รายงาน

ระบบส่งข้อมูลเซนเซอร์แบบไร้สายสำหรับการเฝ้าระวังสภาพแวดล้อม

จัดทำโดย

นาย พิษณุ โพธิ์อยู่ 1660903517

นาย ศรัณย์วิชญ์ วังวรรณรัตน์ 1660901370

อาจาร์ยที่ปรึกษา

ผศ.ดร.สุพจน์ สุขโพธารมณ์

รายงานเล่มนี้เป็นส่วนหนึ่งของรายวิชา EL335 โครงงานวิศวกรรมศาสตร์

หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์และหุ่นยนต์

สำนึกวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยกรุงเทพ

ประจำภาคการศึกษาที่ 1 ปีการศึกษา 2568

ชื่อโครงการ : ระบบส่งข้อมูลเซนเซอร์แบบไร้สายสำหรับการเฝ้าระวังสภาพแวดล้อม
ชื่อ : นาย พิษณุ โพธิ์อยู่ รหัสประจำตัว 1660903517
: นาย ศรัณย์วิชญ์ วังวรรณรัตน์ รหัสประจำตัว 1660901370
สาขาวิชา : วิศวกรรมคอมพิวเตอร์และหุ่นยนต์
ที่ปรึกษาโครงงาน : ผศ.ดร.สุพจน์ สุขโพธารมณ์
ปีการศึกษา : 2568

บทคัดย่อ

โครงงานเรื่อง ระบบส่งข้อมูลเซนเซอร์แบบไร้สายสำหรับการเฝ้าระวังสภาพแวดล้อม มีวัตถุประสงค์เพื่อออกแบบและพัฒนาระบบตรวจวัดข้อมูลจากเซนเซอร์หลายชนิด เช่น เซนเซอร์อุณหภูมิและความชื้น (DHT11) เซนเซอร์วัดระดับน้ำ และเซนเซอร์อัลตราโซนิก (HC-SR04) เพื่อนำค่าที่ได้มาประมวลผลและส่งข้อมูลแบบไร้สายด้วยโมดูล nRF24L01 ระหว่างบอร์ด Arduino Nano สองตัว โดยใช้โมดูล AMS1117 3.3V เป็นตัวจ่ายไฟแรงดันคงที่ให้กับระบบสื่อสารไร้สาย จากนั้นข้อมูลทั้งหมดจะถูกส่งต่อเพื่อแสดงผลในรูปแบบ Dashboard แบบเรียลไทม์ ซึ่งผู้ใช้สามารถดูค่าของเซนเซอร์ได้ผ่านหน้าจอคอมพิวเตอร์หรืออุปกรณ์อื่น ๆ

ผลการดำเนินงานพบว่าระบบสามารถสื่อสารข้อมูลระหว่างโหนดส่งและโหนดรับได้อย่างถูกต้อง มีความเสถียรของสัญญาณ และสามารถแสดงผลค่าที่ตรวจวัดได้อย่างต่อเนื่อง โดยแสดงผลผ่าน Dashboard ที่ออกแบบให้เข้าใจง่าย เหมาะสำหรับนำไปประยุกต์ใช้ในงานเฝ้าระวังสิ่งแวดล้อม โรงเรือนเกษตรอัจฉริยะ หรือระบบตรวจติดตามภายในอาคาร

สารบัญ

เรื่อง	หน้า
บทคัดย่อ	ก
สารบัญ	ข
สารบัญ (ต่อ)	ค
สารบัญรูปภาพ	ง
บทที่1 บทนำ	
1.1 ที่มาและความสำคัญของโครงงาน	1
1.2 วัตถุประสงค์ของโครงงาน	1
1.3 ขอบเขตของโครงงาน	1
1.4 ประโยชน์ที่คาดว่าจะได้รับ	1
บทที่ 2 แนวคิดทฤษฎีที่เกี่ยวข้อง	
2.1 ไมโครคอนโทรลเลอร์ (Arduino Nano)	2
2.2 เซนเซอร์ (Sensors)	2
2.3 โมดูลสื่อสารไร้สาย nRF24L01	3
2.4 ทฤษฎีระบบ Dashboard	3
2.5 หลักการและแนวคิดอื่น ๆ	3
บทที่ 3 การออกแบบและขั้นตอนการดำเนินงาน	
3.1 ภาพรวมของระบบ ระบบแบ่งเป็น 3 ส่วน	4
3.2 ฮาร์ดแวร์ที่ใช้	4
3.3 การเขียนโปรแกรม	4
3.4 การทดสอบระบบ	5
บทที่ 4 ผลการดำเนินงาน	
4.1 ผลการออกแบบระบบ	6
4.1.1 Test Case 1 – ทดสอบเซนเซอร์ DHT11	6
4.1.2 Test Case 2 – ทดสอบเซนเซอร์ HC-SR04	6
4.1.3 Test Case 3 – ทดสอบเซนเซอร์วัดระดับน้ำ (Water Level Sensor)	6
4.1.4 Test Case 4 – ทดสอบการส่งข้อมูลผ่าน NRF24L01	6
4.1.5 Test Case 5 – ทดสอบการแสดงผล Dashboard	7
4.2 ผลการทดสอบการทำงาน	7

สารบัญ (ต่อ)

4.3 สรุปผลการทำงาน 7

บทที่ 5 ข้อเสนอแนะ ผลการดำเนินงาน

5.1 ระยะทางการสื่อสารของ NRF24L01 8

5.2.ความแม่นยำของเซนเซอร์บางตัว 8

5.3 การออกแบบ PCB ให้รองรับสัญญาณ RF ได้ดีขึ้น 8

5.4. ระบบ Dashboard ควรมีระบบ Auto Refresh ที่เหมาะสม 8

บรรณานุกรม 9

ภารผนวก 10

ภารผนวก ก (ข้อมูลทั้งหมด) 11

สารบัญ รูปภาพ

รูปที่

3.1 Receiver 4

3.2 Transmitter 4

3.4 Dash Board 4

3.5 Dash Board 5

4.1 ค่าที่อ่านจากSensor 7

บทที่ 1

บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

โครงงานนี้มีวัตถุประสงค์เพื่อให้นักศึกษาสามารถออกแบบระบบที่สามารถ ตรวจวัดค่าจากเซนเซอร์หลายชนิด และส่งข้อมูลแบบไร้สายไปยังสถานีแม่ข่าย (Base Station) เพื่อแสดงผลข้อมูลบน Dashboard แบบเรียลไทม์ โดยใช้ไมโครคอนโทรลเลอร์ Arduino Nano และโมดูลสื่อสารไร้สาย nRF24L01 เป็นองค์ประกอบหลัก พร้อมทั้งออกแบบแผงวงจรพิมพ์ (PCB) เพื่อให้ระบบมีความเป็นมืออาชีพและพร้อมใช้งาน

การพัฒนาโครงงานนี้ช่วยให้นักศึกษาได้ฝึกกระบวนการทางวิศวกรรม ทั้งการออกแบบวงจรไฟฟ้า, การเขียนโปรแกรมไมโครคอนโทรลเลอร์, การสื่อสารไร้สาย และการสร้าง Dashboard ซึ่งเป็นทักษะสำคัญในสายงานด้านวิศวกรรมคอมพิวเตอร์และหุ่นยนต์

1.2 วัตถุประสงค์ของโครงงาน

1. เพื่อออกแบบและสร้างแผงวงจรพิมพ์ (PCB) สำหรับไมโครคอนโทรลเลอร์และเซนเซอร์
2. เพื่อเขียนโปรแกรมบน Arduino Nano สำหรับอ่านค่าจากเซนเซอร์ 3 ชนิด ได้แก่ DHT11, HC-SR04 และ Water Level Sensor
3. เพื่อพัฒนาโมดูลสื่อสารไร้สายด้วย nRF24L01 สำหรับรับ–ส่งข้อมูลระหว่างสถานีตรวจวัด (Sensor Node) และสถานีแม่ข่าย (Base Station)
4. เพื่อสร้าง Dashboard แสดงผลข้อมูลจากเซนเซอร์แบบเรียลไทม์บนคอมพิวเตอร์

1.3 ขอบเขตของโครงงาน

1. ระบบสามารถอ่านค่าและส่งข้อมูลจากเซนเซอร์ 3 ชนิด ได้แก่ DHT11, HC-SR04 และ Water Level Sensor
2. ใช้บอร์ด Arduino Nano เป็นไมโครคอนโทรลเลอร์ทั้งตัวส่งและตัวรับ
3. ส่งข้อมูลแบบไร้สายผ่าน NRF24L01 โดย Sensor Node จะส่งข้อมูลไปยัง Base Station ทุก ๆ 2-5 วินาที
4. Base Station แสดงข้อมูลบน Dashboard แบบเรียลไทม์ แต่ไม่รวมถึงการวิเคราะห์ข้อมูลขั้นสูงหรือการแจ้งเตือนอัตโนมัติ
5. ระบบออกแบบเพื่อใช้งานใน สภาพแวดล้อมภายในอาคาร หรือพื้นที่จำกัด ไม่ได้ทดสอบในพื้นที่เปิดขนาดใหญ่

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้ต้นแบบระบบเฝ้าระวังและแสดงผลข้อมูลจากเซนเซอร์แบบไร้สายที่สามารถนำไปต่อยอดได้
2. นักศึกษาได้รับประสบการณ์จริงในการออกแบบวงจรอิเล็กทรอนิกส์และแผง PCB
3. พัฒนาความเข้าใจเกี่ยวกับการสื่อสารไร้สายและการส่งข้อมูลแบบ IoT
4. ได้เรียนรู้การพัฒนา Dashboard สำหรับแสดงผลข้อมูลแบบเรียลไทม์

บทที่ 2

แนวคิดทฤษฎีที่เกี่ยวข้อง

2.1 ไมโครคอนโทรลเลอร์ (Arduino Nano)

Arduino Nano เป็นไมโครคอนโทรลเลอร์ที่ใช้ชิป ATmega328P มีขนาดเล็ก เหมาะสำหรับงานต้นแบบ สามารถเชื่อมต่อกับเซนเซอร์ได้หลากหลาย ผ่านพอร์ตดิจิทัลและแอนะล็อก มีการเขียนโปรแกรมผ่าน Arduino IDE ด้วยภาษา C/C++ แบบง่ายต่อการใช้งาน และส่วนประกอบหลัก

ใช้ ATmega328P เป็นหน่วยประมวลผลหลัก

แรงดันทำงาน 5V

พอร์ต I/O 14 Digital I/O, 8 Analog Input

การเชื่อมต่อ USB สำหรับโปรแกรมและ Serial Communication

ความสามารถสำคัญ อ่านค่าเซนเซอร์, ควบคุมแอคชูเอเตอร์, สื่อสารแบบไร้สายผ่าน SPI/I2C

Arduino Nano เป็นหัวใจหลักของ Sensor Node และ Base Station ในโปรเจคนี้ ใช้ประมวลผลและส่งต่อข้อมูลเซนเซอร์แบบเรียลไทม์

2.2 เซนเซอร์ (Sensors) ระบบนี้ใช้เซนเซอร์ 3 ชนิดหลัก ได้แก่

DHT11

1. ใช้วัด อุณหภูมิและความชื้น
2. ใช้เซนเซอร์ความต้านทานและเทอร์มิสเตอร์ในตัว
3. การเชื่อมต่อ Digital I/O
4. Output ค่าอุณหภูมิ (°C) และความชื้น (%RH)

HC-SR04

1. เซนเซอร์ วัดระยะทางด้วยคลื่นอัลตราโซนิก
2. ประกอบด้วย Trig Pin (ส่งสัญญาณ) และ Echo Pin (รับสัญญาณ)
3. การทำงาน วัดเวลาที่คลื่นเสียงสะท้อนกลับ ⟶ คำนวณระยะทาง

Water Level Sensor

1. วัดระดับน้ำในถังหรือภาชนะ
2. ประเภท แบบ Resistive หรือ Capacitive
3. Output Analog Voltage หรือ Digital Signal

2.3 โมดูลสื่อสารไร้สาย nRF24L01

nRF24L01 เป็นโมดูลสื่อสารไร้สายย่านความถี่ 2.4GHz ใช้มาตรฐาน SPI ในการเชื่อมต่อกับไมโครคอนโทรลเลอร์ รองรับการสื่อสารแบบ point-to-point หรือ multi-node มีอัตราการส่งข้อมูลสูงสุด 2Mbps และระยะทางการส่งได้ประมาณ 100 เมตรในพื้นที่โล่ง

2.4 ทฤษฎีระบบ Dashboard

Dashboard คืออินเทอร์เฟซที่ใช้แสดงผลข้อมูลจากระบบตรวจวัดในรูปแบบกราฟ ตัวเลข หรือเกจแบบเรียลไทม์ โดยเชื่อมต่อกับไมโครคอนโทรลเลอร์ผ่าน Serial Port หรือโปรโตคอลการสื่อสารอื่น ๆ เพื่อให้ผู้ใช้สามารถมอนิเตอร์ข้อมูลได้ง่ายและสะดวก รายละเอียดในการออกแบบ

1. แสดงค่าในรูปแบบตัวเลข, กราฟเส้น, หรือเกจวัด
2. ใช้เทคโนโลยี Serial Communication จาก Arduino ⟶ คอมพิวเตอร์
3. ประโยชน์: ทำให้ผู้ใช้เข้าใจสถานะของสภาพแวดล้อมได้ทันที
4. สามารถต่อยอดเพื่อเพิ่มฟังก์ชันแจ้งเตือนหรือเก็บข้อมูลย้อนหลัง

2.5 หลักการและแนวคิดอื่น ๆ

1. การประมวลผลสัญญาณ (Signal Processing): แปลงค่า Analog ⟶ Digital เพื่อให้ Arduino สามารถประมวลผล
2. SPI Communication: การสื่อสารระหว่าง Arduino และ NRF24L01
3. หลักการออกแบบ PCB: วาง Layout ของอุปกรณ์อย่างเหมาะสม ลดเสียงรบกวนและการรบกวนสัญญาณ

บทที่ 3

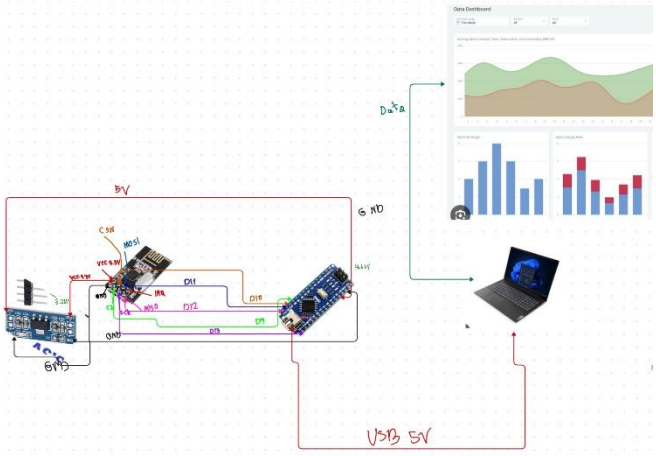
การออกแบบและขั้นตอนการดำเนินงาน

3.1 ภาพรวมของระบบ ระบบแบ่งเป็น 3 ส่วน ได้แก่

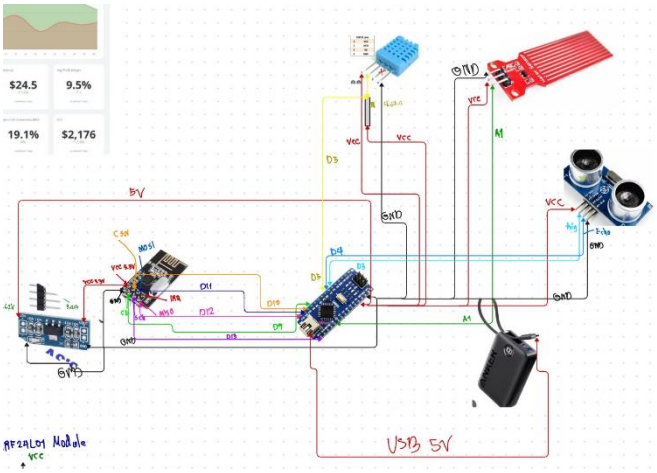
1. Sensor Node: อ่านค่าจากเซนเซอร์และส่งข้อมูลผ่าน nRF24L01
2. Base Station: รับข้อมูลจาก Sensor Node และส่งต่อไปยังคอมพิวเตอร์เพื่อแสดงผล
3. PCB ในการนำอุปกรณ์ทั้งหมดมาไว้ในที่เดียวเพื่อทำงานตามต้องการ

3.2 ฮาร์ดแวร์ที่ใช้

4. Arduino Nano จำนวน 2 ตัว
5. nRF24L01 จำนวน 2 ตัว
6. AMS1117-3.3V จำนวน 2 ตัว
7. DHT11 Sensor
8. Ultrasonic Sensor HC-SR04
9. Water Level Sensor
10. คอมพิวเตอร์สำหรับ Dashboard



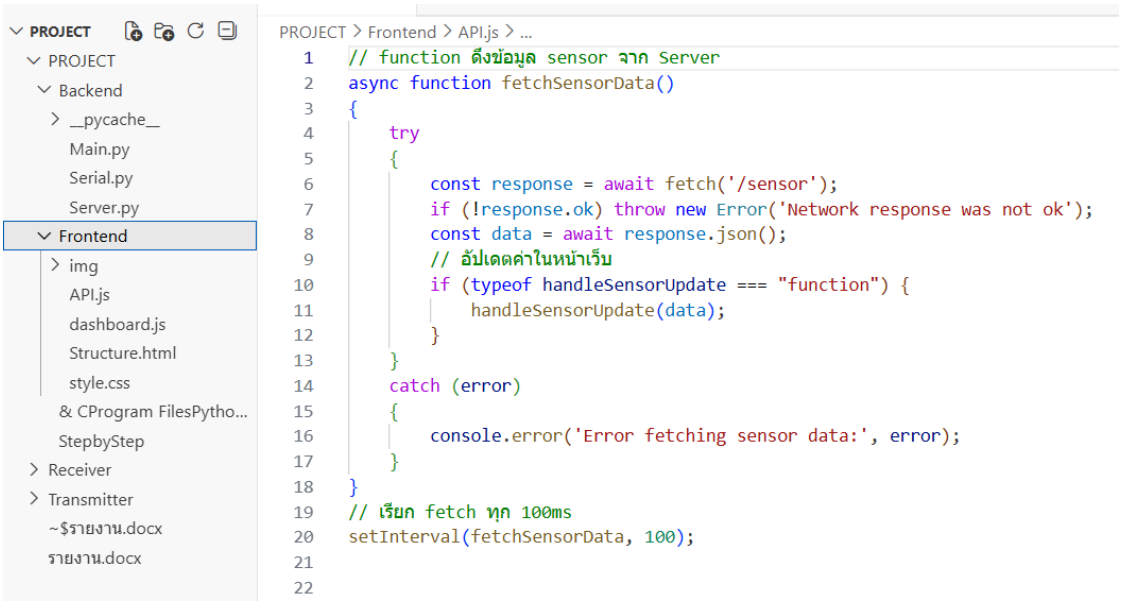
ภาพที่3.1 Receiver



ภาพที่3.2 Transmitter

3.3 การเขียนโปรแกรม

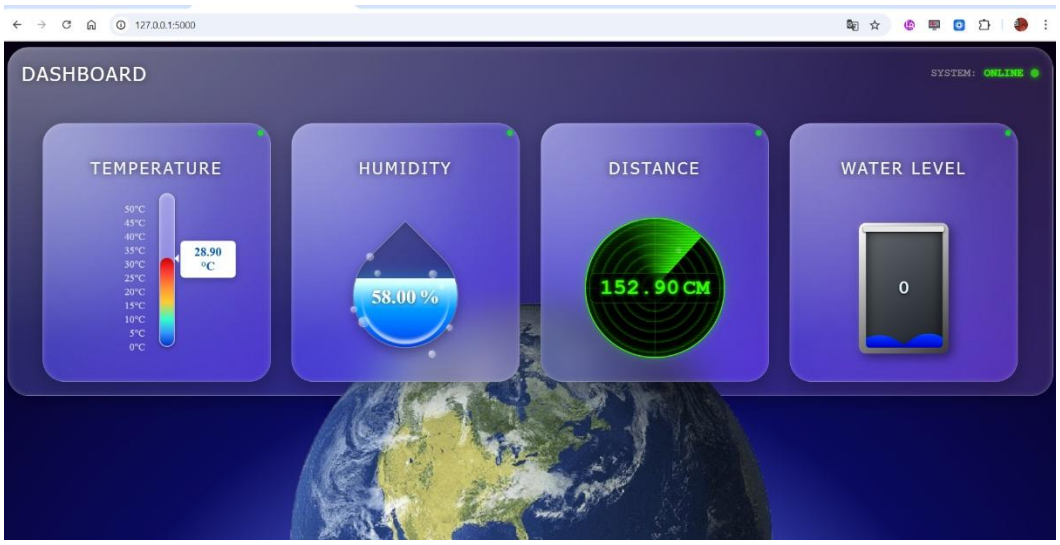
1. ฝั่ง Sensor Node: อ่านค่าจากเซนเซอร์ทั้ง 3 ตัว และส่งข้อมูลทุก 2 วินาทีผ่าน nRF24L01
2. ฝั่ง Base Station: รับข้อมูลจาก nRF24L01 แล้วส่งต่อผ่าน Serial ไปยังคอมพิวเตอร์
3. ฝั่ง Dashboard: ใช้ Server Flask , HTML ,CSS ,JS ในการแสดงผลและสร้างให้ผู้ใช้งานได้เห็น



ภาพที่3.4 Dash Board

3.4 การทดสอบระบบ

1. ทดสอบการอ่านค่าจากเซนเซอร์แต่ละตัว
2. ทดสอบการรับ–ส่งข้อมูลไร้สายระหว่าง Arduino ทั้งสอง
3. ตรวจสอบการแสดงผลบน Dashboard
4. บันทึกผลและวิเคราะห์ความแม่นยำของข้อมูล



ภาพที่3.5 Dash Board

บทที่ 4

ผลการดำเนินงาน

4.1 ผลการออกแบบระบบ

เพื่อประเมินประสิทธิภาพของระบบ ได้กำหนดการทดสอบออกเป็น 3 ส่วนใหญ่ ได้แก่

การทดสอบความถูกต้องของเซนเซอร์

การทดสอบความเสถียรของการส่งข้อมูลผ่าน NRF24L01

การทดสอบการแสดงผลบน Dashboard

4.1.1 Test Case 1 – ทดสอบเซนเซอร์ DHT11

วัตถุประสงค์: ตรวจสอบว่าเซนเซอร์อ่านอุณหภูมิและความชื้นได้ถูกต้อง วิธีทดสอบ:

วาง DHT11 ในห้องที่มีอุณหภูมิและความชื้นคงที่

เปรียบเทียบค่ากับเครื่องวัดมาตรฐาน

เกณฑ์ผ่าน: ค่าคลาดเคลื่อนไม่เกิน ±2°C และ ±5%RH

4.1.2 Test Case 2 – ทดสอบเซนเซอร์ HC-SR04

วัตถุประสงค์: วัดความแม่นยำของระยะทาง วิธีทดสอบ:

ใช้ไม้บรรทัดวัดระยะตั้งแต่ 5–100 cm

เปรียบเทียบค่าที่ HC-SR04 วัดได้

เกณฑ์ผ่าน: ค่าคลาดเคลื่อนไม่เกิน ±3 cm

4.1.3 Test Case 3 – ทดสอบเซนเซอร์วัดระดับน้ำ (Water Level Sensor)

วัตถุประสงค์: วัดระดับน้ำที่ต่างกัน 3 ระดับ วิธีทดสอบ:

ใส่น้ำในภาชนะระดับ ต่ำ–กลาง–สูง

บันทึกค่าที่ sensor ส่งออก

เกณฑ์ผ่าน: ค่าต้องเปลี่ยนตามระดับน้ำ “ชัดเจนและต่อเนื่อง”

4.1.4 Test Case 4 – ทดสอบการส่งข้อมูลผ่าน NRF24L01

วัตถุประสงค์: ทดสอบความเสถียรในการส่งข้อมูล วิธีทดสอบ:

ส่งข้อมูล 100 ครั้งจาก Sensor Node ⟶ Base Station

เก็บสถิติ Packet Loss

เกณฑ์ผ่าน: Packet Loss ≤ 5%

4.1.5 Test Case 5 – ทดสอบการแสดงผล Dashboard

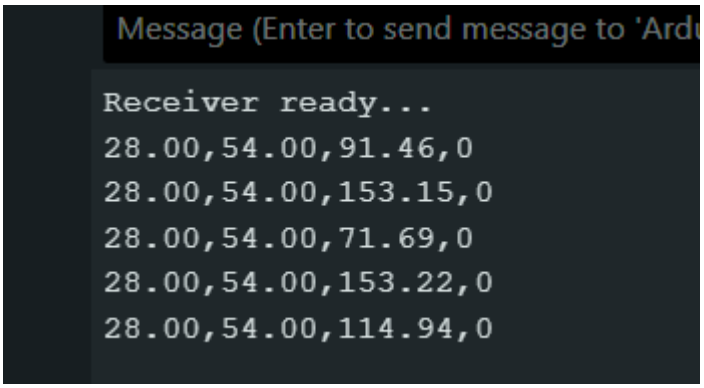
วัตถุประสงค์: ตรวจสอบว่า Dashboard แสดงผลเรียลไทม์ตามข้อมูลที่ได้รับ วิธีทดสอบ:

ปรับสภาพแวดล้อม เช่น ใช้ลมเป่าที่ DHT11, เอาของไปขวาง HC-SR04

ดูว่ากราฟและค่าบน Dashboard อัปเดตทันทีหรือไม่

4.2 ผลการทดสอบการทำงาน

1. ระบบสามารถอ่านค่าจากเซนเซอร์ DHT11, HC-SR04 และ Water Level ได้ถูกต้อง
2. การสื่อสารระหว่าง Arduino ทั้งสองผ่าน nRF24L01 มีความเสถียรในระยะไม่เกิน 15 เมตรในอาคาร
3. Dashboard สามารถแสดงผลค่าทุก 2 วินาทีในรูปแบบกราฟเส้นและตัวเลขแบบเรียลไทม์



ภาพที่4.1 ค่าที่อ่านจากSensor

4.3 สรุปผลการทำงาน

ระบบเฝ้าระวังและแสดงผลข้อมูลเซนเซอร์ไร้สายที่พัฒนาขึ้นสามารถทำงานได้ตามวัตถุประสงค์ที่ตั้งไว้ทุกข้อ ทั้งในส่วนของฮาร์ดแวร์ ซอฟต์แวร์ และการแสดงผลข้อมูล โดยสามารถนำไปประยุกต์ใช้ในงานตรวจวัดสิ่งแวดล้อมหรือระบบอัตโนมัติในอนาคตได้

บทที่ 5

ข้อเสนอแนะ ผลการดำเนินงาน

จากการพัฒนา “ระบบส่งข้อมูลเซนเซอร์แบบไร้สายสำหรับการเฝ้าระวังสภาพแวดล้อม” พบประเด็นที่ควรปรับปรุงเพื่อเพิ่มความเสถียรและประสิทธิภาพของระบบ ดังนี้:

5.1 ระยะทางการสื่อสารของ NRF24L01

แม้โมดูล NRF24L01 จะสื่อสารได้ดีในระยะใกล้–ปานกลาง แต่เมื่อต้องใช้งานในพื้นที่ที่มีสิ่งกีดขวางหรือสัญญาณรบกวน อาจทำให้สัญญาณลดลง จึงควรเพิ่ม

1. ชุดเสาอากาศแบบ PA/LNA
2. การกำหนด Channel ที่เหมาะสม
3. การทำ Retransmission หรือ Auto-ACK

5.2.ความแม่นยำของเซนเซอร์บางตัว

1. เซนเซอร์ DHT11 มีข้อจำกัดด้านความละเอียดและความเร็ว
2. เซนเซอร์ Ultrasonic อาจมีค่าคลาดเคลื่อนเมื่อวัดวัตถุมีผิวดูดซับเสียง แนะนำให้สอบเทียบค่าก่อนใช้งานจริง (Calibration)

5.3 การออกแบบ PCB ให้รองรับสัญญาณ RF ได้ดีขึ้น

1. ควรแยก Ground ของโมดูล RF ให้เหมาะสม
2. เพิ่ม Capacitor ใกล้ขา Vcc/GND เพื่อ ลด noise
3. วางตำแหน่ง NRF24L01 ให้ห่างจาก Switching Power หรือเส้นทางกระแสสูง

5.4. ระบบ Dashboard ควรมีระบบ Auto Refresh ที่เหมาะสม

หาก Refresh ถี่เกินไปอาจทำให้ข้อมูลกระตุกหรือโหลดสูง ควรกำหนด

1. Interval 1–3 วินาทีต่อการอัปเดต
2. ตรวจสอบว่าไม่มีการร้องขอข้อมูลซ้ำซ้อน

ควรมีทดสอบการทำงานในสถานการณ์จริง (Field Test)

เพื่อประเมินความแม่นยำ ความเสถียร และปัญหาที่เกิดขึ้นจริง เช่น

1. แสงแดด
2. ความชื้นสูง
3. สิ่งกีดขวาง
4. สัญญาณรบกวนจากอุปกรณ์อื่น

บรรณานุกรม

Datasheet ของ nRF24L01 2.4 GHz

https://cdn.sparkfun.com/datasheets/Wireless/Nordic/nRF24L01_Product_Specification_v2_0.pdf

Datasheet ของ AMS117 3.3V Module

electrothinks.com/2023/05/ams1117-dc-3-3v-regulator-module.html

https://mm.digikey.com/Volume0/opasdata/d220001/medias/docus/5011/AMS1117.pdf

Datasheet ของ Arduino Nano

https://www.es.co.th/schemetic/pdf/armb-0022.pdf

Datasheet ของ Water MJT Level

https://app.box.com/s/4m48dgi9wc84rrmsaiua

Datasheet ของ DHT11 Module

https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf?srsltid=AfmBOopW1dir9dXQGL9NNO3loYjgtTu_SPINT06-RwVUUEaQeNNESFN1

Datasheet ของ DHT11 Module

https://www.handsontec.com/dataspecs/HC-SR04-Ultrasonic.pdf

ภาคผนวก

ภารผนวก ก

(ข้อมูลทั้งหมด)

Code Arduino Transmitter

ไฟล์ Backend

https://drive.google.com/drive/folders/1JOEieTMoPAzK7erw_s12dLEsj6aZu7jT?usp=drive_link

ไฟล์ Frontend

https://drive.google.com/drive/folders/1TA-7EWhBy3nICzed1bUJ4Z8xYUa3HFin?usp=drive_link

```
#include <SPI.h> // SPI สำหรับสื่อสารกับ nRF24L01
#include <nRF24L01.h> // สำหรับควบคุมโมดูล nRF24L01
#include <RF24.h> // RF24 ที่ช่วยให้ใช้งาน nRF24L01 ง่ายขึ้น
#include "DHT.h" // สำหรับอ่านค่า DHT11 / DHT22 (Temperature + Humidity)

#define DHTPIN 5          // กำหนดขา DATA ของ DHT11
#define DHTTYPE DHT11  // กำหนดชนิด sensor DHT11

DHT dht(DHTPIN, DHTTYPE); // สร้าง object dht เรียกใช้ฟังก์ชันอ่านค่า(Temperature +
Humidity)

// CE = 9, CSN = 10 ของ nRF24L01
RF24 radio(9, 10);
const byte address[6] = "00001"; // address ของข้อมูล Transmitter และ Receiver
ติดต่อกันได้

// Sensor pins
const int waterPin = A1;      // Water Sensor
const int trigPin = 3;        // HC-SR04
const int echoPin = 4;        // HC-SR04

// Structure ส่งข้อมูล sensor 3 ตัว ผ่าน nRF24L01 ได้ในครั้งเดียว
struct SensorData
{
  float temperatureC;
  float humidity;
  float distanceCM;
  int waterLevel;
};

void setup()
{
  Serial.begin(115200);
  Serial.println("Transmitter ready...");
  dht.begin(); // เริ่มการสื่อสารกับ DHT11
  radio.begin(); //เริ่ม nRF24L01
  radio.openWritingPipe(address);  // ตั้ง address ปลายทาง
  radio.setPALevel(RF24_PA_LOW);   // กำลังส่ง (LOW/MIN/HIGH/MAX)
  radio.stopListening();           // หยุดฟังเพื่อเข้าสู่โหมดส่ง

  pinMode(trigPin, OUTPUT); // HC-SR04 pins 3
  pinMode(echoPin, INPUT); // HC-SR04 pins 4
}

// HC-SR04 คำนวณ calculate distance
float readDistanceCM()
```

```
{
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);  // pulse 2 µs จาก trigPin
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10); // pulse 10 µs จาก trigPin
  digitalWrite(trigPin, LOW);

  long duration = pulseIn(echoPin, HIGH); //  วัดเวลาที่ echo กลับ
  float distance = duration * 0.034 / 2; // แปลงเป็น cm
  return distance;
}

void loop()
{
  SensorData data;

 // อ่าน DHT22
  data.temperatureC = dht.readTemperature(); // °C
  data.humidity = dht.readHumidity();        // % RH

  if (isnan(data.temperatureC) || isnan(data.humidity))  //ไม่ให้เกิดค่า NaN
  {
    Serial.println("DHT read error!");
    delay(2000);
    return;
  }

 // อ่าน HC-SR04
  data.distanceCM = readDistanceCM();

  int total = 0;  // “ตัด noise” ของ Water Sensor
  for (int i = 0; i < 5; i++)
  {
    total += analogRead(waterPin);
    delay(10);
  }
  data.waterLevel = total / 5;

 // อ่าน Water Sensor
  //ata.waterLevel = analogRead(waterPin); // 0–1023

 // ส่งข้อมูล
  bool YES = radio.write(&data, sizeof(data)); // ส่งข้อมูลทั้งหมดใน data ผ่าน
nRF24L01
  if(YES) // ส่งสำเร็จ
  {
```

```
    Serial.print("Sent Temp: ");
    Serial.print(data.temperatureC);
    Serial.println(" °C");

    Serial.print("Sent Humidity: ");
    Serial.print(data.humidity);
    Serial.println(" %");

    Serial.print("Sent Distance: ");
    Serial.print(data.distanceCM);
    Serial.println(" cm");

    Serial.print("Sent Water: ");
    Serial.print(data.waterLevel);
    Serial.println(" Level");

    Serial.println(" ");
  }
  else
  {
    Serial.println("Send failed!");  // ไม่สำเร็จ แสดง
  }
  delay(1000);
}
```

Code Arduino Receiver

```
#include <SPI.h> // SPI สำหรับสื่อสารกับ nRF24L01
#include <nRF24L01.h> // สำหรับควบคุมโมดูล nRF24L01
#include <RF24.h> // RF24 ที่ช่วยให้ใช้งาน nRF24L01 ง่ายขึ้น

// CE = 9, CSN = 10
RF24 radio(9, 10);
const byte address[6] = "00001"; // address ของข้อมูล Transmitter และ Receiver
ติดต่อกันได้

// Structure ตรงกับตัวส่งข้อมูล sensor 3 ตัว เพื่อรับค่าได้ถูกต้อง ผ่าน nRF24L01 ได้ในครั้งเดียว
struct SensorData
{
  float temperatureC;
  float humidity;
  float distanceCM;
  int waterLevel;
};

void setup()
{
  Serial.begin(115200);
  Serial.println("Receiver ready...");
  radio.begin();
  radio.openReadingPipe(0, address); // ตั้ง address pipe 0 เดียวกับตัวส่ง
  radio.setPALevel(RF24_PA_LOW);
  radio.startListening();             // เปิดโหมดฟังข้อมูลจาก Transmitter
}

void loop()
{
  if (radio.available())  // เช็คว่ามีข้อมูลจาก Transmitter หรือไม่
  {
    SensorData data;
    radio.read(&data, sizeof(data)); //อ่านข้อมูลทั้งหมดลงใน data

  // เพิ่ม/แก้ไขด้านล่าง หลังอ่านค่า sensor เรียบร้อย
    Serial.print(data.temperatureC);
    Serial.print(",");
    Serial.print(data.humidity);
    Serial.print(",");
    Serial.print(data.distanceCM);
    Serial.print(",");
    Serial.println(data.waterLevel);
  }
}
```