คู่มือทางด้านเทคนิค
(TECHNICAL MANUAL)

สถานีวัดอากาศขนาดย่อมแบบ Real-Time
Mini Real-Time Weather Station

จัดทำโดยนิสิตสาขาวิชาวิศวกรรมโยธา - ชลประทาน

นางสาวกัญญาพัชร นิลเวช	รหัสนิสิต 6120500898
นางสาวนภัสสร รัตนพันธ์	รหัสนิสิต 6120500987
นายนภัสกร ชุลี	รหัสนิสิต 6120502149

เครื่องมือนี้เป็นผลงานจากวิชาปัญหาพิเศษ 02207498

ภาควิชาวิศวกรรมชลประทาน คณะวิศวกรรมศาสตร์ กำแพงแสน

มหาวิทยาลัยเกษตรศาสตร์ วิทยาเขตกำแพงแสน พ.ศ.2564

คำนำ

คู่มือฉบับนี้เป็นเอกสารประกอบเครื่องมือสถานีวัดอากาศขนาดย่อมแบบ REAL-TIME (MINI REAL-TIME WEATHER STATION) ซึ่งเป็นส่วนหนึ่งของรายวิชาปัญหาพิเศษ 02207498 โดย วัตถุประสงค์ของการจัดทำสถานีวัดอากาศขนาดย่อมแบบ REAL-TIME เนื่องมาจากประเทศไทยเป็นประเทศที่ถือว่าเป็นประเทศแห่งเกษตรกรรม ประชากรกว่าครึ่งหนึ่งมีอาชีพในการทำการเกษตร เพื่อการแข่งขันทางเศรษฐกิจอย่างไม่หยุดยั้งของคนในประเทศนั้น จึงมีการหาแนวทางในการนำตัวช่วยมาเพิ่มความสะดวกในการผลิตผลผลิตทางการเกษตร การใช้เทคโนโลยีเพื่ออำนวยความสะดวกก็เป็นอีกแนวทางหนึ่งของการทำการเกษตร ตามแนวคิดของคนในยุคสมัยใหม่ การพัฒนาระบบในไร่นาที่ต้องมีความง่าย สะดวกและประหยัด ซึ่งในการผลิตพืชผลตามงานด้านชลประทานจะมีปัจจัยที่เกี่ยวเนื่องอยู่หลายปัจจัย การเก็บค่าพารามิเตอร์เพื่อมาออกแบบการให้น้ำ สารอาหารแก่พืช หรือสภาพแวดล้อมที่เหมาะสมแก่การเจริญเติบโตของพืช ผู้ศึกษาจึงมีแนวคิดในการเก็บค่าพารามิเตอร์บางตัวที่จำเป็นในการคำนวณการให้น้ำแก่พืช และสภาพแวดล้อมที่เหมาะสมพืชจะเจริญเติบโตได้อย่างมีประสิทธิภาพ โดยขอบเขตความสามารถของสถานีวัดอากาศขนาดย่อมแบบ REAL-TIME คือสามารถทำการวัดอุณหภูมิ ความชื้นอากาศ ความเร็วลมและตำแหน่งพิกัดของสถานี ณ จุดที่ทำการติดตั้งโดยการใช้อุปกรณ์เซนเซอร์เป็นตัวอ่านค่าพารามิเตอร์และให้ส่งค่าเข้าสู่โปรแกรม VISUAL STUDIO CODE และส่งค่าให้แสดงผลที่สามารถเข้าดูได้ทุกเวลาผ่านแอปพลิเคชัน BLYNK ทางสมาร์ทโฟน และคำนวณออกมาเป็นปริมาณการใช้น้ำของพืชอ้างอิง (REFERENCE CROP EVAPOTRANSPIRATION, ETO) จากนั้นส่งค่าให้ไปเก็บใน GOOGLE SHEETS เพื่อให้สามารถนำค่าที่เก็บมาได้ไปวิเคราะห์ ในการพัฒนาระบบชลประทานไร่นาต่อไป

คำนำ

ข้อดีของสถานีวัดอากาศขนาดย่อมแบบ REAL-TIME (MINI REAL-TIME WEATHER STATION) คือติดตั้งง่าย ใช้งานได้จริงในราคาที่ไม่สูงมาก สามารถทราบค่าที่วัดได้แบบ REAL-TIME ผ่านสมาร์ทโฟน เมื่อทำการติดตั้งและให้เครื่องมือทำการตรวจวัด เก็บค่าพารามิเตอร์ที่จำเป็นต่อการนำมาวิเคราะห์ ซึ่งค่าพารามิเตอร์ที่วัดจะนำมาคำนวณหาค่าความต้องการใช้น้ำของพืชอ้างอิง (REFERENCE CROP EVAPOTRANSPIRATION, ETO) เพื่อนำไปคำนวณต่อถึงการใช้น้ำของพืชแต่ละชนิดได้ โดยส่งค่าเข้าไปเก็บใน GOOGLE SHEETS อย่างเป็นระบบ

ข้อด้อยของสถานีวัดอากาศขนาดย่อมแบบ REAL-TIME (MINI REAL-TIME WEATHER STATION) คือด้วยงบประมาณที่มีอยู่อย่างจำกัดจึงทำให้การเลือกใช้เกรดของฮาร์ดแวร์ที่ไม่ได้มาตรฐานมากนัก อาจทำให้เกิดการชำรุดของอุปกรณ์ได้ง่าย และในส่วนของ GPS MODULE จะสามารถตรวจวัดได้ในเฉพาะวันที่ท้องฟ้ามีความปลอดโปร่งซึ่งเป็นข้อจำกัดของฮาร์ดแวร์ประเภทนี้

ในการพัฒนาสถานีวัดอากาศขนาดย่อมแบบ REAL-TIME (MINI REAL-TIME WEATHER STATION) จะอาศัยองค์ความรู้การเขียดโค้ดด้วยภาษา C++ การเลือกใช้งาน LIBRARIES ในโปรแกรม ARDUINO IDE 1.8.8 การเขียนโค้ดเพื่อส่งค่าแสดงผลผ่านแอปพลิเคชัน BLYNK และส่งค่าไปเก็บใน GOOGLE SHEETS และใช้องค์ความรู้ทางด้านชลประทานในด้านการนำค่ามาคำนวณปริมาณการใช้น้ำของพืชอ้างอิง (REFERENCE CROP EVAPOTRANSPIRATION, ETO)

คณะผู้จัดทำ

กลุ่มที่ 1

กิตติกรรมประกาศ

คู่มือประกอบสถานีวัดอากาศขนาดย่อมแบบ REAL-TIME ฉบับนี้สำเร็จสมบูรณ์ได้ด้วยดีเพราะได้รับความกรุณาชี้แนะและช่วยเหลืออย่างดียิ่งจาก รศ.ดร.วราวุธ วุฒิวณิชย์ ผศ.นิมิตร เฉิดฉันท์พิพัฒน์ และนายไพศาล ขันตีสา ผู้ที่ให้คำแนะนำและตรวจสอบแก้ไขข้อบกพร่อง ตั้งแต่เริ่มต้นจนสำเร็จเรียบร้อย คณะผู้จัดทำขอกราบขอบพระคุณด้วยความเคารพอย่างสูงไว้ ณ โอกาสนี้

ขอขอบคุณนายธนกร ทองดอนใหม่ ช่างเทคนิคปฏิบัติงานของภาควิชาวิศวกรรมชลประทาน ผู้ให้ความช่วยเหลือและสนับสนุนการสร้างสถานีวัดอากาศขนาดย่อมแบบ REAL-TIME ในด้านการเชื่อมฐานและประกอบสถานีจนสำเร็จลุล่วง

ขอขอบคุณนายสมัชฌา จับใจ นิสิตคณะศิลปะศาสตร์และวิทยาศาสตร์ สาขาวิชาวิทยาการคอมพิวเตอร์ ผู้ที่ให้ความช่วยเหลือและคำปรึกษาในด้านการเขียน CODE เพื่อใช้งานเซนเซอร์จนสามารถตรวจวัดค่าพารามิเตอร์ที่ใช้ในสถานีวัดอากาศได้ครบทุกพารามิเตอร์

และขอขอบคุณเจ้าของเอกสารและงานวิจัยของทุกท่าน ผู้ที่คณะผู้จัดทำได้ทำการศึกษาค้นคว้าและนำมาอ้างอิงในการจัดทำผลงานจนประกอบเป็นสถานีวัดอากาศที่สามารถใช้งานได้จริง

คณะผู้จัดทำกลุ่มที่ 1 ขอขอบพระคุณทุกท่านเป็นอย่างสูงที่ให้การสนับสนุน เอื้อเฟื้อและให้ความอนุเคราะห์ความช่วยเหลือจนกระทั่งการประดิษฐ์สถานีวัดอากาศขนาดย่อมแบบ REAL-TIME สำเร็จลุล่วงผ่านไปได้ด้วยดี

คณะผู้จัดทำ

กลุ่มที่ 1

TABLE OF CONTENTS

1. ขั้นตอนการออกแบบและสร้าง
สถานีวัดอากาศขนาดย่อมแบบ REAL-TIME

01 จัดหา/ซื้ออุปกรณ์ ที่ต้องใช้ในการออกแบบ

อุปกรณ์ที่ใช้ในการออกแบบสถานีวัดอากาศขนาดย่อมแบบ Real-Time ทั้งหมด ระบุดังตารางที่ 1 อุปกรณ์ที่ใช้ในการตรวจวัด เก็บข้อมูลและส่งข้อมูล

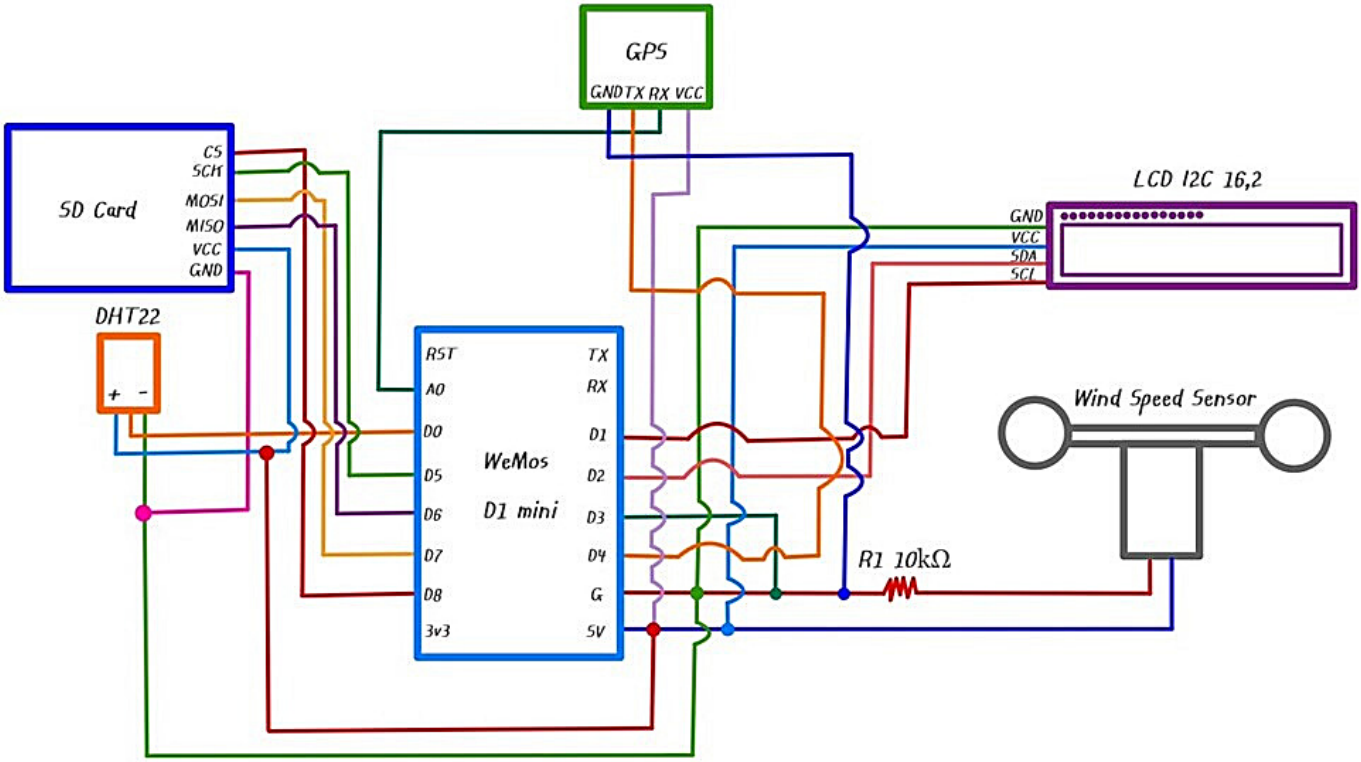
02 ทำการทดสอบ Senser แต่ละตัว

ด้วยการเขียน Code และทำการ Compile เพื่อตรวจสอบว่า Sensor แต่ละชนิดทำงานได้ปกติหรือไม่ และบันทึก Code ไว้ ทั้งนี้ Sensor ที่ต้องทำการตรวจสอบ ได้แก่

- DHT22 Sensor
- SD Card module
- Wind Speed Sensor
- LCD I2C
- GPS Module

ซึ่งสามารถดูตัวอย่าง Code เพื่อทดสอบ Sensor แต่ละชนิดได้ในหัวข้อภาคผนวก ข Code ที่ใช้ในการทดสอบ Sensor

03 ทำการประกอบ Sensor ทั้งหมดเข้าด้วยกัน ดังแสดงในรูปที่ 1 และนำ Code ทั้งหมดมาเขียนรวมเป็นหนึ่งเดียว ซึ่ง Code ทั้งหมดแสดงอยู่ในหัวข้อที่ 3.3 Total Code



รูปที่ 1 ไดอะแกรมแสดงการประกอบ Sensor ทั้งหมดเข้าด้วยกัน

04 ทำการทดสอบ Compile ให้แสดงผลค่าต่างๆ บนหน้าจอ LCD

เมื่อทำการ Compile Code ดังแสดงไว้ในหัวข้อที่ 3.3 Total Code แล้ว จะได้ผลค่าต่างๆ แสดงบนหน้าจอ LCD แสดงดังรูปที่ 2

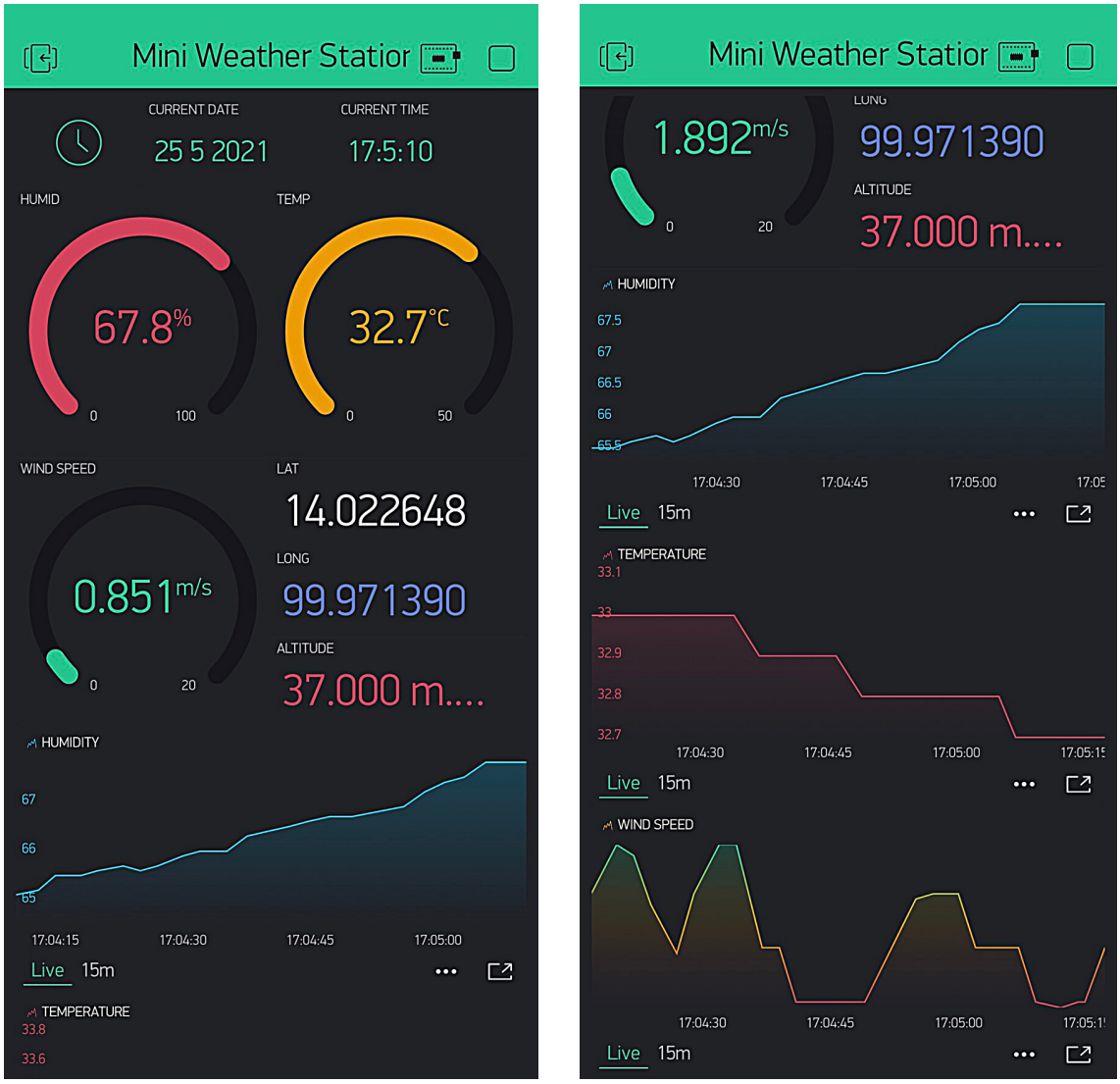
รูป 2 แสดงผลค่าอุณหภูมิ ความชื้นสัมพัทธ์ และความเร็วลมบนหน้าจอ LCD

05 ทำการเชื่อมต่อ Sensor เข้ากับแอพพลิเคชัน Blynk เพื่อส่งค่าไปแสดงผลแบบ Real-Time ผ่าน Smartphone

ขั้นตอนการเชื่อมต่อ Sensor เข้ากับแอพพลิเคชัน Blynk มีดังนี้

1. Download Application: Blynk
2. ทำการ Create New Account และ Login
3. ทำการสร้าง New Project เพื่อแสดงค่าแบบ Real-Time ผ่าน Smartphone
4. ทำการเขียน Code ดังแสดงในหัวข้อที่ 3.3 Total Code ทั้งนี้ Code ในส่วน char auth[] = "Auth Token ที่ส่งไปยัง E-mail ที่ใช้ลงทะเบียนแอพพลิเคชัน Blynk ";

เมื่อทำการ Compile code ดังกล่าว จะได้ผลค่าต่างๆ แสดงบนแอพพลิเคชัน Blynk ดังรูปที่ 3 และ 4

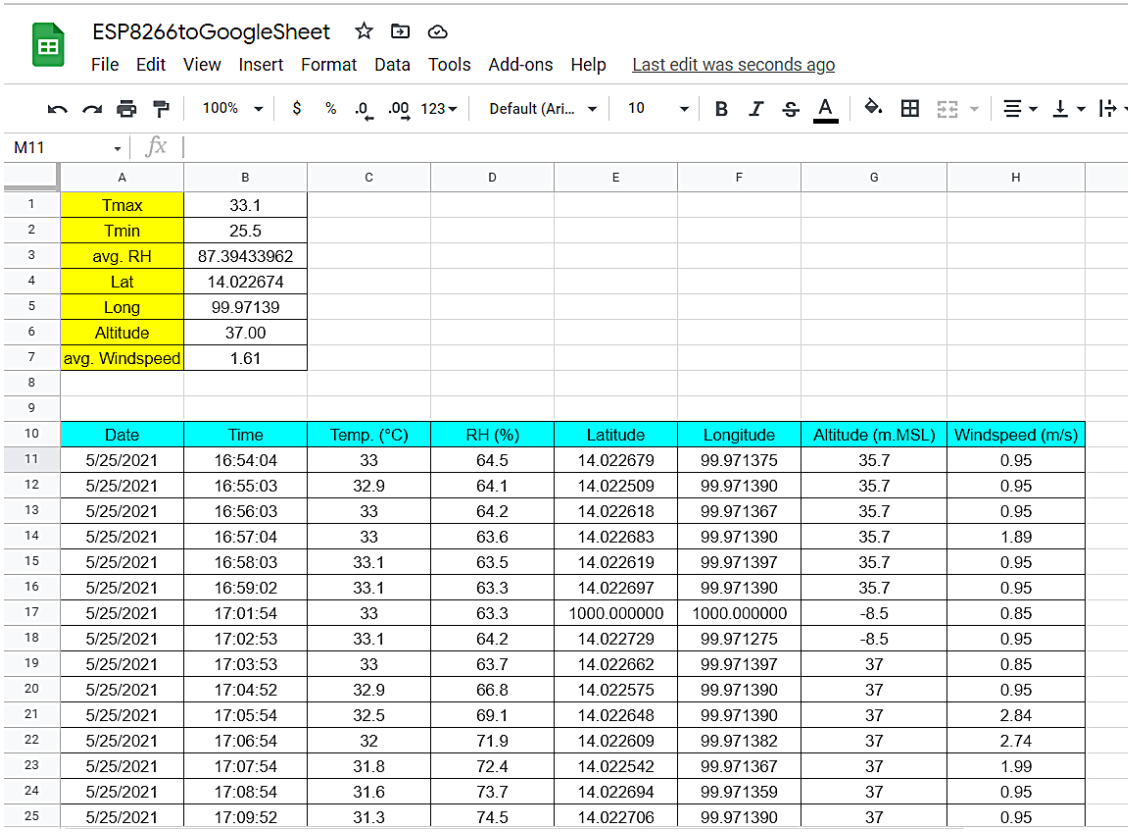


รูปที่ 3 และ 4 แสดงค่าแบบ Real-Time บนแอพพลิเคชัน Blynk

06 ทำการเชื่อมต่อ Sensor เข้ากับโปรแกรม Google Sheets เพื่อทำการบันทึกค่าลงใน Spreadsheet มีขั้นตอนดังนี้

1. สร้าง Sheets ใน Google sheet
2. สร้างหัวข้อที่หัวตารางว่าต้องการให้ส่งค่าอะไรเข้ามาบ้าง
3. กดเลือก เครื่องมือ --> โปรแกรมแก้ไขสคริปต์ --> เขียนสคริปต์ดังแสดงในหัวข้อที่ 3.4 Script ที่ใช้เขียนใน App Script เพื่อรับค่าเข้า Googlesheet และคำนวณ ETo
4. ในสคริปต์ส่วน var sheet id = 'xxx'; ให้ใส่ Spreadsheet ID
5. บันทึกและทำการตั้งชื่อ Script
6. กดเลือก เผยแพร่ --> ใช้งานเป็นแบบแอปพลิเคชันเว็บ... --> Anyone --> Deploy --> จะได้ URL ของสคริปต์
7. นำ URL ของสคริปต์ ไปแก้ไขใน Code ส่วน String GAS_ID = "xxx"; ในหัวข้อที่ 3.3 Total Code

เมื่อทำการ Compile Code ดังแสดงไว้ในหัวข้อที่ 3.3 Total Code แล้ว จะได้ผลค่าต่างๆ แสดงบน Google sheets ใน Spreadsheet ที่ชื่อว่า "DATA" ดังรูปที่ 5



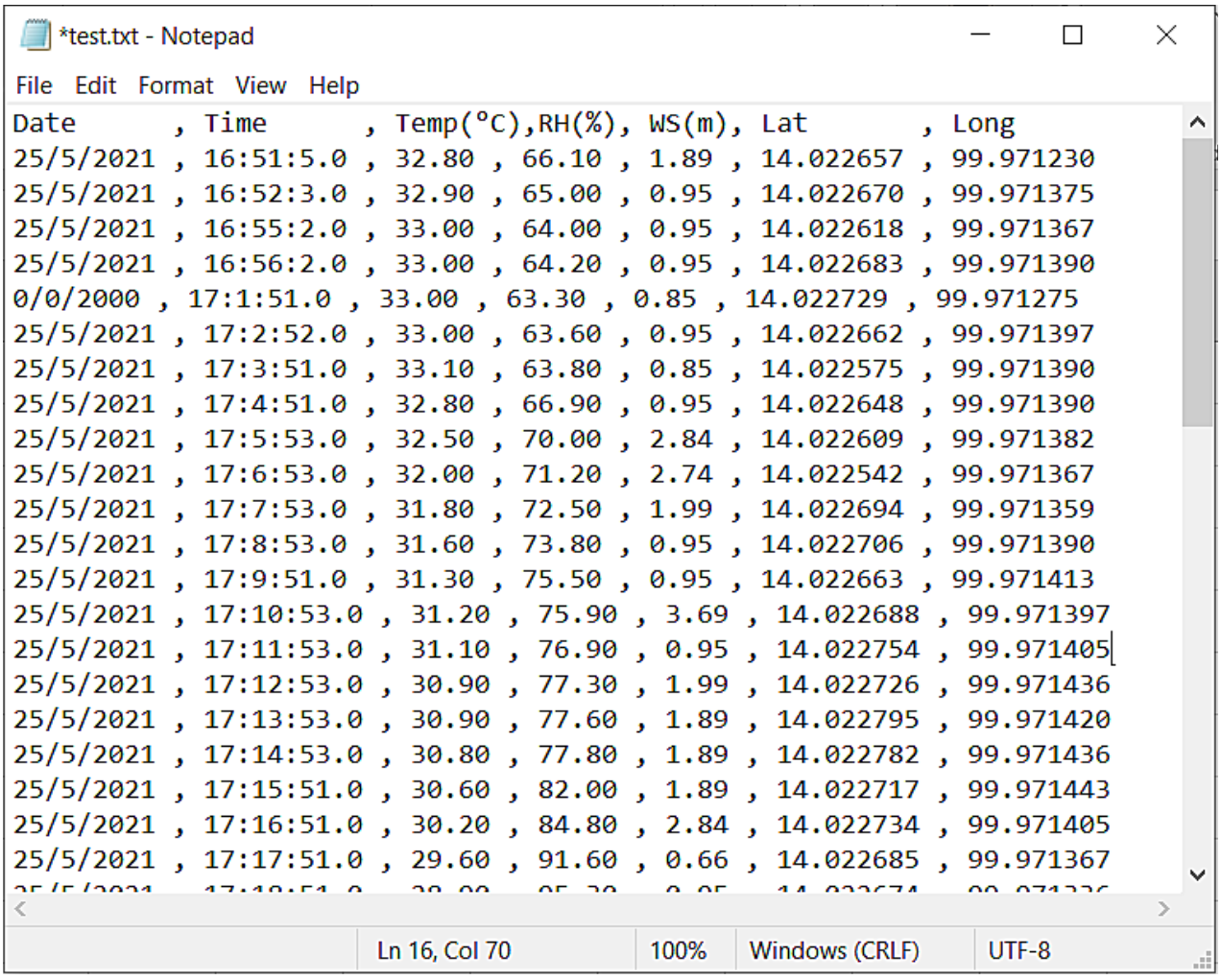
	A	B	C	D	E	F	G	H
1	Tmax	33.1						
2	Tmin	25.5						
3	avg. RH	87.39433962						
4	Lat	14.022674						
5	Long	99.97139						
6	Altitude	37.00						
7	avg. Windspeed	1.61						
8								
9								
10	Date	Time	Temp. (°C)	RH (%)	Latitude	Longitude	Altitude (m.MSL)	Windspeed (m/s)
11	5/25/2021	16:54:04	33	64.5	14.022679	99.971375	35.7	0.95
12	5/25/2021	16:55:03	32.9	64.1	14.022509	99.971390	35.7	0.95
13	5/25/2021	16:56:03	33	64.2	14.022618	99.971367	35.7	0.95
14	5/25/2021	16:57:04	33	63.6	14.022683	99.971390	35.7	1.89
15	5/25/2021	16:58:03	33.1	63.5	14.022619	99.971397	35.7	0.95
16	5/25/2021	16:59:02	33.1	63.3	14.022697	99.971390	35.7	0.95
17	5/25/2021	17:01:54	33	63.3	1000.000000	1000.000000	-8.5	0.85
18	5/25/2021	17:02:53	33.1	64.2	14.022729	99.971275	-8.5	0.95
19	5/25/2021	17:03:53	33	63.7	14.022662	99.971397	37	0.85
20	5/25/2021	17:04:52	32.9	66.8	14.022575	99.971390	37	0.95
21	5/25/2021	17:05:54	32.5	69.1	14.022648	99.971390	37	2.84
22	5/25/2021	17:06:54	32	71.9	14.022609	99.971382	37	2.74
23	5/25/2021	17:07:54	31.8	72.4	14.022542	99.971367	37	1.99
24	5/25/2021	17:08:54	31.6	73.7	14.022694	99.971359	37	0.95
25	5/25/2021	17:09:52	31.3	74.5	14.022706	99.971390	37	0.95

รูปที่ 5 แสดงผลค่าต่างๆบน Google sheets

07 ทำการเชื่อมต่อ Sensor เพื่อนรับค่าผ่าน SD Card มีขั้นตอนดังนี้

1. เปิด SD Card ขึ้นมาแล้วสร้างไฟล์ test.txt ขึ้นมาเพื่อให้ Arduino เขียนข้อมูลและอ่านข้อมูล
2. ทำการเขียน Code ดังแสดงในหัวข้อที่ 3.3 Total Code

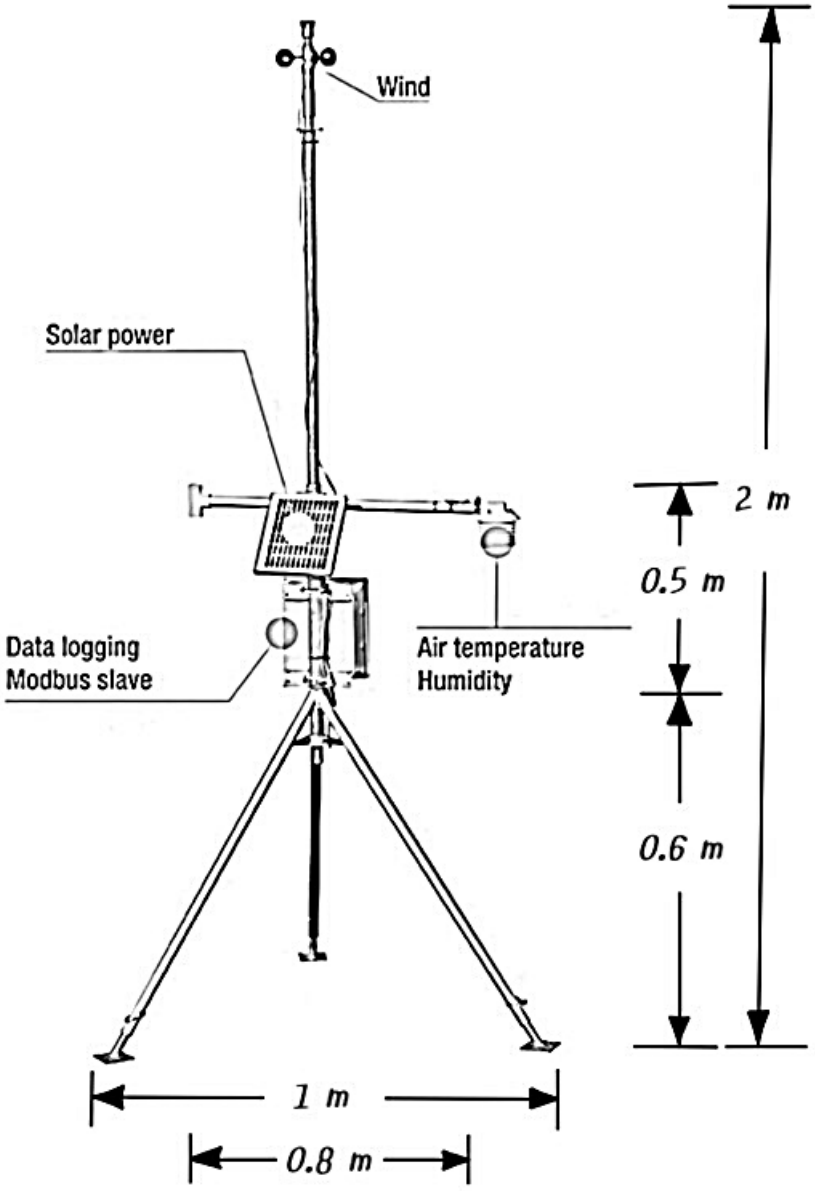
เมื่อทำการ Compile Code ดังแสดงไว้ในหัวข้อที่ 3.3 Total Code แล้ว จะได้ผลค่าต่างๆ แสดงบน SD Card ดังรูปที่ 6



รูปที่ 6 แสดงบันทึกผลค่าต่างๆบน SD Card

08 ออกแบบลักษณะโครงสร้างของสถานีวัดอากาศขนาดย่อม ให้เหมาะสมกับการใช้งาน

ในการออกแบบลักษณะโครงสร้างของสถานีวัดอากาศขนาดย่อม สามารถออกแบบได้ตามต้องการ โดยคำนึงถึงความแข็งแรง และเหมาะสมกับการใช้งาน โดยในคู่มือเล่มนี้ได้มีตัวอย่างการออกแบบโครงสร้างของสถานีวัดอากาศขนาดย่อม ดังแสดงในรูปที่ 7

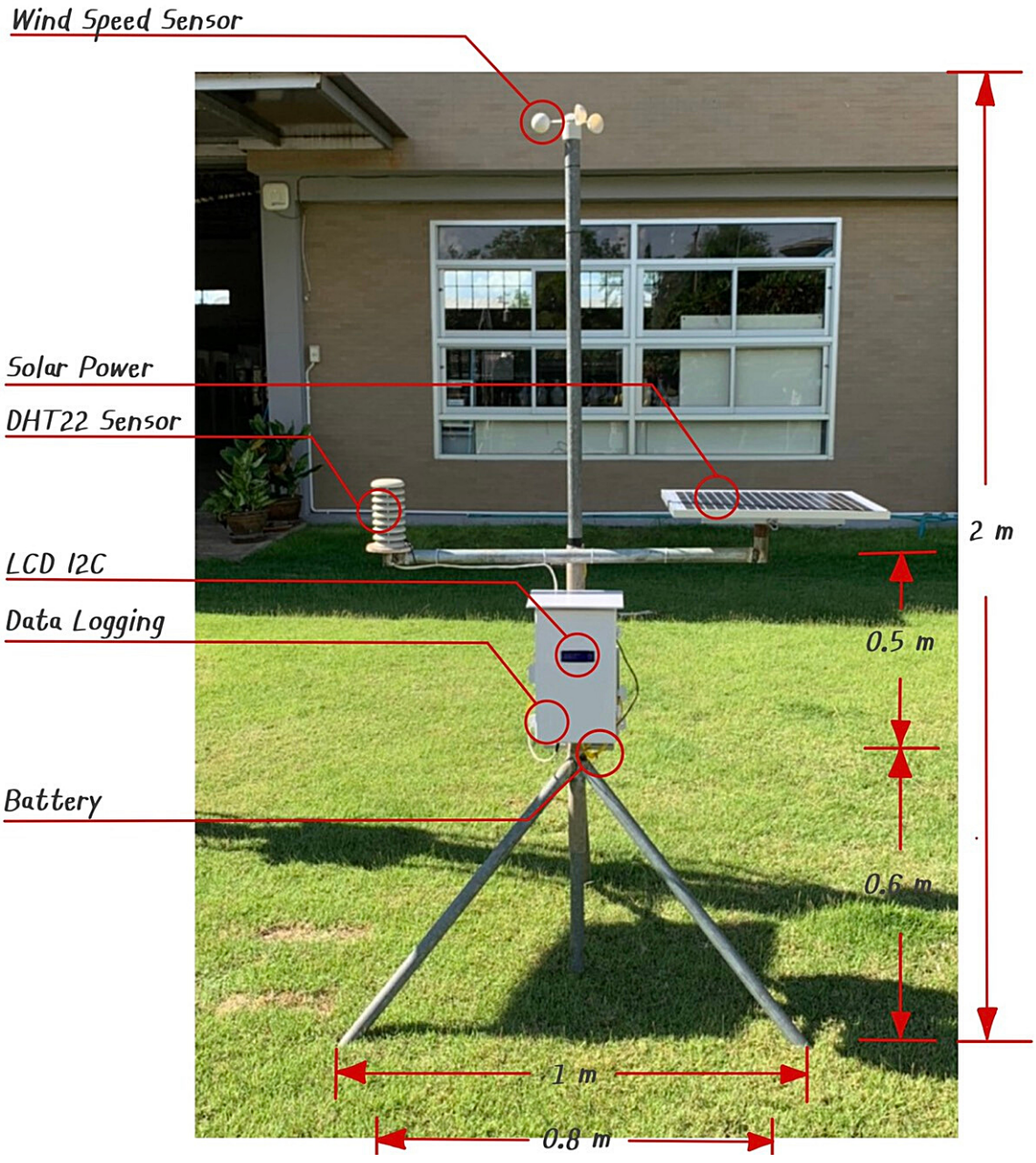


รูปที่ 7 ออกแบบลักษณะโครงสร้างของสถานีวัดอากาศขนาดย่อม

09

ทำการประกอบอุปกรณ์ทั้งหมด เพื่อสร้างสถานีวัดอากาศขนาดย่อมตามลักษณะที่ออกแบบไว้

เมื่อทำการออกแบบลักษณะโครงสร้างของสถานีวัดอากาศขนาดย่อมเรียบร้อยแล้ว จึงทำการประกอบอุปกรณ์ทั้งหมด ได้แก่ Sensor ทั้งหมด แบตเตอรี่ และแผงโซลาร์เซลล์ ตามลักษณะที่ออกแบบไว้ โดยกำหนดให้ตำแหน่งที่ตั้ง Wind Speed Sensor มีความสูงจากระดับพื้นดิน 2 เมตร ดังแสดงในรูปที่ 8



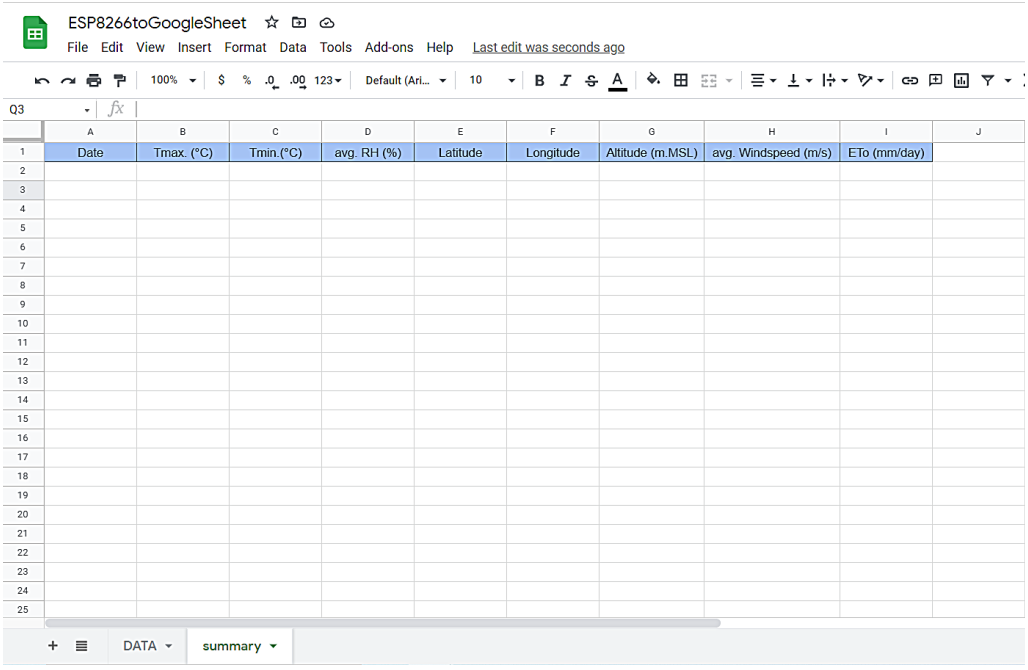
รูปที่ 8 สถานีวัดอากาศขนาดย่อมแบบ Real-Time

10 นำไปทดสอบเพื่อเปรียบเทียบ ตรวจสอบปัญหา และแก้ไขต่อไป

เมื่อทำการประกอบอุปกรณ์ทั้งหมด เพื่อสร้างสถานีวัดอากาศขนาดย่อมเรียบร้อยแล้ว ต้องนำสถานีวัดอากาศขนาดย่อมไปทดสอบเพื่อเปรียบเทียบค่าความถูกต้องกับสถานีวัดอากาศที่เชื่อถือได้ เช่น สถานีวัดอากาศของกรมอุตุนิยมวิทยา เป็นต้น เพื่อตรวจสอบว่าค่าที่วัดได้จากสถานีวัดอากาศขนาดย่อมที่สร้างขึ้น มีค่าแตกต่างกับสถานีอุตุนิยมวิทยามากน้อยเพียงใด เพื่อตรวจสอบ และแก้ไขปัญหาที่อาจเกิดขึ้นต่อไป ซึ่งผลการเปรียบเทียบค่าพารามิเตอร์ที่วัดได้จาก RID-KU001 Weather Station ณ สถานีอุตุนิยมวิทยา จ.นครปฐม และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station) แสดงดังตารางที่ 3 ในภาคผนวก ค ผลการเปรียบเทียบ

11 คำนวณ ETo

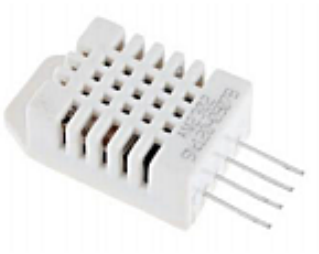
เมื่อได้ค่าที่ถูกต้องเชื่อถือได้ จึงใช้องค์ความรู้ทางด้านชลประทาน ในการนำค่ามาคำนวณปริมาณการใช้น้ำของพืชอ้างอิง (Reference Crop Evapotranspiration, ETo) โดยค่า ETo ที่ได้จะแสดงผลเป็นรายวันบน Spreadsheet ที่ชื่อว่า "Summary" ดังแสดงในรูปที่ 9



รูปที่ 9 แสดงผลค่า ETo เป็นรายวันบน Spreadsheet ที่ชื่อว่า "Summary"

2. รายละเอียดเครื่องมือ

ตางรางที่ 1 อุปกรณ์ที่ใช้ในการตรวจวัด เก็บข้อมูลและส่งข้อมูล

เครื่องมือและอุปกรณ์	Specification	หน้าที่ของอุปกรณ์
1. WeMos D1 Mini 	– Chip: ESP-8266EX – Compatible with Arduino IDE – Compatible NodeMCU firmware – GPIO pins: 11 – 1 pin ADC (0 V to 3.3 V) – Max Input Voltage: 24 V – Operating Frequency: 80 / 160 MHz – Flash Memory: 4MB – I2C interface – Size: 34.2 x 25.6 mm	- ควบคุมการทำงานของเซนเซอร์ทั้งหมด
2. DHT22 Sensor 	– Operating Voltage: 3.5V to 5.5V – Operating current: 0.3mA (measuring) 60uA (standby) – Output: Serial data – Temperature Range: -40℃ to 80℃ – Humidity Range: 0% to 100% – Resolution: Temperature and Humidity both are 16-bit – Accuracy: ±0.5℃ and ±1%	- สำหรับตรวจวัดอุณหภูมิและความชื้นสัมพัทธ์
3. Anemometer Wind Speed Sensor รุ่น WH-SP-WS01 	– Type: Anemometer – Material: Plastic – Accuracy: ± 1-3 cm	- สำหรับตรวจวัดความเร็วลม

เครื่องมือและอุปกรณ์	Specification	หน้าที่ของอุปกรณ์
4. SD Card Reader 	– Interface level 5V or 3.3V – Application Support Micro SD card (2G), micro SDHC card (32G) – Power Supply 4.5V to 5.5V, 3.3V voltage – support Micro SD card, SDHC Micro card (high speed card) – board level conversion circuit, that is, the interface can be 5V or 3.3V – power supply for the 4.5V, 5.5V plate load 3.3V voltage regulator circuit	- สำหรับอ่านข้อมูลที่เก็บไว้ใน SD Card
5. SD Card 	– ความจุ 4GB – ขนาดการ์ด microSDHC/SDXC 0.43" x 0.59" x 0.039" (11 x 15 x 1 มม.) – ขนาดหัวต่อ SD 0.94" x 1.26" x 0.08" (24 x 32 x 2.1 มม.) – พิกัดความเร็ว Class 4: อัตราการถ่ายโอนข้อมูลขั้นต่ำ 4MB/s – อุณหภูมิการทำงาน -13°F ถึง 185°F (-25°C ถึง 85°C) – อุณหภูมิในการจัดเก็บ -40°F ถึง 185°F – (-40°C ถึง 85°C)	- เก็บข้อมูล
6. จอแสดงผล LCD I2C 16×2 	– 16 characters x 2 lines – DC Power Input 5V – Operating Current: 30mA (typical) – Operating Current (Backlight Only): 20mA (typical) – Display Size (PCB): 80 x 36mm (3.1" x 1.4") – Display Height w/ PCB: 18mm (0.7") – Display Bezel: 72 x 25mm – (2.8 x 1")	- แสดงผลข้อมูลที่ตรวจวัดได้

เครื่องมือและอุปกรณ์	Specification	หน้าที่ของอุปกรณ์
7. Resistor 10 KΩ 	– Resistance (Ohm): 10000 – Tolerance: 5% – Composition: Carbon Film – Size: Standard – Mounting Feature: Through Hole – Packing Method: Cut Tape – Power Rating: 0.25 W – Resistance: 10 KΩ	- ต้านการไหลผ่านของกระแสไฟฟ้าที่เข้าไปยังอุปกรณ์อิเล็กทรอนิกส์
8. สาย USB 	– USB Micro-B connectors – 5-pin devices – support high-speed data transfers of 480 Mbit/s	- เชื่อมต่อระหว่างแบตเตอรี่กับบอร์ด WeMos D1 Mini
9. GPS Module 	– Power Supply Range: 3 V to 5 V – Model: GY-GPS6MV2 – Ceramic antenna – EEPROM for saving the configuration data when powered off – Backup battery – LED signal indicator – Antenna Size: 25 x 25 mm – Module Size: 25 x 35 mm – Mounting Hole Diameter: 3 mm – Default Baud Rate: 9600 bps	- ตรวจวัดตำแหน่งที่ตั้งของสถานีวัดอากาศ
10. 12V 12Ah Battery 	– ขนาด 9 x 15 x 9.5 cm. – น้ำหนัก 3.5 kg – ระดับแรงดันชาร์จ 13.5-13.8 โวลล์ – กระแสชาร์จ (Initial Current)1.2-2.40A MAX – การใช้งานกระแสต่อเนื่อง ที่ 0.6 A ได้ 20 ชั่วโมง – การใช้งานกระแสเต็ม ที่ 12 A ได้ 1 ชั่วโมง	- สำรองพลังงานไฟฟ้า

เครื่องมือและอุปกรณ์	Specification	หน้าที่ของอุปกรณ์
11. Solar Charger Controller 	– Operating temperature 0 - 50℃ – Storage temperature -10- 60℃ – Charge controller type: Two step charging algorithm – Battery temperature compensation 4 - 5mV/℃/cell – Solar Module size (Max): 40Wp – Indications: Changing and low battery	- รับพลังงานไฟฟ้ากระแสตรง (DC) จากแผงโซล่าเซลล์แล้วควบคุมการชาร์จประจุไฟฟ้าให้กับแบตเตอรี่อย่างเหมาะสม
12. Solar Cell 20W 	– ความยาวสาย 75 cm – Pmax : 20Wp – Vmp : 18V – Imp : 1.15A – Voc : 21V – Isc : 1.3A	- ใช้พลังงานแสงอาทิตย์ในการชาร์จแบตเตอรี่

3. CODE

- 3.1 โปรแกรมที่ใช้ในการเขียน CODE

รูปที่ 10 โปรแกรม ARDUINO VERSION 1.8.13

ดาวน์โหลดได้ที่ WEBSITE : HTTPS://WWW.ARDUINO.CC/EN/SOFTWARE

- 3.2 LIBRARY ที่ใช้ในการเขียน CODE

01

ESP8266 VERSION 2.7.4

WRITTEN BY ESP8266 COMMUNITY

- 3.2 LIBRARY ที่ใช้ในการเขียน CODE (ต่อ)

02 SOFTWARESERIAL VERSION=1.0
WRITTEN BY PAUL STOFFREGEN

03 TINYGPS
WRITTEN BY WAYNE HOLDER

04 SD
WRITTEN BY SPARKFUN ELECTRONICS

05 LIQUIDCRYSTAL I2C VERSION=1.1.2
WRITTEN BY FRANK DE BRABANDER

06 BLYNK VERSION=1.0.0-BETA.3
WRITTEN BY VOLODYMYR SHYMANSKYY

07 DHT LIBRARY MIT LICENSE
WRITTEN BY ADAFRUIT INDUSTRIES

08 TIME STABLE 1.6
WRITTEN BY PAULSTOFFREGEN

- 3.3 TOTAL CODE

```
#define BLYNK_PRINT Serial

#include <SPI.h>

#include <DHT.h>

#include <ESP8266WiFi.h>

#include <WiFiClientSecure.h>

#include <BlynkSimpleEsp8266.h>

#include <LiquidCrystal_I2C.h>

#include <Wire.h>

#include <SoftwareSerial.h>

#include <TinyGPS.h>

#include <SD.h>

#include <TimeLib.h>                    /* Program code related to Real Time
Clock (RTC). */

#include <WidgetRTC.h>                   /* Communication code with Blynk
Real Time Clock Widget */

#define DHTPIN D0

#define DHTTYPE DHT22

char auth[] = "6yOomrVt8L0bD6VPc3nYunvLaLIEYXqf";

LiquidCrystal_I2C lcd(0x27,20,4);
```

- 3.3 TOTAL CODE (ต่อ)

```
SoftwareSerial mySerial(D4, A0);//(RX,TX)

TinyGPS gps;

void gpsdump(TinyGPS &gps);

void printFloat(double f, int digits = 2);

void GPSdata();

const int RecordTime = 1; //Define Measuring Time (Seconds)

const int SensorPin = D3;  //Define Interrupt Pin (2 or 3 @ Arduino Uno)

const int chipSelect = D8;

volatile int InterruptCounter;

float WindSpeed;

float WS;

char ssid[] = "Complex Hydrologic Unit";

char pass[] = "SIP6230021";

//----------------------------------------Host & httpsPort

const char* host = "script.google.com";

const int httpsPort = 443;

// Initialize DHT sensor.

DHT dht(DHTPIN,DHTTYPE,12);

BlynkTimer timer;
```

- 3.3 TOTAL CODE (ต่อ)

```
WidgetRTC rtc;

unsigned long startMillis;                    /* start counting time for display
refresh*/

unsigned long currentMillis;                  /* current counting time for display
refresh */

const unsigned long period = 1000;            // refresh every X seconds (in
seconds) Default 60000 = 1 minute

WiFiClientSecure client; //--> Create a WiFiClientSecure object.

String GAS_ID = "AKfycby-ldoR7WMjFK-
A7VBhCWXxUVQ_lnTw3mP9Z3fEPC3hf99okGAvHOjKOFM4jJi4vQr-Ag";//-->
spreadsheet script ID

int count=0;

BLYNK_CONNECTED()                             /* When Blynk server is
connected, initiate Real Time Clock function */

{ rtc.begin(); }

void Display()

{

  float flat, flon ;

  gps.f_get_position(&flat, &flon);

  String  Strflat = String(flat, 6);
```

- 3.3 TOTAL CODE (ต่อ)

```
  String  Strflon = String(flon, 6);
  meassure();
if(currentMillis - startMillis > period)                    /* For every 1
second, run the set of code*/
  {
    String currentTime = String(hour()) + ":" + minute() + ":" + second();       /* Define
"currentTime" by combining hour, minute and second */
    String currentDate = String(day()) + " " + month() + " " + year();           /* Define
"currentDate" by combining day, month, and year */
    Blynk.virtualWrite(V1, currentTime);                    /* Send Time
parameters to Virtual Pin V1 on Blynk App */
    Blynk.virtualWrite(V2, currentDate);                    /* Send Date
parameters to Virtual Pin V2 on Blynk App */
    int getSecond = second();                               /* Assign "getHour"
as the hour now */
    if (getSecond > 30)
    { digitalWrite(2,HIGH);}                                /* Turn OFF the LED if
seconds count is more than 30 */
    if (getSecond < 30)
    { digitalWrite(2,LOW); }                                /* Turn ON the LED if
the seconds count is less than 30 */
```

- 3.3 TOTAL CODE (ต่อ)

```
  startMillis = millis();                                    /* Reset time for the
next counting cycle */

 }

 Blynk.virtualWrite(V4, WS);

 Blynk.virtualWrite(V6, dht.readTemperature());

 Blynk.virtualWrite(V5, dht.readHumidity());

 Blynk.virtualWrite(V8, Strflat);

 Blynk.virtualWrite(V9, Strflon);

 Blynk.virtualWrite(V10, gps.f_altitude());

 lcd.setCursor(0, 0);

 lcd.print("T:");

 lcd.print(dht.readTemperature());

 lcd.print(" ");

 lcd.print("H:");

 lcd.print(dht.readHumidity());

 lcd.print("%");

 lcd.setCursor(0, 1);

 lcd.print("WS:");

 lcd.print(WS);
```

- 3.3 TOTAL CODE (ต่อ)

```
  lcd.setCursor(10, 1);
  lcd.print(" m/s");
}
void sendSensor()
{
  float h = dht.readHumidity();
  float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit
  float flat, flon ;
  if (isnan(h) || isnan(t))
  {
    Serial.println("Failed to read from DHT sensor!");
    return;
  }
  // You can send any value at any time.
  // Please don't send more that 10 values per second.
  gps.f_get_position(&flat, &flon);
  Serial.print("Temp. = ");
  Serial.print(t);
  Serial.print(" C ");
```

- 3.3 TOTAL CODE (ต่อ)

```
  Serial.print("; Humidity = ");
  Serial.print(h);
  Serial.println(" %");
  Serial.print("Wind Speed: ");
  Serial.print(WS);       //Speed in km/h
  Serial.print(" km/h - ");
  Serial.print(WS); //Speed in m/s
  Serial.println(" m/s");
  GPSdata();
  String  Strflat = String(flat, 6);
  String  Strflon = String(flon, 6);
     sendData(t,h,Strflat,Strflon,gps.f_altitude(),WS);
}
void sendData(float string_temp,float string_humi,String string_Lat,String string_Long,
float string_mMSL ,float string_Windspeed)
{
  Serial.println("==========");
  Serial.print("connecting to ");
  Serial.println(host);
```

- 3.3 TOTAL CODE (ต่อ)

```
//----------------------------------Connect to Google host
if (!client.connect(host, httpsPort)) {
  Serial.println("connection failed");
  return;
}
String url = "/macros/s/" + GAS_ID + "/exec?temp=" + string_temp + "&humi=" + string_humi + "&Lat=" + string_Lat + "&Long=" + string_Long + "&mMSL=" + string_mMSL + "&Windspeed=" + string_Windspeed ;
Serial.print("requesting URL: ");
Serial.println(url);
client.print(String("GET ") + url + " HTTP/1.1\r\n" +
       "Host: " + host + "\r\n" +
       "User-Agent: BuildFailureDetectorESP8266\r\n" +
       "Connection: close\r\n\r\n");
Serial.println("request sent");
while (client.connected()) {
  String line = client.readStringUntil('\n');
  if (line == "\r") {
    Serial.println("headers received");
```

- 3.3 TOTAL CODE (ต่อ)

```
      break;
    }
  }
  String line = client.readStringUntil('\n');
  if (line.startsWith("{\"state\":\"success\"")) {
    Serial.println("esp8266/Arduino CI successfull!");
  } else {
    Serial.println("esp8266/Arduino CI has failed");
  }
  Serial.print("reply was : ");
  Serial.println(line);
  Serial.println("closing connection");
  Serial.println("==========");
  Serial.println();
}
void GPSdata()
{
    bool newdata = false;
    unsigned long start = millis();
```

- 3.3 TOTAL CODE (ต่อ)

```
// Every 5 seconds we print an update
while (millis() - start < 5000)
{
      if (mySerial.available())
  {
    char c = mySerial.read();
    //Serial.print(c); // uncomment to see raw GPS data
    if (gps.encode(c))
    {
        newdata = true;
        break; // uncomment to print new data immediately!
    }
  }
}
if (newdata)
  {
    Serial.println("Acquired Data");
    Serial.println("-----------");
    gpsdump(gps);
```

- 3.3 TOTAL CODE (ต่อ)

```
    Serial.println("-----------");
    Serial.println();
  }
}
void setup()
{
  Serial.begin(115200);
  //delay(500);
  mySerial.begin(9600);
  //delay(1000);
  Serial.println("uBlox Neo 6M");
  pinMode(SensorPin,INPUT);
  Serial.print("Testing TinyGPS library v. ");
  Serial.println(TinyGPS::library_version());
  Serial.println();
  Serial.print("Sizeof(gpsobject) = ");
  Serial.println(sizeof(TinyGPS));
  Serial.println();
  dht.begin();
```

- 3.3 TOTAL CODE (ต่อ)

```
  Blynk.begin(auth, ssid, pass, "pkdb10.thddns.net", 8334);
  Wire.begin();
  lcd.init();
  lcd.backlight();
//  WiFi.begin(ssid, password); //--> Connect to your WiFi router
  Serial.println("");
//----------------------------------Wait for connection
  Serial.print("Connecting");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
  }
  //----------------------------------
  //----------------------------------If successfully connected to the wifi router, the IP
Address that will be visited is displayed in the serial monitor
  Serial.println("");
  Serial.print("Successfully connected to : ");
  Serial.println(ssid);
  Serial.print("IP address: ");
```

- 3.3 TOTAL CODE (ต่อ)

```
  Serial.println(WiFi.localIP());
  Serial.println();
  //-----------------------------------
  startMillis = millis();


  client.setInsecure();
  timer.setInterval(500L, Display);
  timer.setInterval(60000L, sendSensor);
  timer.setInterval(60000L, sendSD);
}
//============================================================
================ void loop
void loop()
{
  Blynk.run();
  timer.run();
}
void sendSD()
{
```

- 3.3 TOTAL CODE (ต่อ)

```
while (!Serial) {

}

if (!SD.begin(chipSelect)) {

  Serial.println("Initialization failed!");

  return ;

}


String dataString = "";

float h = dht.readHumidity();

float t = dht.readTemperature();

byte month, day, hour, minute, second, hundredths;

int year;

float flat, flon;

gps.f_get_position(&flat, &flon);

gps.crack_datetime(&year, &month, &day, &hour, &minute, &second, &hundredths);

String  Strflat = String(flat, 6);

String  strflon = String(flon, 6);

dataString += static_cast<int>(day);

dataString += "/";
```

- 3.3 TOTAL CODE (ต่อ)

```
dataString += static_cast<int>(month);

dataString += "/";

dataString += (year);

dataString += " , ";

dataString += static_cast<int>(hour + 7);

dataString += ":";

dataString += static_cast<int>(minute);

dataString += ":";

dataString += static_cast<int>(second);

dataString += ".";

dataString += static_cast<int>(hundredths);

dataString += " , ";

dataString += String(t);

dataString += " , ";

dataString += String(h);

dataString += " , ";

dataString += (WS);

dataString += " , ";

dataString += String(flat, 6);
```

- 3.3 TOTAL CODE (ต่อ)

```
  dataString += " , ";
  dataString += String(flon, 6);
  File dataFile = SD.open("test.txt", FILE_WRITE);
  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
    Serial.println(dataString);
    Serial.println("==========");
    Serial.println();
  }
   else {
    Serial.println("error opening datalog.txt");
    Serial.println("==========");
    Serial.println();
  }
}
void gpsdump(TinyGPS &gps)
{
    float flat, flon;
```

- 3.3 TOTAL CODE (ต่อ)

```
unsigned long age, date, time, chars;
int year;
byte month, day, hour, minute, second, hundredths;
unsigned short sentences, failed;
gps.f_get_position(&flat, &flon, &age);
Serial.print("Lat/Long(float): ");
printFloat(flat, 6);
Serial.print(", ");
printFloat(flon, 6);
Serial.print(" Fix age: ");
Serial.print(age);
Serial.println("ms.");
gps.crack_datetime(&year, &month, &day, &hour, &minute, &second, &hundredths,
&age);
Serial.print("Date: ");
Serial.print(static_cast<int>(month));
Serial.print("/");
Serial.print(static_cast<int>(day));
Serial.print("/");
```

- 3.3 TOTAL CODE (ต่อ)

```
Serial.print(year);

Serial.print(" Time: ");

Serial.print(static_cast<int>(hour + 7));

Serial.print(":"); //Serial.print("UTC +08:00 Malaysia");

Serial.print(static_cast<int>(minute));

Serial.print(":");

Serial.print(static_cast<int>(second));

Serial.print(".");

Serial.print(static_cast<int>(hundredths));

Serial.print(" UTC +07:00 Bangkok");

Serial.print(" Fix age: ");

Serial.print(age);

Serial.println("ms.");

Serial.print("Alt(cm): ");

Serial.print(gps.altitude());

Serial.print(" Alt(m): ");

printFloat(gps.f_altitude());

Serial.println();

Serial.print("Speed (mps): ");
```

- 3.3 TOTAL CODE (ต่อ)

```
    printFloat(gps.f_speed_mps());
    Serial.println();
    gps.stats(&chars, &sentences, &failed);
    Serial.print("Stats: characters: ");
    Serial.print(chars);
    Serial.print(" sentences: ");
    Serial.print(sentences);
    Serial.print(" failed checksum: ");
    Serial.println(failed);
}
void printFloat(double number, int digits)
{
// Handle negative numbers
  if (number < 0.0)
  {
    Serial.print('-');
    number = -number;
  }
  // Round correctly so that print(1.999, 2) prints as "2.00"
```

- 3.3 TOTAL CODE (ต่อ)

```
double rounding = 0.5;
for (uint8_t i = 0; i < digits; ++i)
    rounding /= 10.0;
    number += rounding;
// Extract the integer part of the number and print it
unsigned long int_part = (unsigned long)number;
double remainder = number - (double)int_part;
Serial.print(int_part);
 // Print the decimal point, but only if there are digits beyond
 if (digits > 0)
      Serial.print(".");
 // Extract digits from the remainder one at a time
 while (digits-- > 0)
 {
     remainder *= 10.0;
     int toPrint = int(remainder);
     Serial.print(toPrint);
     remainder -= toPrint;
 }
```

- 3.3 TOTAL CODE (ต่อ)

```
}
void meassure()
{
  InterruptCounter = 0;
  attachInterrupt(digitalPinToInterrupt(SensorPin), countup, RISING);
  delay(1000 * RecordTime);
  detachInterrupt(digitalPinToInterrupt(SensorPin));
  WindSpeed = (float)InterruptCounter / (float)RecordTime * 2.4; //Windspeed (km/h)
  WS = (WindSpeed/3.6)*0.141881278113128;                    // Windspeed (m/s) *
adj.factor
}
ICACHE_RAM_ATTR void countup() {
  InterruptCounter++;
}
```

- 3.4 Script ที่ใช้เขียนใน App Script เพื่อรับค่าเข้า Google sheets และคำนวณ ETo

```
function doGet(e){
  Logger.log( JSON.stringify(e) );
    var Curr_Datex = new Date();
    var Curr_Timex = Utilities.formatDate(Curr_Datex, "Asia/Bangkok", 'HH:mm');
    if(Curr_Timex=="07:00"){reset();}
  var result = 'Ok';
  if (e.parameter == 'undefined'){
    result = 'No Parameters';
  }
  else {
   var sheet_id = 'XXXXX';        // Spreadsheet ID
    var Curr_Date = new Date();
    var Curr_Time = Utilities.formatDate(Curr_Date, "Asia/Bangkok", 'HH:mm:ss');
    var sheet = SpreadsheetApp.openById(sheet_id);
    var traget1 = sheet.getSheetByName("DATA");
    var newRow = traget1.getLastRow() + 1;
    var rowData = [];
    rowData[0] = Curr_Date; // Date in column A
    rowData[1] = Curr_Time; // Time in column B
    for (var param in e.parameter){
      Logger.log('In for loop, param=' + param);
      var value = stripQuotes(e.parameter[param]);
      Logger.log(param + ':' + e.parameter[param]);
      switch (param){
        case 'temp':
          rowData[2] = value; // Temperature in column C
          result = 'temp Written on column C';
```

- 3.4 Script ที่ใช้เขียนใน App Script เพื่อรับค่าเข้า Google sheets และคำนวณ ETo (ต่อ)

```
      break;
    case 'humi':
      rowData[3] = value; // Relative Humidity in column D
      result = 'humidity Written on column D';
      break;
    case 'Lat':
     rowData[4] = value; // Latitude in column E
      result = 'Lat Written on column E';
      break;
    case 'Long':
     rowData[5] = value; // Longtitude in column F
      result = 'Long Written on column F';
      break;
    case 'mMSL':
     rowData[6] = value; // altitude in column G
      result = 'mMSL Written on column G';
      break;
    case 'Windspeed':
     rowData[7] = value; // Windspeed in column H
      result = 'Windspeed Written on column H';
      break;
  }
 }
 Logger.log(JSON.stringify(rowData));
 var newRange = traget1.getRange(newRow, 1, 1, rowData.length);
 newRange.setValues([rowData]);
}
```

- 3.4 Script ที่ใช้เขียนใน App Script เพื่อรับค่าเข้า Google sheets และคำนวณ ETo (ต่อ)

```
  return ContentService.createTextOutput(result);
}
function stripQuotes( value ) {
  return value.replace(/^["']|['"]$/g, "");
}
function myETo(date,lat,altitude,tmax,tmin,rh,windspeed)
{
  var pi=3.14159265358979;
  var date = new Date();
  var D=date.getDate();
  var M=date.getMonth()+1;
  var Y=date.getFullYear();
  if(M<3){j = parseInt(275*M/9-30+D);}
  else if(M>2){if(Y%4 == 0){j = parseInt(275*M/9-30+D)-1;} else{j = parseInt(275*M/9-
30+D)-2;}}
  var e= 2.71828182845905
  var es_max = 0.6108*Math.pow(e,(17.27*tmax/(tmax+237.3)));
  var es_min = 0.6108*Math.pow(e,(17.27*tmin/(tmin+237.3)));
  var es=0.5*(es_max+es_min);
  var tmean=0.5*(tmax+tmin);
  var delta=4098*es/Math.pow(tmean+237.3,2);
  var ea=es*rh/100;
  var u2=windspeed;
  var P=(293-0.0065*altitude)/293;
  var P=101.3*Math.pow(P,5.256);
  var gamma=P*0.665/1000;
  var a=2*pi/365;
```

- 3.4 Script ที่ใช้เขียนใน App Script เพื่อรับค่าเข้า Google sheets และคำนวณ ETo (ต่อ)

```
  var solar_angle = 0.409*Math.sin(2*pi/365*j-1.39);
  var d_r=pi/180;
  var lat=lat*d_r;
  var ws = Math.acos(-Math.tan(lat)*Math.tan(solar_angle));
  var Gsc = 0.082 ;
  var dr = 1+0.033*Math.cos(2*pi/365*j);
  var Ra = (24*60/pi)*Gsc*dr*(ws*Math.sin(lat)*Math.sin(solar_angle)+Math.cos(lat)*Mat
h.cos(solar_angle)*Math.sin(ws));
  var N = (24/pi)*ws ;
  var Rs = 0.16*((tmax-tmin)^1/2)*Ra;
  var Rso = (0.75+2*Math.pow(10,-5)*altitude)*Ra;
  var SB_constant = 4.903*Math.pow(10,-9);
  var kmax = tmax+273.16;
  var kmin = tmin+273.16;
  var Rnl = SB_constant*0.5*(Math.pow(kmax,4)+Math.pow(kmin,4))*(0.34-
0.14*Math.pow(ea,0.5))*(1.35*(Rs/Rso)-0.35);
  var Rns = (1-0.23)*Rs;
  var Rn = Rns-Rnl;
  var G = 0;
  var denominator=delta+gamma*(1+0.34*u2);
  var solar=0.408*delta*(Rn-G)/denominator;
  var aero=gamma*(900/(tmean+273.16))*u2*(es-ea)/denominator;
  var eto=solar+aero;
  return eto;
}
function reset(){
```

- 3.4 Script ที่ใช้เขียนใน App Script เพื่อรับค่าเข้า Google sheets และคำนวณ ETo (ต่อ)

```
function reset(){
  var sheet_id = 'XXXXX';        // Spreadsheet ID
    var sheet = SpreadsheetApp.openById(sheet_id);
    var traget1 = sheet.getSheetByName("DATA");
    var traget2 = sheet.getSheetByName("summary");
     var date = traget1.getRange(12,1).getValue();
     var tmax = traget1.getRange(1,2).getValue();
     var tmin = traget1.getRange(2,2).getValue();
     var avgrh = traget1.getRange(3,2).getValue();
     var lat = traget1.getRange(4,2).getValue();
     var lon = traget1.getRange(5,2).getValue();
     var alt = traget1.getRange(6,2).getValue();
     var avgws = traget1.getRange(7,2).getValue();
     var eto = myETo(date,lat,alt,tmax,tmin,avgrh,avgws);
     var next = traget2.getLastRow()+1;
     traget2.getRange(next,1).setValue(date);
     traget2.getRange(next,2).setValue(tmax);
     traget2.getRange(next,3).setValue(tmin);
     traget2.getRange(next,4).setValue(avgrh);
     traget2.getRange(next,5).setValue(lat);
     traget2.getRange(next,6).setValue(lon);
     traget2.getRange(next,7).setValue(alt);
     traget2.getRange(next,8).setValue(avgws);
     traget2.getRange(next,9).setValue(eto);
     traget1.getRange('A11:H10000').clearContent();
}
```

4. แหล่งที่ซื้ออุปกรณ์และราคา

ตางรางที่ 2 แหล่งที่ซื้ออุปกรณ์และราคา

เครื่องมือและอุปกรณ์	แหล่งที่ซื้อ	ราคา (บาท)
1. WeMos D1 Mini	https://shopee.co.th/product/66839018/6945694042?smtt=0.44999811-1620203008.9	76
2. DHT22 Sensor	https://shopee.co.th/product/117988183/2664904166?smtt=0.244853150-1620202956.9	128
3. WH-SP-WS01 Anemometer Wind Speed Sensor	https://shopee.co.th/product/217586058/3882233550?smtt=0.44999811-1620202871.9	663
4. SD Card Reader	https://shopee.co.th/product/56441528/2451089550?smtt=0.44999811-1620203063.9	27
5. SD Card	https://shopee.co.th/product/65908805/4152381951?smtt=0.44999811-1620203154.9	78
6. LCD I2C 16×2	https://shopee.co.th/product/117988183/1884276985?smtt=0.244853150-1620203048.9	88
7. Resistor 10KOhm	https://shopee.co.th/product/270502312/4759876658?smtt=0.44999811-1620203952.9	19
8. สาย USB	https://shopee.co.th/product/313432440/7955846333?smtt=0.44999811-1620204144.9	35
9. GPS Module	โมดูล GPS Ublox 6M, 3V-5V power supply universal | Lazada.co.th	219
10. แบตเตอรี่ 12V 12AH	https://shopee.co.th/product/252046118/4143066551?smtt=0.44999811-1620964237.9	420
11. Solar Charger Controller 30A	https://shopee.co.th/product/399459090/8029249574?smtt=0.44999811-1620964302.9	136
12. Solar Cell 20W	https://shopee.co.th/product/322627514/4058330629?smtt=0.44999811-1620964279.9	369

5. ข้อเสนอแนะในการพัฒนาต่อไป

01 เลือกใช้ฮาร์ดแวร์ที่มีเกรดได้มาตรฐานเพื่อความคงทนของอุปกรณ์ที่นำมาใช้งาน

02 เพิ่มเซนเซอร์วัดความเข้มแสงอาทิตย์เพื่อที่สามารถใช้การคำนวณปริมาณการใช้น้ำของพืชอ้างอิงโดยให้มีความแม่นยำในการคำนวณมากขึ้น

03 ออกแบบสถานีวัดอากาศขนาดย่อมแบบ Real-Time ให้มีความแข็งแรงให้สามารถวางอยู่ได้ในทุกสภาพอากาศ เพื่อไม่ให้สถานีได้รับความเสียหายเนื่องจากคุณภาพของวัสดุที่นำมาใช้

04 หาวิธีการระบุตำแหน่งของสถานีวัดอากาศขนาดย่อมแบบ Real-Time ในวันที่ท้องฟ้าไม่ปลอดโปร่ง เนื่องจาก GPS Module ที่นำมาใช้ไม่สามารถระบุพิกัดตำแหน่งได้ในวันที่ท้องฟ้ามีเมฆครึ้ม

05 จาก Total Code ดังแสดงในหัวข้อที่ 3.3 ทำให้ทุกครั้งที่ทำการเคลื่อนย้ายสถานีวัดอากาศไปติดตั้งในบริเวณที่ต้องการตรวจวัดใหม่ จะต้องทำการเชื่อมต่อ WiFi ใหม่ในบริเวณนั้น ซึ่งจะต้องทำการแก้ไข Code ในส่วนของ WiFi แล้วจึงทำการ Upload Code ลง WeMos D1 mini ใหม่อีกครั้ง จึงเสนอให้มีการปรับใช้ WiFiManager library เพื่อให้ไม่ต้องทำการแก้ไข Code ทุกครั้งที่ทำการเคลื่อนย้ายสถานีวัดอากาศ

บรรณานุกรม

กิตติพงษ์ สุวรรณราช. 2563. EP37: **รายงานสภาพอากาศจากอุปกรณ์ Internet of Things ESP8266 และ Google Sheets ผ่าน Google Data Studio.** แหล่งที่มา: https://www.youtube.com/watch?v=vS9--BNMX54, 8 มีนาคม 2564.

กิตติพงษ์ สุวรรณราช. 2563. **NodeMCU ESP8266 ส่งค่าอุณหภูมิและความชื้นไปเก็บใน Google Sheets อัตโนมัติ.** แหล่งที่มา: https://www.youtube.com/watch?v=bFskgdfvFEc, 6 มีนาคม 2564.

ประภาส สุวรรณเพชร. 2561. **งานครั้งที่ 44 [iot#19 GoogleSheetAPI] บันทึกค่าลงบน Google Sheet.** แหล่งที่มา: https://www.praphas.com/forum/index.php?topic=355.0, 16 มีนาคม 2564.

นิรนาม. 2562. **สอนใช้งาน Arduino แสดงข้อความ และ ค่า Sensor ต่างๆออกจอ LCD 1602 แบบ I2C.** แหล่งที่มา: https://www.myarduino.net/article/97/สอนใช้งาน-arduino-แสดงข้อความ-และ-ค่า-sensor-ต่างๆออกจอ-lcd-1602-แบบ-i2c, 6 มีนาคม 2564.

ทรงธรรมการไฟฟ้า. 2021. หัวแร้ง-ตะกั่วบัตกรี. แหล่งที่มา: songthamelec.com/category/237/เครื่องมือสำหรับช่าง/หัวแร้ง-ตะกั่วบัตกรี, 12 พฤษภาคม 2564.

AB-Marker. 2562. Prototype PCB Board 4x6 cm สีเขียว แผ่นปริ้นไข่ปลา แผ่นปริ้นอเนกประสงค์. แหล่งที่มา: https://www.ab.in.th/product/39/prototype-pcb-board-4x6-cm-สีเขียว-แผ่นปริ้นไข่ปลา-แผ่นปริ้นอเนกประสงค์, 12 พฤษภาคม 2564.

Addicore. 2019. **Male-Male Jumper Wires - 40 x 200mm (7.8in).** Available Source: https://www.addicore.com/Male-Male-Jumper-Wires-40-x-200mm-7-8in-p/217.htm, 12 May 2021.

AliExpress. n.d. UNI-T UT33B + multimetro Digital. Available Source: https://es.aliexpress.com/i/33004287377.html, 12 May 2021.

Arduino4. 2021. **GY-NEO6MV2 Ublox GPS Module.** Available Source: https://www.arduino4.com/product/231/gy-neo6mv2-ublox-gps-module, 12 May 2021.

BB.lighting21. ม.ป.ป. ตู้กันน้ำพลาสติกมีที่กันฝน Nano-102w. แหล่งที่มา: https://www.lazada.co.th/products/nano-102w-i868674134.html, 12 พฤษภาคม 2564.

Components101. 2018. **Breadboard.** Available Source: https://components101.com/misc/breadboard-connections-uses-guide, 12 May 2021.

Components101. 2018. **DHT22 – Temperature and Humidity Sensor.** Available Source: https://components101.com/sensors/dht22-pinout-specs-datasheet, 12 May 2021.

บรรณานุกรม

Crystal Technologies. 2017. DHT22 Temperature and Humidity Sensor. Available Source: https://th.cytron.io/c-sensor/p-dht22-temperature-and-humidity-sensor, 12 May 2021.

Element14. QUALTEK ELECTRONICS 3025030-03 USB CORD. n.d. Available Source: https://th.element14.com/qualtek-electronics/3025030-03/usb-cord-2-0-plug-a-micro-b-914mm/dp/2809820, 12 May 2021.

Engineering360. n.d. **USB Cables Information.** Available Source: https://www.globalspec.com/learnmore/electrical_electronic_components/wires_cables /usb_cables, 12 May 2021.

ESR Electronic Components Ltd. n.d. **Electric Soldering Irons.** Available Source: https://www.esr.co.uk/electronics/tools-soldering-electric.htm, 12 May 2021.

FOG. 2020. solar charger controller. แหล่งที่มา: https://www.lazada.co.th/products/solar-charger-controller-10a-12v24v-2usb-5v-i232914928.html, 12 พฤษภาคม 2564.

IndiaMART. n.d. Jumper Wire (M-M, M-F, F-F). Available Source: https://www.indiamart.com/proddetail/jumper-wire-m-m-m-f-f-f-15359553673.html, 12 May 2021.

Jameco Electronics. n.d. **Resistor Carbon Film 10k Ohm 1/4 Watt 5%.** Available Source: https://www.jameco.com/z/CF1-4W103JRC-Jameco-Valuepro-Resistor-Carbon-Film-10k-Ohm-1-4-Watt-5-_691104.html, 12 May 2021.

James Home Renovation. หัวแร้งบัดกรี ตะกั่ว และน้ำยาเชื่อมประสาน. แหล่งที่มา: jameshomerenovation.com/โปรโมชั่น-ตราหัวนก-ตะกั่/, 12 พฤษภาคม 2564.

JIB Computer Group. 2017. **4 GB MICRO SD CARD KINGSTON CLASS 4 (SDC4/4GB).** Available Source: https://www.jib.co.th/web/index.php/product/readProduct/5161/20/4-Gb-Micro-Sd-Card-Kingston-Class-4, 12 May 2021.

KJ MALL. 2019. แผงโซล่าเซลล์ 20W 12V. แหล่งที่มา: https://www.lazada.co.th/products/20w-12v-poly-25-solar-cell-solar-panel-polycrystalline-solar-panel-i720232643.html, 12 พฤษภาคม 2564.

Made-in-China. 2018. 6 Pin Read and Write Micro SD Card Module Mini TF Card Reader. Available Source: https://kuongshun.en.made-in-china.com/product/JjfxBFtAbMcs/China-6-Pin-Read-and-Write-Micro-SD-Card-Module-Mini-TF-Card-Reader.html, 12 May 2021.

บรรณานุกรม

Myarduino. 2560. สอนใช้งาน GPS Module GY-NEO6MV2 กับ Arduino. แหล่งที่มา: myarduino.net/article/65/สอนใช้งาน-gps-module-gy-neo6mv2-กับ-arduino, 12 พฤษภาคม 2564.

OMICRON. 2020. EARLY WARNING COMPLETE SYSTEM. Available Source: https://omicroning.co/documentos/EARLY-WARNING-COMPLETE-SOLUTION.pdf, 12 May 2021.

Open Impulse. 2012. **WeMos D1 Mini ESP8266 Development Board.** Available Source: https://www.openimpulse.com/blog/products-page/product-category/wemos-d1-mini-esp8266-development-board/, 12 May 2021.

OurEdublog. 2021. **SOLAR CHARGE CONTROLLER.** Available Source: https://blog.oureducation.in/solar-charge-controller-2/, 12 May 2021.

ProtoSupplies. 2021. **PCB, 8 x 12 cm Universal Prototype Board.** Available Source: https://protosupplies.com/product/pcb-8-x-12-cm-universal-prototype-board/, 12 May 2021.

TimUR Team. 2019. Pengenalan Microcontroller. Available Source: https://timur.ilearning.me/2019/10/04/pengenalan-mikrokontroler/, 12 May 2021.

Wasan DIY. 2563. **Win Speed Sensor วัดความเร็วลม ด้วยเซ็นเซอร์วัดลม วัดค่าระดับความเร็ว Arduino LCD #DIY Arduino EP.38.** แหล่งที่มา: https://www.youtube.com/watch?v=fL5DP9ZwVtI, 11 มีนาคม 2564.

Work in. 2560. Resistors 10K Ohm 1/4W 5% Carbon Film. แหล่งที่มา: http://www.workin.in.th/product/329/resistors-10k-ohm-1-4w-5-carbon-film, 12 พฤษภาคม 2564.

ภาคผนวก ก รายการคำนวณ

- **คำนวณกำลังไฟฟ้าที่ต้องใช้ของสถานีวัดอากาศขนาดย่อม**

คำนวณกำลังไฟฟ้าที่ต้องใช้ (คิดจากกำลังสูงสุดที่ใช้) แสดงการคำนวณได้ดังนี้

- Wemos D1 mini = 0.15A × 5V × 24Hrs.
- GY-GPS6MV2 = 0.1A × 5V × 24Hrs.
- LCD-I2C 16,2 = 0.0015A × 5V × 24Hrs.
- DHT22 = 0.16A × 5V × 24Hrs.
- Micro SD Card Adapter = 0.2A × 5V × 24Hrs.
- Wind Speed Sensor = 0.1A × 5V × 24Hrs.

รวมทั้งสิ้น = 85.38 Watt-hr.

ดังนั้นใช้ที่ = 90 Watt-hr.

- กำหนดชั่วโมงแสงแดด = 5 Hrs.

ดังนั้น ต้องใช้ Solar Cell ขนาด = $\frac{90\ Watt-hr.}{5Hrs.}$ = 18 Watt

ดังนั้นเลือกใช้ Solar Cell ขนาด 20W/18V/1.1A

- ขนาดของ Deep Cycle Battery = $\frac{90\ Watt-hr.}{12V \times 0.6 \times 0.85}$ = **15 Ah**

- ขนาดของ Solar Charger = **เลือกให้มีขนาดใหญ่กว่า Solar Cell จึงเลือกที่ 12V, 30A**

- ระยะเวลาการใช้งาน Battery

ความจุ Battery = 12000 mAh

กระแสไฟอุปกรณ์รวม = 711.5 mA

ให้อัตราการสิ้นเปลืองพลังงาน = 0.7

ดังนั้น t = $\frac{12000\ mAh}{711.5\ mA} \times 0.7$ = **11.8 Hrs.**

- **คำนวณความสามารถในการบันทึกข้อมูลของ SD Card ความจุ 4 GB**

การบันทึกข้อมูลของ SD Card ความจุ 4 GB แสดงการคำนวณได้ดังนี้

SD Card ความจุ 4 GB ในความจุ 1 ชุดข้อมูลใช้ความจุ 135 byte

ดังนั้น จำนวนข้อมูลที่บันทึกได้ $= \frac{4\times10^9}{135}$

$=$ 29629629 ชุดข้อมูล

ทำการส่งข้อมูลทุก 1 นาที จะจัดเก็บข้อมูลได้ $= \frac{29629629}{1440}$

$=$ 20576 วัน

สามารถบันทึกข้อมูลได้ทั้งหมด $= \frac{20576}{365}$

$=$ 56 ปี

ภาคผนวก ข CODE ที่ใช้ทดสอบ SENSOR

- Code ที่ใช้ในการทดสอบ DHT22 & SD Card Sensor

```
#include <ESP8266WiFi.h>

#include <SPI.h>

#include <SD.h>

#include "DHT.h"

const int chipSelect = D8;  // used for Arduino

//const int chipSelect = D8;  // used for ESP8266

//#define DHTPIN D4     // used for Arduino

#define DHTPIN D0    // used for ESP8266

#define DHTYPE DHT22

DHT dht(DHTPIN, DHTYPE,12);

void setup() {

  Serial.begin(115200);

  while (!Serial) {

  }

  if (!SD.begin(chipSelect)) {

    Serial.println("Initialization failed!");

    while (1);

  }
```

- Code ที่ใช้ในการทดสอบ DHT22 & SD Card Sensor (ต่อ)

```
 dht.begin();
}
void loop() {
  String dataString = "";
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  dataString += String(t);
  dataString += ",";
  dataString += String(h);
  File dataFile = SD.open("test.txt", FILE_WRITE);
  if (dataFile) {
    dataFile.println(dataString);
    dataFile.close();
    Serial.println(dataString);
  }
  else {
    Serial.println("error opening datalog.txt");
  }
  delay(3000);     }
```

- Code ที่ใช้ในการทดสอบ Wind Speed & LCD I2C Sensor

```
#include<ESP8266WiFi.h>

#include <LiquidCrystal_I2C.h>

const int RecordTime = 3; //Define Measuring Time (Seconds)

const int SensorPin = D3;  //Define Interrupt Pin (2 or 3 @ Arduino Uno)

LiquidCrystal_I2C lcd(0x27,20,4);

volatile int InterruptCounter;

float WindSpeed;

void setup()

{

  Serial.begin(115200);

   lcd.init();

  lcd.backlight();

}

void meassure()

{

  InterruptCounter = 0;

  attachInterrupt(digitalPinToInterrupt(SensorPin), countup, RISING);

  delay(1000 * RecordTime);
```

- Code ที่ใช้ในการทดสอบ Wind Speed & LCD I2C Sensor (ต่อ)

```
  detachInterrupt(digitalPinToInterrupt(SensorPin));
  WindSpeed = (float)InterruptCounter / (float)RecordTime * 2.4;
}
ICACHE_RAM_ATTR void countup() {
  InterruptCounter++;
}
void loop()
{
  meassure();
  Serial.print("Wind Speed: ");
  Serial.print(WindSpeed);       //Speed in km/h
  Serial.print(" km/h - ");
  Serial.print(WindSpeed / 3.6); //Speed in m/s
  Serial.println(" m/s");
  lcd.setCursor(0, 0);
  lcd.print(WindSpeed / 3.6);
}
```

- Code ที่ใช้ในการทดสอบ GPS Sensor

```
#include <ESP8266WiFi.h>

#include <SoftwareSerial.h>

#include <TinyGPS.h>

SoftwareSerial mySerial(D4, A0);//(RX,TX)

TinyGPS gps;

void gpsdump(TinyGPS &gps);

void printFloat(double f, int digits = 2);

void setup()

{

// Oploen serial communications and wait for port to open:

Serial.begin(115200);

// set the data rate for the SoftwareSerial port

mySerial.begin(9600);

delay(1000);

Serial.println("uBlox Neo 6M");

Serial.print("Testing TinyGPS library v. ");

Serial.println(TinyGPS::library_version());

Serial.println("by http://www.arduino.codemobiles.com");
```

- Code ที่ใช้ในการทดสอบ GPS Sensor (ต่อ)

```
Serial.println();
Serial.print("Sizeof(gpsobject) = ");
Serial.println(sizeof(TinyGPS));
Serial.println();
}
void loop() // run over and over
{
  gpsdump(gps);
bool newdata = false;
unsigned long start = millis();
// Every 5 seconds we print an update
while (millis() - start < 5000)
{
if (mySerial.available())
{
char c = mySerial.read();
//Serial.print(c); // uncomment to see raw GPS data
if (gps.encode(c))
{
```

- Code ที่ใช้ในการทดสอบ GPS Sensor (ต่อ)

```
newdata = true;
break; // uncomment to print new data immediately!
}
}
}
if (newdata)
{
Serial.println("Acquired Data");
Serial.println("-------------");
gpsdump(gps);
Serial.println("-------------");
Serial.println();
}
}
void gpsdump(TinyGPS &gps)
{
long lat, lon;
float flat, flon;
unsigned long age, date, time, chars;
```

- Code ที่ใช้ในการทดสอบ GPS Sensor (ต่อ)

```
int year;

byte month, day, hour, minute, second, hundredths;

unsigned short sentences, failed;

gps.get_position(&lat, &lon, &age);

Serial.print("Lat/Long(10^-5 deg): ");

Serial.print(lat);

Serial.print(", ");

Serial.print(lon);

Serial.print(" Fix age: ");

Serial.print(age);

Serial.println("ms.");

// On Arduino, GPS characters may be lost during lengthy Serial.print()

// On Teensy, Serial prints to USB, which has large output buffering and

// runs very fast, so it's not necessary to worry about missing 4800

// baud GPS characters.

gps.f_get_position(&flat, &flon, &age);

Serial.print("Lat/Long(float): ");

printFloat(flat, 5);

Serial.print(", ");
```

- Code ที่ใช้ในการทดสอบ GPS Sensor (ต่อ)

```
printFloat(flon, 5);

Serial.print(" Fix age: ");

Serial.print(age);

Serial.println("ms.");

gps.get_datetime(&date, &time, &age);

Serial.print("Date(ddmmyy): ");

Serial.print(date);

Serial.print(" Time(hhmmsscc): ");

Serial.print(time);

Serial.print(" Fix age: ");

Serial.print(age);

Serial.println("ms.");

gps.crack_datetime(&year, &month, &day, &hour, &minute, &second, &hundredths, &age);

Serial.print("Date: ");

Serial.print(static_cast<int>(month));

Serial.print("/");

Serial.print(static_cast<int>(day));

Serial.print("/");

Serial.print(year);
```

- Code ที่ใช้ในการทดสอบ GPS Sensor (ต่อ)

```
Serial.print(" Time: ");

Serial.print(static_cast<int>(hour + 7));

Serial.print(":"); //Serial.print("UTC +08:00 Malaysia");

Serial.print(static_cast<int>(minute));

Serial.print(":");

Serial.print(static_cast<int>(second));

Serial.print(".");

Serial.print(static_cast<int>(hundredths));

Serial.print(" UTC +07:00 Bangkok");

Serial.print(" Fix age: ");

Serial.print(age);

Serial.println("ms.");

Serial.print("Alt(cm): ");

Serial.print(gps.altitude());

Serial.print(" Course(10^-2 deg): ");

Serial.print(gps.course());

Serial.print(" Speed(10^-2 knots): ");

Serial.println(gps.speed());

Serial.print("Alt(float): ");
```

- Code ที่ใช้ในการทดสอบ GPS Sensor (ต่อ)

```
printFloat(gps.f_altitude());
Serial.print(" Course(float): ");
printFloat(gps.f_course());
Serial.println();
Serial.print("Speed(knots): ");
printFloat(gps.f_speed_knots());
Serial.print(" (mph): ");
printFloat(gps.f_speed_mph());
Serial.print(" (mps): ");
printFloat(gps.f_speed_mps());
Serial.print(" (kmph): ");
printFloat(gps.f_speed_kmph());
Serial.println();
gps.stats(&chars, &sentences, &failed);
Serial.print("Stats: characters: ");
Serial.print(chars);
Serial.print(" sentences: ");
Serial.print(sentences);
Serial.print(" failed checksum: ");
```

- Code ที่ใช้ในการทดสอบ GPS Sensor (ต่อ)

```
Serial.println(failed);

}

void printFloat(double number, int digits)

{

// Handle negative numbers

if (number < 0.0)

{

Serial.print('-');

number = -number;

}

// Round correctly so that print(1.999, 2) prints as "2.00"

double rounding = 0.5;

for (uint8_t i = 0; i < digits; ++i)

rounding /= 10.0;

number += rounding;

// Extract the integer part of the number and print it

unsigned long int_part = (unsigned long)number;

double remainder = number - (double)int_part;

Serial.print(int_part);
```

- Code ที่ใช้ในการทดสอบ GPS Sensor (ต่อ)

```
// Print the decimal point, but only if there are digits beyond

if (digits > 0)

Serial.print(".");

// Extract digits from the remainder one at a time

while (digits-- > 0)

{

remainder *= 10.0;

int toPrint = int(remainder);

Serial.print(toPrint);

remainder -= toPrint;

}

}
```

- Code ที่ใช้ในการทดสอบ Blynk

```
#define BLYNK_PRINT Serial

#include <SPI.h>

#include <DHT.h>

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>

#include <LiquidCrystal_I2C.h>

#include <Wire.h>

#define DHTPIN D6

#define DHTTYPE DHT22

char auth[] = "XXXX";

LiquidCrystal_I2C lcd(0x27,16,2);

char ssid[] = "XXXX";

char pass[] = "XXXX";

DHT dht(DHTPIN,DHTTYPE,12);

BlynkTimer timer;

void sendSensor()

{

  float h = dht.readHumidity();
```

- Code ที่ใช้ในการทดสอบ Blynk (ต่อ)

```
float t = dht.readTemperature(); // or dht.readTemperature(true) for Fahrenheit

if (isnan(h) || isnan(t)) {

  Serial.println("Failed to read from DHT sensor!");

  return;

}

// You can send any value at any time.

// Please don't send more that 10 values per second.

Blynk.virtualWrite(V6, t);

Blynk.virtualWrite(V5, h);

Serial.print(t);

Serial.print(";");

Serial.println(h);

lcd.setCursor(0, 0);

lcd.print("Hum:      ");

lcd.setCursor(4, 0);

lcd.print(h);

lcd.setCursor(9, 0);

lcd.print("%");

lcd.setCursor(0, 1);
```

- Code ที่ใช้ในการทดสอบ Blynk (ต่อ)

```
  lcd.print("Temp:     ");
  lcd.setCursor(5, 1);
  lcd.print(t);
  lcd.setCursor(9, 1);
  lcd.print("C");
  delay(2000);
}
void setup()
{ Serial.begin(115200);
  dht.begin();
  timer.setInterval(1000L, sendSensor);
  Blynk.begin(auth, ssid, pass, "pkdb10.thddns.net", 8334);
  Wire.begin();
  lcd.init();
  lcd.backlight(); }
void loop() {
  Blynk.run();
  timer.run();
}
```

- Code ที่ใช้ในการทดสอบ Google sheets

```
#include <ESP8266WiFi.h>

#include <WiFiClientSecure.h>

#include "DHT.h"

//------------------------------------

//#define DHTTYPE DHT11   // DHT 11

//#define DHTTYPE DHT21   // DHT 21 (AM2301)

#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321

const int DHTPin = D5;

 // String t;

#define ON_Board_LED D4  //--> Defining an On Board LED, used for indicators when the process of connecting to a wifi router

//------------------------------------SSID  dan Password wifi mu gan.

const char* ssid = "XXXX"; //--> Nama Wifi / SSID.

const char* password = "XXXX"; //-->  Password wifi .

//------------------------------------

//------------------------------------Host  & httpsPort

const char* host = "script.google.com";

const int httpsPort = 443;
```

- Code ที่ใช้ในการทดสอบ Google sheets (ต่อ)

```
//----------------------------------

// Initialize DHT sensor.

DHT dht(DHTPin, DHTTYPE,12);

WiFiClientSecure client; //--> Create a WiFiClientSecure object.

// Timers auxiliar variables

long now = millis();

long lastMeasure = 0;

String GAS_ID = "XXXX";  //--> spreadsheet script ID

//============================================  void setup

void setup() {

  // put your setup code here, to run once:

  Serial.begin(115200);

  delay(500);

  dht.begin();

  WiFi.begin(ssid, password); //--> Connect to your WiFi router

  Serial.println("");


  pinMode(ON_Board_LED,OUTPUT); //--> On Board LED port Direction output

  digitalWrite(ON_Board_LED, HIGH); //-->
```

- Code ที่ใช้ในการทดสอบ Google sheets (ต่อ)

```
pinMode(ON_Board_LED,OUTPUT); //--> On Board LED port Direction output
digitalWrite(ON_Board_LED, HIGH); //-->
//----------------------------------------Wait  for connection
Serial.print("Connecting");
while (WiFi.status() != WL_CONNECTED) {
  Serial.print(".");
  //----------------------------------------Make  the On Board Flashing LED on the process of connecting to the wifi  router.
  digitalWrite(ON_Board_LED, LOW);
  delay(250);
  digitalWrite(ON_Board_LED, HIGH);
  delay(250);
  //----------------------------------------
}
//----------------------------------------
digitalWrite(ON_Board_LED, HIGH); //--> Turn off the On Board LED when it is connected to the wifi router.
//----------------------------------------If  successfully connected to the wifi router, the IP Address that will be visited is displayed in the serial monitor
Serial.println("");
```

- Code ที่ใช้ในการทดสอบ Google sheets (ต่อ)

```
  Serial.print("Successfully connected to : ");
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.println();
  //----------------------------------------
  client.setInsecure();
}
//=====================================================================
=========
//=====================================================================
========= void loop
void loop() {
  now = millis();
  // Publishes new temperature and humidity every 3 seconds
  if (now - lastMeasure > 60000) {
    lastMeasure = now;
    // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
    float h = dht.readHumidity();
```

- Code ที่ใช้ในการทดสอบ Google sheets (ต่อ)

```
    // Read temperature as Celsius (the default)
    float t = dht.readTemperature();
    // Check if any reads failed and exit early (to try again).
    if (isnan(h) || isnan(t) ) {
      Serial.println("Failed to read from DHT sensor!");
      return;
    }
    Serial.print("Humidity: ");
    Serial.print(h);
    Serial.print(" %\t Temperature: ");
    Serial.print(t);
    Serial.print(" *C ");
     sendData(t,h);
  }
}
//*****
//=================================================================
=========
```

- Code ที่ใช้ในการทดสอบ Google sheets (ต่อ)

```
void sendData(float value,float value2) {
  Serial.println("==========");
  Serial.print("connecting to ");
  Serial.println(host);
  //----------------------------------Connect  to Google host
  if (!client.connect(host, httpsPort)) {
    Serial.println("connection failed");
    return;
  }
  //----------------------------------
  //----------------------------------Proses  dan kirim  data
  float string_temp = value;
  float string_humi = value2;
  String url = "/macros/s/" + GAS_ID + "/exec?temp=" + string_temp + "&humi="+string_humi; //
2 variables
  Serial.print("requesting URL: ");
  Serial.println(url);
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
         "Host: " + host + "\r\n" +
```

- Code ที่ใช้ในการทดสอบ Google sheets (ต่อ)

```
        "User-Agent: BuildFailureDetectorESP8266\r\n" +
        "Connection: close\r\n\r\n");
Serial.println("request sent");
//----------------------------------
//----------------------------------
while (client.connected()) {
  String line = client.readStringUntil('\n');
  if (line == "\r") {
    Serial.println("headers received");
    break;
  }
}
String line = client.readStringUntil('\n');
if (line.startsWith("{\"state\":\"success\"")) {
  Serial.println("esp8266/Arduino CI successfull!");
} else {
  Serial.println("esp8266/Arduino CI has failed");
}
Serial.print("reply was : ");
```

- Code ที่ใช้ในการทดสอบ Google sheets (ต่อ)

```
  Serial.println(line);

  Serial.println("closing connection");

  Serial.println("==========");

  Serial.println();

  //-----------------------------------

}

//================================================
```

ภาคผนวก ค ผลการเปรียบเทียบ

- ข้อมูลแสดงผลการเปรียบเทียบค่าพารามิเตอร์ที่ได้จากการตรวจวัด

ตารางที่ 3 แสดงผลการเปรียบเทียบค่าพารามิเตอร์ที่วัดได้จาก RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

time	RID-KU001 Weather Station			Mini Real-Time Weather Station		
	Temperature (°C)	Humidity (%)	Wind Speed (m/s)	Temperature (°C)	Humidity (%)	Wind Speed (m/s)
12:00	34.04	64.18	3.22	34.2	68.8	24.67
12:01	34.12	62.96	3.22	34.3	65.5	24
12:02	34.18	62.12	3.62	33.9	66.6	24
12:03	34.1	63.82	2.82	33.9	68.3	26
12:04	34.07	62.87	3.22	34	66.3	12.67
12:05	34.09	61.68	4.83	33.9	65.4	18
12:06	34.08	63.79	2.01	34	66.1	18
12:07	34.11	63.67	2.01	34.3	65.9	23.33
12:08	34.17	63.05	2.82	34.4	70.6	12
12:09	34.24	62.66	4.02	34.4	67.4	23.33
12:10	34.3	64.24	2.01	34.6	67.6	12
12:11	34.41	62.79	1.21	34.8	64.5	11.33
12:12	34.52	62.93	3.62	34.8	65	24
12:13	34.53	60.73	4.43	34.4	66.9	19.33
12:14	34.47	63.11	2.01	34.5	67.8	23.33
12:15	34.41	62.34	1.21	34.6	64.4	6
12:16	34.32	62.73	3.62	34.4	63	18
12:17	34.3	62.45	2.01	34.4	63.4	17.33
12:18	34.23	62.28	4.02	34.3	64.1	22.67
12:19	34.17	62.34	2.82	34.2	64.9	18.67
12:20	34.1	61.92	4.02	34	65.4	26
12:21	34.09	63.97	1.61	34.2	69	0
12:22	34.17	63.47	1.61	34.4	65.3	12
12:23	34.21	62.63	1.61	34.3	66.3	25.33
12:24	34.23	61.89	2.01	34.2	64.2	37.33
12:25	34.22	65.27	0.8	34.4	68.8	18

ตารางที่ 3 แสดงผลการเปรียบเทียบค่าพารามิเตอร์ที่วัดได้จาก RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station) (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

time	RID-KU001 Weather Station			Mini Real-Time Weather Station		
	Temperature (°C)	Humidity (%)	Wind Speed (m/s)	Temperature (°C)	Humidity (%)	Wind Speed (m/s)
12:26	34.22	61.02	2.41	34.2	66.6	6.67
12:27	34.21	62.1	2.01	34.3	64.1	12.67
12:28	34.17	63.23	0.8	34.3	67.6	18
12:29	34.23	65.43	2.41	34.5	65.3	12.67
12:30	34.3	62.43	2.82	34.5	63.7	23.33
12:31	34.3	62.81	2.01	34.7	67.2	24.67
12:32	34.45	63.73	2.82	34.8	67.8	12.67
12:33	34.59	61.29	4.43	35	66.2	23.33
12:34	34.71	61.66	4.02	34.7	63.6	34.67
12:35	34.76	61.89	3.62	34.5	62.4	43.33
12:36	34.74	61	3.22	34.5	64.3	28.67
12:37	34.68	60.19	3.22	34.3	63.6	25.33
12:38	34.65	61.3	3.22	34.4	64.3	26
12:39	34.62	63.35	2.41	34.5	66.7	12
12:40	34.63	60.88	2.41	34.5	63.8	18.67
12:41	34.66	59.57	3.22	34.5	64.5	33.33
12:42	34.64	59.84	3.22	34.4	60.9	13.33
12:43	34.6	59.42	1.61	34.7	63.7	12.67
12:44	34.6	62.43	1.61	35	65.5	12.67
12:45	34.68	62.01	2.01	35.2	62.4	23.33
12:46	34.8	61.83	2.82	35.3	63	32.67
12:47	34.91	61.96	2.82	35.4	63.1	30
12:48	34.99	60.85	3.22	35.1	62.8	23.33
12:49	35	59.72	2.41	34.9	63.2	6.67
12:50	34.93	60.25	4.83	34.8	63	41.33
12:51	34.88	61.45	3.22	34.8	65.2	30
12:52	34.92	60.41	2.82	34.9	60.6	25.33
12:53	34.93	59.93	2.82	34.9	61.6	18.67
12:54	34.91	61.92	3.22	34.9	63	37.33
12:55	34.9	60.14	3.22	34.9	64	23.33

ตารางที่ 3 แสดงผลการเปรียบเทียบค่าพารามิเตอร์ที่วัดได้จาก RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station) (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

time	RID-KU001 Weather Station			Mini Real-Time Weather Station		
	Temperature (°C)	Humidity (%)	Wind Speed (m/s)	Temperature (°C)	Humidity (%)	Wind Speed (m/s)
12:56	34.85	58.59	2.41	34.8	65.9	12.67
12:57	34.88	62.99	3.22	35.3	65.1	18.67
12:58	34.97	62.16	4.43	35.3	62.6	24
12:59	35.06	59.72	2.41	35	62.5	16.67
13:00	35.06	61.44	3.22	35	63.8	17.33
13:01	35.06	60.58	5.23	35	65.1	18
13:02	35.11	60.94	3.62	35.1	66.1	12
13:03	35.16	59.63	3.62	34.9	63.4	11.33
13:04	35.11	61.03	2.01	35.1	65.5	29.33
13:05	35.12	61.96	2.41	35	63.8	17.33
13:06	35.15	60.94	4.02	35.3	64.9	38
13:07	35.21	60.67	5.63	35.4	64.1	24
13:08	35.35	59.63	2.41	35.3	60.7	36
13:09	35.37	59.6	2.82	35.1	62.8	12.67
13:10	35.3	58.7	2.41	35	60.9	18
13:11	35.21	60.53	2.01	34.9	61.9	30
13:12	35.14	59.25	0.8	35	63.8	12
13:13	35.09	59.16	4.02	35	62.6	24
13:14	35.07	62.16	2.01	35.3	63	12
13:15	35.11	60.19	4.43	35.4	60.8	30.67
13:16	35.18	60.08	2.82	35.5	60.5	25.33
13:17	35.21	58.47	2.82	35.2	60.4	18.67
13:18	35.12	58.41	0.8	35	60.7	32
13:19	35.07	59.27	3.22	35	59.5	12.67
13:20	35	62.19	1.21	35.2	64.3	12
13:21	35.04	61.03	3.62	35.3	64.2	18.67
13:22	35.14	61.09	5.23	35.4	61.5	48
13:23	35.21	60.28	2.01	35.4	63.2	30.67
13:24	35.29	60.05	5.23	35.1	59.8	36.67
13:25	35.31	58.56	2.82	35	60.8	11.33

ตารางที่ 3 แสดงผลการเปรียบเทียบค่าพารามิเตอร์ที่วัดได้จาก RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station) (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

time	RID-KU001 Weather Station			Mini Real-Time Weather Station		
	Temperature (°C)	Humidity (%)	Wind Speed (m/s)	Temperature (°C)	Humidity (%)	Wind Speed (m/s)
13:26	35.3	59.22	2.01	35.2	64.3	23.33
13:27	35.31	59.84	6.04	35.2	60.3	18
13:28	35.31	58.23	4.02	35.1	59.6	29.33
13:29	35.31	57.93	3.22	35.2	60	36.67
13:30	35.22	58.62	2.01	35.2	60.4	12
13:31	35.26	60.25	2.82	35.6	60.7	12.67
13:32	35.39	58.26	2.41	35.8	58.6	18.67
13:33	35.45	57.75	2.82	35.8	59.3	25.33
13:34	35.54	59.66	3.22	35.9	59.5	24
13:35	35.63	58.29	3.62	35.5	60.8	20
13:36	35.64	61.69	2.01	35.9	64.2	12
13:37	35.68	60.2	2.01	36.2	60.3	12.67
13:38	35.81	61.45	0.8	36.4	64.2	12
13:39	35.95	58.91	4.83	36.4	59	18.67
13:40	36.04	58.2	2.01	36.3	59.1	12.67
13:41	36.14	58.38	2.82	36.7	60.3	12
13:42	36.27	59.09	2.41	36.5	61.3	13.33
13:43	36.3	58.11	2.41	36.5	58	18.67
13:44	36.27	57.21	3.22	36.3	57.9	12.67
13:45	36.21	56.64	2.41	36.2	58.6	36
13:46	36.16	57.48	6.04	35.9	58.9	46.67
13:47	36.13	56.91	5.63	35.8	57.4	34
13:48	36.07	56.55	4.43	35.8	58.4	33.33
13:49	36	57.22	3.62	36	59	6.67
13:50	35.98	56.91	2.82	36.1	57	32
13:51	36	58.08	2.82	36.3	57.3	32
13:52	36.03	57.39	4.02	36.2	58.7	18
13:53	36.08	57.39	3.62	36.3	55.4	24
13:54	36.13	58.35	2.82	36.5	55.8	11.33
13:55	36.17	57.06	3.22	36.5	59.2	12

ตารางที่ 3 แสดงผลการเปรียบเทียบค่าพารามิเตอร์ที่วัดได้จาก RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station) (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

time	RID-KU001 Weather Station			Mini Real-Time Weather Station		
	Temperature (°C)	Humidity (%)	Wind Speed (m/s)	Temperature (°C)	Humidity (%)	Wind Speed (m/s)
13:56	36.25	55.71	2.41	36.8	55	11.33
13:57	36.36	56.1	2.82	36.9	54.4	51.33
13:58	36.44	54.61	4.02	36.4	54.7	36.67
13:59	36.49	56.19	3.22	36.4	55.6	42.67
14:00	36.55	55.23	5.63	36.4	54.9	32
14:01	36.53	53.47	4.02	36.2	54.4	30.67
14:02	36.55	54.94	6.84	36.6	54	42
14:03	36.57	52.04	6.04	36.3	52.2	58
14:04	36.58	52.43	4.02	36.5	52.9	36
14:05	36.62	51.43	6.04	36.3	52.1	50.67
14:06	36.58	53.71	3.22	36.5	54	18.67
14:07	36.62	52.96	5.63	36.7	52.5	44.67
14:08	36.71	51.84	8.05	36.4	52.2	34
14:09	36.73	52.56	5.63	36.6	52.6	40
14:10	36.71	53.02	4.83	36.3	53.1	23.33
14:11	36.64	53.26	3.62	36.3	53	42.67
14:12	36.58	52.81	3.22	36.4	52.4	28
14:13	36.6	54.22	2.41	36.7	53	22
14:14	36.69	53.88	4.43	36.9	52.6	24
14:15	36.72	52.04	5.63	36.7	51.2	24
14:16	36.7	52.59	5.23	36.5	52.9	22.67
14:17	36.66	51.53	4.83	36.4	53.1	23.33
14:18	36.62	52.25	4.02	36.7	53.2	18
14:19	36.63	53.23	5.23	36.7	52.8	12.67
14:20	36.67	53.17	2.82	36.8	50.7	19.33
14:21	36.69	53.59	2.82	37	54.1	16.67
14:22	36.72	52.63	3.62	37	53	40
14:23	36.72	52.38	3.22	36.9	51.1	13.33
14:24	36.75	54.01	3.22	36.9	54.2	28.67
14:25	36.82	52.31	2.82	37	51.5	6

ตารางที่ 3 แสดงผลการเปรียบเทียบค่าพารามิเตอร์ที่วัดได้จาก RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station) (ต่อ)

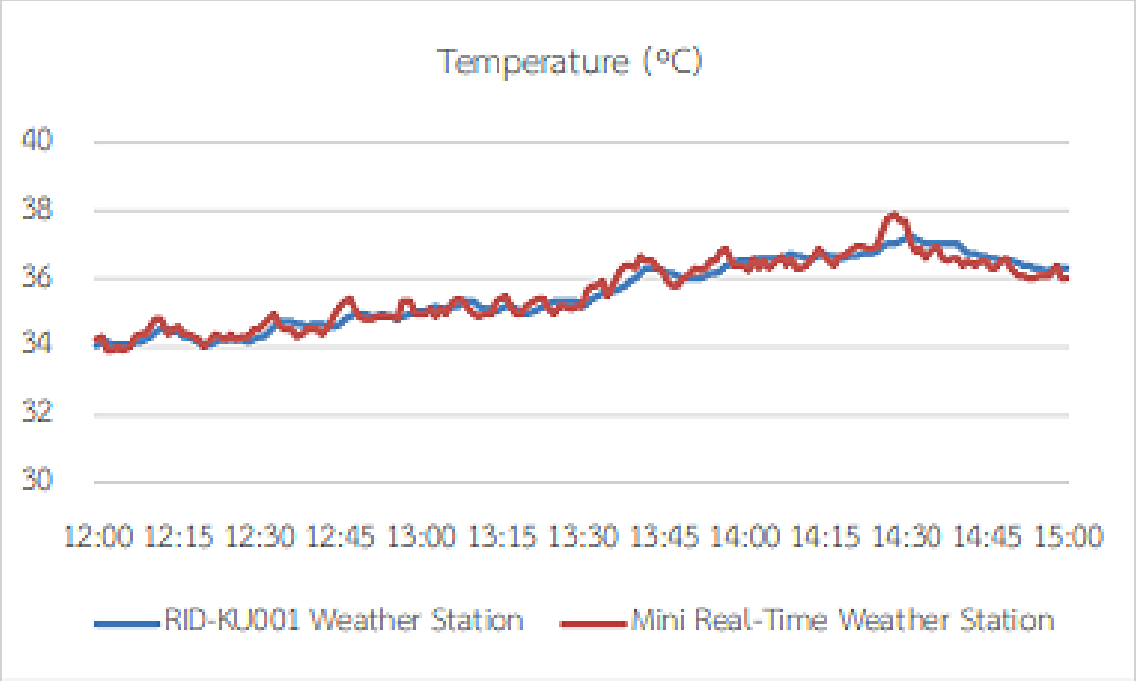
ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

time	RID-KU001 Weather Station			Mini Real-Time Weather Station		
	Temperature (°C)	Humidity (%)	Wind Speed (m/s)	Temperature (°C)	Humidity (%)	Wind Speed (m/s)
14:26	36.97	55.61	0.8	37.4	52	11.33
14:27	37.05	55.77	0.4	37.8	50.3	17.33
14:28	37.05	51.48	2.82	37.9	48.4	24
14:29	37.15	51.54	3.22	37.7	48.9	24
14:30	37.2	52.71	2.01	37.7	52.1	18
14:31	37.28	53.3	5.63	37.1	52	24.67
14:32	37.22	51.95	5.23	36.8	53.5	30.67
14:33	37.13	52.43	4.43	36.9	53.2	18
14:34	37.06	52.57	4.43	36.6	52.7	34
14:35	37.05	52.86	4.02	36.8	54.4	24
14:36	37.05	55.35	5.23	37	53.1	30.67
14:37	37.05	52.07	6.04	36.6	52.8	44
14:38	37.05	53.1	4.83	36.5	52.3	35.33
14:39	37.05	51.75	6.84	36.6	52.4	41.33
14:40	37.05	52.62	4.43	36.6	53.6	36
14:41	36.88	51.93	3.22	36.4	51.8	24.67
14:42	36.78	51.11	2.82	36.5	52.3	36.67
14:43	36.72	52.07	3.22	36.4	51.5	35.33
14:44	36.68	51.46	5.63	36.5	51.4	30
14:45	36.66	51.77	2.82	36.5	51.8	18
14:46	36.62	51.63	4.83	36.3	51.9	22.67
14:47	36.57	52.66	2.82	36.3	52.3	18
14:48	36.52	52.93	1.61	36.5	52.1	24.67
14:49	36.53	52.94	4.43	36.6	51.1	32
14:50	36.51	51.64	5.23	36.3	52	12
14:51	36.44	53.14	2.82	36.1	53.9	29.33
14:52	36.39	52.85	4.43	36.1	52.6	12
14:53	36.36	53.11	5.63	36	51.5	34
14:54	36.31	52.94	1.61	36	52.5	16.67
14:55	36.27	53.2	3.22	36.1	52	12

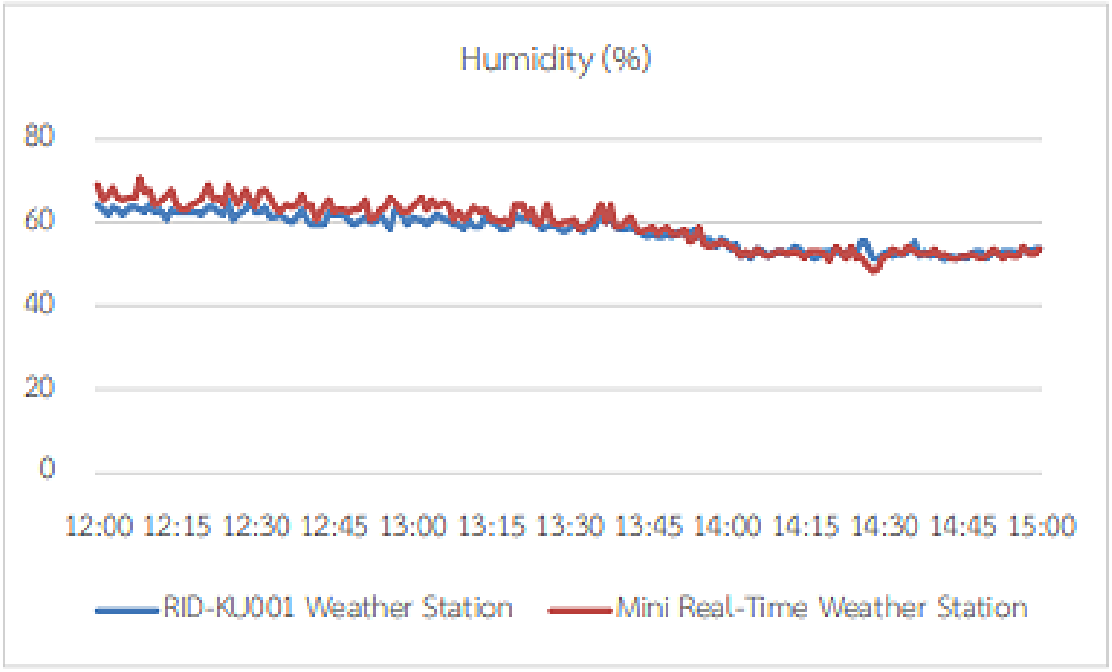
ตารางที่ 3 แสดงผลการเปรียบเทียบค่าพารามิเตอร์ที่วัดได้จาก RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station) (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

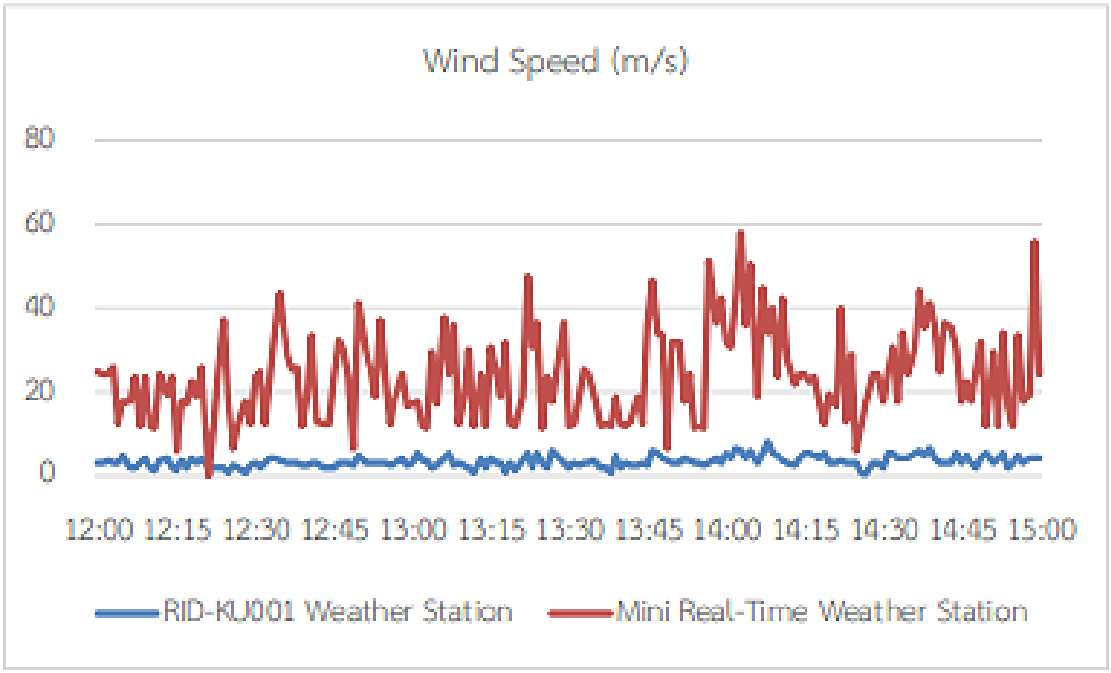
time	RID-KU001 Weather Station			Mini Real-Time Weather Station		
	Temperature (°C)	Humidity (%)	Wind Speed (m/s)	Temperature (°C)	Humidity (%)	Wind Speed (m/s)
14:56	36.26	52.44	4.83	36.1	51.9	33.33
14:57	36.26	53.98	2.82	36.1	54.2	18
14:58	36.28	53.36	4.02	36.4	52.6	19.33
14:59	36.32	53.65	4.43	36	52.8	56
15:00	36.3	53.59	4.02	36	53.9	24



กราฟที่ 1 แสดงการเปรียบเทียบค่าอุณหภูมิระหว่างสถานี RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station)



กราฟที่ 2 แสดงการเปรียบเทียบค่าความชื้นสัมพัทธ์ระหว่างสถานี RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station)



กราฟที่ 3 แสดงการเปรียบเทียบค่าความเร็วลมระหว่างสถานี RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time (Mini Real-Time Weather Station)

หมายเหตุ: เนื่องจากค่าความเร็วลมที่วัดได้จากสถานีวัดอากาศขนาดย่อมแบบ Real-Time มีค่าที่คลาดเคลื่อนและแปรปรวนสูง จึงต้องทำการปรับเทียบหาค่าแฟกเตอร์ ดังภาคผนวก ง.

ภาคผนวก ง ผลการปรับเทียบ

- ปรับเทียบหาค่าแฟกเตอร์ความเร็วลมของสถานีวัดอากาศขนาดย่อมแบบ Real-Time

โดยการใช้ข้อมูลจาก RID-KU001 Weather Station เป็นค่าที่ใช้ในการหาแฟกเตอร์ ใช้ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. ดังแสดงในตารางที่ 4 ได้ผลการทดสอบ ดังนี้

- ค่าเฉลี่ยความเร็วลมของสถานีวัดอากาศขนาดย่อมแบบ Real-Time = 23.706 m/s
- ค่าเฉลี่ยความเร็วลมของ RID-KU001 Weather Station = 3.363 m/s
- ได้ค่าแฟกเตอร์ความเร็วลม = 0.14188 m/s

ตารางที่ 4 แสดงผลค่าความเร็วลมที่ได้จากการปรับเทียบด้วยค่าแฟกเตอร์ ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

Time	Wind Speed RID-KU001 (m/s)	Wind Speed Mini Real-Time (m/s)
12:00	3.22	24.67
12:01	3.22	24.00
12:02	3.62	24.00
12:03	2.82	26.00
12:04	3.22	12.67
12:05	4.83	18.00
12:06	2.01	18.00
12:07	2.01	23.33
12:08	2.82	12.00
12:09	4.02	23.33
12:10	2.01	12.00
12:11	1.21	11.33
12:12	3.62	24.00
12:13	4.43	19.33
12:14	2.01	23.33
12:15	1.21	6.00
12:16	3.62	18.00
12:17	2.01	17.33
12:18	4.02	22.67

ตารางที่ 4 แสดงผลค่าความเร็วลมที่ได้จากการปรับเทียบด้วยค่าแฟกเตอร์ (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

Time	Wind Speed RID-KU001 (m/s)	Wind Speed Mini Real-Time (m/s)
12:19	2.82	18.67
12:20	4.02	26.00
12:21	1.61	0.00
12:22	1.61	12.00
12:23	1.61	25.33
12:24	2.01	37.33
12:25	0.80	18.00
12:26	2.41	6.67
12:27	2.01	12.67
12:28	0.80	18.00
12:29	2.41	12.67
12:30	2.82	23.33
12:31	2.01	24.67
12:32	2.82	12.67
12:33	4.43	23.33
12:34	4.02	34.67
12:35	3.62	43.33
12:36	3.22	28.67
12:37	3.22	25.33
12:38	3.22	26.00
12:39	2.41	12.00
12:40	2.41	18.67
12:41	3.22	33.33
12:42	3.22	13.33
12:43	1.61	12.67
12:44	1.61	12.67
12:45	2.01	23.33
12:46	2.82	32.67
12:47	2.82	30.00
12:48	3.22	23.33
12:49	2.41	6.67
12:50	4.83	41.33

ตารางที่ 4 แสดงผลค่าความเร็วลมที่ได้จากการปรับเทียบด้วยค่าแฟกเตอร์ (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

Time	Wind Speed RID-KU001 (m/s)	Wind Speed Mini Real-Time (m/s)
12:51	3.22	30.00
12:52	2.82	25.33
12:53	2.82	18.67
12:54	3.22	37.33
12:55	3.22	23.33
12:56	2.41	12.67
12:57	3.22	18.67
12:58	4.43	24.00
12:59	2.41	16.67
13:00	3.22	17.33
13:01	5.23	18.00
13:02	3.62	12.00
13:03	3.62	11.33
13:04	2.01	29.33
13:05	2.41	17.33
13:06	4.02	38.00
13:07	5.63	24.00
13:08	2.41	36.00
13:09	2.82	12.67
13:10	2.41	18.00
13:11	2.01	30.00
13:12	0.80	12.00
13:13	4.02	24.00
13:14	2.01	12.00
13:15	4.43	30.67
13:16	2.82	25.33
13:17	2.82	18.67
13:18	0.80	32.00
13:19	3.22	12.67
13:20	1.21	12.00
13:21	3.62	18.67
13:22	5.23	48.00

ตารางที่ 4 แสดงผลค่าความเร็วลมที่ได้จากการปรับเทียบด้วยค่าแฟกเตอร์ (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

Time	Wind Speed RID-KU001 (m/s)	Wind Speed Mini Real-Time (m/s)
13:23	2.01	30.67
13:24	5.23	36.67
13:25	2.82	11.33
13:26	2.01	23.33
13:27	6.04	18.00
13:28	4.02	29.33
13:29	3.22	36.67
13:30	2.01	12.00
13:31	2.82	12.67
13:32	2.41	18.67
13:33	2.82	25.33
13:34	3.22	24.00
13:35	3.62	20.00
13:36	2.01	12.00
13:37	2.01	12.67
13:38	0.80	12.00
13:39	4.83	18.67
13:40	2.01	12.67
13:41	2.82	12.00
13:42	2.41	13.33
13:43	2.41	18.67
13:44	3.22	12.67
13:45	2.41	36.00
13:46	6.04	46.67
13:47	5.63	34.00
13:48	4.43	33.33
13:49	3.62	6.67
13:50	2.82	32.00
13:51	2.82	32.00
13:52	4.02	18.00
13:53	3.62	24.00
13:54	2.82	11.33

ตารางที่ 4 แสดงผลค่าความเร็วลมที่ได้จากการปรับเทียบด้วยค่าแฟกเตอร์ (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

Time	Wind Speed RID-KU001 (m/s)	Wind Speed Mini Real-Time (m/s)
13:55	3.22	12.00
13:56	2.41	11.33
13:57	2.82	51.33
13:58	4.02	36.67
13:59	3.22	42.67
14:00	5.63	32.00
14:01	4.02	30.67
14:02	6.84	42.00
14:03	6.04	58.00
14:04	4.02	36.00
14:05	6.04	50.67
14:06	3.22	18.67
14:07	5.63	44.67
14:08	8.05	34.00
14:09	5.63	40.00
14:10	4.83	23.33
14:11	3.62	42.67
14:12	3.22	28.00
14:13	2.41	22.00
14:14	4.43	24.00
14:15	5.63	24.00
14:16	5.23	22.67
14:17	4.83	23.33
14:18	4.02	18.00
14:19	5.23	12.67
14:20	2.82	19.33
14:21	2.82	16.67
14:22	3.62	40.00
14:23	3.22	13.33
14:24	3.22	28.67
14:25	2.82	6.00
14:26	0.80	11.33

ตารางที่ 4 แสดงผลค่าความเร็วลมที่ได้จากการปรับเทียบด้วยค่าแฟกเตอร์ (ต่อ)

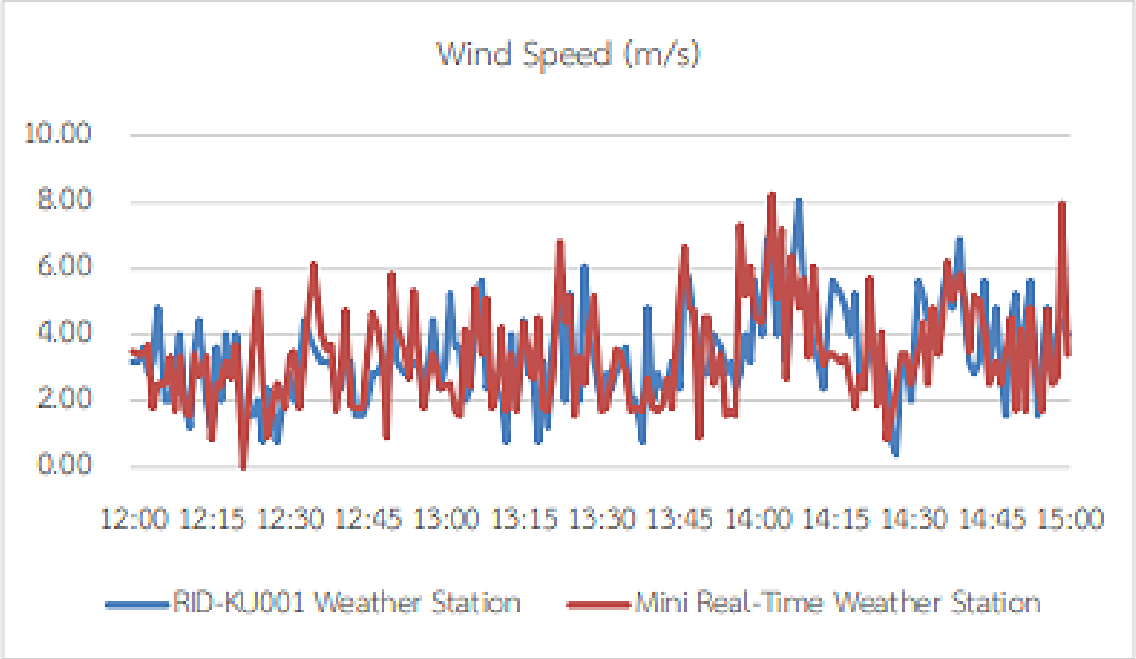
ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

Time	Wind Speed RID-KU001 (m/s)	Wind Speed Mini Real-Time (m/s)
14:27	0.40	17.33
14:28	2.82	24.00
14:29	3.22	24.00
14:30	2.01	18.00
14:31	5.63	24.67
14:32	5.23	30.67
14:33	4.43	18.00
14:34	4.43	34.00
14:35	4.02	24.00
14:36	5.23	30.67
14:37	6.04	44.00
14:38	4.83	35.33
14:39	6.84	41.33
14:40	4.43	36.00
14:41	3.22	24.67
14:42	2.82	36.67
14:43	3.22	35.33
14:44	5.63	30.00
14:45	2.82	18.00
14:46	4.83	22.67
14:47	2.82	18.00
14:48	1.61	24.67
14:49	4.43	32.00
14:50	5.23	12.00
14:51	2.82	29.33
14:52	4.43	12.00
14:53	5.63	34.00
14:54	1.61	16.67
14:55	3.22	12.00
14:56	4.83	33.33
14:57	2.82	18.00
14:58	4.02	19.33

ตารางที่ 4 แสดงผลค่าความเร็วลมที่ได้จากการปรับเทียบด้วยค่าแฟกเตอร์ (ต่อ)

ข้อมูล ณ วันที่ 24 พฤษภาคม 2564 เวลา 12.00 – 15.00 น. (ราย 1 นาที)

Time	Wind Speed RID-KU001 (m/s)	Wind Speed Mini Real-Time (m/s)
14:59	4.43	56.00
15:00	4.02	24.00
12:00	3.22	24.67



กราฟที่ 4 แสดงการเปรียบเทียบค่าความเร็วลมระหว่างสถานี RID-KU001 Weather Station และ สถานีวัดอากาศขนาดย่อมแบบ Real-Time ที่ปรับเทียบด้วยค่าแฟกเตอร์