ระบบตรวจวัดการใช้ปริมาณไฟฟ้าด้วยดิจิทัลมิเตอร์ผ่าน NETPIE

ELECTRIC QUANTITIES MEASUREMENT SYSTEM

WITH DIGITAL METER VIA NETPIE

นายชุติพงศ์ ชัยวิชิตชลกุล รหัส 57362958

นายพีรพล มีแก้ว รหัส 57363283

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมไฟฟ้า ภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร

ปีการศึกษา2560

ใบรับรองปริญญานิพนธ์

ชื่อหัวข้อโครงงาน ระบบตรวจวัดการใช้ปริมาณไฟฟ้าด้วยดิจิตอลมิเตอร์ผ่าน NETPIE

ผู้ดำเนินโครงงาน นายชุติพงศ์ ชัยวิชิตชลกุล รหัส 57362958

นายพีรพล มีแก้ว รหัส 57363283

ที่ปรึกษาโครงงาน ผู้ช่วยศาสตราจารย์ ดร.ปิยดนัย ภาชนะพรรณ์

สาขาวิชา วิศวกรรมไฟฟ้า

ภาควิชา วิศวกรรมไฟฟ้าและคอมพิวเตอร์

ปีการศึกษา 2560

.....

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยนเรศวร อนุมัติให้ปริญญานิพนธ์ฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมไฟฟ้า

.....ที่ปรึกษาโครงงาน

(ผู้ช่วยศาสตราจารย์ ดร.ปิยดนัย ภาชนะพรรณ์)

.....กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.พนัส นัถฤทธิ์)

.....กรรมการ

(ดร.จิรวดี ผลประเสริฐ)

ชื่อหัวข้อโครงงาน	ระบบตรวจวัดการใช้ปริมาณไฟฟ้าด้วยดิจิตอลมิเตอร์ผ่าน NETPIE
ผู้ดำเนินโครงงาน	นายชุติพงศ์ ชัยวิชิตชลกุล รหัส 57362958
	นายพีรพล มีแก้ว รหัส 57363283
ที่ปรึกษาโครงงาน	ผู้ช่วยศาสตราจารย์ ดร.ปิยดนัย ภาชนะพรรณ์
สาขาวิชา	วิศวกรรมไฟฟ้า
ภาควิชา	วิศวกรรมไฟฟ้าและคอมพิวเตอร์
ปีการศึกษา	2560

.....

บทคัดย่อ

โครงงานนี้ได้ทำการศึกษาและพัฒนาระบบตรวจวัดการใช้ปริมาณไฟฟ้าด้วยดิจิตอลมิเตอร์ 1 เฟส ผ่าน NETPIE โดยที่ระบบประกอบไปด้วยดิจิตอลมิเตอร์ 1 เฟส จะส่งค่าปริมาณทางไฟฟ้า ได้แก่ ค่าแรงดันไฟฟ้า, กระแสไฟฟ้า, กำลังไฟฟ้า, ประกอบกำลังไฟฟ้า และพลังงานไฟฟ้าผ่านมาตรฐานการสื่อสาร RS485 ให้กับไมโครคอนโทรลเลอร์ ซึ่งตัวไมโครคอนโทรลเลอร์ที่ใช้คือแผงวงจรอาดูโน่ ชนิด ATMega 2560 R3 เมื่อรับค่าปริมาณทางไฟฟ้าจากมิเตอร์เข้ามาแล้วจะทำการประมวลผล โดยที่ค่าปริมาณทางไฟฟ้านั้นจะถูกส่งไปจัดเก็บในการ์ดหน่วยความจำ (SD Card) ของมอดูลเก็บบันทึกข้อมูล (Data Logger Shield) ทุก ๆ 15 วินาที พร้อมทั้ง ส่งค่าผ่านระบบสัญญาณไวไฟ (WI-FI) โดยใช้มอดูลสื่อสารไร้สาย (NodeMCU ESP8266) ไปแสดงผลออกทางหน้าเว็บ NETPIE รวมถึงจัดเก็บไว้ที่ NETPIE ทุก ๆ 10 นาที

Project title	Electric Quantities Measurement System With Digital Meter Via NETPIE
Name	Mr.Chutipong Chaiwichitchonkoon ID.57362958
	Mr.Pheeraphon Meekaew ID.57363283
Project advisor	Asst. Prof. Dr. Piyadanai Pachanapan
Major	Electrical Engineering
Department	Electrical and Computer Engineering
Academic year	2017

.....

Abstract

A power measurement based on NETPIE is developed in this Project. The system consists of digital meter 1 phase which will sent the amount of electric quantities, such as voltage, current, power, power factor and total active energy to a microcontroller via is the RS485 communication standard. In addition the which the microcontroller via is the ATMega 2560 R3 circuit board. The microcontroller will record the amount of electric quantities received from the digital meter, on the memory card (SD card) in every 15 seconds. As the same time, the microcontroller will transmitted the same measured data to the NETPIE cloud platform, by using the wireless communication module (nodeMCU ESP8266) through the Wi-Fi system. The measured data on the NETPIE wabpage will be update in every 10 minutes.

กิตติกรรมประกาศ

โครงงานนี้สำเร็จลุล่วงไปได้ด้วยดีจาก ผู้ช่วยศาสตราจารย์ ดร.ปิยดนัย ภาชนะพรรณ์ อาจารย์ที่ปรึกษาโครงงาน ซึ่งเป็นผู้ที่เอาใจใส่ในทุกรายละเอียดของการดำเนินโครงงาน โดยให้ทั้งความรู้และคำแนะนำในการแก้ไขปัญหาต่างๆอย่างต่อเนื่อง รวมถึงแนะนำการเขียนปริญญานิพนธ์ การใช้ภาษา และการตรวจทานแก้ไขอย่างละเอียดจนได้ปริญญานิพนธ์เป็นรูปเล่มสมบูรณ์ ผู้ดำเนินโครงงานจึงขอขอบพระคุณเป็นอย่างสูง

ขอขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดร.พนัส นัถฤทธิ์ และ ดร.จิรวดี ผลประเสริฐ เป็นคณะกรรมการในการสอบโครงงานที่ให้คำแนะนำชี้แนะ แนวทาง และข้อคิดเห็นต่าง ๆ รวมถึงการตรวจรูปเล่มปริญญานิพนธ์ที่เป็นประโยชน์ในโครงงานนี้ ให้โครงงานนี้สมบูรณ์ยิ่งขึ้น

ขอขอบคุณเพื่อนๆ สาขาวิศวกรรมคอมพิวเตอร์ และขอขอบคุณภาควิชาวิศวกรรมไฟฟ้าและคอมพิวเตอร์ที่ให้ยืมเครื่องวัดในการทดสอบชิ้นงานที่สร้างขึ้น รวมถึงอำนวยความสะดวกในการใช้ห้องปฏิบัติการจนกระทั่งการทดสอบต่าง ๆ สิ้นสุดลง

ในท้ายที่สุดนี้ เหนือสิ่งอื่นใด ผู้ดำเนินโครงงานขอกราบขอบพระคุณบิดาและมารดา ซึ่งให้การสนับสนุนในทุกด้านเกี่ยวกับการศึกษาของผู้ดำเนินโครงงาน รวมทั้งมอบความเมตตา และคอยเป็นกำลังใจให้จนประสบความสำเร็จในวันนี้

ผู้ดำเนินโครงงาน
นายชุติพงศ์ ชัยวิชิตชลกุล
นายพีรพล มีแก้ว

มิถุนายน 2561

สารบัญ

หน้า

ใบรับรองปริญญานิพนธ์ ..... ก
บทคัดย่อภาษาไทย ..... ข
บทคัดย่อภาษาอังกฤษ ..... ค
กิตติกรรมประกาศ ..... ง
สารบัญ ..... จ
สารบัญตาราง ..... ซ
สารบัญรูป ..... ฌ

บทที่ 1 บทนำ 1
1.1 ความเป็นมาและความสำคัญของโครงงาน 1
1.2 วัตถุประสงค์ของโครงงาน 2
1.3 ประโยชน์ที่คาดว่าจะได้รับ 2
1.4 ขอบเขตการทำโครงงาน 2
1.5 ขั้นตอนการดำเนินงาน 3
1.6 แผนการดำเนินงาน 3
1.7 ผลที่คาดว่าจะได้รับ 4
1.8 รายละเอียดงบประมาณตลอดโครงงาน 5

บทที่ 2 ทฤษฎีหลักการและเอกสารที่เกี่ยวข้อง 6
2.1 มิเตอร์ไฟฟ้า 6
2.2 มาตรฐาน RS-485 7
2.3 มอดูลแปลงสัญญาณ 9
2.4 การติดต่อสื่อสารแบบ Master/Slave 9

สารบัญ (ต่อ)

หน้า

2.5 ไมโครคอนโทรลเลอร์ 10
2.6 มอดูลเก็บบันทึกข้อมูล 12
2.7 มอดูลสื่อสารไร้สาย 13
2.8 NETPIE 15

บทที่ 3 ขั้นตอนการดำเนินการ 21
3.1 ขั้นตอนการทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้า 21
3.2 โครงสร้างการทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้าทั้งหมด 24
3.3 กระบวนการทดลอง 40
3.4 การทำงานของโปรแกรม 42

บทที่ 4 ผลการทดลอง 44
4.1 การแสดงผลรูปกราฟผ่านทาง NETPIE 44
4.2 ผลการทดลองการเก็บข้อมูล 47

บทที่ 5 สรุปผลและข้อเสนอแนะ 60
5.1 สรุปผลการทดลอง 60
5.2 ประเมินผล 61
5.3 ปัญหา ข้อเสนอแนะ และแนวทางแก้ไข 61
5.4 แนวทางในการพัฒนาต่อไป 61

เอกสารอ้างอิง 62

สารบัญ (ต่อ)

หน้า

ภาคผนวก ก รหัสของโปรแกรมในการแสดงผลค่าปริมาณทางไฟฟ้า 63

ภาคผนวก ข เอกสารข้อมูล SDM120 Modbus 80

ภาคผนวก ค เอกสารข้อมูล ESP8266 nodeMCU 88

ภาคผนวก ง เอกสารข้อมูล RS-485 91

ประวัติผู้ดำเนินโครงงาน 96

สารบัญตาราง

ตารางที่ หน้า

3.1 โครงสร้างรหัสโปรโตคอลส่วนของการส่งข้อมูลปริมาณทางไฟฟ้า 26

3.2 รหัสโปรโตคอลมิเตอร์ส่วนของมิเตอร์ส่งข้อมูลแรงดันไฟฟ้าตอบกลับมา 27

สารบัญรูป

รูปที่ หน้า

1.1 รูปวงจรในระบบ 2
2.1 มิเตอร์ไฟฟ้าแบบดิจิทัล รุ่น SDM-120 7
2.2 โครงสร้างมิเตอร์ไฟฟ้าแบบดิจิทัล รุ่น SDM-120 7
2.3 เปรียบเทียบมาตรฐาน RS232 RS422 และ RS485 8
2.4 มอดูลแปลงสัญญาณ (RS 485) 9
2.5 การติดต่อสื่อสารแบบ Master/Slave 10
2.6 ไมโครคอนโทรลเลอร์อาดูโน่ Mega 2560 11
2.7 แผงวงจรมอดูลเก็บบันทึกข้อมูล 13
2.8 แผงวงจร ESP8266 14
2.9 เครือข่าย NETPIE 15
2.10 เว็บแอพพลิเคชั่นควบคุมการเปิด / ปิดไฟ 16
2.11 ระบบควบคุมอุณหภูมิในห้องตรวจสอบคุณภาพสินค้า 17
2.12 กล่องระบบ Location Mapping ติดตั้งไว้ที่ชั้นวางสินค้าในคลัง 17
2.13 การแสดงผลผ่านหน้าการจัดการของ Feed 18
2.14 รูปแสดงการเลือกประเภทของ Widget 19
2.15 การแสดงผลผ่านหน้าการจัดการของ Freeboard 20
3.1 ขั้นตอนการทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้า 21
3.2 การเชื่อมต่อวงจรแสดงค่าปริมาณทางไฟฟ้า 22
3.3 แผงวงจรเมื่อต่อจริง 23
3.4 โครงสร้างการทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้าทั้งหมด 24
3.5 ตารางที่อยู่ข้อมูลมอดบัส โปรโตคอล ในโหมด RTU ของมิเตอร์รุ่น SDM-120 25
3.6 โปรแกรมคำนวณ CRC ของการส่งโปรโตคอลปริมาณทางไฟฟ้า 26
3.7 รหัสโปรโตคอลมิเตอร์ส่วนของมิเตอร์ส่งข้อมูลปริมาณไฟฟ้าและตอบกลับ 27

สารบัญรูป (ต่อ)

รูปที่ หน้า

3.8 ค่าแรงดันไฟฟ้าที่นำมาแปลงเป็นเลขฐานสิบ 28
3.9 พอร์ตสื่อสารระหว่างแผงวงอาดูโน่กับติจิตอลมิเตอร์ 29
3.10 การต่อระหว่างมอดูลบันทึกข้อมูลเข้ากับแผงวงจรอาดูโน่ 30
3.11 การต่อระหว่างมอดูลสื่อสารไร้สาย กับ NETPIE 31
3.12 รูปแสดงการเข้าใช้งาน Feed จากเมนู Resources 31
3.13 รูปแสดงการสร้างหรือเพิ่ม Feed ใหม่ 32
3.14 รูปแสดงการตั้งชื่อ Feed 32
3.15 รูปแสดงหน้าตั้งค่า Feed 32
3.16 รูปแสดงตัวอย่าง Feed ที่สร้างขึ้นด้วย 5 Field 33
3.17 รูปแสดงการให้สิทธิ์การเข้าถึง Feed กับ AppID 34
3.18 นำ App ID ไปใช้งานร่วมกับมอดูลสื่อสารไร้สาย 34
3.19 แสดงหน้าต่าง Serial Monitor 35
3.20 รูปแสดงการเข้าใช้งาน Freeboard จากเมนู Resources 36
3.21 รูปแสดงการเพิ่ม Datasource 36
3.22 แสดงการเลือกประเภทของ Datasource 37
3.23 แสดงการระบุข้อมูลของ Datasource 38
3.24 แสดงการเพิ่ม Panel สำหรับสร้าง Widget 38
3.25 แสดงการเลือกประเภทของ Widget 39
3.26 แสดงหน้ากระดานข้อมูลของ NETPIE Freeboard 39
3.27 การนำโหลดความต้านทานต่อกับระบบตรวจวัดการใช้พลังงานไฟฟ้าด้วยดิจิทัลมิเตอร์ 40
3.28 การนำโหลดหลอดไฟและโหลดตัวเหนี่ยวนำต่อกับระบบตรวจวัดการใช้ปริมาณไฟฟ้า 41
3.29 การนำโหลดหลอดไฟและโหลดตัวเก็บประจุต่อกับระบบตรวจวัดการใช้ปริมาณไฟฟ้า 41
3.30 ภายในระบบตรวจวัดการใช้พลังงานไฟฟ้าด้วยดิจิทัลมิเตอร์ 42

สารบัญรูป (ต่อ)

รูปที่ หน้า

3.31 แผนภาพแสดงภาพรวมการทำงานของโปรแกรม 43

4.1 หน้าจอของดิจิตอลมิเตอร์ 44

4.2 ค่าปริมาณทางไฟฟ้าที่แผงวงจรอาดูโน่ได้รับจากดิจิตอลมิเตอร์ 44

4.3 ค่าปริมาณทางไฟฟ้าที่มอดูลสื่อสารไร้สายได้รับจากแผงวงจรอาดูโน่ 45

4.4 ผลการแสดงค่าปริมาณทางไฟฟ้าเป็นรูปกราฟเส้นของ NETPIE 46

4.5 ผลการแสดงค่าปริมาณทางไฟฟ้าเป็นกระดานข้อมูลของ NETPIE 46

4.6 แสดงไฟล์ในหน่วยความจำ 47

4.7 แสดงข้อมูลที่บันทึกในไฟล์วันที่ 20 พฤษภาคม 2561 (เริ่ม เวลา 0.00 น.) 48

4.8 แสดงข้อมูลกราฟค่าแรงดันไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561 50

4.9 แสดงข้อมูลกราฟค่าแรงดันไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561 50

4.10 แสดงข้อมูลกราฟค่ากำลังไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561 51

4.11 แสดงข้อมูลกราฟค่ากำลังไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561 51

4.12 แสดงข้อมูลกราฟค่ากระแสไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561 52

4.13 แสดงข้อมูลกราฟค่ากระแสไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561 52

4.14 แสดงข้อมูลกราฟค่าประกอบกำลังไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561 53

4.15 แสดงข้อมูลกราฟค่าประกอบกำลังไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561 53

4.16 แสดงข้อมูลกราฟค่าพลังงานไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561 54

4.17 แสดงข้อมูลกราฟค่าพลังงานไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561 54

4.18 แสดงข้อมูลกราฟค่าแรงดันไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561 55

4.19 แสดงข้อมูลกราฟค่าแรงดันไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561 55

4.20 แสดงข้อมูลกราฟค่ากำลังไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561 56

4.21 แสดงข้อมูลกราฟค่ากำลังไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561 56

4.22 แสดงข้อมูลกราฟค่ากระแสไฟฟ้าที่ที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561 57

สารบัญรูป (ต่อ)

รูปที่ หน้า

4.23 แสดงข้อมูลกราฟค่ากระแสไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561 57

4.24 แสดงข้อมูลกราฟค่าประกอบกำลังไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561 ... 58

4.25 แสดงข้อมูลกราฟค่าประกอบกำลังไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561 58

4.26 แสดงข้อมูลกราฟค่าพลังงานไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561 59

4.27 แสดงข้อมูลกราฟค่าพลังงานไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561 59

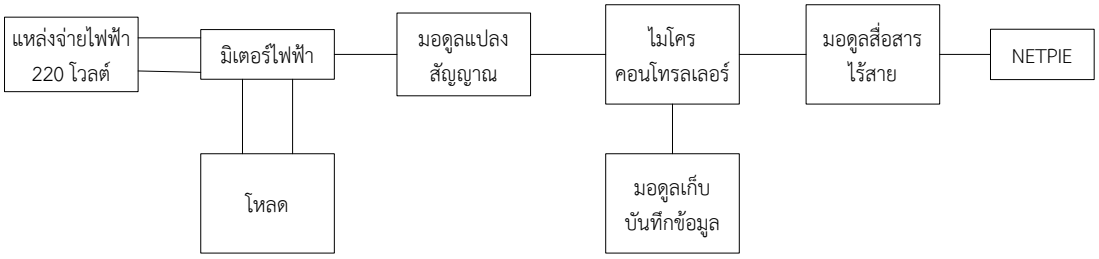
บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของโครงงาน

ในการดำรงชีวิตในแต่ละวัน ทุกคนจำเป็นต้องใช้เครื่องใช้ไฟฟ้า อุปกรณ์ไอที เทคโนโลยีต่าง ๆ หรืออุปกรณ์ที่เกี่ยวข้องทางไฟฟ้า เช่น โทรทัศน์ ตู้เย็น คอมพิวเตอร์ โทรศัพท์เคลื่อนที่ อื่น ๆ อีกมากมาย เพื่อความสะดวกในการดำรงชีวิต ซึ่งไฟฟ้าที่ทุกคนใช้กันทุกวันนั้นเป็นสิ่งสำคัญในการช่วยพัฒนาคุณภาพชีวิต รวมถึงเศรษฐกิจทั้งในและต่างประเทศ ในการตรวจสอบการใช้ปริมาณไฟฟ้าของเครื่องใช้ไฟฟ้า อุปกรณ์ไอที เทคโนโลยีต่าง ๆ หรืออุปกรณ์ที่เกี่ยวข้องทางไฟฟ้านั้น จำเป็นต้องใช้เครื่องมือวัดปริมาณทางไฟฟ้ามาวัดค่า นั่นคือ มิเตอร์ไฟฟ้า (Kilowatt-Hour Meter)

ในปัจจุบันมิเตอร์ไฟฟ้า มีอยู่ 2 ประเภท คือ อนาล็อก และดิจิทัล โดยสามารถแบ่งตามระบบไฟฟ้าได้อีก 2 ประเภท คือ 1 เฟส และ 3 เฟส ซึ่งมิเตอร์ไฟฟ้านี้เป็นเครื่องมือวัดปริมาณไฟฟ้าที่แสดงการใช้ไฟฟ้าให้แก่ผู้ใช้ได้ทราบถึงปริมาณการใช้ไฟฟ้าทั้งหมดจำนวนกี่หน่วย แต่ปัจจุบันนั้นมิเตอร์ไฟฟ้าแบบจานหมุนไม่สามารถคำนวณค่าไฟฟ้าและแสดงหน่วยที่ใช้ในแต่ละช่วงเวลาที่เราต้องการดูค่าไฟฟ้าได้ ในการที่จะคิดค่าไฟฟ้าจะต้องทำการอ่านด้วยสายตาเพื่อทำการจดบันทึก แล้วนำค่าที่ได้ไปคำนวณจำนวนการใช้ รวมถึงค่าใช้จ่าย

จากปัญหาข้างต้น โครงงานนี้จึงได้มีการพัฒนานำมิเตอร์ไฟฟ้าแบบดิจิทัลมาใช้เพื่อให้สามารถส่งผ่านข้อมูลผ่านระบบสัญญาณไวไฟ (WI-FI) ได้ เนื่องจากมิเตอร์ไฟฟ้าแบบดิจิทัลสามารถส่งสัญญาณข้อมูลปริมาณทางไฟฟ้าต่าง ๆ เช่น แรงดัน ,กระแส, กำลังไฟฟ้า, ค่าประกอบกำลัง และค่าพลังงานไฟฟ้า ผ่านมาตรฐาน RS-485 ไปที่แผงวงจรอาดูโน่ ให้ทำการจัดเก็บข้อมูลลงในการ์ดหน่วยความจำ และทำการอัพโหลดส่งข้อมูลผ่านระบบสัญญาณไวไฟโดยใช้ไวไฟโมดูล (WI-FI Module) สู่ NETPIE ได้ โดยจะทำให้เราสามารถเข้าไปตรวจสอบปริมาณค่าไฟฟ้าได้ตลอดเวลา ซึ่งจะทำให้ข้อผิดพลาดในการจดบันทึกลดลง รวมถึงประหยัดเวลาในการจดบันทึกหน้ามิเตอร์ไฟฟ้า อีกทั้งยังสามารถนำข้อมูลที่ได้ไปวางแผนในการช่วยอนุรักษ์หรือประหยัดพลังงานในแต่ละวันได้อีกด้วย



รูปที่ 1.1 รูปวงจรในระบบ

1.2 วัตถุประสงค์ของโครงงาน

1. ประยุกต์การใช้ NETPIE เพื่อแสดงค่าปริมาณการใช้ไฟฟ้า ได้แก่ ค่าแรงดัน ,ค่ากระแส, ค่ากำลังไฟฟ้า, ค่าประกอบกำลัง และค่าพลังงานไฟฟ้าได้
2. ใช้คอนโทรลเลอร์แปลงค่าเพื่อการจัดเก็บค่าปริมาณทางไฟฟ้าจากดิจิทัลมิเตอร์ได้

1.3 ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถนำเอาอุปกรณ์มิเตอร์ไฟฟ้าแบบดิจิทัลที่แสดงค่าปริมาณทางไฟฟ้าไปทำการแสดงผลที่ NETPIE ได้
2. สามารถนำแผงวงจรอาดูโน่ ประยุกต์ใช้เกี่ยวข้องกับมิเตอร์ไฟฟ้าแบบดิจิทัล รวมถึงพัฒนามิเตอร์ไฟฟ้าแบบดิจิทัลให้เป็นอุปกรณ์อิเล็กทรอนิกส์สามารถสื่อสารพูดคุยกันเองได้
3. สามารถเขียนโปรแกรมสำหรับแผงวงจรอาดูโน่ เพื่อหาค่าปริมาณทางไฟฟ้า และจัดเก็บค่าที่วัดได้
4. สามารถนำข้อมูลที่ได้จากระบบตรวจวัดการใช้งานพลังงานไฟฟ้าไปช่วยในการวางแผนการอนุรักษ์หรือประหยัดพลังงาน

1.4 ขอบเขตการทำโครงงาน

พัฒนาอุปกรณ์ระบบตรวจวัดการใช้ปริมาณไฟฟ้าเชื่อมต่อกับเครื่องวัดปริมาณทางไฟฟ้า 1 เฟส เพื่อแสดงค่าปริมาณทางไฟฟ้าได้แก่ แรงดัน, กระแส, กำลังไฟฟ้า, ค่าประกอบกำลัง และค่าพลังงานไฟฟ้า ผ่าน NETPIE รวมถึงการจัดเก็บค่าที่ได้ทุก ๆ 15 วินาที ลงในการ์ดหน่วยความจำ

1.5 ขั้นตอนการดำเนินงาน

1. ศึกษาทำความเข้าใจรวมถึงรวบรวมข้อมูลเกี่ยวกับเครื่องวัด การนำข้อมูลที่วัดได้ออกจากเครื่องวัด และการนำไปใช้กับแผงวงจรอาดูโน่
2. วางแผนออกแบบวงจรและโปรแกรมที่จะใช้ รวมถึงต่อวงจรและทดสอบการทำงานของเครื่องวัด
3. ทดสอบการทำงานของวงจร และตรวจดูผลข้อมูลที่ได้จากวงจรวัดปริมาณทางไฟฟ้า
4. ทำการต่อวงจรเพิ่มในส่วนที่เกี่ยวกับแผงวงจรอาดูโน่ และทดสอบการทำงานของวงจรทั้งหมด
5. ทดสอบการทำงานของวงจรทั้งหมด โดยใช้ชุดทดสอบ(โหลด) และทำการวัดค่าปริมาณทางไฟฟ้าและส่งข้อมูลอัพโหลดสู่ NETPIE รวมถึงจัดเก็บบันทึกผลข้อมูลที่ได้ เป็นระยะเวลา 1 สัปดาห์ต่อเนื่อง
6. วิเคราะห์และสรุปผลที่ได้จากการจัดเก็บข้อมูล และเสนอโครงงานให้อาจารย์ที่ปรึกษาตรวจสอบ
7. เขียนโครงร่างปริญญานิพนธ์
8. จัดทำรูปเล่มปริญญานิพนธ์

1.6 แผนการดำเนินงาน

กิจกรรม	ปี 2560 - 2561									
	ส.ค.	ก.ย.	ต.ค.	พ.ย.	ธ.ค.	ม.ค.	ก.พ.	มี.ค.	เม.ย.	พ.ค.
1. ศึกษาทำความเข้าใจรวมถึงรวบรวมข้อมูลเกี่ยวกับเครื่องวัด การนำข้อมูลที่วัดได้ออกจากเครื่องวัด และการนำไปใช้กับแผงวงจรอาดูโน่										
2. วางแผนออกแบบวงจรและโปรแกรมที่จะใช้ รวมถึงต่อวงจรและทดสอบการทำงานของเครื่องวัด										

3. ทดสอบการทำงานของวงจร และตรวจดูผลข้อมูลที่ได้จากวงจรวัดปริมาณทางไฟฟ้า										
4. ทำการต่อวงจรเพิ่มในส่วนที่เกี่ยวกับแผงวงจรอาดูโน่ และทดสอบการทำงานของวงจรทั้งหมด										
5. ทดสอบการทำงานของวงจรทั้งหมด โดยใช้ชุดทดสอบ(โหลด) และทำการวัดค่าปริมาณทางไฟฟ้าและส่งข้อมูลอัพโหลดสู่ NETPIE รวมถึงจัดเก็บบันทึกผลข้อมูลที่ได้										
6. วิเคราะห์และสรุปผลที่ได้จากการจัดเก็บข้อมูล และเสนอโครงงานให้อาจารย์ที่ปรึกษาตรวจสอบ										
7. เขียนโครงร่างปริญญานิพนธ์										
8. จัดทำรูปเล่มปริญญานิพนธ์										

1.7 ผลที่คาดว่าจะได้รับ

1. เข้าใจการติดต่อสื่อสารระหว่าง มิเตอร์ไฟฟ้าแบบดิจิทัลกับ NETPIE
2. ใช้ระบบตรวจวัดการใช้งานปริมาณไฟฟ้าดูค่าที่มิเตอร์ไฟฟ้าแบบดิจิทัลวัดได้ผ่านหน้าเว็บ NETPIE
3. สามารถตรวจสอบข้อมูลการใช้พลังงานไฟฟ้าได้ตลอดเวลา และสามารถตรวจสอบข้อมูลการใช้พลังงานไฟฟ้าย้อนหลังได้ เพื่อวางแผนการอนุรักษ์หรือประหยัดพลังงาน

1.8 รายละเอียดงบประมาณตลอดโครงงาน

1. ไมโครคอนโทรลเลอร์	200	บาท
2. มิเตอร์วัดปริมาณไฟฟ้า	1,400	บาท
3. มอดูลสื่อสารไร้สายและมอดูลบันทึกข้อมูล	340	บาท
รวมเป็นเงิน (หนึ่งพันเก้าร้อยสี่สิบบาทถ้วน)	1,940	บาท

หมายเหตุ: ถัวเฉลี่ยทุกรายการ

บทที่ 2
ทฤษฎีหลักการและเอกสารที่เกี่ยวข้อง

ในการจัดทำโครงงานนี้จะพัฒนาระบบที่สามารถเชื่อมต่อกับมิเตอร์ไฟฟ้าแบบดิจิทัล 1 เฟส เพื่อให้แสดงผลปริมาณทางไฟฟ้า ผ่าน NETPIE และจัดเก็บข้อมูลลงในการ์ดความจำ โดยเชื่อมผ่านระบบสัญญาณไวไฟ (WI-FI) จะแสดงผลรายละเอียดของอุปกรณ์ในเบื้องต้นจะถูกอธิบายในบทนี้

2.1 มิเตอร์ไฟฟ้า

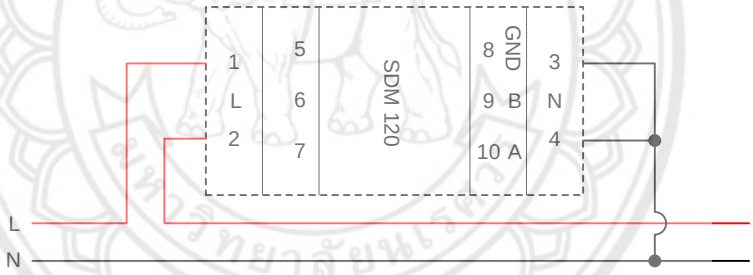
เป็นอุปกรณ์แสดงค่าพารามิเตอร์และปริมาณพลังงานไฟฟ้า เช่น แรงดัน, กระแส, กำลังงานไฟฟ้าจริง, กำลังงานไฟฟ้ารีแอคทีฟ เป็นต้น เพื่อให้ทราบถึงค่าทางไฟฟ้าในกระบวนการผลิตและการใช้พลังงานไฟฟ้าได้ โดยส่วนใหญ่แล้วในภาคอุตสาหกรรม จะนำมิเตอร์ไฟฟ้าไปใช้ในการควบคุมหรือปรับปรุงการใช้พลังงานไฟฟ้า เพื่อให้เกิดประสิทธิภาพในการทำงานได้อย่างเต็มที่ อีกทั้งยังเป็นการช่วยจัดการพลังงาน ซึ่งเป็นไปตามมาตรฐานของ ISO 50001 โดย มิเตอร์ไฟฟ้านั้น สามารถแบ่งออกได้เป็น 2 แบบ คือ มิเตอร์ไฟฟ้าแบบอนาล็อก และ มิเตอร์ไฟฟ้าแบบดิจิทัล โดยในโครงงานนี้ได้เลือกใช้ มิเตอร์ไฟฟ้าแบบดิจิทัล รุ่น SDM-120 สำหรับวัดค่าพารามิเตอร์ เพื่อการแสดงค่าปริมาณทางไฟฟ้า (บริษัท ศิลาไมโคร จำกัด, 2556)

คุณสมบัติมิเตอร์ไฟฟ้ารุ่น SDM-120

1. มิเตอร์ไฟฟ้าวัดกำลังไฟแบบ 1 เฟส
2. ใช้กับไฟ 220 โวลต์ 50/60 เฮิร์ท กระแส 0.25 ถึง 5 แอมแปร์
3. แสดงกำลังไฟฟ้า, แรงดัน, กระแส, ความถี่ และ เพาว์เวอร์แฟคเตอร์ได้
4. แสดงผลด้วย LCD ตัวเลข 6 หลัก ความสูง 5 มิลลิเมตร
5. อ่านข้อมูลและตั้งค่าผ่าน RS485 (Modbus) ได้
6. มี Pulse Output 2 ช่อง
7. ขนาดกว้าง 17.5 x สูง 119 x ลึก 62 มิลลิเมตร

รูปที่ 2.1 มิเตอร์ไฟฟ้าแบบดิจิทัล รุ่น SDM-120
ที่มา : http://smicrothai.com/smicro_3.php

การเชื่อมต่อมิเตอร์ไฟฟ้า รุ่น SDM-120 สามารถแสดงได้ดังรูปที่ 2.2 โดยหมายเลข 1 และ 2 จะเชื่อมต่อกับสายแหล่งจ่ายไฟฟ้า 220 โวลต์ หมายเลข 3 และ 4 จะเชื่อมต่อกับสายนิวตรอน หมายเลขที่ 5-8 ไม่ถูกนำมาใช้งาน และหมายเลข 9 กับ 10 จะเชื่อมต่อสัญญาณเอาท์พุตบนมาตรฐาน RS485 กับ อินพุตของมอดูลแปลงสัญญาณเพื่อแปลงเป็นมาตรฐาน RS232



รูปที่ 2.2 โครงสร้างมิเตอร์ไฟฟ้าแบบดิจิทัล รุ่น SDM-120
ที่มา : http://www.smicrothai.com/smicro_3.php

2.2 มาตรฐาน RS-485

มาตรฐาน RS485 เป็นหนึ่งในมาตรฐานการสื่อสารแบบอนุกรม (Serial Communication) เป็นระบบบัสที่พัฒนาต่อมาจาก RS422 และ RS232 เพื่อตอบสนองต่อความต้องการใช้งาน ที่ต้องการเชื่อมต่ออุปกรณ์หลาย ๆ ตัวบนเครือข่ายเดียวกันเข้าด้วยกัน โดยมีระยะทางการสื่อสารที่ไกลขึ้น และมีความเร็วรับส่งข้อมูลที่สูงขึ้น เมื่อเทียบกับมาตรฐานการสื่อสาร RS232 และ RS422

RS485 เป็นการสื่อสารอนุกรมของมิเตอร์ไฟฟ้าแบบดิจิทัลรุ่น SDM-120 ภายในจะมีการติดต่อสื่อสารโดยใช้โปรโตคอลมอดบัส ที่จะส่งค่าพารามิเตอร์ต่าง ๆ ไปที่คอมพิวเตอร์ แต่การที่จะเชื่อมต่อเข้ากับคอมพิวเตอร์ได้นั้น ต้องมีการแปลงจาก RS485 ไปเป็น RS232 ซึ่งใช้มอดูลแปลง

สัญญาณ (RS485) เพื่อที่จะเข้ากับพอร์ตขนานของคอมพิวเตอร์ได้ การส่งข้อมูลจากมิเตอร์ไปยังคอมพิวเตอร์ จะเป็นการส่งข้อมูลแบบอนุกรม โดยเป็นการส่งแบบไม่ประสานเวลา (Asynchronous) ลักษณะการส่งจะมีสายสัญญาณ 2 เส้น คือ RXD TXD และ RS485 จะไม่เทียบสัญญาณจากค่ากราวด์เหมือน RS232 แต่จะเทียบสัญญาณระหว่างสาย 2 เส้นที่ใช้ส่ง ทำให้ RS485 ทำงานได้แบบ Half duplex นั่นคือการสื่อสารข้อมูล 2 ทิศทางจะต้องแยกการรับหรือการส่งข้อมูลออกจากกัน ไม่สามารถรับส่งข้อมูลได้เวลาเดียวกันยกตัวอย่างเช่น วิทยุสื่อสาร ส่วนแบบ Full duplex จะเป็นการรับ-ส่งข้อมูลได้ในเวลาเดียวกัน ยกตัวอย่างเช่นโทรศัพท์

การส่งข้อมูลบนมาตรฐาน RS485 จะใช้สายสัญญาณแบบสายคู่พันเกลียวระยะทางในการส่งข้อมูลสูงสุด จะอยู่ที่ 1200 เมตร ที่ระยะนี้ความเร็วในการส่งข้อมูลจะอยู่ที่ 100 กิโลบิตต่อวินาที เพียงพอต่อการใช้งานในโรงงานอุตสาหกรรมและระยะการส่งสัญญาณได้ถูกพัฒนาขึ้นมาก ซึ่งเหนือกว่ามาตรฐานรุ่นเก่าอย่าง RS232 ที่สามารถส่งสัญญาณได้เพียง 15 เมตร เท่านั้น ดังรูปที่ 2.3 มาตรฐาน RS485 มีประโยชน์เป็นอย่างมากสำหรับระบบงานที่มีเครื่องวัดหลายตัวเชื่อมต่อบนสายสัญญาณที่เป็นบัสเดียวกัน แต่อย่างไรก็ตามจะต้องมีความระมัดระวังอย่างเป็นพิเศษในการตั้งค่าซอฟต์แวร์เพื่อป้องกันไม่ให้หลายอุปกรณ์ส่งข้อมูลในเวลาเดียวกันวิธีการที่ใช้กันส่วนใหญ่จะกำหนดให้อุปกรณ์หรือจุดเชื่อมต่อตัวอุปกรณ์ตัวหนึ่งทำตัวเป็นตัวแม่ (บริษัท โอเมก้า เมชเชอริ่ง อินสทรูเม้นท์ จำกัด, พฤศจิกายน 2557)

ตารางเปรียบเทียบคุณสมบัติ RS232 RS423 RS422 RS485

	RS232	RS423	RS422	RS485
Differential	no	no	yes	yes
Max number of drivers	1	1	1	32
Max number of receivers	1	10	10	32
Modes of operation	half duplex full duplex	half duplex	half duplex	half duplex
Network topology	point-to-point	multidrop	multidrop	multipoint
Max distance (acc. standard)	15 m	1200 m	1200 m	1200 m
Max speed at 12 m	20 kbs	100 kbs	10 Mbs	35 Mbs
Max speed at 1200 m	(1 kbs)	1 kbs	100 kbs	100 kbs
Max slew rate	30 V/µs	adjustable	n/a	n/a
Receiver input resistance	3..7 kΩ	≧ 4 kΩ	≧ 4 kΩ	≧ 12 kΩ
Driver load impedance	3..7 kΩ	≧ 450 Ω	100 Ω	54 Ω
Receiver input sensitivity	±3 V	±200 mV	±200 mV	±200 mV
Receiver input range	±15 V	±12 V	±10 V	-7..12 V
Max driver output voltage	±25 V	±6 V	±6 V	-7..12 V
Min driver output voltage (with load)	±5 V	±3.6 V	±2.0 V	±1.5 V

รูปที่ 2.3 เปรียบเทียบมาตรฐาน RS232 RS422 และ RS485

ที่มา : https://www.omi.co.th/image/upload/Image/article_046_rs232_compared.jpg

2.3 มอดูลแปลงสัญญาณ

มอดูลแปลงสัญญาณ AC-3B485 เป็นมอดูลแปลงสัญญาณจาก RS485 เป็น RS232 เป็นมาตรฐานที่รับ/ส่งข้อมูลในแบบที่เรียกว่า Half duplex คือ สามารถรับและส่งข้อมูลได้ทีละอย่างเท่านั้นไม่สามารถทำทั้งสองอย่างได้ในเวลาเดียวกัน มีลักษณะคล้ายวิทยุสื่อสารที่ต้องคอยสลับกันพูดทีละครั้ง (บริษัท โอเมก้า เมชเซอริ่ง อินสทรูเม้นท์ จำกัด, พฤศจิกายน 2557)

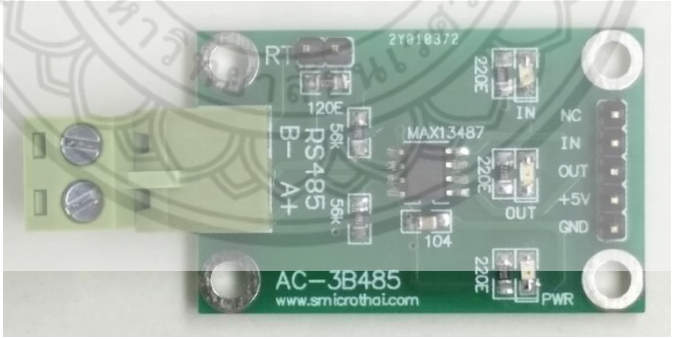
สำหรับการรับ/ส่งข้อมูลดิจิทัลแบบ RS485 นั้น จะส่งข้อมูลโดยใช้สายไฟเพียงแค่ 2 เส้นคือ A และ B เป็นตัวบอกค่ารหัสดิจิทัล (Digital code) โดยใช้ความแตกต่างของแรงดันไฟฟ้าระหว่างขั้ว A และ B

เมื่อ Va - Vb ได้แรงดันไฟฟ้าน้อยกว่า -200 มิลลิโวลต์ คือสัญญาณดิจิทัลเป็น 1

เมื่อ Va - Vb ได้แรงดันไฟฟ้ามากกว่า +200 มิลลิโวลต์ คือสัญญาณดิจิทัลเป็น 0

สามารถแสดงโครงสร้างหน่วยประมวลผลกลางได้ดังรูปที่ 2.4

1. ขา A คือ รับค่าสัญญาณมอดบัส โปรโตคอล
2. ขา B คือ รับค่าสัญญาณมอดบัส โปรโตคอล
3. ขา +5 โวลต์ คือ รับแรงดันกระแสตรง 5 โวลต์
4. ขา GND คือ กราวด์
5. ขา IN คือ ขารับสัญญาณทางไฟฟ้า
6. ขา OUT คือ ขาส่งสัญญาณทางไฟฟ้า



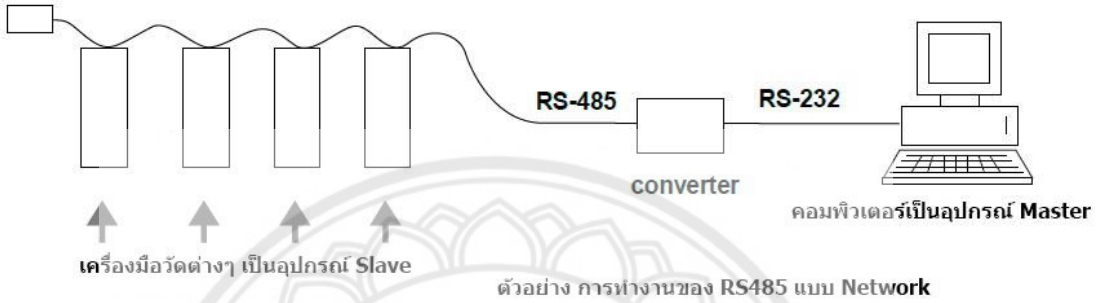
รูปที่ 2.4 มอดูลแปลงสัญญาณ (RS 485)

ที่มา : https://www.banggood.com/th/5V-MAX485

2.4 การติดต่อสื่อสารแบบ Master/Slave

เป็นการสื่อสารข้อมูลในลักษณะ Master/Slave ซึ่งเป็นการสื่อสารจากอุปกรณ์แม่ (Master) เครื่องเดียว ส่วนใหญ่มักเป็นซอฟต์แวร์คอมพิวเตอร์หรืออุปกรณ์แสดงผล HMI ไปยังอุปกรณ์ลูก (Slave) ได้หลาย ๆ เครื่อง โดยสามารถกำหนดหมายเลขอุปกรณ์ได้สูงสุด 255 เครื่อง การติดต่อสื่อสารแบบ Master/Slave โดยที่ Slave แต่ละตัวจะมีหมายเลข Address ของตัวเอง

และเมื่อตัว Master ต้องการสั่งการตัว Slave ตัว Master จะส่งชุดคำสั่งพร้อมระบุหมายเลข Address ไปยังอุปกรณ์ Slave ทุกตัว เมื่ออุปกรณ์ Slave ได้รับคำสั่งและคำสั่งนั้นมีหมายเลข Address ตรงกับตัวเอง อุปกรณ์ Slave ถึงจะทำตามคำสั่งของ Masterเป็นลำดับไป โดยการประยุกต์ใช้กับโครงงานนี้จะใช้ไมโครคอนโทรลเลอร์เป็นอุปกรณ์ Master และ ดิจิทัลมิเตอร์เป็นอุปกรณ์ Slave (บริษัท โอเมก้า เมชเซอริ่ง อินสทรูเม้นท์ จำกัด, พฤศจิกายน 2557) แสดงดังรูปที่ 2.5



รูปที่ 2.5 การติดต่อสื่อสารแบบ Master/Slave

ที่มา : https://www.omi.co.th/image/upload/Image/article_057_RS485-wiring-network.jpg

2.5 ไมโครคอนโทรลเลอร์

อาดูโน่ Mega 2560 เป็นแผงวงจรไมโครคอนโทรลเลอร์ที่ใช้ชิพ ATmaga2560 ซึ่งมี 54 ดิจิทัลอินพุต/เอาท์พุต โดยในขาเหล่านั้นสามารถใช้งานเป็น PWM ได้ 15 ขา, อนาล็อคอินพุต 16 ขา, UART 4 ชุด โดยความถี่คริสตรัลบนแผงวงจรคือ 16 MHz เชื่อมต่อข้อมูลระหว่างคอมพิวเตอร์ผ่านพอร์ต USB บนแผงวงจรได้โดยตรง อีกทั้งรูปแบบการออกแบบยังออกแบบให้รองรับการสวมกับชิลด์ต่าง ๆได้โดยตรง ทำให้สามารถพัฒนาระบบต่าง ๆ ได้อย่างรวดเร็วและ เรียบร้อยสวยงาม โดยรองรับการพัฒนาโปรแกรมบนแพลตฟอร์มอาดูโน่ อย่างเต็มรูปแบบ

แผงวงจรอาดูโน่ Mega 2560 จะเหมือนกับอาดูโน่ Mega ADK ต่างกันตรงที่บนแผงวงจรไม่มี USB Host มาให้ การที่โปรแกรมยังต้องทำผ่านโปรโตคอล UART อยู่บนแผงวงจรใช้ชิปไอซีไมโครคอนโทรเลอร์หมายเลข ATmega2560 เป็นแผงวงจรอาดูโน่ ที่ออกแบบมาสำหรับงานที่ต้องใช้ IO มากกว่าอาดูโน่ Uno R3 เช่น งานที่ต้องการรับสัญญาณจากเซ็นเซอร์ หรือควบคุมเซอร์โวมอเตอร์หลาย ๆ ตัว ทำให้ Pin IO ของแผงวงจรอาดูโน่ Uno R3 ไม่สามารถรองรับได้ ทั้งนี้แผงวงจร ATMega 2560 ยังมีความหน่วยความจำแบบแฟลช มากกว่าอาดูโน่ Uno R3 ทำให้สามารถเขียนโค้ดโปรแกรมเข้าไปได้มากกว่า ในความเร็วของ MCU ที่เท่ากัน (เอกสารประกอบการสอนวิชาไมโครคอนโทรลเลอร์เบื้องต้น, พฤษภาคม 2556) แผงวงอาดูโน่ Mega 2560 แสดงดังรูปที่ 2.6

รูปที่ 2.6 ไมโครคอนโทรลเลอร์อาดูโน่ Mega 2560
ที่มา : www.Thaieasyelec.com

ข้อมูลจำเพาะ

1. ชิปไอซีไมโครคอนโทรเลอร์	ATmega2560
2. ใช้แรงดันไฟฟ้า	5 โวลต์
3. รองรับการจ่ายแรงดันไฟฟ้า (ที่แนะนำ)	7 – 12 โวลต์
4. รองรับการจ่ายแรงดันไฟฟ้า (ที่จำกัด)	6 – 20 โวลต์
5. พอร์ต Digital I/O	54 พอร์ต (มี 15 พอร์ต PWM output)
6. พอร์ต Analog Input	16 พอร์ต
7. กระแสไฟฟ้าที่รวมที่จ่ายได้ทุกพอร์ต	40 มิลลิแอมแปร์
8. กระแสไฟฟ้าที่จ่ายได้ในพอร์ต 3.3 V	50 มิลลิแอมแปร์
9. พื้นที่โปรแกรมภายใน	256 กิโลไบต์
10.พื้นที่แรม	8 กิโลไบต์
11.พื้นที่หน่วยความจำถาวร (EEPROM)	4 กิโลไบต์
12.ความถี่คริสตัล	16 เมกะเฮิร์ท

2.6 มอดูลเก็บบันทึกข้อมูล

มอดูลเก็บบันทึกข้อมูล (Data logger shield) ใช้สำหรับเก็บค่าต่าง ๆ ตามเวลาที่ต้องการ เช่น ค่าของเซ็นเซอร์ ต่าง ๆ โดยค่าที่ต้องการจะบันทึกค่าลงบนการ์ดหน่วยความจำในชิลด์ ประกอบด้วย นาฬิกาเวลาจริงช่องเสียบการ์ดหน่วยความจำและช่องใส่ถ่านสำรอง เพื่อช่วยให้ Real Time Clock (RTC) ยังทำงานอยู่ในกรณีไม่มีไฟเลี้ยงแผงวงจร ใช้งานได้กับแผงวงจรอาดูโน่ UNO, Duemilanove, Diecimila, Leonardo, ADK/Mega R3 (ต่อสายเพิ่ม)

RTC ก็คือ อุปกรณ์ที่ให้ค่าเวลาตามฐานเวลาจริง ซึ่งการทำงานของมันจะทำงานโดยตัวกำเนิดสัญญาณนาฬิกาภายนอก ที่ต่อเข้าไปหรือบางตัวจะมีตัวกำเนิดสัญญาณนาฬิกาภายในตัวของมันเอง ซึ่งจะทำงานที่ความถี่ 32.768 กิโลเฮิร์ท ทำให้การนับเวลาของมันค่อนข้างแม่นยำ สำหรับงานที่เกี่ยวกับการบันทึกเวลา หรือมอดูลเก็บบันทึกข้อมูล ตัวอุปกรณ์นาฬิกาเวลาจริงจึงมีความจำเป็นมาก

ในโครงงานนี้จะนำแผงวงจรมอดูลเก็บบันทึกข้อมูล มาทำงานร่วมกับแผงวงจรอาดูโน่ mega 2560 ซึ่งในการที่จะสามารถทำให้บันทึกค่าลงในการ์ดหน่วยความจำได้นั้น จะใช้แผงวงจรมอดูลเก็บบันทึกข้อมูล ดังรูป 2.7 ที่แผงวงจรอิเล็กทรอนิกส์ ที่ใช้สำหรับบันทึกข้อมูลลงในการ์ดหน่วยความจำ พร้อมกับมีนาฬิกาเวลาจริง (Arduitronics, พฤษภาคม 2560)

การใช้ขาติดต่อแลกเปลี่ยนข้อมูลกับแผงวงจรอาดูโน่

1. MOSI (Master In Slave Out) สลาฟส่งข้อมูลถึงมาสเตอร์ ขา Digital Pin 11
2. MISO (Master Out Slave In) มาสเตอร์ส่งข้อมูลถึงสลาฟ ขา Digital Pin 12
3. CLK การเชื่อมต่อสัญญาณพัลส์นาฬิกา ขา Digital Pin 13
4. CS การเลือกชิพสำหรับการ์ดหน่วยความจำ Digital Pin 10

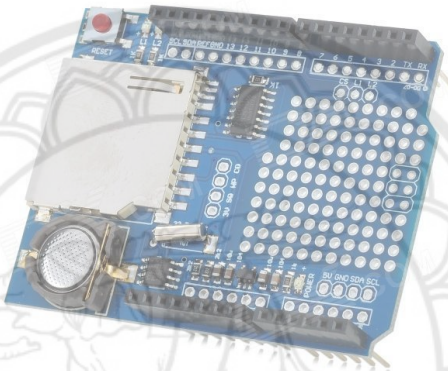
ประโยชน์ของแผงวงจรมอดูลเก็บบันทึกข้อมูล

5. ช่วยลดเวลาการทำงาน ที่ต้องคอยจดบันทึกค่าตามช่วงเวลาที่กำหนดไว้
6. สามารถวัดค่าได้ต่อเนื่อง ตลอด 24 ชั่วโมง และช่วยจัดเก็บข้อมูลให้เป็นหมวดหมู่รวมถึงแสดงผลได้อย่างเป็นระเบียบ อีกทั้งยังช่วยลดความผิดพลาดในการจดบันทึก
7. สามารถดูแนวโน้มการเพิ่มขึ้น ลดลงของค่าที่สนใจได้ เช่น อุณหภูมิ หรือความชื้นสัมพัทธ์

การใช้งานของแผงวงจรมอดูลเก็บบันทึกข้อมูล

1. มีนาฬิกาเวลาจริง โดยมีชิพ คริสตัล และแบตเตอรี่สำรองสำหรับการทำงาน
2. สามารถใช้การ์ดหน่วยความจำ SD/MMC ได้สูงสุดที่ 32 กิกะไบต์
3. มีปุ่มรีเซ็ต ซึ่งในการกดรีเซ็ตในแต่ละครั้งจะเป็นการเริ่มการทำงานใหม่

4. มี Shifter ที่ช่วยป้องกันการ์ดหน่วยความจำเกิดความเสียหายจากกรณีที่ไฟเกิน ซึ่งในการทำงานจะใช้แรงดันไฟฟ้าเพียง 3 โวลต์
5. ที่ขา 3 โวลต์ เป็นแรงดันไฟฟ้าจากตัวควบคุมแผงวงจร
6. ที่ขา SQ เป็นสัญญาณรูปสี่เหลี่ยมจากนาฬิกาเวลาจริง
7. ที่ขา WP เป็นขาตรวจเช็คการ์ดหน่วยความจำ
8. ที่ขา CD เป็นขาตรวจสอบหาการ์ดหน่วยความจำ
9. ที่ขา CS เป็นขา Pin เลือกชิพสำหรับการ์ดหน่วยความจำ
10. L1 และ L2 เป็นตัวเลือกใช้ไฟ แอลอีดี



รูปที่ 2.7 แผงวงจรมอดูลเก็บบันทึกข้อมูล
ที่มา : https://www.arduitronics.com/product/179/data-logger-shield

2.7 มอดูลสื่อสารไร้สาย

ESP8266 เป็นชื่อของชิพไอซีบนแผงวงจรของมอดูล ซึ่งไอซี ESP8266 ไม่มีพื้นที่โปรแกรมในตัว ทำให้ต้องใช้ไอซีภายนอก ในการเก็บโปรแกรม ที่ใช้การเชื่อมต่อผ่านโปรโตคอล SPI (หมายความว่าสัญญาณสามารถส่งหากันได้ระหว่าง Master และ Slave ได้อย่างต่อเนื่อง) ซึ่งสาเหตุนี้เองทำให้มอดูล ESP8266 มีพื้นที่โปรแกรมมากกว่าไอซีไมโครคอนโทรลเลอร์เบอร์อื่น ๆ มอดูลสื่อสารไร้สายแสดงดังรูปที่ 2.8 (ayarafun, สิงหาคม 2558)

ESP8266 ทำงานที่แรงดันไฟฟ้า 3.3 โวลต์ - 3.6 โวลต์ การนำไปใช้งานร่วมกับเซ็นเซอร์อื่น ๆ ที่ใช้แรงดัน 5 โวลต์ ต้องใช้วงจรแบ่งแรงดันมาช่วย เพื่อไม่ให้มอดูลพังเสียหาย กระแสที่มอดูลใช้งานสูงสุดคือ 200 มิลลิแอมแปร์ ความถี่คริสตอล 40 เมกะเฮิร์ท ทำให้เมื่อนำไปใช้งานอุปกรณ์ที่ทำงานรวดเร็วตามความถี่ เช่น แอลซีดีทำให้การแสดงผลข้อมูลรวดเร็วกว่าไมโครคอนโทรลเลอร์ยอดนิยมอาดูโน่มาก

คุณสมบัติ

1. รองรับมาตรฐาน IEEE802.11
2. รองรับการทำงานแบบ WiFi Direct (P2P) และ SoftAP
3. วงจรสวิตช์ TR, สายอากาศที่มี Balun, LNA หรือวงจรขยายสัญญาณรบกวนต่ำ, วงจรขยายกำลังและแมตชิ่งเน็ตเวิร์กเพื่อการรับคลื่นวิทยุอย่างมีประสิทธิภาพ
4. วงจรเฟสล็อกลูปและวงจรควบคุมสัญญาณออสซิลเลเตอร์แบบดิจิทัล (DCXO) เพื่อช่วยบริหารจัดการด้านความถี่
5. วงจรบริหารกำลังไฟฟ้าและวงจรควบคุมไฟเลี้ยงคงที่ เพื่อช่วยให้วงจร Wi-Fi ใช้กำลังได้อย่างเหมาะสม โดยปกติต้องการกระแสไฟฟ้า 0.9 มิลลิแอมแปร์ ในขณะแสตนด์บาย, 135-215 มิลลิแอมแปร์ ขณะส่งข้อมูล, 60 มิลลิแอมแปร์ เมื่อรับข้อมูล, 1 มิลลิแอมแปร์ ในโหมดประหยัดพลังงาน และ 0.5 ไมโครแอมแปร์ ในขณะปิด
6. กำลังส่ง +19.5 เดซิเบลต่อเมตร เมื่อทำงานในโหมด 802.11b

รูปที่ 2.8 แผงวงจร ESP8266

ที่มา : https://netpie.gitbooks.io/nodemcu-esp8266-on-netpie/content/lab-6.html

ข้อดีของมอดูลสื่อสารไร้สาย

1. สามารถกด upload sketch ได้ เชื่อมต่อแผงวงจร USB กับคอมพิวเตอร์ใช้งานง่าย ขนาดของแผงวงจรต่อลง Protoboard ได้
2. ชิบภายใน ESP8266 มี CPU ขนาด 32 บิต แตกต่างจาก อาดูโน่ ที่เป็น CPU 8 บิต
3. ถึงแม้ขา I/O จะไม่มากเท่าของ อาดูโน่ แต่เราสามารถเขียนโปรแกรมลงบนขา GPIO ได้ทุกขาพอๆกัน เป็นข้อดีที่เพิ่มมาจากความต้องการใช้ WIFI เชื่อมต่อเมื่อต้องการเล่นอาดูโน่ ทำให้ต้องซื้อมอดูลไวไฟเพิ่มนั่นคือ NodeMCU (ESP8266) มีต้นทุนต่ำกว่ามาก
4. มีอุปกรณ์หลายอย่างที่ใช้งานที่แรงดัน +3.3 โวลต์ เป็นส่วนใหญ่ ดังนั้นเราสามารถนำ NodeMCU (ESP8266) มาใช้เชื่อมต่อได้โดยตรง

2.8 NETPIE

NETPIE (Network Platform for Internet of Everything) คือ cloud platform (การให้บริการด้าน Platform สำหรับผู้ใช้งานที่ทำงานด้านซอฟต์แวร์และแอพพลิเคชันโดยผู้ให้บริการ Cloud จะจัดเตรียมสิ่งที่จำเป็นต้องใช้ในการพัฒนาซอฟต์แวร์และแอพพลิเคชันเอาไว้ให้) ถูกออกแบบและพัฒนาขึ้นโดยศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ (NECTEC) เพื่ออำนวยให้เกิดการสื่อสารระหว่างอุปกรณ์ในเครือข่าย IoT (Internet of Things) โดยมีประโยชน์ต่อนักพัฒนาและอุตสาหกรรมไทย อาทิ NETPIE ช่วยให้อุปกรณ์สามารถคุยกันได้โดยผู้พัฒนาไม่ต้องกังวลว่าอุปกรณ์นั้นจะอยู่ที่ใดเพียงนำ NETPIE library ไปติดตั้งในอุปกรณ์ NETPIE จะรับหน้าที่ดูแลการเชื่อมต่อให้ทั้งหมดไม่ว่าอุปกรณ์นั้นจะอยู่ในเครือข่ายชนิดใดลักษณะใด หรือแม้กระทั่งเคลื่อนย้ายไปอยู่ที่ใดผู้พัฒนาสามารถตัดปัญหากวนใจในการที่จะต้องมาออกแบบการเข้าถึงอุปกรณ์จากระยะไกล ไม่เพียงเท่านั้น NETPIE ยังช่วยให้การเริ่มต้นใช้งานเป็นไปโดยง่ายโดยการออกแบบให้อุปกรณ์ถูกค้นพบและเข้าสู่บริการโดยอัตโนมัติ NETPIE ถูกออกแบบให้ผู้พัฒนาสามารถออกแบบได้เองทั้งหมดเช่น สิ่งใดมีสิทธิคุยกับสิ่งใด สิ่งใดมีสิทธิหรือไม่เพียงใดในการอ่านหรือเขียนข้อมูลและสิทธิเหล่านี้จะมีอายุเท่าใดหรือถูกเพิกถอนภายใต้เงื่อนไขใดเป็นต้น

NETPIE มีสถาปัตยกรรมเป็น cloud อย่างแท้จริงในทุก ๆ ระดับของระบบ ทำให้เกิดความยืดหยุ่นและคล่องตัวสูงในการขยายตัวนอกจากนี้ โมดูลต่าง ๆ ยังถูกออกแบบให้ทำงานแยกจากกันและสื่อสารกันด้วยวิธีการ ส่งแบบอะซิงโครนัส ช่วยให้แพลตฟอร์มมีความเชื่อถือได้สูงสามารถนำไปใช้ซ้ำและพัฒนาต่อเติมได้ง่าย ดังนั้นผู้พัฒนาไม่จำเป็นต้องกังวลกับการขยายตัวเพื่อรับโหลดที่เพิ่มขึ้นในระบบอีกต่อไป นอกจากนี้ทางเน็คเท็คจะเปิด NETPIE library ในรูปแบบซอฟต์แวร์ที่เปิดเผยให้นักพัฒนาสามารถนำไปปรับปรุงต่อให้ตรงกับความต้องการใช้งานโดยเปิดโอกาสให้นำไปใช้ในเชิงพาณิชย์ได้ (NECTEC, กันยายน 2559) เครือข่าย NETPIE แสดงดังรูปที่ 2.9

รูปที่ 2.9 เครือข่าย NETPIE

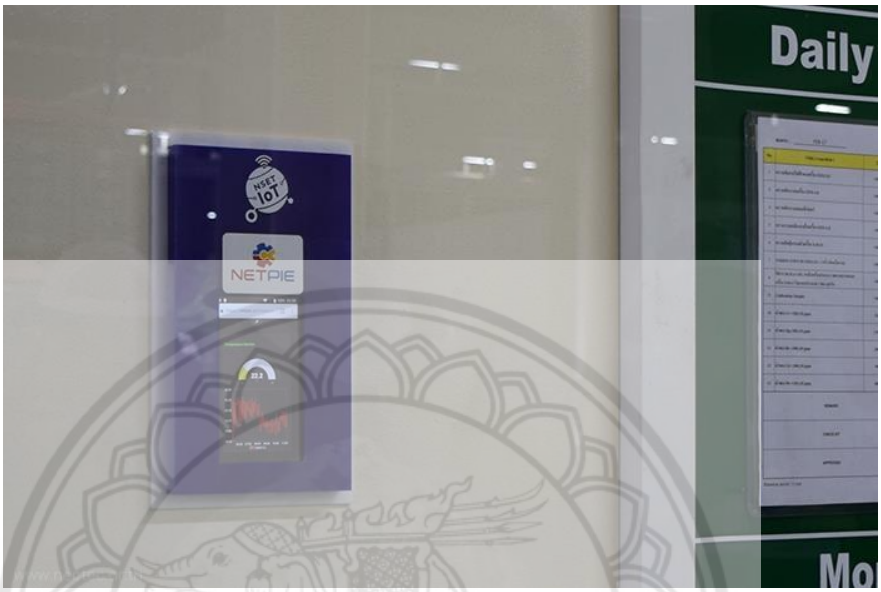
ที่มา : https://1.bp.blogspot.com/netpie-platform.png

ผลงานที่เกิดจากการประยุกต์ใช้งานแพลตฟอร์ม NETPIE ในภาคอุตสาหกรรมการผลิตโดยบริษัท นิเด็ค ชิบาอุระ อิเล็คทรอนิกส์ (ประเทศไทย) นำ NETPIE มาช่วยพัฒนาโรงงานเพื่อก้าวไปสู่เป้าหมายของการเป็น Smart Factory ตัวอย่างการใช้งาน NETPIE ที่นิเด็คฯ มีดังนี้ (NECTEC, มีนาคม 2560)

1. ระบบควบคุมการเปิด / ปิดไฟในห้องและบริเวณทางเดินด้วยเว็บแอพพลิเคชั่นบนมือถือ แสดงดังรูปที่ 2.10

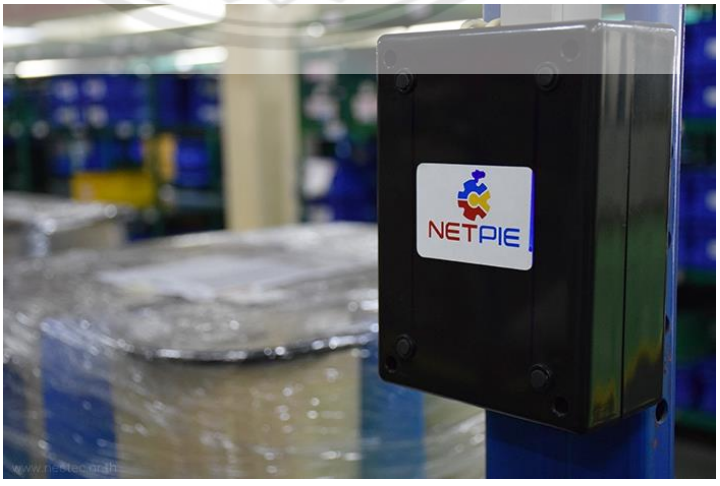
รูปที่ 2.10 เว็บแอพพลิเคชั่นควบคุมการเปิด / ปิดไฟ
ที่มา : https://www.nectec.or.th/sectionImage/4643

2. ระบบควบคุมอุณหภูมิในห้องตรวจสอบคุณภาพสินค้า เดิมพนักงานจะต้องเข้าไปจดบันทึกอุณหภูมิในห้อง ทุก ๆ 2 ชั่วโมง ทำให้อาจได้ข้อมูลที่คลาดเคลื่อนจากการวัดที่ไม่ตรงเวลา ปัจจุบันนิเด็คฯ ใช้ NETPIE เข้ามาช่วยในการวัดอุณหภูมิและบันทึกข้อมูลตามเวลาที่กำหนด อีกทั้งยังทำให้สามารถเรียกดูข้อมูลย้อนหลังได้ แสดงดังรูปที่ 2.11



รูปที่ 2.11 ระบบควบคุมอุณหภูมิในห้องตรวจสอบคุณภาพสินค้า
ที่มา : https://www.nectec.or.th/sectionImage/4645

3. ระบบจัดการคลังสินค้า ทำการบันทึกข้อมูลสินค้าด้วย RFID ติดตั้งชุด IoT ไว้ที่ชั้นวางสินค้า มีระบบ Location Mapping จะทำให้ทราบข้อมูลของสินค้าว่าเป็นสินค้าใด จำนวนเท่าไหร่และเก็บไว้ตรงไหนบ้างแสดงดังรูปที่ 2.12

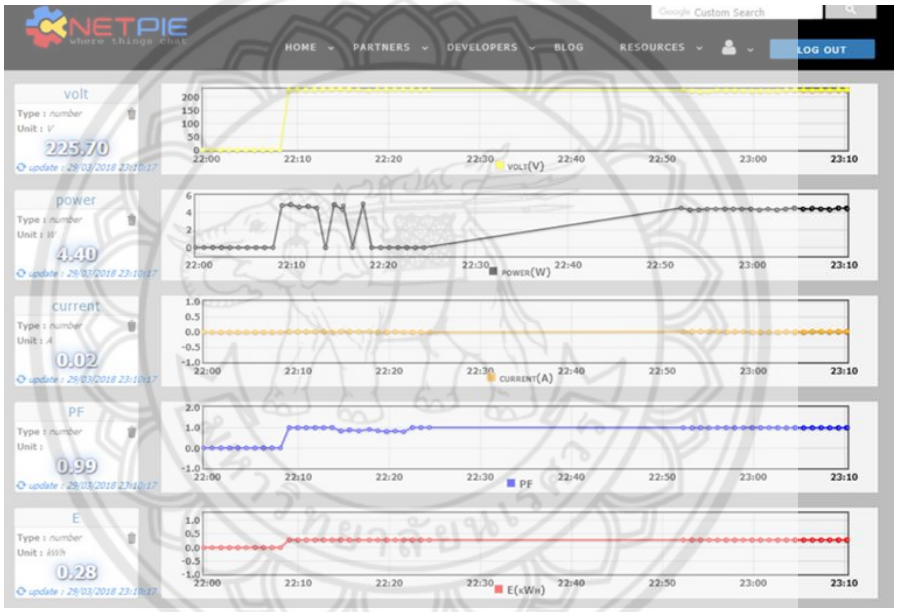


รูปที่ 2.12 กล่องระบบ Location Mapping ติดตั้งไว้ที่ชั้นวางสินค้าในคลัง
ที่มา : https://www.nectec.or.th/sectionImage/4647

2.8.1 วิธีการใช้ข้อมูลของ NETPIE

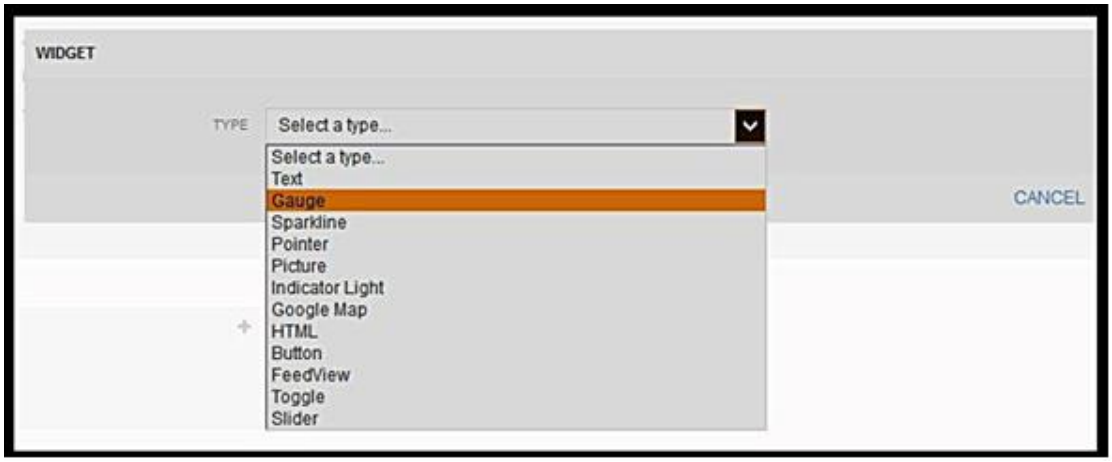
NETPIE มีวิธีใช้งานข้อมูลอยู่ 2 แบบดังนี้

วิธีที่ 1 การแสดงผลผ่านหน้าการจัดการของ Feed ใช้สำหรับการตรวจสอบหรือการแสดงผลของข้อมูล โดยเฉพาะอย่างยิ่งข้อมูลแบบเวลาจริง (Real time) ซึ่งจะต้องเก็บรวบรวมด้วยอัตราความถี่ที่เหมาะสมเพื่อให้การตรวจสอบหรือการแสดงผลตรงตามความต้องการในปัจจุบัน แพลตฟอร์ม NETPIE มีบริการที่สามารถเก็บข้อมูลและแสดงผลที่เรียกว่า Feed ซึ่งทำหน้าที่เสมือนถังเก็บข้อมูลประเภทอนุกรมเวลา กล่าวคือ เป็นชุดข้อมูลหรือค่าตัวแปร ณ เวลาต่าง ๆ เช่น อุณหภูมิอากาศในช่วงเวลาต่าง ๆ ของวัน เป็นต้น ข้อมูลเหล่านี้จะถูกเก็บแบบต่อเนื่องสะสมกันไปตลอด และสามารถเรียกออกมาดูในช่วงเวลาใดก็ได้ การแสดงผลผ่านหน้าการจัดการของ Feed (NECTEC, กันยายน 2559) แสดงดังรูปที่ 2.13



รูปที่ 2.13 การแสดงผลผ่านหน้าการจัดการของ Feed

วิธีที่ 2 การแสดงผลผ่านหน้า Freeboard นอกจากดูค่าผ่านหน้าการจัดการ Feed แล้ว เรายังสามารถดึงค่าของ Feed ไปแสดงผลบน NETPIE Freeboard ร่วมกับประเภทของกระดานข้อมูลต่าง ๆ ได้อีกด้วย โดยประเภทของกระดานข้อมูล (Widget) แสดงดังรูปที่ 2.14



รูปที่ 2.14 รูปแสดงการเลือกประเภทของ Widget

ที่มา : https://netpie.gitbooks.io/5-freeboard/content/assets/i7.jpg

จากรูปที่ 2.13 จะสามารถอธิบายการนำไปใช้งานได้ดังนี้

Widget ชนิด Text เป็นการแสดงผลในรูปแบบข้อความลักษณะของข้อความ

Widget ชนิด Gauge เป็นการแสดงผลในรูปแบบการวัดค่า กำหนดค่าต่ำสุดสูงสุดได้

Widget ชนิด Sparkline เป็นการแสดงผลในรูปแบบเส้นกราฟ และเส้นกราฟแสดงตามข้อมูลที่ระบุจุด

Widget ชนิด Pointer เป็นการแสดงผลในรูปแบบเข็มชี้ ซึ่งกำหนดตำแหน่งเข็มชี้ได้ตั้งแต่0-359 แต่ถ้าเกิน 360 ก็จะเริ่มต้นตำแหน่ง 0 ใหม่อีกครั้ง

Widget ชนิด Indicator Light เป็นการแสดงผลในรูปแบบสถานะ เปิด/ปิด เพื่อระบุสถานะของการทำงาน เช่น สถานะไฟ สถานะเครื่องจักรทำงาน

Widget ชนิด Google Map เป็นการแสดงผลในรูปแบบแผนที่ ใช้สำหรับระบุตำแหน่งที่ตั้งอุปกรณ์ หรือ car gps tracker HTML

Widget ชนิด HTML เป็นการแสดงผลในรูปแบบหน้าเว็บ HTML สามารถเขียนโค้ดเป็นภาษาHTML หรือ Javascript ได้ FeedView

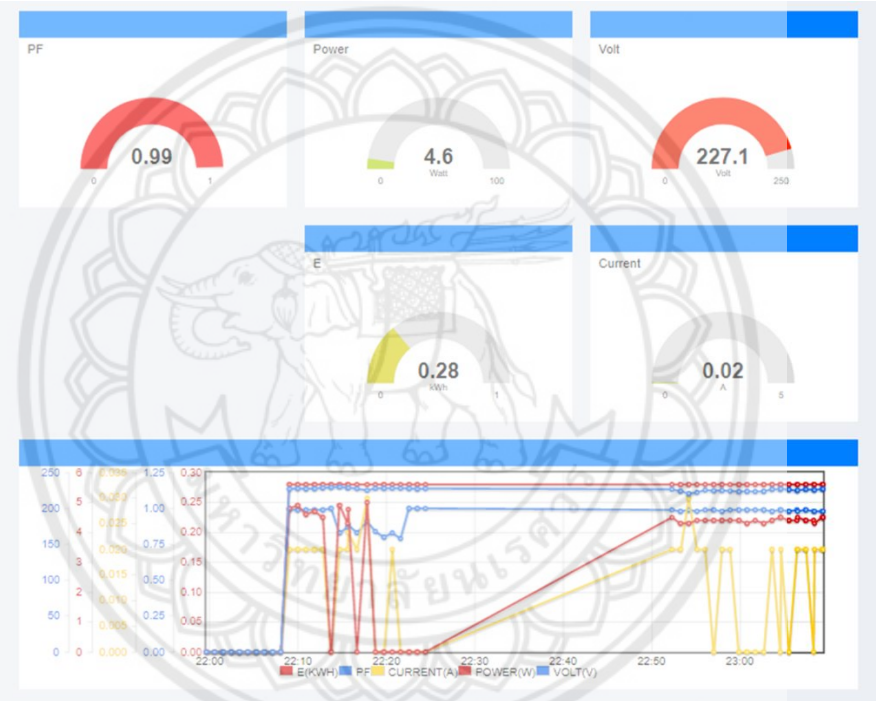
Widget ชนิด Button เป็นการแสดงผลในรูปแบบปุ่มกด กำหนดการกระทำที่จะทำงานเมื่อสั่งงานได้สไลด์

Widget ชนิด Toggle เป็นการแสดงผลในรูปแบบปุ่มกด 2 สถานะ กำหนดการกระทำที่จะทำงานได้ทั้งเปิด/ปิด

Widget ชนิด Slider เป็นการแสดงผลในรูปแบบปุ่มกดแบบ slide กำหนดระดับค่าสำหรับการใช้งาน เช่น ความคุมความเร็วการทำงานรอบมอเตอร์ กำหนดระดับแสงสว่างของหลอดไฟเป็นต้น

Widget ชนิด FeedView เป็นการแสดงผล Feed ในรูปแบบกราฟ เลือกแสดงผลกราฟได้หลายเส้น ตามค่าที่เก็บใน Feed

Freeboard เป็น เว็บแอพพลิเคชั่น ที่สามารถสร้างกระดานข้อมูล เพื่อแสดงผลสำหรับ IoT แอปพลิเคชั่นโดยสามารถใช้เป็นกระดานส่วนตัว สามารถวางปุ่มกดสวิตช์ไว้ใช้สำหรับควบคุมอุปกรณ์ หรือวางหน้าปัดเพื่อแสดงผลข้อมูลต่าง ๆ ที่ได้จากอุปกรณ์ เช่น เซ็นเซอร์ในระบบ IoT นอกจากนี้ยังสามารถแสดงผลเป็นกราฟได้ ส่วนหน้ากระดานหรือ Dashboard นั้น สามารถปรับแต่งได้โดยง่าย เพียงแค่ป้อนข้อมูลเข้าหรือกำหนดค่าสั่งก็สามารถทำงานได้ โดยที่ผู้ใช้ไม่จำเป็นต้องเขียน HTML เว็บเพจเองและที่สำคัญคือข้อมูลนั้นมีการอัพเดทเป็นค่าเวลาจริง มีความเสถียรและเชื่อถือได้ และเป็นซอฟต์แวร์ที่เปิดเผย ซึ่งทำให้นักพัฒนาสามารถต่อยอดให้ดียิ่งขึ้นได้อีกด้วย การแสดงผลผ่านหน้าการจัดการของ Freeboard โดยใช้ประเภทของกระดานข้อมูลเป็น Gauge และ FeedView แสดงดังรูปที่ 2.15



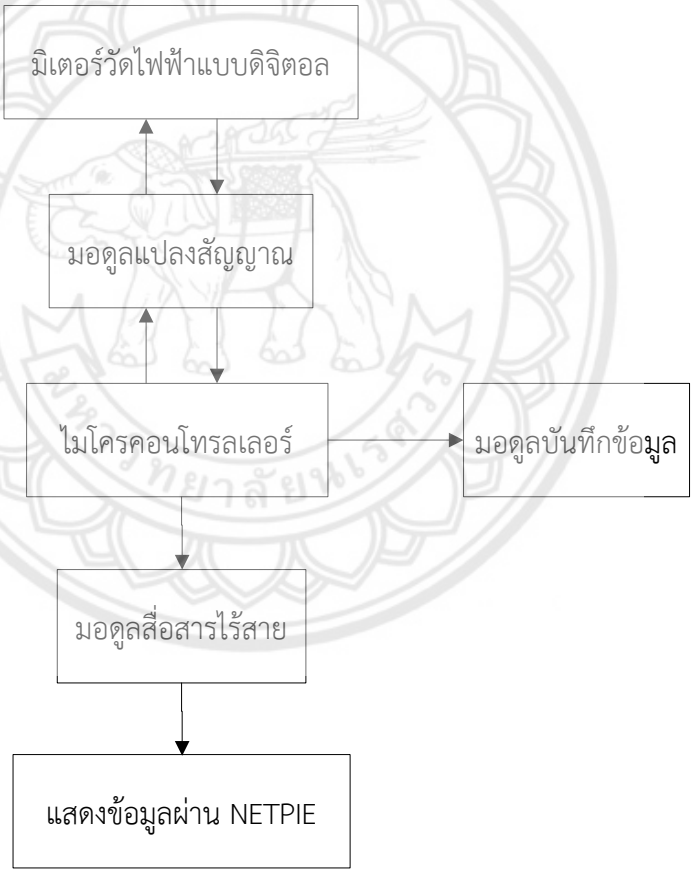
รูปที่ 2.15 การแสดงผลผ่านหน้าการจัดการของ Freeboard

บทที่ 3
ขั้นตอนการดำเนินการ

จากการศึกษาหลักการและทฤษฎีที่เกี่ยวข้องต่าง ๆ ในบทที่ 2 นำมาสู่การเขียนโปรแกรมสำหรับแสดงผลค่าปริมาณทางไฟฟ้า ได้แก่ แรงดัน, กระแส, กำลังไฟฟ้า, ค่าประกอบกำลัง และค่าพลังงานไฟฟ้า ออกผ่านทาง NETPIE พร้อมทั้งจัดเก็บลงในการ์ดหน่วยความจำ ซึ่งจะกล่าวรายละเอียดในหัวข้อต่อไป

3.1 ขั้นตอนการทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้า

ขั้นตอนการทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้า แสดงได้ดังรูปที่ 3.1

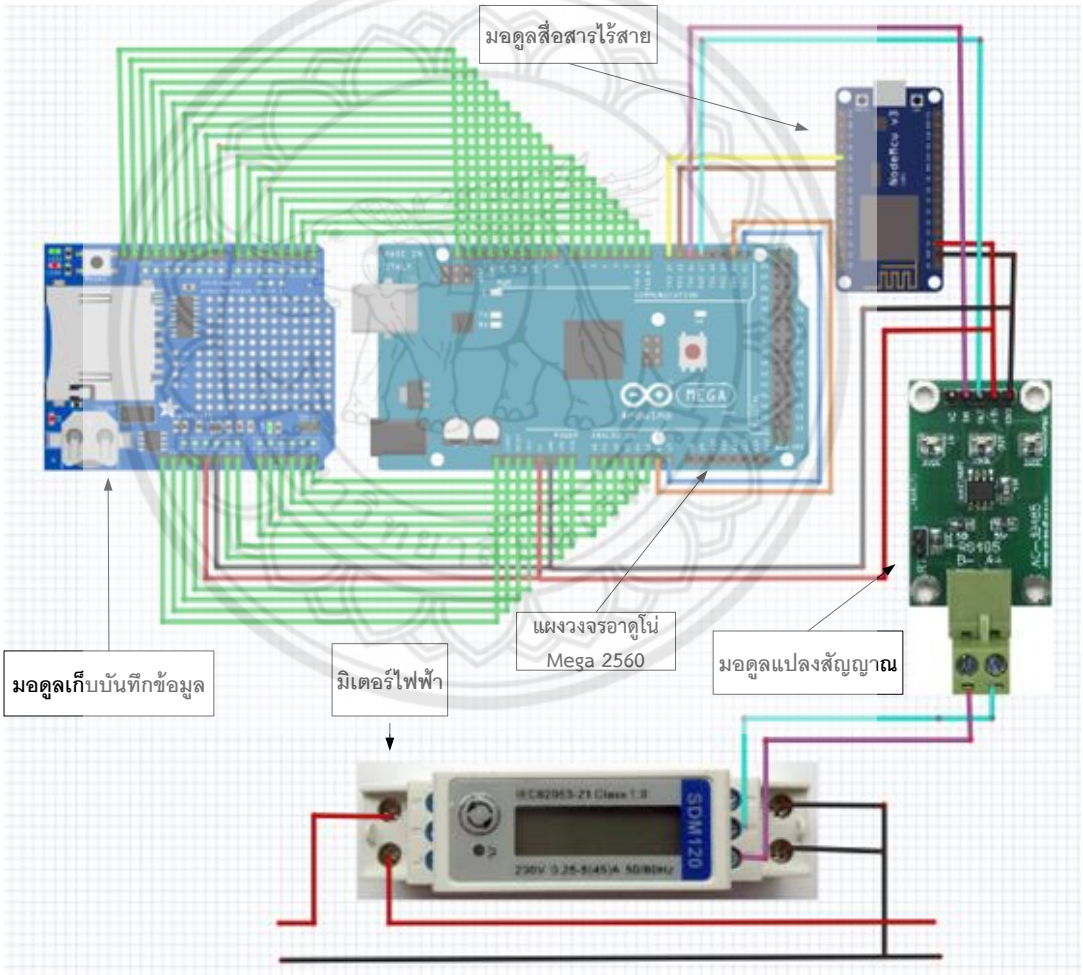


รูปที่ 3.1 ขั้นตอนการทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้า

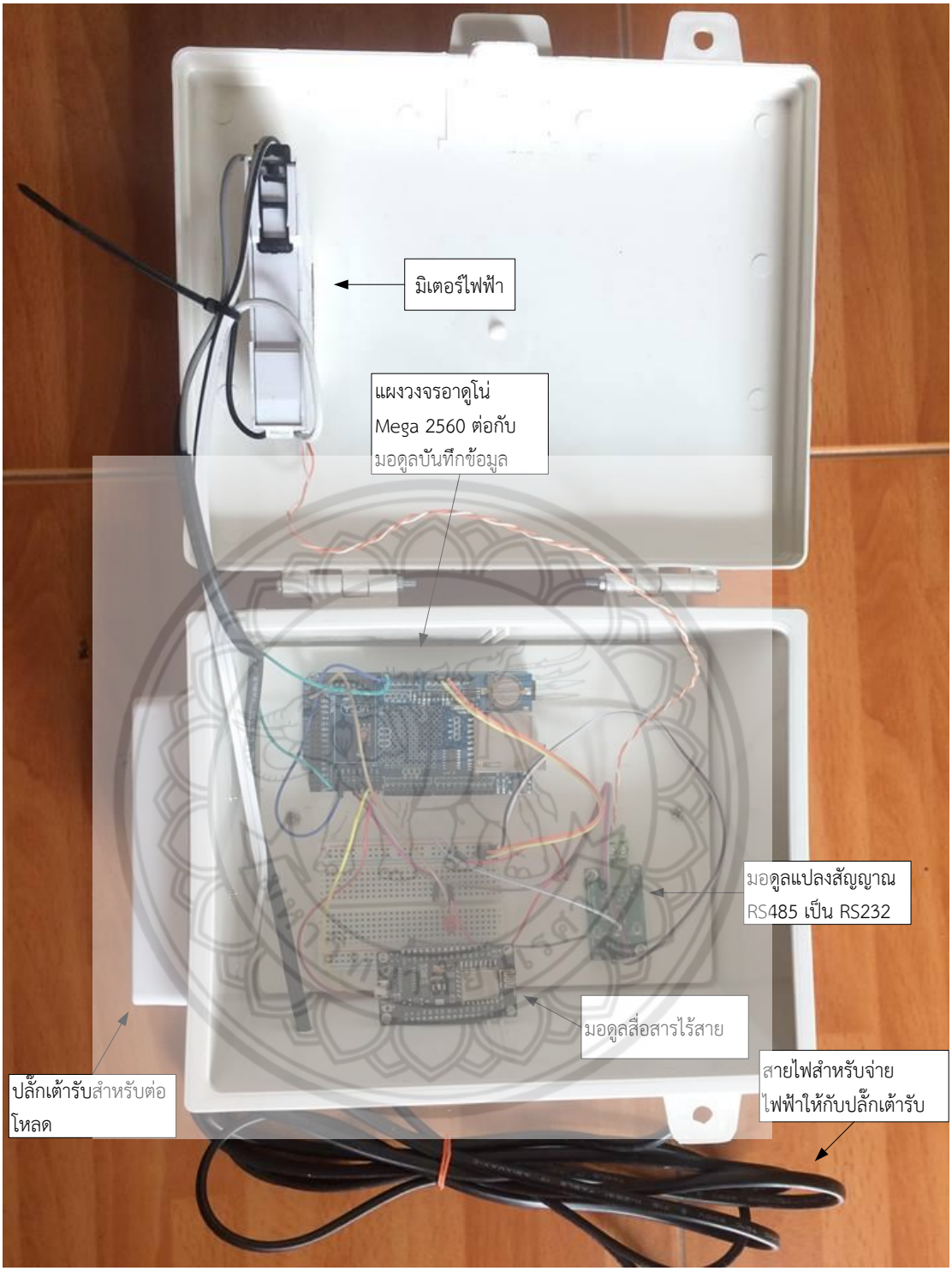
การทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้า สามารถอธิบายได้ดังนี้

1. นำสัญญาณมอดบัส โปรโตคอล แบบ RTU ที่ส่งมาจากมาตรวัดไฟฟ้าในรูปแบบเลขฐานสิบหก มาแปลงสัญญาณที่มอดูลแปลงสัญญาณให้เป็นการสื่อสารแบบอนุกรม เพื่อที่ไมโครคอนโทรลเลอร์และมาตรวัดไฟฟ้าสามารถสื่อสารรับค่าข้อมูลได้
2. สัญญาณที่ได้จากมอดูลแปลงสัญญาณ ถูกส่งไปประมวลผลโดยไมโครคอนโทรลเลอร์ เพื่อแสดงค่าพารามิเตอร์ต่าง ๆ ตามที่ผู้ดำเนินโครงงานสนใจ นอกจากนี้มีการบันทึกข้อมูลลงบนมอดูลบันทึกข้อมูล สำหรับการย้อนมาดูการใช้ไฟฟ้าที่ถูกบันทึกไว้ได้
3. จากนั้นข้อมูลปริมาณทางไฟฟ้าและพารามิเตอร์ต่าง ๆ ถูกส่งผ่านมอดูลสื่อสารไร้สายไปแสดงค่าบน NETPIE

การเชื่อมต่อวงจรแสดงค่าปริมาณทางไฟฟ้า แสดงได้ดังรูป 3.2 และ 3.3



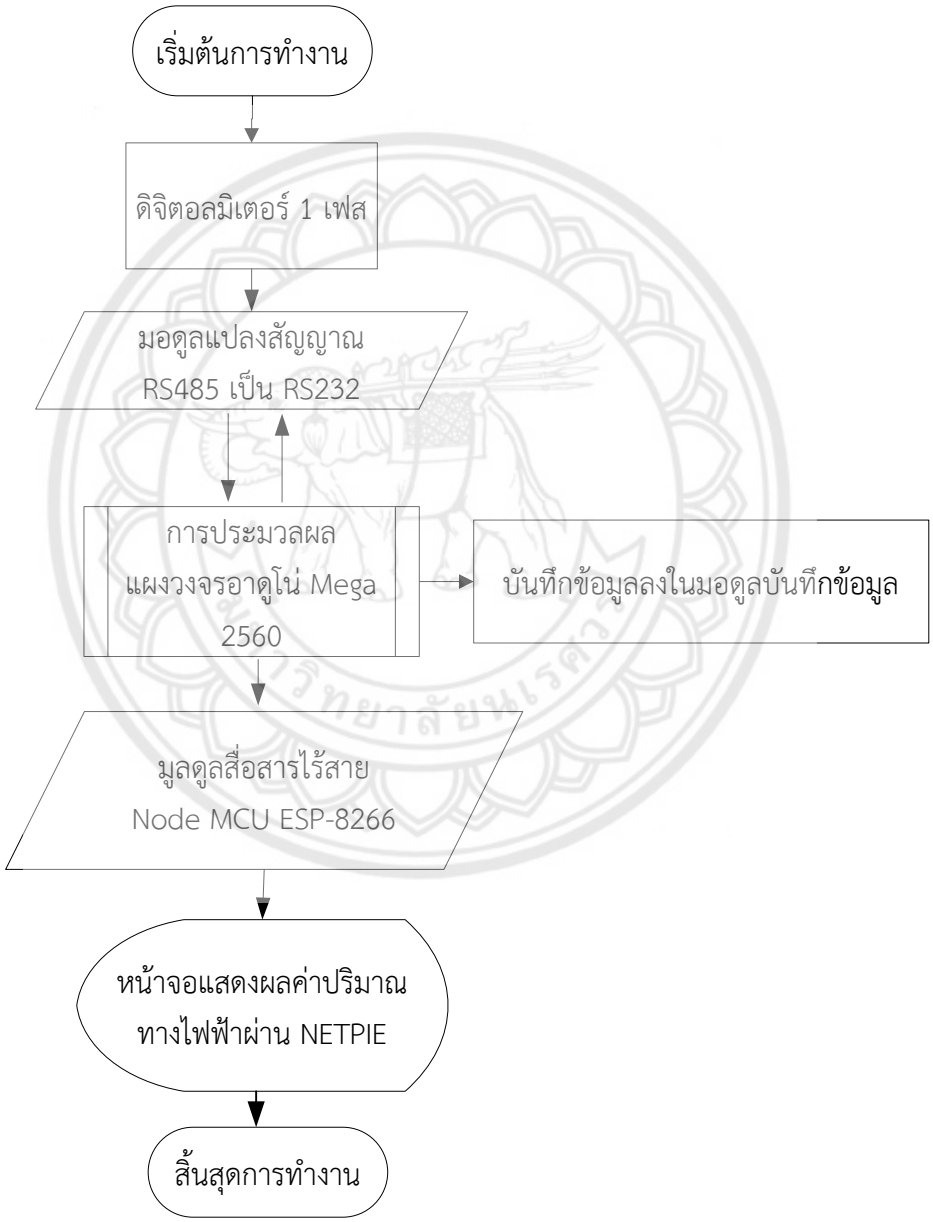
รูปที่ 3.2 การเชื่อมต่อวงจรแสดงค่าปริมาณทางไฟฟ้า



รูปที่ 3.3 แผงวงจรเมื่อต่อจริง

3.2 โครงสร้างการทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้าทั้งหมด

การทำงานของโครงสร้างวงจรแสดงค่าปริมาณทางไฟฟ้า ขั้นตอนแรกทำการต่อวงจรดังรูปที่ 3.2 เปิดโปรแกรมอาดูโน่ IDE ทำการเรียกค่าปริมาณทางไฟฟ้าจากดิจิทัลมิเตอร์โดยผ่านโมดูลแปลงสัญญาณ RS-485 จากนั้นค่าปริมาณทางไฟฟ้าที่ได้รับจากดิจิทัลมิเตอร์จะถูกส่งไปจัดเก็บที่มอดูลบันทึกข้อมูลในช่วงเวลาที่กำหนด ทุก ๆ 15 วินาที และ ถูกส่งไปยังมอดูลสื่อสารไร้สายเพื่อส่งต่อข้อมูลไปแสดงผลผ่านหน้าจอที่ NETPIE ดังรูปที่ 3.4



รูปที่ 3.4 โครงสร้างการทำงานของวงจรแสดงค่าปริมาณทางไฟฟ้าทั้งหมด

3.2.1 โครงสร้างการทำงานส่วนของพอร์ตสื่อสารระหว่างแผงวงอาดูโน่ กับ ดิจิทัลมิเตอร์

ดิจิทัลมิเตอร์ รุ่น SDM-120 ส่งสัญญาณข้อมูลแบบมอดบัสโปรโตคอลในโหมด RTU ประกอบด้วยข้อมูลแสดงตำแหน่งแอดเดรส 1 ไบต์, หมายเลขฟังก์ชัน 1 ไบต์, ข้อมูลที่ทำการรับส่งจำนวนมากสุดไม่เกิน 252 ไบต์ และรหัสตรวจสอบความถูกต้องของข้อมูลแบบการตรวจสอบด้วยส่วนซ้ำซ้อนแบบวน CRC (Cyclical Redundancy Checking) ขนาด 2 ไบต์ ค่า CRC นี้เป็นค่าที่คำนวณมาจากข้อมูลทุกไบต์ ไม่รวมบิต Start, Stop และ Parity Check โดยที่ตัว Slave ตัวที่ส่งข้อมูลออกมาจะสร้างรหัส CRC แล้วส่งตามท้ายไบต์ข้อมูลออกมา หลังจากนั้นเมื่อ Master ได้รับเฟรมข้อมูลและถอดข้อมูลออกจากเฟรมแล้วจะทำการคำนวณค่า CRC ตามสูตรเดียวกับ Slave เพื่อทำการเปรียบเทียบค่า CRC ทั้ง 2 ค่าว่าตรงกันหรือไม่ หากไม่ตรงกันแสดงว่าเกิดความผิดพลาดในการรับส่งข้อมูลในโหมด RTU การรับส่งข้อมูล 1 ไบต์ ไม่ว่าจะเป็นข้อมูลส่วนใดภายในเฟรมจะต้องทำการส่งบิตข้อมูลรวม 11 บิต คือ บิตเริ่มต้น (Start) 1 บิต, บิตข้อมูล 8 บิต, บิตตรวจสอบ Parity ของข้อมูล 1 บิตและบิตหยุด 1 บิต (Stop) 1 บิต หรือหากเลือกแบบไม่มีบิต Parity ก็จะเป็นแบบ Stop แทน 2 บิต สำหรับการกำหนดให้มีบิต Parity นั้น สามารถเลือกเป็นแบบคู่ (Even Parity) หรือคี่ (Odd Parity) ก็ได้ และหากต้องการออกแบบให้สอดคล้องกับอุปกรณ์ที่มีใช้กันทั่วไปมากที่สุด ควรเลือกแบบคู่เพราะเป็นกรณีพิเศษหนึ่งของการตรวจสอบด้วยส่วนซ้ำซ้อนแบบวน โดยที่สามารถปรับเปลี่ยนเป็นแบบคี่หรือไม่มีการตรวจสอบ Parity (No Parity) ได้ด้วย

ที่อยู่ข้อมูลมอดบัส โปรโตคอล แบบ RTUของมิเตอร์รุ่น SDM-120

การดึงที่อยู่ข้อมูลมอดบัส จำเป็นต้องมีการเรียกใช้ค่าเฉพาะของที่อยู่มอดบัสเพื่อระบุตำแหน่งในการนำค่าพารามิเตอร์ที่ต้องการแสดงออกมาได้ดังรูปที่ 3.5

Address Register	Input Register Parameter			Modbus Protocol Start Address Hex	
	Parameter	Units	Format	Hi byte	Lo byte
30001	Voltage	Volts	Float	00	00
30007	Current	Amps	Float	00	06
30013	Active power	Watts	Float	00	0C
30019	Apparent power	VA	Float	00	12
30025	Reactive power	VAr	Float	00	18
30031	Power factor	None	Float	00	1E
30071	Frequency	Hz	Float	00	46
30073	Import active energy	kWh	Float	00	48
30075	Export active energy	kWh	Float	00	4A
30077	Import reactive energy	kvarh	Float	00	4C
30079	Export reactive energy	kvarh	Float	00	4E
30343	Total active energy	kWh	Float	01	56
30345	Total reactive energy	Kvarh	Float	01	58

รูปที่ 3.5 ตารางที่อยู่ข้อมูลมอดบัส โปรโตคอล ในโหมด RTU ของมิเตอร์รุ่น SDM-120
ที่มา : http://www.smicrothai.com/smicro_3.php

รูปแบบข้อมูลมอดบัส โปรโตคอล แบบ RTU จะมีโครงสร้างรหัสโปรโตคอลส่วนของการส่งข้อมูลเพื่อเรียกดูค่าปริมาณทางไฟฟ้าไฟฟ้าจากแผงวงจรอาดูโน่ไปยังดิจิทัลมิเตอร์ ดังตาราง 3.1

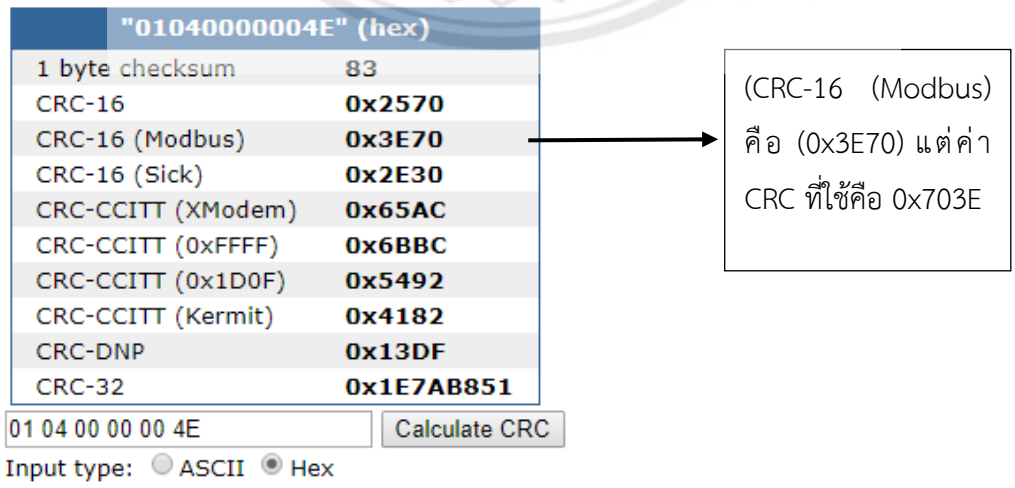
ตารางที่ 3.1 โครงสร้างรหัสโปรโตคอลส่วนของการส่งข้อมูลปริมาณทางไฟฟ้า

Address	Function	Data code				Checkout code	
01	04	00	00	00	4E	70	3E

การถอดรหัสโปรโตคอลจากตารางที่ 3.1

1. Address เป็นเลขฐานสิบหก จะพบว่าได้ค่าคือ 01 นำมาแปลงเป็นเลขฐานสิบ 1 บิต จะเป็น 1 บ่งบอกว่าเป็นอุปกรณ์ตัวที่ 1
2. Function เป็นเลขฐานสิบหก จะพบว่าได้ค่าคือ 04 นำมาแปลงเป็นเลขฐานสิบ 1 บิต จะเป็น 4 เป็นคำสั่งให้อ่านข้อมูล
3. Data code เป็นเลขฐานสิบหก โดย 2 ไบต์แรก คือ 00 00 บ่งบอกถึงจะเริ่มต้นเรียกค่าที่ Address 00 00 และ 2 ไบต์หลัง คือ 00 4E บ่งบอกถึงการเรียกค่าถึง Address 00 4E นั่นคือ จะเรียกค่าแสดงค่าแรงดันไฟฟ้าถึงค่าพลังงานไฟฟ้า ดังรูปที่ 3.5
4. Checkout code รหัสตรวจสอบความถูกต้องของข้อมูลแบบ CRC (Cyclical Redundancy Checking) ขนาด 2 ไบต์ ค่า CRC นี้เป็นค่าที่คำนวณมาจากข้อมูลทุกไบต์

เมื่อคำนวณการถอดรหัสเพื่อหาค่าได้แล้วจะมาคำนวณค่า CRC เลือก Input type : Hex แล้วนำไบต์ที่ 1-6 มาใส่ในช่อง Calculate CRC จะปรากฏค่าCRC แบบกลับไบต์ ที่บรรทัด (CRC-16 (Modbus) คือ 0x3E70) แต่ค่า CRC ที่ใช้คือ 0x703E แสดงดังรูปที่ 3.6



รูปที่ 3.6 โปรแกรมคำนวณ CRC ของการส่งโปรโตคอลปริมาณทางไฟฟ้า

จะได้รหัสโปรโตคอลทั้งหมด 8 ไบต์ รหัส 8 ไบต์นี้จะเป็นรหัสที่ส่งไปหามิเตอร์เพื่อให้มิเตอร์ส่งข้อมูลกลับมา แสดงดังรูปที่ 3.7

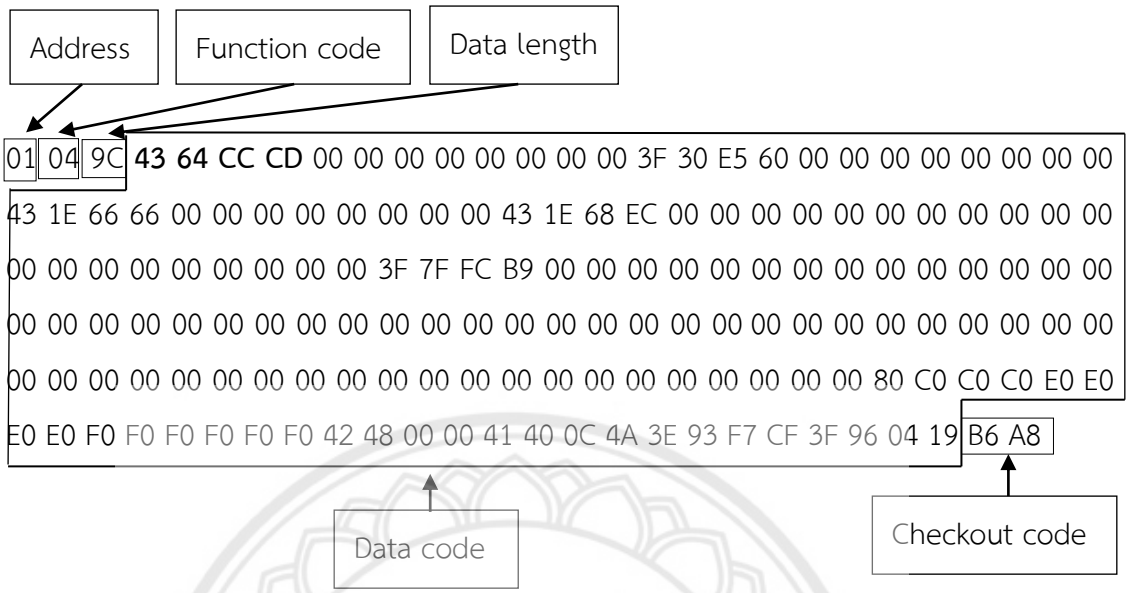
```
5/28/2018 22:04:31.568 [TX] - 01 04 00 00 00 4E 70 3E
5/28/2018 22:04:31.789 [RX] - 01 04 9C 43 64 CC CD 00 00 00 00 00 00 00 00 3F 30 E5 60 00 00 00 00 00 00 00 00 43 1E 66 66 00 00 00 00 00 00 00 00
43 1E 68 EC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3F 7F FC B9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 C0 C0 C0 E0 E0 E0 E0
F0 F0 F0 F0 F0 F0 42 48 00 00 41 40 0C 4A 3E 93 F7 CF 3F 96 04 19 B6 A8
```

รูปที่ 3.7 รหัสโปรโตคอลมิเตอร์ส่วนของมิเตอร์ส่งข้อมูลปริมาณไฟฟ้าและตอบกลับ

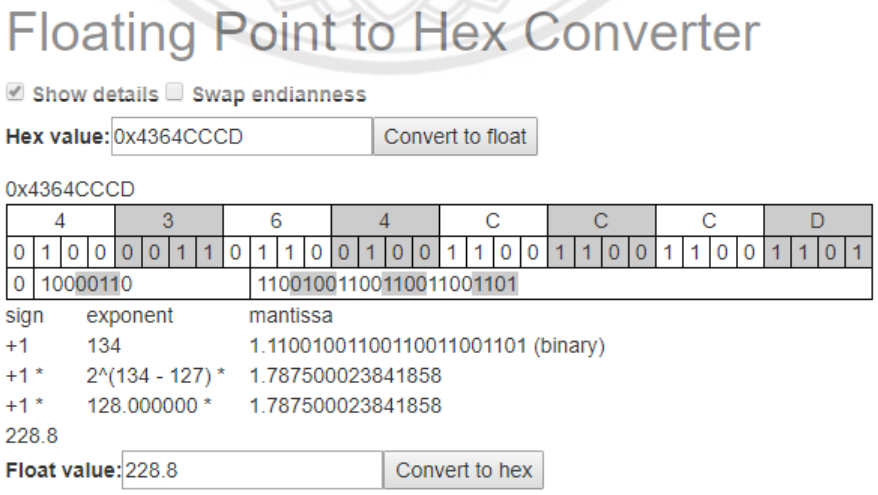
ตารางที่ 3.2 รหัสโปรโตคอลมิเตอร์ส่วนของมิเตอร์ส่งข้อมูลแรงดันไฟฟ้าตอบกลับมา

Address	Function code	Data length	Data	Checkout code	
01	04	9C	43 64 CC CD 00 00 00 00 00 00 00 00 3F 30 E5 60 00 00 00 00 00 00 00 00 43 1E 66 66 00 00 00 00 00 00 00 00 43 1E 68 EC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3F 7F FC B9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 80 C0 C0 C0 E0 E0 E0 E0 F0 F0 F0 F0 F0 F0 42 48 00 00 41 40 0C 4A 3E 93 F7 CF 3F 96 04 19	B6	A8

Receive : Data message เป็นรูปแบบการถอดรหัสโปรโตคอลที่มิเตอร์ส่งข้อมูลกลับมาแสดงดังตารางที่ 3.2



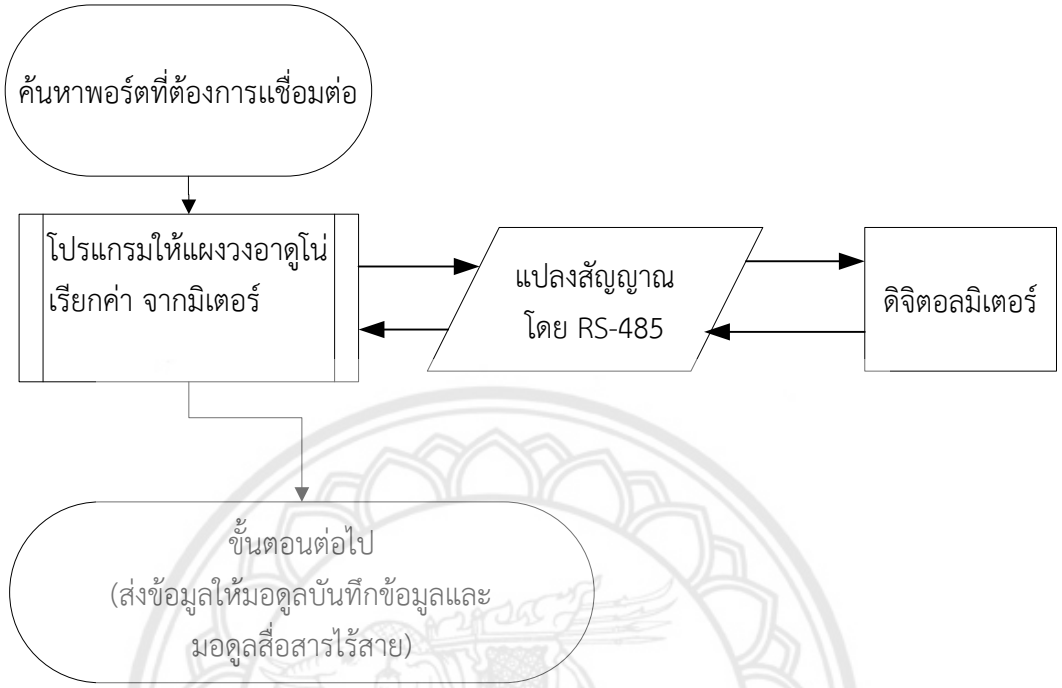
หลักการอ่านค่าปริมาณทางไฟฟ้าที่มิเตอร์ส่งข้อมูลกลับมาเป็นข้อมูลเลขฐานสิบหก 161 ไบต์ โดยไบต์ที่ 1 บ่งบอกว่าเป็นอุปกรณ์ตัวที่ 1, ไบต์ที่ 2 เป็นคำสั่งให้อ่านข้อมูล, ไบต์ที่ 3 เป็นเลขฐานสิบหก คือ 9C แปลงเป็นเลขฐานสิบจะได้ 156 จะบ่งบอกว่าข้อมูลที่ได้มามีกี่ไบต์ นั่นคือข้อมูลมี 156 ไบต์ใน Data code การจะนำข้อมูลที่ Data code มาคำนวณหาค่าปริมาณทางไฟฟ้าโดยใช้ไบต์ที่ 4-7 เท่ากับค่าแรงดันไฟฟ้าคือ 43 64 CC CD มาแปลงเป็นเลขฐานสิบที่เป็นทศนิยมได้ 228.8 โวลต์ แสดงดังรูปที่ 3.8 และ ไบต์ที่ 160-161 เป็นค่า CRC นี้เป็นค่าที่คำนวณมาจากข้อมูลทุกไบต์



รูปที่ 3.8 ค่าแรงดันไฟฟ้าที่นำมาแปลงเป็นเลขฐานสิบ

ที่มา : https://gregstoll.dyndns.org/~gregstoll/floattohex/

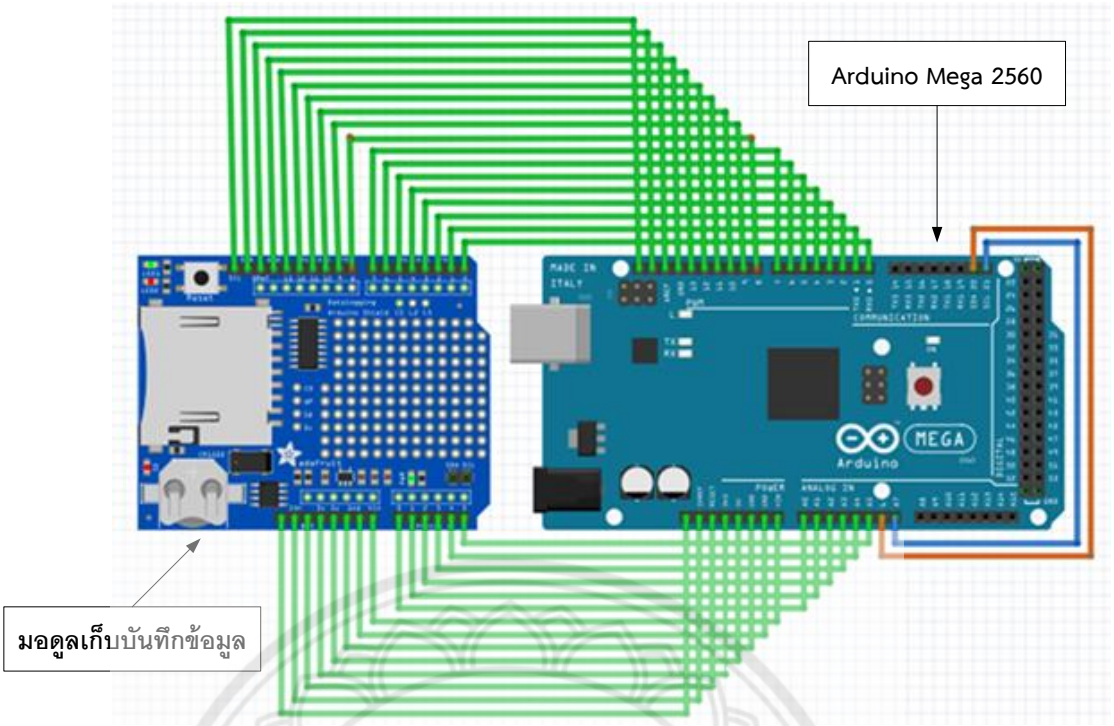
ส่วนประกอบของวงจรรับสัญญาณประกอบด้วย 3 ส่วน ได้แก่ ชุดแปลงสัญญาณทางไฟฟ้า ชุดประมวลผลการทำงาน และดิจิทัลมิเตอร์ รุ่น SDM-120 สามารถแสดงได้ดังรูปที่ 3.9



รูปที่ 3.9 พอร์ตสื่อสารระหว่างแผงวงจรอาดูโน่ กับ และติจิตอลมิเตอร์

3.2.2 โครงสร้างสื่อสารระหว่างแผงวงจรอาดูโน่ กับ มอดูลบันทึกข้อมูล

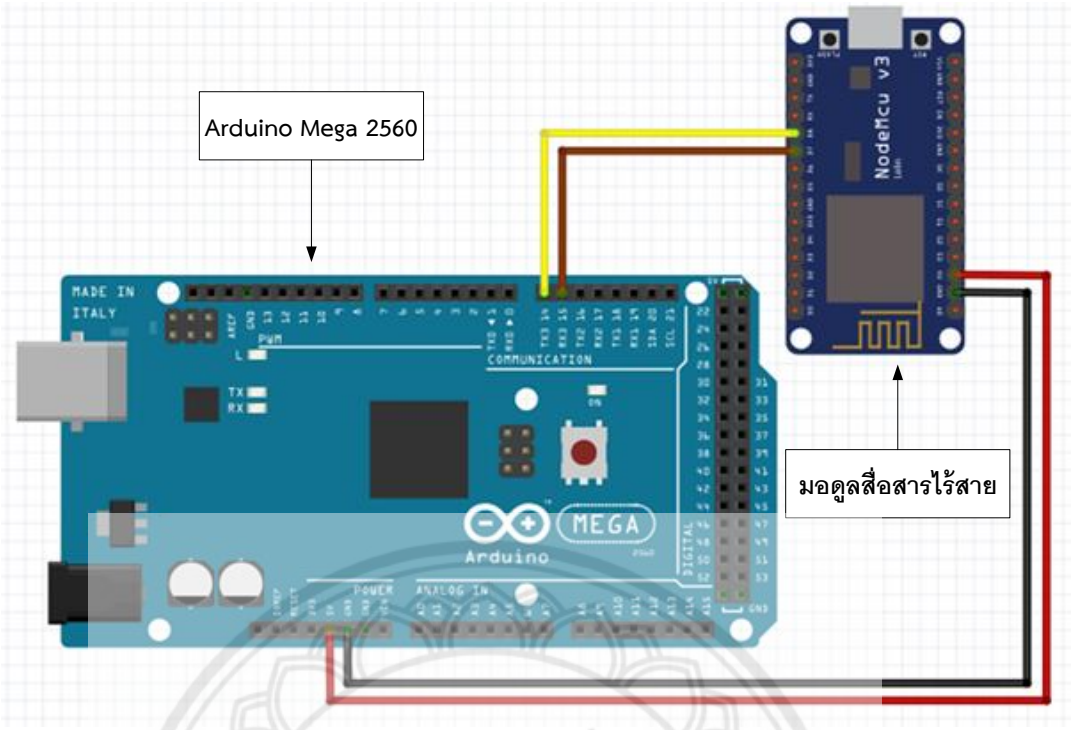
ส่วนจัดเก็บบันทึกข้อมูลนี้จะใช้ มอดูลบันทึกข้อมูลเสียบเข้ากับแผงวงจรอาดูโน่ และทำการโปรแกรมเพื่อบันทึกข้อมูลลงในการ์ดหน่วยความจำ โดย บันทึกข้อมูลเวลา วัน เดือน ปี ตามเวลาปัจจุบัน คือตามเวลาในเครื่องคอมพิวเตอร์ค่าปริมาณทางไฟฟ้า ได้แก่ แรงดัน, กระแส, กำลังไฟฟ้า, ค่าประกอบกำลัง และค่าพลังงานไฟฟ้า การต่อระหว่างมอดูลบันทึกข้อมูลเข้ากับแผงวงจร อาดูโน่ แสดงดังรูปที่ 3.10



รูปที่ 3.10 การต่อระหว่างมอดูลบันทึกข้อมูลเข้ากับแผงวงจรอาดูโน่

3.2.3 โครงสร้างสื่อสารระหว่าง มอดูลสื่อสารไร้สาย กับ NETPIE

ส่วนการเชื่อมต่อระหว่าง มอดูลสื่อสารไร้สาย กับ NETPIE นี้ มอดูลสื่อสารไร้สายจะรับข้อมูลปริมาณทางไฟฟ้าจากดิจิทัลมิเตอร์ โดยใช้แผงวงจรอาดูโน่เป็นอุปกรณ์ส่งข้อมูล จากนั้นมอดูลสื่อสารไร้สายจะส่งข้อมูลปริมาณทางไฟฟ้าให้กับ NETPIE ผ่านระบบสัญญาณ WI-FI แสดงผลเป็นกราฟและส่วนของหน้ากระดานหรือ Dashboard ข้อมูลนั้นมีการอัพเดทที่เวลาจริงตอนนั้น การต่อระหว่างมอดูลสื่อสารไร้สาย กับ NETPIE แสดงดังรูปที่ 3.11



รูปที่ 3.11 การต่อระหว่างมอดูลสื่อสารไร้สาย กับ NETPIE

3.2.3.1 วิธีที่การใช้ NETPIE Feed ผ่านทางหน้าเว็บ NETPIE

ก. เข้าสู่หน้าเว็บ https://netpie.io/#contact โดยเลือก FEEDS จากเมนู RESOURCES แสดงดังรูปที่ 3.12

รูปที่ 3.12 รูปแสดงการเข้าใช้งาน Feed จากเมนู Resources

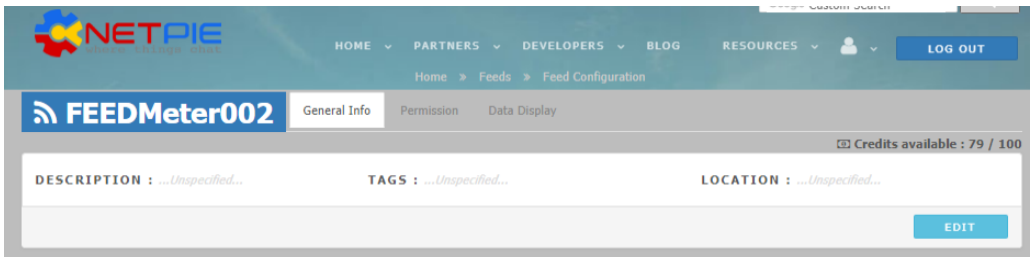
ข. สร้าง Feed ใหม่ หรือเพิ่ม Feed ด้วยการคลิกที่เครื่องหมาย + ที่มุมบนของ Feed แสดงดังรูปที่ 3.13

รูปที่ 3.13 รูปแสดงการสร้างหรือเพิ่ม Feed ใหม่

ค. ตั้งชื่อ Feed โดยที่ชื่อนี้จะต้องไม่ซ้ำกับที่เคยมีมาก่อน และห้ามซ้ำกับของผู้ใช้อื่น เมื่อตั้งชื่อแล้วให้คลิก CREATE เพื่อสร้าง Feed แสดงดังรูปที่ 3.14

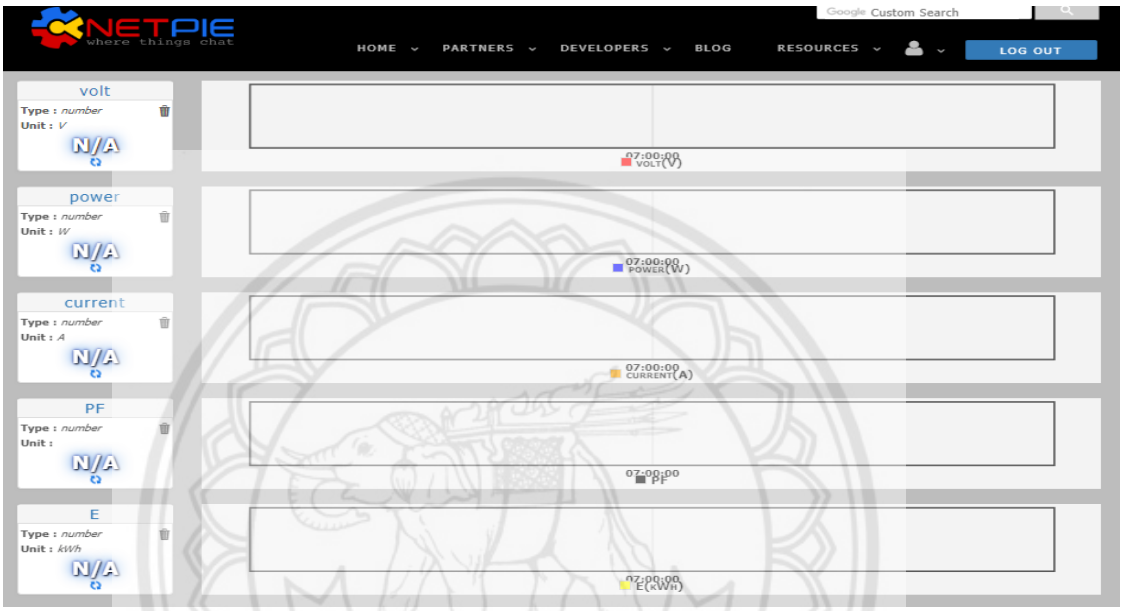
รูปที่ 3.14 รูปแสดงการตั้งชื่อ Feed

ง. เมื่อสร้าง Feed สำเร็จ จะเข้าสู่หน้าตั้งค่า Feed ตามภาพข้างล่าง ในตัวอย่างนี้ ได้สร้างสร้าง Feed ที่มีชื่อว่า “FEEDMete002” ซึ่งจะไปปรากฏในหน้ารวม Feed เมื่อเราเข้ามาในครั้งต่อไปเราสามารถกลับมาที่หน้าตั้งค่านี้ได้อีกจากหน้ารวมของ Feed โดยคลิกที่เครื่องหมายรูปประแจด้านขวามือหลังชื่อ Feed ที่ต้องการ แสดงดังรูปที่ 3.15



รูปที่ 3.15 รูปแสดงหน้าตั้งค่า Feed

จ. เนื่องจาก Feed ที่สร้างขึ้นใหม่นี้ยังไม่สามารถรับค่าใด ๆ เข้ามาได้ จึงต้องสร้าง Field ขึ้นมารับข้อมูล ให้กดปุ่ม + ADD จะปรากฏหน้าต่างให้ตั้งค่าต่าง ๆ ในตัวอย่างนี้ เป็นการสร้าง Field จำนวน 5 Field ที่มีชื่อว่า Volt, Power, Current, PF และ E (หรือจะตั้งเป็นชื่ออะไรก็ได้) หากมีหน่วยก็ระบุไว้เพื่อการแสดงผลต่อไป โดย ณ ขณะนี้ยังรองรับเฉพาะข้อมูลชนิดตัวเลข (Number) เท่านั้น โดยในอนาคตมีอาจมีการพัฒนาเพิ่มเติมให้รองรับข้อมูลชนิดอื่น ๆ แสดงดังรูปที่ 3.16

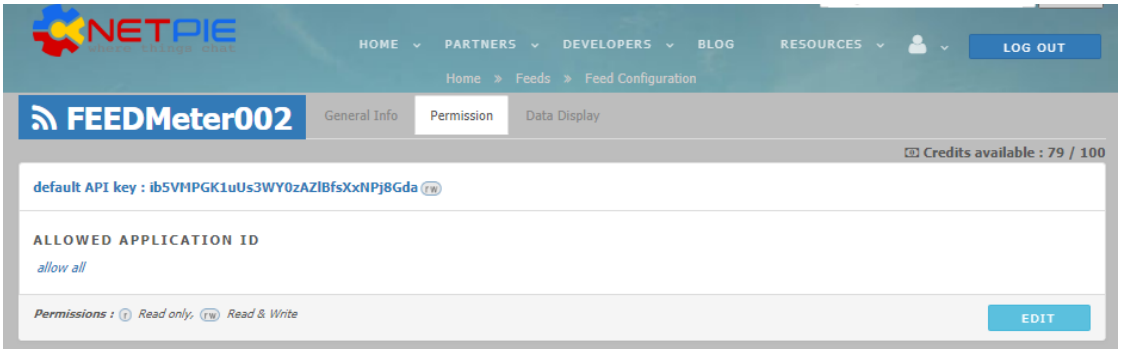


รูปที่ 3.16 รูปแสดงตัวอย่าง Feed ที่สร้างขึ้นด้วย 5 Field

ฉ. การกำหนดสิทธิ์ในการเข้าถึง Feed เนื่องจาก Feed แต่ละอันที่สร้างขึ้นมานั้น เป็น Resource อิสระที่ไม่ขึ้นกับ AppID ดังนั้นในการนำไปใช้งานจำเป็นต้องมีเรื่องของการกำหนดสิทธิ์ การให้สิทธิ์กับอุปกรณ์ที่จะเข้ามาอ่านหรือเขียน Feed มีด้วยกัน 2 วิธี ได้แก่

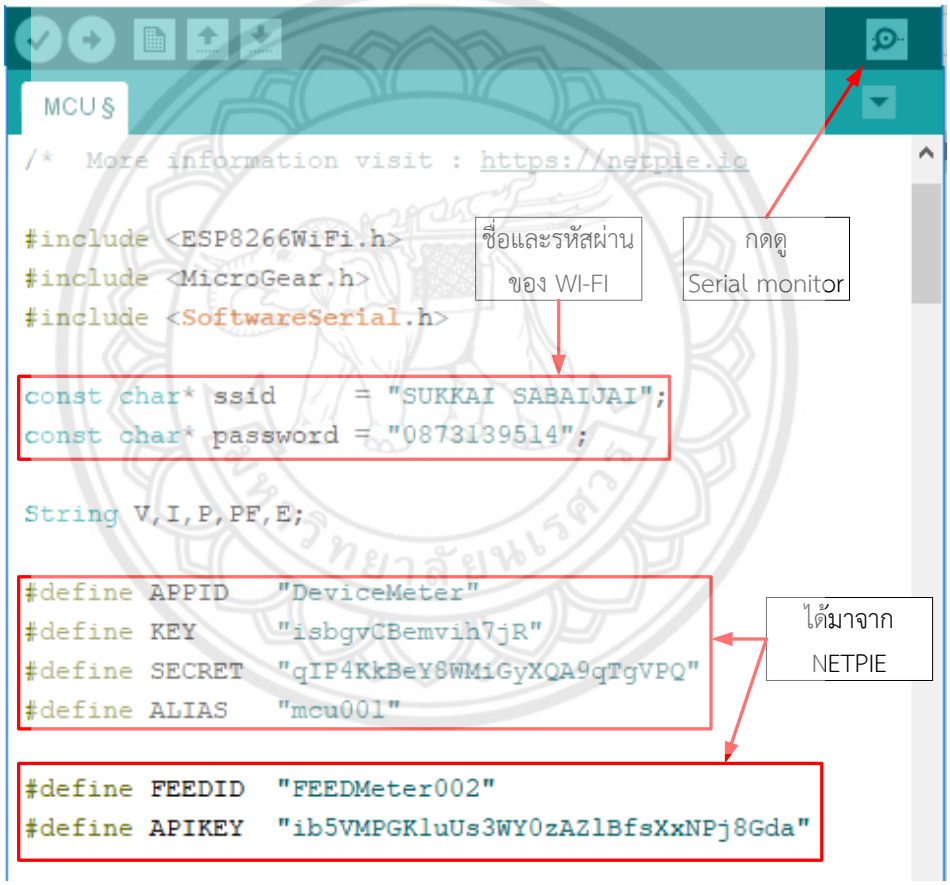
ฉ.1 การใช้ API Key ที่ปรากฏอยู่ในแท็บ Permission วิธีนี้จะอนุญาตให้ Client ที่มี API key สามารถอ่านเขียน Feed นี้ได้ วิธีการนำ API key ไปใช้

ฉ.2 การให้สิทธิ์กับ AppID ใช้ได้กับเฉพาะ AppID ของเจ้าของ Feed เท่านั้น วิธีนี้มีไว้เพื่ออำนวยความสะดวกในขั้นตอนการพัฒนาเนื่องจากเป็นการให้สิทธิ์กันผ่านเว็บ จึงไม่ต้องเข้าไปแก้โค้ดโปรแกรมการให้สิทธิ์กับ AppID จะทำให้ทุกอุปกรณ์ใน AppID นั้นมีสิทธิ์อ่านหรือเขียน Feed นี้เหมือนกันหมด การตั้งค่าทำได้โดยคลิกที่แท็บ Permission และคลิกที่ EDIT จากนั้นพิมพ์ AppID ที่ต้องการให้สิทธิ์ หรือเลือกจากเมนู Drop-Down และคลิก SAVE ดังรูปที่ 3.17 เป็นการให้สิทธิ์กับ AppID ที่ชื่อ “allow all” ดังนั้นทุกอุปกรณ์ภายใต้ AppID นี้ จะสามารถอ่านเขียน Feed ได้โดยอัตโนมัติ การอนุญาติแบบนี้จะเป็นการแจกสิทธิ์ในรูปแบบเดียวกับการนำ Default API key ของ Feed ไปใช้



รูปที่ 3.17 รูปแสดงการให้สิทธิ์การเข้าถึง Feed กับ AppID

ช. การนำ AppID ไปใช้งานร่วมกับ มอดูลสื่อสารไร้สาย ดังรูปที่ 3.18 และ 3.19



รูปที่ 3.18 นำ App ID ไปใช้งานร่วมกับมอดูลสื่อสารไร้สาย

รูปที่ 3.19 แสดงหน้าต่าง Serial Monitor

3.2.3.2 วิธีที่การใช้ NETPIE Freeboard ผ่านทางหน้าเว็บ NETPIE

ก. เข้าสู่ระบบ NETPIE Account แล้วไปที่เมนู RESOURCES แล้วเลือกที่ FREEBOARDS แสดงดังรูปที่ 3.20

รูปที่ 3.20 รูปแสดงการเข้าใช้งาน Freeboard จากเมนู Resources

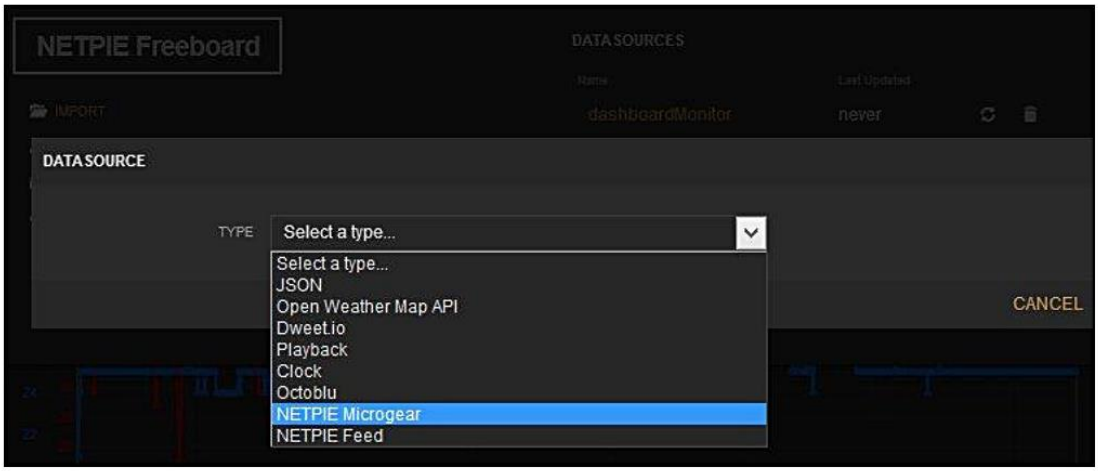
ข. คลิกเครื่องหมาย + เพื่อสร้าง Freeboard ขึ้นมาใหม่

ค. ตั้งชื่อ Freeboard แล้วกดปุ่ม CREATE

ง. ADD เป็นเมนูสำหรับเพิ่ม Datasource ที่เป็นแหล่งข้อมูลที่จะเชื่อมต่อเพื่อดึงออกมาแสดง แสดงดังรูปที่ 3.21

รูปที่ 3.21 รูปแสดงการเพิ่ม Datasource

จ. ด้านล่าง DATASOURCES คลิกที่ ADD จะปรากฏ Datasource Type ชนิดต่างๆ ให้เลือกเป็น NETPIE Microgear แสดงดังรูปที่ 3.22



รูปที่ 3.22 แสดงการเลือกประเภทของ Datasource

ฉ. ใส่ข้อมูลสำหรับ Datasource ซึ่งประกอบด้วย

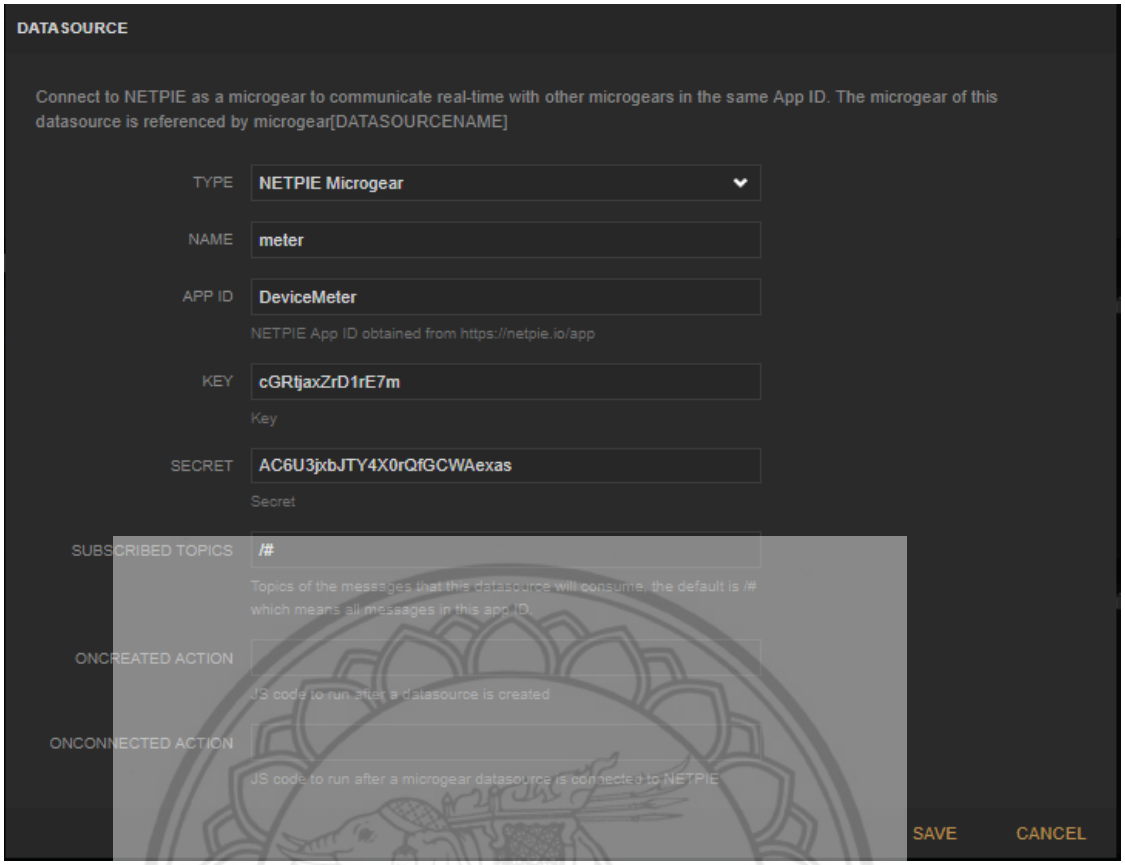
ฉ.1 NAME คือ ชื่อเรียก Datasource ที่ใช้อ้างอิง (ไม่เกิน 16 ตัวอักษร) ในตัวอย่างในภาพด้านล่างคือ meter

ฉ.2 APP ID คือ App ID ที่ได้สร้างผ่านหน้าในตัวอย่างในภาพด้านล่างคือ Device Meter

ฉ.3 KEY คือ Key ที่ได้จากการสร้าง App Key บนเว็บ NETPIE

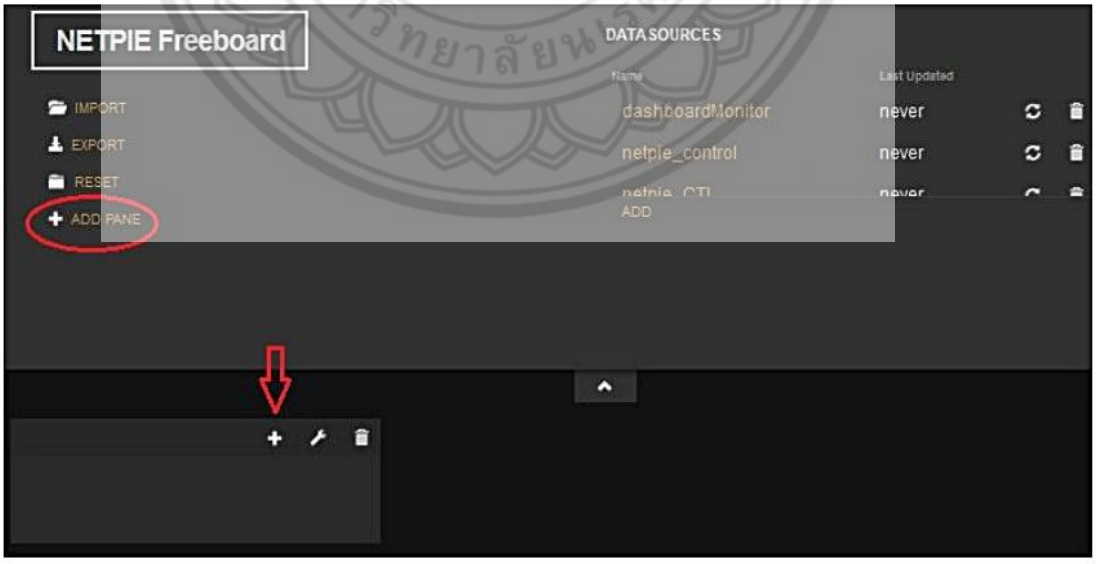
ฉ.4 SECRET คือ Secret ของ Key บนเว็บ NETPIE

ฉ.5 SUBSCRIBED TOPIC คือ Topic ที่ใช้สำหรับการรับส่งข้อมูลที่อยู่ภายใน APPID นั้น ๆ กรณีนี้ใช้เป็น /# มีความหมายว่า รับข้อความจากทุก Topic



รูปที่ 3.23 แสดงการระบุข้อมูลของ Datasource

ช. เพิ่ม Panel สำหรับสร้าง Widget ด้วยการคลิกที่ ADD PANE จะปรากฏ Panel เพิ่มขึ้นมาด้านล่าง แสดงดังรูปที่ 3.24

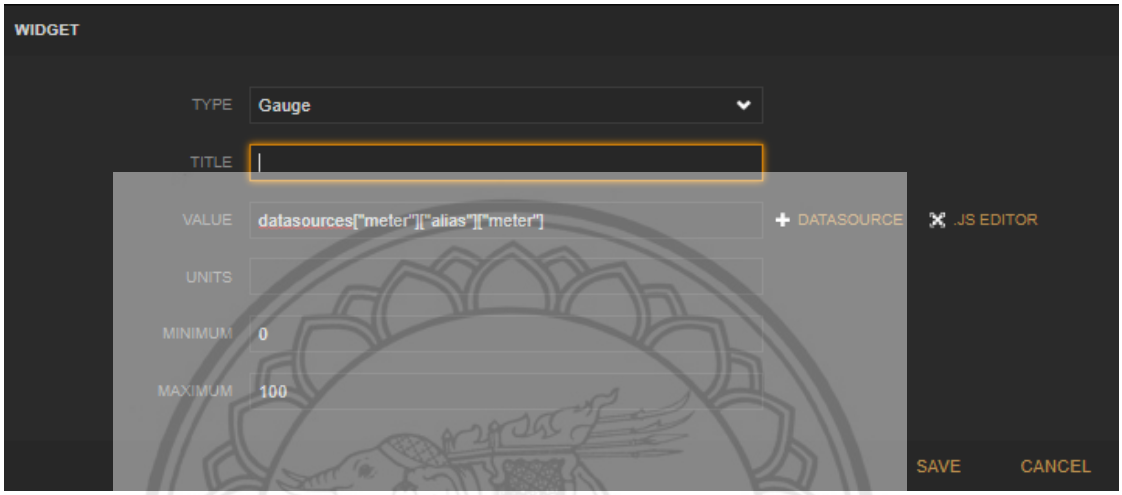


รูปที่ 3.24 แสดงการเพิ่ม Panel สำหรับสร้าง Widget

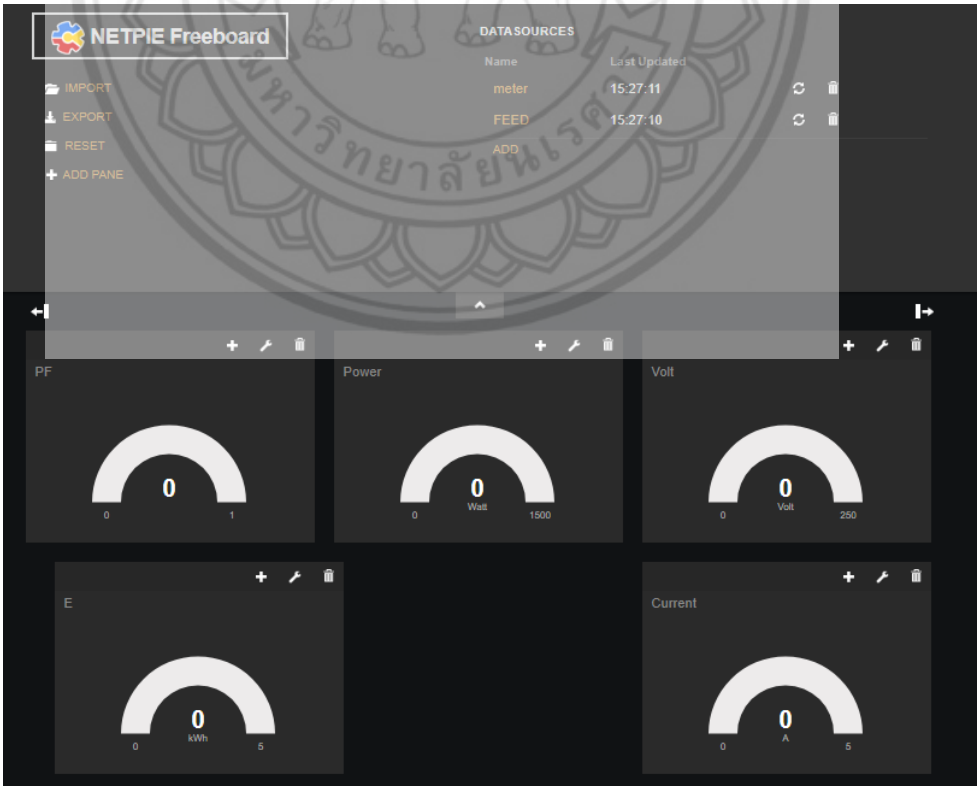
ซ. เพิ่ม Widget บน Panel ที่สร้างขึ้นใหม่โดย คลิกที่เครื่องหมาย + และเลือกชนิดของ Widget เช่น Gauge กรอกข้อความลงไปดังรูปที่ 3.25 แล้วกด Save จะได้หน้าเว็ปดังรูปที่ 3.26

ซ.1 TITLE ตั้งชื่อให้ Widget นี้

ซ.2 VALUE ให้คลิกที่ + DATASOURCE จะขึ้นรายชื่อของ Datasource ที่ได้สร้างไว้ และสามารถเลือกชื่อที่มีอยู่



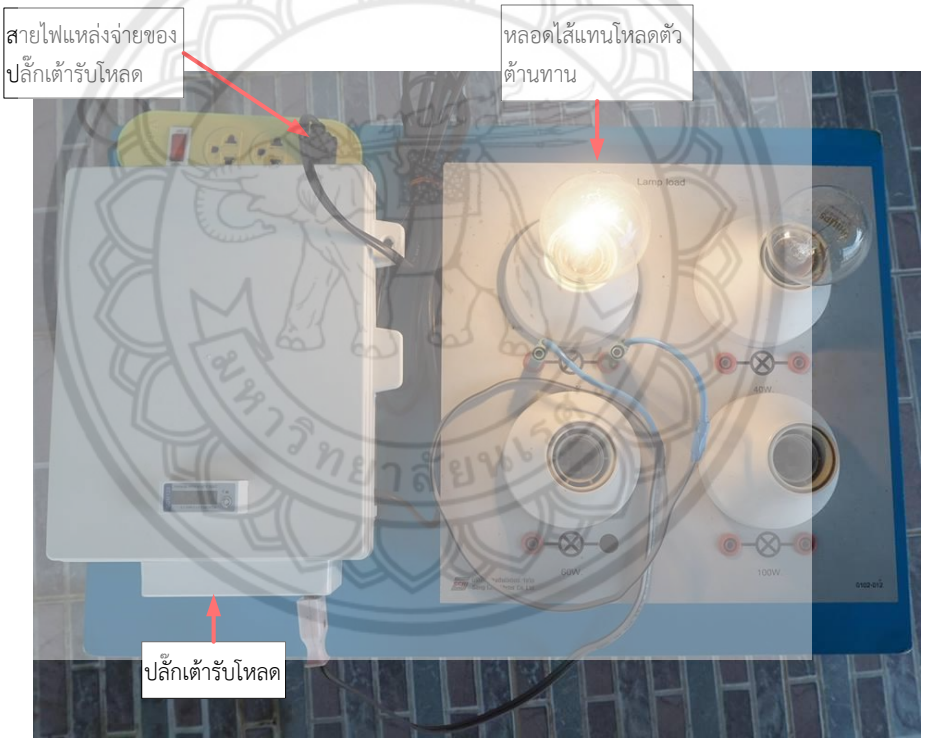
รูปที่ 3.25 แสดงการเลือกประเภทของ Widget



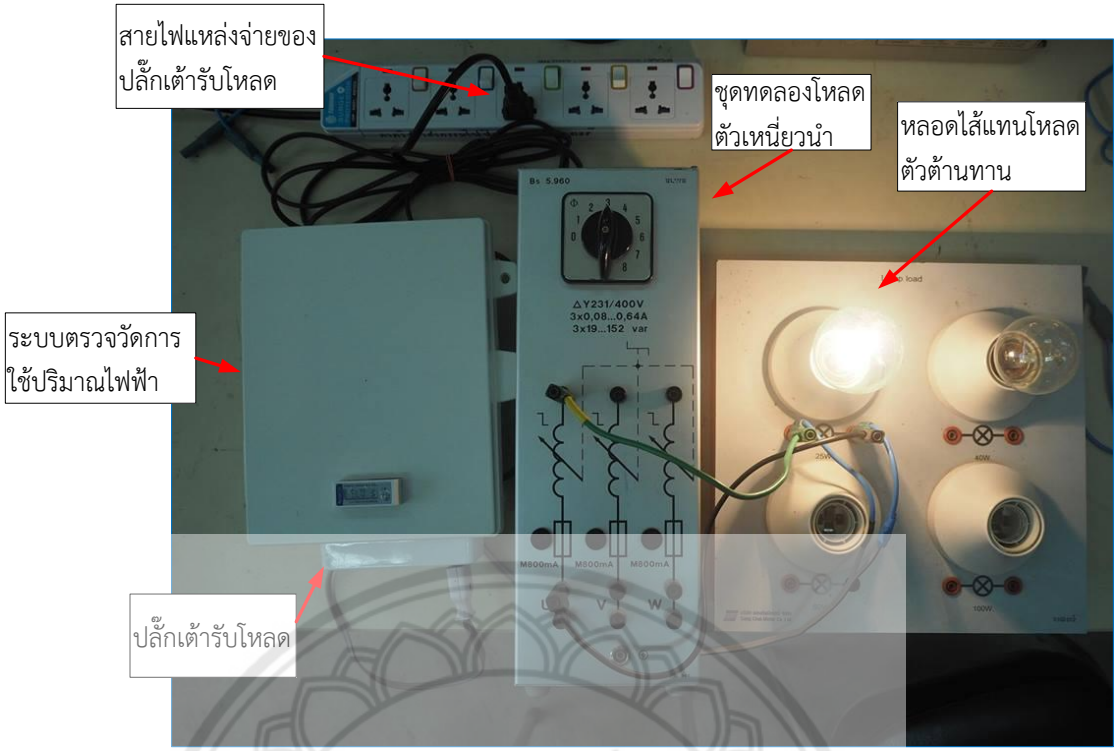
รูปที่ 3.26 แสดงหน้ากระดานข้อมูลของ NETPIE Freeboard

3.3 กระบวนการทดลอง

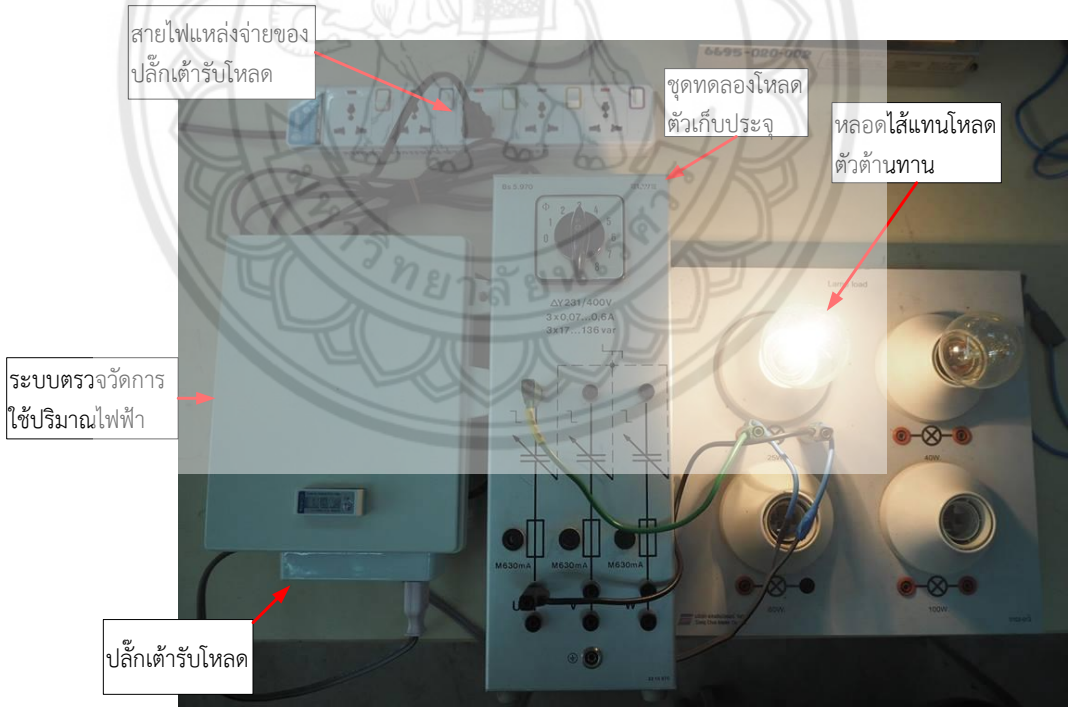
ในการทดลองจะใช้ภาวะโหลดที่เป็นอุปกรณ์เครื่องใช้ไฟฟ้าภายในบ้านเรือน เช่น อุปกรณ์ชาร์จมือถือ, พัดลม, หลอดไส้ขนาด 60 วัตต์, ชุดทดลองโหลดตัวเหนี่ยวนำขนาด 1.97 เฮนรี และ ชุดทดลองโหลดตัวเก็บประจุขนาด 2.38 ไมโครฟารัด เป็นต้น การทดลองกับระบบตรวจวัดการใช้พลังปริมาณไฟฟ้าด้วยดิจิทัลมิเตอร์ผ่าน NETPIE จะทำโดยการนำภาระโหลดที่กล่าวข้างต้นเชื่อมต่อเข้ากับปลั๊กเต้ารับซึ่งต่ออยู่กับดิจิทัลมิเตอร์ รุ่น Eastron SDM120 เพื่อทำการวัดปริมาณไฟฟ้า การนำโหลดหลอดไส้แทนโหลดความต้านทานต่อกับระบบตรวจวัดการใช้พลังงานไฟฟ้าแสดงดังรูปที่ 3.27 การนำโหลดหลอดไฟและโหลดตัวเหนี่ยวนำต่อกับระบบตรวจวัดการใช้ปริมาณงานไฟฟ้าแสดงดังรูปที่ 3.28 และ การนำโหลดหลอดไฟและโหลดตัวเก็บประจุต่อกับระบบตรวจวัดการใช้ปริมาณไฟฟ้าแสดงดังรูปที่ 3.29 จะทำการทดลองเป็นเวลา7 วัน ส่วนของระบบตรวจวัดการใช้ปริมาณไฟฟ้าในรูปที่ 3.30 นั้น แผงวงจรอาดูโน่ กับ มอดูลสื่อสารไร้สายจะใช้แหล่งจ่ายแยกจากกัน



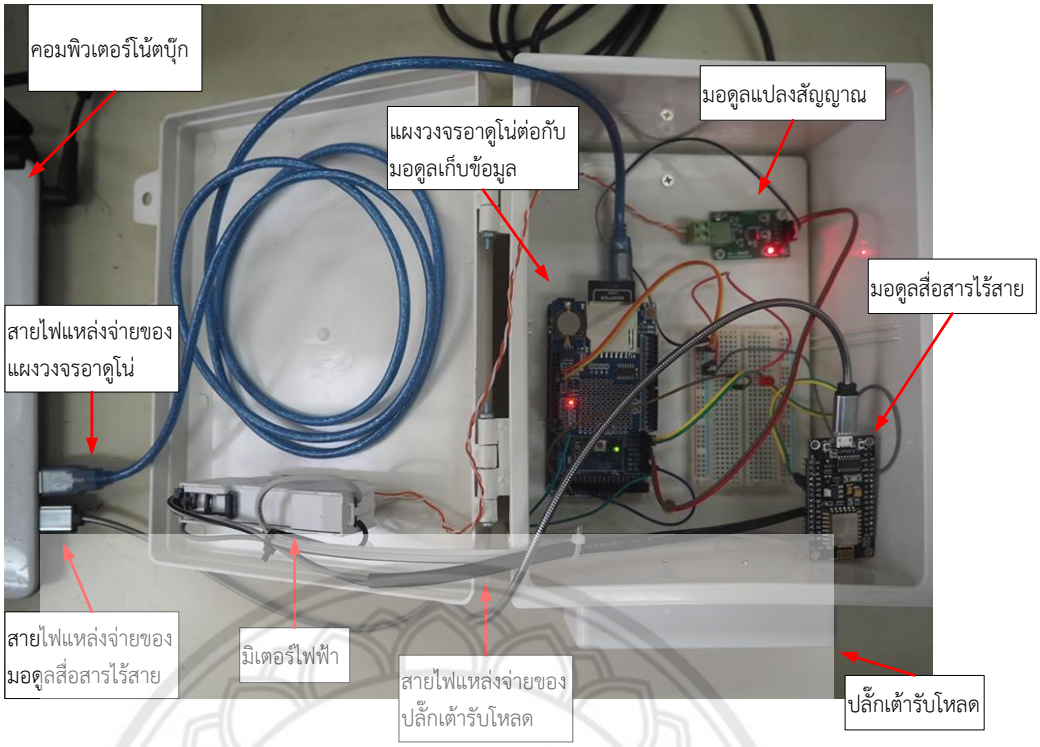
รูปที่ 3.27 การนำโหลดความต้านทานต่อกับระบบตรวจวัดการใช้พลังงานไฟฟ้าด้วยดิจิทัลมิเตอร์



รูปที่ 3.28 การนำโหลดหลอดไฟและโหลดตัวเหนี่ยวนำต่อกับระบบตรวจวัดการใช้ปริมาณไฟฟ้า



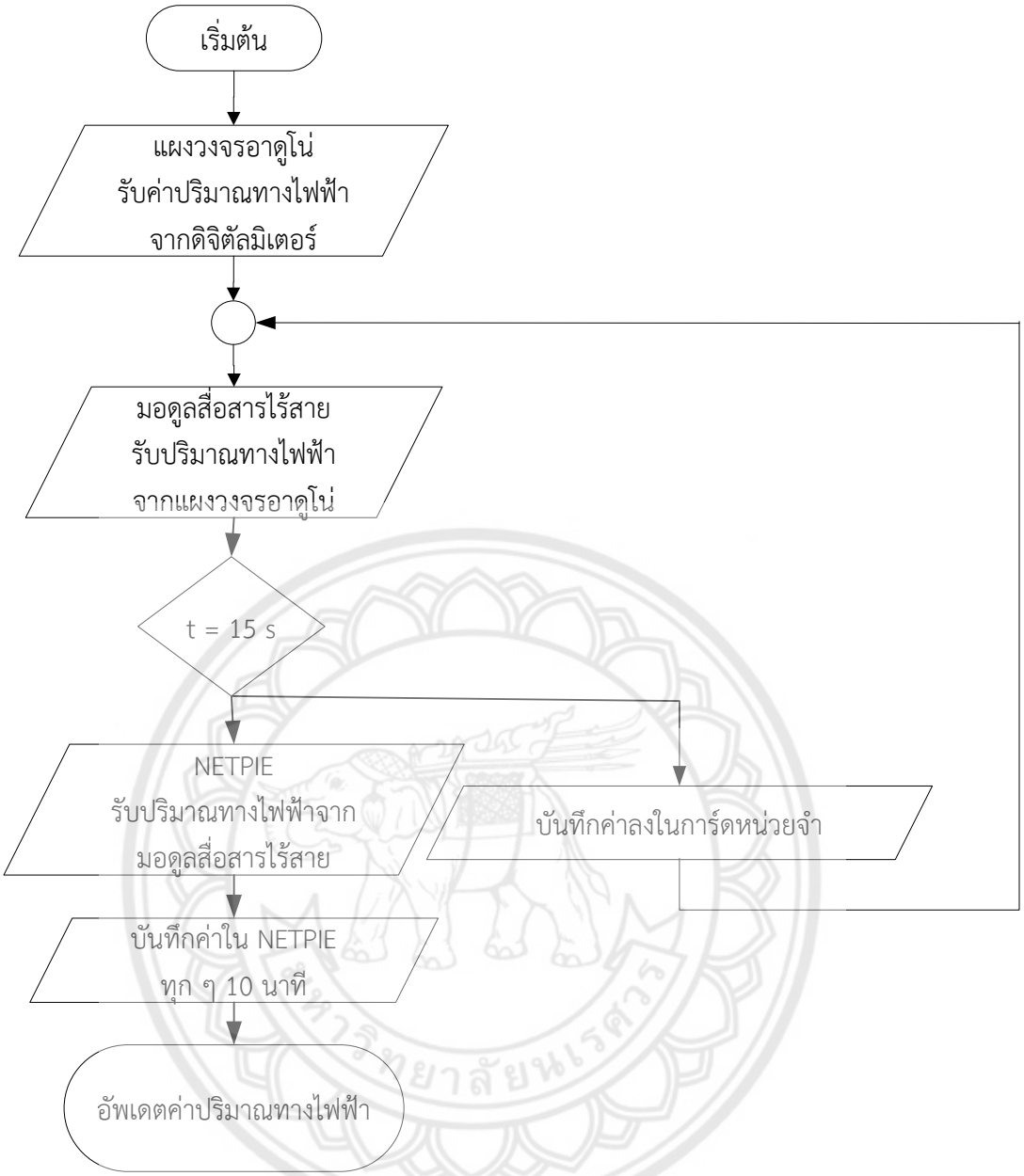
รูปที่ 3.29 การนำโหลดหลอดไฟและโหลดตัวเก็บประจุต่อกับระบบตรวจวัดการใช้ปริมาณไฟฟ้า



รูปที่ 3.30 ภายในระบบตรวจวัดการใช้พลังงานไฟฟ้าด้วยดิจิทัลมิเตอร์

3.4 การทำงานของโปรแกรม

การทำงานเริ่มจากแผงวงจรอาดูโน่ รับค่าปริมาณทางไฟฟ้าจากดิจิทัลมิเตอร์ แล้วมอดูลสื่อสารไร้สายกับมอดูลบันทึกข้อมูลรับค่าปริมาณทางไฟฟ้าจากแผงวงจรอาดูโน่ มอดูลบันทึกข้อมูลจะบันทึกค่าทุก ๆ 15 วินาที และ NETPIE รับค่าปริมาณทางไฟฟ้าจากมอดูลสื่อสารไร้สายแล้วบันทึกค่าไว้ที่ NETPIE ทุก ๆ 10 นาที การทำงานของโปรแกรมแสดงดังรูปที่ 3.31



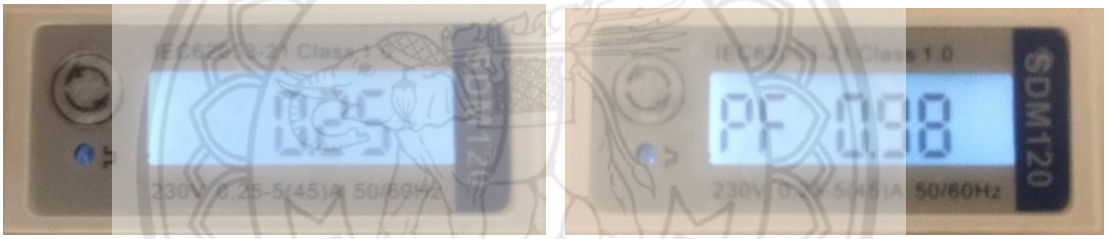
รูปที่ 3.31 แผนภาพแสดงภาพรวมการทำงานของโปรแกรม

บทที่ 4
ผลการทดลอง

ในบทนี้จะเป็น การออกแบบชุดทดลองต่อชุดทดสอบเข้ากับโหลด เพื่อเก็บค่าปริมาณทางไฟฟ้า โดยการทดสอบแบ่งออกเป็น 2 รูปแบบ คือ รูปแบบการแสดงผลรูปกราฟผ่านทาง NETPIE และ รูปแบบของการเก็บข้อมูล รายละเอียดปลีกย่อยของผลการทดลอง มีดังต่อไปนี้

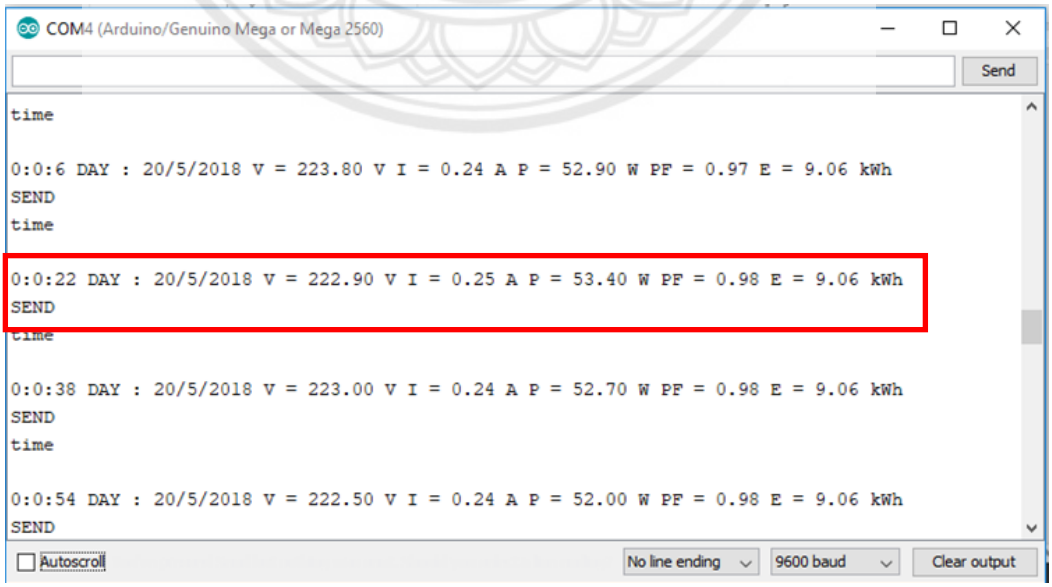
4.1 การแสดงผลรูปกราฟผ่านทาง NETPIE

จากการทดสอบการแสดงผลรูปกราฟผ่านทาง NETPIE จะได้ผลการแสดงค่าปริมาณทางไฟฟ้าเป็นรูปกราฟเส้นที่จะพล็อตทุก ๆ 10 นาที แสดงดังรูปที่ 4.4 และค่าปริมาณไฟฟ้านั้นจะบันทึกข้อมูลไว้ที่ NETPIE นอกจากนี้ยังแสดงผลเป็นหน้ากระดานข้อมูล ค่าปริมาณทางไฟฟ้านั้นมีการอัพเดทที่เวลาจริงตอนนั้น แสดงดังรูปที่ 4.5 โดยขึ้นอยู่การส่งข้อมูลของชุดทดสอบ



ก) ค่ากระแสไฟฟ้า ข) ค่าประกอบกำลังไฟฟ้า

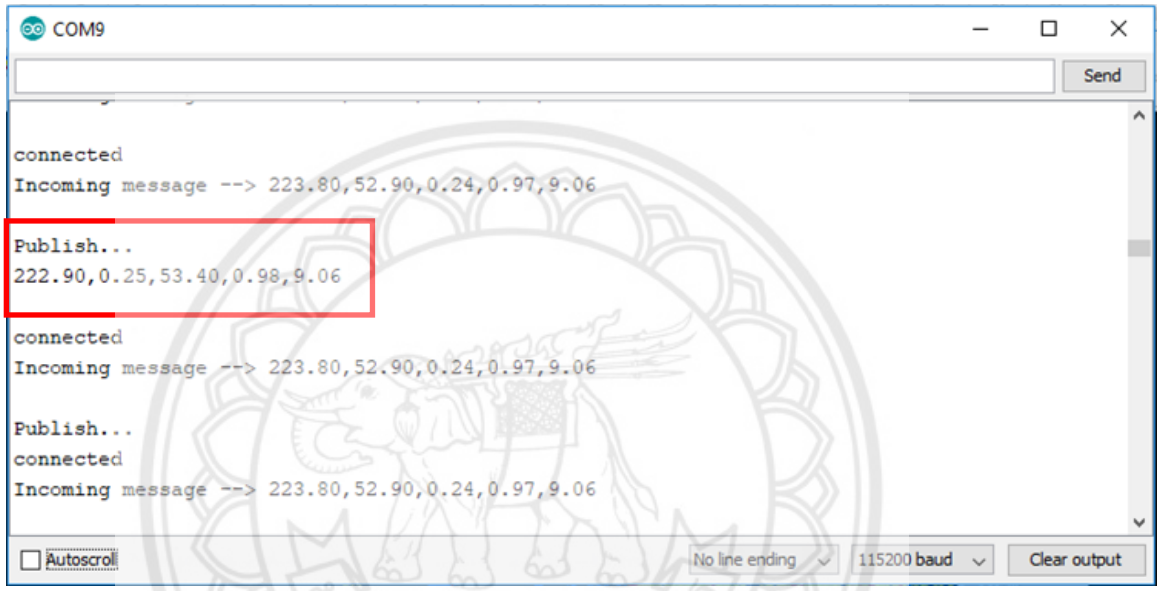
รูปที่ 4.1 หน้าจอของดิจิตอลมิเตอร์



รูปที่ 4.2 ค่าปริมาณทางไฟฟ้าที่แผงวงจรอาดูโน่ได้รับจากดิจิตอลมิเตอร์

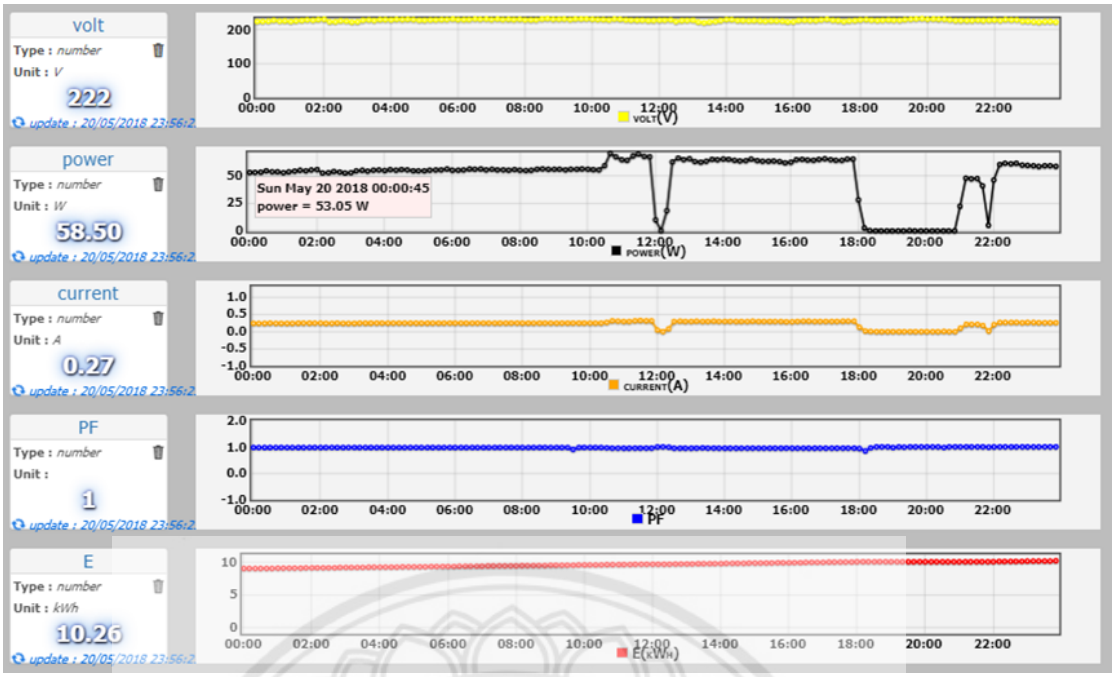
ผลการทดสอบ จากรูปที่ 4.1 พบว่า ค่าปริมาณทางไฟฟ้าฟ้าที่ส่งจากดิจิตอลมิเตอร์ให้กับแผงวงจรอาดูโน่ แสดงให้เห็นเพียงค่ากระแสไฟฟ้าและค่าประกอบกำลังไฟฟ้าซึ่งมีค่าตรงกับรูปที่ 4.2 ที่แสดงให้เห็นเพียง 2 ค่านั้นเนื่องจากค่าแรงดันไฟฟ้า, กำลังไฟฟ้า และค่าพลังงานไฟฟ้ามีค่าไม่คงที่เปลี่ยนแปลงตลอดเวลา

ผลการทดสอบ จากรูปที่ 4.2 พบว่า ค่าปริมาณทางไฟฟ้าฟ้าที่ส่งจากดิจิตอลมิเตอร์ให้กับแผงวงจรอาดูโน่ ทุก ๆ 15 วินาที และจัดเก็บไว้ในการ์ดความจำพร้อมทั้งส่งค่าปริมาณทางไฟฟ้าให้กับมอดูลสื่อสารไร้สาย

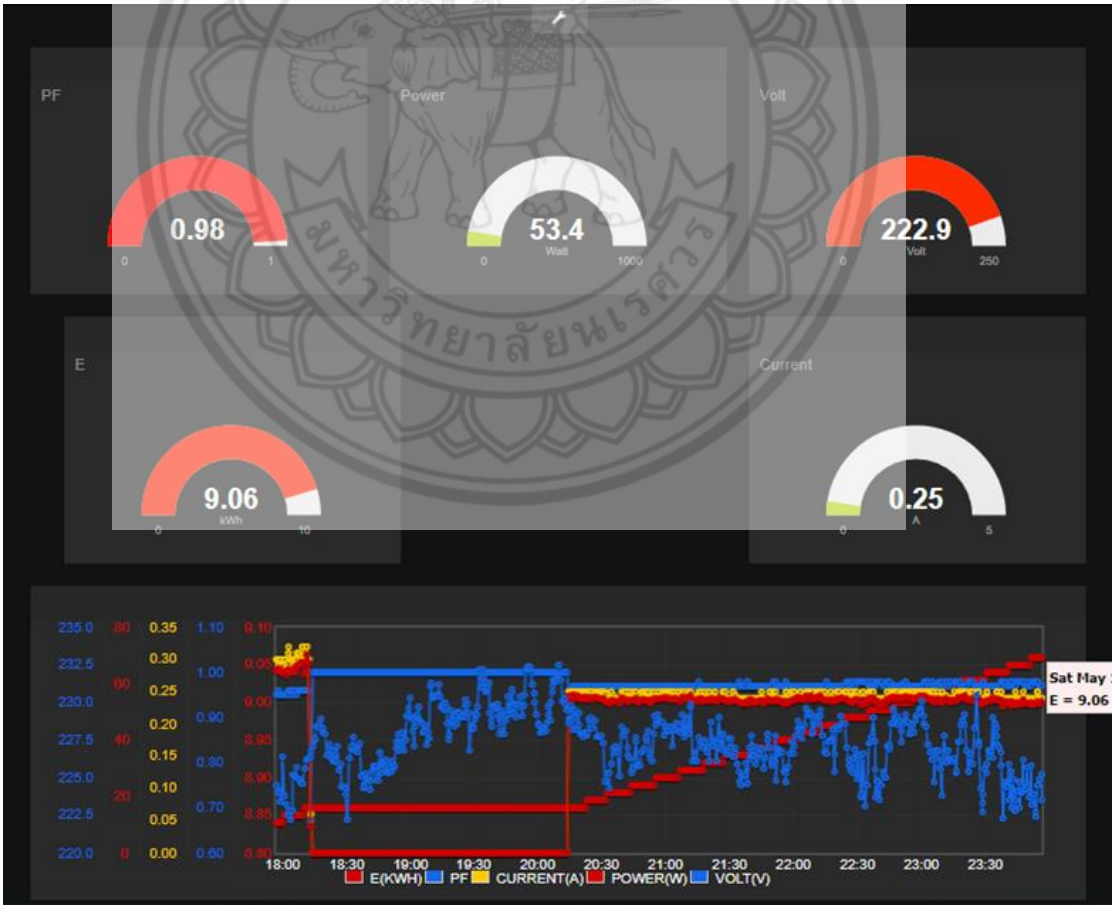


รูปที่ 4.3 ค่าปริมาณทางไฟฟ้าที่มอดูลสื่อสารไร้สายได้รับจากแผงวงจรอาดูโน่

ผลการทดสอบ จากรูปที่ 4.3 พบว่า ค่าปริมาณทางไฟฟ้าฟ้าที่ได้รับจากแผงวงจรอาดูโน่ จะถูกส่งให้กับ NETPIE ทุก ๆ 15 วินาที จะมีค่าตรงกับรูปที่ 4.2



รูปที่ 4.4 ผลการแสดงค่าปริมาณทางไฟฟ้าเป็นรูปกราฟเส้นของ NETPIE



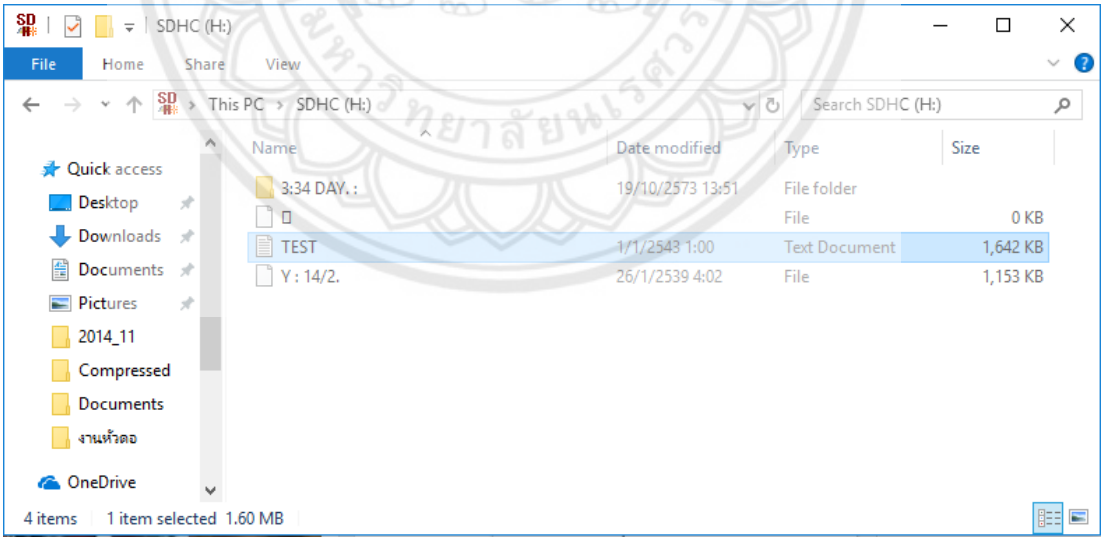
รูปที่ 4.5 ผลการแสดงค่าปริมาณทางไฟฟ้าเป็นกระดานข้อมูลของ NETPIE

ผลการทดสอบ การแสดงผลของ NETPIE จากรูปที่ 4.4 พบว่า ชุดการทดลองสามารถที่จะแสดงผลออกมาทาง NETPIE เป็นกราฟเส้นได้อย่างชัดเจน และมีค่าพารามิเตอร์ต่าง ๆ ถูกต้องตามหน้าจอของมิเตอร์ ชุดการทดลองสามารถวัดค่าปริมาณทางไฟฟ้า โดยที่จุดบนกราฟเส้นหมายความว่าจุดข้อมูลต่าง ๆ ที่เก็บมาในช่วงเวลาหน้าต่างละ 10 นาทีจะถูกนามาเฉลี่ยเป็นจุดเดียวไม่มีความคลาดเคลื่อนในการแสดงผล และ สามารถแสดงผลแบบติดตามได้ หากมีการเปลี่ยนแปลงของค่าทันทีทันใด

ผลการทดสอบ จากรูปที่ 4.5 พบว่า ชุดการทดลองสามารถที่จะแสดงผลออกมาทาง NETPIE เป็นหน้ากระดานข้อมูลแสดงเป็นตัวเลข และ กราฟรวมของค่าพารามิเตอร์ต่าง ๆ ซึ่ง หน้ากระดานข้อมูลแสดงเป็นตัวเลขนั้นจะการเปลี่ยนแปลงของค่าทันทีทันใดหากได้รับข้อมูลจากมอดูลสื่อสารไร้สาย และกระดานข้อมูลแสดงเป็นกราฟรวมจะมีค่าตรงกับรูปที่ 4.2 และ 4.3

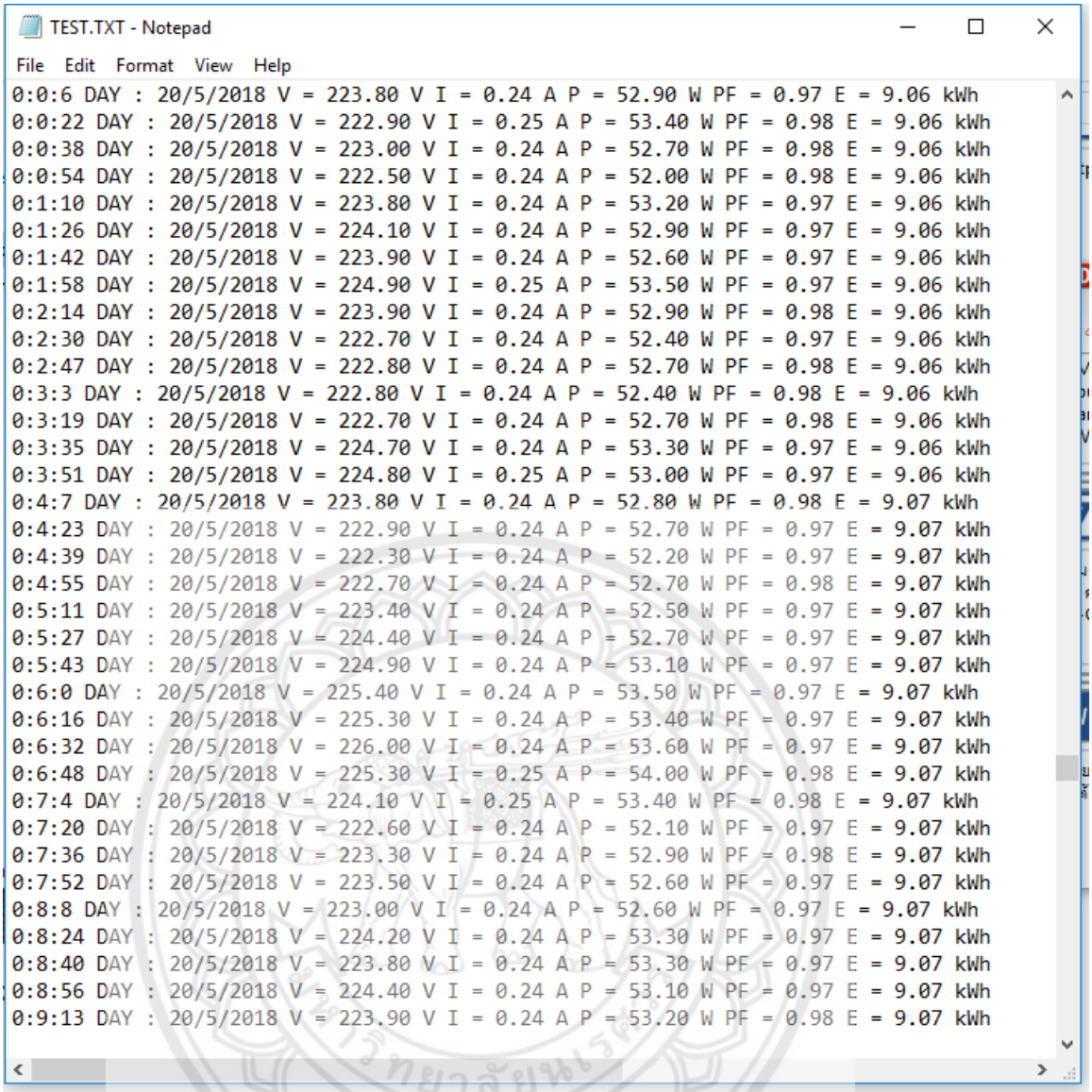
4.2 ผลการทดลองการเก็บข้อมูล

จากการทดลองวัดค่าปริมาณทางไฟฟ้าที่ทำการทดลองโดยใช้โหลดอุปกรณ์เครื่องใช้ไฟฟ้าภายในบ้าน ทดลองเปิดชุดทดสอบทิ้งไว้เป็นเวลา 7 วัน โปรแกรมที่เขียนจะสั่งให้บันทึกข้อมูลลงในหน่วยความจำทุก 15 วินาที โดยจะบันทึกเป็นค่าเวลา,แรงดัน, กระแส, กำลังไฟฟ้า, ค่าประกอบกำลัง และค่าพลังงานไฟฟ้า โปรแกรมจะสั่งให้บันทึกเป็น ปี เดือน และวันที่ ที่เวลาจริงตอนนั้น และบันทึกข้อมูลลงในไฟล์ แสดงดังรูปที่ 4.7

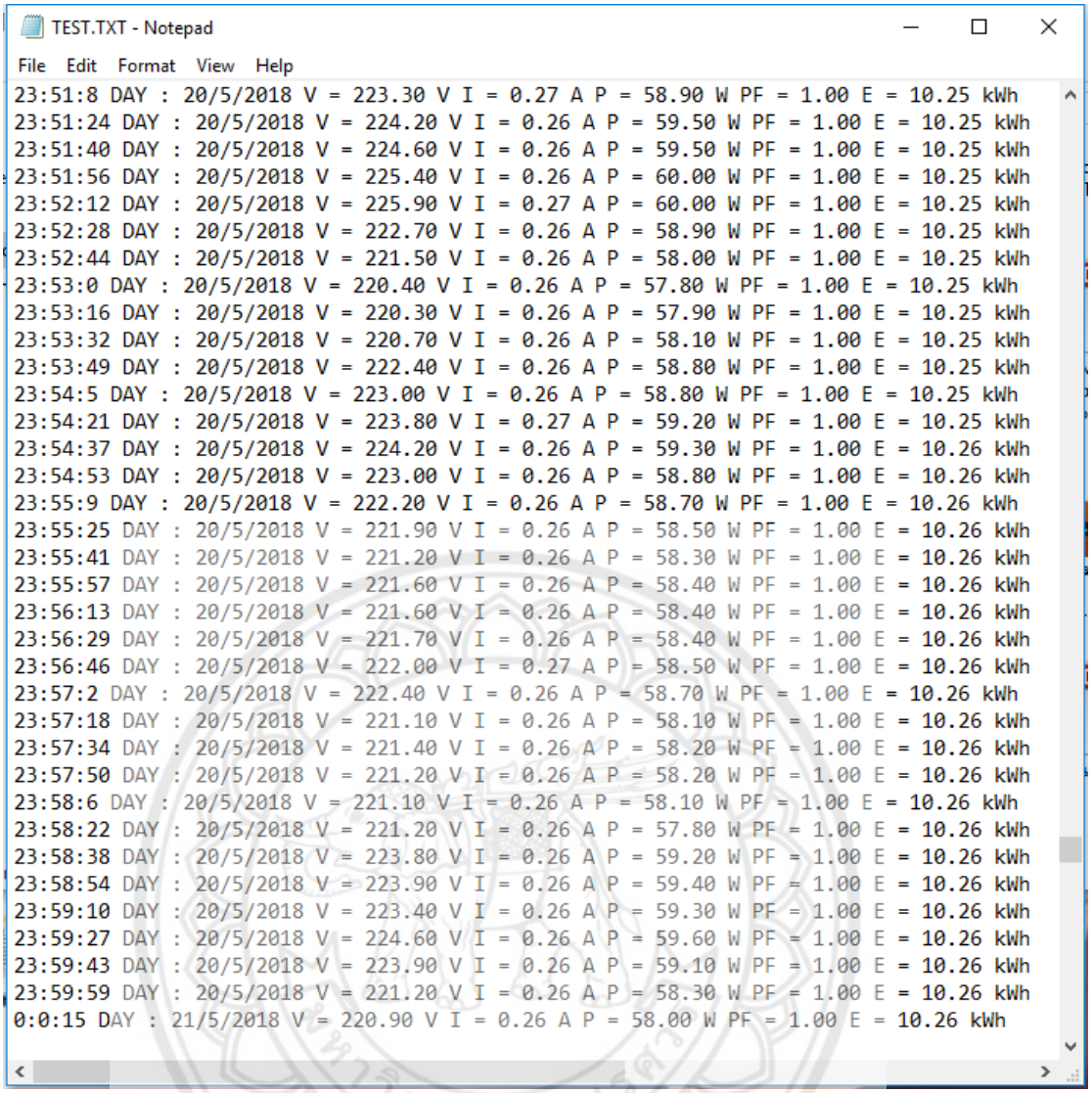


รูปที่ 4.6 แสดงไฟล์ในหน่วยความจำ

ผลการทดสอบ จากรูปที่ 4.6 เป็นการแสดงไฟล์ในหน่วยความจำ โดยข้อมูลนั้นจะถูกบันทึกไว้ในไฟล์ที่มีชื่อว่า TEST.TXT



รูปที่ 4.7 แสดงข้อมูลที่บันทึกในไฟล์วันที่ 20 พฤษภาคม 2561 (เริ่ม เวลา 0.00 น.)

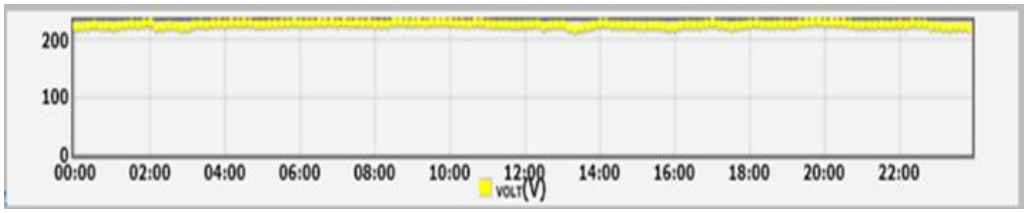


รูปที่ 4.7 (ต่อ) แสดงข้อมูลที่บันทึกในไฟล์วันที่ 20 พฤษภาคม 2561 (สิ้นสุด เวลา 23.59 น.)

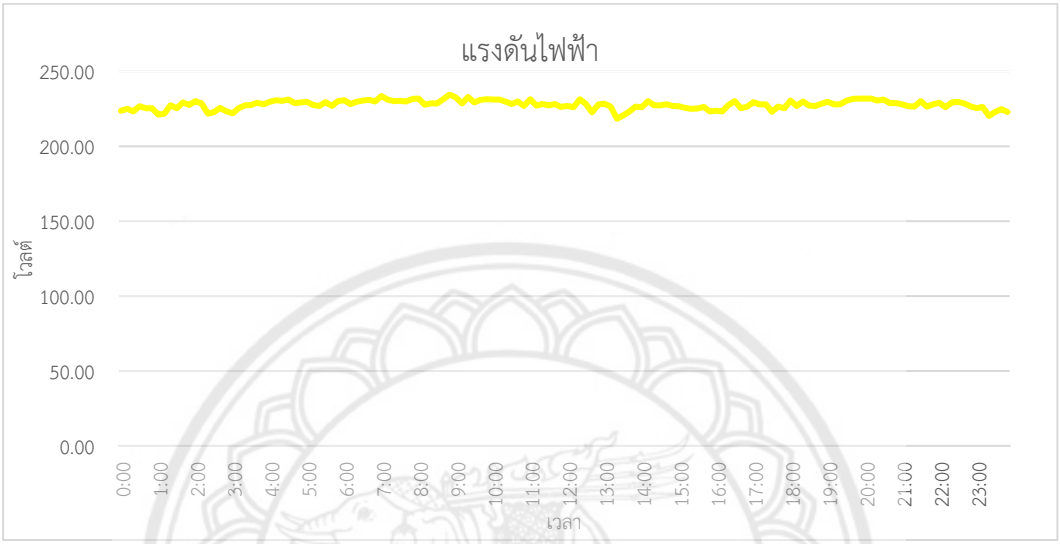
ผลการทดสอบ จากรูปที่ 4.7 เป็นการแสดงข้อมูลที่บันทึกในไฟล์วันที่ 20 พฤษภาคม 2561 โดยข้อมูลที่บันทึกไว้ในไฟล์จะถูกบันทึกทุก ๆ 15 วินาที

4.2.1 การเปรียบเทียบค่าปริมาณทางไฟฟ้าระหว่างข้อมูลกราฟที่หน้าเว็บ NETPIE กับข้อมูลกราฟกำลังไฟฟ้าที่บันทึกในไฟล์ 1 วัน

การเปรียบเทียบค่าปริมาณทางไฟฟ้าระหว่างข้อมูลกราฟที่หน้าเว็บ NETPIE กับ ข้อมูลกราฟกำลังไฟฟ้าที่บันทึกในไฟล์ 1 วัน คือของวันที่ 20 พฤษภาคม 2561 จะแสดงกราฟเปรียบเทียบค่าปริมาณไฟฟ้าที่เปลี่ยนแปลงไปใน 1 วัน ได้แก่ ค่าแรงดันไฟฟ้า, กระแสไฟฟ้า, กำลังไฟฟ้า, ประกอบกำลังไฟฟ้า และค่าพลังงานไฟฟ้า จะแสดงดังนี้

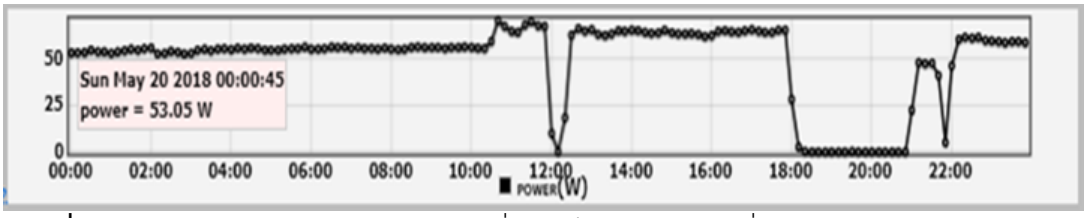


รูปที่ 4.8 แสดงข้อมูลกราฟค่าแรงดันไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561

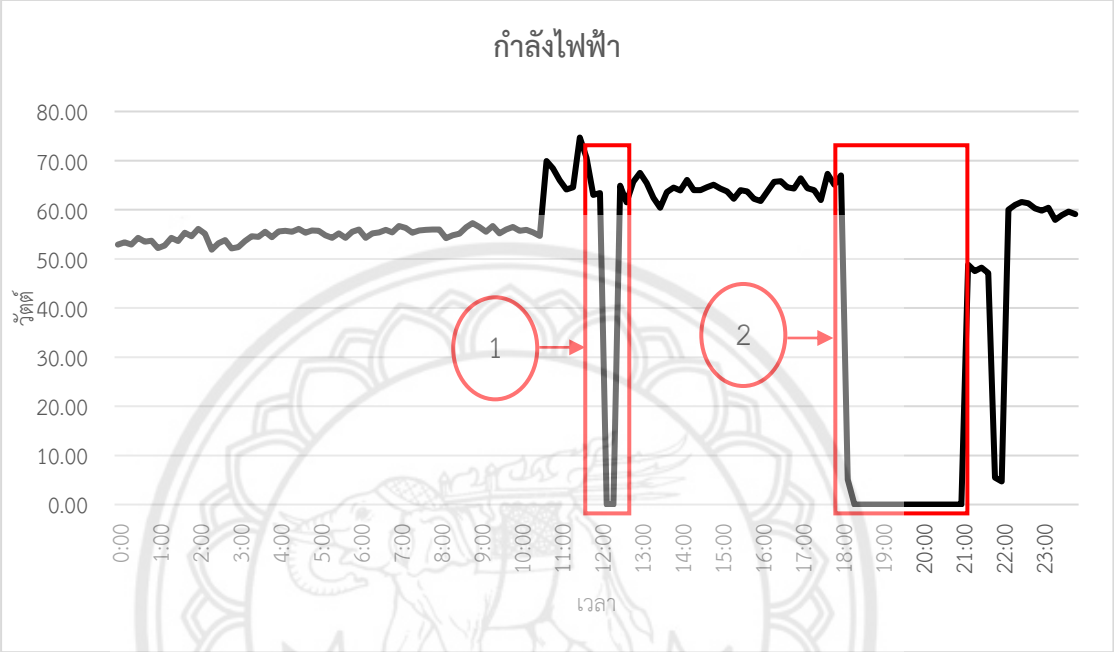


รูปที่ 4.9 แสดงข้อมูลกราฟค่าแรงดันไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561

จากรูปกราฟที่ 4.8 เปรียบเทียบกับรูปกราฟที่ 4.9 พบว่า มีลักษณะรูปกราฟค่าแรงดันไฟฟ้าใกล้เคียงกัน รูปกราฟค่าแรงดันไฟฟ้าที่หน้าเว็บ NETPIE เราไม่สามารถเลือกความละเอียดของช่วงกราฟได้เนื่องจากเว็บ NETPIE จะสร้างกราฟให้เอง ส่วนรูปกราฟค่าแรงดันไฟฟ้าที่บันทึกในไฟล์ จะมีความละเอียดมากกว่าเนื่องจากเรานำค่าแรงดันไฟฟ้าที่บันทึกในไฟล์มาสร้างกราฟเอง ค่าแรงดันไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าอยู่ระหว่าง 220-230 โวลต์

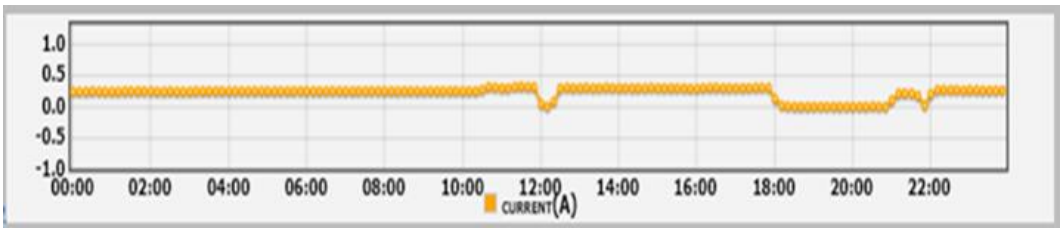


รูปที่ 4.10 แสดงข้อมูลกราฟค่ากำลังไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561

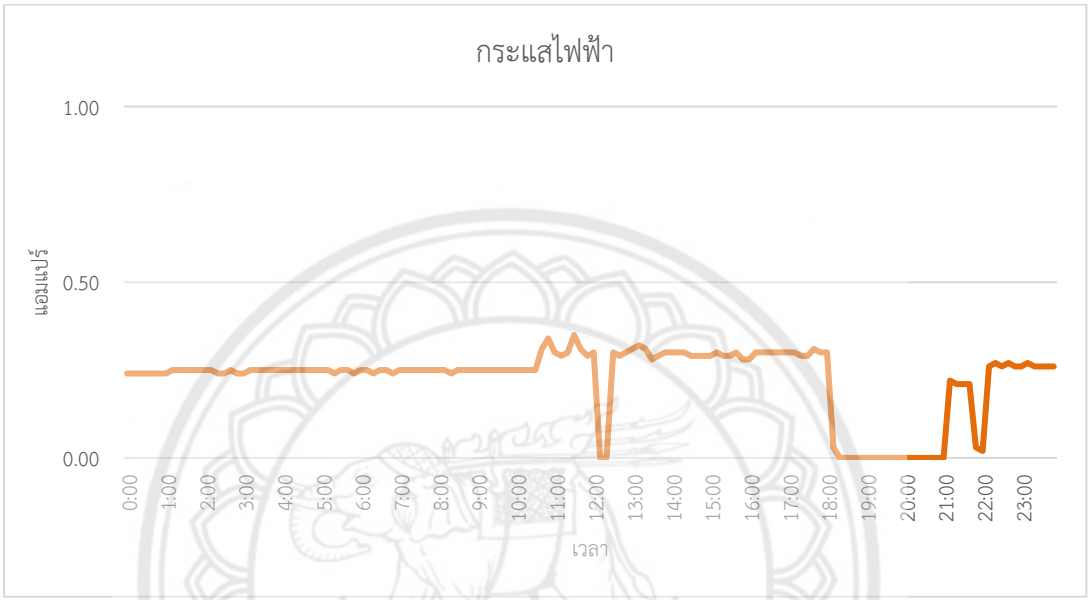


รูปที่ 4.11 แสดงข้อมูลกราฟค่ากำลังไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561

จากรูปกราฟที่ 4.10 เปรียบเทียบกับรูปกราฟที่ 4.11 พบว่า มีลักษณะรูปกราฟค่ากำลังไฟฟ้าใกล้เคียงกัน ค่ากำลังไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าเพิ่มขึ้นหรือลดลงนั้นจะขึ้นอยู่กับโหลดที่ผู้ใช้นำมาทดสอบว่าจะใช้กำลังไฟฟ้ามากหรือน้อยเพียงใด ส่วนหมายเลข 1 ค่ากำลังไฟฟ้ามีค่าลดลงเป็นศูนย์เนื่องจากผู้ทำการทดลองไม่มีการใช้งานโหลดใด ๆ เพราะออกไปรับประทานอาหารกลางวัน และ หมายเลข 2 ค่าลดลงเป็นศูนย์เนื่องจากผู้ทำการทดลองไม่มีการใช้งานโหลดใด ๆ เพราะออกไปออกกำลังกายและรับประทานอาหารเย็น

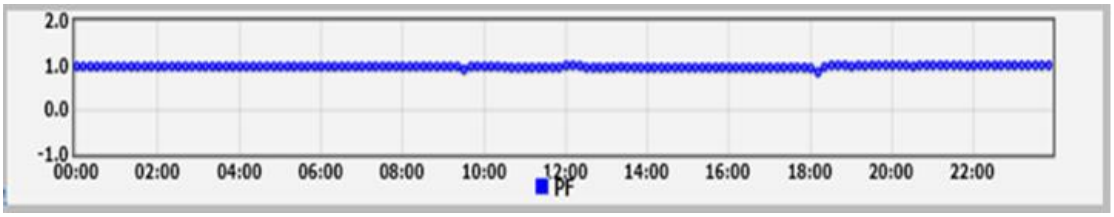


รูปที่ 4.12 แสดงข้อมูลกราฟค่ากระแสไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561

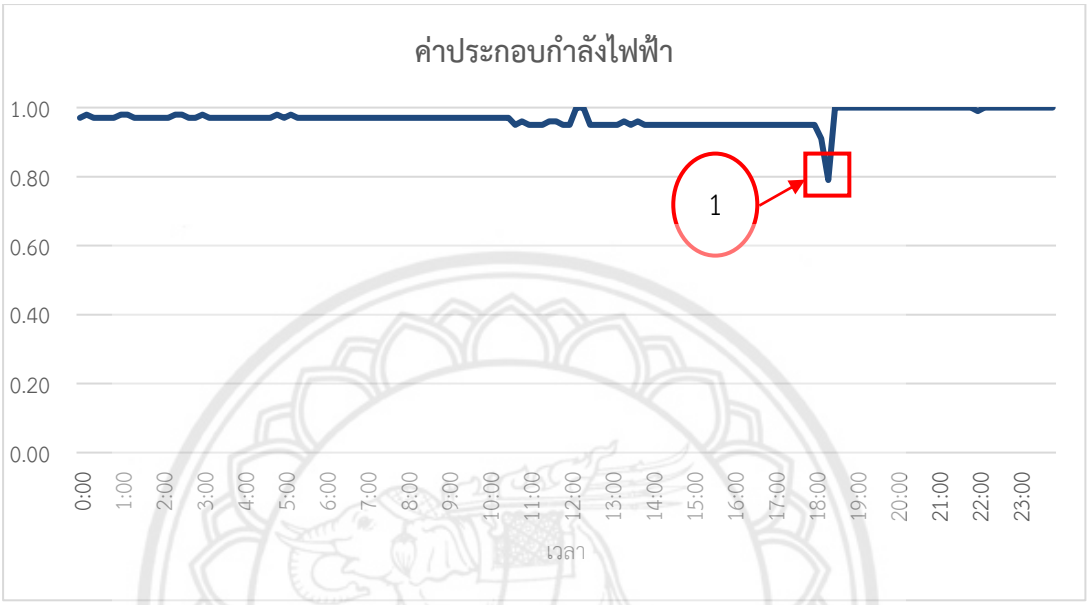


รูปที่ 4.13 แสดงข้อมูลกราฟค่ากระแสไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561

จากรูปกราฟที่ 4.12 เปรียบเทียบกับรูปกราฟที่ 4.13 พบว่า มีลักษณะรูปกราฟค่ากระแสไฟฟ้าใกล้เคียงกัน ค่ากระแสไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าเพิ่มขึ้นหรือลดลงนั้นจะขึ้นอยู่กับโหลดที่ผู้ใช้นำมาทดสอบว่าจะใช้กระแสไฟฟ้ามากหรือน้อยเพียงใด

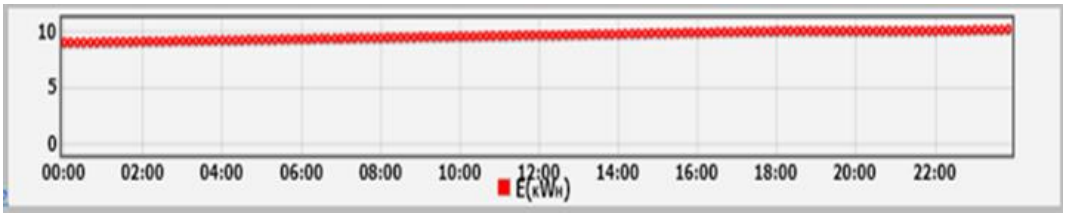


รูปที่ 4.14 แสดงข้อมูลกราฟค่าประกอบกำลังไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561

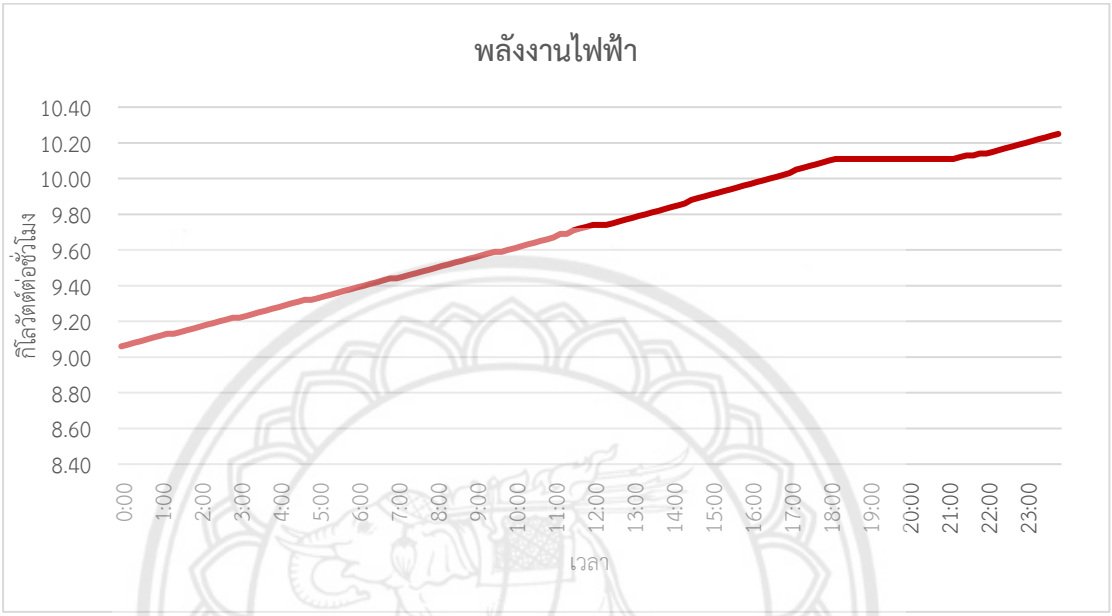


รูปที่ 4.15 แสดงข้อมูลกราฟค่าประกอบกำลังไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561

จากรูปกราฟที่ 4.14 เปรียบเทียบกับรูปกราฟที่ 4.15 พบว่า มีลักษณะรูปกราฟค่าประกอบกำลังไฟฟ้าใกล้เคียงกัน ค่าประกอบกำลังไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าเพิ่มขึ้นหรือลดลงนั้นจะขึ้นอยู่กับโหลดที่ผู้ใช้นำมาทดสอบว่า หากโหลดที่นำมาใช้เป็นโหลดชนิดความต้านทาน ค่าประกอบกำลังไฟฟ้า จะมีค่าเป็น 1 และ โหลดที่นำมาใช้เป็นโหลดชนิดตัวเหนี่ยวนำ หรือ ชนิดตัวเก็บประจุ ค่าประกอบกำลังไฟฟ้าจะมีค่าน้อยกว่า 1 ส่วนหมายเลข 1 มีค่าลดลงนั้นผู้ทำการทดลองได้ใช้โหลดชนิดไม่เป็นเชิงเส้น (non-liner load) เพียงอย่างเดียวคือชาร์จแบตเตอรี่โทรศัพท์มือถือ



รูปที่ 4.16 แสดงข้อมูลกราฟค่าพลังงานไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 20 พฤษภาคม 2561

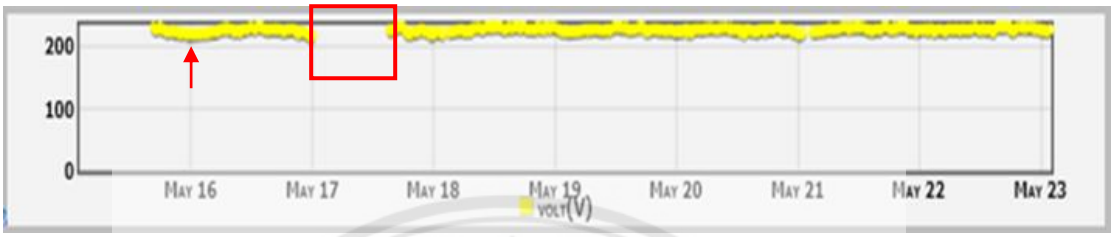


รูปที่ 4.17 แสดงข้อมูลกราฟค่าพลังงานไฟฟ้าที่บันทึกในไฟล์ วันที่ 20 พฤษภาคม 2561

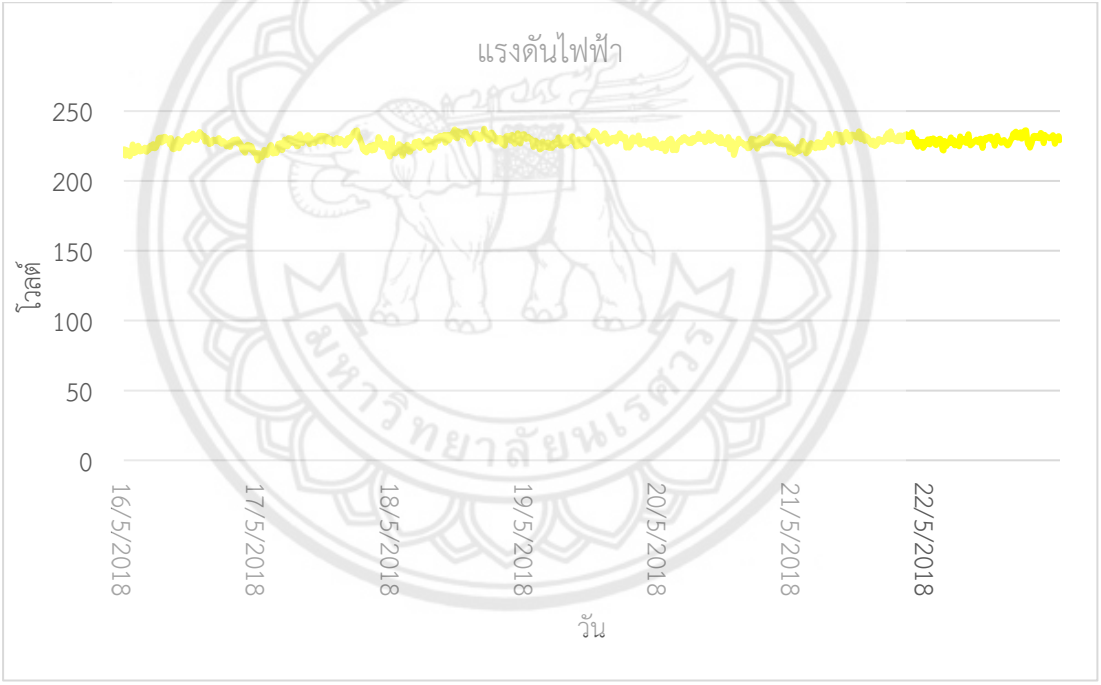
จากรูปกราฟที่ 4.16 เปรียบเทียบกับรูปกราฟที่ 4.17 พบว่า มีลักษณะรูปกราฟค่าพลังงานไฟฟ้าใกล้เคียงกัน ค่าพลังงานไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าที่เพิ่มขึ้นนั้นเนื่องจากเป็นการสะสมค่าพลังงานของโหลดที่ผู้ใช้นำมาทดลองซึ่งจะมีค่าเพิ่มขึ้นเรื่อย ๆ

4.2.2 การเปรียบเทียบค่าปริมาณทางไฟฟ้าระหว่างข้อมูลกราฟที่หน้าเว็บ NETPIE กับข้อมูลกราฟกำลังไฟฟ้าที่บันทึกในไฟล์ 7 วัน

การเปรียบเทียบค่าปริมาณทางไฟฟ้าระหว่างข้อมูลกราฟที่หน้าเว็บ NETPIE กับ ข้อมูลกราฟกำลังไฟฟ้าที่บันทึกในไฟล์ 7 วัน คือรหว่างวันที่ 16-22 พฤษภาคม 2561 จะแสดงกราฟเปรียบเทียบค่าปริมาณไฟฟ้าที่เปลี่ยนแปลงไปใน 7 วัน ได้แก่ ค่าแรงดันไฟฟ้า, กระแสไฟฟ้า, กำลังไฟฟ้า, ประกอบกำลังไฟฟ้า และค่าพลังงานไฟฟ้า จะแสดงดังนี้



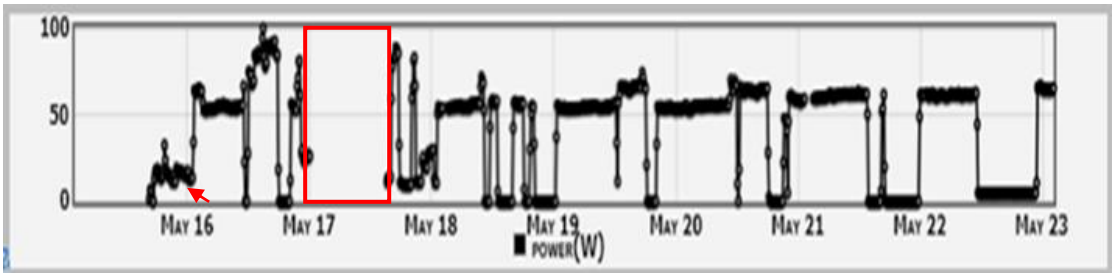
รูปที่ 4.18 แสดงข้อมูลกราฟค่าแรงดันไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561



รูปที่ 4.19 แสดงข้อมูลกราฟค่าแรงดันไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561

จากรูปกราฟที่ 4.18 เปรียบเทียบกับรูปกราฟที่ 4.19 พบว่า มีลักษณะรูปกราฟค่าแรงดันไฟฟ้าใกล้เคียงกัน รูปกราฟค่าแรงดันไฟฟ้าที่หน้าเว็บ NETPIE เราไม่สามารถเลือกความละเอียดของช่วงกราฟได้เนื่องจากเว็บ NETPIE จะสร้างกราฟให้เองโดยในรูปกราฟที่ 4.18 นั้นจะเริ่มเปรียบเทียบตั้งแต่จุดที่มีลูกศร และมีบางส่วนของค่าแรงดันไฟฟ้าที่หายไปในกรอบสี่เหลี่ยมของวันที่ 17 พฤษภาคม 2561 เนื่องจากสัญญาณอินเตอร์เน็ตเกิดการขัดข้องทำให้มอดูลสื่อสารไร้สายไม่สามารถส่งค่าแรงดันไฟฟ้าให้กับเว็บ NETPIE ได้ ส่วนรูปกราฟค่าแรงดันไฟฟ้าที่บันทึกในไฟล์

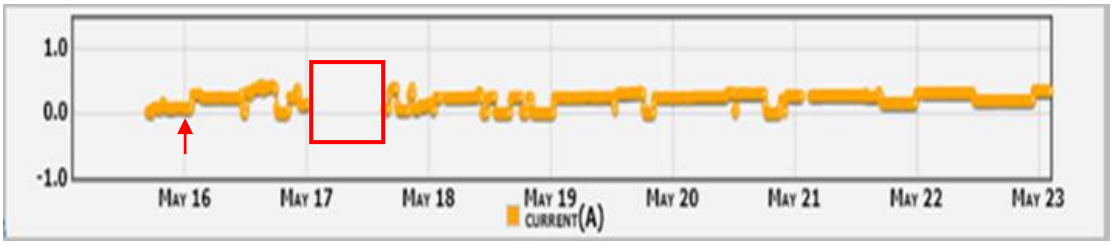
ค่าแรงดันไฟฟ้าจะมีข้อมูลที่ขาดหายไป และมีความละเอียดมากกว่าเนื่องจากเรานำค่าแรงดันไฟฟ้าที่บันทึกในไฟล์มาสร้างกราฟเอง ค่าแรงดันไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าอยู่ระหว่าง 220-230 โวลต์



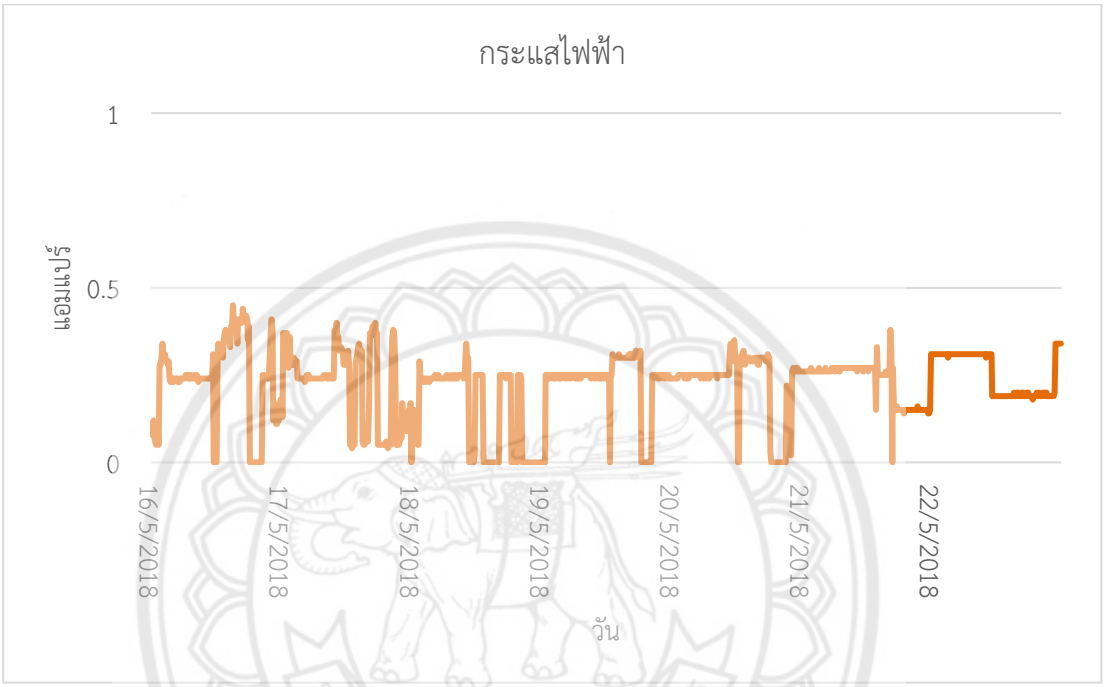
รูปที่ 4.20 แสดงข้อมูลกราฟค่ากำลังไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561

รูปที่ 4.21 แสดงข้อมูลกราฟค่ากำลังไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561

จากรูปกราฟที่ 4.20 เปรียบเทียบกับรูปกราฟที่ 4.21 พบว่า มีลักษณะรูปกราฟค่ากำลังไฟฟ้าใกล้เคียงกัน โดยในรูปกราฟที่ 4.20 นั้นจะเริ่มเปรียบเทียบตั้งแต่จุดที่มีลูกศร มีบางส่วนของค่ากำลังไฟฟ้าที่หายไปในกรอบสี่เหลี่ยมของวันที่ 17 พฤษภาคม 2561 เนื่องจากสัญญาณอินเตอร์เน็ตเกิดการขัดข้องทำให้มอดูลสื่อสารไร้สายไม่สามารถส่งค่ากำลังไฟฟ้าให้กับเว็บ NETPIE ได้ แต่ค่ากำลังไฟฟ้าที่หายไปนั้นยังบันทึกไว้ในไฟล์ ค่ากำลังไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าเพิ่มขึ้นหรือลดลงนั้นจะขึ้นอยู่กับโหลดที่ผู้ใช้นำมาทดสอบว่าจะใช้กำลังไฟฟ้ามากหรือน้อยเพียงใด

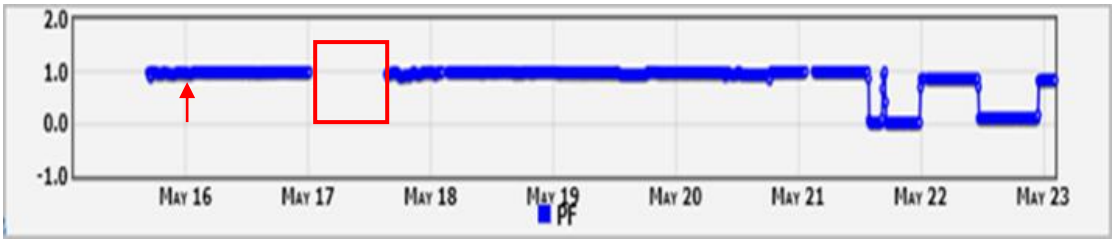


รูปที่ 4.22 แสดงข้อมูลกราฟค่ากระแสไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561

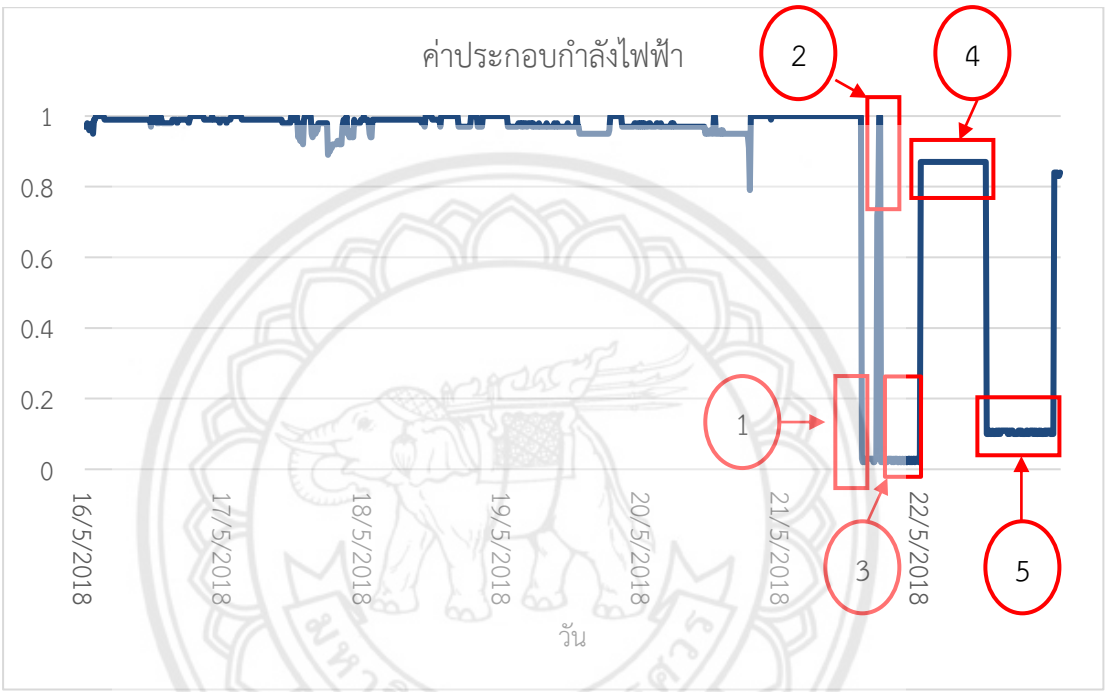


รูปที่ 4.23 แสดงข้อมูลกราฟค่ากระแสไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561

จากรูปกราฟที่ 4.22 เปรียบเทียบกับรูปกราฟที่ 4.23 พบว่า มีลักษณะรูปกราฟค่ากระแสไฟฟ้าใกล้เคียงกัน โดยในรูปกราฟที่ 4.22 นั้นจะเริ่มเปรียบเทียบตั้งแต่จุดที่มีลูกศร มีบางส่วนของค่ากระแสไฟฟ้าที่หายไปในกรอบสี่เหลี่ยมของวันที่ 17 พฤษภาคม 2561 เนื่องจากสัญญาณอินเตอร์เน็ตเกิดการขัดข้องทำให้มอดูลสื่อสารไร้สายไม่สามารถส่งค่ากระแสไฟฟ้าให้กับเว็บ NETPIE ได้ แต่ค่ากระแสไฟฟ้าที่หายไปนั้นยังบันทึกไว้ในไฟล์ ค่ากระแสไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าเพิ่มขึ้นหรือลดลงนั้นจะขึ้นอยู่กับโหลดที่ผู้ใช้นำมาทดสอบว่าจะใช้กระแสไฟฟ้ามากหรือน้อยเพียงใด



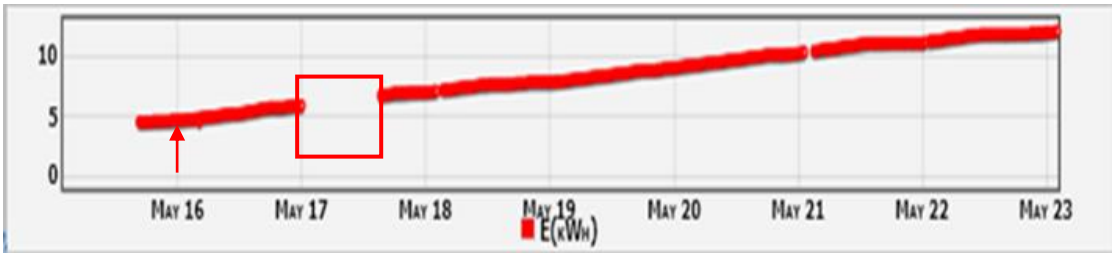
รูปที่ 4.24 แสดงข้อมูลกราฟค่าประกอบกำลังไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561



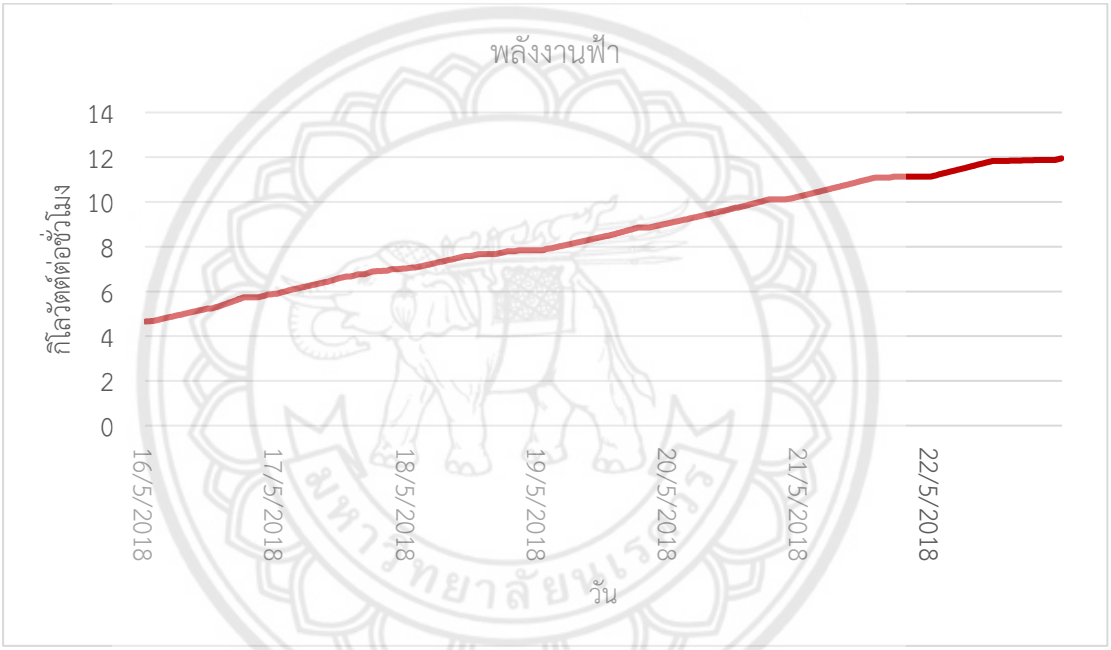
รูปที่ 4.25 แสดงข้อมูลกราฟค่าประกอบกำลังไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561

จากรูปกราฟที่ 4.24 เปรียบเทียบกับรูปกราฟที่ 4.25 พบว่า มีลักษณะรูปกราฟค่าประกอบกำลังไฟฟ้าใกล้เคียงกัน โดยในรูปกราฟที่ 4.24 นั้นจะเริ่มเปรียบเทียบตั้งแต่จุดที่มีลูกศรมีบางส่วนของค่าประกอบกำลังไฟฟ้าที่หายไปในกรอบสี่เหลี่ยมของวันที่ 17 พฤษภาคม 2561 เนื่องจากสัญญาณอินเตอร์เน็ตเกิดการขัดข้องทำให้มอดูลสื่อสารไร้สายไม่สามารถส่งค่าประกอบกำลังไฟฟ้าให้กับเว็บ NETPIE ได้ แต่ค่าประกอบกำลังไฟฟ้าที่หายไปนั้นยังบันทึกไว้ในไฟล์ ค่าประกอบกำลังไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าเพิ่มขึ้นหรือลดลงนั้นจะขึ้นอยู่กับโหลดที่ผู้ใช้นำมาทดสอบว่า หากโหลดที่นำมาใช้เป็นโหลดชนิดความต้านทานค่าประกอบกำลังไฟฟ้า จะมีค่าเป็น 1 และ โหลดที่นำมาใช้เป็นโหลดชนิดตัวเหนี่ยวนำ หรือ ชนิดตัวเก็บประจุค่าประกอบกำลังไฟฟ้าจะมีค่าน้อยกว่า 1 โดยหมายเลข 1 และ 3 เป็นการต่อโหลดชุดทดลองตัวเก็บประจุเพียงอย่างเดียวทำให้ค่าประกอบกำลังไฟฟ้าจะมีค่าน้อยมาก หมายเลข 2 เป็นการถอดชุดทดลองออกทำให้ค่าประกอบกำลังไฟฟ้ามีค่าเท่ากับ 1 หมายเลข 4 เป็นการต่อโหลดหลอดไฟและ

โหลดตัวเก็บประจุทำให้ค่าประกอบกำลังไฟฟ้ามีค่าประมาณ 0.85 และหมายเลข 5 เป็นการต่อโหลดชุดทดลองเหนี่ยวนำเพียงอย่างเดียวทำให้ค่าประกอบกำลังไฟฟ้าจะมีค่าน้อยมาก



รูปที่ 4.26 แสดงข้อมูลกราฟค่าพลังงานไฟฟ้าที่หน้าเว็บ NETPIE วันที่ 16-22 พฤษภาคม 2561



รูปที่ 4.27 แสดงข้อมูลกราฟค่าพลังงานไฟฟ้าที่บันทึกในไฟล์ วันที่ 16-22 พฤษภาคม 2561

จากรูปกราฟที่ 4.26 เปรียบเทียบกับรูปกราฟที่ 4.27 พบว่า มีลักษณะรูปกราฟค่าพลังงานไฟฟ้าใกล้เคียงกัน โดยในรูปกราฟที่ 4.26 นั้นจะเริ่มเปรียบเทียบตั้งแต่จุดที่มีลูกศร มีบางส่วนของค่าพลังงานไฟฟ้าที่หายไปในกรอบสี่เหลี่ยมของวันที่ 17 พฤษภาคม 2561 เนื่องจากสัญญาณอินเตอร์เน็ตเกิดการขัดข้องทำให้มอดูลสื่อสารไร้สายไม่สามารถส่งค่าพลังงานไฟฟ้าให้กับเว็บ NETPIE ได้ แต่ค่าค่าพลังงานไฟฟ้าที่หายไปนั้นยังบันทึกไว้ในไฟล์ ค่าพลังงานไฟฟ้าที่สังเกตได้จากกราฟทั้งสองจะมีการเปลี่ยนแปลงค่าที่เพิ่มขึ้นนั้นเนื่องจากเป็นการสะสมค่าพลังงานของโหลดที่ผู้ใช้นำมาทดลองซึ่งจะมีค่าเพิ่มขึ้นเรื่อย ๆ

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 สรุปผลการทดลอง

จากการศึกษาชุดทดลองระบบตรวจวัดการใช้พลังงานไฟฟ้าด้วยดิจิทัลมิเตอร์ผ่าน NETPIE คือการทดลองวัดค่าปริมาณทางไฟฟ้า ได้แก่ แรงดัน, กระแส, กำลังไฟฟ้า, ค่าประกอบกำลัง, และค่าพลังงานไฟฟ้า ของภาระโหลดต่าง ๆ ที่นำมาทดลอง โดยต่อผ่านมิเตอร์วัดพลังแบบดิจิทัล แล้วทำการบันทึกค่าและแสดงผลผ่านทาง NETPIE ในรูปแบบกราฟและกระดานข้อมูล เมื่อพิจารณาการทดลองพบว่า

ค่าปริมาณทางไฟฟ้าจากผลการทดลอง โดยขึ้นอยู่กับภาระโหลดต่าง ๆ ที่นำมาเชื่อมต่อเข้ากับระบบ ซึ่งค่าปริมาณทางไฟฟ้าที่แสดงในรูปกราฟและกระดานข้อมูลของ NETPIE มีค่าตรงกับที่เก็บข้อมูลได้จากมิเตอร์ แม้มีการเปลี่ยนแปลงภาระโหลดแบบกะทันหัน ค่าปริมาณทางไฟฟ้าจะเปลี่ยนแปลงเร็วช้าขึ้นกับจำนวนภาระโหลดและการตั้งค่าเวลาให้ค่าปริมาณทางไฟฟ้าแสดงผล การแสดงผลของNETPIE ยังสามารถแสดงผลแบบติดตามผลได้อย่างต่อเนื่อง และสามารถแสดงผลย้อนหลังได้

ค่าแรงดันไฟฟ้า ที่วัดได้จากการทดลอง จะมีปริมาณไม่เท่ากันในทุกการทดลอง เนื่องจากมีการเปลี่ยนค่าของภาระที่นำมาต่อให้มากขึ้นหรือลดลง และแรงดันไฟฟ้าของระบบไฟฟ้า มีการเปลี่ยนแปลงอยู่ตลอดเวลา

ค่ากระแสไฟฟ้า จากผลการทดลองจะขึ้นอยู่กับภาระโหลดถ้าภาระโหลดกินกระแสมาก ๆ ค่ากระแสไฟฟ้าจะแสดงบนหน้าจออย่างเห็นได้ชัด ส่วนใหญ่ค่าที่แสดงจะเป็นค่าเดียวกันเพราะว่าเมื่อต่อภาระโหลดหนึ่งครั้งโปรแกรมกว่าจะบันทึกค่าอัตโนมัติไปเรื่อย ๆ จนกว่ามีการเพิ่มภาระโหลดหรือลดภาระโหลด

ค่ากำลังไฟฟ้า จากผลการทดลองจะมีปริมาณไม่เท่ากัน จะขึ้นอยู่กับภาระโหลด ถ้าภาระโหลดที่ใช้มีค่ากระแสไฟฟ้ามากก็จะให้ค่ากำลังไฟฟ้าเพิ่มมากขึ้นตามไปด้วย

ค่าประกอบกำลัง จากการทดลอง ค่าประกอบกำลังเพิ่มขึ้นหรือลดลงนั้นจะขึ้นอยู่กับโหลดที่นำมาทดสอบว่า หากโหลดที่นำมาใช้เป็นโหลดชนิดความต้านทาน ค่าประกอบกำลังไฟฟ้า จะมีค่าเป็น 1 และ โหลดที่นำมาใช้เป็นโหลดชนิดตัวเหนี่ยวนำ หรือ ชนิดตัวเก็บประจุ ค่าประกอบกำลังไฟฟ้าจะมีค่าน้อยกว่า 1 ซึ่งแสดงค่าเป็นค่าบวกเพียงอย่างเดียว ไม่ระบุว่าระบบไฟฟ้าว่านำหน้า (Leading) หรือล้าหลัง (Lagging)

ค่าพลังงานไฟฟ้า จากการทดลอง เป็นการสะสมค่าพลังงานของโหลดที่นำมาทดลองซึ่งจะมีค่าเพิ่มขึ้นเรื่อย ๆ

5.2 ประเมินผล

จากการดำเนินงานเทียบกับวัตถุประสงค์ได้ผลดังนี้

1. สามารถนำระบบตรวจวัดการใช้พลังงานไฟฟ้ามาใช้ร่วมกับร่วมมิเตอร์ไฟฟ้า 1 เฟสได้ โดยมีการแสดงค่า แรงดัน, กระแส, กำลังไฟฟ้า, ค่าประกอบกำลัง, และค่าพลังงานไฟฟ้า ผ่าน NETPIE และจัดเก็บลงในการ์ดความจำได้

2. มีความรู้ความเข้าใจเกี่ยวกับระบบสื่อสาร เพื่อทำการติดต่อสื่อสาร ระบบตรวจวัดการใช้พลังงานไฟฟ้า กับติมิเตอร์วัดพลังงานไฟฟ้าแบบดิจิทัลได้

5.3 ปัญหา ข้อเสนอแนะ และแนวทางแก้ไข

1. ปัญหาจากการเชื่อมต่อระบบสื่อสารที่ผิดพลาด ทำให้อุปกรณ์บางส่วนได้รับความเสียหาย จึงต้องทำการซ่อมแซม ทำให้ชุดทดสอบที่ออกมาไม่เรียบร้อยเท่าที่ควรจะเป็น และมีอันตรายจากกระแสไฟฟ้าอยู่มาก โดยเฉพาะจุดเชื่อมต่อ จึงต้องทำการซื้ออุปกรณ์ฉนวนมาใส่ชุดเชื่อมต่อเพื่อความปลอดภัย

2. การติดต่อสื่อสารระหว่างอุปกรณ์สื่อสารนั้นมีจังหวะการรับส่งสัญญาณไม่ตรงกัน ทำให้อุปกรณ์สื่อสารไม่สามารถรับ-ส่งข้อมูลระหว่างกันได้ แก้ไขโดยใช้อุปกรณ์แปลงสัญญาณเพื่อให้อุปกรณ์สื่อสารสามารถรับ-ส่งข้อมูลกันได้

3. อุปกรณ์มีความเปราะบางมาก ในการเชื่อมต่อหรือเคลื่อนย้ายอุปกรณ์ทุกครั้ง ต้องระมัดระวังเป็นพิเศษ

4. ไม่สามารถปรับเปลี่ยนสเกลและสีของกราฟที่ NETPIE ได้ เนื่องผู้ให้บริการ Cloud platorm จะจัดเตรียมสิ่งที่จำเป็นต้องใช้ในการพัฒนาซอฟต์แวร์และแอพพลิเคชันเอาไว้ให้

5.4 แนวทางในการพัฒนาต่อไป

จากชุดทดลองระบบตรวจวัดการใช้พลังงานไฟฟ้าด้วยดิจิทัลมิเตอร์ผ่าน NETPIE สามารถนำไปต่อยอดพัฒนาต่อได้ โดยอาจจะออกแบบโปรแกรมให้โมดูลสื่อสารไร้สายสามารถค้นหาสัญญาณ WI-FI ได้โดยอัตโนมัติ เพื่อให้ผู้ใช้งานเพียงแค่ใส่รหัสผ่าน การจัดเก็บข้อมูลให้อยู่ในไฟล์รูปแบบอื่นที่ทำให้สามารถนำข้อมูลออกไปใช้ประโยชน์ได้ง่าย หรืออาจพัฒนาทำแอปพลิเคชัน นำไปใช้ในสมาร์ทโฟน เพื่อที่จะเฝ้าดูการเปลี่ยนแปลง หรือทำการแจ้งเตือนหากเกิดปัญหาจากค่าพลังงาน และออกแบบโปรแกรมโดยการนำค่าพลังงานไฟฟ้าที่ใช้ไปคำนวณหาค่าไฟฟ้าแล้วแสดงค่าที่เว็บ NETPIE ตามค่าที่ใช้งานในปัจจุบันขณะนั้น

เอกสารอ้างอิง

บริษัท ศิลาไมโคร จำกัด, (ตุลาคม 2556), **มิเตอร์ไฟฟ้า,** สืบค้นเมื่อวันที่ 20 พฤศจิกายน 2560, จาก http://www.smicrothai.com

เอกสารประกอบการสอนวิชาไมโครคอนโทรลเลอร์เบื้องต้น, (พฤษภาคม 2556), **ไมโครคอนโทรลเลอร์,** สืบค้นเมื่อวันที่ 20 พฤศจิกายน 2560, จาก http://www.sbt.ac.th/new/sites/default/files/TNP_Unit_1.pdf

บริษัท โอเมก้า เมชเซอริ่ง อินสทรูเม้นท์ จำกัด, (พฤศจิกายน 2557), **มาตรฐาน RS485**, สืบค้นเมื่อวันที่ 21 พฤศจิกายน 2560, จาก https://www.omi.co.th

Arduitronics. (พฤษภาคม 2560), **มอดูลเก็บบันทึกข้อมูล**, สืบค้นเมื่อวันที่ 22 พฤศจิกายน 2560, จาก https://www.arduitronics.com/product/179/data-logger-shield

Ayarafun, (สิงหาคม 2558), **มอดูลสื่อสารไร้สาย,** สืบค้นเมื่อวันที่ 23 พฤศจิกายน 2560, จาก http://www.ayarafun.com/2015/08/introduction-arduino-esp8266-nodemcu/

NECTEC, (กันยายน 2559), **NETPIE,** สืบค้นเมื่อวันที่ 24 พฤศจิกายน 2560, จาก https://www.nectec.or.th/innovation/innovation-software/netpie.html

NECTEC, (มีนาคม 2560), **NETPIE,** สืบค้นเมื่อวันที่ 8 มิถุนายน 2560, จาก https://www.nectec.or.th/news/news-pr-news/netpie-nidec.html

ภาคผนวก ก
รหัสของโปรแกรมในการแสดงผลค่าปริมาณทางไฟฟ้า

โปรแกรมหลักที่ใช้กับแผงวงจรอาดูโน่

```
#include <DS1307RTC.h>      //ไลบรารีที่ต้องติดตั้งเพิ่มแสดงรายระเอียดท้ายโปรแกรม
#include <TimeLib.h>        //ไลบรารีที่ต้องติดตั้งเพิ่มแสดงรายระเอียดท้ายโปรแกรม
#include <Wire.h>           //ไลบรารีการติดต่อสื่อสารกับอุปกรณ์I2C/TWI
#include <SPI.h>            //ไลบรารีการติดต่อสื่อสารกับอุปกรณ์SPI
#include <SD.h>             //ไลบรารีการ์ดหน่วยความจำ
#include "REG_SDM120.h"     //ไลบรารีเรียกใช้รีจิสเตอร์จากมิตอร์
#include <ModbusMaster.h>  //ไลบรารีโปรโตคอลสื่อสาร
ModbusMaster node;  //การกำหนดตัวแปรต่างๆ
File myFile;
const int chipSelect = 10;
String time ;
String Day = "";
tmElements_t tm;
//01 04 00 00 01 56 71 3F // Test 30001
//--------------------------------------------
// Convent 32bit to float
//--------------------------------------------
// ฟังก์ชันแปลงเลขฐานสิบหกเป็นฐานสิบ
float HexTofloat(uint32_t x) {
                                    return (*(float*)&x);
                                }
uint32_t FloatTohex(float x) {
```

```
                return (*(uint32_t*)&x);
            }
//---------------------------------------------
float Read_Meter_float(char addr , uint16_t  REG) {
        float i = 0;
        uint8_t j, result;
        uint16_t data[2];
       uint32_t value = 0;
        node.begin(addr, Serial2);
  result = node.readInputRegisters (REG, 2); ///< Modbus function 0x04 Read Input
Registers
  delay(500);
  if (result == node.ku8MBSuccess) {
    for (j = 0; j < 2; j++)
    {
      data[j] = node.getResponseBuffer(j);
    }
    value = data[0];
    value = value << 16;
    value = value + data[1];
    i = HexTofloat(value);
    //Serial.println("Connec modbus Ok.");
    return i;
```

```
  } else {
    Serial.print("Connec modbus fail. REG >>> "); Serial.println(REG, HEX); // Debug
    delay(1000);
    return 0;
  }
}
void GET_METER() {     // Update read all data
  delay(1000);                              //
    for (char i = 0; i < Total_of_Reg ; i++){
      DATA_METER [i] = Read_Meter_float(ID_meter, Reg_addr[i]);
 ID_METER_ALL=X
    }
}
void setup() {
  Serial2.begin(2400); // กำหนดความเร็วในการสื่อสาร
  Serial.begin(9600); // กำหนดความเร็วในการสื่อสาร
  Serial3.begin(9600); // กำหนดความเร็วในการสื่อสาร
  while (!Serial) ; // wait for serial
  delay(200);
  Serial.println("ArduinoAll DataLogger Shield Test");
  pinMode(SS, OUTPUT);
// ตรวจสอบการใช้การ์ดหน่วยความจำ
  if (!SD.begin(10, 11, 12, 13)) {
```

```
    Serial.println("SD Card initialization failed!");
    return;
  }
  Serial.println("SD Card OK.");
  //ReadText();
}
void loop() {
  //Serial.println("LOOP");
  GET_METER();
  Serial.println(); // แสดงค่าปริมาณทางไฟฟ้าที่มอนิเตอร์
  /*Serial.print("Voltage = "); Serial.print(DATA_METER[0]);Serial.println(" VAC");
  Serial.print("Current = "); Serial.print(DATA_METER[1]);Serial.println(" Amps");
  Serial.print("Active Power= "); Serial.print(DATA_METER[2]);Serial.println(" Watts");
  Serial.print("Power Factor = "); Serial.println(DATA_METER[3]);
  Serial.print("Total Active Energy = "); Serial.print(DATA_METER[5]);Serial.println("
kWh");*/
  time = Now()+" SenSerValue"; // เก็บค่าลงบนการ์ดความจำ
  String temp = Now() + " V = " + String(DATA_METER[0]) + " V" +
    " I = " + String(DATA_METER[1]) + " A" +
    " P = " + String(DATA_METER[2]) + " W" +
    " PF = " + String(DATA_METER[3]) +
    " E = " + String(DATA_METER[5]) + " kWh";
   String sentToMCU = String(DATA_METER[0]) + "," +
```

```
    String(DATA_METER[1]) + "," +
    String(DATA_METER[2]) + "," +
    String(DATA_METER[3]) + "," +
    String(DATA_METER[5]);
  Serial.println(temp);
  if (!Serial3.available()) {
    Serial3.println(sentToMCU);
    Serial.println("SEND");
  }
  else {
    Serial.println("NO SEND");
  }
  delay(5000);
  WriteText(temp);
  delay(6000);
  Serial.println("time");
}
void ReadText(){
  // re-open the file for reading:
char dayCopy[50];
  Day += tmYearToCalendar(tm.Year);
  if (tm.Month < 10) {
    Day += "0";
```

```
}
  Day += tm.Month;
  if (tm.Day < 10) {
    Day += "0";
  }
  Day += tm.Day;
  Day += ".txt";
 Serial.println(Day);
  Day.toCharArray(dayCopy, 50);
  myFile = SD.open(dayCopy);
  if (myFile) {
    Serial.println(Day);    // read from the file until there's nothing else in it:
    while (myFile.available()) {
      Serial.write(myFile.read());
    }
    // close the file:
    myFile.close();
  }
  else {
    // if the file didn't open, print an error:
    Serial.println("error opening test.txt");
  }
}
```

```
void WriteText(String txt){
  //Serial.println(txt);
char dayCopy[50];
  Day += tmYearToCalendar(tm.Year);
  if (tm.Month < 10) {
    Day += "0";
  }
  Day += tm.Month;
  if (tm.Day < 10) {
    Day += "0";
  }
  Day += tm.Day;
  Day += ".txt";
  Serial.println(Day);
  Day.toCharArray(dayCopy, 50);
  myFile = SD.open(dayCopy, FILE_WRITE);
  if (myFile) {
    myFile.println(txt);
    myFile.close();  else {
    // if the file didn't open, print an error:
    Serial.println("error opening test.txt");
  }
}
```

```
//สร้างตัวแปรสตริงเวลาNow()
String Now(){
  //Serial.println("now()");
  String time = "";
  if (RTC.read(tm)) { //ถ้าสามารถอ่านข้อมูลจากRTCได้สร้างตัวแปรสริงtime
     //   time = String(tm.Hour+":"+tm.Minute+":"+tm.Secnd+" DAY :
"+tm.Day+"/"+tm.Month+"/"+tmYearToCalendar(tm.Year));
    time+=tm.Hour;
    time+=":";
    time+=tm.Minute;
    time+=":";
    time+=tm.Second;
    time+=" DAY : ";
    time+=tm.Day;
    time+="/";
    time+=tm.Month;
    time+="/";
    time+=tmYearToCalendar(tm.Year);
      }
  else { //ถ้าไม่สามารถอ่านข้อมูลจากRTCให้แสดงผล
    time = "NO";
    if (RTC.chipPresent()) {
      Serial.println("The DS1307 is stopped.  Please run the SetTime");
```

```
      Serial.println("example to initialize the time and begin running.");
      Serial.println();
    }
    else {
      Serial.println("DS1307 read error!  Please check the circuitry.");
      Serial.println();
    }
  }
  //Serial.println(time);
  return time;
}
/*-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
```

ไลบรารี่ REG_SDM120.h จะถูกไปใช้ใน #include "REG_SDM120.h" ของโปรแกรมหลัก

```
 #include "REG_SDM120.h"

#define ID_meter 0x01

#define Total_of_Reg 6

#define Reg_Volt                0x0000 // 0.

#define Reg_Current             0x0006 // 1.

#define Reg_ActivePower         0x000C // 2.

 #define Reg_PowerFactor        0x001E // 3.

#define Reg_Frequency           0x0046 // 4.

#define Reg_TotalActiveEnergy   0x0156 // 5.

uint16_t const Reg_addr[6] = {

Reg_Volt, Reg_Current,

Reg_ActivePower,

Reg_PowerFactor,

Reg_Frequency,

Reg_TotalActiveEnergy

};

 float DATA_METER [Total_of_Reg] ;
```

โปรแกรมส่งข้อมูลให้ NETPIE ที่ใช้กับมอดูลสื่อสารไร้สาย

```
/*  NETPIE ESP8266 basic sample                            */
/*  More information visit : https://netpie.io             */
#include <ESP8266WiFi.h>              //ไลบรารีเรียกใช้ ESP8266WiFi
#include <MicroGear.h>                //ไลบรารีเรียกใช้ MicroGear
#include <SoftwareSerial.h>           //ไลบรารีเรียกใช้ขาเชื่อมต่อพอตเพิ่ม
const char* ssid     = "SUKKAI SABAIJAI Floor3";      // ชื่อ Wi-Fi ที่ต้องการเชื่อมต่อ
const char* password = "0873139514";                  // รหัสผ่าน Wi-Fi
String V,I,P,PF,E;                                    //  ตัวแปรที่ต้องการส่งค่า
//ค่าสำหรับเชื่อมต่อ NETPIE
#define APPID   "DeviceMeter"
#define KEY     "cGRtjaxZrD1rE7m"
#define SECRET  "AC6U3jxbJTY4X0rQfGCWAexas"
#define ALIAS   "mcu001"
#define FEEDID  "FEEDMeter002"
#define APIKEY  "ib5VMPGK1uUs3WY0zAZlBfsXxNPj8Gda"
// ประกาศตัวแปรต่าง ๆ
WiFiClient client;
SoftwareSerial magaCore(13,15);
int timer = 0;
MicroGear microgear(client);
//ถ้ามีข้อมูลใหม่มาถึงให้ทำดังนี้
void onMsghandler(char *topic, uint8_t* msg, unsigned int msglen) {
```

```
    Serial.print("Incoming message --> ");
    msg[msglen] = '\0';
    Serial.println((char *)msg);
}
void onFoundgear(char *attribute, uint8_t* msg, unsigned int msglen) {
    Serial.print("Found new member --> ");
    for (int i=0; i<msglen; i++)
        Serial.print((char)msg[i]);
    Serial.println();
}
void onLostgear(char *attribute, uint8_t* msg, unsigned int msglen) {
    Serial.print("Lost member --> ");
    for (int i=0; i<msglen; i++)
        Serial.print((char)msg[i]);
    Serial.println();
}
/* When a microgear is connected, do this */ เมื่อเชื่อมต่อไมโครเกียแล้วให้ทำต่อไปนี้
void onConnected(char *attribute, uint8_t* msg, unsigned int msglen) {
    Serial.println("Connected to NETPIE...");
    /* Set the alias of this microgear ALIAS */
    microgear.setAlias(ALIAS);
}
// การตั้งค่าส่วนต่าง ๆ ให้กับ Wi-Fi ,microgear และ พอร์ตสื่อสาร
```

```
void setup() {
     /* Add Event listeners */
    /* Call onMsghandler() when new message arraives */
    microgear.on(MESSAGE,onMsghandler);
  /* Call onFoundgear() when new gear appear */
    microgear.on(PRESENT,onFoundgear);
 /* Call onLostgear() when some gear goes offline */
    microgear.on(ABSENT,onLostgear);
 /* Call onConnected() when NETPIE connection is established */
    microgear.on(CONNECTED,onConnected);
    Serial.begin(115200);
    Serial.println("Starting...");
    /* Initial WIFI, this is just a basic method to configure WIFI on ESP8266.
*/
    // ส่วนในการเชื่อมต่อ Wi-Fi
    if (WiFi.begin(ssid, password)) {
        while (WiFi.status() != WL_CONNECTED) {
            delay(500);
            Serial.print(".");
        }
    }
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
```

```
    Serial.println(WiFi.localIP());
    /* Initial with KEY, SECRET and also set the ALIAS here */
    microgear.init(KEY,SECRET,ALIAS);
    /* connect to NETPIE to a specific APPID */
    microgear.connect(APPID);
    magaCore.begin(9600);
}
void loop() {
 // การสื่อสารระหว่าง Node MCU กับ ไมโครคอลโทเลอร์ (Arduino Mega 2560)
   if (magaCore.available()) {
      String resiveFormMaga = magaCore.readString();
    V = getValue(resiveFormMaga, ',', 0);
    I = getValue(resiveFormMaga, ',', 1);
    P = getValue(resiveFormMaga, ',', 2);
    PF = getValue(resiveFormMaga, ',', 3);
    E = getValue(resiveFormMaga, ',', 4);
    }
    /* To check if the microgear is still connected */ / ตรวจสอบการเชื่อมต่อกับเว็บไซต์
NETPIE
    if (microgear.connected()) {
        Serial.println("connected");
        /* Call this method regularly otherwise the connection may be lost */
        microgear.loop();
```

```
        if (timer >= 1000) {
            Serial.println("Publish...");
            /* Chat with the microgear named ALIAS which is myself */
// ตรวจสอบว่าได้รับข้อมูลจากมิเตอร์หรือไม่
  if (V && I && P && PF && E && V != "" && I != "" && P != "" && PF != "" && E != "") {
              String data2feed = "{\"volt\":" + (String)V + ",\"power\":" + (String)P +
",\"current\":" + (String)I + ",\"PF\":" + (String)PF + ",\"E\":" + (String)E + "}";
// ส่งข้อมูลเพื่อทำการเขียนกราฟที่ NETPIE
              microgear.writeFeed(FEEDID,data2feed);
              String data2freeboard = (String)V + "," + (String)P + "," + (String)I + "," +
(String)PF + "," + (String)E;
// ส่งข้อมูลเพื่อแสดงผลที่ NETPIE
              microgear.chat(ALIAS,String(data2freeboard));
              Serial.println(data2freeboard);
            }
            Serial.println("No Data...");
            timer = 0;
        }
        else timer += 100;
    }
// ถ้าการเชื่อมต่อขัดข้องจะทำการเชื่อมต่อใหม่อีกครั้ง
 else {
        Serial.println("connection lost, reconnect...");
```

```
        if (timer >= 5000) {
            microgear.connect(APPID);
            timer = 0;
        }
        else timer += 100;
    }
    delay(100);
}
// ฟังก์ชันสำหรับแยกข้อมูลเป็นส่วนจากข้อมูลที่ถูกส่งมาจาก Arduino Mega 2560
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

เอกสารข้อมูล SDM120 Modbus

SDM120-Modbus

Single-Phase Multifunction DIN rail Meter

- Measures kWh, Kvarh, KW, Kvar, KVA, PF, Hz, dmd, V, A, etc.
- Di-directional measurement IMP & EXP
- Two pulse outputs
- RS485 Modbus
- Din rail mounting 17.5mm
- 45A direct connection
- Better than Class 1 accuracy

User Manual V2.5

2015

Application

The energy-meters “with a blue back-lighted LCD screen for prefect reading” are used to measure single-phase like residential, utility and Industrial application. The unit measures and displays various important electrical parameters, and provide a RS485 communication port for remote reading and monitoring. Bi-directional energy measurement makes the unit a good choice for solar PV energy metering. The compact design and din rail installation provides a easy and economical solution for your metering demand.

General Specifications

Voltage AC (Un)	230V
Voltage Range	176~276V AC
Base Current (Ib)	5A
Max. Current (Imax)	45A
Mini Current (Imin)	0.25A
Starting current	0.4% of Ib
Power consumption	<2W/10VA
Frequency	50/60Hz(±10%)
AC voltage withstand	4KV for 1 minute
Impulse voltage withstand	6KV-1.2uS wavform
Overcurrent withstand	30Imax for 0.01s
Pulse output rate	1000imp/kWh (default)
	100/10/1 imp/kWh/kVarh (configurable)
Display	LCD with blue backlit
Max. Reading	99999.9kWh

Accuracy

Voltage	0.5% of range maximum
Current	0.5% of nominal
Frequency	0.2% of mid-frequency
Power factor	1% of Unity
Active power	1% of range maximum
Reactive power	1% of range maximum
Apparent power	1% of range maximum
Active energy	Class 1 IEC62053-21
	Class B EN50470-3
Reactive energy	1% of range maximum

Environment

Operating temperature	-25℃ to +55℃
Storage and transportation temperature	-40℃ to +70℃
Reference temperature	23℃±2℃
Relative humidity	0 to 95%, non-condensing
Altitude	up to 2500m
Warm up time	10s
Installation category	CAT II
Mechanical Environment	M1
Electromagnetic environment	E2
Degree of pollution	2

Output

Pulse Output

The meter provides two pulse outputs. Both pulse outputs are passive type.

Pulse output 1 is configurable. The pulse output can be set to generate pulses to represent total / import/export kWh or kVarh.

The pulse constant can be set to generate 1 pulse per: 0.001(default) /0.01/0.1/1kWh/kVarh.

Pulse width: 200/100/60ms

Pulse output 2 is non-configurable. It is fixed up with total kWh. The constant is 1000imp/kWh.

RS485 output for Modbus RTU

The meter provides a RS485 port for remote communication. Modbus RTU is the protocol applied. For Modbus RTU, the following RS485 communication parameters can be configured from the Set-up menu.

Baud rate: 1200, 2400, 4800, 9600

Parity: NONE/EVEN/ODD

Stop bits: 1 or 2

Modbus Address: 1 to 247

Mechanics

Din rail dimensions	17.5x119x62 (WxHxD) DIN 43880
Mounting	DIN rail 35mm
Sealing	IP51 (indoor)
Material	self-extinguishing UL94V-0

Initialization Display

When it is powered on, the meter will initialize and do self-checking.

1		Full Screen It will last for 3 seconds.
2		Software version It will last for 3 seconds.

After the self-checking program, the meter display will show the total active energy (kWh)

Scroll Display by button

There is a button on the front of the meter. After initialization and self-checking program, the meter display the measured values. The default page is total kWh. If the user wants to check other information, he needs to press the scroll button on the front panel.

	Click the button, the LCD display will scroll the measurements.
	Keep pressing the button for 3 seconds, the meter will get into set-up mode.

1		Total active energy（kWh） Display format：0000.00→9999.99→10000.0→99999.9→0000.00
1-1		Import active energy（kWh） Display format：0000.00→9999.99→10000.0→99999.9→0000.00
1-2		Export active energy（kwh） Display format：0000.00→9999.99→10000.0→99999.9→0000.00

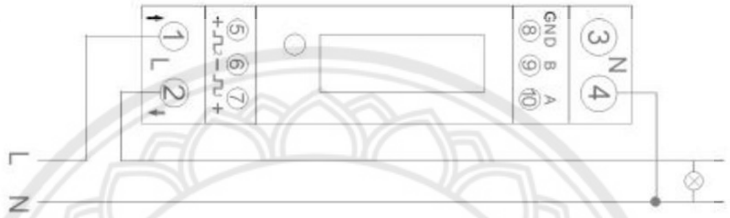
2		Voltage (V)
3		Current (A)
4		Active power (W)
5		Frequency (F)
6		Power factor (PF)
7		Modbus Address (ID) Default: 001
8		Baud rate Default : 2400bps
9		Parity None/even/odd are optional Default: none

Set-up Mode

To get into Set-up Mode, the user need keep pressing the button for 3 seconds, the meter LCD will shows "-SET-".

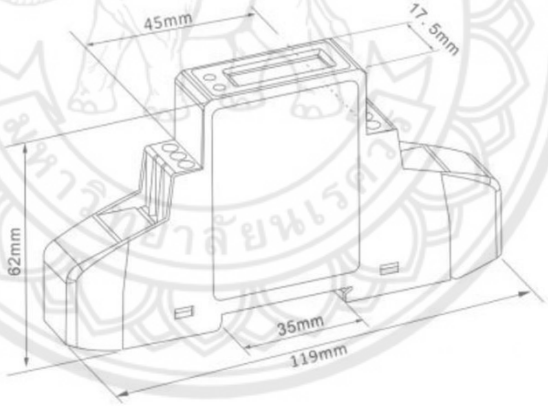
The user can program the meter parameters by sending correct command via RS485 port.
The protocol is Modbus RTU. For the details. Please look at the *"SDM120-Modbus protocol"*.

Wiring diagram

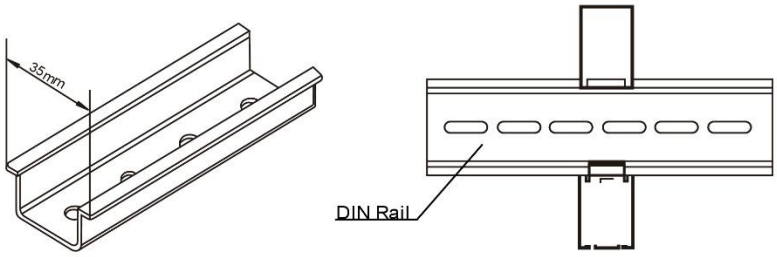


1：L-in 2: L-out 3 / 4: N 5 / 6 / 7: PO 2+ / Com / PO 1+ 8 / 9 / 10：GND/RS485 B/A

Dimensions



Installation



ภาคผนวก ค

เอกสารข้อมูล ESP8266 nodeMCU

Handson Technology

User Manual V1.2

ESP8266 NodeMCU WiFi Devkit

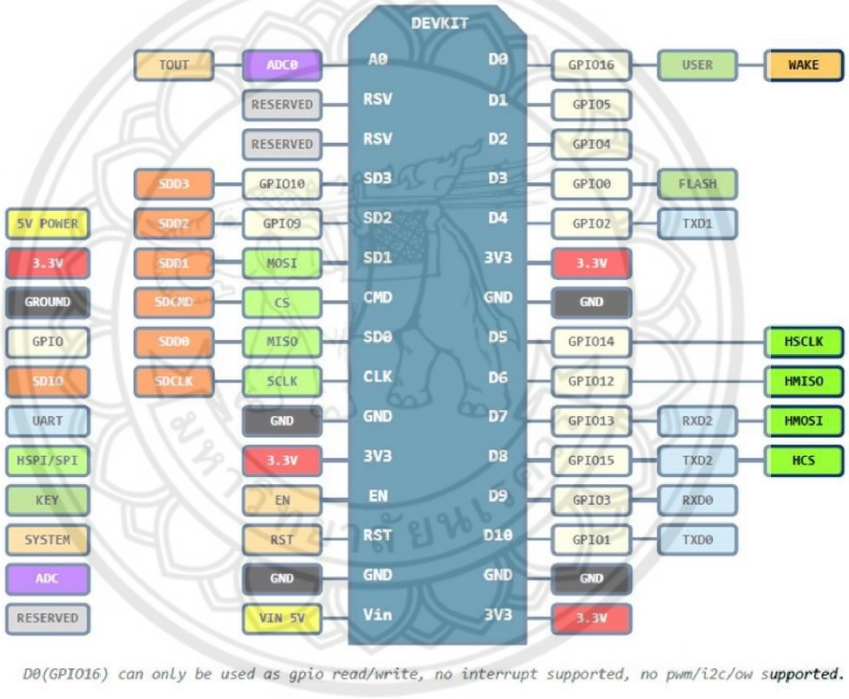
The ESP8266 is the name of a micro controller designed by Espressif Systems. The ESP8266 itself is a self-contained WiFi networking solution offering as a bridge from existing micro controller to WiFi and is also capable of running self-contained applications.

This module comes with a built in USB connector and a rich assortment of pin-outs. With a micro USB cable, you can connect NodeMCU devkit to your laptop and flash it without any trouble, just like Arduino. It is also immediately breadboard friendly.

1. Specification:

- Voltage:3.3V.
- Wi-Fi Direct (P2P), soft-AP.
- Current consumption: 10uA~170mA.
- Flash memory attachable: 16MB max (512K normal).
- Integrated TCP/IP protocol stack.
- Processor: Tensilica L106 32-bit.
- Processor speed: 80~160MHz.
- RAM: 32K + 80K.
- GPIOs: 17 (multiplexed with other functions).
- Analog to Digital: 1 input with 1024 step resolution.
- +19.5dBm output power in 802.11b mode
- 802.11 support: b/g/n.
- Maximum concurrent TCP connections: 5.

2. Pin Definition:



D0(GPIO16) can only be used as gpio read/write, no interrupt supported, no pwm/i2c/ow supported.

3. Using Arduino IDE

ภาคผนวก ง

เอกสารข้อมูล RS-485

19-0740; Rev 0; 1/07

MAXIM

Half-Duplex RS-485-/RS-422-Compatible Transceiver with AutoDirection Control

General Description

The MAX13487E/MAX13488E +5V, half-duplex, ±15kV ESD-protected RS-485/RS-422-compatible transceivers feature one driver and one receiver. The MAX13487E/MAX13488E include a hot-swap capability to eliminate false transitions on the bus during power-up or live insertion.

The MAX13487E/MAX13488E feature Maxim's proprietary AutoDirection control. This architecture makes the devices ideal for applications, such as isolated RS-485 ports, where the driver input is used in conjunction with the driver-enable signal to drive the differential bus.

The MAX13487E features reduced slew-rate drivers that minimize EMI and reduce reflections caused by improperly terminated cables, allowing error-free transmission up to 500kbps. The MAX13488E driver slew rate is not limited, allowing transmit speeds up to 16Mbps.

The MAX13487E/MAX13488E feature a 1/4-unit load receiver input impedance, allowing up to 128 transceivers on the bus. These devices are intended for half-duplex communications. All driver outputs are protected to ±15kV ESD using the Human Body Model. The MAX13487E/MAX13488E are available in an 8-pin SO package. The devices operate over the extended -40°C to +85°C temperature range.

Applications

Isolated RS-485 Interfaces
Utility Meters
Industrial Controls
Industrial Motor Drives
Automated HVAC Systems

Pin Configuration and Typical Application Circuit appear at end of data sheet.

Features

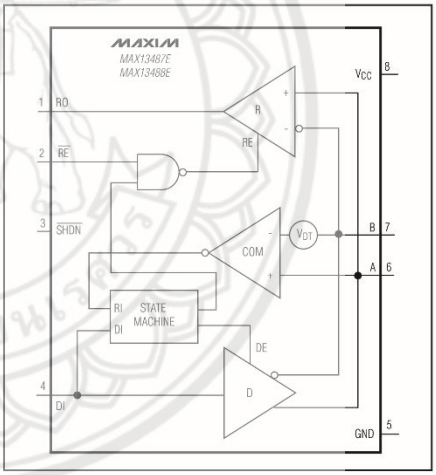
- **+5V Operation**
- **AutoDirection Enables Driver Automatically on Transmission**
- **Hot-Swappable for Telecom Applications**
- **Enhanced Slew-Rate Limiting Facilitates Error-Free Data Transmission (MAX13487E)**
- **High-Speed Version (MAX13488E) Allows for Transmission Speeds Up to 16Mbps**
- **Extended ESD Protection for RS-485 I/O Pins ±15kV Human Body Model**
- **1/4-Unit Load, Allowing Up to 128 Transceivers on the Bus**
- **8-Pin SO Package**

Ordering Information/ Selector Guide

PART	PIN-PACKAGE	SLEW-RATE LIMITED	PKG CODE
MAX13487EESA+	8 SO	Yes	S8-2
MAX13488EESA+	8 SO	No	S8-2

+Denotes a lead-free package
All devices operate over the -40°C to +85°C temperature range.

Functional Diagram



For pricing, delivery, and ordering information, please contact Maxim/Dallas Direct! at 1-888-629-4642, or visit Maxim's website at www.maxim-ic.com.

Half-Duplex RS-485-/RS-422-Compatible Transceiver with AutoDirection Control

ABSOLUTE MAXIMUM RATINGS

(All voltages referenced to GND.)
Supply Voltage V_{CC} ..... +6V
SHDN, RE, DI ..... -0.3V to +6
A, B ..... -8V to +13V
Short-Circuit Duration (RO, A, B) to GND ..... Continuous
Continuous Power Dissipation (T_A = +70°C)
8-Pin SO (derate 5.9mW/°C above +70°C) 471mW

Operating Temperature Range ..... -40°C to +85°C
Junction Temperature ..... +150°C
Storage Temperature Range ..... -65°C to +150°C
Lead Temperature (soldering 10s) ..... +300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS

(V_{CC} = +5V ±5%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted. Typical values are at V_{CC} = +5V and T_A = +25°C.) (Note 1)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
DRIVER						
Differential Driver Output	V_{OD}	R_{DIFF} = 100Ω, Figure 1	2.0		V_{CC}	V
		R_{DIFF} = 54Ω, Figure 1	1.5			
		No load			V_{CC}	
Driver Common-Mode Output Voltage	V_{OC}	R_L = 100Ω or 54Ω, Figure 1		V_{CC} / 2	3	V
Driver Disable Threshold	V_{DT}	Figure 2 (Note 2)	+0.6		+1	V
Input-High Voltage	V_{IH}	DI, $\overline{SHDN}$, $\overline{RE}$	2.0			V
Input-Low Voltage	V_{IL}	DI, $\overline{SHDN}$, $\overline{RE}$			0.8	V
Input Current	I_{IN}	DI, $\overline{SHDN}$, $\overline{RE}$			±1	µA
Driver Short-Circuit Output Current (Note 3)	I_{OSD}	0V ≤ V_{OUT} ≤ +12V	+50		+250	mA
		-7V ≤ V_{OUT} ≤ 0V	-250		-50	
Driver Short-Circuit Foldback Output Current (Note 3)	I_{OSDF}	(V_{CC} - 1V) ≤ V_{OUT} ≤ +12V	20			mA
		-7V ≤ V_{OUT} ≤ 0V			-20	
RECEIVER						
Input Current (A and B)	$I_{A, B}$	DI = V_{CC}, V_{CC} = GND or +5V; V_{IN} = +12V			250	µA
		DI = V_{CC}, V_{CC} = GND or +5V; V_{IN} = -7V	-200			
Receiver Differential Threshold Voltage	V_{TH}	-7V ≤ V_{CM} ≤ +12V	-200		+200	mV
Receiver Input Hysteresis	ΔV_{TH}	V_A + V_B = 0V		25		mV
Output-High Voltage	V_{OH}	I_O = -1.6mA, V_A - V_B > V_{TH}	V_{CC} - 1.5			V
Output-Low Voltage	V_{OL}	I_O = 1mA, V_A - V_B < -V_{TH}			0.4	V
Tri-State Output Current at Receiver	I_{OZR}	0V ≤ V_O ≤ V_{CC}			±1	µA
Receiver Input Resistance	R_{IN}	-7V ≤ V_{CM} ≤ +12V	48			kΩ
Receiver Output Short-Circuit Current	I_{OSR}	0V ≤ V_{RO} ≤ V_{CC}	±7		±95	mA

ELECTRICAL CHARACTERISTICS (continued)

(V_{CC} = +5V ±5%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted. Typical values are at V_{CC} = +5V and T_A = +25°C.) (Note 1)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
POWER SUPPLY						
Supply Voltage	V_{CC}		4.75		5.25	V
Supply Current	I_{CC}	$\overline{SHDN}$ = 1, $\overline{RE}$ = 0, no load			4.5	mA
Shutdown Supply Current	$I_{\overline{SHDN}}$	$\overline{SHDN}$ = 0			10	µA
ESD PROTECTION						
ESD Protection (A, B)		Air Gap Discharge IEC61000-4-2 (MAX13487E)		±15		kV
		Human Body Model		±15		
ESD Protection (All Other Pins)		Human Body Model		±2		kV

SWITCHING CHARACTERISTICS—MAX13487E

(V_{CC} = +5V ±5%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted. Typical values are at V_{CC} = +5V and T_A = +25°C.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
DRIVER						
Driver Propagation Delay	t_{DPLH}	R_L = 110Ω, C_L = 50pF, Figures 2 and 3	200		1000	ns
	t_{DPHL}		200		1000	
Driver Differential Output Rise or Fall Time	t_{HL}	R_L = 110Ω, C_L = 50pF, Figures 2 and 3	200		900	ns
	t_{LH}		200		900	
Maximum Data Rate			500			kbps
Driver Disable Delay	t_{DDD}	Figure 3			2500	ns
Driver Enable from Shutdown to Output High	$t_{DZH(SHDN)}$	Figure 4			5.5	µs
Driver Enable from Shutdown to Output Low	$t_{DZL(SHDN)}$	Figure 4			5.5	µs
Time to Shutdown	t_{SHDN}		50	340	700	ns
RECEIVER						
Receiver Propagation Delay	t_{RPLH}	C_L = 15pF, Figures 5 and 6			80	ns
	t_{RPHL}				80	
Receiver Output Skew	t_{RSKEW}	C_L = 15pF, Figure 6			13	ns
Maximum Data Rate			500			kbps
Receiver Enable to Output High	t_{RZH}	Figure 7			50	ns
Receiver Enable to Output Low	t_{RZL}	Figure 7			50	ns
Receiver Disable Time from High	t_{RHZ}	Figure 7			50	ns
Receiver Disable Time from Low	t_{RLZ}	Figure 7			50	ns
Receiver Enable from Shutdown to Output High	$t_{RZH(SHDN)}$	Figure 8			2200	ns

SWITCHING CHARACTERISTICS—MAX13487E (continued)

(V_{CC} = +5V ±5%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted. Typical values are at V_{CC} = +5V and T_A = +25°C.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Receiver Enable from Shutdown to Output Low	$t_{RZL(SHDN)}$	Figure 8			2200	ns
Receiver Enable Delay	t_{RED}	Figure 3			70	ns
Time to Shutdown	t_{SHDN}		50	340	700	ns

SWITCHING CHARACTERISTICS—MAX13488E

(V_{CC} = +5V ±5%, T_A = T_{MIN} to T_{MAX}, unless otherwise noted. Typical values are at V_{CC} = +5V and T_A = +25°C.)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
DRIVER						
Driver Propagation Delay	t_{DPLH}	R_L = 110Ω, C_L = 50pF, Figures 2 and 3			50	ns
	t_{DPHL}				50	
Driver Differential Output Rise or Fall Time	t_{HL}	R_L = 110Ω, C_L = 50pF, Figures 2 and 3			15	ns
	t_{LH}				15	
Maximum Data Rate			16			Mbps
Driver Disable Delay	t_{DDD}	Figure 3			70	ns
Driver Enable from Shutdown to Output High	$t_{DZH(SHDN)}$	Figure 4			2.2	µs
Driver Enable from Shutdown to Output Low	$t_{DZL(SHDN)}$	Figure 4			2.2	µs
Time to Shutdown	t_{SHDN}		50	340	700	ns
RECEIVER						
Receiver Propagation Delay	t_{RPLH}	C_L = 15pF, Figures 5 and 6			80	ns
	t_{RPHL}				80	
Receiver Output Skew	t_{RSKEW}	C_L = 15pF, Figure 6			13	ns
Maximum Data Rate			16			Mbps
Receiver Enable to Output High	t_{RZH}	Figure 7			50	ns
Receiver Enable to Output Low	t_{RZL}	Figure 7			50	ns
Receiver Disable Time from High	t_{RHZ}	Figure 7			50	ns
Receiver Disable Time from Low	t_{RLZ}	Figure 7			50	ns
Receiver Enable from Shutdown to Output High	$t_{RZH(SHDN)}$	Figure 8			2200	ns