังก์ชัน sendUpdateData()

|  |
| --- |
| **รูปที่ 3.46** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (6) |

จากรูปที่ 3.46 ถ้าข้อมูลที่ได้รับจาก STA Node มี Keyword ตัวแรกเท่ากับ “P” หมายความว่าต้องการ เปิด/ปิด การทำงานของปั๊ม (Pump) โดยมีรายละเอียดการทำงานดังต่อไปนี้ ขั้นแรกจะตรวจสอบว่าถ้าค่าควบคุมการทำงานของปั๊ม (pumpState) มีค่าเท่ากับ false (pumpState == false) หรือสั่งให้ปั๊มหยุดทำงาน ก็จะเปลี่ยนค่าควบคุมการทำงานของปั๊ม (pumpState) ให้มีค่าเท่ากับ ture (pumpState = true) เพื่อให้ปั๊มกลับมาทำงาน เช่นเดียวกันถ้าปัจจุบันค่าควบคุมการทำงานของปั๊ม (pumpState) มีค่าเท่ากับ ture ก็จะเปลี่ยนค่าควบคุมการทำงานของปั๊ม (pumpState) ให้มีค่าเท่ากับ false กล่าวคือเป็นการทำงานแบบสลับ (toggle) ค่าควบคุมการทำงานของปั๊ม (pumpState) เมื่อเปลี่ยนสำเร็จแล้วก็จะส่งข้อมูลการเปลี่ยนแปลงกลับไปยัง STA Node โดยใช้ฟังก์ชัน sendUpdateData()

|  |
| --- |
| **รูปที่ 3.47** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (7) |

จากรูปที่ 3.47 ถ้าข้อมูลที่ได้รับจาก STA Node มี Keyword ตัวแรกเท่ากับ “F” หมายความว่าต้องการ เปิด/ปิด การทำงานของพัดลม (Fan) โดยมีรายละเอียดการทำงานดังต่อไปนี้ ขั้นแรกจะตรวจสอบว่าถ้าค่าควบคุมการทำงานของพัดลม (fanState) มีค่าเท่ากับ false (fanState == false) หรือสั่งให้พัดลมหยุดทำงาน ก็จะเปลี่ยนค่าควบคุมการทำงานของพัดลม (fanState) ให้มีค่าเท่ากับ ture (fanState = true) เพื่อให้พัดลมกลับมาทำงาน เช่นเดียวกันถ้าปัจจุบันค่าควบคุมการทำงานของพัดลม (fanState) มีค่าเท่ากับ ture ก็จะเปลี่ยนค่าควบคุมการทำงานของพัดลม (fanState) ให้มีค่าเท่ากับ false กล่าวคือเป็นการทำงานแบบสลับ (toggle) ค่าควบคุมการทำงานของพัดลม (fanState) เมื่อเปลี่ยนสำเร็จแล้วก็จะส่งข้อมูลการเปลี่ยนแปลงกลับไปยัง STA Node โดยใช้ฟังก์ชัน sendUpdateData()

ในส่วนต่อมาถ้าข้อมูลที่ได้รับจาก STA Node มี Keyword ตัวแรกเท่ากับ “R” หมายความว่า STA Node ต้องการอัปเดตข้อมูลการเปลี่ยนแปลงโดยให้ Mushroom Node ส่งข้อมูลการเปลี่ยนแปลงไปให้

|  |
| --- |
| **รูปที่ 3.48** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (8) |

จากรูปที่ 3.48 ถ้าข้อมูลที่ได้รับจาก STA Node มี Keyword ตัวแรกเท่ากับ “L” หมายความว่าต้องการเปลี่ยนแปลงค่าความสว่างโดยมีรายละเอียดการทำงานดังต่อไปนี้ เมื่อเจอ Keyword “L” จะนำข้อมูลต่อท้ายจำนวน 3 หลักมาเปลี่ยนค่าความสว่างที่กำหนดหรือกำหนดระดับความเข้มของแสง โดยจะทำการตรวจสอบค่าความสว่างที่กำหนดหากมีค่าต่ำกว่า 100 ก็จะนำเลข 0 มีแทรกด้านหน้าของข้อมูลเพื่อไม่ให้ความยาวของข้อมูลเปลี่ยนแปลง จากนั้นทำการบันทึกค่าความสว่างที่ถูกส่งมาจาก STA Node ลงใน EEPROM และส่งข้อมูลการเปลี่ยนแปลงกับไปยัง STA Node

**3.6.2 การออกแบบการสื่อสาร Lora ที่ STA Node**

ในการออกแบบการสื่อสารผ่าน Lora ที่ STA Node ผู้จัดทำจะขออธิบายรายละเอียดต่างๆ โดยใช้ Flowchart ประกอบการอธิบาย

|  |
| --- |
| **รูปที่ 3.49** แสดงภาพ Flowchart การทำงานของ STA Node (1) |

จากรูปที่ 3.49 แสดงการทำงานของ STA Node เริ่มต้นจะส่ง Keyword “R” ไปยัง Mushroom Node เพื่อขออัปเดตข้อมูลก่อนเริ่มระบบ โดยข้อมูลที่ส่งไปมีรูปแบบดังนี้ X1R เมื่อ X1 คือ IP Address ของ Mushroom Node เพื่อให้ทราบว่าต้องการส่งข้อมูลไปยัง Mushroom Node และ Keyword “R” คือ Keyword ที่ส่งไปเพื่อให้ Mushroom Node ทราบว่าต้องการอัปเดตข้อมูล

ขั้นตอนต่อไปจะตรวจสอบว่ามีข้อมูลส่งมาจาก Serial Port หรือ Windows Application หรือไม่ถ้ามีก็จะส่งผ่านข้อมูลไปยัง Mushroom Node เลยโดยไม่มีการประมวลผลใดๆ ตัวอย่างข้อมูลที่ส่งมาจาก Serial Port มีดังต่อไปนี้

* M คือ Keyword ที่ส่งเมื่อกดปุ่มเปลี่ยนโหมด
* P คือ Keyword ที่ส่งเมื่อกดปุ่มเปิดหรือปิดปั๊ม
* F คือ Keyword ที่ส่งเมื่อกดปุ่มเปิดหรือปิดพัดลม
* Y คือ Keyword ที่ส่งเมื่อเปลี่ยนค่าอุณหภูมิต่ำสุดตามด้วยตัวเลขจำนวน 2 หลัก เช่น Y25 เมื่อ 25 คือค่าที่ต้องการเปลี่ยน
* T คือ Keyword ที่ส่งเมื่อเปลี่ยนค่าอุณหภูมิสูงสุดตามด้วยตัวเลขจำนวน 2 หลัก เช่น T32 เมื่อ 32 คือค่าที่ต้องการเปลี่ยน
* H คือ Keyword ที่ส่งเมื่อเปลี่ยนค่าความชื้นต่ำสุดตามด้วยตัวเลขจำนวน 2 หลัก เช่น H80 เมื่อ 80 คือค่าที่ต้องการเปลี่ยน
* J คือ Keyword ที่ส่งเมื่อเปลี่ยนค่าความชื้นสูงสุดตามด้วยตัวเลขจำนวน 2 หลัก เช่น J90 เมื่อ 90 คือค่าที่ต้องการเปลี่ยน
* L คือ Keyword ที่ส่งเมื่อต้องการเปลี่ยนค่าความสว่างหรือระดับความสว่างตามด้วยตัวเลขจำนวน 2 หลักหรือ 3 หลัก เช่น L90, L200

จากรายละเอียดด้านบนเป็นข้อมูลจาก Serial Port เมื่อได้รับข้อมูลจาก Serial Port แล้วก็จะส่งข้อมูลต่อไปยัง Mushroom Node ทันทีดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 3.50** แสดงภาพโปรแกรมส่วนที่รับข้อมูลจาก Serial Port |

จากรูปที่ 3.50 จะรับข้อมูลจาก Serial Port แล้วส่งข้อมูลไปยัง Mushroom Node ด้วยฟังก์ชัน loraSend() โดยข้อมูลที่ส่งจะมีรูปแบบ (Data Format) ดังนี้ X1M เมื่อ X1 คือ IP Address ของ Mushroom Node และ M คือ Keyword ที่ส่งไปเพื่อควบคุมการทำงานของระบบดังรายละเอียดด้านบน จากนั้นจะล้างข้อมูลที่ได้รับจาก Serial Port

เมื่อระบบทำงานผ่านขั้นตอนต่างๆ ด้านบนมาแล้วก็ขั้นตอนต่อไปจะทำการตรวจสอบเงื่อนไขข้อต่อไปคือการตรวจสอบว่าข้อมูลส่งมาจาก Mushroom Node หรือไม่ ดังรูปที่ 3.49 ถ้าไม่มีข้อมูลที่ส่งมาก็จะส่งข้อมูลเพื่อขออัปเดตข้อมูล (C3) ทุกๆ 60 วินาที ถ้ามีข้อมูลส่งมาจาก Mushroom Node ก็จะทำการหา IP Address -ของตัวเอง (STA Node IP Address) ว่าข้อมูลที่ส่งมาเป็นข้อมูลของตัวเองหรือไม่ ถ้าใช่ก็จะนำข้อมูลที่ได้มาประมวลผลต่อไป

|  |
| --- |
| **รูปที่ 3.51** แสดงภาพโปรแกรมส่วนที่ตรวจสอบข้อมูลที่ส่งมายัง STA Node |

|  |
| --- |
| **รูปที่ 3.52** แสดงภาพ Flowchart การทำงานของ STA Node (2) |

เมื่อได้รับข้อมูลที่ส่งมาจาก Mushroom Node แล้วก็จะนำข้อมูลที่ได้มาเก็บไว้ในตัวแปลต่างๆ โดยใช้ฟังก์ substring() เพื่อนำข้อมูลที่ได้ไปใช้งานบน Web Application และ Windows Application ตัวอย่างรูปแบบของข้อมูล (Data Format) ที่ส่งมาจาก Mushroom Node ดังรูปที่ 3.53

|  |
| --- |
| **รูปที่ 3.53** แสดงภาพรูปแบบของข้อมูล (Data Format) ที่ส่งมาจาก Mushroom Node |

จากรูปที่ 3.53 แสดงรูปแบบของข้อมูลที่ส่งมาจาก Mushroom Node เมื่อได้รับข้อมูลก็จะนำข้อมูลที่ได้มา substring() ดังรูปที่ 3.54

|  |  |
| --- | --- |
|  |  |
| **รูปที่ 3.54** แสดงภาพการใช้ฟังก์ชัน substring() กับข้อมูลที่ได้รับจาก Mushroom Node | |

จากรูปที่ 3.54 แสดงการใช้ฟังก์ชัน substring() เพื่อนำข้อมูลที่ได้รับมาเก็บไว้ในตัวแปลที่ถูกต้องโดยมีรายละเอียดดังต่อไปนี้

* ตัวแปล node เป็นตัวแปลที่ใช้เก็บข้อมูลว่าข้อมูลที่ได้รับถูกส่งมาจาก Node ไหน โดยการ substring(0, 2) จากตำแหน่งที่ 0 ถึงตำแหน่งที่ 2
* ตัวแปล temp เป็นค่าอุณหภูมิ โดยการ substring(3, 8) จากตำแหน่งที่ 3 ถึงตำแหน่งที่ 8 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Float
* ตัวแปล humi เป็นค่าความชื้น โดยการ substring(9, 14) จากตำแหน่งที่ 9 ถึงตำแหน่งที่ 14 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Float
* ตัวแปล set\_temp\_max เป็นค่าอุณหภูมิสูงสุด โดยการ substring(14, 16) จากตำแหน่งที่ 14 ถึงตำแหน่งที่ 16 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Int
* ตัวแปล set\_humi\_min เป็นค่าความชื้นต่ำสุด โดยการ substring(16, 18) จากตำแหน่งที่ 16 ถึงตำแหน่งที่ 18 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Int
* ตัวแปล ctrlMode เป็นโหมดการทำงานของระบบควบคุมอุณหภูมิและความชื้นภายในโรงเรือน โดยการ substring(19, 20) จากตำแหน่งที่ 19 ถึงตำแหน่งที่ 20 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Int
* ตัวแปล pumpState เป็นค่าสถานะการทำงานของปั๊มที่ได้จากวงจรตัวต้านทานตรวจสอบกระแสโดยการ substring(21, 22) จากตำแหน่งที่ 21 ถึงตำแหน่งที่ 22 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Int
* ตัวแปล fanState เป็นค่าสถานะการทำงานของพัดลมที่ได้จากวงจรตัวต้านทานตรวจสอบกระแสโดยการ substring(23, 24) จากตำแหน่งที่ 23 ถึงตำแหน่งที่ 24 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Int
* ตัวแปล set\_temp\_min เป็นค่าอุณหภูมิต่ำสุด โดยการ substring(24, 26) จากตำแหน่งที่ 24 ถึงตำแหน่งที่ 26 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Int
* ตัวแปล set\_humi\_max เป็นค่าความชื้นต่ำสุด โดยการ substring(26, 28) จากตำแหน่งที่ 26 ถึงตำแหน่งที่ 28 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Int
* ตัวแปล lux เป็นค่าความสว่างที่ได้จากเซ็นเซอร์ BH1750FVI โดยการ substring(29, 36) จากตำแหน่งที่ 29 ถึงตำแหน่งที่ 36 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Float
* ตัวแปล set\_lux เป็นค่าความสว่างที่กำหนดไว้ โดยการ substring(36, 39) จากตำแหน่งที่ 36 ถึงตำแหน่งที่ 39 แล้วเปลี่ยนประเภทของข้อมูลเป็นประเภท Int

เมื่อจัดการกลับข้อมูลที่ส่งมาเรียบร้อยแล้วขั้นตอนต่อไปคือการส่ง Keyword เพื่อขออัปเดตข้อมูลทุกๆ 60 วินาที ดังรูปที่ 3.55

|  |
| --- |
| **รูปที่ 3.55** แสดงภาพโปรแกรมที่ใช้ในการส่ง Keyword ทุก 60 วินาที |

จากรูปที่ 3.55 มีหลักการทำงานคือ STA Node จะส่ง Keyword “R” ทุกๆ 60 วินาทีโดยใช้ฟังก์ชัย millis() ในการนับเวลา ซึ่งตัวแปล node1 มีค่าเท่ากับ X1 คือ IP Address ของ Mushroom Node รูปแบบข้อมูล (Data Format) คือ X1R

|  |
| --- |
| **รูปที่ 3.56** แสดงภาพ Flowchart การทำงานของ STA Node (3) |

จากรุปที่ 3.55 เมื่อ Keyword เพื่อขออัปเดตข้อมูลทุกๆ 60 วินาทีแล้ว ขั้นตอนต่อไปคือการตรวจสอบว่ามีการควบคุมการทำงานของระบบจากเว็บแอปพลิเคชัน (Web Application) หรือไม่ โดยมีรายละเอียดการทำงานดังต่อไปนี้

* ถ้ามีการกดปุ่มเปลี่ยนโหมดการทำงานจากเว็บแอปพลิเคชัน (Web Application) จะส่ง Keyword “M” ไปยัง Mushroom Node เพื่อแจ้งว่าผู้ใช้ต้องการเปลี่ยนโหมดการทำงาน โดยมีรูปแบบของข้อมูลที่ส่งดังนี้ X1M
* ถ้ามีการกดปุ่ม เปิด/ปิด ปั๊มจากเว็บแอปพลิเคชัน (Web Application) จะส่ง Keyword “M” ไปยัง Mushroom Node เพื่อแจ้งว่าผู้ใช้ต้องการ เปิด/ปิด ปั๊ม โดยมีรูปแบบของข้อมูลที่ส่งดังนี้ X1P
* ถ้ามีการกดปุ่ม เปิด/ปิด พัดลมจากเว็บแอปพลิเคชัน (Web Application) จะส่ง Keyword “F” ไปยัง Mushroom Node เพื่อแจ้งว่าผู้ใช้ต้องการ เปิด/ปิด พัดลม โดยมีรูปแบบของข้อมูลที่ส่งดังนี้ X1F
* ถ้ามีการกดปุ่มขออัปเดตข้อมูลจากเว็บแอปพลิเคชัน (Web Application) จะส่ง Keyword “R” ไปยัง Mushroom Node เพื่อแจ้งว่าผู้ใช้ต้องการอัปเดตข้อมูลโดยมีรูปแบบของข้อมูลที่ส่งดังนี้ X1R

|  |
| --- |
| **รูปที่ 3.57** แสดงภาพ Flowchart การทำงานของ STA Node (4) |

จากรูปที่ 3.57 หากมีการกรอกข้อมูลจากเว็บแอปพลิเคชัน (Web Application) STA Node จะส่งข้อมูลต่อไปยัง Mushroom Node เพื่อเปลี่ยนแปลงข้อมูลตามที่ผู้ใช้กรอกในแต่ละ input โดยมีรายละเอียดดังนี้

* ถ้ามีการกรอกข้อมูลที่ input1 จากเว็บแอปพลิเคชัน (Web Application) จะส่ง Keyword “T” + “ข้อมูลที่กรอก” ไปยัง Mushroom Node เพื่อแจ้งว่าผู้ใช้ต้องการเปลี่ยนค่าอุณหภูมิสูงสุด โดยมีรูปแบบของข้อมูลที่ส่งดังนี้ X1T32
* ถ้ามีการกรอกข้อมูลที่ input2 จากเว็บแอปพลิเคชัน (Web Application) จะส่ง Keyword “H” + “ข้อมูลที่กรอก” ไปยัง Mushroom Node เพื่อแจ้งว่าผู้ใช้ต้องการเปลี่ยนค่าความชื้นต่ำสุด โดยมีรูปแบบของข้อมูลที่ส่งดังนี้ X1H80
* ถ้ามีการกรอกข้อมูลที่ input3 จากเว็บแอปพลิเคชัน (Web Application) จะส่ง Keyword “Y” + “ข้อมูลที่กรอก” ไปยัง Mushroom Node เพื่อแจ้งว่าผู้ใช้ต้องการเปลี่ยนค่าอุณหภูมิต่ำสุด โดยมีรูปแบบของข้อมูลที่ส่งดังนี้ X1Y25
* ถ้ามีการกรอกข้อมูลที่ input4 จากเว็บแอปพลิเคชัน (Web Application) จะส่ง Keyword “J” + “ข้อมูลที่กรอก” ไปยัง Mushroom Node เพื่อแจ้งว่าผู้ใช้ต้องการเปลี่ยนค่าความชื้นสูงสุด โดยมีรูปแบบของข้อมูลที่ส่งดังนี้ X1J90
* ถ้ามีการกรอกข้อมูลที่ input5 จากเว็บแอปพลิเคชัน (Web Application) จะส่ง Keyword “L” + “ข้อมูลที่กรอก” ไปยัง Mushroom Node เพื่อแจ้งว่าผู้ใช้ต้องการเปลี่ยนค่าความความสว่าง โดยมีรูปแบบของข้อมูลที่ส่งดังนี้ X1L90, X1L200

**3.7 การออกแบบและสร้างเว็บแอปพลิเคชัน**

ในโครงงานนี้ผู้จัดทำได้ใช้ภาษา HTML CSS Javascript และ PHP ในการสร้าง Web Application และใช้แอปพลิเคชัน ngrok ที่ติดไว้บน Raspberry Pi ทำ Port forwarding ของ Web Application เพื่อให้สามารถใช้งานเว็บแอปพลิเคชันได้จากทุกที่ที่สามารถใช้งานอินเทอร์เน็ตได้ Web Application ส่วนที่แสดงค่าสถานะต่างๆและควบคุมการทำงานของระบบจะถูกเก็บอยู่ที่ ESP32LoRa และ Web Application ส่วนที่เกี่ยวกับฐานข้อมูล เช่น บันทึกข้อมูลลงในฐานข้อมูล นำข้อมูลในฐานข้อมูลมาแสดงบน Web Application จะถูกเก็บอยู่ที่ Raspberry Pi 3 Model B เพราะใช้ Raspberry Pi เป็นฐานข้อมูลดังรูปที่ 3.58

|  |
| --- |
| **รูปที่ 3.58** แสดงภาพโครงสร้างการทำงานของ Web Application |

**3.7.1 การสื่อสารระหว่าง Server และ Client**

การสื่อสารระหว่าง Server และ Client เป็นการสื่อสารผ่านทาง Hypertext Transfer Protocol (HTTP) โดยมีรายละเอียดการทำงานดังรูปที่ 3.59

|  |
| --- |
| **รูปที่ 3.59** แสดงภาพ Flowchart การสื่อสารระหว่าง Server และ Client |

จากรูปที่ 3.59 แสดงภาพ Flowchart การสื่อสารระหว่าง Server และ Client โดยมีรายละเอียดการทำงานดังต่อไปนี้

เริ่มต้นการทำงาน Client จะส่ง HTTP GET Method เพื่อขออัปเดตข้อมูลทุกๆ 5 วินาที เพื่อให้หน้าเว็บมีการอัปเดตข้อมูลโดยใช้ฟังก์ setInterval() ในภาษา JavaScript จากนั้นใช้ Ajax (Asynchronous JavaScript And XML) ในการส่ง HTTP GET Method ไปยัง Server เมื่อ Server ตอบกลับมาจะนำข้อมูลที่ได้ไปแสดงบนหน้าเว็บ ดังภาพด้านล่าง

|  |
| --- |
| **รูปที่ 3.60** แสดงภาพตัวอย่างโปรแกรมส่วนที่ใช้ในการขออัปเดตข้อมูลทุกๆ 5 วินาที |

จากรูปที่ 3.60 เป็นตัวอย่างการขออัปเดตค่าอุณหภูมิจาก Server ทุกๆ 5 วินาทีเพื่อให้ค่าอุณหภูมิที่แสดงบนหน้าเว็บมีการอัปเดต นอกจากนี้ยังมีค่าต่างๆที่ขออัปเดตทุกๆ 5 วินาทีเหมือนค่าอุณหภูมิ เช่น อุณหภูมิต่ำสุด อุณหภูมิสูงสุด ความชื้น ความชื้นต่ำสุด ความชื้นสูงสุด ความสว่าง ความสว่างที่กำหนด สถานะการทำงานของปั๊มและพัดลม โหมดการทำงาน ดังรูปที่ 3.61

|  |
| --- |
| **รูปที่ 3.61** แสดงภาพข้อมูลต่างๆบนหน้าเว็บที่อัปเดตทุกๆ 5 วินาที |

จากรูปด้านบนแสดงค่าตัวเลขและสถานะต่างๆ Client ขออัปเดตไปยัง Server ทุกๆ 5 วินาที ขั้นตอนต่อไปจะทำตรวจสอบว่ามีการความคุมการทำงานของระบบหรือไม่โดยมีรายละเอียดดังต่อไปนี้

|  |
| --- |
| **รูปที่ 3.62** แสดงภาพฟังก์ที่ถูกเรียกใช้เมื่อกดปุ่มเปลี่ยนโหมดการทำงาน |

จากรูปที่ 3.62 ถ้า Client กดปุ่มเปลี่ยนโหมดการทำงาน (change mode) จะเรียกใช้ฟังก์ชัน getmode() ที่เขียนด้วยภาษา JavaScript โดยฟังก์ชันมีการทำงานดังนี้ เมื่อมีการเรียกใช้จะส่ง HTTP GET Method “/M” ไปยัง Server โดยใช้ Ajax จากนั้น Server จะส่ง Keyword “M” ผ่าน Lora ไปยัง Mushroom Node เพื่อขอเปลี่ยนโหมดการทำงาน เมื่อ Server ตอบกลับมายัง Client ว่าส่งข้อมูลสำเร็จจะแจ้งเตือนดังรูปที่ 3.63

|  |
| --- |
| **รูปที่ 3.63** แสดงภาพการแจ้งเตือนเมื่อกดปุ่มเปลี่ยนโหมดการทำงาน |

|  |
| --- |
| **รูปที่ 3.64** แสดงภาพฟังก์ที่ถูกเรียกใช้เมื่อกดปุ่มขออัปเดตข้อมูล |

จากรูปที่ 3.64 ถ้า Client กดปุ่มขออัปเดตข้อมูล (request update) จะเรียกใช้ฟังก์ชัน requestupdate() ที่เขียนด้วยภาษา JavaScript โดยฟังก์ชันมีการทำงานดังนี้ เมื่อมีการเรียกใช้จะส่ง HTTP GET Method “/requestupdate” ไปยัง Server โดยใช้ Ajax จากนั้น Server จะส่ง Keyword “R” ผ่าน Lora ไปยัง Mushroom Node เพื่อขออัปเดตข้อมูล

|  |
| --- |
| **รูปที่ 3.65** แสดงภาพฟังก์ที่ถูกเรียกใช้เมื่อกดปุ่ม เปิด/ปิด ปั๊ม |

จากรูปที่ 3.65 ถ้า Client กดปุ่มควบคุมการทำงานของปั๊ม (control PUMP) จะเรียกใช้ฟังก์ชัน toggleCheckbox\_pump(element) ที่เขียนด้วยภาษา JavaScript โดยฟังก์ชันมีการทำงานดังนี้ เมื่อมีการเรียกใช้จะส่ง HTTP GET Method “/P” ไปยัง Server โดยใช้ Ajax จากนั้น Server จะส่ง Keyword “P” ผ่าน Lora ไปยัง Mushroom Node เพื่อควบคุมการทำงานของปั๊ม

|  |
| --- |
| **รูปที่ 3.66** แสดงภาพฟังก์ที่ถูกเรียกใช้เมื่อกดปุ่ม เปิด/ปิด พัดลม |

จากรูปที่ 3.66 ถ้า Client กดปุ่มควบคุมการทำงานของพัดลม (control FAN) จะเรียกใช้ฟังก์ชัน toggleCheckbox\_fan(element) ที่เขียนด้วยภาษา JavaScript โดยฟังก์ชันมีการทำงานดังนี้ เมื่อมีการเรียกใช้จะส่ง HTTP GET Method “/F” ไปยัง Server โดยใช้ Ajax จากนั้น Server จะส่ง Keyword “F” ผ่าน Lora ไปยัง Mushroom Node เพื่อควบคุมการทำงานของพัดลม

**3.7.2 การส่ง url ของ Web Application และ Database Server ไปยัง LINE Application**

เพื่อความสะดวกในการเข้าถึงเว็บแอปพลิเคชัน (Web Application) เพราะเมื่อใช้โปรแกรม Ngrok ทำ Port forwarding แล้วจะได้ random url มา url จะทำการเปลี่ยนไปทุกครั้งเมื่อมีการปิดหรือเปิดใช้งาน Ngrok ตัวอย่าง url ที่ได้จากการ random คือ https://e9d707f7cc96.ap.ngrok.io/ ช่องทางที่ผู้ใช้สามารถดู url ที่ได้ 3 อยู่สามช่องทาง 1. ดูที่หน้าโปรแกรม Ngrok 2. ดูที่เว็บของ Ngrok 3. ดูที่ localhost ของเครื่องที่ติดตั้งโปรแกรม Ngrok โดยเข้าไปที่ url ดังนี้ http://localhost:4040/api/tunnels

จากทั้ง 3 วิธีที่กล่าวมาผู้จัดทำจึงได้นำวิธีที่ 3 มาประยุกต์ใช้งานเพื่อส่ง url ของเว็บแอปพลิเคชัน (Web Application) ไปยัง LINE Application เพื่อให้สะดวกต่อการเข้าถึงโดยมีขั้นตอนดังรูปที่ 3.67

|  |
| --- |
| **รูปที่ 3.67** แสดงภาพ Flowchart ขั้นตอนการส่ง url ไปยัง LINE Application |

จากรูป 3.67 แสดงภาพ Flowchart ขั้นตอนการส่ง url ไปยัง LINE Application โดยมีรายละเอียดดังต่อไปนี้ ขั้นแรกตรวจสอบการเชื่อม WiFi ถ้าเชื่อมต่อแล้วจะส่ง HTTP GET Method ไปยังไฟล์ public-url.php ที่อยู่ใน Database Server ตัวอย่างที่อยู่ไฟล์ public-url.php คือ http://1d7113101b75.ap.ngrok.io/public-url.php หรือผ่าน Local IP Address เพราะ ESP32LoRa และ Raspberry Pi 3 b เชื่อมต่อกับ Access Point เดียวกัน ตัวอย่าง Local IP Address http://192.168.43.181/public-url.php โดยไฟล์ public-url.php มีรายละเอีอดดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 3.68** แสดงภาพไฟล์ public-url.php |

จากรูปที่ 3.68 โปรแกรมที่เขียนด้วยภาษา PHP โดยมีหลักการทำงานคือเรียกไปที่ http://localhost:4040/api/tunnels จะได้ข้อมูลรายละเอียดต่างๆ เกี่ยวกับ Web Application เช่น public\_url คือตัวแปลที่เก็บค่า url ที่ได้จากโปรแกรม Ngrok ตัวแปล public\_url จะถูกเก็บอยู่ในรูปแบบ JSON ดังรูปที่ 3.69

|  |
| --- |
| **รูปที่ 3.69** แสดงภาพข้อมูล JSON ที่ได้จาก http://localhost:4040/api/tunnels |

จึงต้องทำการ decode ก่อนที่จะนำมาใช้งานโดยใช้ฟังก์ชัน json\_decode ข้อมูลที่ได้จาก http://localhost:4040/api/tunnels ก่อนจากนั้นนำค่าที่ได้มาเก็บไว้ในตัวแปลต่างๆ แล้วสร้าง array เพื่อใช้สำหรับเก็บตัวแปลต่างๆเพื่อให้ง่ายต่อกการแปลงข้อมูลกลับไปเป็น JSON อีกครั้งโดยใช้ฟังก์ json\_encode ผลลัพธ์ที่ได้จากไฟล์ public-url.php ดังรูปที่ 3.70

|  |
| --- |
| **รูปที่ 3.70** แสดงภาพผลลัพธ์ที่ได้จากไฟล์ public-url.php |

เมื่อ Web Application (ESP32LoRa) ใช้ HTTP GET Method ไปยังไฟล์ public-url.php ที่อยู่บน Database Server (Raspberry Pi) แล้วจะนำผลลัพธ์ที่ได้มา decode แล้วเก็บค่าที่ได้ไว้ในตัวแปล tunnelsNameArr[] publicURLArr[] และ ดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 3.71** แสดงภาพโปรแกรมส่วนที่จัดการจัด url ที่ได้จากโปรแกรม Ngrok |

เมื่อได้ url ของ Web Application และ Database Server แล้วจะส่ง url ที่ได้ไปยัง LINE Application โดยใช้ฟังก์ชัน NotifyLine()

|  |
| --- |
| **รูปที่ 3.72** แสดงภาพตัวอย่าง url ของ Web Server และ Database Server ที่ส่งไปยัง  LINE Application |

**3.7.3 การใช้โปรแกรม Ngrok ในการทำ Port forwarding**

โปรแกรม Ngrok เป็น Tool Open Source พัฒนาโดย GitHub ซึ่งอำนวยความสะดวกให้บุคคลอื่นสามารถเข้าใช้งาน Website หรือ Application ที่กำลังทำงานอยู่บนเครื่อง Localhost โดยบุคคลอื่นสามารถเข้าใช้งาน Website หรือ Application กำลังทำงานอยู่บนเครื่อง Localhost ผ่านทาง URL ของทาง Ngrok โดยที่ทาง Ngrok จะทำการสุ่มสร้าง URL ขึ้นมา และ URL ที่ได้มานั้น จะทำการเปลี่ยนไปทุกครั้งเมื่อมีการปิดหรือเปิดใช้งาน Ngrok

ในโครงงานนี้ผู้จัดทำได้ใช้โปรแกรม Ngrok ที่ติดตั้งบน Raspberry Pi ในการทำ Port forwarding โดยมีรายละเอียดดังต่อไปนี้

ขั้นตอนที่ 1 สมัครสมาชิกกับ Ngrok เพื่อนำ authtoken ที่ได้มากรอกใน ngrok configuration file ให้สารถใช้งานโปรแกรมได้ตลอดเวลา

|  |
| --- |
| **รูปที่ 3.73** แสดงภาพ Authtoken ที่ได้จากการสมัครสมาชิก |

ขั้นตอนที่ 2 แก้ไขไฟล์ ngrok configuration file

|  |
| --- |
| **รูปที่ 3.74** แสดงภาพ ไฟล์ ngrok configuration file |

จากรูปที่ 3.74 แก้ไขไฟล์ ngrok configuration file นำ Authtoken ที่ได้จากการสมัครสมาชิก มากรอกใน ngrok configuration file เลือกพื้นที่ของ Ngrok Server โดยกำหนด region: ap หรือ Asia pacific เพื่อการใช้งานอย่างมีประสิทธิภาพ กำหนด IP Address ภายใน Local Area Network ที่ต้องการทำ Port forwarding มีรายละเอียดดังนี้

* การทำ Port forwarding ของ Web Server (ESP32LoRa) ต้องกำหนด IP Address ของ ESP32LoRa ตามด้วย Port เช่น addr: 192.168.43.100:80
* การทำ Port forwarding ของ Database Server (Raspberry Pi) กำหนดเฉพาะ Port เช่น addr: 80 หมายความว่าต้องการทำ Port forwarding IP Address ของเครื่องที่ติดตั้งโปรแกรม Ngrok ซึ่งก็คือ Raspberry Pi

ขั้นตอนที่ 3 สร้างไฟล์ autostart.sh เพื่อใช้ในการ run command ./ngrok start –all ./ngrok start –all เป็นคำที่สั่งให้โปรแกรม Ngrok เริ่มการทำงานจาก ngrok configuration file

|  |
| --- |
| รูปที่ 3.75 แสดงภาพไฟล์ autostart.sh |

ขั้นตอนที่ 4 ใช้ Crontab ในการ run คำสั่งที่อยู่ในไฟล์ autostart.sh Crontab คือคำสั่งที่จะทำงานตามเวลาที่กำหนด

|  |
| --- |
| **รูปที่ 3.76** แสดงภาพการใช้ Crontab ในการ run คำสั่ง |

จากรูปที่ 3.76 กรอบสีแดงคือส่วนที่เพิ่มเข้าไปใน Crontab โดยมีรายละเอียดคือการสั่งให้ Raspberry Pi run คำสังที่อยู่ในไฟล์ autostart.sh เมื่อ reboot หรือ เมื่อ Raspberry Pi เริ่มทำงาน

จากทั้ง 4 ขั้นตอนที่กล่าวมาผลลัพธ์ที่ได้คือโปรแกรม Ngrok ที่บน Raspberry Pi จะทำ Port forwarding โดยอัตโนมัติเมื่อ Raspberry Pi เริ่มทำงาน เพื่อช่วยอำนวยความสะดวกโดยไม่ต้องสั่งให้โปรแกรม Ngork ทำงาน

|  |
| --- |
| **รูปที่ 3.77** แสดงภาพผลลัพธ์ที่ได้จากการทำ Port forwarding |

รูปที่ 3.77 คือผลลัพธ์ที่ได้จากการทำ Port forwarding โดย url แรกคือ Public url ของ Web Application และ url ที่สองคือ Public url ของ Database Server

**3.8 การออกแบบและสร้างวินโดว์แอปพลิเคชัน**

ในโครงงานนี้ผู้จัดทำได้มีวัตถุประสงค์ในการนำวินโดว์แอปพลิเคชัน (Windows Application) มาประยุกต์ใช้ภายในโครงงาน ใช้ภาษา Visual Basic ของ .NET Core ในการสร้างวินโดว์แอปพลิเคชัน (Windows Application)

วินโดว์แอปพลิเคชัน (Windows Application) สามารถใช้งานได้กับ ESP32LoRa ได้ทั้งสองตัว ดังรูปที่ 3.78 โดยสื่อสารกันผ่าน Serial Port



**รูปที่ 3.78** แสดงภาพโครงสร้างของระบบ

จากรูปที่ 3.78 จะเห็นว่าวินโดว์แอปพลิเคชัน (Windows Application) สามารถใช้งานได้กับ ESP32LoRa (Mushroom Node) และ ESP32LoRa (STA Node) โดยใช้งานผ่านทาง Serial Port

หากใช้งานกับวินโดว์แอปพลิเคชัน (Windows Application) กับ ESP32LoRa (Mushroom Node) เมื่อควบคุมการทำงาน เช่น กดปุ่มเปลี่ยนโหมดการทำงาน วินโดว์แอปพลิเคชัน (Windows Application) จะส่งข้อมูลผ่าน Serial Port มายัง ESP32LoRa จากนั้น ESP32LoRa จะนำข้อมูลที่ได้มาประมวลผลตามข้อมูลที่ได้

หากใช้งานกับวินโดว์แอปพลิเคชัน (Windows Application) กับ ESP32LoRa (STA Node) เมื่อควบคุมการทำงาน เช่น กดปุ่มเปลี่ยนโหมดการทำงาน วินโดว์แอปพลิเคชัน (Windows Application) จะส่งข้อมูลผ่าน Serial Port มายัง ESP32LoRa จากนั้น ESP32LoRa จะส่งข้อมูลที่ได้จาก Serial Port ผ่าน LoRa ไปยังยัง ESP32LoRa (Mushroom Node) จากนั้นจะนำข้อมูลที่ได้มาประมวลผล

**3.8.1 การทำงานของวินโดว์แอปพลิเคชัน (Windows Application)**

ขั้นตอนการทำงานของวินโดว์แอปพลิเคชัน (Windows Application) มีรายละเอียดการทำงานดดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 3.79** แสดงภาพ Flowchart การทำงานของวินโดว์แอปพลิเคชัน (1) |
| **รูปที่ 3.80** แสดงภาพ Flowchart การทำงานของวินโดว์แอปพลิเคชัน (2) |

จากรูปที่ 3.79 และ 3.80 แสดงภาพ Flowchart การทำงานของวินโดว์แอปพลิเคชันโดยมีรายละเอียดดังต่อไปนี้ เมื่อแอปพลิเคชันเริ่มต้นทำงานจะให้กรอก Username และ Password ก่อนเพื่อป้องกันความปลอดภัย ดังรูปที่ 3.81

|  |
| --- |
| **รูปที่ 3.81** แสดงภาพหน้า Login ของวินโดว์แอปพลิเคชัน |

เมื่อทำการ Login สำเร็จจะแสดงหน้าแรกของวินโดว์แอปพลิเคชันดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 3.82** แสดงภาพวินโดว์แอปพลิเคชันเมื่อยังไม่ได้เชื่อมต่อกับ Serial Port |

จากรูปที่ 3.82 แสดงภาพของวินโดว์แอปพลิเคชันเมื่อยังไม่เชื่อต่อกับ Serial Port เมื่อทำการเชื่อมต่อแล้วจะแสดงค่าสถานะต่างๆ ของโรงเรือนดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 3.83** แสดงภาพวินโดว์แอปพลิเคชันเมื่อเชื่อมต่อกับ Serial Port แล้ว |

จากรูปที่ 3.83 แสดงภาพวินโดว์แอปพลิเคชันเมื่อเชื่อมต่อกับ Serial Port แล้ว ESP32LoRa จะส่งข้อมูลผ่าน Serial Port มายังวินโดว์แอปพลิเคชันโดยส่งฟังก์ชัน Serial.print() ตัวอย่างข้อมูลคือ T30.50H78.703075M0P1F02685L200.33200 จากนั้นวินโดว์แอปพลิเคชันจะนำข้อมูลที่ได้ประมวลผลดังรูปที่ 3.84 แล้วนำมาแสดงบนหน้าวินโดว์แอปพลิเคชัน

|  |  |
| --- | --- |
|  |  |
| **รูปที่ 3.84** แสดงภาพส่วนของโปรแกรม Visual Basic ที่จัดการกับข้อมูลที่ส่งมาจาก Serial Port | |

ขั้นตอนต่อมาหลังจากที่จัดการกับข้อมูลที่ส่งมาจาก Serial Port แล้วจะตรวจสอบว่ามีการควบคุมการทำงานของระบบหรือไม่โดยมีรายละเอียดดังต่อไปนี้

เมื่อกดปุ่มเปลี่ยนโหมดการทำงาน (Change mode) วินโดว์แอปพลิเคชันจะส่ง Keyword “M” ผ่าน Serial Port ไปยัง ESP32LoRa โดยใช้ฟังก์ชัน SerialPort1.Write("M") เพื่อให้ ESP32LoRa นำ Keyword ที่ได้ไปประมวลผลต่อไปดังรูปที่ 3.85

|  |
| --- |
| **รูปที่ 3.85** แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกดปุ่มเปลี่ยนโหมด |

เมื่อกดปุ่มเปิดปั๊ม (Turn on Pump) หรือ ปุ่มปิดปั๊ม (Turn off Pump) วินโดว์แอปพลิเคชันจะส่ง Keyword “P” ผ่าน Serial Port ไปยัง ESP32LoRa โดยใช้ฟังก์ชัน SerialPort1.Write("P") เพื่อให้ ESP32LoRa นำ Keyword ที่ได้ไปประมวลผลต่อไปดังรูปที่ 3.86

|  |
| --- |
| **รูปที่ 3.86** แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกดปุ่ม เปิด/ปิด ปั๊ม |

เมื่อกดปุ่มเปิดพัดลม (Turn on Fan) หรือ ปุ่มปิดพัดลม (Turn off Fan) วินโดว์แอปพลิเคชันจะส่ง Keyword “F” ผ่าน Serial Port ไปยัง ESP32LoRa โดยใช้ฟังก์ชัน SerialPort1.Write("F") เพื่อให้ ESP32LoRa นำ Keyword ที่ได้ไปประมวลผลต่อไปดังรูปที่ 3.87

|  |
| --- |
| **รูปที่ 3.87** แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกดปุ่ม เปิด/ปิด พัดลม |

เมื่อกำหนดค่าอุณหภูมิต่ำสุดวินโดว์แอปพลิเคชันจะส่ง Keyword “Y” + ข้อมูลที่กรอก ผ่าน Serial Port ไปยัง ESP32LoRa โดยใช้ฟังก์ชัน SerialPort1.Write("Y" + TextBox1.Text) เพื่อให้ ESP32LoRa นำ Keyword ที่ได้ไปประมวลผลต่อไปดังรูปที่ 3.88

|  |
| --- |
| **รูปที่ 3.88** แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าอุณหภูมิ  ต่ำสุด |

เมื่อกำหนดค่าอุณหภูมิสูงสุดวินโดว์แอปพลิเคชันจะส่ง Keyword “T” + ข้อมูลที่กรอก ผ่าน Serial Port ไปยัง ESP32LoRa โดยใช้ฟังก์ชัน SerialPort1.Write("T" + TextBox4.Text) เพื่อให้ ESP32LoRa นำ Keyword ที่ได้ไปประมวลผลต่อไปดังรูปที่ 3.89

|  |
| --- |
| **รูปที่ 3.89** แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าอุณหภูมิ  สูงสุด |

เมื่อกำหนดค่าความชื้นต่ำสุดวินโดว์แอปพลิเคชันจะส่ง Keyword “H” + ข้อมูลที่กรอก ผ่าน Serial Port ไปยัง ESP32LoRa โดยใช้ฟังก์ชัน SerialPort1.Write("H" + TextBox2.Text) เพื่อให้ ESP32LoRa นำ Keyword ที่ได้ไปประมวลผลต่อไปดังรูปที่ 3.90

|  |
| --- |
| **รูปที่ 3.90** แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าความชื้น  ต่ำสุด |

เมื่อกำหนดค่าความชื้นสูงสุดวินโดว์แอปพลิเคชันจะส่ง Keyword “J” + ข้อมูลที่กรอก ผ่าน Serial Port ไปยัง ESP32LoRa โดยใช้ฟังก์ชัน SerialPort1.Write("J" + TextBox3.Text) เพื่อให้ ESP32LoRa นำ Keyword ที่ได้ไปประมวลผลต่อไปดังรูปที่ 3.91

|  |
| --- |
| **รูปที่ 3.91** แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าความชื้น  สูงสุด |

เมื่อกำหนดค่าความสว่างวินโดว์แอปพลิเคชันจะส่ง Keyword “L” + ข้อมูลที่กรอก ผ่าน Serial Port ไปยัง ESP32LoRa โดยใช้ฟังก์ชัน SerialPort1.Write("L" + TextBoxSetLux.Text) เพื่อให้ ESP32LoRa นำ Keyword ที่ได้ไปประมวลผลต่อไปดังรูปที่ 3.92

|  |
| --- |
| **รูปที่ 3.92** แสดงภาพการส่งข้อมูลผ่าน Serial Port ไปยัง ESP32LoRa เมื่อกำหนดค่าความสว่าง |

**3.9 การออกแบบฐานข้อมูลและการบันทึกข้อมูล**

**3.9.1 การออกแบบฐานข้อมูล**

ในการออกแบบฐานข้อมูลในโครงงานนี้จะใช้แค่ 1 ตาราง (entity) เท่านั้นเพื่อใช้เก็บข้อมูลต่างๆที่เกี่ยวข้องเช่น ค่าอุณหภูมิ (temp) ค่าความชื้น (humi) ค่าความสว่าง (lux) ค่าอุณหภูมิต่ำสุด (temp\_limit\_min) ค่าอุณหภูมิสูงสุด (temp\_limit\_max) ค่าความชื้นต่ำสุด (humi\_limit\_min) ค่าความชื้นสูงสุด (humi\_imit\_max) ค่าความสว่างที่กำหนด (set\_lux) โหมดการทำงาน (ctrl\_mode) สถานะการทำงานของปั๊ม (pump\_state) สถานการณ์ทำงานของพัดลม (fan\_state) และเวลาที่บันทึกข้อมูล (save\_time) ดังรูปที่ 3.93 โดยตารางมีชื่อว่า SensorData

|  |
| --- |
| **รูปที่ 3.93** แสดงภาพโครงสร้างของฐานข้อมูล |

จากรูปที่ 3.93 โครงสร้างของฐานข้อมูลโดยมี Primary key (PK) คือ id สามารถบันทึกข้อมูลเป็นตัวเลขจำนวนเต็มแบบไม่มีเครื่องหมายไม่เกิน 6 หลัก เป็นลำดับของข้อมูลแบบเพิ่มขึ้นอัตโนมัติ (auto increment) มี save\_time attribute ใช้ในการเก็นค่าวันเวลาที่บันทึกข้อมูล และ #temp #humi #lux #temp\_limit\_min #temp\_limit\_max #humi\_limit\_min #humi\_limit\_max #set\_lux #ctrl\_mode #pump\_state #fan\_state attribute เป็น attribute ที่ใช้ในการเก็บข้อมูลต่างๆภายในระบบ

**ตารางที่ 3.9** แสดง Data Dictionary SensorData

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribute | Description | Type | Example | Key |
| id | ลำดับของข้อมูล | int(6) | 1 | PK |
| temp | อุณหภูมิ | varchar(5) | 32.10 |  |
| humi | ความชื้น | varchar(5) | 80.70 |  |
| lux | ความสว่าง | varchar(6) | 213.22 |  |
| temp\_limit\_min | อุณหภูมิต่ำสุด | varchar(2) | 20 |  |
| temp\_limit\_max | อุณหภูมิสูงสุด | varchar(2) | 29 |  |
| humi\_limit\_min | ความชื้นต่ำสุด | varchar(2) | 80 |  |
| humi\_limit\_max | ความชื้นสูงสุด | varchar(2) | 85 |  |
| Set\_lux | ความสว่างที่กำหนด | varchar(5) | 200 |  |
| ctrl\_mode | โหมดการทำงาน | varchar(6) | AUTO |  |
| pump\_state | สถานะการทำงานของปั๊ม | varchar(3) | ON |  |
| fan\_state | สถานะการทำงานของพัดลม | varchar(3) | ON |  |
| save\_time | วันเวลาที่บันทึกข้อมูล | timestamp | 2020-09-08 18:00:00 |  |

|  |
| --- |
| **รูปที่ 3.94** แสดงภาพตัวอย่างข้อมูลเก็บอยู่ภายในฐานข้อมูล |

**3.9.2 การบันทึกข้อมูลลงในฐานข้อมูล**

ในการจัดทำโครงงานนี้ผู้จัดทำได้ใช้ Raspberry pi 3 b เป็นฐานข้อมูลเพื่อเก็บค่าต่างๆและใช้ไมโครคอนโทรลเลอร์ ESP32 LoRa เป็นเว็บเซิร์ฟเวอร์ (Web Server) โดยจะบันทึกข้อมูลลงในฐานข้อมูลทุกๆชั่วโมง เช่น 00.00, 01.00, 02.00, 12.00, 13.00 เป็นต้น ดัง Flowchart ด้านล่าง

|  |
| --- |
| **รูปที่ 3.95** แสดงภาพ Flowchart ขั้นตอนการบันทึกข้อมูลลงในฐานข้อมูล |
| **รูปที่ 3.96** แสดงภาพการบันทึกข้อมูลลงในฐานข้อมูลทุกชั่วโมง |

จากรูปที่ 3.96 จะเป็นการดึงค่าเวลาจริงมาจากเซิร์ฟเวอร์ NTP (Network Time Protocol) มาใช้ในการตรวจสอบเพื่อบันทึกข้อมูล โดยจะตรวจสอบเวลาในหลักนาทีและวินาที เช่น ถ้าเวลาในหน่วยนาทีมีค่าเท่ากับ 00 และเวลาในหน่วยวินาทีมีค่าเท่ากับ 00 ก็จะทำการบันทึกข้อมูลโดยเรียกใช้ฟังก์ชัน insertDB() ก็จะสามารถบันทึกข้อมูลทุกๆชั่วโมงได้แล้ว

การบันทึกข้อมูลจากเว็บเซิร์ฟเวอร์ (Web Server) ไปยังฐานข้อมูลที่อยู่บน Raspberry pi 3 b ทุกๆชั่วโมงโดยใช้ฟังก์ชัน insertDB() ที่เขียนด้วยภาษา C++ ดังภาพที่ 3.97



**รูปที่ 3.97** แสดงภาพฟังก์ชันที่ใช้บันทึกข้อมูลลงในฐานข้อมูล

จากรูปที่ 3.97 เป็นฟังก์ชันที่ทำหน้าบันทึกข้อมูลลงในฐานข้อมูลที่อยู่บน Raspberry pi 3 b โดยตัวแปลที่เป็นประเภท String ที่มีชื่อว่า serverName เป็นตัวแปลที่เก็บที่อยู่ของไฟล์ PHP ที่มีชื่อว่า post-esp-data.php ทำหน้าที่บันทึกข้อมูลลงในฐานข้อมูลที่อยู่บน Raspberry pi โดยที่อยู่นี้ได้จากการทำ port forwarding โดยใช้โปรแกรม Ngrok ตัวอย่างที่อยู่ http://bc14e80e8393.ap.ngrok.io/post-esp-data.php

ขั้นตอนต่อมาเป็นการจัดรูปแบบเพื่อเตรียมการก่อนการบันทึกข้อมูลผ่าน HTTP POST REQUEST ตัวแปลที่เป็นประเภท String ที่มีชื่อว่า httpRequestData เป็นตัวแปลที่เก็บข้อมูลต่างๆ ที่จะบันทึกลงในฐานข้อมูล ตัวแปลที่มีชื่อว่า httpResponseCode จะทำหน้าที่ POST ข้อมูลไปยังไฟล์ post-esp-data.php แล้วเก็บค่า http response เช่นเซิร์ฟเวอร์ตอบกลับว่า 200 OK แสดงว่าบันทึกข้อมูลสำเร็จ จากนั้นทำการแจ้งเตือนสถานะการบันทึกข้อมูลทาง LINE แอปพลิเคชัน โดยเรียกใช้ฟังก์ชัน NotifyLine()

|  |
| --- |
| **รุปที่ 3.98** แสดงภาพไฟล์ post-esp-data.php บางส่วน |

จากรูปที่ 3.98 แสดง code บางส่วนของไฟล์ post-esp-data.php โดยมีตัวแปลที่มีชื่อว่า $api\_key\_value ที่มีค่าเท่ากับ “tPmAT5Ab3j7F9” เป็นค่าที่ใช้ตรวจสอบว่าข้อมูลที่ส่งมา มาจากแหล่งที่มีที่ถูกต้อง เช่น ถ้า ESP32 LoRa ต้องการที่จะบันทึกข้อมูลลงในฐานข้อมูลจะต้องส่งค่า $api\_key\_value มาด้วยผ่าน HTTP POST

ต่อมาไฟล์ post-esp-data.php จะตรวจสอบว่าถ้ามีการ POST มาจะตรวจสอบ $api\_key\_value ถ้าตรงกันจะทำการเชื่อมต่อฐานข้อมูลแล้วบันทึกข้อมูล ถ้าไม่ตรงกันจะแสดงข้อความ "Wrong API Key provided." เพื่อเป็นการป้องกันฐานข้อมูลเบื่องต้น

**3.10 การออกแบบ Use Case Diagram**

|  |
| --- |
| **รูปที่ 3.99** แสดงภาพ Use Case Diagram |

รายละเอียด Use Case Diagram จากตารางที่ 3.10 ถึงตารางที่ 3.16 ด้านล่างต่อไปนี้แสดงถึงรายละเอียดการใช้งานของแต่ละ Use Case ตามรูปแบบของ UML Use Case Description

**ตารางที่ 3.10** แสดงLogin Use Case

|  |  |
| --- | --- |
| Use Case Name | Login |
| Actors | User |
| Use Case Purpose | เพื่อเข้าสู่ระบบ |
| Pre-conditions | - |
| Post-conditions | เข้าสู่ระบบเพื่อดูค่าสถานะต่างๆ ควบคุมการทำงานของอุปกรณ์ กำหนดค่าสูงสุดต่ำสุดของอุณหภูมิและความชื้น กำหนดค่าความสว่าง |
| Main Course | 1.กรอก Username และ Password  2.กดปุ่ม Login  3.เข้าสู่ระบบเสร็จสิ้น |
| Exceptions | Username หรือ Password ไม่ถูกต้อง |

**ตารางที่ 3.11** แสดงLogout Use Case

|  |  |
| --- | --- |
| Use Case Name | Logout |
| Actors | User |
| Use Case Purpose | เพื่อใช้ในการออกจากระบบ |
| Pre-conditions | เข้าสู่ระบบมาแล้วก่อนหน้า |
| Post-conditions | ออกจากระบบสำเร็จ |
| Main Course | 1.กดปุ่ม Logout  2.ออกจากระบบสำเร็จ |
| Exceptions | - |

**ตารางที่ 3.12** แสดง View Dashboard Use Case

|  |  |
| --- | --- |
| Use Case Name | View Dashboard |
| Actors | User |
| Use Case Purpose | เพื่อใช้ในการแสดงสถานะต่างๆ และจัดการระบบ |
| Pre-conditions | เข้าสู่ระบบมาแล้วก่อนหน้า |
| Post-conditions | - |
| Main Course | แสดงค่าสถานะต่างๆ เช่น อุณหภูมิ ความชื้น แสงสว่าง สถานะการทำงานของอุปกรณ์ |
| Exceptions | แสดงค่าผิดพลาด |

**ตารางที่ 3.13** แสดง Control Use Case

|  |  |
| --- | --- |
| Use Case Name | Control |
| Actors | User |
| Use Case Purpose | เพื่อใช้ในการควบคุมการทำงานของระบบ เช่น เปลี่ยนโหมดการทำงาน ความคุมการเปิดปิดของอุปกรณ์ |
| Pre-conditions | ใช้ร่วมกับ View Dashboard |
| Post-conditions | - |
| Main Course | 1.กดปุ่ม change mode  2.ถ้าโหมดการทำงานแบบอัตโนมัติระบบจะควบคุมการทำงานของอุปกรณ์โดยอัตโนมัติ เมื่อค่าอุณหภูมิและความชื้นต่ำกว่าหรือสูงกว่าที่กำหนด ถ้าโหมดการทำงานแบบแมนนวลผู้ใช้จะสามารถสั่งเปิดปิดอุปกรณ์ได้ตามต้องการ |
| Exceptions | เกิดความผิดพลาดภายในระบบ |

**ตารางที่ 3.14** แสดง Setting Use Case

|  |  |
| --- | --- |
| Use Case Name | Setting |
| Actors | User |
| Use Case Purpose | เพื่อใช้ในการกำหนดค่าสูงสุดต่ำสุดของอุณหภูมิและความชื้น กำหนดค่าความสว่าง ตั้งค่าการแจ้งเตือนผ่าน LINE Application |
| Pre-conditions | ใช้ร่วมกับ View Dashboard |
| Post-conditions | - |
| Main Course | 1.กำหนดค่าสูงสุดต่ำสุด  2.กำหนดค่าสูงสุดต่ำสุดสำเร็จ |
| Exceptions | เกิดความผิดพลาดภายในระบบ |

**ตารางที่ 3.15** แสดง View Data Use Case

|  |  |
| --- | --- |
| Use Case Name | View Data |
| Actors | User |
| Use Case Purpose | เพื่อใช้ในการแสดงผลข้อมูลที่อยู่ใน database ในรูปแบบต่างๆ |
| Pre-conditions | ใช้ร่วมกับ View Dashboard |
| Post-conditions | - |
| Main Course | 1.กดปุ่ม datalogger หรือ chart  2.แสดงข้อมูลในรูปแบบต่างๆ |
| Exceptions | ไม่สามารถติดต่อฐานข้อมูลได้ |

**ตารางที่ 3.16** แสดง View Status Use Case

|  |  |
| --- | --- |
| Use Case Name | View Status |
| Actors | User, Guest |
| Use Case Purpose | เพื่อใช้ในการแสดงสถานะบนจอ LCD Display ที่โรงเพาะเห็ด |
| Pre-conditions | - |
| Post-conditions | - |
| Main Course | 1.ดูค่าสถานะบนจอ LCD Display |
| Exceptions | เกิดความผิดพลาดภายในระบบ |

**บทที่ 4**

**ผลการดำเนินงาน**

หลังจากที่ได้ดำเนินงานออกแบบโรงเรือนเพาะเห็ด เขียนโปรแกรมควบคุมการทำงานระบบภายในโรงเรือน ออกแบบและสร้างเว็บแอปพลิเคชันและวินโดว์แอปพลิเคชันเพื่อมอนิเตอร์และความคุมการทำงานของโรงเรือนเพาะเห็ด ในบทนี้เป็นการทดลองส่วนต่างๆที่ได้ดำเนินงานในบทที่ 3 และนำระบบทั้งหมดมาใช้งานร่วมกัน โดยได้ทำการทดลองดังนี้

1. การทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือนและปุ่มควบคุมที่กล่อง STA
2. การทดลองควบคุมระบบภายในโรงเรือนจากวินโดว์แอปพลิเคชัน (Windows Application)
3. การทดลองควบคุมระบบภายในโรงเรือนจากเว็บแอปพลิเคชัน (Web Application)
4. การทดลองเพาะเห็ด

**4.1 การทดลอง****ควบคุมระบบภายในโรงเรือนจาก****ปุ่มควบคุมหน้าโรงเรือนและปุ่มควบคุมที่กล่อง STA**

**4.1.1 วัตถุประสงค์**

เพื่อทดลองควบคุมระบบภายในโรงเรือนเพาะเห็ดจากปุ่มควบคุมหน้าโรงเรือนและปุ่มควบคุมที่กล่อง STA

**4.1.2 วิธีการทดลอง**

ควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือนและปุ่มควบคุมที่กล่อง STA โดยปุ่มทั้ง 2 ที่ สามารถควบคุมโหมดการทำงานของระบบ เปิดปิดพัดลมและปั๊ม

1.) ควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือน

|  |
| --- |
| **รูปที่ 4.1** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือน (1) |

จากรูปที่ 4.1 จะเห็นว่าโหมดการทำงานของระบบเป็นโหมดการทำงานแบบ AUTO จากนั้นทำการทดลองเปลี่ยนโหมดการทำงานของระบบโดยกดที่ปุ่มสีเขียว (Mode) ระบบจะเปลี่ยนโหมดการทำงานเป็นโหมด MANUAL ดังรูปที่ 4.2

|  |
| --- |
| **รูปที่ 4.2** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือน (2) |

จากรูปที่ 4.2 จะเปลี่ยนโหมดการทำงานเป็นโหมด MANUAL จากนั้นทำการทดลองควบคุมการทำงานของปั๊มโดยการสั่งปิดปั๊มกดที่ปุ่มสีเหลืองด้านขาว (Pump) ดังรูปที่ 4.ป3

|  |
| --- |
| **รูปที่ 4.3** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือน (3) |

|  |
| --- |
| **รูปที่ 4.4** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือน (4) |

จากรูปที่ 4.4 จะเห็นว่าสถานการณ์ทำงานของปั๊มมีค่าเท่ากับ 0 หมายความว่าปั๊มหยุดทำงาน จากนั้นทำการควบคุมการทำงานของพัดลม โดยการสั่งเปิดพัดลมทำการกดที่ปุ่มสีเหลืองทางด้านซ้าย (Fan) ดังรูปที่ 4.5

|  |
| --- |
| **รูปที่ 4.5** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือน (5) |

|  |
| --- |
| **รูปที่ 4.6** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือน (6) |

จากรูปที่ 4.6 จะเห็นว่าสถานการณ์ทำงานของพัดลมมีค่าเท่ากับ 1 หมายความว่าพัดลมกำลังทำงาน

2.) ควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมที่กล่อง STA

|  |  |
| --- | --- |
|  |  |
| **รูปที่ 4.7** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมที่กล่อง STA (1) | |

จากรูปที่ 4.7 จะเห็นว่าโหมดการทำงานของระบบเป็นโหมดการทำงานแบบ AUTO จากนั้นทำการทดลองเปลี่ยนโหมดการทำงานของระบบโดยกดที่ปุ่ม Mode ระบบจะเปลี่ยนโหมดการทำงานเป็นโหมด MANUAL ดังรูปที่ 4.8

|  |
| --- |
| **รูปที่ 4.8** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมที่กล่อง STA (2) |

จากรูปที่ 4.8 จะเปลี่ยนโหมดการทำงานเป็นโหมด MANUAL จากนั้นทำการทดลองควบคุมการทำงานของปั๊มโดยการสั่งปิดปั๊มกดที่ปุ่ม Pump ดังรูปที่ 4.9

|  |  |
| --- | --- |
|  |  |
| **รูปที่ 4.9** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมที่กล่อง STA (3) | |

|  |
| --- |
| **รูปที่ 4.10** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมที่กล่อง STA (4) |

จากรูปที่ 4.10 จะเห็นว่าสถานการณ์ทำงานของปั๊มมีค่าเท่ากับ 1 หมายความว่าปั๊มกำลังทำงาน จากนั้นทำการควบคุมการทำงานของพัดลม โดยการสั่งปิดพัดลมทำการกดที่ปุ่ม Fan ดังรูปที่ 4.11

|  |  |
| --- | --- |
|  |  |
| **รูปที่ 4.11** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมที่กล่อง STA (5) | |
| **รูปที่ 4.12** แสดงภาพการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมที่กล่อง STA (6) | |

จากรูปที่ 4.12 จะเห็นว่าสถานการณ์ทำงานของพัดลมมีค่าเท่ากับ 0 หมายความว่าพัดลมหยุดทำงาน

**4.1.3 ผลการทดลอง**

จากการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือนและปุ่มควบคุมที่กล่อง STA ได้ผลการทดลองคือ สามารถควบคุมการทำงานของระบบภายในโรงเรือนเพาะเห็ดได้

**4.2 การทดลองควบคุมระบบภายในโรงเรือนจากวินโดว์แอปพลิเคชัน (Windows Application)**

**4.2.1 วัตถุประสงค์**

เพื่อทดลองใช้งาน Windows Application ในการควบคุมระบบภายในโรงเรือนเพาะเห็ด เช่น การเปลี่ยนโหมดการทำงาน เปิด/ปิดปั๊มพ่นหมอกและพัดลมระบายอากาศ กำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด กำหนดค่าความชื้นต่ำสุดและความชื้นสูงสุด และการกำหนดค่าความสว่าง เป็นต้น

**4.2.2 วิธีการทดลอง**

ทดลองใช้งาน Windows Application ในการควบคุมระบบภายในโรงเรือนเพาะเห็ด ขั้นตอนแรกเมื่อเข้าใช้งาน Windows Application จะต้องทำการเข้าสู่ระบบก่อนดังรูปที่ 4.13

|  |
| --- |
| **รูปที่ 4.13** แสดงภาพหน้า Login ของวินโดว์แอปพลิเคชัน |

จากรูปที่ 4.13 จะต้องทำการกรอก Username และ Password ให้ถูกต้องก่อนเข้าใช้งาน Windows Application หากกรอก Username หรือ Password ไม่ถูกต้องจะแสดงข้อความแจ้งเตือนดังรูปที่ 4.14

|  |
| --- |
| **รูปที่ 4.14** แสดงภาพแสดงข้อความแจ้งเตือนเมื่อกรอกข้อมูลไม่ถูกต้อง |
| **รูปที่ 4.15** แสดงภาพวินโดว์แอปพลิเคชันเมื่อยังไม่ได้เชื่อมต่อกับ Serial Port |

จากรูปที่ 4.15 เมื่อทำการกรอก Username และ Password ถูกต้องแล้วจะเข้าสู่หน้าแรกของ Windows Application ที่ยังไม่เชื่อมต่อกับ Serial Port เมื่อเชื่อมต่อกับ Serial Port แล้ว ESP32LoRa จะส่งข้อมูลผ่าน Serial Port เพื่อมาแสดงบน Windows Application ดังรูปที่ 4.16

|  |
| --- |
| **รูปที่ 4.16** แสดงภาพวินโดว์แอปพลิเคชันเมื่อเชื่อมต่อกับ Serial Port |

จากรูปที่ 4.16 เมื่อวินโดว์แอปพลิเคชัน (Windows Application)ทำการเชื่อมต่อกับ Serial Port แล้วจะนำข้อมูลที่ได้มาแสดง เช่น อุณหภูมิ ความชื้น ความสว่าง โหมดการทำงาน ค่าต่างๆที่กำหนดไว้ เป็นต้น ขั้นตอนต่อมาทำการทดลองควบคุมการทำงานของระบบผ่านทางวินโดว์แอปพลิเคชัน (Windows Application)

1. ทดลองเปลี่ยนโหมดการทำงาน

จากรูปที่ 4.16 จะเห็นว่าโหมดการทำงานเป็นโหมด AUTO จากนั้นเมื่อคลิกที่ปุ่มเปลี่ยนโหมดการทำงาน (Change mode) ดังรูปที่ 4.17

|  |
| --- |
| **รูปที่ 4.17** แสดงภาพปุ่มเปลี่ยนโหมดการทำงานของวินโดว์แอปพลิเคชัน |

จากรูปที่ 4.17 เมื่อทำการกดที่ปุ่มเปลี่ยนโหมดการทำงาน (Change mode) วินโดว์แอปพลิเคชัน (Windows Application) จะเปลี่ยนโหมดการทำงานจากโหมด AUTO เป็นโหมด MANUAL และแสดงปุ่มควบคุม เปิด/ปิด พัดลมและปั๊มดังรูปที่ 4.18

|  |
| --- |
| **รูปที่ 4.18** แสดงภาพวินโดว์แอปพลิเคชันเมื่อโหมดการทำงานเป็นโหมด MANUAL |

1. ทดลองควบคุมพัดลมและปั๊ม

จากรูปที่ 4.18 จะเห็นว่าปุ่มควบคุมพัดลม (Turn off Fan) เป็นสีแดงหมายความว่าพัดลมกำลังทำงาน จากนั้นทำงานสั่งปิดพัดลมโดยกดที่ปุ่มควบคุมพัดลม (Turn off Fan) ดังรูปที่ 4.19

|  |
| --- |
| **รูปที่ 4.19** แสดงภาพปุ่มควบคุมพัดลม (Turn off Fan) ขณะที่พัดลมกำลังทำงาน |

|  |
| --- |
| **รูปที่ 4.20** แสดงภาพวินโดว์แอปพลิเคชันหลังจากสั่งปิดพัดลม |

จากรูปที่ 4.20 เมื่อกดที่ปุ่มควบคุมพัดลม (Turn off Fan) วินโดว์แอปพลิเคชัน (Windows Application) จะเปลี่ยนสถานการณ์ทำงานของพัดลมเป็น OFF ตัวสีแดง ต่อมาทำการควบคุมปั๊มโดยกดที่ปุ่มควบคุมปั๊ม (Turn on Pump) ดังรูปที่ 4.21

|  |
| --- |
| **รูปที่ 4.21** แสดงภาพปุ่มควบคุมปั๊ม (Turn on Fan) ขณะที่ปั๊มไม่ทำงาน |
| **รูปที่ 4.22** แสดงภาพวินโดว์แอปพลิเคชันหลังจากสั่งเปิดปั๊ม |

จากรูปที่ 4.22 เมื่อกดที่ปุ่มควบคุมปั๊ม (Turn on Pump) วินโดว์แอปพลิเคชัน (Windows Application) จะเปลี่ยนสถานการณ์ทำงานของปั๊มเป็น ON ตัวสีเขียว

1. ทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด

ทดลองกำหนดค่าอุณหภูมิต่ำสุดโดยค่าอุณหภูมิต่ำสุดมีค่าเท่ากับ 25 องศาเซลเซียส (°C) จากนั้นทำการเปลี่ยนค่าอุณหภูมิต่ำสุดเป็น 26 องศาเซลเซียส (°C) โดยกรอกที่ textbox ดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 4.23** แสดงภาพการทดลองเปลี่ยนแปลงค่าอุณหภูมิต่ำสุด (1) |
| **รูปที่ 4.24** แสดงภาพการทดลองเปลี่ยนแปลงค่าอุณหภูมิต่ำสุด (2) |

จากรูปที่ 4.24 เมื่อทำการทดลองเปลี่ยนแปลงค่าอุณหภูมิต่ำสุดแล้ววินโดว์แอปพลิเคชัน (Windows Application) จะทำการแสดงค่าอุณหภูมิต่ำสุดตามที่ทำการทดลอง ขั้นตอนต่อมาทำการทดลองเปลี่ยนแปลงค่าอุณหภูมิสูงสุด โดยทำการทดลองกำหนดค่าอุณหภูมิสูงสุดโดยค่าอุณหภูมิสูงสุดมีค่าเท่ากับ 30 องศาเซลเซียส (°C) จากนั้นทำการเปลี่ยนค่าอุณหภูมิสูงสุดเป็น 31 องศาเซลเซียส (°C) โดยกรอกที่ textbox ดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 4.25** แสดงภาพการทดลองเปลี่ยนแปลงค่าอุณหภูมิสูงสุด (1) |

|  |
| --- |
| **รูปที่ 4.26** แสดงภาพการทดลองเปลี่ยนแปลงค่าอุณหภูมิสูงสุด (2) |

จากรูปที่ 4.26 เมื่อทำการทดลองเปลี่ยนแปลงค่าอุณหภูมิสูงสุดแล้ววินโดว์แอปพลิเคชัน (Windows Application) จะทำการแสดงค่าอุณหภูมิสูงสุดตามที่ทำการทดลอง

1. ทดลองกำหนดค่าความชื้นต่ำสุดและความชื้นสูงสุด

ทดลองกำหนดค่าความชื้นต่ำสุดโดยค่าความชื้นต่ำสุดมีค่าเท่ากับ 80 เปอร์เซ็นต์ (%) จากนั้นทำการเปลี่ยนค่าความชื้นต่ำสุดเป็น 81 เปอร์เซ็นต์ (%) โดยกรอกที่ textbox ดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 4.27** แสดงภาพการทดลองเปลี่ยนแปลงค่าความชื้นต่ำสุด (1) |
| **รูปที่ 4.27** แสดงภาพการทดลองเปลี่ยนแปลงค่าความชื้นต่ำสุด (2) |

จากรูปที่ 4.27 เมื่อทำการทดลองเปลี่ยนแปลงค่าความชื้นต่ำสุดแล้ววินโดว์แอปพลิเคชัน (Windows Application) จะทำการแสดงค่าความชื้นต่ำสุดตามที่ทำการทดลอง

ขั้นตอนต่อมาทำการทดลองเปลี่ยนแปลงค่าความชื้นสูงสุด โดยทำการทดลองกำหนดค่าความชื้นสูงสุดโดยค่าความชื้นสูงสุดมีค่าเท่ากับ 90 เปอร์เซ็นต์ (%) จากนั้นทำการเปลี่ยนค่าความชื้นสูงสุดเป็น 91 เปอร์เซ็นต์ (%) โดยกรอกที่ textbox ดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 4.28** แสดงภาพการทดลองเปลี่ยนแปลงค่าความชื้นสูงสุด (1) |

|  |
| --- |
| **รูปที่ 4.29** แสดงภาพการทดลองเปลี่ยนแปลงค่าความชื้นสูงสุด (2) |

จากรูปที่ 4.29 เมื่อทำการทดลองเปลี่ยนแปลงค่าความชื้นสูงสุดแล้ววินโดว์แอปพลิเคชัน (Windows Application) จะทำการแสดงค่าความชื้นสูงสุดตามที่ทำการทดลอง

1. ทดลองกำหนดค่าความสว่าง

ทดลองกำหนดค่าความสว่างโดยค่าความสว่างมีค่าเท่ากับ 200 ลักซ์ (lx) จากนั้นทำการเปลี่ยนค่าความสว่างเป็น 230 ลักซ์ (lx) โดยกรอกที่ textbox ดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 4.30** แสดงภาพการทดลองเปลี่ยนแปลงค่าความสว่าง (1) |

|  |
| --- |
| **รูปที่ 4.31** แสดงภาพการทดลองเปลี่ยนแปลงค่าความสว่าง (2) |

จากรูปที่ 4.31 เมื่อทำการทดลองเปลี่ยนแปลงค่าความสว่างแล้ววินโดว์แอปพลิเคชัน (Windows Application) จะทำการแสดงค่าความสว่างตามที่ทำการทดลอง

**4.2.3 ผลการทดลอง**

จากการทดลองควบคุมระบบภายในโรงเรือนจากวินโดว์แอปพลิเคชัน (Windows Application) ได้ผลการทดลองคือ วินโดว์แอปพลิเคชัน (Windows Application) สามารถควบคุมการทำงานของระบบภายในโรงเรือนเพาะเห็ดได้

**4.3** **การทดลองควบคุมระบบภายในโรงเรือนจากเว็บแอปพลิเคชัน (Web Application)**

**4.3.1 วัตถุประสงค์**

เพื่อทดลองใช้งานเว็บแอปพลิเคชัน (Web Application) ในการควบคุมระบบภายในโรงเรือนเพาะเห็ด เช่น การเปลี่ยนโหมดการทำงาน เปิด/ปิดปั๊มพ่นหมอกและพัดลมระบายอากาศ กำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด กำหนดค่าความชื้นต่ำสุดและความชื้นสูงสุด และดูประวัติข้อมูลในฐานข้อมูล เป็นต้น

**4.3.2 วิธีการทดลอง**

ทดลองใช้งานเว็บแอปพลิเคชัน (Web Application) ในการควบคุมระบบภายในโรงเรือนเพาะเห็ด ขั้นตอนแรกเมื่อเข้าใช้งานเว็บแอปพลิเคชัน (Web Application) จะต้องทำการเข้าสู่ระบบก่อนดังรูปที่ 4.32

|  |
| --- |
| **รูปที่ 4.32** แสดงภาพหน้าเข้าสู่ระบบก่อนใช้งานเว็บแอปพลิเคชัน (Web Application) |

จากรูปที่ 4.32 จะต้องทำการกรอก Username และ Password ให้ถูกต้องก่อนเข้าใช้งาน เว็บแอปพลิเคชัน (Web Application) หากกรอก Username หรือ Password ไม่ถูกต้องระบบจะเด้งกลับมาให้กรอกอีกครั้ง

|  |
| --- |
| **รูปที่ 4.33** แสดงภาพหน้าแรกของเว็บแอปพลิเคชัน (Web Application) |

จากรูปที่ 4.33 เมื่อทำการกรอก Username และ Password ถูกต้องแล้วจะเข้าสู่หน้าแรกของเว็บแอปพลิเคชัน (Web Application)

1. ทดลองเปลี่ยนโหมดการทำงาน

จากรูปที่ 4.33 จะเห็นว่าโหมดการทำงานเป็นโหมด AUTO จากนั้นเมื่อคลิกที่ปุ่มเปลี่ยนโหมดการทำงาน (change mode) ดังรูปที่ 4.34

|  |
| --- |
| **รูปที่ 4.34** แสดงภาพการทดลองเปลี่ยนแปลงโหมดการทำงาน (1) |

จากรูปที่ 4.34 เมื่อทำการกดที่ปุ่มเปลี่ยนโหมดการทำงาน (Change mode) เว็บแอปพลิเคชัน (Web Application) จะเปลี่ยนโหมดการทำงานจากโหมด AUTO เป็นโหมด MANUAL และแสดงปุ่มควบคุม เปิด/ปิด พัดลมและปั๊มดังรูปที่ 4.35

|  |
| --- |
| **รูปที่ 4.35** แสดงภาพการทดลองเปลี่ยนแปลงโหมดการทำงาน (2) |

1. ทดลองควบคุมพัดลมและปั๊ม

จากรูปที่ 4.35 จะเห็นว่าเว็บแอปพลิเคชัน (Web Application) จะแสดงปุ่มควบคุมพัดลม (Control -FAN) และควบคุมปั๊ม (Control PUMP) ปุ่มควบคุมเป็นสีแดงหมายความว่าอุปปกรณ์กำลังทำงาน จากนั้นทำงานสั่งปิดพัดลมโดยกดที่ปุ่มควบคุมพัดลม (Control -FAN) ดังรูปที่ 4.36

|  |
| --- |
| **รูปที่ 4.36** แสดงภาพการทดลองควบคุมพัดลมและปั๊ม (1) |
| **รูปที่ 4.37** แสดงภาพการทดลองควบคุมพัดลมและปั๊ม (2) |

จากรูปที่ 4.37 เว็บแอปพลิเคชัน (Web Application) เมื่อสั่งปิดพัดลมแล้วจะทำการแสดงสถานะการทำงานของพัดลมเป็น OFF และแสดงปุ่มควบคุมพัดลม (Control -FAN) เป็นสีเทา ขั้นตอนต่อมาทำการทดลองควบคุมปั๊ม จากรูปที่ 4.ป สาถานะการทำงานของปั๊มเป็น OFF ดังนั้นทำการทดลองสั่งเปิดปั๊มโดยกดที่ปุ่มควบคุมปั๊ม (Control -PUMP) ดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 4.38** แสดงภาพการทดลองควบคุมพัดลมและปั๊ม (3) |

|  |
| --- |
| **รูปที่ 4.39** แสดงภาพการทดลองควบคุมพัดลมและปั๊ม (4) |

จากรูปที่ 4.39 เว็บแอปพลิเคชัน (Web Application) เมื่อสั่งเปิดปั๊มแล้วจะทำการแสดงสถานะการทำงานของปั๊มเป็น ON และแสดงปุ่มควบคุมปั๊ม (Control -PUMP) เป็นสีแดง

1. ทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด

ทำการทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุดโดยการคลิกที่แถบเมนูนำทางที่อยู่ด้านบนซ้ายของหน้าจอดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 4.40** แสดงภาพการทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด (1) |

เมื่อทำการคลิกที่แถบเมนูนำทางเว็บแอปพลิเคชันจะแสดงเมนูนำทางขึ้นมาดังรูปที่ 4.41

|  |
| --- |
| **รูปที่ 4.41** แสดงภาพการทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด (2) |

จากรูปที่ 4.41 เมื่อเว็บแอปพลิเคชัน (Web Application) แสดงเมนูนำทางขึ้นมาแล้วทำการคลิกที่ Setting ดังรูปที่แสดงด้านล่างเพื่อทำการกำหนดค่าต่างๆของระบบ

|  |
| --- |
| **รูปที่ 4.42** แสดงภาพการทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด (3) |

จากรูปที่ 4.42 เมื่อคลิกที่ Setting เว็บแอปพลิเคชัน (Web Application) จะแสดงหน้า Setting ขึ้นมาดังรูปที่แสดงด้านล่าง โดยหน้า Setting สามารถกำหนดค่าต่างๆภายในระบบและเปิดปิดการแจ้งเตือนของ LINE Notify ได้

|  |
| --- |
| **รูปที่ 4.43** แสดงภาพการทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด (4) |

ทดลองกำหนดค่าอุณหภูมิต่ำสุดโดยค่าอุณหภูมิต่ำสุดมีค่าเท่ากับ 25 องศาเซลเซียส (°C) จากนั้นทำการเปลี่ยนค่าอุณหภูมิต่ำสุดเป็น 26 องศาเซลเซียส (°C) โดยกรอกที่ textbox ดังรูปที่แสดงด้านล่าง

|  |
| --- |
| **รูปที่ 4.44** แสดงภาพการทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด (5) |
| **รูปที่ 4.45** แสดงภาพการทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด (6) |

จากรูปที่ 4.45 เมื่อทำการทดลองเปลี่ยนแปลงค่าอุณหภูมิต่ำสุดแล้วเว็บแอปพลิเคชัน (Web Application) จะทำการแสดงค่าอุณหภูมิต่ำสุดตามที่ทำการทดลอง ขั้นตอนต่อมาทำการทดลองเปลี่ยนแปลงค่าอุณหภูมิสูงสุด โดยทำการทดลองกำหนดค่าอุณหภูมิสูงสุดโดยค่าอุณหภูมิสูงสุดมีค่าเท่ากับ 30 องศาเซลเซียส (°C) จากนั้นทำการเปลี่ยนค่าอุณหภูมิสูงสุดเป็น 31 องศาเซลเซียส (°C) โดยกรอกที่ textbox ดังรูปที่ 4.46

|  |
| --- |
| **รูปที่ 4.46** แสดงภาพการทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด (7) |

|  |
| --- |
| **รูปที่ 4.47** แสดงภาพการทดลองกำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด (8) |

จากรูปที่ 4.47 เมื่อทำการทดลองเปลี่ยนแปลงค่าอุณหภูมิสูงสุดแล้วเว็บแอปพลิเคชัน (Web Application) จะทำการแสดงค่าอุณหภูมิสูงสุดตามที่ทำการทดลอง

1. ทดลองกำหนดค่าความชื้นต่ำสุดและความชื้นสูงสุด

ทดลองกำหนดค่าความชื้นต่ำสุดโดยค่าความชื้นต่ำสุดมีค่าเท่ากับ 80 เปอร์เซ็นต์ (%) จากนั้นทำการเปลี่ยนค่าความชื้นต่ำสุดเป็น 82 เปอร์เซ็นต์ (%) โดยกรอกที่ textbox ดังรูปที่ 4.48

|  |
| --- |
| **รูปที่ 4.48** แสดงภาพการทดลองเปลี่ยนแปลงค่าความชื้นต่ำสุด (1) |

|  |
| --- |
| **รูปที่ 4.49** แสดงภาพการทดลองเปลี่ยนแปลงค่าความชื้นต่ำสุด (2) |

จากรูปที่ 4.49 เมื่อทำการทดลองเปลี่ยนแปลงค่าความชื้นต่ำสุดแล้วเว็บแอปพลิเคชัน (Web Application) จะทำการแสดงค่าความชื้นต่ำสุดตามที่ทำการทดลอง

ขั้นตอนต่อมาทำการทดลองเปลี่ยนแปลงค่าความชื้นสูงสุด โดยทำการทดลองกำหนดค่าความชื้นสูงสุดโดยค่าความชื้นสูงสุดมีค่าเท่ากับ 90 เปอร์เซ็นต์ (%) จากนั้นทำการเปลี่ยนค่าความชื้นสูงสุดเป็น 89 เปอร์เซ็นต์ (%) โดยกรอกที่ textbox ดังรูปที่ 4.50

|  |
| --- |
| **รูปที่ 4.50** แสดงภาพการทดลองเปลี่ยนแปลงค่าความชื้นสูงสุด (1) |
| **รูปที่ 4.51** แสดงภาพการทดลองเปลี่ยนแปลงค่าความชื้นสูงสุด (2) |

จากรูปที่ 4.51 เมื่อทำการทดลองเปลี่ยนแปลงค่าความชื้นสูงสุดแล้วเว็บแอปพลิเคชัน (Web Application) จะทำการแสดงค่าความชื้นสูงสุดตามที่ทำการทดลอง

1. ทดลองกำหนดค่าความสว่าง

ทดลองกำหนดค่าความสว่างโดยค่าความสว่างมีค่าเท่ากับ 200 ลักซ์ (lx) จากนั้นทำการเปลี่ยนค่าความสว่างเป็น 210 ลักซ์ (lx) โดยกรอกที่ textbox ดังรูปที่ 4.52

|  |
| --- |
| **รูปที่ 4.52** แสดงภาพการทดลองเปลี่ยนแปลงค่าความสว่าง (1) |

|  |
| --- |
| **รูปที่ 4.53** แสดงภาพการทดลองเปลี่ยนแปลงค่าความสว่าง (2) |

จากรูปที่ 4.53 เมื่อทำการทดลองเปลี่ยนแปลงค่าความสว่างแล้วเว็บแอปพลิเคชัน (Web Application) จะทำการแสดงค่าความสว่างตามที่ทำการทดลอง

**4.3.3 ผลการทดลอง**

จากการทดลองควบคุมระบบภายในโรงเรือนจากเว็บแอปพลิเคชัน (Web Application) ได้ผลการทดลองคือ เว็บแอปพลิเคชัน (Web Application) สามารถควบคุมการทำงานของระบบภายในโรงเรือนเพาะเห็ดได้

**4.4** **การทดลองเพาะเห็ด**

**4.4.1 วัตถุประสงค์**

เพื่อทดสอบประสิทธิภาพของโรงเรือนเพาะเห็ดอัจฉริยะในการทดลองเพาะเห็ดจริง โดยใช้เห็ดนางฟ้าในการทำการทดลอง

**4.4.2 วิธีการทดลอง**

ทำการทดลองเพาะเห็ดนางฟ้าระหว่างวันที่ 8 ตุลาคม พ.ศ. 2563 ถึงวันที่ 21 ตุลาคม พ.ศ. 2563 รวมทั้งหมด 14 วัน

การรักษาสภาพแวดล้อมภายในโรงเรือนเพาะเห็ดให้เหมาะสมต่อการออกดอกของเห็ด จะต้องควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอก เช่น อุณหภูมิ ความชื้น และแสงสว่าง โดยผู้จัดทำได้นำเห็ดนางฟ้ามาใช้ในการทำการทดลองเพาะเห็ด จากการศึกษาสภาพแวดล้อมที่เหมาะสมกับการออกดอกของเห็ดนางฟ้าสามารถสรุปได้ดังนี้ อุณหภูมิควรอยู่ระหว่าง 25-30 องศาเซลเซียส (°C) ความชื้นสัมพัทธ์ 80-90 เปอร์เซ็นต์ (%) แสงสว่างที่มีความเข้มของแสงประมาณ 200 lux เป็นแสงสีน้ำเงินจะมีทำให้เห็ดออกดอกได้ดีที่สุด

**ตารางที่ 4.1** แสดงค่าอุณหภูมิและความชื้นที่มีผลต่อการออกดอกของเห็ดนางฟ้า

|  |  |  |
| --- | --- | --- |
| ชื่อเห็ด | ระยะที่เจริญเป็ดดอกเห็ด | |
| เห็ดนางฟ้า | อุณหภูมิ (°C) | ความชื้น (%) |
| 25-30 | 80-90 |

ที่มา (Peter Oei, 2005)

|  |
| --- |
| **รูปที่ 4.54** แสดงภาพผลการทดลองการเจริญเติบโตของเห็ดต่อแสงแต่ละสี  ที่มา (http://slowacki.kielce.eu/IB/PSlusarczyk.pdf) |

จากรูปที่ 4.54 จะเห็นว่าแสงสีน้ำเงินมีผลให้เห็ดสามารถเจริญเติมโตได้ดีมากกว่าแสงสีอื่นโครงสร้างของระบบที่ใช้ในการควบคุมสภาพแวดล้อมภายในโรงเรือนจะประกอบด้วยเซนเซอร์วัดอุณหภูมิและความชื้น เพื่อวัดอุณหภูมิและความชื้นภายในโรงเรือนระบบปิด จากนั้นนำข้อมูลอุณหภูมิและความชื้นที่ได้ไปเปรียบเทียบกับค่าที่ตั้งไว้คือ อุณหภูมิควรอยู่ระหว่าง 25-30 องศาเซลเซียส (°C) ความชื้นสัมพัทธ์ 80-90 เปอร์เซ็นต์ (%) หากอุณหภูมิและความชื้นไม่อยู่ภายในช่วงที่กำหนด จะควบคุมการทำงานของปี๊มพ่นหมอกและพัดลมระบายอากาศ เพื่อใช้อุณหภูมิและความชื้นอยู่ในช่วงที่กำหนด โดยการควบคุมการทำงานของปั๊มพ่นหมอกและพัดลมระบายอากาศจะมีการทำงานตามตารางที่ 4.2

**ตารางที่ 4.2** แสดงการทำงานของปั๊มพ่นหมอกและพัดลมในโหมดการทำงานแบบอัตโนมัติ (AUTO)

|  |  |  |  |
| --- | --- | --- | --- |
| **อุณหภูมิ(°C)** | **ความชื้น(%)** | **ปั๊มพ่นหมอก** | **พัดลม** |
| < 25 | < 80 | ON | OFF |
| < 25 | 80-90 | OFF | OFF |
| < 25 | > 90 | OFF | OFF |
| 25-30 | < 80 | ON | OFF |
| 25-30 | 80-90 | OFF | OFF |
| 25-30 | > 90 | OFF | OFF |
| > 30 | < 80 | ON | ON |
| > 30 | 80-90 | ON | ON |
| > 30 | > 90 | ON | ON |

ในการควบคุมแสงสว่างภายในโรงเรือนผู้จัดทำได้นำ LED ขนาด 5 mm สีน้ำเงินมาใช้ในการให้แสงสว่างภายในโรงเรือน โดยกำหนดค่าความเข้มของแสงที่ 200 lux หากค่าความเข้มของแสงน้อยกว่า 200 lux ก็จะเพิ่มความสว่างของหลอด LED หากความสว่างมากกว่า 200 lux ก็จะลดความสว่างของหลอด LED ทำการควบคุมแสงสว่าง 12 ชั่วโมงต่อวันตั้งแต่เวลา 06.00-18.00 น.

การทดลองเพาะเห็ดผู้จัดทำได้ทำการเปรียบเทียบการเจริญเติบโตของเห็ดโดยทำการเพาะเห็ดนางฟ้าจำนวน 20 ก้อน โดยที่ 10 ก้อนจะเพาะภายในโรงเรือนที่สามารถควบคุมอุณหภูมิความชื้นและแสงสว่างได้ อีก 10 จะเพาะภายในโรงเรือนที่ไม่มีการควบคุมอุณหภูมิความชื้นและแสงสว่างดังรูปที่ 4.55 จากนั้นทำการเปรียบเทียบความกว้างของดอกและน้ำหนักของเห็ดเมื่อทำการเก็บเกี่ยวผลผลิต

|  |
| --- |
| **รูปที่ 4.55** แสดงภาพของโรงเรือนที่สร้างขึ้นเพื่อใช้ในการเปรียบเทียบผลการทดลอง |
|  |
|  |

**4.4.3 ผลการทดลอง**

ผลการทดลองเพาะเห็ดนางฟ้าจะแบ่งออกเป็น 2 ช่วงเพราะทำการเพาะเห็ดทั้งหมด 2 รุ่นเริ่มเปิดดอกเห็ดวันที่ 8 ตุลาคม 2563 โดยเห็ดรุ่นที่ 1 ทำการเก็บเกี่ยวผลผลิตวันที่ 11 ตุลาคม 2563 รุ่นที่ 2 ทำการเก็บเกี่ยวผลผลิตวันที่ 21 ตุลาคม 2563

1. ผลการทดลองเพาะเห็ดรุ่นที่ 1

|  |  |
| --- | --- |
| (ก.) | (ข.) |
| **รูปที่ 4.56** แสดงภาพผลการทดลองวันที่ 9 ตุลาคม 2563  (ก.) ผลการทดลองภายในโรงเรือนที่มีการควบคุมสภาพแวดล้อม  (ข.) ผลการทดลองภายในโรงเรือนที่ไม่มีการควบคุมสภาพแวดล้อม | |

|  |  |
| --- | --- |
| (ก.) | (ข.) |
| **รูปที่ 4.57** แสดงภาพผลการทดลองวันที่ 10 ตุลาคม 2563  (ก.) ผลการทดลองภายในโรงเรือนที่มีการควบคุมสภาพแวดล้อม  (ข.) ผลการทดลองภายในโรงเรือนที่ไม่มีการควบคุมสภาพแวดล้อม | |
| (ก.) | (ข.) |
| **รูปที่ 4.58** แสดงภาพผลการทดลองวันที่ 11 ตุลาคม 2563  (ก.) ผลการทดลองภายในโรงเรือนที่มีการควบคุมสภาพแวดล้อม  (ข.) ผลการทดลองภายในโรงเรือนที่ไม่มีการควบคุมสภาพแวดล้อม | |

|  |  |
| --- | --- |
| (ก.) | (ข.) |
| **รูปที่ 4.59** แสดงภาพผลผลิตที่ได้จากการทดลองเพาะเห็ดนางฟ้า  (ก.) ผลผลิตที่ได้จากโรงเรือนที่มีการควบคุมสภาพแวดล้อมน้ำหนัก 0.7 กิโลกรัม  (ข.) ผลผลิตที่ได้จากโรงเรือนที่ไม่มีการควบคุมสภาพแวดล้อมน้ำหนัก 0.4 กิโลกรัม | |

จากผลการทดลองด้านบนสามารถสรุปผลการทดลองได้ดังนี้ การเพาะเห็ดในโรงเรือนที่มีการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยก้อนละ 6.18 เซนติเมตร มีน้ำหนักทั้งหมด 0.7 กิโลกรัม การเพาะเห็ดในโรงเรือนที่ไม่มีกาการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยก้อนละ 5.67 เซนติเมตร มีน้ำหนักทั้งหมด 0.4 กิโลกรัม

1. ผลการทดลองเพาะเห็ดรุ่นที่ 2

|  |  |
| --- | --- |
| (ก.) | (ข.) |
| **รูปที่ 4.60** แสดงภาพผลการทดลองวันที่ 19 ตุลาคม 2563  (ก.) ผลการทดลองภายในโรงเรือนที่มีการควบคุมสภาพแวดล้อม  (ข.) ผลการทดลองภายในโรงเรือนที่ไม่มีการควบคุมสภาพแวดล้อม | |

|  |  |
| --- | --- |
| (ก.) | (ข.) |
| **รูปที่ 4.61** แสดงภาพผลการทดลองวันที่ 20 ตุลาคม 2563  (ก.) ผลการทดลองภายในโรงเรือนที่มีการควบคุมสภาพแวดล้อม  (ข.) ผลการทดลองภายในโรงเรือนที่ไม่มีการควบคุมสภาพแวดล้อม | |
| (ก.) | (ข.) |
| **รูปที่ 4.62** แสดงภาพผลการทดลองวันที่ 21 ตุลาคม 2563  (ก.) ผลการทดลองภายในโรงเรือนที่มีการควบคุมสภาพแวดล้อม  (ข.) ผลการทดลองภายในโรงเรือนที่ไม่มีการควบคุมสภาพแวดล้อม | |

|  |  |
| --- | --- |
| (ก.) | (ข.) |
| **รูปที่ 4.63** แสดงภาพผลผลิตที่ได้จากการทดลองเพาะเห็ดนางฟ้า  (ก.) ผลผลิตที่ได้จากโรงเรือนที่มีการควบคุมสภาพแวดล้อมน้ำหนัก 0.36 กิโลกรัม  (ข.) ผลผลิตที่ได้จากโรงเรือนที่ไม่มีการควบคุมสภาพแวดล้อมน้ำหนัก 0.2 กิโลกรัม | |

จากผลการทดลองด้านบนสามารถสรุปผลการทดลองได้ดังนี้ การเพาะเห็ดในโรงเรือนที่มีการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยก้อนละ 9.14 เซนติเมตร มีน้ำหนักทั้งหมด 0.36 กิโลกรัม การเพาะเห็ดในโรงเรือนที่ไม่มีกาการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยก้อนละ 7.22 เซนติเมตร มีน้ำหนักทั้งหมด 0.2 กิโลกรัม

จากการทดลองเพาะเห็ดนางฟ้าระหว่างวันที่ 8 ตุลาคม พ.ศ. 2563 ถึงวันที่ 21 ตุลาคม พ.ศ. 2563 รวมทั้งหมด 14 วัน สามารถสรุปผลการทดลองได้ดังนี้

รุ่นที่ 1 ทำการเก็บเกี่ยวผลผลิตวันที่ 11 ตุลาคม 2563 การเพาะเห็ดในโรงเรือนที่มีการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ด มีความกว้างของดอกเฉลี่ยก้อนละ 6.18 เซนติเมตร มีน้ำหนักทั้งหมด 0.7 กิโลกรัม การเพาะเห็ดในโรงเรือนที่ไม่มีกาการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยก้อนละ 5.67 เซนติเมตร มีน้ำหนักทั้งหมด 0.4 กิโลกรัม

รุ่นที่ 2 ทำการเก็บเกี่ยวผลผลิตวันที่ 21 ตุลาคม 2563 การเพาะเห็ดในโรงเรือนที่มีการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ด มีความกว้างของดอกเฉลี่ยก้อนละ 9.14 เซนติเมตร มีน้ำหนักทั้งหมด 0.36 กิโลกรัม การเพาะเห็ดในโรงเรือนที่ไม่มีกาการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยก้อนละ 7.22 เซนติเมตร มีน้ำหนักทั้งหมด 0.2 กิโลกรัม

ดังนั้นการเพาะเห็ดในโรงเรือนที่มีการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ด มีความกว้างของดอกเฉลี่ยเท่ากับ 7.66 เซนติเมตร มีน้ำหนักทั้งหมดเท่ากับ 1.06 กิโลกรัม การเพาะเห็ดในโรงเรือนที่ไม่มีกาการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยเท่ากับ 6.45 เซนติเมตร มีน้ำหนักทั้งหมดเท่ากับ 0.6 กิโลกรัม

**บทที่ 5**

**สรุปผลการดำเนินงาน**

**5.1 สรุปผล**

ปริญญานิพนธ์นี้วัตถุประสงค์เพื่อออกแบบและสร้างเว็บแอปพลิเคชันและวินโดว์แอปพลิเคชันสำหรับมอนิเตอร์และควบคุมระบบโรงเรือนเพาะเห็ด ออกแบบและสร้างโรงเรือนที่ใช้สำหรับเพาะเห็ด ออกแบบระบบควบคุมอุณหภูมิความชื้นและแสงสว่างภายในโรงเรือนเพาะเห็ด ศึกษาและประยุกต์ใช้งาน Lora Communication สำหรับการควบคุมโรงเรือนเพาะเห็ด และเพื่อศึกษาการพัฒนาเว็บแอปพลิเคชันบนไมโครคอนโทรลเลอร์ ESP32LoRa

การทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือนและปุ่มควบคุมที่กล่อง STA เช่น ควบคุมโหมดการทำงานของระบบ เปิดปิดพัดลมและปั๊ม จากการทดลองควบคุมระบบภายในโรงเรือนจากปุ่มควบคุมหน้าโรงเรือนและปุ่มควบคุมที่กล่อง STA ได้ผลการทดลองคือ สามารถควบคุมการทำงานของระบบภายในโรงเรือนเพาะเห็ดได้

การทดลองควบคุมระบบภายในโรงเรือนจากวินโดว์แอปพลิเคชัน (Windows Application) เช่น การเปลี่ยนโหมดการทำงาน เปิด/ปิดปั๊มพ่นหมอกและพัดลมระบายอากาศ กำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด กำหนดค่าความชื้นต่ำสุดและความชื้นสูงสุด และการกำหนดค่าความสว่าง เป็นต้น จากการทดลองควบคุมระบบภายในโรงเรือนจากวินโดว์แอปพลิเคชัน (Windows Application) ได้ผลการทดลองคือ วินโดว์แอปพลิเคชัน (Windows Application) สามารถควบคุมการทำงานของระบบภายในโรงเรือนเพาะเห็ดได้

การทดลองควบคุมระบบภายในโรงเรือนจากเว็บแอปพลิเคชัน (Web Application) เห็ด เช่น การเปลี่ยนโหมดการทำงาน เปิด/ปิดปั๊มพ่นหมอกและพัดลมระบายอากาศ กำหนดค่าอุณหภูมิต่ำสุดและอุณหภูมิสูงสุด กำหนดค่าความชื้นต่ำสุดและความชื้นสูงสุด และดูประวัติข้อมูลในฐานข้อมูล เป็นต้น จากการทดลองควบคุมระบบภายในโรงเรือนจากเว็บแอปพลิเคชัน (Web Application) ได้ผลการทดลองคือ เว็บแอปพลิเคชัน (Web Application) สามารถควบคุมการทำงานของระบบภายในโรงเรือนเพาะเห็ดได้

การทดลองเพาะเห็ดเป็นการทดสอบประสิทธิภาพของโรงเรือนเพาะเห็ดอัจฉริยะในการทดลองเพาะเห็ดจริง โดยใช้เห็ดนางฟ้าในการทำการทดลอง จากการทดลองเพาะเห็ดนางฟ้าระหว่างวันที่ 8 ตุลาคม พ.ศ. 2563 ถึงวันที่ 21 ตุลาคม พ.ศ. 2563 รวมทั้งหมด 14 วัน สามารถสรุปผลการทดลองได้ดังนี้

รุ่นที่ 1 ทำการเก็บเกี่ยวผลผลิตวันที่ 11 ตุลาคม 2563 การเพาะเห็ดในโรงเรือนที่มีการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ด มีความกว้างของดอกเฉลี่ยก้อนละ 6.18 เซนติเมตร มีน้ำหนักทั้งหมด 0.7 กิโลกรัม การเพาะเห็ดในโรงเรือนที่ไม่มีกาการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยก้อนละ 5.67 เซนติเมตร มีน้ำหนักทั้งหมด 0.4 กิโลกรัม

รุ่นที่ 2 ทำการเก็บเกี่ยวผลผลิตวันที่ 21 ตุลาคม 2563 การเพาะเห็ดในโรงเรือนที่มีการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ด มีความกว้างของดอกเฉลี่ยก้อนละ 9.14 เซนติเมตร มีน้ำหนักทั้งหมด 0.36 กิโลกรัม การเพาะเห็ดในโรงเรือนที่ไม่มีกาการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยก้อนละ 7.22 เซนติเมตร มีน้ำหนักทั้งหมด 0.2 กิโลกรัม

ดังนั้นการเพาะเห็ดในโรงเรือนที่มีการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ด มีความกว้างของดอกเฉลี่ยเท่ากับ 7.66 เซนติเมตร มีน้ำหนักทั้งหมดเท่ากับ 1.06 กิโลกรัม การเพาะเห็ดในโรงเรือนที่ไม่มีกาการควบคุมสภาพแวดล้อมให้เหมาะสมต่อการออกดอกของเห็ดโดยมีความกว้างของดอกเฉลี่ยเท่ากับ 6.45 เซนติเมตร มีน้ำหนักทั้งหมดเท่ากับ 0.6 กิโลกรัม

**5.2 ปัญหาและแนวทางแก้ไข**

5.2.1 ในการนำอุปกรณ์ประเภทความเหนี่ยวนำหรือ Inductive load เช่น ปั๊มพ่นหมอกและพัดลมระบายอากาศ มาใช้งานจะต้องคำนึงแรงดันที่พุ่งขึ้นอย่างเฉียบพลันเมื่อกระแสที่ไหลใน Inductive load ลดลงอย่างฉับพลัน เพื่อไม่ให้แรงดันนั้นไหลย้อนกลับเข้าสู่แหล่งจ่ายกระแสไฟแล้วสร้างความเสียหายให้แก่อุปกรณ์ แนวทางแก้ไขนำ flyback diode หรือ snubber diode เบอร์ 1N400x มาต่อขนานกับ Inductive load โดยหันด้านแคโทด (Cathode) ไปทางด้านไปบวก

5.2.2 การเลือกใช้งานตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) เพื่อใช้ในการตรวจสอบสถานะการทำงานของอุปกรณ์ ผู้จัดทำได้นำตัวต้านทานที่มีอัตราทนกำลังวัตต์ต่ำเกินไปทำให้เมื่ออุปกรร์เช่น พัดลมทำงาน จะทำให้ต้านต้านทานเกิดความร้อนสูง แนวทางแก้ไข นำตัวต้านทานที่มีค่าอัตราทนกำลังวัตต์สูงๆ มาใช้งานแทน

**5.3 ข้อเสนอแนะและแนวทางการพัฒนา**

5.3.1 ปริญญานิพนธ์นี้สามารถพัฒนาต่อได้โดยการนำจอสัมผัส (Touch Screen) เช่น จอ Touch Screen (HMI TFT LCD Touch) มาใช้งานในการแสดงผลและควบระบบหน้าโรงเรือนหากนำจอสัมผัส (Touch Screen) มาใช้งานจะช่วยให้สามารถแสดงค่าสาถานะต่างๆ ภายในโรงเรือนได้อย่างครบถ้าน และสามารถสร้างปุ่มเพื่อใช้ในการควบคุมระบบเพื่อลดการใช้งานขา GPIO ของไมโครคอนโทรลเลอร์จากการใช้ปุ่มควบคุมแบบฮาร์ดแวร์

5.3.2 ปริญญานิพนธ์นี้สามารถพัฒนาต่อได้โดยการพัฒนาระบบให้สามารถรองรับโรงเรือนที่มากกว่า 1 โรงเรือน

5.3.3 ในการส่งข้อมูลระหว่างโหนด (node) ควรส่งข้อมูลที่มีรูปแบบ JSON เพื่อเพิ่มความสะดวกในการนำข้อมูลมาใช้งาน จากเดิมจะส่งข้อมูลที่มีรูปแบบเป็นแถวต่อกัน (data raw) การที่จะนำข้อมูลมาใช้งานจะต้องทำการ substring ดังนั้นความยาวของข้อมูลจะต้องคงที่หากความยาวของข้อมูลมีการเปลี่ยนแปลงอาจที่ให้ข้อมูลที่ได้เกิดความผิดพลาด ฉะนั้นการส่งข้อมูลในรูปแบบ JSON จะช่วยแก้ปัญหาที่กล่าวมาได้

**บรรณานุกรม**

[1] ขันติชัย รุจิตระการโชติกุล และชยุต สุรกุล. 2560. **“ไอโอทีแพลทฟอร์มโรงเพาะเห็ด”**.ปริญญาตรี สาขาวิชาวิศวกรรมคอมพิวเตอร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

[2] วีรศักดิ์ ฟองเงิน สุรพงษ์ เพ็ชร์หาญ และรัฐสิทธิ์ ยะจ่อ. 2560. **“****การประยุกต์ใช้เทคโนโลยีไอโอทีควบคุมฟาร์มอัจฉริยะในโรงเรือนเพาะเห็ดนางฟ้า”**. ปริญญาตรี คณะวิทยาศาสตร์ มหาวิทยาลัยราชภัฏลำปาง

[3] Oran Chieochan and Anukit Saokaew. 2017. **“****IOT for Smart Farm: A case study of the Lingzhi Mushroom Farm at Maejo University”**. Bachelor’s Thesis, Chiang Mai University, Thailand.

[4] Parvati Bhandari and Megha Kimothi. 2018. **“****IOT BASED DESIGN IMPLEMENTATION OF MUSHROOM FARM MONITORING USING ARDUINO MICROCONTROLLERS & SENSORS”**. Bachelor’s Thesis, Dept. of Wireless Mobile Communication Engg, GRD IMT, Dehradun, India

[5] สำนักงานพัฒนาธุรกรรมทางอิเล็กทรอนิกส์ (สพธอ.) (2563) **ETDA เผย ปี 62 คนไทยใช้อินเทอร์เน็ตเพิ่มขึ้นเฉลี่ย 10 ชั่วโมง 22 นาที Gen Y ครองแชมป์ 5 ปีซ้อน** [ออนไลน์] เข้าถึงได้จากURL: https://www.etda.or.th/content/thailand-internet-user-behavior-2019-press-release.html (8 สิงหาคม 2563)

[6] กรมส่งเสริมการเกษตร. (ม.ป.ป.) **เห็ดนางฟ้า** [ออนไลน์] เข้าถึงได้จาก URL:http://www.servicelink.doae.go.th/webpage/book%20PDF/mushroom/m015.pdf (8 สิงหาคม 2563)

[7] Nobita [นามแฝง] (2560) **ESP32 เบื้องต้น** [ออนไลน์] เข้าถึงได้จาก URL:http://share.psu.ac.th/blog/service-engineer/44239 (8 สิงหาคม 2563)

[8] components101 [นางแฝง] (2560) **SX1278 LoRa RF Module** [ออนไลน์] เข้าถึงได้จากURL:https://components101.com/wireless/sx1278-lora-rf-module-features-dimension-datasheet (8 สิงหาคม 2563)

[9] raspberrypi [นามแฝง] (2558) **Raspberry Pi 3 Model B** [ออนไลน์] เข้าถึงได้จาก URL: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/ (8 สิงหาคม 2563)

[10] AOSONG [นามแฝง] (ม.ป.ป.) **Digital temperature and humidity Sensor AM2315 Product Manual** [ออนไลน์] เข้าถึงได้จาก

URL: https://cdn-shop.adafruit.com/datasheets/AM2315.pdf (10 สิงหาคม 2563)

[11] KimJaeHa [นามแฝง] (2561) **ความแตกต่างระหว่าง เว็บไซต์ กับ เว็บแอพฯ** [ออนไลน์] เข้าถึงได้จาก URL: https://1th.me/Txf8W (10 สิงหาคม 2563)

[12] admin [นามแฝง] (2556) **มารู้จักกับ Windows Application กันเถอะ** [ออนไลน์] เข้าถึงได้จากURL: https://1th.me/GhE4a (10 สิงหาคม 2563)

[13] SUNFOUNDER [นามแฝง] (2560) **4 Channel 5V Relay Module** [ออนไลน์] เข้าถึงได้

จากURL: http://wiki.sunfounder.cc/index.php?title=4\_Channel\_5V\_

Relay\_Module (10 สิงหาคม 2563)

[14] Benne de Bakker (2562) **How to control a character I2C LCD with Arduino** [ออนไลน์] เข้าถึงได้จาก URL: https://www.makerguides.com/character-i2c-lcd-arduino-tutorial/ (10 สิงหาคม 2563)

[15] MarcusCode [นามแฝง] (2558) **แนะนำภาษา C++** [ออนไลน์] เข้าถึงได้จาก

URL: https://1th.me/KCVtf (10 สิงหาคม 2563)

[16] MarcusCode [นามแฝง] (2559) **ภาษา Visual Basic** [ออนไลน์] เข้าถึงได้จาก

URL: http://marcuscode.com/lang/visual-basic (10 สิงหาคม 2563)

[17] สมหวัง อริสริยวงศ์ (2553) **ความรู้เกี่ยวกับความชื้นเบื้องต้น** [ออนไลน์] เข้าถึงได้จากURL:http://ir.swu.ac.th/jspui/bitstream/123456789/2626/2/Somwang\_A\_R420645.pdf (21 สิงหาคม 2563)

[18] kritsada arjchariyaphat (2561) **LoRA, LoRaWAN คืออะไร มารู้จักกันดีกว่า** [ออนไลน์] เข้าถึงได้จาก URL: https://1th.me/2OTK0 pdf (21 สิงหาคม 2563)

[19] thawatchait (ม.ป.ป.) **บทที่ 7 วงจรแปลงแอนะล็อกเป็นดิจิตอลและวงจรแปลงดิจิตอลเป็นแอนะล็อก (Analog to Digital Converter and Digital to Analog Converter Circuits)** [ออนไลน์] เข้าถึงได้จาก URL:

http://pws.npru.ac.th/thawatchait/data/files/Chapter%207\_ADC%20and%20DAC%20circuits%2001.pdf (21 สิงหาคม 2563)

[20] Sara Santos (2562) **ESP32 ADC – Read Analog Values with Arduino IDE** [ออนไลน์] เข้าถึงได้จาก URL: https://randomnerdtutorials.com/esp32-adc-analog-read-arduino-ide/ (22 สิงหาคม 2563)

[21] งานระบบ (2554) **เครื่องปั๊มน้ำ(Pump)คืออะไร** [ออนไลน์] เข้าถึงได้จาก

URL: http://xn--72c0bda7bwel.blogspot.com/2011/08/pump.html

(22 สิงหาคม 2563)

[22] (ม.ป.ป.) **ทฤษฎีพัดลม** [ออนไลน์] เข้าถึงได้จาก URL: https://1th.me/De4vq

(22 สิงหาคม 2563)

[23] Thaimicrotron.com [นามแฝง] (ม.ป.ป.) **การเชื่อมต่ออุปกรณ์แบบ I2C (I2C)** [ออนไลน์] เข้าถึงได้จาก URL: https://1th.me/g9dEB (22 สิงหาคม 2563)

[24] วิทูลย์ งามขำ (ม.ป.ป.) **PHP คืออะไร?** [ออนไลน์] เข้าถึงได้จาก URL: https://sites.google.com/a/chaiwit.ac.th/php/unit1/bi-khwam-ru-thi-1

(22 สิงหาคม 2563)

[25] mindphp [นามแฝง] (2560) **JavaScript คืออะไร จาวา สคริปต์ คือ ภาษาคอมพิวเตอร์สำหรับการเขียนโปรแกรมบนระบบอินเทอร์เน็ต** [ออนไลน์] เข้าถึงได้จาก URL: https://1th.me/J2fg0 (23 สิงหาคม 2563)

[26] Admin [นามแฝง] (2554) **ajax คืออะไร** [ออนไลน์] เข้าถึงได้จาก URL: http://jquery-thaigroup.blogspot.com/2011/10/ajax.html (23 สิงหาคม 2563)

[27] mindphp [นามแฝง] (2560) **HTML คืออะไร เอชทีเอ็มแอล ภาษาคอมพิวเตอร์ที่ใช้ในการสร้างเว็บเพจ ใช้เขียนโปรแกรม ย่อมาจากอะไร** [ออนไลน์] เข้าถึงได้จากURL: https://1th.me/7lVEz (23 สิงหาคม 2563)

[28] mindphp [นามแฝง] (2560) **CSS คืออะไร ซีเอสเอส คือ ภาษาที่ใช้ในการจัดรูปแบบเอกสาร HTML ให้มีความสวยงาม** [ออนไลน์] เข้าถึงได้จากURL: https://1th.me/sj6sd (23 สิงหาคม 2563)

[29] support [นามแฝง] (2562) **การติดตั้ง Ngrok ให้ Localhost สามารถ Online ได้ตลอดเวลา**[ออนไลน์] เข้าถึงได้จาก URL: https://www.hostpacific.com/how-to-install-nrok-on-localhost/ (23 สิงหาคม 2563)

[30] mindphp [นามแฝง] (2562) **Database คืออะไร ระบบฐานข้อมูล คือ การเก็บรวบรวมข้อมูลไว้ด้วยกัน อย่างมีระบบ** [ออนไลน์] เข้าถึงได้จากURL: https://1th.me/ep1d4 (23 สิงหาคม 2563)