อุปกรณ์ตรวจวัดอุณหภูมิโดยใช้เทคโนโลยีสมองกลฝังตัว
TEMPERATURE MEASURING DEVICE USING EMBEDDED SYSTEM

นายชนาวุธ แก่นจันทร์แดง
นายชลเทพ จันทร์สิริ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมอัตโนมัติ
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2560

อุปกรณ์ตรวจวัดอุณหภูมิโดยใช้เทคโนโลยีสมองกลฝังตัว

TEMPERATURE MEASURING DEVICE USING EMBEDDED SYSTEM

นายชนาวุธ แก่นจันทร์แดง
นายชลเทพ จันทร์สิริ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมอัตโนมัติ
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2560

TEMPERATURE MEASURING DEVICE USING EMBEDDED SYSTEM

CHANAWUT KAENJANDANG
CHOLATHEP JUNSIRI

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENT FOR THE DEGREE OF
BACHELOR OF ENGINEERING IN AUTOMATION ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2017

สาขาวิชาวิศวกรรมอัตโนมัติ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

.....

หัวข้อปริญญานิพนธ์ อุปกรณ์ตรวจวัดอุณหภูมิโดยใช้เทคโนโลยีสมองกลฝังตัว
TEMPERATURE MEASURING DEVICE USING EMBEDDED SYSTEM

นักศึกษาผู้จัดทำ นายชนาวุธ แก่นจันทร์แดง **รหัสนักศึกษา** 57010247
นายชลเทพ จันทร์สิริ **รหัสนักศึกษา** 57010268

ปริญญา วิศวกรรมศาสตรบัณฑิต

สาขาวิชา วิศวกรรมอัตโนมัติ

ปีการศึกษา 2560

อาจารย์ผู้ควบคุมวิทยานิพนธ์	ลายมือชื่อ
รศ.ดร.พิทยา ปานนิล	
ดร.อภินัย ฤกษ์รัตน์	

หัวข้อปริญญานิพนธ์	อุปกรณ์ตรวจวัดอุณหภูมิโดยใช้เทคโนโลยีสมองกลฝังตัว		
นักศึกษาผู้จัดทำ	นายชนาวุธ	แก่นจันทร์แดง	**รหัสนักศึกษา** 57010247
	นายชลเทพ	จันทร์สิริ	**รหัสนักศึกษา** 57010268
อาจารย์ที่ปรึกษา	รศ.ดร.พิทยา	ปานนิล	
	ดร.อภินัย	ฤกษ์รัตน์	
ปีการศึกษา	2560		

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้นำเสนอเทคนิคในการสร้างระบบวัดอุณหภูมิแบบไร้สายโดยใช้เทคโนโลยีสมองกลฝังตัว โดยระบบที่นำเสนอประกอบด้วยเซ็นเซอร์วัดอุณหภูมิรุ่น PT1000 วงจรบริดจ์ และโมดูล Arduino ESPino32 สำหรับใช้เป็นตัวควบคุมและอุปกรณ์ในการรับส่งสัญญาณ Wi-Fi เพื่อส่งข้อมูลและสถานะของระบบไปยัง SCADA Software โดยใช้โปรโตคอล Modbus TCP / IP ในการสื่อสารข้อมูลระหว่างโมดูล Arduino ESPino32 กับ SCADA Software สำหรับในส่วนของข้อมูลประกอบไปด้วยสถานะของระบบและอุณหภูมิที่ทำการวัด ซึ่งจะถูกส่งไปยัง SCADA Software เพื่อทำการแสดงผล จากผลการทดสอบระบบวัดอุณหภูมิแบบไร้สายโดยใช้อุปกรณ์ฝังตัวที่พัฒนาขึ้นกับพลานต์ควบคุมอุณหภูมิยืนยันให้เห็นว่าระบบที่นำเสนอสามารถทำงานได้อย่างถูกต้องและสามารถแสดงสถานะของอุปกรณ์และอุณหภูมิที่ทำการวัดด้วย SCADA Software ได้โดยมีค่าความผิดพลาดสูงสุดของการวัดประมาณ -2.35%

Thesis Title	TEMPERATURE MEASURING DEVICE USING EMBEDDED SYSTEM		
Student Name	Mr. Chanawut Kaenjandang	**Student ID**	57010247
	Mr. Chonlathep Junsiri	**Student ID**	57010268
Thesis Advisors	Assoc. Prof. Dr. Pittaya Pannil		
	Dr. Apinai Rerkratn		
Academic Year	2017		

ABSTRACT

This project presents a technique to implement the wireless temperature measurement system based on embedded device. The proposed system consists of the temperature sensor model PT1000, a bridge circuit and an Arduino ESPino32 module. The Arduino ESPino32 module is employed as the controller and Wi-Fi Transceiver to transmit data and system status to SCADA software. The Modbus TCP/IP protocol is used for communication between Arduino ESPino32 module and SCADA software. The transmitted data are system status and temperature, which are monitored by SCADA software. The temperature control plant is employed to test the performance of the proposed system. Experimental results verify that the proposed system works efficiently and is able to monitor device status and temperature via SCADA software with maximum error about -2.35 %

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยความกรุณาจาก รศ.ดร.พิทยา ปานนิล และ ดร.อภินัย ฤกษ์รัตน์ อาจารย์ที่ปรึกษา ที่ได้ให้คำแนะนำ ข้อเสนอแนะ ตลอดจนแก้ไขข้อบกพร่องต่าง ๆ มาโดยตลอดจนโครงงานเล่มนี้เสร็จสมบูรณ์ ขอขอบคุณ บิดา มารดา เพื่อน ๆ ที่คอยช่วยเหลือและเป็นกำลังใจที่ดีเสมอมา สุดท้ายนี้ขอขอบคุณอาจารย์และบุคลากร ภาควิชาวิศวกรรมอัตโนมัติทุกท่านตลอดระยะเวลา 4 ปี สำหรับความรู้ ประสบการณ์ คำปรึกษาคำแนะนำ ทั้งเรื่องการเรียนและแนวคิดในการดำเนินชีวิต ที่สามารถนำไปใช้ในการทำโครงงานได้และสำเร็จลุล่วงไปได้ด้วยดี จึงขอขอบคุณไว้ ณ ที่นี้ หากมีข้อผิดพลาดประการใดให้ถือเป็นความบกพร่องของผู้จัดทำ และ ขออภัยมา ณ ที่นี้ด้วย

คณะผู้จัดทำ

สารบัญ

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญ(ต่อ)

หน้า

สารบัญตาราง

หน้า

สารบัญรูป

รูปที่ **หน้า**

สารบัญรูป(ต่อ)

รูปที่ **หน้า**

สารบัญรูป(ต่อ)

รูปที่ **หน้า**

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญ

ในปัจจุบันเทคโนโลยีระบบสมองกลฝังตัว หรือ Embedded System ที่ถูกนำมาใช้กันอย่างแพร่หลายในชีวิตประจำวัน โดยคำว่าระบบสมองกลฝังตัว หมายถึง ระบบที่ใช้ตัวประมวลผล(CPU) มาควบคุมการทำงานของระบบใดระบบหนึ่งที่สนใจ ซึ่งรวมไปถึงการใช้ Microcontroller และ Microprocessor ในก่อนหน้านี้ด้วย ระบบสมองกลแบบฝังตัวเป็นระบบที่ใช้ตัวประมวลผล เพิ่มประสิทธิภาพในการทำงานของอุปกรณ์การวัดและควบคุม และมีซอฟต์แวร์ของตนเองเพื่อเขียนโปรแกรมควบคุมการทำงาน ทำให้การพัฒนาวงจรอิเล็กทรอนิกส์ ระบบการวัดและควบคุม หรือการสังเกตุการณ์ (Monitoring) ทำได้ง่ายขึ้น ราคาถูกและมีความยืดหยุ่นในการทำงานมากขึ้น การนำบอร์ดไมโครคอนโทรลเลอร์ หรือบอร์ดประมวลผลอื่น ๆ ฝังตัวเข้าไปในวงจรหรืออุปกรณ์นั้น ๆ เพื่อเพิ่มความสามารถในการทำงาน ยกตัวอย่างเช่น เตาไมโครเวฟ ที่ปัจจุบันมีโหมดการปรุงอาหารให้เลือกมากมาย หรือระบบเบรค ABS ที่ทำให้การขับขี่ปลอดภัยมากขึ้น จึงมีแนวคิดการนำระบบสมองกลฝังตัวมาใช้ในงานอุตสาหกรรมอัตโนมัติ คือการนำมาผนวกเพิ่มความสามารถให้กับเซนเซอร์วัดอุณหภูมิประเภท RTD ให้มีความสามารถที่คล้ายคลึงกับ Smart Device มากขึ้น

1.2 วัตถุประสงค์ปริญญานิพนธ์

1. ศึกษาระบบเทคโนโลยีสมองกลฝังตัวเพื่อใช้สร้างอุปกรณ์ตรวจวัดอุณหภูมิ
2. พัฒนาโปรแกรมในการสื่อสารด้วยโปรโตคอล Modbus TCP/IP ด้วย Arduino IDE
3. สร้างส่วนแสดงผลด้วยโปรแกรม ArchestrA เพื่อแสดงค่าอุณหภูมิ และ สถานะของอุปกรณ์ตรวจวัดอุณหภูมิ

1.3 ขอบเขตปริญญานิพนธ์

1. สร้างอุปกรณ์ตรวจวัดอุณหภูมิด้วย Sensor PT1000 และ Arduino ESPino32 พร้อมส่งข้อมูลไปยังส่วนแสดงผลผ่านโปรโตคอล Modbus TCP/IP
2. สามารถแสดงค่าอุณหภูมิและกราฟผ่านส่วนแสดงผลด้วยโปรแกรม ArchestrA ในหน่วยมาตรฐาน พร้อมทั้งสถานะของอุปกรณ์ตรวจวัดอุณหภูมิ
3. มีระบบตรวจสอบการทำงานผิดปกติของอุปกรณ์ตรวจวัดอุณหภูมิ พร้อมแจ้งเตือนให้ผู้ใช้ทราบผ่านส่วนแสดงผล

1.4 ขั้นตอนการศึกษา

ขั้นตอนการศึกษามีรายละเอียดดังนี้

ตารางที่ 1.1 ขั้นตอนการดำเนินงาน

ขั้นตอนการดำเนินงาน	ระยะเวลาดำเนินการ																			
	ส.ค.				ก.ย.				ต.ค.				พ.ย.				ธ.ค.			
	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1. วางแผนการทำงาน																				
2. ทดลองเชื่อมต่อกับ OPC server																				
3. ทดลองเชื่อม Modbus TCP/IP																				
4. เขียน System Platform																				
5. ติดตั้งและทดสอบการทำงานของส่วนแสดงผล																				
6. จัดทำรูปเล่มรายงาน																				

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. ได้ศึกษาและประยุกต์ใช้งานในระบบเทคโนโลยีระบบสมองกลฝังตัว
2. ได้ศึกษาการเขียนภาษา C หรือภาษาอื่น ๆ จากการใช้โปรแกรม Arduino IDE
3. ได้รับความรู้จากการพัฒนาและเพิ่มขีดความสามารถของอุปกรณ์ตรวจวัดอุณหภูมิผ่านการเขียนเงื่อนไขในส่วนแสดงผล

1.6 รายละเอียดของปริญญานิพนธ์

รายงานฉบับนี้จัดทำทั้งหมด 5 บท โดยที่แต่ละบทมีรายละเอียดดังต่อไปนี้

บทที่ 1 วัตถุประสงค์และขอบเขตของปริญญานิพนธ์

กล่าวถึง ที่มาและความสำคัญของปริญญานิพนธ์ในหัวเรื่องการนำเทคโนโลยีระบบสมองกลฝังตัวประยุกต์ใช้งาน โดยมีวัตถุประสงค์ในการออกแบบและสร้างอุปกรณ์ตรวจวัดอุณหภูมิด้วยระบบสมองกลฝังตัวและขอบเขตในการทำงาน

บทที่ 2 ทฤษฎีที่เกี่ยวข้อง

กล่าวถึงทฤษฎีต่าง ๆ ที่เกี่ยวข้องในการนำมาใช้สร้างอุปกรณ์วัดอุณหภูมิ เช่น ระบบฝังตัวในชีวิตประจำวัน การประยุกต์ใช้ ระบบการส่งข้อมูลผ่าน Protocol ของ Modbus และส่วนแสดงผลที่เขียนด้วย ArchestrA

บทที่ 3 ขั้นตอนการดำเนินงาน

กล่าวถึงการสร้างอุปกรณ์วัดอุณหภูมิด้วยระบบสมองกลฝังตัว ตั้งแต่วงจรภายใน การเขียนโปรแกรมรับค่าผ่าน Arduino IDE ลงบอร์ด ESPino32 และการดึงค่า Parameter จาก Modbus ไปใช้ในส่วนแสดงผล

บทที่ 4 การทดลอง และ ผลการทดลอง

กล่าวถึงการทดสอบอุปกรณ์ตรวจวัดอุณหภูมิ โดยการวัดเทียบกับ Digital Thermometer และการทดสอบฟังก์ชั่นต่าง ๆ (Diagnostic Massage, Trend, Historian, Device Status)

บทที่ 5 สรุปผลการทำงานและข้อเสนอแนะ

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 อุปกรณ์วัดอุณหภูมิในอุตสาหกรรม

อุปกรณ์วัดอุณหภูมิในอุตสาหกรรม มีหลากหลายชนิดแตกต่างกันขึ้นอยู่กับวัตถุประสงค์การใช้งานและความละเอียดของงานที่ต้องการจะวัด โดยมีการจำแนกประเภทของอุปกรณ์จากวัสดุที่ทำหรือวิธีการใช้งาน เช่น เทอร์โมคัปเปิ้ล (Thermocouple) , RTD และ อื่น ๆ

รูปที่ 2.1 เทอร์โมคัปเปิ้ล

2.1.1 เทอร์โมคัปเปิ้ล

คืออุปกรณ์วัดอุณหภูมิที่มีส่วนประกอบจากเส้นลวดโลหะต่างชนิดกันสองชนิดต่อเข้าด้วยกันที่ปลายข้างหนึ่ง ส่วนปลายอีกข้างหนึ่งจะถูกนำไปต่อใช้งาน ซึ่งปลายของเส้นลวดที่ต่อเข้าด้วยกันนี้เรียกว่า Hot Junction ส่วนปลายด้านที่ต่อไปใช้งานจะเรียกว่า Cold Junction ซึ่งเมื่อจุดต่อที่ Hot Junction ถูกนำไปใช้วัดอุณหภุมิและเมื่ออุณหภูมิที่วัดมีค่าสูงขึ้นก็จะทำให้เกิดแรงดันไฟฟ้าวัดค่าได้ที่จุด Cold Junction ซึ่งหลักการทำงานของเทอร์โมคัปเปิลคืออาศัยความแตกต่างของอุณหภูมิในการสร้างแรงเคลื่อนไฟฟ้าขึ้น การที่แรงเคลื่อนไฟฟ้าค่าหนึ่งจะอ้างอิงเป็นอุณหภูมิค่าหนึ่งได้ แสดงว่าความแตกต่างของอุณหภูมิที่เกิดขึ้นนั้นจะต้องอ้างอิงกับอุณหภูมิค่าคงที่ค่าหนึ่งเสมอ โดยเรียกอุณหภูมิคงที่ที่ใช้อ้างอิงนี้ว่า Reference Junction และได้มีการกำหนดให้ Reference Junction ให้เป็น 0°C เพื่อเป็นการเทียบอุณหภูมิ เพื่อให้การวัดอุณหภูมิเกิดแรงเคลื่อนไฟฟ้าที่เป็นมาตรฐานเดียวกัน และกำหนดเป็นตารางมาตรฐานแสดงค่าอุณหภูมิเทียบกับแรงเคลื่อนไฟฟ้าที่วัดได้ แต่โดยทั่วไป เทอร์โมคัปเปิลจะทำการวัดที่อุณหภูมิห้อง (เช่น 25°C) นั่นคือไม่ได้เทียบกับ 0 °C แสดงว่าค่าแรงเคลื่อนไฟฟ้าที่ได้ยังไม่ถูกต้อง

หากนำไปอ่านค่าอุณหภูมิจากตารางมาตรฐานจะผิดพลาด จึงจำเป็นต้องมีการรักษา Reference Junction เพื่อให้การวัดอุณหภูมิเทียบกับ 0°C ตลอดเวลา

ย่านการใช้งานและคุณลักษณะของเทอร์โมคัปเปิลแสดงในตารางที่ 2.1

ตารางที่ 2.1 ตารางเปรียบเทียบชนิดของเทอร์โมคัปเปิล

ชนิดเทอร์โมคัปเปิล	ย่านอุณหภูมิที่ใช้งาน (°C)	ย่านอุณหภูมิ (°C)	ค่าความผิดพลาด (°C)	ค่าความไวสูงสุด (μV/°C)
R	-50 ถึง1768.1	-50.0 ถึง 250.0	-0.02 ถึง 0.02	6
		250.0 ถึง 1200.0	-0.005 ถึง 0.005	
		1064.0 ถึง 1664.5	-0.0005ถึง0.001	
		1664.5 ถึง 1768.1	-0.001 ถึง 0.002	
J	-210 ถึง 1200	-210.0 ถึง 0.0	-0.05 ถึง 0.03	50
		0.0 ถึง 760.0	-0.04 ถึง 0.04	
		760.0 ถึง 1200.0	-0.04 ถึง 0.03	
K	-270 ถึง 1372	-270.0 ถึง 0.0	-0.02 ถึง 0.04	50
		0.0 ถึง 500.0	-0.05 ถึง 0.04	
		500.0 ถึง 1372.0	-0.05 ถึง 0.06	
T	-270 ถึง 400	-200.0 ถึง 0.0	-0.02 ถึง 0.04	60
		0.0 ถึง 400.0	-0.03 ถึง 0.03	

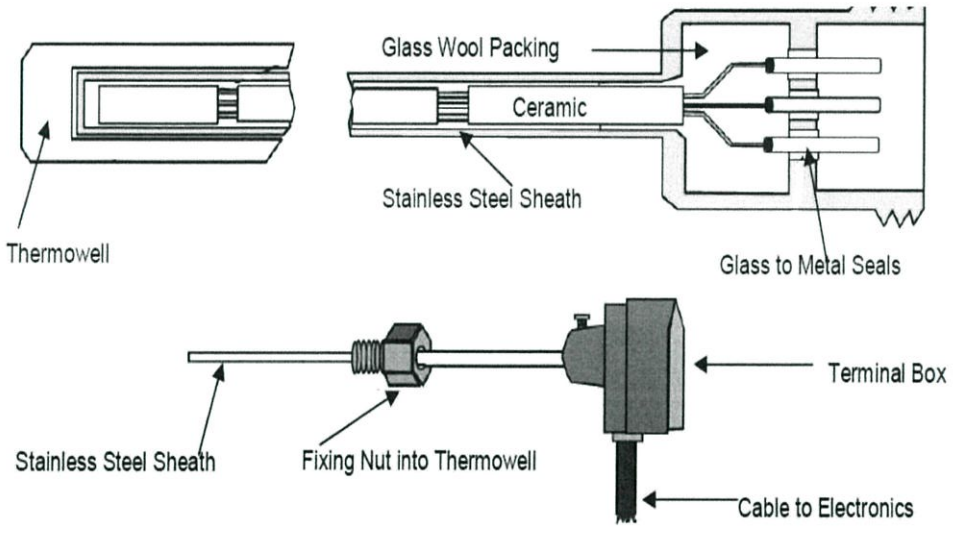
2.1.2 RTD

รูปที่ 2.2 RTD Pt100

RTD เป็นคำที่ย่อมาจาก Resistance Temperature Detector ซึ่งมีความหมายว่า อุปกรณ์ที่ใช้ในการวัดอุณหภูมิแล้วเปลี่ยนแปลงเป็นค่าความต้านทาน โดยค่าความต้านทานจะมีค่าเพิ่มมากขึ้นเมื่ออุณหภูมิมีค่าสูงขึ้น ซึ่ง RTD โดยทั่วไปแล้วจะทำมาจากแพลตตินั่ม เนื่องจากโลหะชนิดนี้จะให้ผลตอบสนองออกมาค่อนข้างเป็นเป็นเส้นตรง ดังนั้นในงานอุตสาหกรรมโดยทั่วไปเราจึงมักจะพบเจอ RTD ที่เรียกว่า RTD Pt100 Ohm ได้บ่อยครั้งซึ่ง RTD Pt100 มีความหมายมาจาก RTD ที่ทำจากแพลตตินั่มโดยวัดอุณหภูมิที่ 0 °C สัญญาณทางไฟฟ้าออกมาที่ 100 Ohm ดังนั้นจึงมักจะพบเจอบ่อยๆ ที่จะเรียกตัววัดอุณหภูมิชนิดนี้สั้น ๆ ว่า RTD หรือ PT100 ซึ่งโดยทั่วไปแล้วเซ็นเซอร์ชนิดนี้นิยมใช้งานวัดในช่วงอุณหภูมิ -50...400 °C

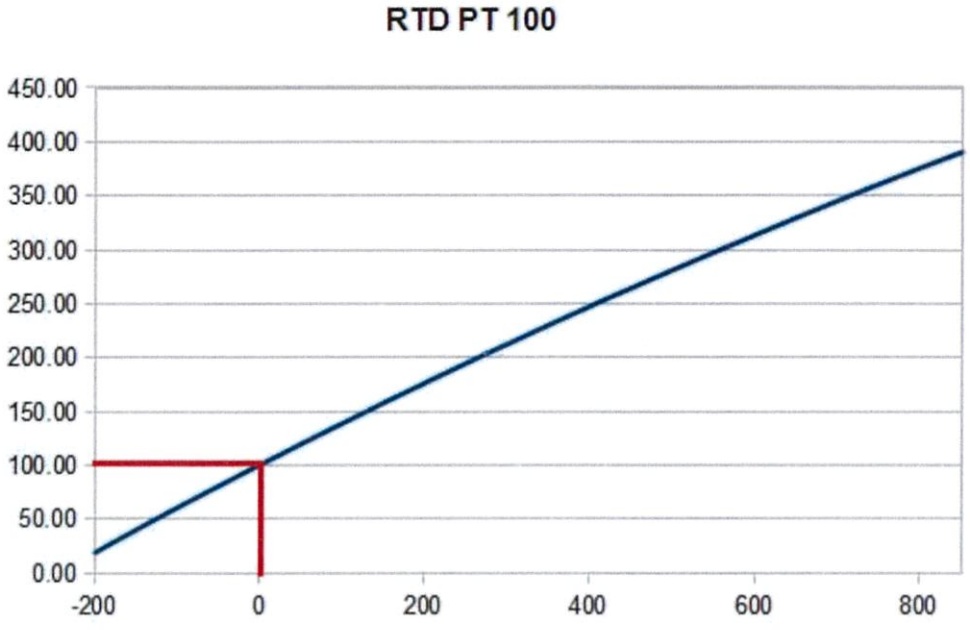
รูปที่ 2.3 อุปกรณ์วัดอุณหภูมิ Pt1000

PRT ย่อมาจาก Platinum Resistance Thermometers ซึ่งเรียกสั้น ๆ ว่า Pt ซึ่งคือเซนเซอร์วัดอุณหภูมิซึ่งใช้ลวดความต้านทานเป็นแพลตตินั่ม เนื่องจากมีความเที่ยงตรงและมีความเป็นเชิงเส้นสูงที่สุดของกราฟระหว่างอุณหภูมิกับความต้านทานโดยทั่วไปแล้วเราสามารถจำแนก RTD ตามลักษณะการผลิตได้ 2 ประเภทคือ แบบ Wire wound ซึ่งเป็น RTD ที่ประกอบด้วยขดลวดแพลตตินั่มที่เล็กมาก ๆ (0.0005-0.0015 นิ้ว) ทำเป็นลักษณะคล้ายสปริงบรรจุอยู่ใน ceramic mandrel หรืออีกแบบหนึ่งจะพันขดแพลตตินัมอยู่ข้างนอก ceramic mandrel และฉาบด้วยวัสดุที่เป็นฉนวนไฟฟ้าหุ้มอีกทีเพื่อป้องกันความเสียหายที่อาจเกิดกับขดลวด และแบบที่สอง Thin Film เป็น RTD ที่ใช้แพลตตินัมแปรรูปเป็นแผ่นฟิล์มบาง ๆ เชื่อมต่อกับสาย Lead วางไว้บน Ceramic Substrate เคลือบด้วยแก้วหรืออีพ็อกซี่บริเวณจุดต่อ และเครือบแก้วบริเวณแผ่นฟิล์ม แต่เนื่องจากราคาค่อนข้างแพงเมื่อเปรียบเทียบกับวัสดุชนิดอื่น จึงนิยมใช้ Pt100 ในงานวัดที่ต้องการความละเอียด ส่วน Pt1000 เหมาะสมกับงานที่ต้องการความละเอียดมากขึ้นกว่า Pt100



รูปที่ 2.4 PRT Structure

RTD จำแนกได้หลายแบบด้วยกัน โดยชื่อเรียกจะขึ้นอยู่กับค่าความต้านทาน เช่น Pt1000 นั้นคือ RTD ที่มีค่าความต้านทานที่ 1000 โอมห์ และ Pt100 นั้นคือ RTD ที่มีค่าความต้านทานที่ 100 โอมห์ตามลำดับ โดยทั่วไปในอุตสาหกรรม RTD แบบที่นิยมใช้กันมากที่สุดคือ Pt100 โอห์ม ซึ่งหมายถึงที่อุณหภูมิ 0 °C จะมีค่าความต้านทาน100 โอห์ม และที่ 100 °C จะมีความต้านทานเป็นค่าใกล้ศูนย์ ซึ่งมีอัตราการเปลี่ยนแปลงค่าความต้านทานเป็นเชิงเส้นโดยเฉลี่ย 0.385 โอห์มต่อ 1 °C มีช่วงอุณหภูมิการวัดประมาณ -250 ถึง 800 °C



รูปที่ 2.5 กราฟความต้านทานเทียบกับอุณหภูมิ

จากรูปที่ 2.5 จะพบว่ากราฟของ RTD Pt100 มีอัตราการเพิ่มขึ้นของความต้านทานเป็นเชิงเส้นสูง ทำให้การวัดค่ามีความแม่นยำ ดังนั้นเมื่อเราทราบค่าหนึ่ง ๆ แล้วจะสามารถนำค่านั้น ๆ มาแทนลงบนสมการเส้นตรงจึงจะสามารถทราบค่าที่เหลือจากสมการได้ โดยหลักการของความต้านทานของลวดที่เปลี่ยนไปตามอุณหภูมิเป็นไปตามสมการดังนี้

$$R_t = R_0 (1 + \alpha t) \qquad (2.1)$$

เมื่อ

R_t = ค่าความต้านทานของลวดโลหะที่อุณหภูมิ t °C

R_0 = ค่าความต้านของลวดโลหะที่อุณหภูมิ 0 °C

α = 0.00385 Ω/°C (สัมประสิทธิ์การเปลี่ยนแปลงค่าความต้านทานไฟฟ้าต่ออุณหภูมิ 1 °C)

หรือสามารถนำความต้านทานที่ออกจาก Pt1000 เมื่อนำไปวัดอุณหภูมิมาเทียบค่าอุณหภูมิจากตาราง Datasheet ซึ่งจะเป็นการเทียบความต้านทานกับอุณหภูมิโดยตรงดังแสดงในรูปที่ 2.6

°C	0	-1	-2	-3	-4	-5	-6	-7	-8	-9
-100	602,56									
-90	643,00	638,96	634,92	630,88	626,84	622,80	618,76	614,71	610,66	606,61
-80	683,25	679,24	675,22	671,20	667,17	663,15	659,12	655,09	651,06	647,03
-70	723,35	719,34	715,34	711,34	707,33	703,32	699,31	695,30	691,29	687,27
-60	763,28	759,29	755,30	751,32	747,32	743,33	739,34	735,34	731,34	727,35
-50	803,06	799,09	795,12	791,14	787,17	783,19	779,21	775,23	771,25	767,26
-40	842,71	838,75	834,79	830,83	826,87	822,90	818,94	814,97	811,00	807,03
-30	882,22	878,27	874,33	870,38	866,43	862,48	858,53	854,57	850,62	846,66
-20	921,60	917,67	913,73	909,80	905,86	901,92	897,99	894,04	890,10	886,16
-10	960,86	956,94	953,02	949,09	945,17	941,24	937,32	933,39	929,46	925,53
0	1.000,00	996,09	992,18	988,27	984,36	980,44	976,53	972,61	968,70	964,78
°C	0	-1	-2	-3	-4	-5	-6	-7	-8	-9

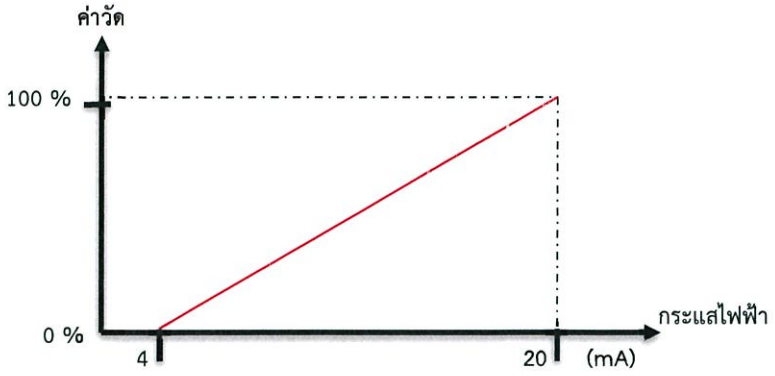
°C	0	1	2	3	4	5	6	7	8	9
0	1.000,00	1.003,91	1.007,81	1.011,72	1.015,62	1.019,53	1.023,43	1.027,33	1.031,23	1.035,13
10	1.039,03	1.042,92	1.046,82	1.050,71	1.054,60	1.058,49	1.062,38	1.066,27	1.070,16	1.074,05
20	1.077,94	1.081,82	1.085,70	1.089,59	1.093,47	1.097,35	1.101,23	1.105,10	1.108,98	1.112,86
30	1.116,73	1.120,60	1.124,47	1.128,35	1.132,21	1.136,08	1.139,95	1.143,82	1.147,68	1.151,55
40	1.155,41	1.159,27	1.163,13	1.166,99	1.170,85	1.174,70	1.178,56	1.182,41	1.186,27	1.190,12
50	1.193,97	1.197,82	1.201,67	1.205,52	1.209,36	1.213,21	1.217,05	1.220,90	1.224,74	1.228,58
60	1.232,42	1.236,26	1.240,09	1.243,93	1.247,77	1.251,60	1.255,43	1.259,26	1.263,09	1.266,92
70	1.270,75	1.274,58	1.278,40	1.282,23	1.286,05	1.289,87	1.293,70	1.297,52	1.301,33	1.305,15
80	1.308,97	1.312,78	1.316,60	1.320,41	1.324,22	1.328,03	1.331,84	1.335,65	1.339,46	1.343,26
90	1.347,07	1.350,87	1.354,68	1.358,48	1.362,28	1.366,08	1.369,87	1.373,67	1.377,47	1.381,26
°C	0	1	2	3	4	5	6	7	8	9
100	1.385,05	1.388,85	1.392,64	1.396,43	1.400,22	1.404,00	1.407,79	1.411,58	1.415,36	1.419,14
110	1.422,93	1.426,71	1.430,49	1.434,26	1.438,04	1.441,82	1.445,59	1.449,37	1.453,14	1.456,91
120	1.460,68	1.464,45	1.468,22	1.471,98	1.475,75	1.479,51	1.483,28	1.487,04	1.490,80	1.494,56
130	1.498,32	1.502,08	1.505,83	1.509,59	1.513,34	1.517,10	1.520,85	1.524,60	1.528,35	1.532,10
140	1.535,84	1.539,59	1.543,33	1.547,08	1.550,82	1.554,56	1.558,30	1.562,04	1.565,78	1.569,52
150	1.573,25	1.576,99	1.580,72	1.584,45	1.588,18	1.591,91	1.595,64	1.599,37	1.603,09	1.606,82
160	1.610,54	1.614,27	1.617,99	1.621,71	1.625,43	1.629,15	1.632,86	1.636,58	1.640,30	1.644,01
170	1.647,72	1.651,43	1.655,14	1.658,85	1.662,56	1.666,27	1.669,97	1.673,68	1.677,38	1.681,08
180	1.684,78	1.688,48	1.692,18	1.695,88	1.699,58	1.703,27	1.706,96	1.710,66	1.714,35	1.718,04
190	1.721,73	1.725,42	1.729,10	1.732,79	1.736,48	1.740,16	1.743,84	1.747,52	1.751,20	1.754,88
°C	0	1	2	3	4	5	6	7	8	9
200	1.758,56	1.762,24	1.765,91	1.769,59	1.773,26	1.776,93	1.780,60	1.784,27	1.787,94	1.791,61
210	1.795,28	1.798,94	1.802,60	1.806,27	1.809,93	1.813,59	1.817,25	1.820,91	1.824,56	1.828,22
220	1.831,88	1.835,53	1.839,18	1.842,83	1.846,48	1.850,13	1.853,78	1.857,43	1.861,07	1.864,72
230	1.868,36	1.872,00	1.875,64	1.879,28	1.882,92	1.886,56	1.890,19	1.893,83	1.897,46	1.901,10
240	1.904,73	1.908,36	1.911,99	1.915,62	1.919,24	1.922,87	1.926,49	1.930,12	1.933,74	1.937,36
250	1.940,98	1.944,60	1.948,22	1.951,83	1.955,45	1.959,06	1.962,68	1.966,29	1.969,90	1.973,51
260	1.977,12	1.980,73	1.984,33	1.987,94	1.991,54	1.995,14	1.998,75	2.002,35	2.005,95	2.009,54
270	2.013,14	2.016,74	2.020,33	2.023,93	2.027,52	2.031,11	2.034,70	2.038,29	2.041,88	2.045,46
280	2.049,05	2.052,63	2.056,22	2.059,80	2.063,38	2.066,96	2.070,54	2.074,11	2.077,69	2.081,27
290	2.084,84	2.088,41	2.091,98	2.095,55	2.099,12	2.102,69	2.106,26	2.109,82	2.113,39	2.116,95

รูปที่ 2.6 Pt1000 resistance table

2.1.3 Smart Device (Temperature Transmitter)

รูปที่ 2.7 Honeywell XYR 6000

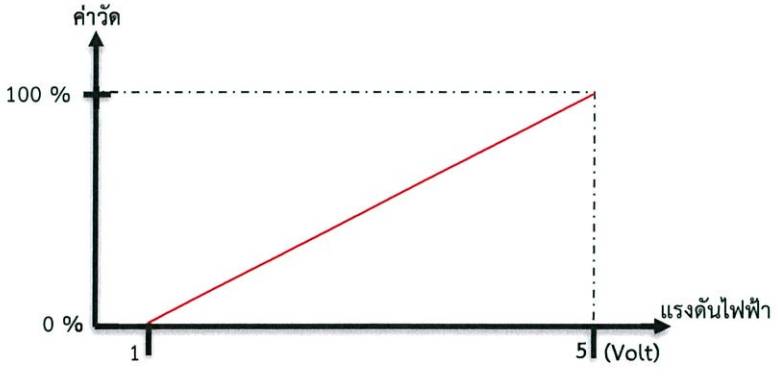
Temperature Transmitter คืออุปกรณ์การวัดอุณหภูมิที่มีวงจรอิเล็คทรอนิกส์ภายใน สามารถทำงานร่วมกับ Sensor และ Actuator (เช่น Thermocouple, RTD, และ Thermistor) เพื่อทำการวัดค่าอุณหภูมิหรือควบคุมโดยสามารถใช้ร่วมกับ PLC, DCS, PC, loop controller หรือ data logger เพื่อเก็บบันทึกข้อมูลไว้วินิจฉัยปัญหาภายหลังได้ ภายในอุปกรณ์ชนิดนี้จะมีการแยกส่วนการทำงานกันเป็นสามส่วน คือวงจรขยายสัญญาญ (amplifier) วงจรกรองสัญญาณรบกวน (noise filter) และวงจรแปลงสัญญาณมาตรฐานเป็นสัญญาณที่นิยมใช้ 4-20mA หรือ 0-10V DC ยกตัวอย่างเช่นจากกระแส 4mA จะแสดงเอาท์พุทออกเป็นอุณหภูมิ 0°C และที่กระแส 20mA จะหมายถึงอุณหภูมิ 100°C เนื่องจากระบบควบคุมในอุตสาหกรรมประกอบด้วย อุปกรณ์ควบคุมหลายชนิดต่อพ่วงกันเป็นระบบและอุปกรณ์เหล่านี้จำเป็นต้องมีการส่งและรับสัญญาญวัดแบบ Analog ระหว่างกันและกัน ดังนั้นจึงจำเป็นต้องมีการกำหนดมาตรฐานสัญญาณวัดแบบ Analog ให้เป็นสากลเพื่อที่บริษัทผู้ผลิตอุปกรณ์ควบคุมจะได้ยึดถือเป็นมาตรฐานในการออกแบบอุปกรณ์ของตนเอง เพื่อให้สามารถต่อพ่วงและใช้งานร่วมกับอุปกรณ์อื่น ๆ ได้



รูปที่ 2.8 ความสัมพันธ์ระหว่างค่าวัดและสัญญาณกระแสมาตรฐาน

ในรูปที่ 2.8 เป็นการส่งสัญญาณในรูปของกระแสตรง (DC Current) มาตรฐานที่นิยมใช้ในอุตสาหกรรมคือ 4-20mA หมายความว่าเมื่อค่าวัดเป็น 0% ก็จะเท่ากับกระแส 4 mA และหากวัดค่าได้เป็น 100% เท่ากับ 20 mA โดยค่าวัดได้จะอยู่ในช่วง 0-100% จะสัมพันธ์เชิงเส้นกับกระแส 4-20mA เพื่อความแม่นยำ

ข้อดีของการส่งสัญญาณเป็นกระแสคือ สามารถส่งสัญญาณไปได้ระยะไกล ๆ ได้ ความต้านทานของสายส่งสัญญาณจะไม่ทำให้ค่าวัดผิดพลาดและการถูกสัญญาณรบกวนจะน้อยกว่าการส่งสัญญาณจะไม่ทำให้ค่าวัดผิดพลาดและการถูกสัญญาณรบกวนจะน้อยกว่าการส่งเป็นแรงดันไฟฟ้า นอกจากมาตรฐาน 4-20mA แล้วยังมีมาตรฐานแบบอื่น ๆ อีก เช่น 0-20mA,10-50 mA,0-1 mA แต่ไม่ค่อยได้รับความนิยม



รูปที่ 2.9 ความสัมพันธ์ระหว่างค่าวัดและแรงดันไฟฟ้ามาตรฐาน

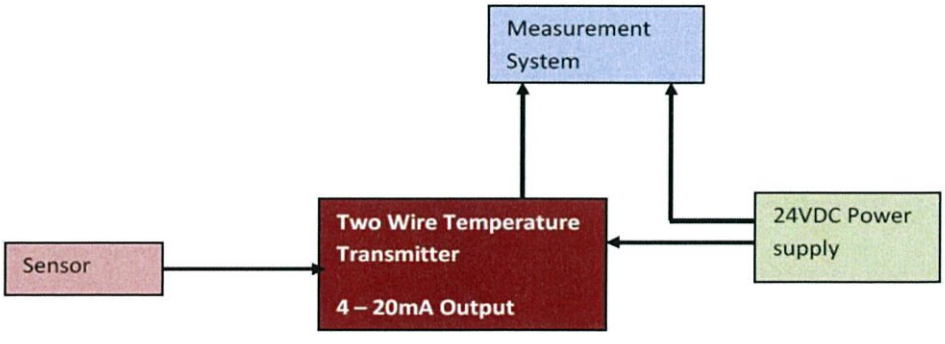
ในรูปที่ 2.9 แสดงความสัมพันธ์ของค่าแรงดันกับค่าวัด โดยรูปแบบสัญญาณจะอยู่ในรูปของแรงดันไฟฟ้า มาตรฐานที่นิยมใช้คือ 1-5 Vdc หมายความว่าเมื่อค่าวัดเป็น 0% ก็จะเท่ากับแรงดันที่ 1

Vdc และค่าวัดเป็น 100% เท่ากับแรงดัน 5 VDC การใช้สัญญาณมาตรฐานแบบแรงดันนี้ไม่เหมาะกับการที่ต้องส่งสัญญาณระยะไกล เนื่องจากความต้านทานของสายสัญญาณจะทำให้ค่าวัดผิดไปและถูกสัญญาณรบกวนได้ง่าย จึงเหมาะกับการส่งสัญญาณระยะใกล้ และมีการต่อเข้าอุปกรณ์รับสัญญาณหลากหลาย เนื่องจากสะดวกในการติดตั้งจึงนิยมใช้

รูปที่ 2.10 Transmitters

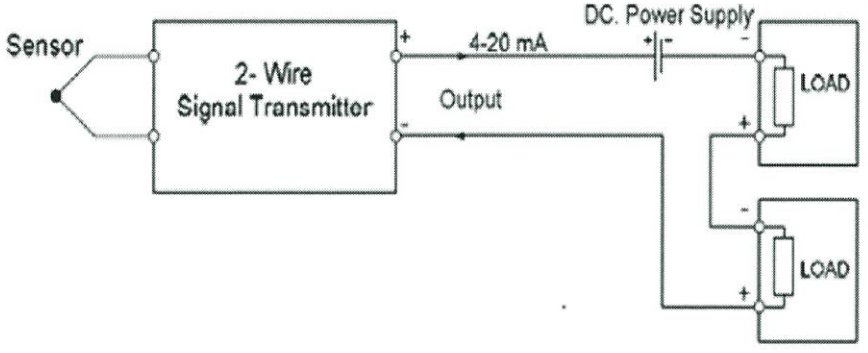
2.1.4 อุปกรณ์แปลงสัญญาณมาตรฐาน (Transmitters)

ในรูปที่ 2.10 คืออุปกรณ์ที่ทำหน้าที่แปลงสัญญาณวัดเซ็นเซอร์แบบต่าง ๆ มาเป็นสัญญาณมาตรฐานของ Transmitters ซึ่งมีหลายชนิดและเรียกตามชื่อของการวัดสิ่งเหล่านั้นตามเซ็นเซอร์ที่ Transmitters ใช้แปลงสัญญาณอุณหภูมิจาก Thermocouple มาเป็นสัญญาณมาตรฐาน RTD Transmitter จากนั้นใช้แปลงสัญญาณจากอุณหภูมิจาก RTD Sensor มาเป็นสัญญาณมาตรฐาน PH Transmitter ใช้แปลงสัญญาณค่า PH จาก PH Sensor เพื่อส่งค่ามาเป็นสัญญาณมาตรฐาน รูปที่ 2.11 คือการแสดงถึงการทำงาน

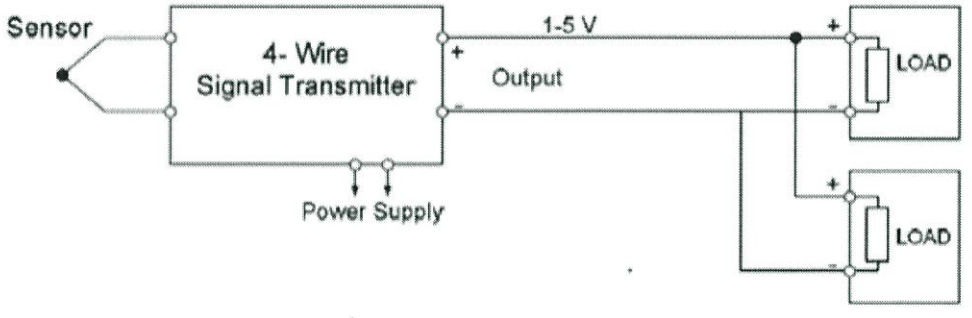


รูปที่ 2.11 การแปลงสัญญาณของ Transmitters

Temperature Transmitter โดยทั่วไปมีอยู่ 2 ชนิดเรียกตามจำนวนสายที่ต่อกับ Signal Transmitter ตัวอย่างการส่งในรูปที่ 2.12 คือการต่อแบบ 2-Wire Signal Transmitter มีการใช้สายเพียง 2 เส้น ซึ่งสายที่ใช้เป็นสัญญาณ Output ของ Transmitter และเป็นสายของ Power Supply สำหรับจ่ายเลี้ยงวงจรอิเล็กทรอนิกส์ภายในตัว Transmitter Output Signal ของ Transmitter แบบนี้เป็นสัญญาณ 4-20mA เท่านั้น ข้อดีของ Transmitter แบบนี้คือประหยัดสายในการติดตั้ง



รูปที่ 2.12 2-Wire Signal Transmitter

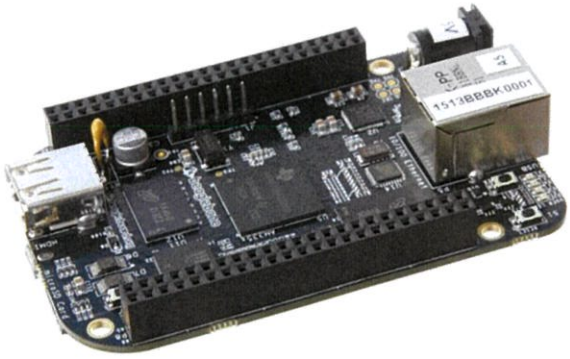


รูปที่ 2.13 4-Wire Signal Transmitter

การต่ออีกรูปแบบหนึ่งคือแบบ 4-Wire Signal Transmitter จะใช้สายสัญญาณ 2 เส้น และสาย Power Supply อีก 2 เส้นแยกกันสัญญาณ Output ของ 4-Wire Signal Transmitter มีทั้งที่เป็นสัญญาณกระแสไฟฟ้ามาตรฐาน และสัญญาณแรงดันไฟฟ้ามาตรฐาน สัญญาณ Output ของ 4-Wire Signal Transmitter มีทั้งเป็นสัญญาณกระแสไฟฟ้ามาตรฐานและสัญญาณแรงดันไฟฟ้ามาตรฐาน สัญญาณจะแตกต่างกัน

2.2 ระบบสมองกลฝังตัว (Embedded System)

ระบบสมองกลแบบฝังตัวหมายถึง ระบบที่ใช้ ตัวประมวลผล (CPU) มาควบคุมการทำงานของระบบใดระบบหนึ่งที่สนใจ ซึ่งรวมไปถึงการใช้ Microcontroller และ Microprocessor ในก่อนหน้านี้ด้วย ระบบสมองกลแบบฝังตัวเป็นระบบที่ใช้ตัวประมวลผล เพิ่มประสิทธิภาพในการทำงานของอุปกรณ์การวัดและควบคุม และ มีซอร์ฟแวร์ของตนเองเพื่อเขียนโปรแกรมควบคุมการทำงาน ทำให้การพัฒนาวงจรอิเล็กทรอนิกส์ ระบบการวัดและควบคุม หรือการสังเกตุการณ์ (Monitoring) ทำได้ง่ายขึ้น ราคาถูกลง และมีความยืดหยุ่นในการทำงานมากขึ้น



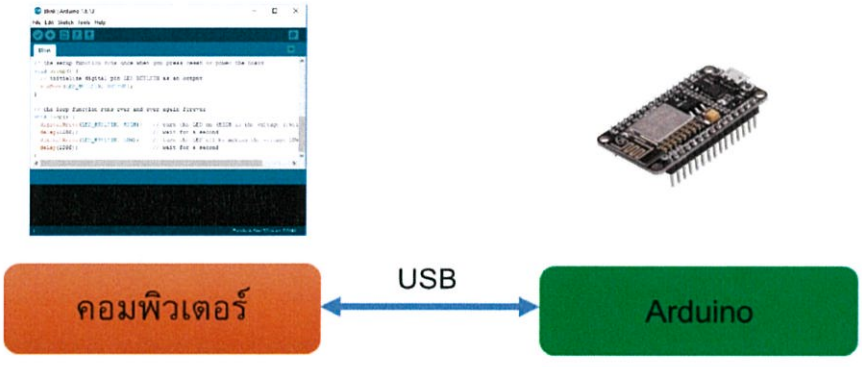
รูปที่ 2.14 Embedded system board

ปัจจุบันทั้งฮาร์ดแวร์และซอฟแวร์กำลังพัฒนาอย่างต่อเนื่อง ซึ่งระบบสมองกลฝังตัวมีการใช้งานอย่างแพร่หลายในปัจจุบัน เช่น ในเครื่องปรับอากาศ ระบบสมองกลฝังตัวที่คอยสั่งงานให้อุณหภูมิห้องเท่ากับอุณหภูมิที่ตั้งไว้ รวมถึงตัดการทำงานของคอมเพรสเซอร์แอร์ หรือในหม้อหุงข้าวและเตาไมโครเวฟที่มีโหมดการทำอาหารแตกต่างกัน ขึ้นอยู่กับคำสั่งที่เขียนไว้ในระบบสมองกลฝังตัวและยังมีแนวโน้มที่ระบบสมองกลจะถูกนำไปใช้ในอุตสาหกรรมโรงงาน หุ่นยนต์ รวมไปถึงยุทโธปกรณ์ เนื่องจากความสามารถที่หลากหลาย ความยืดหยุ่นในการใช้งาน รวมถึงราคาที่ถูก ระบบสมองกลฝังตัวใช้บอร์ดที่ทำหน้าที่เป็นตัวประมวลผล และอุปกรณ์อิเลคทรอนิคส์ต่าง ๆ เป็นอุปกรณ์เสริมเมื่อเพิ่มฟังก์ชั่นที่ปัจจุบันมีบอร์ดอยู่หลากหลาย ขึ้นอยู่กับความประสงค์ในการใช้งาน

2.2.1 Arduino

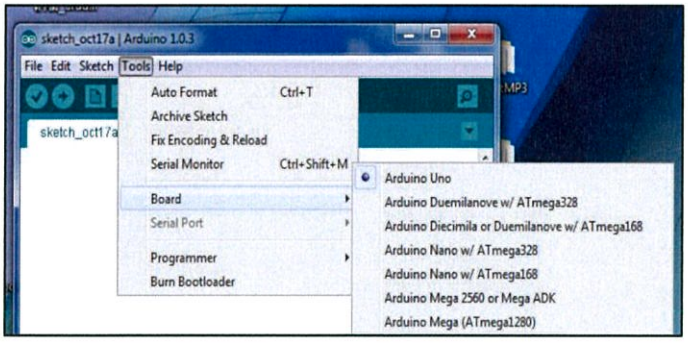
Arduino เป็นบอร์ดขนาดเล็กไมโครคอนโทรเลอร์ตระกูล AVR ที่สามารถเรียนรู้ได้ง่ายมีโครงสร้างระบบ Software และ Hardware เป็นแบบเปิดเผยข้อมูล (Opensource) ซึ่งเปิดกว้างให้หลาย ๆ คนมีส่วนร่วมในการพัฒนา บอร์ด Arduino เหมาะสำหรับงานที่มีการคำนวณน้อยและระบบไม่ซับซ้อนมากนัก Arduino จะมีบอร์ดเสริมเพื่อเพิ่มฟังก์ชั่นการทำงานของระบบได้ เรียกว่า Arduino shield เช่น Arduino Ethernet Shield จะเป็นการเสริมฟังก์ชั่น และพอร์ตเพื่อการสื่อสารแบบ Local Area Network(LAN) ดังรูปที่ 2.15 ที่แสดงด้านล่าง ซึ่งทางขวาคือ Ethernet Shield

รูปที่ 2.15 Arduino UNO Board(ทางซ้าย), Arduino Ethernet Shield(ทางขวา)

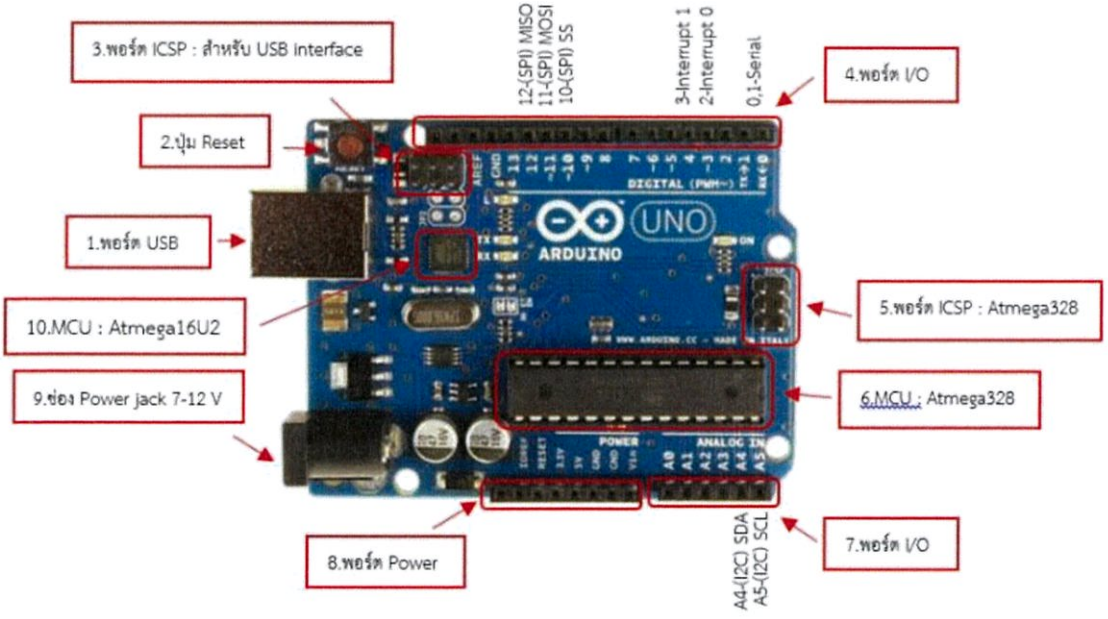


รูปที่ 2.16 การเชื่อมต่อคอมพิวเตอร์และ Arduino

การเขียนโปรแกรมจะเขียนผ่านซอฟต์แวร์ Arduino IDE ซึ่งเป็นฟรีแวร์ที่สามารถดาวน์โหลดได้จาก Website หลักของ Arduino เมื่อเขียนโปรแกรมที่จะใช้เสร็จแล้วให้เลือกบอร์ดที่จะอัปโหลดให้ตรงกับบอร์ดที่ใช้งานอยู่ เช่นหากเป็นบอร์ด UNO ให้เลือกดังรูปที่ 2.17 แล้วจึงอัปโหลดตามลำดับ กรณีที่เลือกบอร์ดหรือ baud rate ไม่ตรงกับตัวบอร์ด จะไม่สามารถอัปโหลดได้หรือเกิดข้อผิดพลาดขึ้น



รูปที่ 2.17 Arduino IDE



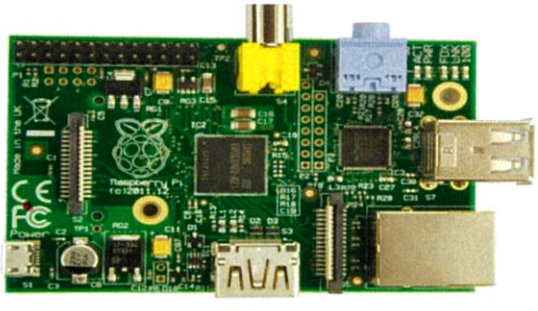
รูปที่ 2.18 UNO Layout Board

1. USBPort: ใช้สำหรับต่อกับ Computer เพื่ออัปโหลดโปรแกรมเข้า MCU และจ่ายไฟ
2. Reset Button: เป็นปุ่ม Reset ใช้กดเมื่อต้องการให้ MCU เริ่มการทำงานใหม่
3. ICSP Port เป็นพอร์ตที่ใช้โปรแกรม Visual Com port บน Atmega16U2
4. I/O Port Digital I/O ตั้งแต่ขา D0 ถึง D13 นอกจากนี้ บาง Pin จะทำหน้าที่อื่น ๆ

5. ICSP Port: Atmega328 เป็นพอร์ตที่ใช้โปรแกรม Bootloader
6. MCU: Atmega328 เป็น MCU ที่ใช้บนบอร์ด Arduino
7. I/O Port: นอกจากจะเป็น Digital I/O แล้วยังเปลี่ยนเป็นช่องรับสัญญาณอนาล็อก
8. Power Port: ไฟเลี้ยงของบอร์ด ประกอบด้วยขาไฟเลี้ยง +3.3 V, +5V, GND, Vin
9. Power Jack: รับไฟจาก Adapter โดยที่แรงดันอยู่ระหว่าง 7-12 V
10. MCU ของ Atmega16U2 เป็น MCU ที่ทำหน้าที่เป็น USB to Serial

2.2.2 Raspberry Pi

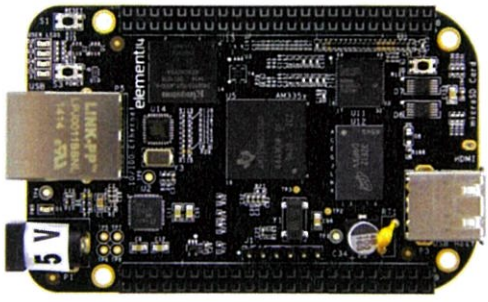
ใช้กับงานที่มีความซับซ้อนมากยิ่งขึ้นความสามารถเทียบเท่าคอมพิวเตอร์ขนาดเล็กที่สามารถต่อขึ้นจอได้ด้วยสาย HDMI หรือสายวิดีโอสามสีปกติแต่จะได้ความละเอียดที่ต่ำกว่า ภาษาที่ใช้เขียนโปรแกรมคือภาษาไพทอน (Python) ซึ่งจะมารองรับทันทีที่เปิดเครื่องถือว่าเป็นฮาร์ดแวร์และซอฟต์แวร์ไปในตัว Raspberry Pi สามารถเป็นสมองกลฝังตัวใน หุ่นยนต์ เครื่องดนตรี เครื่องตรวจจับสภาพอากาศ หรือแม้กระทั่งติดกล้องเข้าไปเพื่อทำกล้องวงจรปิดแล้วส่งค่าขึ้นอินเตอร์เน็ตก็สามารถทำได้



รูปที่ 2.19 Raspberry Pi Board

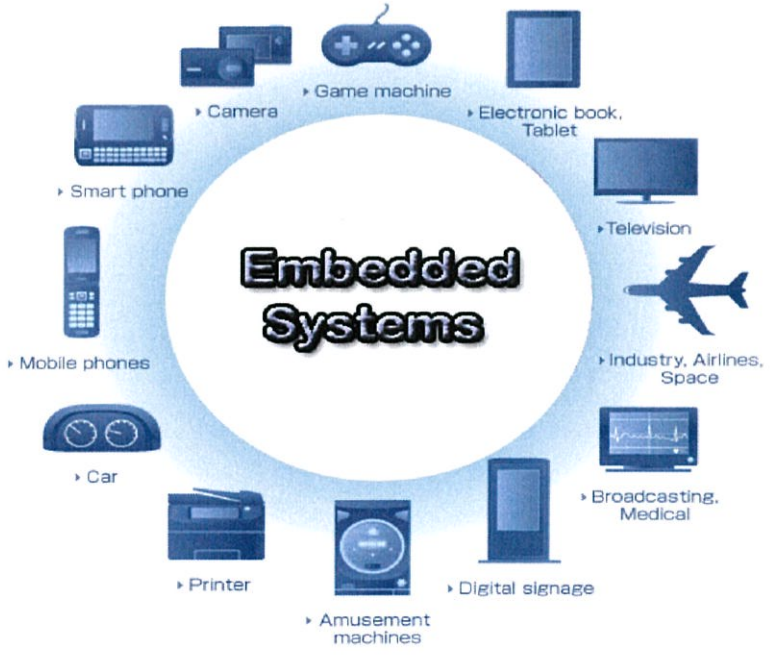
2.2.3 Beagle bone Black

เป็น Single Board สำหรับ Linux ที่มีความสามารถเทียบเท่ากับเครื่อง PC นอกจากนั้นยังมี Port รองรับกับการเชื่อมต่อภายนอกอย่างมากมายและมีระบบซอร์ฟแวร์เป็นแบบ Open source เช่นเดียวกับ Arduino สามารถใช้ภาษาที่หลากหลายในการเขียนโปรแกรม เช่น C, C++, Shell Script หรือ Python ในการพัฒนาได้โดยตัวบอร์ดจะเป็นดังรูปที่ 2.20



รูปที่ 2.20 BeagleBone Board

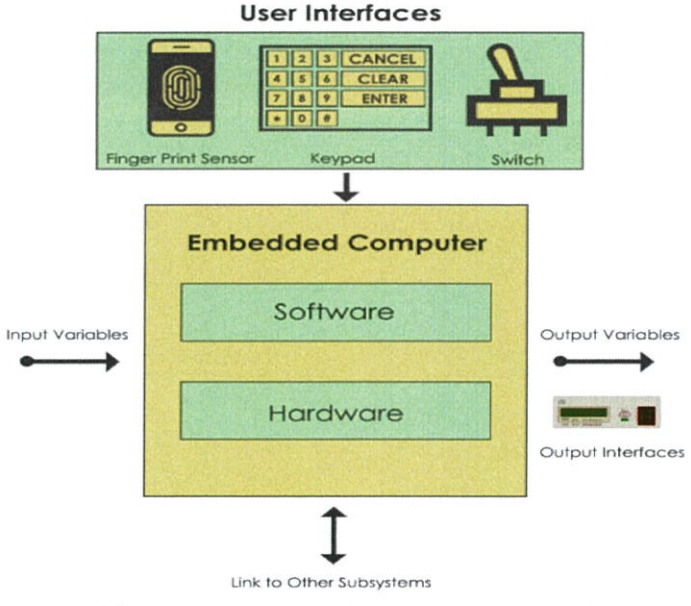
เทคโนโลยีระบบสมองกลฝังตัวเป็นองค์ความรู้ที่เกิดขึ้นมาไม่นาน ทำให้มีผู้เชี่ยวชาญทางด้านระบบสมองกลฝังตัวน้อยและอีกทั้งยังต้องบูรณาการเอาองค์ความรู้ด้านอื่นมาประยุกต์ใช้ด้วย เพราะฉะนั้นบุคคลที่จะนำระบบสมองกลฝังตัวไปประยุกต์ใช้ในเรื่องต่าง ๆ จำเป็นที่จะต้องมีความรู้ทั้งด้าน Software และ Hardware เป็นอย่างดีเพื่อที่จะสามารถออกแบบระบบให้ทำงานได้ เช่น วิศวกรคอมพิวเตอร์หรือโปรแกรมเมอร์ก็จะมีความถนัดทาง Software ส่วนวิศวกรอิเล็กทรอนิกส์หรือช่างอิเล็กทรอนิกส์ก็จะมีความถนัดทางด้าน Hardware เท่านั้น เพื่อให้ใช้งานระบบคอมพิวเตอร์แบบฝังตัวได้ดีจึงจะต้องมีการเรียนรู้ความรู้ทั้งสองด้านและจะต้องมีความรู้เฉพาะทางด้านระบบสมองกลฝังตัวอีกด้วย เมื่อมีความรู้ทั้งสองด้านแล้วก็จะสามารถทำระบบให้มีประสิทธิภาพสูงได้



รูปที่ 2.21 ในปัจจุบันมีการใช้งานระบบสมองกลฝังตัวในหลายอุปกรณ์

รูปที่ 2.21 บอกแสดงถึงระบบสมองกลฝังตัวในชีวิตประจำวัน เนื่องจากปัจจุบันมีการใช้ระบบสมองกลอย่างแพร่หลาย ทั้งในสมาร์ทโฟน รถยนต์ อุตสาหกรรม การบิน และ อวกาศ จึงทำให้ระบบสมองกลฝังตัวเป็นที่รู้จักมากขึ้นเรื่อย ๆ และนำมาใช้งานมากขึ้นเพื่อเพิ่มประสิทธิภาพการทำงานและประหยัดค่าใช้จ่ายมากขึ้น

โดยวิธีการใช้คือนำไปฝังร่วมกันกับวงจรอุปกรณ์เครื่องมือเครื่องใช้ไฟฟ้า และ เครื่องเล่นอิเล็กทรอนิกส์ต่าง ๆ ทั้งนี้เพื่อเพิ่ม ความฉลาดและความสามารถต่าง ๆ ให้กับอุปกรณ์เหล่านั้น ระบบสมองกลฝังตัวแม้จะไม่ใช่เครื่องคอมพิวเตอร์ แต่ก็มีระบบคอมพิวเตอร์ประมวลผลอยู่ภายในอาจจะเป็นเพียงไมโครโพรเซสเซอร์ (Microprocessor) หรือ ชิป (chip) ธรรมดาหรือ โพรเซสเซอร์ (Processor) ที่ประกอบด้วยชิปที่มีวงจรซับซ้อนโดยจะมีหลักการทำงาน คือมีสัญญาณข้อมูลขาเข้า (Input) จากอุปกรณ์เซ็นเซอร์ (Sensor) เข้าสู่ระบบและมีสัญญาณผลลัพธ์ (Output) ของระบบไปควบคุมบังคับ (Actuator) สวิตช์ของเครื่องควบคุมต่าง ๆ เช่นสวิตช์เครื่องจักรหรือวาล์วควบคุมทิศทางการไหลของท่อต่าง ๆ มากมายดังรูปที่ 2.22

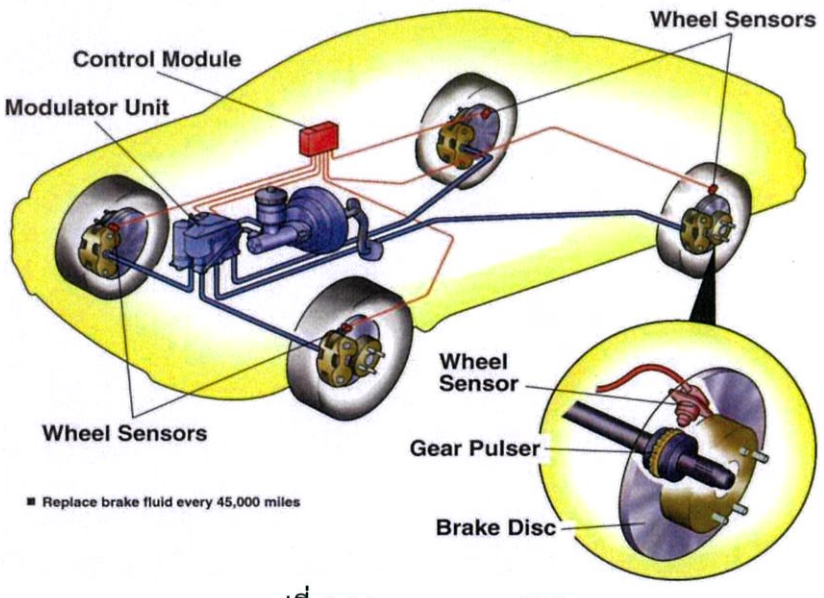


รูปที่ 2.22 ลักษณะการทำงานระบบสมองกลฝังตัว

ในรูปที่ 2.23 คือตัวอย่างการนำสมองกลฝังตัวไปประยุกต์ใช้จักรยานไฟฟ้า (Electric bike) ซึ่งจะทำให้แตกต่างจากจักรยานทั่วไปตรงที่จักรยานไฟฟ้ามีการติดตั้งเพิ่มชุดมอเตอร์ไฟฟ้า ชุดควบคุมมอเตอร์และแบตเตอรี่พลังงานที่เหลือจากการใช้ขณะปั่นจะถูกนำมาเก็บสะสมไว้ในแบตเตอรี่ สมองกลฝังตัวจะทำการควบคุมการจ่ายกระแสไฟฟ้าให้กับมอเตอร์ คุณภาพของจักรยานไฟฟ้าขึ้นอยู่กับความสามารถของสมองกลในควบคุมการทำงานของมอเตอร์ให้เก็บกระแสและจ่ายกระแส

รูปที่ 2.23 จักรยานไฟฟ้า

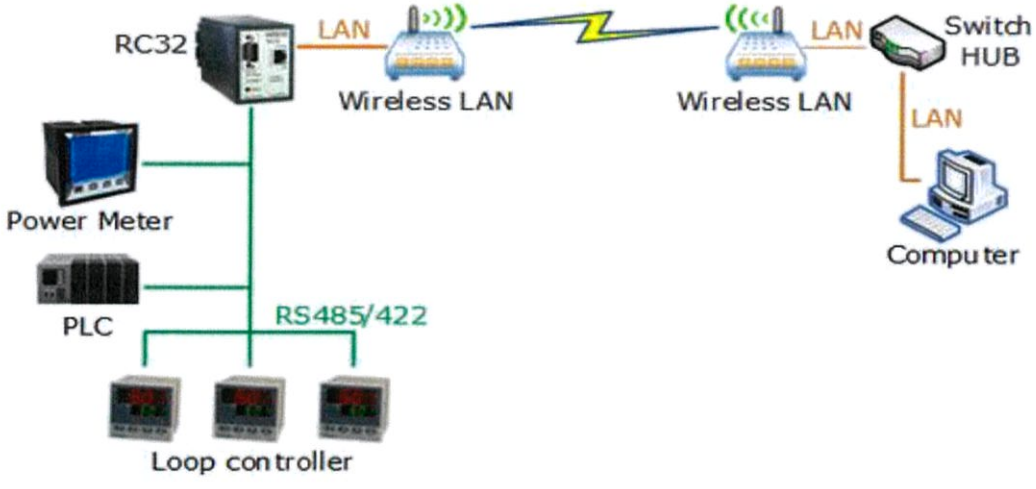
ระบบเบรค Anti-lock Brake System หรือที่รู้จักในชื่อเบรค ABS ในรูปที่ 2.24 เป็นระบบความปลอดภัยซึ่งป้องกันปัญหาการลื่นไถลในขณะเบรกโดยไม่ให้ล้อของยานยนต์เกิดการล็อกหรือหยุดหมุนระหว่างการเบรกทำงานโดยการตรวจจับการล็อกของล้อแล้วไปควบคุมแรงดัน Hydraulic ให้ยกเลิกการเบรกเป็นระยะสั้น ๆ เพื่อความปลอดภัย



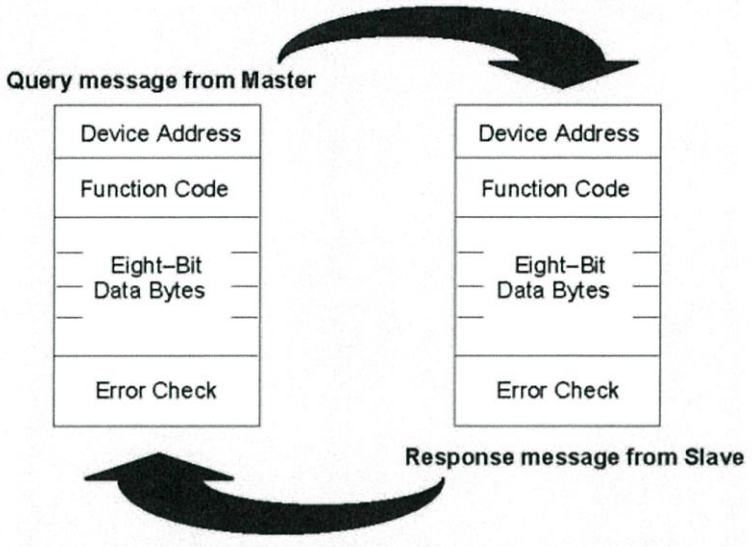
รูปที่ 2.24 ระบบเบรค ABS

2.3 Modbus

โปรโตคอล Modbus เป็นโปรโตคอลเพื่อสื่อสารข้อมูลอินพุต/เอาต์พุตและ Register ภายใน PLC ซึ่งถูกคิดค้นโดยบริษัท Modicon ที่ปัจจุบันคือบริษัท Schneider Electric โปรโตคอล Modbus ได้เป็นที่ยอมรับในการติดต่อสื่อสารที่เป็นแบบ Network Protocol อันเนื่องมาจาก Modbus เป็นระบบเปิด, ไม่มีค่าใช้จ่าย, เชื่อมต่อและพัฒนาง่าย พร้อมทั้งยังสามารถนำโปรโตรคอลนี้ไปประยุกต์ใช้งานในอุปกรณ์อื่น ๆ เช่น Digital Power Meter, RTU (Remote Terminal Unit), Remote I/O, PLC เป็นต้น นอกจากนี้ Modbus ยังสามารถรองรับและใช้งานร่วมกับ Application จำพวก SCADA และ HMI Software ได้อีกด้วย Modbus เป็นการสื่อสารข้อมูลในลักษณะ Master/Slave ซึ่งเป็นการสื่อสารจากอุปกรณ์แม่ (Master) เครื่องเดียว ส่วนใหญ่มักเป็นซอฟต์แวร์คอมพิวเตอร์หรืออุปกรณ์แสดงผล HMI ไปยังอุปกรณ์ลูก (Slave) ได้หลาย ๆ เครื่อง การรับข้อมูลใน Modbus มีให้ปรับเปลี่ยนสองรูปแบบการทำงานคือการส่งข้อมูลแบบ ASCII เรียกว่า Modbus ASCII และอีกรูปแบบหนึ่งคือการส่งแบบเลขฐานสองคือ Modbus RTU ทั้ง 2 โหมดนี้มีความแตกต่างกันที่การกำหนดรูปแบบของชุดข้อมูลภายในเฟรม การใช้งานจะเลือกโหมดใดก็ได้แต่มีเงื่อนไขว่า อุปกรณ์ทุกตัวที่ต่อร่วมกันอยู่ในบัสหรือเครือข่ายเดียวกัน จะต้องตั้งให้เลือกใช้โหมดเดียวกันทั้งหมด



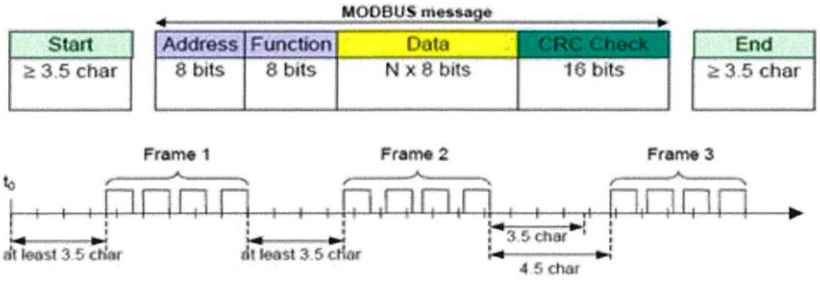
รูปที่ 2.25 Modbus Network



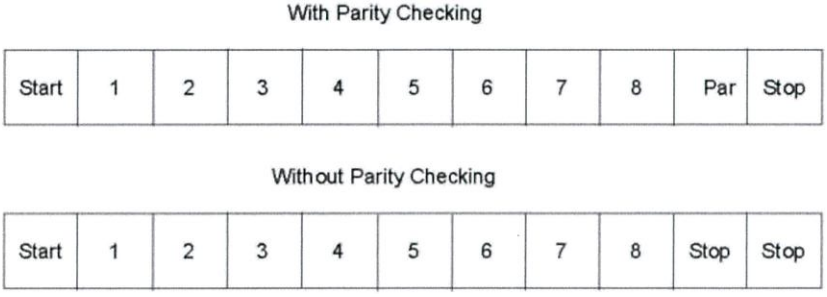
รูปที่ 2.26 รูปแบบการส่งข้อมูล ของ Modbus

2.3.1 Modbus RTU

เฟรมการส่งข้อมูลในการทำงานโหมด RTU ประกอบด้วยข้อมูลแสดงแอดเดรส 1 ไบต์, หมายเลขฟังก์ชัน 1 ไบต์, ข้อมูลที่ทำการรับส่งจำนวนมากสุดไม่เกิน 252 ไบต์ และรหัสตรวจสอบความถูกต้องของข้อมูลแบบ 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 หากไม่ตรงกันแสดงว่าเกิดความผิดพลาดในการรับส่งข้อมูลในการทำงานโหมด 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 ควรเลือกแบบคู่โดยที่สามารถปรับเปลี่ยนเป็นแบบคี่หรือไม่มีการตรวจสอบ Parity (No Parity) ได้ด้วย โดยลักษณะเฟรมการส่งข้อมูลจะแสดงให้เห็นในรูปที่ 2.27



รูปที่ 2.27 ลักษณะเฟรมข้อมูลของ Modbus RTU

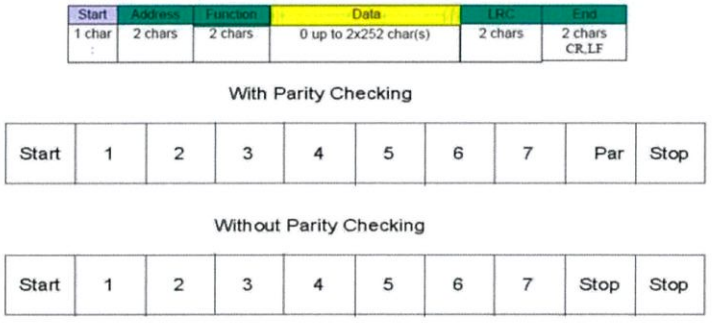


รูปที่ 2.28 ลักษณะข้อมูลแต่ละไบต์ของ Modbus RTU

2.3.2 Modbus ASCII

การรับส่งข้อมูลในโหมด ASCII นั้นมีความแตกต่างจากโหมด RTU ตรงที่ในโหมด RTU ข้อมูลที่จะส่งขนาด 1 ไบต์ นำมารวมกับบิตประกอบต่าง ๆ ก็สามารถส่งออกไปได้เลย แต่สำหรับโหมด ASCII จะมองข้อมูล 1 ไบต์ นั้นออกมาเป็นตัวอักษร 2 ตัว เช่น ค่า 0x5B ซึ่งเป็นเลขฐานสิบหก ก็จะถูกมองเป็นตัวอักษร '5' และตัวอักษร 'B' จากนั้นก็จะทำการค้นหารหัส ASCII ของตัวอักษรทั้ง 2 ตัวนั้น ซึ่งได้แก่ 0x35 สำหรับ '5' และ 0x42 สำหรับ 'B' แล้วทำการส่งรหัส ASCII ทั้ง 2 ค่านี้ออกไป ซึ่งจะได้ผลเท่ากับการส่งค่า 0x5B ซึ่งเป็นข้อมูลขนาด 1 ไบต์ ในโหมด RTU จะเห็นได้ว่าการส่งข้อมูลในโหมด ASCII จะต้องทำงานมากกว่าการส่งข้อมูลในโหมด RTU ซึ่งทำให้อัตราเร็วในการสื่อสารมีค่าต่ำกว่า สาเหตุที่เป็นแบบนี้ก็เพราะว่า โหมด ASCII ได้ถูกออกแบบมาสำหรับอุปกรณ์ที่ไม่มีความสามารถในการกำหนดช่วงระยะห่างของเวลาในการส่งเฟรมข้อมูล อย่างเช่นในโหมด RTU ที่อุปกรณ์สามารถกำหนดได้ว่าจะส่งเฟรมข้อมูลแต่ละเฟรมออกมาด้วยเวลาห่างกันเท่าใด และอุปกรณ์ที่รับข้อมูลก็ต้องสามารถตรวจจับและแยกแยะได้ว่าเฟรมข้อมูลแต่ละเฟรมที่รับเข้ามานั้นมีระยะเวลาห่างกันภายในช่วงเวลาที่กำหนดหรือไม่ เพื่อทำให้

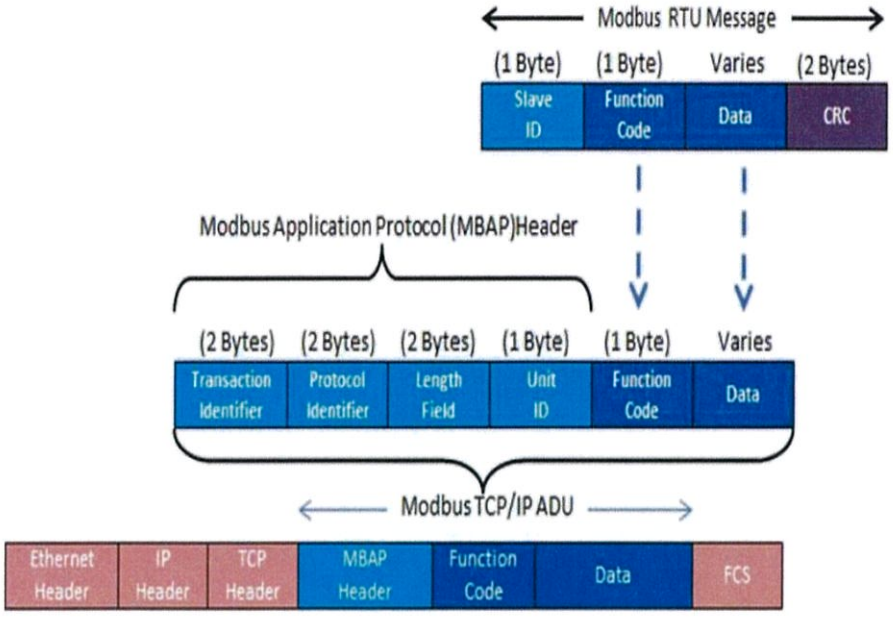
สามารถตรวจสอบหาจุดเริ่มต้นและจุดสิ้นสุดของเฟรมข้อมูลแต่ละเฟรมได้ แต่ในความเป็นจริงยังมีอุปกรณ์อีกหลายชนิดที่ไม่มีความสามารถพิเศษนี้ จึงต้องใช้วิธีอื่นที่จะช่วยให้สามารถรับรู้จุดเริ่มต้นและจุดสิ้นสุดของเฟรมข้อมูลได้ นั่นได้แก่โหมด ASCII ซึ่งในโหมดนี้จะเริ่มต้นเฟรมข้อมูลด้วยการส่งรหัส ASCII ที่กำหนดให้หมายถึงจุดเริ่มต้น คือ 0x3A ซึ่งตรงกับตัวอักษร ':' ตามด้วยแอดเดรสของ Slave, หมายเลขฟังก์ชัน, ข้อมูล, รหัสตรวจสอง RLC และรหัส ASCII 2 ตัว ที่กำหนดให้หมายถึงจุดสิ้นสุด คือ รหัส 0x0D และ 0x0A คือรหัส CR (Carriage Return) และ LF (Line Feed) ตามลำดับ โดยในขณะที่บัสข้อมูลว่างจากการรับส่งข้อมูล อุปกรณ์ทุกตัวจะคอยตรวจสอบข้อมูลในบัสว่ามีการส่งรหัส ASCII ของ ':' ออกมาหรือไม่ ถ้ามีก็จะรับรู้ว่าขณะนี้ได้มีการเริ่มต้นส่งเฟรมข้อมูลออกมาแล้ว ก็จะเข้ากระบวนการรับข้อมูลต่อไป



รูปที่ 2.29 ลักษณะเฟรมข้อมูลของ Modbus ASCII

2.3.3 การสื่อสารของ Modbus TCP/IP

Modbus TCP/IP คือโปรโตคอลที่คล้ายคลึงกับ Modbus RTU แต่เชื่อมต่อด้วย TCP เพื่อทำงานบนระบบ Ethernet ซึ่งโครงสร้าง Message ของ Modbus จะทำหน้าที่เป็น Application Protocol ที่จะถูกส่งผ่านไปพร้อมกับ TCP/IP หรือ Transmission Control Protocol และ Internet Protocol ซึ่งเป็นตัวกลางที่ใช้ในการส่ง Message ของ Modbus TCP/IP ในทางปฏิบัติ Modbus TCP จะฝัง Modbus RTU data frame ร่วมไปกับ TCP frame โดยไม่ต้องใช้ Modbus checksum แต่จะใช้ Checksum ของ TCP แทนดังแสดงในรูปที่ 2.30



รูปที่ 2.30 Modbus Application Protocol (MBAP)

จากรูปภาพที่ 2.30 MBAP จะประกอบด้วยข้อมูล 7 bytes ซึ่งจะวางหน้า Modbus RTU message โดยมีรายละเอียดดังหัวข้อต่อไปนี้

1. Transaction/invocation Identifier (2 Bytes): ใช้จับคู่การแลกเปลี่ยนข้อมูลเมื่อมี Message หลาย ๆ ชุด ถูกส่งออกมาด้วย TCP เดียวกันด้วย Client ตัวหนึ่งโดยไม่ต้องรอลำดับการ Response

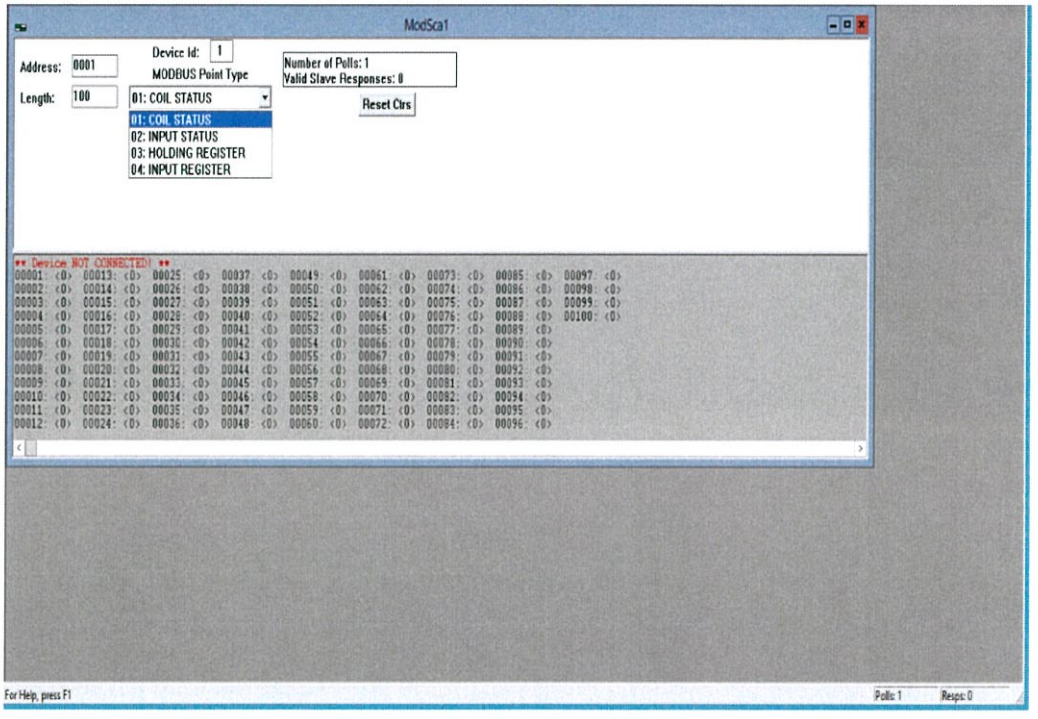
2. Protocol Identifier (2 bytes): ในส่วนจะมีค่าเป็น 0 เสมอ

3. Length (2 bytes): เป็นการระบุจำนวน Byte ที่รวมจำนวน Byte ของ unit identifier, function code, และ data fields

4. Unit Identifier (1 byte): เป็นการระบุ ID ของ server สื่อสารอาจเป็น 00 ถึง FF

2.3.4 ModScanโปรแกรมอ่าน/เขียน Modbus

ในทางอุตสาหกรรมมักจะใช้การสื่อสาร RS485 และใช้ Protocol ของ Modbus ในการสื่อสารส่งข้อมูลและบางโอกาสก็ต้องการจะอ่านค่าจาก Instrument หรือ PLC (Slave Device) ขึ้นมาเพื่อดูค่า Parameter ที่กล่าวถึง จึงมีโปรแกรมอ่านค่า Modbus เพื่อสะดวกต่อการตรวจดูค่าที่ส่งมาจากอุปกรณ์ต่าง ๆ โดยรูปที่ 2.31 คือตัวหน้าต่าง Monitoring ค่า Parameter สามารถเลือก Baud Rate ได้ตั้งแต่ 110-256 Kbps



รูปที่ 2.31 ModScan

2.4 ระบบ Human Machine Interface (HMI)

รูปที่ 2.32 System Platform logo

HMI คือ การใช้งานร่วมกันระหว่าง PLC กับเครื่องคอมพิวเตอร์จึงเรียกว่า HMI ที่ย่อมาจาก Human Machine Interface โดยนำคอมพิวเตอร์มาเป็นอุปกรณ์ที่ใช้ในการติดต่อระหว่างผู้ใช้งานกับเครื่องจักร เพื่อควบคุมและเป็นจอแสดงผล HMI รวมไปถึง SCADA เกิดจากความต้องการของผู้ใช้งานที่ต้องการเข้าไปควบคุมระบบที่ PLC เป็นตัวควบคุมอยู่ โดย HMI นั้นจะเป็นการนำข้อมูลจาก PLC ส่งผ่านโครงข่ายของการสื่อสารแบบต่าง ๆ และทำการถอดรหัสและรวบรวมข้อมูลในรูปแบบต่าง ๆ เข้าด้วยกัน

งานอุตสาหกรรมในปัจจุบันเกือบทุกประเภทจะมีระบบควบคุมอัตโนมัติที่ใช้ PLC เป็นตัวควบคุม และจะต้องใช้งานร่วมกัน กับ HMI โดยใช้ HMI เป็นตัวสื่อสารระหว่างผู้ใช้งานกับระบบ Module PLC หรือจอแสดงผลต่าง ๆ โดยให้ PLC สั่งงาน ไปที่เครื่องจักรอีกที เพื่อนำไปใช้งานกับเครื่องจักรต่าง ๆ ใน Line ผลิต โดยที่ทาง Energy Scope เลือกใช้ HMI ที่เชื่อมต่อกับ PLC ต่าง ๆ ได้ทุกยี่ห้อผ่านทาง Digital Communication Ports เช่น RS485, RS232,MODBUS,PROFIBUS,ETHERNET และยังสามารถเชื่อมต่อกับพอร์ต USB ได้โดยตรง

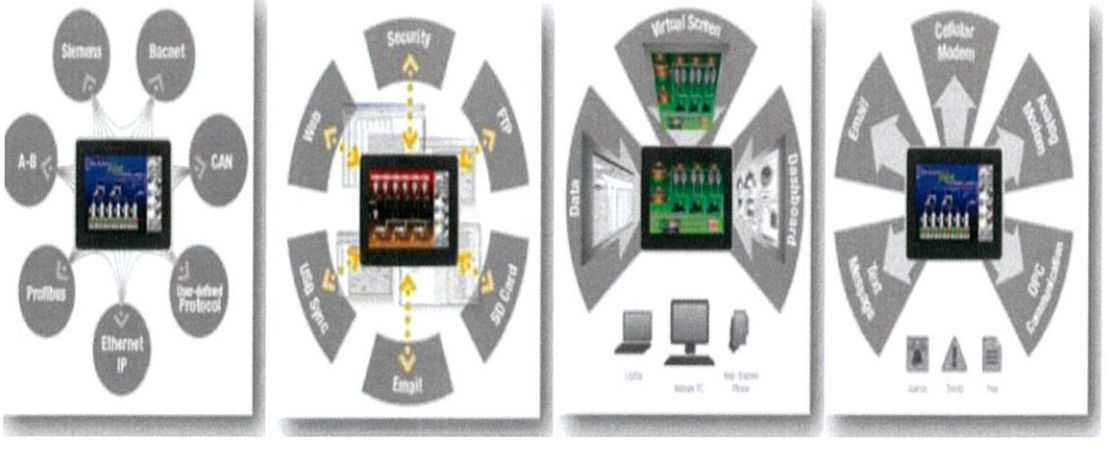
รูปที่ 2.33 ตัวอย่าง intouch HMI

2.4.1 ลักษณะการสื่อสาร

สามารถสื่อสารข้อมูลกับอุปกรณ์อื่น ๆ ในลักษณะแบบดิจิตอล โดยมีรูปแบบของสัญญาณให้เลือกหลายแบบและสามารถสื่อสารข้อมูลกับอุปกรณ์ต่าง ๆ ทุกยี่ห้อได้อย่างมีประสิทธิภาพ สามารถต่อได้ทั้งอุปกรณ์ PLC, Meter, Controller และอีกมากมายตามการนำไปใช้งานประเภทต่าง ๆ โดยอุปกรณ์HMI เพียงตัวเดียวก็สามารถควบคุม หรืออ่านค่าตัวอุปกรณ์ฮาร์ดแวร์อื่น ๆ ที่ต่อเชื่อมอยู่ได้อย่างง่ายดายผ่านการเชื่อมต่อทางเครือข่ายอินเตอร์เน็ต, LAN หรือ Wireless

2.4.2 การเชื่อมต่อและการเก็บค่า

สามารถเพิ่มความสะดวกให้กับผู้ใช้งานในการสังเกตุการณ์ค่าหรือควบคุมกระบวนการผลิตจากระยะไกลโดยการเชื่อมต่อผ่านมือถือหรือแท็บเล็ตใช้เว็บบราวเซอร์มาตรฐานตัวใดก็ได้ในการดูค่าหรือควบคุมป้อนกลับโดยหน้าจอแสดงผลโชว์ หน้าตาใช้งานสะดวก มีการสั่งงานที่เข้าใจง่าย สามารถส่งข้อความ SMS หรือ email แจ้งเตือนให้กับบุคคลที่เกี่ยวข้อง และ ดูค่าที่หน้าจอ, ค่าที่บันทึกไว้ใน Memory Card หรือควบคุมแก้ไขเปลี่ยนค่าได้แม้ไม่ได้อยู่ที่หน้างาน การเก็บค่าจะทำโดยกระบวนการผลิตต่าง ๆ ใน รูปแบบไฟล์ Excel เพื่อวินิจฉัยปัญหา รวมไปถึงการเข้าถึงข้อมูล (Data logger) ผ่านทาง Web Browser ได้อย่างง่ายดาย ทำให้สะดวกในการทราบข้อมูล แม้ไม่ได้อยู่ที่หน้างานไลน์ผลิต



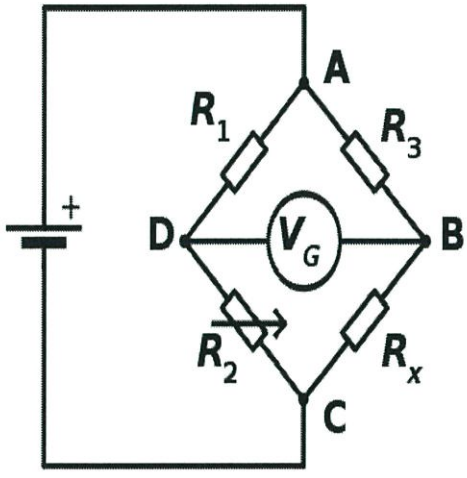
รูปที่ 2.34 ฟังก์ชั่นการทำงานของ HMI

2.5 วงจรวีตสโตนบริดจ์ (Wheatstone bridge)

วงจรบริดจ์เป็นวงจรที่นิยมนำมาใช้ในการวัดค่าความต้านทาน โดยใช้หลักการ บริดจ์สมดุล (Bridge Balance) หรือที่เรียกว่า วีตสโตนบริดจ์ (Wheatstone bridge) ซึ่งมีลักษณะของวงจรพื้นฐานดังแสดงในรูปด้านล่าง วงจรบริดจ์มีหลักการทำงานคือ เมื่อบริดจ์สมดุล กระแสที่ไหลผ่านกัลวานอมิเตอร์ (G) ในวงจรจะมีค่าเท่ากับศูนย์ ซึ่งเงื่อนไขดังกล่าวจะเกิดขึ้นก็ต่อเมื่อความต้านทานเป็นดังสมการที่ 2.2

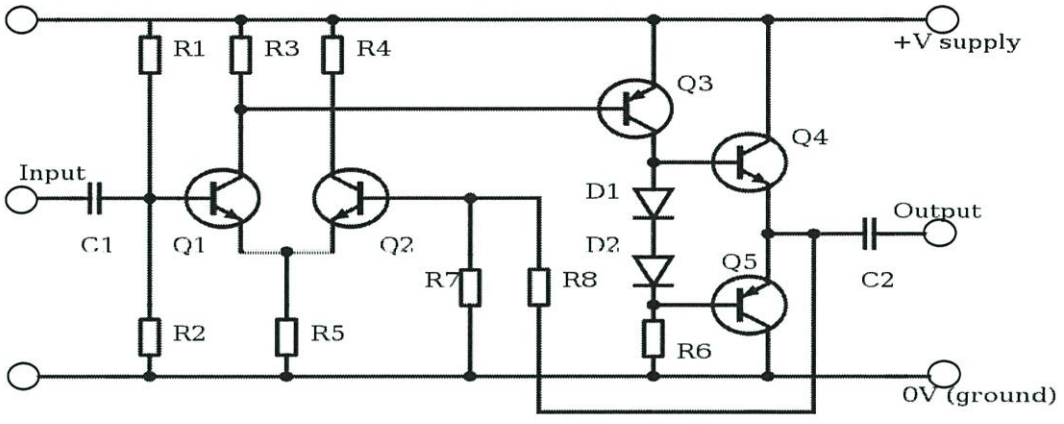
$$R1/R2 - R3/R4 \qquad (2.2)$$

ในทางปฏิบัติ R1 คือความต้านทานที่ต้องการวัด R3 และ R4 จะเท่ากัน ส่วน R2 เป็นตัวต้านทานที่สามารถปรับเปลี่ยนค่าได้เพื่อทำให้กระแสที่กัลวานอมิเตอร์เป็นศูนย์ให้เข้ากับเงื่อนไขดังกล่าวไว้ในสมการที่ 2.2



รูปที่ 2.35 วงจรวีทสโตนบริดจ์

2.6 วงจรขยายสัญญาณ (Amplifier)



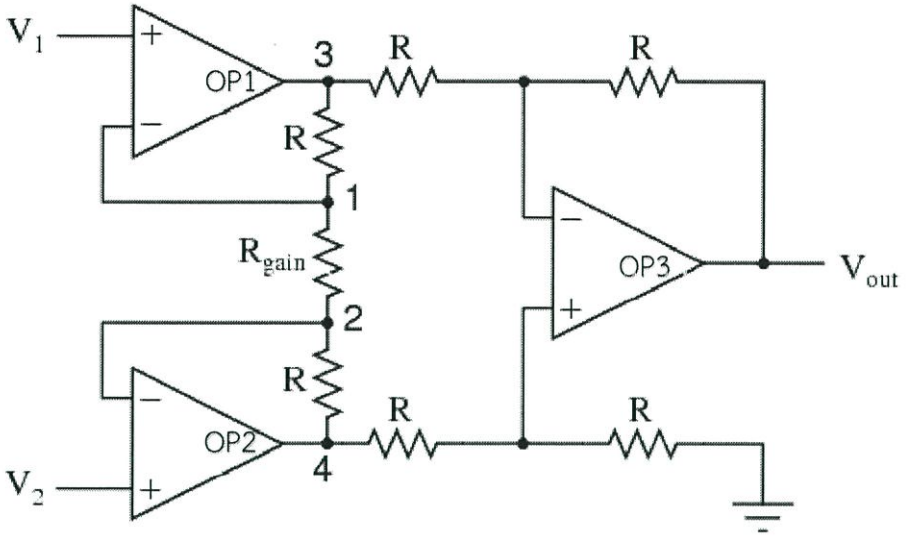
รูปที่ 2.36 วงจร Amplifier

วงจรขยายสัญญาณ (Electronic Amplifier or Amplifier) เป็นอุปกรณ์วงจรอิเล็กทรอนิกส์ที่ช่วยเพิ่มขนาดหรือกำลังของสัญญาณที่ส่ง โดยการใช้พลังงานจากแหล่งจ่ายไฟและการควบคุมสัญญาณเอาต์พุทให้มีรูปร่างเหมือนสัญญาณอินพุท แต่มีขนาดใหญ่กว่า ตัวขยายอิเล็กทรอนิกส์มี 4 ประเภทพื้นฐานได้แก่ วงจรขยายแรงดัน, วงจรขยายกระแส, วงจรขยาย transconductance และวงจรขยาย trans resistance ความแตกต่างอยู่ที่สัญญาณเอาต์พุตจะแทนความหมายของสัญญาณอินพุทแบบเชิงเส้นหรือแบบกราฟเอ็กซ์โปเนนเชียล ตัวขยายสัญญาณยังสามารถถูกแยกประเภทโดยการแทนที่ทางกายภาพในขบวนของสัญญาณด้วย

ออปแอมป์ เป็นอุปกรณ์อิเล็กทรอนิกส์ชนิดหนึ่งที่มีการทำงานเป็นแบบ Voltage-controlled voltage source ซึ่งสามารถประยุกต์ใช้กับสัญญาณได้ดังนี้

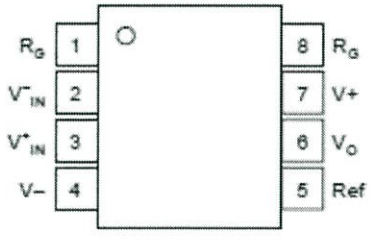
1. รวมสัญญาณ (Sum signal)
2. อนุพันธ์สัญญาณ (Differentiate signal)
3. อินทิเกรตสัญญาณ (Integrate signal)
4. ขยายสัญญาณ(Amplify signal)

2.7 วงจร Instrument Amplifier



รูปที่ 2.37 วงจร Instrument Amplifier

วงจร Instrument Amplifier เป็นวงจรรวม หรือ Integrated Circuit สำเร็จรูปมีหน้าที่ในการขยายสัญญาณ โดยในรูปที่ 2.37 จะใช้ออปแอมป์ 3 ตัวในการสร้างวงจร Instrument Amp



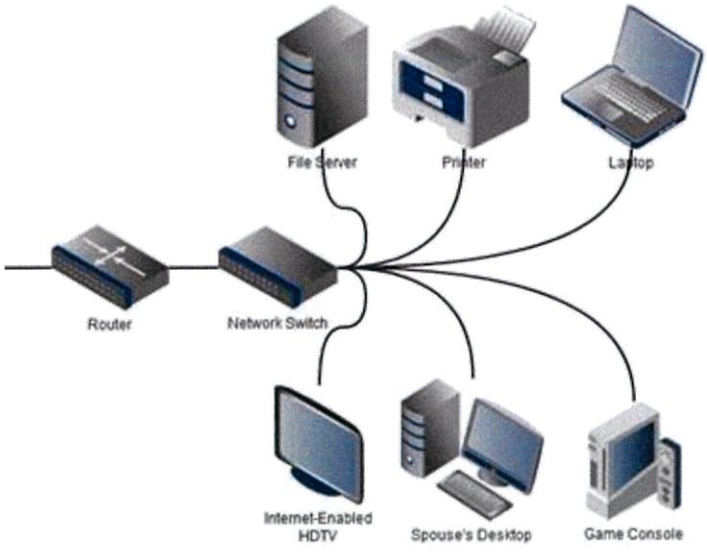
รูปที่ 2.38 ตัวอย่าง IC INA126

2.8 เราเตอร์ (Router)

รูปที่ 2.39 เราเตอร์

เราเตอร์เป็นอุปกรณ์ที่ซับซ้อนกว่าบริดจ์ มีหน้าที่เชื่อมต่อ LAN หลาย ๆ เครือข่ายเข้าด้วยกัน คล้ายกับสวิตช์แต่จะมีส่วนเพิ่มเติมขึ้นมาคือเราเตอร์สามารถเชื่อมต่อ LAN ที่ใช้โปรโตคอลในการรับส่งข้อมูลเหมือนกัน เช่นการใช้ข้อตกลง Modbus RTU ในการสื่อสารแต่ใช้สื่อส่งข้อมูลหรือสายส่งต่างชนิดกันได้ เช่น เชื่อมต่อ Ethernet LAN ที่ใช้รับส่งข้อมูลแบบ UTP เข้ากับ Ethernet อีกเครือข่ายหนึ่งที่ใช้สายข้อมูลแบบ coaxial cable ได้

เราเตอร์ทำงานเสมือนเป็น Node หนึ่งใน LAN ซึ่งจะทำการรับข้อมูลเข้ามาแล้วส่งต่อไปยังปลายทาง โดยอาจส่งในรูปแบบของ packet ที่ต่างออกไปจากเดิม เพื่อไปผ่านสายสัญญาณแบบอื่น ๆ เช่น สายโทรศัพท์ที่ต่อผ่านโมเด็มก็ได้ ดังนั้นเราจึงอาจใช้เราเตอร์เพื่อเชื่อมต่อ LAN หลาย ๆ แบบเข้าด้วยกันได้ด้วยและจากการที่มันทำตัวเสมือนเป็น Node หนึ่งใน LAN ทำให้เราเตอร์สามารถทำงานอื่น ๆ อีกมาก ยกตัวอย่างเช่น รวบรวมข้อมูลเพื่อหาเส้นทางที่ดีที่สุดในการส่งข้อมูลต่อ หรือตรวจสอบว่าข้อมูลที่เข้ามานั้นมาจากไหน ควรจะให้ผ่านหรือไม่ เพื่อช่วยในเรื่องการรักษาความปลอดภัยด้วย สิ่งที่แตกต่างกันระหว่างบริดจ์กับเราเตอร์คือ เราเตอร์มีการทำงานที่สูงกว่าคือ ในระดับชั้นที่ 3 ของ โมเดล OSI นั่นคือ Network Layer โดยจะใช้ logical address หรือ Network Layer address ซึ่งเป็นที่อยู่ซึ่งตั้งโดยซอฟต์แวร์ที่ผู้ใช้แต่ละเครื่องจะตั้งขึ้นให้โปรโตคอลในระดับ Network Layer



รูปที่ 2.40 ตัวอย่างการเชื่อมต่อวง LAN

หน้าที่หลักของเราเตอร์ คือการหาเส้นทางที่ดีที่สุดในการส่งผ่านข้อมูลให้รวดเร็วและเป็นตัวกลางในการส่งต่อข้อมูลไปยังเครือข่ายอื่นๆที่ต้องการจะสื่อสารด้วย โดยในแต่ละเครือข่ายจะมีรูปแบบของ packet ที่แตกต่างกันตามโปรโตคอลที่ทำงานในระดับสูง (ตั้งแต่เลเยอร์ที่ 3 ขึ้นไป) เช่น IP, IPX หรือ AppleTalk

รูปที่ 2.41 ช่องทางส่งข้อมูลที่สะดวก

จากรูปที่ 2.41 เมื่อมีการส่งข้อมูลก็จะบรรจุข้อมูลนั้นเป็น packet ในรูปแบบของเลเยอร์ที่ 2 เมื่อเราเตอร์ได้รับข้อมูลก็จะตรวจดูใน packet นี้เพื่อจะทราบว่าใช้โปรโตคอลแบบใด ซึ่งเราเตอร์จะเข้าใจ Protocol ต่าง ๆ แล้ว จากนั้นก็จะตรวจดูเส้นทางส่งข้อมูลจากตาราง routing table ว่าจะต้องส่งข้อมูลนี้ไปยังเครือข่ายใดต่อจึงจะถึงปลายทางได้ แล้วจึงบรรจุข้อมูลลงเป็น packet ตามรูปแบบของเลเยอร์ที่ 2 อีกครั้งเพื่อส่งต่อไปยังเครือข่ายถัดไป

ข้อดีของการใช้เราเตอร์

1. การไหลเวียนของข้อมูลจะไม่รบกวนการไหลเวียนข้อมูลของอีกเครือข่าย LAN อื่น
2. มีความคล่องตัวในการทำงานสูง
3. กำหนดความสำคัญในการส่งได้ โปรโตคอลความสำคัญสูงสามารถลัดคิวส่งออกไปได้ก่อน
4. การปิดกั้นเครือข่ายซึ่งเป็นการรักษาความปลอดภัยวิธีหนึ่ง
5. การเลือกเส้นทางในการที่จะส่งข้อมูล สามารถใช้เราเตอร์ช่วยในการเลือกเส้นทางที่ดีที่สุด

ข้อเสียของการใช้เราเตอร์

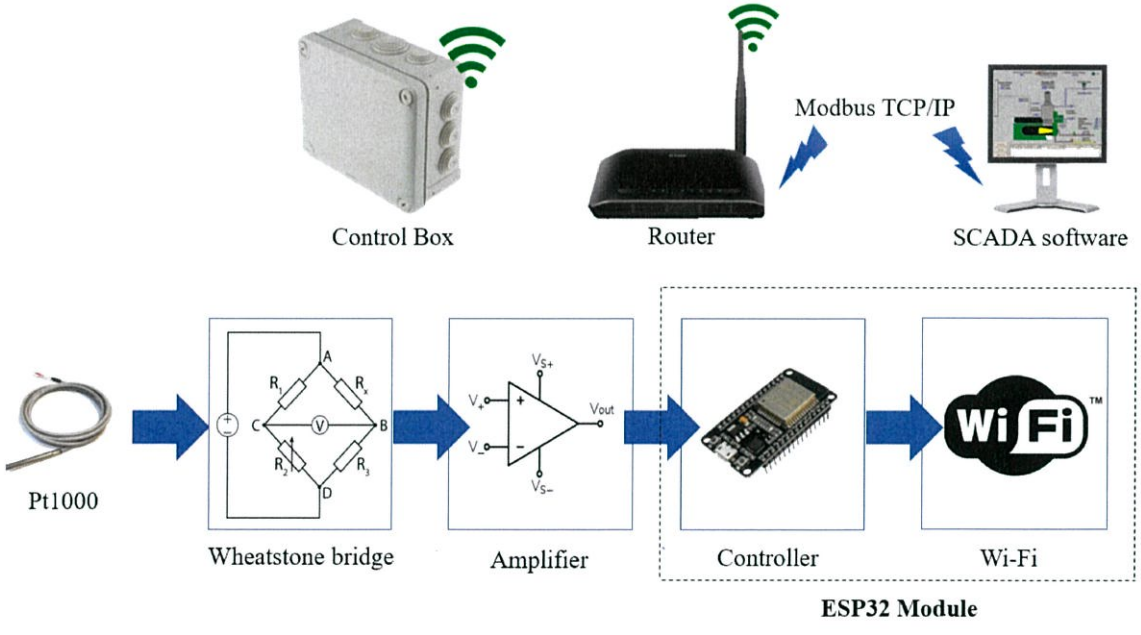
1. เราเตอร์ทำงานภายใต้ OSI เท่านั้น ไม่ติดต่อหรือส่งข้อมูลในรูปแบบของโปรโตคอลที่ไม่รู้จัก
2. ราคาแพงกว่าสวิตช์และฮับมาก

บทที่ 3
ขั้นตอนการดำเนินงาน

3.1 กล่าวนำ

โครงงานนำระบบเทคโนโลยีสมองกลฝังตัวมาประยุกต์ใช้เข้ากับอุปกรณ์วัดอุณหภูมิแบบ RTD Pt1000 โดยจะมีการใช้วงจรบริดจ์ในการปรับ Range หรือย่านของการวัด และมีวงจร Instrument Amplifier มาทำหน้าที่ขยายสัญญาณก่อนบอร์ด ESPino32 ที่ทำหน้าที่เป็นตัวประมวลผลของระบบสมองกลฝังตัวนี้และยังทำการเปรียบเทียบค่า และสร้างเลเยอร์ก่อนส่งออกมาเป็นเฟรมในโปรโตคอลการสื่อสารแบบ Modbus TCP/IP โดยโปรแกรมเหล่านี้ ถูกเขียนไว้ใน Arduino IDE ก่อนจะ Compile คำสั่งให้บอร์ด ESPino32 ได้ทำงาน และส่วนของงานซอฟแวร์จะเป็นการเขียนใน Wonderware System Platform เพื่อสังเกตุการณ์ค่า Parameter การทำ Device Diagnostic รวมไปถึงการเก็บ Historian เพื่อทำ Data log

3.2 โครงสร้างของระบบ

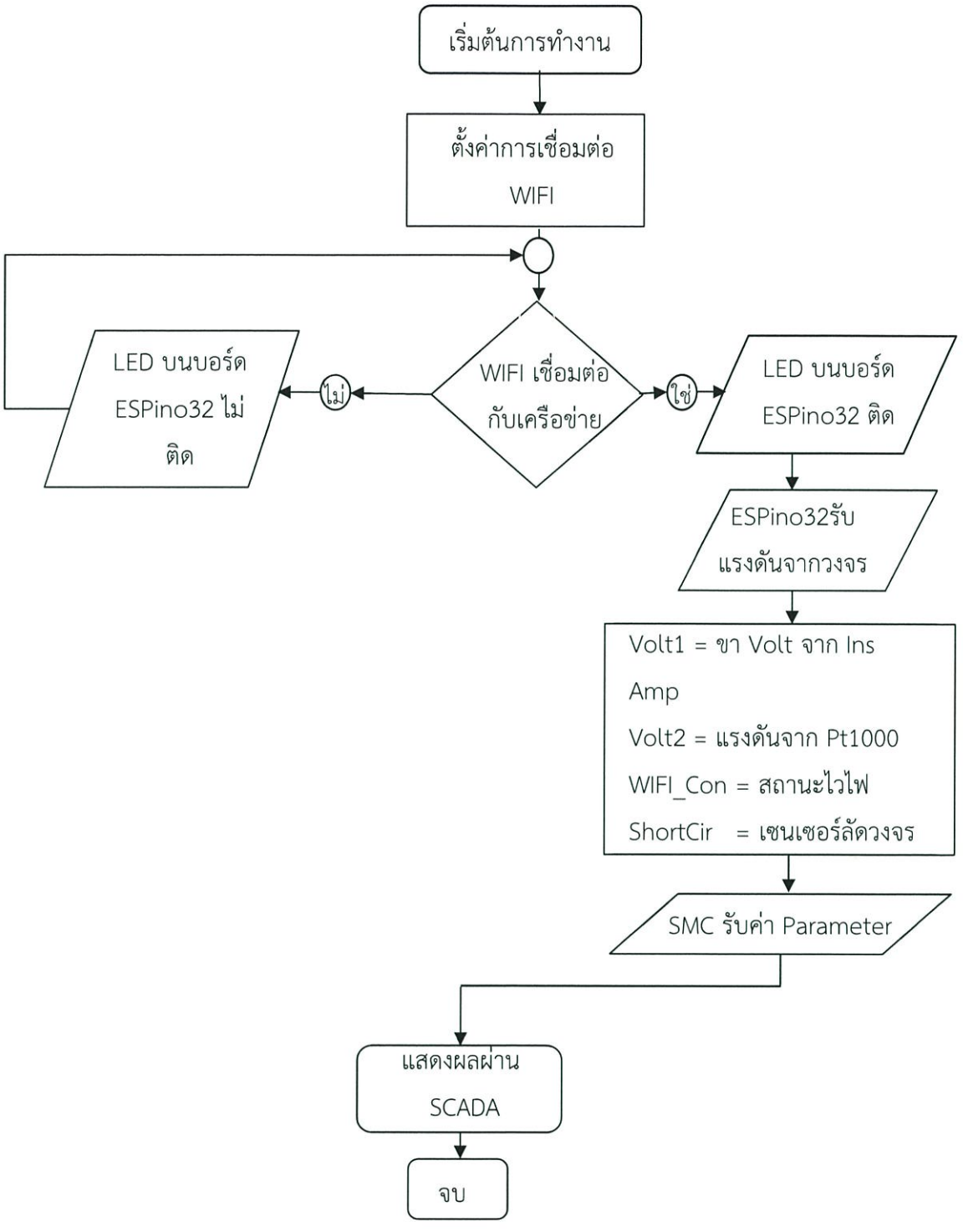


รูปที่ 3.1 อุปกรณ์ที่ติดตั้งใช้งานในระบบ

ตารางที่ 3.1 อุปกรณ์ที่ติดตั้งใช้งานในระบบ

อุปกรณ์	หน้าที่การทำงาน
Arduino ESPino32	เป็นตัวประมวลผลและส่งค่าผ่าน WIFI (Modbus TCP/IP)
Pt1000	อุปกรณ์ในการวัด
วงจร Bridge	เพื่อปรับ Range ของสัญญาณในย่านที่ต้องการ
วงจร Amplifier	เพื่อขยายสัญญาณ
คอมพิวเตอร์	ทำหน้าที่รับค่าจาก Router และขึ้นแสดงผลบนส่วนแสดงผล
Router	ทำหน้าที่เสมือน Gateway ของ Smart Device

จากรูปที่ 3.1 หลักการทำงานของระบบแบบง่าย เริ่มจากอุปกรณ์การวัดอุณหภูมิ Pt1000 ได้รับค่าความต้านทานซึ่งเปลี่ยนแปลงตามอุณหภูมิ จากนั้นเข้าวงจรวีตสโตนบริดจ์ และวงจร Instrument Amplifier ก่อนที่ Arduino ESPino32 Module จะส่งค่า Volt สุดท้ายที่ได้รับผ่าน Modbus TCP/IP protocol และคอมพิวเตอร์ที่ได้รับค่าผ่าน Router จะนำค่าที่ได้รับขึ้นแสดงผลบนสกาดา ซึ่งสามารถดูลำดับการทำงานเพิ่มเติมได้จาก flow chart ในรูปที่ 3.2



รูปที่ 3.2 Flow Chart

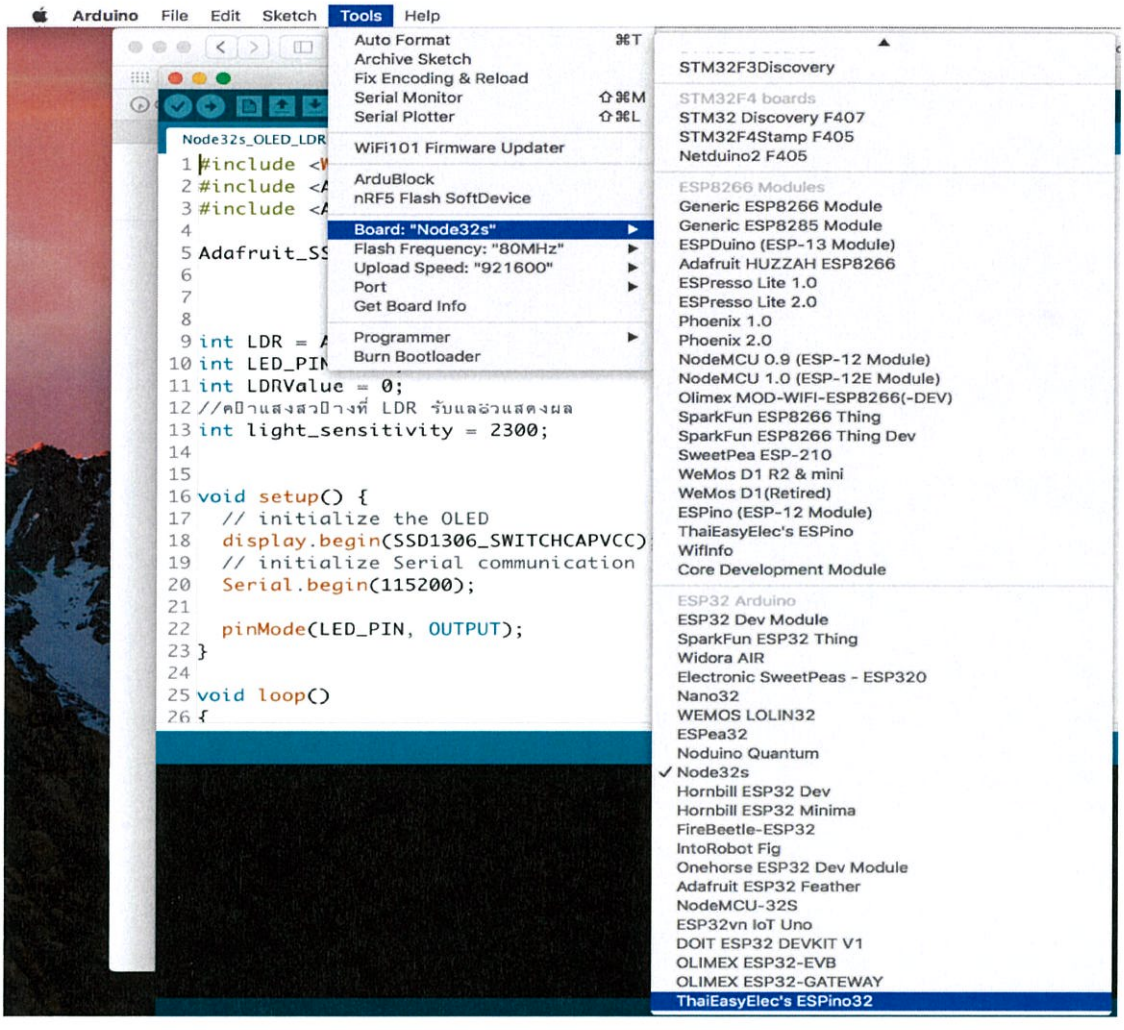
3.3 Hardware

3.3.1 Arduino ESPino32

บอร์ดที่ใช้ในการทำการประมวลผลวงจรคือบอร์ดรุ่น ESPino32 เป็นบอร์ดที่พัฒนาเพิ่มเติมจาก ESPin0 ซึ่งเป็นบอร์ดที่พัฒนาโดยคนไทย มีความเสถียรสูงและหา Libraries ได้ง่าย โดยมีความสามาถเพิ่มเติมดังนี้

1. ความสามารถในการสื่อสารเพิ่มเติมคือ Bluetooth
2. มี DAC เพิ่ม 2 pin
3. เพิ่ม pin Analog จากเดิมโดยมี 17 pin มีความละเอียด 12 bits หรือ 0–4095
4. สามารถปรับจ่ายไฟเลี้ยงวงจรได้จาก jumper

รูปที่ 3.3 ESPino32

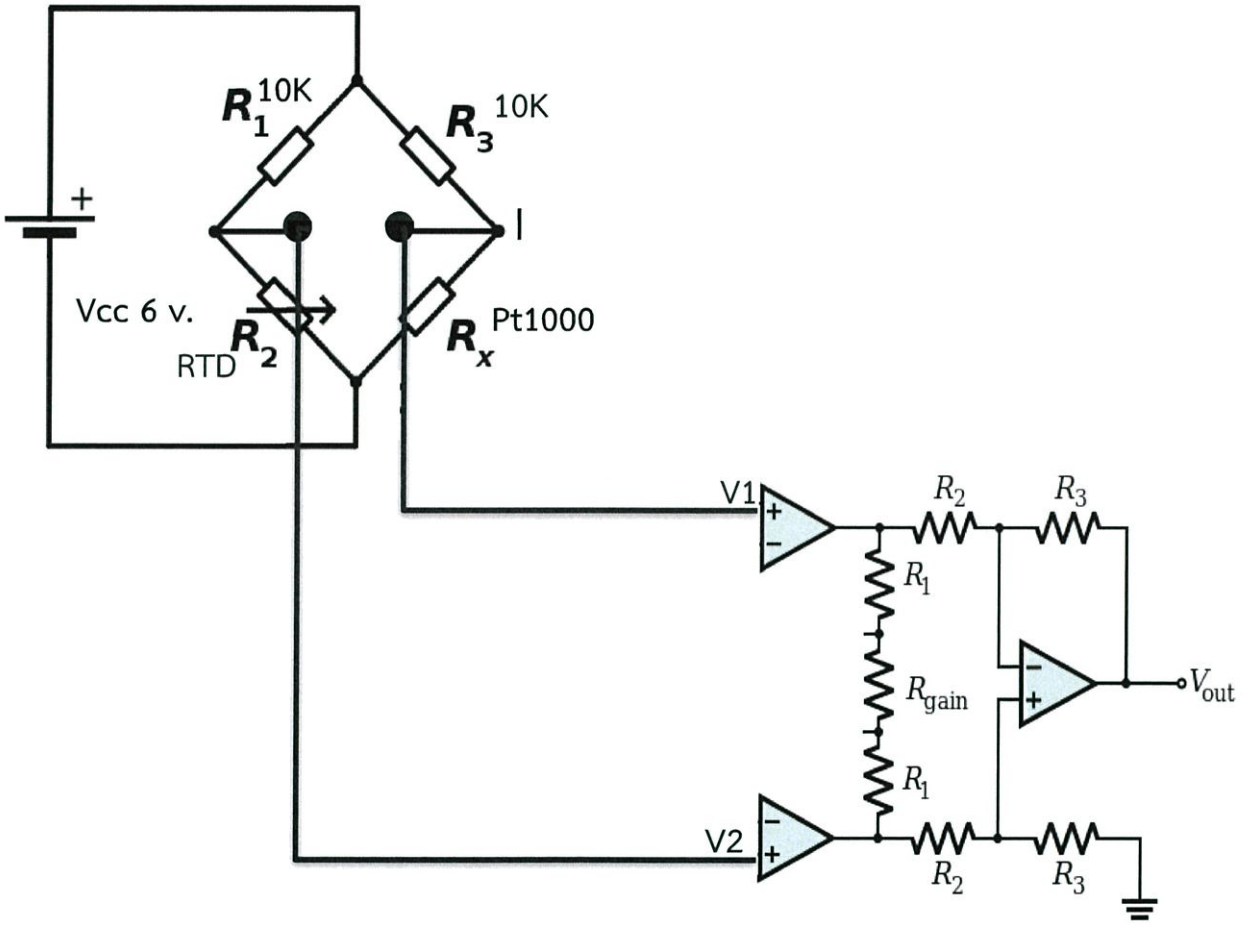


รูปที่ 3.4 Select Board

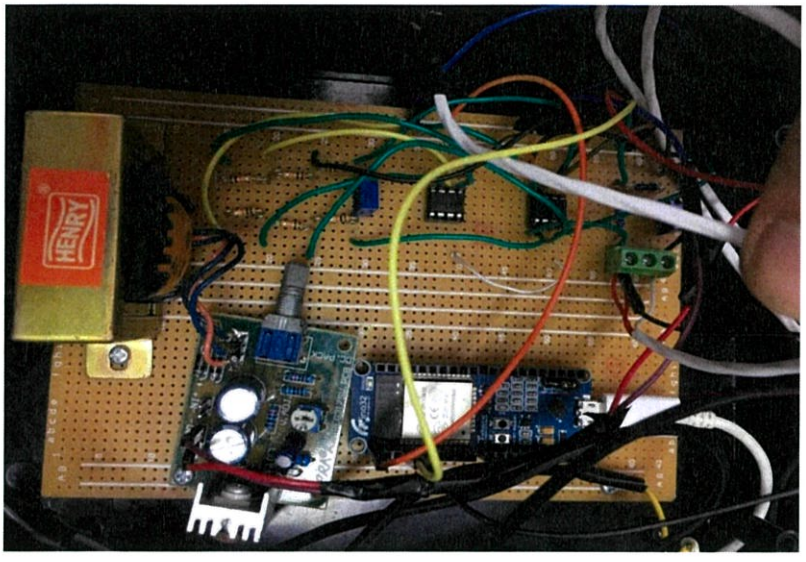
เนื่องจาก Arduino เป็น open source การเชื่อมต่อบอร์ดกับ Arduino IDE ก็สามารถทำได้ง่าย โดย Libraries สำหรับการดึงข้อมูลหรือชุดคำสั่งต่าง ๆ สามารถดาวน์โหลดได้ที่ github.com รวมไปถึง Core Component ที่จำเป็นก็สามารถดาวน์โหลดได้เช่นกัน ซึ่งทั้งสองอย่างข้างต้น เป็นสิ่งจำเป็นในการ Compile โปรแกรมลงบอร์ด ThaiEasyElec's ESPino32

3.3.2 Circuit Diagram

จากรูปที่ 3.5 คือวงจรภายในกล่องควบคุมที่นำมาใช้ มีวงจรบริดจ์เพื่อทำการปรับ GAIN ให้ได้ Range ที่ต้องการแต่ Volt ที่ออกมาต้องไม่เกิน 3.6 V เนื่องจากตัวบอร์ด Arduino ESPino32 ไม่สามารถรับมากกว่านี้ได้ (ยกเว้นต่อตัวต้านทานเพิ่ม) โดยจากการทดลอง ทำให้ทราบว่าที่ Pt1000 วัดอุณหภูมิ 100°C แรงดันที่ออกจาก Instrument Amplifier จะมีค่าเท่ากับ 2.904 V. และที่ 0°C จะมีค่าแรงดันประมาณ 0.036 V. ซึ่งถือเป็น Range ที่มีความละเอียดพอสมควร



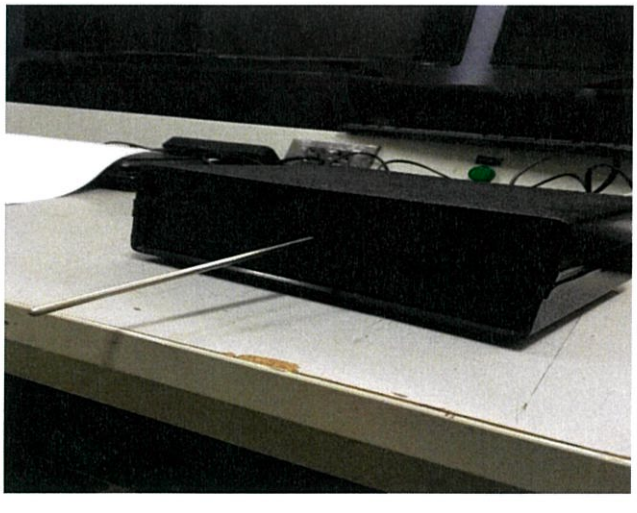
รูปที่ 3.5 Circuit Diagram



รูปที่ 3.6 วงจรภายในกล่อง

3.3.3 วงจรภายในกล่อง

เนื่องมาจากอุปกรณ์ที่ใช้เป็นอุปกรณ์อิเลคทรอนิกส์ จึงมีการนำ Power Supply มาใส่ร่วมในวงจรด้วย เพื่อแปลงไฟจากไฟบ้านทั่วไป 220 V. ให้เป็นไฟเลี้ยงแผ่นวงจร 6 V. และมีการนำ Arduino ESPino32 มาใส่ร่วมภายในกล่อง โดยจะมีการนำสายออกไปนอกกล่องเพียงสองสาย คือสายไฟเลี้ยงวงจรและสายUART ที่จะใช้ในกรณีที่มีการอัพเดตข้อมูลโปรแกรมเพิ่มเติม โดยถ้าไม่การอัพเดตเพิ่ม ก็ไม่ต้องใช้สาย UART เข้ามา โดยนอกกล่องจะมีเพียง Sensor Pt1000 ที่ทำหน้าที่วัดอุณหภูมิดังรูปที่ 3.7



รูปที่ 3.7 กล่องอุปกรณ์วัดอุณหภูมิระบบสมองกลฝังตัว

3.4 การเขียนโค้ดในโปรแกรม Arduino IDE

modbus_ip

```
#include <WiFi.h>
#include <Modbus.h>


// Modbus Registers Offsets (0-9999)
const int TEST_HREG = 100;


//ModbusIP object
ModbusIP mb;

void setup() {
  Serial.begin(115200);

  mb.config("your_ssid", "your_password");

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());

  mb.addHreg(TEST_HREG, 0xABCD);
}
```

รูปที่ 3.8 ต้นแบบการสื่อสาร Modbus

รูปที่ 3.8 คือโปรแกรมที่เขียนขึ้นมาเพื่อใช้ในการสื่อสารผ่าน Modbus TCP/IP โดยโปรโตคอล TCP เป็นโปรโตคอลที่ใช้รับ/ส่งข้อมูลที่นิยมใช้งานมากที่สุดในโลกอินเตอร์เนตเนื่องจากการรับ/ส่งข้อมูลบนโปรโตคอล TCP จะมีการตรวจสอบผลข้อมูลทุกครั้ง ข้อมูลที่ได้จึงมีความถูกต้องตามลำดับเนื่องจากมีการแบ่งชั้นข้อมูลเป็น Layer ที่เข้าใจตรงกันในการสื่อสาร ทำให้ถูกใช้เป็นพื้นฐานของโปรโตคอลประยุกต์ต่าง ๆ

การใช้งานโปรโตคอล TCP จะแบ่งออกเป็น 2 ฝั่ง คือ

1. TCP Server คือฝั่งที่ทำหน้าที่เปิดพอร์ตและรอการเชื่อมต่อเข้ามาเพื่อสื่อสารด้วย การใช้ ESP32ino32 ทำเป็น TCP Server จะสะดวกต่อการรับ-ส่งข้อมูลกับอุปกรณ์อื่น ๆ มากที่สุด

2. TCP Client คือฝั่งที่ทำหน้าที่เชื่อมต่อไปยัง Server เพื่อเริ่มการสื่อสาร มักจะหมายถึงอุปกรณ์ที่สามารถเคลื่อนที่ได้ หรือเลือกสื่อสารได้ เช่น คอมพิวเตอร์ สมาร์ทโฟนการสร้าง TCP Server จะต้องเรียกใช้ไลบารี่ WIFI.h ทุกครั้งที่ทำการคอมไพล์ข้อมูลหรือเรียกใช้งาน ซึ่งการใช้งานจะต้องสร้างออปเจคของคลาส WiFiServer ขึ้นมานอกฟังก์ชั่นย่อย มีรูปแบบดังนี้

WiFiServer: WiFiServer (int port=80, int max_clients=4);

sketch_feb20a

```
#include<WiFi.h>

#define WIFI_STA_NAME "PI_LAB"
#define WIFI_STA_PASS "PILAB2561"

void setup() {
  Serial.begin(115200);
  pinMode(LED_BUILTIN, OUTPUT);

  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WIFI_STA_NAME);

  WiFi.mode(WIFI_STA);
  WiFi.begin(WIFI_STA_NAME, WIFI_STA_PASS);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
    digitalWrite(LED_BUILTIN, !digitalRead(LED_BUILTIN));
  }

  digitalWrite(LED_BUILTIN, HIGH);
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
```

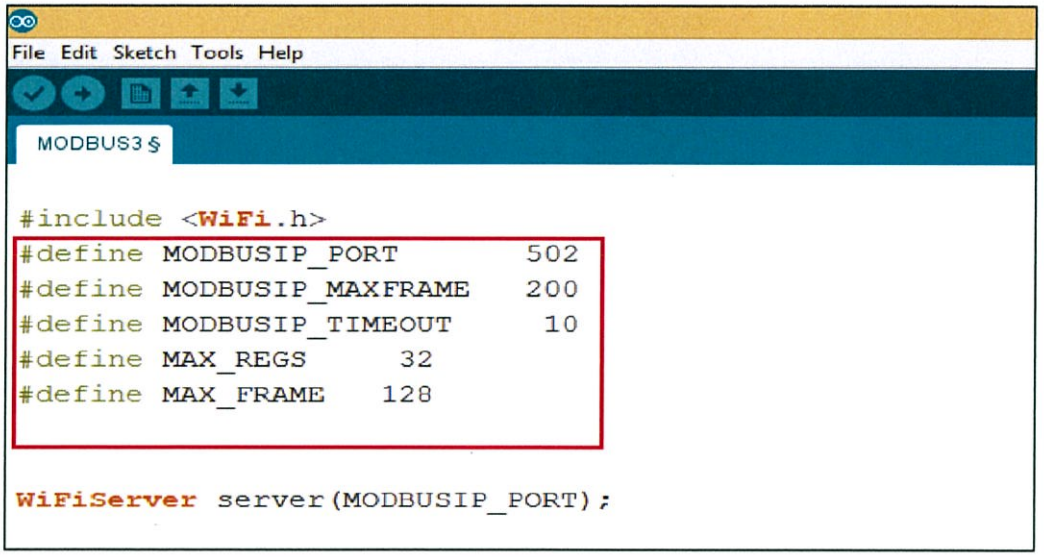
รูปที่ 3.9 ทดลองใช้กับวง LAN ของภาควิชา

การประกาศตัวแปรฟังก์ชั่น Modus TCP/IPโดยจะเห็นว่าในส่วนสำคัญจะมีลักษณะตัวแปรเป็น modbus tcp_port คือการเลือก port ในการเชื่อมต่อว่าต้องการเชื่อมต่อ port ใดซึ่ง port การสื่อสารที่เข้าใจตรงกันสำหรับการสื่อสารแบบ Modbus คือ port 502 และการประกาศตัวแปรฟังก์ชั่นนั้นจะนำไปสู่ในรูปที่ 3.10

```
sketch_mar20a §
#include < WiFi.h >

const char* ssid = "**********";
const char* password = "**********";
int ModbusTCP_port = 502;

//////// Required for Modbus TCP / IP /// Requerido para Modbus TCP/IP /////////
#define maxInputRegister 20
#define maxHoldingRegister 20

#define MB_FC_NONE 0
#define MB_FC_READ_REGISTERS 3 //implemented
#define MB_FC_WRITE_REGISTER 6 //implemented
#define MB_FC_WRITE_MULTIPLE_REGISTERS 16 //implemented
//
// MODBUS Error Codes
//
#define MB_EC_NONE 0
#define MB_EC_ILLEGAL_FUNCTION 1
#define MB_EC_ILLEGAL_DATA_ADDRESS 2
#define MB_EC_ILLEGAL_DATA_VALUE 3
#define MB_EC_SLAVE_DEVICE_FAILURE 4
//
// MODBUS MBAP offsets
//
#define MB_TCP_TID 0
#define MB_TCP_PID 2
#define MB_TCP_LEN 4
#define MB_TCP_UID 6
#define MB_TCP_FUNC 7
#define MB_TCP_REGISTER_START 8
#define MB_TCP_REGISTER_NUMBER 10
```

รูปที่ 3.10 การประกาศตัวแปรฟังก์ชั่น code Modbus TCP/IP

ในส่วนของการส่งค่าเริ่มต้นจุดที่สำคัญนั้นคือการเรียกใช้ไลบรารี่ที่จะใช้ในการทำงาน การประมวลผลคล้าย ๆ การประกาศตัวแปร <stdio.h> ในภาษา C++ นั่นคือ Libraries <WiFi.h> เพื่อที่จะสามารถเชื่อมต่อไปยังเครือข่ายไร้สายได้ และที่วงสีแดงด้านล่างคือค่าตัวแปรต่าง ๆ ที่ใช้ในการกำหนดช่องทางการสื่อสาร เช่น MODBUSIP_PORT = 502 คือการเลือกช่องทางในการสื่อสาร โดย Port 502 คือข้อตกลงของ MODBUS ตามที่กล่าวไว้ข้างต้น



รูปที่ 3.11 การประกาศเรียกใช้ Libraries

ในส่วนของการเชื่อมต่อ WIFI นั้น จะมีฟังก์ชั่น Code ดังนี้ โดยในบรรทัดที่ 3 จะเป็นการตรวจสอบว่า WIFI ได้เชื่อมต่อหรือยัง เพราะ ใน While เมื่อ WIFI. Status () ยังไม่ตรงกับ เงื่อนไข WL_CONNECTED จะมี LED ในขาที่ 16 ติดอยู่นั้นคือยังไม่สามารถเชื่อมต่อได้

```
void Re_WIFI_Connect(){

  //WiFi.config(ip, gateway, subnet);
  mb.config("Heys", "11110000");

 while (WiFi.status() != WL_CONNECTED) {
    delay(100);
    digitalWrite(16 , HIGH);

  }

 IPAddress local_ip = {192,168,1,5};    //ตั้งค่า IP
 IPAddress gateway={192,168,1,1};       //ตั้งค่า IP Gateway
 IPAddress subnet={255,255,255,0};      //ตั้งค่า Subnet
 WiFi.config(local_ip,gateway,subnet);    //setค่าไปยังโมดูล
```

รูปที่ 3.12 เซ็ตค่า IPAddress และ Config ค่าต่างให้กับตัว โมดูล ESP32

ในวงกลมสีแดงคือการเซ็ตค่า IPAddress และ Config ค่าต่างให้กับตัวโมดูล ESP32

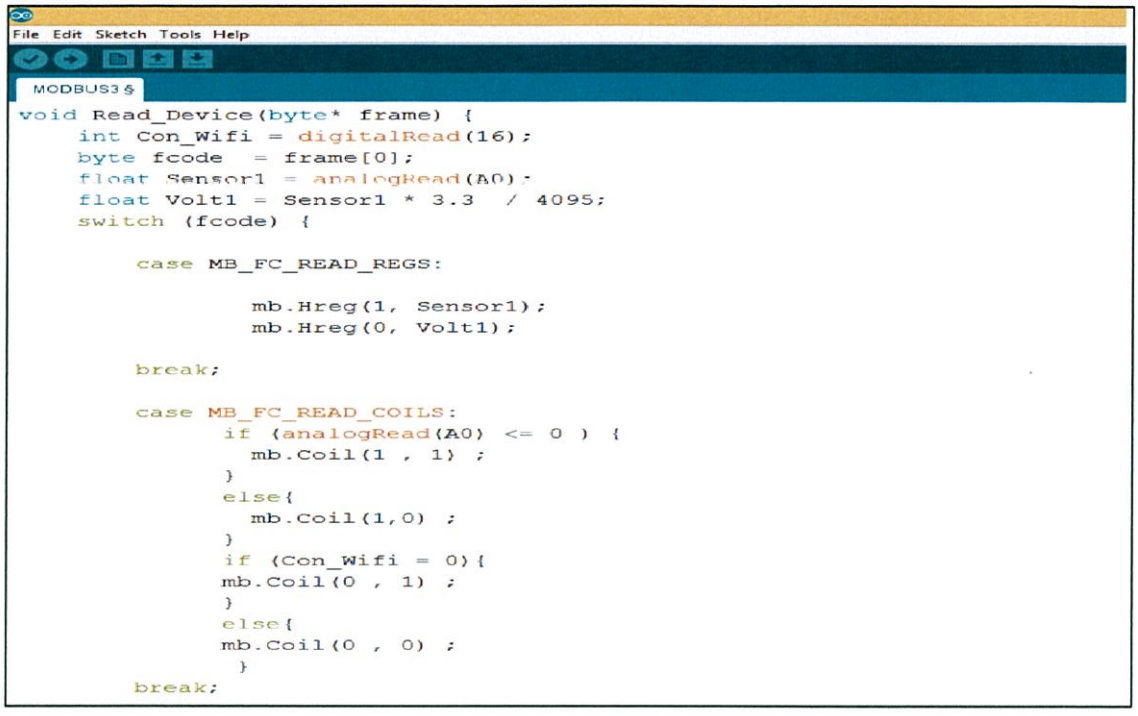
File Edit Sketch Tools Help

MODBUS3§

```
//Function Codes
enum {
    MB_FC_READ_COILS        = 0x01, // Read Coils (Output) Status 0xxxx
    MB_FC_READ_INPUT_STAT   = 0x02, // Read Input Status (Discrete Inputs) 1xxxx
    MB_FC_READ_REGS         = 0x03, // Read Holding Registers 4xxxx
    MB_FC_READ_INPUT_REGS   = 0x04, // Read Input Registers 3xxxx
    MB_FC_WRITE_COIL        = 0x05, // Write Single Coil (Output) 0xxxx
    MB_FC_WRITE_REG         = 0x06, // Preset Single Register 4xxxx
    MB_FC_WRITE_COILS       = 0x0F, // Write Multiple Coils (Outputs) 0xxxx
    MB_FC_WRITE_REGS        = 0x10, // Write block of contiguous registers 4xxxx
};

//Exception Codes
enum {
    MB_EX_ILLEGAL_FUNCTION = 0x01, // Function Code not Supported
    MB_EX_ILLEGAL_ADDRESS  = 0x02, // Output Address not exists
    MB_EX_ILLEGAL_VALUE    = 0x03, // Output Value not in Range
    MB_EX_SLAVE_FAILURE    = 0x04, // Slave Deive Fails to process request
};

//Reply Types
enum {
    MB_REPLY_OFF    = 0x01,
    MB_REPLY_ECHO   = 0x02,
    MB_REPLY_NORMAL = 0x03,
};
```

รูปที่ 3.13 Function Code Modbus

ส่วนของรูป 3.13 จะเป็นการประกาศ Function Code ของ Modbus ที่ใช้ในการสื่อสาร จะแบ่งเป็น Read / Write โดยทุกตัวจะมี Function Read หมด แต่ตัวที่มี Function Write จะมีแค่ Coil Status กับ Holding Register เท่านั้น



รูปที่ 3.14 การอ่านค่าที่กำหนดแล้วส่งไปยัง Address

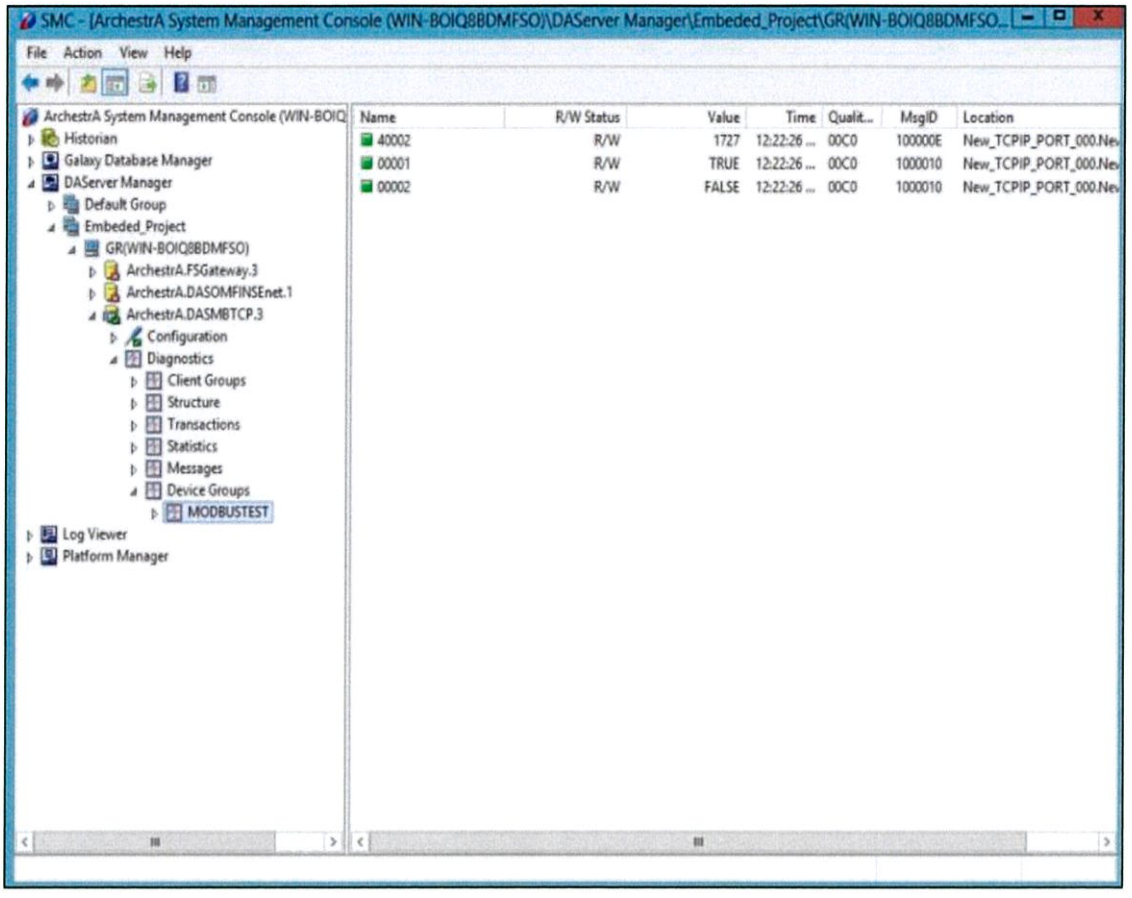
ภายในรูปที่ 3.14 โปรแกรมส่วนนี้จะเป็น ในส่วนของ การอ่านค่าที่กำหนด แล้วส่งไปยัง Address ต่างที่เราอยากจะส่งโดยถ้าสังเกตุ ตัวแปร Sensor1 จะเป็นตัวแปรที่อ่านค่า Analog โดย ค่าAnalog ที่อ่านได้นั้น จะเป็น บิต 12 บิต นั้นก็คือ 0-4095 โดยเราจะต้องนำมาแปลงค่าให้เป็นค่า แรงดันจริง โดยคูณค่าแรงดันไฟฟ้าเข้ากับค่าที่อ่านได้แล้วหารด้วย 4095 ก็จะได้ค่าแรงดันไฟฟ้าที่เราต้องการ

สมมุติให้ค่า 40002 จาก HOLDING REGISTER เป็น T จะได้สมการคำนวณหาค่า Volt ดังนี้

$$\text{Volt} = T\,\frac{3.3}{4095} \tag{3.1}$$

3.5 ส่วนแสดงผล

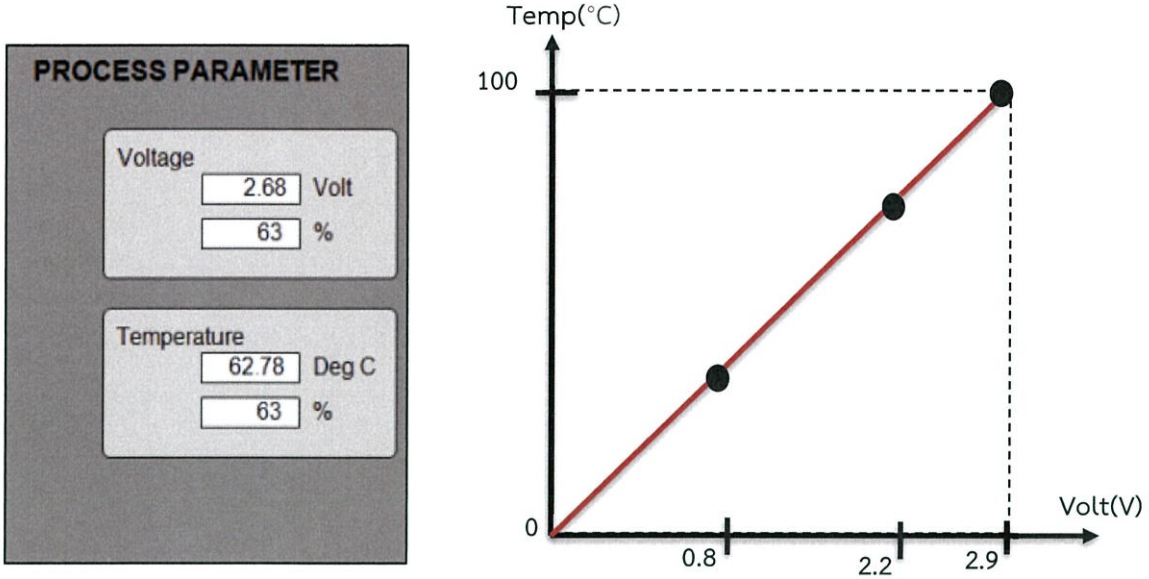
3.5.1 Tag name ในส่วนแสดงผล



รูปที่ 3.15 ค่าที่ดึงมาจาก Modbus

ก่อนจะเข้าสู่ระบบ HMI ที่เขียนไว้ให้เปิดโปรแกรม SMC หรือ System Management Console ที่ทำหน้าที่เป็นตัวกลางการสื่อสาร (Drivers) ให้กับ Modbus TCP/IP และส่วนแสดงผลของ ArchestrA โดย Device Group ที่แสดงมาจากการเขียนโปรแกรมลงใน ESPino32 ซึ่งมีการเขียนกำหนดค่าและเงื่อนไขไว้ 3 ตัว ดังนี้ 40001 คือค่าที่มาจาก HOLDING REGISTER เป็นค่า Volt จากวงจร ในที่นี้จะนำมาเข้าสูตรแปลงตามสมการเส้นตรงเพื่อให้ได้ค่าอุณหภูมิออกมา 00001 คือค่าที่เขียนโปรแกรมไว้ว่าหาก WIFI เชื่อมต่อเมื่อไหร่ให้ขึ้นค่า “TRUE”มีค่าเป็น “1” และ Tag name สุดท้าย 00002 คือสถานะของ Sensor ว่าอุปกรณ์วัด Pt1000 มีการทำงานผิดปกติหรือไม่ โดยถ้า Sensor มีการลัดวงจร Value จะเปลี่ยนจาก “FALSE” เป็น“TRUE”

3.5.2 หน้าต่างแสดงค่า PV



รูปที่ 3.16 หน้าต่างแสดงค่า Parameter และวิธีคำนวณ

ค่าตัวแปร Volt ที่ปรากฏดึงตัวแปรมาจาก Arduino โดยตรงผ่าน Modbus TCP/IP โดย Tag ที่แสดงใน Modbus ของค่า Volt คือ 40001 และ 40002 โดยที่ 40001 เป็นค่าแรงดันในรูปแบบ Integer (เป็นตัวเลขปัดเศษ ไม่มีจุดทศนิยม) และ 40002 เป็นค่าแรงดันในรูปแบบ Float (แสดงจุดทศนิยม)

ค่าตัวแปร Temperature คือการนำค่า Volt มาเข้าสมการเส้นตรงเพื่อแปลงค่าให้อยู่ในหน่วยของอุณหภูมิ (สามารถเขียนสคริปในโปรแกรม ArchestrA System Platform ได้เลย)

สูตรคำนวณ

หาความชัน

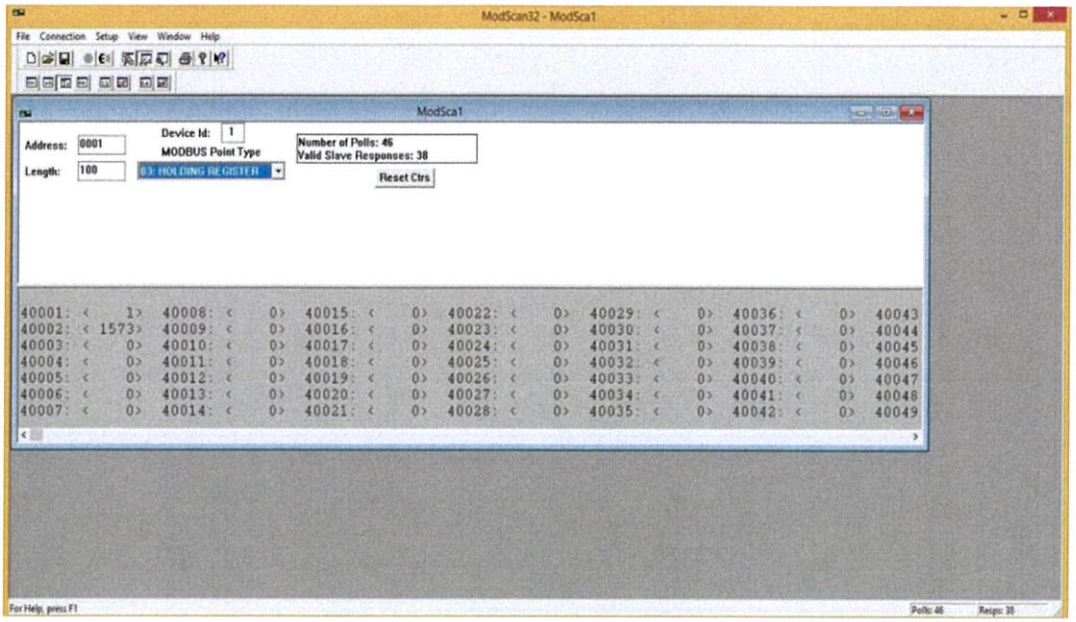
$$\frac{y2-y1}{x2-x1} = \frac{100}{2.848} = 35.1 \quad (3.2)$$

จากสมการเส้นตรง

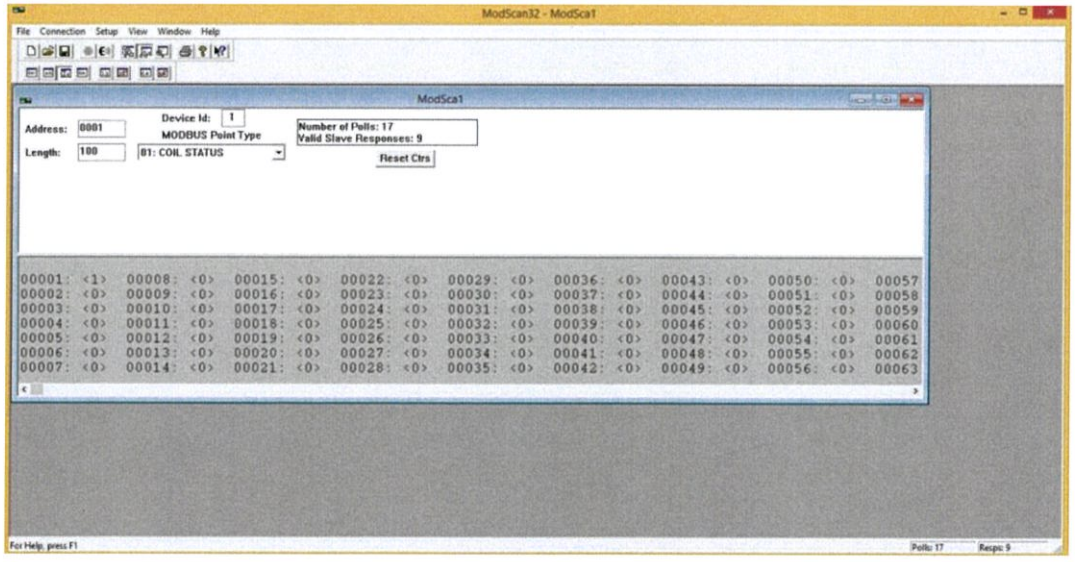
$$Y = mx + c \quad (3.3)$$

จะได้สมการ

$$\text{Temperature} = 35.1(\text{Volt}) - 1.825 \quad (3.4)$$

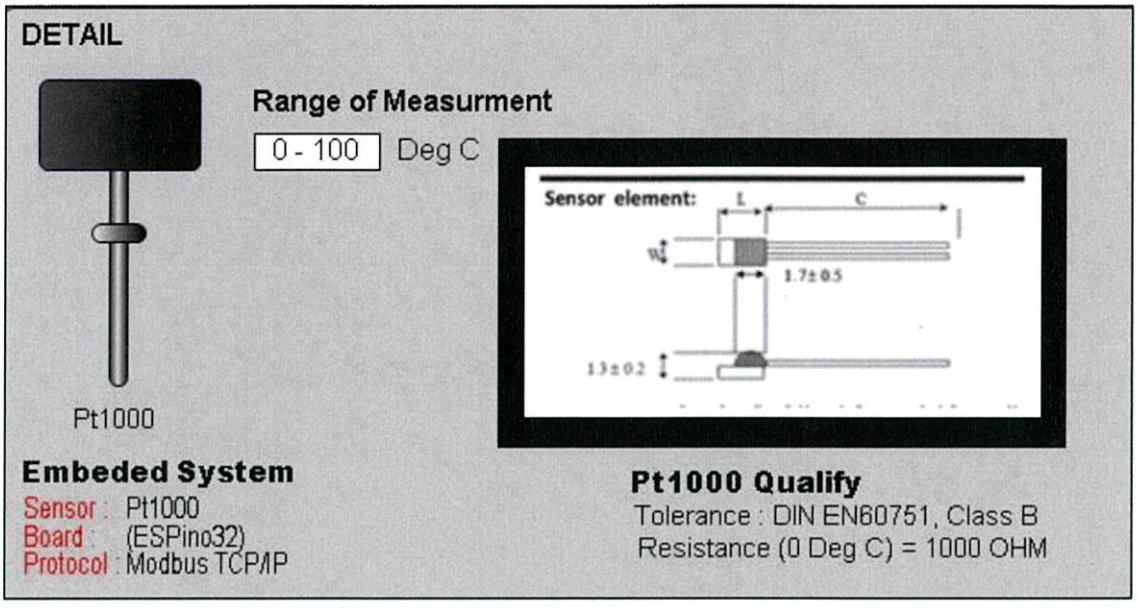


รูปที่ 3.17 ค่าจาก Holding Register



รูปที่ 3.18 ค่าจาก Coil Status

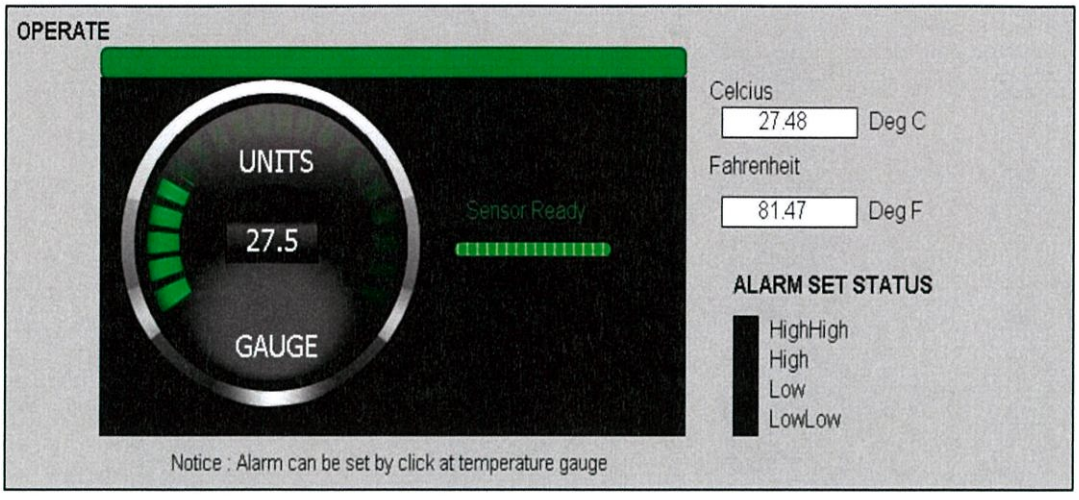
3.5.3 รายละเอียดโดยรวมและอุปกรณ์วัด Pt1000



รูปที่ 3.19 Equipment Detail

ส่วนบ่งบอกรายละเอียดและคุณลักษณะของอุปกรณ์ ซึ่งในที่นี้ อุปกรณ์ที่เป็น Interface กับกระบวนการโดยตรงคือ Pt1000 จากนั้นประมวลผลและสร้างชั้นข้อมูล (Layers) ของการสื่อสารขึ้นมาเพื่อเป็นข้อตกลงที่ตรงกันระหว่างผู้ส่งและผู้รับ ก่อนจะส่งค่าตัวแปรกระบวนการ ผ่านระบบไร้สาย Modbus TCP/IP Protocol โดยมีย่านการวัดอยู่ที่ 0 – 100 Degree Celsius เนื่องจากมีการกำหนดย่านการวัดจากการใช้วงจรแบ่งแรงดัน Wheat Stone Bridge มีคุณลักษณะเฉพาะทางฮาร์ดแวร์ของ Sensor และบอกค่าความแม่นยำตามมาตรฐาน

3.5.4 หน้าต่าง Monitoring และ Operate



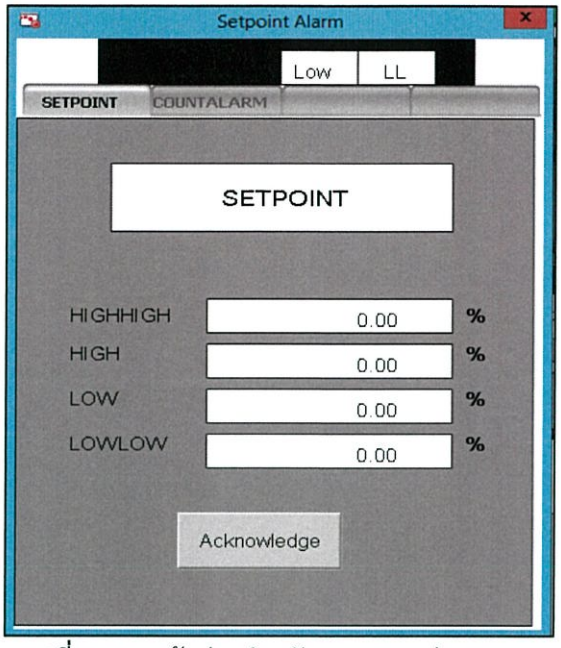
รูปที่ 3.20 เกจบอกอุณหภูมิ

เกจบอกอุณหภูมิและสเตตัสการทำงานของ Sensor พร้อมทั้งบอกอุณหภูมิในสองหน่วยที่ใช้กันอย่างสากลคือองศาฟาเรนไฮต์ (Fahrenheit) และองศาเซลเซียส (Celsius) โดยค่าองศาฟาเรนไฮต์ ได้มาจากสูตรคำนวณการแปลงหน่วย

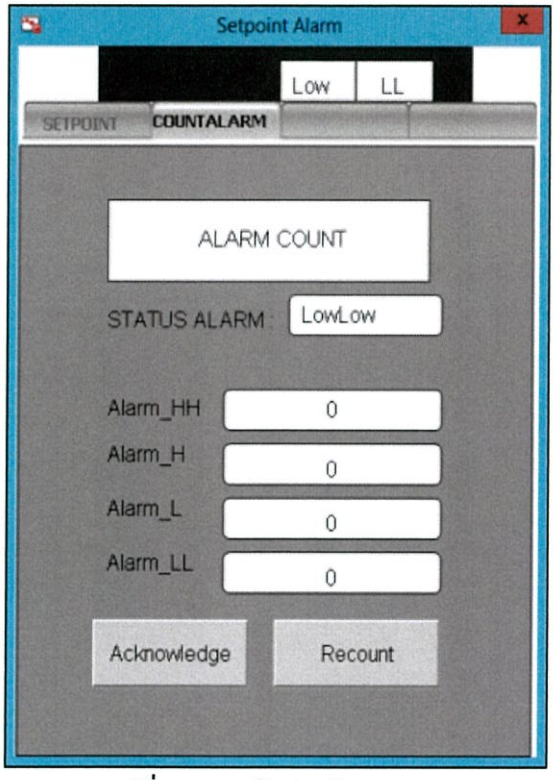
Celsius to Fahrenheit

$$T\,(\text{Fahrenheit}) = T\,(\text{Celsius})\ \frac{9}{5}\ + 32 \qquad (3.5)$$

ด้านล่างของอุณหภูมิทั้งสองหน่วยคือสถานะของกระบวนการ แบ่งเป็นสี่สถานะ ยกตัวอย่างเช่น เมื่ออุณหภูมิเกินค่าที่ตั้งไว้ในสถานะ High ไฟสีแดงจะขึ้นบ่งบอกสถานะหรือแจ้งเตือน โดยค่าของสถานะเหล่านี้สามารถ Set ไว้ได้ โดยการคลิกที่หน้าปัดแสดงผล จะแสดงหน้าต่างสำหรับการ Set ค่า Alarm และ Alarm Count เพื่อเก็บเป็นข้อมูลว่าเคยเกิดปัญหา Alarm กี่ครั้ง เพื่อนำไปใช้วิเคราะห์ปัญหาได้

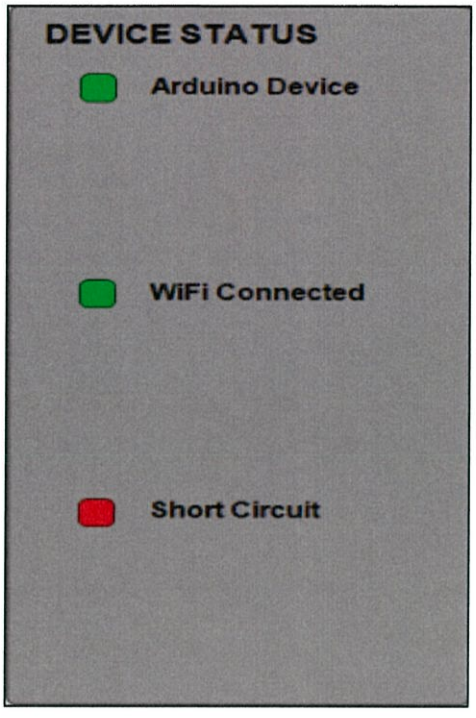


รูปที่ 3.21 หน้าต่างสำหรับการ Set ค่า Alarm



รูปที่ 3.22 หน้าต่างนับ Alarm

3.5.5 สัญญาณไฟบ่งบอกสถานะอุปกรณ์

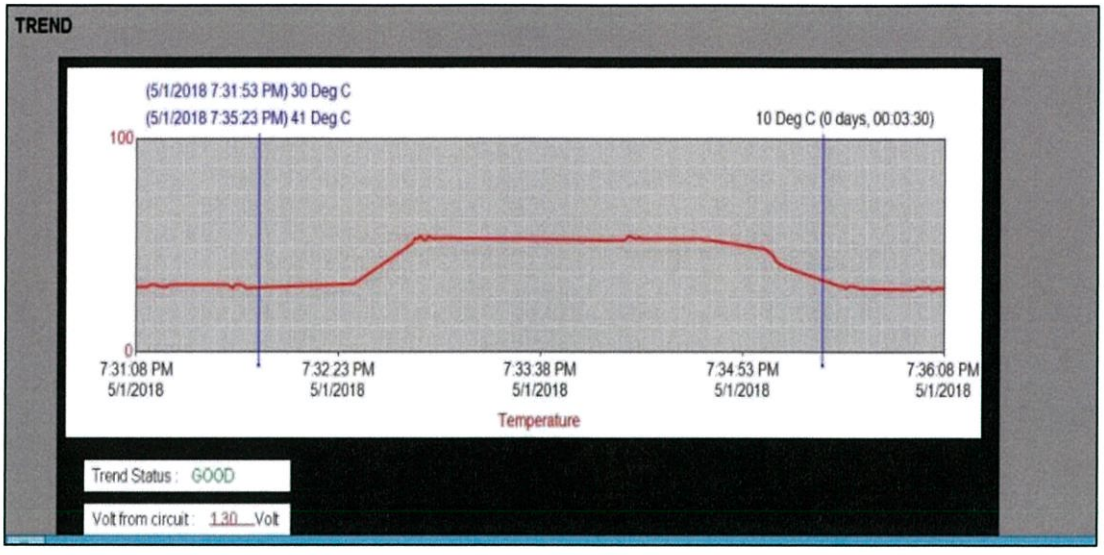


รูปที่ 3.23 Device Status

ส่วนที่บ่งบอกสถานะ เปิด/ปิด หรือการเชื่อมต่อของอุปกรณ์ โดยที่ดึงค่ามาจาก COIL STATUS ของ Modbus TCP/IP ซึ่งถูกเขียนโปรแกรมให้ส่งค่ามาแล้วจาก Arduino IDE ซึ่งต่างจากค่า Parameter ที่เป็น Volt ที่จะถูกดึงค่ามาจาก HOLDING REGISTER ของ Modbus

1. สัญญาณไฟ Arduino Device หมายถึงตัวบอร์ด ESPino32 ต่อกับวงจรหรือไม่
2. สัญญาณไฟ WIFI Connected หมายถึง สถานะของการส่งข้อมูลผ่านทาง WIFI
3. สัญญาณไฟ Short Circuit หมายถึง มีการลัดวงจรไฟฟ้าภายในอุปกรณ์

3.5.6 กราฟแสดงสถานะ



รูปที่ 3.24 Trend Status

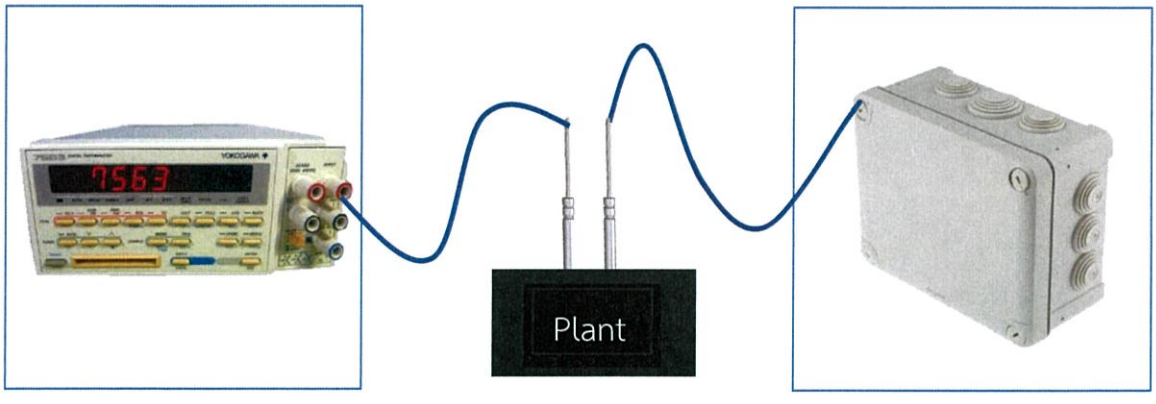
กราฟที่แสดงดังรูปที่ 3.24 คือกราฟแสดงอุณหภูมิภายในช่วงเวลา 3 นาที โดยตัวแปรที่นำมาใช้ในการแสดงผลกราฟคือตัวแปร Temp ที่ประกาศเป็นตัวแปร Global ใน System Platform ซึ่งนอกจากจะมีการแสดงผลที่หน้าต่างนี้แล้ว ยังมีการเก็บ Historian เข้าใน SMC ให้สามารถดูค่าย้อนหลังเพื่อวิเคราะห์ปัญหาได้ ที่ขอบด้านซ้ายล่างของหน้าจอแสดงกราฟอุณหภูมิ คือ สเตตัสของกราฟ และ แรงดันที่ถูกส่งเข้ามา

บทที่ 4
การทดลองและผลการทดลอง

สำหรับในบทนี้จะเป็นการทดสอบระบบวัดอุณหภูมิแบบไร้สายโดยใช้อุปกรณ์ฝังตัวที่สร้างขึ้นมากับแพลนต์ควบคุมอุณหภูมิ โดยในการทดสอบระบบจะแยกเป็นสามส่วนคือ

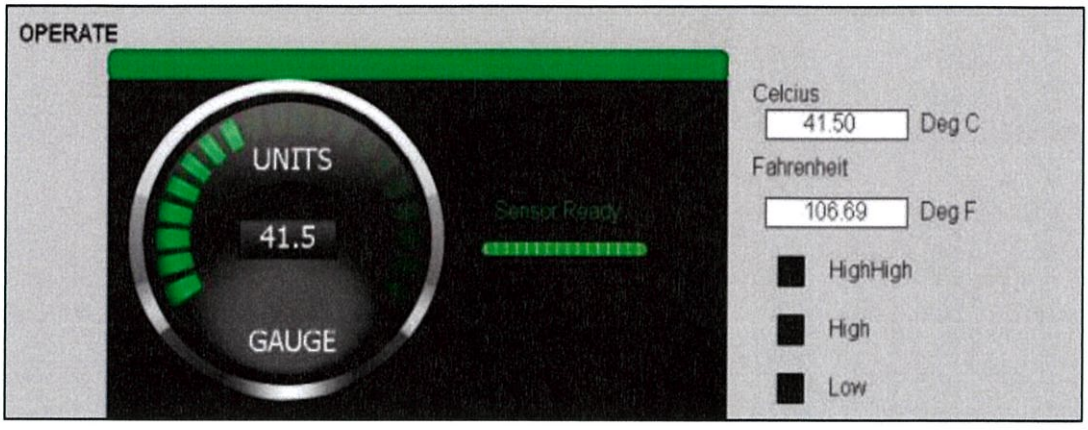
- การทดลองวัดอุณหภูมิเทียบกับอุปกรณ์วัดอุณหภูมิมาตรฐานแบบดิจิตอล
- การทดสอบการทำงานของส่วนแสดงผลของอุณหภูมิผ่านทาง SCADA Software
- การทดสอบการทำงานของส่วนแสดงผลสถานะการทำงานของระบบผ่านทาง SCADA Software

4.1 การทดลองวัดอุณหภูมิเทียบกับอุปกรณ์วัดอุณหภูมิมาตรฐานแบบดิจิตอล



รูปที่ 4.1 การทดลองเทียบอุณหภูมิ

โดยการทดลองนี้เพื่อหาค่าความแม่นยำของระบบวัดอุณหภูมิแบบไร้สายโดยใช้อุปกรณ์ฝังตัวที่สร้างขึ้น โดยการทดลองจะทำการเปรียบเทียบค่าอุณหภูมิที่วัดได้กับอุปกรณ์วัดอุณหภูมิแบบดิจิตอล Yokogawa 7563 digital thermometer โดยจะทำการปรับค่าอุณหภูมิของแพลนต์ควบคุมอุณหภูมิตั้งแต่ 0-100 องศาเซลเซียส เพื่อตรวจสอบว่าค่าอุณหภูมิที่วัดได้จากระบบที่สร้างขึ้นมามีค่าความคลาดเคลื่อนจากค่าอุณหภูมิที่วัดได้จากอุปกรณ์วัดอุณหภูมิมาตรฐานแบบดิจิตอลเท่าไร โดยผลการทดลองที่ได้สามารถแสดงได้ดังรูปที่ 4.2-4.3 และตารางที่ 4.1 ตามลำดับ



รูปที่ 4.2 หน้าปัดแสดงผลการวัดของระบบที่สร้างขึ้นมา

รูปที่ 4.3 Yokogawa 7563 digital thermometer

ตารางที่ 4.1 ผลการทดลองการวัดอุณหภูมิเทียบกับอุปกรณ์วัดอุณหภูมิมาตรฐานแบบดิจิตอล

อุณหภูมิ(°c)	ระบบที่นำเสนอ (°c)	Yokogawa 7563 (°c)	ค่าความผิดพลาดของอุณหภูมิที่วัดได้
0	-	-	-
25	24.9	25.5	-2.35 %
50	50.7	50.4	0.60 %
75	76.1	75.2	1.20 %
100	100	99.8	0.2 %

จากตารางที่ 4.1 แสดงผลการทดลองการวัดอุณหภูมิเทียบกับอุปกรณ์วัดอุณหภูมิมาตรฐานแบบดิจิตอล โดยเมื่อทำการคำนวณค่าความผิดพลาดของอุณหภมิที่วัดได้จากระบบที่สร้างขึ้นมาเมื่อเทียบกับค่าที่ได้จากอุปกรณ์วัดอุณหภูมิมาตรฐานแบบดิจิตอลได้ จะมีค่าความผิดพลาดสูงสุดประมาณ -2.35 %

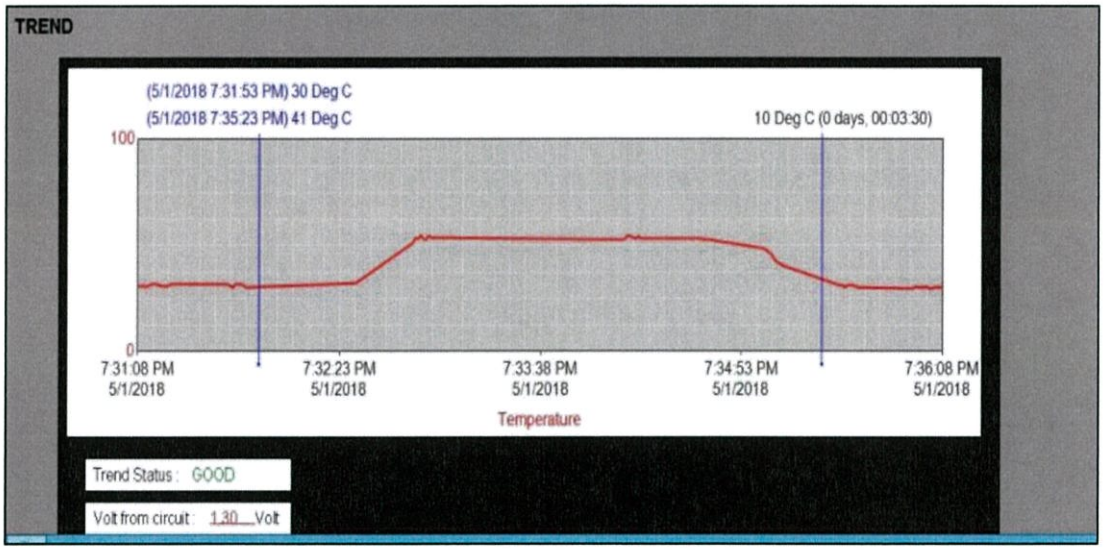
4.2 การทดสอบการทำงานของส่วนกราฟแสดงผลของอุณหภูมิผ่าน SCADA Software

สำหรับในการทดสอบนี้จะทำการทดสอบการแสดงกราฟอุณหภูมิที่วัดได้โดยระบบวัดอุณหภูมิแบบไร้สาย โดยใช้อุปกรณ์ฝังตัวที่สร้างขึ้นมา โดยการทดสอบจะแบ่งออกเป็น 2 กรณีคือ

- สถานะการทำงานปกติ
- สถานะการทำงานผิดปกติ (ระบบวัดอุณหภูมิไม่สามารถเชื่อมต่อกับ SCADA Software)

4.2.1 การทดสอบกรณีสถานะการทำงานปกติ

สำหรับการทดสอบนี้จะทำการวัดอุณหภูมิจากแพลนต์ควบคุมอุณหภูมิโดยระบบที่พัฒนาขึ้น แล้วทำเก็บข้อมูลและแสดงผลผ่านกราฟบนหน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software ซึ่งผลการทดสอบที่ได้สามารถแสดงได้ดังรูปที่ 4.4

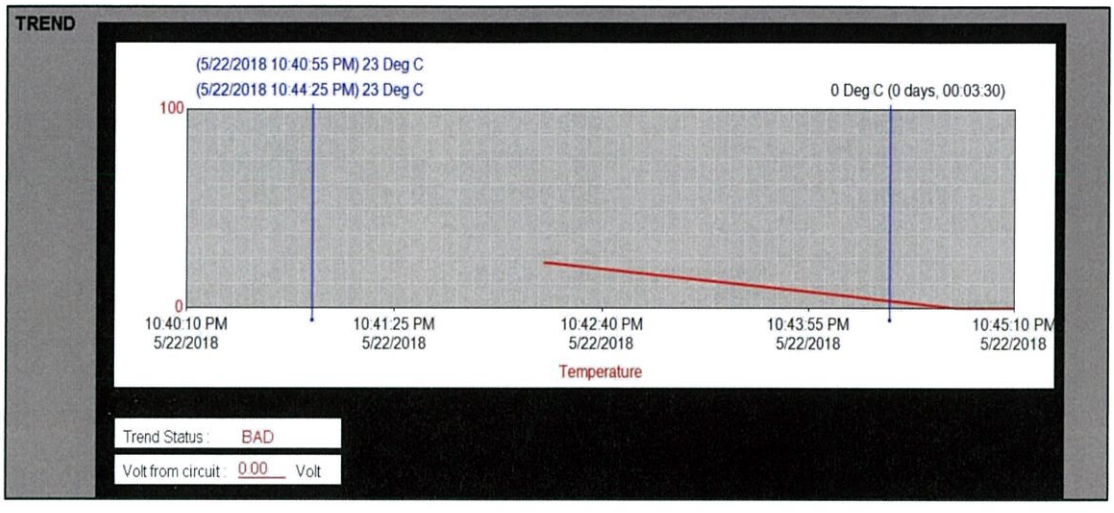


รูปที่ 4.4 การแสดงผลบนหน้าจอแสดงผลกรณีสถานะการทำงานปกติ

จากรูปที่ 4.4 ในกรณีที่ระบบทำงานปกติ ระบบจะแสดงสถานะเป็น “Good” ให้ผู้ใช้รับทราบว่าขณะนี้มีการส่งข้อมูลของอุณหภูมิที่วัดได้เข้ามาอย่างต่อเนื่อง

4.2.2 การทดสอบกรณีสถานะการทำงานผิดปกติ (ระบบวัดอุณหภูมิไม่สามารถเชื่อมต่อกับ SCADA Software)

สำหรับการทดสอบนี้จะทำการวัดอุณหภูมิจากแพลนต์ควบคุมอุณหภูมิโดยระบบที่พัฒนาขึ้น แล้วทำเก็บข้อมูลและแสดงผลผ่านกราฟบนหน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software แต่จะทำปิดเน็ตเวิร์คสวิตช์เพื่อตัดสัญญาณการเชื่อมต่อ WIFI ของระบบทิ้ง ซึ่งผลการทดสอบที่ได้สามารถแสดงได้ดังรูปที่ 4.5



รูปที่ 4.5 ผลการทดสอบกรณีสถานะการทำงานผิดปกติ (ระบบวัดอุณหภูมิไม่สามารถเชื่อมต่อกับ SCADA Software)

จากรูปที่ 4.5 เมื่อเกิดความผิดปกติกับการทำงานของระบบ เช่น ระบบวัดอุณหภูมิไม่สามารถเชื่อมต่อกับ SCADA Software ระบบแสดงผลจะแสดงสถานะ “Bad” ให้ผู้ใช้รับทราบว่าขณะนี้ไม่มีการส่งข้อมูลของอุณหภูมิที่วัดได้เข้ามา

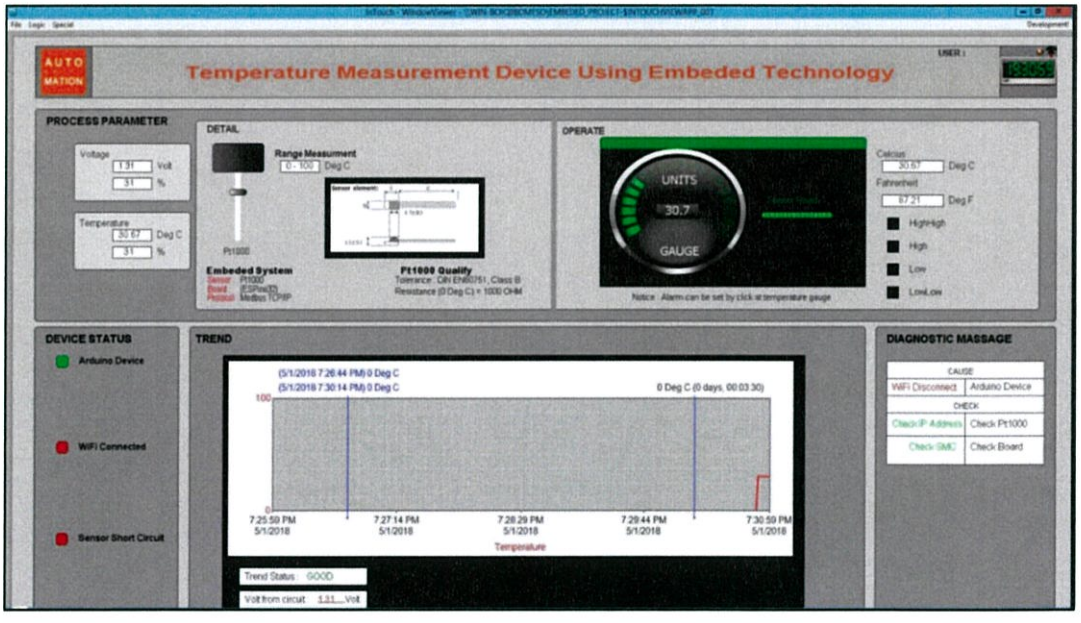
4.3 การทดสอบการทำงานของส่วนแสดงผลสถานะการทำงานของระบบผ่านทาง SCADA Software

สำหรับในการทดสอบนี้จะทำการทดสอบการทำงานของส่วนแสดงผลสถานะการทำงานระบบผ่านทาง SCADA Software ของระบบวัดอุณหภูมิแบบไร้สายโดยใช้อุปกรณ์ฝังตัวที่สร้างขึ้นมา โดยการทดสอบจะแบ่งออกเป็น 3 กรณีคือ

- สถานะการทำงานของระบบปกติ
- สถานะการทำงานผิดปกติ (ระบบวัดอุณหภูมิไม่สามารถเชื่อมต่อกับ SCADA Software)
- สถานะการทำงานผิดปกติ (เซนเซอร์ PT1000 ลัดวงจร)

4.3.1 การทดสอบกรณีสถานะการทำงานปกติ

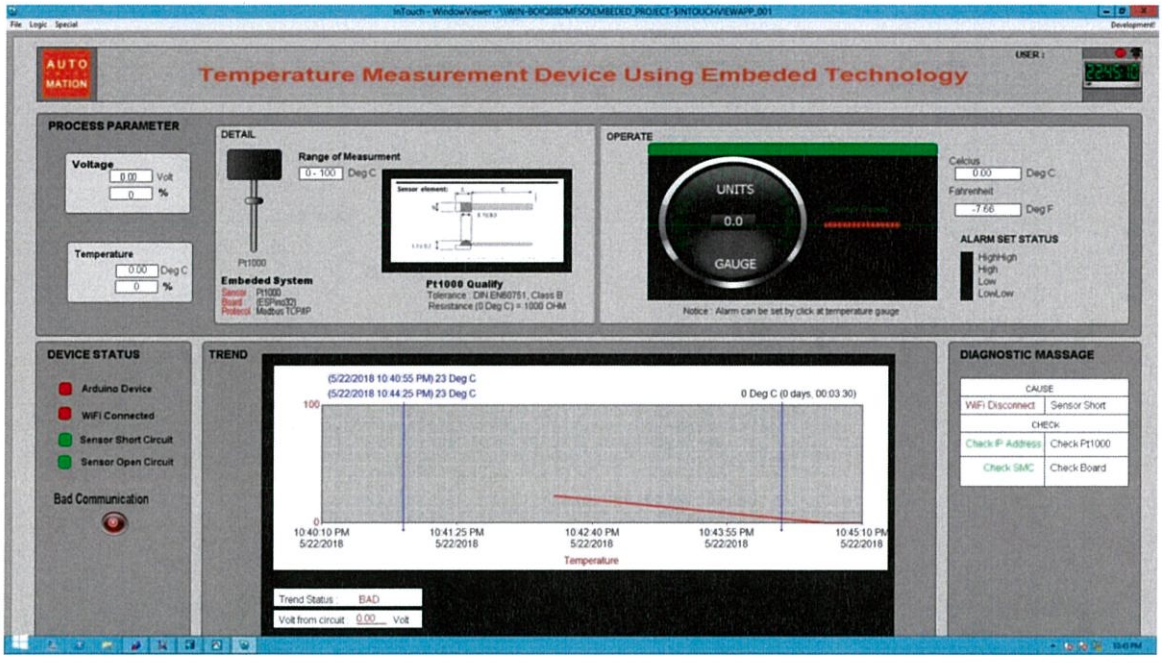
สำหรับการทดสอบนี้จะทำการวัดอุณหภูมิจากแพลนต์ควบคุมอุณหภูมิโดยระบบที่พัฒนาขึ้น แล้วทำการบันทึกหน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software ซึ่งผลการทดสอบที่ได้สามารถแสดงได้ดังรูปที่ 4.6



รูปที่ 4.6 หน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software กรณีสถานะการทำงานปกติ

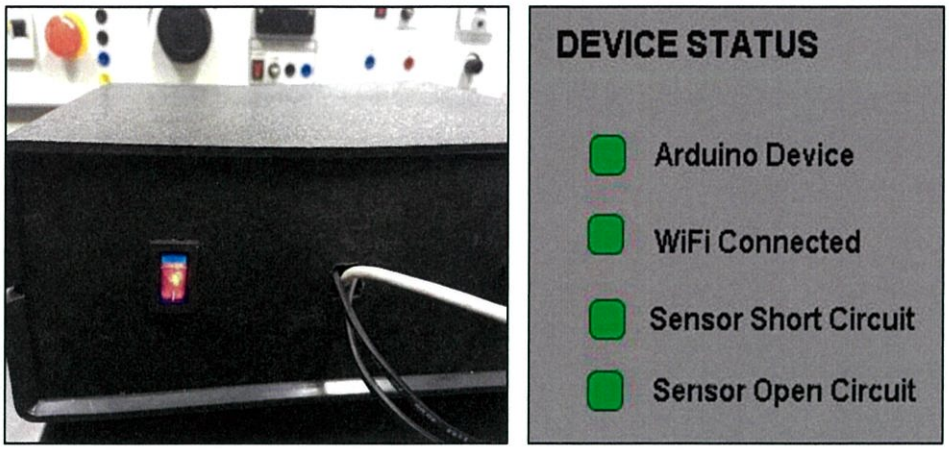
4.3.2 สถานะการทำงานผิดปกติ (ระบบวัดอุณหภูมิไม่สามารถเชื่อมต่อกับ SCADA Software)

สำหรับการทดสอบนี้จะทำการวัดอุณหภูมิจากแพลนต์ควบคุมอุณหภูมิโดยระบบที่พัฒนาขึ้น แล้วทำการบันทึกหน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software แต่จะทำปิดเน็ตเวิร์คสวิตช์เพื่อตัดสัญญาณการเชื่อมต่อ WIFI ของระบบทิ้ง ซึ่งผลการทดสอบที่ได้สามารถแสดงได้ดังรูปที่ 4.7

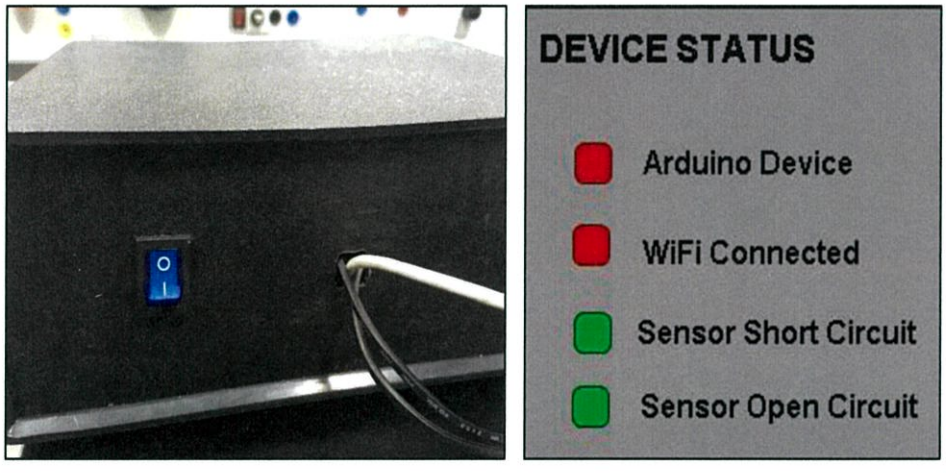


รูปที่ 4.7 หน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software กรณีสถานะการทำงานผิดปกติ (ระบบวัดอุณหภูมิไม่สามารถเชื่อมต่อกับ SCADA Software)

จากรูปที่ 4.7 หน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software กรณีสถานะการทำงานผิดปกติ (ระบบวัดอุณหภูมิไม่สามารถเชื่อมต่อกับ SCADA Software) จะแสดงสถานะด้วยไฟแสดงผลสีแดงที่ Arduino device และ WIFI Connected เพื่อแจ้งให้ผู้ใช้งานทราบ โดยในการทดสอบเพิ่มเติมโดยการเปิดเน็ตเวิร์คสวิตช์ให้สัญญาณการเชื่อมต่อ WIFI ปกติแล้วทำการปิด-เปิดระบบวัดอุณหภูมิแทนจะได้ผลการทดสอบเหมือนกันดังแสดงในรูป 4.8 และ รูปที่ 4.9

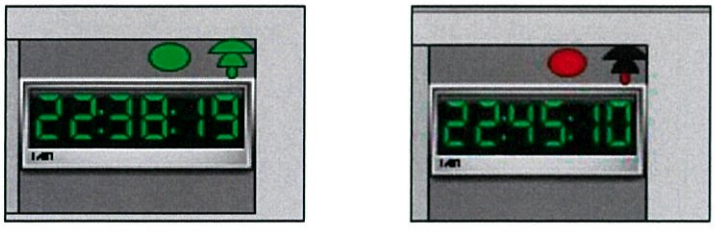


รูปที่ 4.8 สถานะการทำงานปกติ (เปิดระบบวัดอุณหภูมิ)

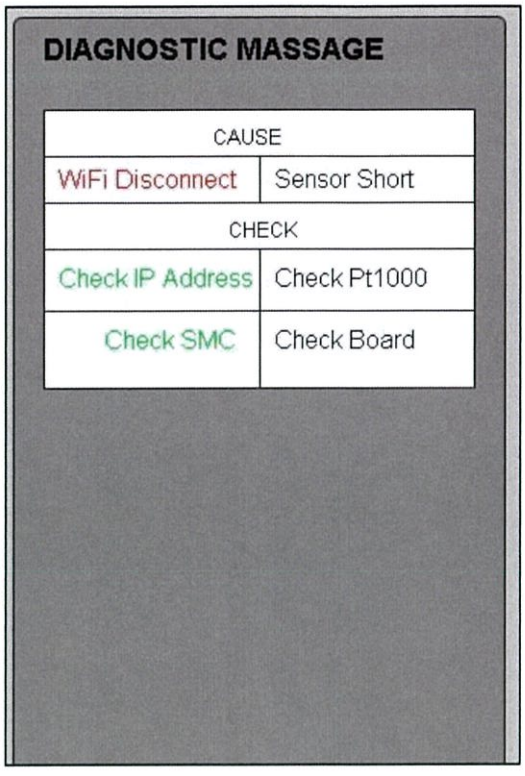


รูปที่ 4.9 สถานะการทำงานผิดปกติ (ปิดระบบวัดอุณหภูมิ)

นอกจากนี้ผู้ใช้งานยังสามารถโดยสามารถดูสถานะการเชื่อมต่อได้จาก Quick Bar ที่มุมซ้ายบนของหน้าจอแสดงผล เพื่อดูการเชื่อมต่อ WIFI ว่าปกติดีหรือไม่ดังแสดงในรูปที่ 4.10 พร้อมทั้งข้อแนะนำสำหรับผู้ใช้งานในการตรวจสอบระบบดังแสดงในรูปที่ 4.11



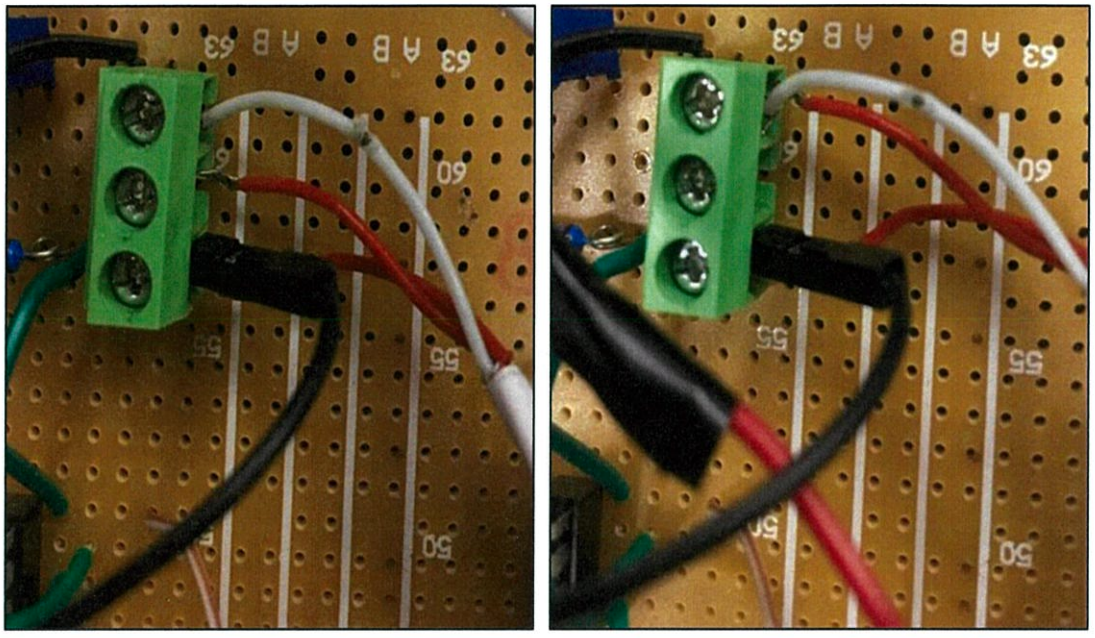
รูปที่ 4.10 Quick Bar เพื่อบอกสถานะ WIFI



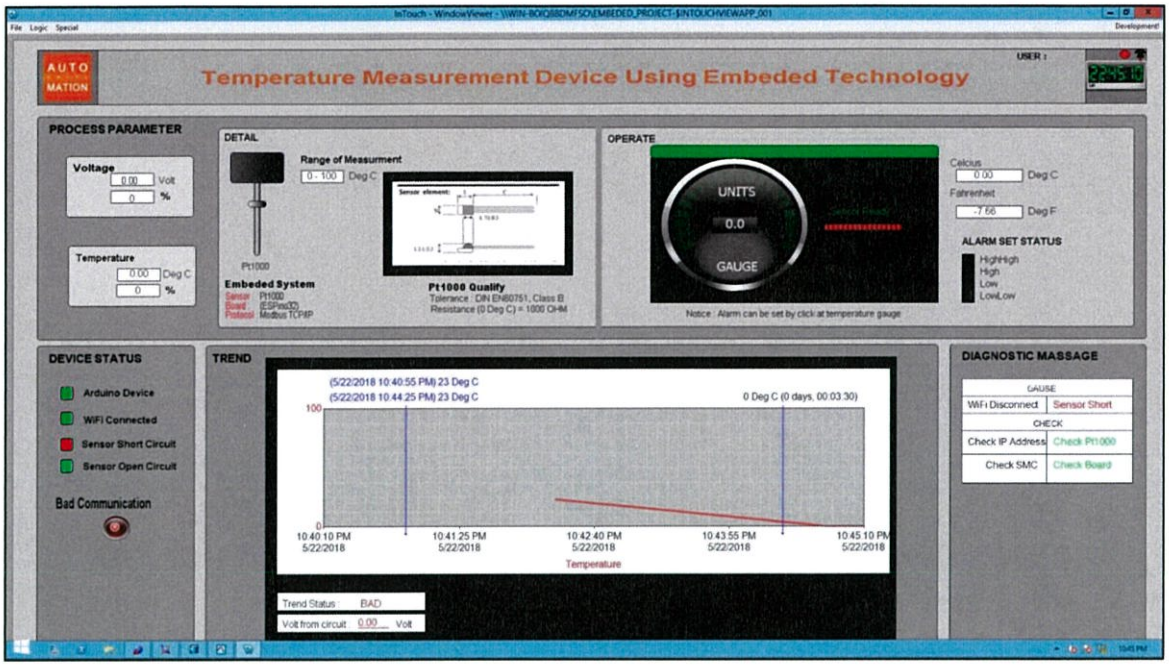
รูปที่ 4.11 Diagnostic Massage แสดงข้อแนะนำสำหรับผู้ใช้งานในการตรวจสอบระบบ

4.3.3 การทดสอบกรณีสถานะการทำงานผิดปกติ (เซนเซอร์ PT1000 ลัดวงจร)

สำหรับการทดสอบนี้จะทำการวัดอุณหภูมิจากแพลนต์ควบคุมอุณหภูมิโดยระบบที่พัฒนาขึ้น แล้วทำการบันทึกหน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software แต่จะทำต่อลัดวงจรที่เซนเซอร์ PT1000 ดังแสดงในรูปที่ 4.12 โดยผลการทดสอบที่ได้สามารถแสดงได้ดังรูปที่ 4.13

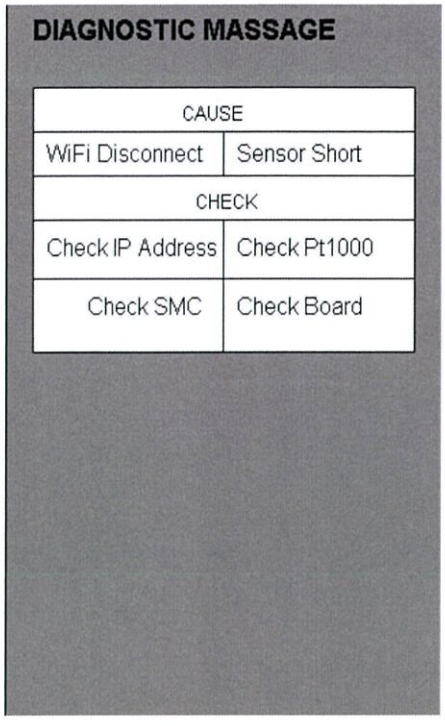


รูปที่ 4.12 การต่อลัดวงจรที่เซนเซอร์ PT1000

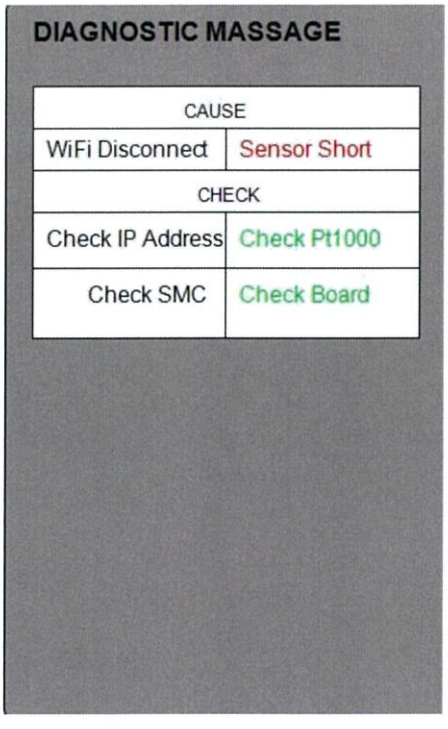


รูปที่ 4.13 หน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software กรณีสถานะการทำงานผิดปกติ (เซนเซอร์ PT1000 ลัดวงจร)

จากรูปที่ 4.13 หน้าจอแสดงผลที่พัฒนาขึ้นมาโดย SCADA Software กรณีสถานะการทำงานผิดปกติ (เซนเซอร์ PT1000 ลัดวงจร) จะแสดงสถานะด้วยไฟแสดงผลสีแดงที่ Sensor short circuit เพื่อแจ้งให้ผู้ใช้งานทราบ พร้อมทั้งข้อแนะนำสำหรับผู้ใช้งานในการตรวจสอบระบบดังแสดงในรูปที่ 4.14 และรูปที่ 4.15 ตามลำดับ



รูปที่ 4.14 Diagnostic Massage กรณีปกติ



รูปที่ 4.15 Diagnostic Massage กรณี Sensor Short

บทที่ 5
บทสรุปและข้อเสนอแนะ

5.1 บทสรุป

จากการดำเนินงานพบว่าระบบสมองกลฝังตัวสามารถนำมาประยุกต์ใช้กับอุตสาหกรรมได้ โดยความสามารถของตัวอุปกรณ์วัดจะเพิ่มขึ้นเท่าใด ขึ้นอยู่กับความสามารถในการเขียนโปรแกรมลงบนบอร์ด Arduino เนื่องจากในโครงงานมีการทำวงจรทั้งหมดขึ้นมาเอง ซึ่งค่าความต้านทานในวงจรไม่สามารถปรับให้ตรงที่สุดตามที่คำนวณได้ ทำให้เกิดความไม่แม่นยำในการวัดเกิดขึ้น (เปรียบเทียบกับอุปกรณ์วัดแบบดิจิตอล Yokogawa 7563 digital thermometer) โดยมีค่าความผิดพลาดสูงสุดประมาณ -2.35% รวมไปถึงตัวบอร์ด Arduino ที่การส่งค่าจะมีการส่งค่าตัวแปร และมีการส่งค่า NULL (ไม่ส่งค่า) สลับกันอยู่บ่อย จึงถือเป็นข้อผิดพลาดทางฮาร์ดแวร์ที่แก้ไขได้ยาก ทำให้การนำไปประยุกต์ใช้ยังต้องพัฒนาด้านวงจรให้สมบูรณ์มากขึ้น และใช้บอร์ดไมโครคอนโทรลเลอร์ ที่มีประสิทธิภาพมากขึ้น

5.2 ปัญหาและอุปสรรค

5.2.1 การจะเขียนโปรแกรมลงบอร์ด Arduino จำเป็นต้องมี Components และ Libraries เยอะ และคอมพิวเตอร์บางเครื่องไม่สามารถติดตั้งส่วนประกอบเหล่านี้ได้

5.2.2 การปรับรอบขยายของวงจรแบ่งกระแสภายในยังไม่แม่นยำมากพอเนื่องจากผลของความต้านทานแบบปรับค่า

5.2.3 บอร์ดที่ใช้มีปัญหาบ่อยครั้ง ไม่มีเสถียรภาพในการรับ-ส่งข้อมูล

5.3 ข้อเสนอแนะ

5.3.1 เลือกใช้บอร์ดที่เสถียรภาพในการรับ-ส่งข้อมูลสูง

5.3.2 ควรใช้อุปกรณ์ที่เหมาะสมและแม่นยำที่สุดเพื่อลดความคลาดเคลื่อนของวงจรขยายสัญญาณ

5.3.3 ผู้สนใจโครงงานนี้ควรศึกษาเรื่องการเขียนโปรแกรมเพิ่มเติม เพื่อเพิ่มความสามารถให้ระบบฝังตัวได้

เอกสารอ้างอิง

[1] รู้จัก Embedded Systems

แหล่งที่มา : http://www.namwalab.com/2014/05/08/%E0%B8%

[2] บทความ Arduino คืออะไร? ตอนที่1 แนะนำเพื่อนใหม่ที่ชื่อ Arduino

แหล่งที่มา : http://www.thaieasyelec.com/article-wiki/basic-electronics/%E0%B8%A1-

[3] ทำความรู้จัก Raspberry Pi

แหล่งที่มา : http://www2.crma.ac.th/itd/Know/RBPI/index.asp

[4] PLC Protocol: การสื่อสารแบบ Modbus Protocol

แหล่งที่มา : https://riverplusblog.com/2011/08/18/plc-protocol% A-modbus-protocol/

[5] OPC System

แหล่งที่มา : http://automationreview.blogspot.com/ole-for-process-control-opc.html