ระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

นายกำพล นนแก้ว

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมเมคคาทรอนิกส์
มหาวิทยาลัยเทคโนโลยีสุรนารี
ปีการศึกษา 2564

TRACKING AND DATA LOGGER SYSTEM FOR ICE TRUCK

KAMPON NONKEAW

A Thesis Submitted in Partial Fulfillment of the Requirements for the
Degree of Master of Engineering in Mechatronics Engineering
Suranaree University of Technology
Academic Year 2021

ระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

มหาวิทยาลัยเทคโนโลยีสุรนารี อนุมัติให้นับวิทยานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญามหาบัณฑิต

คณะกรรมการสอบวิทยานิพนธ์

(ผศ. ดร.โศรฎา แข็งการ)
ประธานกรรมการ

(รศ. ดร.จิระพล ศรีเสริฐผล)
กรรมการ (อาจารย์ที่ปรึกษาวิทยานิพนธ์)

(รศ. ดร.บัณฑิต กฤตาคม)
กรรมการ

(รศ. ดร.ฉัตรชัย โชติษฐยางกูร)
รองอธิการบดีฝ่ายวิชาการและประกันคุณภาพ

(รศ. ดร.พรศิริ จงกล)
คณบดีสำนักวิชาวิศวกรรมศาสตร์

กำพล นนแก้ว : ระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง (TRACKING AND DATA LOGGER SYSTEM FOR ICE TRUCK) อาจารย์ที่ปรึกษา : รองศาสตราจารย์ ดร. จิระพล ศรีเสริฐผล, 118 หน้า.

คำสำคัญ: ระบบติดตามจีพีเอส/รถตู้บรรทุกน้ำแข็ง/ไมโครคอนโทรลเลอร์/อินเทอร์เน็ตของสรรพสิ่ง

ปัจจุบันผู้ประกอบการธุรกิจขนส่งน้ำแข็งมักประสบปัญหาเกี่ยวกับพฤติกรรมของพนักงานขับรถตู้บรรทุกน้ำแข็งในเรื่องของการขับรถออกนอกเส้นทางที่กำหนด การไปรับจ้างขนน้ำแข็งจากคู่แข่งโดยไม่ได้รับอนุญาต รวมถึงการขนส่งสินค้าประเภทอาหารแช่แข็งในบางครั้งได้รับความเสียหาย อันมีสาเหตุสำคัญหลักมาจากการสูญเสียอุณหภูมิภายในตู้บรรทุกน้ำแข็ง ส่งผลกระทบให้คุณภาพอาหารแช่แข็งลดลง เกิดการเน่าเสีย ส่งผลให้เสียความเชื่อมั่นจากลูกค้า ดังนั้น งานวิจัยนี้เล็งเห็นถึงความสำคัญของปัญหาดังกล่าวจึงได้นำเสนอการออกแบบและพัฒนาระบบติดตามพิกัดตำแหน่งเส้นทางการเดินรถที่สามารถรับส่งข้อมูลจำนวนการเปิดปิดประตู และอุณหภูมิภายในตู้รถบรรทุกน้ำแข็งด้วยระบบ IoT รวมไปถึงการแสดงผลข้อมูลผ่านเว็บแอปพลิเคชันเพื่อเฝ้าระวังและติดตามรถตู้บรรทุกน้ำแข็งได้อย่างต่อเนื่อง และทันท่วงที

สาขาวิชา วิศวกรรมเมคคาทรอนิกส์ ลายมือชื่อนักศึกษา กำพล

ปีการศึกษา 2564 ลายมือชื่ออาจารย์ที่ปรึกษา

KAMPON NONKEAW : TRACKING AND DATA LOGGER SYSTEM FOR ICE TRUCK. THESIS ADVISOR : ASSOC. PROF. JIRAPHON SRISERTPOL, Ph.D., 118 PP.

Keywords: Gps Tracking/Ice Truck/Microcontroller/Internet Of Things

At present, ice transport entrepreneurs often confront problems with ice truck drivers' behavior in terms of driving out the path, hiring to transport ice from competitors without permission, and the transportation of frozen food items problem sometimes damaged. This is mainly caused by the loss of temperature inside the ice tanker. As a result, the quality of frozen food is reduced and spoilage. Affect the confidence of customers. Therefore, this research recognizes the importance of such problems, it is proposed to design and develop a system to track the location of the route of the truck that can send and receive data about the number of opening and closing doors and the temperature inside the ice truck with an IoT system, including displaying data via web applications to monitor and track the ice truck in real-time.

School of Mechatronic Engineering

Academic Year 2021

Student's Signature Kampon

Advisor's Signature

กิตติกรรมประกาศ

วิทยานิพนธ์นี้สำเร็จลุล่วงตามวัตถุประสงค์ทุกประการ ผู้วิจัยขอขอบพระคุณ และขอขอบคุณบุคคลต่าง ๆ ที่ให้คำแนะนำ และช่วยเหลืออย่างดียิ่ง ทั้งด้านวิชาการและด้านการดำเนินงานวิจัย ดังนี้

ขอขอบพระคุณ มหาวิทยาลัยเทคโนโลยีสุรนารี และห้างหุ้นส่วนจำกัด อาร์เอพี เอ็นเตอร์ไพรส์ แอนด์ เซอร์วิสเซส ให้การสนับสนุนสถานที่และอุปกรณ์สำหรับงานวิจัย

ขอขอบพระคุณ รองศาสตราจารย์ ดร.จิระพล ศรีเสริฐผล อาจารย์ที่ปรึกษาวิทยานิพนธ์ ที่เมตตาให้โอกาสให้ความรู้ และคำปรึกษาแนะนำ ในการทำงานวิจัย รวมไปถึงคำแนะนำในการดำรงชีวิตแก่ผู้วิจัยตลอดจนสำเร็จการศึกษา

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.โศรฎา แข็งการ ประธานกรรมการสอบวิทยานิพนธ์ ที่กรุณาให้คำแนะนำและตรวจทานเนื้อหาวิทยานิพนธ์ให้ถูกต้องโดยสมบูรณ์

ขอขอบพระคุณ ห้างหุ้นส่วนจำกัด เคราการช่าง ที่อำนวยความสะดวกในด้านสถานที่และอุปกรณ์สำหรับงานวิจัย

และที่ขาดไม่ได้ขอกราบขอบพระคุณพ่อณัฐวุฒิ นนแก้ว และคุณแม่อุบลวรรณ นนแก้ว ที่ให้การอบรมเลี้ยงดู และส่งเสริมสนับสนุนการศึกษาเป็นอย่างดีมาโดยตลอดจนทำให้ผู้วิจัยประสบความสำเร็จในชีวิตตลอดมา

กำพล นนแก้ว

สารบัญ

หน้า

บทคัดย่อ (ภาษาไทย) ..... ก
บทคัดย่อ (ภาษาอังกฤษ) ..... ข
กิตติกรรมประกาศ ..... ค
สารบัญ ..... ง
สารบัญตาราง ..... ช
สารบัญรูป ..... ฌ
บทที่
1 บทนำ 1
1.1 ที่มาและความสำคัญของปัญหาการวิจัย 1
1.2 วัตถุประสงค์ของงานวิจัย 3
1.3 ขอบเขตของงานวิจัย 3
1.4 ระเบียบวิธีวิจัย 4
1.5 สถานที่ทำงานวิจัย 4
1.6 ประโยชน์ที่คาดว่าจะได้รับ 4
1.7 การจัดทำรูปเล่มวิทยานิพนธ์ 5
2 ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง 6
2.1 กล่าวนำ 6
2.2 ความรู้เกี่ยวกับไมโครคอนโทรลเลอร์ 6
2.2.1 อาดูอีโน่ (Arduino) 7
2.2.1.1 อาดูอีโน่ ยูโน่ อาร์สาม (Arduino Uno R3) 7
2.2.1.2 อาดูอีโน่ เมก้า 2560 (Arduino Maga 2560) 9
2.2.2 โหนด เอ็มซียู (NodeMCU) 11
2.2.2.1 NodeMCU V2 ESP8266 Development Kit ESP-12F/N 11
2.3 เซนเซอร์ตรวจวัดอุณหภูมิ 13
2.3.1 เทอร์โมคัปเปิล (Thermocouples) 13

สารบัญ (ต่อ)

หน้า

2.3.2 Resistance Temperature Detector (RTD) 14
2.3.3 Thermistor 15
2.3.4 Digital temperature sensor 16
2.4 เซนเซอร์ตรวจจับ (Proximity Sensors) 19
2.4.1 Magnetic Reed Switch 19
2.5 จอแสดงผลภาพ (3.5 Inch TFT Color Screen Module 320x480) 20
2.6 บัสเซอร์ โมดูล (Active Buzzer Module) 21
2.7 โมดูลนาฬิกา DS3231 (DS3231 module) 22
2.8 Pocket Wi-Fi 23
2.9 Constant current/voltage 5-36V 3A buck regulator with Display 24
2.10 ระบบการหาตำแหน่งทั่วโลก GPS (Global Positioning System) 25
2.10.1 ประวัติความเป็นมาของ GPS 25
2.10.2 หลักการทำงาน GPS 26
2.10.2.1 ดาวเทียม (Space Segment) 27
2.10.2.2 สถานีภาคพื้นดิน (Control Segment) 27
2.10.2.3 อุปกรณ์สัญญาณ GPS (User Segment) 27
2.10.3 โมดูล GPS Ublox NEO-7M 28
2.10.4 GPS Antenna 29
2.10.5 ปัจจัยที่ส่งผลต่อสัญญาณ GPS 30
2.10.6 ประเภทของการใช้งาน GPS ในปัจจุบัน 31
2.11 โปรแกรม Arduino ide 32
2.12 โปรโตคอล (Protocol) 33
2.12.1 HTTP (Hypertext Transfer Protocol) 33
2.12.1.1 REST API (Representational State Transfer) 34
2.12.2 MQTT (Message Queue Telemetry Transport) 35

สารบัญ (ต่อ)

หน้า

2.13 โมดูลตัวอ่านการ์ด (MicroSD Card Adapter) 36
2.14 ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง 36
2.15 สรุป 40
3 วิธีการดำเนินงานวิจัย 42
3.1 กล่าวนำ 42
3.2 การออกแบบระบบ 42
3.3 ระบบตรวจวัดค่าเซนเซอร์ต่าง ๆ 44
3.3.1 ตัวตรวจวัดพิกัดตำแหน่งทางภูมิศาสตร์ 44
3.3.2 ตัวตรวจสอบวันที่และเวลา 45
3.3.3 ตัวตรวจสอบการเปิดปิดประตู 45
3.3.4 ตัวตรวจวัดอุณหภูมิ 45
3.4 ระบบส่งข้อมูลไปยัง Server 45
3.5 ระบบจอแสดงผลและการแจ้งเตือน 46
3.6 ระบบบันทึกข้อมูลสำรอง 47
3.7 สรุป 47
4 ผลการทดลอง และการวิเคราะห์ผลการทดลอง 49
4.1 กล่าวนำ 49
4.2 การทดสอบ Lab Tests 49
4.2.1 การทดสอบโมดูล GPS Ublox NEO-7M 49
4.2.2 การทดสอบเซนเซอร์ Magnetic Reed Switch BR-1021 55
4.2.3 การทดสอบเซนเซอร์ DS18B20 56
4.3 การทดสอบภาคสนาม 62
4.3.1 การทดสอบภาคสนามตำแหน่งทางภูมิศาสตร์ 62
4.3.2 การทดสอบภาคสนามเซนเซอร์ตรวจจับประตู 66
4.3.3 การทดสอบภาคสนามเซนเซอร์อุณหภูมิ 68
4.3.4 การทดสอบภาคสนามระบบบันทึกข้อมูลสำรอง 70
4.3.5 การทดสอบภาคสนาม ระบบจอแสดงผลสถานะทำงานอุปกรณ์ 72

สารบัญ (ต่อ)

หน้า

4.4 สรุป 73
5 สรุป 74
5.1 สรุป 74
5.1.1 ระบบตำแหน่งทางภูมิศาสตร์ 74
5.1.2 ระบบเซนเซอร์ตรวจจับประตู 74
5.1.3 ระบบเซนเซอร์ตรวจวัดอุณหภูมิ 75
5.1.4 ระบบบันทึกข้อมูลสำรอง 75
5.1.5 ระบบแสดงผลและการแจ้งเตือน 75
5.1.5.1 ระบบแสดงผล 75
5.1.5.2 ระบบการแจ้งเตือนด้วยเสียง 76
5.1.5.3 ระบบเว็บแอปพลิเคชัน 76
5.2 ข้อเสนอแนะ 76
5.2.1 ข้อเสนอแนะเกี่ยวกับกล่องชุดอุปกรณ์ต้นแบบ 76
รายการอ้างอิง 77
ภาคผนวก
ภาคผนวก ก. โค้ดโปรแกรมสำหรับไมโครคอนโทรลเลอร์ 78
ภาคผนวก ข. การติดตั้งชุดอุปกรณ์ต้นแบบ และการทดสอบภาคสนาม 105
ภาคผนวก ค. บทความที่ได้รับการตีพิมพ์เผยแพร่ 112
ประวัติผู้เขียน 118

สารบัญตาราง

ตารางที่ **หน้า**

4.1 ตารางสรุปการทดสอบการตรวจพิกัดตำแหน่งทางภูมิศาสตร์ 54
4.2 ตารางสรุปผลการทดสอบเซนเซอร์ Magnetic Reed Switch BR-1021 56
4.3 ตารางสรุปค่าความผิดพลาดของการทดสอบตรวจวัดอุณหภูมิ 61
4.4 ตารางสรุปการทดสอบตำแหน่งทางภูมิศาสตร์ภาคสนาม 62
4.5 ตารางสรุปการทดสอบเซนเซอร์ตรวจจับประตูภาคสนาม 66
4.6 ตารางสรุปการทดสอบเซนเซอร์อุณหภูมิภาคสนาม 68

สารบัญรูป

รูปที่ **หน้า**

1.1 ภายนอก รถตู้บรรทุกน้ำแข็งประเภท สี่ล้อ 1
1.2 ภายในด้านหน้า รถตู้บรรทุกน้ำแข็งประเภท สี่ล้อ 2
1.3 ด้านหลัง รถตู้บรรทุกน้ำแข็งประเภท สี่ล้อ 2
1.4 แผนผังรถตู้บรรทุกน้ำแข็งประเภท สี่ล้อ 3
2.1 โครงสร้างโดยทั่วไปของไมโครคอนโทรลเลอร์ 6
2.2 บอร์ด Arduino Uno R3 7
2.3 โครงสร้างพื้นฐานของบอร์ด Arduino Uno R3 8
2.4 บอร์ด Arduino Mega 2560 9
2.5 โครงสร้างพื้นฐานของบอร์ด Arduino Mega 2560 10
2.6 บอร์ด NodeMCU V2 ESP8266 Development Kit ESP-12F/N 11
2.7 โครงสร้างพื้นฐานของบอร์ด NodeMCU V2 ESP8266 Development Kit ESP-12F/N 12
2.8 หลักการทำงานของ Thermocouples 14
2.9 ตัวอย่าง Thermocouples 14
2.10 หลักการทำงานของ Resistance Temperature Detector 14
2.11 ตัวอย่าง Resistance Temperature Detector 15
2.12 ตัวอย่าง Thermistor 15
2.13 ตัวอย่าง Digital temperature sensor 16
2.14 บล็อกไดอะแกรมของ DS18B20 16
2.15 หลักการทำงานของเซนเซอร์อุณหภูมิ DS18B20 17
2.16 แผนภาพวงจรของอินเทอร์เฟซ DS18B20 หลายตัวในเทคโนโลยี 1-Wire 18
2.17 หลักการทำงานของ Magnetic sensor 19
2.18 ตัวอย่าง Magnetic Reed Switch 19
2.19 ตัวอย่าง จอแสดงผลภาพ 3.5 Inch TFT Color Screen Module 320x480 20
2.20 ตัวอย่าง Active Buzzer Module 21

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

2.21 แผนภาพวงจรของอินเทอร์เฟซ Arduino กับ Active Buzzer Module 21
2.22 ตัวอย่างโมดูลนาฬิกา DS3231 22
2.23 แผนภาพวงจรของอินเทอร์เฟซ Arduino กับ DS3231 Real Time Clock Module 23
2.24 ตัวอย่าง Pocket Wi-Fi 23
2.25 ตัวอย่าง Constant current/voltage 5-36V 3A buck regulator with Display 24
2.26 หลักการการคำนวณหาพิกัดตำแหน่งของGPS 26
2.27 หลักการทำงาน GPS ในการหาพิกัดตำแหน่ง 26
2.28 ตัวอย่างโมดูล GPS Ublox NEO-7M 28
2.29 การอินเทอร์เฟซ Arduino กับ โมดูล GPS Ublox NEO-7M 28
2.30 เสาอากาศ GPS แบบ Passive Antenna 29
2.31 เสาอากาศ GPS แบบ Active Antenna 29
2.32 วงจรพื้นฐานของ Active Antenna 30
2.33 หลักการทำงาน GPS Tracking System 31
2.34 โปรแกรม Arduino ide 32
2.35 หลักการทำงาน HTTP 33
2.36 การนำ REST API ไปใช้งาน 34
2.37 หลักการทำงาน MQTT 35
2.38 ตัวอย่าง โมดูลตัวอ่านการ์ด Micro SD 36
3.1 สถาปัตยกรรมระบบ 43
3.2 ชุดอุปกรณ์ต้นแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง 43
3.3 แผนผังตำแหน่งการติดตั้งชุดอุปกรณ์ต้นแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง 44
3.4 ตัวอย่าง Web Application 45
3.5 กระบวนการทำงานแสดงผลข้อมูลบน Web Application 46
3.6 การออกแบบสถานะทำงานของอุปกรณ์ปกติ 46
3.7 การออกแบบสถานะทำงานของอุปกรณ์ผิดปกติ 47
4.1 สถานที่ทดสอบ Lab Tests พิกัดตำแหน่งทางภูมิศาสตร์ 49

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

4.2 กราฟเปรียบเทียบความผิดพลาดพิกัดตำแหน่งของ GPS 50
4.3 กราฟเปรียบเทียบความผิดพลาดความเร็วของ GPS 51
4.4 กราฟเปรียบเทียบความเร็วของ GPS ที่ความเร็วคงที่ 20 km/h 51
4.5 กราฟเปรียบเทียบความผิดพลาดความเร็วของ GPS ที่ความเร็วคงที่ 20 km/h 52
4.6 กราฟเปรียบเทียบความเร็วของ GPS ที่ความเร็วคงที่ 30 km/h 52
4.7 กราฟเปรียบเทียบความผิดพลาดความเร็วของ GPS ที่ความเร็วคงที่ 30 km/h 53
4.8 กราฟเปรียบเทียบความเร็วของ GPS ที่ความเร็วคงที่ 40 km/h 53
4.9 กราฟเปรียบเทียบความผิดพลาดความเร็วของ GPS ที่ความเร็วคงที่ 40 km/h 54
4.10 ทดสอบระยะการตรวจจับเซนเซอร์ Magnetic Reed Switch BR-1021 55
4.11 เครื่อง Water Bath ชุดทดลองการแลกเปลี่ยนความเย็น 56
4.12 เครื่อง Water Bath ชุดทดลองการแลกเปลี่ยนความร้อน 57
4.13 กราฟ Random Test ทั้งสามเซนเซอร์ตรวจวัดอุณหภูมิ 57
4.14 กราฟ Static Calibration Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 1 58
4.15 กราฟ Static Calibration Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 2 58
4.16 กราฟ Static Calibration Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 3 59
4.17 กราฟ Sequential Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 1 59
4.18 กราฟ Sequential Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 2 60
4.19 กราฟ Sequential Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 3 60
4.20 สมการหาค่า Overall instrument error 61
4.21 ตำแหน่งการทดสอบภาคสนาม 63
4.22 เส้นทางการเดินรถภาคสนามบนเว็บแอปพลิเคชัน 63
4.23 ตำแหน่งการทดสอบภาคสนามบนเว็บแอปพลิเคชัน 64
4.24 ตำแหน่งการเปิดประตูการทดสอบภาคสนาม บนเว็บแอปพลิเคชัน 64
4.25 กราฟความเร็วการทดสอบภาคสนามบนเว็บแอปพลิเคชัน 65
4.26 ผลจำนวนการเปิดประตูภาคสนามบนเว็บแอปพลิเคชัน 67
4.27 ผลอุณหภูมิภาคสนามบนเว็บแอปพลิเคชัน 68
4.28 กราฟผลอุณหภูมิภาคสนามบนเว็บแอปพลิเคชัน 69

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

4.29 กราฟ อุณหภูมิจากข้อมูลสำรองในโมดูล SD Card 70
4.30 กราฟตรวจจับการเปิดปิดประตูจากโมดูล SD Card 71
4.31 กราฟความเร็วจากข้อมูลสำรองโมดูล SD Card 71
4.32 จอแสดงผลสถานะทำงานอุปกรณ์ ขณะปิดประตู 72
4.33 จอแสดงผลสถานะทำงานอุปกรณ์ ขณะเปิดประตู 72
ข.1 แหล่งจ่ายไฟชุดอุปกรณ์ต้นแบบหัวเสียบที่จุดบุหรี่ 12v/24v (10A) 106
ข.2 ตำแหน่งการติดตั้งกล่องอุปกรณ์ควบคุมใต้ที่ที่นั่ง 106
ข.3 ตำแหน่งเซนเซอร์ตรวจจับประตูบานที่ 1 และ2 (ประตูด้านหลัง) 107
ข.4 ตำแหน่งเซนเซอร์ตรวจจับประตูบานที่ 3 (ประตูด้านข้าง) 107
ข.5 เสาอากาศสำหรับโมดูล จีพีเอส 108
ข.6 ภายในชุดอุปกรณ์ติดตั้งเซนเซอร์ตรวจวัดอุณหภูมิ 108
ข.7 ภายนอกชุดอุปกรณ์ติดตั้งเซนเซอร์ตรวจวัดอุณหภูมิ 109
ข.8 น้ำแข็งที่ใช้ทดสอบภาคสนามจำนวน 30 กระสอบ 109
ข.9 DE-3003 DIGITAL THERMOMETER TYPE K 110
ข.10 ตัวอย่างการตรวจวัดอุณหภูมิโดยใช้ DE-3003 DIGITAL THERMOMETER TYPE K 110
ข.11 การบันทึกวีดีโอหน้าจอระบบแสดงผล 111

บทที่ 1
บทนำ

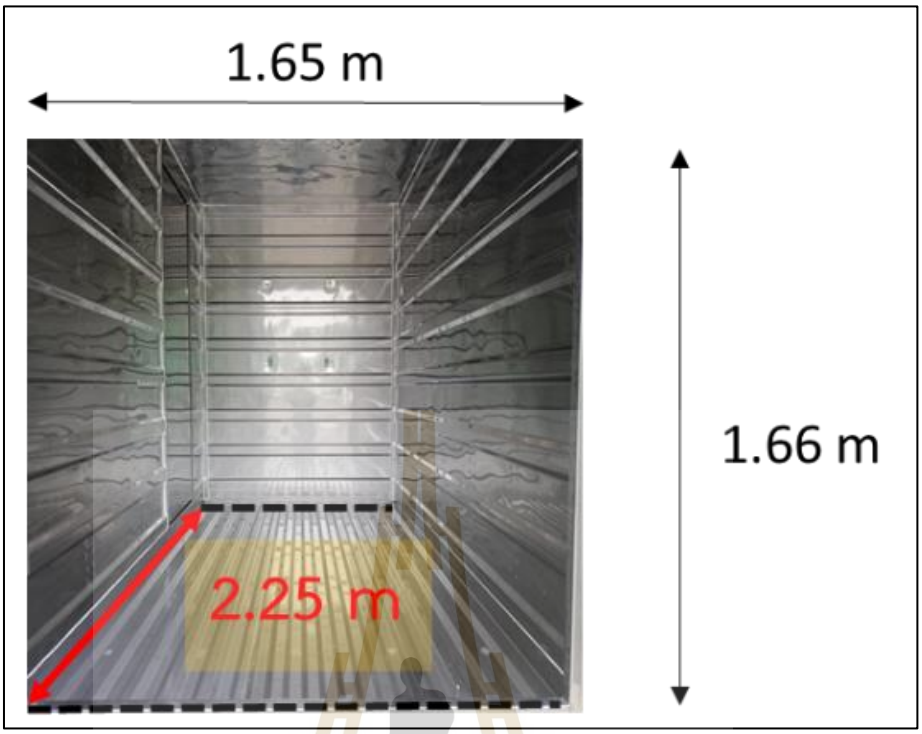
1.1 ความเป็นมาและความสำคัญของปัญหาการวิจัย

ปัจจุบันจากการสำรวจรถตู้บรรทุกน้ำแข็งพบว่าพนักงานขับรถตู้บรรทุกน้ำแข็ง มีพฤติกรรมการขับขี่ที่ไม่เหมาะสม เช่น การขับรถตู้บรรทุกน้ำแข็งออกนอกเส้นทางการไปรับจ้างขนน้ำแข็งจากคู่แข่งโดยไม่ได้รับอนุญาต รวมถึงการบรรทุกขนสินค้าประเภทอาหารแช่แข็งในบางครั้ง อาหารแช่แข็งได้รับความเสียหาย ซึ่งมีสาเหตุสำคัญหลักมาจากการสูญเสียอุณหภูมิภายในตู้บรรทุกน้ำแข็ง ส่งผลกระทบให้คุณภาพอาหารแช่แข็งลดลง เกิดการเน่าเสีย ซึ่งทำให้เสียความเชื่อมั่นจากลูกค้า ดังนั้นงานวิจัยนี้ได้นำเสนอออกแบบและพัฒนาระบบติดตามพิกัดตำแหน่งการเดินรถที่สามารถรับส่งข้อมูลการเปิดปิดประตู และอุณหภูมิภายในตู้รถบรรทุกน้ำแข็งด้วยระบบ IoT ซึ่งรถตู้บรรทุกน้ำแข็งที่ใช้ติดตั้งชุดอุปกรณ์ต้นแบบเป็นรถยนต์ประเภท 4 ล้อ ดังรูปที่ 1.1 และรูปที่ 1.2 มีบานประตูรถตู้น้ำแข็งทั้งหมด 3 บาน โดยบานประตูที่ 1 และ2 อยู่ด้านหลังรถตู้บรรทุกน้ำแข็งดังรูปที่ 1.3 ส่วนบานประตูที่ 3 อยู่บริเวณด้านข้างของรถตู้บรรทุกน้ำแข็งดังรูปที่ 1.1 และรถตู้บรรทุกน้ำแข็งมีขนาด 1.65 x 2.25 x 1.66 เมตร ดังรูปที่ 1.4

รูปที่ 1.1 ภายนอก รถตู้บรรทุกน้ำแข็งประเภท สี่ล้อ

รูปที่ 1.2 ภายในด้านหน้า รถตู้บรรทุกน้ำแข็งประเภท สี่ล้อ

รูปที่ 1.3 ด้านหลัง รถตู้บรรทุกน้ำแข็งประเภท สี่ล้อ



รูปที่ 1.4 แผนผังรถตู้บรรทุกน้ำแข็งประเภท สี่ล้อ

1.2 วัตถุประสงค์ของงานวิจัย

1.2.1. เพื่อออกแบบและสร้างระบบติดตามพิกัดตำแหน่งของรถตู้บรรทุกน้ำแข็งในการบันทึกข้อมูลจำนวนการเปิดปิดประตู และอุณหภูมิภายในรถบรรทุกน้ำแข็ง

1.2.2. เพื่อออกแบบและสร้างระบบอินเทอร์เน็ตของสรรพสิ่ง (IOT) ส่งข้อมูลไปยังฐานข้อมูลออนไลน์ และแสดงผลข้อมูลบนเว็บแอปพลิเคชัน

1.2.3. เพื่อออกแบบและสร้างระบบแจ้งเตือนด้วยเสียงเมื่อมีการเปิดประตูรถตู้บรรทุกน้ำแข็ง และจอแสดงผลการทำงานของชุดอุปกรณ์ต้นแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

1.3 ขอบเขตของงานวิจัย

1.3.1. ตรวจวัดอุณหภูมิช่วง -10 ถึง 80 องศาเซลเซียส ภายในตู้รถบรรทุกน้ำแข็งที่มีค่าความผิดพลาดไม่เกิน 2 องศาเซลเซียส

1.3.2. ตรวจจับตำแหน่งเส้นทางการเดินรถและความเร็วในการขับขี่ของคนขับรถตู้บรรทุกน้ำแข็ง ที่มีค่าความผิดพลาดตำแหน่ง ไม่เกิน 5 เมตร และค่าความผิดพลาดความเร็วไม่เกิน 10 กิโลเมตรต่อชั่วโมง

1.3.3. ตรวจสอบจำนวนการเปิดปิดประตูของรถตู้บรรทุกน้ำแข็ง

1.3.4. ส่งข้อมูลจากรถบรรทุกน้ำแข็งด้วยระบบ IOT

1.4 ระเบียบวิธีการวิจัย

1.4.1. ศึกษาวิจัยเกี่ยวกับระบบติดตามรถ การเลือกใช้อุปกรณ์ในการตรวจวัดอุณหภูมิ จำนวนการเปิดปิดประตู และการส่งข้อมูลไปยังฐานข้อมูลออนไลน์ ด้วยระบบอินเทอร์เน็ตของสรรพสิ่ง (IOT)

1.4.2. ออกแบบสถาปัตยกรรมขอบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

1.4.3. พิจารณาการเลือกใช้อุปกรณ์ในสถาปัตยกรรมของระบบ

1.4.4. ออกแบบและทดสอบเทียบอุปกรณ์ระบบเซนเซอร์ตรวจวัดอุณหภูมิ และระบบโมดูล จีพีเอส ในการระบุตำแหน่งทางภูมิศาสตร์

1.4.5. ออกแบบและสร้างชุดอุปกรณ์ต้นแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

1.4.6. เขียนโปรแกรมควบคุมการทำงานชุดอุปกรณ์ต้นแบบของระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

1.4.7. ออกแบบและทดสอบการทำงานภาคสนามชุดอุปกรณ์ต้นแบบของระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

1.4.8. รวบรวมข้อมูล วิเคราะห์ และสรุปผลการวิจัย

1.4.9. จัดทำรูปเล่มวิทยานิพนธ์

1.4.10. สอบวิทยานิพนธ์

1.5 สถานที่ทำงานวิจัย

1.5.1. ห้างหุ้นส่วนจำกัด เคราการช่าง

1.5.2. ห้างหุ้นส่วนจำกัด อาร์เอพี เอ็นเตอร์ไพรส์ แอนด์ เซอร์วิสเซส

1.5.3. มหาวิทยาลัยเทคโนโลยีสุรนารี

1.6 ประโยชน์ที่คาดว่าจะได้รับ

1.6.1. เพิ่มประสิทธิภาพรถตู้บรรทุกน้ำแข็ง

1.6.2. สามารถตรวจสอบเส้นทางการเดินรถ จำนวนการเปิดปิดประตูและอุณหภูมิรถตู้บรรทุกน้ำแข็งได้แบบเวลาเรียลไทม์ผ่านเว็บแอปพลิเคชัน

1.6.3. สามารถแจ้งเตือนด้วยสัญญาณเสียง เมื่อพนักงานลืมปิดหรือปิดประตูไม่สนิทของรถตู้บรรทุกน้ำแข็ง และมีสถานะแจ้งเตือนเซนเซอร์ต่างๆ ผ่านจอแสดงผล TFT LCD

1.6.4. สามารถนำข้อมูลไปประเมินและแก้ไขปัญหาอุณหภูมิภายในรถตู้บรรทุกน้ำแข็งได้ทันที

1.7 การจัดทำรูปเล่มวิทยานิพนธ์

วิทยานิพนธ์นี้ประกอบด้วย 5 บท 3 ภาคผนวก ซึ่งมีรายละเอียดโดยย่อดังนี้

บทที่ 1 เป็นบทนำกล่าวถึงความสำคัญของปัญหา วัตถุประสงค์ และเป้าหมายของงานวิจัยวิทยานิพนธ์ ตลอดจนขอบเขต และประโยชน์ที่คาดว่าจะได้รับจากงานวิจัยนี้

บทที่ 2 กล่าวถึงทฤษฎีที่เกี่ยวข้องกับไมโครคอนโทรลเลอร์ เซนเซอร์ตรวจวัดอุณหภูมิ เซนเซอร์ตรวจจับวัตถุ ระบบพิกัดตำแหน่ง อินเทอร์เน็ตทุกสรรพสิ่ง และงานวิจัยที่เกี่ยวข้อง

บทที่ 3 กล่าวถึงการออกแบบ และพัฒนาระบบติดตามพิกัดตำแหน่งการเดินรถที่สามารถรับส่งข้อมูลการเปิดปิดประตู และอุณหภูมิภายในตู้รถบรรทุกน้ำแข็งด้วยระบบ IOT

บทที่ 4 ผลการดำเนินการวิจัยและการอภิปรายผล การทดสอบตรวจวัดพิกัดตำแหน่งทางภูมิศาสตร์ ตรวจวัดอุณหภูมิ และผลทดสอบการทำงานขอบระบบ

บทที่ 5 บทสรุปและข้อเสนอแนะ

ภาคผนวก ก. โค้ดโปรแกรมสำหรับไมโครคอนโทรลเลอร์

ภาคผนวก ข. การติดตั้งชุดอุปกรณ์ต้นแบบ และการทดสอบภาคสนาม

ภาคผนวก ค. บทความที่ได้รับการตีพิมพ์เผยแพร่

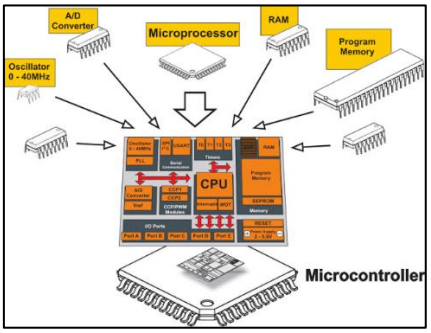
บทที่ 2
ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

2.1 กล่าวนำ

อินเทอร์เน็ตทุกสรรพสิ่ง (Internet of Things) หมายถึง การที่อุปกรณ์ต่าง ๆ เชื่อมโยงกันบนเครือข่ายอินเตอร์เน็ต สถาปัตยกรรมเกี่ยวกับอินเทอร์เน็ตในทุกสิ่ง มีองค์ประกอบหลัก 3 องค์ประกอบ 1. สิ่งต่าง ๆ (Things) คืออุปกรณ์ที่ใช้ในการเชื่อมต่อ 2. เครือข่าย (Networks) คือ ช่องทางในการเชื่อมต่อสิ่งต่าง ๆ ไปยังระบบคลาวด์ และ 3. ระบบคลาวด์ (Cloud) คือ เซิร์ฟเวอร์ทำหน้าที่ในการรวมและเก็บข้อมูล ในบทความนี้จะกล่าวถึงปริทัศน์วรรณกรรมเกี่ยวกับการนำระบบอินเทอร์เน็ตในทุกสิ่งมาประยุกต์ใช้งาน การตรวจวัดอุณหภูมิ การตรวจจับวัตถุ รวมไปถึงระบบพิกัดทางภูมิศาสตร์ เป็นต้น

2.2 ความรู้เกี่ยวกับไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (Microcontroller) เขียนสัญญาลักษณ์เป็น μC หรือ MCU คืออุปกรณ์ขนาดเล็กที่มีความสามารถคล้ายคลึงกับระบบคอมพิวเตอร์ โดยมีส่วนประกอบสำคัญได้แก่ ซีพียู หน่วยความจำ และพอร์ต เข้าไว้ด้วยกันดังรูปที่ 2.1 รวมถึงยังมีโปรแกรมคำสั่งเพื่อใช้ในการควบคุมขา Input/Output เพื่อสั่งงานให้ไปควบคุมอุปกรณ์ต่าง ๆ ประยุกต์ใช้งานได้หลากหลายทั้งด้านช่องสัญญาณดิจิทัล และแอนะล็อก ยกตัวอย่างเช่น ระบบสัญญาณตอบรับอัตโนมัติ เป็นต้น ในยุคปัจจุบันสามารถเชื่อมต่อกับระบบเน็ตเวิร์ก ผ่านระบบเครือข่ายอินอินเตอร์เน็ตได้ด้วย



รูปที่ 2.1 โครงสร้างโดยทั่วไปของไมโครคอนโทรลเลอร์

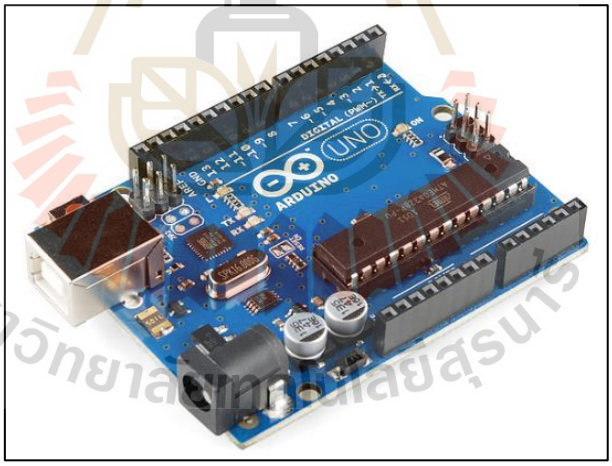
2.2.1 อาดูอีโน่ (Arduino)

อาดูอีโน่ (Arduino) เป็นไมโครคอนโทรลเลอร์ (Micro Controller) บอร์ดแบบสำเร็จรูป ถูกสร้างมาจาก คอนโทรลเลอร์ ตระกูล ARM ของ ATMEL เป็นแบบ Opensource ทั้งด้าน ฮาร์ดแวร์ และซอฟต์แวร์ ที่สามารถนำไปพัฒนาต่อกับอุปกรณ์ต่าง ๆ ได้ และความสามารถในการเพิ่ม boot loader เข้าไปที่ตัว ARM จึงทำให้การ upload code เข้าตัวบอร์ดสามารถทำได้ง่ายขึ้น การพัฒนา ซอฟต์แวร์ ที่จะใช้ในการควบคุมตัวบอร์ด ก็จะเป็นในลักษณะของ C++ และตัวบอร์ดยังสามารถนำโมดูลมาต่อเพิ่ม เพื่อเพิ่มความสามารถต่าง ๆ ได้

จุดเด่นของบอร์ด Arduino

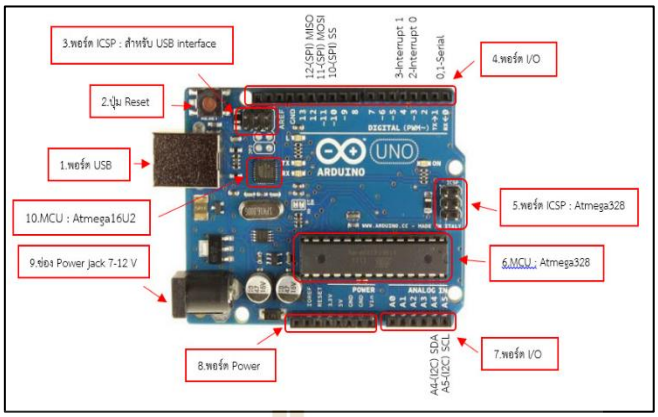
1. เป็น Opensource ทำให้ง่ายต่อการนำไปพัฒนาได้หลากหลายด้าน
2. Cross Platform สามารถพัฒนาโปรแกรมบนระบบ OS ต่าง ๆ ได้
3. มีราคาไม่แพง
4. ง่ายต่อการพัฒนามีชุดคำสั่งพื้นฐาน โครงสร้างของโปรแกรมไม่ซับซ้อน

2.2.1.1 อาดุอีโน่ ยูโน่ อาร์สาม (Arduino Uno R3)



รูปที่ 2.2 บอร์ด Arduino Uno R3

Arduino Uno R3 เป็นบอร์ดที่ได้รับความนิยมมากที่สุด เนื่องจากราคาของบอร์ดที่ไม่แพง และ Library ต่าง ๆ ที่ใช้งานถูกพัฒนาอ้างอิงกับบอร์ดนี้เป็นหลักแสดงตัวอย่างบอร์ดดังรูปที่ 2.2



รูปที่ 2.3 โครงสร้างพื้นฐานของบอร์ด Arduino Uno R3

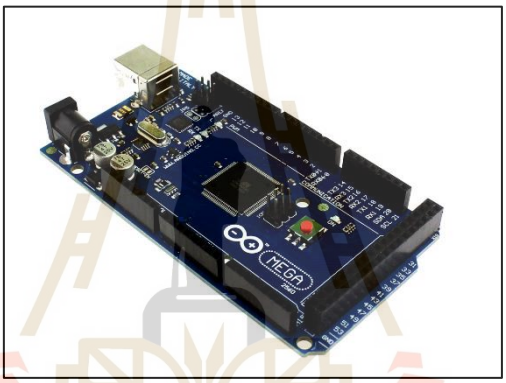
โครงสร้างพื้นฐานของบอร์ด Arduino Uno R3 แสดงดังรูปที่ 2.3

1. USB Port ใช้สำหรับเชื่อมต่อกับ คอมพิวเตอร์ เพื่ออัปโหลดโปรแกรมเข้า MCU และจ่ายไฟให้กับบอร์ด
2. Reset Button เป็นปุ่ม Reset ใช้กดเมื่อต้องการให้ MCU เริ่มการทำงานใหม่
3. ICSP Port เป็นพอร์ตที่ใช้โปรแกรม Visual Com port บน Atmega16U2
4. I/O Port Digital I/O ตั้งแต่ขา D0 ถึง D13 นอกจากนี้ บาง Pin จะทำหน้าที่อื่น ๆ เพิ่มเติมด้วย เช่น Pin0,1 เป็นขา Tx,Rx Serial, Pin3,5,6,9,10 และ 11 เป็นขา PWM
5. ICSP Port Atmega328 เป็นพอร์ตที่ใช้โปรแกรม Boot loader
6. MCU Atmega328 เป็น MCU ที่ใช้บนบอร์ด Arduino
7. I/O Port ช่องรับสัญญาณแอนะล็อก ตั้งแต่ขา A0-A5
8. Power Port ไฟเลี้ยงของบอร์ดเมื่อต้องการจ่ายไฟให้กับวงจรภายนอก ประกอบด้วยขา +3.3 V, +5V, GND, Vin
9. Power Jack รับไฟจาก Adapter โดยที่แรงดันอยู่ระหว่าง 7-12 V
10. MCU Atmega16U2 เป็น MCU ที่ทำหน้าที่เป็น USB to Serial โดย Atmega328 จะติดต่อกับ คอมพิวเตอร์ ผ่าน Atmega16U2

คุณสมบัติทั่วไปของบอร์ด Arduino Uno R3

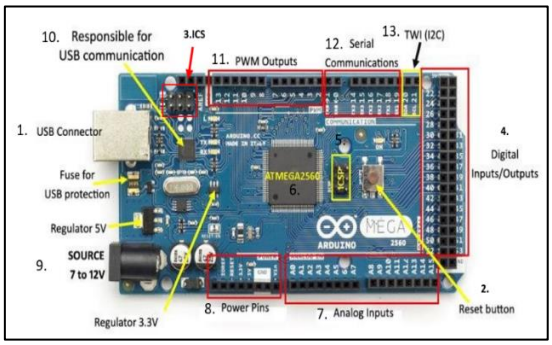
1. ใช้ชิป ATmega328 รันที่ความถี่ 16 MHz
2. หน่วยความจำแฟลช 32 KB แรม 2 KB
3. บอร์ดใช้ไฟเลี้ยง 7ถึง 12 V
4. มีระดับแรงดันไฟฟ้าในการทำงานและขาสัญญาณอยู่ที่ 5 V
5. Digital I/O 14 ขา (เป็น PWM ได้6 ขา)
6. Analog I/O 6 ขา Serial UART 1 ชุด เป็นพอร์ตสื่อสารอนุกรม

2.2.1.2 อาดูอีโน่ เมก้า 2560 (Arduino Maga 2560)



รูปที่ 2.4 บอร์ด Arduino Mega 2560

บอร์ด Arduino Mega 2560 เป็นไมโครคอนโทรลเลอร์ บอร์ดที่ออกแบบมาสำหรับงานที่ต้องใช้ I/O Port มากกว่า Arduino Uno R3 เช่น งานที่ต้องการรับสัญญาณจาก Sensor หรือควบคุมมอเตอร์ Servo หลาย ๆ ตัว ทำให้ I/O Port ของบอร์ด Arduino Uno R3 ไม่สามารถรองรับได้ ทั้งนี้บอร์ด Arduino Mega 2560 ยังมีความหน่วยความจำแบบ Flash มากกว่า Arduino Uno R3 ทำให้สามารถเขียนโค้ดโปรแกรมเข้าไปได้มากกว่า ในความเร็วของ MCU ที่เท่ากัน แสดงตัวอย่างบอร์ดดังรูปที่ 2.4



รูปที่ 2.5 โครงสร้างพื้นฐานของบอร์ด Arduino Mega 2560

โครงสร้างพื้นฐานของบอร์ด Arduino Mega 2560 แสดงดังรูปที่ 2.5

1. USB Port ใช้สำหรับเชื่อมต่อกับ คอมพิวเตอร์ เพื่ออัปโหลดลดโปรแกรมเข้า MCU และจ่ายไฟให้กับบอร์ด
2. Reset Button เป็นปุ่ม Reset ใช้กดเมื่อต้องการให้ MCU เริ่มการทำงานใหม่
3. ICSP Port เป็นพอร์ตที่ใช้โปรแกรม Visual Com port
4. I/O Port Digital I/O
5. ICSP Port Atmega328 เป็นพอร์ตที่ใช้โปรแกรม Boot loader
6. MCU Atmega2560 เป็น MCU ที่ใช้บนบอร์ด
7. I/O Port ช่องรับสัญญาณอนาล็อก ตั้งแต่ขา A0-A15
8. Power Port ไฟเลี้ยงบอร์ดเมื่อต้องการจ่ายไฟให้กับวงจรภายนอก ขา +3.3 V, +5V, GND, Vin
9. Power Jack รับไฟจาก Adapter โดยที่แรงดันอยู่ระหว่าง 7-12 V
10. MCU เป็น MCU ที่ทำหน้าที่เป็น USB to Serial โดย Atmega328 จะติดต่อกับคอมพิวเตอร์ ผ่าน Atmega16U2
11. PWM 2 ถึง 13 และ 44 ถึง 46 ให้ output PWM output 8-bits
12. I/O Port ดิจิตอลสามารถรับสัญญาณดิจิตอลหรือถ่ายโอนสัญญาณดิจิตอล ขา 0 ชื่อ Rx และ ขา 1 ที่ชื่อ Tx คือขาที่รับและส่งของ UART (Universal Asynchronous Receiver และ Transmitter) ตามลำดับขา 14, 16, 18 และขา 15, 17, 19 ก็เป็น Tx และ Rx ตามลำดับดังนั้นจึงมีทั้งหมดสี่ UARTS บนบอร์ด
13. TWI รองรับการเชื่อมต่อแบบ TWI (I2C) 20 (SDA) and 21 (SCL).

คุณสมบัติทั่วไปของบอร์ด Arduino Mega 2560

1. ไมโครคอนโทรลเลอร์ Atmega2560 ความเร็วสัญญาณนาฬิกา 16 MHz
2. หน่วยความจำแฟลช 256 KB, SRAM 8 KB, EEPROM 4 KB
3. แรงดันไฟฟ้าอินพุตที่แจ็ค SOURCE 7 ถึง 12 โวลต์
4. แรงดันไฟฟ้า5 โวลต์ กระแสตรงสำหรับ 3.3V Pin 50 mA
5. Digital I/O 54 ขา
6. Analog I/O 16 ขา

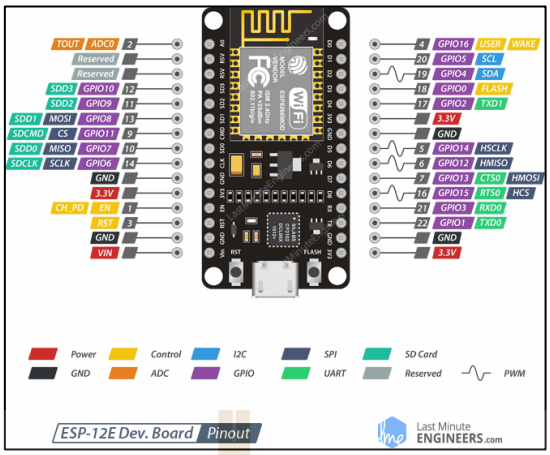
2.2.2 โหนด เอ็มซียู (NodeMCU)

NodeMCU เป็นไมโครคอนโทรลเลอร์แพลตฟอร์มหนึ่งที่ใช้ช่วยในการสร้างโปรเจค Internet of Things (IoT) ซึ่งเป็น Open Source สามารถเขียนโปรแกรมด้วยภาษา Lauได้ และมาพร้อมโมดูล Wi-Fi ESP8266 ซึ่งเป็นหัวใจสำคัญในการเชื่อมต่อสัญญาณอินเทอร์เน็ตโดยที่ NodeMCU มีลักษณะคล้าย Arduino ตรงที่มี I/O Port ภายในตัวและสามารถทำงานร่วมกับโปรแกรม Arduino IDE ได้จึงทำให้สามารถใช้ภาษา C/C++ ในการเขียนโปรแกรม

2.2.2.1 NodeMCU V2 ESP8266 Development Kit ESP-12F/N

NodeMCU V2 ESP8266 Development Kit ESP-12F/N เป็นบอร์ดเชื่อมต่อ Wi-Fi ยอดนิยมด้วยโมดูล ESP8266 แสดงตัวอย่างบอร์ดดังรูปที่ 2.6 และเหมาะสำหรับนำไปพัฒนาโปรเจค Internet of Things ได้อย่างง่ายดายเพราะ บอร์ดมี USB to UART มาให้พร้อมใช้งานอยู่แล้ว บอร์ดรุ่นนี้จะใช้ชิป ESP-12F สามารถรับสัญญาณได้ดีขึ้น 30-50%

รูปที่ 2.6 บอร์ด NodeMCU V2 ESP8266 Development Kit ESP-12F/N



รูปที่ 2.7 โครงสร้างพื้นฐานของบอร์ด NodeMCU V2 ESP8266 Development Kit ESP-12F/N

โครงสร้างพื้นฐานของบอร์ด NodeMCU V2 ESP8266 Development Kit ESP-12F/N แสดงดังรูปที่ 2.7

1. Power เมื่อต้องการจ่ายไฟให้กับวงจรภายนอก ขาไฟเลี้ยง +3.3 V, +5V, GND, Vin

2. Control ใช้สำหรับควบคุม NodeMCU/ESP8266 เช่น RST pin ใช้เพื่อรีเซ็ตชิป ESP8266

3. I2C ใช้สำหรับเชื่อมต่อเซนเซอร์ I2C และอุปกรณ์ต่อพ่วง สามารถรับรู้ฟังก์ชันการทำงานของอินเทอร์เฟซ I2C โดยทางโปรแกรม และความถี่สัญญาณนาฬิกาสูงสุด 100 kHz

4. SPI มี SPI สองตัว (SPI และ HSPI) ในโหมดสเลฟ และมาสเตอร์ SPI เหล่านี้ยังสนับสนุนคุณสมบัติ SPI ทั่วไปต่อไปนี้ โหมดจับเวลา 4 โหมดของการถ่ายโอนรูปแบบ SPI สูงสุด 80 MHz และนาฬิกาแบ่ง 80 MHz สูงสุด 64-Byte FIFO

5. SDIO Pins Secure Digital Input/Output Interface (SDIO) ซึ่งใช้เพื่อเชื่อมต่อการ์ด SD โดยตรง รองรับ 4-bit 25 MHz SDIO v1.1 และ 4-bit 50 MHz SDIO v2.0

6. GND คือขากราวด์ของ NodeMCU/ESP8266

7. ADC NodeMCU ถูกฝังด้วย SAR ADC ที่มีความแม่นยำ 10 บิต ฟังก์ชันทั้งสองนี้สามารถใช้งานได้

8. GPIO NodeMCU/ESP8266 มีขา GPIO 17 pin ซึ่งสามารถกำหนดให้กับฟังก์ชันต่าง ๆ เช่น I2C, I2S, UART, PWM, รีโมทคอนโทรล IR, ไฟ LED และปุ่ม

9. UART อินเทอร์เฟซ UART 2 แบบ (UART0 และ UART1) ซึ่งให้การสื่อสารแบบอะซิงโครนัส (RS232 และ RS485) และสามารถสื่อสารได้สูงสุด 4.5 Mbps UART0 (ขา TXD0, RXD0, RST0 & CTS0) ใช้สำหรับการสื่อสาร

10. PWM บอร์ดมี 4 ช่องสัญญาณของการปรับความกว้างพัลส์ (PWM) เอาต์พุต PWM สามารถนำไปใช้โดยทางโปรแกรมและใช้สำหรับขับมอเตอร์ดิจิตอลและไฟ LED ช่วงความถี่ PWM ปรับได้ตั้งแต่ 1000 μs ถึง 10000 μs (100 Hz และ 1 kHz)

คุณสมบัติทั่วไปของบอร์ด NodeMCU V2 ESP8266 Development Kit ESP-12F/N

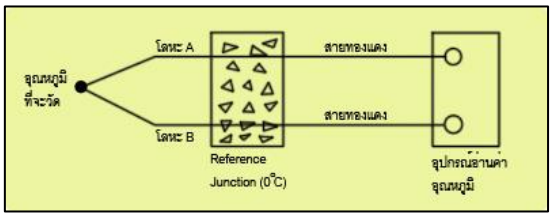
1. ใช้โมดูล ESP8266-12F ซึ่งมีหน่วยความจำ flash 4 MBytes
2. ใช้ชิป USB to UART เบอร์ CP2102 ของบริษัท SILICON LABS
3. ระดับสัญญาณลอจิกของสัญญาณอินพุตและเอาต์พุต (GPIO) 3.3 V
4. สามารถใช้ไฟเลี้ยงบอร์ดจากพอร์ต USB 5V DC หรือ แหล่งจ่ายไฟภายนอก 5-10V DC
5. สามารถพัฒนาโปรแกรมโดยใช้โปรแกรม Arduino IDE ได้
6. ขนาดของบอร์ด 25.4 mm x 48.26 mm

2.3 เซนเซอร์ตรวจวัดอุณหภูมิ

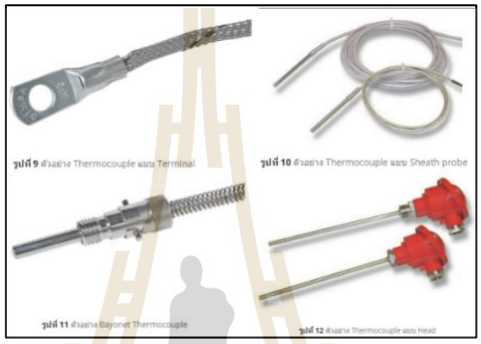
เซนเซอร์ตรวจวัดอุณหภูมิ เริ่มต้นมาจากความต้องการในอุตสาหกรรมเครื่องปรับอากาศ ต่อมาจึงได้มีการพัฒนาเซนเซอร์ตรวจวัดที่มีคุณสมบัติหลายอย่าง (Multisensor) การตรวจวัดอุณหภูมิใช้รูปแบบการเปลี่ยนแปลงระดับแรงดันไฟฟ้าจากสัญญาณอนาล็อกไปสู่สัญญาณดิจิตอล โดยสัมพันธ์กับอุณหภูมิ โดยมีรูปแบบใหญ่ ๆ ของเซนเซอร์อยู่ด้วยกัน 4 รูปแบบคือ

2.3.1 เทอร์โมคัปเปิล (Thermocouples)

เซนเซอร์ตรวจวัดอุณหภูมิ ที่อาศัยความแตกต่างของอุณหภูมิในการสร้างแรงเคลื่อนไฟฟ้าขึ้น การที่แรงเคลื่อนไฟฟ้าค่าหนึ่งจะอ้างอิงเป็นอุณหภูมิค่าหนึ่งได้ แสดงว่าความแตกต่างของอุณหภูมิที่เกิดขึ้นนั้นจะต้องอ้างอิงกับอุณหภูมิค่าคงที่ค่าหนึ่งเสมอ โดยเรียกอุณหภูมิคงที่ใช้อ้างอิงนี้ว่า Reference Junction และได้มีการกำหนด Reference Junction ให้เป็น 0 °C เพื่อให้การวัดอุณหภูมิเกิดแรงเคลื่อนไฟฟ้าที่เป็นมาตรฐานเดียวกัน และกำหนดเป็นตารางมาตรฐานแสดงค่าอุณหภูมิเทียบกับแรงเคลื่อนไฟฟ้าที่วัดได้ แต่โดยทั่วไป เทอร์โมคัปเปิลจะทำการวัดที่อุณหภูมิห้อง (เช่น 25 °C) นั่นคือไม่ได้เทียบกับ 0 °C แสดงว่าค่าแรงเคลื่อนไฟฟ้าที่ได้ยังไม่ถูกต้อง หากนำไปอ่านค่าอุณหภูมิจากตารางมาตรฐานจะผิดพลาด จึงจำเป็นต้องมีการรักษา Reference Junction เพื่อให้การวัดอุณหภูมิเทียบกับ 0 °C ตลอดเวลา การรักษา Reference Junction ด้วยน้ำแข็งบริสุทธิ์ ดังรูปที่ 2.8 และตัวอย่างของ Thermocouples ดังรูปที่ 2.9



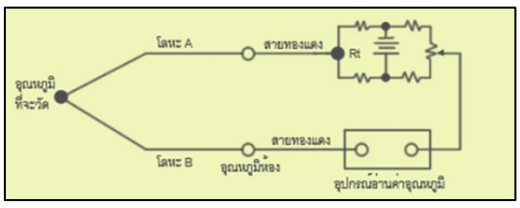
รูปที่ 2.8 หลักการทำงานของ Thermocouples



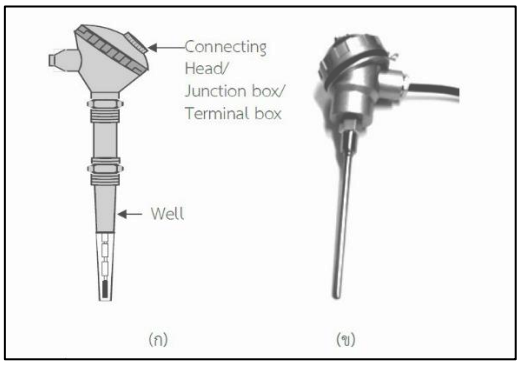
รูปที่ 2.9 ตัวอย่าง Thermocouples

2.3.2 Resistance Temperature Detector (RTD)

เซนเซอร์อุณหภูมิที่อาศัยหลักการเปลี่ยนแปลงค่าความต้านทานของโลหะ โดยความต้านทานจะมีค่าเพิ่มขึ้นตามอุณหภูมิ เรียกว่า สัมประสิทธิ์การเปลี่ยนแปลงอุณหภูมิเชิงบวกดังรูปที่ 2.10 นิยมนำไปใช้ในการวัดอุณหภูมิในช่วง -270 to 850 °C วัสดุที่นำมาใช้จะเป็นโลหะที่มีความต้านทานจำเพาะต่ำ เช่น ทังสเตน นิกเกิล แพลทินัม และลักษณะตัวอย่าง RTD ดังรูปที่ 2.11



รูปที่ 2.10 หลักการทำงานของ Resistance Temperature Detector



รูปที่ 2.11 ตัวอย่าง Resistance Temperature Detector

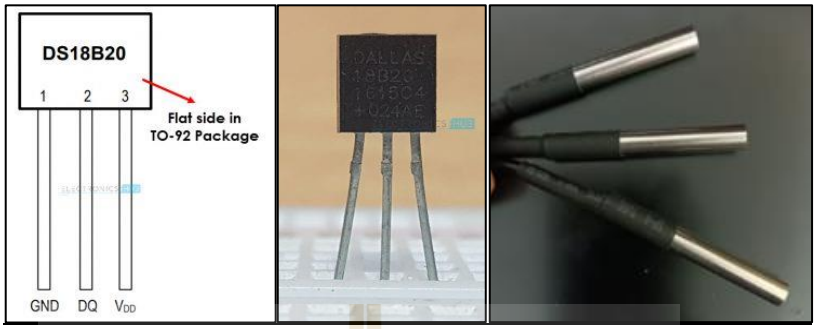
2.3.3 Thermistor

เป็นอุปกรณ์ความต้านทานชนิดที่สามารถเปลี่ยนค่าความต้านทานเมื่อได้รับความร้อน โดยที่ค่าความต้านทานจะเปลี่ยนแปลงแบบไม่เป็นเชิงเส้นกับอุณหภูมิ ตัวอย่าง Thermistor ดังรูปที่ 2.12 แบ่งเป็น 2 ลักษณะ คือ Positive Temperature Comitial (PTC) เป็นชนิดที่ปกติจะมีค่าความต้านทานต่ำ เมื่อได้รับความร้อนจะทำให้มีค่าความต้านทานสูงขึ้นตามลำดับอุณหภูมิ นำไปใช้ตรวจสอบระดับความร้อน หรือทำให้เกิดความร้อนขึ้นเพื่อควบคุมการจ่ายแรงดันไฟฟ้าให้กับขดลวด เช่น วงจรล้างสนามแม่เหล็กอัตโนมัติของเครื่องรับโทรทัศน์สี (Degaussing coil) เป็นต้น

Negative Temperature Comitial (NTC) เป็นชนิดที่ปกติจะมีความต้านทานสูงเมื่อได้รับความร้อน ค่าความต้านทานจะต่ำลง ใช้งานด้านการตรวจสอบความร้อนเพื่อควบคุมระดับการทำงาน เช่น ในวงจรขยายเสียงที่ดีใช้ตรวจจับความร้อนที่เกิดจากการทำงานแล้วป้อนกลับไปลดการทำงานของวงจรให้น้อยลง เพื่ออุปกรณ์หลักจะไม่เกิดความร้อนมากจนเกินไป

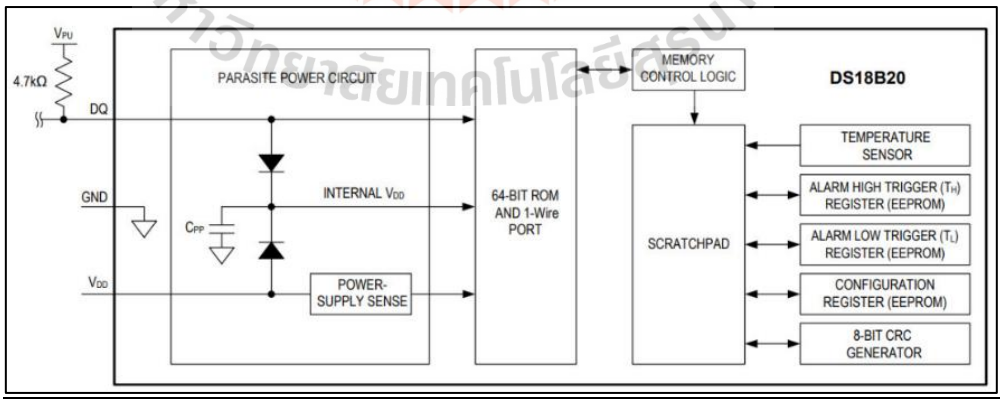
รูปที่ 2.12 ตัวอย่าง Thermistor

2.3.4 Digital temperature sensor

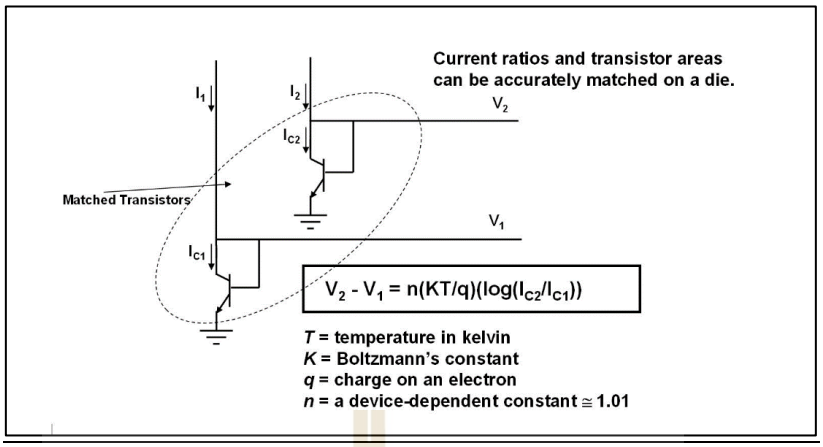


รูปที่ 2.13 ตัวอย่าง Digital temperature sensor

เซนเซอร์อุณหภูมิแบบดิจิตอล DS18B20 คือ เซนเซอร์อุณหภูมิแบบดิจิตอลสามขา ดังรูปที่ 2.13 ประกอบไปด้วยขาไฟเลี้ยง 3-5v ขากราวด์ และขาข้อมูลที่ใช้โปรโตคอลแบบสายเดี่ยวและสามารถใช้วัดอุณหภูมิในช่วง -67°F ถึง +257°F หรือ -55°C ถึง +125°C ด้วยความแม่นยำ ± 5% เครื่องวัดอุณหภูมิดิจิตอล DS18B20 ให้การอ่านอุณหภูมิ 9 ถึง 12 บิต (กำหนดค่าได้) ซึ่งระบุอุณหภูมิของอุปกรณ์ มันสื่อสารผ่านบัส 1-Wire ฟังก์ชันหลักของ DS18B20 คือเซนเซอร์อุณหภูมิแบบ Direct-to-Digital ความละเอียดของเซนเซอร์อุณหภูมิที่ผู้ใช้กำหนดค่าได้คือ 9, 10, 11 หรือ 12 บิต ซึ่งสอดคล้องกับการเพิ่มขึ้นทีละ 0.5°C, 0.25°C, 0.125°C และ 0.0625°C ตามลำดับ ความละเอียดเริ่มต้นเมื่อเปิดเครื่องคือ 12 บิต



รูปที่ 2.14 บล็อกไดอะแกรมของ DS18B20

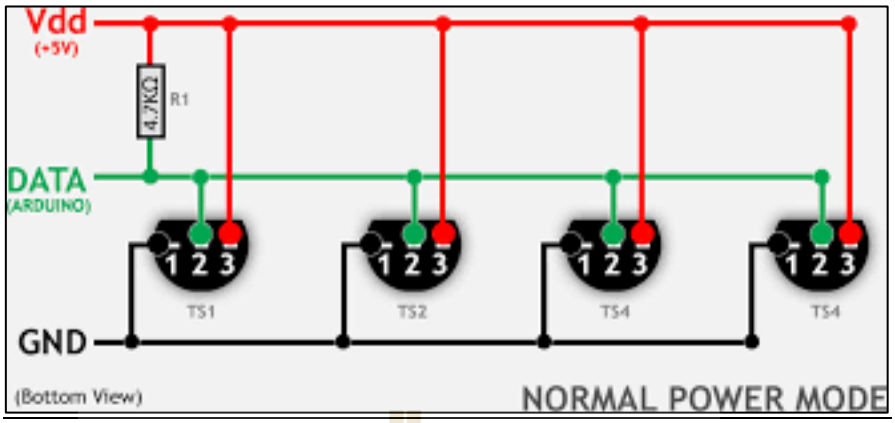


รูปที่ 2.15 หลักการทำงานของเซนเซอร์อุณหภูมิ DS18B20

จากรูปที่ 2.14 แสดงแผนภาพบล็อกการทำงานของเซนเซอร์อุณหภูมิ DS18B20 ประกอบด้วย วงจรจ่ายไฟ ROM 64 บิต ตัวควบคุมหน่วยความจำ เซนเซอร์อุณหภูมิหลัก และพื้นที่ Scratchpad ซึ่งมีตัวบันทึกอุณหภูมิและรีจิสเตอร์อื่น ๆ อีกสองสามตัวสำหรับการจัดเก็บการกำหนดค่าและการเตือนทริกเกอร์สูงและต่ำ ROM 64 บิตเก็บรหัสซีเรียลเฉพาะของอุปกรณ์ เอาต์พุตดิจิตอลจากเซนเซอร์อุณหภูมิจะถูกเก็บไว้ใน 2-Byte Temperature Register ยังประกอบด้วย 1 ไบต์สำหรับการลงทะเบียน Alarm HIGH Trigger, Alarm LOW Trigger register และ Configuration register แต่ละอัน ซึ่ง DS18B20 เป็นเซนเซอร์ตรวจวัดอุณหภูมิแบบใช้ช่องว่างแถบซิลิกอน เซนเซอร์มีพื้นฐานอยู่บนหลักการที่ว่าแรงดันตกคร่อมของซิลิคอนไดโอดขึ้นอยู่กับอุณหภูมิ โดยการวัดแรงดันตกคร่อมไปข้างหน้าในไดโอด จะคำนวณอุณหภูมิของซิลิกอนเซนเซอร์ใช้ทรานซิสเตอร์ที่เข้าคู่กันสองตัวพร้อมพฤติกรรมแรงดันไฟฟ้าที่ทราบทั่วอุณหภูมิ ความแตกต่างของแรงดันไฟฟ้าทั้งสองนั้นจะถูกถ่ายและแปลงเป็นค่าดิจิตอลดังแสดงในรูปที่ 2.15

วิธีการใช้งานเซนเซอร์อุณหภูมิ DS18B20 กับไมโครคอนโทรลเลอร์

เซนเซอร์ทำงานด้วยวิธีการสื่อสารแบบ 1-Wire ต้องการเพียงขาข้อมูลที่เชื่อมต่อกับไมโครคอนโทรลเลอร์ที่มีตัวต้านทานแบบดึงขึ้นและอีกสองขาใช้สำหรับจ่ายไฟตัวต้านทานแบบดึงขึ้นใช้เพื่อให้สายอยู่ในสถานะสูงเมื่อไม่ได้ใช้งานบัส ตามรูปที่ 2.16 ค่าอุณหภูมิที่วัดโดยเซนเซอร์จะถูกเก็บไว้ในรีจิสเตอร์ขนาด 2 ไบต์ภายในเซนเซอร์ ข้อมูลนี้สามารถอ่านได้โดยใช้วิธี 1-wire โดยส่งข้อมูลตามลำดับ มีคำสั่งสองประเภทที่จะส่งเพื่ออ่านค่า คำสั่งหนึ่งคือคำสั่ง ROM และอีกคำสั่งหนึ่งคือคำสั่งฟังก์ชัน ค่าแอดเดรสของหน่วยความจำ ROM แต่ละอันพร้อมกับลำดับ



รูปที่ 2.16 แผนภาพวงจรของอินเทอร์เฟซ DS18B20 หลายตัวในเทคโนโลยี 1-Wire

คุณสมบัติทั่วไปของ DS18B20 Digital temperature sensor

1. เซนเซอร์อุณหภูมิดิจิตอลที่ตั้งโปรแกรมได้
2. สื่อสารโดยใช้วิธี 1-Wire
3. แรงดันไฟฟ้าที่ใช้งานได้ 3V ถึง 5V
4. ช่วงอุณหภูมิ -55 °C ถึง +125 °C
5. ความแม่นยำ ±0.5°C
6. ความละเอียดเอาต์พุต 9 บิตถึง 12 บิต
7. ที่อยู่ 64 บิตที่ไม่ซ้ำกันทำให้สามารถมัลติเพล็กซ์ได้
8. เวลาในการแปลง 750ms ที่ 12-bit
9. ตัวเลือกการเตือนที่ตั้งโปรแกรมได้
10. มีทั้งแบบ To-92, SOP และเซนเซอร์แบบกันน้ำ
11. Operating voltage 3V to 5V

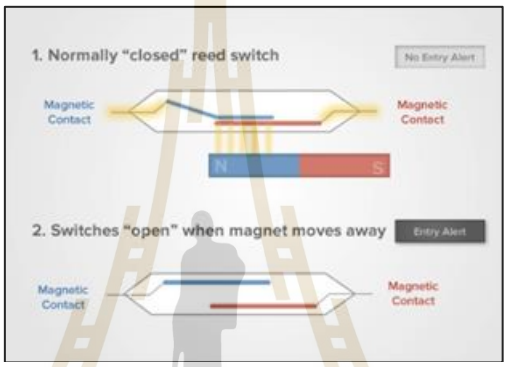
จุดเด่น ของ DS18B20 Digital temperature sensor

1. การตรวจวัดอุณหภูมิในสภาพแวดล้อมที่ยากลำบาก
2. การตรวจวัดอุณหภูมิของเหลว
3. แอปพลิเคชันที่ต้องตรวจวัดอุณหภูมิหลายจุด

2.4 เซนเซอร์ตรวจจับ (Proximity Sensors)

เซนเซอร์ตรวจจับ คือ เซนเซอร์ตรวจจับวัตถุ โดยปราศจากการสัมผัส ทำให้ทราบถึงตำแหน่งของวัตถุหรือสามารถระบุได้ว่ามีวัตถุใดผ่านเข้ามาในตำแหน่งที่กำหนดไว้หรือไม่ ส่วนใหญ่จะใช้กับงานตรวจจับ ตำแหน่ง ระดับ ขนาด และรูปร่าง

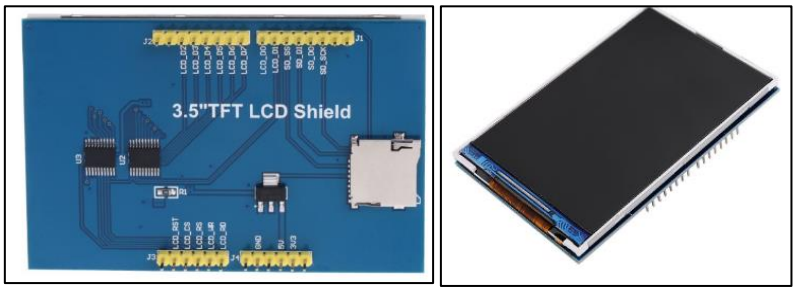
2.4.1 Magnetic Reed Switch คือ เซนเซอร์ตรวจจับเฉพาะแม่เหล็กหลักการทำงานดังรูปที่2.17 เมื่อมีแม่เหล็กอยู่ในระยะของ Magnetic Contact จะเชื่อมติดกัน และเมื่อสัมผัสแม่เหล็กไม่เจอ Magnetic Contact จะไม่เชื่อมติดกัน และตัวอย่าง Magnetic Reed Switch ดังรูป 2.18



รูปที่ 2.17 หลักการทำงานของ Magnetic sensor

รูปที่ 2.18 ตัวอย่าง Magnetic Reed Switch

2.5 จอแสดงผลภาพ (3.5 Inch TFT Color Screen Module 320x480)



รูปที่ 2.19 ตัวอย่าง จอแสดงผลภาพ 3.5 Inch TFT Color Screen Module 320x480

TFT Color Screen Module คือ โมดูลจอแสดงผลภาพ ดังรูป 2.19 ที่สามรถใช้ร่วมกับไมโครคอนโทรลเลอร์ได้ โดยที่ TFT LCD (Thin-Film-Translator Liquid Crystal Display) ประเภทนี้เรียกอีกอย่างว่า LCD แบบแอคทีฟเมทริกซ์ สามารถย้อนกลับบางพิกเซลในขณะที่ใช้พิกเซลอื่นได้ ส่งผลให้ใช้พลังงานเพียงเล็กน้อยในการทำงาน

จุดเด่นของจอแสดงผลภาพแบบ TFT LCD

1. ใช้พลังงานน้อยกว่ามากในการทำงานเมื่อใช้จากหน้าจอที่ใหญ่ขึ้น
2. สร้างภาพที่คมชัดที่จะไม่มีปัญหากับการมองเห็น
3. หน้าจอที่ใช้เทคโนโลยี TFT มีการออกแบบและรูปลักษณ์ที่น่าดึงดูดใจมาก

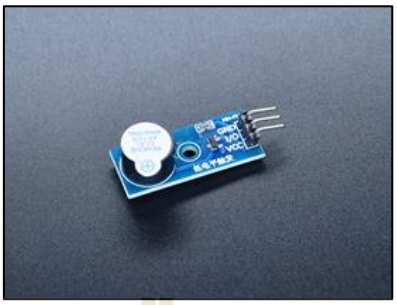
จุดด้อยของจอแสดงผลภาพแบบ TFT LCD

1. ไม่สามารถสร้างมุมมองภาพที่กว้างขึ้นได้ เป็น ภาพในหน้าจอ TFT จะบิดเบือน
2. หน้าจอ TFT สำหรับการพิมพ์ภาพ เทคโนโลยีไม่สามารถแสดงสีได้เหมือนรูปภาพจริง

คุณสมบัติทั่วไปของ TFT Color Screen Module

1. รองรับการใช้งานติดตั้งโดยตรงกับ Arduino Mega2560
2. หน้าจอสีขนาด 3.5 นิ้ว ความละเอียด 320x480 HD
3. ชิปแปลงระดับ OnBorad สำหรับ 3.3-5V MCU
4. เข้ากันได้กับระดับแรงดันไฟทำงาน 3.3-5V
5. เข้ากันได้กับ UTFT / UTFT Buttons / Utouch Library สำหรับ Arduino
6. มีช่องเสียบการ์ด SD ให้พร้อม
7. พร้อมวงจร SPI FLASH

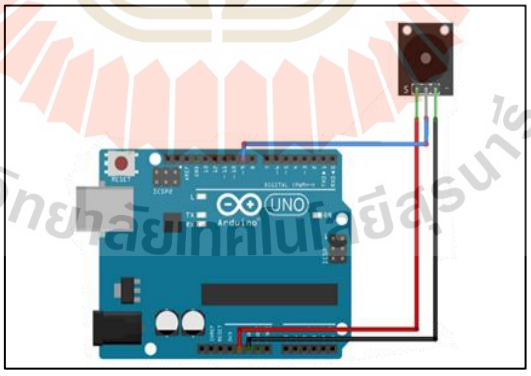
2.6 บัสเซอร์ โมดูล (Active Buzzer Module)



รูปที่ 2.20 ตัวอย่าง Active Buzzer Module

โมดูล Active Buzzer เป็นอุปกรณ์ส่งสัญญาณเสียง มีลักษณะดังรูป 2.20 อยู่ในรูปแบบของเพียโซอิเล็กทริกเครื่องกลหรือไฟฟ้า ผ่านขาเพียงยูนิตเดียว โมดูล Buzzer แบบแอคทีฟ 5V โมดูล Buzzer แอคทีฟชิ้นส่วนอิเล็กทรอนิกส์ 5V DC จะสร้าง เสียงโทนเดียวเมื่อได้รับความถี่สัญญาณสูง แต่สามารถใช้โมดูลออดแบบพาสซีฟเพื่อสร้างโทนเสียงต่าง ๆ ได้ ประกอบด้วยออดแบบเพียโซอิเล็กทริกพร้อมออสซิลเลเตอร์ในตัวเพื่อสร้างเสียง 2.5 kHz เมื่อสัญญาณสูง

วิธีการใช้งาน Active Buzzer Module กับไมโครคอนโทรลเลอร์



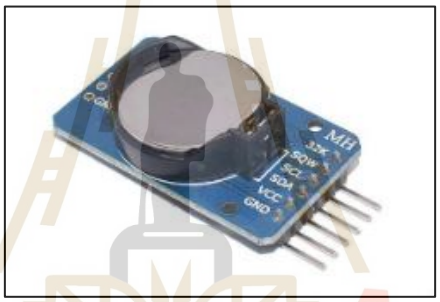
รูปที่ 2.21 แผนภาพวงจรของอินเทอร์เฟซ Arduino กับ Active Buzzer Module

เนื่องจากโมดูล Active Buzzer ได้รับการออกแบบมาเพื่อเปิดและปิด Buzzer จึงสามารถนำมาใช้ในวงจรสัญญาณเตือนและสัญญาณเสียงได้ โมดูลออดจะดึงกระแสมากกว่ากระแสสูงสุดของ Arduino ดังนั้นจึงจำเป็นต้องเข้าร่วม Buzzer กับArduinoโดยใช้ทรานซิสเตอร์ โดยการเชื่อมต่อสาย Data ต่อเข้า Digital pin 1 ต่อไฟเลี้ยงและกราวด์ ตามรูปที่ 2.21

คุณสมบัติทั่วไปของ Active Buzzer Module

1. แรงดันไฟฟ้า ที่ใช้งาน 3.5V - 5.5V
2. กระแสไฟสูงสุด 30mA 5V DC
3. เอาต์พุตเสียงขั้นต่ำ 85dB ที่ 10 ซม.
4. อุณหภูมิในการจัดเก็บ -30℃ ถึง 105℃
5. ความถี่เรโซแนนซ์ 2500Hz ± 300Hz
6. อุณหภูมิในการทำงาน -20 ℃ ถึง 70 ℃
7. ขนาด 18.5 มม.x15 มม.

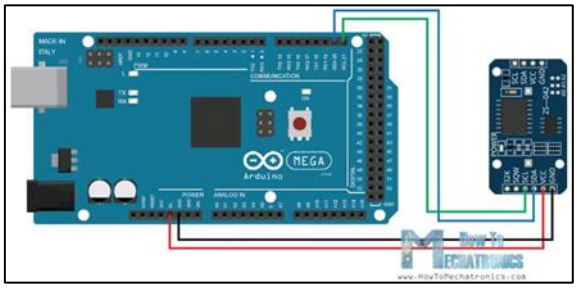
2.7 โมดูลนาฬิกา DS3231 (DS3231 module)



รูปที่ 2.22 ตัวอย่างโมดูลนาฬิกา DS3231

DS3231 Real Time Clock Module เป็นนาฬิกาเรียลไทม์ราคาถูกและแม่นยำสูง ซึ่งสามารถรักษาชั่วโมง นาที และวินาที ตลอดจนข้อมูลวัน เดือน และปีได้ มีการชดเชยอัตโนมัติสำหรับปีอธิกสุรทินและสำหรับเดือนที่มีเวลาน้อยกว่า 31 วัน มีลักษณะดังรูปที่ 2.22

วิธีการใช้งาน DS3231 Real Time Clock Module กับไมโครคอนโทรลเลอร์



รูปที่ 2.23 แผนภาพวงจรของอินเทอร์เฟซ Arduino กับ DS3231 Real Time Clock Module

จากรูปที่ 2.23 แผนภาพวงจรประกอบไปด้วย VCC pin และ GND pin สำหรับจ่ายไฟให้กับโมดูล และ pinการสื่อสาร I2C สองตัว, SDA และ SCL

คุณสมบัติทั่วไปของ DS3231 Real Time Clock Module

1. ทำงานได้ทั้ง 3.3 หรือ 5 V อินพุตแบตเตอรี่คือ 3V และแบตเตอรี่ CR2032 3V ทั่วไป
2. โมดูลนี้ใช้โปรโตคอลการสื่อสาร I2C

2.8 Pocket Wi-Fi

รูปที่ 2.24 ตัวอย่าง Pocket Wi-Fi

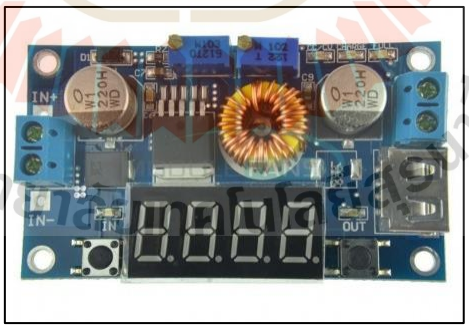
Pocket Wi-Fi คืออุปกรณ์ที่ทำงานโดยใช้ Subscriber Identity Module หรือที่เรียกกันทั่วไปว่า SIM การ์ด ซิมการ์ดเชื่อมต่อกับเครือข่ายมือถือหรือเซลลูลาร์ในลักษณะเดียวกับที่สมาร์ท

โฟนทำ เมื่ออุปกรณ์เชื่อมต่อกับอินเทอร์เน็ตผ่านซิมการ์ดแล้ว ก็สามารถใช้งานอินเทอร์เน็ตได้ ดังรูปที่ 2.24

คุณสมบัติทั่วไปของ Pocket Wi-Fi 4G

1. ขอบเขต 30 เมตร
2. มาตรการ 802.11n
3. รูปแบบ SIM card ใส่ซิมแล้วใช้งานได้ทันที
4. รองรับ4G/LTE FDD 2100/1800/900/800/850 MHz
5. รองรับ3G/DC-HSPA+/HSPA+/HSPA/UMTS 2100/1900/900/850 MHz
6. รองรับ2G/EDGE/GPRS/GSM 1900/1800/900/850 MHz
7. ความเร็ว 150Mbps บนเครือข่าย4G/LTE FDD (MAX)
8. ความเร็ว 42Mbpsบนเครือข่าย3G/HSPA+/DC-HSPA+ (MAX)
9. รองรับ Wi-Fi b/g/nบน2.4Ghz ใช้งานได้พร้อมกัน10เครื่อง
10. มีช่อง micro USB/ Micro SD card
11. มีแบตเตอรี่ภายในตัว ขนาด 2,100 mAh

2.9 Constant current/voltage 5-36V 3A buck regulator with Display



รูปที่ 2.25 ตัวอย่าง Constant current/voltage 5-36V 3A buck regulator with Display

Constant current/voltage 5-36V 3A buck regulator with Display คืออุปกรณ์แปลงแรงดันไฟฟ้าจากสูงให้ต่ำลงโดยช่วงแรงดันไฟฟ้าอินพุตของตัวแปลง คือ DC 5V-36V ช่วงแรงดันเอาต์พุตคือ DC 1.25V-32V ปรับต่อเนื่องได้ พร้อมกับมัลติมิเตอร์โมดูลควบคุมแรงดันไฟฟ้า สามารถใช้งานได้โดยตรงโดยไม่ต้องใช้มัลติมิเตอร์ภายนอก ตัวอย่างดังรูปที่ 2.25

คุณสมบัติทั่วไปของ Constant current/voltage 5-36V 3A buck regulator with Display

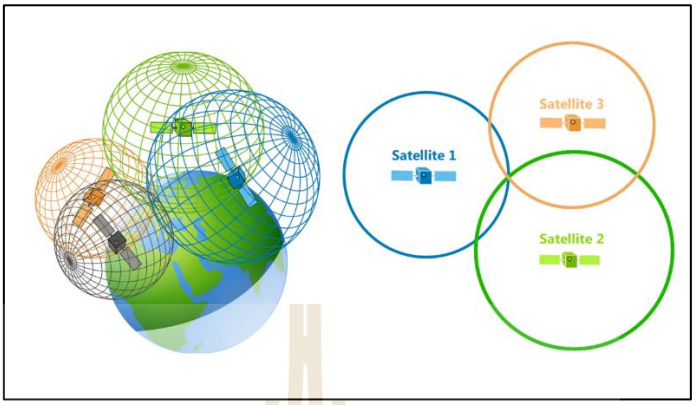
1. ช่วงแรงดันไฟฟ้า อินพุต5-36VDC
2. ช่วงแรงดันไฟฟ้าขาออก1.25-32VDC ปรับได้
3. กระแสไฟขาออก 0-5A
4. กำลังขับ 75W
5. ประสิทธิภาพสูงถึง 96%
6. ฟังก์ชันปิดระบบระบายความร้อนในตัว
7. สร้างขึ้นในฟังก์ชันจำกัดกระแส
8. มีฟังก์ชันป้องกันการลัดวงจรของเอาต์พุตในตัว

2.10 ระบบการหาตำแหน่งทั่วโลก GPS (Global Positioning System)

2.10.1 ประวัติความเป็นมาของ GPS

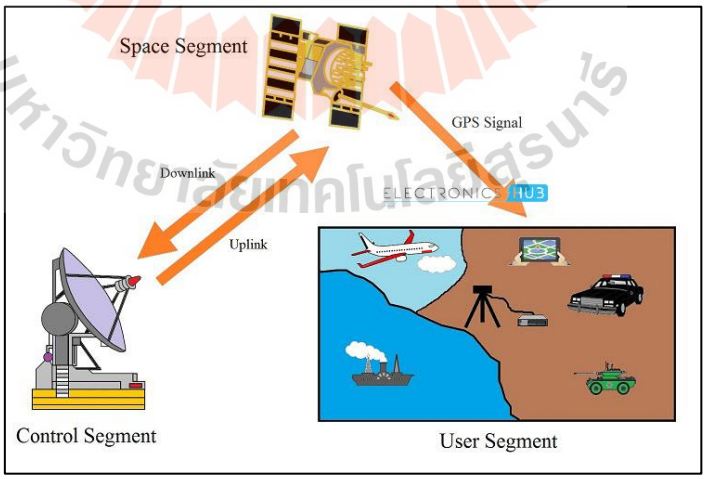
ระบบ GPS เป็นระบบการบอกตำแหน่งพิกัดทางภูมิศาสตร์บนโลกจากอุปกรณ์รับส่งสัญญาณที่ทำงานร่วมกับระบบดาวเทียมกว่า 30 ดวง โคจรอยู่เหนือพื้นดินที่ระดับความสูงกว่า 20,000 กิโลเมตร โดยจะมีดาวเทียมอย่างน้อย 3-4 ดวง ทำหน้าที่รับส่งสัญญาณกับอุปกรณ์ GPS เพื่อระบุตำแหน่งพิกัด ละติจูด (Latitude) ลองจิจูด (Longitude) และระดับความสูงจากระดับน้ำทะเล (Altitude) อยู่ตลอดเวลา และเป็นระบบดาวเทียมนำทางทั่วโลก (GNSS) ที่ให้การระบุตำแหน่ง การนำทาง และระบบจับเวลา (PNT) ระบบ GPS เริ่มต้นขึ้นตั้งแต่ปี 1957 จากนักวิทยาศาสตร์ชาวสหรัฐฯ กระทั่งถึงปี 1960 ได้เริ่มต้นการทดสอบใช้งานจริงภายในกองทัพเรือของสหรัฐฯ ต่อมาในปี 1983 เกิดเหตุการณ์เครื่องบินโคเรียนแอร์ไลน์ เที่ยวบินที่ 007 ของเกาหลีใต้ได้เกิดการพัดหลงเข้าไปยังน่านฟ้าของสหภาพโซเวียตก่อนถูกยิงตกลงมา จำนวนผู้โดยสาร 269 คนเสียชีวิตทั้งหมด นั่นทำให้ประธานาธิบดีโรนัลด์ เรแกน ของสหรัฐฯ ออกมาประกาศว่า หากระบบ GPS มีการพัฒนาจนเสร็จสมบูรณ์เขาจะอนุญาตให้ประชาชนทั่ว ๆ ไป ได้ใช้งานกัน นั่นจึงทำให้เรื่องของ GPS ค่อย ๆ ถูกพัฒนาให้เข้ามาสู่เชิงพาณิชย์ และในยุคปัจจุบันยังมีระบบนำทางด้วยดาวเทียมอื่น ๆ เช่น GLONASS ของรัสเซีย Galileo ของยุโรปและ BeiDou ของจีน แต่ Global Positioning System (GPS) ของสหรัฐอเมริกาและ Russian Global Navigation Satellite System (GLONASS) เป็นระบบนำทางด้วยดาวเทียมที่ทำงานได้อย่างสมบูรณ์เพียงระบบเดียวที่มีกลุ่มดาว 32 ดวงและ ดาวเทียม 27 ดวง ตามลำดับ

2.10.2 หลักการทำงาน GPS



รูปที่ 2.26 หลักการการคำนวณหาพิกัดตำแหน่งของGPS

จากรูปที่ 2.26 ในการคำนวณหาพิกัดตำแหน่ง 2 มิติของคุณ (ละติจูดและลองติจูด) และติดตามการเคลื่อนที่ ตัวรับสัญญาณ GPS ต้องถูกล็อกเข้ากับสัญญาณของดาวเทียมอย่างน้อย 3 ดวง และด้วยดาวเทียม 4 ดวงขึ้นไป ตัวรับสัญญาณจะสามารถระบุตำแหน่ง 3 มิติของคุณ (ละติจูด ลองติจูด และระดับความสูง) โดยทั่วไปแล้ว ตัวรับสัญญาณ GPS จะติดตามดาวเทียม 8 ดวงขึ้นไป แต่นั่นก็ขึ้นอยู่กับเวลาในแต่ละวันและสถานที่บนโลกที่คุณอยู่



รูปที่ 2.27 หลักการทำงาน GPS ในการหาพิกัดตำแหน่ง

จากรูปที่ 2.27 แสดงถึงหลักการทำงาน GPS ในการหาพิกัดตำแหน่ง โดยมีการทำงานร่วมกันทั้งหมด 3 ส่วน คือ

2.10.2.1 ดาวเทียม (Space Segment) เป็นส่วนของอากาศ

Space Segment (SS) ของ GPS ประกอบด้วยกลุ่มดาวดาวเทียม 24 ดวงที่โคจรรอบโลกในวงโคจรเป็นวงกลมโดยประมาณ ดาวเทียมถูกวางในระนาบการโคจร 6 ระนาบ โดยแต่ละระนาบการโคจรประกอบด้วยดาวเทียมสี่ดวงความเอียงของระนาบโคจรและตำแหน่งของดาวเทียมถูกจัดเรียงในลักษณะเฉพาะเพื่อให้ดาวเทียมอย่างน้อยหกดวงอยู่ในแนวสายตาจากตำแหน่งใด ๆ บนโลกเสมอ เมื่อมาถึงการจัดเรียงกลุ่มดาวในอวกาศ ดาวเทียม GPS จะถูกวางไว้ใน Medium Earth Orbit (MEO) ที่ระดับความสูงประมาณ 20,000 กม. เพื่อเพิ่มความซ้ำซ้อนและปรับปรุงความแม่นยำ จำนวนดาวเทียม GPS ในกลุ่มดาวทั้งหมดได้เพิ่มขึ้นเป็น 32 ดวง โดยในจำนวนนี้มีดาวเทียมทั้งหมด 31 ดวงที่ปฏิบัติการอยู่สัญญาณ GPS ประกอบไปด้วยข้อมูลที่แตกต่างกัน 3 ชนิด ดังนี้

1. รหัสสุ่มเทียม คือรหัสระบุตัวตนที่ระบุว่าดาวเทียมดวงไหนกำลังถ่าย ทอดข้อมูลจะสามารถดูได้ว่าดาวเทียมดวงไหนที่ได้รับสัญญาณมาบนหน้าดาวเทียมของอุปกรณ์

2. ข้อมูลปฏิทินดาวเคราะห์ เป็นข้อมูลที่จำเป็นเพื่อระบุตำแหน่งดาวเทียม และมอบข้อมูลสำคัญเกี่ยวกับสภาพของดาวเทียม วัน และเวลาปัจจุบัน

3. ข้อมูลปูม บอกตัวรับสัญญาณ GPS ว่าดาวเทียมดวงไหนจะไปอยู่ ตรงไหนในเวลาไหนตลอดทั้งวัน และแสดงข้อมูลการโคจรสำหรับดาวเทียมดวงดังกล่าวและดาวเทียมดวงอื่น ๆ ทั้งหมดในระบบ

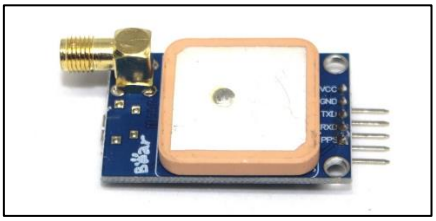
2.10.2.2 สถานีภาคพื้นดิน (Control Segment) เป็นส่วนควบคุม

ส่วนควบคุม (CS) ของ GPS ประกอบด้วยเครือข่ายสถานีตรวจสอบและควบคุมและติดตามทั่วโลก งานหลักของส่วนควบคุมคือการติดตามตำแหน่งของดาวเทียม GPS และดูแลให้อยู่ในวงโคจรที่เหมาะสมโดยใช้คำสั่งการหลบหลีก นอกจากนี้ ระบบควบคุมยังกำหนดและรักษาความสมบูรณ์ของระบบออนบอร์ด สภาพบรรยากาศ ข้อมูลจากนาฬิกาอะตอมและพารามิเตอร์อื่น กลุ่มควบคุม GPS ถูกแบ่งออกเป็นสี่ระบบย่อยดังนี้ สถานีควบคุมหลักใหม่ (NMCS) สถานีควบคุมหลักสำรอง (AMCS) เสาอากาศภาคพื้นดิน (GA) สี่ตัว และเครือข่ายสถานีตรวจสอบ (MSs) ทั่วโลก

2.10.2.3 อุปกรณ์สัญญาณ GPS (User Segment) เป็นส่วนของการใช้งาน

ส่วนผู้ใช้ของระบบ GPS ประกอบด้วยผู้ใช้ปลายทางของเทคโนโลยี เช่น พลเรือนและการทหารสำหรับการนำทาง การวางตำแหน่งและเวลาที่แม่นยำหรือมาตรฐาน โดยทั่วไปในการเข้าถึงบริการ GPS ผู้ใช้จะต้องติดตั้งเครื่องรับ GPS เช่น โมดูล GPS แบบสแตนด์อะโลน โทรศัพท์มือถือที่เปิดใช้งาน GPS และคอนโซล GPS โดยเฉพาะอุปกรณ์รับสัญญาณ GPS ที่ใช้ในโครงงาน

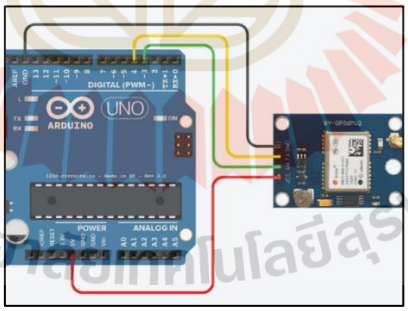
2.10.3 โมดูล GPS Ublox NEO-7M



รูปที่ 2.28 ตัวอย่างโมดูล GPS Ublox NEO-7M

โมดูล GPS Ublox NEO-7M คือ โมดูลเครื่องรับ GPS ดังรูปที่ 2.28 เป็นอุปกรณ์ที่ใช้ระบุตำแหน่งของดาวเทียมหลายดวงคำนวณ ระยะทางจากกันและกัน แล้วคำนวณระยะทางจากดาวเทียมเหล่านั้นเอง กระบวนการระบุตำแหน่งนี้เป็นสิ่งที่ทำให้เครื่องรับ GPS มีความแม่นยำมาก โดยนอกจากมีระบบ GPS หลักแล้ว ยังสนับสนุนระบบ GLONASS (ระบบของรัสเซีย) ด้วย ในการรันครั้งแรก จะใช้เวลาประมาณ 10 นาที เพื่อรอ GPS

การใช้งานโมดูล GPS Ublox NEO-7M กับ Arduino



รูปที่ 2.29 การอินเทอร์เฟซ Arduino กับ โมดูล GPS Ublox NEO-7M

การใช้งานโมดูล GPS Ublox NEO-7M กับ บอร์ด Arduino ทำได้โดยการต่อขาRx และ Tx เข้ากับขา Digital ของบอร์ด Arduino ต่อขา VCC กับขาไฟเลี้ยง 5V ของบอร์ด Arduino และต่อขา GND เข้ากับ GND ของบอร์ด Arduino ดังรูปที่ 2.29

จุดเด่นของ โมดูล GPS Ublox NEO-7M

1. สามารถตั้งค่าพารามิเตอร์ผ่านพอร์ตอนุกรมและบันทึกใน EEPROM
2. ด้วยอินเทอร์เฟซ SMA คุณสามารถเชื่อมต่อเสาอากาศได้หลากหลาย
3. สำรองแบตเตอรี่แบบชาร์จไฟได้บนเครื่องด้วย micro USB
4. มีเสาอากาศเซรามิกออนบอร์ด

คุณสมบัติทั่วไปของโมดูล GPS Ublox NEO-7M

1. แรงดันไฟฟ้า 3.3 ถึง 5 VDC (หรือโดยสาย USB)
2. การเชื่อมต่อ 5 VCC , GND , TX, RX
3. อัตราบอร์ดเริ่มต้น 9600
4. ขนาด 4 x 2.5 x 1.5 เซนติเมตร
5. น้ำหนัก 15 กรัม

2.10.4 GPS Antenna

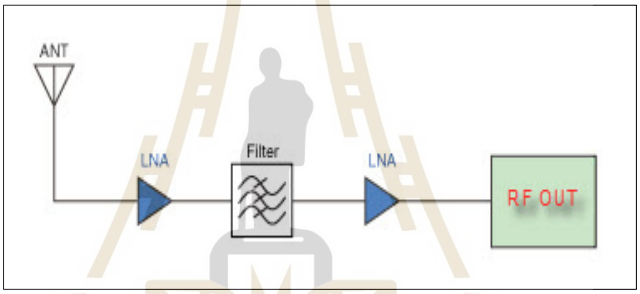
รูปที่ 2.30 เสาอากาศ GPS แบบ Passive Antenna

รูปที่ 2.31 เสาอากาศ GPS แบบ Active Antenna

เสาอากาศ GPS เป็นอุปกรณ์ที่รับสัญญาณวิทยุจากดาวเทียม GPS ที่ความถี่ต่างกัน เสาอากาศขยายสัญญาณและแปลงเป็นสัญญาณอิเล็กทรอนิกส์เพื่อให้เครื่องรับ GPS สามารถตีความได้ จากนั้นเครื่องรับ GPS จะใช้สัญญาณเหล่านี้ในการประเมินตำแหน่งของเครื่องรับอย่างแม่นยำ ประเภทของเสาอากาศ GPS แบ่งออกเป็น 2 ประเภท ดังนี้

1.Pasive Antenna คือ เสาอากาศ GPS ที่ไม่มีวงจรภายใน ทำด้วยเซรามิค และโลหะมีสายนำสัญญาณ ต่อตรงมาที่ วงจรภาครับของ GPS ได้โดยตรง ข้อดีของสายอากาศแบบ Passive Antenna คือ ไม่ต้องใช้ไฟเลี้ยง ดังรูปที่ 2.30

2. Active Antenna คือ เสาอากาศ ที่มีวงจรขยายสัญญาณรบกวนต่ำ อยู่ข้างใน เรียกว่า Low Noise Amplifier เรียกสั้น ๆ ว่า LNA จำเป็นต้องใช้ไฟเลี้ยง หากไม่ต่อไฟเลี้ยง วงจร LNA จะไม่ทำงานส่งผลให้ สายอากาศรับสัญญาณไม่ได้ ดังรูปที่ 2.31



รูปที่ 2.32 วงจรพื้นฐานของ Active Antenna

ข้อดีของ Active Antenna คือ รับสัญญาณ GPS และ ขยายสัญญาณให้ชัดเจน ก่อนส่งไปให้ภาค RF ของ GPS ซึ่งส่งผลให้ GPS รับสัญญาณ ได้ดีขึ้น FIX สัญญาณได้เร็วขึ้น แต่ โมดูล จีพีเอส จะต้องมีวงจรจ่ายไฟเลี้ยงออกมากับสายนำสัญญาณเพื่อเลี้ยง Active Antenna ดังรูปที่ 2.32

2.10.5 ปัจจัยที่ส่งผลต่อสัญญาณ GPS

1. ความล่าช้าของชั้นบรรยากาศไอโอโนสเฟียร์และโทรโพสเฟียร์ สัญญาณดาวเทียมจะช้าลงเมื่อผ่านชั้นบรรยากาศดังกล่าว ระบบ GPS จะใช้โมเดลในตัวในการแก้ไขความผิดพลาดชนิดนี้บางส่วน

2. สัญญาณกระจายหลายเส้นทาง สัญญาณ GPS อาจสะท้อนจากวัตถุเช่น ตึกสูง หรือเนินใหญ่ที่ปรากฏก่อนจะไปถึงตัวรับสัญญาณ ซึ่งจะเพิ่มเวลาในการเดินทางของสัญญาณ และก่อให้เกิดข้อผิดพลาดได้

3. ความผิดพลาดจากนาฬิกาตัวรับสัญญาณ นาฬิกาในตัวรับสัญญาณอาจมีการระบุเวลาผิดพลาดเล็กน้อยเนื่องจากแม่นยำน้อยกว่านาฬิกาปรมาณูในดาวเทียม GPS

4. ความผิดพลาดของการโคจร ตำแหน่งดาวเทียมที่อาจจะไม่แม่นยำ

5. จำนวนดาวเทียมที่อยู่ในทัศนวิสัย ยิ่งมีดาวเทียมหลายดวงที่อยู่ในทัศนวิสัยของตัวรับ GPS ก็ยิ่งแม่นยำขึ้น เมื่อสัญญาณถูกกั้น คุณอาจได้รับตำแหน่งที่ผิดพลาดหรือไม่สามารถอ่านตำแหน่งได้เลย หน่วย GPS ส่วนใหญ่มักจะทำงานใต้น้ำหรือใต้ดินไม่ได้ แต่ตัวรับสัญญาณรุ่นใหม่ที่ไวต่อสัมผัสสูงนั้นสามารถติดตามสัญญาณบางชนิดเมื่ออยู่ในอาคาร

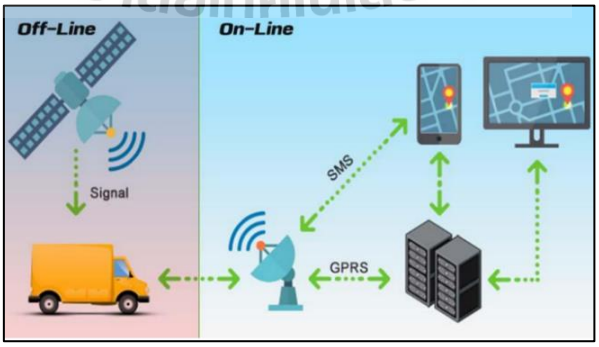
6. เรขาคณิต/แสงเงาของดาวเทียม สัญญาณดาวเทียมจะมีประสิทธิภาพมากขึ้นเมื่อดาวเทียมอยู่ในตำแหน่งมุมกว้างสัมพันธ์กับดวงอื่น ๆ มากกว่าเรียงเป็นเส้นตรงหรือเกาะกลุ่มกันแคบ ๆ

7. การเลือกให้บริการ ครั้งหนึ่งกระทรวงกลาโหมของสหรัฐฯได้ใช้มาตรการการเลือกให้บริการ (Selective Availability) กับดาวเทียมต่าง ๆ ทำให้สัญญาณแม่นยำน้อยลงเพื่อป้องกันไม่ให้ศัตรู ใช้สัญญาณ GPS ความแม่นยำสูงได้ รัฐบาลยกเลิกมาตรการการเลือกให้บริการในเดือนพฤษภาคม ปี 2000 ส่งผลให้การรับสัญญาณ GPS ของพลเรือนแม่นยำขึ้น

2.10.6 ประเภทของการใช้งาน GPS ในปัจจุบัน

1.GPS Navigator คือ อุปกรณ์และระบบนำทาง เป็น GPS ที่เราคุ้นเคยกันมาก เนื่องจากจะใช้งานกับรถยนต์ทั่วไปสำหรับบอกแผนที่เส้นทางต่าง ๆ ที่เราต้องการเดินทางไปแต่ไม่แน่ใจว่าต้องเดินทางผ่านถนนเส้นไหน เลี้ยวเข้าซอยอะไร การทำงานหลักของ GPS คือ ต้องมีการบอกพิกัดตำแหน่งปลายทางที่เราต้องการไปเมื่อบอกพิกัดเรียบร้อย GPS จะทำการคำนวณเส้นทางที่เหมาะสมพร้อมระยะทางจากจุดที่คุณอยู่และบอกเวลาได้ด้วยว่าต้องใช้ประมาณกี่นาทีจึงจะถึงจุดหมายปลายทางตามต้องการ

2.GPS Tracking System คือ อุปกรณ์ระบบติดตามยานพาหนะสามารถแยกย่อยออกได้อีก 2 แบบ ดังรูปที่ 2.33

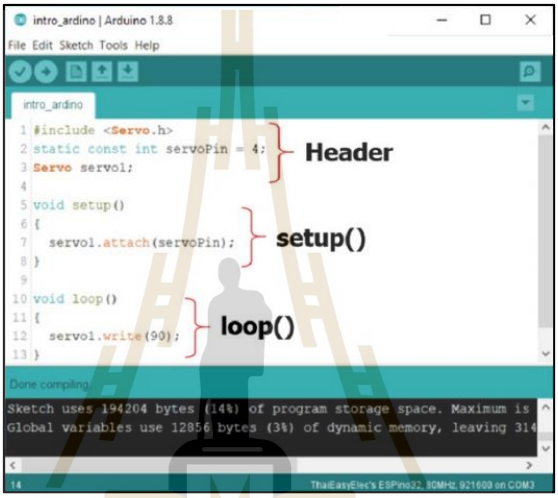


รูปที่ 2.33 หลักการทำงาน GPS Tracking System

1.อุปกรณ์ติดตามรถแบบออฟไลน์ เอาไว้ใช้เพื่อตรวจสอบประวัติการเดินทางแต่จะตรวจสอบเรื่องของตำแหน่งที่ตั้งของที่อยู่เครื่อง GPS นั้น ๆ ไม่ได้

2.อุปกรณ์ติดตามรถแบบออนไลน์ อุปกรณ์ตัวนี้ทำงานร่วมกับมือถือนั่นทำให้สามารถดูในส่วนของประวัติการเดินทางรวมถึงตำแหน่งที่ตั้งปัจจุบันของตัวอุปกรณ์ GPS ได้

2.11 โปรแกรม Arduino ide



รูปที่ 2.34 โปรแกรม Arduino ide

Arduino Integrated Development Environment (IDE) เป็นแอปพลิเคชันข้ามแพลตฟอร์ม (สำหรับ Windows, macOS, Linux) ที่เขียนด้วยฟังก์ชันจาก C และ C++ ใช้สำหรับเขียนและอัปโหลดโปรแกรมไปยังบอร์ดที่เข้ากันได้กับ Arduino บอร์ดพัฒนาผู้จำหน่ายรายอื่นด้วยซอร์สโค้ดสำหรับ IDE เผยแพร่ภายใต้ GNU General Public License เวอร์ชัน 2 Arduino IDE รองรับภาษา C และ C++ โดยใช้กฎพิเศษของการจัดโครงสร้างโค้ด Arduino IDE จัดหาไลบรารีซอฟต์แวร์จากโครงการ Wiring โค้ดที่ผู้ใช้เขียนขึ้นต้องการเพียงสองฟังก์ชันพื้นฐาน สำหรับการเริ่มต้นสเก็ตช์และลูปโปรแกรมหลัก ด้วยความนิยมที่เพิ่มขึ้นของ Arduino ในฐานะแพลตฟอร์มซอฟต์แวร์ ผู้ขายรายอื่นจึงเริ่มใช้คอมไพเลอร์และเครื่องมือโอเพ่นซอร์สแบบกำหนดเองที่สามารถสร้างและอัปโหลดไปยังไมโครคอนโทรลเลอร์อื่นได้

จากรูปที่ 2.34 แสดงถึงโครงสร้างโปรแกรม Arduino IDE ที่สำคัญมี 3 ส่วน คือ

1. Header เป็นส่วนแรกของโปรแกรมที่ถูกเรียกใช้งานจึงเป็นส่วนของโปรแกรมที่ใช้สำหรับเรียกใช้งานไลบรารี ประกาศตัวแปร กำหนดค่าของตัวแปร เป็นต้น

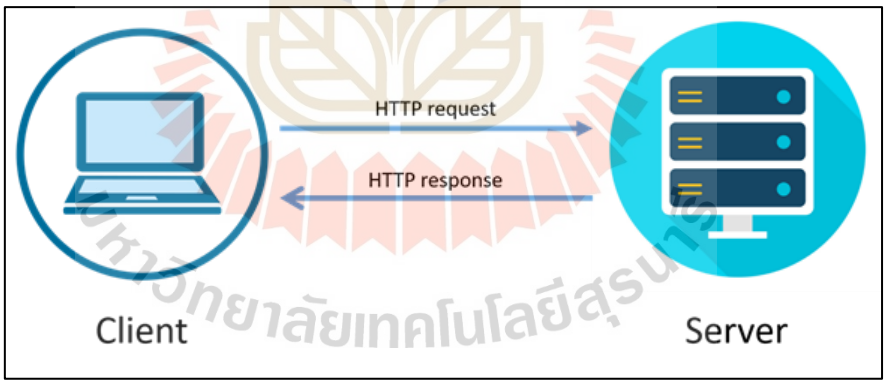
2. setup () เป็นฟังก์ชันบังคับสำหรับโปรแกรมที่พัฒนาด้วย Arduino IDE ฟังก์ชันนี้จะทำงานหลังจากทำงานในส่วนของการประกาศตัวแปร หรือเรียกใช้ไลบรารีในส่วน Header สำเร็จ ฟังก์ชันนี้จะทำงานเพียงครั้งเดียวจนกว่าจะมีการรีเซ็ตหรือจ่ายไฟให้บอร์ดใหม่อีกครั้ง

3. loop () เป็นฟังก์ชันบังคับสำหรับโปรแกรมที่พัฒนาด้วย Arduino IDE ฟังก์ชันนี้จะทำงานหลังจากฟังก์ชัน setup () ฟังก์ชันนี้จะวนรอบทำงานแบบไม่รู้จบ ตัวอย่างการใช้งานเช่น วนรอบตรวจสอบการกดสวิตซ์วนรอบนับจำนวนจากการอ่านค่าเซนเซอร์นับจำนวน วนรอบสั่งงานสถานะขา GPIO ของบอร์ด ESPino32 เป็นต้น

2.12 โปรโตคอล (Protocol)

โปรโตคอล คือ ข้อกำหนดหรือข้อตกลงในการสื่อสารระหว่างคอมพิวเตอร์ หรือภาษาสื่อสารที่ใช้เป็น ภาษากลางในการสื่อสารระหว่างคอมพิวเตอร์ด้วยกัน การที่เครื่องคอมพิวเตอร์ที่ถูกเชื่อมโยงกันไว้ในระบบจะสามารถติดต่อสื่อสารกันได้นั้น

2.12.1 HTTP ย่อมาจาก Hypertext Transfer Protocol คือ โปรโตคอลการส่งข้อความนี้ใช้ข้อความธรรมดาเพื่อส่งคำสั่งและข้อมูลโดยใช้ข้อความส่วนหัวซึ่งรวมถึงประเภทของคำขอที่ส่ง ประเภทเนื้อหา และข้อมูลเบราว์เซอร์

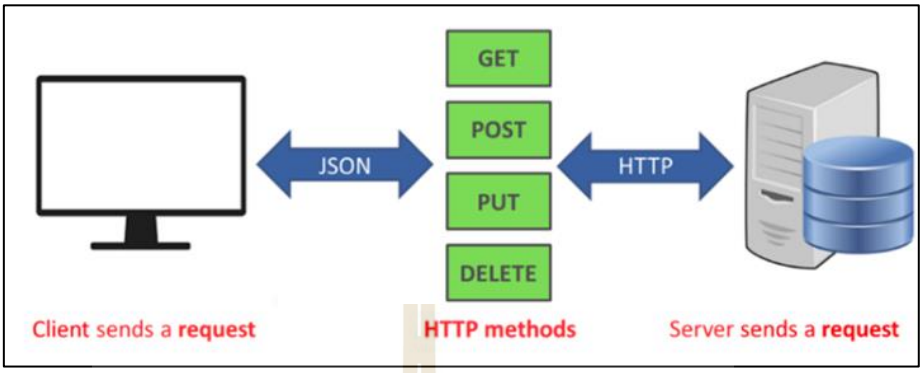


รูปที่ 2.35 หลักการทำงาน HTTP

จากรูปที่ 2.35 แสดงหลักการทำงานการสื่อสาร แบบ HTTP ซึ่งเกี่ยวข้องกับคำสำคัญสองคำคือ ไคลเอนต์และเซิร์ฟเวอร์

1. ไคลเอนต์ คือ ผู้ส่งคำขอ ตัวอย่างเช่น เบราว์เซอร์
2. เซิร์ฟเวอร์ คือ ผู้ที่ได้รับคำขอและส่งการตอบกลับ โดยทั่วไปเซิร์ฟเวอร์คือชิ้นส่วนของรหัสที่รับผิดชอบในการยอมรับคำขอและส่งการตอบกลับ

2.12.1.1 REST API



รูปที่ 2.36 การนำ REST API ไปใช้งาน

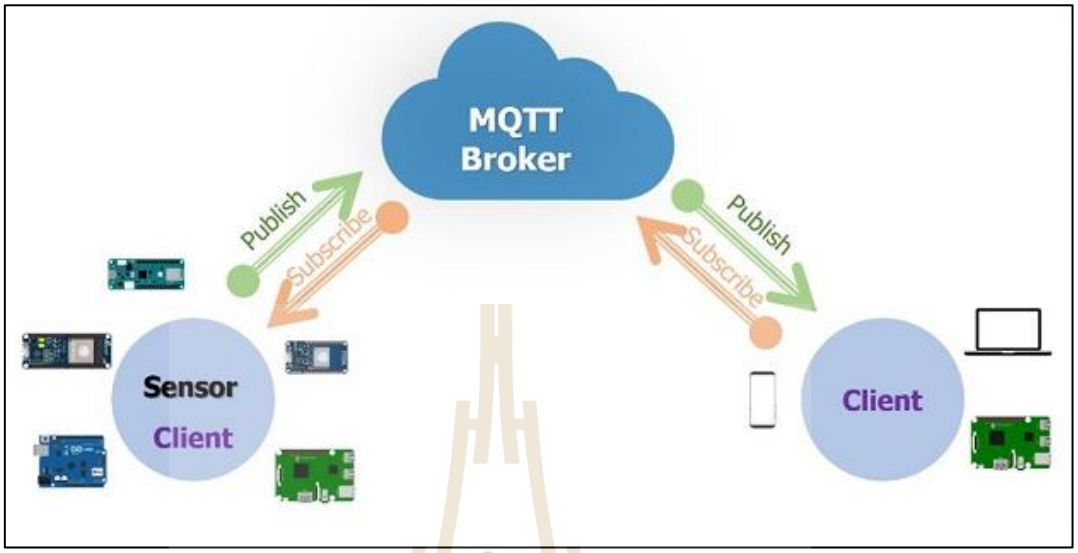
REST API คำว่า REST ย่อมาจาก Representational State Transfer และ API ย่อมาจาก application programming interface เป็นวิธีในการสร้าง web service โดยอาศัยรูปแบบของ HTTP Method โดยมีชุดคำสั่งสำคัญดังนี้ GET คือ เรียกดูข้อมูล POST คือ เพิ่มข้อมูล PUT คือ แก้ไขข้อมูล และ DELETE คือ ลบข้อมูล ในการทำงาน และจะส่งค่ากลับมาเป็น JSON หรือ XML ส่งผลให้สามารถรับส่งข้อมูลไปมาข้ามแพลตฟอร์ม ได้อย่างสะดวก เพราะเป็นการเรียกผ่าน HTTP Protocol ที่ใช้ในการเรียกใช้เว็บไซต์ การทำงานของ REST นั้นจะอาศัย URI/URL ของ request เพื่อเป็นตัวค้นหาและประมวลผลแล้วตอบกลับโดย response ดังรูปที่ 2.36

การรับส่งข้อมูลไปมาข้ามแพลตฟอร์ม คือการเก็บหรือรับส่งข้อมูลโดยที่ภาษาโปรแกรมส่วนใหญ่สามารถนำไปใช้งานได้ และมนุษย์ก็สามารถอ่านเข้าใจได้ แบ่งออกเป็น 2 แบบ

1. JSON เป็นการจัดเก็บข้อมูลมีโครงสร้างที่ไม่ซับซ้อน ทำให้เก็บข้อมูลนั้นสั้นกระชับ ไม่ต้องใช้พื้นที่ในการเก็บโครงสร้างของข้อมูลมากเกินความจำเป็นทำให้ได้ประโยชน์ตามมาคือสามารถนำไปใช้งานได้เร็ว โดยเฉพาะเมื่อนำไปใช้ใน JavaScript จะสามารถแปลงเป็น JavaScript Object และใช้งานได้ทันที

2. XML เป็นการเก็บข้อมูลโดยเก็บไว้ใน tag ที่จะต้องมี <tag> เปิดและ </tag> ปิด เหมือนกับ HTML ทำให้การเก็บข้อมูลแต่ละตัวต้องใช้พื้นที่มากขึ้น ยิ่งเป็นข้อมูลขนาดใหญ่ที่มีความความซับซ้อน โดยการนำไปใช้งานกับ JavaScript มีความยุ่งยากและช้ากว่า JSON โดยจะต้องทำการดึงข้อมูล XML Document และ ทำการวนซ้ำเพื่อ เข้าไปทำการเก็บข้อมูลมาใส่ตัวแปรไว้อีกที

2.12.2 MQTT



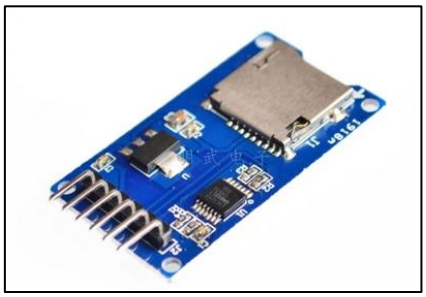
รูปที่ 2.37 หลักการทำงาน MQTT

MQTT (Message Queue Telemetry Transport) เป็นโปรโตคอลในการส่งข้อมูลที่พัฒนามาเพื่อใช้ในระบบ IOT มันทำงานแบบ Broker and Clients Network มันถูกออกแบบให้สามารถส่งข้อมูลแบบ Real-Time ในปริมาณข้อมูลที่น้อย ทำให้ใช้พลังงานต่ำมันถูกพัฒนามาจาก TCP/IP ที่มีการส่งข้อมูลแบบ One-To-One ทำให้สิ้นเปลืองทรัพยากรมากซึ่งไม่เหมาะกับระบบ IOT เนื่องจากในระบบ IOT มีการส่งข้อมูลตลอดเวลา และ 1 อุปกรณ์อาจรับหรือส่งข้อมูลไปยังหลายอุปกรณ์ หรือการส่งข้อมูลแบบ One-To-All โดยอุปกรณ์ทุกตัวที่ทำการ Subscriber ไปยัง Topic ใด ๆ บน Broker จะได้รับข้อมูลที่ Publisher ส่งให้ Topic นั้น ๆ บน Broker ทั้งหมด

จากรูปที่ 2.37 แสดงหลักการทำงานการสื่อสาร แบบ MQTT ซึ่งมีส่วนประกอบดังนี้

1. Broker (Server) คือตัวกลางในการรับข้อมูลจาก Publisher และส่งข้อมูลให้ Subscriber
2. Publisher คือตัวส่งข้อมูลให้กับ Topic ที่อยู่ใน Broker เรียกว่าการ Publish
3. Subscriber คือตัวรับข้อมูลจาก Topic ที่อยู่ใน Broker เรียกว่าการ Subscribe

2.13 โมดูลตัวอ่านการ์ด (MicroSD Card Adapter)



รูปที่ 2.38 ตัวอย่าง โมดูลตัวอ่านการ์ด Micro SD

โมดูลตัวอ่านการ์ด (MicroSD Card Adapter) คือโมดูลอินเทอร์เฟซ SPI ผ่านไดร์เวอร์ระบบไฟล์ โดยนิยมมาใช้งานร่วมกับระบบไมโครคอนโทรลเลอร์เพื่อให้การ์ด MicroSD อ่านและเขียนไฟล์ได้ด้วยไมโครคอนโทรลเลอร์ แสดงตัวอย่างดังรูปที่ 2.38

คุณสมบัติทั่วไปของ โมดูลตัวอ่านการ์ด Micro SD

1. แผงวงจรควบคุมแรงดันไฟฟ้า 3.3V
2. การสื่อสาร อินเทอร์เฟซ SPI มาตรฐาน
3. บัส SPI MISO, MOSI, SCK
4. CS ชิปเลือกสัญญาณขา

2.14 ปริทัศน์วรรณกรรมและงานวิจัยที่เกี่ยวข้อง

ในปี ค.ศ 2021 M.S. Ahmed [1] ได้ศึกษาการออกแบบระบบเวลาจริง (Real time) ด้วยอินเทอร์เน็ตของสรรพสิ่ง (Internet of Things) โดยพิจารณาประสิทธิภาพของโปรโตคอล Suggested System เทียบกับโปรโตคอล Message Queuing Telemetry Transport (MQTT) โดยเพลตฟอร์มการสร้างต้นแบบใช้ Nucleo Board Arm Cortex-M40 เป็นอุปกรณ์ใน Suggested System ซึ่งระบบแนะนำมีโครงสร้างแบบโหนด แบ่งออกเป็น 4 ประเภท 1. โหนดเซนเซอร์ (Sensor nodes) เป็นการส่งค่าข้อมูลไปยังระบบ เช่น เซนเซอร์วัดอุณหภูมิ เซนเซอร์เปิดปิดประตู 2. โหนดแอคชูเอเตอร์ (Actuator nodes) เป็นการทำงานตามค่าที่ได้รับจากระบบ เช่น มอเตอร์ จอแสดงผลภาพ 3. โหนดไฮบริด (Hybrid nodes) มีความสามารถในการส่งและรับข้อมูลในระบบ เช่น ไมโครคอนโทรลเลอร์ 4. โหนดสังเกตการณ์ (Monitoring node) เป็นการควบคุม และตรวจสอบ

โหนดของระบบ อาทิ สมาร์ทโฟน หน้าจอสัมผัส ในงานวิจัยจะทำการตั้งค่าการทดสอบที่เป็น 1 โหนด และ 1 เซิร์ฟเวอร์ สองโปรโตคอล โดย 1 คือ โปรโตคอลแนะนำ 2 คือ โปรโตคอล MQTT เริ่มทดสอบโดยการส่งข้อความจากโปรโตคอลทั้งสองระบบไปยังเซิร์ฟเวอร์ แล้วทำการวิเคราะห์เปรียบเทียบผลเวลาล่าช้าในการส่งข้อความไปยังเซิร์ฟเวอร์ทั้งสองโปรโตคอล ผลที่ได้พบว่า ระบบโปรโตคอลแบบ MQTT มีขนาดข้อมูลที่ต่ำกว่าโปรโตคอลที่แนะนำดั้งนั้นความล่าช้าของข้อความจึงน้อยกว่าของระบบโปรโตคอลแนะนำ

ในปี ค.ศ 2020 Saif Allah H. AlMetwally และคณะ [2] ได้ศึกษาระบบการจัดการคุณภาพน้ำจาก IOT แบบ Real Time เนื่องจากปัญหาคุณภาพน้ำได้รับผลกระทบจากกิจกรรมทางอุตสาหกรรมที่ส่งผลโดยตรงต่อน้ำดื่มกระจายไปยังน้ำประปาของเมือง เพื่อให้สามารถควบคุมคุณภาพของน้ำดื่มได้โดยผ่านการตรวจสอบคุณภาพน้ำด้วยการนำระบบ IOT มาบูรณาการใช้เซนเซอร์เพื่อตรวจสอบและควบคุมปัจจัยคุณภาพน้ำได้ในเวลาจริง ในงานวิจัยนี้ได้ออกแบบจำลองการตรวจสอบและควบคุมปัจจัยประสิทธิภาพของน้ำ โดยมีสถาปัตยกรรมระบบ เซนเซอร์วัดอุณหภูมิ เซนเซอร์วัดค่า pH เพื่อใช้วัดความเป็นกรดด่างของน้ำ เซนเซอร์ระดับน้ำอัลตราโซนิก เพื่อใช้ในการวัดระดับน้ำ เซนเซอร์วัดการไหลของน้ำเพื่อใช้วัดปริมาณน้ำที่จะไหลผ่านออกกจากระบบ เซนเซอร์วัดความขุ่นใช้วัดความขุ่นมัวของน้ำ วาล์วน้ำโซลินอยด์ ใช้ควบคุมการไหลของน้ำ เครื่องกรองน้ำ ใช้สำหรับบำบัดน้ำเสีย คอนโทรลเลอร์ที่ใช้คือ Arduino Uno สมมติฐานที่เสนอคือการเชื่อมต่อแบบจำลองทางเข้าของน้ำทำการตรวจสอบ HP อุณหภูมิ และระดับน้ำและตรวจสอบการไหลของน้ำ เมื่อเซนเซอร์วัดค่าความขุ่นตัวแรกจะส่งค่าไปที่ตัวควบคุมเพื่อทำการเปรียบเทียบค่าความขุ่นที่เซนเซอร์ที่ 2 ซึ่งตัวควบคุมจะเป็นตัวตัดสินใจในการที่จะเปิดหรือปิดวาล์วเพื่อให้น้ำไหลเข้าตัวกรองหรือไม่ จากนั้นน้ำที่บำบัดแล้วจะถูกเก็บไว้ในถังที่สองจึงจะสามารถนำมาบริโภคได้ความปลอดภัย ระบบควบคุมคุณภาพนี้สามารถเพิ่มประสิทธิภาพของน้ำระบบการจัดการ นอกจากนี้การตรวจสอบตามเวลาจริงและแก้ไขความข้อผิดพลาดที่เกิดจากทำงานของมนุษย์ โดยใช้ต้นแบบโมเดลนี้เพื่อทำการรวบรวมข้อมูลและวิเคราะห์ผล

ในปี ค.ศ 2014 M. Behzad และ คณะ [3] ศึกษาวิจัยเกี่ยวกับการออกแบบและพัฒนาระบบติดตามโดยใช้ต้นทุนต่ำ เนื่องจากปัญหาพื้นฐานที่เกิดขึ้นในระบบติดตามในปัจจุบันคือต้นทุน เพื่อแก้ไขปัญหานี้ อัลกอริทึมตำแหน่งที่ไม่มี GPS การใช้เฉพาะสัญญาณ RF ใน WSN สำหรับการประมาณตำแหน่งของโหนด อาจส่งผลให้เกิดข้อผิดพลาดของระยะทาง และยังมีข้อจำกัดในเรื่องของแบตเตอรี่ที่จำกัดและการพกพายังเป็นคอขวดในระบบ WSN ในการวิจัยครั้งนี้ จึงนำระบบ GPS Tracking ซึ่งมีราคาถูกกว่ามาใช้งาน โดยมีฟังก์ชันการทำงานที่มากกว่าระบบติดตามที่มีอยู่ทั่วไป ระบบประกอบไปด้วยเครื่องรับ GPS และโมเด็ม GSM ที่เชื่อมต่อกับ ไมโครคอนโทรลเลอร์ ATmega-16 ในการติดตามยานพาหนะ หลักการคือ เจ้าของรถส่ง SMS ไปยังระบบติดตามที่ติดตั้งภายในรถ

เพื่อให้ไมโครคอนโทรลเลอร์นำพิกัดลองจิจูด และละติจูดของตำแหน่งปัจจุบันจาก GPS ส่งไปทาง SMS โดยผ่านโมเด็ม GSM ไปยังเซิร์ฟเวอร์กลับไปยังเจ้าของรถ ซึ่งพิกัดตำแหน่งรถจะอยู่บน Google Maps สำหรับผู้ใช้โทรศัพท์ Android และเมื่อถูกขโมยรถ เจ้าของสามารถปิดสวิตช์กุญแจหลัก และตรวจสอบความเร็วของรถได้ โดยสามารถติดตามยานพาหนะบน Google Maps ได้อย่างง่าย โดยไม่ต้องใช้อินเทอร์เน็ต ผลปรากฏว่าการพัฒนาระบบติดตามมีค่าใช้จ่าย ประมาณ 80 ดอลลาร์สหรัฐ (2,700 บาท) ซึ่งมีค่าใช้จ่ายน้อยกว่าระบบติดตามที่มีอยู่มาก การทดสอบระบบพบว่าระบบติดตามพาหนะ สามารถตรวจสอบสถานะพิกัดตำแหน่งและตอบกลับไปยังบนเว็บพอร์ทัล โดยใช้แผนที่ Google ได้สำเร็จทุกครั้ง เมื่อทำการเปรียบเทียบคุณลักษณะของระบบกับบริษัทระบบติดตามหลายแห่ง โดยพิจารณาจากต้นทุนโดยรวม การควบคุมบริการ การสร้างความน่าเชื่อถือ การรับรองความถูกต้อง และการทำงานตลอด 24 ชั่วโมง ดังนั้นการออกแบบระบบติดตามพาหนะนี้ เป็นการออกแบบระบบติดตามที่มีประสิทธิภาพ

ในปี ค.ศ 2014 José Fernando Mendoza [4] ศึกษาวิจัยเกี่ยวกับการพัฒนาสถาปัตยกรรมสำหรับซอฟต์แวร์ฝังตัวในไมโครคอนโทรลเลอร์สำหรับ Internet of Things (IoT) ในการเก็บรวบรวมหมอก (Fog Water Collection) เนื่องจากปัญหาการขาดแคลนน้ำดื่มสะอาดในบางพื้นที่ทำให้เกิดสภาพแวดล้อม สุขภาพ และสังคมต่าง ๆ ปัญหา สถานการณ์นี้เกิดขึ้นเนื่องจากแหล่งน้ำจำนวนมากปนเปื้อนหรือหมดไปเนื่องจากการขาดแคลนครั้งนี้ประชากรที่อยู่ห่างไกลและในชนบทจำนวนมากต้องพึ่งพาเกวียนน้ำหรือบ่อน้ำที่ไม่ได้ คุณภาพ และปริมาณที่คนต้องการ จึงมีแนวคิดในการแก้ปัญหาดังกล่าวด้วย การรวบรวมละอองน้ำ (FWC) ทำให้สามารถรับน้ำในเขตแห้ง โดยการควบแน่นหมอกให้เป็นหยดน้ำ ผ่านการควบคุมร่วมด้วยกับระบบ IoT ข้อดีคือไม่ต้องใช้พลังงานไฟฟ้าในการสกัดหรือลำเลียงน้ำ นอกจากความเสี่ยงของการปนเปื้อนจะต่ำเมื่อเทียบกับเทคนิคอื่น ดังนั้นระบบ FWC เป็นทางเลือกที่เหมาะสมในพื้นที่ภูเขาที่มีหมอกบ่อย งานวิจัยนี้ นำเสนอสถาปัตยกรรมซอฟต์แวร์สำหรับการพัฒนาระบบฝังตัวที่ทำงานบนไมโครคอนโทรลเลอร์สถาปัตยกรรมไม่ได้ขึ้นอยู่กับโดเมนเฉพาะและเน้นที่ข้อกำหนดด้านการทำงานของซอฟต์แวร์เน้นคุณสมบัติคุณภาพ การออกแบบชั้นของสถาปัตยกรรมทำให้สามารถระบุส่วนประกอบที่ซอฟต์แวร์ต้องมีและวิธีที่แต่ละรายการต้องโต้ตอบกับองค์ประกอบของการเก็บข้อมูลบัตร กระบวนการประเมินผลอนุญาตให้ระบุได้ว่าการใช้สถาปัตยกรรมทำให้สามารถปฏิบัติตามได้คุณลักษณะด้านคุณภาพ เช่น ความสามารถในการบำรุงรักษา ความปลอดภัย ความสามารถในการปรับขนาด และอื่น ๆ ในระหว่างการประเมินคุณภาพของแอตทริบิวต์ความพร้อมใช้งานได้รับการตอบสนองเป็นส่วนใหญ่เนื่องจากข้อมูลถูกจับอย่างสม่ำเสมอในช่วง 2 เดือน บนในทางกลับกัน การเก็บข้อมูลทำให้นักวิจัย CLEM – SENA พบว่าความชื้น อุณหภูมิ และปริมาตรของน้ำที่เก็บได้ (ลิตร) ระหว่างเวลา 17.00 น. ถึง 08.00 น. อุณหภูมิเฉลี่ยของสถานที่นี้จะแกว่งไปมาระหว่าง 19 °C ถึง 23 °C มีน้ำควบแน่นเป็นหมอกที่หมุนเวียนไม่คงที่

ความชื้นสัมพัทธ์เฉลี่ยอยู่ระหว่าง 80% ถึง 100% ปริมาณเฉลี่ยที่เก็บได้น้ำในแต่ละ FCW คือ 87 ลิตรต่อวัน ระหว่างเวลา 01.00 ถึง 04.00 น. อุณหภูมิถึงระดับต่ำสุด (13.5 °C) และความชื้นถึงระดับสูงสุด (99%) น้ำที่เก็บรวบรวมสามารถดื่มได้เนื่องจากมีระดับบริสุทธิ์กว่าน้ำฝนเพราะมีแร่ธาตุต่ำ นอกจากนี้ยังสามารถใช้น้ำนี้ในกิจกรรมการเกษตรอื่น ๆ

ในปี ค.ศ 2021 Mirza Jabbar Aziz Baig และ คณะ [5] ศึกษาวิจัยเกี่ยวกับการออกแบบและใช้ระบบ IOT ร่วมกับแพลตฟอร์มการซื้อขายพลังงานแบบ P2P บนบล็อกเชนโดยใช้ ESP32-S2 Node-Red และโปรโตคอล MQTT โดยการรับข้อมูลแบบเรียลไทม์ในการเฝ้าติดตามและการควบคุมพลังงานที่สร้างขึ้นในพื้นที่ห่างไกลในการซื้อขายพลังงานโดยใช้แพลตฟอร์ม IOT ซึ่งใช้ไมโครคอนโทรลเลอร์ ESP32-S2 ในการรับค่าจากเซนเซอร์ตรวจวัดแรงดันไฟฟ้าประมวลผลข้อมูลและถ่ายโอนข้อมูลผ่าน Wi-Fi โดยใช้โปรโตคอล Message Queuing Telemetry Transport (MQTT) ไปยังฐานข้อมูลออนไลน์ ซึ่งเหมาะสำหรับแอปพลิเคชัน IOT ที่มีทรัพยากรจำกัดและใช้ Node-Red ในการดึงข้อมูลจากฐานข้อมูลออนไลน์มาแสดงผลลัพธ์บนเว็บแอปพลิเคชันในการซื้อขายพลังงานแบบ P2Pโดยวิธีการ HTTP จากการทดสอบได้พิสูจน์แล้วว่าแพลตฟอร์มการซื้อขายพลังงานแบบ P2P ที่เสนอสามารถทำงานได้อย่างสมบูรณ์

ในปี ค.ศ 2011 Ambade Shruti Dinkar and S.A Shaikh [6] ศึกษาวิจัยการออกแบบและการใช้งานระบบติดตามยานพาหนะการใช้ GPS เพื่อพัฒนาระบุตำแหน่งยานพาหนะอัตโนมัติโดยใช้ GPS สำหรับข้อมูลตำแหน่งและระบบรักษาความปลอดภัยแจ้งเตือนเจ้าของรถโดยส่ง SMS เมื่อมีผู้บุกรุกเข้ามายานพาหนะ โดยงานวิจัยนี้มีการใช้ สถาปัตยกรรม ดังนี้ GM900-GPS พรอกซิมิตี้เซนเซอร์สำหรับช่วยจอดรถเซนเซอร์สั่นสะเทือน Ultrasonic Sensors ไมโครคอนโทรลเลอร์ ARM9 microcontroller เมื่อทำการทดสอบพบว่า In-Vehicle Software โปรแกรม ถูกนำมาใช้มีปัญหาเนื่องจากไม่สามารถตรวจจับยานพาหนะที่จะติดตามได้หากพบรถบางคัน มีลักณะใกล้เคียงเหมือนกับคันในรุ่น ความน่าจะเป็นของข้อผิดพลาดจะเพิ่มหลายเท่า ถ้ามีสิ่งรบกวนที่ขอบภาพ

ในปี ค.ศ 2022 Carlos A. Hernández-Morales และคณะ [7] ได้ศึกษาวิจัยเกี่ยวกับการออกแบบและปรับใช้ระบบตรวจสอบด้วย IoT ที่ใช้งานได้จริงสำหรับการเพาะปลูกที่ได้รับการคุ้มครอง ที่สามารถคาดการณ์สำหรับการใช้งานทางเกษตร มีสถาปัตยกรรมที่มีประสิทธิภาพประกอบไปด้วยการตรวจจับ เครือข่ายการประมวลผล และแอปพลิเคชันที่มีค่าใช้จ่ายไม่สูง เพื่อแสดงถึงความเป็นไปได้ ระบบ IoT ถูกสร้างขึ้นทดสอบทดลองและตรวจสอบโดยการตรวจสอบอุณหภูมิและความชื้นของเรือนกระจก สำหรับกรณีศึกษาอุปกรณ์ IoT จะรวมเฉพาะเซนเซอร์อุณหภูมิ PT1000 และเซนเซอร์ความชื้น DHT21 ซึ่งเชื่อมต่อผ่านอินเทอร์เฟซ SPI กับไมโครคอนโทรลเลอร์ Arduino Uno ที่ติดตั้งโมดูล LPWAN ช่วยให้สามารถส่งข้อมูลผ่านอินเทอร์เน็ตไปยังแพลตฟอร์ม IoT โดยใช้โครงสร้างพื้นฐานเครือข่าย Sigfox โดยอุปกรณ์แต่ละเครื่องมี ID เฉพาะสำหรับการกำหนดเส้นทาง

การรับส่งข้อมูลและได้ใช้ RESTful API ที่ทำหน้าที่โต้ตอบกับแบ็กเอนด์ขึ้นกับคำขอ HTTPS REST (PUT, GET, DELETE หรือ POST) และจัดเตรียมอินเทอร์เฟซจากโปรโตคอลไปยังแอปพลิเคชัน

ในปี ค.ศ 2021 Nawzad K. Al-Salihi [8] ได้ศึกษาวิจัยเกี่ยวกับการปรับปรุงความทนทานต่อความผิดพลาดในระบบกำหนดตำแหน่งด้วย IOT ได้นำเสนอกลไกลสำหรับการติดตามวัตถุโดยพิจารณาโมดูล IoT และ GPS ซึ่งสถาปัตยกรรมประกอบไปด้วย Arduino Uno เป็นหน่วยประมวลผล รับค่าข้อมูลพิกัด GPS จากโมดูล GPS และส่งข้อมูลไปยัง ESP8266 เพื่อทำการเชื่อมต่ออินเทอร์เน็ตผ่าน Wi-Fi ยังหน้าเว็บ แบบจำลองของระบบได้เพิ่มชุด โมดูล GPS เพิ่ม 1 ชุดในสถาปัตยกรรม ทำเปรียบค่าพิกัดก่อนทำการส่งข้อมูลไปยัง ESP8266 เพื่อลดความผิดพลาดของข้อมูลจากการจำลองกลไกลการตรวจสอบความถูกต้องและการปรับปรุงความผิดพลาดน้ำได้ถึง 77.4%

ในปี ค.ศ 2021 Victor Chang [9] ได้ศึกษาวิจัยเกี่ยวกับระบบเซนเซอร์ IoT ทางอุตสาหกรรมสำหรับวัดอุณหภูมิสูง นำเสนอการออกแบบและการใช้งานระบบเซนเซอร์ Arduino ในการบันทึกอุณหภูมิและตำแหน่งของภาชนะเพื่อวัตถุประสงค์ทางโลหะวิทยา สถาปัตยกรรมประกอบไปด้วย Arduino Uno เป็นหน่วยประมวลผลข้อมูล เซนเซอร์ตรวจวัดอุณหภูมิเทอร์โมคัปเปิล K-Type โมดูล GPS ใช้ระบุพิกัดตำแหน่ง โมดูลนาฬิกาแบบเรียลไทม์ โมดูล SD Card ใช้สำหรับบันทึกค่าข้อมูลเมื่อเชื่อมต่อล้มเหลว และโมดูล Bluetooth DSD TECH HC-05 ใช้ในการสื่อสารข้อมูลจาก Arduino Uno ไปยังแอปพลิเคชัน Windows Presentation Foundation (WPF) แล้วทำการส่งข้อมูลไปยัง SQL Database

ในปี ค.ศ 2021 Amit Kumer Podder [10] ได้ศึกษาวิจัยเกี่ยวกับ ระบบเกษตรอัจฉริยะที่ใช้ IoT สำหรับการตรวจสอบพารามิเตอร์การทำฟาร์มในเมือง ได้นำเสนอระบบ Smart Agro Tech บน IoT ในบริบทของการทำฟาร์มในเมืองที่พิจารณาความชื้นอุณหภูมิในอากาศและความชื้นในดิน โดยมีโครงสร้างสถาปัตยกรรมเลือก วัดอุณหภูมิและความชื้นในอากาศด้วยเซนเซอร์ DHT11 วัดความชื้นในดินด้วยเซนเซอร์ Soil Moisture ส่งค่าไปยัง ESP8266 เป็นตัวรับค่าเซนเซอร์ต่าง ๆ ทำการสั่งงานให้กับ Water Pump ทำงานตามเงื่อนไข และทำการเชื่อมต่ออินเทอร์เน็ตผ่านระบบ Wi-Fi ในการส่งข้อมูลไปยังเว็บเซิร์ฟเวอร์ ThinkSpeak ในการจัดเก็บและแสดงผลข้อมูล

2.15 สรุป

บทนี้ได้นำเสนอทฤษฎี และปริทัศน์วรรณกรรมเกี่ยวกับการนำไมโครคอนโทรลเลอร์มาประยุกต์ใช้กับระบบอินเทอร์เน็ต จากการพิจารณาอุปกรณ์ตรวจวัดอุณหภูมิ หลากหลายประเภท ได้ตัดสินใจเลือกเซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 เนื่องจากมีราคาถูกสามารถกันน้ำได้และที่สำคัญสามารถทนกับช่วงอุณหภูมิต่ำได้เหมาะกับการนำไปใช้งานในรถตู้บรรทุกน้ำแข็งที่มีสภาวะอุณหภูมิต่ำและมีความชื้นสูง การตรวจจับวัตถุพิจารณาเลือกใช้เซนเซอร์แบบตรวจจับคลื่นแม่เหล็ก Magnetic

Sensor เนื่องจากประตูรถตู้บรรทุกน้ำแข็งมีการเปิดปิดตลอดเวลาจึงไม่เหมาะที่จะนำเซนเซอร์สัมผัสกับประตูโดยตรง ดังนั้นเพื่อให้อายุการใช้งานที่ยาวนานจึงเลือกใช้เซนเซอร์ตรวจจับคลื่นแม่เหล็กที่ไม่การสัมผัสกับประตูโดยตรง และระบบพิกัดทางภูมิศาสตร์เลือกใช้โมดูลจีพีเอส Ublox NEO-7M ที่มีเสาอากาศภายในตัวและมีช่องต่อสำหรับเสาอากาศภายนอกด้วยเพื่อเพิ่มความสามารถในการรับสัญญาณจีพีเอสได้ดีมากยิ่งขึ้น รวมไปถึงการเลือกใช้โปรโตคอล Message Queuing Telemetry Transport (MQTT) ในการส่งข้อมูลไปยังฐานข้อมูลออนไลน์เนื่องจากมีความเหมาะสมสำหรับแอปพลิเคชัน IOT ที่มีทรัพยากรจำกัดผ่านอินเทอร์เน็ต ทั้งนี้เพื่อเป็นประโยชน์และแนวทางแก่ผู้วิจัยในการดำเนินงานวิจัยในบทต่อ ๆ ไป

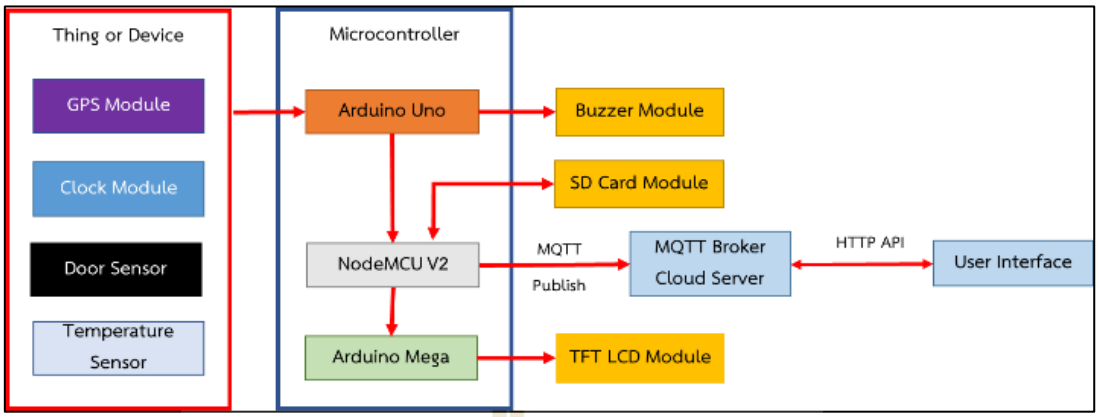
บทที่ 3
วิธีการดำเนินการวิจัย

3.1 กล่าวนำ

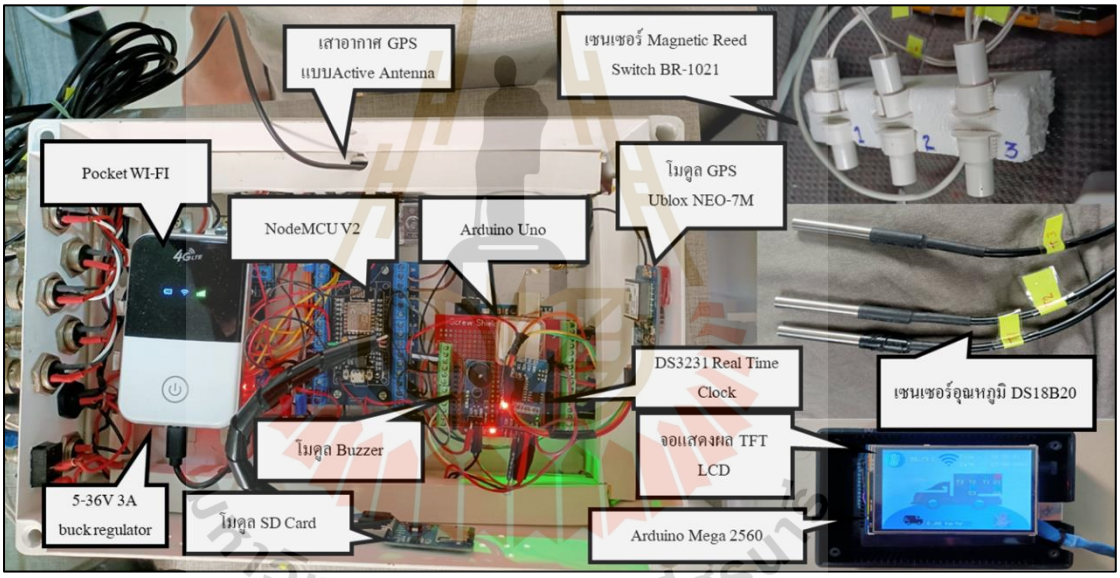
งานวิจัยนี้ได้ทำการออกแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง เพื่อนำไปใช้ตรวจสอบการสูญเสียอุณหภูมิภายในตู้น้ำแข็ง เส้นทางการเดินรถ และพฤติกรรมการเปิดปิดประตูรถตู้บรรทุกน้ำแข็ง ดังนั้นในกระบวนการศึกษาจึงเริ่มต้นจากการออกแบบระบบในภาพรวม การพิจารณาเลือกใช้อุปกรณ์เซนเซอร์ต่าง ๆ ให้มีความเหมาะสมกับสภาพแวดล้อมภายในตู้บรรทุกน้ำแข็ง การออกแบบระบบตรวจวัดค่าเซนเซอร์ต่าง ๆ ระบบส่งข้อมูลไปยังฐานข้อมูลออนไลน์ ระบบแสดงผลและการแจ้งเตือนตามลำดับ

3.2 การออกแบบระบบ

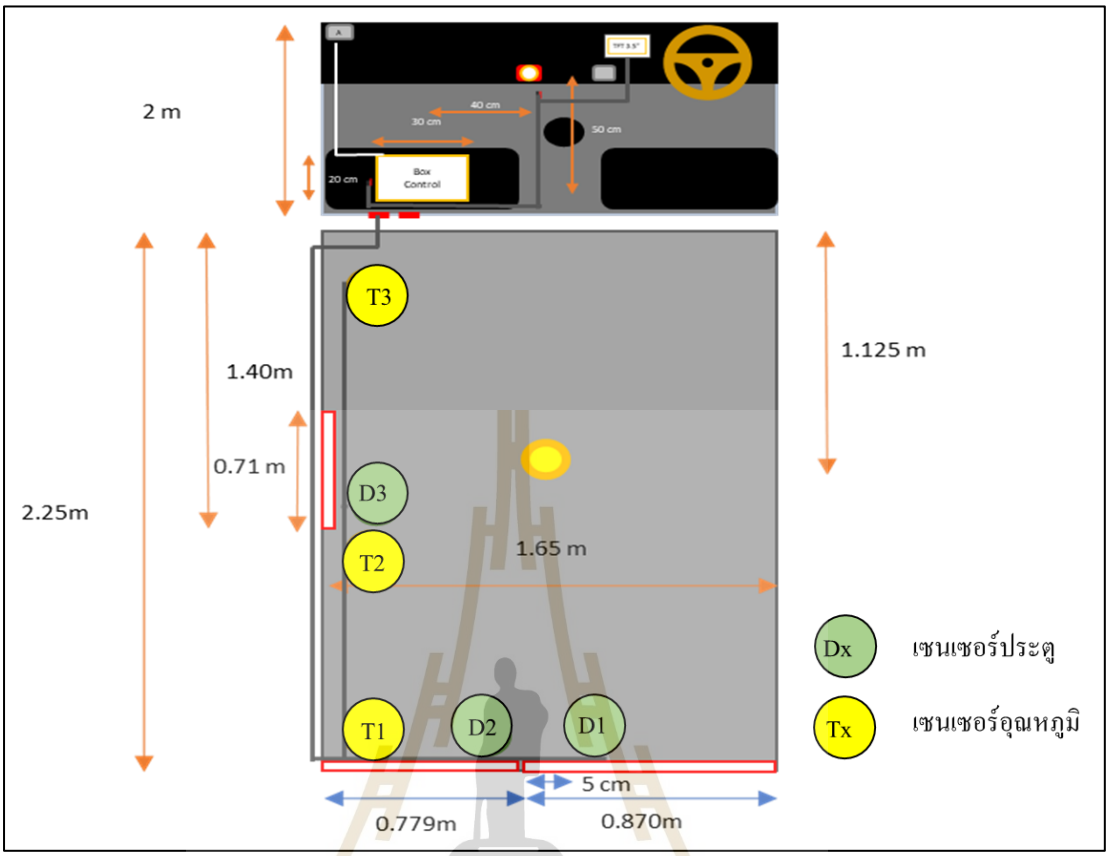
การออกแบบระบบได้พิจารณาถึงการทำงานของการออกแบบโครงสร้างสถาปัตยกรรม แสดงในรูปที่ 3.1 และรูปที่ 3.2 คือส่วนประกอบที่สำคัญในชุดอุปกรณ์ต้นแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง ที่สามารถทำการเชื่อมต่ออุปกรณ์ตรวจวัดค่าเซนเซอร์ และโมดูลต่าง ๆ ด้วย ไมโครคอนโทรลเลอร์ Arduino Uno R3 ในการตรวจเช็คการเปิดปิดประตู กรณีประตูไม่ได้ปิดอยู่โมดูล Active Buzzer จะทำงานส่งเสียงเตือนจนกว่าประตูจะปิด และดำเนินการส่งค่าข้อมูลค่าเซนเซอร์และโมดูลต่าง ๆ ไปยัง NodeMCU V2 บันทึกข้อมูลลง โมดูล SD Card และส่งค่าข้อมูลไปยังฐานข้อมูลออนไลน์ ด้วยโปโตคอล Message Queuing Telemetry Transport (MQTT) ผ่านอินเทอร์เน็ตมาตรฐานการสื่อสารแบบไร้สาย IEEE 802.11 จาก Pocket Wi-Fi และทำการส่งค่าข้อมูลไปยัง Arduino Mega 2560 เพื่อใช้ในการแสดงผลหน้าจอสถานะทำงานของเซนเซอร์และโมดูลต่าง ๆ ผ่าน โมดูล TFT LCD



รูปที่ 3.1 สถาปัตยกรรมระบบ



รูปที่ 3.2 ชุดอุปกรณ์ต้นแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง



รูปที่ 3.3 แผนผังตำแหน่งการติดตั้งชุดอุปกรณ์ต้นแบบระบบติดตาม และเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

3.3 ระบบตรวจวัดค่าเซนเซอร์ต่าง ๆ

การพิจารณาเลือกอุปกรณ์ในระบบ มีข้อพิจารณาจากลักษณะการใช้งานที่เหมาะสมต่อสภาพแวดล้อมใช้งานจริง และมีความเที่ยงตรงของข้อมูลในการตรวจวัด โดยอุปกรณ์ประมวลผลสำหรับรับค่าเซนเซอร์ และโมดูลต่าง ๆ ในงานวิจัยนี้เลือกใช้ Arduino Uno R3 ซึ่งเป็นไมโครคอนโทรลเลอร์ ที่มีราคาของบอร์ดที่ไม่แพง มี ไลบรารีต่าง ๆ ที่ใช้งานถูกพัฒนาอ้างอิงกับบอร์ดนี้เป็นหลัก และได้กำหนดตัวตรวจวัดระบบดังนี้

3.3.1 ตัวตรวจวัดพิกัดตำแหน่งทางภูมิศาสตร์

เลือกใช้ โมดูล GPS Ublox NEO-7M เป็นอุปกรณ์รับสัญญาณ GPS ที่สามารถตรวจจับสัญญาณดาวเทียมได้สูงสุด 56 ดวง มีค่าความแม่นยำของตำแหน่ง 2.5 เมตร ความแม่นยำของความเร็ว 0.1 เมตรต่อวินาที และความถี่ของสัญญาณอยู่ในช่วงระหว่าง 0.25 Hz ถึง 10 MHz

3.3.2 ตัวตรวจสอบวันที่และเวลา

เลือกใช้โมดูล DS3231 Real Time Clock ที่มีค่าความคลาดเคลื่อนไม่เกิน 2PPM (2 วินาที/1,000,000 วินาที)

3.3.3 ตัวตรวจสอบการเปิดปิดประตู

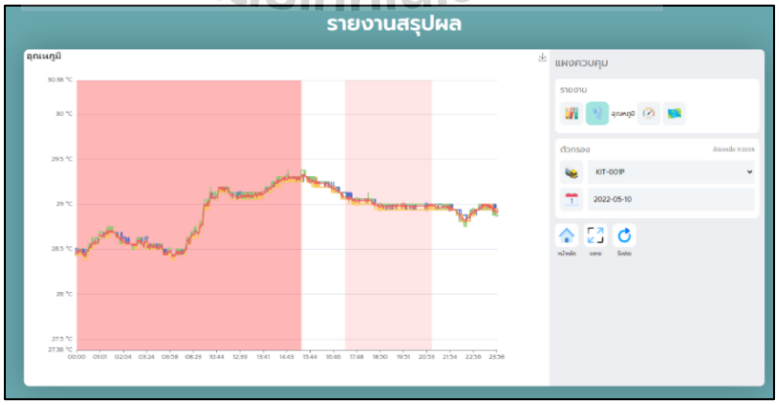
เลือกใช้เซนเซอร์ตรวจจับคลื่นแม่เหล็ก Magnetic Reed Switch BR-1021 มีค่าระยะตรวจจับแม่เหล็ก ช่วง 0 ถึง 21 มม. โดยติดตั้งเซนเซอร์ตรวจจับ Magnetic Reed Switch BR-1021 จำนวนทั้งหมด 3 ชุด ที่บริเวณประตูรถตู้บรรทุกน้ำแข็งดังรูปที่ 3.3

3.3.4 ตัวตรวจวัดอุณหภูมิ

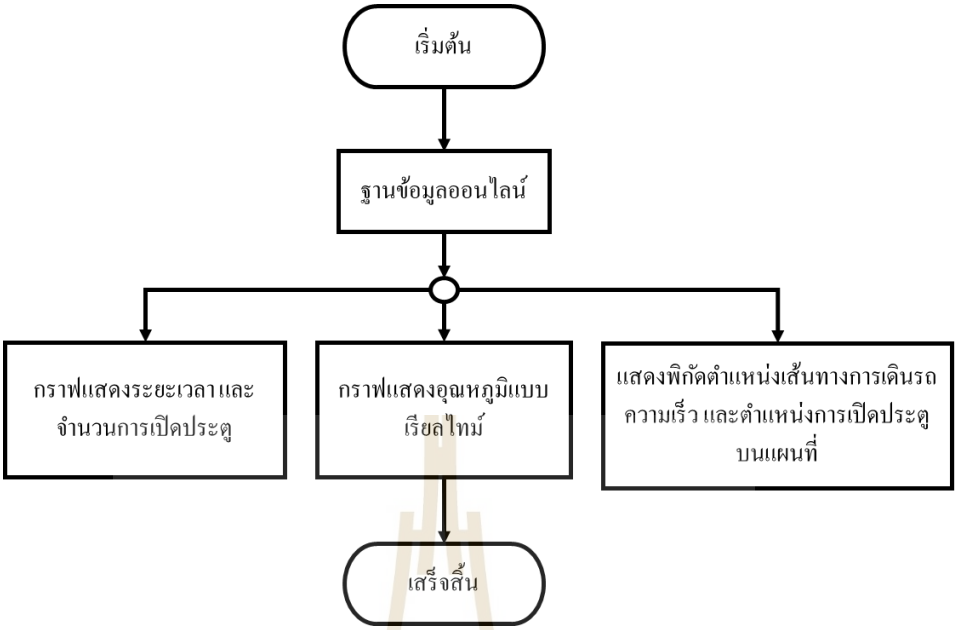
เลือกใช้ดิจิตอลเซนเซอร์ DS18B20 ที่มีค่าความผิดพลาด ± 0.5 °C ค่าความถูกต้องระหว่าง -10 °C ถึง +85 °C โดยติดตั้งเซนเซอร์ DS18B20 จำนวนทั้งหมด 3 ชุดภายในรถตู้บรรทุกน้ำแข็งดังรูปที่ 3.3

3.4 ระบบส่งข้อมูลไปยัง ฐานข้อมูลออนไลน์

การส่งข้อมูลไปยังฐานข้อมูลออนไลน์ งานวิจัยนี้เลือกใช้อุปกรณ์ประมวลผล NodeMCU V2 ซึ่งเป็นไมโครคอนโทรลเลอร์ ที่มีโมดูล WI-FI ESP8266 ทำให้สามารถรับค่าข้อมูล ส่งไปยัง ฐานข้อมูลออนไลน์ ด้วยโปโตคอล Message Queuing Telemetry Transport (MQTT) ที่มีความเหมาะสมสำหรับแอปพลิเคชัน IOT ที่มีทรัพยากรจำกัดผ่านอินเทอร์เน็ตมาตรฐานการสื่อสารแบบไร้สาย IEEE 802.11 จาก Pocket Wi-Fi ได้ และใช้เว็บแอปพลิเคชันในการแสดงผลข้อมูลแบบเรียลไทม์ ดังรูปที่ 3.4โดยมีการแสดงผลบนเว็บแอปพลิเคชัน 3 ส่วนด้วยกันดังนี้ 1. กราฟแสดงระยะเวลาและจำนวนการเปิดประตู 2. กราฟแสดงอุณหภูมิแบบเรียลไทม์ และ 3. แสดงพิกัดเส้นทางเดินรถความเร็วและตำแหน่งการเปิดปิดประตูบนแผนที่ ดังรูปที่ 3.5



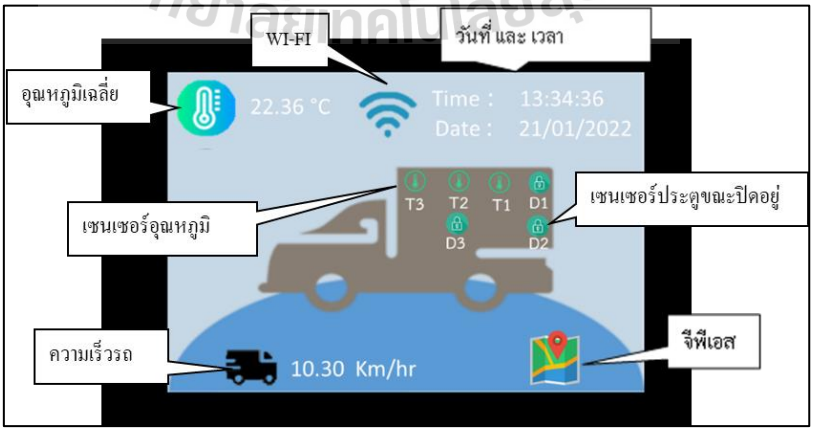
รูปที่ 3.4 ตัวอย่าง Web Application



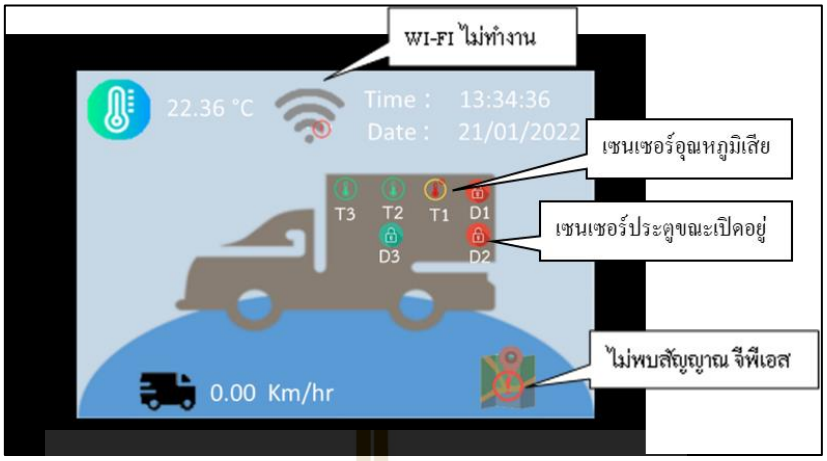
รูปที่ 3.5 กระบวนการทำงานแสดงผลข้อมูลบน Web Application

3.5 ระบบจอแสดงผลและการแจ้งเตือน

ระบบแสดงผล เป็นการทำงานร่วมกันระหว่าง อุปกรณ์ประมวลผลไมโครคอนโทรลเลอร์ Arduino Mega 2560 ร่วมกับโมดูล TFT LCD3.5 นิ้ว ในการแสดงการทำงานของอุปกรณ์ปกติดังรูปที่ 3.6 และแสดงการทำงานของอุปกรณ์ผิดปกติดังรูปที่ 3.7 ระบบการแจ้งเตือน ได้ทำการพิจารณาเลือกใช้อุปกรณ์ส่งสัญญาณเสียง โมดูล Active Buzzer โดยจะส่งเสียงเตือนเมื่อมีการเปิดประตูรถตู้บรรทุกน้ำแข็งทิ้งไว้ และจะหยุดส่งสัญญาณเสียงเมื่อประตูปิดอยู่



รูปที่ 3.6 การออกแบบสถานะทำงานของอุปกรณ์ปกติ



รูปที่ 3.7 การออกแบบสถานะทำงานของอุปกรณ์ผิดปกติ

3.6 ระบบบันทึกข้อมูลสำรอง

ระบบบันทึกข้อมูลสำรองมีความสำคัญเพื่อใช้ในการตรวจสอบข้อมูลย้อนหลังกรณีส่งข้อมูลไปยังฐานข้อมูลออนไลน์ไม่สำเร็จ โดยพิจารณาใช้ NodeMCU V2 ประมวลผลร่วมกับโมดูล SD Card ในการบันทึกข้อมูลสำรอง

3.7 สรุป

ในบทนี้ได้นำเสนอเกี่ยวกับการออกแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง ซึ่งการดำเนินการวิจัยจะต้องมีความรู้ความเข้าใจเกี่ยวกับทฤษฎีที่เกี่ยวข้อง เพื่อนำมาปรับปรุงประยุกต์ใช้กับงานวิจัย โดยหลักกการในการวิจัยนี้ แบ่งออกเป็น 3 ส่วนดังนี้

ส่วนที่ 1 การศึกษาปัญหาที่เกี่ยวข้องกับการใช้งานของรถตู้บรรทุกน้ำแข็งในปัจจุบัน เพื่อให้เข้าใจถึงปัญหาที่เกิดขึ้นและค้นหาแนวทางแก้ไขให้ดีกว่าเดิมโดยอาจจะได้ด้วยวิธีการปรับปรุงระบบวิธีการเดิมหรือ ด้วยการออกแบบระบบใหม่ขึ้นมาทดแทนระบบวิธีการเดิมดังนั้นงานวิจัยนี้เสนอการออกแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

ส่วนที่ 2 การศึกษาทฤษฎีและศึกษาปริทัศน์วรรณกรรมความรู้ที่นักวิจัยผู้อื่นได้ทำการทดลองศึกษามาก่อนหน้านี้เกี่ยวกับระบบติดตามจีพีเอส รถบรรทุกน้ำแข็งไมโครคอนโทรลเลอร์ และอินเทอร์เน็ตของสรรพสิ่ง โดยจะต้องมีความเข้าใจถึงทฤษฎีและความรู้ต่าง ๆ เหล่านี้เป็นอย่างดี

ส่วนที่ 3 การดำเนินงานวิจัยไปตามแผนที่ได้วางไว้ โดยจะมีขั้นตอนการทำงานวิจัยดังนี้

ขั้นตอนที่ 1 การออกแบบสถาปัตยกรรมขอบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง ซึ่งพิจารณาถึงความเป็นไปได้ให้มีความสอดคล้องกับทฤษฎีและศึกษาปริทัศน์วรรณกรรมที่ได้ศึกษามา

ขั้นตอนที่ 2 พิจารณาการเลือกใช้อุปกรณ์ในสถาปัตยกรรมของระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง ให้มีความเหมาะสมกับการนำไปใช้งาน

ขั้นตอนที่ 3 ออกแบบและทดสอบเทียบอุปกรณ์ระบบเซนเซอร์ตรวจวัดอุณหภูมิและระบบโมดูล จีพีเอส ในการระบุตำแหน่งทางภูมิศาสตร์

ขั้นตอนที่ 4 ออกแบบและทดลองระบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

ขั้นตอนที่ 5 ประเมินประสิทธิภาพการทำงานของระบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

ขั้นตอนที่ 6 สรุปผลการทดลอง

บทที่ 4
ผลการทดลอง และการวิเคราะห์ผลการทดลอง

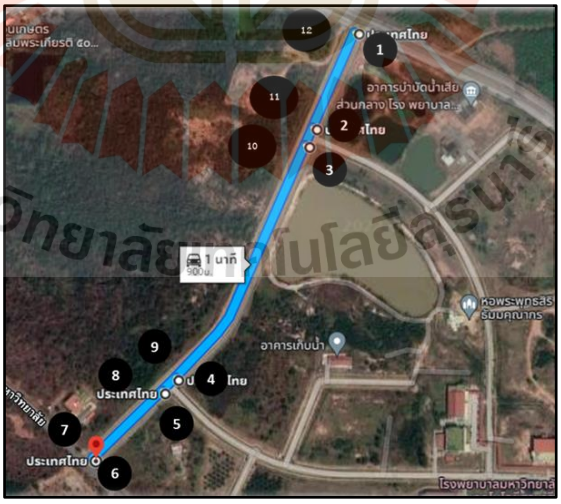
4.1 กล่าวนำ

ในบทนี้จะเป็นการนำเสนอผลการทดสอบ Lab Tests และการทดสอบภาคสนาม

ส่วนที่ 1 การทดสอบ Lab Tests ประกอบไปด้วย การทดสอบโมดูล GPS Ublox NEO-7M การทดสอบเซนเซอร์ตรวจจับ Magnetic Reed Switch BR-1021 และการทดสอบเซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 ส่วนที่ 2 การทดสอบภาคสนาม ประกอบไปด้วย การทดสอบภาคสนามเซนเซอร์ตรวจจับประตู การทดสอบภาคสนามเซนเซอร์ตรวจจับประตู การทดสอบภาคสนามระบบบันทึกข้อมูลสำรอง การทดสอบภาคสนามระบบแสดงผลและการแจ้งเตือนการทำงานของชุดอุปกรณ์ต้นแบบ

4.2 การทดสอบ Lab Tests

4.2.1 การทดสอบโมดูล GPS Ublox NEO-7M



รูปที่ 4.1 สถานที่ทดสอบ Lab Tests พิกัดตำแหน่งทางภูมิศาสตร์ ถนน โรงพยาบาลมหาวิทยาลัยเทคโนโลยีสุรนารี

การทดสอบโมดูล GPS Ublox NEO-7M เปรียบเทียบกับ พิกัดตำแหน่งบน Google map ซึ่งจะต้องอยู่ในขอบเขตที่ยอมรับได้ในการนำไปใช้งานของความผิดพลาดของตำแหน่งไม่เกิน 5 เมตร และความผิดพลาดของความเร็วไม่เกิน 10 กิโลเมตรต่อชั่วโมง การทดสอบได้พิจารณาพิกัดตำแหน่งบริเวณ ถนนโรงพยาบาลมหาวิทยาลัยเทคโนโลยีสุรนารี ดังแสดงในรูปที่ 4.1 ในการทดสอบจะแบ่งออกเป็น 2 การทดสอบ

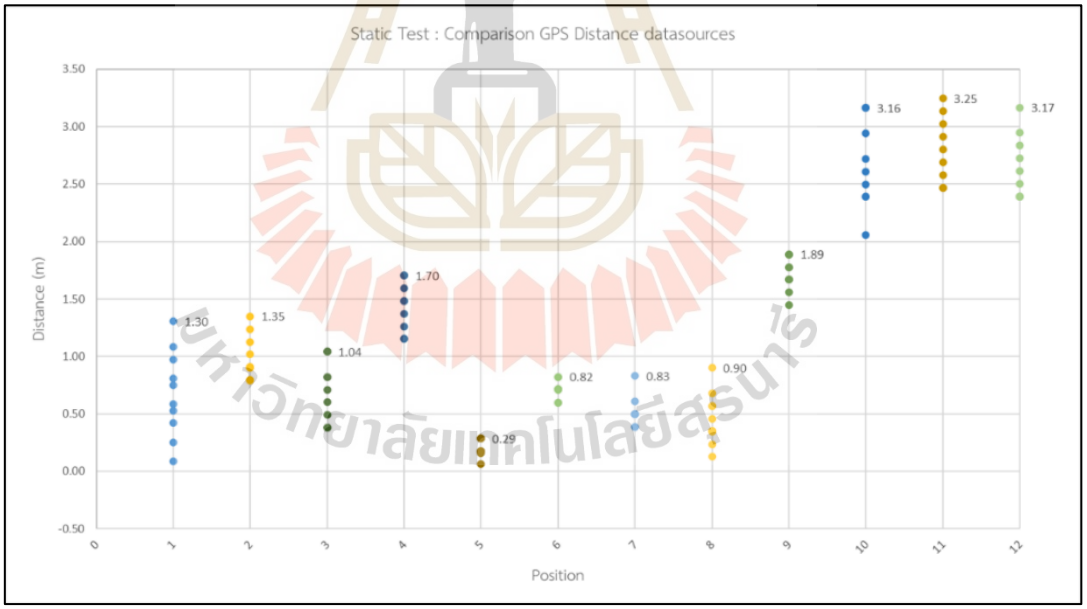
1.การทดสอบแบบ Static Test

การทดสอบเก็บค่าพิกัดตำแหน่ง ขณะอยู่กับที่ในแต่ละตำแหน่งทั้งหมด 12 จุด แล้วนำค่าพิกัดตำแหน่งที่เก็บได้ไปทำการเปรียบเทียบกับพิกัดตำแหน่งบน Google map

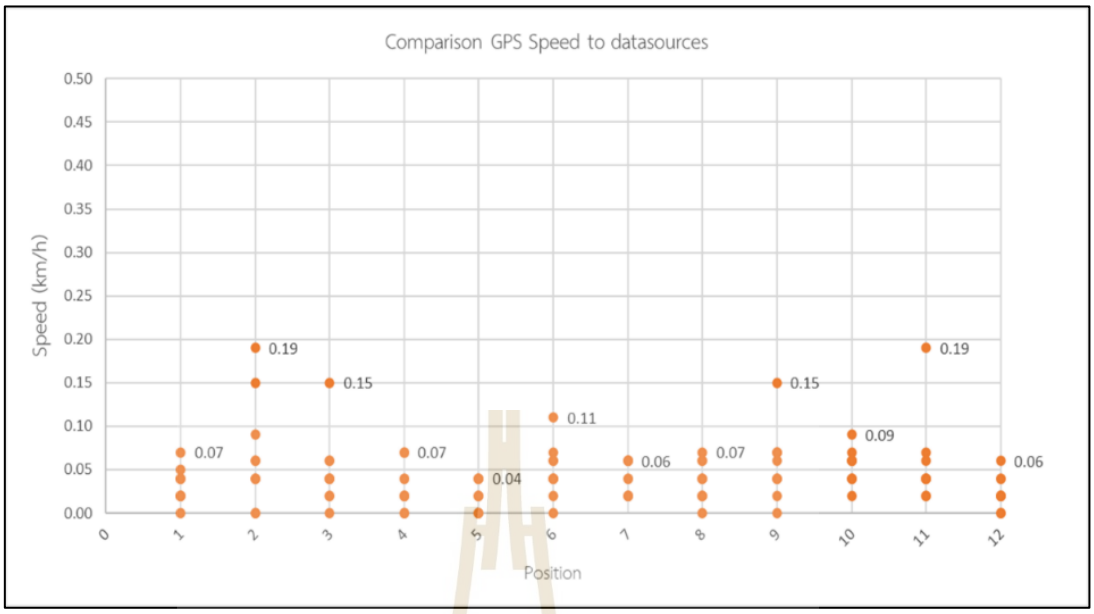
2.การทดสอบแบบ Dynamic Test

การทดสอบเก็บค่าความเร็ว 20 30 และ 40 กิโลเมตรต่อชั่วโมง ขณะเคลื่อนที่ด้วยความเร็วคงที่ผ่านเส้นทางทั้ง 12 จุด เพื่อเปรียบเทียบความผิดพลาดความเร็วของ GPS

1. ผลการทดสอบขณะรถอยู่กับที่ (Static Test)

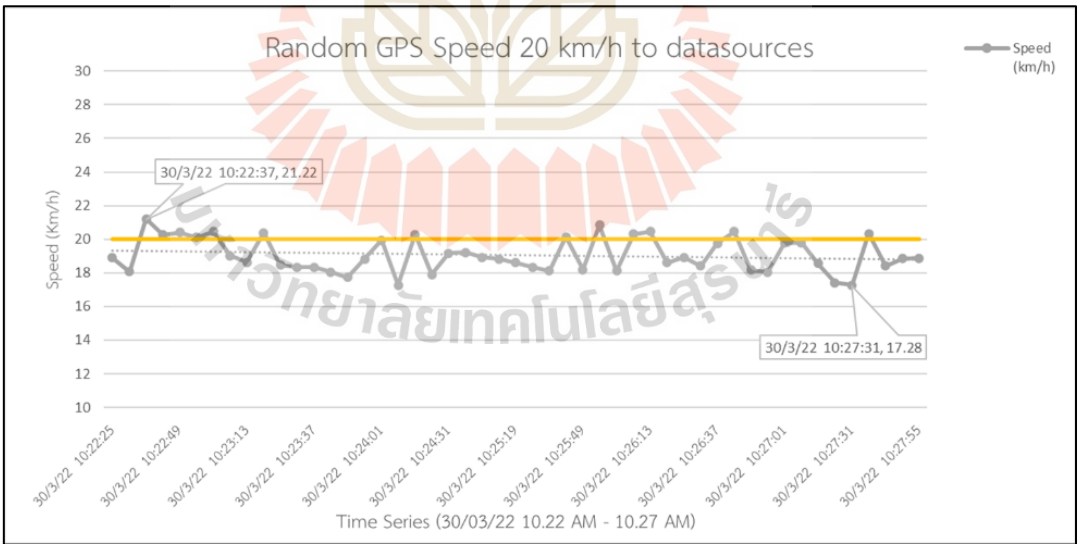


รูปที่ 4.2 กราฟเปรียบเทียบความผิดพลาดพิกัดตำแหน่งของ GPS ขณะอยู่กับที่

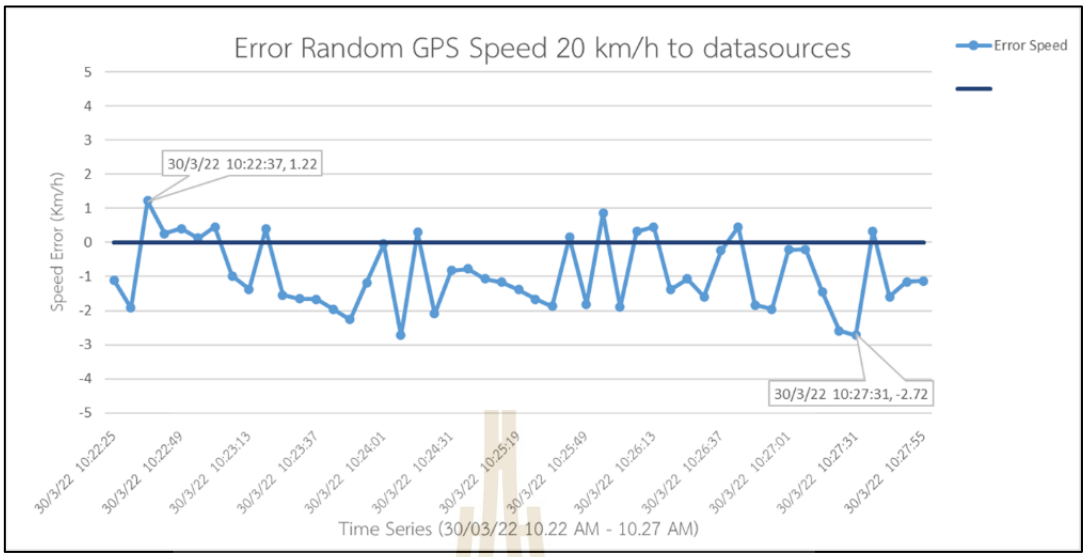


รูปที่ 4.3 กราฟเปรียบเทียบความผิดพลาดความเร็วของ GPS ขณะอยู่กับที่

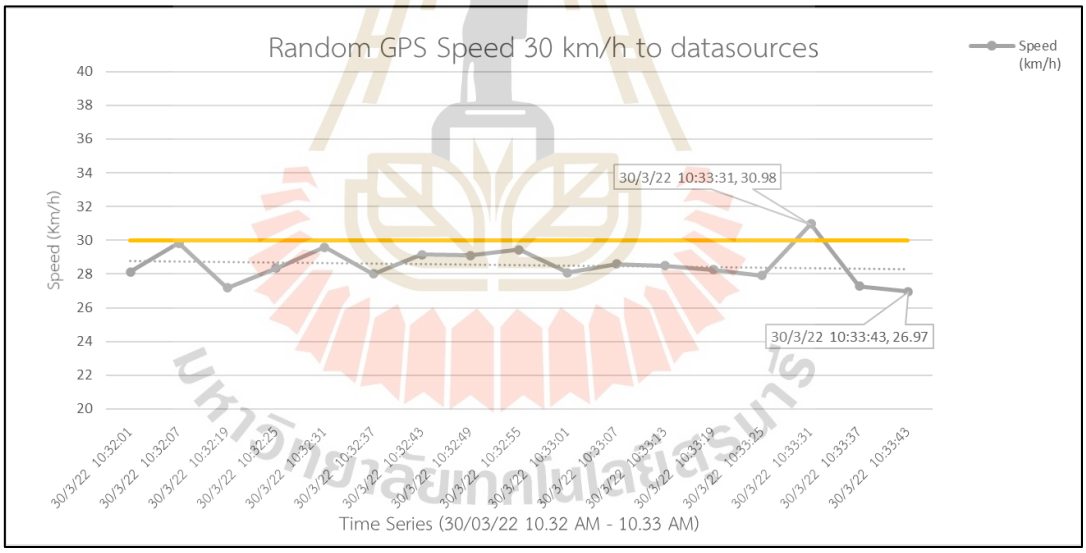
2. ผลการทดสอบขณะรถเคลื่อนที่ (Dynamic Test)



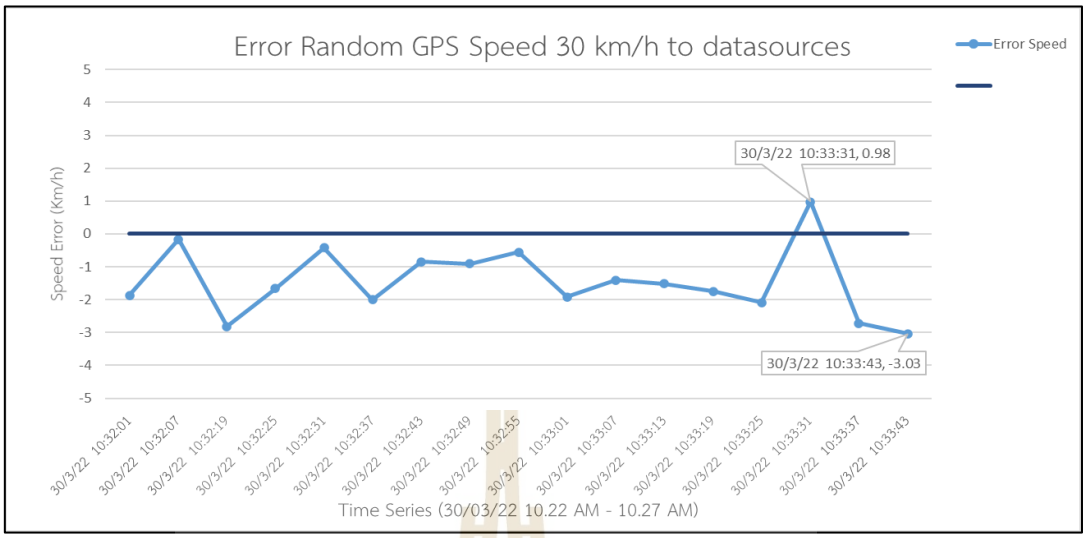
รูปที่ 4.4 กราฟเปรียบเทียบความเร็วของ GPS ที่ความเร็วคงที่ 20 km/h



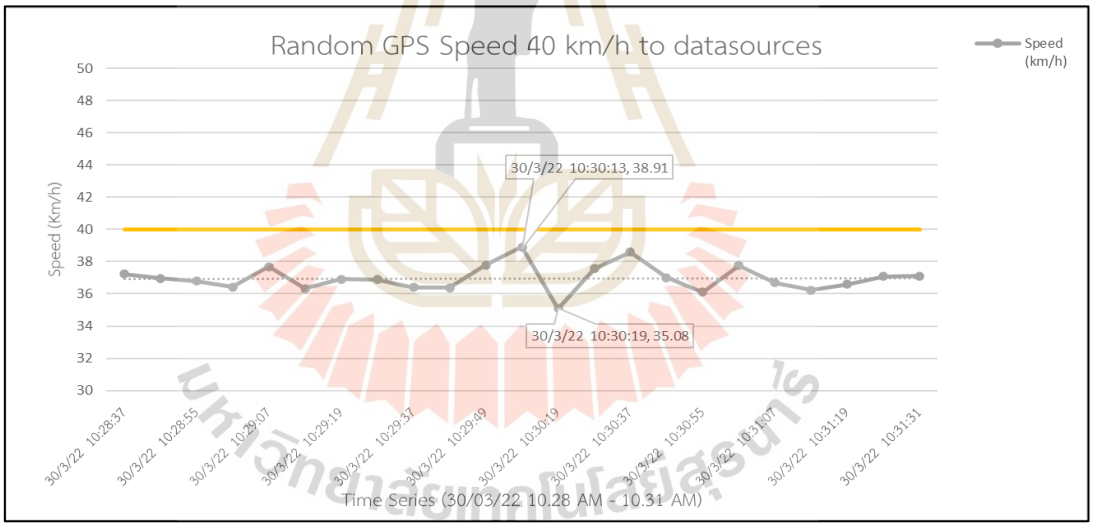
รูปที่ 4.5 กราฟเปรียบเทียบความผิดพลาดความเร็วของ GPS ที่ความเร็วคงที่ 20 km/h



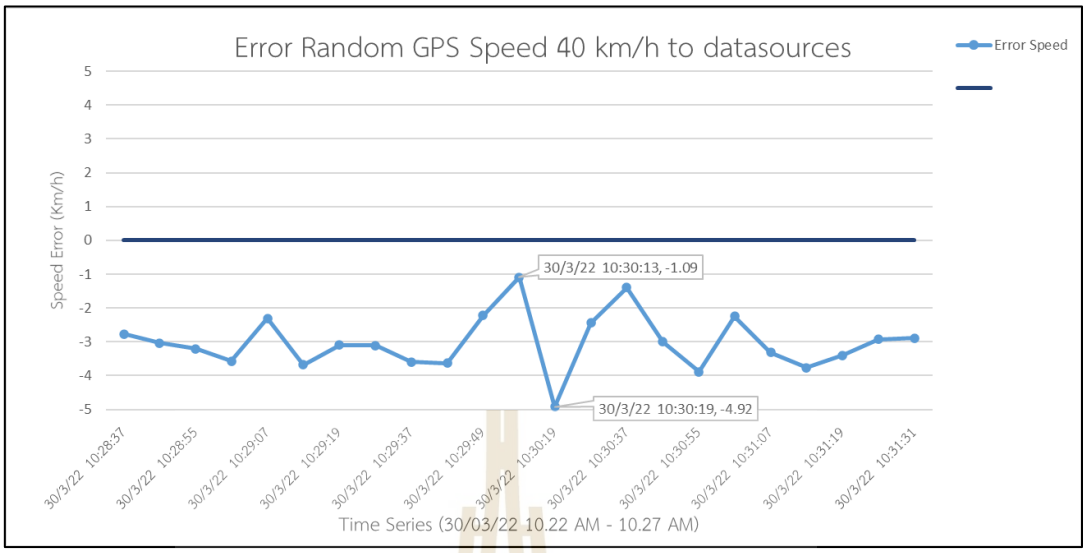
รูปที่ 4.6 กราฟเปรียบเทียบความเร็วของ GPS ที่ความเร็วคงที่ 30 km/h



รูปที่ 4.7 กราฟเปรียบเทียบความผิดพลาดความเร็วของ GPS ที่ความเร็วคงที่ 30 km/h



รูปที่ 4.8 กราฟเปรียบเทียบความเร็วของ GPS ที่ความเร็วคงที่ 40 km/h



รูปที่ 4.9 กราฟเปรียบเทียบความผิดพลาดความเร็วของ GPS ที่ความเร็วคงที่ 40 km/h

ตารางที่ 4.1 สรุปการทดสอบการตรวจพิกัดตำแหน่งทางภูมิศาสตร์

Test	Random Speed (Km/h)	Max Position Error (m)	Max Speed Error (Km/h)
Static	-	3.25	0.19
Dynamic	20	-	2.72
	30	-	3.03
	40	-	4.92

การทดสอบแบบ Static Test ทำการทดสอบตำแหน่งขณะอยู่กับที่ จุดทดสอบที่ 11 มีค่าความผิดพลาดตำแหน่งสูงสุดอยู่ 3.25 เมตร ดังรูปที่ 4.2 และจุดทดสอบที่ 2 และ 11 มีค่าความผิดพลาดความเร็วสูงสุดอยู่ที่ 0.19 กิโลเมตรต่อชั่วโมง ดังรูปที่ 4.3

การทดสอบแบบ Dynamic Test ทำการทดสอบขณะเคลื่อนที่ด้วยความเร็วคงที่ 20 กิโลเมตรต่อชั่วโมง พบว่ามีค่าความเร็วจากการทดสอบสูงสุด 21.22 และต่ำสุด 17.28 กิโลเมตรต่อชั่วโมง ดังรูปที่ 4.4 และมีค่าความผิดพลาดความเร็วสูงสุดอยู่ที่ 2.72 กิโลเมตรต่อชั่วโมง ดังรูปที่ 4.5

ทำการทดสอบขณะเคลื่อนที่ด้วยความเร็วคงที่ 30 กิโลเมตรต่อชั่วโมง พบว่ามีค่าความเร็วจากการทดสอบสูงสุด 30.98 และต่ำสุด 26.97 กิโลเมตรต่อชั่วโมง ดังรูปที่ 4.6 และมีค่าความผิดพลาดความเร็วสูงสุดอยู่ที่ 3.03 กิโลเมตรต่อชั่วโมง ดังรูปที่ 4.7

ทำการทดสอบขณะเคลื่อนที่ด้วยความเร็วคงที่ 40 กิโลเมตรต่อชั่วโมง พบว่ามีค่าความเร็วจากการทดสอบสูงสุด 38.91 และต่ำสุด 35.08 กิโลเมตรต่อชั่วโมง ดังรูปที่ 4.8 และมีค่าความผิดพลาดความเร็วสูงสุดอยู่ที่ 4.92 กิโลเมตรต่อชั่วโมง ดังรูปที่ 4.9

การทดสอบเก็บค่าความเร็ว 20 30 และ 40 กิโลเมตรต่อชั่วโมง ขณะเคลื่อนที่ด้วยความเร็วคงที่ผ่านเส้นทางทั้ง 12 จุด เพื่อเปรียบเทียบความผิดพลาดความเร็วของ GPS ที่ความเร็วคงที่ ผลการทดสอบทั้งสองการทดสอบสามารถสรุปได้ว่าความผิดพลาดพิกัดตำแหน่งของ GPS ขณะอยู่กับที่สูงสุดมีค่าเท่ากับ 3.25 เมตร ค่าความผิดพลาดความเร็วของ GPS ที่ความเร็วคงที่ 20 30 และ 40 กิโลเมตรต่อชั่วโมง อยู่ที่ 2.72 3.03 และ 4.92 กิโลเมตรต่อชั่วโมงตามลำดับ ดังตารางที่ 4.1

4.2.2 การทดสอบเซนเซอร์ Magnetic Reed Switch BR-1021

รูปที่ 4.10 ทดสอบระยะการตรวจจับเซนเซอร์ Magnetic Reed Switch BR-1021

ตารางที่ 4.2 สรุปผลการทดสอบเซนเซอร์ Magnetic Reed Switch BR-1021

Door Sensor 1					
Distance (mm)	0	10	20	21	25
Status LED	OFF	OFF	OFF	ON	ON
Door Sensor 2					
Distance (mm)	0	10	20	21	25
Status LED	OFF	OFF	OFF	ON	ON
Door Sensor 3					
Distance (mm)	0	10	20	21	25
Status LED	OFF	OFF	OFF	ON	ON

การทดสอบระยะการตรวจจับของเซนเซอร์ Magnetic Reed Switch BR-1021 โดยทำการวัดระยะห่างระหว่างหัวเซนเซอร์กับหัวแม่เหล็ก เมื่อแม่เหล็กอยู่ในระยะการตรวจจับแม่เหล็กไฟ LED จะดับลง และเมื่อแม่เหล็กอยู่นอกระยะการตรวจจับของหัวเซนเซอร์ ไฟ LED จะสว่าง ดังรูปที่ 4.10 ผลการทดสอบพบว่า ระยะการตรวจจับทั้ง 3 เซนเซอร์อยู่ที่ 0 ถึง 20 มิลลิเมตร ซึ่งระยะช่องว่างระหว่างประตูรถบรรทุกน้ำแข็งอยู่ที่ 16 มิลลิเมตร สรุปได้ดังตารางที่ 4.2

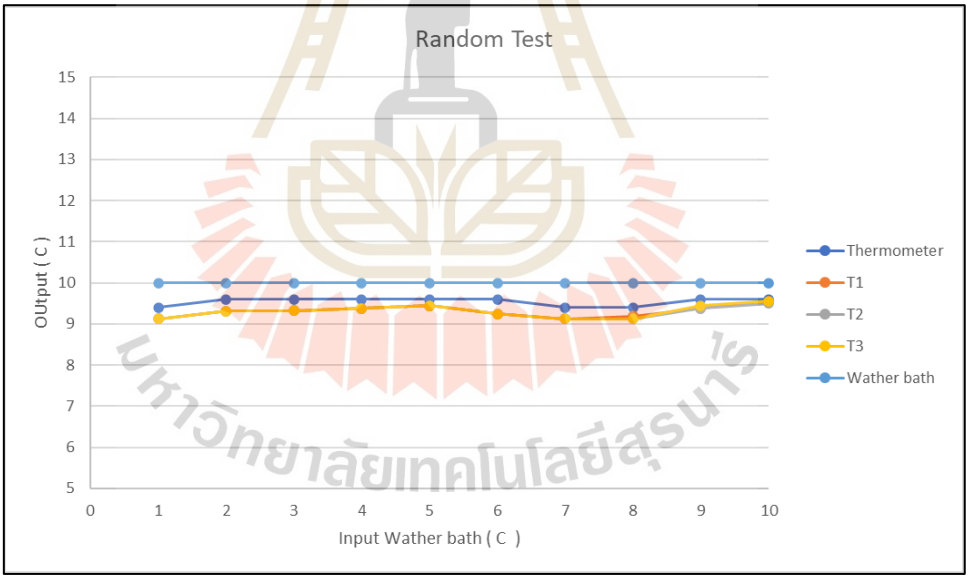
4.2.3 การทดสอบเซนเซอร์ DS18B20

ได้พิจารณาเลือกใช้ เครื่อง Water Bath ชุดทดลองการแลกเปลี่ยนความเย็น ทดสอบการตรวจวัดอุณหภูมิ ย่านอุณหภูมิต่ำ ช่วง 5 ถึง 20 °C ดังรูปที่ 4.11 และWater Bath ชุดทดลองการแลกเปลี่ยนความร้อนในการทดสอบการตรวจวัดอุณหภูมิ ย่านอุณหภูมิสูง ช่วง 30 ถึง 80 °C ดังรูปที่ 4.12

รูปที่ 4.11 เครื่อง Water Bath ชุดทดลองการแลกเปลี่ยนความเย็น

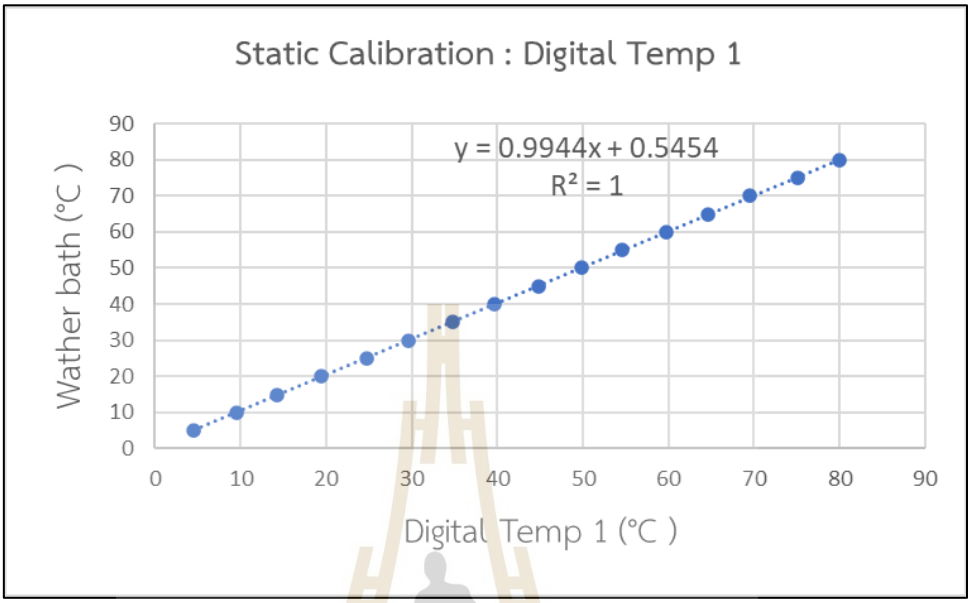
รูปที่ 4.12 เครื่อง Water Bath ชุดทดลองการแลกเปลี่ยนความร้อน

1. ผลการทดสอบสุ่มตัวอย่าง (Random Test)

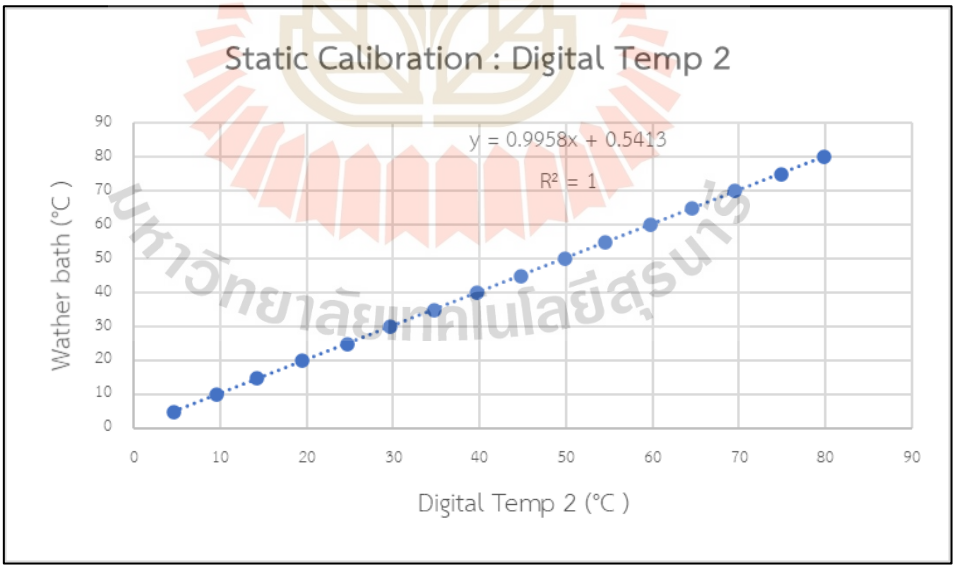


รูปที่ 4.13 กราฟ Random Test ทั้งสามเซนเซอร์ตรวจวัดอุณหภูมิ

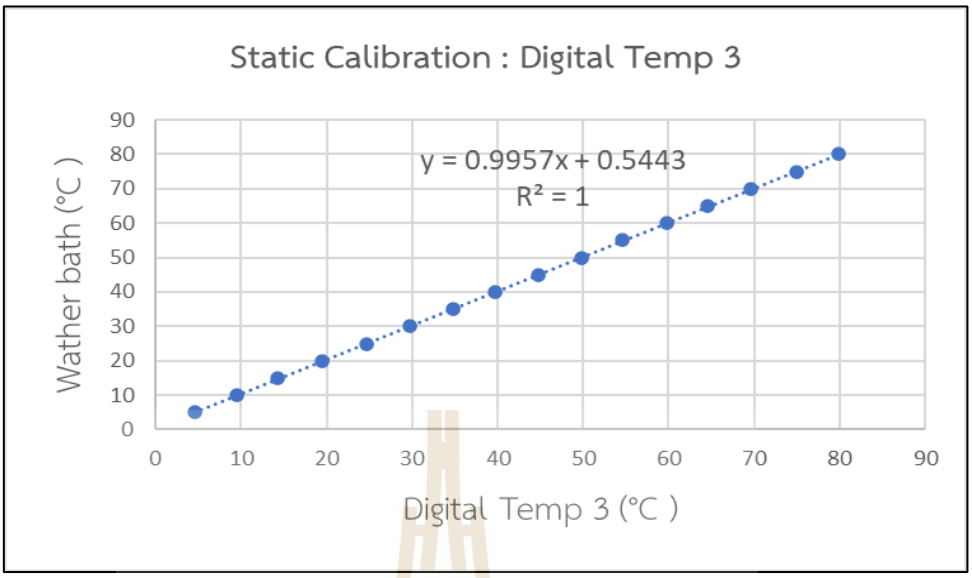
2. ผลการทดสอบเทียบสถิติ (Static Calibration Test)



รูปที่ 4.14 กราฟ Static Calibration Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 1

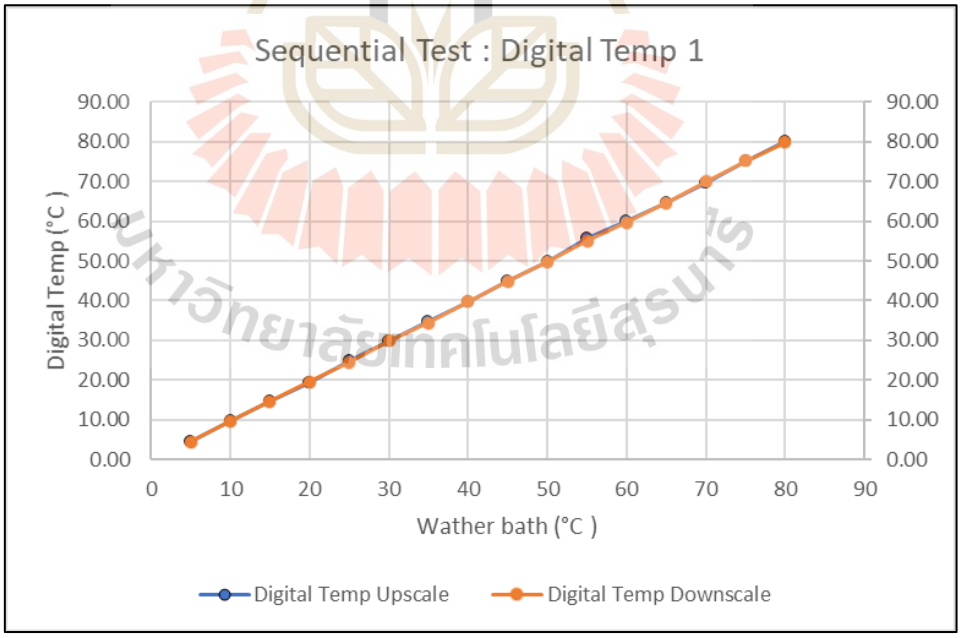


รูปที่ 4.15 กราฟ Static Calibration Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 2

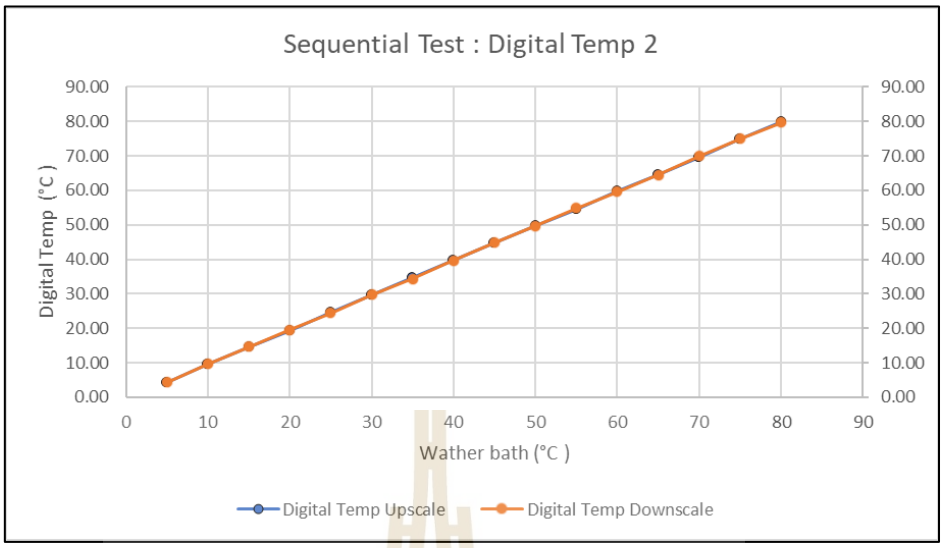


รูปที่ 4.16 กราฟStatic Calibration Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 3

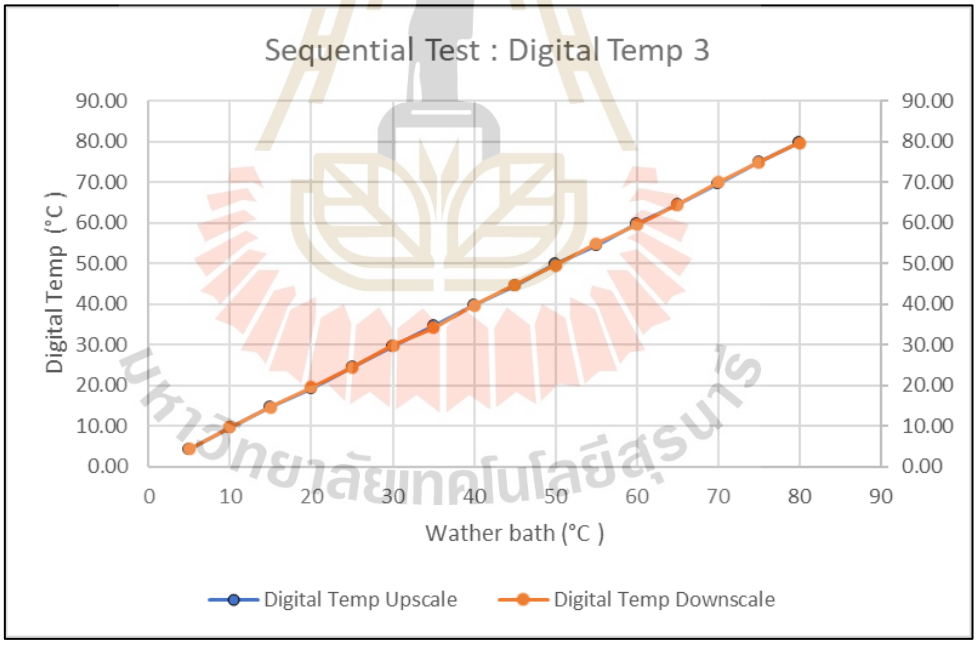
3. ผลการทดสอบแบบสะสม (Sequential Test)



รูปที่ 4.17 กราฟ Sequential Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 1



รูปที่ 4.18 กราฟ Sequential Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 2



รูปที่ 4.19 กราฟ Sequential Test เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 3

ตารางที่ 4.3 สรุปค่าความผิดพลาดของการทดสอบตรวจวัดอุณหภูมิ

Details Error	Thermometer (°C)	DS18B20 Sensors (°C)		
		1	2	3
Sensivity error (eS)	0.400	0.750	0.810	0.810
Zero shift error (eZ)	0.280	0.550	0.540	0.540
Linearity error (eL)	0.008	0.006	0.004	4.000
Random error (eR)	0.600	0.870	0.870	0.870
Bias error (eB)	0.460	0.690	0.700	0.690
Hysteresis error (eH)	0.200	0.690	0.380	0.500
Overall instument error	0.920	1.600	1.530	1.560

การทดสอบ Random Test โดยพิจารณาที่ อุณหภูมิ 10°C ซึ่งจะสังเกตเห็นได้ว่าค่าอุณหภูมิทั้งสามเซนเซอร์วัดอุณหภูมิต่ำกว่าอุณหภูมิที่กำหนด อยู่ในช่วง 9°C ถึง 10°C ดังรูป 4.13

การทดสอบ Static Calibration จะได้สมการเชิงเส้นโดยที่ y คือค่าอุณหภูมิที่ควรจะเป็น X คือค่าที่ได้จากเซนเซอร์ตรวจวัดอุณหภูมิ

เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 1 ; Y1 = 0.9944x1 + 0.5454 (1) ดังรูปที่ 4.14

เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 2 ; Y2 = 0.9958x2 + 0.5413 (2) ดังรูปที่ 4.15

เซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 3 ; Y3 = 0.9957x3 + 0.5443 (3) ดังรูปที่ 4.16

การทดสอบ Sequential Test เป็นวิธีการนำเซนเซอร์วัดอุณหภูมิตรวจวัดอุณหภูมิจากอุณหภูมิสูงไปอุณหภูมิต่ำเปรียบเทียบกับการนำเซนเซอร์วัดอุณหภูมิตรวจวัดอุณหภูมิจากต่ำไปอุณหภูมิสูง ดังรูปที่ 4.17 4.18 และ 4.19 ตามลำดับ

$$u_c = \sqrt{e_1^2 + e_2^2 + \ldots + e_m^2}$$

รูปที่ 4.20 สมการหาค่า Overall instrument error

จากการทดสอบทั้งสามการทดสอบสามารถหาค่าความผิดพลาดต่าง ๆ ของเซนเซอร์ตรวจวัดอุณหภูมิได้ดังตารางที่ 4.3 และนำค่าความผิดพลาดต่าง ๆ มาหาผลสรุปค่าความผิดพลาด (Overall

instrument error) ดังสมการในรูปที่ 4.20 ซึ่งสามารถสรุปค่าความผิดพลาดเซนเซอร์ตรวจวัดอุณหภูมิตัวที่ 1 2 และ3 มีค่าดังต่อไปนี้ 1.60°C 1.53°C และ1.56°C ตามลำดับ

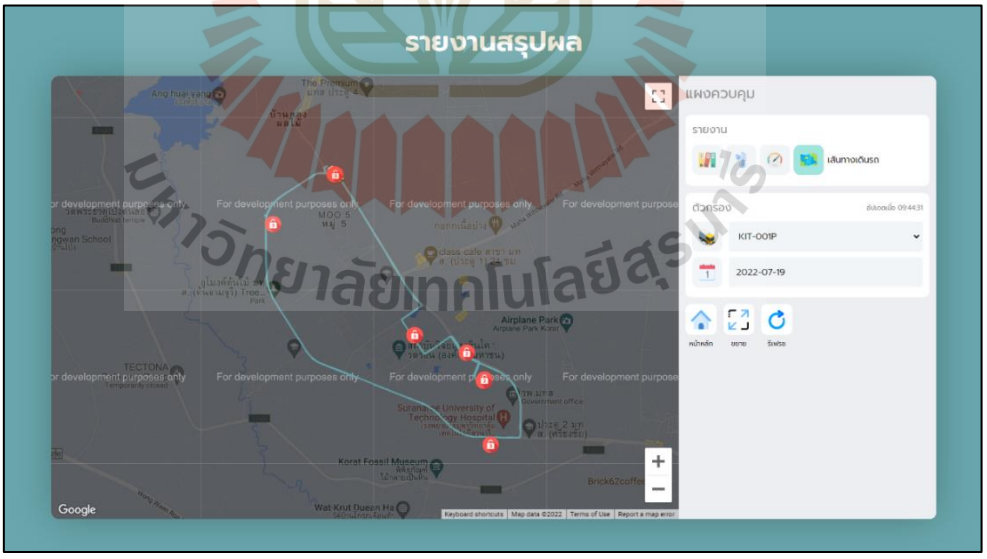
4.3 การทดสอบภาคสนาม

4.3.1 การทดสอบตำแหน่งทางภูมิศาสตร์ภาคสนาม

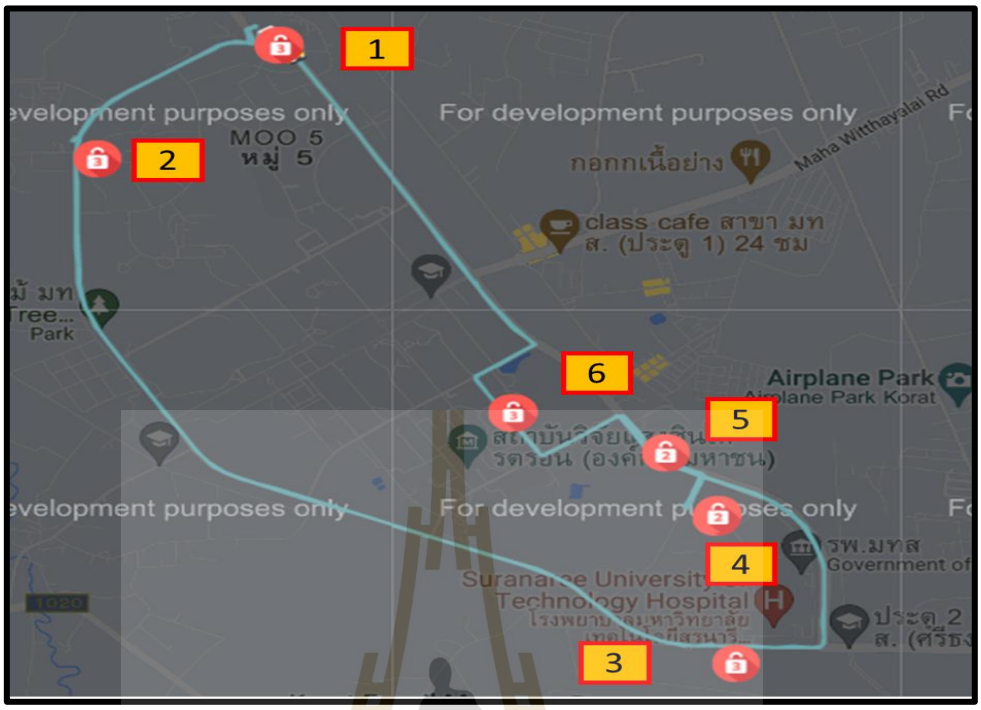
ตารางที่ 4.4 สรุปการทดสอบตำแหน่งทางภูมิศาสตร์ภาคสนาม

ครั้งที่	ตำแหน่ง	สถานที่	พิกัดตำแหน่งอ้างอิง	พิกัดตำแหน่งทดสอบ	ความผิดพลาดตำแหน่ง (m)	เซนเซอร์ประตู		
						1	2	3
1	1	โรงอาหารกาสะลองคำ	14.896299, 102.0131039	14.896312,102.013121	2.45	-	-	เปิด
	2	ฟาร์มมอ	14.890258, 102.005195	14.890246,102.00517	3.04	-	-	เปิด
	3	โรงกัญชา	14.863147, 102.032836	14.863167,102.03281	3.11	-	-	เปิด
	4	ถนนโรงพยาบาล	14.871089, 102.031962	14.871052,102.03200	2.08	เปิด	เปิด	-
	5	โรงเรียนสุรวิวัฒน์	14.874435,102.029758	14.874415,102.029774	3.06	เปิด	เปิด	-
	6	เทคโนธานี	14.876594, 102.023195	14.876587,102.02316	2.90	เปิด	เปิด	เปิด
2	1	โรงอาหารกาสะลองคำ	14.896299, 102.0131039	14.896305,102.01309	1.66	-	-	เปิด
	2	ฟาร์มมอ	14.890258, 102.005195	14.89027,102.005200	1.29	-	-	เปิด
	3	โรงกัญชา	14.863147, 102.032836	14.863116,102.03286	3.14	-	-	เปิด
	4	ถนนโรงพยาบาล	14.871089, 102.031962	14.871111,102.031974	3.03	เปิด	เปิด	-
	5	โรงเรียนสุรวิวัฒน์	14.874435,102.029758	14.874427,102.02979	3.04	เปิด	เปิด	-
	6	เทคโนธานี	14.876594, 102.023195	14.876590,102.023169	2.88	เปิด	เปิด	เปิด
3	1	โรงอาหารกาสะลองคำ	14.896299, 102.0131039	14.896312,102.01312	2.50	-	-	เปิด
	2	ฟาร์มมอ	14.890258, 102.005195	14.890288,102.005205	3.20	-	-	เปิด
	3	โรงกัญชา	14.863147, 102.032836	14.863152,102.03282	1.67	-	-	เปิด
	4	ถนนโรงพยาบาล	14.871089, 102.031962	14.871072,102.031950	2.75	เปิด	เปิด	-
	5	โรงเรียนสุรวิวัฒน์	14.874435,102.029758	14.874456,102.02977	2.61	เปิด	เปิด	-
	6	เทคโนธานี	14.876594, 102.023195	14.876616,102.0232	2.49	เปิด	เปิด	เปิด

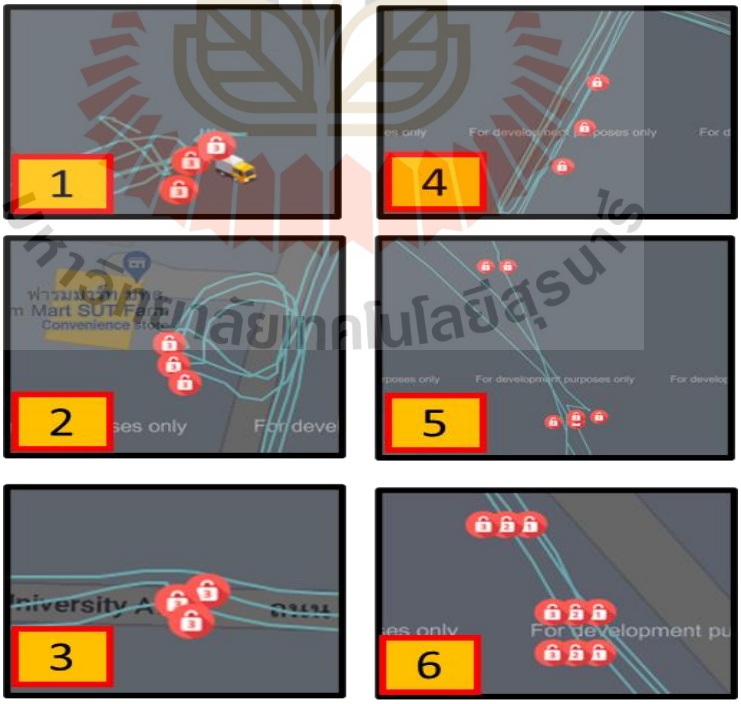
รูปที่ 4.21 ตำแหน่งการทดสอบภาคสนาม



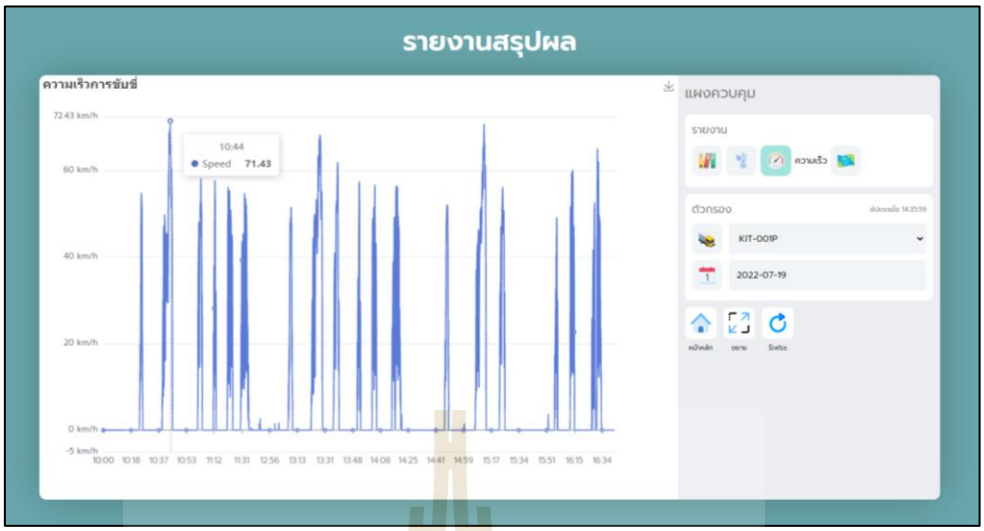
รูปที่ 4.22 เส้นทางการเดินรถภาคสนามบนเว็บแอปพลิเคชัน



รูปที่ 4.23 ตำแหน่งการทดสอบภาคสนามบนเว็บแอปพลิเคชัน



รูปที่ 4.24 ตำแหน่งการเปิดประตูการทดสอบภาคสนาม บนเว็บแอปพลิเคชัน



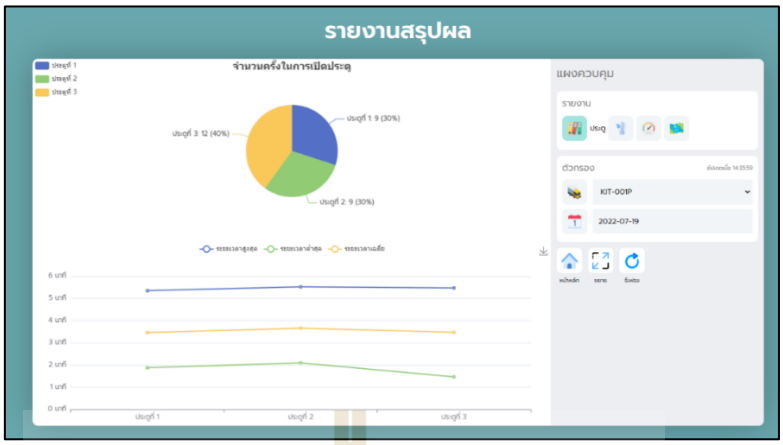
รูปที่ 4.25 กราฟความเร็วการทดสอบภาคสนามบนเว็บแอปพลิเคชัน

ผลการทดสอบภาคสนามแสดงให้เห็นถึงเส้นทางการเดินรถบนเว็บแอปพลิเคชัน ดังรูปที่ 4.22 และ 4.23 เป็นไปตามตำแหน่งการทดสอบภาคสนามที่กำหนดดังรูปที่ 4.21 และเมื่อทำการขยายแผนที่บนเว็บแอปพลิเคชันดังรูปที่ 4.24 พบว่าจำนวนตำแหน่งการเปิดประตูในแต่ละจุดเป็นไปตามตารางที่ 4.4 กล่าวคือ จุดที่ 1 โรงอาหารกาสะลองคำ ทำการเปิดเฉพาะประตูที่ 3 จุดที่ 2 ฟาร์มมอ ทำการเปิดเฉพาะประตูที่ 3 โรงกัญชาทำการเปิดเฉพาะประตูที่ 3 จุดที่ 4 ถนนโรงพยาบาล ทำการเปิดประตูที่ 1 และ 2 จุดที่ 5 โรงเรียนสุรวิวัฒน์ ทำการเปิดประตูที่ 1 และ 2 จุดที่ 6 เทคโนธานี ทำการเปิดทั้ง 3 ประตู และจากการทดสอบ มีความผิดพลาดของตำแหน่งสูงสุดอยู่ที่ 3.20 เมตร ในรอบการทดสอบที่ 3 ซึ่งมีสาเหตุมาจากการทดสอบรอบที่ 3 มีฝนตกและเมฆมากตลอดการทดสอบ ส่งผลให้ประสิทธิภาพในการตรวจจับสัญญาณจีพีเอสลดน้อยลง ทั้งนี้ ยังมีความผิดพลาดที่เกิดจากคนขับรถในการจอดรถแต่ละตำแหน่งจุดทดสอบด้วย และสามารถตรวจสอบความเร็วในการขับรถตู้บรรทุกน้ำแข็งแสดงผ่านเว็บแอปพลิเคชันดังรูปที่ 4.25

4.3.2 การทดสอบภาคสนามเซนเซอร์ตรวจจับประตู

ตารางที่ 4.5 สรุปการทดสอบเซนเซอร์ตรวจจับประตูภาคสนาม

ครั้งที่	ตำแหน่ง	สถานที่	เวลาเริ่ม	เวลาสิ้นสุด	ระยะเวลา (นาที)	เซนเซอร์ประตู		
						1	2	3
1	1	โรงอาหารกาสะลองคำ	10:15:19	10:16:48	0:01:29	-	-	เปิด
	2	ฟาร์มมอ	10:34:56	10:36:24	0:01:28	-	-	เปิด
	3	โรงกัญชา	10:57:14	10:59:02	0:01:48	-	-	เปิด
	4	ถนนโรงพยาบาล	11:08:36	11:10:55	0:02:19	เปิด	เปิด	-
	5	โรงเรียนสุรวิวัฒน์	11:19:18	11:21:11	0:01:53	เปิด	เปิด	-
	6	เทคโนธานี	11:27:57	11:30:00	0:02:03	เปิด	เปิด	เปิด
2	1	โรงอาหารกาสะลองคำ	13:05:00	13:08:20	0:03:20	-	-	เปิด
	2	ฟาร์มมอ	13:18:51	13:22:12	0:03:21	-	-	เปิด
	3	โรงกัญชา	13:33:13	13:36:36	0:03:23	-	-	เปิด
	4	ถนนโรงพยาบาล	13:48:46	13:51:56	0:03:10	เปิด	เปิด	-
	5	โรงเรียนสุรวิวัฒน์	13:59:37	14:02:46	0:03:09	เปิด	เปิด	-
	6	เทคโนธานี	14:12:24	14:15:51	0:03:27	เปิด	เปิด	เปิด
3	1	โรงอาหารกาสะลองคำ	14:43:07	14:48:22	0:05:15	-	-	เปิด
	2	ฟาร์มมอ	15:01:50	15:07:08	0:05:18	-	-	เปิด
	3	โรงกัญชา	15:18:12	15:23:31	0:05:19	-	-	เปิด
	4	ถนนโรงพยาบาล	15:53:58	15:59:19	0:05:21	เปิด	เปิด	-
	5	โรงเรียนสุรวิวัฒน์	16:07:03	16:12:10	0:05:07	เปิด	เปิด	-
	6	เทคโนธานี	16:20:32	16:26:00	0:05:28	เปิด	เปิด	เปิด



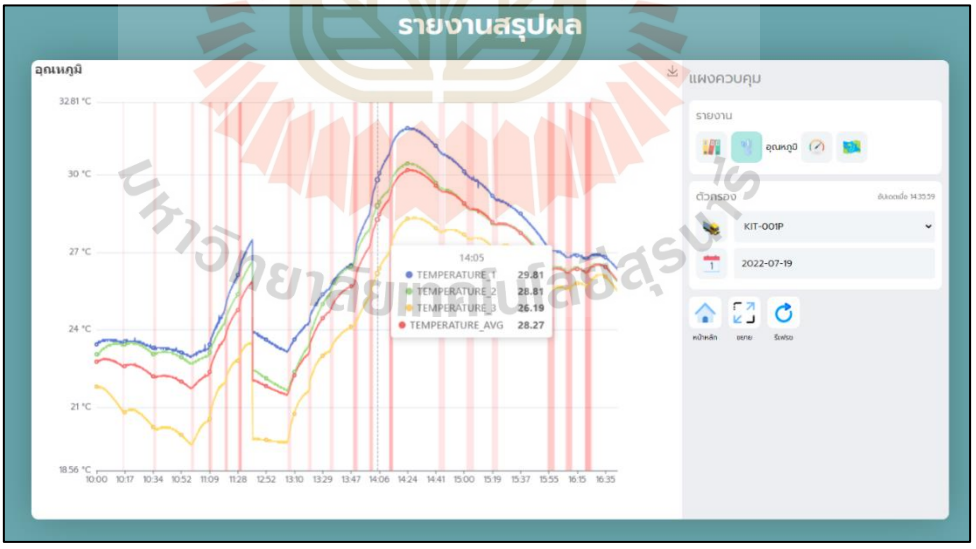
รูปที่ 4.26 ผลจำนวนการเปิดประตูภาคสนามบนเว็บแอปพลิเคชัน

จากรูปที่ 4.26 แสดงผลการทดสอบภาคสนามเกี่ยวกับจำนวนการเปิดประตู และระยะเวลาในการเปิดประตู จะสังเกตเห็นได้ว่าการทดสอบภาคสนามมีการเปิดประตูที่ 1 จำนวน 9 ครั้ง คิดเป็น 30 เปอร์เซ็นต์ของการเปิดประตูทั้งหมดการเปิดประตูที่ 2 จำนวน 9 ครั้ง คิดเป็น 30 เปอร์เซ็นต์ ของการเปิดประตูทั้งหมด และการเปิดประตูที่ 3 จำนวน 12 ครั้ง คิดเป็น 40 เปอร์เซ็นต์ของการเปิดประตูทั้งหมด ซึ่งมีความสอดคล้องกันกับตารางที่ 4.5 และเวลาสูงสุดในการเปิดแต่ละบานอยู่ที่ 5.35 5.52 และ 5.47 นาที แต่เวลาในการบันทึกในตารางที่ 4.5 เวลาสูงสุดอยู่ที่ 5.28 นาที มีความแตกต่างของเวลา 7 ถึง 24 วินาที เกิดความผิดพลาดจากผู้ทำการทดสอบขณะเปลี่ยนสถานะการเปิดปิดประตู ซึ่งส่งผลให้เกิดความคลาดเคลื่อนของเวลา

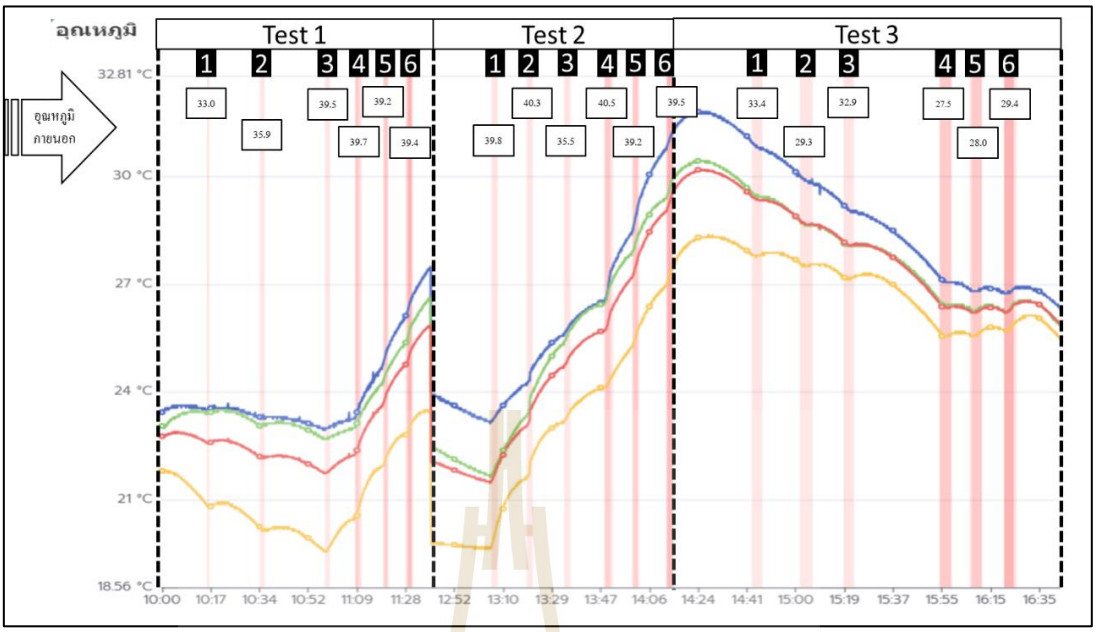
4.3.3 การทดสอบภาคสนามเซนเซอร์อุณหภูมิ

ตารางที่ 4.6 สรุปการทดสอบเซนเซอร์อุณหภูมิภาคสนาม

ครั้งที่	ตำแหน่ง	สถานที่	เวลาเริ่ม	เวลาสิ้นสุด	ระยะเวลา (นาที)	เซนเซอร์ประดู			เซนเซอร์ DS18B20			DE-3003 DIGITAL THERMOMETER TYPE K				ความผิดพลาด เซนเซอร์ DS18B20		
						1	2	3	1	2	3	1	2	3	นอกรอ	1	2	3
1	1	โรงอาหารกาสะลองคำ	10:15:19	10:16:48	0:01:29	-	-	เปิด	23.25	23.44	20.94	23.50	23.50	21.50	33.00	0.25	-0.06	-0.56
	2	ฟาร์มมอ	10:34:56	10:36:24	0:01:28	-	-	เปิด	23.50	23.06	20.19	23.80	23.80	21.30	35.90	0.30	-0.74	-1.11
	3	โรงกัญชา	10:57:14	10:59:02	0:01:48	-	-	เปิด	23.00	22.69	19.69	23.50	24.20	21.10	39.50	0.50	-1.51	-1.41
	4	ถนนโรงพยาบาล	11:08:36	11:10:55	0:02:19	เปิด	เปิด	-	23.44	23.13	20.56	24.20	23.50	20.90	39.70	0.76	-0.37	-0.34
	5	โรงเรียนสุรวิวัฒน์	11:19:18	11:21:11	0:01:53	เปิด	เปิด	-	24.94	24.44	22.06	25.20	24.80	22.80	39.20	0.26	-0.36	-0.74
	6	เทคโนธานี	11:27:57	11:30:00	0:02:03	เปิด	เปิด	เปิด	26.25	25.50	22.88	26.90	26.40	23.80	39.40	0.65	-0.90	-0.92
2	1	โรงอาหารกาสะลองคำ	13:05:00	13:08:20	0:03:20	-	-	เปิด	23.25	21.75	19.88	23.50	22.80	21.30	39.80	0.25	-1.05	-1.42
	2	ฟาร์มมอ	13:18:51	13:22:12	0:03:21	-	-	เปิด	24.31	23.44	21.69	24.90	23.80	22.50	40.30	0.59	-0.36	-0.81
	3	โรงกัญชา	13:33:13	13:36:36	0:03:23	-	-	เปิด	25.75	25.56	23.31	25.90	25.90	23.60	35.50	0.15	-0.34	-0.29
	4	ถนนโรงพยาบาล	13:48:46	13:51:56	0:03:10	เปิด	เปิด	-	27.00	26.81	24.25	27.40	28.30	25.10	40.50	0.40	-1.49	-0.85
	5	โรงเรียนสุรวิวัฒน์	13:59:37	14:02:46	0:03:09	เปิด	เปิด	-	28.94	28.19	25.56	29.70	28.90	26.80	39.20	0.76	-0.71	-1.24
	6	เทคโนธานี	14:12:24	14:15:51	0:03:27	เปิด	เปิด	เปิด	30.94	29.56	27.19	31.00	30.10	28.70	39.50	0.06	-0.54	-1.51
3	1	โรงอาหารกาสะลองคำ	14:43:07	14:48:22	0:05:15	-	-	เปิด	30.81	29.44	27.81	31.00	30.00	27.90	33.40	0.19	-0.56	-0.09
	2	ฟาร์มมอ	15:01:50	15:07:08	0:05:18	-	-	เปิด	29.88	28.63	27.50	30.00	29.00	28.50	29.30	0.12	-0.37	-1.00
	3	โรงกัญชา	15:18:12	15:23:31	0:05:19	-	-	เปิด	29.06	28.06	27.13	29.50	28.50	28.00	32.90	0.44	-0.44	-0.87
	4	ถนนโรงพยาบาล	15:53:58	15:59:19	0:05:21	เปิด	เปิด	-	27.06	26.44	25.56	27.50	27.00	26.40	27.50	0.44	-0.56	-0.84
	5	โรงเรียนสุรวิวัฒน์	16:07:03	16:12:10	0:05:07	เปิด	เปิด	-	26.81	26.31	25.63	27.10	27.60	27.00	28.00	0.29	-1.29	-1.37
	6	เทคโนธานี	16:20:32	16:26:00	0:05:28	เปิด	เปิด	เปิด	26.81	26.25	25.75	27.10	26.90	26.70	29.40	0.29	-0.65	-0.95



รูปที่ 4.27 ผลอุณหภูมิภาคสนามบนเว็บแอปพลิเคชัน



รูปที่ 4.28 กราฟผลอุณหภูมิภาคสนามบนเว็บแอปพลิเคชัน

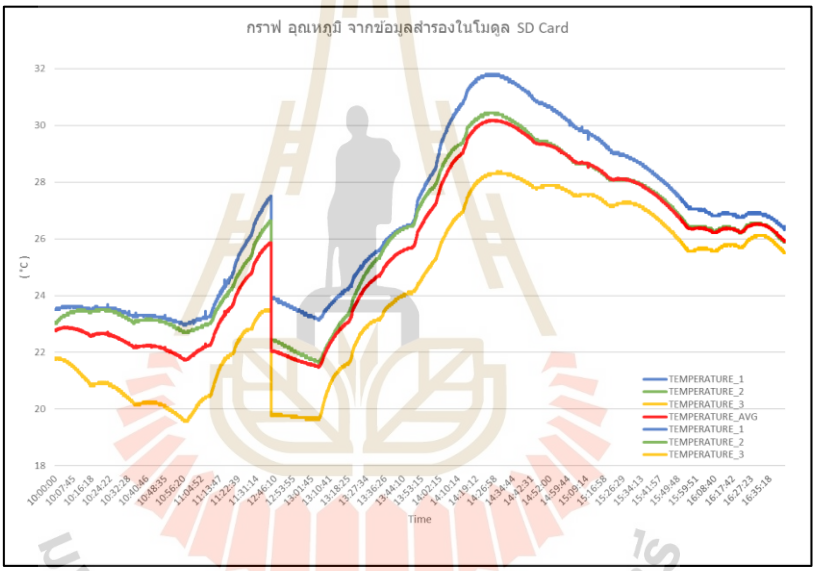
การทดสอบภาคสนามแสดงให้เห็นอุณหภูมิที่เปลี่ยนแปลงไปตามช่วงเวลาดังรูปที่ 4.27 บนเว็บแอปพลิเคชันและทำการวิเคราะห์ผลกราฟอุณหภูมิเทียบกับตารางที่ 4.6 ดังรูปที่ 4.28 จะมีการทดสอบทั้งหมด 3 รอบและมีการใช้เครื่องมือตรวจวัดอุณหภูมิ Digital Thermometer Type K ในการตรวจวัดอุณหภูมิบริเวณตำแหน่งติดตั้งเซนเซอร์ตรวจวัดอุณหภูมิภายในรถตู้น้ำแข็ง และอุณหภูมิภายนอกรถตู้บรรทุกน้ำแข็ง สังเกตเห็นได้ว่าการทดสอบรอบที่ 1 มีขนาดความกว้างแถบสีน้อยที่สุดและการทดสอบรอบที่ 3 มีขนาดแถบสีกว้างที่สุดนั้นเป็นเพราะว่าได้ทำการออกแบบการทดสอบให้รอบที่ 1 ใช้เวลาในการเปิดประตูแต่ละจุด 1 นาที รอบที่ 2 ใช้เวลาในการเปิดประตูแต่ละจุด 3 นาที และรอบที่ 3 ใช้เวลาในการเปิดประตูแต่ละจุด 5 นาที ในแต่ละรอบจะสังเกตเห็นได้ว่า แถบสีจุดที่ 1 2 และ3 ที่เป็นการเปิดประตูบานเดียวจะเป็นสีแดงอ่อนที่สุด แถบสี่จุดที่ 4 และ5ที่เป็นการเปิดประตู 2 บานพร้อมกันจะเป็นสีแดงเข้มปานกลาง ส่วนจุดที่6 แถบสีจะเป็นสีแดงเข้มที่สุดเป็นการเปิดประตูทั้ง 3 บานพร้อมกัน

การทดสอบรอบที่ 1 ดังรูปที่ 4.28 ในช่วงแรก การทดสอบจุดที่ 1 2 และ3 มีแนวโน้มอุณหภูมิสูงขึ้นในช่วงเปิดประตู และอุณหภูมิลดต่ำลง ในช่วงปิดประตูกราฟในช่วงที่สอง การทดสอบจุดที่ 4 5 และ6 มีแนวโน้มอุณหภูมิสูงขึ้นในช่วงเปิดประตู และอุณหภูมิค่อย ๆ สูงขึ้นในช่วงปิดประตู เนื่องจาก อุณหภูมิภายนอกเริ่มสูงขึ้นประมาณ 3.5 – 6.5 °C ส่งผลให้อุณหภูมิภายในตู้รถบรรทุกน้ำแข็งสูญเสียความเย็น

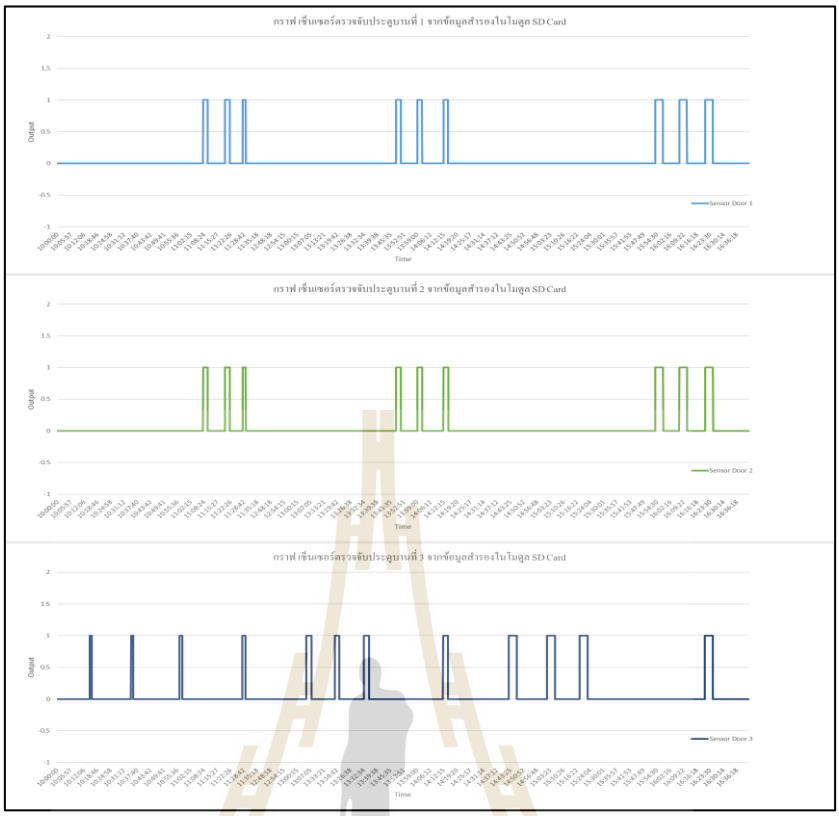
การทดสอบรอบที่ 2 ดังรูปที่ 4.28 ก่อนทำการทดสอบรอบที่ 2 ได้ทำการปิดระบบชุดอุปกรณ์และปิดประตูตั้งแต่เวลา 11.28 น. และเริ่มทดสอบรอบที่ 2 เวลา 12.52 น. ตลอดช่วงการทดสอบกราฟการทดสอบมีแนวโน้มอุณหภูมิสูงขึ้นในช่วงเปิดประตู และอุณหภูมิค่อย ๆ สูงขึ้นในช่วงปิดประตู เนื่องจากอุณหภูมิภายนอกที่สูง ส่งผลให้อุณหภูมิภายในตู้รถบรรทุกน้ำแข็งสูญเสียความเย็น

การทดสอบรอบที่ 3 ดังรูปที่ 4.28 ขณะทำการทดสอบมีฝนตกตลอดเวลา กราฟมีแนวโน้มอุณหภูมิสูงขึ้นเล็กน้อยในช่วงเปิดประตู และอุณหภูมิลดต่ำลงในช่วงปิดประตู ตลอดการทดสอบกราฟอุณหภูมิมีแนวทางอุณหภูมิต่ำ ลงตลอดเวลา นั้น เป็นเพราะอุณหภูมิภายนอกลดต่ำลงประมาณ 6 - 12 ℃ เนื่องจากฝนตก ทำให้อุณหภูมิภายในตู้บรรทุกน้ำแข็งสูญเสียความเย็นลดน้อยลง

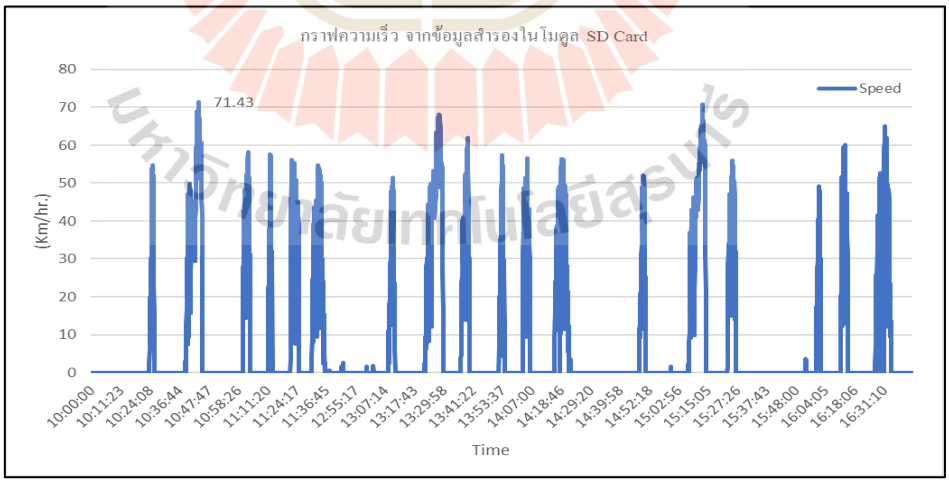
4.3.4 การทดสอบภาคสนามระบบบันทึกข้อมูลสำรอง



รูปที่ 4.29 กราฟ อุณหภูมิจากข้อมูลสำรองในโมดูล SD Card



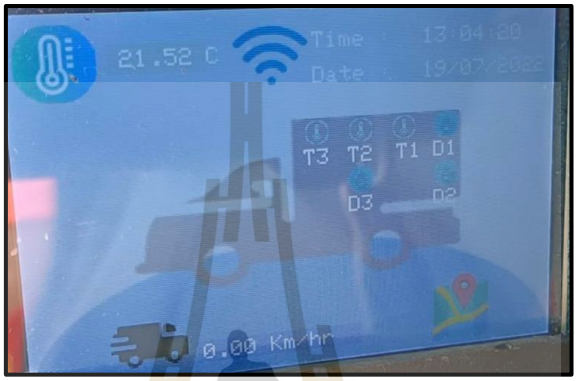
รูปที่ 4.30 กราฟตรวจจับการเปิดปิดประตูจากโมดูล SD Card



รูปที่ 4.31 กราฟความเร็วจากข้อมูลสำรองโมดูล SD Card

ผลการทดสอบนำข้อมูลสำรองจากโมดูล SD Card มาพล็อตกราฟ อุณหภูมิจากข้อมูลสำรองในโมดูล SD Card ดังรูปที่ 4.29 กราฟตรวจจับการเปิดปิดประตูจากโมดูล SD Card ดังรูปที่ 4.30 และกราฟความเร็ว จากข้อมูลสำรองโมดูล SD Card ดังรูปที่ 4.31 พบว่ามีความสอดคล้องกับผลทดสอบภาคสนามบนเว็บแอปพลิเคชัน

4.3.5 การทดสอบภาคสนาม ระบบจอแสดงผลสถานะทำงานอุปกรณ์



รูปที่ 4.32 จอแสดงผลสถานะทำงานอุปกรณ์ ขณะปิดประตู

รูปที่ 4.33 จอแสดงผลสถานะทำงานอุปกรณ์ ขณะเปิดประตู

ผลการทดสอบภาคสนามสามารถแสดงสถานะทำงานของอุปกรณ์ได้ปกติดังรูปที่ 4.32 และเมื่อมีการเปิดประตู สัญลักษณ์ที่หน้าจอแสดงผลจะเปลี่ยนจากสีเขียวเป็นเปิดสีแดงแทนดังรูปที่ 4.33 ในการอัปเดต การเปลี่ยนแปลงข้อมูลทุก ๆ 3 วินาที

4.4 สรุป

ในบทนี้นำเสนอการผลการนำเสนอผลการทดสอบ Lab Tests และการทดสอบภาคสนาม

ส่วนที่ 1 การทดสอบ Lab Tests ประกอบไปด้วย

การทดสอบโมดูล GPS Ublox NEO-7M ความผิดพลาดอยู่กับที่สูงสุด 3.25 เมตร และความผิดพลาดความเร็วขณะเคลื่อนที่สูงสุด 4.92 กิโลเมตรต่อชั่วโมง ณ ความเร็วคงที่ 40 กิโลเมตรต่อชั่วโมง ซึ่งอยู่ในขอบเขตที่สามารถยอมรับได้

การทดสอบเซนเซอร์ตรวจจับ Magnetic Reed Switch BR-1021 ระยะการตรวจจับทั้ง 3 เซนเซอร์อยู่ที่ 0 ถึง 20 มม. ซึ่งระยะช่องว่างระหว่างประตูรถบรรทุกน้ำแข็งอยู่ที่ 16 มม. อยู่ในขอบเขตที่สามารถใช้ในการตรวจจับได้

การทดสอบเซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 ความผิดพลาดเซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 ตัวที่ 1 2 และ3 มีค่าความผิดพลาดตามลำดับดังนี้ 1.60℃ 1.53℃ และ1.56℃ ซึ่งอยู่ในขอบเขตที่สามารถยอมรับได้

ส่วนที่ 2 การทดสอบภาคสนาม ประกอบไปด้วย

การทดสอบภาคสนามตำแหน่งทางภูมิศาสตร์ การทดสอบรอบที่ 1 2 และ 3 มีความผิดพลาดของตำแหน่งสูงสุดอยู่ที่ 3.11 3.14 และ 3.20 เมตรตามลำดับ ซึ่งจะเห็นได้ว่าในรอบที่ 3 มีความผิดพลาดของตำแหน่งสูงสุดอาจมีผลมาจากสภาพอากาศมีเมฆมากและฝนตกตลอดการทดสอบทำให้ประสิทธิภาพในการรับสัญญาณดาวเทียมจีพีเอสลดลง

การทดสอบภาคสนามเซนเซอร์ตรวจจับประตู มีความถูกต้องในการนับจำนวนการเปิดปิดประตู และมีเวลาสูงสุดในการเปิดแต่ละบานอยู่ที่ 5.35 5.52 และ 5.47 นาที แต่เวลาในการบันทึกในตารางที่ 4.5 เวลาสูงสุดอยู่ที่ 5.28 นาที มีความแตกต่างของเวลา 7 ถึง 24 วินาที เกิดความผิดพลาดจากผู้ทำการทดสอบขณะเปลี่ยนสถานะการเปิดปิดประตูซึ่งส่งผลให้เกิดความคลาดเคลื่อนของเวลา

การทดสอบเซนเซอร์ตรวจวัดอุณหภูมิ โดยใช้เครื่องมือวัดอุณหภูมิเ DE-3003 DIGITAL THERMOMETER TYPE K ในการทดสอบเทียบวัดอุณหภูมิบริเวณตำแหน่งติดตั้งเซนเซอร์ DS18B20 ภายในรถตู้บรรทุกน้ำแข็งมีค่าความผิดพลาดเซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 ตัวที่ 1 2 และ3 มีค่าความผิดพลาดตามลำดับดังนี้ 0.76℃ 1.51℃ และ1.51℃ ตามลำดับ

การทดสอบภาคสนามระบบบันทึกข้อมูลสำรอง สามารถนำข้อมูลมาพล็อตกราฟเปรียบเทียบกันกับข้อมูลแสดงบนเว็บแอปพลิเคชันมีความสอดคล้องกัน

การทดสอบภาคสนามระบบจอแสดงผลสถานะทำงาน จอแสดงผล TFT LCD สามารถแจ้งเตือนการทำงานของชุดอุปกรณ์ต้นแบบได้ถูกต้อง

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 สรุป

งานวิทยานิพนธ์นี้ ได้นำเสนอการออกแบบและสร้างชุดอุปกรณ์ต้นแบบสำหรับระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง เพื่อนำไปใช้ตรวจสอบการสูญเสียอุณหภูมิภายในตู้รถบรรทุกน้ำแข็ง เส้นทางการเดินรถ และจำนวนการเปิดปิดประตูโดยประกอบไปด้วยส่วนสำคัญ 5 ระบบดังนี้

5.1.1 ระบบตำแหน่งทางภูมิศาสตร์

อุปกรณ์ที่เลือกใช้คือ GPS Ublox NEO-7M เป็นอุปกรณ์รับสัญญาณ จีพีเอส ที่มีช่องต่อสำหรับเสาอากาศภายนอกในการเพิ่มประสิทธิภาพในการรับสัญญาณจีพีเอส

การทดสอบ Lab Tests ทำการทดสอบโมดูล GPS Ublox NEO-7M บันทึกข้อมูลลงในโมดูล SD Card เปรียบเทียบกับตำแหน่งบน Google Maps พบว่าค่าความผิดพลาดตำแหน่ง ขณะอยู่กับที่สูงสุด 3.25 เมตร

การทดสอบภาคสนามตำแหน่งทางภูมิศาสตร์ มีการทดสอบโดยการนำค่าพิกัดที่ได้จากฐานข้อมูลออนไลน์มาเปรียบเทียบกับตำแหน่งบน Google Maps พบว่ามีค่าความผิดพลาดตำแหน่ง ขณะอยู่กับที่สูงสุด 3.20 เมตร ที่สภาพภูมิอากาศเมฆมากและฝนตกทำให้สรุปได้ว่าสภาพภูมิอากาศมีผลต่อประสิทธิภาพการรับสัญญาณดาวเทียมจีพีเอส

สรุปผลการทดสอบ Lab Tests และการทดสอบภาคสนามตำแหน่งทางภูมิศาสตร์ มีค่าความผิดพลาดของตำแหน่งไม่เกิน 5 เมตร ตามขอบเขตที่กำหนด และการแสดงผลข้อมูลเส้นทางการเดินรถและตำแหน่งการเปิดประตูบนเว็บแอปพลิเคชันสามารถแสดงผลได้อย่างต้องถูกต้อง

5.1.2 ระบบเซนเซอร์ตรวจจับประตู

อุปกรณ์ที่เลือกใช้คือเซนเซอร์ตรวจจับ Magnetic Reed Switch BR-1021 ในการตรวจจับการเปิดปิดประตูรถตู้บรรทุกน้ำแข็งพิจารณาเลือกใช้เซนเซอร์แบบตรวจจับคลื่นแม่เหล็ก เนื่องจากประตูรถตู้บรรทุกน้ำแข็งมีการเปิดปิดตลอดเวลาจึงเหมาะกับการใช้เซนเซอร์ที่ไม่สัมผัสกับประตูโดยตรงเพื่อยืดอายุการใช้งานของเซนเซอร์

การทดสอบ Lab Tests เซนเซอร์ตรวจจับ Magnetic Reed Switch BR-1021 วัดระยะการตรวจจับสูงสุดทั้ง 3 ตัวอยู่ที่ 20 มิลลิเมตร

การทดสอบภาคสนามเซนเซอร์ตรวจจับ Magnetic Reed Switch BR-1021 สามารถตรวจจับการเปิดปิดประตูได้ถูกต้องและแม่นยำตลอดระยะเวลาทดสอบภาคสนาม

สรุปผลการทดสอบ Lab Tests และการทดสอบภาคสนาม เซนเซอร์ตรวจจับประตู สามารถใช้งานในการตรวจจับการเปิดปิดประตูรถตู้บรรทุกน้ำแข็งได้ถูกต้องทุกครั้งขณะทำการทดสอบภาคสนาม

5.1.3 ระบบเซนเซอร์ตรวจวัดอุณหภูมิ

อุปกรณ์ที่เลือกใช้คือเซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 เนื่องจากมีราคาถูก สามารถกันน้ำได้และที่สำคัญสามารถทนกับช่วงอุณหภูมิต่ำได้เหมาะกับการนำไปใช้งานตรวจวัดอุณหภูมิภายในรถตู้บรรทุกน้ำแข็งที่มีสภาวะอุณหภูมิต่ำ ความชื้นสูง

การทดสอบ Lab Tests เซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 เปรียบเทียบกับเครื่อง Water Bath พบว่าความผิดพลาดเซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 ตัวที่ 1 2 และ3 มีค่าความผิดพลาดตามลำดับดังนี้ 1.60°C 1.53°C และ1.56°C

การทดสอบภาคสนามเซนเซอร์ตรวจวัดอุณหภูมิ โดยใช้รถบรรทุกน้ำแข็งบรรทุกถุงน้ำแข็งทั้งหมด 30 ถุง คิดเป็นร้อยละ 30 ของปริมาณที่รถบรรทุกน้ำแข็งสามารถบรรจุได้ และใช้เครื่องมือวัดอุณหภูมิ DE-3003 DIGITAL THERMOMETER TYPE K ในการทดสอบเทียบวัดอุณหภูมิบริเวณตำแหน่งติดตั้งเซนเซอร์ DS18B20 ภายในรถตู้บรรทุกน้ำแข็งมีค่าความผิดพลาดเซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 ตัวที่ 1 2 และ3 มีค่าความผิดพลาดตามลำดับดังนี้ 0.76°C 1.51°C และ 1.51°C ตามลำดับ

สรุปผลการทดสอบการทดสอบ Lab Tests และการทดสอบภาคสนามเซนเซอร์ตรวจวัดอุณหภูมิ DS18B20 สามารถใช้งานได้ถูกต้องและแม่นยำในขอบเขตที่กำหนดไม่เกิน 2 °C

5.1.4 ระบบบันทึกข้อมูลสำรอง

อุปกรณ์ที่เลือกใช้คือ โมดูล Micro SD Card SPI เนื่องจาก ราคาถูกและรองรับการใช้งานกับ NodeMcu 32 V2

การทดสอบภาคสนาม ระบบบันทึกข้อมูลสำรอง สามารถนำข้อมูลมาพล็อตกราฟเปรียบเทียบกันกับข้อมูลแสดงบนเว็บแอปพลิเคชันมีความสอดคล้องกัน

5.1.5 ระบบแสดงผลและการแจ้งเตือน

5.1.5.1 ระบบแสดงผล

อุปกรณ์ที่เลือกใช้คือ TFT LCD 3.5 นิ้ว สำหรับแสดงผลสถานะทำงานอุปกรณ์ เนื่องจากจอใช้พลังงานน้อย ราคาถูก และรองรับการใช้งานกับ Arduino Mega 2560 ได้โดยตรง

การทดสอบภาคสนามระบบแสดงผล การแสดงผลสถานะทำงานอุปกรณ์สามารถทำงานได้ปกติ ตลอดการทดสอบ ทำการแสดงผลใหม่ ทุก ๆ 3 วินาที

5.1.5.2 ระบบการแจ้งเตือนด้วยเสียง

อุปกรณ์ที่เลือกใช้คือ โมดูล Active Buzzer สำหรับแจ้งเตือนด้วยเสียง เนื่องจากใช้พลังงานต่ำ ราคาถูก และรองรับการใช้งานกับ Arduino

การทดสอบภาคสนามระบบการแจ้งเตือนด้วยเสียง เมื่อมีการเปิดประตูอยู่จะทำการส่งเสียงเตือนตลอดเวลาจนกว่าจะทำการปิดประตู สามารถใช้งานได้ปกติ

5.1.5.3 ระบบเว็บแอปพลิเคชัน

สามารถส่งข้อมูลจากชุดอุปกรณ์ต้นแบบผ่านเครือข่ายไร้สาย Pocket WI-FI ไปยังเซิร์ฟเวอร์ด้วยโปโตคอล MQTT และสามารถแสดงผลข้อมูลผ่านเว็บแอปพลิเคชันทำได้อย่างมีประสิทธิภาพ

5.2 ข้อเสนอแนะ

5.2.1 ข้อเสนอแนะเกี่ยวกับกล่องชุดอุปกรณ์ต้นแบบ

อย่างไรก็ตามงานวิจัยนี้เป็นเพียงต้นแบบของการออกแบบระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง ขนาดของกล่องชุดอุปกรณ์มีขนาดใหญ่ทำให้ต้องใช้พื้นที่ในการติดตั้งค่อนข้างมากในการพัฒนาต่อไปควรทำการลดขนาดกล่องควบคุมให้มีความเหมาะสมมากยิ่งขึ้น

รายการอ้างอิง

[1] M.S. Ahmed, Designing of internet of things for real time system, Materials Today: Proceedings, https://doi.org/10.1016/j.matpr.2021.03.527 (2021).

[2] Saif Allah H. AlMetwally, Procedia CIRP 91 (2020). 478 – 485.

[3] Behzada , et al Procedia Computer Science 34 (2014). 220 – 227.

[4] José Fernando Mendoza, et al Procedia Computer Science 109C (2014). 1092 – 1097.

[5] Mirza Jabbar Aziz Baig, Design and implementation of an open-Source IoT and blockchain-based peer-to-peer energy trading platform using ESP32-S2, Node-Red and, MQTT protocol, Energy Reports 7 (2021) 5733-5746.

[6] Ambade Shruti Dinkar and S.A Shaikh University Of Pune,India SSN 2224-896X (online) Vol 1, No.3, 2011

[7] Carlos A. Hernández-Morales, Design and deployment of a practical IoT-based monitoring system for protected cultivations, Computer Communications 186 (2022) 51–64.

[8] Nawzad K. Al-Salihi, Improvement of the Fault Tolerance in IoT Based Positioning Systems by Applying for Redundancy in the Controller Layer, Baghdad Science Journal 2021, 18(4): 1303-1316.

[9] Victor Chang, An industrial IoT sensor system for high-temperature measurement, Computers and Electrical Engineering 95 (2021) 107439.

[10] Amit Kumer Podder, IoT based smart agrotech system for verification of Urban farming parameters, Microprocessors and Microsystems 82 (2021) 104025.

ภาคผนวก ก

โค้ดโปรแกรมสำหรับไมโครคอนโทรลเลอร์

ก.1 โค้ดสำหรับการอ่านข้อมูลจากเซนเซอร์ต่าง ๆ บน Arduino Uno

```
#include <SoftwareSerial.h>
#include <DS18B20.h>
#include <OneWire.h>
#include <DallasTemperature.h>
#include <Wire.h>
#include <RTClib.h>
#include <TinyGPS++.h>
#include <avr/wdt.h>
static const uint32_t GPSBaud = 9600;
// The serial connection to the GPS device
SoftwareSerial GpsSerial(RXPin, TXPin);
static const int RXPin = 2, TXPin = 3;
int door1, door2, door3 ;
const int Sensor_Door1 = 7;
const int Sensor_Door2 = 8;
const int Sensor_Door3 = 9;
long double GPS_La;
long double GPS_Lo;
long double coor_distance ;
float Speed ;
int GPS_Status ;
char DATE[40] ;
char TIME[40];

SoftwareSerial linkSerial(4, 6);
RTC_DS1307 RTC;

#define ONE_WIRE_BUS 5
OneWire oneWire(ONE_WIRE_BUS);
```

```
DallasTemperature sensors(&oneWire);
float tempC, Temp1, Temp2, Temp3, T_AVG;

int buzzer = 13;

void setup() {
  wdt_enable(WDTO_8S);
  Serial.println("Arduino Reset");
  linkSerial.begin(4800);
  Serial.begin(9600);
  GpsSerial.begin(GPSBaud);
  pinMode(buzzer, OUTPUT);
  setupDoor();
  setupTime();
  delay(1000);
}
void loop() {
  wdt_disable();
  digitalWrite(buzzer, HIGH);
  getTime();
  getDoor();
  getGPS();
  //delay(1000);
  wdt_reset(); //Reset
}

void setupDoor() {
  pinMode(Sensor_Door1, INPUT_PULLUP);
  pinMode(Sensor_Door2, INPUT_PULLUP);
  pinMode(Sensor_Door3, INPUT_PULLUP);
}
```

```
void setupTime() {
  Wire.begin(); // Start the I2C
  RTC.begin();  // Init RTC
}

long startTime = 0;
long timeout = 5000;
long increment = 10;

void getGPS() {
  if (GpsSerial.available() > 0) {
    while (GpsSerial.available() > 0) {
      startTime += increment;
      gps.encode(GpsSerial.read());
      if (gps.location.isUpdated()) {
        startTime = 0;
        GPS_La = gps.location.lat();
        GPS_Lo = gps.location.lng();
        Speed = gps.speed.kmph();
        getTemperature();
        GPS_Status = 1;
        String payload = "[" + String(DATE) + "," + String(TIME)  + "," + String(Temp1) +
"," + String(Temp2) + "," + String(Temp3) + "," + String(T_AVG) + "," + String(door1) +
"," + String(door2) + "," + String(door3)  + "," + String((double)GPS_La, 6) + "," +
String((double)GPS_Lo, 6) + "," + String(Speed) + "," + String(GPS_Status) + "]";
        Serial.println(payload);
        linkSerial.print(payload);
        delay(500);
        alarm();
      } else if (startTime >= timeout) {
        Serial.println(String(startTime) + " > " + String(timeout));
```

```
        if (startTime != timeout + increment) {
          getTemperature();
          GPS_La = 0;
          GPS_Lo = 0;
          Speed = 0;
          GPS_Status = 0;
          String payload = "[" + String(DATE) + "," + String(TIME)  + "," + String(Temp1)
+ "," + String(Temp2) + "," + String(Temp3) + "," + String(T_AVG) + "," + String(door1)
+ "," + String(door2) + "," + String(door3)  + "," + String((double)GPS_La, 6) + "," +
String((double)GPS_Lo, 6) + "," + String(Speed) + "," + String(GPS_Status) + "]";
          Serial.println(payload);
          linkSerial.print(payload);
          delay(500);
          alarm();
        }
        startTime = 0;
      }
    }
  }
}

void getDoor() {
  door1 = digitalRead(Sensor_Door1) ;
  door2 = digitalRead(Sensor_Door2) ;
  door3 = digitalRead(Sensor_Door3) ;

}

void getTemperature() {
  sensors.requestTemperatures();
  // Addresses of 3 DS18B20s
```

```
byte Thermo1[8] = { 0x28, 0x09, 0x4F, 0xDF, 0x0D, 0x00, 0x00, 0x85 };
byte Thermo2[8] = { 0x28, 0xB6, 0xDC, 0xB4, 0x0D, 0x00, 0x00, 0x5B };
byte Thermo3[8] = { 0x28, 0xB1, 0x4E, 0xDF, 0x0D, 0x00, 0x00, 0xEE };

float T1 = sensors.getTempC(Thermo1);
float T2 = sensors.getTempC(Thermo2);
float T3 = sensors.getTempC(Thermo3);

if ((T1 != T2) && (T1 != T3)&&(T2 != T3)) {

  Temp1 = T1;
  Temp2 = T2;
  Temp3 = T3;



  T_AVG ;
  if (Temp1 < -20 && Temp2 < -20 && Temp3 < -20) {
    T_AVG = 0 ;
  }
  else if (Temp1 < -20 && Temp2 < -20 ) {
    T_AVG = Temp3 ;
  }
  else if (Temp1 < -20 && Temp3 < -20 ) {
    T_AVG = Temp2 ;
  }
  else if (Temp2 < -20 && Temp3 < -20 ) {
    T_AVG = Temp1 ;
  }
  else if (Temp1 < -20 ) {
    T_AVG = (Temp2 + Temp3) / 2 ;
  }
```

```
    else if (Temp2 < -20 ) {
      T_AVG = (Temp1 + Temp3) / 2 ;
    }
    else if (Temp3 < -20 ) {
      T_AVG = (Temp1 + Temp2) / 2 ;
    }
    else {
      T_AVG = (Temp1 + Temp2 + Temp3) / 3 ;
    }
  }
}

void getTime () {
  DateTime now = RTC.now();
  int T_year = now.year();
  byte T_month = now.month();
  byte T_day = now.day();
  byte T_hour = now.hour();
  byte T_minute = now.minute();
  byte T_second = now.second();
  TIME[40]; sprintf(TIME, "%02d:%02d:%02d", T_hour, T_minute, T_second);
  DATE[40]; sprintf(DATE, "%02d/%02d/%04d", T_day, T_month, T_year);
}

void alarm() {
  if (door1 | door2 | door3) {
    digitalWrite(buzzer, LOW);
    tone(buzzer, 500, 500);
    delay(1000);
  }
}
```

ก.2 โค้ดสำหรับการส่งข้อมูลไปยังเซิร์ฟเวอร์ผ่าน NodeMCU 32

```
#include <ArduinoJson.h>
#include <SoftwareSerial.h>
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include <Arduino_JSON.h>
#include <PubSubClient.h>
#include <SPI.h>
#include <SD.h>
#define MQTT_SERVER   "45.91.134.248"
#define MQTT_PORT     8841
#define MQTT_USERNAME "krao"
#define MQTT_PASSWORD "mqtt-krao"
#define MQTT_NAME     "KIT-001P"
int day_num = 20 ;
SoftwareSerial linkSerial(D4, D3); // TX, RX เลขน้อยนำ A1 to ESP
SoftwareSerial MegaSerial(D1, D2); // TX, RX เลขน้อยนำ ESP to Mega

File myFile;
String values ;
String values2 ;
String values_SD ;
int recNum = 0;
int A;
int B;

unsigned long previousTime = millis();

const int chipSelect = D8;
```

```
WiFiClient wifiClient;
PubSubClient mqtt(wifiClient);

int wifi_status =WiFi.status();
String filename = "logs14.csv";
String str ;
void setup() {
  pinMode(SS, OUTPUT);
  linkSerial.begin(4800);
  MegaSerial.begin(4800);
  Serial.begin(9600);
  Serial.print("Initializing SD card...");
  if (!SD.begin(chipSelect)) {
    Serial.println("initialization failed!");
    while (1);
  }
  Serial.println("initialization done.");
  if (!SD.exists(filename)) {
    myFile = SD.open(filename, FILE_WRITE);
    if (myFile) {
      myFile.close();
    } else {
      Serial.println("Error to open " + filename + "...");
    }
  } else {
    Serial.println(filename + " is already exist.");
  }
  delay(500);
```

```
  setupWifi();
  mqtt.setServer(MQTT_SERVER, MQTT_PORT);
}

void loop() {
  wifi_status = WiFi.status();

  //Receive Data by Arduino Uno
  if (linkSerial.available()) {
    while (linkSerial.available() > 0) {
      values = linkSerial.readString();
      values_SD = values + "r" ;
      stringHandle(values);
      Backup();
      Serial.println(wifi_status);
    }
  }
  else if (linkSerial.available() == 0) {
    //Serial.println("No Data");
  }
}

void stringHandle(String val) {
  //Serial.println(val.length());
  String values = val.substring(1, val.length());
  String date = getValue(values, ',', 0);
  String time = getValue(values, ',', 1);
  float t1 = getValue(values, ',', 2).toFloat();
  float t2 = getValue(values, ',', 3).toFloat();
  float t3 = getValue(values, ',', 4).toFloat();
  float t_avg = getValue(values, ',', 5).toFloat();
```

```
  int d1 = getValue(values, ',', 6).toInt();
  int d2 = getValue(values, ',', 7).toInt();
  int d3 = getValue(values, ',', 8).toInt();
  float speed = getValue(values, ',', 11).toFloat();
  int gps_status = getValue(values, ',', 12).toInt();
  wifi_status = WiFi.status();
  values2 = ("[" + date + "," + time + ","  + String(t1) + ","  + String(t2) + ","  +
String(t3) + "," + String(t_avg) + "," + String(d1) + "," + String(d2) + "," + String(d3) + ","
+ String(speed) + "," + String(gps_status) + "," + String(wifi_status) + "]");
  MegaSerial.print(values2);
}

String getValue(String data, char separator, int index)
{
  int found = 0;
  int strIndex[] = {0, -1};
  int maxIndex = data.length() - 1;

  for (int i = 0; i <= maxIndex && found <= index; i++) {
    if (data.charAt(i) == separator || i == maxIndex) {
      found++;
      strIndex[0] = strIndex[1] + 1;
      strIndex[1] = (i == maxIndex) ? i + 1 : i;
    }
  }

  return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
}

void Backup() {
unsigned long currentTime = millis(); // or millis()
```

```
if (((wifi_status == 0)||(wifi_status == 1))&&(currentTime - previousTime > 30000) ) {

  myFile = SD.open("backupB.txt", FILE_WRITE);
  if (myFile) {
    Serial.print("backupB_Wifi off : ");
    Serial.println(values_SD);
    myFile.println(values_SD);
    mqttPayload("ice_truck/values", values);
    myFile.close();
    A = 1;
    B = 1;
    previousTime = currentTime;
  }
  if (A == 1) {
    myFile = SD.open("backupA.txt", FILE_WRITE);
    if (myFile) {
      myFile.println(values);
      Serial.print("backupA_Wifi off : ");
      Serial.println(values);
      myFile.close();
      A = 0;

    }
  }
}

else if ((wifi_status != 0)&&(wifi_status != 1)&& (B == 0)) {
  Serial.print("Send data to MQTT: ");
  mqttPayload("ice_truck/values", values);
  myFile = SD.open("backupA.txt", FILE_WRITE);
  if (myFile) {
```

```
    myFile.println(values);
    Serial.print("backupA_Wifi on :");
    Serial.println(values);
    myFile.close();
  }

}
else if ((wifi_status != 0)&&(wifi_status != 1) &&(B == 1)) {
  //Read data SD crad line by line and Send data to MQTT
  Serial.println("if ((Status_WIFI != 0) && (B == 1))");
  myFile = SD.open("backupB.txt");
  if (myFile) {
    Serial.println("backupB.txt");
    while (myFile.available()) {
      String list = myFile.readStringUntil('r');
      Serial.print("Send data to MQTT: ");
      Serial.println(list);
      mqttPayload("ice_truck/rd_sdcard", list);
      delay(500);
      recNum++; // Count the record
      Serial.println(recNum);
    }
    myFile.close();
  } else {
    Serial.println("error opening test.txt");
  }
  Serial.println("remove backupB.txt");
  SD.remove("backupB.txt");
  recNum = 0 ;
  B = 0;
  delay(500);
```

```
  }}
void setupWifi() {
  const char* ssid = "4GMIFI_6803";
  const char* pass = "1234567890";
  WiFi.begin(ssid, pass); //Connect SSID and Pass
  Serial.print("Wi-Fi connected.");
  Serial.print("IP Address : ");
  Serial.println(WiFi.localIP());  //  Print IP
}
void mqttPayload(char* topic, String payload) {
  if (WiFi.status() == WL_CONNECTED) {
    if (mqtt.connect(MQTT_NAME, MQTT_USERNAME, MQTT_PASSWORD)) {
      Serial.print("\n Publish message: ");
      if (mqtt.publish(topic, payload.c_str()) == true) {
        Serial.println("Success sending");
      } else {
        Serial.println("Fail sending");
      }} }}
```

ก.3 โค้ดสำหรับการจอแสดงสถานะทำงานอุปกรณ์ บน Arduino Mega 2560

```
#include <MCUFRIEND_kbv.h>   // Hardware-specific library
MCUFRIEND_kbv tft;

#include <Arduino.h>
#include <Adafruit_GFX.h> // Core graphics library
#include <SPI.h>          // f.k. for Arduino-1.5.2
#include <SD.h>
#include <TimeLib.h>

File myFile;
File datetimeVar; // instance of a file

const int SD_CS = 53;

String values;
String values2;
int wifi_status ;
String SD_File ;
#include <ArduinoJson.h>
#include <SoftwareSerial.h>
SoftwareSerial ESPSerial(10, 11); // TX, RX  Arduino send data to NodMCU32
//#include <avr/pgmspace.h>
#define BLACK   0x0000
#define BLUE    0x001F
#define RED     0xF800
#define GREEN   0x07E0
#define CYAN    0x07FF
#define MAGENTA 0xF81F
#define YELLOW  0xFFE0
```

```
#define WHITE    0xFFFF

#define BG1 0xC6D7
#define CAR_BG 0x877d

File root;
int pathlen;
uint8_t spi_save;
int bg1[] = {198, 215, 230};
int bg2[] = {91, 155, 213};
int car_bg[] = {135, 125, 114};

byte isFinish = 0;
String testDate;

String started_datetime = "";
tmElements_t startedDate;

unsigned long previousTime = millis();
long timeInterval = 5000;
void setup()

{
  uint16_t ID;
  ESPSerial.begin(4800);
  //MegaSerial.begin(4800);
  Serial.begin(9600);
  pinMode(SS, OUTPUT);
  if (!SD.begin(SD_CS)) {
    return;
  }
```

```
  ID = tft.readID();
  Serial.println(ID, HEX);
  if (ID == 0x0D3D3) ID = 0x9481;
  tft.begin(ID);
  tft.fillScreen(0x0000);
  tft.setRotation(-1);

  bool good = SD.begin(SD_CS);
  if (!good) {
    Serial.print(F("cannot start SD"));
    isFinish = 0;
    while (1);
    Serial.print("Initializing SD card...");
  }
  bmpDraw("main1.bmp", 0, 0, true, NULL, false);
  isFinish = 1;
  bmpDraw("temp.bmp", 5, 5, true, bg1, false);
  bmpDraw("car.bmp", 65, 260, false, bg2, false);
  showmsgXY(325, 122, 2, ("T1"));
  showmsgXY(280, 122, 2, ("T2"));
  showmsgXY(240, 122, 2, ("T3"));
  showmsgXY(360, 122, 2, ("D1"));
  showmsgXY(360, 167, 2, ("D2"));
  showmsgXY(280, 167, 2, ("D3"));
  showmsgXY(220, 290, 2.5, ("Km/hr"));
  showmsgXY(250, 50, 2.5, ("Date :"));
  showmsgXY(250, 20, 2.5, ("Time :"));
  showmsgXY(160, 30, 2, ("C"));
}
unsigned long previousTime_Uno;
```

```
void loop() {
  unsigned long currentTime = millis(); // or millis()
  while (ESPSerial.available() > 0) {
    values = ESPSerial.readString();
    stringHandle(values);
   delay(500);
  }
}



void writeDateTime (String datetime) {
  File file = SD.open("datetime.var", FILE_WRITE);
  Serial.print("Writing to datetime.var... " + datetime);
  file.println(datetime);
  file.close();
  Serial.println(" done.");
}

String readFileTxt (String filename) {
  File file = SD.open(filename);
  String buffer;
  while (file.available()) {
    buffer = file.readStringUntil('\n');
  }
  file.close();
  return buffer;
}

void stringHandle(String val) {
  Serial.println(val.length());
```

```
  if (val.length() <= 70 & val[0] == '[' & (val[val.length() - 4] == ']' || val[val.length()] == ']' || val[val.length() - 1] == ']' || val[val.length() - 2] == ']' || val[val.length() - 3] == ']')) {
    String values = val.substring(1, val.length());
    String date = getValue(values, ',', 0);
    String time = getValue(values, ',', 1);
    float t1 = getValue(values, ',', 2).toFloat();
    float t2 = getValue(values, ',', 3).toFloat();
    float t3 = getValue(values, ',', 4).toFloat();
    float t_avg = getValue(values, ',', 5).toFloat();
    int d1 = getValue(values, ',', 7).toInt();
    int d2 = getValue(values, ',', 6).toInt();
    int d3 = getValue(values, ',', 8).toInt();
    float speed = getValue(values, ',', 9).toFloat();
    int gps_status = getValue(values, ',', 10).toInt();
    wifi_status = getValue(values, ',', 11).toInt();
    setData(date, time, t1, t2, t3, t_avg, d1, d2, d3, speed, gps_status, wifi_status);
  }
}

void setData(String date, String time, float t1, float t2, float t3, float t_avg, int d1, int d2, int d3, float speed, int gps_status, int wifi_status) {
  //Temp Screen
  tft.fillRect(80, 25, 80, 25, 0xC6D7);
  showmsgXY(90, 30, 2, String(t_avg));

  //Speed Screen
  tft.fillRect(140, 287, 150, 20, 0x5b9b);
  showmsgXY(145, 290, 2.5, String(speed) + " Km/hr");

  // Door
```

```
  Door(d1, 360, 140);
  Door(d2, 360, 95);
  Door(d3, 280, 140);

  // Temp
  Temp(t1, 320, 95);
  Temp(t2, 280, 95);
  Temp(t3, 240, 95);

  // Wifi
  setWifi(wifi_status);

  // Date & Time
  if(date.length() == 10){
  tft.fillRect(340, 50, 130, 20, 0xC6D7);
  showmsgXY(350, 50, 2.5, date);
  }
  tft.fillRect(340, 20, 130, 20, 0xC6D7);
  showmsgXY(350, 20, 2.5, time);

  // GPS
  Map(gps_status);

  // SD Card
  //SDCard();
}

void showmsgXY(int x, int y, int sz, String msg) {
  int16_t x1, y1;
  uint16_t wid, ht;
  tft.setCursor(x, y);
```

```
  tft.setTextSize(sz);
  tft.print(msg);
}

void setWifi(int wifi_status) {
  if ((wifi_status == 0)||(wifi_status == 1)) {
    bmpDraw("wifi_off.bmp", 180, 5, false, bg1, false); // 70
  } else {
    bmpDraw("wifi_on.bmp", 180, 5, false, bg1, false); // 70
  }
}

void SDCard() {
  //int status_SD_Card = (SD.begin(SD_CS));
  if ((myFile == 1) && (wifi_status != 3) ) {
    bmpDraw("SDWrite.bmp", 5, 80, false, bg1, false); // 70
  } else {
    bmpDraw("SDon3.bmp", 5, 80, false, bg1, false); // 70
  }
}

void Door(int status, int x, int y) {
  if (status == 1) {
    bmpDraw("Dooroff.bmp", x, y, false, car_bg, false); // D1
  } else {
    bmpDraw("Dooron.bmp", x, y, false, car_bg, false); // D1
  }
}

void Temp(float temp, int x, int y) {
  if (temp == -127) {
```

```
    bmpDraw("Tempoff.bmp", x, y, false, car_bg, false); // D1
  } else {
    bmpDraw("TempOn.bmp", x, y, false, car_bg, false); // D1
  }
}



void Map(int GPS_Status) {

  if (GPS_Status == 0) {
    bmpDraw("Mapoff4.bmp", 350, 250, false, bg2, false); // 75x75
  } else {
    bmpDraw("Mapon3.bmp", 350, 250, false, bg2, false); // 75x75
  }
}



//// Draw BMP Image
#define BUFFPIXEL 20

void bmpDraw(char filename[], int x, int y, bool isBlack, int bg[], bool clean) {
  int bg_size = sizeof(bg) / sizeof(int);
  File   bmpFile;
  int    bmpWidth, bmpHeight;   // W+H in pixels
  uint8_t  bmpDepth;         // Bit depth (currently must be 24)
  uint32_t bmpImageoffset;       // Start of image data in file
  uint32_t rowSize;          // Not always = bmpWidth; may have padding
  uint8_t  sdbuffer[3 * BUFFPIXEL]; // pixel in buffer (R+G+B per pixel)
  uint16_t lcdbuffer[BUFFPIXEL];  // pixel out buffer (16-bit per pixel)
  uint8_t  buffidx = sizeof(sdbuffer); // Current position in sdbuffer
  boolean  goodBmp = false;      // Set to true on valid header parse
```

```
boolean  flip  = true;      // BMP is stored bottom-to-top
int     w, h, row, col;
uint8_t  r, g, b;
uint32_t pos = 0, startTime = millis();
uint8_t  lcdidx = 0;
boolean  first = true;

if ((x >= tft.width()) || (y >= tft.height())) return;
SPCR = spi_save;
if ((bmpFile = SD.open(filename)) == NULL) {
  Serial.print(F("File not found"));
  return;
}

// Parse BMP header
if (read16(bmpFile) == 0x4D42) { // BMP signature
  read32(bmpFile);
  (void)read32(bmpFile); // Read & ignore creator bytes
  bmpImageoffset = read32(bmpFile); // Start of image data
  read32(bmpFile);
  bmpWidth  = read32(bmpFile);
  bmpHeight = read32(bmpFile);
  if (read16(bmpFile) == 1) { // # planes -- must be '1'
    bmpDepth = read16(bmpFile); // bits per pixel
    if ((bmpDepth == 24) && (read32(bmpFile) == 0)) { // 0 = uncompressed

      goodBmp = true; // Supported BMP format -- proceed!
      // BMP rows are padded (if needed) to 4-byte boundary
      rowSize = (bmpWidth * 3 + 3) & ~3;

      // If bmpHeight is negative, image is in top-down order.
```

```
// This is not canon but has been observed in the wild.
if (bmpHeight < 0) {
  bmpHeight = -bmpHeight;
  flip     = false;
}
// Crop area to be loaded
w = bmpWidth;
h = bmpHeight;
if ((x + w - 1) >= tft.width())  w = tft.width()  - x;
if ((y + h - 1) >= tft.height()) h = tft.height() - y;
// Set TFT address window to clipped image bounds
SPCR = 0;
tft.setAddrWindow(x, y, x + w - 1, y + h - 1);

for (row = 0; row < h; row++) { // For each scanline...
  if (flip) // Bitmap is stored bottom-to-top order (normal BMP)
    pos = bmpImageoffset + (bmpHeight - 1 - row) * rowSize;
  else   // Bitmap is stored top-to-bottom
    pos = bmpImageoffset + row * rowSize;
  SPCR = spi_save;
  if (bmpFile.position() != pos) { // Need seek?
    bmpFile.seek(pos);
    buffidx = sizeof(sdbuffer); // Force buffer reload
  }

  for (col = 0; col < w; col++) { // For each column...
    // Time to read more pixel data?
    if (buffidx >= sizeof(sdbuffer)) { // Indeed
      // Push LCD buffer to the display first
      if (lcdidx > 0) {
        SPCR = 0;
```

```
            tft.pushColors(lcdbuffer, lcdidx, first);
            lcdidx = 0;
            first  = false;
          }
          SPCR = spi_save;
          bmpFile.read(sdbuffer, sizeof(sdbuffer));
          buffidx = 0; // Set index to beginning
        }




        b = sdbuffer[buffidx++];
        g = sdbuffer[buffidx++];
        r = sdbuffer[buffidx++];
        int gray_threshold = 150;
        if (isBlack & r == 0 & g == 0 & b == 0 & bg != NULL) {
          lcdbuffer[lcdidx++] = tft.color565(bg[0], bg[1], bg[2]);
        } else if (!isBlack & r >= gray_threshold & g >= gray_threshold & b >=
gray_threshold & bg != NULL) {
          lcdbuffer[lcdidx++] = tft.color565(bg[0], bg[1], bg[2]);
        } else if (clean) {
          lcdbuffer[lcdidx++] = tft.color565(bg[0], bg[1], bg[2]);
        } else {
          lcdbuffer[lcdidx++] = tft.color565(r, g, b);
        }
      } // end pixel
    } // end scanline
    // Write any remaining data to LCD
    if (lcdidx > 0) {
      SPCR = 0;
      tft.pushColors(lcdbuffer, lcdidx, first);
```

```
        }
      } // end goodBmp
    }
  }
  bmpFile.close();
  if (!goodBmp) Serial.println(F("BMP format not recognized."));
}
uint16_t read16(File f) {
  uint16_t result;
  ((uint8_t *)&result)[0] = f.read(); // LSB
  ((uint8_t *)&result)[1] = f.read(); // MSB
  return result;
}
uint32_t read32(File f) {
  uint32_t result;
  ((uint8_t *)&result)[0] = f.read(); // LSB
  ((uint8_t *)&result)[1] = f.read();
  ((uint8_t *)&result)[2] = f.read();
  ((uint8_t *)&result)[3] = f.read(); // MSB
  return result;
}
String getValue(String data, char separator, int index)
{
  int found = 0;
  int strIndex[] = {0, -1};
  int maxIndex = data.length() - 1;
  for (int i = 0; i <= maxIndex && found <= index; i++) {
    if (data.charAt(i) == separator || i == maxIndex) {
      found++;
      strIndex[0] = strIndex[1] + 1;
      strIndex[1] = (i == maxIndex) ? i + 1 : i;
```

```
    } }
  return found > index ? data.substring(strIndex[0], strIndex[1]) : "";
}
```

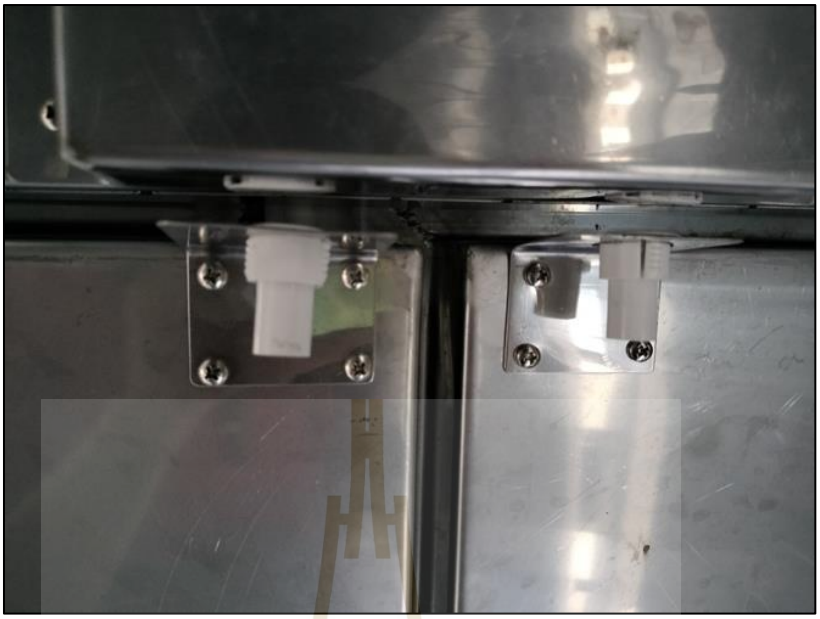
ภาคผนวก ข

การติดตั้งชุดอุปกรณ์ต้นแบบ และการทดสอบภาคสนาม

ข.1 การติดตั้งชุดอุปกรณ์ต้นแบบ

รูปที่ ข.1 แหล่งจ่ายไฟชุดอุปกรณ์ต้นแบบหัวเสียบที่จุดบุหรี่ 12v/24v (10A)

รูปที่ ข.2 ตำแหน่งการติดตั้งกล่องอุปกรณ์ควบคุมใต้ที่นั่ง

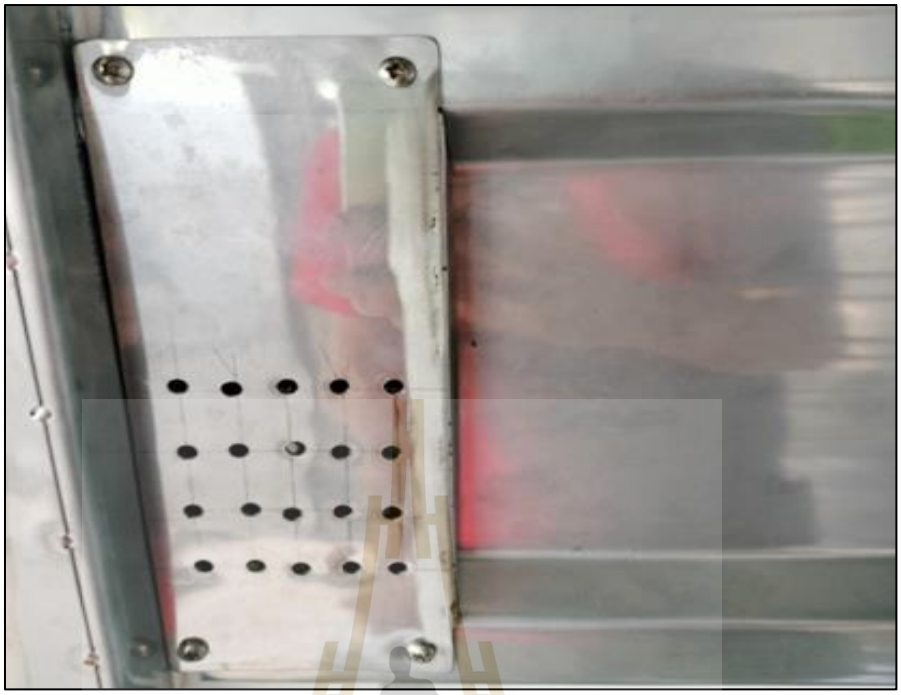


รูปที่ ข.3 ตำแหน่งเซนเซอร์ตรวจจับประตูบานที่ 1 และ2 (ประตูด้านหลัง)

รูปที่ ข.4 ตำแหน่งเซนเซอร์ตรวจจับประตูบานที่ 3 (ประตูด้านข้าง)

รูปที่ ข.5 เสาอากาศสำหรับโมดูล จีพีเอส

รูปที่ ข.6 ภายในชุดอุปกรณ์ติดตั้งเซนเซอร์ตรวจวัดอุณหภูมิ

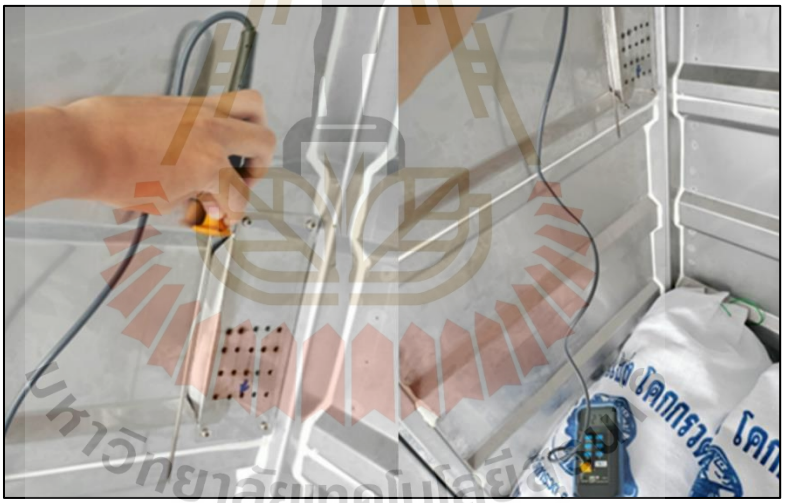


รูปที่ ข.7 ภายนอกชุดอุปกรณ์ติดตั้งเซนเซอร์ตรวจวัดอุณหภูมิ

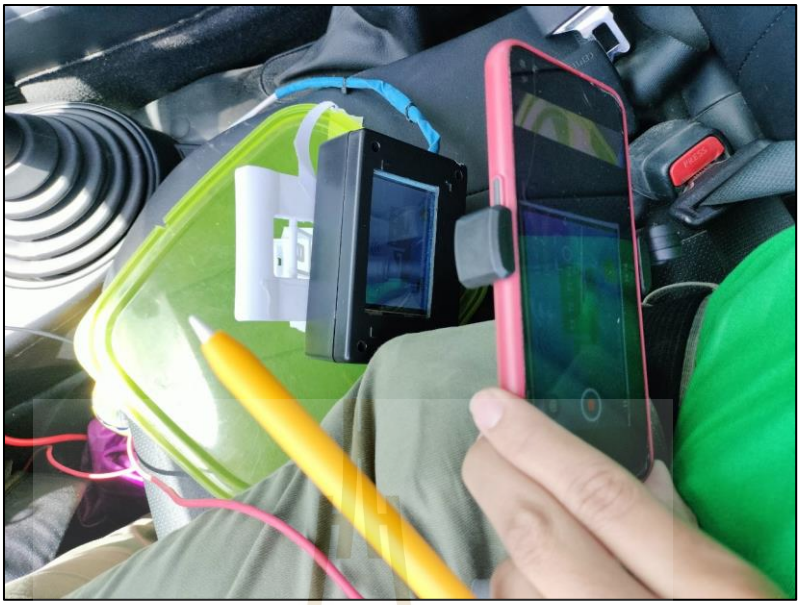
ข.2 การทดสอบภาคสนาม

รูปที่ ข.8 น้ำแข็งที่ใช้ทดสอบภาคสนามจำนวน 30 กระสอบ

รูปที่ ข.9 DE-3003 DIGITAL THERMOMETER TYPE K



รูปที่ ข.10 ตัวอย่างการตรวจวัดอุณหภูมิโดยใช้ DE-3003 DIGITAL THERMOMETER TYPE K



รูปที่ ข.11 การบันทึกวีดีโอหน้าจอระบบแสดงผล

ภาคผนวก ค

บทความที่ได้รับการตีพิมพ์เผยแพร่


ENGCON RMUTP
6th RMUTP
CONFERENCE
ON ENGINEERING
AND TECHNOLOGY
2022
"NEW FRONTIERS
OF ENGINEERING
SCIENCE
TECHNOLOGY
AND INNOVATION"
RMUTP
เกียรติบัตรฉบับนี้มอบไว้เพื่อแสดงว่า
กำพล นนแก้ว พีรพงษ์ พิมพ์อุบ และ จิระพล ศรีเลริฐผล
บทความเรื่อง
ระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง
ได้เข้าร่วมนำเสนอบทความ
สาขาวิชาวิศวกรรมเมคคาทรอนิกส์และระบบการผลิตอัตโนมัติ
การประชุมวิชาการวิศวกรรมศาสตร์และเทคโนโลยี มทร.พระนคร ครั้งที่ 6
(6th RMUTP Conference on Engineering and Technology)
ในวันที่ 27 พฤษภาคม 2565
ณ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร
(รูปแบบออนไลน์)
(ผู้ช่วยศาสตราจารย์ ดร.ณัฐพงศ์ พันธุนะ)
รักษาราชการแทน
คณบดีคณะวิศวกรรมศาสตร์
มหาวิทยาลัยเทคโนโลยีราชมงคลพระนคร
ประธานคณะกรรมการจัดการประชุม
6th RMUTP Conference on Engineering and Technology

บทความวิจัย

การประชุมวิชาการวิศวกรรมศาสตร์และเทคโนโลยี มทร.พระนคร ครั้งที่ 6
Proceedings of the 6th RMUTP Conference on Engineering and Technology

ระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง

Tracking and data logger system for Ice truck.

กำพล นนแก้ว*, พีรพงษ์ พิมพ์ดูบ และ จิระพล ศรีเสริมผล
หลักสูตรวิศวกรรมเมคคาทรอนิกส์ สาขาวิชาวิศวกรรมเครื่องกล สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี
ติดต่อ: aum.kampon@gmail.com

บทคัดย่อ

การวิจัยในครั้งนี้มีวัตถุประสงค์เพื่อ (1) ออกแบบและสร้างระบบติดตามพิกัดตำแหน่งของรถบรรทุกน้ำแข็งและระบบบันทึกข้อมูลการเปิดปิดประตูอุณหภูมิภายในตู้รถบรรทุกน้ำแข็ง (2) ออกแบบและสร้างระบบอินเทอร์เน็ตของสรรพสิ่ง (IoT) ส่งข้อมูลไปยังเซิร์ฟเวอร์ และแสดงผลข้อมูล (3) ออกแบบและสร้างระบบแจ้งเตือนด้วยเสียงเมื่อมีการเปิดประตูและจอแสดงผลสถานะการทำงานอุปกรณ์ ในการพัฒนาระบบแบ่งองค์ประกอบสำคัญอยู่ 3 ส่วน คือ (1) การตรวจวัดค่าเซ็นเซอร์ต่าง ๆ (2) ระบบส่งข้อมูลไปยังเซิร์ฟเวอร์ (3) ระบบจอแสดงผลและการแจ้งเตือน และ (4) ระบบบันทึกข้อมูลสำรอง โดยสถาปัตยกรรมระบบนี้จะสามารถตรวจสอบพิกัดตำแหน่งเส้นทางการเดินรถ และสามารถตรวจสอบจำนวนครั้งในการเปิดปิดประตู ที่เป็นปัจจัยส่งผลต่ออุณหภูมิภายในตู้บรรทุกน้ำแข็ง ซึ่งสามารถนำข้อมูลเหล่านี้ไปประเมินติดตามเส้นทางการเดินรถ และแก้ไขอุณหภูมิได้ทันก่อนส่งผลกระทบต่อน้ำแข็งละลายหรืออาหารแช่แข็งเน่าเสียภายในตู้บรรทุกน้ำแข็งได้

คำหลัก: ระบบติดตามจีพีเอส รถบรรทุกน้ำแข็ง ไมโครคอนโทรลเลอร์ อินเทอร์เน็ตของสรรพสิ่ง

Abstract

The objectives of this research were (1) to design and build a location tracking system for the ice truck and the temperature data logger inside of the truck (2) to design and build an Internet of Things (IoT) system to collect and send the data to a server, and (3) to design and build a sound alarm system when detecting the door is opened and display the device operation status. There are 3 parts of system development (1) Sensor measurement (2) Transmitting the data to the server (3) the display system and (4) the data logger system. This system architecture can trackback the location coordinates of the route and can check the number of times opening the door. The duration of time in opening and closing the door which is a factor affecting the temperature inside the ice truck. These data can be used to assess, track routes, and correct temperatures before they adversely affect the melting ice or spoiled frozen food inside the ice truck.

Keywords: GPS tracking, Ice truck, Microcontroller, Internet of Things

1. บทนำ

ปัจจุบันจากการสำรวจรถบรรทุกน้ำแข็งพบว่าพนักงานขับรถตู้บรรทุกน้ำแข็ง มีพฤติกรรมการขับขี่ที่ไม่เหมาะสม เช่น การขับรถตู้บรรทุกน้ำแข็งออกนอกเส้นทางการไปรับจ้างขนน้ำแข็งจากคู่แข่งโดยไม่ได้รับอนุญาต รวมถึงการบรรทุกขนสินค้าประเภทอาหารแช่แข็งในบางครั้งอาหารแช่แข็งได้รับความเสียหาย ซึ่งมีสาเหตุสำคัญหลักมาจากการสูญเสียอุณหภูมิภายในตู้บรรทุกน้ำแข็ง ส่งผลกระทบให้คุณภาพอาหารแช่แข็งลดลง เกิดการเน่าเสีย ซึ่งทำให้เสียความเชื่อมั่นจากลูกค้า ดังนั้นงานวิจัยนี้ได้นำเสนอออกแบบและพัฒนาระบบติดตามพิกัดตำแหน่ง[3] การเดินรถที่สามารถรับส่งข้อมูลการเปิดปิดประตูและอุณหภูมิ [2] [6] ภายในตู้รถบรรทุกน้ำแข็งด้วยระบบ IoT

2. วิธีการศึกษา

ผลการศึกษาปัจจัยที่ส่งผลต่อปัญหาที่พบ สามารถสรุปได้ว่ามีปัจจัยหลักอยู่ 2 ประการ คือ การสูญเสียอุณหภูมิภายในตู้น้ำแข็ง และพฤติกรรมการขับขี่ที่ไม่เหมาะสม ดังนั้นในกระบวนการศึกษาจึงเริ่มต้นจากการออกแบบระบบในภาพรวม การพิจารณาเลือกใช้อุปกรณ์เซ็นเซอร์ต่างๆ ให้มีความเหมาะสมกับสภาพแวดล้อมภายในตู้บรรทุกน้ำแข็ง การออกแบบ ระบบตรวจวัดค่าเซ็นเซอร์ต่าง ๆ ระบบส่งข้อมูลไปยัง Server และระบบจอแสดงผลข้อมูลตามลำดับ

3. การออกแบบระบบ

การออกแบบระบบได้พิจารณาถึงการทำงานของการออกแบบโครงสร้างสถาปัตยกรรม ดังแสดงในรูปที่ 1 ที่สามารถเชื่อมต่ออุปกรณ์ตรวจวัดค่าเซนเซอร์และโมดูลต่าง ๆ ด้วย ไมโครคอนโทรลเลอร์ [4] Arduino Uno R3 ในการตรวจเช็คการเปิดปิดประตู กรณีประตูไม่ได้ปิดอยู่โมดูล Buzzer จะทำงานส่งเสียงเตือนจนกว่าประตูจะปิดและ ดำเนินการส่งค่าข้อมูลค่าเซนเซอร์และโมดูลต่าง ๆ ไปยัง NodeMCU V2 บันทึกข้อมูลลงโมดูล SD Card และส่งค่าข้อมูลไปยังเซิร์ฟเวอร์ ด้วยโปโตคอล Message Queuing Telemetry Transport (MQTT) [1],[5] ผ่านอินเทอร์เน็ต

มาตรฐานการสื่อสารแบบไร้สาย IEEE 802.11 จาก Pocket Wi-Fi และทำการส่งค่าข้อมูลไปยัง Arduino Mega 2560 เพื่อใช้ในการแสดงผลหน้าจอสถานะทำงานของเซนเซอร์และโมดูลต่าง ๆ ผ่าน โมดูล TFT LCD

รูปที่ 1 สถาปัตยกรรมระบบ

3.1 ระบบตรวจวัดค่าเซ็นเซอร์ต่าง ๆ

การพิจารณาเลือกอุปกรณ์ในระบบ มีข้อพิจารณาจากลักษณะการใช้งานที่เหมาะสมต่อสภาพแวดล้อมใช้งานจริง และมีความเที่ยงตรงของข้อมูลในการตรวจวัด โดยอุปกรณ์ประมวลผลสำหรับรับค่าเซ็นเซอร์และโมดูลต่าง ๆ ในงานวิจัยนี้เลือกใช้ Arduino Uno R3 ซึ่งเป็น Arduino Microcontroller ที่มีราคาของบอร์ดที่ไม่แพงมี Library ต่าง ๆ ที่ใช้งานถูกพัฒนาอ้างอิงกับบอร์ดนี้เป็นหลัก และได้กำหนดตัวตรวจวัดระบบดังนี้

1. ตัวตรวจวัดพิกัดตำแหน่งทางภูมิศาสตร์ และความเร็ว เลือกใช้โมดูล GPS Ublox NEO-7M เป็นอุปกรณ์รับสัญญาณ GPS ที่สามารถตรวจจับสัญญาณดาวเทียมได้สูงสุด 56 ดวง มีค่าความแม่นยำของตำแหน่ง 2.5 เมตร ความแม่นยำของความเร็ว 0.1 เมตรต่อวินาที และความถี่ของสัญญาณอยู่ในช่วงระหว่าง 0.25 Hz ถึง 10 MHz
2. ตัวตรวจสอบวันที่และเวลา เลือกใช้โมดูล DS3231 Real Time Clock ที่มีค่าความคลาดเคลื่อนไม่เกิน 2PPM (2 วินาที/1,000,000 วินาที)
3. ตัวตรวจสอบการเปิดปิดประตู เลือกใช้เซ็นเซอร์ Magnetic Reed Switch BR-1021 มีค่าระยะตรวจจับแม่เหล็กอย่าง 0 ถึง 21 มม.
4. ตัวตรวจวัดอุณหภูมิ เลือกใช้ดิจิตอลเซ็นเซอร์ DS18B20 ที่มีค่าความผิดพลาด ±0.5 °C ค่าความถูกต้องระหว่าง -10 °C ถึง +85 °C

3.2 ระบบส่งข้อมูลไปยัง Server

การส่งข้อมูลไปยังเซิร์ฟเวอร์ งานวิจัยนี้เลือกใช้อุปกรณ์ประมวลผล NodeMCU V2 ซึ่งเป็น Microcontroller ที่มี โมดูล WI-FI ESP8266 ทำให้สามารถรับค่าข้อมูล ส่งไปยัง Server ด้วยโปโตคอล Message Queuing Telemetry Transport (MQTT) ที่มีความเหมาะสมสำหรับแอปพลิเคชัน IOT ที่มีทรัพยากรจำกัดผ่านอินเทอร์เน็ตมาตรฐานการสื่อสารแบบไร้สาย IEEE 802.11 จาก Pocket Wi-Fi ได้ และใช้ Web Application ในการแสดงผลข้อมูลแบบเรียลไทม์ ดังแสดงในรูปที่ 2

รูปที่ 2 Web Application

3.3 ระบบจอแสดงผลและการแจ้งเตือน

ระบบแสดงผล เป็นการทำงานร่วมกันระหว่าง อุปกรณ์ประมวลผล Microcontroller Arduino Mega 2560 ร่วมกับโมดูล TFT LCD3.5" ในการแสดงสถานการณ์ทำงานของอุปกรณ์ดังแสดงในรูปที่ 3 ระบบการแจ้งเตือน ได้ทำการพิจารณาเลือกใช้อุปกรณ์ส่งสัญญาณเสียง โมดูล Active Buzzer โดยจะส่งเสียงเตือนเมื่อมีการเปิดประตูตู้บรรทุกน้ำแข็งทิ้งไว้ และจะหยุดส่งสัญญาณเสียงเมื่อประตูปิด

รูปที่ 3 แสดงจอแสดงผลสถานะการทำงานของอุปกรณ์

3.4 ระบบบันทึกข้อมูลสำรอง

ระบบบันทึกข้อมูลสำรองมีความสำคัญเพื่อใช้ในการตรวจสอบข้อมูลย้อนหลังกรณีส่งข้อมูลไปยังเซิร์ฟเวอร์ไม่สำเร็จ โดยพิจารณาใช้ NodeMCU V2 ประมวลผลร่วมกับโมดูล SD Card ในการบันทึกข้อมูลสำรอง

4. ผลการทดสอบ

ระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็ง ที่ได้ออกแบบตามงานวิจัยนี้มีผลการศึกษา และผลการทดลองของอุปกรณ์ต่าง ๆ ในระบบ ดังนี้

4.1 ผลการทดสอบตัวตรวจวัดพิกัดตำแหน่งทางภูมิศาสตร์

การทดสอบโมดูล GPS 7M เปรียบเทียบกับ พิกัดตำแหน่งบน Google map ซึ่งขอบเขตที่ยอมรับได้ในการนำไปใช้งานของความคลาดเคลื่อนของตำแหน่งไม่เกิน 5 เมตร และความคลาดเคลื่อนของความเร็วไม่เกิน 5 กิโลเมตรต่อชั่วโมง การทดสอบได้พิจารณาพิกัด

ตำแหน่งบริเวณ ถนนโรงพยาบาลมหาวิทยาลัยเทคโนโลยีสุรนารี ดังแสดงในรูปที่ 4 ในการทดสอบจะแบ่งออกเป็น 2 การทดสอบ

1.การทดสอบแบบ Static Test

การทดสอบเก็บค่าพิกัดตำแหน่งแบบอยู่กับที่ในแต่ละตำแหน่งทั้งหมด 12 จุด แล้วนำค่าพิกัดตำแหน่งที่เก็บได้ไปทำการเปรียบเทียบกับพิกัดตำแหน่งบน Google map

2.การทดสอบแบบ Dynamic Test

การทดสอบเก็บค่าความเร็ว 20 30 และ 40 กิโลเมตรต่อชั่วโมง ขณะเคลื่อนที่ด้วยความเร็วคงที่ผ่านเส้นทางทั้ง 12 จุด เพื่อเปรียบเทียบความคลาดเคลื่อนความเร็วของ GPS ที่ความเร็วคงที่

รูปที่ 4 แสดงพิกัดตำแหน่งบน Google map ทดสอบ

ผลการทดสอบทั้งสองการทดสอบสามารถสรุปได้ว่าความคลาดเคลื่อนพิกัดตำแหน่งของ GPS ขณะอยู่กับที่สูงสุดมีค่าเท่ากับ 3.25 เมตร ค่าความคลาดเคลื่อนความเร็วของ GPS ที่ความเร็วคงที่ 20 30 และ 40 กิโลเมตรต่อชั่วโมง อยู่ที่ 2.72 3.03 และ 4.92 กิโลเมตรต่อชั่วโมง ตามลำดับ ดังแสดงในตารางที่ 1

ตารางที่ 1 สรุปค่าความผิดพลาดของการทดสอบ GPS

Test	Random Speed (km/h)	Max Position Error (m)	Max Speed Error (km/h)
Static	-	3.25	0.19
Dynamic	20	-	2.72
	30	-	3.03
	40	-	4.92

4.2 ผลการทดสอบตัวตรวจวัดอุณหภูมิ

เซ็นเซอร์ตรวจวัดอุณหภูมิสอบเทียบกับ Water Bath ในช่วงอุณหภูมิ 5°C ถึง 80°C การทดสอบ Static Calibration จะได้สมการเชิงเส้นโดยที่ y คือค่าอุณหภูมิที่ควรจะเป็น X คือค่าที่ได้จากเซ็นเซอร์ตรวจวัดอุณหภูมิ

เซ็นเซอร์ตรวจวัดอุณหภูมิตัวที่ 1 ; $Y1 = 0.9944x1 + 0.5454$ (1)

เซ็นเซอร์ตรวจวัดอุณหภูมิตัวที่ 2 ; $Y2 = 0.9958x2 + 0.5413$ (2)

เซ็นเซอร์ตรวจวัดอุณหภูมิตัวที่ 3 ; $Y3 = 0.9957x3 + 0.5443$ (3)

การทดสอบ Random Test โดยพิจารณาที่ อุณหภูมิ 10°C ซึ่งจะสังเกตเห็นได้ว่าค่าอุณหภูมิทั้งสามเซ็นเซอร์วัดอุณหภูมิต่ำกว่าอุณหภูมิที่กำหนด อยู่ในช่วง 9°C ถึง 10°C

การทดสอบ Sequential Test เป็นวิธีการนำเซ็นเซอร์วัดอุณหภูมิตรวจวัดอุณหภูมิจากสูงไปอุณหภูมิต่ำ การเปรียบเทียบกับ การนำเซ็นเซอร์วัดอุณหภูมิตรวจวัดอุณหภูมิจากต่ำไปอุณหภูมิสูง

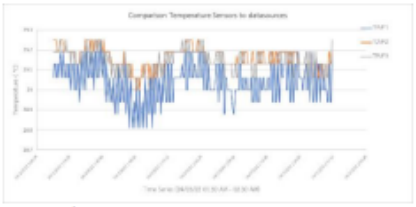
จากการทดสอบทั้งสามการทดสอบสามารถสรุปค่าความผิดพลาดของเซ็นเซอร์ตรวจวัดอุณหภูมิตัวที่ 1 2 และ3 มีค่าดังต่อไปนี้ 1.60°C 1.53°C และ1.56°C ดังแสดงในตารางที่ 2

ตารางที่ 2 สรุปค่าความผิดพลาดการตรวจวัดอุณหภูมิ

Detail Error	Thermometer (°C)	Temperature Sensor (°C)		
		T1	T2	T3
Sensitivity error	0.4	0.75	0.81	0.81
Zero shift error	0.28	0.55	0.54	0.54
Linearity error	0.008	0.006	0.004	0.004
Random error	0.6	0.87	0.87	0.87
Bias error	0.46	0.69	0.70	0.69
Hysteresis error	0.20	0.69	0.38	0.50
Overall error	0.92	1.60	1.53	1.56

4.3 ผลการทดสอบการทำงานของระบบ

รูปที่ 5 แสดงความสัมพันธ์เซ็นเซอร์วัดอุณหภูมิกับช่วงเวลาพฤติกรรมของเซ็นเซอร์ตรวจวัดอุณหภูมิทั้งสามตัวมีพฤติกรรมเป็นไปในทิศทางเดียวกัน รูปที่ 6 แสดงความสัมพันธ์เซ็นเซอร์ประตูกับช่วงเวลา ในช่วงแรกเมื่อไม่มีแม่เหล็กมากระทำกับเซ็นเซอร์ ระบบจะส่งสัญญาณทริกเกอร์เท่ากับ 1 และเมื่อเวลาผ่านไปนำเซ็นเซอร์ตรวจจับเจอแม่เหล็ก ระบบจะส่งสัญญาณทริกเกอร์เท่ากับ 0 จะเห็นได้ว่าเซ็นเซอร์ทั้งสามตัวมีแนวโน้มไปในทิศทางเดียวกัน รูปที่ 7 แสดงความสัมพันธ์พิกัดตำแหน่งกับช่วงเวลา ขณะอยู่กับที่ มีค่าความผิดพลาดตำแหน่งสูงสุดเท่ากับ 4.85 เมตร และรูปที่ 8 แสดงความสัมพันธ์ความเร็วกับช่วงเวลา ขณะอยู่กับที่มีค่าความผิดพลาดความเร็วสูงสุดเท่ากับ 3.52 กิโลเมตรต่อชั่วโมง



รูปที่ 5 ความสัมพันธ์เซ็นเซอร์วัดอุณหภูมิกับช่วงเวลา

รูปที่ 6 ความสัมพันธ์เซ็นเซอร์ประตูกับช่วงเวลา

รูปที่ 7 ความสัมพันธ์พิกัดตำแหน่งกับช่วงเวลา

รูปที่ 8 ความสัมพันธ์ความเร็วกับช่วงเวลา

5. สรุป

งานวิจัยนี้นำเสนอเกี่ยวกับระบบติดตามและเก็บข้อมูลสำหรับรถตู้บรรทุกน้ำแข็งจะได้ระบบที่มีความถูกต้องของการตรวจวัดอุณหภูมิ มีค่าความผิดพลาดไม่เกิน 1.60°C ซึ่งมีความแม่นยำน้อยกว่าสเปคของเซ็นเซอร์อาจเกิดจากปัจจัยต่างๆเช่นความผิดพลาดของอุปกรณ์วัดและความผิดพลาดที่เกิดจากผู้กระทำการทดสอบ การตรวจจับพิกัดตำแหน่ง มีค่าความผิดพลาดไม่เกิน 3.25 เมตร และความเร็วมีค่าความผิดพลาดไม่เกิน 4.95 กิโลเมตรต่อชั่วโมง การส่งข้อมูลระหว่างอุปกรณ์ประมวลผล และผ่านระบบเครือข่ายไร้สายไปยังเซิร์ฟเวอร์ สามารถทำได้อย่างมีประสิทธิภาพ

6. กิตติกรรมประกาศ

งานวิจัยนี้ได้รับการสนับสนุนทุนจาก ห้างหุ้นส่วนจำกัด เศราการช่าง จำกัด และมหาวิทยาลัยเทคโนโลยีสุรนารี

เอกสารอ้างอิง

[1] M.S. Ahmed, Designing of internet of things for real time system, Materials Today: Proceedings, https://doi.org/10.1016/j.matpr.2021.03.527 (2021).

[2] Saif Allah H. AlMetwally, Procedia CIRP 91 (2020). 478 – 485.

[3] Behzada , et al Procedia Computer Science 34 (2014). 220 – 227.

[4] José Fernando Mendoza, et al Procedia Computer Science 109C (2014).

[5] Mirza Jabbar Aziz Baig, Design and implementation of an open-Source IoT and blockchain-based peer-to-peer energy trading platform using ESP32-S2, Node-Red and, MQTT protocol, Energy Reports 7 (2021) 5733-5746.

[6] Victor Chang, An industrial IoT sensor system for high-temperature measurement, Computers and Electrical Engineering 95 (2021) 107439.

ประวัติผู้เขียนบทความ นายกำพล นนแก้ว เป็นนักศึกษาระดับปริญญาโท หลักสูตรวิศวกรรมเมคคาทรอนิกส์ มหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา สำเร็จการศึกษาระดับปริญญาตรีวิศวกรรมศาสตรบัณฑิต (วิศวกรรมการผลิตอัตโนมัติ และหุ่นยนต์) จากมหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา เมื่อ พ.ศ. 2562

ประวัติผู้เขียน

นายกำพล นนแก้ว เกิดเมื่อวันที่ 22 พฤษภาคม พ.ศ.2540 ที่อำเภอพิชัย จังหวัดอุตรดิตถ์ เริ่มการศึกษาชั้นประถมศึกษาที่โรงเรียนบ้านในเมือง อำเภอพิชัย จังหวัดอุตรดิตถ์ ระดับมัธยมศึกษาระดับตอนต้นและตอนปลายที่โรงเรียนพิชัย อำเภอพิชัย จังหวัดอุตรดิตถ์ และสำเร็จการศึกษาวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมการผลิตอัตโนมัติและหุ่นยนต์ สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี จังหวัดนครราชสีมา เมื่อปี พ.ศ. 2563 และเข้าศึกษาต่อในระดับวิศวกรรม ศาสตรมหาบัณฑิต (หลักสูตรวิศวกรรมเมคคาทรอนิกส์) สาขาวิชาวิศวกรรมเมคคาทรอนิกส์ ณ สถาบันการศึกษาเดิม