**บทที่ 3**

**วิธีการดำเนินงาน**

ในบทนี้จะกล่าวถึงขั้นตอนและวิธีการดำเนินงานจัดทำโครงงานตลอดจนการออกแบบโครงสร้างและส่วนต่างๆ ของโครงงาน ซึ่งได้รับการศึกษาข้อมูลและทฤษฎีที่เกี่ยวข้องในบทที่ 2 จึงได้นำข้อมูลที่ได้มาศึกษาและวิเคราะห์ออกแบบเพื่อเลือกใช้วัสดุอุปกรณ์ที่จะนำมาทำโครงงาน ให้มีความถูกต้องเหมาะสมกับลักษณะงานและดำเนินการสร้างโครงงานให้มีประสิทธิภาพและเป็นไปตามวัตถุประสงค์และขอบเขตของโครงงาน

**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 โครงสร้างระบบ**

|  |
| --- |
| **รูปที่ 3.1** แสดงภาพโครงสร้างระบบ |

จากรูปที่ 3.1 แสดงภาพโครงสร้างระบบของโครงงานโรงเพาะเห็ดอัจฉริยะโดยใช้เทคโนโลยีการสื่อสารไร้สายระยะไกล โดยภาพรวมของระบบประกอบด้วยโรงเรือนเพาะเห็ดที่สามารถควบคุมสภาพแวดล้อมภายในโรงเรือนได้โดยใช้ไมโครคอนโทรลเลอร์ (ESP32LoRa) เป็นตัวควบคุมระบบและทำหน้าที่ในการสื่อสารกับพื้นที่ที่มีสัญญาณอินเทอร์เน็ตผ่านลอรา (LoRa)

พื้นที่ที่มีสัญญาณอินเทอร์เน็ตจะประกอบไปด้วยไมโครคอนโทรลเลอร์ (ESP32LoRa) ทำหน้าที่สื่อสารระหว่างพื้นที่ที่มีสัญญาณอินเทอร์เน็ตและโรงเรือนและเป็น Web Server ส่วน Raspberry Pi ทำหน้าที่เป็นฐานข้อมูลเพื่อให้สามารถเรียกดูค่าสภาพแวดล้อมย้อนหลังได้ และใช้โปรแกรม ngrok ในการทำ Port forwarding เพื่อให้สามารถใช้งานเว็บแอปพลิเคชัน (Web Application) ได้จากทุกที่ที่สามารถใช้งานอินเทอร์เน็ต

เว็บแอปพลิเคชัน (Web Application) สามารถมอนิเตอร์ค่าสภาพแวดล้อมภายในโรงเรือนและควบคุมระบบ โดยผู้จัดทำได้ใช้ภาษา HTML และ JavaScript ในการสร้างเว็บแอปพลิเคชัน (Web Application) จากนั้นนำเว็บแอปพลิเคชัน (Web Application) เก็บไว้ในหน่วยความจำ (Flash Memory) ของไมโครคอนโทรลเลอร์ (ESP32LoRa) และใช้ ESPAsyncWebServer library เพื่อให้ไมโครคอนโทรลเลอร์ (ESP32LoRa) สามารถทำหน้าที่เป็น Web Server ได้

วินโดว์แอปพลิเคชัน (Windows Application) สามารถมอนิเตอร์ค่าสภาพแวดล้อมภายในโรงเรือนและควบคุมระบบ โดยผู้จัดทำได้ใช้ภาษา Visual Basic ของ .NET Core ในการสร้างวินโดว์แอปพลิเคชัน (Windows Application) วินโดว์แอปพลิเคชัน (Windows Application) สามารถใช้งานได้กับไมโครคอนโทรลเลอร์ (ESP32LoRa) ทั้งสองตัวผ่านทาง Serial Port หากใช้งานวินโดว์แอปพลิเคชัน (Windows Application) กับไมโครคอนโทรลเลอร์ (ESP32LoRa) (Mushroom Node) เมื่อควบคุมการทำงาน เช่น กดปุ่มเปลี่ยนโหมดการทำงาน วินโดว์แอปพลิเคชัน (Windows Application) จะส่งข้อมูลผ่าน Serial Port มายังไมโครคอนโทรลเลอร์ (ESP32LoRa) จากนั้น ไมโครคอนโทรลเลอร์ (ESP32LoRa) จะนำข้อมูลที่ได้มาประมวลผลเพื่อควบคุมการทำงานของระบบ หากใช้งานกับวินโดว์แอปพลิเคชัน (Windows Application) กับ ไมโครคอนโทรลเลอร์ (ESP32LoRa) (STA Node) เมื่อควบคุมการทำงาน เช่น กดปุ่มเปลี่ยนโหมดการทำงาน วินโดว์แอปพลิเคชัน (Windows Application) จะส่งข้อมูลผ่าน Serial Port มายังไมโครคอนโทรลเลอร์ (ESP32LoRa) จากนั้น ไมโครคอนโทรลเลอร์ (ESP32LoRa) จะส่งข้อมูลที่ได้จาก Serial Port ผ่าน LoRa ไปยังยังไมโครคอนโทรลเลอร์ (ESP32LoRa) (Mushroom Node) จากนั้นจะนำข้อมูลที่ได้มาประมวลผล

ฐานข้อมูล (Database) ในโครงงานนี้ผู้จัดทำได้ใช้ Raspberry Pi เป็นฐานข้อมูล (Database) เพื่อให้สามารถดูค่าสภาพแวดล้อมย้อนหลังได้ ใช้ภาษา PHP ในการบันทึกข้อมูลลงในฐานข้อมูล (Database) การนำข้อมูลจากฐานข้อมูล (Database) มาแสดงในรูปแบบของตารางและกราฟที่สามารถเลือกดูข้อมูลได้ตามช่วงเวลาที่ต้องการ

การแจ้งเตือนไปยัง Line Application (Line Notify Service) ผู้จัดทำได้ใช้ Line Notify Service ในการแจ้งเตือนสถานะต่างๆ ไปยัง Line Application เช่น สาถานะ การบันทึกข้อมูล การเปลี่ยนสถานะการทำงานของอุปกรณ์ควบคุมสภาพแวดล้อม หรือการเปลี่ยนโหมดการทำงานของระบบ

**3.3 โรงเรือนเพาะเห็ด (Mushroom Farm)**

**3.3.1 การออกแบบโรงเรือนเพาะเห็ด**

|  |
| --- |
| **รูปที่ 3.2** แสดงภาพการออกแบบโรงเพาะเห็ด |

จากรูปที่ 3.2 แสดงภาพการออกแบบโรงเพาะเห็ด ในโครงงานนี้ผู้จัดทำได้ใช้โปรแกรม Google Sketchup ในการออกแบบโรงเรือนสำหรับ เพาะเห็ดขนาด (กว้าง x ยาว x สูง) 200x150x210 เซนติเมตร โดยได้ออกแบบให้เป็นโรงเรือนแบบปิด สามารถควบคุมสภาพแวดล้อมต่างๆภายในโรงเรือนได้เช่น อุณหภูมิ ความชื้น และแสง โครงสร้างของ โรงเรือนเพาะเห็ดเป็นเหล็กกล่องขนาด 1x1 นิ้ว เพื่อโรงเรือนมีความแข็งแรงและ สามารถใช้งานได้ในระยะยาว จากรูปจะเห็นว่ามีตัวเลขกำกับในแต่ละส่วนโดยมีรายละเอียดดังนี้ ส่วนที่ 1 เป็นส่วนของหัวพ่นหมอกใช้ในการควบคุมความชื้นในโรงเรือน ส่วนที่ 2 พัดลมระบายอากาศใช้ในการควบคุมอุณหภูมิและความชื้นภายในโรงเรือน ส่วนที่ 3 หลอดไฟใช้ในการควบคุมความสว่างในโรงเรือน ส่วนที่ 4 ชั้นสำหรับวางก้อนเห็ด ส่วนที่ 5 ประตูทางเข้าโรงเรือน ส่วนที่ 6 หมุดยกระดับโรงเรือนเพื่อไม่ให้โรงเรือนอยู่ติดกับพื้นจนเกินไปเพื่อป้องกันการเกิดสนิม

**3.3.2 การออกแบบฮาร์ดแวร์ระบบควบคุมสภาพแวดล้อมภายในโรงเรือนเพาะเห็ด**

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

|  |
| --- |
| **รูปที่ 3.3** แสดงภาพส่วนประกอบของฮาร์ดแวร์ระบบควบคุมสภาพแวดล้อมภายในโรงเรือน |

**3.3.2.1 การอ่านค่าอุณหภูมิและความชื้นจากเซ็นเซอร์ AM2315**

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

|  |
| --- |
| **รูปที่ 3.4** แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และ AM2315 |

การเชื่อมต่อระหว่าง ESP32LoRa และ AM2315

|  |  |  |
| --- | --- | --- |
| 5V | ต่อกับ | VCC |
| GND | ต่อกับ | GND |
| SDA(GPIO21) | ต่อกับ | SDA |
| SCL(GPIO22) | ต่อกับ | SCL |

|  |
| --- |
| **รูปที่ 3.5** แสดงภาพโปรแกรมที่ใช้อ่านค่าจากเซ็นเซอร์ AM2315 |

จากรูปที่ 3.5 เป็นโปรแกรมภาษา C++ ที่ใช้ในการอ่านค่าจากเซ็นเซอร์ AM2315 โดยมีการเรียกใช้ไลบรารี่ (Library) Wire.h และ Adafruit\_AM2315.h ต่อมาส่วนที่สำคัญที่สุดคือส่วนของการปลุกการทำงานของเซ็นเซอร์ดังรูปที่แสดงด้านบน หากไม่ปลุกการทำงานของเซ็นเซอร์เมื่อไม่มีการใช้งานเซ็นเซอร์เป็นเวลานานจะส่งผลให้ไม่สามารถใช้งานได้

**3.3.2.2 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 3.6** แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และ BH1750FVI |

การเชื่อมต่อระหว่าง ESP32LoRa และ BH1750FVI

|  |  |  |
| --- | --- | --- |
| 5V | ต่อกับ | VCC |
| GND | ต่อกับ | GND |
| SDA(GPIO21) | ต่อกับ | SDA |
| SCL(GPIO22) | ต่อกับ | SCL |

|  |
| --- |
| **รูปที่ 3.7** แสดงภาพโปรแกรมที่ใช้อ่านค่าจากเซ็นเซอร์ BH1750FVI |

จากรูปที่ 3.7 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 (ลักซ**ฺ**)

**3.3.2.3 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 3.8** แสดงภาพ Schematic ของการควบคุมพัดลมและปั๊มพ่นหมอก |

การเชื่อมต่อระหว่าง ESP32LoRa และ 4-Channel Relay Module

|  |  |  |
| --- | --- | --- |
| 5V | ต่อกับ | DC+ |
| GND | ต่อกับ | DC- |
| GPIO33 | ต่อกับ | IN4 |
| GPIO4 | ต่อกับ | IN3 |

|  |
| --- |
| **รูปที่ 3.9** แสดงภาพโปรแกรมส่วนที่ใช้ควบคุมการทำงานของ Relay |

จากรูปที่ 3.9 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

**3.3.2.4 การทำงานของปั๊มพ่นหมอกและพัดลมในโหมดการทำงานแบบอัตโนมัติ (AUTO)**

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

**ตารางที่ 3.2** 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 3.10** แสดงภาพโปรแกรมที่ใช้ควบคุมการทำงานของปั๊มพ่นหมอกและพัดลม |

จากรุปที่ 3.10 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

**3.3.2.5 การควบคุมความสว่างภายในโรงเรือนเพาะเห็ด**

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

|  |
| --- |
| **รูปที่ 3.11** แสดงภาพวงจร LED Driver |

จากรูปที่ 3.11 จะต้องคำนวณหาค่าของตัวต้านทานที่จะนำมาต่ออนุกรมกับหลอด LED เพื่อจำกัดกระแสตามความต้องการของหลอด LED โดยสามารถคำนวณได้จากสมการ โดยกำหนดให้ ,

จาก

จะได้

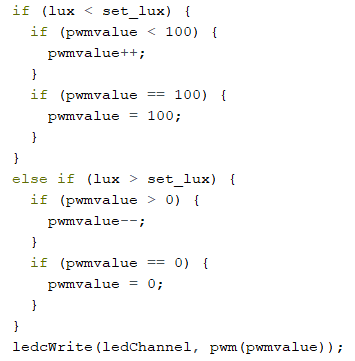
แทนค่า

ดังนั้นค่าของตัวต้านทานที่จะนำมาต่ออนุกรมกับหลอด LED มีค่าเท่ากับ 440 โอห์ม

|  |
| --- |
| **รูปที่ 3.12** แสดงภาพ Schematic ของการควบคุมความสว่างภายในโรงเรือนเพาะเห็ด |

การเชื่อมต่อระหว่าง ESP32LoRa และ L298N

|  |  |  |
| --- | --- | --- |
| GPIO25 | ต่อกับ | IN1 |
| GND | ต่อกับ | GND |
| GND | ต่อกับ | IN2 |



**รูปที่ 3.13** แสดงภาพโปรแกรมที่ใช้ในการควบคุมความสว่างของหลอดไฟ

จากรูปที่ 3.13 โปรแกรมจะทำการตรวจสอบว่าความสว่างต่ำกว่าหรือสูงกว่าค่าที่กำหนดหรือไม่ หากต่ำกว่าค่าที่กำหนดจะทำการเพิ่มค่า pwmvalue เพื่อเพิ่มความสว่างของหลอด LED หากสูงกว่าค่าที่กำหนดจะทำการลดค่า pwmvalue เพื่อลดความสว่างของหลอด LED

**3.3.2.6 การใช้จอแอลซี (LCD) แสดงผล**

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

|  |
| --- |
| **รูปที่ 3.14** แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และ จอแอลซีดี (LCD) |

การเชื่อมต่อระหว่าง ESP32LoRa และ จอแอลซีดี (LCD)

|  |  |  |
| --- | --- | --- |
| 5V | ต่อกับ | VCC |
| GND | ต่อกับ | GND |
| SDA (GPIO21) | ต่อกับ | SDA |
| SCL (GPIO22) | ต่อกับ | SCL |

|  |
| --- |
| **รูปที่ 3.15** แสดงภาพโปรแกรมทดสอบการทำงานของจอแอลซีดี (LCD) |

จากรูปที่ 3.15 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

**3.3.2.7 การอ่านค่าแรงดันเอาต์พุตจากวงจรตัวต้านทานตรวจสอบกระแส**

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

|  |
| --- |
| **รูปที่ 3.16** แสดงภาพ Schematic การเชื่อต่อระหว่าง ESP32LoRa และวงจรตัวต้านทาน  ตรวจสอบกระแส |

การเชื่อมต่อระหว่าง ESP32LoRa และวงจรตัวต้านทานตรวจสอบกระแส

|  |  |  |
| --- | --- | --- |
| GPIO25 | ต่อกับ | OUT (LM358P) PUMP |
| GPIO35 | ต่อกับ | OUT (LM358P) FAN |

|  |
| --- |
| **รูปที่ 3.17** แสดงภาพโปรแกรมทดสอบการอ่านค่าแรงดันเอาต์พุตจากวงจรตัวต้านทานตรวจสอบ  กระแส |

จากรูปที่ 3.17 แสดงภาพโปรแกรมที่ใช้ในการอ่านค่าแรงดันเอาต์พุตจากวงจรตัวต้านทานตรวจสอบกระแส โดยใช้ GPIO25 และ GPIO35 ของ ESP32LoRa ในการอ่านค่าแรงดันที่ได้จากเอาต์พุตของวงจร ใช้ ADC ความละในการอ่าน 12-bit จากนั้นนำค่าที่ได้มาคำนวลเป็นแรงดันที่สามารถอ่านได้จากเอาต์พุตของวงจร

**3.3 การออกแบบวงจรตัวต้านทานตรวจสอบกระแส**

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

|  |
| --- |
| **รูปที่ 3.18** แสดงภาพวงจรตัวต้านทานตรวจสอบกระแส |

จากรูปที่ 3.18 แสดงภาพของแสดงภาพวงจรตัวต้านทานตรวจสอบกระแส โดยจะประกอบด้วย LOAD คือ พัดลมระบายอากาศ หรือ ปั๊มพ่นหมอก RS คือ ตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) และส่วนของวงจรขยายแบบไม่กลับเฟส (Non-inverting amplifier) ใช้เพื่อขยายแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแส

**3.3.1 การ****คำนวณแรงดันเอาต์พุตของวงจรเมื่อใช้งานกับพัดลมระบายอากาศ**

ในโครงงานนี้ผู้จัดทำใช้พัดลมระบายอากาศขนาด 12 นิ้ว 12V 6.6A ในการควบคุมสภาพแวดล้อมภายในโรงเรือนเพาะเห็ด ดังนั้นสามารถคำนวณแรงดันเอาต์พุตของวงจรตัวต้านทานตรวจสอบกระแส ได้จากสมการ

จากสมการด้านบน คือแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) ดังนั้นจึงต้องทราบค่า ก่อน

|  |
| --- |
| **รูปที่ 3.19** แสดงภาพวงจรตัวต้านทานตรวจสอบกระแสเมื่อใช้งานกับพัดลมระบายอากาศ |

จารรูปที่ 3.19 เพื่อให้ไมโครคอนโทรลเลอร์สมารถแรงดันเอาต์พุตของวงจรได้ โดยสามารถหาอัตราการขยายของวงจรได้จากสมการ

จากสูตร

จะได้

อัตราการขยาย

ดังนั้น

อัตราการขยาย เท่า

สามารหาแรงดันตกคร่อมต้านทานตรวจสอบกระแส หรือ ได้จากกฎของโอห์มดังสมการ

*ดังนั้นหาแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแสจากรูปที่ 3.33 กำหนดให้ กระแส ความต้านทานของ*ต้านทานตรวจสอบกระแส

จากสูตร

*จะได้*

*แทนค่า*

*ดังนั้นแรงดันตกคร่องตัวต้านทานตรวจสอบกระแส มีค่าเท่ากับ ดังนั้นสามารถคำนวณแรงดันเอาต์พุตของวงจรเมื่อใช้งานกับพัดลมระบายอากาศได้จากสมการ*

*แทนค่า*

*ดังนั้นแรงดันเอาต์พุตของวงจรเมื่อใช้งานกับพัดลมระบายอากาศ* () *เท่ากับ สามารถคำนวณ**กำลัง*วัตต์ของตัวต้านทานตรวจสอบกระแส

*จากสูตร*

*จะได้*

*แทนค่า*

*ดังนั้นกำลัง*วัตต์ของตัวต้านทานตรวจสอบกระแส เท่ากับ 2.178

**3.3.2 การคำนวณแรงดันเอาต์พุตของวงจรเมื่อใช้งานกับปั๊มพ่นหมอก**

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

จากสมการด้านบน คือแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแส (Current Sense Resistors) ดังนั้นจึงต้องทราบค่า ก่อน

|  |
| --- |
| **รูปที่ 3.20** แสดงภาพวงจรตัวต้านทานตรวจสอบกระแสเมื่อใช้งานกับปั๊มพ่นหมอก |

จารรูปที่ 3.20 เพื่อให้ไมโครคอนโทรลเลอร์สมารถแรงดันเอาต์พุตของวงจรได้ โดยสามารถหาอัตราการขยายของวงจรได้จากสมการ

จากสูตร

จะได้

อัตราการขยาย

ดังนั้น

อัตราการขยาย เท่า

สามารหาแรงดันตกคร่อมต้านทานตรวจสอบกระแส หรือ ได้จากกฎของโอห์มดังสมการ

*ดังนั้นหาแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแสจากรูปที่ 3.4 กำหนดให้ กระแส ความต้านทานของ*ต้านทานตรวจสอบกระแส

จากสูตร

*จะได้*

*แทนค่า*

*ดังนั้นแรงดันตกคร่อง**ตัวต้านทานตรวจสอบกระแส มีค่าเท่ากับ ดังนั้นสามารถคำนวณแรงดันเอาต์พุตของวงจรเมื่อใช้งานกับปั๊มพ่นหมอกได้จากสมการ*

*แทนค่า*

*ดังนั้นแรงดันเอาต์พุตของวงจรเมื่อใช้งานกับปั๊มพ่นหมอก เท่ากับ* 3.1 *สามารถคำนวณ**กำลัง*วัตต์ของตัวต้านทานตรวจสอบกระแส

*จากสูตร*

*จะได้*

*แทนค่า*

*ดังนั้นกำลัง*วัตต์ของตัวต้านทานตรวจสอบกระแส เท่ากับ 0.2

**3.4 การสื่อสารระหว่างโรงเรือนและพื้นที่ที่มีสัญญาณอินเทอร์เน็ตโดยใช้ลอรา (LoRa Communication)**

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

|  |
| --- |
| **รูปที่ 3.21** แสดงภาพการสื่อสารแบบ Node-to-Node |

จากรูปที่ 3.21 Node แต่ละ Node จะมี IP Address เป็นของตัวเองเพื่อใช้ในการระบุตัวตน เช่น Mushroom Node มี IP Address คือ X1 และ STA Node มี IP Address คือ X0

**3.4.1 การออกแบบการสื่อสาร Lora ที่ Mushroom Node**

ในการออกแบบการสื่อสารผ่าน Lora ที่ Mushroom Node ผู้จัดทำจะขออธิบายรายละเอียดต่างๆ โดยใช้ Flowchart ประกอบการอธิบาย

|  |
| --- |
| **รูปที่ 3.22** แสดงภาพ Flowchart การทำงานของ Mushroom Node (1) |

จากรูปที่ 3.22 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 3.23

|  |
| --- |
| **รูปที่ 3.23** แสดงภาพรูปแบบของข้อมูล (Data Format) |

จากรูปที่ 3.23 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 3.24

|  |  |
| --- | --- |
|  |  |
| **รูปที่ 3.24** แสดงภาพฟังก์ชัน sendUpdateData() | |

จากรูปที่ 3.24 แสดงฟังก์ sendUpdateData() ที่ใช้ส่งข้อมูลเมื่อมีการเปลี่ยนแปลงโดยรายละเอียดมีการทำงานดังต่อไปนี้ ก่อนส่งข้อมูลจะตรวจสอบค่าแรงดันตกคร่อมตัวต้านทานตรวจสอบกระแสก่อนเพื่อให้ได้ค่าสถานะการทำงานของอุปกรณ์ที่ตรงกับค่าจริง จากนั้นจะส่งข้อมูลที่มีรูปแบบ (Data Format) เหมือนรูปที่ 3.23 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 ถ้าไม่มีข้อมูลที่ส่งมาก็จะกลับไปตรวจสอบการเปลี่ยนแปลงของข้อมูล

หากมีข้อมูลส่งมาจาก STA Node ก็จะทำการตรวจสอบข้อมูลโดยหา IP Address -ของตัวเอง (Mushroom Node IP Address) ว่าข้อมูลที่ส่งมาเป็นข้อมูลของตัวเองหรือไม่ ถ้าใช่ก็จะนำข้อมูลที่ได้มาประมวลผลต่อไปดังรูปที่ 3.25 เป็นโปรแกรมส่วนตรวจสอบข้อมูล

|  |
| --- |
| **รูปที่ 3.**25 แสดงภาพโปรแกรมส่วนที่ตรวจสอบข้อมูลที่ส่งมายัง Mushroom Node |
| **รูปที่ 3.26** แสดงภาพ Flowchart การทำงานของ Mushroom Node (2) |
| **รูปที่ 3.27** แสดงภาพ Flowchart การทำงานของ Mushroom Node (3) |

จากรูปที่ 3.26 และ รูปที่ 3.27 เมื่อได้รับข้อมูลที่ส่งมาจาก STA Node แล้วขั้นตอนต่อไปคือการนำข้อมูลมาประมวลผลโดยตัวอย่างของข้อมูลที่ถูกส่งมาจาก STA Node มีดังนี้ X1M, X1Y20, X1T30, X1H80, X1J85, X1P, X1F, หรือ X1R เป็นต้น มีรายละเอียดการทำงานในแต่ละข้อมูลดังนี้

|  |
| --- |
| **รูปที่ 3.28** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (1) |

จากรูปที่ 3.28 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 3.29** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (2) |

จากรูปที่ 3.29 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 3.30** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (3) |

จากรูปที่ 3.30 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 3.31** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (4) |

จากรูปที่ 3.31 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 3.32** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (5) |

จากรูปที่ 3.32 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 3.33** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (6) |

จากรูปที่ 3.33 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 3.34** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (7) |

จากรูปที่ 3.34 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 3.35** แสดงภาพโปรแกรมประมวลผลข้อมูลที่ได้รับจาก STA Node (8) |

จากรูปที่ 3.35 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

**3.4.2 การออกแบบการสื่อสาร Lora ที่ STA Node**

ในการออกแบบการสื่อสารผ่าน Lora ที่ STA Node ผู้จัดทำจะขออธิบายรายละเอียดต่างๆ โดยใช้ Flowchart ประกอบการอธิบาย

|  |
| --- |
| **รูปที่ 3.36** แสดงภาพ Flowchart การทำงานของ STA Node (1) |

จากรูปที่ 3.36 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 3.37** แสดงภาพโปรแกรมส่วนที่รับข้อมูลจาก Serial Port |

จากรูปที่ 3.37 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 3.38** แสดงภาพโปรแกรมส่วนที่ตรวจสอบข้อมูลที่ส่งมายัง STA Node |

|  |
| --- |
| **รูปที่ 3.39** 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 3.40

|  |
| --- |
| **รูปที่ 3.40** แสดงภาพรูปแบบของข้อมูล (Data Format) ที่ส่งมาจาก Mushroom Node |

จากรูปที่ 3.40 แสดงรูปแบบของข้อมูลที่ส่งมาจาก Mushroom Node เมื่อได้รับข้อมูลก็จะนำข้อมูลที่ได้มา substring() ดังรูปที่ 3.41

|  |  |
| --- | --- |
|  |  |
| **รูปที่ 3.41** แสดงภาพการใช้ฟังก์ชัน substring() กับข้อมูลที่ได้รับจาก Mushroom Node | |

จากรูปที่ 3.41 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 3.42

|  |
| --- |
| **รูปที่ 3.42** แสดงภาพโปรแกรมที่ใช้ในการส่ง Keyword ทุก 60 วินาที |

จากรูปที่ 3.42 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 3.43** แสดงภาพ Flowchart การทำงานของ STA Node (3) |

จากรุปที่ 3.43 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 3.44** แสดงภาพ Flowchart การทำงานของ STA Node (4) |

จากรูปที่ 3.44 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

**3.5 เว็บแอปพลิเคชัน (Web Application)**

**3.6 วินโดว์แอปพลิเคชัน (Windows Application)**

**3.7 ฐานข้อมูล (Database)**

**3.8 การแจ้งเตือนไปยัง Line Application (Line Notify Service)**