ระบบควบคุมระดับน้ำแบบไร้สาย
WIRELESS LEVEL CONTROL SYSTEM

วรธัน ธรรมสุนทรี

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต
สาขาวิชาวิศวกรรมระบบควบคุม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2559

ระบบควบคุมระดับน้ำแบบไร้สาย

WIRELESS LEVEL CONTROL SYSTEM

วรธัน ธรรมสุนทรี

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมระบบควบคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2559

WIRELESS LEVEL CONTROL SYSTEM

WORRATHUN THAMSUNTREE

THIS THESIS IS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
BACHELOR OF ENGINEERING IN CONTROL ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2016

ปริญญานิพนธ์ปีการศึกษา 2559

ภาควิชาวิศวกรรมการวัดและควบคุม คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบควบคุมระดับน้ำแบบไร้สาย

WIRELESS LEVEL CONTROL SYSTEM

ผู้จัดทำ นายวรธัน ธรรมสุนทรี 55011333

.....อาจารย์ที่ปรึกษา

(ศาสตราจารย์ ดร.วันชัย ริ้วรุจา)

.....อาจารย์ที่ปรึกษา

(ผู้ช่วยศาสตราจารย์ ดร.วรรณดี เพชรมณีล้ำค่า)

.....อาจารย์ที่ปรึกษา

(ดร.สิริชัย ธรรมารักษ์วัฒนะ)

ระบบควบคุมระดับน้ำแบบไร้สาย

โดย

นายวรธัน ธรรมสุนทรี 55011333

อาจารย์ที่ปรึกษา

ศาสตราจารย์ ดร.วันชัย ริ้วรุจา

ผู้ช่วยศาสตราจารย์ ดร.วรรณดี เพชรมณีล้ำค่า

ดร.สิริชัย ธรรมารักษ์วัฒนะ

ปีการศึกษา 2559

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้เป็นการออกแบบ และดำเนินการสร้างโครงงานขนาดเล็ก เพื่อศึกษาและจำลองระบบควบคุมแบบปิดที่ใช้กันในอุตสาหกรรม โดยในระบบจะประกอบไปด้วย Pressure Sensor Flow Sensor และ Motor ซึ่งใช้วัดค่าแรงดัน อัตราการไหล และกระตุ้นการขับเคลื่อนของน้ำ ตามลำดับ จากนั้นจะส่งสัญญาณไปที่ PLC (Programmable Logic Controller) ซึ่งเป็นตัวควบคุมระดับน้ำภายในถัง ให้มีค่าตรงตามค่าเป้าหมายที่ต้องการ โดยจะใช้ตัวควบคุมแบบ PID ที่มีความนิยม และยอมรับกันในการควบคุมระบบอุตสาหกรรม เนื่องจากมีความแม่นยำสูงในการป้อนข้อมูลต่างๆ ตามค่าเป้าหมายที่ต้องการให้แก่ระบบ

นอกจากนี้ยังได้นำเสนอการเชื่อมต่อระหว่างอุปกรณ์ต่างๆ ในโรงงานอุตสาหกรรมด้วยการรับ – ส่งข้อมูลแบบไร้สายผ่านทางโมดูล Wi-Fi ซึ่งเป็นประโยชน์อย่างมากสำหรับโรงงานอุตสาหกรรมขนาดใหญ่ที่มีการเชื่อมต่อระหว่างอุปกรณ์แบบใช้สาย

WIRELESS LEVEL CONTROL SYSTEM

By

Worrathun Thamsuntree 55011333

Advisors

Prof. Dr. Vanchai Riewruja

Asst. Prof. Dr. Wandee Petchmaneelumka

Dr. Sirichai Thammarakwattana

Academic Year 2016

ABSTRACT

This project presents a design and implementation of a small plant for study and simulation of closed-loop control systems in the industrial work. Transducers used in the model plant consist of two sensors and one motor functioned as flow and level transmitters and actuator, respectively. PLC is used as a controller to control the level of water in the tank to the point where users need. Using PID controller, which is used commonly in industrial control systems, is very precise to control the input of a system to the desired value. Furthermore, the elements in the entire proposed plant are connected to each other without wiring which is very useful in a big plant using wired system.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ สำเร็จลุล่วงไปได้ด้วยการให้ความช่วยเหลือ และคำแนะนำของ ผู้ช่วยศาสตราจารย์ ดร.วรรณดี เพชรมณีล้ำค่า ที่ได้ให้คำปรึกษาเป็นอย่างดีมาตลอด รวมทั้งยังเอื้อเฟื้ออุปกรณ์สำหรับการทำโครงงาน และความช่วยเหลืออื่นๆ ที่เป็นประโยชน์ต่อโครงงาน ทางผู้จัดทำจึงขอขอบพระคุณไว้ ณ โอกาสนี้

ขอขอบพระคุณศาสตราจารย์ ดร.วันชัย ริ้วรุจา และ ดร.สิริชัย ธรรมารักษ์วัฒนะ ที่เป็นอาจารย์ที่ปรึกษาโครงงานร่วม และให้คำแนะนำที่ดีมาโดยตลอด

ขอบคุณพี่ เพื่อน และน้องภาควิชาทุกคนที่ถามไถ่ ให้กำลังใจด้วยความห่วงใยในการทำโครงงาน และให้ความช่วยเหลือเมื่อเกิดปัญหา ช่วยสนับสนุนอุปกรณ์ที่ขาดเหลืออยู่เสมอ

ท้ายนี้ผู้จัดทำขอขอบพระคุณบิดา มารดา และครอบครัว ที่เป็นกำลังใจที่ดีตลอดมา สนับสนุนเรื่องของงบประมาณในการทำโครงงาน และยังสร้างแรงบันดาลใจที่ดีที่สุดในการทำโครงงานนี้ จนสามารถสำเร็จลุล่วง และเสร็จสมบูรณ์ลงได้

ผู้จัดทำ

นายวรธัน ธรรมสุนทรี

สารบัญ

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญ (ต่อ)

สารบัญรูป

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญรูป (ต่อ)

สารบัญตาราง

บทที่ 1

บทนำ

1.1 ความเป็นมาของโครงงาน

ปัจจุบันการติดต่อสื่อสาร และการส่งข้อมูลต่างๆ ผ่านทางเทคโนโลยีการสื่อสารแบบไร้สายเป็นที่ยอมรับ และมีความนิยมอย่างมาก เนื่องจากมีความสะดวก รวดเร็ว ระบบเครือข่ายไม่ยุ่งยากซับซ้อนต่อการส่งผ่านข้อมูลโดยไม่ต้องเคลื่อนย้ายอุปกรณ์ ทางผู้จัดทำจึงได้นำเอาการสื่อสารผ่านทางโมดูล Wi-Fi มาประยุกต์ใช้กับทางด้านอุตสาหกรรม โดยการนำ Wi-Fi Module มาเป็นตัวกลางส่งผ่านข้อมูล ระหว่าง PLC (Programmable Logic Controller) กับอุปกรณ์ต่างๆในระบบ เช่น เซนเซอร์ และมอเตอร์ ซึ่งจะช่วยลดค่าใช้จ่ายในการวางสาย LAN และง่ายต่อการซ่อมบำรุง

เนื่องจากอุปกรณ์ Client Wi-Fi ซึ่งเป็น Wi-Fi สำเร็จรูป ที่ใช้กันอยู่ทั่วไปในการควบคุมระบบนั้นมีราคาค่อนข้างสูง ทางคณะผู้จัดทำจึงได้นำเสนอระบบควบคุมระดับน้ำแบบไร้สาย (Wireless Level Control System) ในราคาต้นทุนที่ต่ำกว่า โดยจำลองตัวอย่างวิธีการใช้งาน Wi-Fi Module มาเป็นตัวกลางในการรับ-ส่งข้อมูล เพื่อควบคุมระบบโดยใช้ PLC ซึ่งจะรับข้อมูลจาก Pressure Sensor และ Flow Sensor มาแสดงผลที่โปรแกรม TIA Portal V13 เพื่อทำการควบคุมมอเตอร์ของปั๊มน้ำ ให้ได้ระดับน้ำภายในถังตามที่ต้องการ

1.2 วัตถุประสงค์ในการทำปริญญานิพนธ์

1. เพื่อออกแบบ และสร้างแบบจำลองระบบควบคุมระดับน้ำอัตโนมัติแบบไร้สายโดยใช้โปรแกรม SolidWorks
2. เพื่อศึกษาการใช้งานโปรแกรม STEP7 TIA Portal V13 และการเขียนคำสั่งในการควบคุมระบบ
3. เพื่อศึกษาการทำงานของตัวควบคุมแบบพีไอดี
4. เพื่อสร้างหน้าจอแสดงผลสำหรับติดตามค่า และควบคุมระบบสำหรับผู้ใช้งาน (Human Interface: HMI) ผ่านโปรแกรม WinCC Runtime Advance V13
5. เพื่อศึกษาการทำงานของไมโครคอนโทรลเลอร์ (Microcontroller) และการรับส่งข้อมูลแบบไร้สาย ผ่าน Wi-Fi Module ระหว่าง PLC กับระบบจำลองเพื่อควบคุมระดับน้ำ
6. เพื่อศึกษาการเขียนโปรแกรมอาดูโน (Arduino)
7. เพื่อศึกษาการทำงาน และการนำมาใช้งานของ Pressure Sensor, Flow Sensor และ Actuator

1.3 ขอบเขตของโครงงาน

1. สร้างระบบจำลองการควบคุมระดับน้ำแบบไร้สายที่ควบคุมระบบผ่านทาง W-Fi Module ได้สำเร็จ และสามารถใช้งานได้จริง

2. เขียนโปรแกรมในการสั่งการพีแอลซี Siemens รุ่น S7-1200 เพื่อควบคุมระดับน้ำในระบบควบคุมระดับน้ำอัตโนมัติแบบไร้สาย

3. สามารถนำ Pressure Sensor, Flow Sensor และ Actuator มาประยุกต์ใช้งานกับระบบควบคุมระดับน้ำแบบไร้สายได้

4. สามารถเขียนโปรแกรมเพื่อให้สามารถรับ-ส่งข้อมูลผ่านทาง Wi-Fi Module ได้

1.4 ขั้นตอนการศึกษา

1. ศึกษาข้อมูลและทฤษฎีที่เกี่ยวข้องกับการทำงานของระบบทั้งหมด ได้แก่ หลักการทำงานของไมโครคอนโทรลเลอร์, การเขียนโปรแกรมอาดูโน, โปรแกรม TIA Portal V13, การรับส่งข้อมูลผ่าน Wi-Fi Module ESP 8266, การทำงานของเซนเซอร์ตลอดจนการออกแบบโครงสร้างของระบบจำลอง ผ่านโปรแกรม SolidWorks

2. วางแผนการทำงาน

3. ออกแบบระบบ วงจร และโครงสร้างของระบบควบคุมระดับน้ำแบบไร้สาย

4. ศึกษาส่วนประกอบ และอุปกรณ์ที่ต้องใช้ในการทำงาน

5. สร้างระบบวงจร, ติดตั้งอุปกรณ์ และเขียนโปรแกรม

6. ทดลอง และบันทึกข้อมูลการทำงาน

7. สรุปผลการทำงาน

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถนำระบบนี้ไปเป็นแนวทางในการประยุกต์ใช้กับอุตสาหกรรมได้ในอนาคต

2. สามารถเขียนโปรแกรมควบคุมการทำงานของ PLC และการเขียนคำสั่งควบคุมระบบผ่าน Wi-Fi Module

3. สามารถสร้างระบบควบคุมระดับน้ำอัตโนมัติแบบไร้สายได้

1.6 รายละเอียดของปริญญานิพนธ์

เนื้อหาที่จะกล่าวในปริญญานิพนธ์ฉบับนี้ประกอบด้วย 5 บท กับอีก 1 ภาคผนวก ดังนี้

บทที่ 1 บทนำ เป็นการกล่าวถึงความเป็นมาของโครงงาน วัตถุประสงค์ของการทำโครงงาน ขอบเขตของโครงงาน ขั้นตอนการศึกษาและจัดทำโครงงาน ประโยชน์ที่คาดว่าจะได้รับและรายละเอียดของปริญญานิพนธ์

บทที่ 2 ทฤษฎี หลักการ และความรู้ที่เกี่ยวข้องกับโครงงานนี้

บทที่ 3 ขั้นตอนการดำเนินงาน

บทที่ 4 ผลการทดลอง

บทที่ 5 บทวิจารณ์และสรุป สรุปการดำเนินการ ปัญหา และแนวทางการปรับปรุงโครงงาน

ภาคผนวก ก โปสเตอร์

บทที่ 2

ทฤษฎีและข้อมูลที่เกี่ยวข้อง

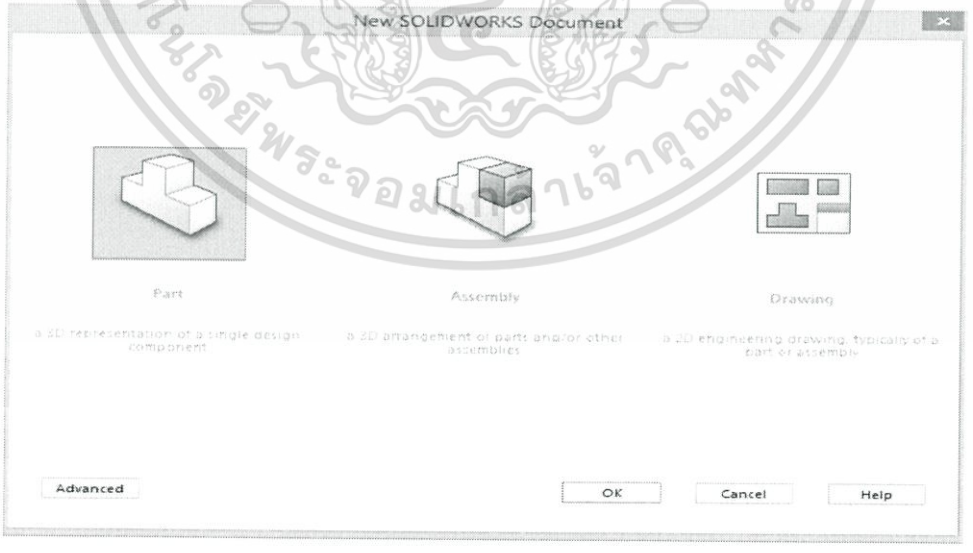
2.1 โปรแกรม SolidWorks

SolidWorks พัฒนาขึ้นในปี 1995 โดยบริษัท Dassault System ในฝรั่งเศสเป็นซอฟต์แวร์เพื่อให้นักออกแบบใช้เป็นเครื่องมือในการออกแบบทางวิศวกรรม เพื่อสร้างตัวอย่างผลิตภัณฑ์จำลองในคอมพิวเตอร์ ก่อนที่จะสร้างผลิตภัณฑ์ต้นแบบจริง โดยตัวซอฟต์แวร์จะจัดอยู่ในตระกูล CAD (Computer Aided Design) ซึ่งสามารถสร้างชิ้นงานจำลองในรูปแบบ 3D Solid Models เป็นแบบงานแยกชิ้น (Part) และแบบงานประกอบ (Assembly) เพื่อนำไปสร้างเป็น 2D Standard Engineering (CADD = Computer Aided Design and Drafting)

โปรแกรม SolidWorks เป็นโปรแกรมที่มีความยืดหยุ่นในการทำงานสูงมากคือ สามารถที่จะทำงานได้มากมายหลายรูปแบบ ไม่ว่าจะเป็นชิ้นงานที่ต้องขึ้นเป็น Solid หรือ Surface ก็มีเครื่องที่รองรับเป็นอย่างดี เมื่อสร้างชิ้นงานเสร็จเรียบร้อยสามารถที่จะประกอบชิ้นงานได้ใน Mode ของชุดคำสั่ง Assembly รวมทั้งผู้ที่ต้องการ Drawing ของชิ้นงาน ก็เพียงลากชิ้นงานมาวางในใบงาน

ลักษณะการทำงานของโปรแกรม SolidWorks

SolidWorks แบ่งหมวดการทำงานหลักออกเป็น 3 หมวดคือ Part, Assembly และ Drawing โดยรูปแบบการทำงานทั้ง 3 หมวดมีลักษณะการใช้งานดังนี้



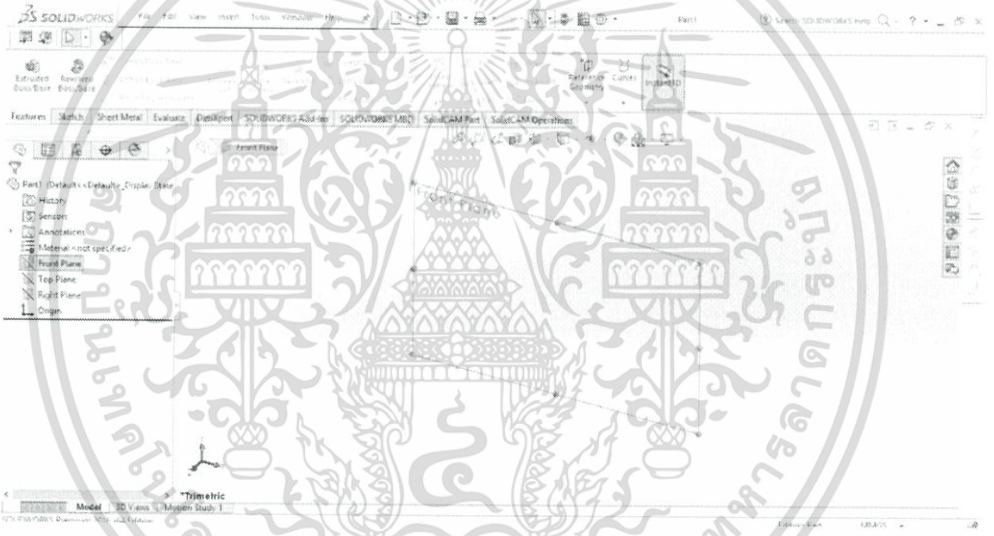
รูปที่ 2.1 รูปแบบการทำงานของโปรแกรม SolidWorks

Part Mode เป็นหมวดการทำงานเริ่มต้นก่อนที่จะก้าวสู่การทำงานในหมวด Assembly และ Drawing ในขั้นนี้จะมีการแบ่งการทำงานออกเป็น 2 ส่วน คือ การใช้ 2D Sketch เพื่อนำไปสู่การสร้างเป็น 3D Feature และมีเงื่อนไขเป็น Feature-Based Modeling และ Parametric โดยมีการอ้างอิงจาก Solid Mode

Feature-Based Modeling คือ การออกแบบซอฟต์แวร์ให้สามารถทราบถึงคุณสมบัติต่างๆของ Solid Model ที่สร้างขึ้นมา เพื่อให้ผู้ใช้งานสามารถเปลี่ยนแปลงและแก้ไข Model ในลำดับการทำงานแต่ละขั้นได้ง่ายและรวดเร็ว

Parametric Model คือ การออกแบบซอฟต์แวร์ซึ่งใช้เงื่อนไขทางคณิตศาสตร์ในการแก้ไขขนาดรูปร่าง ทางเรขาคณิตของ Model ที่สร้างขึ้นมา

Solid Model คือ แบบจำลองบนคอมพิวเตอร์ที่สามารถแสดงค่าต่างๆ เช่น Density, Material, Mass, Weight เป็นต้น และยังสามารถมองเห็น 3D Model ได้ทุกมุมมอง



รูปที่ 2.2 หน้าต่างการทำงานเริ่มต้นของ Part Mode

Assembly Mode เป็นหมวดการทำงานเพื่อนำ Part Model เข้าไปประกอบเป็นเครื่องจักรกลหรือกลไกต่างๆ และมีเงื่อนไขเป็น Feature Base และ Parametric เช่นเดียวกับ Part Model โดย Part Model และ Assembly จะมีความสัมพันธ์ซึ่งกันและกัน เมื่อทำการแก้ไขในหมวดใด หรือมีการประกอบที่ซ้อนหรือทับกันอีกหมวดจะมีการเปลี่ยนแปลงตามการแก้ไขไปด้วย การทำงานใน Assembly สามารถช่วยให้นักออกแบบหรือวิศวกรสามารถตรวจสอบความผิดพลาดในการสร้าง Part ได้โดยการใช้คำสั่งต่างๆ เช่น คำสั่ง Interference Detection เพื่อตรวจสอบการขัดกันเมื่อมีการเคลื่อนที่ โดยใช้คำสั่ง Move Component เพื่อตรวจสอบการเคลื่อนที่ของกลไก คำสั่ง Simulation เพื่อจำลองต้นกำลังในการทำงานจริงของเครื่องจักร หรือหากชิ้นงานจำลองที่ออกแบบมีข้อผิดพลาด ก็สามารถแก้ไข Part ใน Assembly ได้เลย ทำให้การออกแบบเป็นเรื่องง่าย และ

ผู้ออกแบบจะสนุกกับการทำงาน Design การทำงานใน Assembly Mode มีลักษณะการทำงาน 2 กรณีได้แก่

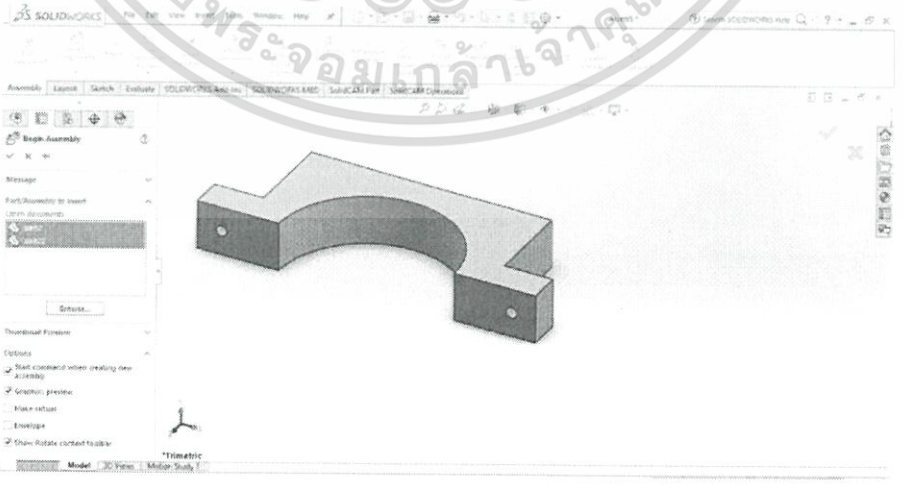
1. Bottom-Up Assembly

เป็นการนำ 3D Model ต่างๆ ที่สร้างเสร็จแล้วใน Part Mode ไปวางในหน้าต่าง Assembly เพื่อทำการประกอบ โดยการใช้คำสั่ง Mate หรือ Smart Mate ซึ่งวิธีนี้การนี้จะเหมาะสำหรับผู้ใช้ในระดับเริ่มต้นหรือขั้น Basic

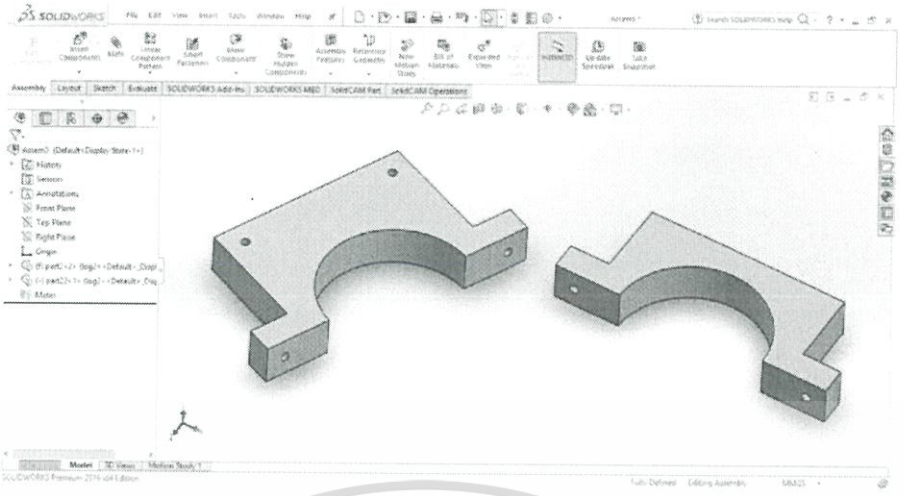
2. Top-Down Assembly

เป็นการสร้าง 2D Sketch เป็นโครงร่างระหว่างชิ้นส่วนต่างๆ ระหว่าง Part หรือการสร้าง Part ใน Assembly โดยให้มีขนาดและรูปร่างที่มีการอ้างอิงกับ Part อื่นๆ ทั้งในส่วน Sketch และ Feature วิธีนี้เหมาะกับผู้ใช้ในระดับ Advance

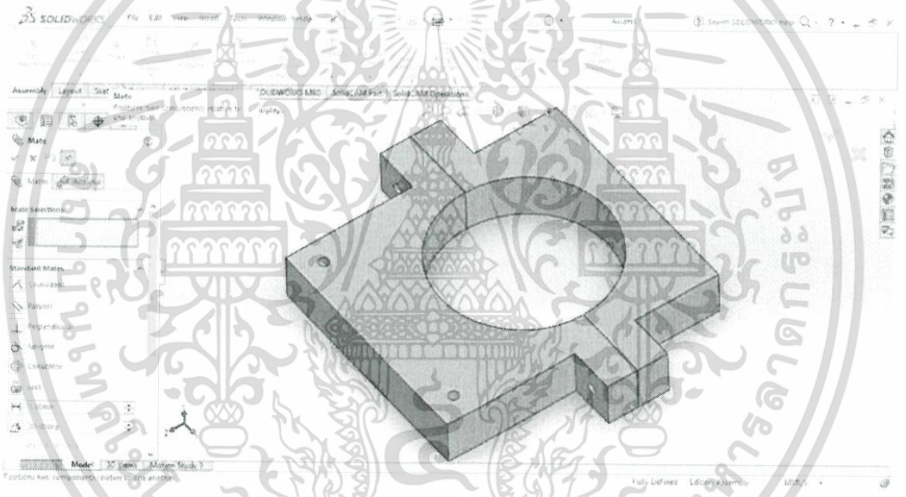
รูปที่ 2.3 หน้าต่างการทำงานเริ่มต้นของ Assembly Mode



รูปที่ 2.4 การเลือกชิ้นงานที่ต้องการประกอบโดยใช้คำสั่ง Browse



รูปที่ 2.5 ชิ้นงานที่ต้องการประกอบ



รูปที่ 2.6 การประกอบชิ้นงานโดยใช้คำสั่ง Mate

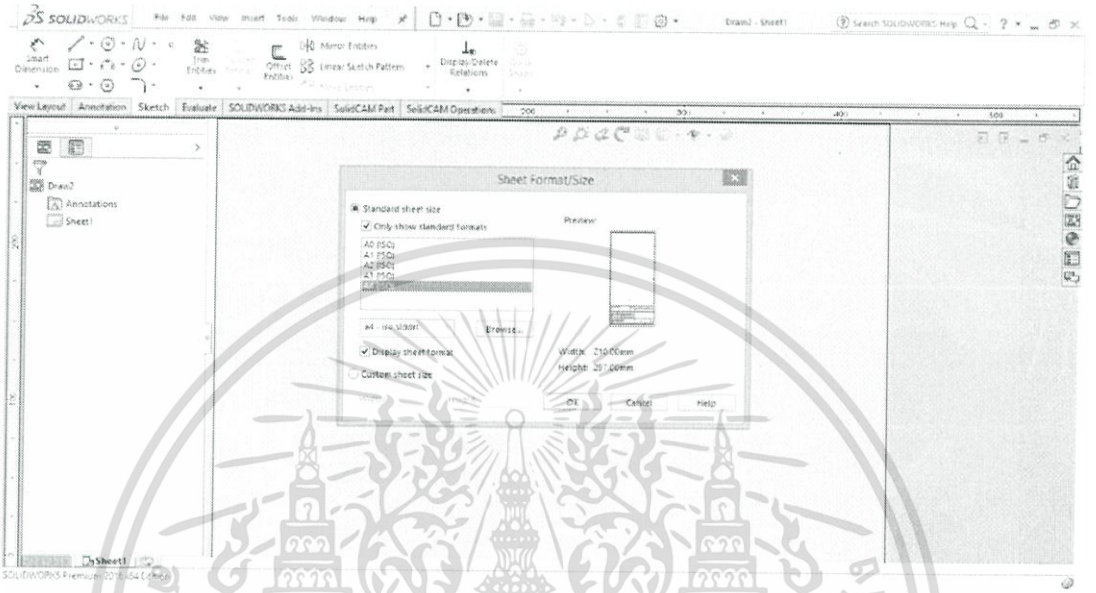
Drawing Mode เป็นหมวดการทำงานเพื่อสร้าง 2D Standard Engineering โดยในหมวดนี้เป็นการสร้างมุมมอง และกำหนดรายละเอียดตามระบบมาตรฐานต่างๆ โดยจะแบ่งการทำงานออกเป็น 2 ส่วนคือ

1. Generative Drafting

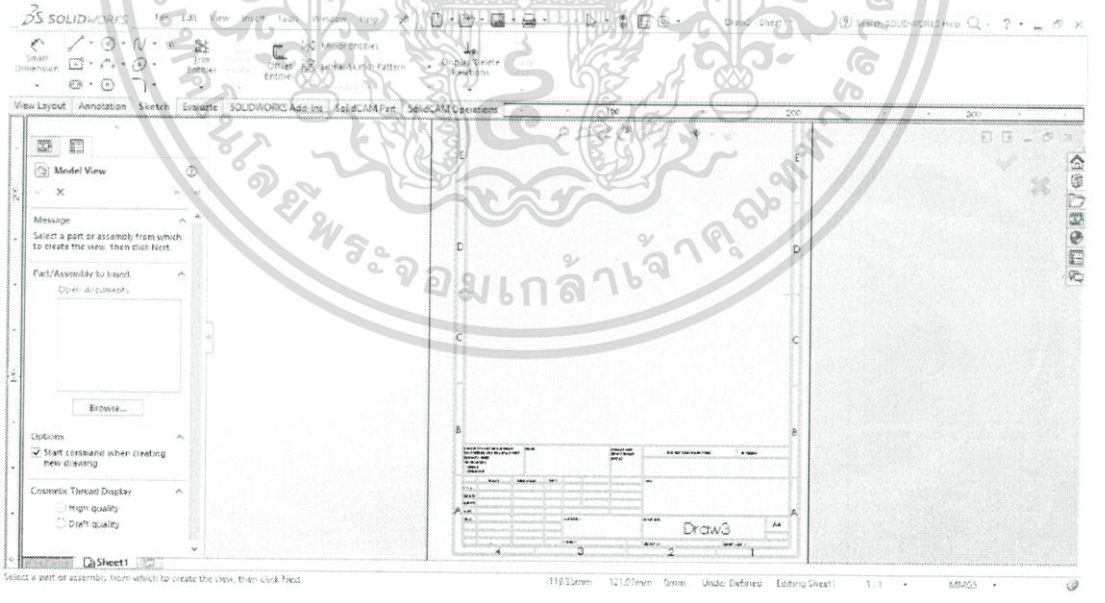
เป็นการสร้าง 2D Sketch และ Interaction Drafting ซึ่งเป็นการนำ 3D Model จาก Part และ Assembly มาวางใน Drawing เพื่อสร้างเป็น 2D Drafting จะมีลักษณะเป็น Parametric และ Relation เช่นกัน แต่จะไม่สามารถใช้คำสั่งใน Drawing Commands ได้ เพราะคำสั่งต่างๆ จะต้องอ้างอิงกับ 3D Model

2. Interaction Drafting

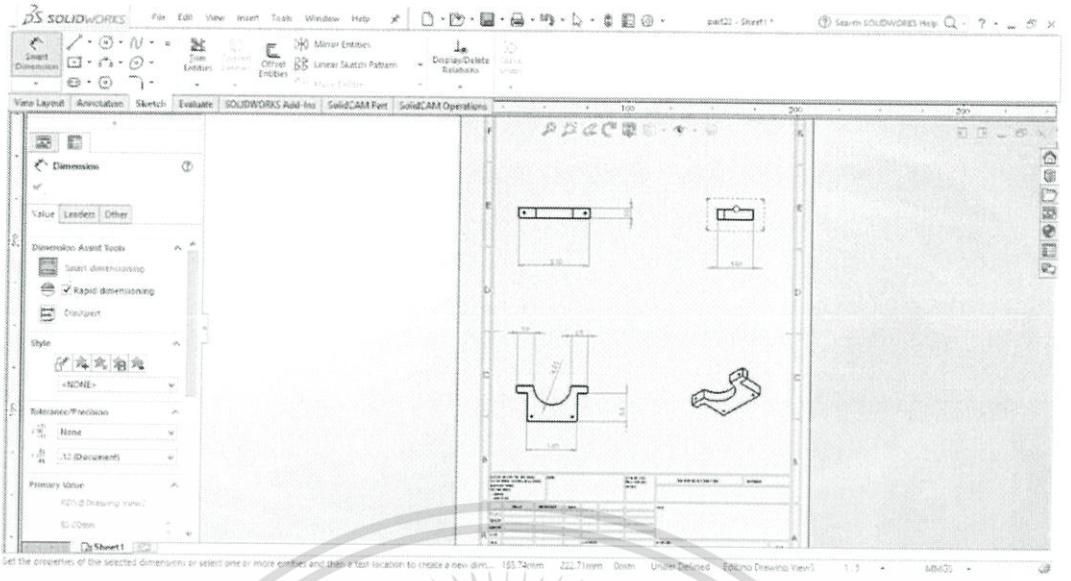
เป็นการนำ 3D Model จาก Part และ Assembly มาวาง Drawing เพื่อสร้างเป็น 2D Drafting การทำงานในหมวดนี้สามารถใช้คำสั่งจาก Annotation Command และ Drawing Command เพื่อสร้างมุมมองและกำหนดรายละเอียดได้โดยอัตโนมัติ



รูปที่ 2.7 รูปหน้าต่างการทำงานเริ่มต้นของ Drawing Mode



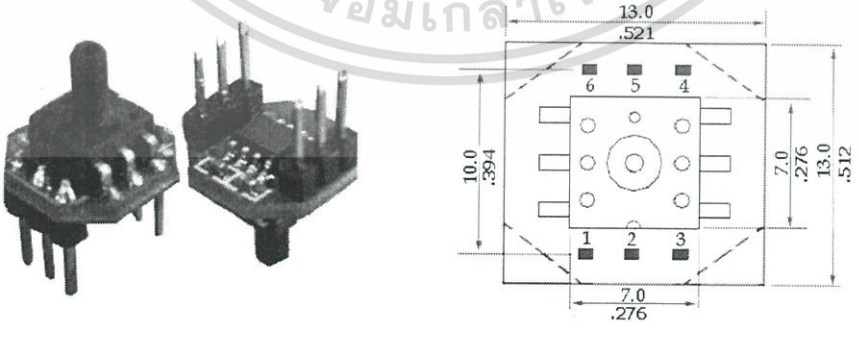
รูปที่ 2.8 การเลือกชิ้นส่วนที่ต้องการสร้าง 2D Standard Engineering โดยใช้คำสั่ง Browse



รูปที่ 2.9 การสร้าง 2D Standard Engineering และกำหนดรายละเอียดตามระบบมาตรฐาน

2.2 โมดูลเซนเซอร์วัดความดัน

XGZP6847 เป็นโมดูลเซนเซอร์วัดความดันที่มีการเชื่อมต่อแบบ Ratiometric Analog สำหรับการอ่านค่าความดันที่มากกว่าช่วงความดันที่กำหนดไว้ มีความสามารถในการวัดซ้ำ, มีความเป็นเชิงเส้น, มีเสถียรภาพ และมีการตอบสนองไว ซึ่งสามารถนำมาใช้ได้โดยตรงในอุปกรณ์ทางการแพทย์, เครื่องออกกำลังกาย, เครื่องใช้ไฟฟ้าในบ้าน และอุปกรณ์นิวเมติกอื่นๆ อีกทั้งยังสามารถนำไปประยุกต์ใช้กับอุปกรณ์อื่นๆ ได้อีกด้วย เนื่องจากมีขนาดเล็ก ราคาต่ำ และประสิทธิภาพดี



รูปที่ 2.10 โมดูล XGZP6847

ตารางที่ 2.1 พารามิเตอร์ประสิทธิภาพของโมดูล XGZP6847

Item	Data	Unit
Output	0.5-4.5	V
Accuracy	±1	%Span
Zero Temp.Coefficient	±0.03	%FS/°C
Long Term Stability	±1	%Span
Over Pressure	3X(≤500kPa)	Rated
	1.5(≥500kPk)	
Cmpensation	-20 ~ 85/-4 ~ 176	°C/°F
Operating Temp	-20 ~ 100/-4 ~212	°C/°F
Storage Temp	-40 ~ 125/-40 ~257	°C/°F

ตารางที่ 2.2 ตำแหน่งขาของโมดูล XGZP6847

1	2	3	4	5	6
N/C	Vdd	GND	Vdd	OUT	GND

2.2.1 หลักการทำงานของสเตรนเกจ

เซนเซอร์วัดความดันโดยทั่วไปใช้หลักการของสเตรนเกจ (Strain Gage) ซึ่งสเตรนเกจเป็นตัวแปลงสัญญาณที่ใช้หลักการของการเปลี่ยนแปลงค่าความต้านทานไฟฟ้าภายในเส้นลวด เพื่อการตรวจวัดความเครียดที่เกิดขึ้นจากแรงที่มากระทำบนเส้นลวดนี้ สเตรนเกจมีการใช้งานในการตรวจวัดได้อย่างกว้างขวาง เช่น การวัดน้ำหนัก ความวัด แรงเชิงกล หรือการเคลื่อนที่

สเตรนเกจสามารถแบ่งออกได้เป็น 2 แบบคือ แบบยึดติด (Bonded Strain Gage) และแบบไม่ยึดติด (Unbounded Strain Gage) ซึ่งสเตรนเกจทั้งสองชนิดจะมีลักษณะของโครงสร้างและการทำงานที่คล้ายกันคือ ทำด้วยเส้นลวดเล็กๆ ขดไปขดมาและนำไปติดกับวัตถุที่ต้องการตรวจวัดความเครียด เมื่อสเตรนเกจถูกดึงให้ยืดออก ความยาวของเส้นลวดจะเพิ่มขึ้นในขณะที่พื้นที่หน้าตัดจะลดลง ผลก็คือความต้านทานของเส้นลวดเพิ่มขึ้นดังสมการที่ (2.1)

$$R = \frac{pL}{A} \tag{2.1}$$

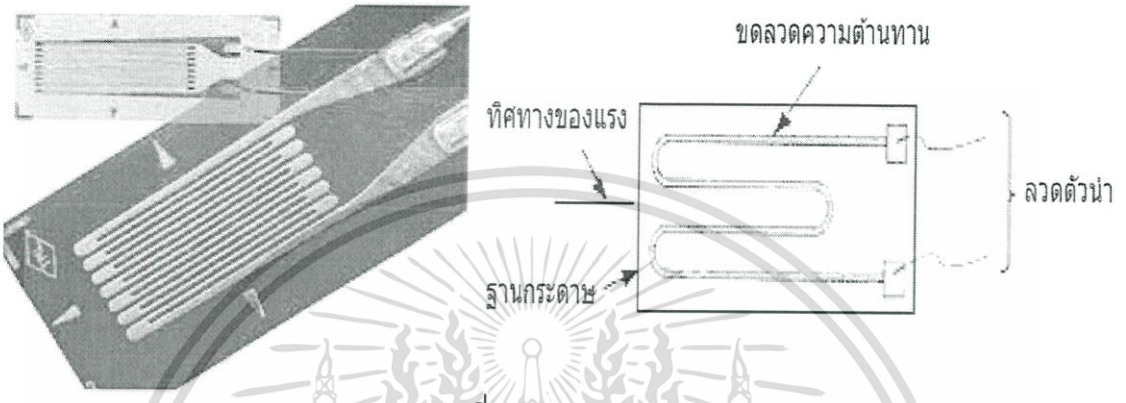
เมื่อ R คือ ความต้านทานของสเตรนเกจ หน่วยโอห์ม (Ω)

p คือ ค่าความต้านทานคงที่ของตัวโลหะ หน่วยโอห์มเมตร (Ωm)

L คือ ความยาวของเส้นลวด หน่วยเมตร (m)

A คือ พื้นที่หน้าตัดของตัวนำ หน่วยตารางเมตร (m^2)

โครงสร้างของสเตรนเกจทำมาจากเส้นลวดเล็กๆ ขดไปมาหลายๆ ครั้ง และยึดติดไว้บนแผ่นกระดาษหรือพลาสติกบางๆ โดยทั่วไปส้นลวดนี้จะมีเส้นผ่าศูนย์กลางประมาณ 0.001 นิ้ว และความต้านทาน 120 Ω



รูปที่ 2.11 สเตรนเกจ

การใช้สเตรนเกจในการตรวจวัดความเครียดวัตถุนั้น จะพิจารณาถึงปริมาณทางกายภาพสองสิ่งคือค่าความต้านทานของเกจ (Gage Resistance) ที่เปลี่ยนแปลงและค่าของความยาวที่เปลี่ยนแปลงซึ่งความสัมพันธ์ระหว่างตัวแปรสองตัวนี้จะแสดงเป็นอัตราส่วนที่เรียกว่า “เกจแฟกเตอร์ (Gage Factor)” ดังสมการทางคณิตศาสตร์ต่อไปนี้

$$K = \frac{\Delta R / R}{\Delta L / L} \tag{2.2}$$

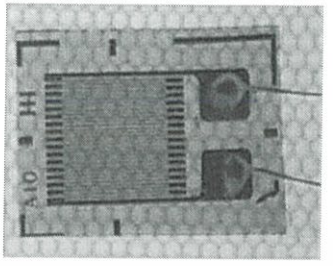
เมื่อ K คือ เกจแฟกเตอร์

R	คือ ความต้านทานเริ่มต้น	หน่วยโอห์ม (Ω)
ΔR	คือ ความต้านทานที่เปลี่ยนแปลง	หน่วยโอห์ม (Ω)
L	คือ ความยาวเริ่มต้น	หน่วยเมตร (m)
ΔL	คือ ความยาวที่เปลี่ยนแปลง	หน่วยเมตร (m)

ในเทอมของ $\Delta L/L$ สามารถกำหนดเป็นหน่วยของความเครียดโดยใช้อักษรแทนด้วย ε ดังนั้นจึงเขียนเป็นสมการได้ใหม่คือ

$$K = \frac{\Delta R / R}{\varepsilon} \tag{2.3}$$

สเตรนเกจจะแบ่งประเภทตามวัสดุที่เป็นเส้นลวดความต้านทานบางๆ ขนาดเล็กมาก หรือทำมาจากฟอยล์โลหะ (Metal Foil) บางๆ มาตัดเป็นรูปซิกแซกคล้ายๆ กับเส้นลวดของสเตรนเกจ สเตรนเกจชนิดนี้สามารถใช้งานในอุณหภูมิที่สูงได้



รูปที่ 2.12 สเตรนเกจที่ทำจากฟอยด์โลหะ

สเตรนเกจชนิดสารกึ่งตัวนำ ทำมาจากผลึกซิลิคอนที่มีเกจแฟกเตอร์สูงประมาณ 50 ถึง 200 สเตรนเกจชนิดนี้จะมีความไวสูงมากคือ ประมาณ 10 เท่าของสเตรนเกจแบบเส้นลวด แต่มีความเป็นเชิงเส้นต่ำมาก จะมีเฉพาะกับงานพิเศษ

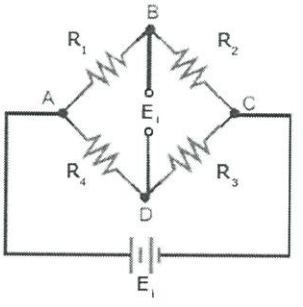
2.2.2 บริดจ์แบบวีทสโตน

วงจรวีทสโตนบริดจ์ (Wheatstone Bridge) เป็นวงจรที่นิยมใช้กับเซนเซอร์ที่มีเอาต์พุตเป็นการเปลี่ยนแปลงค่าความต้านทานไฟฟ้า ในการวัดหาค่าความต้านทานให้เป็นเอาต์พุตในรูปของสัญญาณทางไฟฟ้าเพื่อให้สามารถอ่านค่าและบันทึกสัญญาณได้อย่างแม่นยำ

ภายในวงจรวีทสโตนบริดจ์ ประกอบไปด้วย

1. ตัวต้านทานที่ต่อขนานกัน 2 สาขา แต่ละสาขาประกอบด้วยตัวต้านทาน 2 ตัวต่ออนุกรม
2. แหล่งจ่ายแรงดันไฟฟ้ากระแสตรง (Ei) ต่อขนานกับตัวต้านทานของวงจรทำหน้าที่จ่ายกระแสไฟฟ้าให้ไหลผ่านตัวต้านทานในวงจร

วงจรบริดจ์โดยทั่วไปจะประกอบไปด้วยตัวต้านทาน 4 ตัว มีค่าความต้านทาน R_1, R_2, R_3 และ R_4 ต่อเป็นวงจรบริดจ์ มีแหล่งจ่ายไฟต่อคร่อมแขนของบริดจ์ด้าน A และ C และมีตัวตรวจจับกระแสที่ปลายแขนบริดจ์ด้าน B และ D แสดงตามรูปที่ 2.13



รูปที่ 2.13 วงจรวีทสโตนบริดจ์

2.3 การวัดอัตราการไหล

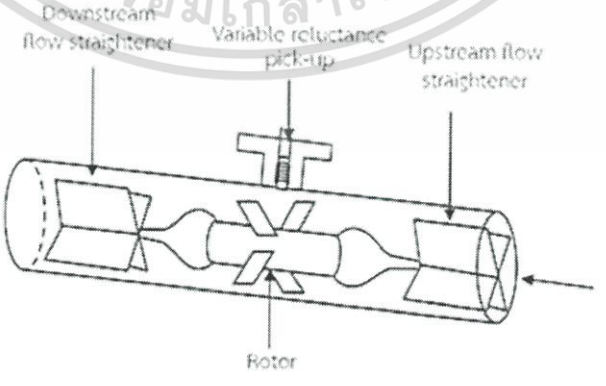
อัตราการไหล (Flow Rate) คือ ปริมาณของไหลที่ไหลผ่านพื้นที่หน้าตัดใดๆ ที่กำหนดต่อหนึ่งหน่วยเวลาอัตราการไหลเชิงปริมาตร (Volume Flow Rate) : Q มีสมการทั่วไป ดังนี้

$$Q = Av \tag{2.4}$$

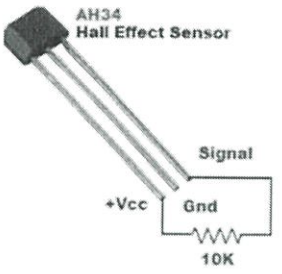
เมื่อ A คือ พื้นที่หน้าตัดการไหล

v คือ ความเร็วเฉลี่ยของการไหล

เครื่องมือวัดอัตราการไหล (Flow Sensor) ที่เลือกใช้ในโครงงานเป็นประเภท เทอร์ไบน์ (Turbine) เป็นเครื่องมือวัดอัตราการไหลที่สามารถใช้วัดอัตราการไหลของของไหลได้โดยตรงแบบหนึ่ง โดยหลักการทำงานของเทอร์ไบน์นั้นจะทำหน้าที่แปลงค่าอัตราการไหลของของไหลให้อยู่ในรูปของสัญญาณไฟฟ้าแต่ต้องใช้ร่วมกับอุปกรณ์ตัวอื่นอีกด้วย ภายในของอุปกรณ์วัดอัตราการไหลแบบนี้จะประกอบไปดวย ใบพัดทำจากวัสดุที่มีคุณสมบัติเป็นสารแม่เหล็กที่ติดตั้งอยู่บนโรเตอร์ ดังแสดงในรูปที่ 2.14 สำหรับการใช้งานนั้น เมื่อมีของไหลไหลเข้าไปในตัวเรือน และผ่านใบพัดที่ติดอยู่กับโรเตอร์จะทำให้ใบพัดหมุน และความเร็วในการหมุนของใบพัดจะเป็นสัดส่วนโดยตรงกับอัตราการตรวจจับแม่เหล็ก ลักษณะเป็นขดลวดพันรอบแม่เหล็กถาวรติดตั้งอยู่ที่ตัวเครื่องมือวัด เช่น Hall Effect Sensor มีหลักการทำงานคือ เมื่อใบหมุนรอบแกนเหล็กจะเกิดสนามแม่เหล็กขึ้น ทำให้ Hall Effect Sensor สามารถตรวจวัดสนามแม่เหล็กที่เกิดขึ้นได้ตามความเร็วในการหมุนของใบพัดแล้วให้สัญญาณพัลส์ออกมาตามความเร็วในการหมุนของใบพัด และใช้ตัวนับแบบดิจิตอลตรวจนับจำนวนพัลส์ที่เกิดขึ้น จากนั้นทำการประมวลผลด้วยไมโครโปรเซสเซอร์แปลงหน่วยให้อยู่ในรูปของปริมาตรซึ่งได้จากการนับจำนวนพัลส์ที่เกิดขึ้น หรืออัตราการไหล ซึ่งได้จากการนับจำนวนพัลส์ในหนึ่งช่วงเวลา หรือเรียกว่า อัตราการเกิดพัลส์



รูปที่ 2.14 โครงสร้างของเครื่องมือวัดอัตราการไหลแบบเทอร์ไบน์



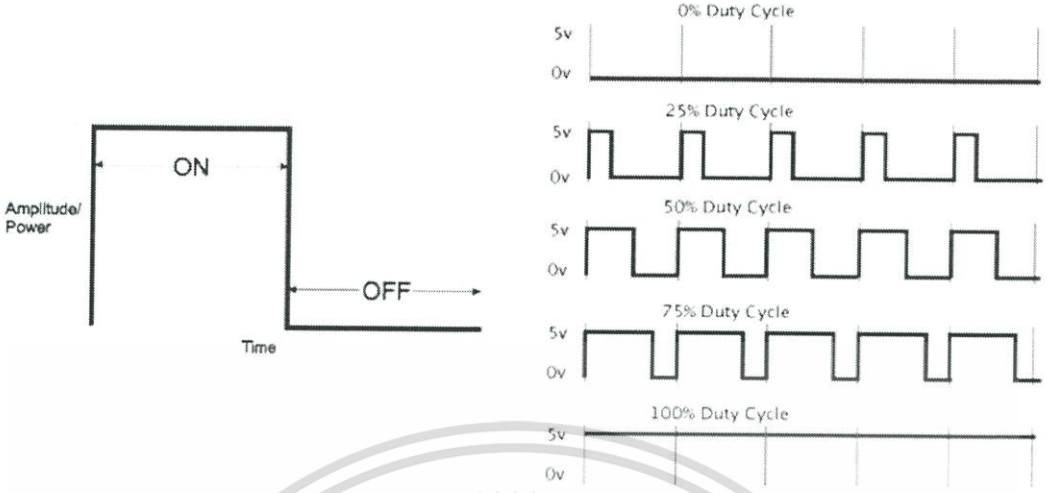
รูปที่ 2.15 Hall Effect Sensor

เครื่องมือวัดการไหลชนิดนี้มีลักษณะความเป็นเชิงเส้น (Linearity) ที่ดีที่อัตราการไหลสูงส่วนที่อัตราการไหลต่ำการไหลที่วัดได้จะได้รับผลกระทบจากแรงต้าน เนื่องจากความเสียดทานโดยทั่วไปเครื่องมือวัดการไหลชนิดเทอร์ไบน์มีย่านการวัด (Range) การไหลของของเหลวอยู่ในช่วง 1 ถึง 100,000 ลิตรต่อนาที และสำหรับการวัดการไหลของก๊าซมีย่านย่านการใช้งานในช่วงง 5 ถึง 100,000 ลิตรต่อนาที ไม่สามารถใช้วัดการไหลของของไหลที่มีสารแขวนลอยปะปนได้ และในการติดตั้งเครื่องมือวัดต้องพิจารณาระยะความตรงของท่อทั้งด้านหน้าก่อนเข้าตัวเครื่องมือวัด และระยะด้านหลัง เพื่อป้องกันไม่ให้ค่าความเร็วที่วัดได้คลาดเคลื่อน (Error)

2.4 สัญญาณ PWM

PWM ย่อมาจาก Pulse Width Modulation คือ การมอดูเลชันทางความกว้างของพัลส์ PWM เป็นการปรับเปลี่ยนที่สัดส่วน และความกว้างของสัญญาณพัลส์ โดยความถี่ของสัญญาณพัลส์จะไม่มีการเปลี่ยนแปลง หรือเป็นการเปลี่ยนแปลงที่ค่าของดิวตี้ไซเคิล (Duty Cycle) นั่นเอง ซึ่งค่าของดิวตี้ไซเคิลคือ ช่วงความกว้างของพัลส์ที่มีสถานะลอจิกสูง โดยคิดสัดส่วนเป็นเปอร์เซนต์จากความกว้างของพัลส์ทั้งหมด ยกตัวอย่างเช่น ถ้าหากค่าดิวตี้ไซเคิลมีค่าเท่ากับ 50% ก็หมายถึงใน 1 รูปสัญญาณพัลส์จะมีช่วงของสัญญาณที่เป็นสถานะลอจิกสูงอยู่ครึ่งหนึ่ง และสถานะลอจิกต่ำอยู่อีกครึ่งหนึ่ง ดังรูปที่ 2.16 และในทำนองเดียวกันถ้าหากค่าดิวตี้ไซเคิลมีค่ามาก หมายความว่าความกว้างของพัลส์ที่เป็นสถานะลอจิกสูงจะมีความกว้างมากขึ้น หากค่าดิวตี้ไซเคิลมีค่าเท่ากับ 100% ก็หมายความว่าจะไม่มีสถานะลอจิกต่ำเลย ซึ่งค่าดิวตี้ไซเคิลสามารถจะหาได้จากค่าความสัมพันธ์ดังนี้

$$\text{ค่าดิวตี้ไซเคิล} = (\text{ช่วงของสัญญาณพัลส์}/\text{คาบเวลาทั้งหมดของสัญญาณ}) \times 100 \qquad (2.5)$$



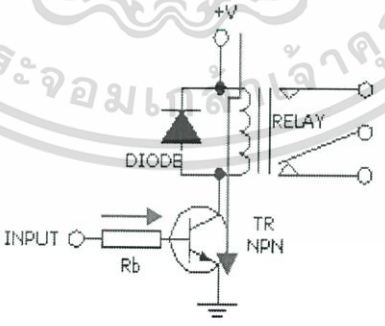
รูปที่ 2.16 กราฟแสดงค่า PWM ในรูปของค่าดิวตี้ไซเคิล

2.5 มอเตอร์กระแสตรง

มอเตอร์กระแสตรง (DC Motor) จะมีหลักการทำงานโดยวิธีการผ่านกระแสให้กับขดลวดในสนามแม่เหล็ก ซึ่งจะทำให้เกิดแรงแม่เหล็ก โดยส่วนของแรงนี้จะขึ้นอยู่กับกระแส และกำลังของสนามแม่เหล็ก

2.5.1 การขับมอเตอร์กระแสตรง

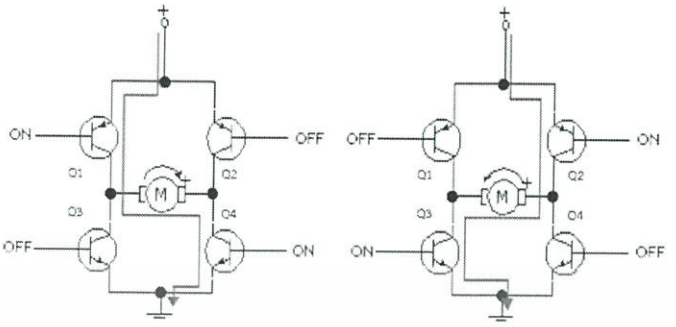
ในการใช้ไอซีไมโครคอนโทรลเลอร์เป็นตัวควบคุมการหมุนของมอเตอร์กระแสตรงนั้น จะต้องมีส่วนของวงจรที่เรียกว่าวงจรขับมอเตอร์ (Driver)



รูปที่ 2.17 การใช้ทรานซิสเตอร์เพื่อขับรีเลย์ให้ทำงาน

จากรูปที่ 2.17 เป็นวงจรขับรีเลย์โดยใช้ทรานซิสเตอร์ทำหน้าที่ขยายกระแส เพราะไม่สามารถจะใช้ขา เอาต์พุตของไมโครคอนโทรลเลอร์ป้อนกระแสไฟที่ขดลวดของรีเลย์โดยตรงได้ เนื่องจากว่ากระแสที่จ่ายออกมาจากขาเอาต์พุตของไมโครคอนโทรลเลอร์มีค่าน้อยเกินไป ดังนั้นจึงต้องมีส่วนของวงจรทรานซิสเตอร์เพื่อที่จะทำการขยายกระแสให้เพียงพอในการป้อนให้กับขดลวดของ

รีเลย์ ส่วนไดโอดนำมาต่อไว้สำหรับป้องกันแรงดันย้อนกลับที่เกิดจากการเหนี่ยวนำของสนามแม่เหล็กในขณะเกิดการยุบตัว ซึ่งอาจจะทำให้ทรานซิสเตอร์เสียหายได้



รูปที่ 2.18 การใช้ทรานซิสเตอร์เป็นวงจรขับ และกำหนดทิศทางของมอเตอร์กระแสตรง

จากรูปที่ 2.18 เป็นวงจรลิเนียร์บริดจ์แอมป์ ซึ่งจะประกอบไปด้วยทรานซิสเตอร์กำลัง 4 ตัวที่ทำหน้าที่ขับ และควบคุมทิศทางการหมุนของมอเตอร์ ถ้ากำหนดให้ทรานซิสเตอร์ Q1 และ Q4 อยู่ในสภาวะทำงาน (Active) กระแสไฟฟ้าจะไหลผ่านทรานซิสเตอร์จากซ้ายไปขวา โดยผ่านมอเตอร์กระแสตรงทำให้มอเตอร์หมุนไปทางขวา ในทำนองเดียวกันถ้าทำให้ทรานซิสเตอร์ Q2 และ Q3 อยู่ในสภาวะทำงาน (Active) กระแสไฟฟ้าก็จะไหลจากทางขวาไปทางซ้ายซึ่งจะส่งผลให้มอเตอร์กลับทิศทางการหมุนจากทางขวาไปทางซ้าย

2.5.2 การควบคุมความเร็วของมอเตอร์กระแสตรง

การควบคุมความเร็วของมอเตอร์กระแสตรงมีหลายวิธีด้วยกัน ซึ่งอาจจะใช้วิธีการควบคุมแบบพื้นฐานทั่วไป เช่น การควบคุมด้วยวิธีการใช้ตัวต้านทานปรับค่าโดยต่ออนุกรมกับมอเตอร์หรือใช้วิธีการการควบคุมโดยการเปลี่ยนค่าของระดับแรงดันที่ป้อนให้กับมอเตอร์ แต่การควบคุมในวิธีดังกล่าวถึงแม้ว่าจะควบคุมความเร็วมอเตอร์ให้คงที่ได้ แต่ที่ความเร็วต่ำจะส่งผลให้แรงบิดต่ำไปด้วย ดังนั้นจึงเลือกใช้วิธีการควบคุมโดยการจ่ายกระแสไฟให้กับมอเตอร์เป็นช่วงๆ โดยอาศัยกระแสไฟที่ป้อนให้กับมอเตอร์ให้เป็นค่าเฉลี่ยที่เกิดขึ้นในแต่ละช่วง ซึ่งเรียกว่าวิธีการของการมอดูเลชันทางความกว้างของพัลส์ (Pulse Width Modulation, PWM)

การมอดูเลชันทางความกว้างของพัลส์ (PWM) จะเป็นการปรับเปลี่ยนที่สัดส่วน และความกว้างของสัญญาณพัลส์ โดยความถี่ของสัญญาณพัลส์จะไม่มีการเปลี่ยนแปลงหรือเป็นการเปลี่ยนแปลงที่ค่าของดิวตี้ไซเคิล (Duty Cycle) นั้นเอง ซึ่งค่าของดิวตี้ไซเคิล คือ ช่วงความกว้างของพัลส์ที่มีสถานะลอจิกสูง โดยคิดสัดส่วนเป็นเปอร์เซ็นต์จากความกว้างของพัลส์ทั้งหมด ยกตัวอย่างเช่น ถ้าหากค่าดิวตี้ไซเคิลมีค่าเท่ากับ 50% จะหมายถึงใน 1 รูปสัญญาณพัลส์จะมีช่วงของสัญญาณที่เป็นสถานะลอจิกสูงอยู่ครึ่งหนึ่ง และสถานะลอจิกต่ำอยู่อีกครึ่งหนึ่ง และในทำนองเดียวกันถ้าหากค่าดิวตี้ไซเคิลมีค่ามาก หมายความว่าความกว้างของพัลส์ที่เป็นสถานะลอจิกสูงจะมีความกว้างมากขึ้น หาก

ค่าดิวตี้ไซเคิลมีค่าเท่ากับ 100% หมายความว่าจะไม่มีสถานะลอจิกต่ำเลย ซึ่งค่าดิวตี้ไซเคิลสามารถหาได้จากค่าความสัมพันธ์ดังนี้

$$\text{ค่าดิวตี้ไซเคิล} = \frac{\text{ช่วงของสัญญาณพัลส์}}{\text{คาบเวลาทั้งหมดของสัญญาณ}} \text{ X } 100\% \qquad (2.6)$$

2.6 เทคโนโลยี Wi-Fi

2.6.1 Wi-Fi

ตั้งแต่อินเตอร์เน็ตมีการใช้งานกันอย่างมาก และนับวันอัตราการใช้อินเตอร์เน็ตก็ได้เพิ่มขึ้นเรื่อยๆ อย่างต่อเนื่อง เป็นแรงบันดาลใจทำให้ Wi-Fi เทคโนโลยีไร้สายได้ถูกพัฒนาขึ้นมา Wi-Fi ย่อมาจากคำว่า Wireless-Fidelity เป็นเทคโนโลยีเครือข่ายไร้สาย ภายใต้เทคโนโลยีการสื่อสาร มาตรฐาน IEEE 802.11 โดยเป็นมาตรฐานที่ถูกอนุมัติให้ใช้จาก IEEE (The Institute of Electrical and Electronics Engineers) เพื่อให้อุปกรณ์คอมพิวเตอร์สามารถสื่อสารกันได้บนมาตรฐานการทำงานแบบเดียวกัน สำหรับเทคโนโลยีเครือข่ายไร้สายนี้จะใช้คลื่นความถี่วิทยุ (Radio Frequency, RF) และคลื่นความถี่อินฟาเรดในการรับส่งข้อมูล คลื่นความถี่วิทยุของเครือข่ายไร้สายจึงสามารถทะลุทะลวงกำแพงหรือสิ่งกีดขวางได้ทำให้การใช้งานบนเครือข่ายไร้สายมีความคล่องตัว และสะดวกสบายมากขึ้น โดยสามารถเชื่อมต่อเข้าสู่เครือข่ายได้ทุกที่ที่มีคลื่นสัญญาณ ข้อมูลจะถูกรับส่งผ่านคลื่นวิทยุความถี่ 2.4 GHz ด้วยความเร็ว 11 Mbps ระยะห่างประมาณ 300 ฟุต

2.6.2 มาตรฐานของเทคโนโลยี Wi-Fi

1. IEEE 802.11b ถูกเผยแพร่เมื่อปี พ.ศ. 2542 ซึ่งเป็นที่รู้จักกันดี และนิยมใช้กันอย่างแพร่หลาย ใช้เทคโนโลยีที่เรียกว่า CCK (Complimentary Code Keying) ผนวกกับ DSSS (Direct Sequence Spread Spectrum) รับส่งข้อมูลได้ด้วยความเร็วสูงสุดที่ 11 Mbps ผ่านคลื่นวิทยุความถี่ 2.4 GHz ซึ่งอุปกรณ์ต่างๆ ที่ใช้คลื่นความถี่นี้ ได้แก่ โทรศัพท์ไร้สาย, Bluetooth มาตรฐาน IEEE 802.11b เป็นที่ใช้กันอยู่ในปัจจุบัน ใช้เครื่องหมายการค้าที่รู้จักกันดีในนาม Wi-Fi โดยใช้ร่วมกับอุปกรณ์ยี่ห้อต่างๆ ที่มีเครื่องหมาย Wi-Fi ได้

2. IEEE 802.11a ถูกเผยแพร่เมื่อปี พ.ศ. 2542 ใช้เทคโนโลยีที่เรียกว่า OFDM (Orthogonal Frequency Division Multiplexing) รับส่งข้อมูลได้ด้วยความเร็วสูงสุดที่ 54 Mbps แต่จะใช้คลื่นวิทยุที่ความถี่ 5 GHz ซึ่งเป็นย่านความถี่สาธารณะสำหรับใช้งานในประเทศสหรัฐอเมริกา ซึ่งมีข้อเสียคือ บางประเทศย่านความถี่ดังกล่าวไม่สามารถนำมาใช้งานได้อย่างสาธารณะ เช่น ประเทศไทย ซึ่งไม่อนุญาตให้มีการใช้งานเนื่องจากความถี่ย่าน 5 GHz ได้ถูกใช้สำหรับกิจการอื่นแล้ว และรัศมีของสัญญาณมีขนาดค่อนข้างสั้นคือ ประมาณ 30 เมตร จึงไม่เป็นที่นิยม

3. IEEE 802.11g ถูกเผยแพร่เมื่อกลางปี พ.ศ. 2546 IEEE 802.11g เป็นมาตรฐานที่จะเข้ามาทดแทนมาตรฐาน IEEE 802.11b ใช้นำเทคโนโลยี OFDM มาประยุกต์ใช้ในช่องสัญญาณวิทยุความถี่ 2.4 GHz ซึ่งอุปกรณ์ IEEE 802.11g WLAN มีความสามารถในการรับส่งข้อมูลด้วยความเร็วสูงสุดที่ 54 Mbps ส่วนรัศมีสัญญาณของอุปกรณ์ IEEE 802.11g WLAN จะอยู่ระหว่างรัศมีสัญญาณของอุปกรณ์ IEEE 802.11a และ IEEE 802.11b เนื่องจากความถี่ 2.4 GHz เป็นย่านความถี่สาธารณะสากล และสามารถที่จะใช้งานร่วมกันกับอุปกรณ์ที่ใช้มาตรฐาน IEEE 802.11b ได้จึงทำให้ไม่จำเป็นที่จะต้องเปลี่ยนแปลงโครงสร้างพื้นฐานของระบบไร้สายที่ใช้กันอยู่เดิมอย่าง IEEE 802.11b เพียงแต่อัพเกรดเฟิร์มแวร์เท่านั้นหรือใช้งานกับ Access Point ได้เช่นเดียวกัน ดังนั้นจึงมีแนวโน้มสูงว่าอุปกรณ์ IEEE 802.11g WLAN จะได้รับความนิยมอย่างแพร่หลายหากมีราคาไม่แพงจนเกินไปและน่าจะมาแทนที่ IEEE 802.11b

4. IEEE 802.11n ในปัจจุบันได้รับการพัฒนาจากมาตรฐาน 802.11b และ 802.11g โดยมีวัตถุประสงค์เพื่อเพิ่มอัตราความเร็วในการรับส่งข้อมูลให้ได้สูงสุด 300-600 Mbps ซึ่งมีความเร็วมากกว่ามาตรฐานเก่าถึงประมาณ 5 เท่า โดยความแตกต่างของ 802.11n กับมาตรฐาน 802.11a/b/g คือ การเพิ่มการสื่อสารทั้งรับและส่งแบบหลายช่องสัญญาณ (MIMO: Multiple-Input Multiple-Output) และทำงานด้วยช่วงห่างของสัญญาณที่ 40 MHz ซึ่งเทคโนโลยี MIMO จะช่วยเพิ่มความสามารถในการรับส่งสัญญาณและกู้คืนข้อมูลจากสัญญาณได้ดีกว่า และเทคโนโลยีแยกแยะสัญญาณและแปลงสัญญาณให้กลายเป็นช่องสัญญาณเดียว (SDM: Spatial Division Multiplexing)

2.6.3 รูปแบบการเชื่อมต่อของระบบเครือข่ายไร้สาย

1. Peer-to-Peer (Ad Hoc Mode)

รูปแบบการเชื่อมต่อแลนไร้สายแบบ Peer to Peer เป็นการเชื่อมต่อแบบโครงข่ายโดยตรงระหว่างเครื่องคอมพิวเตอร์ โดยเครื่องคอมพิวเตอร์แต่ละเครื่องนั้นจะมีความเท่าเทียมกัน สามารถทำงานของตนเองได้ และขอใช้บริการเครื่องอื่นได้ จึงเหมาะสำหรับนำมาใช้งานเพื่อจุดประสงค์ด้านความรวดเร็ว หรือติดตั้งได้โดยง่ายเมื่อไม่มีโครงสร้างพื้นฐานที่จะรองรับ ตัวอย่างเช่น ในศูนย์ประชุมหรือการประชุมที่จัดนอกสถานที่

2. Client/Server (Infrastructure Mode)

ระบบเครือข่ายไร้สายแบบ Client/Server (Infrastructure Mode) มีลักษณะการรับส่งข้อมูลโดยอาศัย Access Point (AP) หรือเรียกว่า "Hot Spot" ทำหน้าที่เป็นสะพานเชื่อมต่อระหว่างระบบเครือข่ายแบบใช้สายกับคอมพิวเตอร์ลูกข่าย (Client) โดยจะกระจายสัญญาณคลื่นวิทยุเพื่อรับ-ส่งข้อมูลเป็นรัศมีโดยรอบ ซึ่ง AP 1 จุด สามารถให้บริการเครื่องลูกข่ายได้ถึง 15-50 อุปกรณ์ เหมาะสำหรับการนำไปขยายเครือข่าย หรือใช้ร่วมกับระบบเครือข่ายแบบใช้สายเดิมใน Office ห้องสมุด หรือในห้องประชุม เพื่อเพิ่มประสิทธิภาพในการทำงานให้มากขึ้น

3. Multiple access points and roaming

เป็นการเพิ่มจุดการติดตั้ง AP ให้มากขึ้น เพื่อให้การรับส่งสัญญาณในบริเวณของเครือข่ายขนาดใหญ่เป็นไปอย่างครอบคลุม

4. Use of an Extension Point

มีคุณสมบัติเหมือนกับ Access Point แต่ไม่ต้องผูกติดไว้กับเครือข่ายไร้สาย

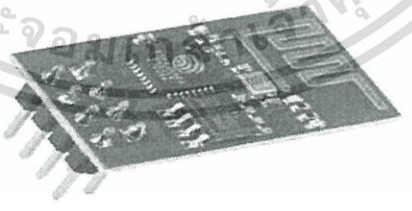
5. The Use of Directional Antennas

ระบบแลนไร้สายแบบนี้เป็นแบบใช้เสาอากาศในการรับส่งสัญญาณระหว่างอาคารที่อยู่ห่างกันโดยการติดตั้งเสาอากาศที่แต่ละอาคาร เพื่อส่งและรับสัญญาณระหว่างกัน

2.7 Wi-Fi Module ESP8266-01

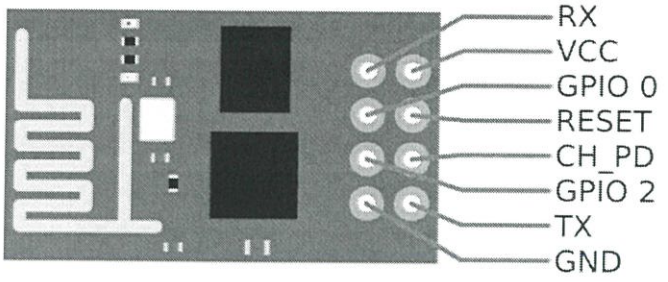
ESP8266 คือ โมดูลที่มีความพิเศษตรงที่ตัวมันสามารถโปรแกรมลงไปได้ ทำให้สามารถนำไปใช้งานแทนไมโครคอนโทรลเลอร์ได้เลย และมีพื้นที่ในการเขียนโปรแกรมได้มากถึง 4MB โดย ESP8266 เป็นชื่อของชิปไอซีบนบอร์ดของโมดูล ซึ่งไอซี ESP8266 ไม่มีพื้นที่โปรแกรม (Flash Memory) ในตัว ทำให้ต้องใช้ไอซีภายนอก (External Flash Memory) ในการเก็บโปรแกรม ที่ใช้การเชื่อมต่อผ่านโปรโตคอล SPI ซึ่งสาเหตุนี้เองทำให้โมดูล ESP8266 มีพื้นที่โปรแกรมมากกว่าไอซีไมโครคอนโทรลเลอร์เบอร์อื่นๆ

ESP8266 ทำงานที่แรงดันไฟฟ้า 3.3V – 3.6V การนำไปใช้งานร่วมกับเซนเซอร์อื่นๆ ที่ใช้แรงดัน 5V ต้องใช้วงจรแบ่งแรงดันมาช่วย เพื่อไม่ให้โมดูลพังเสียหาย กระแสที่โมดูลใช้งานสูงสุดคือ 200mA ความถี่คริสตอล 40MHz ทำให้เมื่อนำไปใช้งานอุปกรณ์ที่ทำงานรวดเร็วตามความถี่ เช่น LCD ทำให้การแสดงผลข้อมูลรวดเร็วกว่าไมโครคอนโทรลเลอร์ Arduino มาก



รูปที่ 2.19 Wi-Fi Module ESP8266-01

รุ่น ESP-01 เป็นรุ่นที่เหมาะสำหรับการเรียนรู้ และเหมาะสำหรับนำไปใช้งานกับงานที่มีโปรแกรมเล็กๆ มีขาทั้งหมด 8 ขา ได้แก่ VCC, CH_PD, Reset, Rx, Tx, GPIO0, GPIO2 และ GND โมดูลนี้ทำงานได้ค่อนข้างที่จะช้ามาก หากมีการเขียนโปรแกรมที่ไม่รัดกุมหรือมีคำสั่งทำงานมากๆ

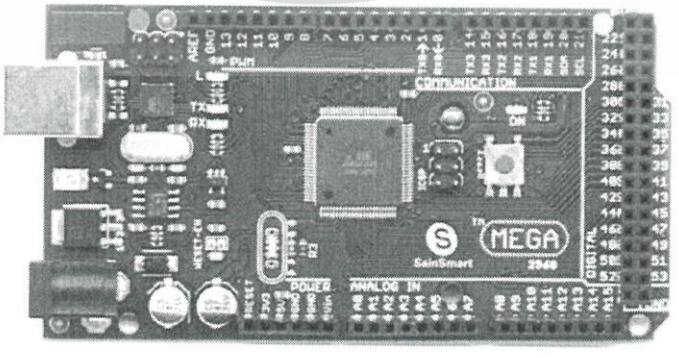


รูปที่ 2.20 ตำแหน่งขาของโมดูล Wi-Fi ESP8266-01

2.8 Arduino Microcontroller

Arduino อ่านว่า (อา-ดู-อิ-โน่ หรือ อาดุยโน่) เป็นบอร์ดไมโครคอนโทรลเลอร์ตระกูล AVR ที่มีการพัฒนาแบบ Open Source คือ มีการเปิดเผยข้อมูลทั้งด้านฮาร์ดแวร์ และซอฟต์แวร์ ตัวบอร์ด Arduino ถูกออกแบบมาให้ใช้งานได้ง่าย ดังนั้นจึงเหมาะสำหรับผู้เริ่มต้นศึกษา ทั้งนี้ผู้ใช้งานยังสามารถดัดแปลงเพิ่มเติม พัฒนาต่อยอดทั้งตัวบอร์ดหรือโปรแกรมต่อได้อีกด้วย ผู้ใช้งานสามารถต่อวงจรอิเล็กทรอนิคส์จากภายนอกแล้วเชื่อมต่อเข้ามาที่ขา I/O ของบอร์ดหรือเพื่อความสะดวกสามารถเลือกต่อกับบอร์ดเสริม (Arduino Shield) ประเภทต่างๆ เช่น Arduino Xbee Shield, Arduino Music Shield, Arduino Relay Shield, Arduino Wireless Shield และ Arduino GPRS Shield เป็นต้น มาต่อบนบอร์ด Arduino แล้วเขียนโปรแกรมพัฒนาต่อได้เลย

Arduino Mega 2560 R3 เป็นบอร์ด Arduino ที่ออกแบบมาสำหรับงานที่ต้องใช้ I/O มากกว่า Arduino Uno R3 เช่น งานที่ต้องการรับสัญญาณจากเซนเซอร์ หรือควบคุมเซอร์โวมอเตอร์หลายๆ ตัว ทำให้ขา I/O ของบอร์ด Arduino Uno R3 ไม่สามารถรองรับได้ ทั้งนี้บอร์ด Mega 2560 R3 ยังมีความหน่วยความจำแบบ Flash มากกว่า Arduino Uno R3 ทำให้สามารถเขียนโปรแกรมเข้าไปได้มากกว่าในความเร็วของ MCU ที่เท่ากัน



รูปที่ 2.21 Arduino Mega 2560 R3

โปรแกรมของ Arduino แบ่งได้เป็นสองส่วนคือ void setup() และ void loop() โดยฟังก์ชัน setup() เมื่อโปรแกรมทำงานจะทำคำสั่งของฟังก์ชันนี้เพียงครั้งเดียวใช้ในการกำหนดค่าเริ่มต้นของการทำงาน ส่วนฟังก์ชัน loop() เป็นส่วนทำงาน โปรแกรมจะทำคำสั่งในฟังก์ชันนี้ต่อเนื่องตลอดเวลา โดยปกติใช้กำหนดโหมดการทำงานของขาต่างๆ กำหนดการสื่อสารแบบอนุกรม ฯลฯ ส่วนของ loop() เป็นโค้ดโปรแกรมที่ทำงาน เช่น อ่านค่าอินพุต ประมวลผลสั่งงานเอาต์พุต ฯลฯ โดยส่วนกำหนดค่าเริ่มต้น เช่น ตัวแปรจะต้องเขียนที่ส่วนหัวของโปรแกรมก่อนถึงตัวฟังก์ชัน นอกจากนั้นยังต้องคำนึงถึงตัวพิมพ์ เล็ก-ใหญ่ ของตัวแปรและชื่อฟังก์ชันให้ถูกต้อง

2.9 Programmable Logic Control (PLC)

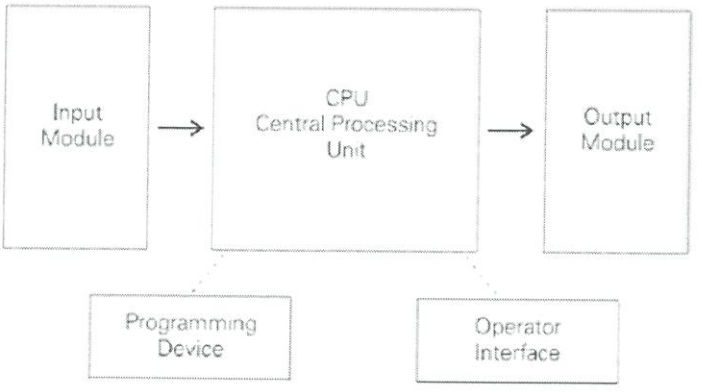
อุตสาหกรรมในยุคแรกๆ การควบคุมการทำงานในโรงงานอุตสาหกรรมทั่วไปอาศัยแรงงานคนในการตรวจวัด คำนวณ และการปรับแต่งเป็นส่วนใหญ่ ซึ่งการควบคุมลักษณะนี้ให้ผลการควบคุมที่ไม่คงที่ และมีความแม่นยำค่อนข้างน้อย เนื่องจากถูกจำกัดด้วยขีดจำกัดความสามารถของมนุษย์ จึงได้มีการนำเครื่องจักรมาใช้งานแทนแรงงานคนมากขึ้น ซึ่งช่วยให้ลดต้นทุนการผลิต และเพิ่มมาตรฐานทั้งในด้านสินค้า และการดำเนินงานของโรงงาน จึงได้มีการคิดวงจรรีเลย์ขึ้นเพื่อให้สามารถควบคุมการทำงานเครื่องจักรให้ทำงานได้อย่างอัตโนมัติ จากนั้นจึงได้พัฒนามาเป็น พีแอลซี ซึ่งทำให้สามารถลดความยุ่งยากทั้งในการติดตั้งโปรแกรม และการบำรุงรักษา เป็นต้น

2.9.1 โครงสร้างทั่วไป และหลักการทำงานพื้นฐานของ PLC

พีแอลซี (Programmable Logic Controller, PLC) เป็นอุปกรณ์ควบคุมการทำงานของเครื่องจักร หรือกระบวนการต่างๆ โดยมีโครงสร้างแบ่งออกเป็น 5 ส่วน ดังนี้

- หน่วยประมวลผลกลาง (Central Processing Unit, CPU)
- หน่วยความจำ (Memory Unit)
- หน่วยอินพุต/เอาต์พุต (Input/Output Unit)
- แหล่งจ่ายพลังงาน (Power Supply)
- อุปกรณ์ติดต่อภายนอก (Peripheral Device)

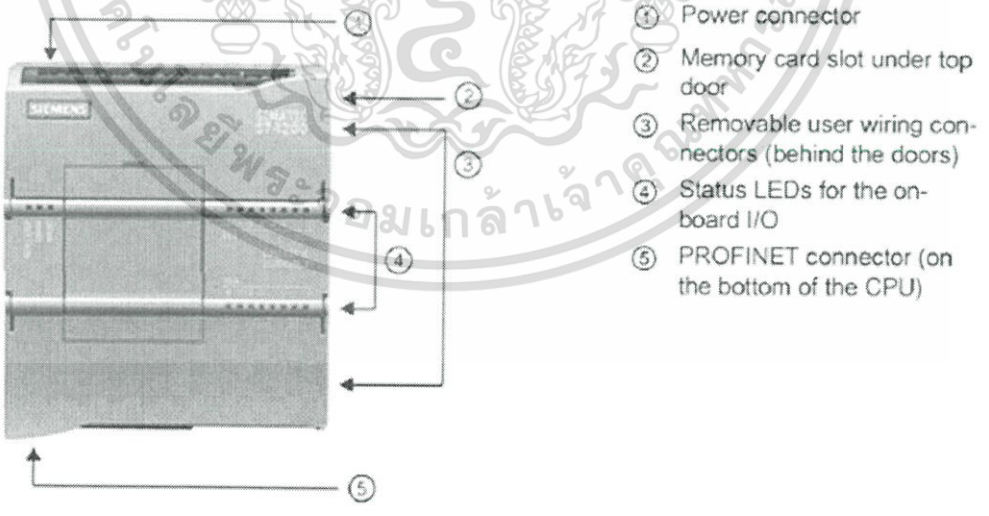
โดยตัวอินพุตจะรับข้อมูลทั้งสัญญาณแบบดิจิตอล และแอนะล็อกจากอุปกรณ์ต่างๆ เช่น สวิตช์เปิด –ปิด, เซนเซอร์ตรวจวัดอุณหภูมิ, เซนเซอร์ตรวจวัดระดับของเหลว หรือเซนเซอร์ตรวจวัดอัตราการไหลของของเหลว เป็นต้น จากนั้นหน่วยประมวลผลกลางจะแปลงสัญญาณนั้นๆ ให้เป็นสัญญาณลอจิก (Logic Signal) และประมวลผลตามโปรแกรมที่ผู้ใช้เขียนไว้ในหน่วยความจำแล้วตัวเอาต์พุตโมดูลจะแปลงคำสั่งควบคุมจากหน่วยประมวลผลกลางกลับเป็นสัญญาณดิจิตอลหรือแอนะล็อกเพื่อนำไปสั่งควบคุมอุปกรณ์ เช่น มอเตอร์ เป็นต้น



รูปที่ 2.22 โครงสร้างการทำงานพื้นฐานของ PLC

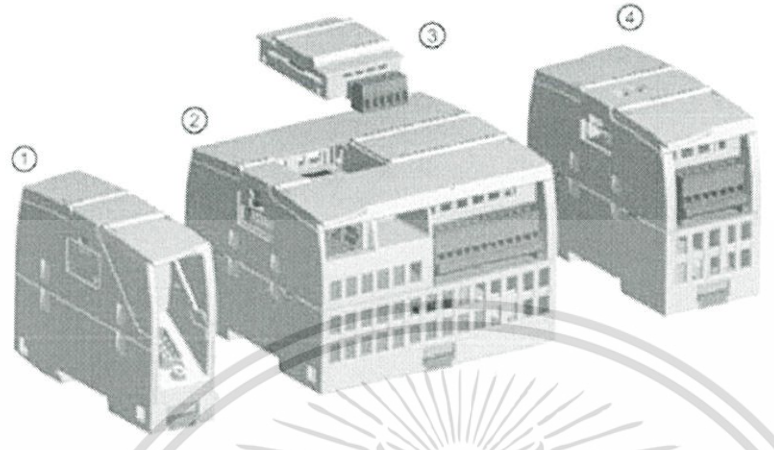
2.9.2 PLC Siemens รุ่น S7-1200

PLC จากบริษัท Siemens รุ่น S7-1200 เป็นพีแอลซีประเภทคอมแพค (Compact Type PLCs) ภายในกล่องประกอบด้วยตัวประมวลผล หน่วยความจำภาคอินพุต/เอาต์พุต และแหล่งจ่ายไฟ โดย PLC รุ่น S7-1200 นี้จะมีพอร์ตเฉพาะสำหรับการติดต่อสื่อสารผ่านเครือข่ายมาตรฐาน PROFINET ภายในตัว PLC คอมแพค แต่หากต้องการติดต่อสื่อสารผ่านมาตรฐาน RS232, RS485,GPRS, IEC, PROFIBUS, DNP3 และ WDC สามารถทำได้โดยการต่อ Communication Port Module เพื่อขยายเพิ่มได้



รูปที่ 2.23 PLC Siemens รุ่น S7-1200

อุปกรณ์ต่อขยาย (Expansion Capability Module) หากมีความจำเป็นที่จะต้องขยับขยายระบบ หรือต้องการเพิ่มความสามารถให้กับเครื่องจักรที่ใช้งาน สามารถเพิ่มอุปกรณ์ต่อขยายได้ เช่น Communication Port, Digital I/O, Analog I/O, RTD และ Thermocouple เป็นต้น



รูปที่ 2.24 การต่ออุปกรณ์ต่อขยาย

2.10 โปรแกรม STEP7 TIA Portal V13

โปรแกรม TIA Portal เป็นโปรแกรมสำหรับเขียนโปรแกรมสำหรับควบคุมพีแอลซีของบริษัทSiemens โดยเฉพาะ ซึ่งมีหลักการทำงานแบบอัตโนมัติเพื่อช่วยเหลือผู้ออกแบบระบบระหว่างการทำงาน ทำให้ผู้ออกแบบสามารถสร้างโปรเจคได้อย่างสะดวก และรวดเร็ว โดยในเวอร์ชั่น 13 นั้นมี Function Block ให้เลือกใช้มากมายตามความต้องการของผู้ออกแบบระบบ และยังสามารถจำลองเครื่องพีแอลซี รวมถึงในส่วนของการเขียนโปรแกรม ซึ่งโปรกรม TIA Portal รองรับการเขียนแบบLadder (LAD), Function Block Diagram (FBD), Structured Text (SCL), Statement List (STL) ซึ่งจะถูกเขียนในส่วน Program Blocks

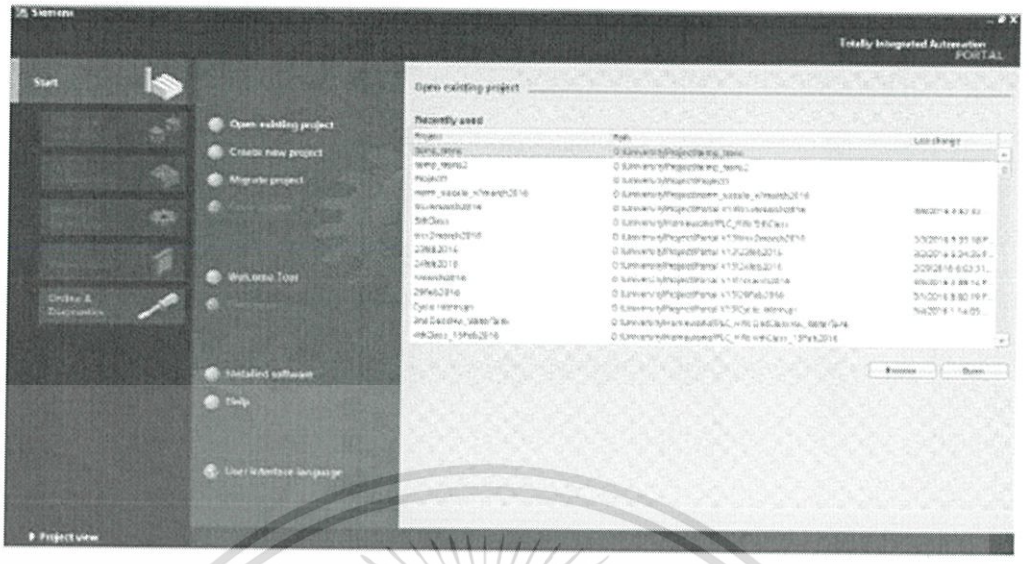
Program Blocks แบ่งเป็น 4 ประเภท

1 Organization Block (OB) – คือ ส่วนหลักที่ใช้ในการรันโปรแกรม PLC

2 Function Block (FB) – เป็นบล็อกที่จะเก็บค่า จาก Data Block โดยถาวรมาเขียนเป็นโปรแกรม ซึ่งสามารถนำมาเรียกใช้ใน OB ได้

3 Function (FC) – ทำงานเหมือนกับ FB แต่ไม่เก็บค่าไว้ใน Data Block

4 Data Block (DB) – เป็นบล็อกสำหรับเก็บข้อมูลต่างๆ ไว้สำหรับการเขียนโปรแกรม เช่น ตัวแปร รวมถึงค่าเริ่มต้นของตัวแปรนั้นๆ



รูปที่ 2.25 หน้าต่างแสดงโปรแกรม TIA Portal V13

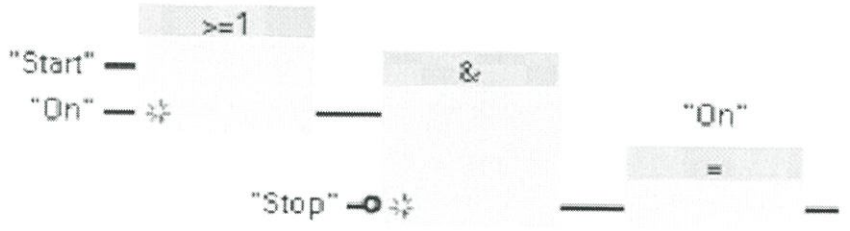
2.10.1 ภาษาที่ใช้ในการเขียนโปรแกรม TIA Portal V13

ภาษามาตรฐานในซอฟแวร์ TIA Portal V13 ที่ใช้ในการเขียนโปรแกรมมี 3 ภาษาต่อไปนี้

Ladder Logic (LAD) เป็นการเขียนโปรแกรมเชิงกราฟิก โดยจะแสดงผลออกมาคล้ายกับแผนผังการทำงานของวงจรรี- เลย์ โดย LAD นี้จะมีฟังก์ชันบล็อกอื่นๆ ให้ เช่น ฟังก์ชันการคำนวณทางคณิตศาสตร์ (Math), ตัวจับเวลา (Timer), ตัวนับ (Counter) เป็นต้น

รูปที่ 2.26 รูปแบบโปรแกรม Ladder Logic (LAD)

Function Block Diagram (FBD) รูปแบบของโปรแกรมเป็นการเขียนโปรแกรมเชิงกราฟิกเช่นเดียวกับการเขียนโปรแกรมแบบ Ladder Logic แต่การแสดงลอจิกต่างๆ จะใช้ในรูปแบบสัญลักษณ์ Boolean



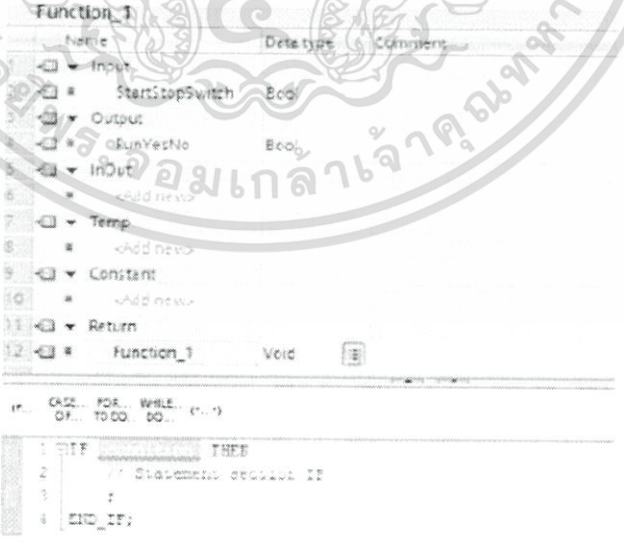
รูปที่ 2.27 รูปแบบการเขียนโปรแกรมแบบ Function

Structure Control Language (SCL) เป็นรูปแบบการเขียนโปรแกรมขั้นสูง โดยใช้พื้นฐานความรู้ภาษาปาสคาล (Pascal Language) ในการเขียน

```
"C" := #A+#B;                          Assigns two local variables to a tag
"Data_block_1".Tag := #A;              Assignment to a data block tag
IF #A > #B THEN "C" := #A;             Condition for the IF-THEN statement
"C" := SQRT (SQR (#A) + SQR (#B));     Parameters for the SQRT instruction
```

รูปที่ 2.28 รูปแบบการเขียนโปรแกรมแบบ Structure Control Language (SCL)

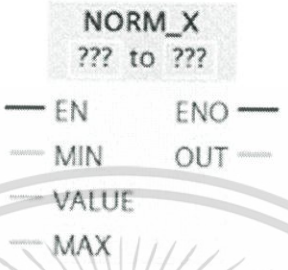
นอกจากนี้ซอฟแวร์ TIA Portal V13 ยังเปิดให้ผู้ใช้สามารถออกแบบรูปแบบของบล็อกโดยใช้ภาษาแบบ Structure Control Language (SCL) นี้ในการออกแบบอีกด้วย



รูปที่ 2.29 หน้าเชื่อมต่อของโปรแกรมภาษา SCL ที่สามารถกำหนดตัวแปรต่างๆ ได้

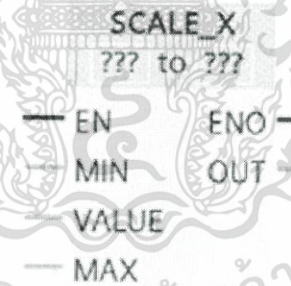
2.10.2 บล็อกคำสั่งพื้นฐานที่เกี่ยวข้อง (Basic Instruction)

NORM_X ใช้สำหรับสเกลค่าโวลต์ที่ได้จากแอนะล็อกขาเข้าที่ได้รับมาจากตัวเซนเซอร์ 0–5 โวลต์ หรือ จากค่าสเกลขาแอนะล็อกขาเข้า 0–13824 ออกเป็นค่าลิเนียร์ 0-1 โดยบล็อกสามารถรับข้อมูลประเภท Int, Dint, SInt, USInt, UInt, UDInt, Real และ LReal ส่วนขาออกสามารถเลือกให้ออกได้เป็นข้อมูลประเภท Real และ LReal



รูปที่ 2.30 ฟังก์ชันบล็อก NORM_X

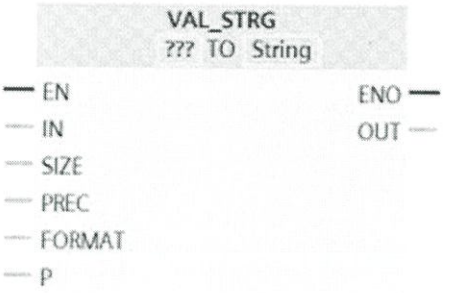
SCALE_X ใช้สำหรับสเกลค่าลิเนียร์ 0-1 ที่ได้จากขาออกของ NORM_X ให้เป็นช่วงที่ต้องการใช้งาน โดยรูปแบบของข้อมูลขาเข้าสามารถรับข้อมูลประเภท Real และ LReal ส่วนขาออกสามารถเลือกให้ออกได้เป็นข้อมูลประเภท Int, Dint, SInt, USInt, UInt, UDInt, Real และ LReal



รูปที่ 2.31 ฟังก์ชันบล็อก SCALE_X

2.10.3 คำสั่งเพิ่มเติมที่เกี่ยวข้อง (Extended Instruction)

VAL_ STRG ใช้สำหรับแปลงรูปแบบข้อมูล (Data Type) จาก USInt, SInt, UInt, Int, UDInt, DInt, Real หรือ LReal เป็น String



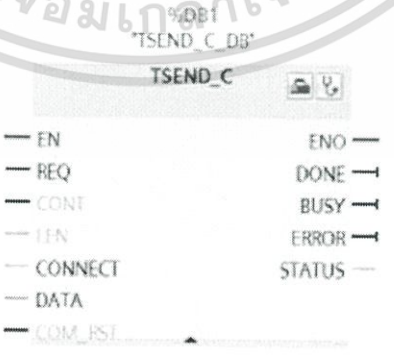
รูปที่ 2.32 ฟังก์ชันบล็อก VAL_STRG

Strg_TO_Char ใช้สำหรับแปลงรูปแบบข้อมูลจาก String เป็น Array of Chars

รูปที่ 2.33 ฟังก์ชันบล็อก Strg_TO_Char

2.10.4 บล็อกการเชื่อมต่อ (Communications)

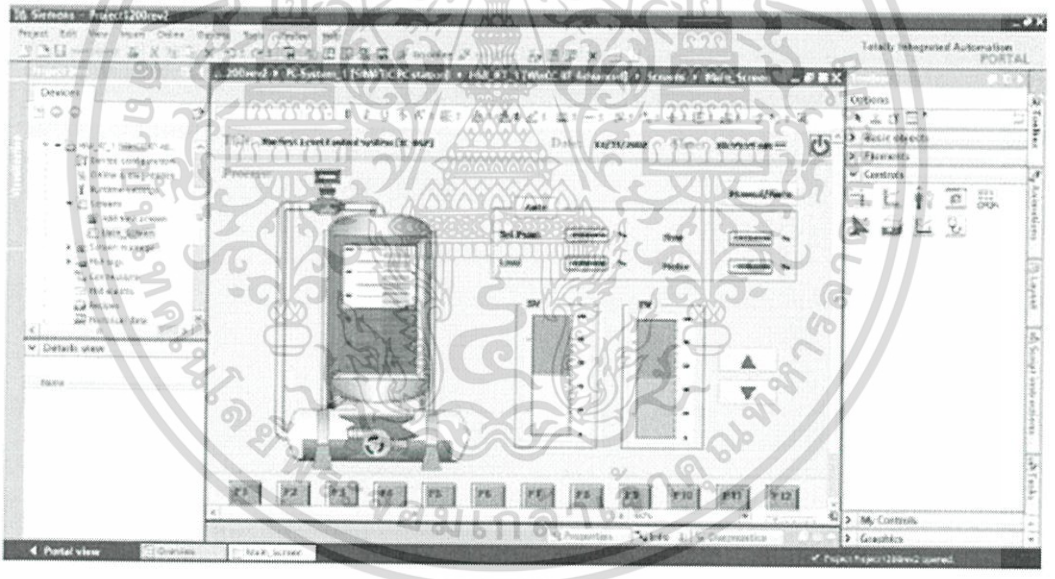
TSEND_C ใช้สำหรับส่งข้อมูลไปยังเครื่องที่ติดต่อผ่านโปรโตคอล TCP/IP ภายในบล็อกจะประกอบไปด้วย 3 บล็อก คือ TCON สำ หรับการสร้างการเชื่อมต่อกับเครื่อง Partner, TSEND สำหรับการส่งข้อมูลไปยังเครื่อง Partner และ TDISCON สำหรับยกเลิกการเชื่อมต่อกับเครื่อง Partner



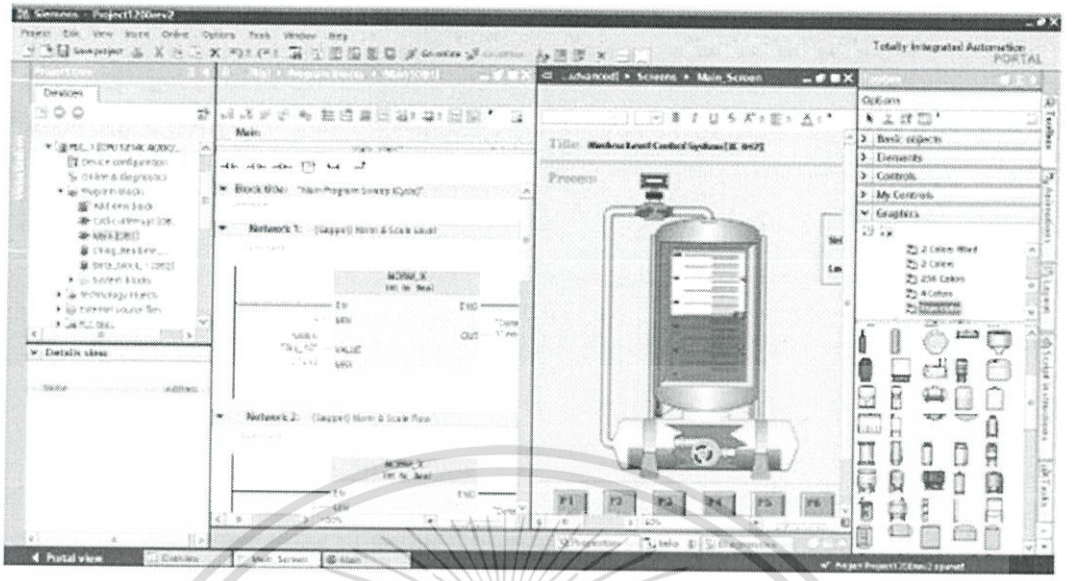
รูปที่ 2.34 ฟังก์ชันบล็อก TSEND_C

2.11 โปรแกรม WinCC Runtime Advanced V13

โปรแกรม WinCC Runtime Advanced V13 เป็นโปรแกรมของบริษัท Siemens ที่พัฒนาขึ้นมาภายใต้คอนเซ็ป "One Engineering Environment" – One Software Project For All Automation Tasks คือ หนึ่งโปรแกรมสำหรับงานอัตโนมัติทั้งหมด โดย WinCC Runtime Advanced V13 เป็นโปรแกรมที่เสริมมากับ STEP7 TIA Portal V13 ใช้สำหรับสร้างหน้าจอแสดงผลสำหรับติดตามค่า และควบคุมระบบสำหรับผู้ใช้งาน (Human Interface, HMI) เนื่องจากเป็นโปรแกรมเดียวกันท้าให้ใช้งานง่าย และสะดวก เพราะสามารถเขียนโปรแกรมแลดเดอร์สำหรับควบคุม และสร้างหน้าเชื่อมต่อได้ภายในโปรแกรมเดียวพร้อมๆ กัน นอกจากนี้ภายในโปรแกรมยังมีรูปภาพกราฟิกที่ใช้ในกระบวนการต่างๆ ในโรงงานอุตสาหกรรมเป็นจำนวนมากที่สามารถเลือกใช้ได้ทันที หรือหากผู้ใช้ต้องการที่จะสร้างกราฟิกของตัวเองเพิ่มก็สามารถทำได้เช่นกัน โดยสามารถสร้างกราฟิกได้จากอุปกรณ์วาดพื้นฐานต่างๆ ภายในตัวโปรแกรม หรือสามารถสร้างจากโปรแกรมอื่นๆ เช่น Adobe Illustrator, Adobe Photoshop ฯลฯ แล้วอิมพอร์ตเข้ามาในโฟลเดอร์ My Graphics Folder ได้



รูปที่ 2.35 หน้าต่างโปรแกรม WinCC Runtime Advanced V13



รูปที่ 2.36 หน้าต่าง WinCC Screen พร้อมกับหน้า Main Block สำหรับเขียนโปรแกรม

2.12 ระบบควบคุมแบบสัดส่วน-ปริพันธ์-อนุพันธ์ (Proportional-Integral-Derivative Controller, PID Controller)

การควบคุมแบบพีไอดีเป็นการควบคุมสัญญาณเอาต์พุตของกระบวนการ โดยใช้แบบสัดส่วนรวมกับแบบปริพันธ์ และแบบอนุพันธ์ (PID Controller) หรือที่เรียก “การควบคุมแบบ 3 เทอม” คือ Proportional (P) หมายถึงการปรับสัดส่วนสัญญาณ, Integral (I) หมายถึง การอินทิเกรตสัญญาณ และ Derivative (D) หมายถึง การอนุพันธ์สัญญาณ แสดงในรูปของสมการได้ดังนี้

$$output = K_P e + K_I \int_0^t edt + K_D \frac{de}{dt} \qquad (2.7)$$

เมื่อ K_P = ค่าคงที่แบบสัดส่วน

K_I = ค่าคงที่ของการอินทิกรัล

K_D = ค่าคงที่ของการอนุพันธ์

$\int_0^t edt$ = การอินทิกรัลค่าความคลาดเคลื่อนในช่วงคาบเวลา

$\frac{de}{dt}$ = อัตราการเปลี่ยนแปลงของความคลาดเคลื่อนเทียบเวลา

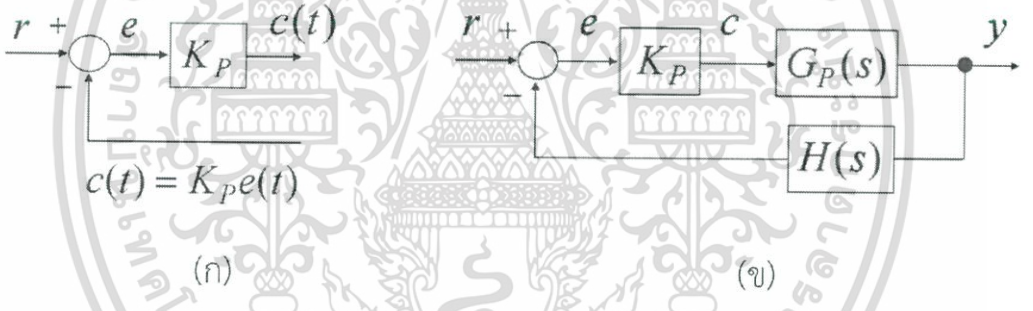
e = ค่าความคลาดเคลื่อนของกระบวนการ โดย e=SV-PV

เมื่อ SV คือค่าเป้าหมาย และ PV คือค่ากระบวนการ

ในสมการพีไอดีโปรแกรมจะนำค่าความผิดพลาด (Error) ของกระบวนการที่ได้จากผลต่างของค่า SV กับ ค่า PV ที่มาจากอุปกรณ์ตรวจวัดมาเป็นตัวแปรในสมการ และทำการส่งสัญญาณแรงดันเอาต์พุต ให้เป็นไปตามสมการพีไอดี และค่าตัวแปร K P , K I , K D ที่ป้อนค่าไว้โปรแกรมจะปรับค่าแรงดันเอาต์พุตจนกว่าค่า SV จะเท่ากับค่า PV โปรแกรมจะรักษาแรงดันเอาต์พุตไว้เพื่อให้ได้ค่าของกระบวนการตามที่ต้องการ

2.12.1 ตัวควบคุมแบบสัดส่วน (Proportional Controller)

ลักษณะการทำงานเป็นการส่งสัญญาณเอาต์พุตออกมาเป็นสัดส่วนโดยตรงกับสัญญาณค่าความผิดพลาด บล็อกไดอะแกรม และฟังก์ชันการทำงานเป็นดังรูปที่ 2.37(ก) ตัวควบคุมแบบนี้จะนำเอาค่าความผิดพลาดระหว่างสัญญาณอ้างอิงกับสัญญาณเอาต์พุตของตัวควบคุม แล้วตัวควบคุมจะทำการสร้างสัญญาณเอาต์พุตด้วยการขยายสัญญาณความผิดพลาดดังกล่าวด้วยค่าอัตราขยายของตัวควบคุมบล็อกไดอะแกรม และลักษณะของการประมวลผลสัญญาณเป็นดังรูปที่ 2.37(ข)

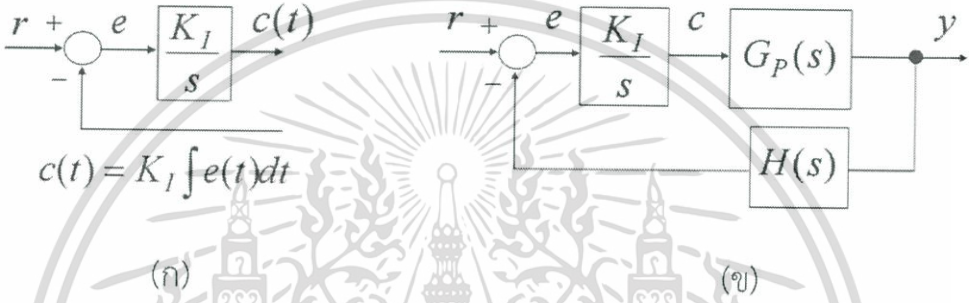


รูปที่ 2.37 ตัวควบคุมแบบสัดส่วน

ถ้านำการควบคุมแบบสัดส่วนไปใช้กับระบบชนิด 0 (System Type 0) ตัวควบคุมแบบนี้จะไม่สามารถขจัดค่าความผิดพลาดในสภาวะคงตัวได้ แต่ก็สามารถทำให้ค่าผิดพลาดดังกล่าวมีค่าน้อยลงด้วยการปรับค่าให้สูง ค่าที่สูงจะเป็นผลให้ค่าเอาต์พุตมากขึ้นตาม (ระบบมีผลตอบสนองเร็วขึ้น) แต่หากค่า มากเกินไประบบอาจจะไม่เสถียรได้ อาจทำให้ระบบมีค่าพุ่งเกินสูง ซึ่งอาจเป็นอันตรายต่อระบบได้ ในทางตรงกันข้ามหากค่าน้อยเกินไปอาจทำให้ระบบตอบสนองช้าเกินไป หรืออาจกล่าวได้ว่าระบบไม่สามารถสู้กับสิ่งรบกวนที่เกิดได้ ตัวอย่างสถานการณ์เช่น การเร่งรอบ เครื่องยนต์ เมื่อมีโหลดมากระทำกับเครื่องยนต์มากขึ้น ตัวคอนโทรลเลอร์จะต้องเร่งรอบเครื่องเพิ่ม เพื่อให้สามารถรองรับการกระทำของโหลดที่เพิ่มมาได้ โดยหากคอนโทรลเลอร์เร่งรอบเครื่องช้าอาจ ส่งผลให้ความเร็วรอบของเครื่องตกลง และส่งผลให้เครื่องดับหรือระบบหยุดทำงานในที่สุด

2.12.2 ตัวควบคุมแบบปริพันธ์ (Integral Controller)

ลักษณะการทำงานเป็นการส่งสัญญาณเอาต์พุตออกมาจากการอินทิเกรตสัญญาณค่าความผิดพลาด บล็อกไดอะแกรม และฟังก์ชันการทำงานเป็นดังรูปที่ 2.38(ก) ตัวควบคุมแบบปริพันธ์จะนำเอาสัญญาณความผิดพลาดระหว่างสัญญาณอ้างอิงกับสัญญาณ เอาต์พุตของตัวควบคุม แล้วตัวควบคุมจะทำการสร้างสัญญาณเอาต์พุตด้วยการอินทิเกรตสัญญาณ ความผิดพลาดดังกล่าวแล้วคูณด้วยค่า อัตราขยายของตัวควบคุม บล็อกไดอะแกรม และลักษณะของ การประมวลผลสัญญาณเป็นดังรูปที่ 2.38(ข)

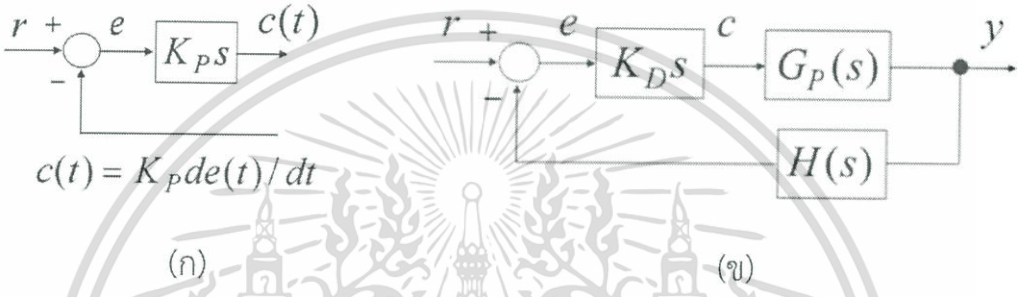


รูปที่ 2.38 ตัวควบคุมแบบปริพันธ์

จุดเด่นของตัวควบคุมแบบนี้คือ ถ้านำไปใช้กับระบบชนิด 0 (System Type 0) ตัวควบคุมแบบนี้จะสามารถขจัดค่าความผิดพลาดในสภาวะคงตัวได้ ข้อด้อยที่อาจเกิดขึ้นในการนำไปใช้งานคือ ตัวควบคุมแบบนี้ไม่สามารถลดผลของการพุ่งเกินของผลตอบสนองได้ และการปรับอัตราขยายให้มีค่าสูงอาจทำให้ผลตอบสนองที่ไม่พึงประสงค์ เช่น การปรับค่าเกนให้สูงขึ้นอาจมีผลทำให้ผลตอบสนองของระบบเกิดการแกว่งตัวได้

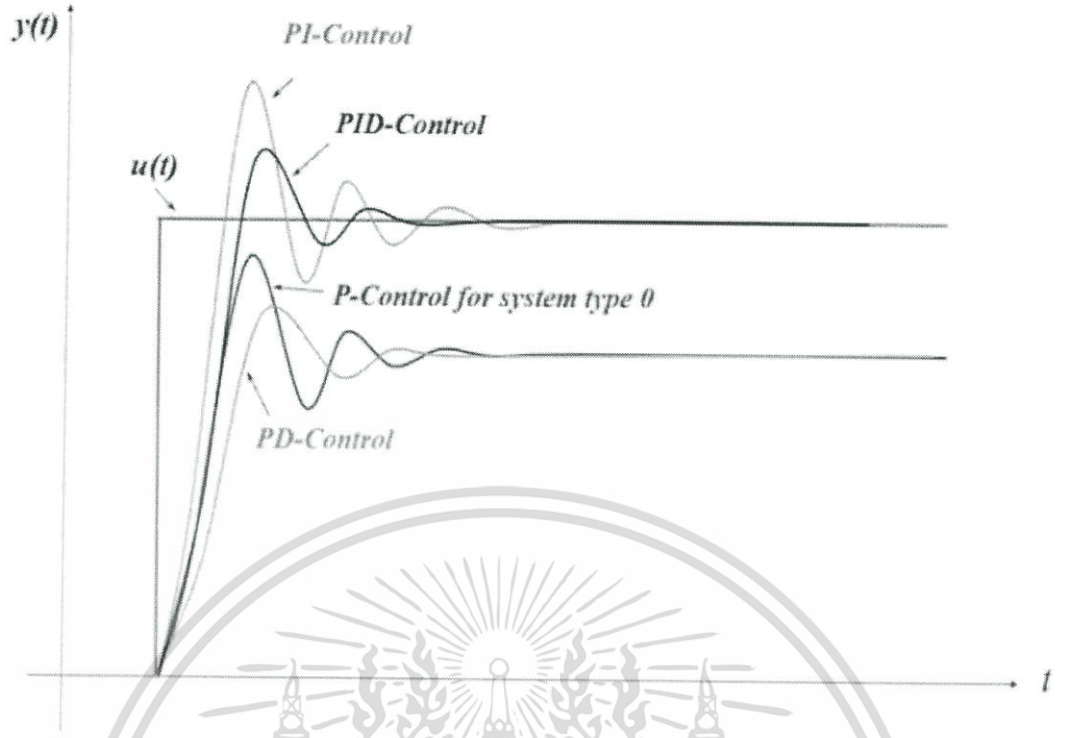
2.12.3 ตัวควบคุมแบบอนุพันธ์ (Derivative Controller)

ลักษณะการทำงานเป็นการส่งสัญญาณเอาต์พุตมาจากการอนุพันธ์ค่าความผิดพลาด บล็อกไดอะแกรม และฟังก์ชันการทำงานเป็นดังรูปที่ 2.39(ก) ตัวควบคุมแบบนี้จะนำเอาค่าสัญญาณความผิดพลาดระหว่างสัญญาณอ้างอิงกับสัญญาณเอาต์พุตมาเป็นอินพุตของตัวควบคุมแล้วตัวควบคุมจะทำการสร้างสัญญาณเอาต์พุตด้วยการอนุพันธ์สัญญาณความผิดพลาดดังกล่าว แล้วคูณด้วยค่าอัตราขยายของตัวควบคุม บล็อกไดอะแกรม และลักษณะของการประมวลผลสัญญาณเป็นดังรูปที่ 2.39(ข)



รูปที่ 2.39 ตัวควบคุมแบบอนุพันธ์

จุดเด่นของการควบคุมแบบนี้คือตัวควบคุมแบบนี้ใช้สำหรับลดผลของค่าพุ่งเกินของผลตอบสนองได้ ลดผลตอบสนองที่มีการเปลี่ยนแปลงไปมาได้ แต่ต้องปรับค่าอัตราขยายให้เหมาะสมด้วย ไม่เช่นนั้นอาจจะทำให้ได้ผลตอบสนองที่ไม่เป็นที่พึงประสงค์ ปัญหาที่อาจเกิดขึ้นในการนำตัวควบคุมแบบนี้ไปใช้คือ ตัวควบคุมแบบนี้จะไม่สามารถขจัดค่าความผิดพลาดในสภาวะคงตัวได้ และการใช้ตัวควบคุมนี้อาจจะทำให้ได้ผลตอบสนองที่ช้าลงได้



รูปที่ 2.40 ผลตอบสนองต่ออินพุตแบบขั้นบันไดในการใช้งานระบบควบคุมพีไอดีในลักษณะต่างๆกัน

จากรูปที่ 2.40 จะเห็นว่าถ้าหากนำตัวควบคุมแบบสัดส่วนไปใช้กับระบบอันดับหนึ่ง ผลตอบสนองที่ได้จะมีค่าความผิดพลาดในสภาวะคงตัว ซึ่งสามารถลดผลกระทบได้โดยการเพิ่มค่าอัตราขยายของตัวควบคุมให้สูงขึ้นสำหรับการนำไปใช้กับระบบที่มีอันดับสูงกว่า และเป็นระบบชนิด0 ค่าความผิดพลาดในสภาวะคงตัวก็จะยังมีอยู่ และการลดผลกระทบด้วยการปรับค่าอัตราขยายของตัวควบคุมแบบสัดส่วน อาจจะส่งผลให้การพุ่งเกินมีค่าสูงขึ้นได้ ถ้าหากใช้ตัวควบคุมร่วมกันระหว่างตัวควบคุมแบบสัดส่วนกับแบบปริพันธ์ (PI Controller) ตัวควบคุมแบบปริพันธ์จะช่วยขจัดค่าความผิดพลาดในสภาวะคงตัว แต่ผลตอบสนองที่ได้จะยังมีค่าพุ่งเกินเหมือนเดิม ถ้าหากใช้ตัวควบคุมร่วมกันระหว่างแบบสัดส่วนกับแบบอนุพันธ์ (PD Controller) การพุ่งเกินของผลตอบสนองจะลดลง แต่ค่าความผิดพลาดในสภาวะคงตัวจะยังคงอยู่ ดังนั้นหากใช้ตัวควบคุมแบบสัดส่วน ปริพันธ์ และอนุพันธ์ร่วมกัน (PID Controller) ด้วยการปรับค่าอัตราขยายให้เหมาะสมกับระบบนั้นๆ ก็จะได้ผลตอบสนองแบบหน่วงต่ำกว่าวิกฤตที่มีค่าพุ่งเกินเหมาะกับระบบนั้นๆ

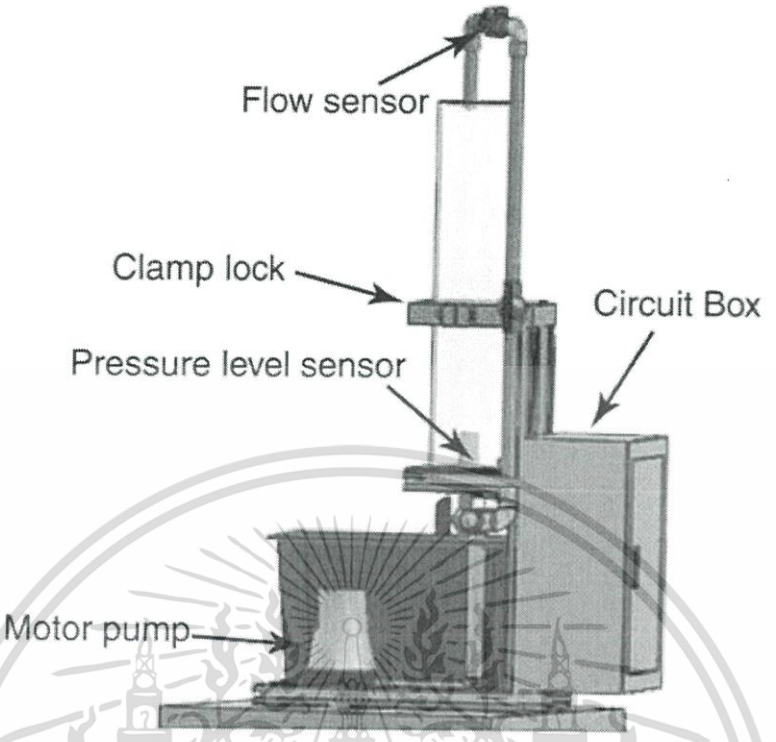
บทที่ 3
ขั้นตอนการดำเนินงาน

ขั้นตอนการดำเนินงานสำหรับโครงงาน ระบบควบคุมระดับน้ำอัตโนมัติแบบไร้สาย เริ่มจากการศึกษาวางแผนระบบ ศึกษาข้อมูลต่างๆที่เกี่ยวข้อง หลังจากนั้นทำการแบ่งงานออกเป็นฝ่ายต่างๆ ดังนี้

ฮาร์ดแวร์ จะทำการออกแบบโครงสร้าง และจัดวางตำแหน่งของอุปกรณ์, เซนเซอร์ต่างๆ ของระบบด้วโปรแกรม SolidWorks และใช้โปรแกรม MakerWear ในการขึ้นรูปชิ้นงาน 3 มิติ ของชิ้นงานบางส่วน

Level, Flow Transmitter และ Actuator ในส่วนของ Level Transmitter นั้นจะใช้ Pressure Sensor เป็นตัววัดแรงดันของระดับน้ำภายในถังแล้วส่งค่าอินพุตให้แก่ระบบ เช่นเดียวกับ Flow Transmitter ที่ใช้ Flow Sensor ในการวัดอัตราการไหลเพื่อส่งค่าอินพุต และ Actuator จะทำหน้าที่ควบคุมความเร็วของมอเตอร์ปั๊มน้ำ ซึ่งในส่วน Sensor และ Actuator นี้จะต้องทำการเขียนโปรแกรมเพื่อรับ-ส่งข้อมูลต่างๆ แก่ระบบผ่านทาง Wi-Fi Module โดยใช้โปรแกรม Arduino ในการเขียนโปรแกรม

ซอฟต์แวร์ ใช้โปรแกรม STEP7 TIA Portal V13 ในการเขียนแลดเดอร์ไดอะแกรม โดยใช้บล็อกลูปพีไอดีในการควบคุมระดับน้ำตามค่าเป้าหมายที่เราต้องการโดยการควบคุมความเร็วมอเตอร์ปั๊มน้ำ ซึ่งสั่งการโดยบล็อก TSEND_C ตามค่าเอาต์พุตที่ได้จากการประมวลผลแบบป้อนกลับของลูปพีไอดี นอกจากนี้ยังใช้โปรแกรม WinCC Runtime Advanced V13 ในการสร้างหน้าต่างสำหรับการติด ตาม และควบคุมระบบเพื่อให้สามารถติดตาม และควบคุมระบบได้ผ่านหน้าจอคอมพิวเตอร์ นอกจากนี้ยังได้สร้างหน้าต่างสำหรับการแสดงกราฟผลตอบสนองที่ได้จากลูปพีไอดีอีกด้วย



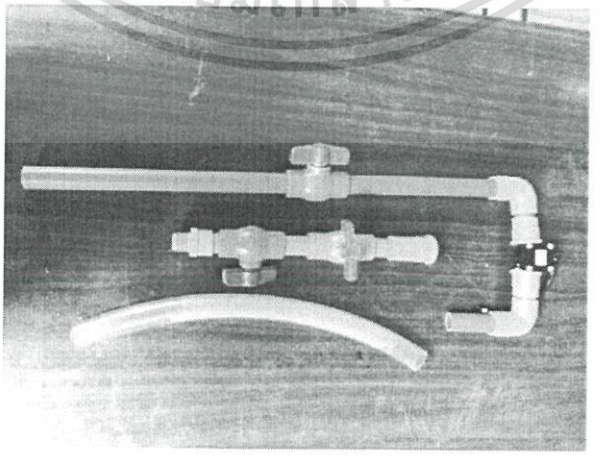
รูปที่ 3.1 จำลองโครงสร้างระบบด้วยโปรแกรม SolidWorks

3.1 การเลือกใช้อุปกรณ์

3.1.1 วาล์วและท่อ PVC

คุณสมบัติ

1. วาล์วประเภทบอลวาล์วขนาด ½ นิ้ว ปรับหมุนไปมาง่ายเพื่อการควบคุมระดับน้ำ
2. ท่อ PVC หาง่าย และมีราคาถูก

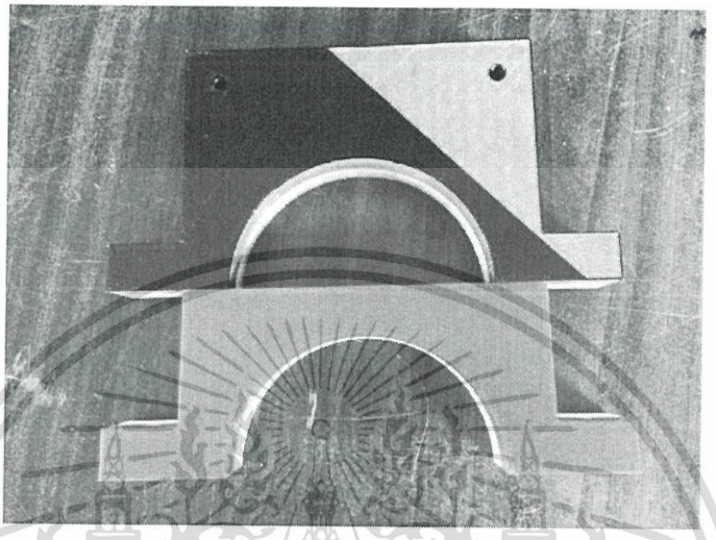


รูปที่ 3.2 ท่อ PVC สำหรับปล่อยน้ำ

3.1.2 Clamp Lock

คุณสมบัติ

1. สามารถล็อคกับแทงค์สำหรับควบคุมระดับน้ำได้พอดี
2. มีความแข็งแรง

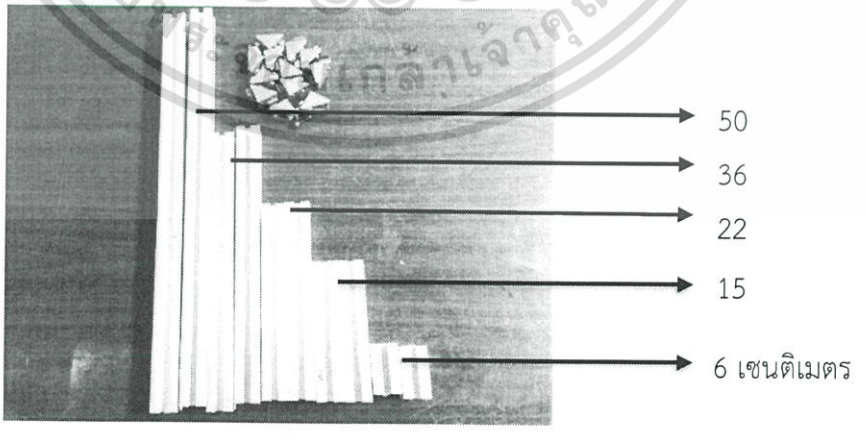


รูปที่ 3.3 Clamp Lock จากเครื่องพิมพ์ 3 มิติ

3.1.3 อลูมิเนียมโปรไฟล์

คุณสมบัติ

1. มีน้ำหนักเบา ไม่เป็นสนิม
2. ประยุกต์ได้กับงานที่หลากหลาย สามารถถอดหรือประกอบได้สะดวก สามารถรับแรงกดได้ดี



รูปที่ 3.4 อลูมิเนียมโปรไฟล์

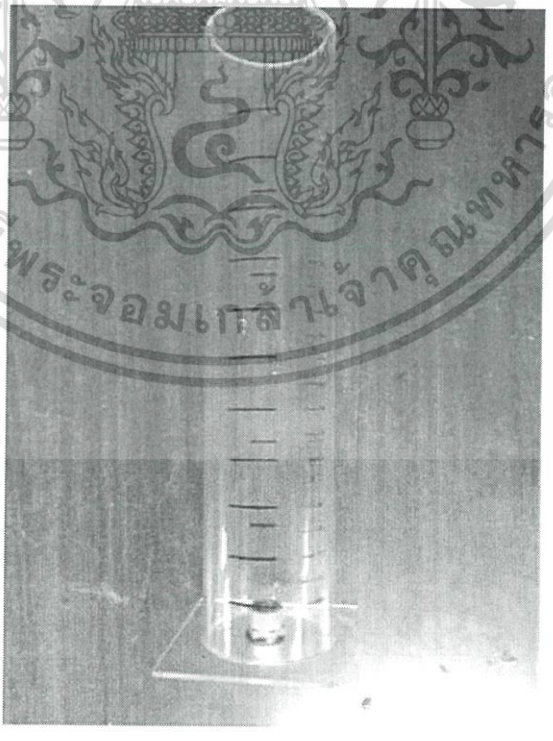
3.1.4 ถังพักน้ำและแทงค์สำหรับควบคุมระดับน้ำ

คุณสมบัติ

1. แทงค์สำหรับควบคุมระดับน้ำที่ใช้เป็นท่ออะคริลิคมีปริมาตรที่ 3,925 ลูกบาศก์เซนติเมตร

2. ถังพักน้ำสำหรับวางปั๊มมีปริมาตรเป็นสองเท่าของแทงค์สำหรับควบคุมระดับน้ำ ซึ่งมีขนาดเท่ากับ 7,850 ลูกบาศก์เซนติเมตร

รูปที่ 3.5 ถังพักน้ำ

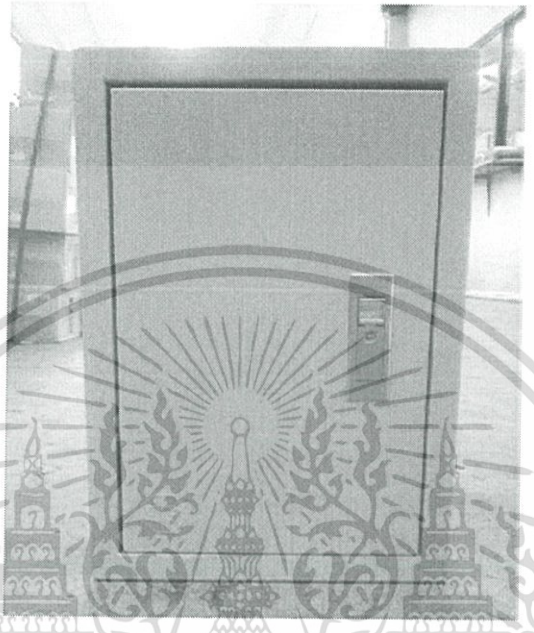


รูปที่ 3.6 แทงค์สำหรับควบคุมระดับน้ำ

3.1.5 ตู้สำหรับใส่แผงวงจร

คุณสมบัติ

1. สามารถระบายความร้อนได้
2. เคลื่อนย้ายได้สะดวก

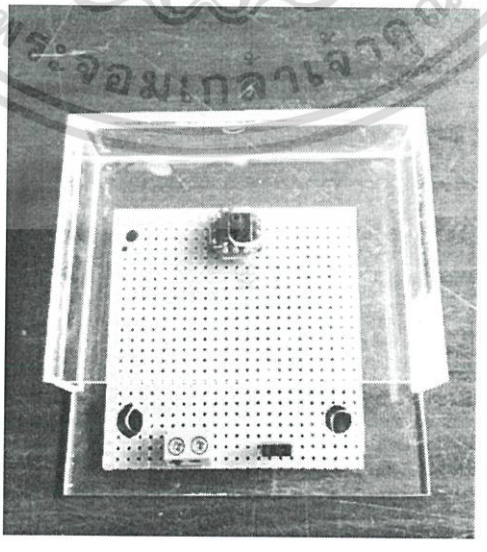


รูปที่ 3.7 ตู้สำหรับใส่แผงวงจร

3.1.6 กล่องสำหรับใส่ Level Transmitter

คุณสมบัติ

1. ป้องกันการกระเด็นของน้ำ



รูปที่ 3.8 กล่องสำหรับใส่ Level Transmitter

3.1.7 เซนเซอร์วัดระดับน้ำ

สำหรับโครงงานนี้ได้เลือกใช้เซนเซอร์วัดความดันน้ำ XGZP6847 Pressure Sensor Module เนื่องจากค่าแรงดันที่คำนวณได้จากการวัดที่ถังจริงของระบบ มีค่าสูงสุด 4.369 kpa ดังนั้นจึงเลือกใช้เซนเซอร์รุ่นนี้ที่มีช่วงแรงดันอยู่ที่ 0-10 kpa ซึ่งใกล้เคียงกับค่าจริง จะทำให้ค่าที่โปรแกรม Arduino อ่านค่าออกมาได้ค่าที่ละเอียด และมองเห็นได้

คุณสมบัติของเซนเซอร์

- Ranges: 0-10 kPa
- MEMS Technology
- Gauge
- DIP Package
- For Non-corrosive Gas or Liquid
- Amplified and Calibrated.
- Temp. Compensated : 0°C~+85°C(32°F~+185°F)
- Directly Application, Low Cost

รูปที่ 3.9 โมดูลวัดความดัน XGZP6847

3.1.8 เซนเซอรวัดอัตราการไหลของน้ำ (Flow Sensor)

โครงงานนี้ได้เลือกใช้เซนเซอรวัดอัตราการไหลของน้ำ (Electronic Flow Sensor Electronic Flow Meter) 1-30 ลิตรต่อนาที 1/2" เพราะสามารถต่อเข้าได้กับท่อขนาด 1/2"

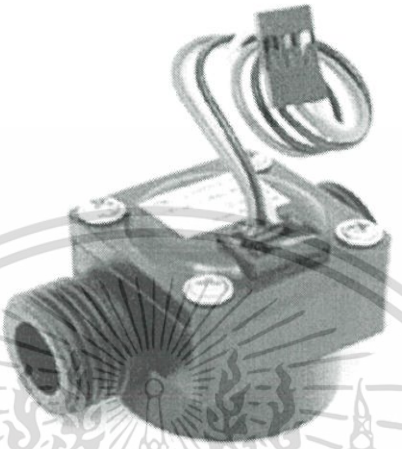
คุณสมบัติของ Flow Sensor

- The Level of The Test Pulse Frequency (Hz) = [8.1Q - 3] ± 10

(Q = Flow Rate L/min.)

- Minimum Rated Working Voltage : DC 3.5 , 5-24 V

- Maximum Current : 15 mA DC 5V
- Voltage Range : DC 5-18 V
- Operating Temp : ≤ 80 °C
- Allow Compression : Water Pressure 1.75 Mpa Below



รูปที่ 3.10 Flow Sensor

3.1.9 High-Power Motor Drive BTS7960 43A

คุณสมบัติของ High-Power Motor Drive

- Operating Voltage 5.5 to 27V (B+)
- Path Resistance of Typ. 16 mΩ at 25°C
- Low Quiescent Current of Typ. 7 μA at 25°C
- PWM Capability of up to 25 kHz Combined with Active Freewheeling
- Switched Mode Current Limitation for Reduced Power Dissipation in Overcurrent
- Current Limitation Level of 43 A Typ.
- Status Flag Diagnosis with Current Sense Capability
- Over Temperature Shut Down with Latch Behaviour
- Under Voltage Shut Down
- Driver Circuit with Logic Level Inputs
- Adjustable Slew Rates for Optimized EMI
- 74AHC244 Schmitt-trigger Octal Buffer/Line Driver for ESD Protection (Inputs Accepts Voltages Higher than VCC)

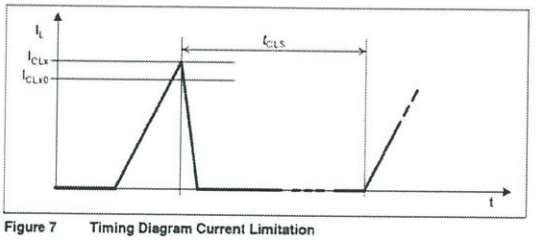


Figure 7 Timing Diagram Current Limitation

Thermal Shut Down							
4.3.10	Thermal shut down junction temperature	T_{jSD}	152	175	200	°C	–
4.3.11	Thermal switch on junction temperature	T_{jSO}	150	–	190	°C	–
4.3.12	Thermal hysteresis	ΔT	–	7	–	K	–
4.3.13	Reset pulse at INH pin (INH low)	t_{reset}	3	–	–	µs	–

4.3.6 Electrical Characteristics - Protection Functions

$-40\ °C < T_j < 150\ °C$; $8\ V < V_S < 18\ V$ (unless otherwise specified)

Pos.	Parameter	Symbol	Limit Values			Unit	Test Conditions
			min.	typ.	max.		
Under Voltage Shut Down							
4.3.1	Switch-ON voltage	$V_{UV(ON)}$	–	–	5.5	V	V_S increasing
4.3.2	Switch-OFF voltage	$V_{UV(OFF)}$	4.0	–	5.4	V	V_S decreasing
4.3.3	ON/OFF hysteresis	$V_{UV(HY)}$	–	0.2	–	V	–
Over Voltage Lock Out							
4.3.4	Switch-ON voltage	$V_{OV(ON)}$	27.5	–	–	V	V_S decreasing
4.3.5	Switch-OFF voltage	$V_{OV(OFF)}$	27.6	–	30	V	V_S increasing
4.3.6	ON/OFF hysteresis	$V_{OV(HY)}$	–	0.2	–	V	–
Current Limitation							
4.3.7	Current limitation detection level high side	I_{CLH0}	47 44 43	62 60 59	84 80 79	A	V_S=13.5 V T_j = -40 °C T_j = 25 °C T_j = 150 °C

รูปที่ 3.11 Datasheet ของชุดขับมอเตอร์ BTS7960 43A

โมดูลขับดีซีมอเตอร์แบบ Full Bridge ใช้ไอซีเบอร์ BTS7960 สำหรับขับดีซีมอเตอร์ที่ต้องการกระแสสูงๆ ใช้สัญญาณ PWM ในการควบคุมความเร็ว สามารถควบคุมมอเตอร์หมุนซ้ายขวาหรือกลับทางได้ และรองรับความเร็วของ PWM ได้ถึง 25 KHz แรงดันไฟเลี้ยงมอเตอร์ 6-27 Vdc กระแสเอาต์พุตสูงสูง 40A Max แต่ในทางปฏิบัติควรใช้กระแสไม่เกิน 20A เพื่อความปลอดภัยแรงดันอินพุต (PWM) สำหรับใช้ควบคุม 5 Vdc โดยไอซี BTS7960 มีระบบป้องกันดังต่อไปนี้

- Under Voltage Shutdown ถ้าแหล่งจ่ายไฟให้มอเตอร์ต่ำกว่า 5.5 V ไอซีจะหยุดทำงานจนกว่าแรงดันจะเพิ่มขึ้นเป็น 5.5V ขึ้นไป

- Over Temperature Protection ถ้าอุณหภูมิภายในตัวไอซีสูงเกินค่าที่กำหนดไว้ ไอซีจะหยุดทำงาน

- Current Limitation มีระบบป้องกันกระแสเกิน 43A

การต่อใช้งานทางด้านเอาต์พุต

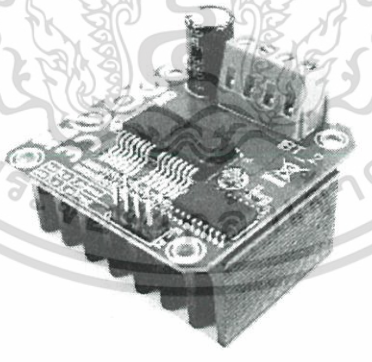
- ขา B+: ขั้วบวกแหล่งจ่ายไฟสำหรับมอเตอร์ (ใช้แรงดัน 6-27 Vdc)
- ขา B- : กราวด์ของแหล่งจ่ายไฟสำหรับมอเตอร์
- ขา M+: ขั้วบวกของมอเตอร์
- ขา M- : ขั้วลบของมอเตอร์

การต่อใช้งานทางด้านอินพุต (ขาควบคุม)

- VCC: +5V
- GND: GND
- R_IS และ L_IS จะเป็นขาเอาต์พุตแสดงสถานะผิดพลาด (Error Signal) กรณีที่กระแสทางเอาต์พุตไหลเกินหรือเกิดการลัดวงจร และตัว IC จะหยุดทำงานเสมอ
- R_EN และ L_EN จะเป็นขาควบคุม Enable (เปิดปิดการทำงานของ Output ทางขวาและซ้ายตามลำดับ): Active High (ต่อ 5V)
- RPWM และ LPWM เป็นขาอินพุตสำหรับต่อสัญญาณ PWM มาควบคุมความเร็วของมอเตอร์

หลักการทำงาน

- จะต้องต่อขา R_EN และ L_EN ด้วย 5V ไว้ (เป็นการ Enable เอาต์พุต)
- ถ้าต้องการให้มอเตอร์หมุนไปทางซ้าย จ่าย PWM (หรือ 5V) ไปที่ขา LPWM โดยที่ขา RPWM ให้ต่อกราวด์ (หรือจ่าย 0 V) ไว้
- ถ้าต้องการให้มอเตอร์หมุนไปทางขวา จ่าย PWM (หรือ 5V) ไปที่ขา RPWM โดยที่ขา LPWM ให้ต่อกราวด์ไว้ (หรือจ่าย 0V)
- ถ้าจ่าย +5V พร้อมกันไปที่ขา LPWM และ RPWM มอเตอร์จะหยุดหมุน



รูปที่ 3.12 High-Power Motor Drive BTS7960 43A

3.1.10 ปั๊มน้ำแบบจุ่ม/แช่

คุณสมบัติของปั๊มน้ำ

- Outlet dia: 25M/M • 1 inch
- Delivery Volume: 70 l/min • 1110gal/hour
- Delivery Head: 4m • 13ft

- Water Temp: 0°C ~ 60°C
- Power: DC-12V
- Current: 5.4A
- Rating: Continous
- Motor Power: 45W/5400rpm
- Weight: 1.3kg

รูปที่ 3.13 ปั๊มน้ำ Toshin Marine Pet รุ่น BL-2512S

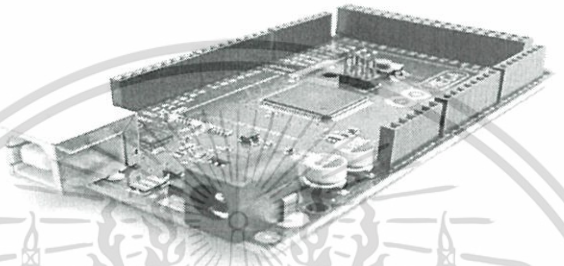
3.1.11 บอร์ด Arduino Mega 2560

เลือกใช้บอร์ด Arduino MEGA 2560 R3 เพราะเป็นอุปกรณ์ที่มีหน่วยประมวลผลและความจำขนาดเล็กภายในตัวเอง สามารถรับ-ส่ง ข้อมูลได้ทั้งแบบดิจิตอลและแอนะล็อก ใช้พลังงานน้อยทำให้เป็นที่นิยม และยังรองรับ Windows ทุกระบบใช้ได้กับทุก Windows xp, Windows 7, Windows 8 และ Windows 10

คุณสมบัติของบอร์ด

- Microcontroller	ATmega2560
- Operating Voltage	5V
- Input Voltage (Recommended)	7-12V
- Input Voltage (Limits)	6-20V
- Digital I/O Pins	54 (of which 14 provide PWM output, 4 UART TTL)
- Analog Input Pins	16
- DC Current per I/O Pin	40 mA
- DC Current for 3.3V Pin	50 mA

- Flash Memory — 256 KB of which 8 KB used by bootloader
- SRAM — 8 KB
- EEPROM — 4 KB
- Clock Speed — 16 MHz



รูปที่ 3.14 บอร์ด Arduino Mega 2560

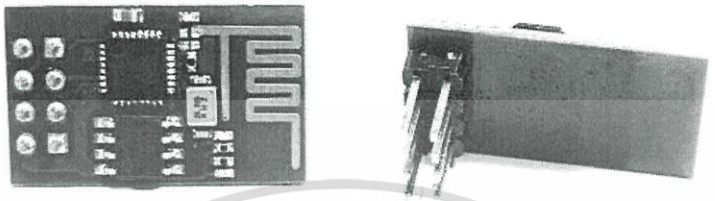
3.1.12 โมดูล Wi-Fi ESP8266

Wi-Fi ESP8266 ESP-01 เป็นโมดูล Wi-Fi ที่มีขนาดเล็ก ใช้พลังงานน้อย และรองรับการใช้งาน ได้หลากหลายรูปแบบทั้ง Client, Access Point และ Client+AP ESP8266 ใช้การเชื่อมต่อด้วย Serial (UART 3.3V) จึงทำให้ง่ายต่อการนำไปใช้งานร่วมกับไมโครคอนโทรลเลอร์ และยังมีรูปแบบคำสั่งแบบ AT Command ที่เชื่อมต่อ TCP/IP ได้ทันที ทำให้เขียนโปรแกรมเชื่อมต่อได้สะดวกยิ่งขึ้น

คุณสมบัติของโมดูล Wi-Fi

- 802.11 b/g/n
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP Protocol Stack
- Integrated TR Switch, Balun, LNA, Power Amplifier and Matching Network
- Integrated PLLs, Regulators, DCXO and Power Management Units
- +19.5dBm Output Power in 802.11b Mode
- Power Down Leakage Current of < C 10uA
- Integrated Low Power 32-bit CPU could be used as Application Processor
- SDIO 1.1/2.0, SPI, UART

- STBC, 1×1 MIMO, 2×1 MIMO
- A-MPDU & A-MSDU Aggregation & 0.4ms Guard Interval
- Wake up and Transmit Packets in < 2ms
- Standby Power Consumption of < 1.0mW (DTIM3)



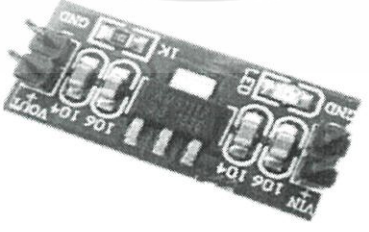
รูปที่ 3.15 โมดูล Wi-Fi ESP8266 ESP-01

3.1.13 Regulator Step Down

เนื่องจาก Wi-Fi Module ESP8266-01 ต้องใช้ไฟ 3.3V จึงต้องเลือกใช้ Regulator Step Down เพื่อแปลงไฟ 5V จากบอร์ด Arduino ให้เป็น 3.3V โดยเลือกใช้ AMS1117-3.3V Power Supply Module ซึ่งเป็นโมดูลแปลงไฟ 4.5 - 7V ไปเป็น 3.3V

คุณสมบัติของตัวแปลงไฟ

- Input: 4.5V dc - 7V (1 V Input Voltage Must be Higher Than to the Output Voltage Above)
- Output: 3.3V, 800 mA (Load Current Should not Exceed 800 ma)
- Size: 2.5 cm x 1.1 cm
- Power Indicator Light (red)



รูปที่ 3.16 โมดูล AMS1117-3.3V Power Supply

3.1.14 แหล่งจ่ายไฟ

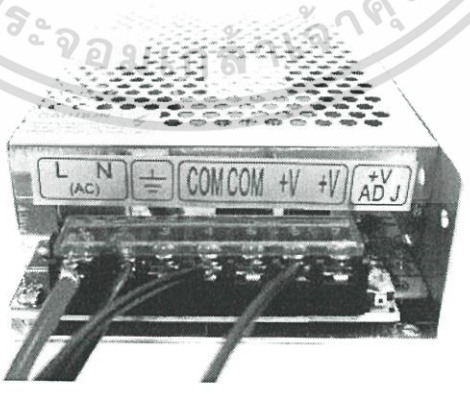
Adaptor อุปกรณ์จ่ายไฟขนาดเล็ก นำมาใช้ในโครงงานนี้เพื่อจ่ายแรงดันขนาด 12V ให้แก่บอร์ด Arduino Mega 2560

คุณสมบัติ

- MODEL : YX-4116C 12A
- INPUT : AC 220V 50Hz
- OUTPUT : DC 12V 500mA

รูปที่ 3.17 Adapter Switching Power Supply รุ่น DSA-0151A-12S

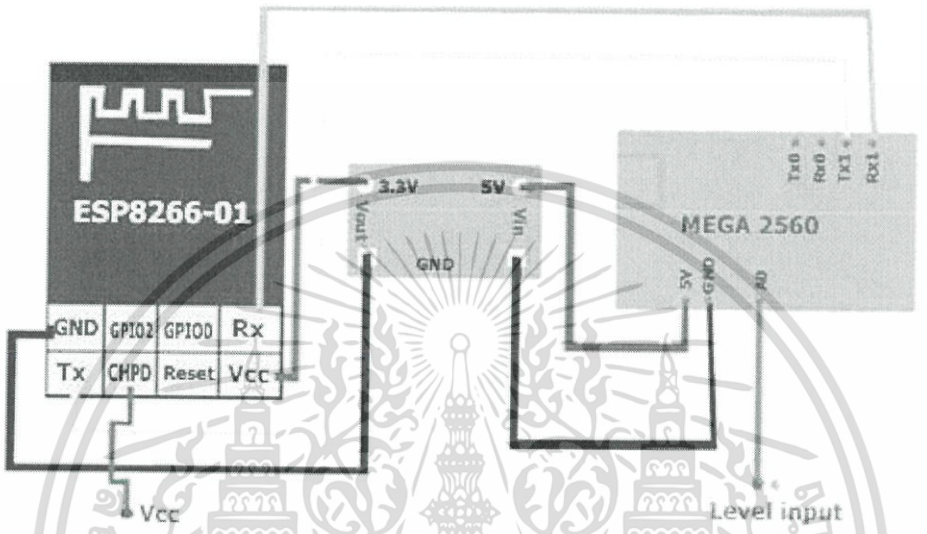
Switching Power Supply เป็นแหล่งจ่ายไฟตรงคงค่าแรงดันแบบหนึ่ง และสามารถเปลี่ยนแรงดันไฟจากไฟสลับโวลต์สูง ให้เป็นแรงดันไฟตรงค่าต่ำเพื่อใช้ในงานอิเล็กทรอนิกส์ได้ ซึ่งการนำมาใช้ในโครงงานนี้เพื่อจ่ายไฟ 12V 10A ให้กับไอซี LM358 ในวงจรบัฟเฟอร์



รูปที่ 3.18 Switching Power Supply 12V 10A 120W

3.2 การทดสอบการรับ - ส่งข้อมูลของ Wi-Fi ESP 8266

ทำการตรวจสอบว่าตัว Wi-Fi Module สามารถใช้งานได้จริง และรับ-ส่งข้อมูลได้ โดยใช้คำสั่ง AT Command ในการทดสอบ กำหนดให้ Wi-Fi เป็น Client และโปรแกรม Hercules เป็น TCP Server หลังจากนั้นต่อวงจรทดสอบ ESP8266 ระหว่างบอร์ด Arduino Mega 2560, Wi-Fi Module, Regulator และคอมพิวเตอร์ ดังรูปที่ 3.19



รูปที่ 3.19 วงจรทดสอบ ESP8266

การเชื่อมต่อผ่าน Serial Port โดยใช้บอร์ด Arduino

1. ต่อไฟเลี้ยง VCC, CH_PD และ GND ของโมดูล Wi-Fi เข้ากับไฟ 3.3V และ GND ของ Regulator ตามลำดับ

2. ต่อขาสัญญาณ RXD ของโมดูล Wi-Fi เข้ากับขา RXD ของ Arduino Mega 2560 R3

3. ต่อขาสัญญาณ TXD ของโมดูล Wi-Fi เข้ากับขา TXD ของ Arduino Mega 2560 R3

4. อัพโหลดโปรแกรมไฟกระพริบเข้าไปในบอร์ด เพื่อป้องกันไม่ให้ตัวไอซีใช้ช่องสัญญาณ Serial Port

5. เปิด Serial Monitor ตั้งค่า Baud Rate 115200 และปรับช่องในรูปให้เป็น Both NL&CR

6. พิมพ์ AT แล้วกด Send ถ้าขึ้นคำว่า OK แสดงว่าบอร์ดใช้งานได้ จึงทดสอบต่อไป

7. เชื่อมต่อ Router

- พิมพ์คำสั่ง AT+CWLAP เพื่อเชื่อมต่อเข้า Router ค้นหา Wi-Fi Hotspot ที่เปิดใช้งานอยู่
- พิมพ์ AT+CWJAP="ECC-208", "profinet" เพื่อเชื่อมต่อเข้ากับ Router
- พิมพ์ AT+CWJAP? เพื่อตรวจสอบว่าสามารถเชื่อมต่อได้หรือไม่ เมื่อเชื่อมต่อเข้ากับ

Router ได้ Router จะให้ไอพีมา และทุกครั้งที่เปิดมาจะเชื่อมต่อกับ Router เองอัตโนมัติ

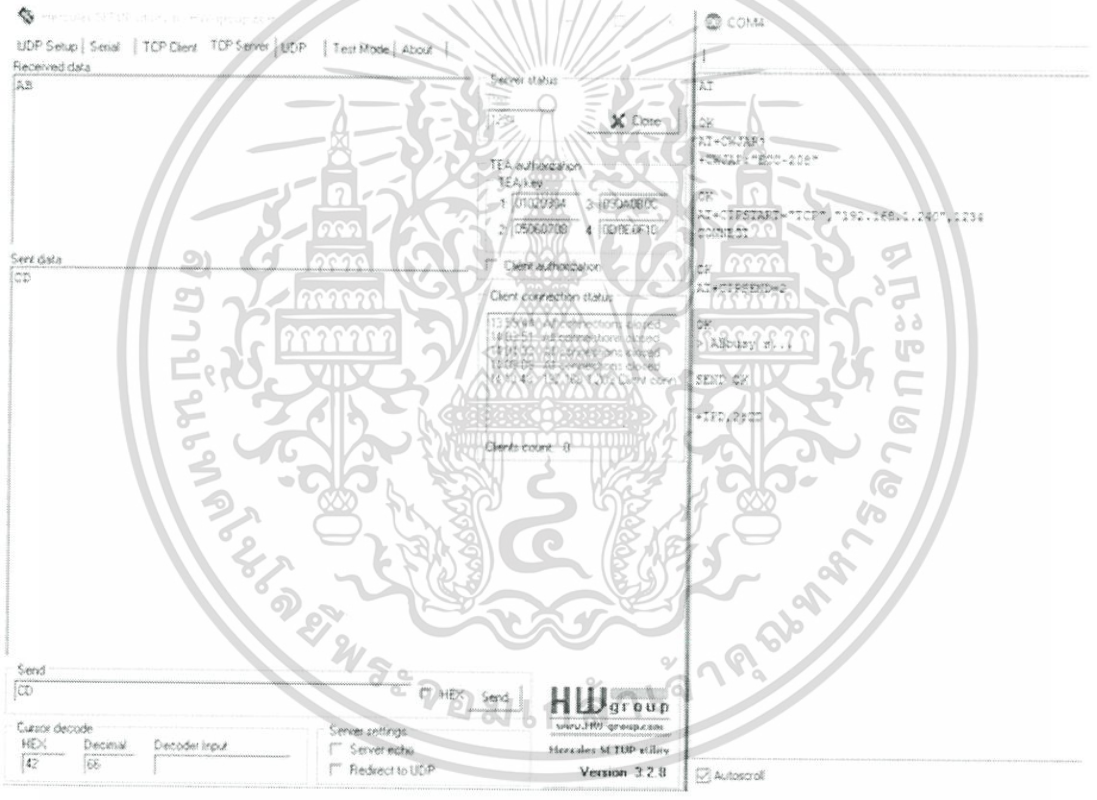
8. Setup ให้ Wi-Fi ESP8266 เป็น TCP Client

- พิมพ์ AT+CIPMODE=1 คำสั่งนี้จะเป็นการเลือกหมวดการทำงานของโมดูล ซึ่งสามารถทำงานได้ 3 โหมด 1 STA, 2 AP และ 3 AP+STA

- พิมพ์ AT+CIPMUX=0 เพื่อเปิดโหมดการเชื่อมต่อแบบหลายจุด

- ทำการเชื่อมต่อกับ Server โดยใช้โปรแกรม Hercules จำลองเป็น TCP Server, IP 192.168.1.240 และ Port 1234 พิมพ์ AT+CIPSTART="TCP", "192.168.1.240", 1234

9. พิมพ์ AT+CIPSEND=2 เพื่อสั่งว่าจะส่งข้อมูลจำนวน 2 Byte หลังจากกด Enter คำสั่งจะมีเครื่องหมาย ">" ให้พิมพ์ข้อมูลที่จะส่ง



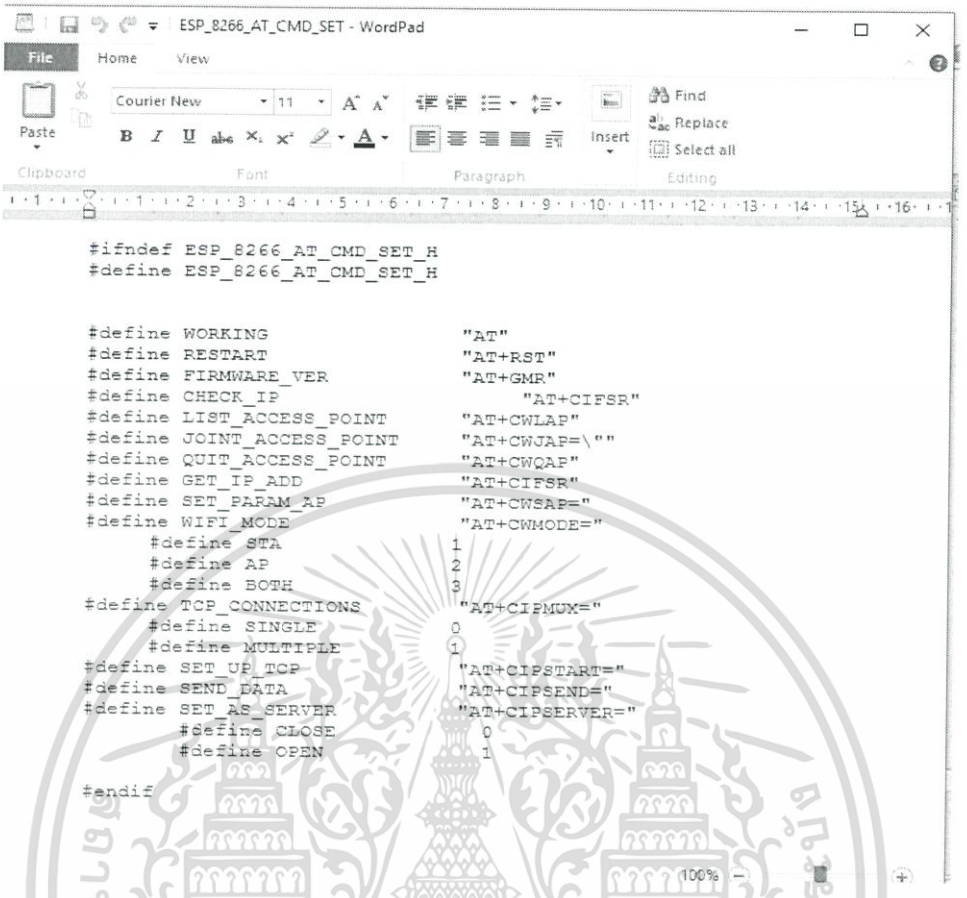
รูปที่ 3.20 ทดสอบการรับส่งข้อมูลของโมดูล Wi-Fi

3.3 การเขียน Library สำหรับ Wi-Fi ESP8266

เขียน ESP_8266 Library เพื่อใช้งาน Wi-Fi Module ESP8266 โดยใช้ Software Serial ทำให้สามารถใช้งานกับ Arduino Mega 2560 R3 ได้ และต้องใช้กับ Wi-Fi ESP8266 ที่ใช้ Firmware Version 0.9.2.2 เป็นต้นไป เนื่องจากเป็น Version ที่มีการพัฒนาให้สามารถปรับค่า

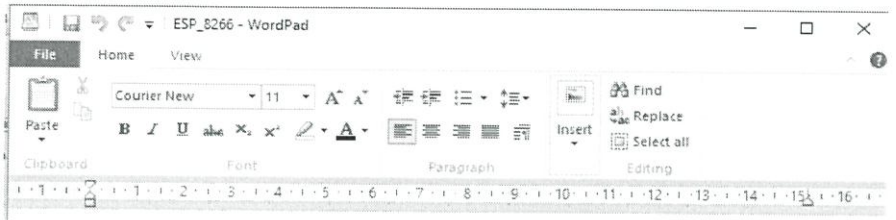
Baud Rate ให้เหมาะสมกับ Software Serial ของ Arduino ซึ่งได้เขียนให้มีความสอดคล้องกับการใช้งานของโครงงานนี้ มีฟังก์ชันต่างๆ ดังนี้

- ESP_8266(long baud) ใช้เพื่อกำหนดความเร็วในการรับ-ส่งข้อมูลระหว่างบอร์ด Arduino และ Wi-Fi ESP8266
- esp.restartESP() ใช้เพื่อ Restart โมดูล
- esp.firmwareESP() ใช้เพื่อตรวจสอบ Firmware ของโมดูลว่าอยู่ Version ไหน
- esp.listAP() ใช้เพื่อสั่งให้โมดูล ESP8266 แสดงรายชื่อ Access Point ที่สแกนพบ
- esp.connectAP(String SSID,String PASS) ใช้เพื่อเชื่อมต่อ ESP8266 เข้ากับ Access Point โดยกำหนดชื่อของ SSID และ Password ให้ตรงกับ Access Point ที่จะเชื่อมต่อ
- esp.disconnectAP() ใช้เพื่อหยุดการเชื่อมต่อ ESP8266 เข้ากับ Access Point
- esp.espIP() ใช้เพื่อแสดง IP ของโมดูล Wi-Fi
- esp.setmode(String MODE) ใช้เพื่อเลือกโหมดการทำงานให้กับโมดูล ESP8266 ซึ่งกำหนดได้ 3 โหมด คือ STA, AP, BOTH
- esp.connectTCP(String TCP) ใช้เพื่อตั้งค่าการเชื่อมต่อ TCP/UDP โมดูล ESP8266 กำหนดได้ 2 โหมด คือ Single และ Multiple
- esp.startClient(String TYPE, String ADDR, String PORT) ใช้เพื่อกำหนดค่าเริ่มต้นการใช้งาน Client Mode ในกรณีนี้กำหนด esp.connectTCP(String TCP) เป็น Single
- esp.startServer(String STATUS, String PORT) ใช้กำหนดค่าเริ่มต้นการใช้งาน Server Mode
- esp.disconnectSV() ใช้หยุดการเชื่อมต่อในกรณี TCP และ UDP
- esp.sendDT(String DATA) ใช้ส่งข้อมูลแบบ String ออกไปทาง TCP และ UDP ในกรณีนี้กำหนด esp.connectTCP(String TCP) เป็น SINGLE
- esp.startSendDATA(int length) ใช้งานร่วมกับ esp.sendDT(String DATA) แต่ฟังก์ชันนี้สามารถกำหนดความยาวของข้อมูลที่จะส่งได้




```
#ifndef ESP_8266_AT_CMD_SET_H
#define ESP_8266_AT_CMD_SET_H


#define WORKING                 "AT"
#define RESTART                 "AT+RST"
#define FIRMWARE_VER            "AT+GMR"
#define CHECK_IP                    "AT+CIFSR"
#define LIST_ACCESS_POINT       "AT+CWLAP"
#define JOINT_ACCESS_POINT      "AT+CWJAP=\""
#define QUIT_ACCESS_POINT       "AT+CWQAP"
#define GET_IP_ADD              "AT+CIFSR"
#define SET_PARAM_AP            "AT+CWSAP="
#define WIFI_MODE               "AT+CWMODE="
      #define STA               1
      #define AP                2
      #define BOTH              3
#define TCP_CONNECTIONS         "AT+CIPMUX="
      #define SINGLE            0
      #define MULTIPLE          1
#define SET_UP_TCP              "AT+CIPSTART="
#define SEND_DATA               "AT+CIPSEND="
#define SET_AS_SERVER           "AT+CIPSERVER="
       #define CLOSE             0
       #define OPEN              1

#endif
```

รูปที่ 3.21 ESP_8266 Library ส่วน AT Command




```
#ifndef ESP_8266_h
#define ESP_8266_h
#if ARDUINO >= 100
#include "Arduino.h"
#else
#include "WProgram.h"
#endif


#include "ESP_8266_AT_CMD_SET.h"


class ESP_8266
{
public :
    ESP_8266();
    ESP_8266(long baud);
    ~ESP_8266();
    void connectAP(String SSID,String PASS);
    void disconnectAP(String SSID,String PASS);
    void checkIP();
    void setmode(String MODE);
    void connectTCP(String TCP);
    void connectSV (String TYPE,String ADDR,String PORT);
    void sendDT(String DATA);
    void startSendDATA(int length);
    void startServer(String STATUS,String PORT);

};
#endif
```

รูปที่ 3.22 ESP_8266 Library ส่วน Header File


```
#include "ESP_8266.h"
ESP_8266::ESP_8266(){}
ESP_8266::ESP_8266(long baud)
{
   Serial1.begin(baud);
   Serial.begin(baud);
}
ESP_8266::~ESP_8266()
{
   Serial1.print("AT+CIPCLOSE");
   delay(1000);
   while (Serial1.available()) {
        Serial.write(Serial1.read());

    }

}
void ESP_8266::connectAP(String SSID,String PASS)
{
    Serial.println("Connecting to access point .....");
    String CMD = JOINT_ACCESS_POINT ;
           CMD += SSID ;
           CMD += "\",\"" ;
           CMD += PASS ;
           CMD += "\"\r\n" ;
    Serial1.print(CMD);
    delay(2000);
 while (Serial1.available()) {
        Serial.write(Serial1.read());
```

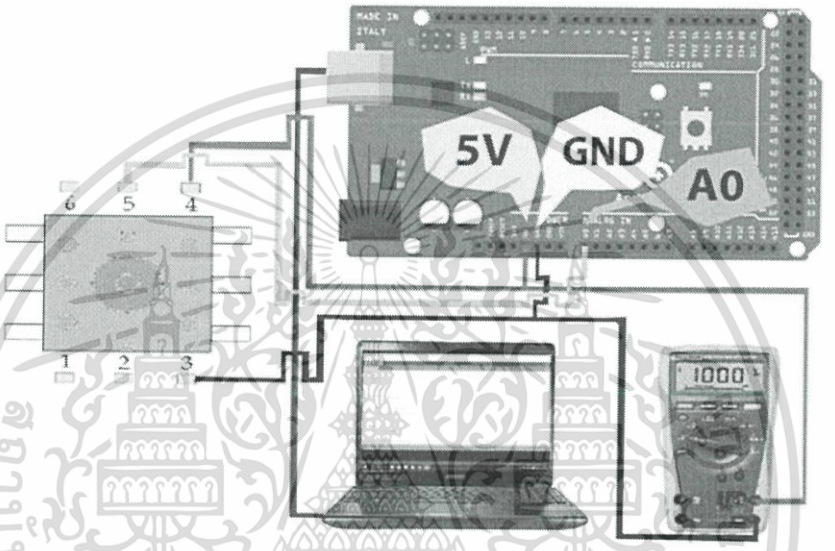
รูปที่ 3.23 ESP_8266 Library ส่วน Source File

3.4 การทดสอบส่วนการวัดระดับ

3.4.1 ทดสอบการใช้งาน Pressure Sensor

1. ทำการศึกษาข้อมูล รายละเอียดการทำงาน การทดสอบ และคุณสมบัติของเซนเซอร์

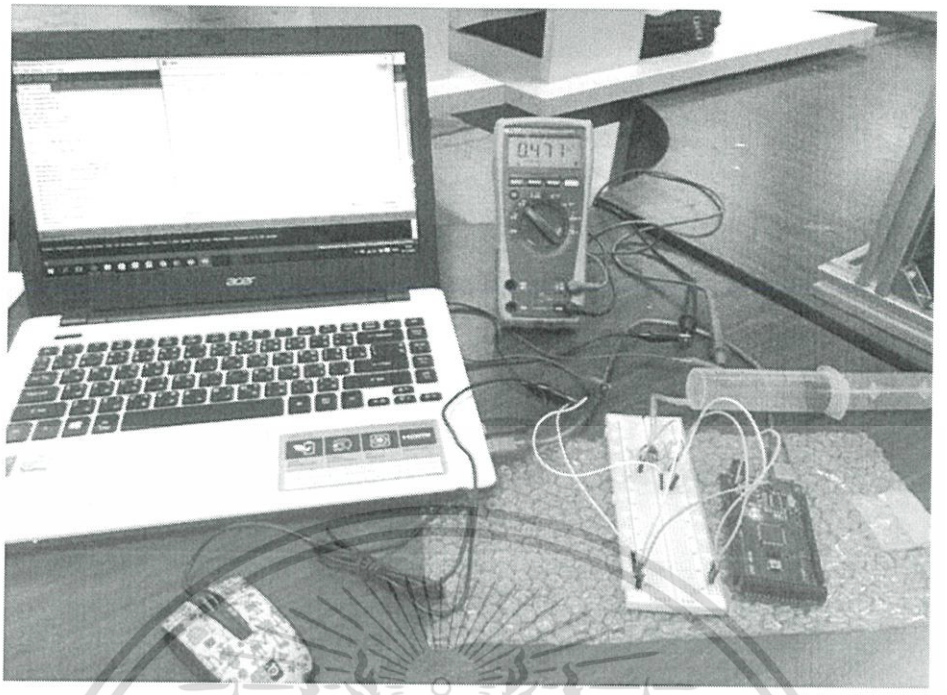
2. ทำการทดสอบเซนเซอร์โดยต่อวงจรทดสอบ Pressure Sensor ดังรูปที่ 3.24 โดยใช้หลอดฉีดยาอัดแรงดันลมเข้าไปในเซนเซอร์ เพื่อดูการเปลี่ยนแปลงค่าแรงดันที่ช่วงต่างๆ กัน ตามสเกลบนหลอดฉีดยา



รูปที่ 3.24 วงจรทดสอบ Pressure Sensor

3. เปิดโปรแกรม Arduino เขียนโปรแกรมเพื่ออ่านค่า Pressure ที่วัดได้จากถัง ซึ่งจะได้ค่าต่ำสุด-สูงสุดอยู่ที่ช่วงๆ หนึ่ง นำค่าที่ได้มาแปลงให้อยู่ในช่วง 0-100% โดยการใช้คำสั่ง Map แล้วนำค่าเปอร์เซ็นต์ที่ได้มาใช้คำสั่ง Map อีกครั้งเพื่อทำให้ค่าออกมาเป็นแรงดันในหน่วยมิลลิโวลต์ อัพโหลดคำสั่งที่ใช้ในการทดลองลงบอร์ด Arduino Mega 2560

4. เปิดหน้าต่าง Serial Monitor เพื่อดูค่าเอาต์พุตที่ได้ และสังเกตค่าที่ได้จาก 2 ทาง คือค่าจากมิเตอร์ รวมถึงค่าจากโปรแกรม Arduino ดังรูปที่ 3.25



รูปที่ 3.25 การทดสอบอ่านค่าของ Pressure Sensor จากมิเตอร์และโปรแกรม Arduino

ทดสอบหาค่าความคลาดเคลื่อนระหว่างค่าแรงดันจากมิเตอร์ และจากโปรแกรม Arduino

1.ต่อเซนเซอร์เข้ากับถังที่ต้องการควบคุมระดับน้ำ และต่อวงจรทดสอบดังรูปที่ 3.24

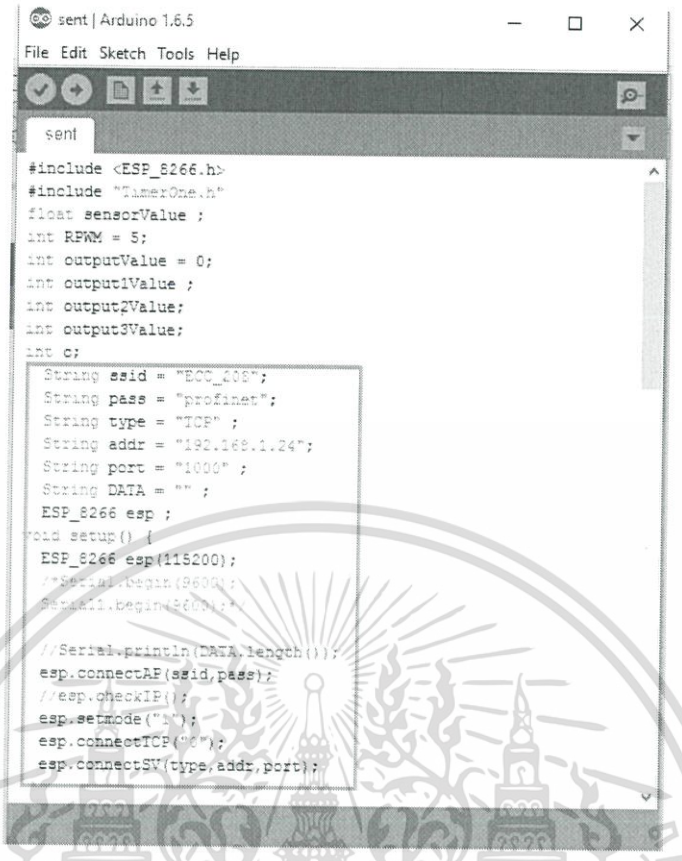
2.เปิดโปรแกรม Arduino อัพโหลดโปรแกรมที่ใช้ในการทดลองลงบอร์ด Arduino Mega 2560 และเปิดหน้าต่าง Serial Monitor เพื่อดูค่าแรงดันที่เกิดขึ้น จากนั้นเติมน้ำลงในถังครั้งละ 0.5 ลิตร ตั้งแต่ที่ระดับน้ำ 0-3.5 ลิตร

3.ทำการบันทึกค่าแรงดันที่ได้จากมิเตอร์ และจากโปรแกรม Arduino เพื่อนำค่าที่ได้มาเปรียบเทียบหาค่าความคลาดเคลื่อนที่เกิดขึ้น โดยค่าที่แสดงออกมาที่โปรแกรมนั้นคือค่าแรงดันของเซนเซอร์ และค่าเปอร์เซ็นต์ของระดับน้ำที่อยู่ในถัง

3.4.2 การทดสอบเขียนโปรแกรมส่งค่า

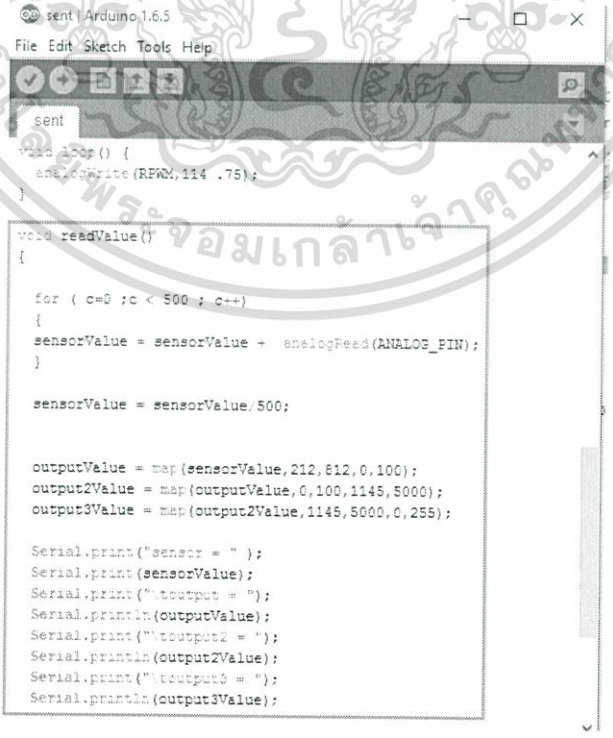
ต่อวงจรเพื่อส่งค่าดังรูปที่ 3.19 โดยโปรแกรมจะแบ่งออกเป็น 3 ส่วน ดังนี้

1. ส่วนการกำหนดรูปแบบการทำงานของโมดูล Wi-Fi ESP8266 ให้เป็น Client โดยนำฟังก์ชันที่เขียนใน ESP8266 Library มาใช้ ซึ่งจะกำหนด IP Address และ Port ที่จำลองเป็น Server โดยคอมพิวเตอร์เครื่องที่จะจำลองเป็น Server จะต้องเชื่อมต่อ Access Point เดียวกันกับ Wi-Fi Module ESP8266 จึงสามารถเชื่อมต่อกันได้



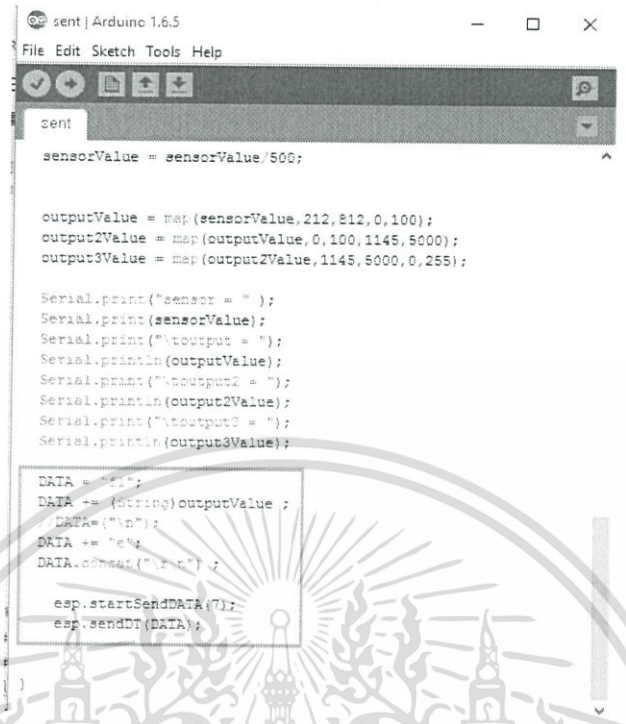
รูปที่ 3.26 โปรแกรมส่วนกำหนดรูปแบบการทำงานของ Wi-Fi Module

2. ส่วนอ่านค่าของเซนเซอร์ และแสดงค่าเอาต์พุตออกมาทาง Serial Monitor



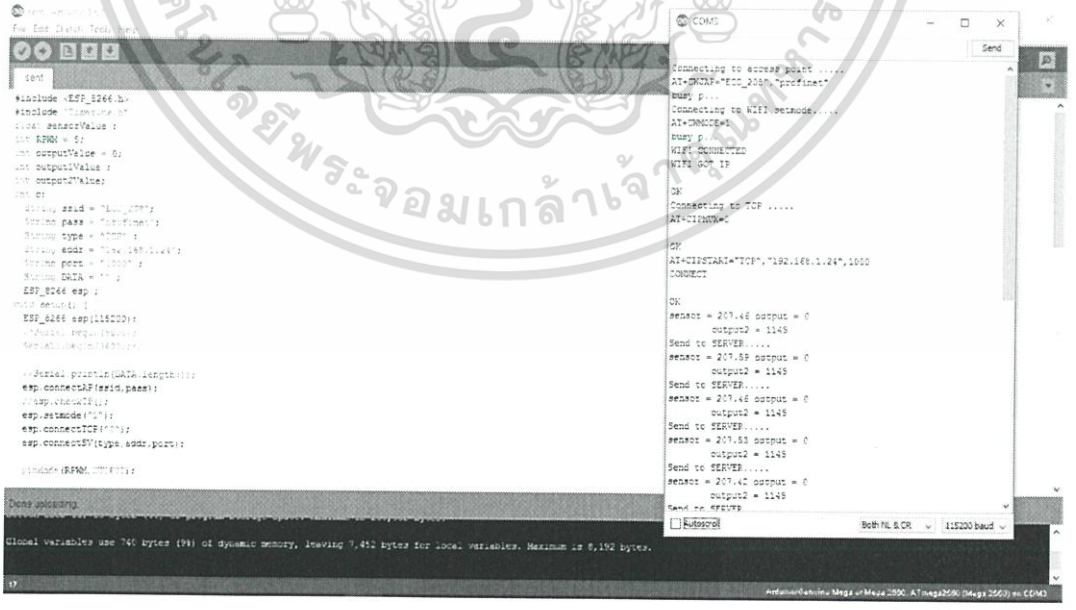
รูปที่ 3.27 โปรแกรมส่วนอ่านค่าเซนเซอร์ และแสดงค่าเอาต์พุต

3. ส่วนส่งค่าเปอร์เซ็นต์ของระดับน้ำในถัง



รูปที่ 3.28 โปรแกรมส่วนส่งค่าเปอร์เซ็นต์ของระดับน้ำในถัง

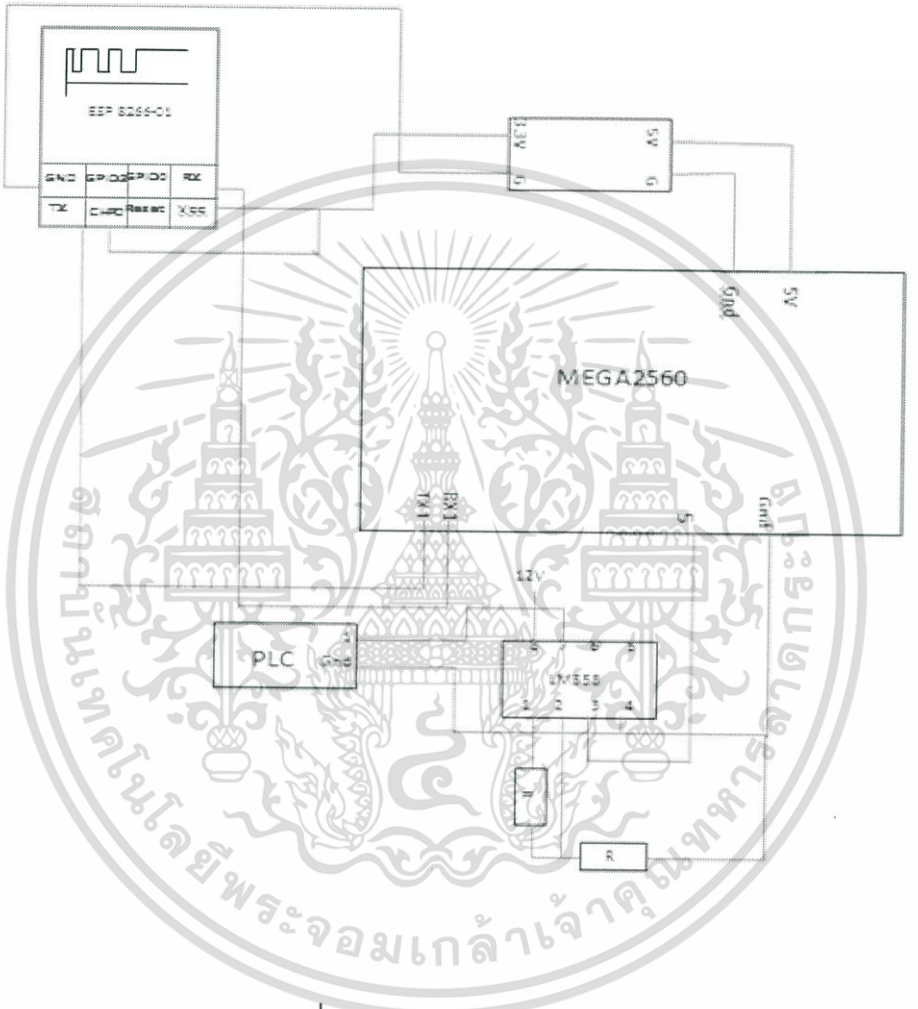
อัพโหลดคำสั่งจากข้อ 2 ลงบอร์ด และเปิดหน้าต่าง Serial Monitor เพื่อดูค่าเอาต์พุตที่เกิดขึ้น และตรวจสอบการเชื่อมต่อระหว่างโมดูล Wi-Fi ตัวที่เป็น Client และ Server ว่าเชื่อมต่อกันได้หรือไม่ ซึ่งหากเชื่อมต่อได้จะขึ้นคำว่า CONNECT



รูปที่ 3.29 หน้าต่างโปรแกรมส่งข้อมูล Wi-Fi

3.4.3 การทดสอบเขียนโปรแกรมรับค่า

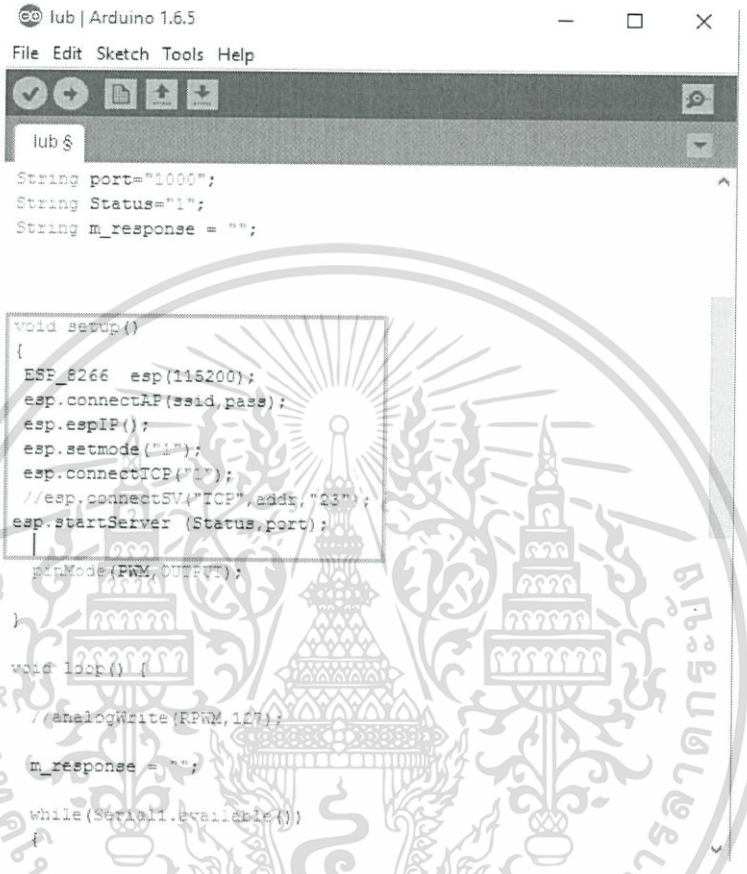
ต่อวงจร Wi-Fi กับวงจรบัฟเฟอร์เข้ากับ PLC โดยต้องต่อวงจรบัฟเฟอร์ให้ขยายแรงดันจากบอร์ด Arduino ในช่วง 0-5V ให้เป็น 0-10V เพื่อให้ PLC สามารถอ่านค่าได้โดยเกิดความคลาดเคลื่อนน้อยมาก ดังรูปที่ 3.30



รูปที่ 3.30 วงจรรับค่าโมดูล W-Fi

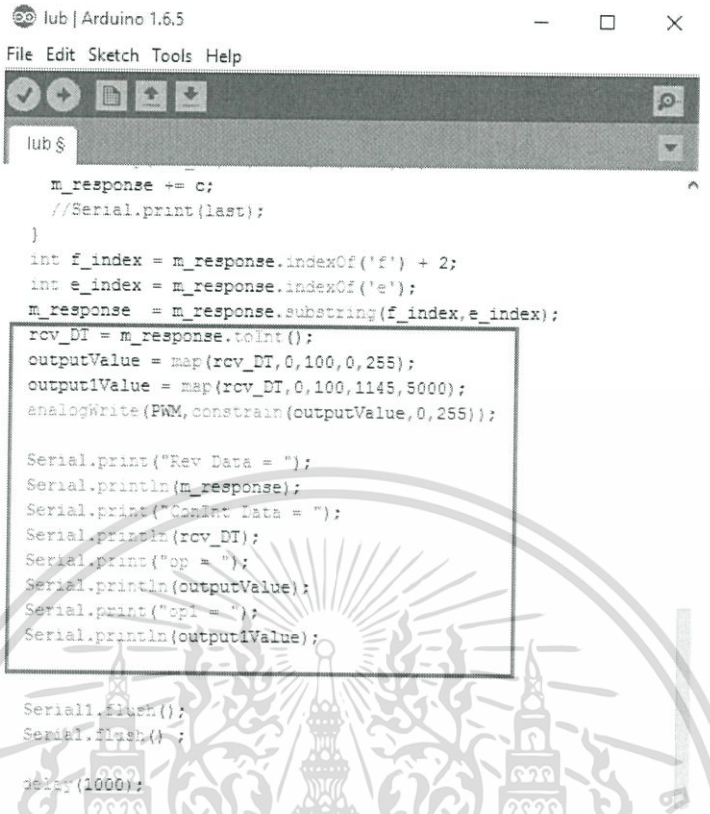
โปรแกรมจะแบ่งออกเป็น 3 ส่วน ดังนี้

1. ส่วนการกำหนดรูปแบบการทำงานของ โมดูล Wi-Fi ESP8266 ให้เป็น Server โดยนำฟังก์ชันที่เขียนใน ESP8266 Library มาใช้งาน



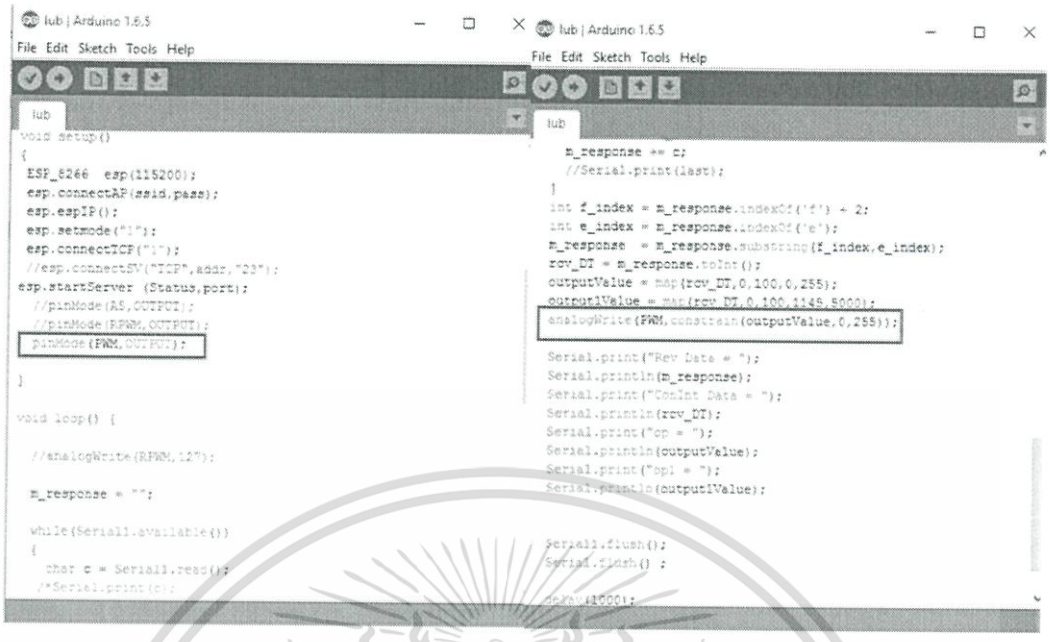
รูปที่ 3.31 โปรแกรมส่วนกำหนดรูปแบบการทำงานของโมดูล Wi-Fi

2. ส่วนรับค่าเปอร์เซ็นต์ และแสดงค่าของระดับน้ำในถัง โดยนำค่าที่ได้มาเก็บไว้ในตัวแปร m_response จากนั้นทำการแปลงค่าเปอร์เซ็นต์ที่รับมาใช้คำสั่ง map แปลงเป็นค่า 0-255



รูปที่ 3.32 โปรแกรมส่วนรับ และแสดงค่าเปอร์เซ็นต์ของระดับน้ำในถัง

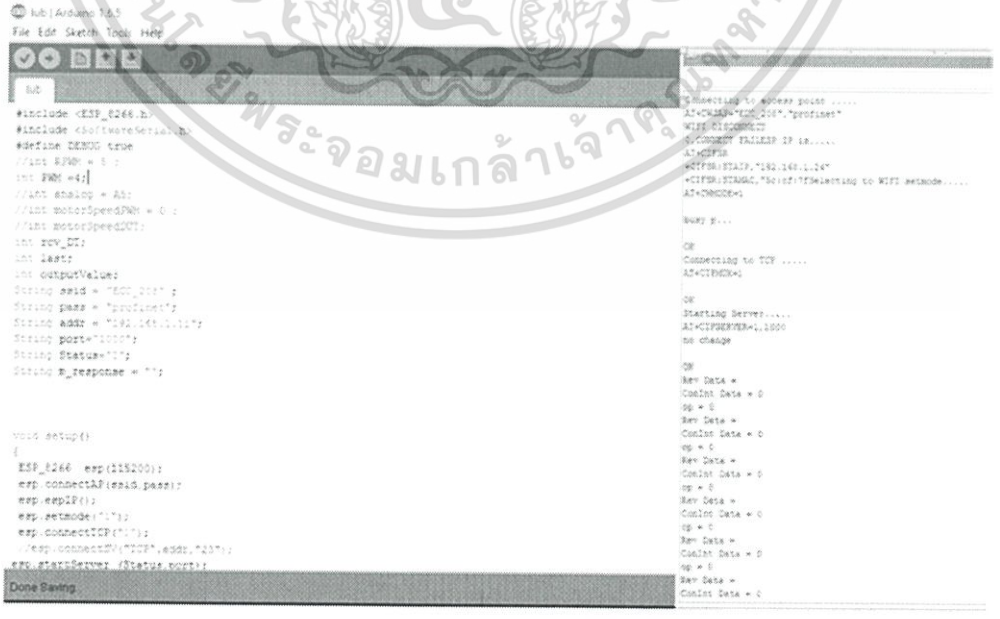
3. ส่วนส่งค่าแรงดัน โดยกำหนดให้ขา PWM ที่บอร์ด Arduino เป็นขาเอาต์พุต ส่งค่าแรงดันออกไปให้กับ PLC โดยสัญญาณ Digital ที่ PWM จะถูกแปลงผ่านตัวแปลงในบอร์ด Arduino ให้เป็นสัญญาณแอนะล็อกเข้าวงจรขยายแรงดันเพื่อขยายแรงดัน แล้วเชื่อมต่อเข้ากับ PLC ซึ่งจะรับอินพุตเข้าเป็นสัญญาณแอนะล็อกเท่านั้น โดย PLC จะมีตัวกรองสัญญาณแบบ RC ทำหน้าที่เฉลี่ยค่าแอนะ-ล็อกที่ได้รับเข้ามา ซึ่งจะมีค่าประมาณ 2 เท่าของค่าที่ออกมาจากขา PWM ที่บอร์ด Arduino



รูปที่ 3.33 โปรแกรมส่วนส่งค่าแรงดันให้กับ PLC

อัพโหลดโปรแกรมลงบอร์ด และเปิดหน้าต่าง Serial Monitor เพื่อดูค่าเอาต์พุตและสามารถตรวจสอบได้ว่าโมดูล Wi-Fi ตัวที่เป็น Client และ Server ว่าสามารถเชื่อมต่อกันได้หรือไม่ หากเชื่อมต่อได้จะขึ้นคำว่า CONNECT

บันทึกค่าแรงดันเอาต์พุตที่อ่านได้ จาก Serial Monitor และที่วัดจากขา PWM และ PLC เพื่อดูค่าที่ออกมาเปรียบเทียบดูความแตกต่างของแรงดันที่เกิดขึ้น



รูปที่ 3.34 หน้าต่างโปรแกรมรับข้อมูลจาก Wi-Fi

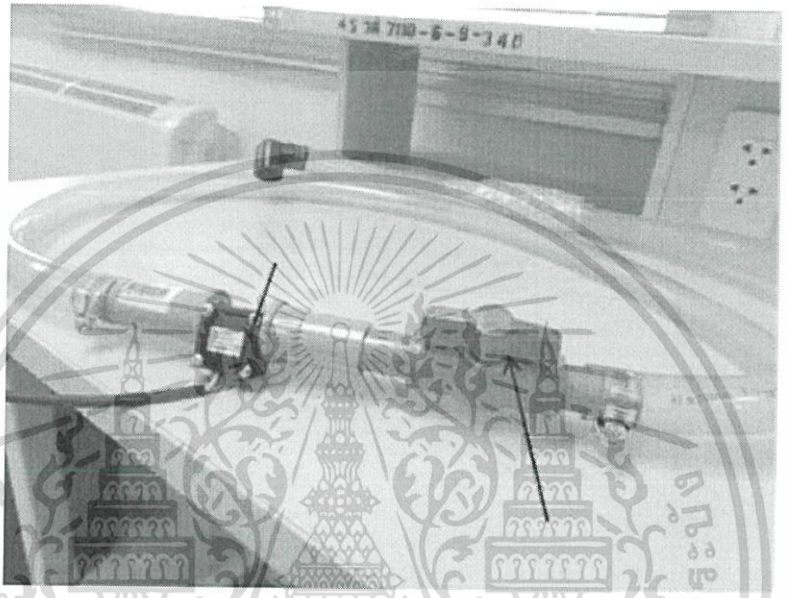
3.5 การทดสอบส่วนการวัดอัตราการไหล

3.5.1 การทดสอบการใช้งาน Flow sensor

3.5.1.1 การหาค่าความคลาดเคลื่อนของอัตราการไหลจากโปรแกรมกับค่าที่วัดได้จริง

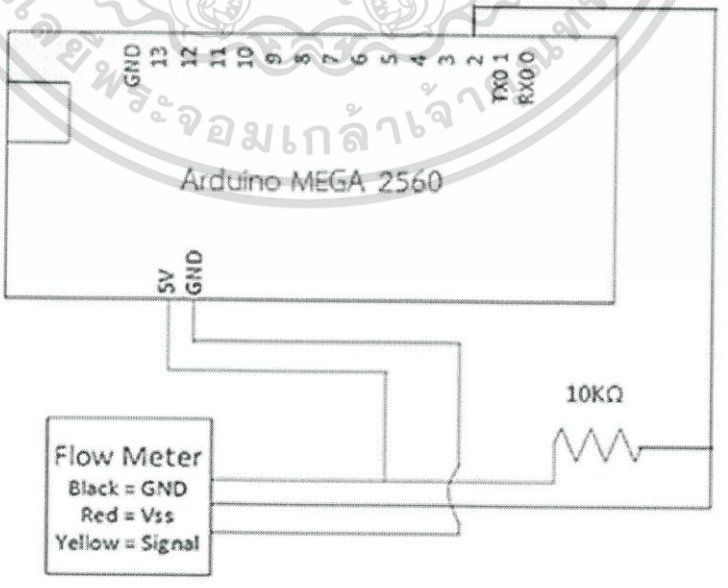
หลังจากที่ศึกษาข้อมูลต่างๆที่เกี่ยวข้องกับ Flow Sensor เพื่อใช้วัดอัตราการไหลของน้ำในท่อ เลือกใช้ Flow Sensor แบบ Turbine Flow Meter ซึ่งขั้นตอนในการทดลองดังนี้

1. ต่อ Flow Sensor, วาล์ว และสายยางดังรูปที่ 3.35



รูปที่ 3.35 การต่ออุปกรณ์

2. ต่อวงจรตามรูปที่ 3.36



รูปที่ 3.36 วงจรทดสอบ Flow Sensor

3. เขียน Code Arduino เพื่อคำนวณค่า Flow Rate ออกมาเป็น L/min และเทียบเป็นเปอร์เซ็นต์

4. ต่อบอรด Arduino กับคอมพิวเตอร์แล้ว Upload Code ลงบอร์ด Arduino

5. ต่อสายยางด้านที่ติดกับวาล์วเข้ากับปั๊มน้ำ

6. ใส่น้ำในถัง 1 ใบแล้วจุ่มปั๊มน้ำลงในถัง โดยสายยางอีกด้านให้ใส่ไว้ในถังใบเดียวกัน

7. ต่อ Supply 12 V กับปั๊มน้ำ

8. ไล่อากาศในสายยางโดยเปิดวาล์วให้น้ำไหลจนเต็มพื้นที่สายยาง

9. นำสายยางด้านที่ไม่ได้ต่อปั๊มไปใส่ในถังเปล่าอีก 1 ใบ

10. เปิดปั๊มน้ำแล้วจึงเปิดวาล์วพร้อม ๆ กับจับเวลา 6 วินาที

11. ใช้ถ้วยตวงวัดปริมาณน้ำในถังที่ผ่าน Flow Sensor แล้วคูณด้วย 10 เพื่อหาค่า Flow Rate ในหน่วย L/min แล้วบันทึกผล

12. อ่านค่า Flow Rate ใน Serial Monitor แล้วบันทึกผล

13. ทำซ้ำตั้งแต่ข้อ 10-12 จำนวน 10 ครั้ง แล้วหาค่าผิดพลาด (Error) โดย

$$\text{Flow Rate} = \frac{\sum(\text{ค่าเฉลี่ย} - \text{ค่าจริง})}{\text{จำนวนครั้งที่ทดลอง}} \quad (3.1)$$

$$\text{ค่าเฉลี่ยหา} = \frac{\sum \text{ค่าจริง}}{\text{จำนวนครั้งที่ทดลอง}} \quad (3.2)$$

14. ทำซ้ำตั้งแต่ข้อ 10-13 แต่เปลี่ยนเวลาในการวัดเป็น 1 นาที (หา Flow Rate จากปริมาณน้ำ โดยปริมาณน้ำที่วัดได้เทียบกับเวลาคือค่า Flow Rate)

3.5.1.2 การหาค่าอัตราการไหลเปรียบเทียบกับเปอร์เซ็นต์การทำงานของปั๊ม

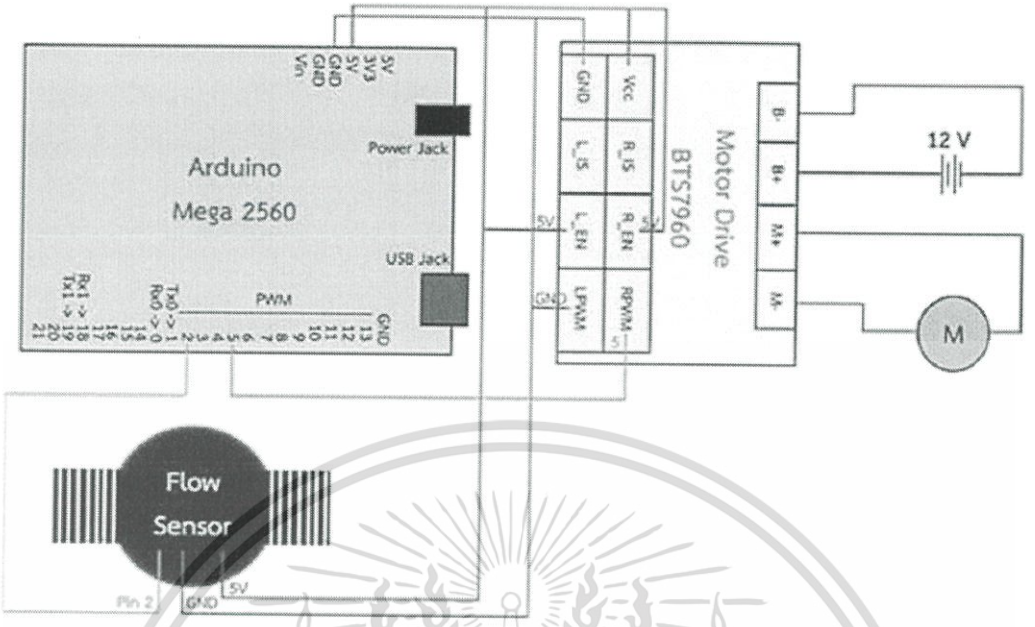
1. ต่อวงจร ดังรูปที่ 3.37

2. ใช้คำสั่ง map เทียบค่าจากเปอร์เซ็นต์ Flow Rate ให้เป็นแรงดันดัน

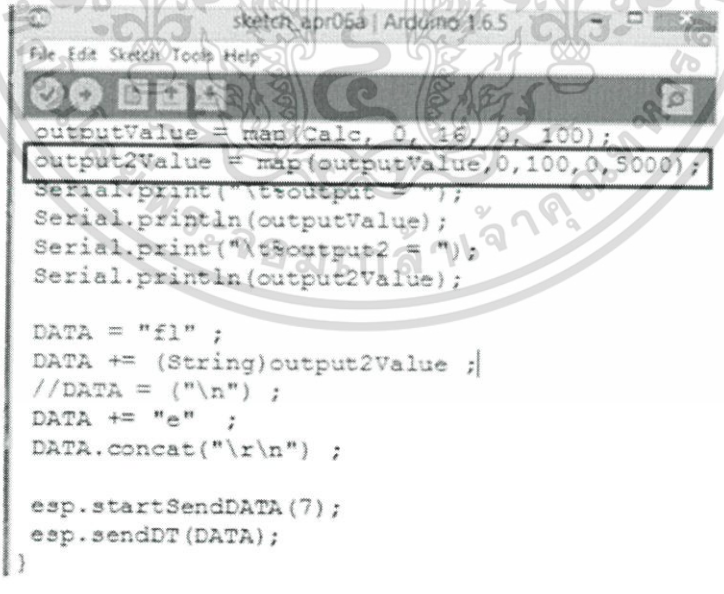
3. ทำการเพิ่มคำสั่งปรับ Speed Motor ลงในโปรแกรมที่ใช้ในการทดลอง ในหัวข้อที่ 3.5

4. กำหนดค่า Speed Motor เริ่มตั้งแต่ 45% ซึ่งเป็นค่าที่ Motor สามารถปั๊มน้ำขึ้นไปถึง Flow Sensor ได้ จากนั้นเปลี่ยนค่าไปเรื่อยๆ ทีละ 5% จนกระทั่งถึง 100%

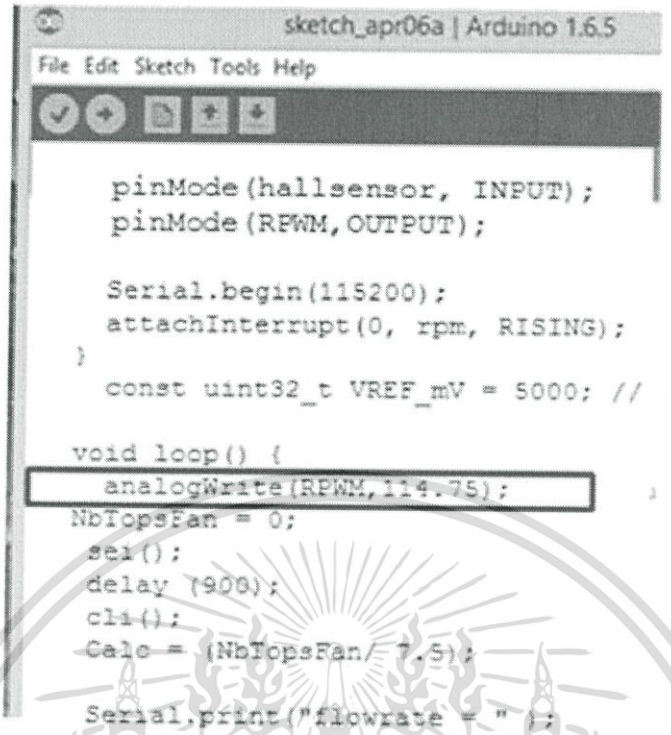
5. บันทึกค่าแรงดัน และเปอร์เซ็นต์ที่อ่านได้ในแต่ละช่วงจากโปรแกรม Arduino



รูปที่ 3.37 วงจร Speed Motor และ Flow Sensor



รูปที่ 3.38 การใช้คำสั่ง map ค่าจากเปอร์เซ็นต์ให้เป็นแรงดัน

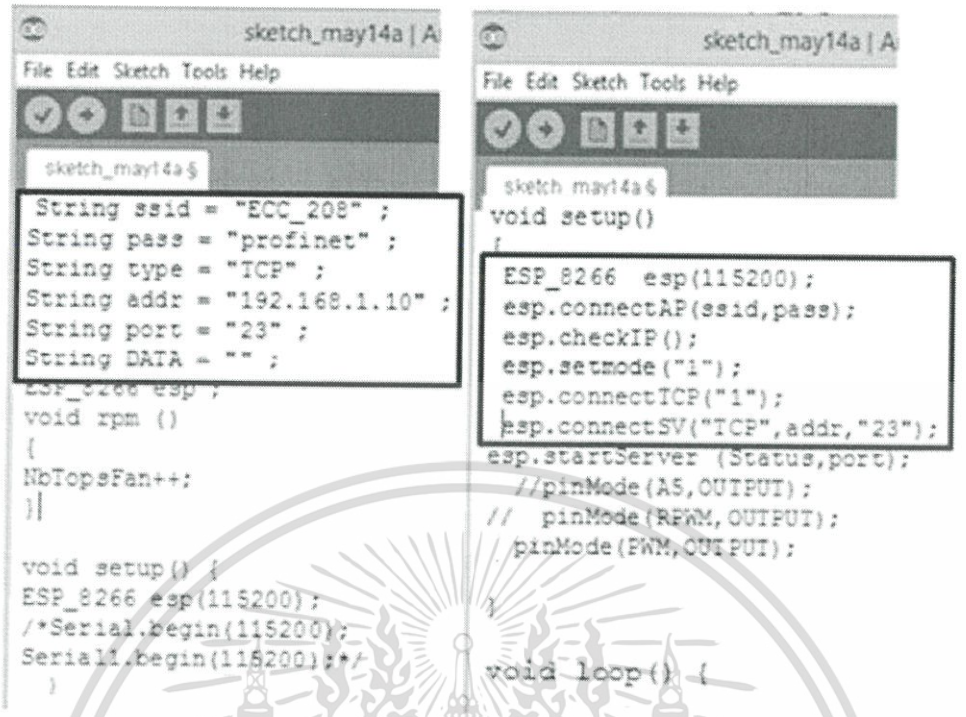


รูปที่ 3.39 การกำหนดค่าเปอร์เซ็นต์การทำงานของปั๊ม

3.5.2 การทดสอบเขียนโปรแกรมส่งค่า

ต่อวงจรตามรูปที่ 3.37 โดยโปรแกรมจะแบ่งออกเป็น 3 ส่วน ดังนี้

1. ส่วนการกำหนดรูปแบบการทำงานของ Wi-Fi Module ESP8266 กำหนดเป็น Client โดยนำ Function ที่เขียนใน ESP8266 Library มาใช้ซึ่งจะกำหนด IP Address และ Port ให้ตรงกับ Wi-Fi Module ESP8266 ที่เป็น Server โดย Wi-Fi Module ESP8266 ทั้งสองฝั่งต้องเชื่อมต่อ Access Point เดียวกัน จึงจะสามารถเชื่อมต่อกันได้

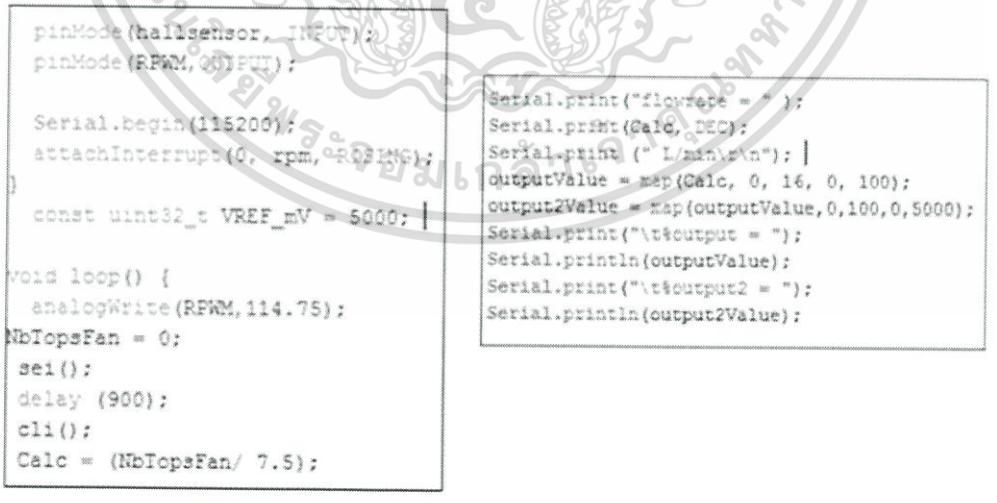


```
String ssid = "ECC_208" ;
String pass = "profinet" ;
String type = "TCP" ;
String addr = "192.168.1.10" ;
String port = "23" ;
String DATA = "" ;
ESP_8266 esp ;
void rpm ()
{
NbTopsFan++;
}

void setup() {
ESP_8266 esp(115200);
/*Serial.begin(115200);
Serial1.begin(115200);*/
}
```

```
void setup()
{
ESP_8266  esp(115200);
esp.connectAP(ssid,pass);
esp.checkIP();
esp.setmode("1");
esp.connectTCP("1");
esp.connectSV("TCP",addr,"23");
esp.startServer (Status,port);
 //pinMode(A5,OUTPUT);
//   pinMode(RPWM,OUTPUT);
 pinMode(PWM,OUTPUT);

}

void loop() {
```

รูปที่ 3.40 การกำหนดรูปแบบการทำงานของ Wi-Fi Module ESP8266 ให้เป็น Client

2. ส่วนการนับรอบการหมุนใบพัดของ Flow Sensor เพื่อนำค่าที่ได้มาคำนวณหาค่า Flow Rate ส่วนแสดงค่า Flow Rate และค่าที่ map เป็นแรงดันออกมาทาง Serial Monitor

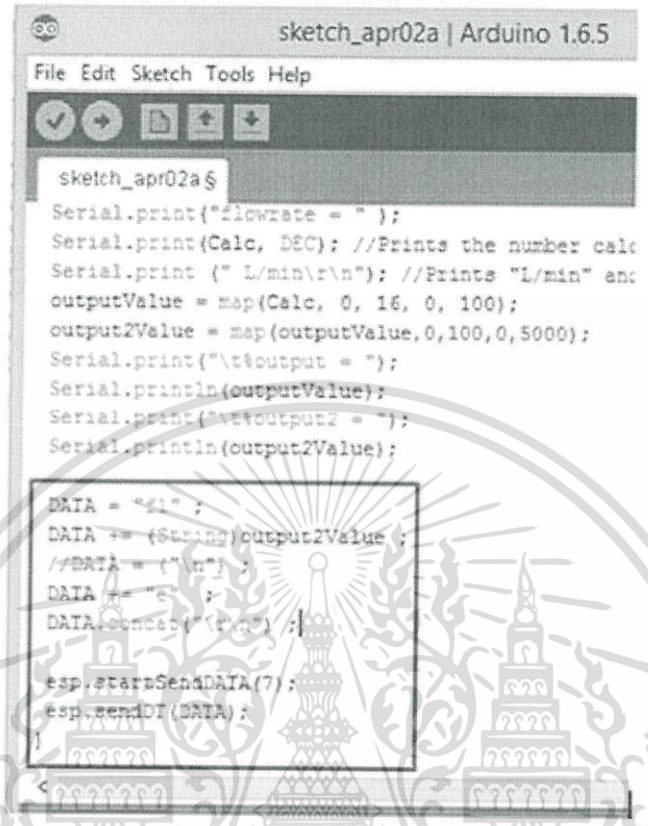


```
pinMode(hallsensor, INPUT);
pinMode(RPWM,OUTPUT);

Serial.begin(115200);
attachInterrupt(0, rpm, RISING);
}
const uint32_t VREF_mV = 5000;

void loop() {
 analogWrite(RPWM,114.75);
NbTopsFan = 0;
sei();
delay (900);
cli();
Calc = (NbTopsFan/ 7.5);
```

```
Serial.print("flowrate = " );
Serial.print(Calc, DEC);
Serial.print (" L/min\r\n");
outputValue = map(Calc, 0, 16, 0, 100);
output2Value = map(outputValue,0,100,0,5000);
Serial.print("\t%output = ");
Serial.println(outputValue);
Serial.print("\t%output2 = ");
Serial.println(output2Value);
```

รูปที่ 3.41 ส่วนการนับรอบการหมุนใบพัดของ Flow Sensor และส่วนแสดงค่า Flow Rate

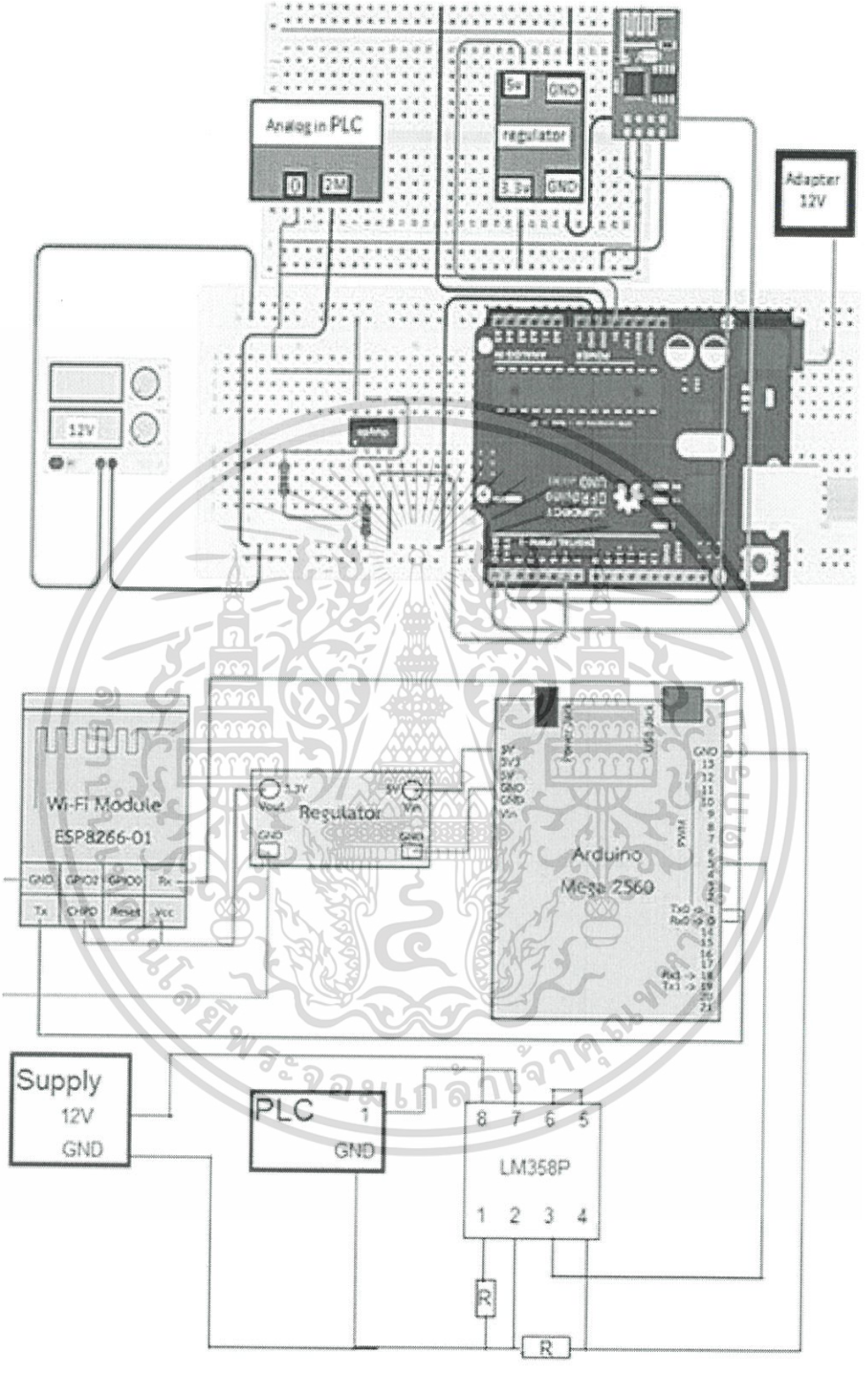
3. ส่วนส่งค่าแรงดันของ Flow Sensor



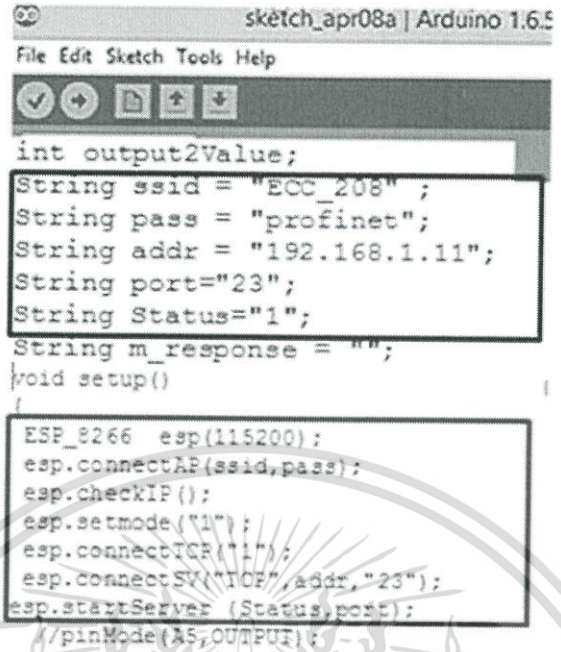
รูปที่ 3.42 ส่วนส่งค่าแรงดันของ Flow Sensor

3.5.3 การทดสอบเขียนโปรแกรมรับค่า

ต่อวงจร Wi-Fi กับวงจร Buffer เข้า PLC โดยวงจร Buffer ต่อเพื่อขยายแรงดันจากบอร์ด Arduino จากช่วง 0-5 V ให้เป็น 0-10 V เพื่อให้ PLC อ่านค่าได้โดยเกิดความคลาดเคลื่อนน้อยดังรูปที่ 3.43

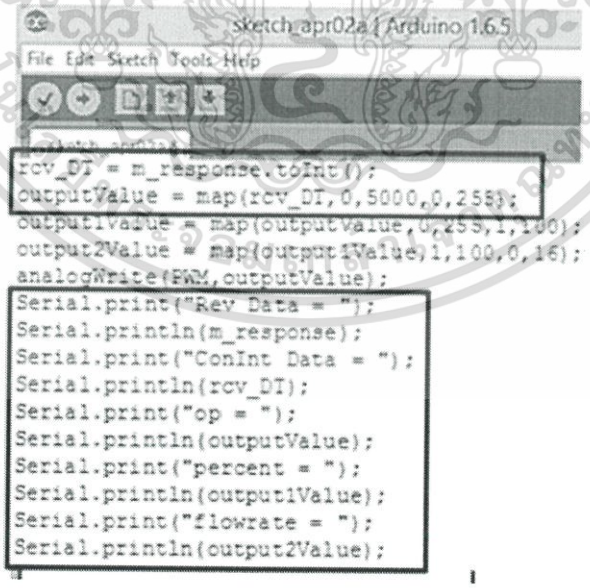


รูปที่ 3.43 การต่อวงจร Wi-Fi กับวงจร Buffer เข้า PLC



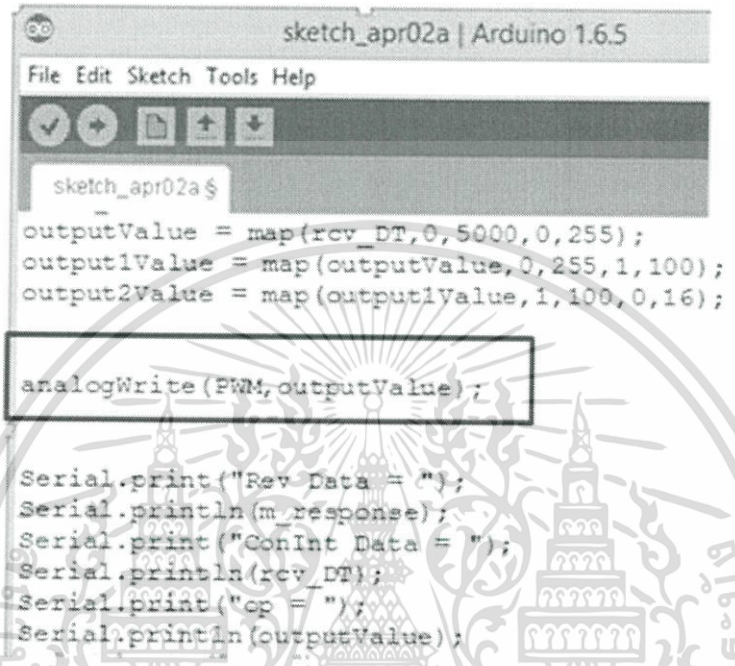
รูปที่ 3.44 กำหนดรูปแบบการทำงานของ Wi-Fi Module ESP8266 ให้เป็น Server

ส่วนรับค่าและแสดงค่าแรงดัน ของ Flow Sensor โดยนำค่าที่ได้มาเก็บไวในตัวแปร m_response จากนั้นทำการแปลงค่าจากแรงดันเป็น 0-255 โดยใช้คำสั่ง map ซึ่งเป็นช่วงของ PWM ที่สามารถส่งเข้า PLC ได้



รูปที่ 3.45 กำหนดรูปแบบการทำงานของ Wi-Fi Module ESP8266 ให้เป็น Server

ส่วนส่งค่าโวลต์ โดยทำการกำหนดให้ขา PWM ที่บอร์ด Arduino เป็นขาเอาต์พุต โดยสัญญาณ Digital ที่อยู่ในขา PWM จะถูกแปลงเป็นสัญญาณ Analog หลังจากนั้นค่าแรงดันจะถูกส่งเข้า Buffer เพื่อขยายแรงดันเขา PLC และเมื่อวัดค่าแรงดันที่ PLC จะพบว่าเท่ากับค่าแรงดันที่หน้า Serial Monitor พอดี



รูปที่ 3.46 ส่วนส่งค่าแรงดัน โดยกำหนดให้ขา PWM ที่บอร์ด Arduino เป็นขาเอาต์พุต

3.6 การทดสอบชุดขับมอเตอร์

3.6.1 การทดสอบชุดขับมอเตอร์ BTS7960 43A

ขั้นตอนการทดลอง

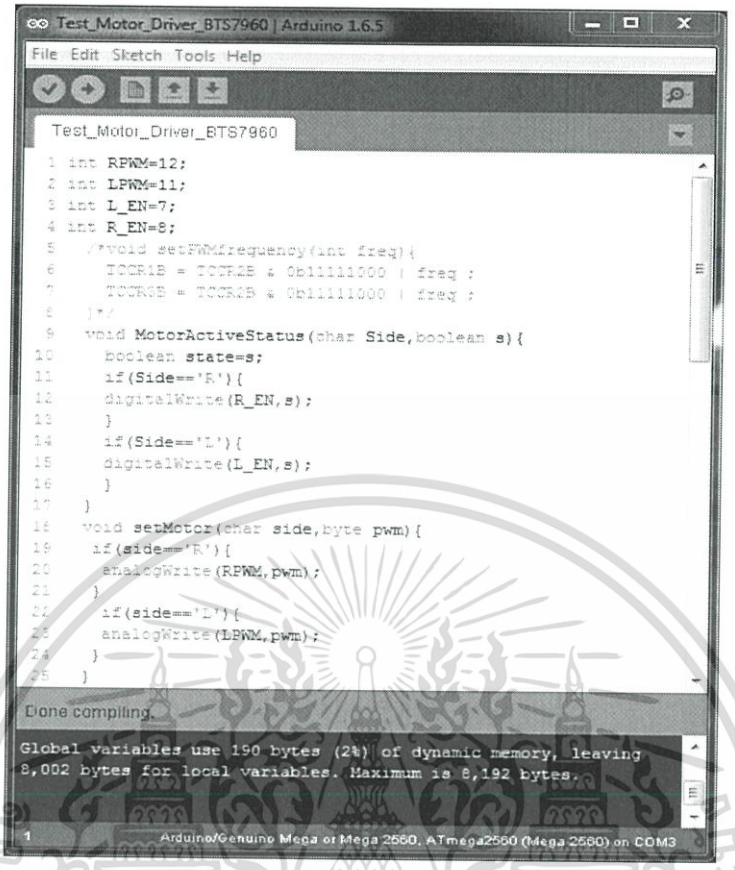
1. ศึกษาคุณสมบัติ และวิธีการใช้งานโมดูลขับมอเตอร์กระแสตรง BTS7960 43A อย่างละเอียด

2. ทดลองเขียนโปรแกรมควบคุมการทำงานของโมดูลขับมอเตอร์กระแสตรง โดยให้มอเตอร์หมุนไปทางขวา และทางซ้ายด้วยความเร็วเพิ่มขึ้นจาก 0-255 PWM สลับกัน โดยสามารถเขียนโปรแกรมได้ดังนี้

```
int RPWM=3;

int LPWM=11;

int L_EN=7;

int R_EN=8;
```

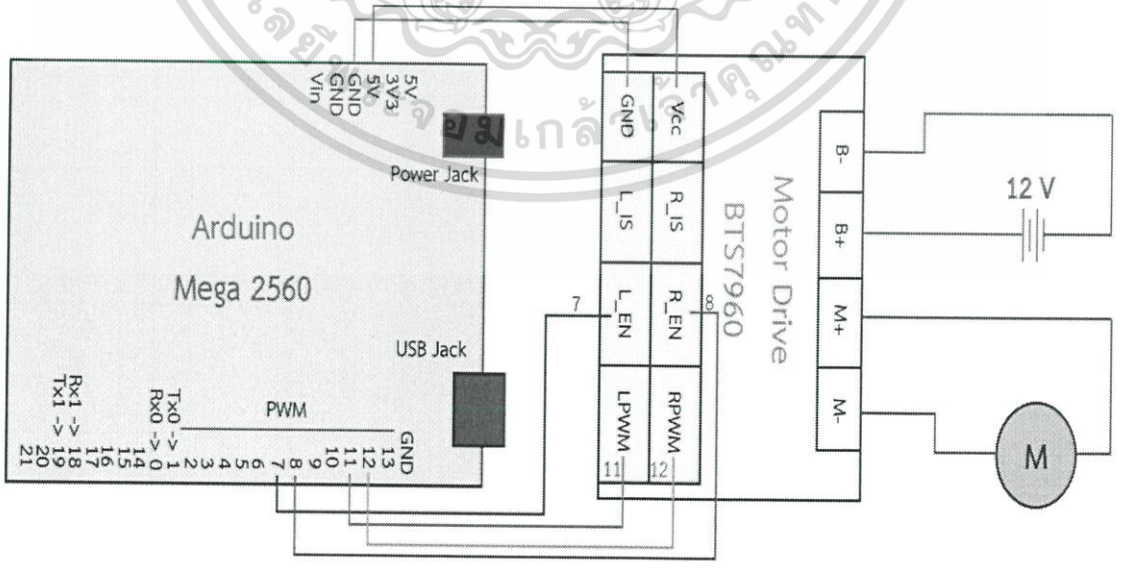
```
void MotorActiveStatus(char
Side,boolean s){

    boolean state=s;
```

```
    if(Side=='R'){
    digitalWrite(R_EN,s);
    }
    if(Side=='L'){
    digitalWrite(L_EN,s);
    }
  }
void setMotor(char side,byte pwm){
   if(side=='R'){
    analogWrite(RPWM,pwm);
   }
    if(side=='L'){
    analogWrite(LPWM,pwm);
   }
  }
 void closeMotor(char side){
    if(side=='R'){
   digitalWrite(RPWM,LOW);
    }
    if(side=='L'){
   digitalWrite(LPWM,LOW);
    }
  }

void setup() {
  for(int i=5;i<9;i++){
  pinMode(i,OUTPUT);
  }
  for(int i=5;i<9;i++){
  digitalWrite(i,LOW);
  }
  delay(1000);
  MotorActiveStatus('R',true);
  MotorActiveStatus('L',true);
  Serial.begin(9600);
  }
void loop() {
  for(int i=0;i<256;i++)
  {
  setMotor('R',i);
  delay(50);
  }
  delay(500);
  closeMotor('R');
  delay(1000);
  for(int i=0;i<256;i++)
  {
  setMotor('L',i);
  delay(50);
  }
  delay(500);
  closeMotor('L');
  delay(1000);
}
```



รูปที่ 3.47 หน้าต่างการเขียนโปรแกรมการควบคุมชุดขับมอเตอร์ BTS7960

3. ต่อวงจรดังรูปที่ 3.48 แล้วอัพโหลดโปรแกรมลงบอร์ด Arduino



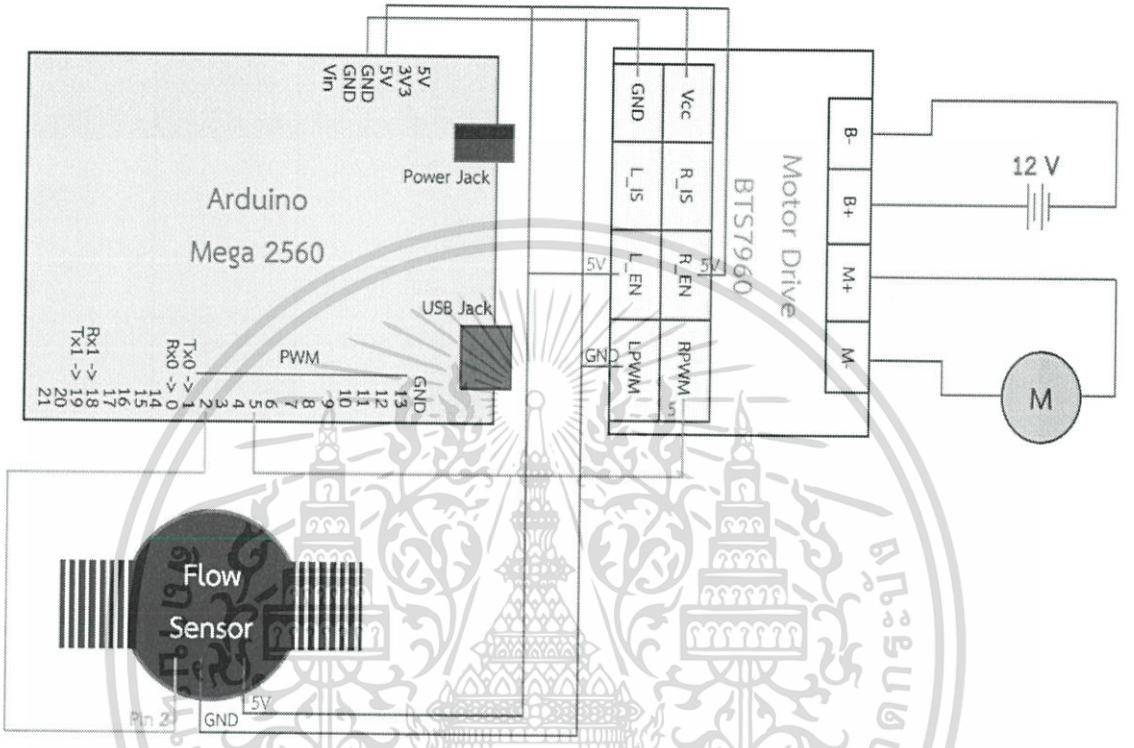
รูปที่ 3.48 การเชื่อมต่อ BTS7960 กับบอร์ด Arduino Mega 2560 R3

4. บันทึกผลการทดลอง

3.6.2 การทดสอบการควบคุมความเร็วมอเตอร์ปั๊มน้ำด้วย Duty Cycle

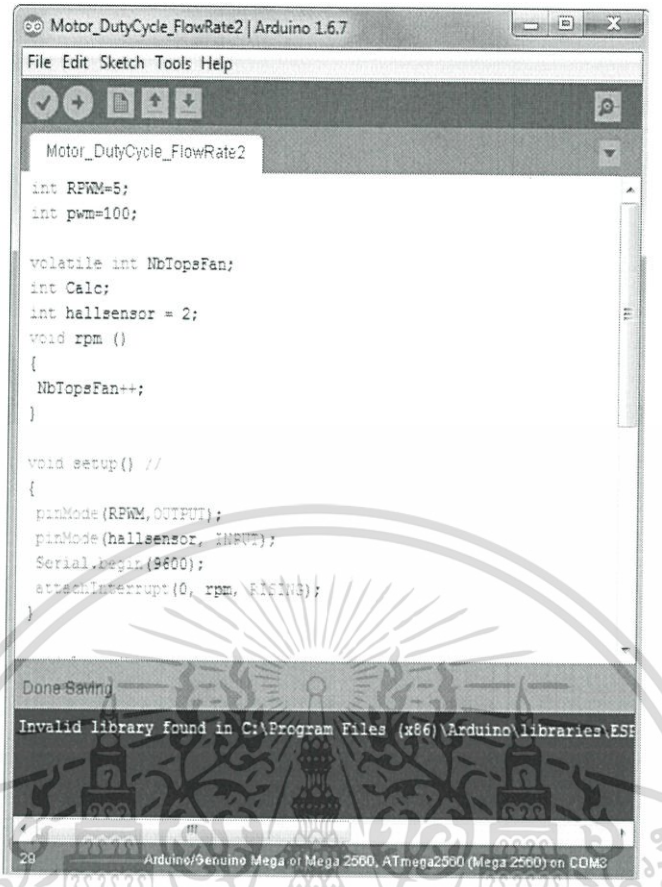
ขั้นตอนการทดลอง

1.จัดเตรียมอุปกรณ์ และต่อวงจรดังรูปที่ 3.49



รูปที่ 3.49 การต่อวงจร Flow Sensor และมอเตอร์ กับบอร์ด Arduino

2. เขียนโปรแกรมควบคุมความเร็วมอเตอร์ปั๊มน้ำด้วย Duty Cycle พร้อมวัดค่าอัตราการไหล โดยนำโปรแกรมควบคุมมอเตอร์ที่ได้เขียนไว้ในการทดสอบชุดขับมอเตอร์ BTS7960 43A มาแก้ไขให้สามารถควบคุมความเร็วมอเตอร์ปั๊มน้ำตามค่า Duty Cycle พร้อมทั้งเพิ่มในส่วนของการอ่านค่าจาก Flow Sensor และคำนวณค่าอัตราการไหล (Flow Rate) ของน้ำในท่อลงในโปรแกรม จากนั้นอัพโหลดโปรแกรมลงในบอร์ด Arduino



รูปที่ 3.50 หน้าต่างการเขียนโปรแกรมควบคุมความเร็วมอเตอร์ปั๊มน้ำด้วย Duty Cycle พร้อมวัดค่าอัตราการไหล

3. เปิด Serial Monitor เพื่อดูเอาต์พุต และบันทึกผลการทดลอง

4. ทำการทดลองกับแบบจำลองระบบจริง โดยทำการทดลองข้อ 1-3 ซ้ำอีกครั้ง แล้วบันทึกผลการทดลอง

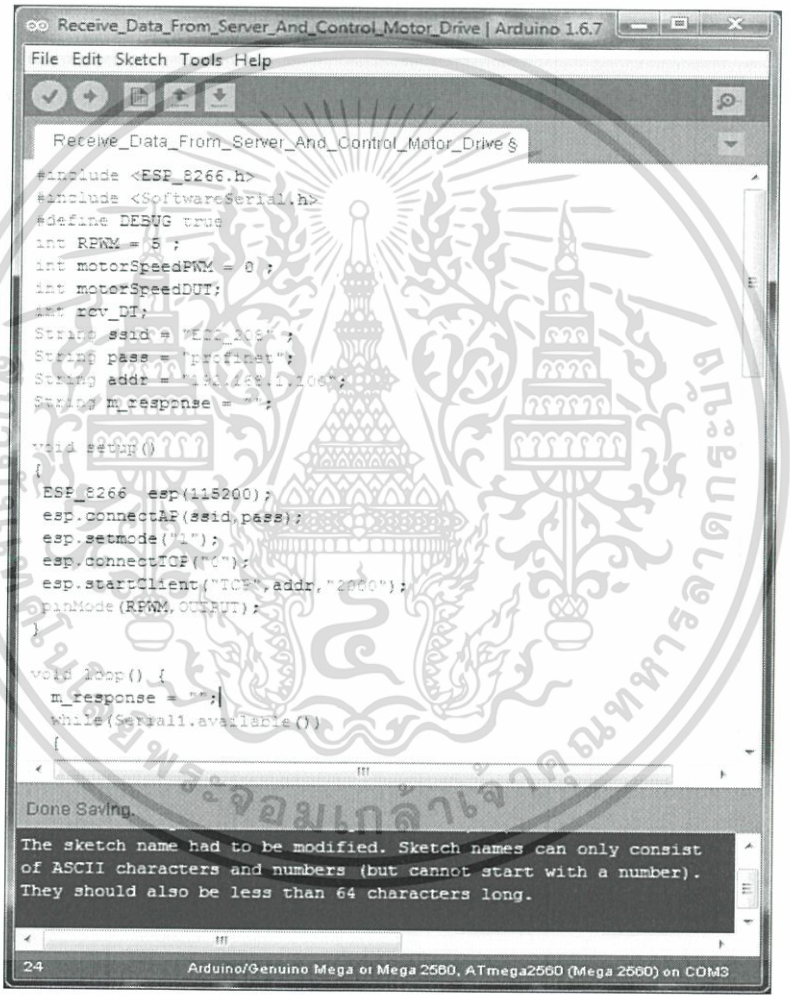
3.6.3 การทดสอบเขียนโปรแกรมรับค่าจาก Server (PLC) เพื่อควบคุมมอเตอร์ปั๊มน้ำ

1. นำโปรแกรมรับค่า Duty Cycle จาก Server (Hercules) เพื่อควบคุมความเร็วมอเตอร์ปั๊มน้ำ ที่ได้เขียนไว้ข้างต้นมาปรับแก้ให้สามารถรับค่าจาก PLC ได้ โดยการเปลี่ยน IP Address และ Port ของ Server ที่จะทำการเชื่อมต่อเป็น IP Address และ Port ของ PLC คือ 192.168.1.100 และ 2000 ตามลำดับ โดย PLC ที่จะทำหน้าที่เป็น Server จะต้องเชื่อมต่อ Access Point เดียวกันกับโมดูล ESP8266 จึงจะทำการเชื่อมต่อกันได้ โดยสามารถเขียนโปรแกรมได้ดังนี้

```
#include <ESP_8266.h>
#include <SoftwareSerial.h>
#define DEBUG true
int RPWM = 5 ;
```

```
int motorSpeedPWM = 0 ;
int motorSpeedDUT;
int rcv_DT;
String ssid = "ECC_208";
String pass = "profinet";
String addr = "192.168.1.100";
String m_response = "";

void setup()
{
 ESP_8266 esp(115200);
 esp.connectAP(ssid,pass);
 esp.setmode("1");
 esp.connectTCP("0");
 esp.startClient("TCP",addr,"2000");
 pinMode(RPWM,OUTPUT);
}
void loop() {
  m_response = "";
  while(Serial1.available())
  {
    char c = Serial1.read();
    Serial.print(c);
    m_response += c;
  }
    if(m_response.equals("0"))
   {
    digitalWrite(RPWM,LOW);
    Serial.println("Motor Stop");
    motorSpeedDUT = 0;
    motorSpeedPWM = 0;
   }
  else
  {
    rcv_DT = m_response.toInt();
    if  ( rcv_DT > 0 )
    {
     motorSpeedDUT = rcv_DT;
digitalWrite(RPWM,HIGH);
     motorSpeedPWM = map(motorSpeedDUT, 0, 100, 0, 255);
analogWrite(RPWM,motorSpeedPWM);
     Serial.println("Motor Start");
    }
  }
 Serial.print("DUTY CIRCLE = ");
```

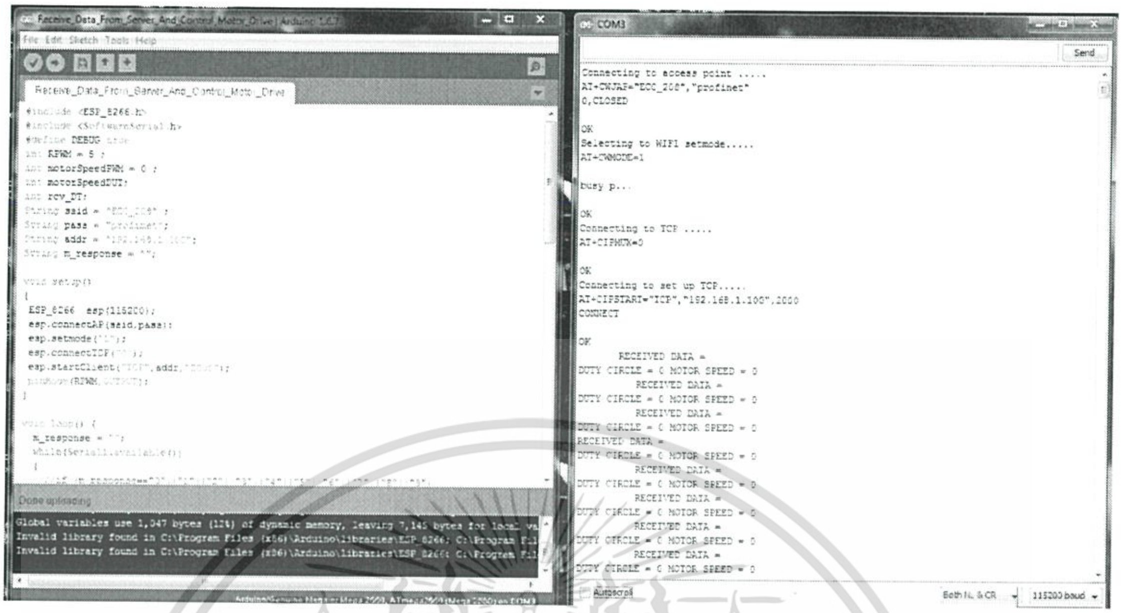
```
    Serial.print(motorSpeedDUT);
    Serial.print("\t");
    Serial.print("MOTOR SPEED = ");
    Serial.println(motorSpeedPWM);
    Serial1.flush();
    Serial.flush() ;
    delay(150);
}
```



รูปที่ 3.51 หน้าต่างโปรแกรมรับค่าที่ทำการแก้ไข IP Address และ Port แล้ว

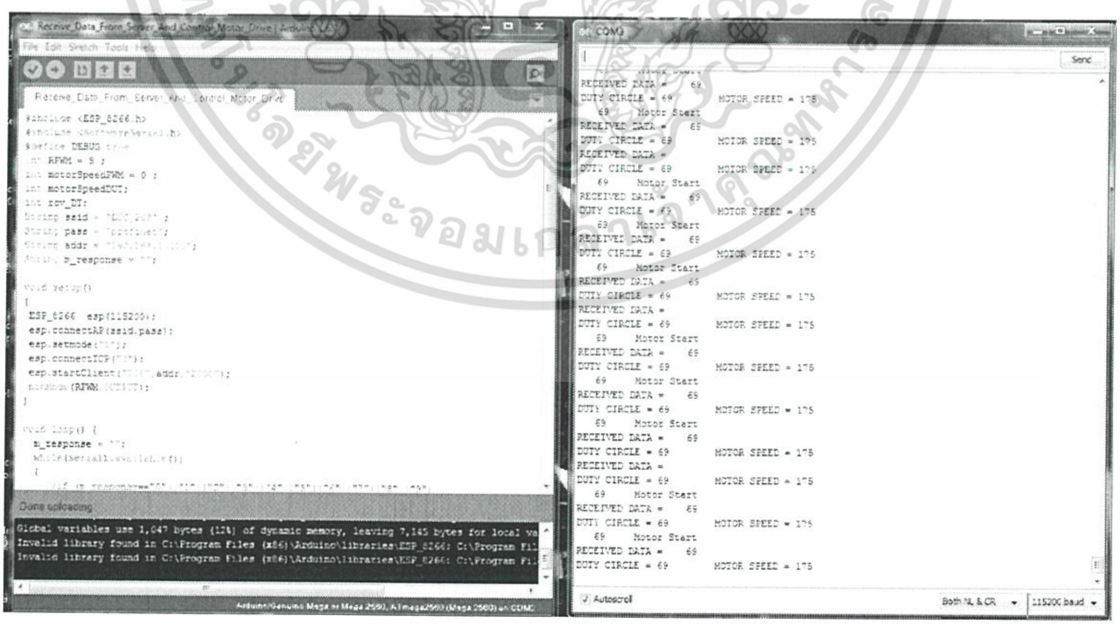
2. ทำการต่อวงจรเช่นเดียวกันกับการทดลองเขียนโปรแกรมรับค่า Duty Cycle จาก Server (PLC) เพื่อควบคุมความเร็วมอเตอร์ปั๊มน้ำ

3. เปิดหน้าต่าง Serial Monitor เพื่อดู Output ของโปรแกรม



รูปที่ 3.52 หน้าต่างโปรแกรมกับ Serial Monitor ของ Arduino (1)

4. เมื่อ PLC ส่งค่า Duty Cycle มา ตัวโมดูลจะทำการรับค่า จากนั้นนำค่าที่ได้ไปแปลงเป็นค่าของ PWM และทำการสั่งมอเตอร์ปั๊มน้ำต่อไป ซึ่งคำสั่งดังกล่าวถูกเขียนไว้ในโปรแกรมรับค่า Duty Cycle แล้ว

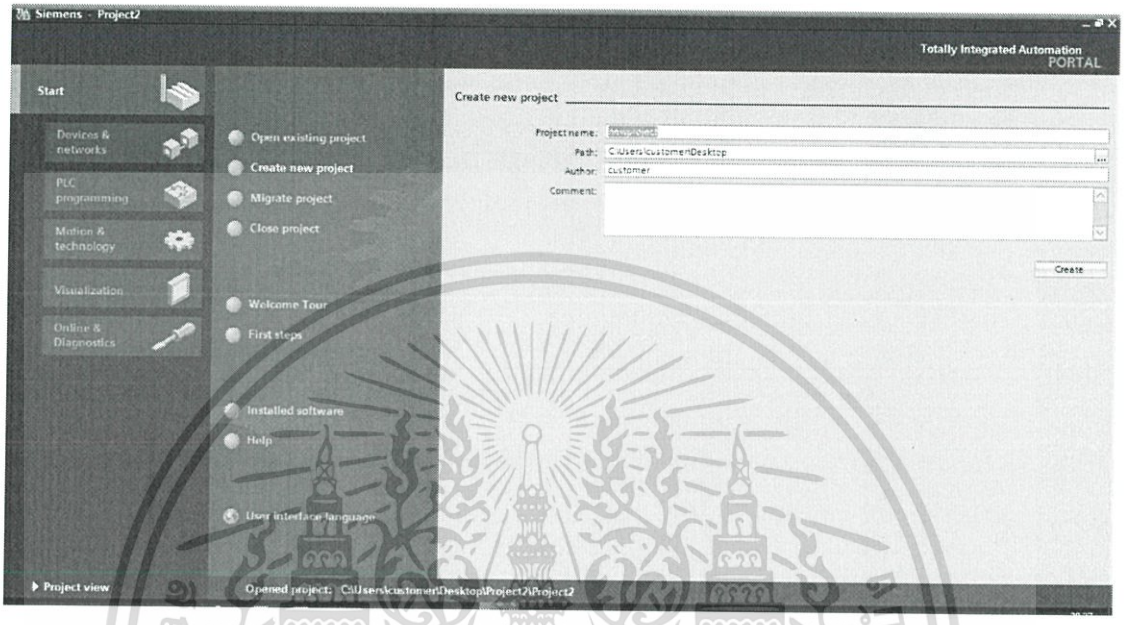


รูปที่ 3.53 หน้าต่างโปรแกรมกับ Serial Monitor ของ Arduino (2)

3.7 โปรแกรม TIA Portal V13

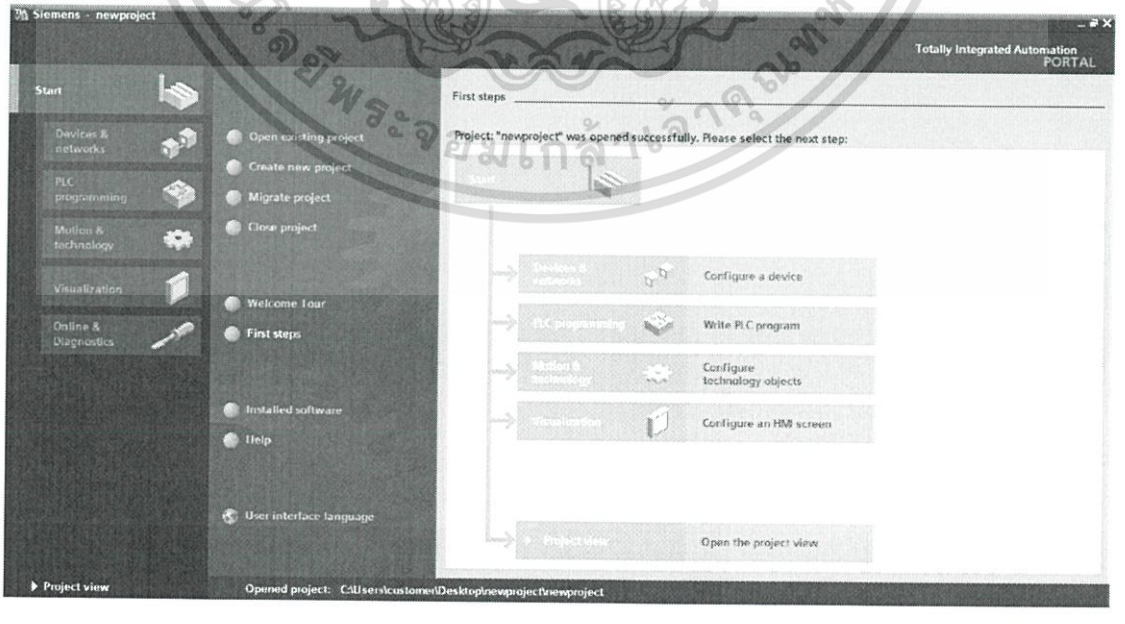
สร้างโปรเจค และตั้งค่าอุปกรณ์

1. กด Create New Project > ตั้งชื่อโปรเจค > กด Create

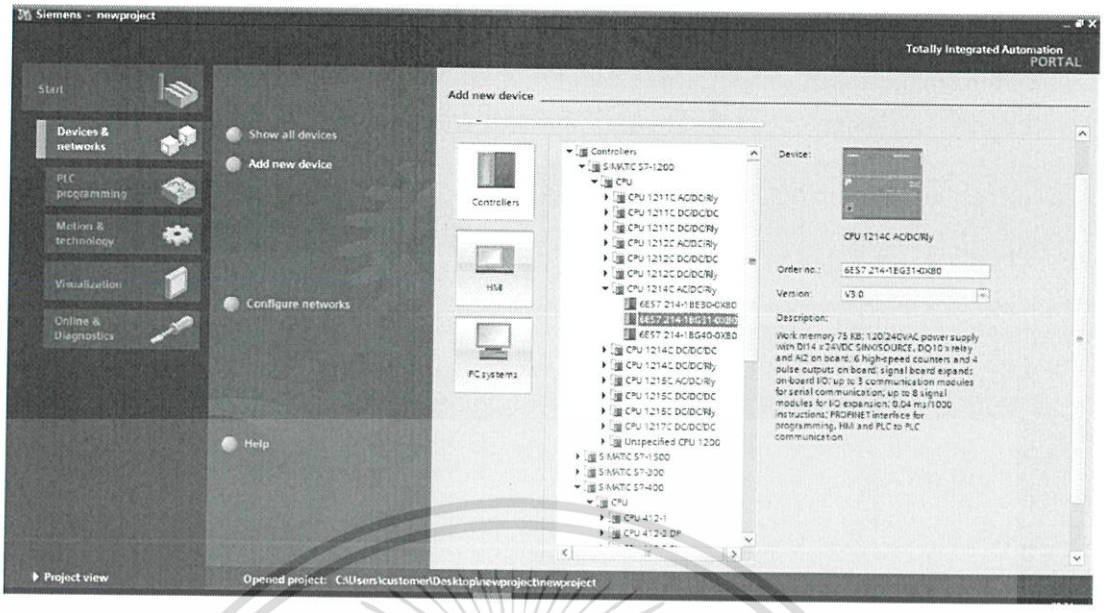


รูปที่ 3.54 หน้าแสดงผล Portal View ในส่วนของการสร้างโปรเจค

2. กด Configure a device > Add new device > Controller > รุ่นของพีแอลซีที่ใช้ทำในโปรเจค ชื่อรุ่น 6ES7 214-1BG31-0XB0 > กด Add

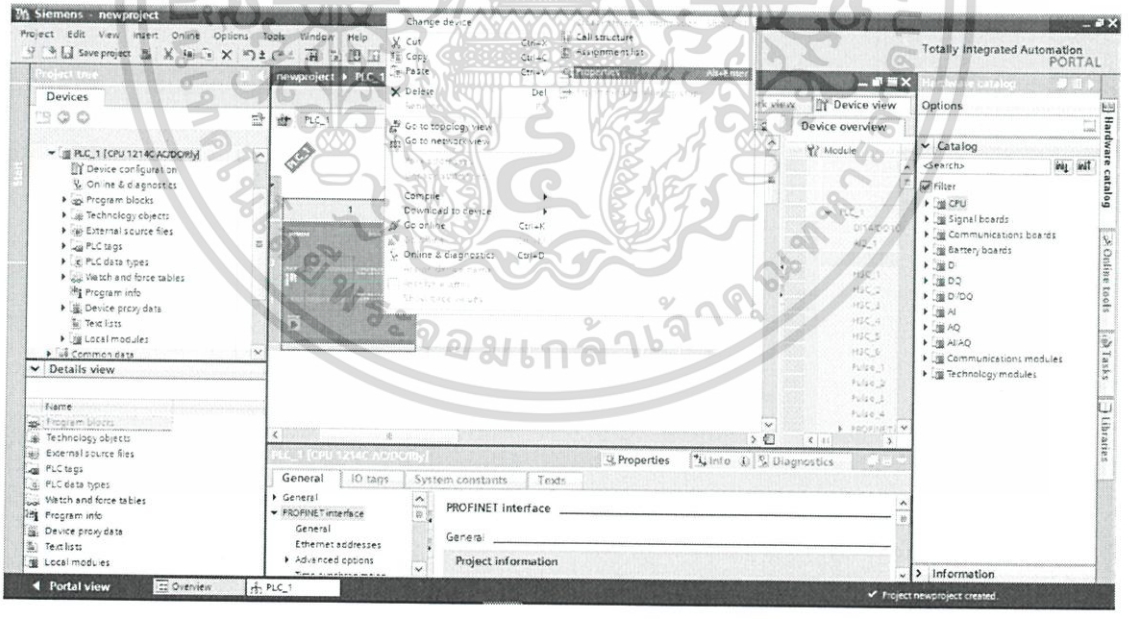


รูปที่ 3.55 หน้าแสดงผล Portal View ในส่วนของการเพิ่มอุปกรณ์

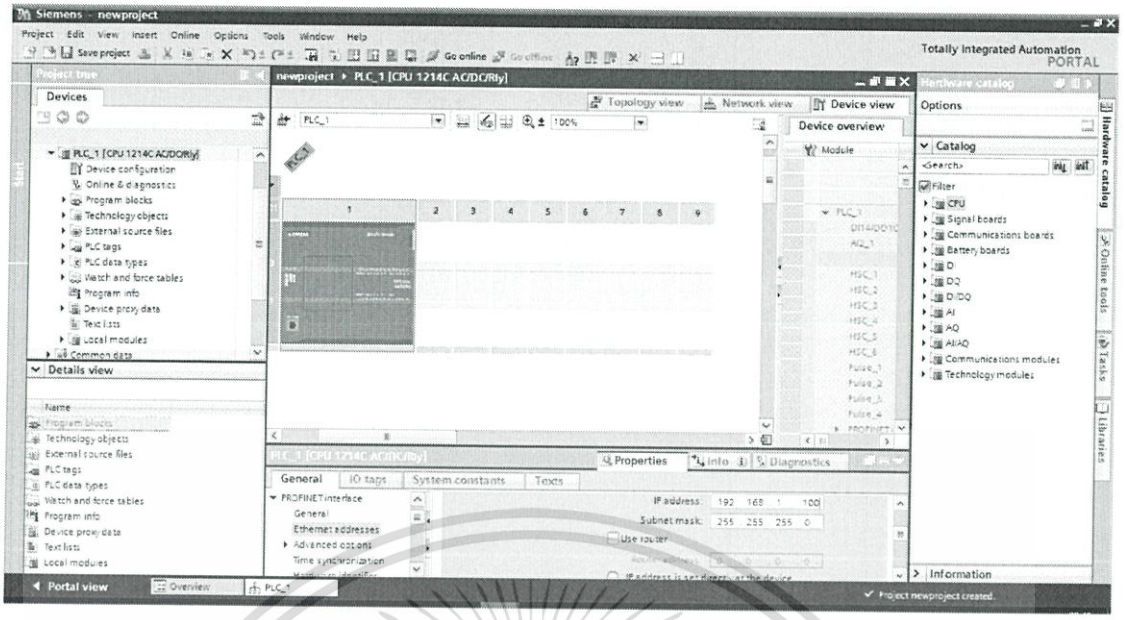


รูปที่ 3.56 หน้าแสดงผล Portal View ในส่วนของการเลือกอุปกรณ์พีแอลซีคอนโทรลเลอร์

3. เมื่อสร้างโปรเจคแล้วจะต้องตั้ง IP Address ของพีแอลซีในโปรแกรมให้ตรงกับเครื่องจริงโดย คลิกขวาที่รูปพีแอลซี > Properties > เลือก PROFINET Interface > Ethernet addresses >IP Protocol > ปรับ IP Address เป็น 192.168.1.100



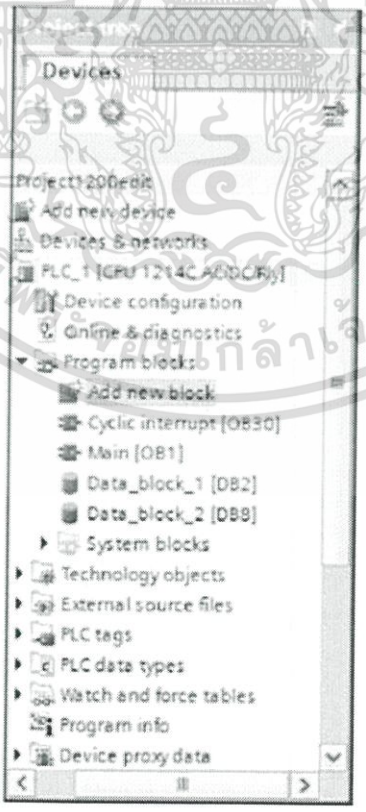
รูปที่ 3.57 การตั้งค่า IP Address ของหน่วยประมวลผลกลางของพีแอลซี (1)



รูปที่ 3.58 การตั้งค่า IP Address ของหน่วยประมวลผลกลางของพีแอลซี (2)

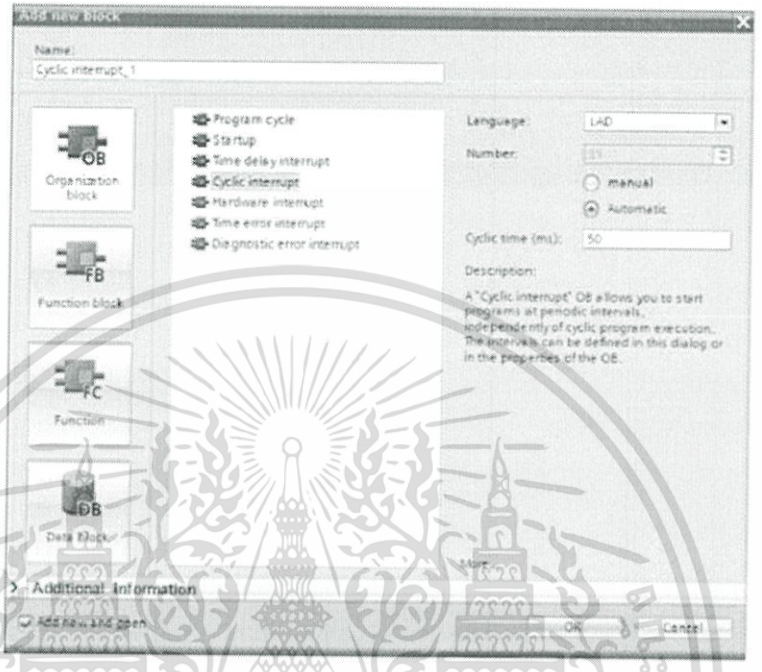
เขียนโปรแกรมพีแอลซีเพื่อควบคุมระบบโดยใช้ตัวควบคุมแบบพีไอดี

1. ที่ Project Tree เลือก PLC_1 > Program Blocks > Add new block



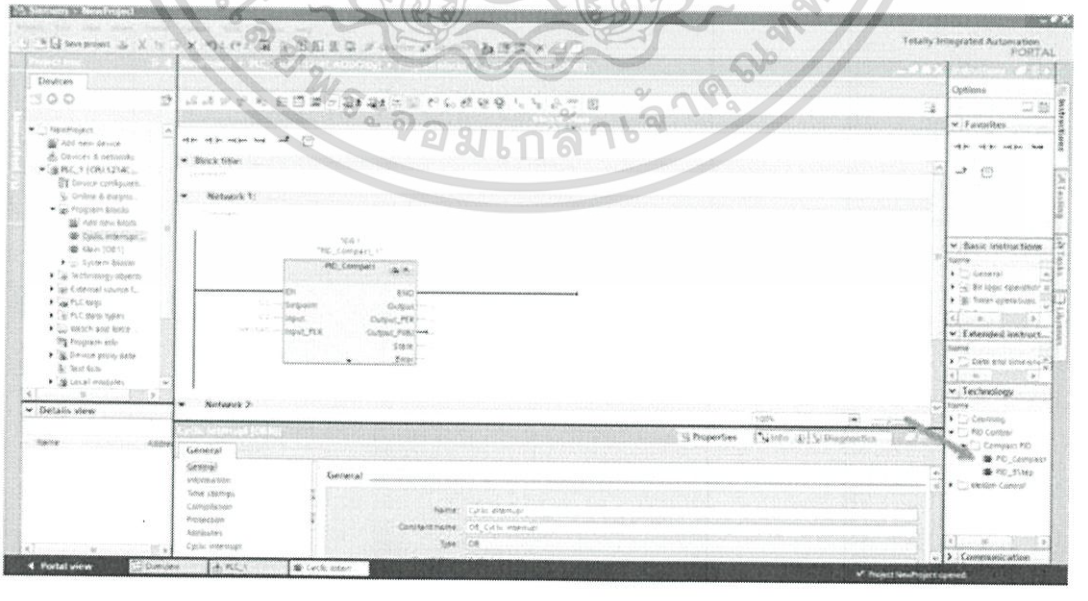
รูปที่ 3.59 การสร้างพื้นที่สำหรับหรับเขียนโปรแกรมควบคุม Main [OB1]

2. เลือก Organization Block > Cyclic Interrupt โดย OB แบบ Cyclic Interrupt จะสั่งให้โปรแกรมที่อยู่ภายใน OB ทำงานในทุกๆ Cyclic Time ที่ตั้งไว้ ดังนั้นในการสร้างคอนโทรลเลอร์แบบพีไอดี จึงจำเป็นต้องสร้างใน OB นี้ เพื่อให้วัดค่าใหม่ตลอดเวลา ซึ่งในโปรเจคนี้จะตั้ง Cyclic Time ไว้ที่ 50 ms



รูปที่ 3.60 การสร้าง Cyclic Interrupt Block

3. เลือก Technology > PID Control > Compact PID > PID_Compact > PID_Compact ไปวางบน Racks



รูปที่ 3.61 การเพิ่มบล็อก PID_Compact

โดย Setpoint – ค่าที่ต้องการให้ระบบลู่เข้าหา

Input – ค่าปัจจุบันของระบบที่ระต้องการควบคุม

Input_PER – ค่าปัจจุบันของระบบที่ระต้องการควบคุมในหน่วย Analog

Output – ค่าที่ต้องสั่งเพื่อควบคุมระบบ

Output_PER – ค่าที่ต้องสั่งเพื่อควบคุมระบบในหน่วย Analog

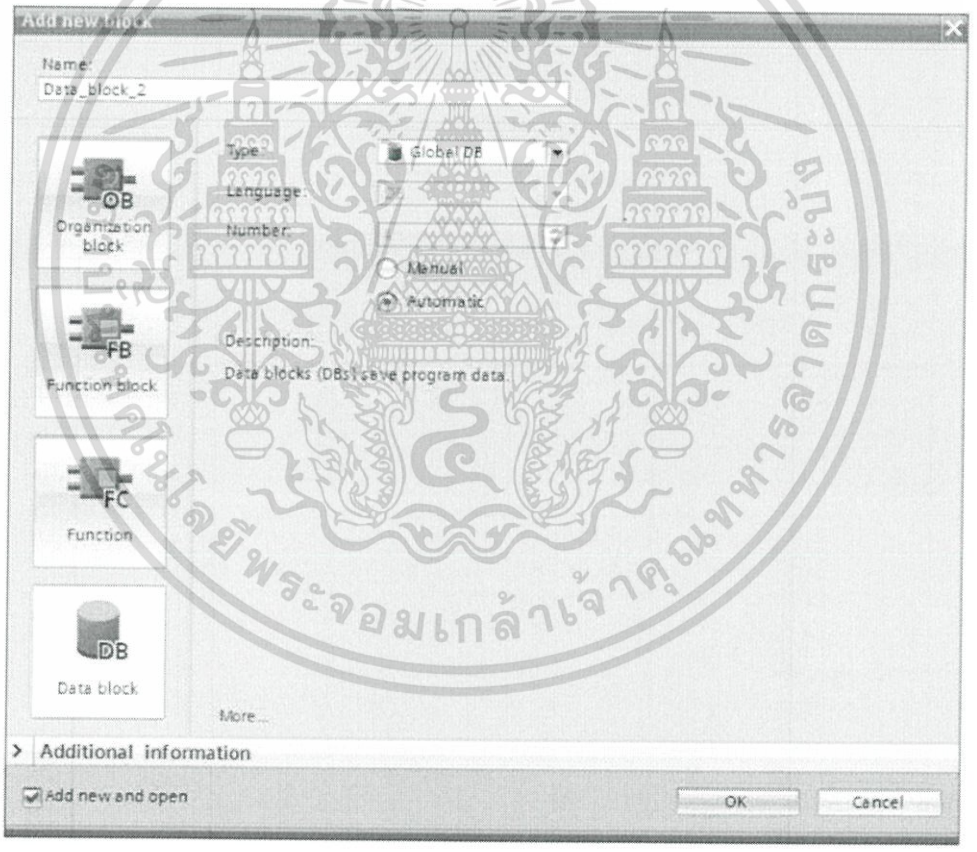
Output_PWM – ค่าที่ต้องสั่งเพื่อควบคุมระบบโดยควบคุมแบบ Boolean

State – สถานะของบล็อก PID_Compact

Error – สถานะ Error ของบล็อก PID_Compact

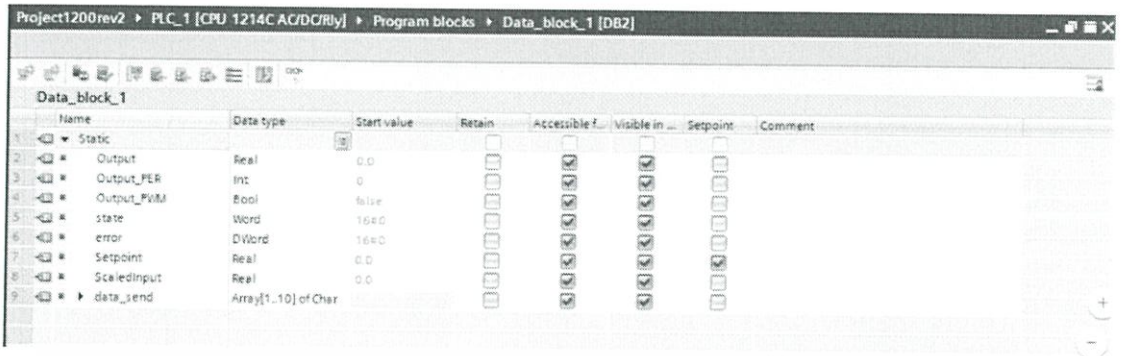
(โดยค่าของ State และ Error สามารถดูได้จาก Manual)

4. กด Add new Block > Data Block > OK



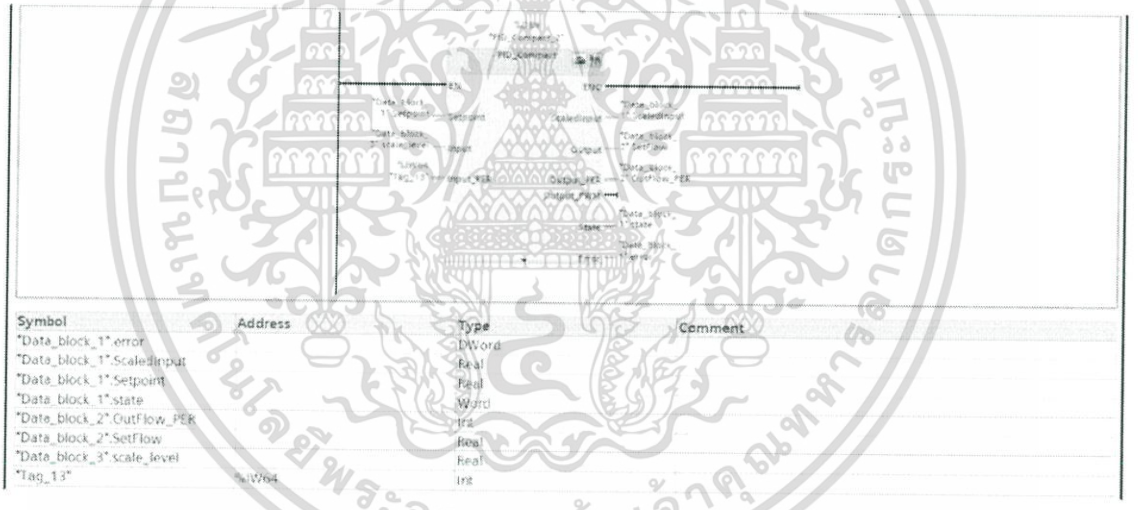
รูปที่ 3.62 การสร้างหน้า Data Block สำหรับตั้งค่าพารามิเตอร์พีไอดี

ตั้งพารามิเตอร์ต่างๆ สำหรับ Primary PLC Controller ดังนี้



รูปที่ 3.63 ตัวแปรสำหรับบล็อกพีไอดีในหน้า Data Block

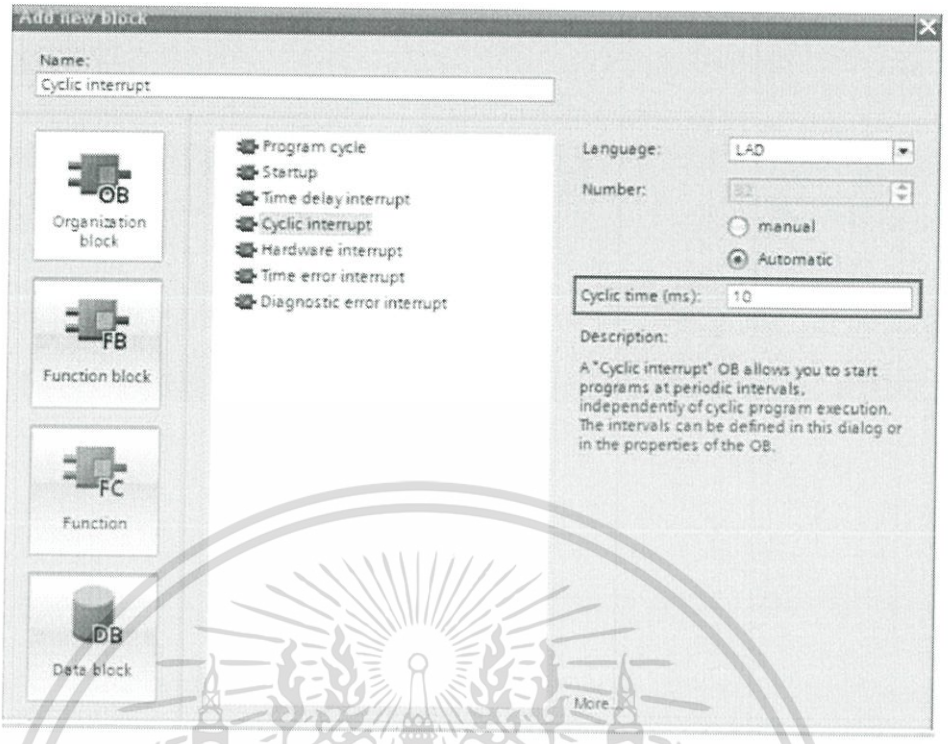
5. นำพารามิเตอร์มาใส่ในบล็อก PID_Compact ด้วยวิธี Drag and Drop ดังนี้



รูปที่ 3.64 การใส่พารามิเตอร์ลงบล็อก PID_Compact

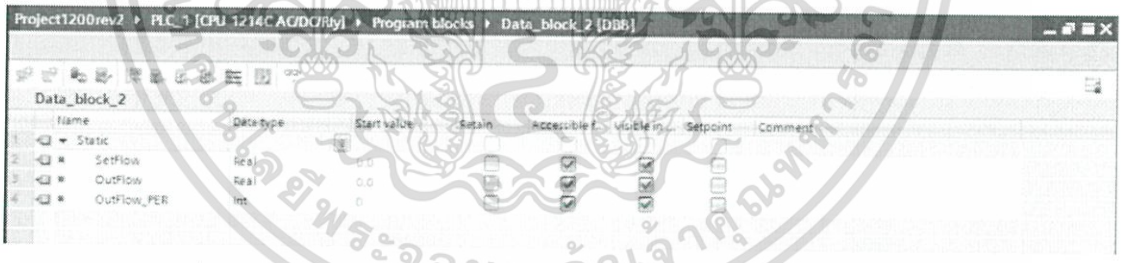
โดย IW64 คือ Address ขา Analog In ของพีแอลซีที่รับค่าจาก Level Sensor

6. สร้าง Cyclic Interrupt Block สำหรับตัวควบคุมแบบพีไอดีอีกตัว เพื่อทำการควบคุมแบบแคสเคด โดยให้ Cyclic time มีค่า 10 ms เนื่องจากเวลาในการตอบสนองของ Secondary Controller จะต้องมีความไวกว่าของ Primary Controller เพื่อให้ได้ระบบที่มีความแม่นยำ

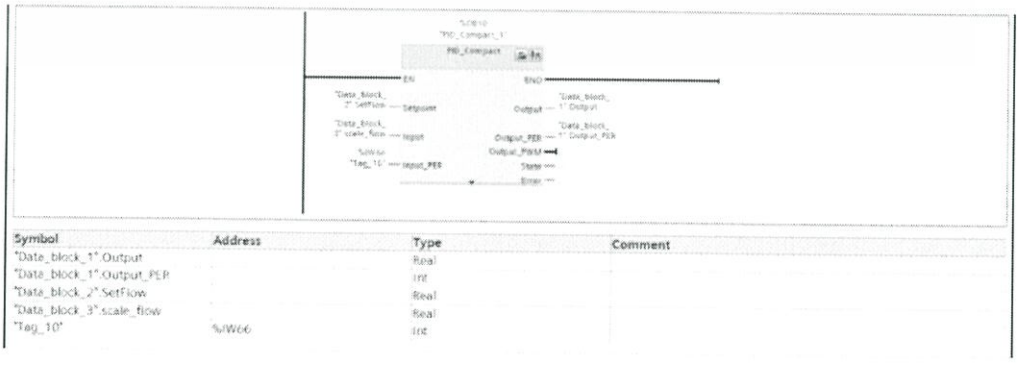


รูปที่ 3.65 การตั้งค่า Cyclic Time

7. สร้าง Data Block เพื่อตั้งพารามิเตอร์สำหรับSecondary PLC Controller และนำพารามิเตอร์ไปใส่ในบล็อก ดังนี้



รูปที่ 3.66 สร้างพารามิเตอร์สำหรับบล็อกพีไอดีสำหรับการควบคุมแบบแคสเคด

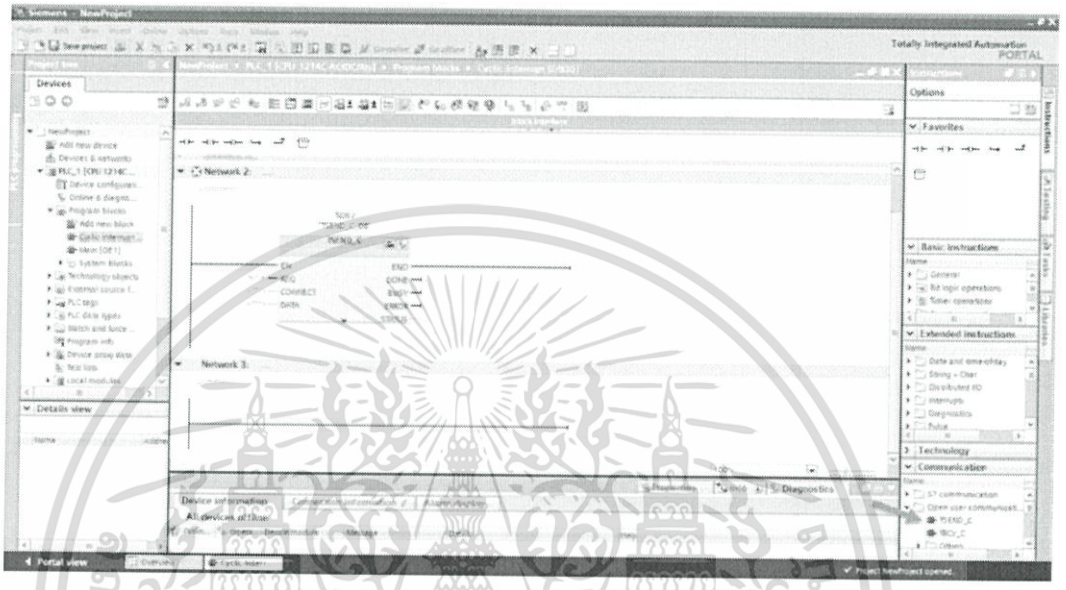


รูปที่ 3.67 การใส่พารามิเตอร์ลงบล็อก PID_Compact สำหรับการควบคุมแบบแคสเคด

โดย IW66 คือ Address ขา Analog In ของพีแอลซีที่รับค่าจาก Flow Sensor

8. ที่ Project Tree เลือก PLC_1 > Program Blocks > Main [OB1]

9. เลือก Communication > Open user communication > TSEND_C ไปวางบน Racks



รูปที่ 3.68 การเพิ่มบล็อก TSEND_C สำหรับการเชื่อมต่ออุปกรณ์ไร้สายผ่านโปรโตคอล TCP/IP

โดย

REQ - ส่งคำขอในการส่งข้อมูล ซึ่งสถานะของขานี้ต้องเป็น TRUE จึงจะทำการส่งข้อมูล

CONT - เป็นตัวอนุญาตการเชื่อมต่อระหว่างพีแอลซีกับพีแอลซี หรือพีซีเครื่องอื่น ซึ่งสถานะของขานี้ต้องเป็น TRUE จึงจะสามารถทำการเชื่อมต่อได้

LEN - ขนาดของข้อมูลที่จะส่งมีหน่วยเป็นบิต

CONNECT - เป็นขาที่เชื่อมกับบล็อกสำรับรูปที่จะสร้างขึ้นหลังทำการตั้งค่าบล็อก TSEND_C

DATA - ข้อมูลที่ต้องการส่ง

DONE - สถานะของบล็อก จะแสดงเป็น TRUE เมื่อบล็อกส่งข้อมูลเสร็จสิ้น

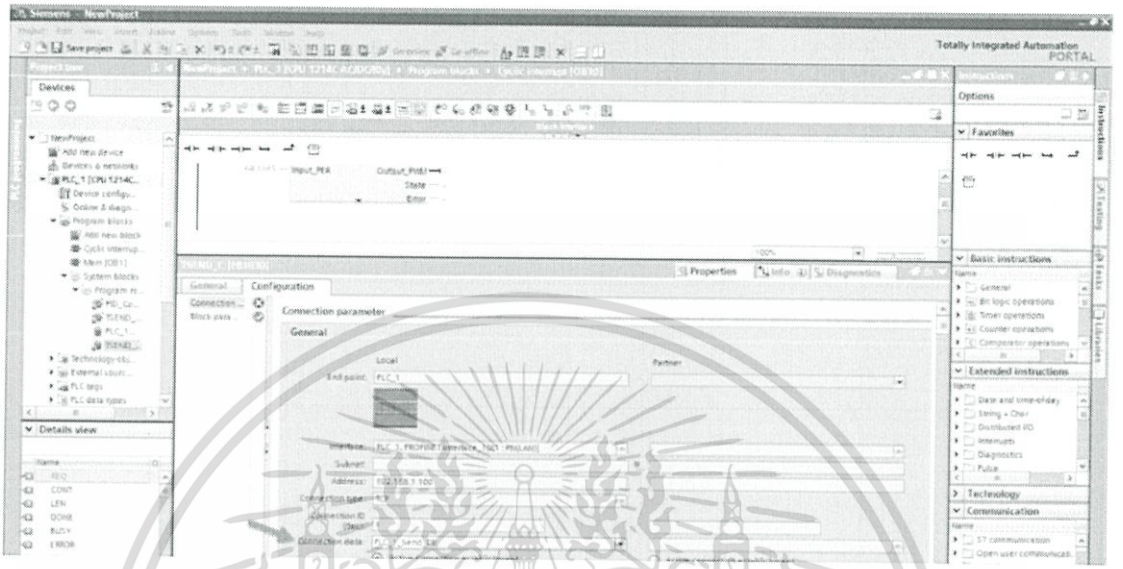
BUSY - สถานะของบล็อก จะแสดงเป็น TRUE เมื่อบล็อกกำลังทำงาน

ERROR - สถานะของบล็อก จะแสดงเป็น TRUE เมื่อบล็อกเกิดข้อผิดพลาด

STATUS - สถานะของบล็อก TSEND_C

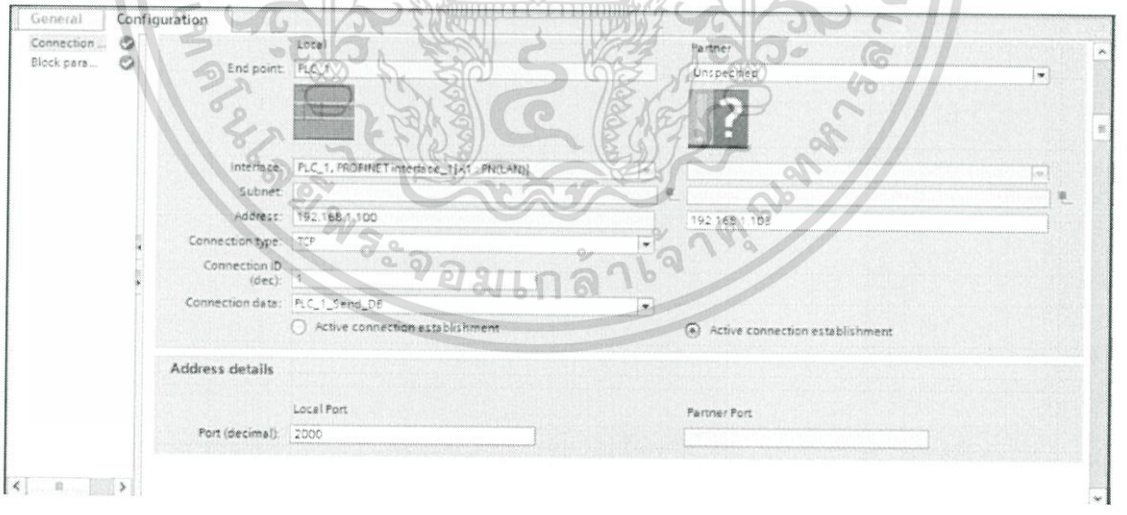
(สามารถดูว่าค่าที่แสดงมีความหมายว่าอย่างไรจากคู่มือการใช้)

10. เลือกบล็อก TSEND_C > Properties > Configuration > Connection Parameter > Connection Data > new โปรแกรมจะสร้างบล็อก PLC1_Send_DB เพื่อเก็บพารามิเตอร์ต่างๆ และเชื่อมกับขา Connect ของบล็อกโดยอัตโนมัติ



รูปที่ 3.69 การตั้งค่าการเชื่อมต่อของบล็อก TSEND_C

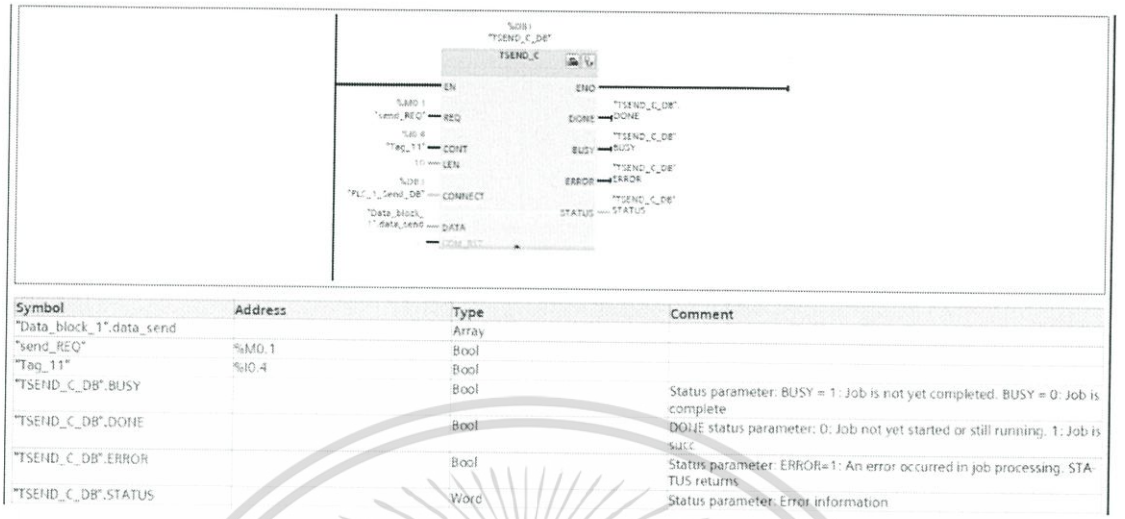
11. ใส่ค่าต่างๆ ดังนี้



รูปที่ 3.70 ค่าการตั้งค่าการเชื่อมต่อบล็อก TSEND_C กับอุปกรณ์

โดย 192.168.1.103 คือ IP ของ Wi-Fi Module ที่ต่อกับมอเตอร์

12. นำพารามิเตอร์มาใส่ในบล็อก TSEND_C ด้วยวิธี Drag and Drop ดังนี้



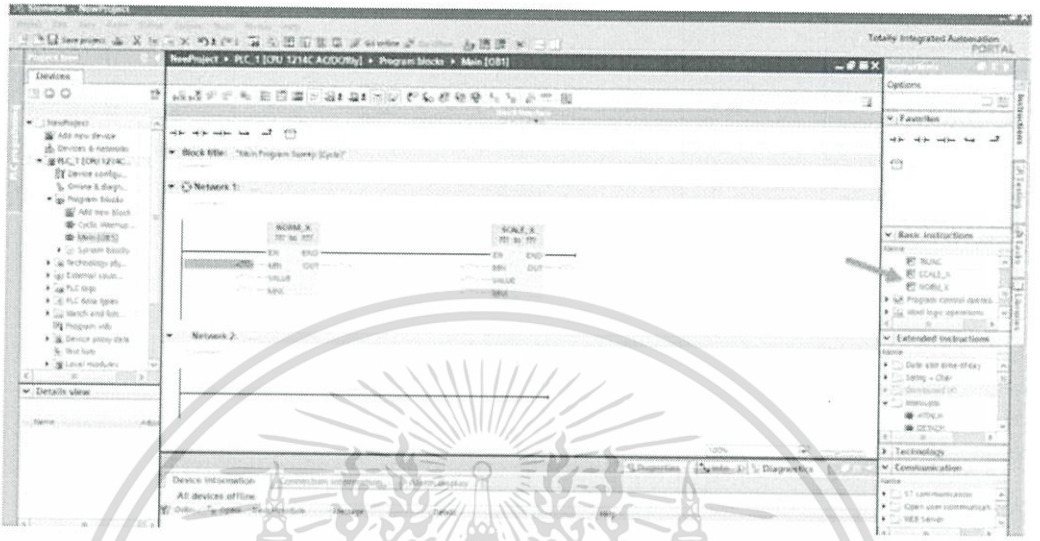
Symbol	Address	Type	Comment
"Data_block_1".data_send		Array	
"send_REQ"	%M0.1	Bool	
"Tag_11"	%I0.4	Bool	
"TSEND_C_DB".BUSY		Bool	Status parameter: BUSY = 1: Job is not yet completed. BUSY = 0: Job is complete
"TSEND_C_DB".DONE		Bool	DONE status parameter: 0: Job not yet started or still running. 1: Job is succ
"TSEND_C_DB".ERROR		Bool	Status parameter: ERROR=1: An error occurred in job processing. STATUS returns
"TSEND_C_DB".STATUS		Word	Status parameter: Error information

รูปที่ 3.71 การใส่พารามิเตอร์ลงบล็อก TSEND_C

13. สร้าง Data Block เพื่อเก็บตัวแปรของแลดเดอร์สำหรับการแสดงผล และลูปต่างๆ

รูปที่ 3.72 สร้างพารามิเตอร์สำหรับบล็อก NORM_X และ SCALE_X

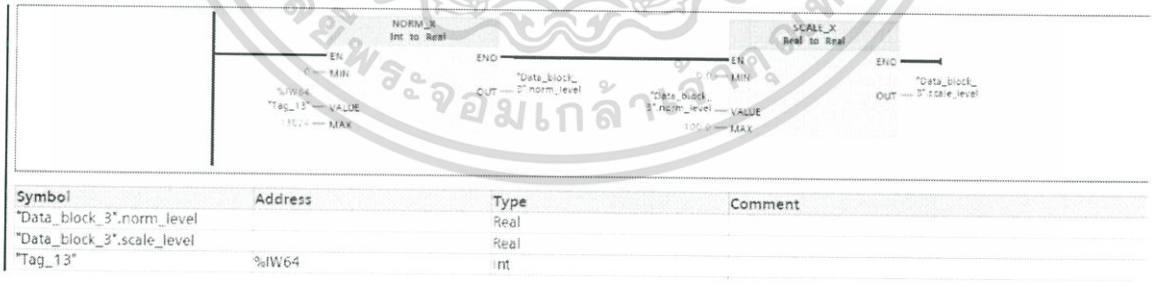
14. เขียนโปรแกรมเพื่อรับค่าแรงดันไฟฟ้าที่ได้จากขา Analog In ของพีแอลซี โดยเลือก Basic Instruction > Conversion Operations > NORM_X และ SCALE_X นำไปวางไว้ดังรูปที่ 3.69



รูปที่ 3.73 การเพิ่มบล็อก NORM_X และ SCALE_X

บล็อก NORM_X จะแปลงค่า VALUE เป็นเปอร์เซ็นต์โดยที่ 100% มีค่าเท่ากับ 1 เมื่อเทียบด้วยค่า MAX และ MIN ส่วนบล็อก SCALE_X จะแปลงค่าเปอร์เซ็นต์ที่ขา VALUE เป็นค่าจริงเมื่อเทียบด้วยค่า MAX และ MIN

15. ใส่ค่าพารามิเตอร์ดังนี้

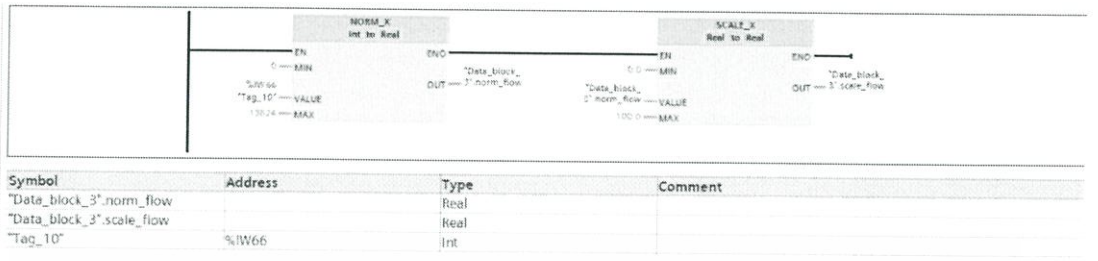


Symbol	Address	Type	Comment
"Data_block_3".norm_level		Real	
"Data_block_3".scale_level		Real	
"Tag_13"	%IW64	Int	

รูปที่ 3.74 การใส่พารามิเตอร์ลงบล็อก NORM_X และ SCALE_X สำหรับ Level

โดย IW64 คือค่าแรงดันไฟฟ้าที่ PLC รับได้จากขา Analog In ซึ่งต่ออยู่กับ Pressure Sensor แบบไร้สาย และมีค่าสูงสุดที่ 27648 จากนั้นจึงทำการแปลงค่าเปอร์เซ็นต์ที่ได้จาก NORM_X เป็นค่าแรงดันไฟฟ้าจริงที่วัดได้จากเครื่องพีแอลซีคือ 0 ถึง 10 โวลต์ แต่ค่าแรงดันไฟฟ้าสูงสุดที่ส่งมาจาก Arduino มีแค่ 5 โวลต์ จึงต้องใส่ค่า MAX เป็น 13824

16. ทำซ้ำอีกครั้งโดยใช้ Address IW66 แทน ซึ่ง IW66 คือค่าแรงดันไฟฟ้าที่พีแอลซีรับได้จากขา Analog In ซึ่งต่ออยู่กับ Flow Sensor แบบไร้สาย และใช้บล็อก SCALE_X แปลงค่าที่อ่านได้เป็น 0 ถึง 100%



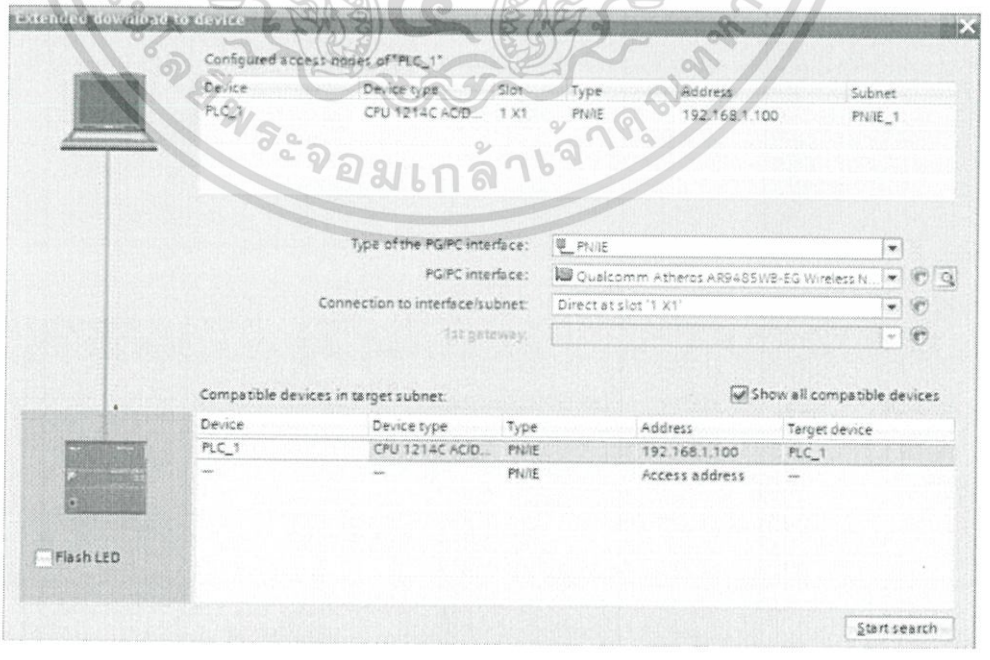
Symbol	Address	Type	Comment
"Data_block_3".norm_flow		Real	
"Data_block_3".scale_flow		Real	
"Tag_10"	%IW66	Int	

รูปที่ 3.75 การใส่พารามิเตอร์ลงบล็อก NORM_X และ SCALE_X สำหรับ Flow

17. ดาวน์โหลดโปรแกรมเข้าเครื่องพีแอลซี

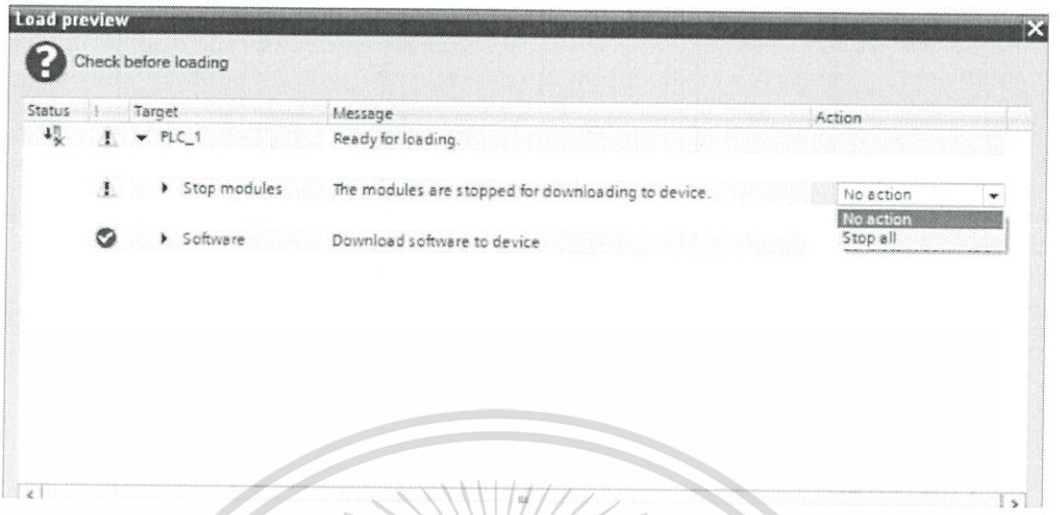
รูปที่ 3.76 การโหลดโปรแกรมลงเครื่องพีแอลซี

เลือก Wireless Network Module ของคอมพิวเตอร์ ที่ช่อง PG/PC Interfaces กด StartSearch แล้วกด Load



รูปที่ 3.77 หน้าต่าง Download to Device

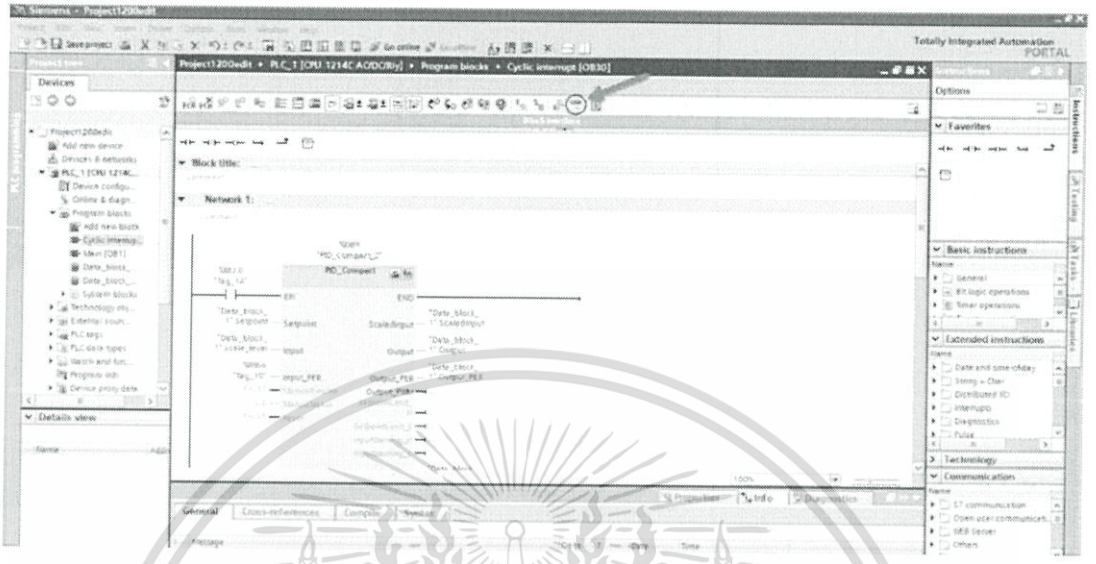
18. กด Stop all > Load > Finish



รูปที่ 3.78 หน้าต่าง Load Preview

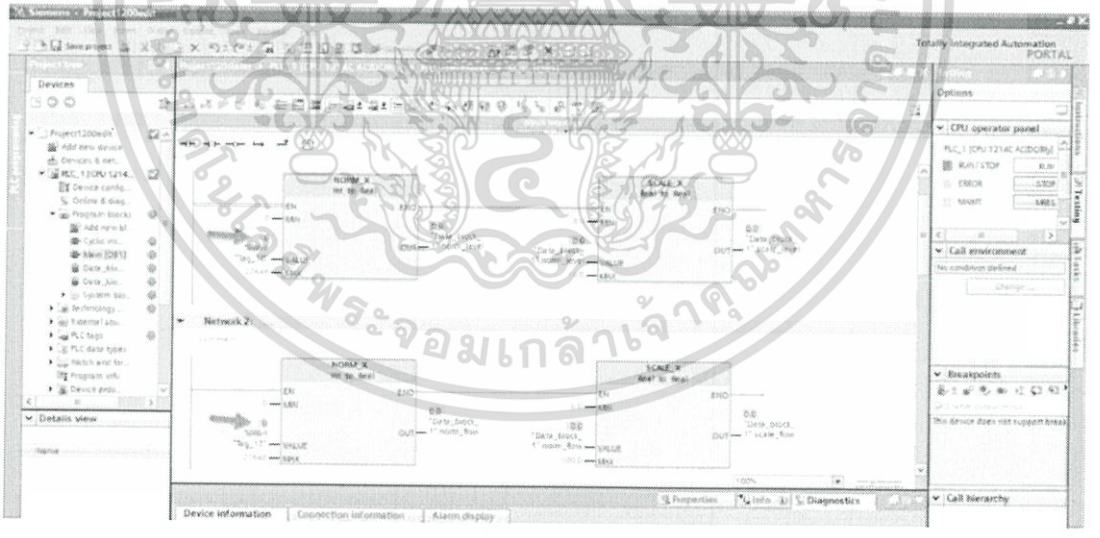
รูปที่ 3.79 หน้าต่าง Load Result

19. กดปุ่ม Monitor all เพื่อตรวจสอบว่าโปรแกรมสามารถรับค่าจากช่อง Analog In ได้หรือไม่ รวมถึงเป็นการตรวจสอบการเชื่อมต่อระหว่าง Wi-Fi Module ด้วย



รูปที่ 3.80 การ Monitor ค่า และการทำงานของโปรแกรมแลดเดอร์

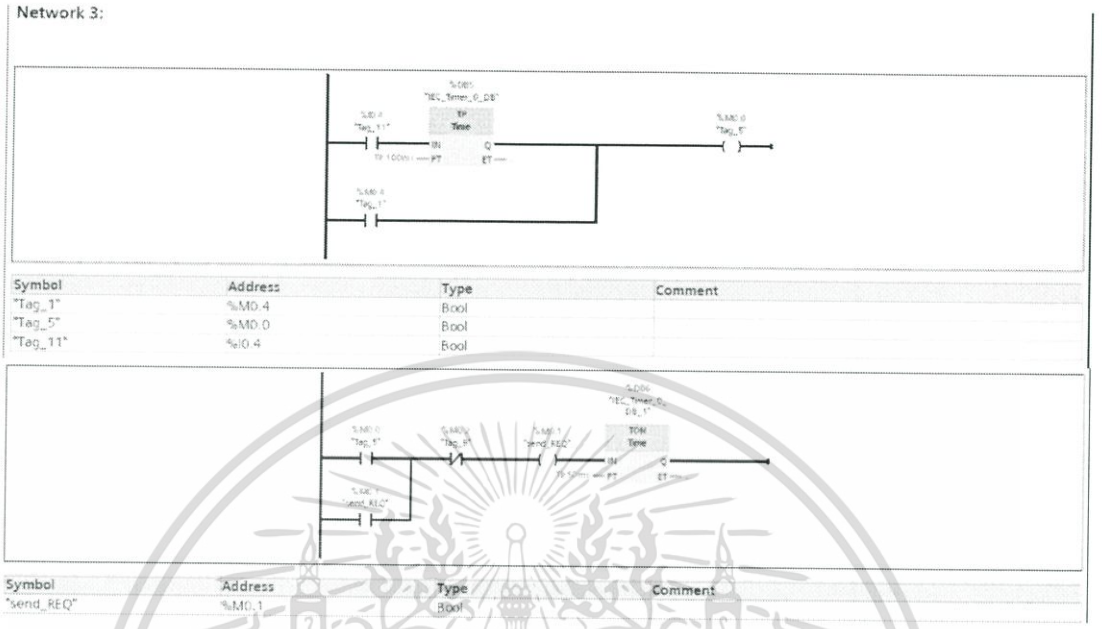
20. ตรวจสอบว่าเลขที่แสดงมีค่าตรงตามที่วัดจากเครื่องพีแอลซีหรือไม่



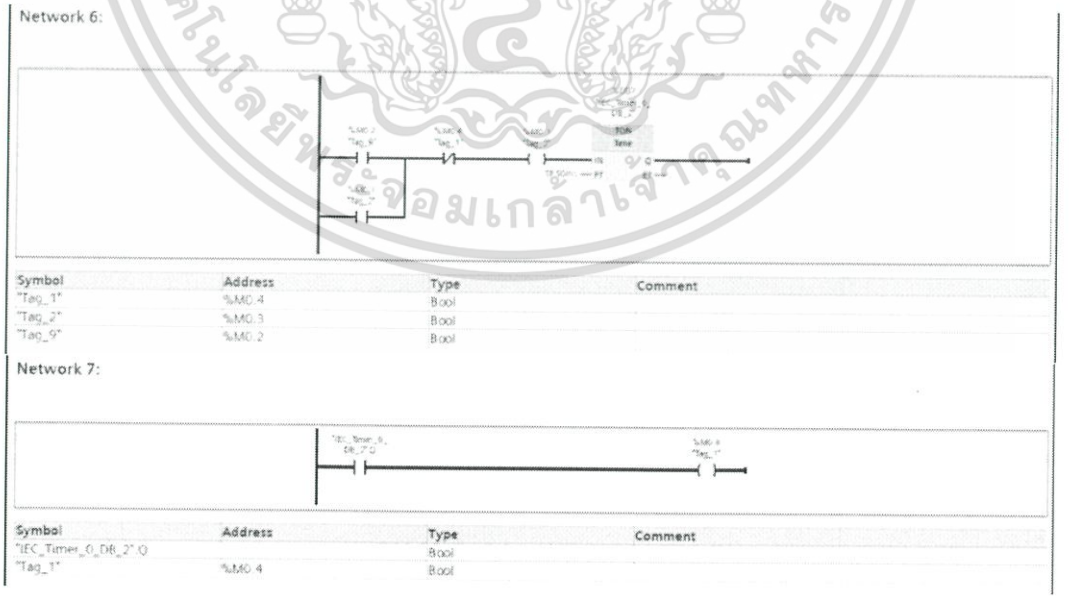
รูปที่ 3.81 ค่าที่เข้าขาแอนะล็อก IW64 และ IW66

จากนั้นกด Monitor all อีกครั้งเพื่อหยุด

21. เขียนโปรแกรมแลดเดอร์ให้บล็อก TSEND_C ส่งข้อมูลเป็นเวลา 50 ms และส่งทุกๆ 50 ms

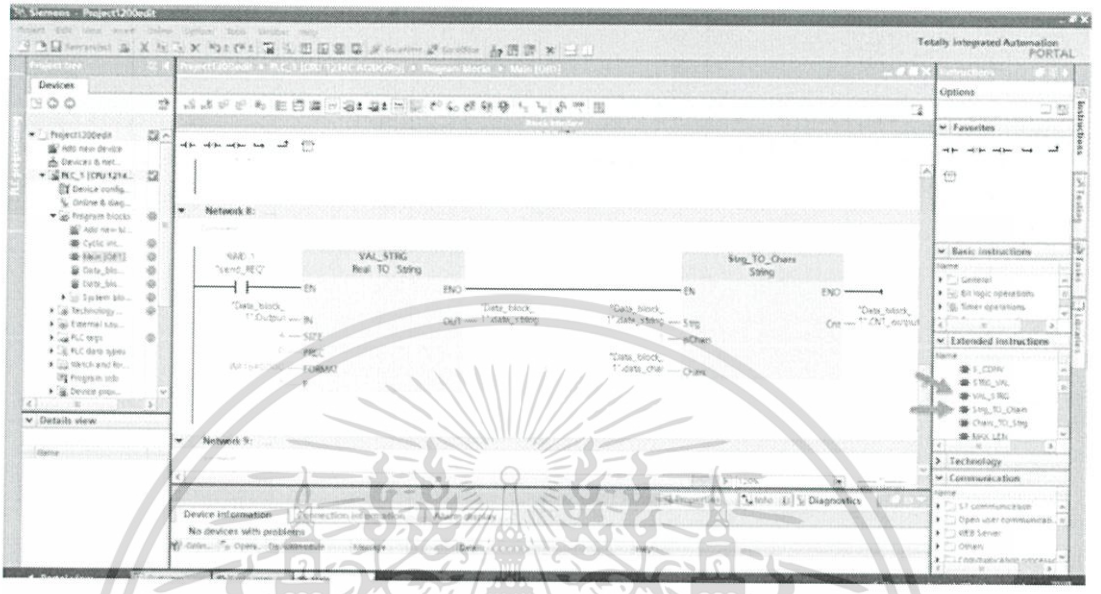


รูปที่ 3.82 โปรแกรมสำหรับบล็อก TSEND_C ส่งข้อมูลเป็นเวลา 50 ms และส่งทุกๆ 50 ms



รูปที่ 3.83 โปรแกรมสำหรับบล็อก TSEND_C ส่งข้อมูลเป็นเวลา 50 ms และส่งทุกๆ 50 ms

22. เลือก Extended instructions > String + Char > VAL_STRG เพื่อแปลงค่าเอาต์พุตของบล็อก PID_Compact จาก Real เป็น String และ Strg_TO_Chars เพื่อแปลง String ให้เป็น Array of Char เพื่อส่งผ่านบล็อก TSEND_C ไปสั่งมอเตอร์



รูปที่ 3.84 การเพิ่มบล็อก VAL_STRG และ Strg_To_Chars

โดย IN – ข้อมูลที่ต้องการแปลงเป็น String

SIZE – ขนาดของ String

PREC – จำนวนเลขหลังจุดทศนิยม

FORMAT – รูปแบบของ String โดยสามารถดูค่าได้จากคู่มือการใช้งาน

P – บิตแรกของ String ที่จะใช้เก็บข้อมูล

OUT – ตำแหน่งที่จะเก็บ String

Strg – String ที่ต้องการแปลงเป็น Char

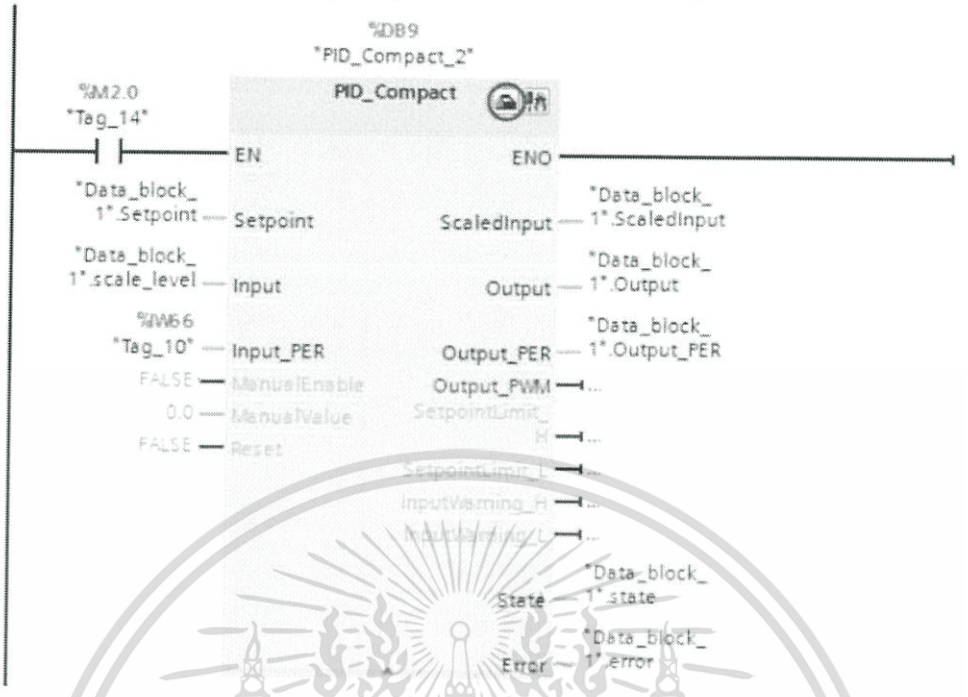
pChars – ตำแหน่งเริ่มต้นใน Array ของ Char ตัวแรก

Chars – ตำแหน่งที่จะเก็บ Chars

Cnt – จำนวนของตัวอักษรที่ซ้ำ

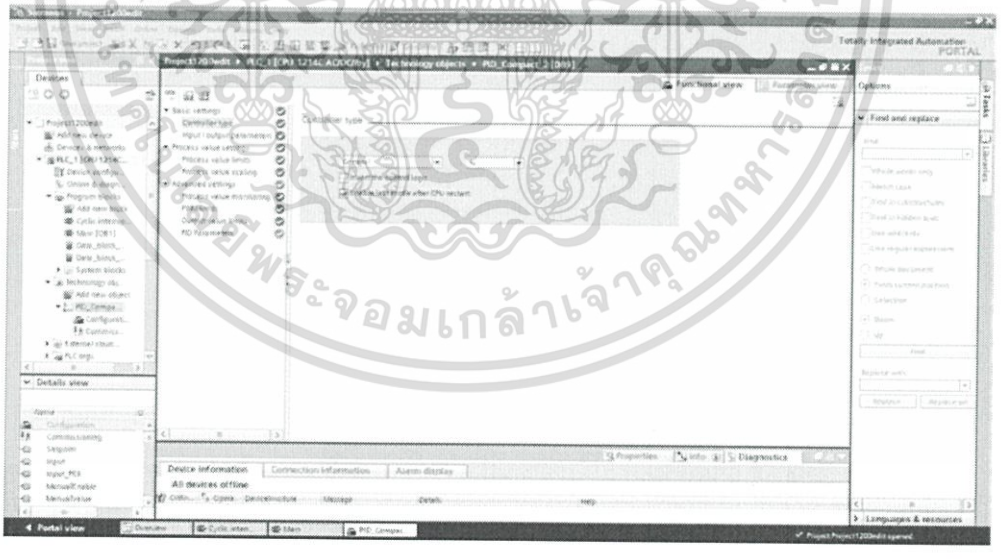
23. ปรับค่าพารามิเตอร์ต่างๆ ของบล็อก PID_Compact โดยต้องปรับที่ Secondary Controller ก่อนแล้วจึงค่อยปรับ Primary Controller

24. เลือกบล็อก PID_Compact > Opens the configuration window

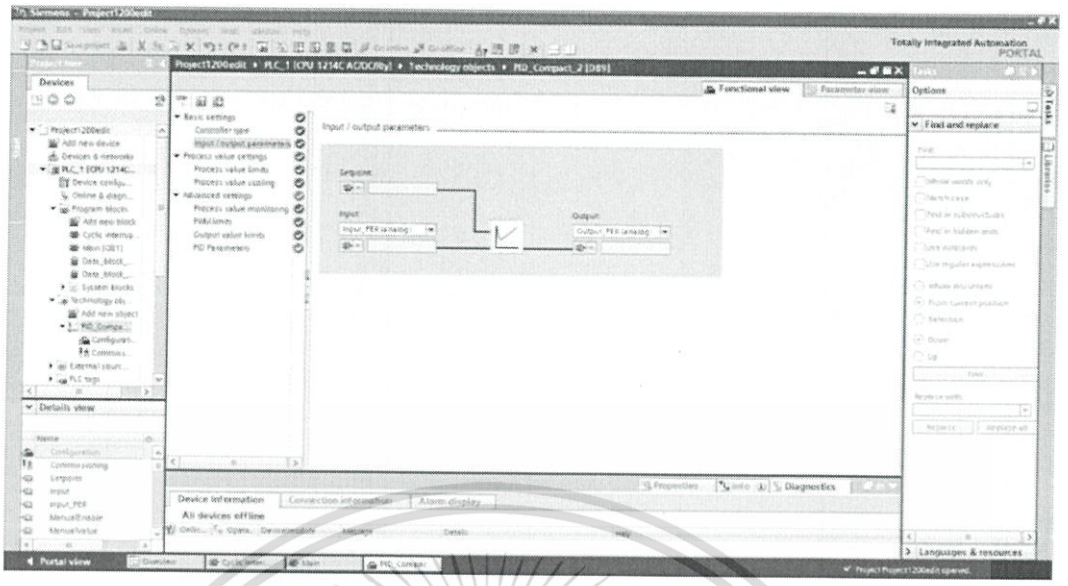


รูปที่ 3.85 การเปิดหน้า configuration window

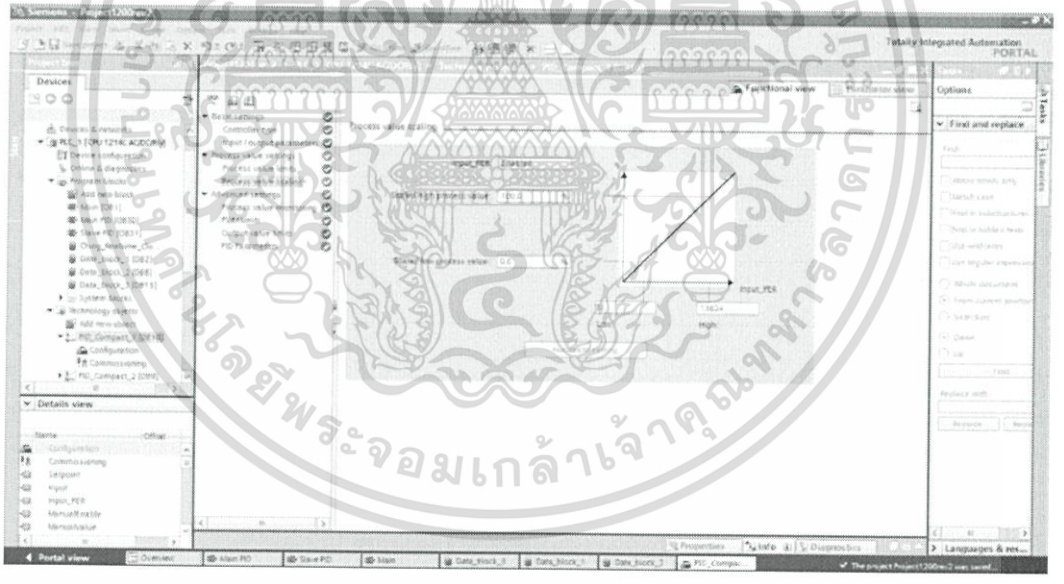
25. ตั้งค่าต่างๆ ดังต่อไปนี้



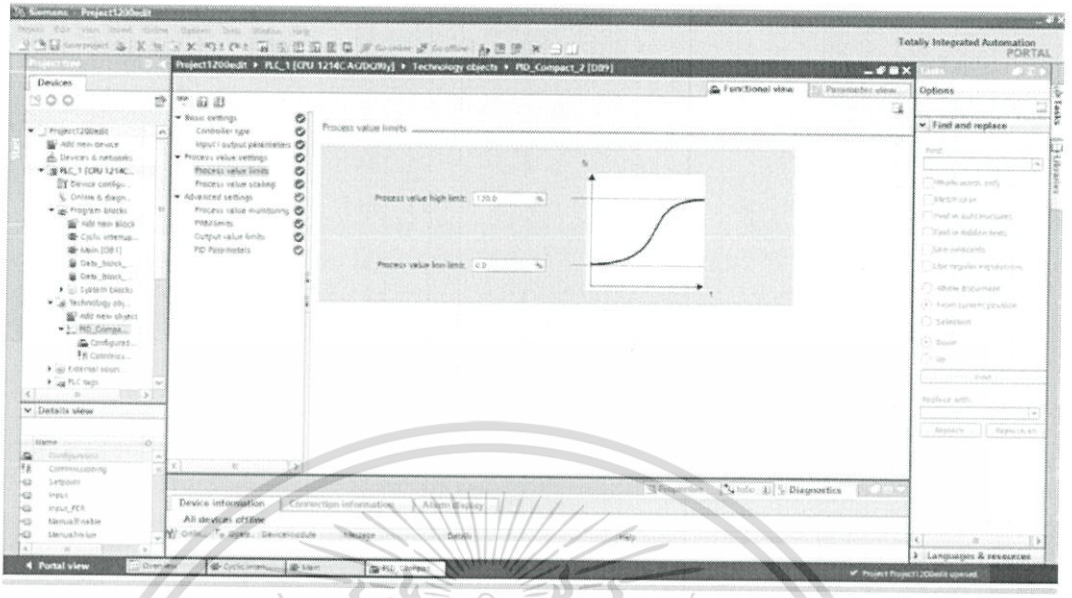
รูปที่ 3.86 หน้าต่าง configuration window



รูปที่ 3.87 การตั้งค่า Input/Output Parameter

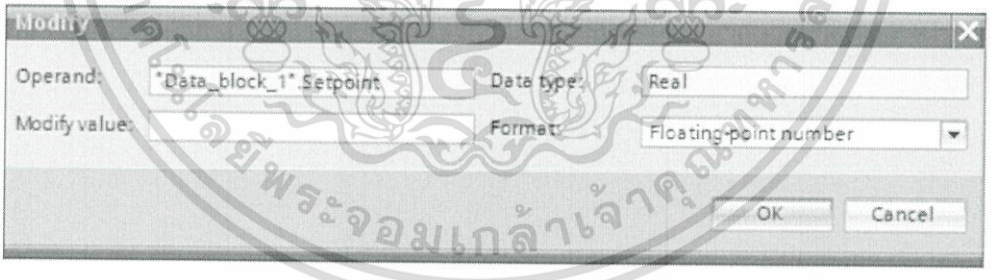


รูปที่ 3.88 การตั้งค่า Process Value Scaling



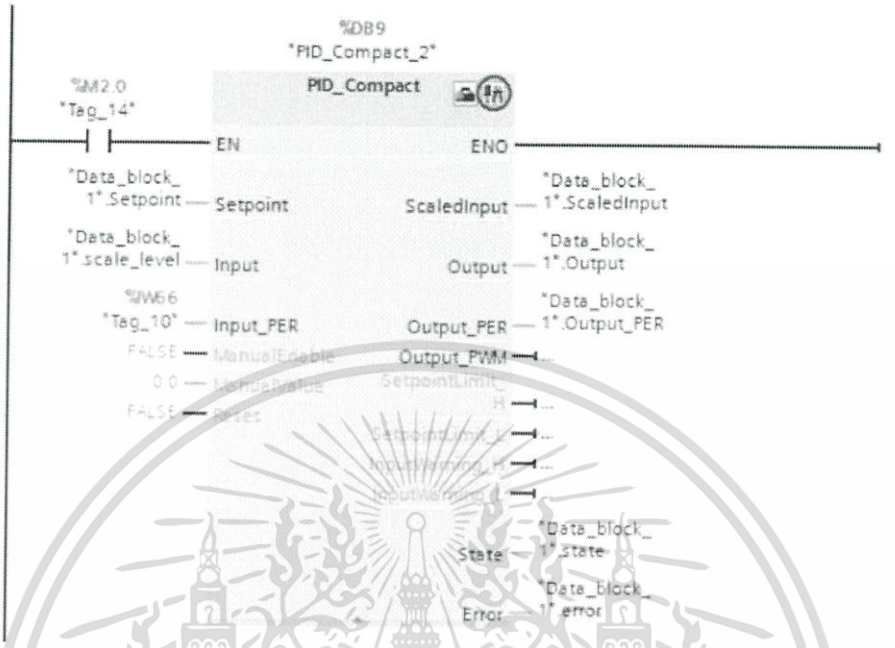
รูปที่ 3.89 การตั้งค่า Process Value Limits

26. ดาวน์โหลดโปรแกรมเข้าพีแอลซี และ Monitor > คลิกขวาที่ Setpoint ของบล็อก PID_Compact > Operands > Modify Operands > ปรับเป็นค่าเป้าหมายที่ต้องการ



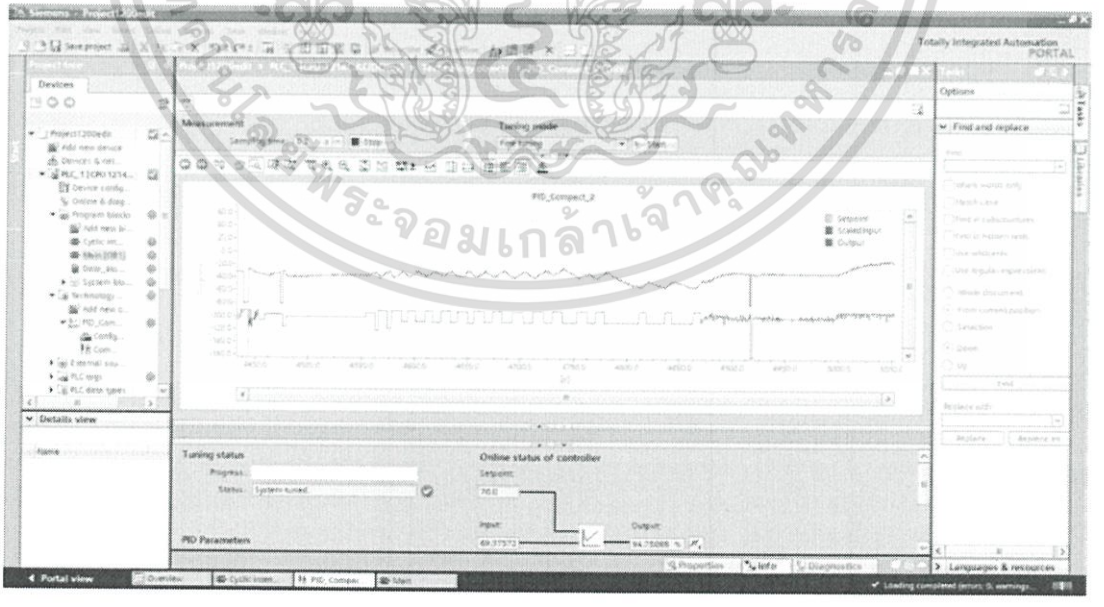
รูปที่ 3.90 การปรับเปลี่ยนค่าเป้าหมาย

27. เลือกบล็อก PID_Compact > Open the commissioning window



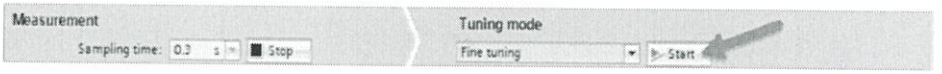
รูปที่ 3.91 การเปิดหน้า commissioning window

28. Measurement > Start เพื่อดูกราฟของระบบ และ Tuning mode > Fine Tuning > Start เพื่อปรับค่าพีไอดีให้เข้ากับระบบ



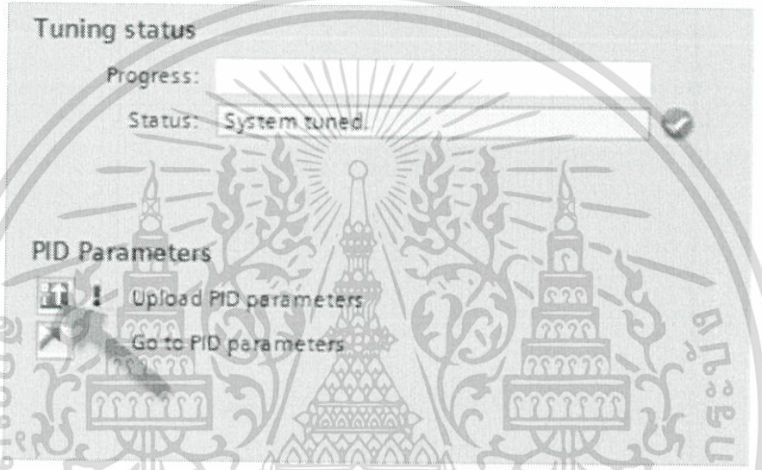
รูปที่ 3.92 การปรับค่าพีไอดี (1)

29. คลิก Start เพื่อเริ่มการปรับค่าพารามิเตอร์อัตโนมัติ



รูปที่ 3.93 การปรับค่าพีไอดี (2)

30. เมื่อปรับค่าเสร็จ คลิก Upload PID Patameters > ดาวน์โหลด > Moniter all

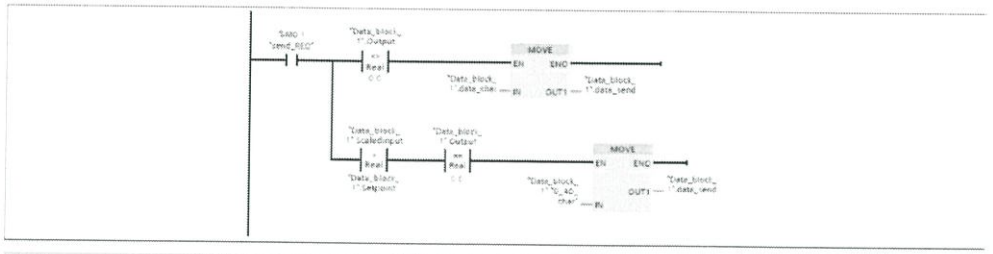


รูปที่ 3.94 การอัพโหลดพารามิเตอร์ของบล็อก PID_Compact

31. ทำเช่นเดียวกันกับบล็อก PID_Compact ของ Primary Controller

32. สร้างเงื่อนไขให้ส่งค่า 40 แทน 0 เมื่อเอาต์พุตของ PID_Compact เป็น 0

เนื่องจากมอเตอร์จะสั่งค่าเก่าเมื่อได้รับค่า 0 จึงใช้สั่งค่า 40 ซึ่งทำให้มอเตอร์ทำงาน แต่มีแรงไม่มากพอจะปั๊มน้ำกลับขึ้นไปได้

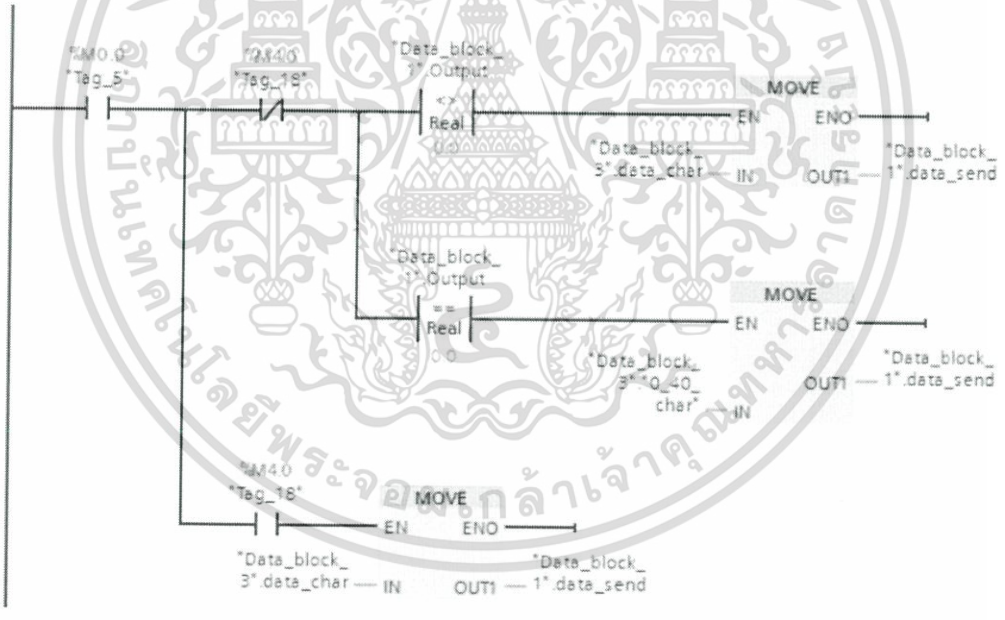


Symbol	Address	Type	Comment
"Data_block_1"."0_40_char"		Array	
"Data_block_1".data_char		Array	
"Data_block_1".data_send		Array	
"Data_block_1".Output		Real	
"Data_block_1".ScaledInput		Real	
"Data_block_1".Setpoint		Real	
"send_REQ"	%M0.1	Bool	

รูปที่ 3.95 เงื่อนไขการสั่งงานมอเตอร์แบบอัตโนมัติ

33. ดาวน์โหลด > Moniter all เพื่อทดสอบ

34. เขียนโปรแกรมเพื่อสร้างเงื่อนไขสำหรับการสั่งมอเตอร์ด้วยตนเอง



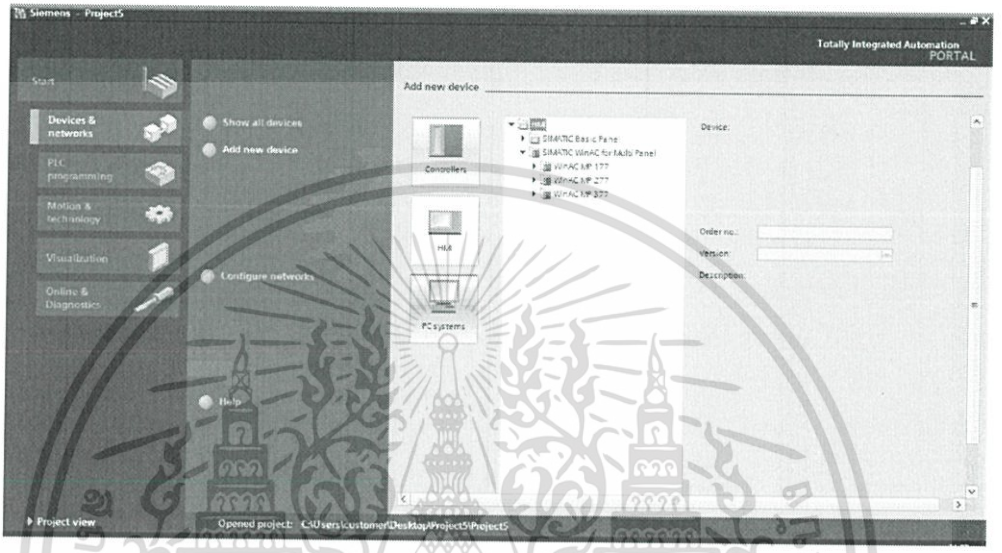
รูปที่ 3.96 แลดเดอร์สำหรับการสั่งงานมอเตอร์แบบไม่อัตโนมัติ (Manual)

3.8 โปรแกรม WinCC RT Advanced

WinCC Runtime Advanced V13 เป็นโปรแกรมที่ใช้สำหรับสร้างหน้ากราฟิก สำหรับติดตาม และควบคุมระบบผ่านหน้าจอคอมพิวเตอร์ หรือที่เรียกว่า Human Interface (HMI)

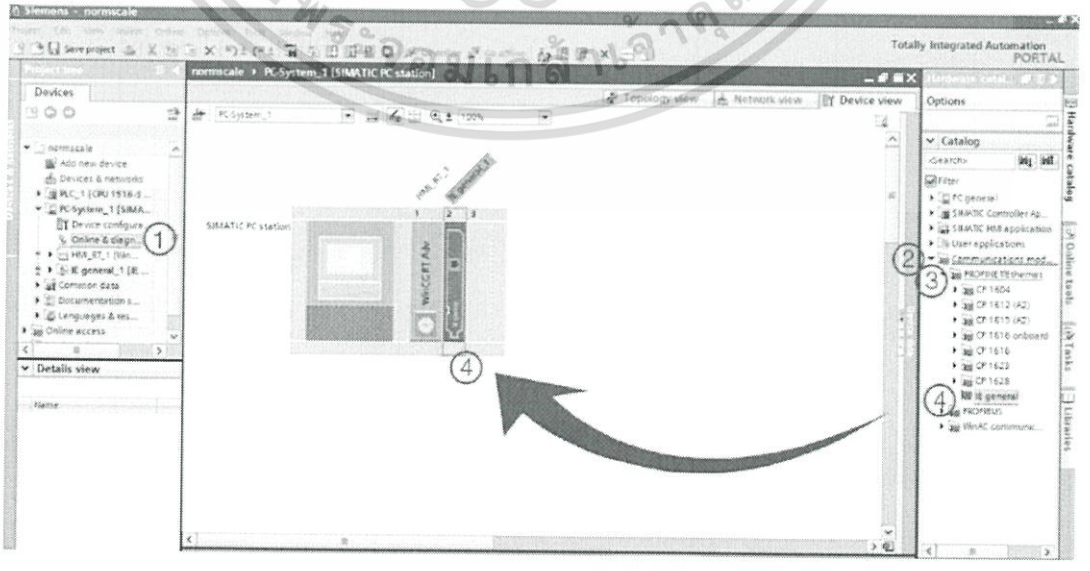
เพิ่มและตั้งค่าอุปกรณ์

1. กด Add new device > PC System > WinCC RT Advanced



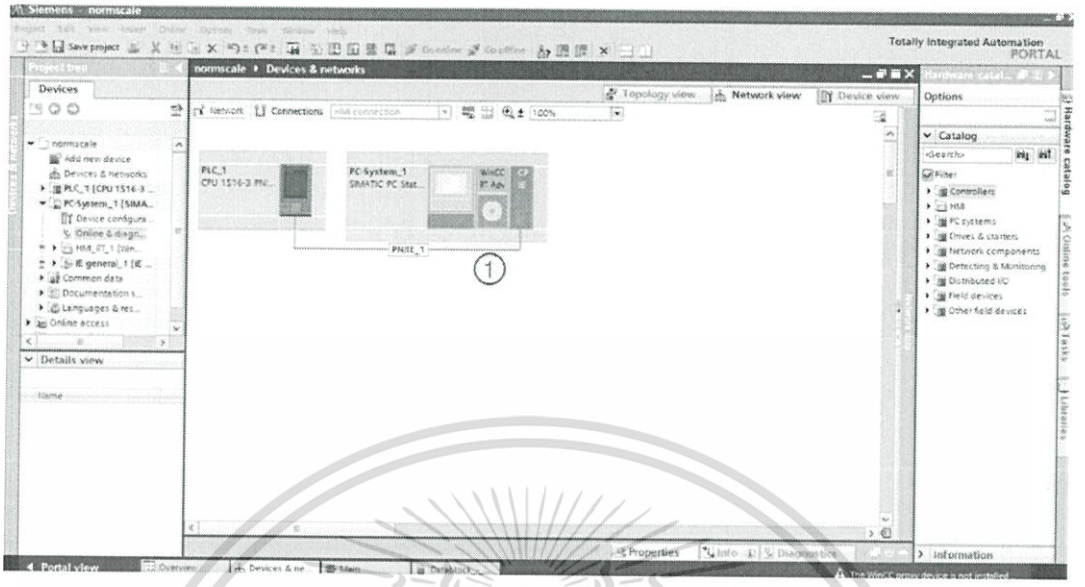
รูปที่ 3.97 หน้าแสดงผล Portal View

2. คลิก Online & diagnostics > Communication modules > PROFINET/Ethernet > ลาก IE general ไปไว้ในช่องที่ 2 แล้วกด ที่มุมซ้ายบนของหน้า Device view



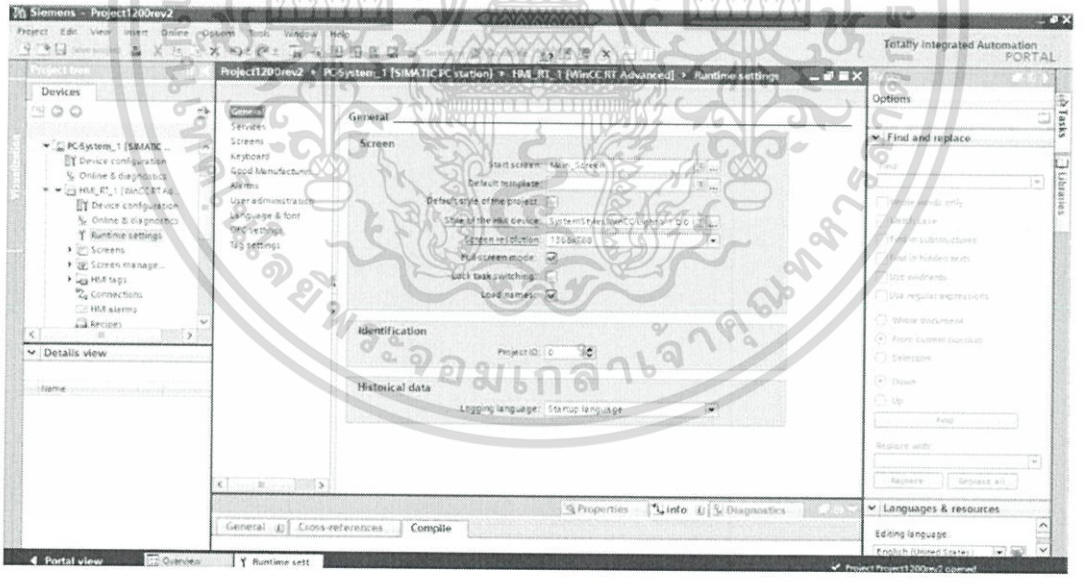
รูปที่ 3.98 หน้าแสดงผล Project View

3.ลากสายเชื่อมระหว่างพีแอลซีกับ Communications modules



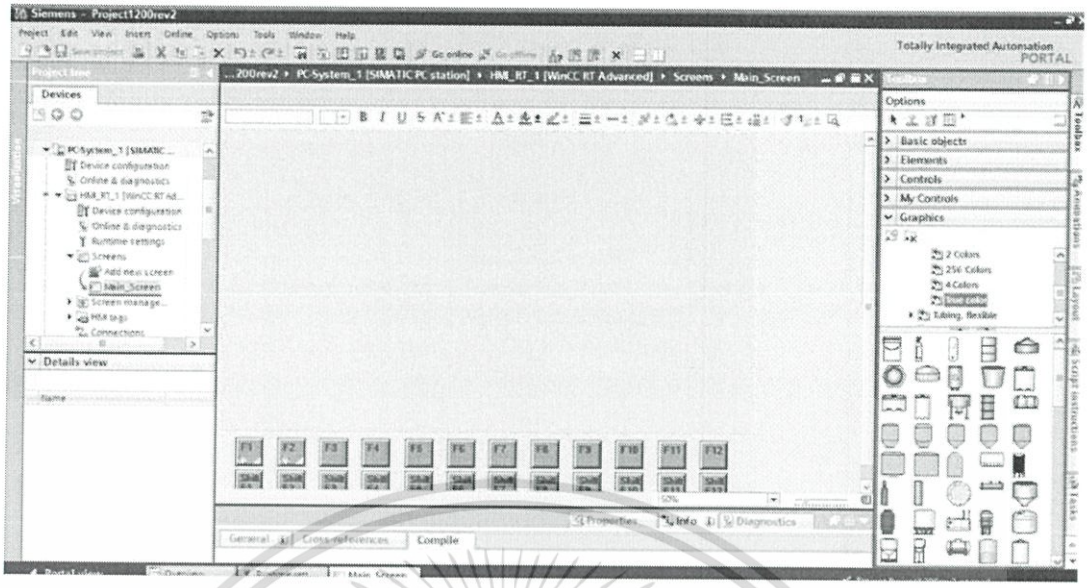
รูปที่ 3.99 หน้า Device & Networks แสดงการเชื่อมต่อของอุปกรณ์ต่างๆ

4. ตั้งค่ารูปแบบหน้าจอ (Style of the HMI Device) และขนาดของหน้าจอ (Screenresolution) ตามต้องการ



รูปที่ 3.100 หน้า Runtime Setting สำหรับตั้งค่าหน้าจอ HMI

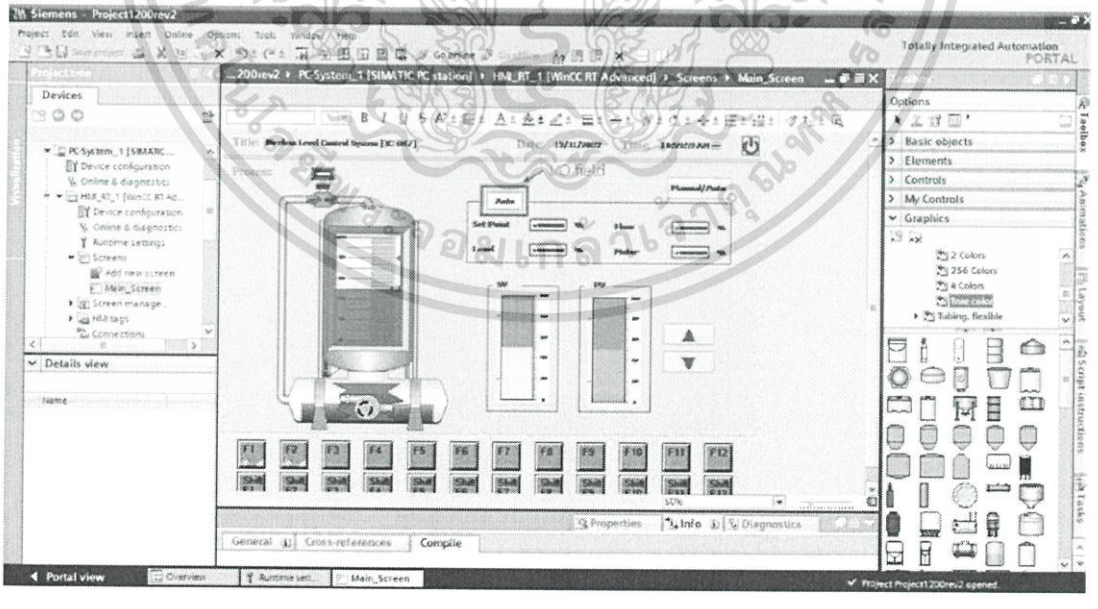
5.กด Screen > Add new screen จากนั้นสร้างหน้า HMI ตามที่ใช้งาน โดยสามารถใช้กราฟิกที่โปรแกรมมีให้ที่แถบ Toolbox ขวามือ



รูปที่ 3.101 หน้า Screen สำหรับสร้าง และปรับแต่งหน้า HMI

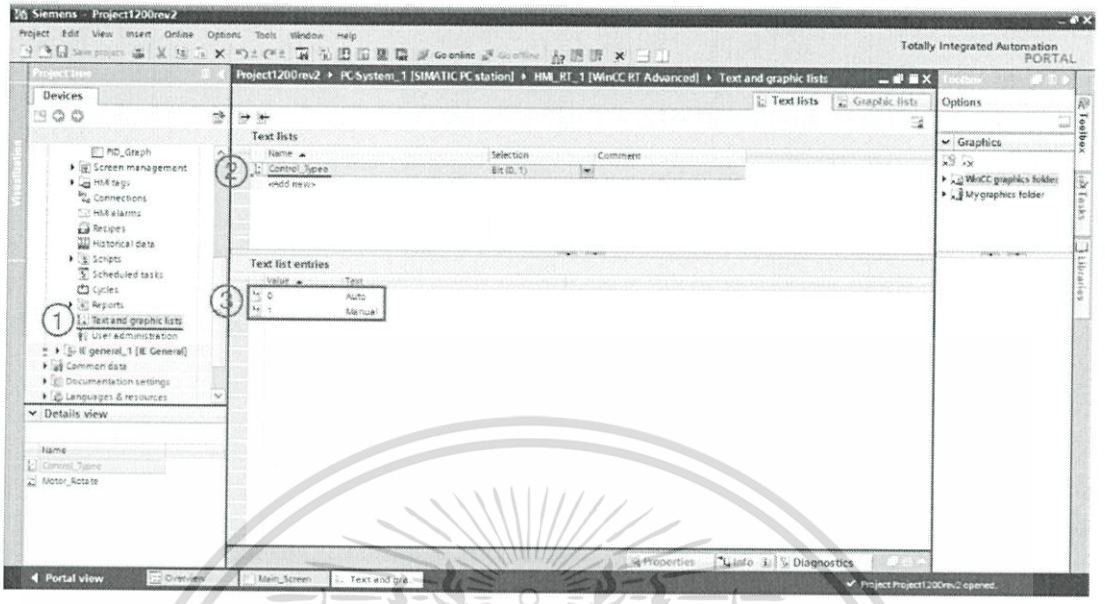
การสร้าง Symbolic I/O Field แสดงสถานะ

Symbolic I/O Field คือ Dynamic Text Box ซึ่งสามารถนำมาใช้ในการแสดงสถานะได้ตามที่ต้องการ โดยค่าเริ่มต้น ของตัว Symbolic I/O Field คือ 0 กับ 1 ทั้งนี้สามารถตั้งค่า Text List ให้ Symbolic I/O Field ได้ตามที่ต้องการ



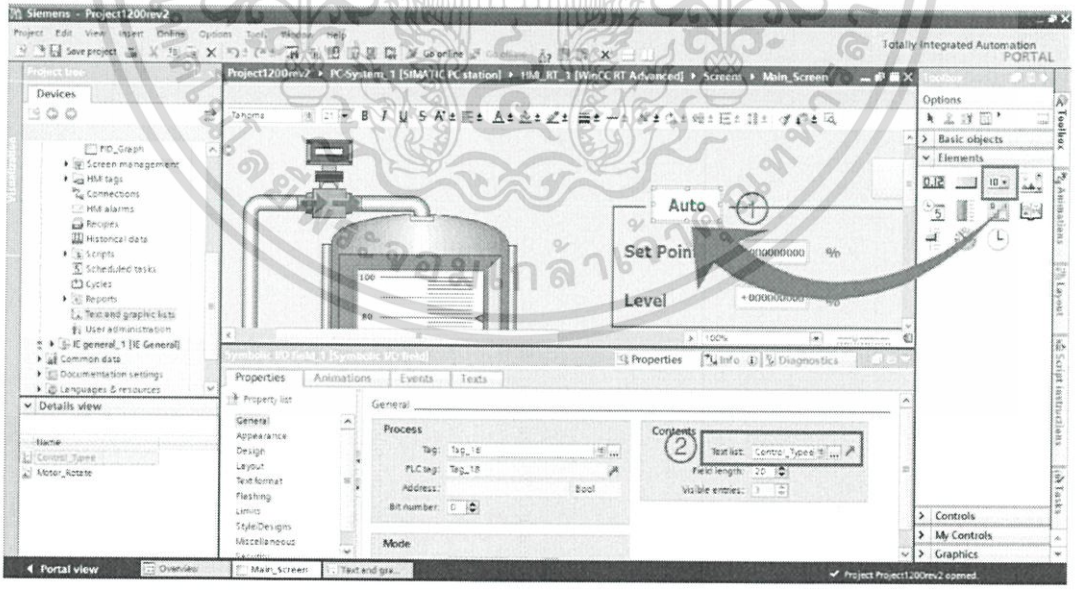
รูปที่ 3.102 ตำแหน่งตัวบอกสถานะควบคุม

1. สร้าง Text List สถานะที่ต้องการจะใช้ในการแสดงผล



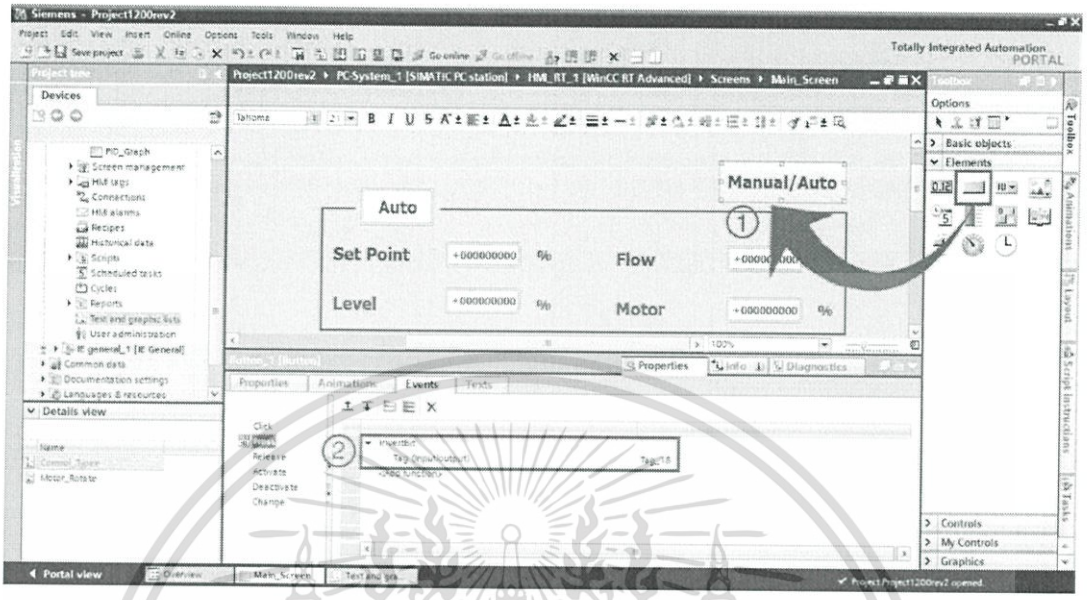
รูปที่ 3.103 หน้า Text and Graphic Lists

2. ลาก Symbolic I/O Field ไปที่พื้นที่หน้าจอสำหรับแสดงผลแล้ว Tag Text List : Control_Typee ที่ทำไว้ในข้อ 1



รูปที่ 3.104 การสร้าง Symbolic I/O Field และ Tag Text List

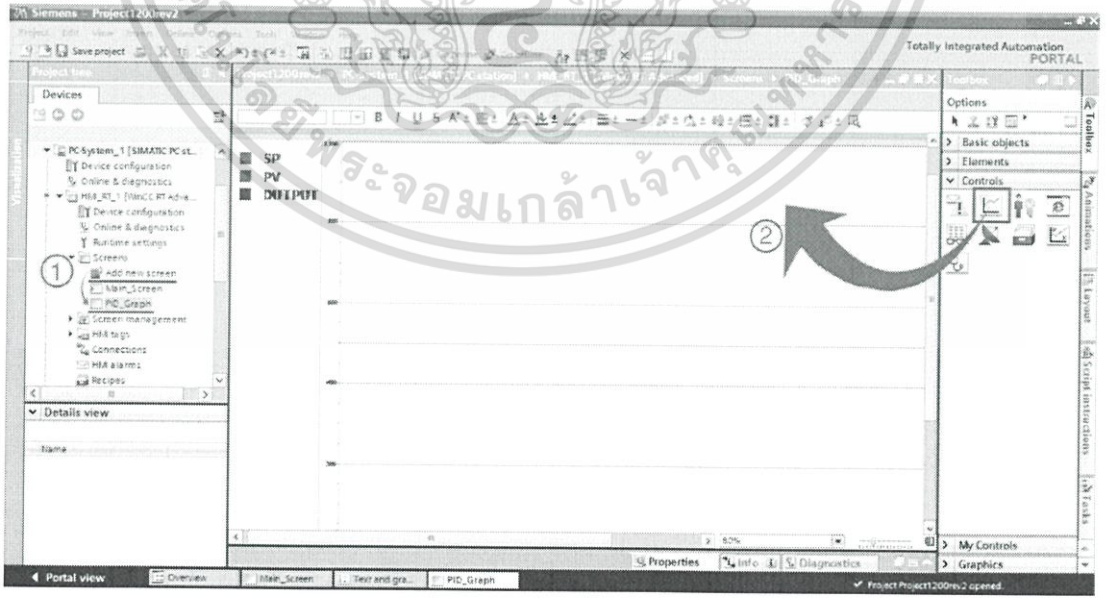
3. สร้าง Button Switch สำหรับการเปลี่ยนรูปแบบการควบคุม โดยการลาก Button ไปที่พื้นที่หน้าจอสำหรับแสดงผล แล้ว Tag Event ขณะกดปุ่ม (Press) เป็น InvertBit



รูปที่ 3.105 วิธีการเพิ่ม Event สำหรับ Button Switch Element

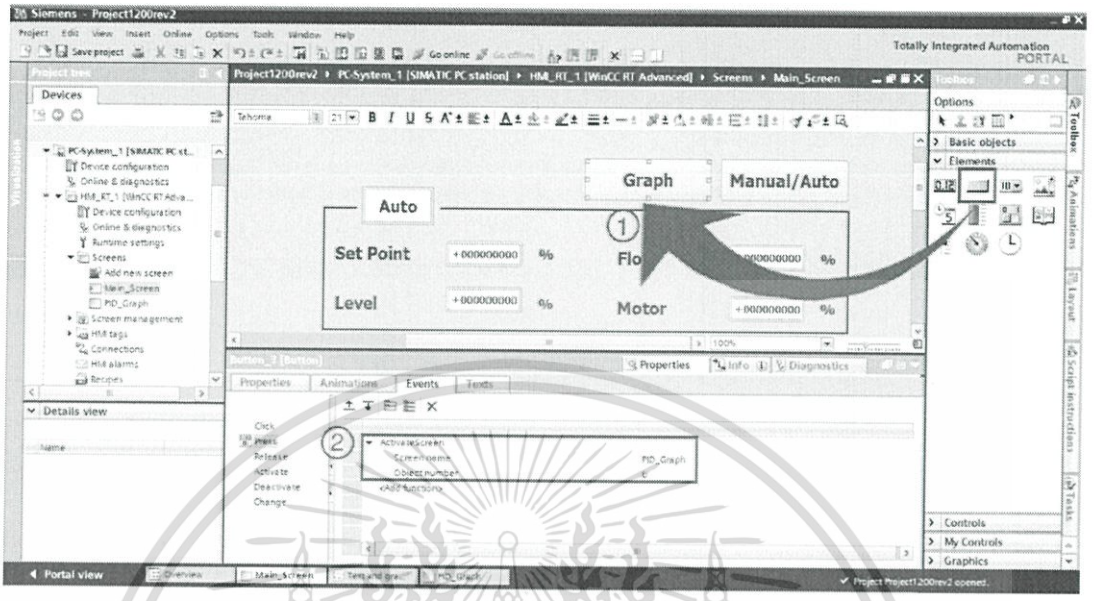
การสร้างกราฟแสดงแนวโน้มบล็อกพีไอดี

1. เลือก Add New Screen > เลือกอุปกรณ์ TrendView

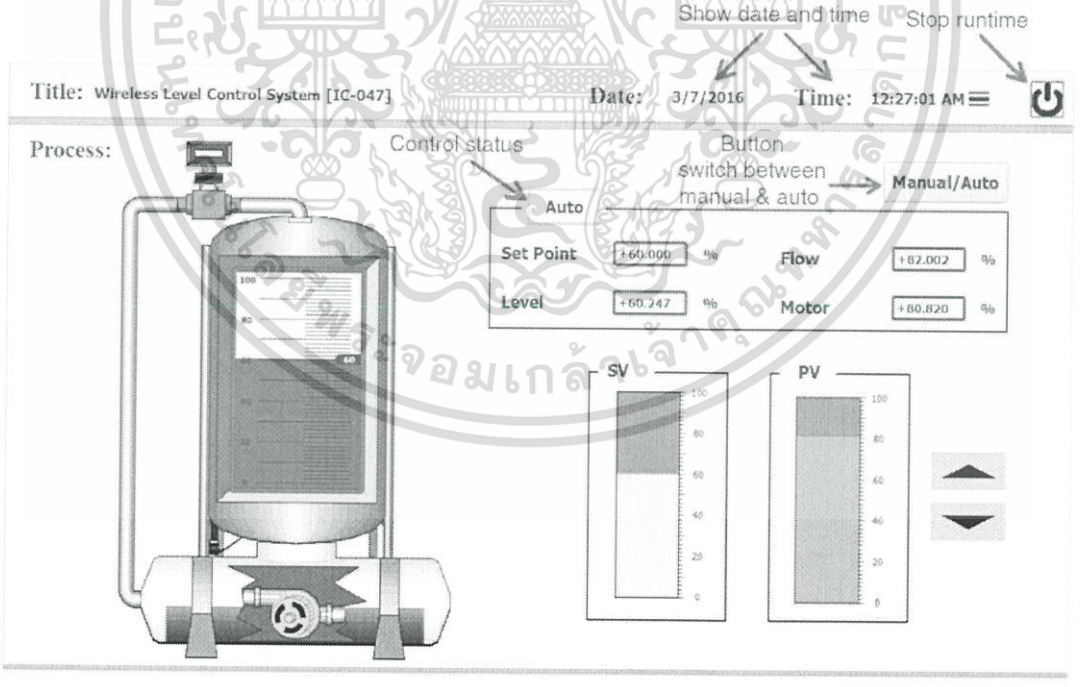


รูปที่ 3.106 การสร้างหน้าแสดงผลแนวโน้มกราฟ

2. สร้างปุ่มสำหรับเรียกใช้งานหน้ากราฟจากหน้า Main_Screen โดยการสร้าง Button แล้วเพิ่ม Event ขณะกดปุ่ม (Press) เป็น ActivateScreen



รูปที่ 3.107 การสร้างปุ่มสำหรับเรียกใช้หน้าจอแสดงแนวโน้มกราฟของบล็อกพีไอดี



รูปที่ 3.108 ส่วนประกอบหน้าเชื่อมต่อของระบบควบคุมระดับน้ำแบบไร้สาย

รูปที่ 3.109 รูปแบบหน้าต่างแสดงกราฟแนวโน้มของบล็อกพีไอดี

บทที่ 4
ผลการดำเนินงาน

4.1 ผลการติดตั้งโครงสร้างระบบควบคุมระดับน้ำแบบไร้สาย

จากการดำเนินงานในการออกแบบโครงสร้างด้วยโปรแกรม SolidWorks ในการประกอบชิ้นส่วนต่างๆ เข้าด้วยกัน เมื่อได้แบบตามที่ต้องการแล้วจึงดำเนินการประกอบชิ้นงานจริง

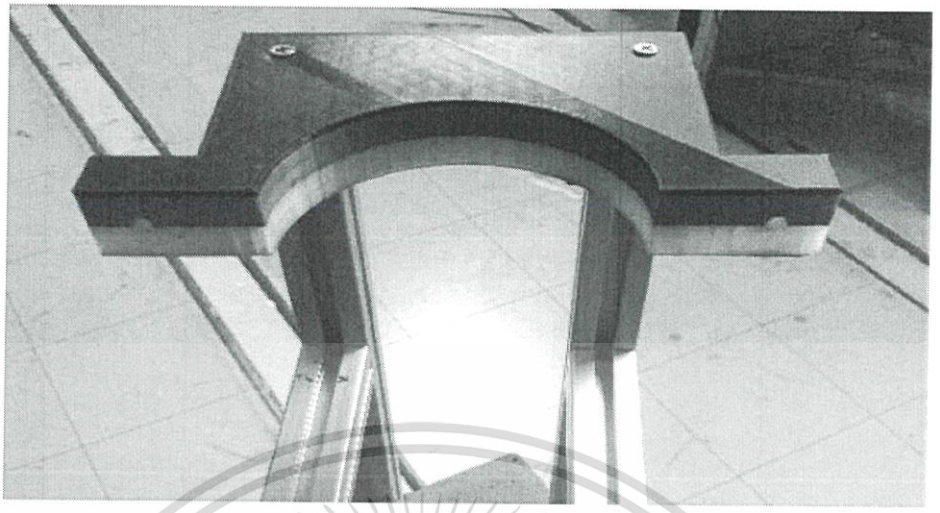
4.1.1 ประกอบอลูมิเนียมโปรไฟล์

สามารถประกอบอลูมิเนียมโปรไฟล์โดยส่วนฐานที่มีขนาด 330 x 160 มิลลิเมตร ให้สามารถล็อคถังพักน้ำได้พอดี และที่วางสำหรับหลอดวัดระดับน้ำสามารถวางและรับน้ำหนักของแทงค์สำหรับควบคุมระดับน้ำได้ โดยที่มีระยะห่างจากถังพักน้ำเท่ากับ 165 มิลลิเมตร เพื่อต่อกับบอลวาล์ว แล้วให้ได้ระยะที่เวลาปล่อยน้ำจะไม่กระเด็นแรงจนเกินไป

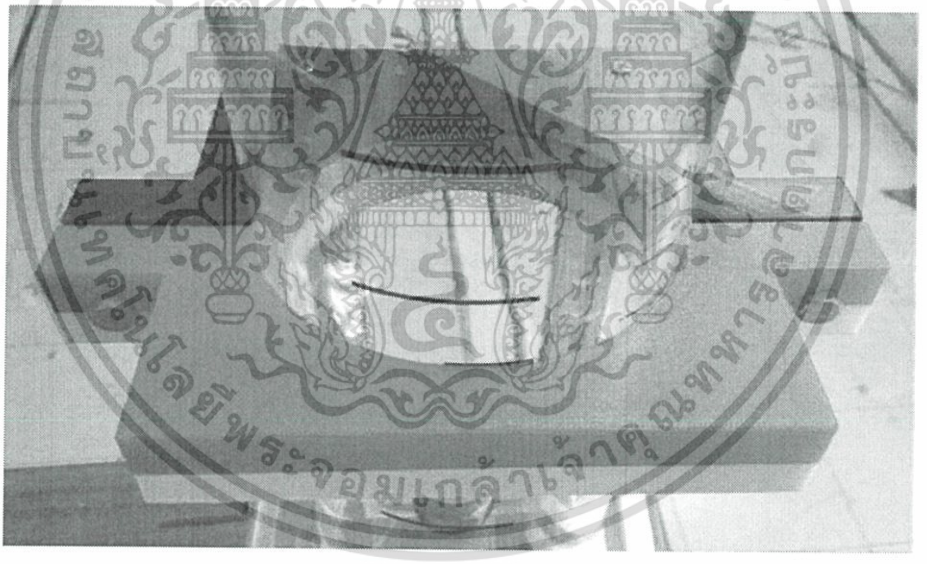
รูปที่ 4.1 ฐานอลูมิเนียมโปรไฟล์

4.1.2 การต่อ Clamp Lock เข้ากับอลูมิเนียมโปรไฟล์และแทงค์สำหรับควบคุมระดับน้ำ

นำ Clamp Lock ที่ได้จากการขึ้นรูปด้วยเครื่องพิมพ์ 3 มิติ โดยแบ่งเป็น 2 ส่วนคือ ส่วนที่ยึดติดกับอลูมิเนียมโปรไฟล์และส่วนที่ล็อคแทงค์สำหรับควบคุมระดับน้ำ

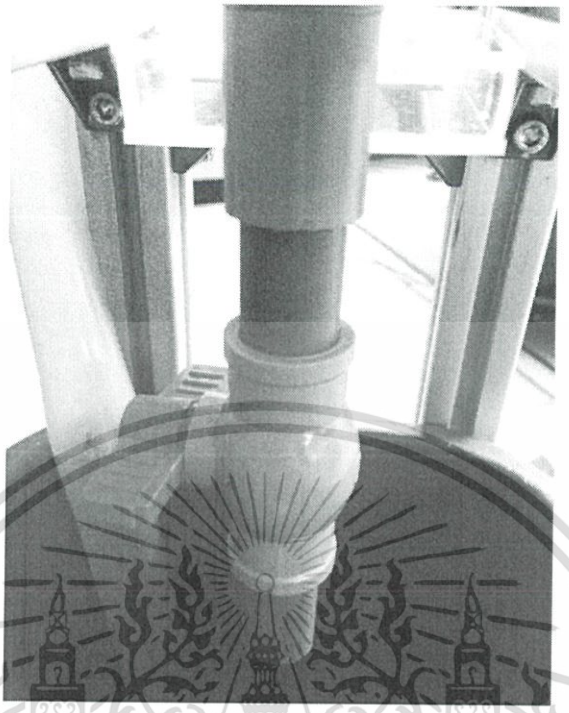


รูปที่ 4.2 Clamp Lock ส่วนที่ยึดกับอลูมิเนียมโปรไฟล์



รูปที่ 4.3 Clamp Lock ส่วนที่ล็อคแทงค์สำหรับควบคุมระดับน้ำ

4.1.3 การต่อบอลวาล์วกับแทงค์สำหรับควบคุมระดับน้ำเพื่อปล่อยน้ำลงถังพักน้ำ

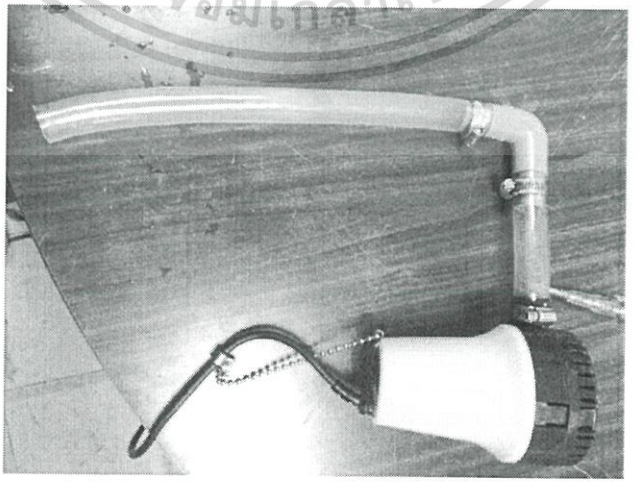


รูปที่ 4.4 ส่วนบอลวาล์วที่ต่อกับแทงค์สำหรับควบคุมระดับน้ำ

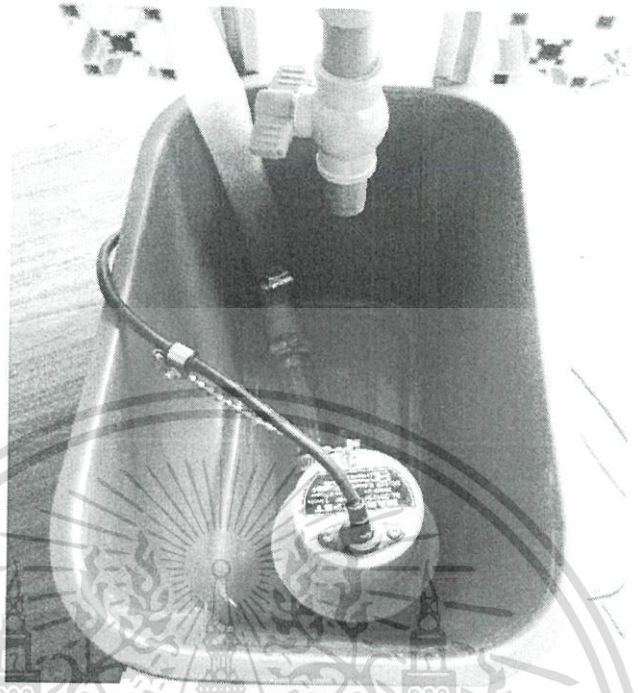
4.2 ผลการติดตั้งชุดอุปกรณ์และแผงควบคุม

สามารถติดตั้งปั๊มน้ำ, Level Sensor, Flow Sensor, วงจรปั๊มน้ำ, วงจรสำหรับ Level Sensor และวงจรสำหรับ Flow Sensor ได้อย่างสมบูรณ์ และได้ทำการทดสอบการทำงานของอุปกรณ์พบว่าอุปกรณ์สามารถทำงานได้ปกติ

4.2.1 การต่อปั๊มน้ำเข้ากับสายยาง และท่อ PVC เพื่อปั๊มน้ำเข้าสู่ท่อวัดระดับน้ำ



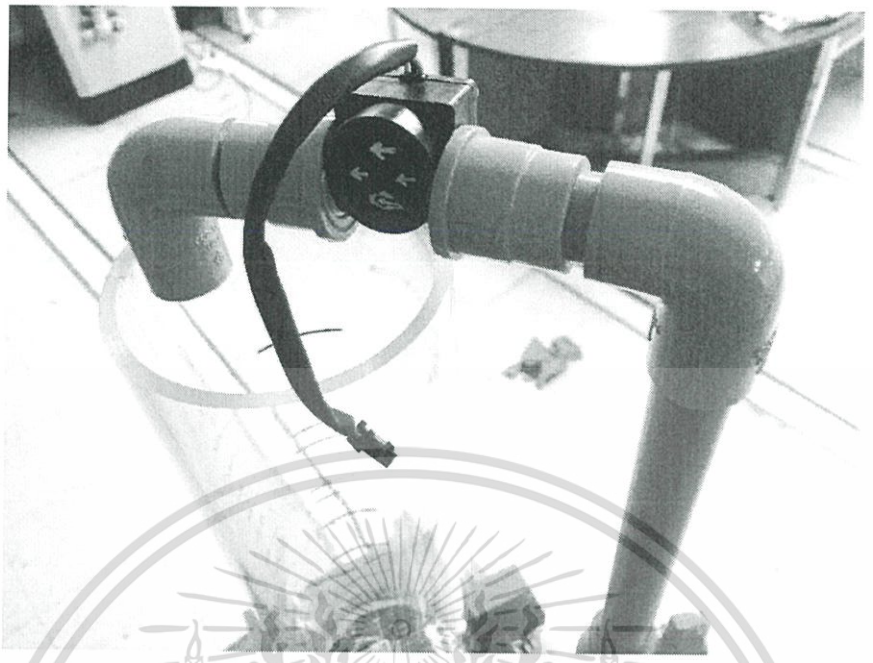
รูปที่ 4.5 ปั๊มน้ำ



รูปที่ 4.6 ตำแหน่งที่วางปั๊มน้ำ

4.2.2 ติดตั้ง Flow Sensor เชื่อมเข้ากับท่อ PVC เพื่อปล่อยน้ำลงแทงค์สำหรับควบคุมระดับน้ำ

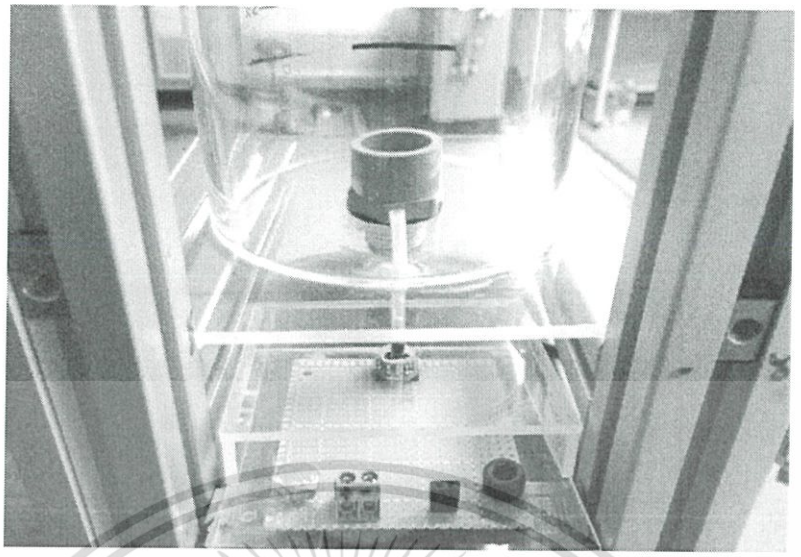
รูปที่ 4.7 การติดตั้ง Flow Sensor



รูปที่ 4.8 ตำแหน่งที่ต่อ Flow Sensor

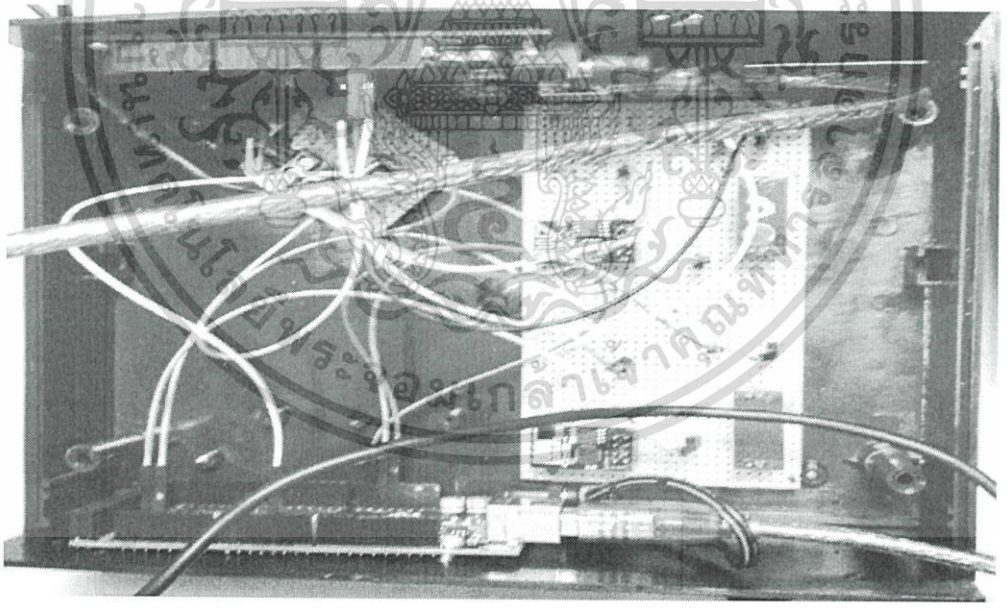
4.2.3 การติดตั้ง Level Sensor เข้าแทงค์สำหรับควบคุมระดับน้ำ

รูปที่ 4.9 การติดตั้ง Level Sensor

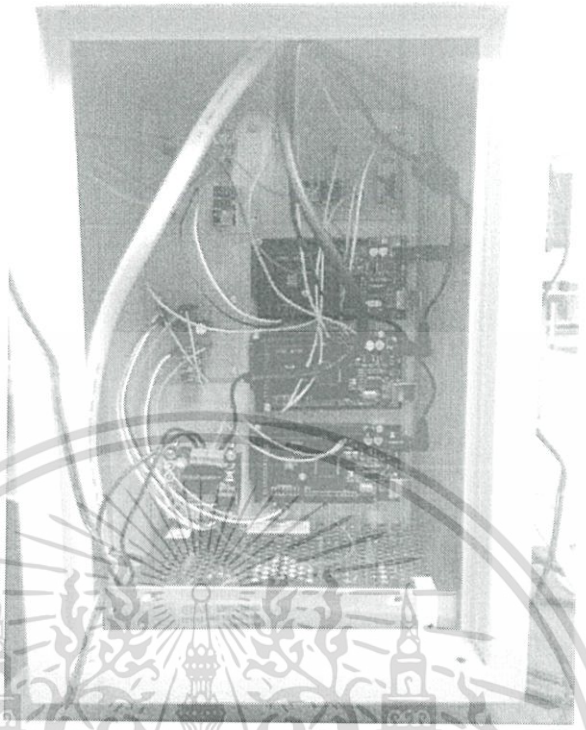


รูปที่ 4.10 ตำแหน่งติดตั้ง Level Sensor

4.2.4 การติดตั้งบอร์ดวงจรเข้ากับตัวเครื่อง



รูปที่ 4.11 บอร์ดวงจรที่ต่อกับเครื่อง PLC



รูปที่ 4.12 บอร์ดวงจรที่ติดตั้งกับฐานอลูมิเนียมโปรไฟล์

รูปที่ 4.13 ภาพรวมของระบบ

4.3 ทดสอบการใช้งาน Pressure Sensor

จากการทดลองเพื่อดูการทำงานของเซนเซอร์นั้น จะดูจากค่า 2 จุด เพื่อเปรียบเทียบกันคือ ค่าที่อ่านได้จากมิเตอร์ และค่าจากโปรแกรม Arduino ซึ่งพบว่าค่าแรงดันที่ได้ออกมาจากมิเตอร์ และจากโปรแกรม Arduino ค่ามีการเปลี่ยนแปลงเพิ่มขึ้น ลดลงไปตามช่วงระยะต่างๆ จากการอัดแรงดันผ่านหลอดฉีดยา โดยค่าที่สามารถอ่านได้อยู่ในช่วงตั้งแต่ 1 – 5V ซึ่งเป็นค่าเอาต์พุตของเซนเซอร์ตามข้อมูลจาก Data Sheet จึงทำให้สรุปได้ว่า Pressure Sensor สามารถทำงานได้ตามปกติ

4.4 ทดลองหาค่าความคลาดเคลื่อนระหว่างค่าจากมิเตอร์ และจากโปรแกรม Arduino

จากการทดลองเพื่อหาค่าความคลาดเคลื่อนนั้นจะได้ค่า ดังตารางที่ 4.1

ตารางที่ 4.1 ผลการทดลองหาค่าความคลาดเคลื่อนที่ปริมาตรน้ำ 0.0 ลิตร

ครั้งที่	แรงดันที่ มิเตอร์ (โวลต์)	ค่าผิดพลาด ที่ได้จากมิเตอร์ (ค่าเฉลี่ย–ค่าจริง)	แรงดันที่ Arduino (โวลต์)	ค่าผิดพลาด ที่ได้จากโปรแกรม (ค่าเฉลี่ย–ค่าจริง)	เปอร์เซ็นต์ ระดับน้ำ ที่ถัง (%)
1	1.135	0.1495	1.146	0	0
2	1.150	0.1345	1.146	0	0
3	1.132	0.1525	1.146	0	0
4	1.164	0.1205	1.146	0	0
5	1.260	0.0245	1.146	0	0
6	1.780	0.4955	1.146	0	0
7	1.234	0.0505	1.146	0	0
8	1.279	0.0055	1.146	0	0
9	1.361	0.0765	1.146	0	0
10	1.350	0.0655	1.146	0	0
เฉลี่ย	1.2845	0.1275	1.146	0	0

$$\text{ค่าเฉลี่ย} = \frac{\sum \text{ค่าจริง}}{\text{จำนวนครั้ง}} \quad (4.1)$$

$$\text{ค่าความคลาดเคลื่อน} = \frac{\sum(\text{ค่าเฉลี่ย} - \text{ค่าจริง})}{\text{จำนวนครั้งที่ทดลอง}} \quad (4.2)$$

$$\text{ค่าความคลาดเคลื่อนคิดเป็นเปอร์เซ็นต์} = \frac{\text{ค่าผิดพลาดเฉลี่ย}}{\text{แรงดันเฉลี่ย}} \text{ X } 100\% \quad (4.3)$$

ค่าความคลาดเคลื่อนที่มิเตอร์ = 9.926 %

ค่าความคลาดเคลื่อนที่โปรแกรม Arduino = 0 %

ตารางที่ 4.2 ผลการทดลองหาค่าความคลาดเคลื่อนที่ปริมาตรน้ำ 0.5 ลิตร

ครั้งที่	แรงดันที่ มิเตอร์ (โวลต์)	ค่าผิดพลาด ที่ได้จากมิเตอร์ (ค่าเฉลี่ย–ค่าจริง)	แรงดันที่ Arduino (โวลต์)	ค่าผิดพลาด ที่ได้จากโปรแกรม (ค่าเฉลี่ย–ค่าจริง)	เปอร์เซ็นต์ ระดับน้ำ ที่ถัง (%)
1	1.599	0.1416	1.647	0.0208	13
2	1.622	0.1186	1.685	0.0172	14
3	1.601	0.1396	1.627	0..0408	13
4	1.664	0.0766	1.685	0.0172	14
5	1.771	0.0304	1.647	0.0208	13
6	1.819	0.0784	1.647	0.0208	13
7	1.819	0.0284	1.685	0.0172	14
8	1.867	0.1264	1.685	0.0172	14
9	1.844	0.1034	1.685	0.0172	14
10	1.850	0.1094	1.685	0.0172	14
เฉลี่ย	1.2845	0.0953	1.146	0.0206	13.6

ค่าความคลาดเคลื่อนที่มิเตอร์ = 5.475 %

ค่าความคลาดเคลื่อนที่โปรแกรม Arduino = 1.186 %

ตารางที่ 4.3 ผลการทดลองหาค่าความคลาดเคลื่อนที่ปริมาตรน้ำ 1.0 ลิตร

ครั้งที่	แรงดันที่มิเตอร์ (โวลต์)	ค่าผิดพลาดที่ได้จากมิเตอร์ (ค่าเฉลี่ย–ค่าจริง)	แรงดันที่ Arduino (โวลต์)	ค่าผิดพลาดที่ได้จากโปรแกรม (ค่าเฉลี่ย–ค่าจริง)	เปอร์เซ็นต์ระดับน้ำที่ถัง (%)
1	2.086	0.1196	2.225	0.0044	28
2	2.072	0.1336	2.186	0.1295	27
3	2.097	0.1086	2.225	0.0044	28
4	2.260	0.0544	2.225	0.0006	28
5	2.289	0.0834	2.225	0.0044	28
6	2.333	0.1274	2.225	0.0044	28
7	2.330	0.1244	2.225	0.0044	28
8	2.290	0.0844	2.225	0.0044	28
9	2.379	0.1734	2.225	0.0044	28
10	1.920	0.2856	2.225	0.0044	28
เฉลี่ย	2.2056	0.1297	2.2206	0.0185	27.9

ค่าความคลาดเคลื่อนที่มิเตอร์ = 5.884 %

ค่าความคลาดเคลื่อนที่โปรแกรม Arduino = 0.834 %

ตารางที่ 4.4 ผลการทดลองหาค่าความคลาดเคลื่อนที่ปริมาตรน้ำ 1.5 ลิตร

ครั้งที่	แรงดันที่มิเตอร์ (โวลต์)	ค่าผิดพลาดที่ได้จากมิเตอร์ (ค่าเฉลี่ย–ค่าจริง)	แรงดันที่ Arduino (โวลต์)	ค่าผิดพลาดที่ได้จากโปรแกรม (ค่าเฉลี่ย–ค่าจริง)	เปอร์เซ็นต์ระดับน้ำที่ถัง (%)
1	2.542	0.0495	2.726	0.0158	41
2	2.562	0.0295	2.726	0.0538	42
3	2.564	0.0275	2.726	0.0538	42
4	2.666	0.0745	2.726	0.0538	42
5	2.275	0.3165	2.726	0.0538	42
6	2.794	0.2025	2.726	0.0538	42
7	2.700	0.1085	2.726	0.0538	42
8	2.577	0.0145	2.803	0.0928	42
9	2.813	0.2215	2.726	0.0538	42
10	2.422	0.1695	2.225	0.4852	42
เฉลี่ย	2.5915	0.1214	2.7102	0.0970	41.9

ค่าความคลาดเคลื่อนที่มิเตอร์ = 4.684 %

ค่าความคลาดเคลื่อนที่โปรแกรม Arduino = 3.579 %

ตารางที่ 4.5 ผลการทดลองหาค่าความคลาดเคลื่อนที่ปริมาตรน้ำ 2.0 ลิตร

ครั้งที่	แรงดันที่มิเตอร์ (โวลต์)	ค่าผิดพลาดที่ได้จากมิเตอร์ (ค่าเฉลี่ย–ค่าจริง)	แรงดันที่ Arduino (โวลต์)	ค่าผิดพลาดที่ได้จากโปรแกรม (ค่าเฉลี่ย–ค่าจริง)	เปอร์เซ็นต์ระดับน้ำที่ถัง (%)
1	3.280	0.1134	3.265	0.0036	55
2	3.046	0.1206	3.304	0.0426	56
3	3.030	0.1366	3.304	0.0426	56
4	3.213	0.0464	3.304	0.0426	56
5	3.090	0.0766	3.304	0.0426	56
6	3.234	0.0674	3.304	0.0426	56
7	3.342	0.1754	3.381	0.1196	57
8	3.268	0.1014	3.342	0.0806	57
9	3.340	0.1734	3.342	00806	57
10	2.823	0.3436	2.764	0.4974	57
เฉลี่ย	3.166	0.0135	3.2614	0.0994	56.3

ค่าความคลาดเคลื่อนที่มิเตอร์ = 4.278 %

ค่าความคลาดเคลื่อนที่โปรแกรม Arduino = 3.050 %

ตารางที่ 4.6 ผลการทดลองหาค่าความคลาดเคลื่อนที่ปริมาตรน้ำ 2.5 ลิตร

ครั้งที่	แรงดันที่มิเตอร์ (โวลต์)	ค่าผิดพลาดที่ได้จากมิเตอร์ (ค่าเฉลี่ย–ค่าจริง)	แรงดันที่ Arduino (โวลต์)	ค่าผิดพลาดที่ได้จากโปรแกรม (ค่าเฉลี่ย–ค่าจริง)	เปอร์เซ็นต์ระดับน้ำที่ถัง (%)
1	3.516	0.1216	3.843	0.0267	70
2	3.523	0.1146	3.843	0.0267	71
3	3.515	0.1226	3.882	0.0657	70
4	3.734	0.0964	3.882	0.0657	71
5	3.738	0.1004	3.882	0.0657	71
6	3.697	0.0594	3.843	0.0267	70
7	3.732	0.0944	3.882	0.0657	71
8	3.770	0.1324	3.882	0.0657	71
9	3.799	0.1614	3.882	0.0657	71
10	3.352	0.2856	3.324	0.4743	71
เฉลี่ย	3.637	0.1288	3.816	0.0948	70

ค่าความคลาดเคลื่อนที่มิเตอร์ = 3.542 %

ค่าความคลาดเคลื่อนที่โปรแกรม Arduino = 2.485 %

ตารางที่ 4.7 ผลการทดลองหาค่าความคลาดเคลื่อนที่ปริมาตรน้ำ 3.0 ลิตร

ครั้งที่	แรงดันที่มิเตอร์ (โวลต์)	ค่าผิดพลาดที่ได้จากมิเตอร์ (ค่าเฉลี่ย–ค่าจริง)	แรงดันที่ Arduino (โวลต์)	ค่าผิดพลาดที่ได้จากโปรแกรม (ค่าเฉลี่ย–ค่าจริง)	เปอร์เซ็นต์ระดับน้ำที่ถัง (%)
1	4.003	0.1528	4.383	0.0197	84
2	4.001	0.1548	4.421	0.0577	85
3	4.051	0.1048	4.421	0.0577	85
4	4.421	0.2652	4.421	0.0577	85
5	4.421	0.2652	4.421	0.0577	85
6	4.146	0.0098	4.421	0.0577	85
7	4.223	0.0672	4.421	0.0577	85
8	4.228	0.0722	4.421	0.0577	85
9	4.269	0.1132	4.421	0.0577	85
10	3.795	0.3608	3.882	0.4813	85
เฉลี่ย	1.2845	0.1566	4.3633	0.0962	84.9

ค่าความคลาดเคลื่อนที่มิเตอร์ = 3.768 %

ค่าความคลาดเคลื่อนที่โปรแกรม Arduino = 2.206 %

ตารางที่ 4.8 ผลการทดลองหาค่าความคลาดเคลื่อนที่ปริมาตรน้ำ 3.5 ลิตร

ครั้งที่	แรงดันที่มิเตอร์ (โวลต์)	ค่าผิดพลาดที่ได้จากมิเตอร์ (ค่าเฉลี่ย–ค่าจริง)	แรงดันที่ Arduino (โวลต์)	ค่าผิดพลาดที่ได้จากโปรแกรม (ค่าเฉลี่ย–ค่าจริง)	เปอร์เซ็นต์ระดับน้ำที่ถัง (%)
1	4.495	0.0808	5.000	0.0117	100
2	4.483	0.0928	4.961	0.0273	99
3	4.500	0.0758	5.000	0.0117	100
4	4.532	0.0438	4.961	0.0273	99
5	4.640	0.0642	4.961	0.0273	99
6	4.487	0.0888	5.000	0.0117	100
7	4.516	0.0598	5.000	0.0117	100
8	4.688	0.1122	5.000	0.0117	100
9	4.715	0.1392	5.000	0.0117	100
10	4.702	0.1262	5.000	0.0117	100
เฉลี่ย	4.576	0.0883	4.988	0.0164	99.7

ค่าความคลาดเคลื่อนที่มิเตอร์ = 1.931 %

ค่าความคลาดเคลื่อนที่โปรแกรม Arduino = 0.328 %

ตารางที่ 4.9 สรุปค่าแรงดันเฉลี่ย และความคลาดเคลื่อนที่ได้

ปริมาตรน้ำ (L)	แรงดันที่มิเตอร์ (V)		แรงดันที่ Arduino (V)		เปอร์เซ็นต์ระดับน้ำที่ถังเฉลี่ย(%)
	ค่าเฉลี่ย	ค่าผิดพลาดเฉลี่ย	ค่าเฉลี่ย	ค่าผิดพลาดเฉลี่ย	
0.0	1.2845	0.1275	1.1460	0.0000	0.0
0.5	1.7406	0.0953	1.7406	0.0206	13.6
1.0	2.2056	0.1295	2.2206	0.0185	27.9
1.5	2.5915	0.1214	2.7102	0.0970	41.9
2.0	3.1666	0.0135	3.2614	0.0995	56.3
2.5	3.6376	0.1289	3.8163	0.0949	70.7
3.0	4.1558	0.1566	4.3633	0.0963	84.9
3.5	4.5758	0.0883	4.9883	0.0164	99.7

จากตารางที่ 4.9 จะพบว่าที่ระดับน้ำสูงขึ้นค่าความคลาดเคลื่อนที่ได้จะมีค่าน้อยลง เนื่องจากความแรงของน้ำที่กระแทกลงมาจากด้านบนนั้นมีระยะทางที่น้อยลง ทำให้ความแรงของน้ำลดลงตามไปด้วย น้ำในถังจึงนิ่งมากยิ่งขึ้น

4.5 ทดสอบหาค่าเปอร์เซ็นต์ของระดับน้ำในถัง เปรียบเทียบกับค่าเปอร์เซ็นต์การทำงานของปั๊ม

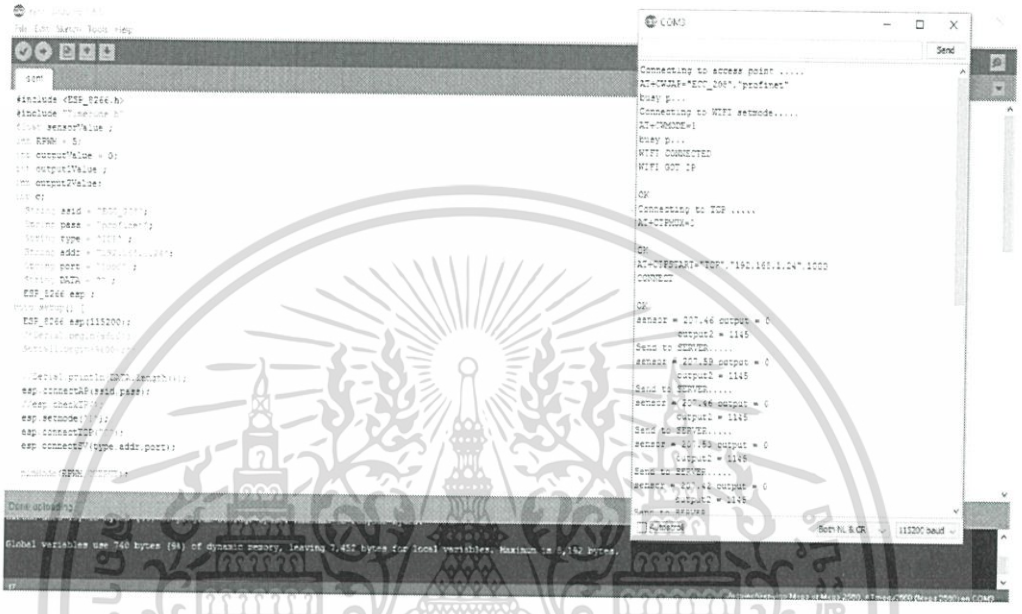
ตารางที่ 4.10 ค่าจาก Pressure Sensor เทียบกับเปอร์เซ็นต์การทำงานของปั๊ม

เปอร์เซ็นต์การทำงานของปั๊ม (%)	เปอร์เซ็นต์ของระดับน้ำในถัง (%)	เอาต์พุตจาก pressure Sensor (มิลลิโวลต์)
50	2	1222
55	3	1260
60	14	1684
65	26	2147
70	40	2687
75	57	3342
80	73	3959
85	86	4807
90	95	4460
95	98	4922
100	100	5000

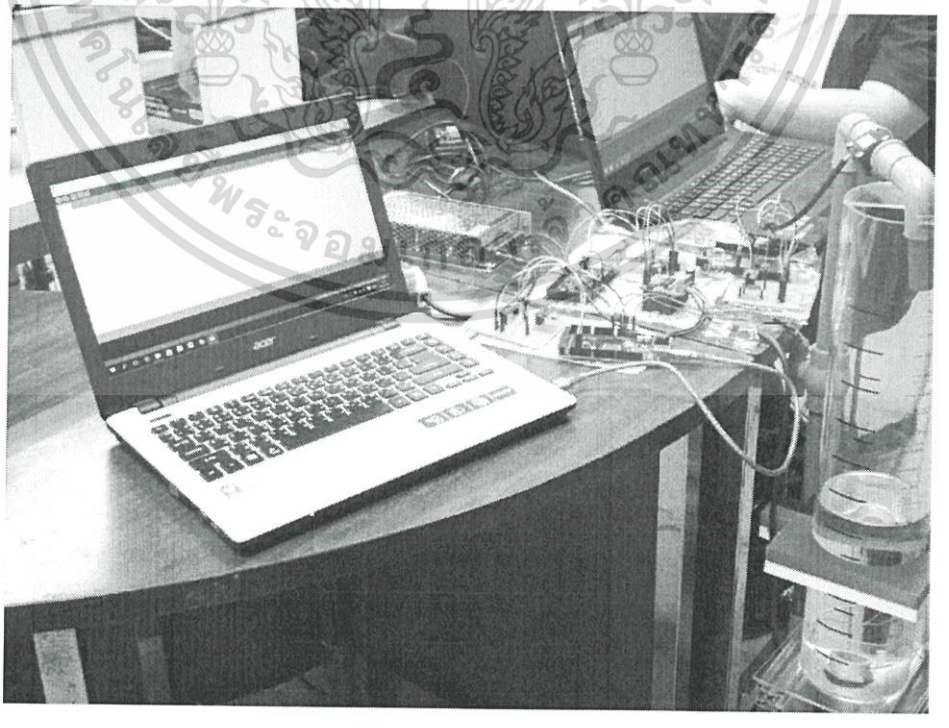
การทดลองนี้ทำเพื่อเปรียบเทียบดูค่าที่ระดับน้ำแต่ละช่วงเพื่อปรับ Speed Motor ในค่าที่สามารถควบคุมระดับน้ำภายในถังให้เป็นไปตามค่าที่ต้องการได้ จากตารางที่ 4.10 จะพบว่าจะต้องใช้ค่า Speed Motor ที่ 50% จึงจะสามารถปั๊มน้ำขึ้นใส่ถังได้ และที่ระดับน้ำสูงขึ้นค่า Speed Motor ก็จะต้องเพิ่มขึ้นเพื่อให้สามารถปั๊มน้ำขึ้นใส่ถังให้สูงเพิ่มขึ้น พบว่าที่ช่วงระดับน้ำใกล้ 100% นั้นจะมีค่าไม่ห่างกันมาก เนื่องจากระดับน้ำที่เพิ่มขึ้นสูงทำให้เกิดแรงกระแทกของน้ำน้อยกว่าที่ระดับน้ำที่ต่ำ ดังนั้นแรงดันน้ำจึงมีความนิ่งมากยิ่งขึ้น โดยที่เปิดท่อน้ำออกที่คงที่

4.6 การทดลองเขียนโปรแกรมส่งค่าให้กับโมดูล Wi-Fi (Level Transmitter)

จากการทดลองเขียนโปรแกรมเพื่อส่งค่าเปอร์เซ็นต์ของระดับน้ำในถังที่วัดได้พบว่า โปรแกรมสามารถส่งข้อมูลระหว่าง Wi-Fi สามารถส่งค่าถึงกันได้ โดยการเชื่อมต่อนั้นจะต้องใช้ระยะเวลาสักระยะหนึ่งเพื่อให้ Wi-Fi ได้ประมวลผลเพื่อเชื่อมต่อกับ Wi-Fi Server และเมื่อพบว่าขึ้น Connect แล้ว ก็จะส่งค่าเปอร์เซ็นต์ของระดับน้ำในถังที่อ่านได้ออกมา



รูปที่ 4.14 หน้าต่างโปรแกรม Arduino กับ Serial Monitor ส่วนส่ง



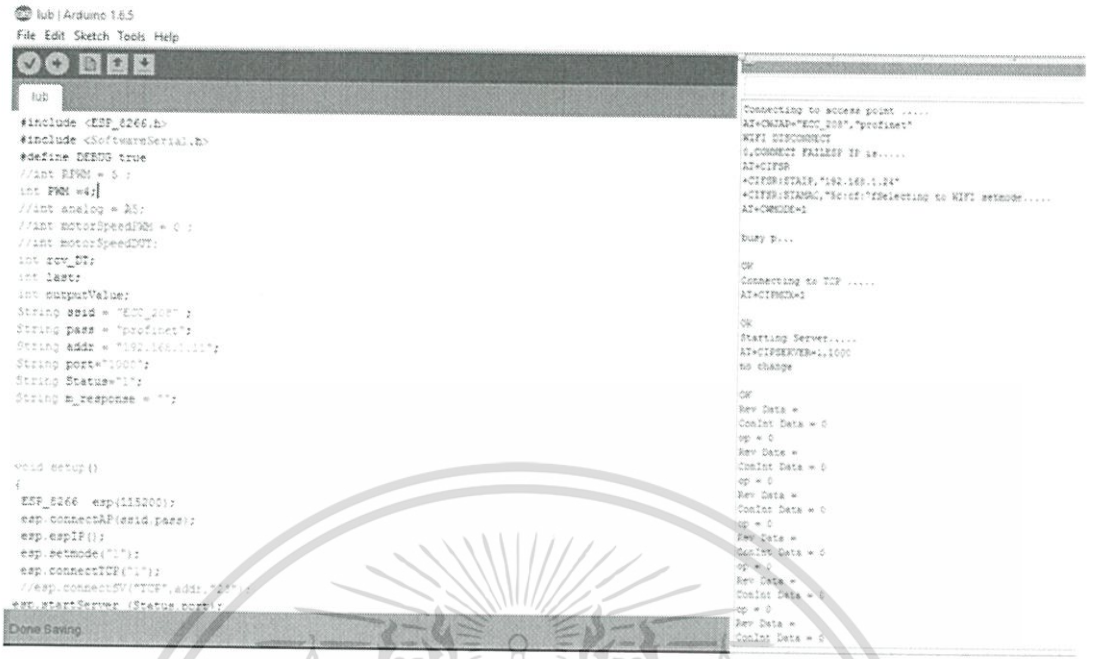
รูปที่ 4.15 การทดลองส่งค่าให้กับโมดูล Wi-Fi

4.7 การทดลองเขียนโปรแกรมรับค่าจากโมดูล Wi-Fi (Level Transmitter)

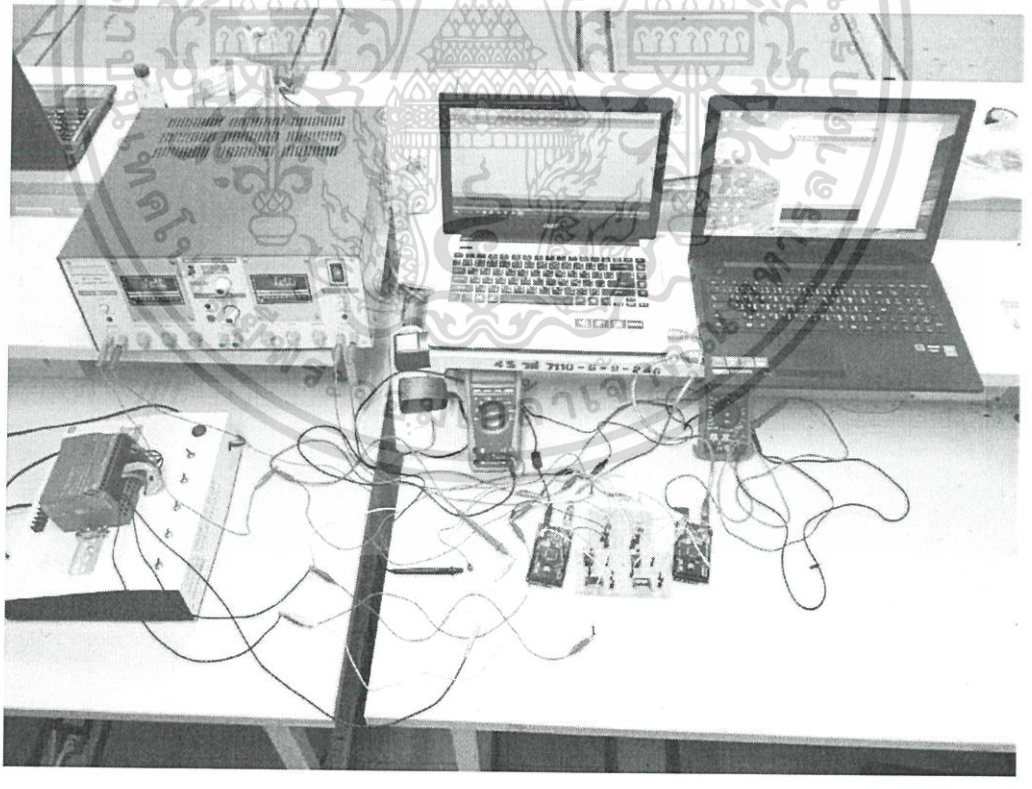
จากการทดลองเขียนโปรแกรมเพื่อรับค่าเปอร์เซ็นต์ของระดับน้ำในถัง พบว่าโปรแกรมสามารถรับข้อมูลได้ โดยที่ Wi-Fi Server จะมีช่วงระยะเวลาการเชื่อมต่อที่ช้ากว่า Wi-Fi Client อยู่ระยะหนึ่ง แต่เมื่อเชื่อมต่อแล้วค่าที่รับมาเป็นค่าที่ได้แบบ Real-time ซึ่งตรงกับค่าที่ส่งมาจากฝั่ง Wi-Fi Client และโปรแกรมสามารถส่งค่าเปอร์เซ็นต์ของระดับน้ำในถังที่รับมาให้กับ PLC ผ่านทาง PWM ได้ตามที่เขียนโปรแกรมไว้บนบอร์ด Arduino และเมื่อทำการวัดค่าแรงดันที่ได้ออกมาจาก PWM และ PLC พบว่าค่าที่ได้จาก PLC มีค่าประมาณสองเท่าของค่าจาก PWM ตามที่ได้ต่อวงจรขยายแรงดัน เพื่อเพิ่มแรงดันไว้

ตารางที่ 4.11 ผลการทดลองวัดค่าแรงดันที่ขา PWM และที่ PLC

เปอร์เซ็นต์ระดับน้ำที่ถัง (%)	ค่าแรงดันที่ Arduino (mV)	ค่าแรงดันที่ PWM (V)	ค่าแรงดันที่ PWM เมื่อคำนวณเป็นสองเท่า (V)	ค่าแรงดันที่ PLC (V)
0	1.15	1.15	2.30	2.35
5	1.30	1.29	2.58	2.73
10	1.48	1.48	2.96	3.16
15	1.63	1.62	2.34	3.31
20	1.81	1.80	3.60	3.70
25	1.98	1.91	3.94	3.98
30	2.14	2.14	4.28	4.34
35	2.23	2.23	4.46	4.51
40	2.31	2.31	4.60	4.65
45	2.42	2.42	4.84	4.92
50	2.50	2.45	4.90	5.02
55	2.75	2.75	5.50	5.58
60	2.86	2.85	5.70	6.06
65	3.23	3.20	6.40	6.44
70	3.45	3.44	6.88	7.09
75	3.75	3.75	7.50	7.57
80	4.05	4.00	8.00	8.04
85	4.35	4.30	8.60	8.48
90	4.50	4.49	8.98	9.06
95	4.70	4.68	9.36	9.52
100	5.00	4.98	9.96	9.99



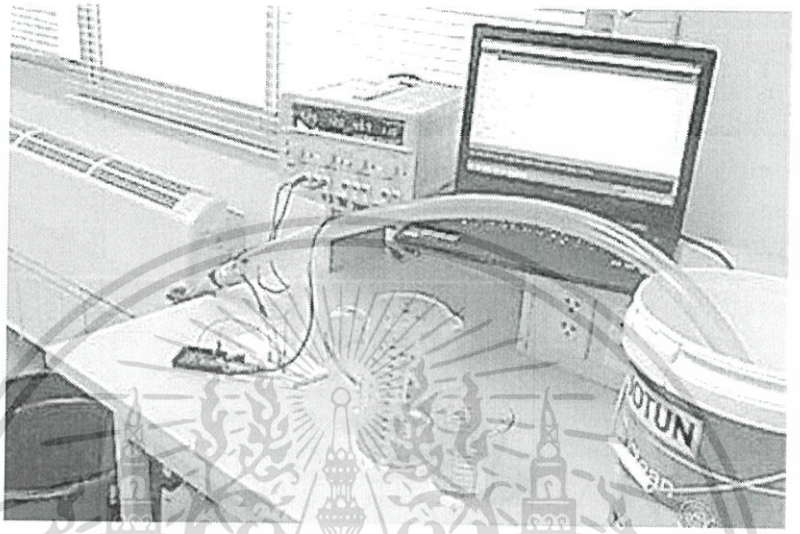
รูปที่ 4.16 หน้าต่างโปรแกรม Arduino กับ Serial Monitor ส่วนรับ



รูปที่ 4.17 การทดลองรับค่าจากโมดูล Wi-Fi

4.8 การทดลอง Flow Sensor เพื่อหาค่า Flow Rate ของระบบ

ทดลองหาค่าความคลาดเคลื่อนระหว่างค่าอัตราการไหลที่ได้จากการเขียนโปรแกรมเปรียบเทียบกับการวัดอัตราการไหลจริง



รูปที่ 4.18 การเซ็ตอุปกรณ์เพื่อทดลอง

ผลการทดลอง

ตารางที่ 4.12 ค่า Flow Rate จากการวัดและจากหน้า Serial Monitor โดยจับเวลา 6 วินาที

ครั้งที่	ปริมาณน้ำที่วัด ได้(L)	เวลา (s)	Flow Rate (วัด) L/min	ค่าผิดพลาดจากการวัด	Flow Rate (monitor) L/min	ค่าผิดพลาดจาก monitor
1	1.265	6	12.65	0.47	12-13	1.45
2	1.25	6	12.5	0.32	11	0.55
3	1.275	6	12.75	0.57	12	0.45
4	1.1	6	11	0.18	11	0.55
5	1.14	6	11.4	0.78	11	0.55
6	1.195	6	11.95	0.23	11-12	0.55
7	1.2	6	12	0.18	11-12	0.45
8	1.275	6	12.75	0.57	11-12	0.05
9	1.13	6	11.3	0.88	11-12	0.55
10	1.15	6	13.5	0.32	12-13	0.45
เฉลี่ย			12.18	0.6	11.55	0.56

ตารางที่ 4.13 ค่า Flow Rate จากการวัดและจากหน้า Serial Monitor โดยจับเวลา 1 นาที

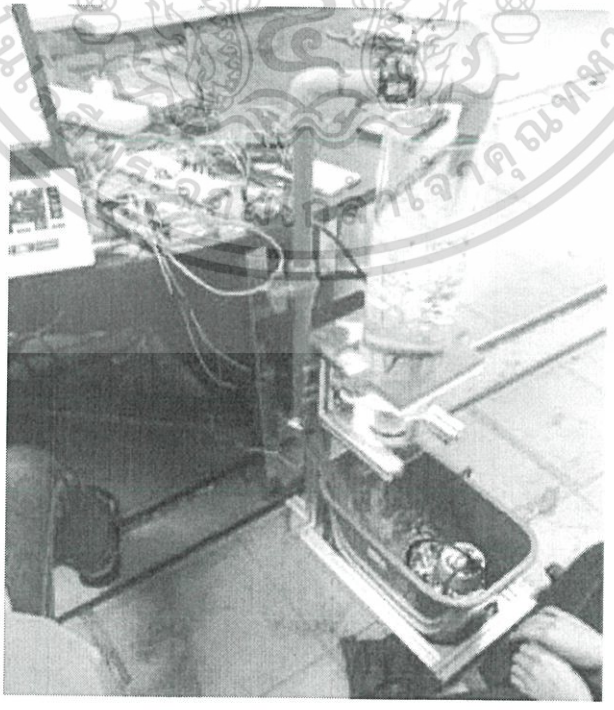
ครั้งที่	ปริมาณน้ำที่วัด ได้(L)	เวลา (s)	Flow Rate (วัด) L/min	ค่าผิดพลาดจากการวัด	Flow Rate (monitor) L/min	ค่าผิดพลาดจาก monitor
1	11.445	1	11.445	0.069	11-12	0
2	11.445	1	11.445	0.079	12	0
3	11.54	1	11.54	0.164	12-13	0
4	11.355	1	11.355	0.021	12	0
5	11.2	1	11.2	0.176	12	0
6	11.51	1	11.51	0.134	12	0
7	11.25	1	11.25	0.126	12	0
8	11.285	1	11.285	0.091	12	0
9	11.3	1	11.3	0.076	12	0
10	11.42	1	11.42	0.044	12	0
เฉลี่ย			11.346	0.098	12	0

จากการทดลองหาค่า Flow Rate ด้วย Flow Sensor และโปรแกรม Arduino พบว่าค่าที่ได้ในแต่ละครั้งมีความใกล้เคียงกัน และค่าผิดพลาดที่ได้มีค่าน้อย (ค่าผิดพลาดส่วนใหญ่ที่เกิดขึ้น เกิดจากความผิดพลาดในการวัดปริมาณน้ำและการจับเวลา) แต่อย่างไรก็ตาม เมื่อส่งค่า Flow Rate ไปให้ PLC จะต้องส่งเป็นค่าเปอร์เซ็นต์ Flow Rate และบอกช่วงค่าผิดพลาดเพื่อความสะดวกในการเขียนโปรแกรมควบคุมต่อไป

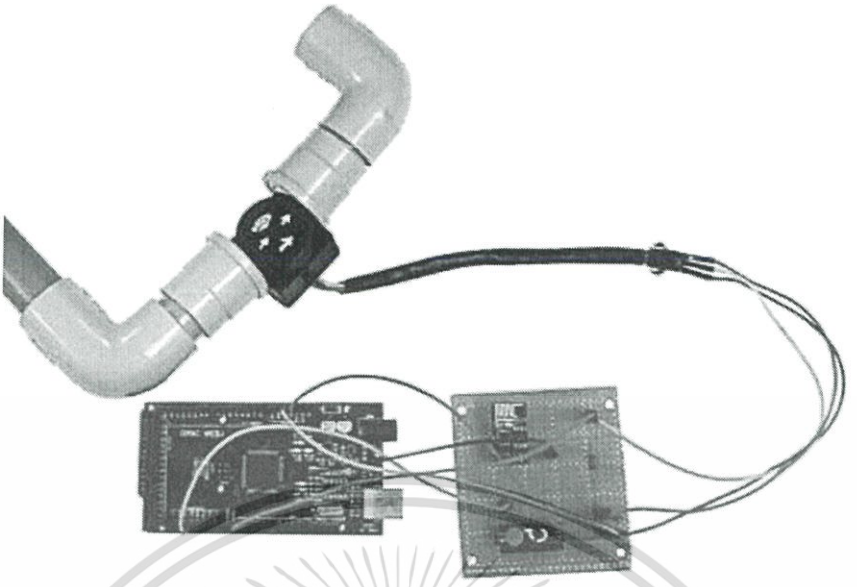
4.9 การทดลองหาค่าแรงดันเอาต์พุตของ Flow Sensor เทียบกับเปอร์เซ็นต์การทำงานของปั๊มน้ำ

ทำการทดลองต่อ Flow Sensor เข้ากับระบบจริงโดยให้ผลลัพธ์ออกมาเป็นค่า Flow Rate (L/min), ค่าเปอร์เซ็นต์ Flow Rate และค่าแรงดันเอาต์พุตจาก Flow Sensor ในหน่วย mV โดยให้น้ำไหลผ่าน Flow Sensor ด้วยความเร็วที่แตกต่างกัน เพื่อเทียบกับเปอร์เซ็นต์การทำงานของปั๊มน้ำ

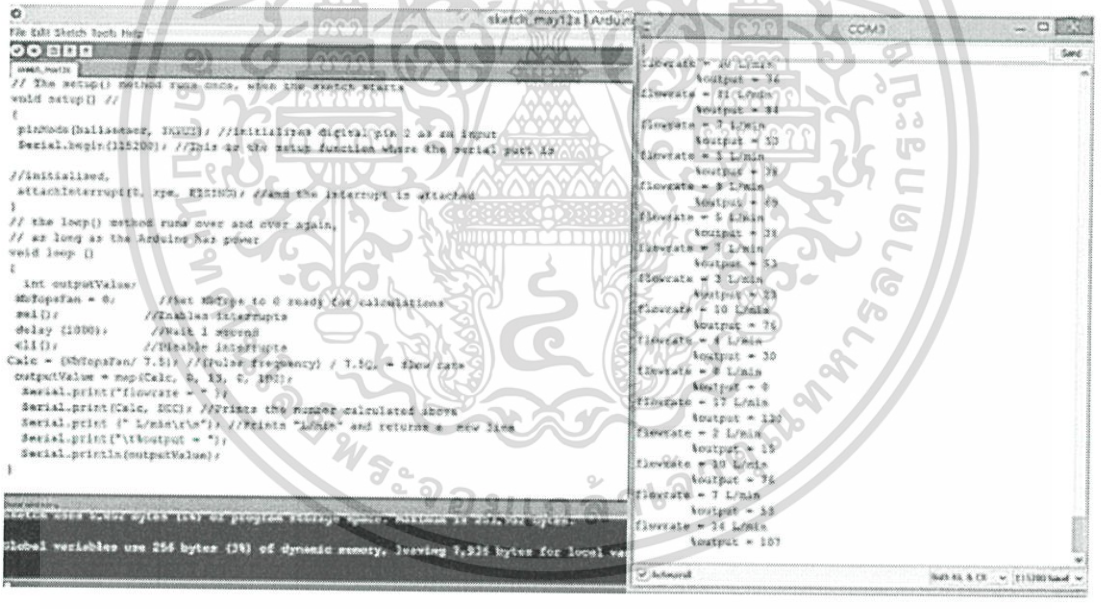
รูปที่ 4.19 ระบบรวมที่ใช้ในการทดลอง



รูปที่ 4.20 ส่วนของระบบถังน้ำ



รูปที่ 4.21 ส่วนของการต่อวงจร



รูปที่ 4.22 หน้าจอโปรแกรม Arduino และผลที่แสดงผ่านหน้า Serial Monitor

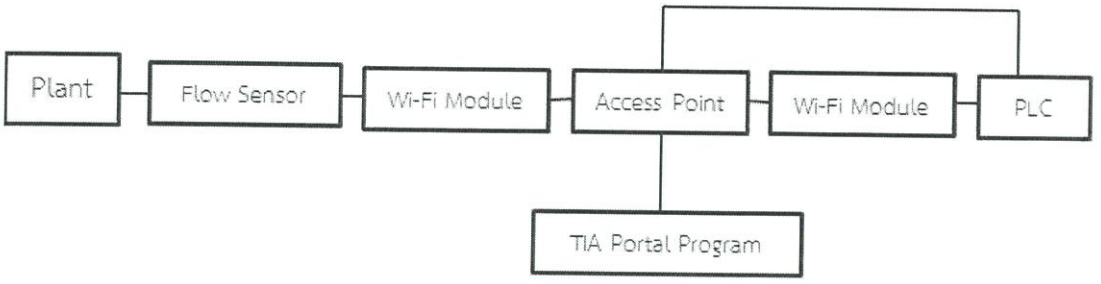
ผลการทดลองหาค่าแรงดันเอาต์พุตของ Flow Sensor เทียบกับเปอร์เซ็นต์การทำงานของปั๊มน้ำ

ตารางที่ 4.14 ผลการทดลองหาค่าแรงดันเอาต์พุตของ Flow Sensor เทียบกับเปอร์เซ็นต์การทำงานของปั๊มน้ำ

เปอร์เซ็นต์การทำงานของ Motor ปั๊มน้ำ (%)	ค่า Flow Rate (L/min)	ค่าแรงดันเอาต์พุตของ Flow Sensor (mV)
45	3	900
50	5	1550
55	6	1850
60	8	2500
65	9	2800
70	10	3100
75	11	3400
80	12	3750
85	13	4050
90	14	4350
95	15	4650
100	16	5000

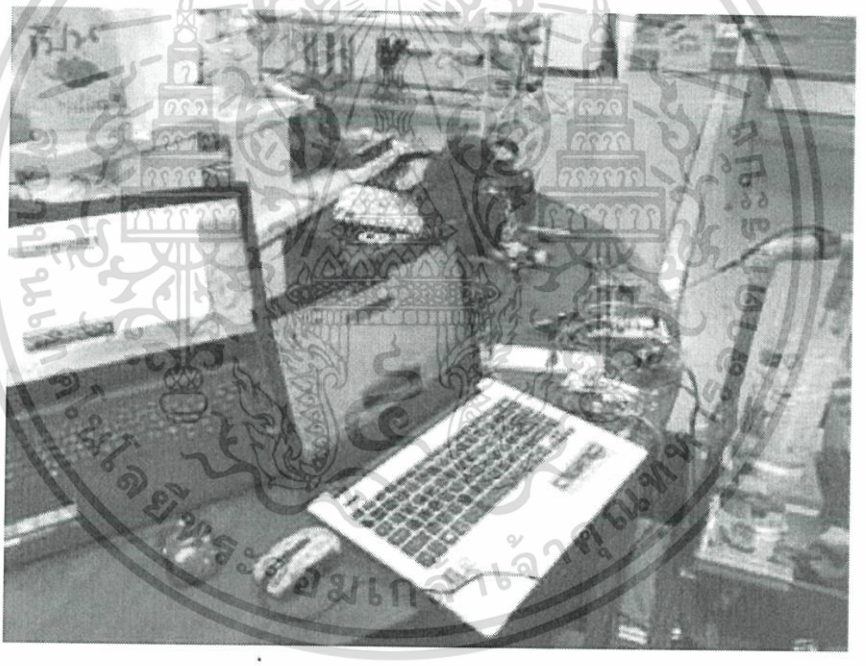
4.10 ผลการทดลอง ส่ง-รับสัญญาณ (ค่าแรงดัน) เข้า Module Wi-Fi แล้วส่งเข้า PLC และโปรแกรม TIA PORTAL

ทำการทดลองส่งค่าแรงดันที่ได้จาก Flow Sensor โดยการส่งค่าแรงดันนั้นจะต้องส่งผ่าน Module Wi-Fi 2 ตัว ซึ่งตัวแรกจะต่อกับบอร์ด Arduino ที่ติดกับ Flow Sensor ส่วนอีกตัวจะต่อกับบอร์ด Arduino ที่ติดกับ PLC และเป็นการส่ง-รับค่าแบบเรียลไทม์ หลังจากนั้นบอร์ด Arduino ที่ติดกับ PLC จึงจะทำการส่งค่าแรงดันที่ได้รับไปที่ PLC โดยผ่านขา PWM ของบอร์ด Arduino เข้าขา Analog in ของ PLC และสุดท้ายโปรแกรม TIA PORTAL จะนำค่าแรงดันที่ PLC ไปแสดงผลและใช้งานต่อไป ดังรูปที่ 4.23

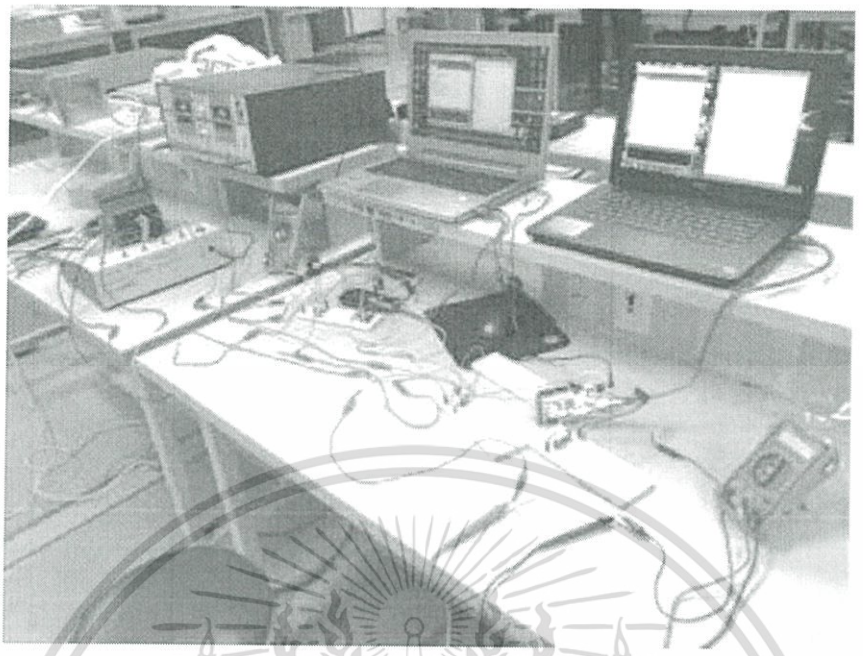


รูปที่ 4.23 การส่งสัญญาณจาก Flow Sensor ไป PLC

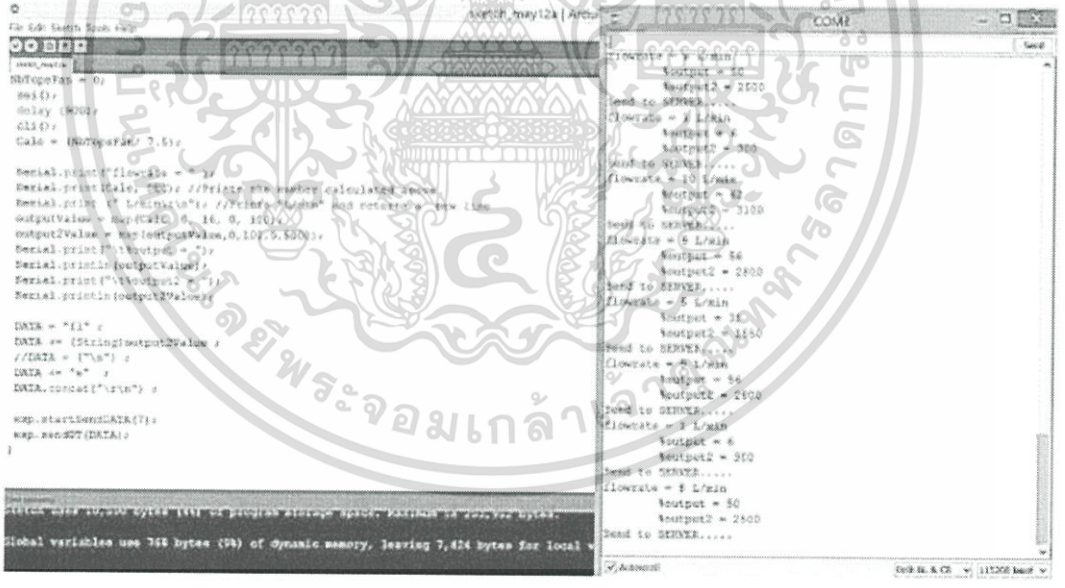
ผลการทดลอง ส่ง-รับ สัญญาณระหว่าง Module Wi-Fi กับ Module Wi-Fi แล้วส่งเข้าPLC และโปรแกรม TIA PORTAL



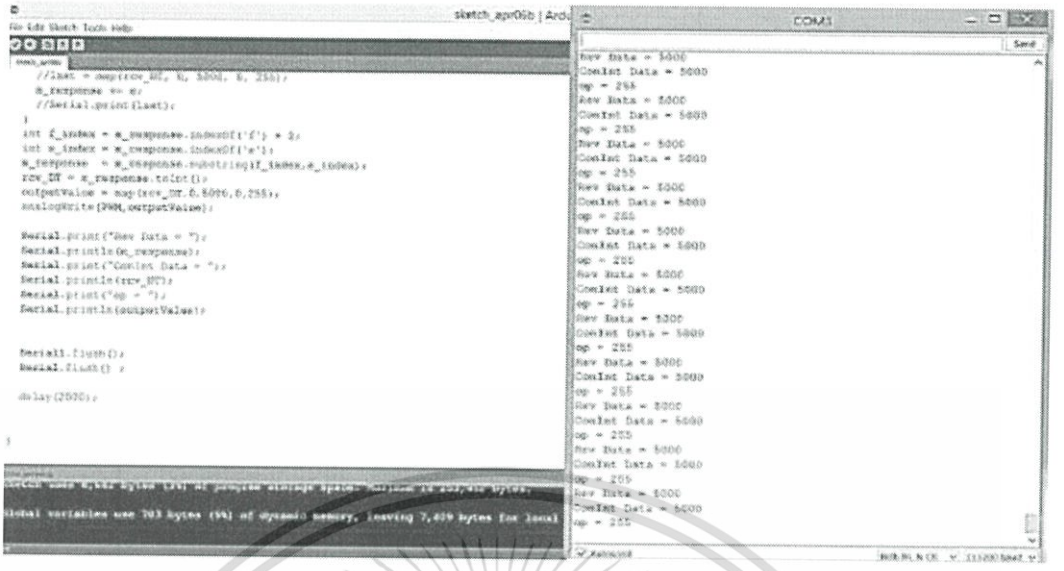
รูปที่ 4.24 ระบบด้านตัวส่งสัญญาณ



รูปที่ 4.25 ระบบด้านตัวรับสัญญาณ



รูปที่ 4.26 หน้าจอโปรแกรม Arduino และผลที่แสดงผ่านหน้า Serial Monitor (ด้านตัวส่งสัญญาณ)

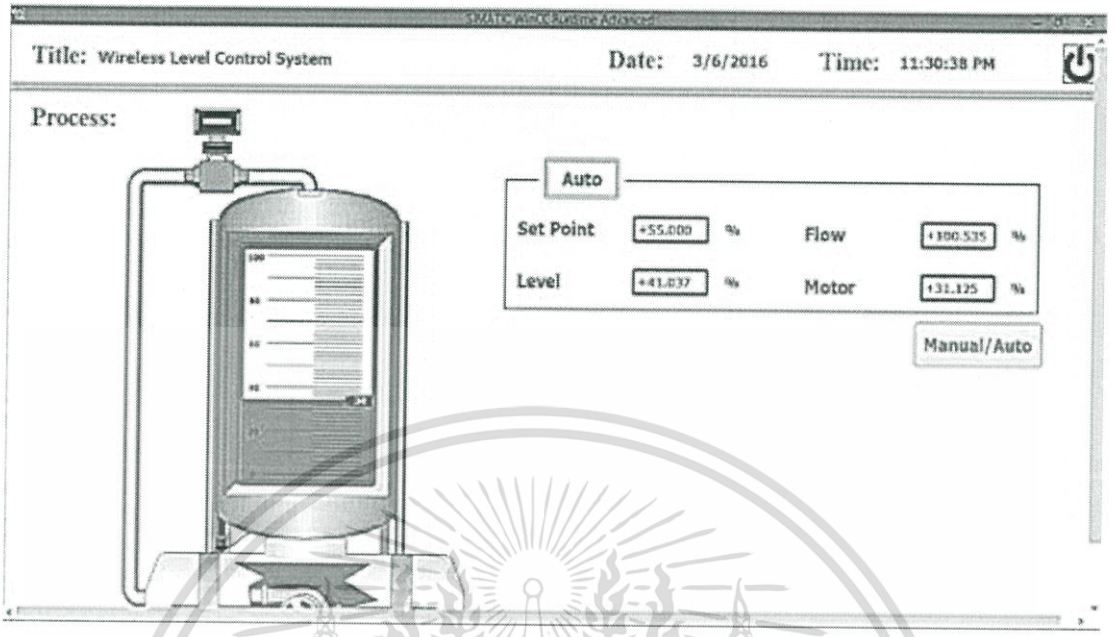


รูปที่ 4.27 หน้าจอโปรแกรม Arduino และผลที่แสดงผ่านหน้า Serial Monitor (ด้านตัวรับสัญญาณ)

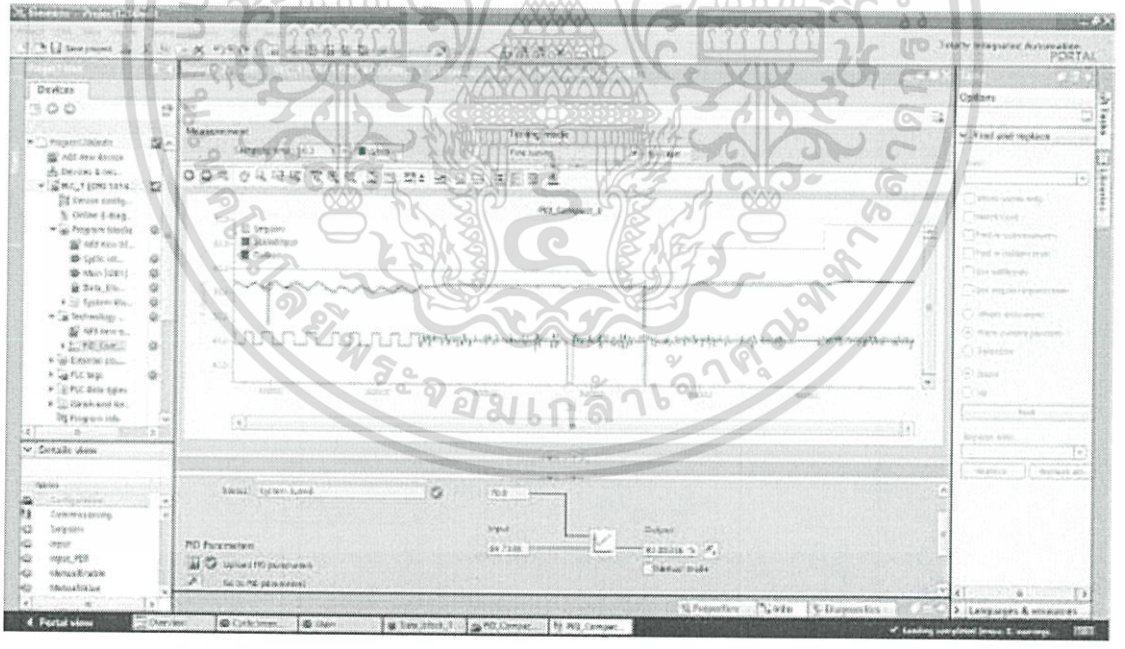
ผลการทดลองหาค่า Volt Output ที่ Module Wi-Fi ฝั่งรับ, ค่า Volt ที่ขา PWM ของบอร์ด Arduino และค่าแรงดันที่เข้า PLC เทียบกับเปอร์เซ็นต์การทำงานของปั๊มน้ำ

ตารางที่ 4.15 ผลการทดลองหาค่าแรงดันที่ Module Wi-Fi ด้านตัวรับ, ค่าแรงดันที่ขา PWM ของบอร์ด Arduino และค่าแรงดันที่เข้า PLC เทียบกับเปอร์เซ็นต์การทำงานของปั๊มน้ำ

เปอร์เซ็นต์การทำงานของ Motor ปั๊มน้ำ (%)	ค่าแรงดันของ Flow Sensor (mV)	ค่าแรงดันที่ขา PWM (V)	ค่าแรงดันที่เข้า PLC (V)
45	1250	1.24	2.63
50	1850	1.85	3.836
55	2150	2.15	4.419
60	2500	2.50	5.120
65	2800	2.79	5.702
70	3100	3.11	6.320
75	3400	3.40	6.910
80	3750	3.76	7.600
85	4050	4.05	8.160
90	4350	4.35	8.700
95	4650	4.66	9.290
100	5000	5.02	10.00



รูปที่ 4.28 หน้าจอ HMI ของ PLC



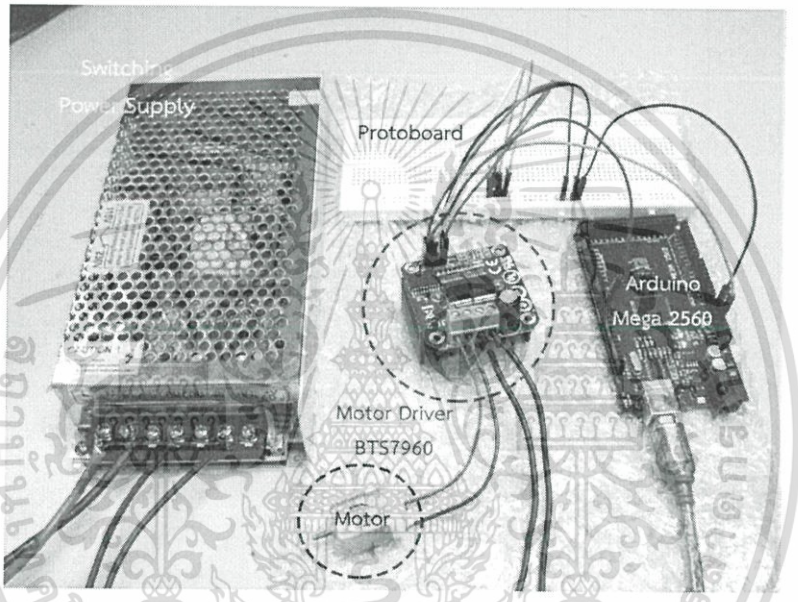
รูปที่ 4.29 หน้าจอโปรแกรม TIA Protal

จากการทดลองพบว่าค่าแรงดันที่ได้จาก Flow Sensor (แสดงที่หน้าจอ Serial Monitor ของ Arduino ด้าน Client หรือด้านส่งสัญญาณ) ซึ่งมีค่าเท่ากันกับค่าแรงดันที่อ่านได้จากหน้าจอ Serial

Monitor ของ Arduino ด้าน Server หรือด้านรับสัญญาณ, หน้า HMI ของ PLC และที่หน้าจอโปรแกรม TIA Protal ซึ่งเป็นไปตาม Code ที่ได้เขียนไว้ในโปรแกรม Arduino

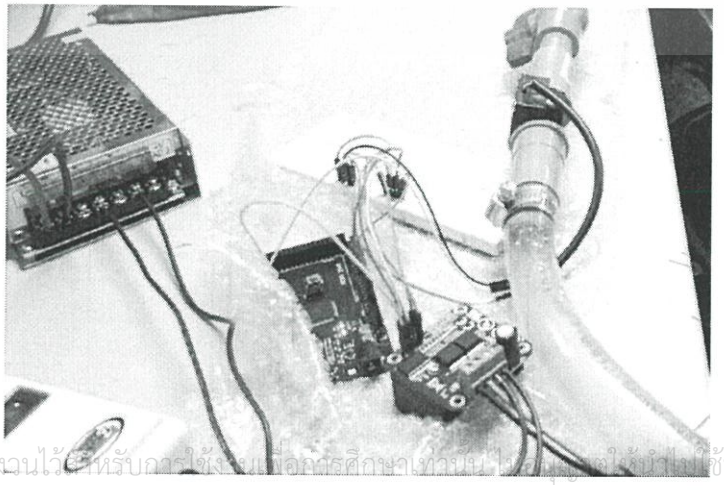
4.11 ผลการทดสอบชุดขับมอเตอร์ BTS7960 43A

จากการทดสอบเขียนโปรแกรมควบคุมชุดขับมอเตอร์ BTS7960 เพื่อไปขับมอเตอร์ พบว่ามอเตอร์สามารถทำงานได้ตามที่โปรแกรมได้เขียนควบคุมไว้ได้อย่างมีประสิทธิภาพ โดยมอเตอร์หมุนไปทางขวา และทางซ้ายด้วยความเร็วเพิ่มขึ้นจาก 0-255 PWM สลับกัน และไม่พบปัญหาใดๆ ระหว่างการทดลอง



รูปที่ 4.30 การต่อวงจรชุดขับมอเตอร์ BTS7960 กับ Arduino Mega 2560 R3

4.12 ผลการทดลองควบคุมความเร็วของมอเตอร์ปั๊มน้ำ โดยใช้ Duty Cycle 0-100% พร้อมทั้งวัดค่าอัตราการไหลของน้ำ



รูปที่ 4.31 การต่อวงจรชุดขับมอเตอร์ BTS7960 กับ Arduino Mega 2560 R3 และ Flow Sensor ก่อนต่อเข้ากับระบบจริง

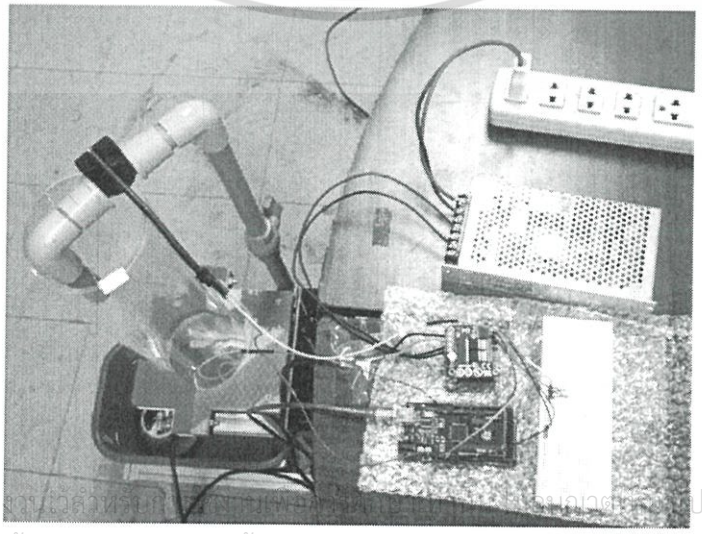
ผลการทดลองครั้งที่ 1

ตารางที่ 4.16 ค่าอัตราการไหลของน้ำที่ Duty Cycle 0-100%

Duty Cycle (%)	Flow Rate (L/min)
10	-
20	-
30	-
40	1
50	4.89
60	7.86
70	9.90
80	11.90
90	13.90
100	15.90

จากการทดลองจะเห็นได้ว่า เมื่อเพิ่ม Duty Cycle จาก 0 - 100% ซึ่งจะทำการเพิ่มขึ้นทีละ 10% และพบว่าในช่วง 0 - 39% มอเตอร์สามารถทำงานได้ แต่ไม่สามารถปั๊มน้ำให้ผ่าน Flow Sensor ได้ อาจจะเกิดจากความต่างของระดับความสูงจากถังน้ำถึง Flow Sensor ที่สูงเกินไป

มอเตอร์สามารถทำงานจนปั๊มน้ำไหลผ่าน Flow Sensor โดยเริ่มต้นที่ค่า Duty Cycle 40% ขึ้นไป และในช่วงแรกที่ปั๊มจะเกิดการหยุดชะงักชั่วครู่ แล้วสามารถทำงานต่อจนครบ 1 นาที โดยโปรแกรมที่สั่งคำสั่งไว้ใน Arduino จะคำนวณ Flow Rate ให้ออกมาโดยอัตโนมัติ ดังที่กล่าวไว้ในตารางที่ 4.1 จากนั้นจึงได้ทำการทดลองโดยการต่อกับระบบจริง



รูปที่ 4.32 การต่อวงจรชุดขับมอเตอร์ BTS7960 กับ Arduino Mega 2560 R3 และ Flow Sensor เข้ากับระบบจริง

ผลการทดลองครั้งที่ 2

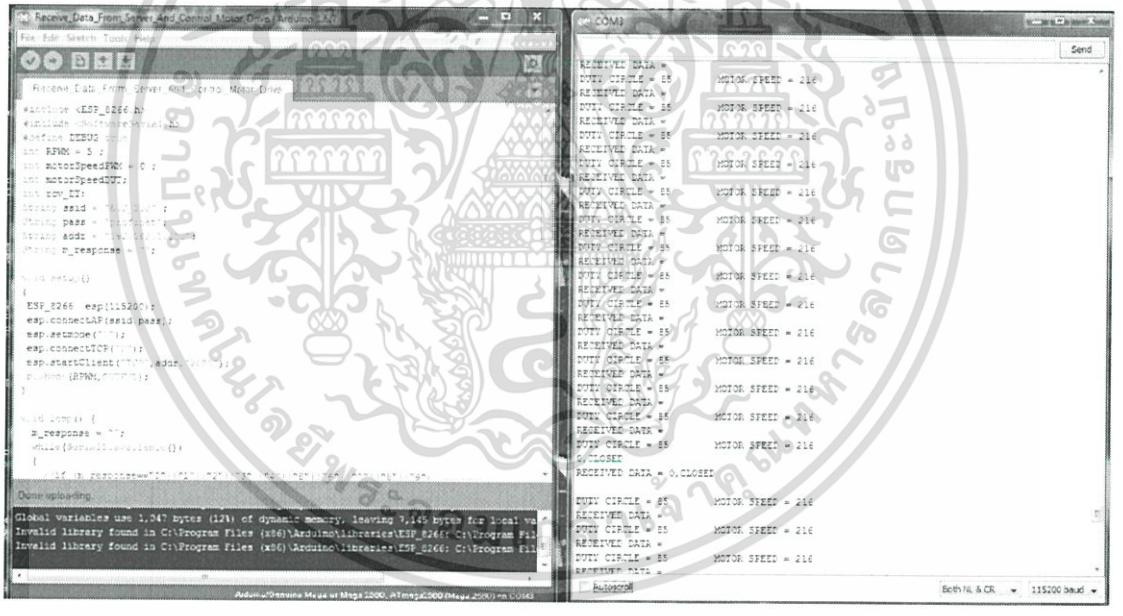
ตารางที่ 4.17 ค่า PWM และค่า Flow Rate ที่ Duty Cycle 60-80%

Duty Cycle (%)	PWM	Control from User	Control from Server
		Flow Rate (L/min)	Flow Rate (L/min)
60	153.00	8	8.4
61	155.55	9	9
62	158.10	9	9
63	160.65	9	9.4
64	163.20	9	10
65	165.75	9.4	10
66	168.30	10	10
67	170.85	10	10
68	173.40	10	11
69	175.95	10	11
70	178.50	11	11
71	181.05	11	11
72	183.60	11	12
73	186.15	11.4	12
74	188.70	12	12
75	191.25	12	12
76	193.80	12	12
77	196.35	12	12
78	198.90	12.7	13
79	201.45	13	13
80	204.00	13	13

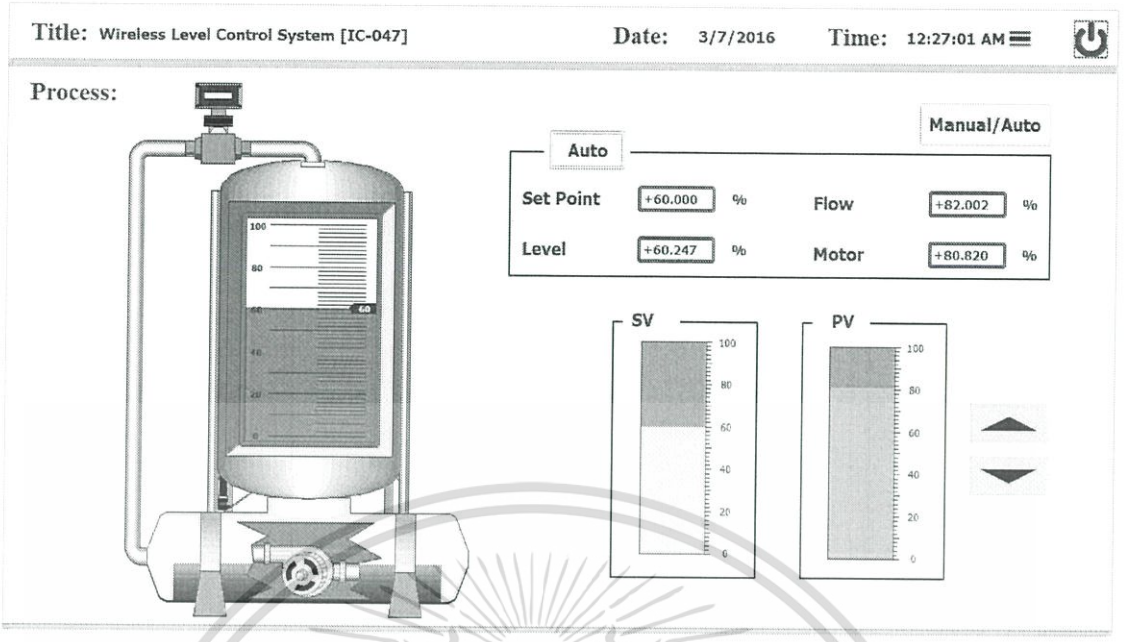
4.13 ผลการทดลองเขียนโปรแกรมรับค่าจาก Server (PLC) เพื่อควบคุมมอเตอร์ปั๊มน้ำ

จากการทดลองสามารถสั่งระบบควบคุม Motor Driver ผ่านการส่งสัญญาณแบบไร้สาย โดยใช้ ESP8266-01 เป็นตัวรับข้อมูล ซึ่งสามารถรับคำสั่งจาก Server (PLC) ส่งมายัง Client (Wi-Fi Module) เพื่อควบคุมมอเตอร์ได้ตามที่ต้องการตาม Duty Cycle จาก 0% จนถึง 100%

ในเริ่มแรกจะพบปัญหาจากการส่งข้อมูลจาก PLC ส่งมายัง Wi-Fi Module โดยเมื่อเปิดดูเอาต์พุตที่ Serial Monitor จะพบว่ามีข้อมูลที่ไม่ต้องการคือ มีสี่เหลี่ยม และเว้นวรรคออกมาพร้อมกับข้อมูลที่ส่งด้วย แก้ปัญหาโดยการแก้ไขที่ระบบส่ง PLC เปลี่ยนจากการส่งข้อมูลแบบ String แปลงมาเป็น Char และอีกปัญหาหนึ่งคือ ตัวโมดูล ESP8266 หลุดการเชื่อมต่อกับ Server แก้ไขโดยเขียนโปรแกรมให้ตัวโมดูลทำการเชื่อมต่อกับ Server ใหม่อีกครั้งโดยอัตโนมัติ เมื่อหลุดการเชื่อมต่อ



รูปที่ 4.33 หน้าต่างโปรแกรม Hercules กับ Serial Monitor ของ Arduino ที่ตัวโมดูล ESP8266 หลุดการเชื่อมต่อกับ Server

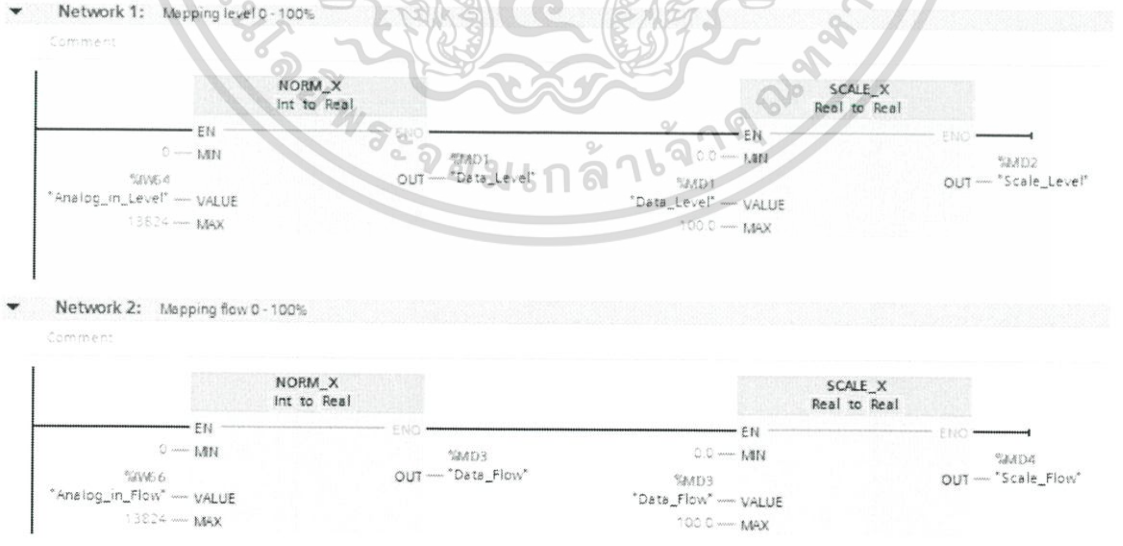


รูปที่ 4.34 หน้าจอ HMI ของ PLC ที่ใช้ควบคุมระบบ

4.14 ขั้นตอนการทดลองเขียนโปรแกรมของ Software

4.14.1 ทดลองการรับค่าที่ขาแอนะล็อกของพีแอลซี โดยบล็อก NORM_X และ SCALE_X

1. เขียนบล็อก NORM_X และ SCALE_X เพื่อทดลองการรับ และอ่านค่าจากขาแอนะล็อกขาเข้า IW64 และ IW66 โดยค่าโวลต์ 0–5 V จะมีค่าอยู่ในช่วง 0–13824 จากนั้นคลิกสัญลักษณ์ Download to device ที่แถบ Toolbar เพื่ออัพโหลดโปรแกรมลงพีแอลซี



รูปที่ 4.35 บล็อก NORM_X & SCALE_X พร้อมใช้งาน และทดสอบ

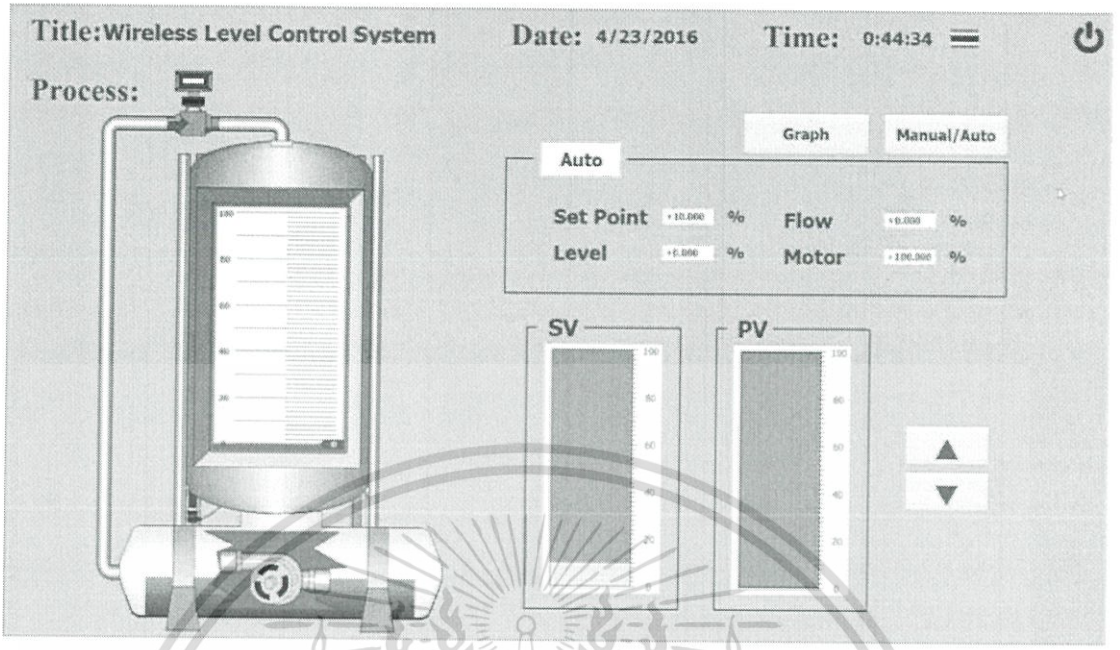
2. กดสัญลักษณ์ Monitoring on/off เพื่อทดลองมอนิเตอร์ค่าโวลต์ตามที่ป้อนเข้าขาอินพุต โดยใช้ฟังก์ชันเจนเนอร์เรเตอร์ (Function Generator) โดยทดลองค่าตังแต่ 0–5 โวลต์หรือ0–13824

ตารางที่ 4.18 ตารางผลการทดลองการรับค่าโวลต์ที่ขาแอนะล็อกพีแอลซี

Exact value		Trial value					
Function generator (V)	PLC rate range	Voltmeter (V)	Error (%)	TIA Portal V13	Error (%)	PLC rate at pin IW66	Error (%)
1.0	2765	1.00	0	1.0	0	2764	0.04
2.0	5530	2.00	0	2.0	0	5563	0.59
3.0	8295	3.02	0.67	3.0	0	8395	1.20
4.0	11060	4.01	0.25	4.0	0	11126	0.59
5.0	13824	5.01	0.2	5.0	0	13890	0.48

4.14.2 ผลการทดลองเปลี่ยนค่าเป้าหมายของระบบเพื่อหา %Overshoot และ %ess

เปิดการทำงาน และควบคุมระบบผ่านหน้าจอ HMI ปรับค่าเป้าหมายตามต้องการ โดยจะเปลี่ยนค่าเป้าหมายทีละ 10 เริ่มจาก 0 ถึง 100 และ จาก 100 ถึง 0 จากนั้นสังเกตกราฟผลตอบสนองที่ได้ของระบบเพื่อหาค่าเปอร์เซ็นต์พุ่งเกิน (%Overshoot) และเปอร์เซ็นต์ความคลาดเคลื่อนที่สภาวะคงที่ (Steady State Error, %ess) ที่ค่าเป้าหมายซึ่งตัวเลขที่ต้องการของทั้งสองค่าจะต้องมีค่าใกล้เคียงกับ 0 ที่สุด



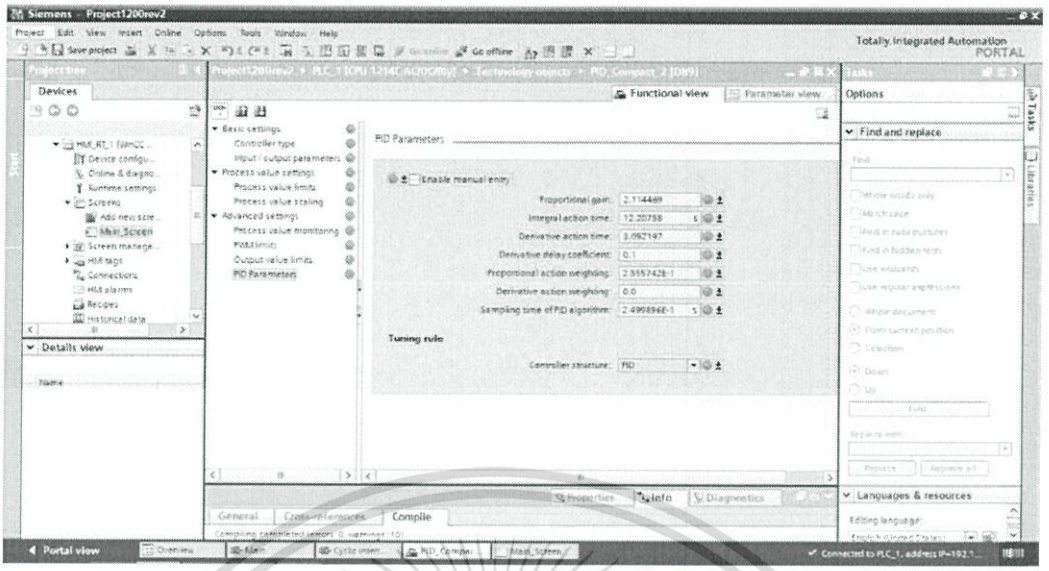
รูปที่ 4.36 หน้าจอ HMI เมื่อปรับค่าเป้าหมายจาก 0 เป็น 10

โดยใช้ค่าพารามิเตอร์พีไอดีของคอนโทรลเลอร์ตัวหลัก ดังนี้

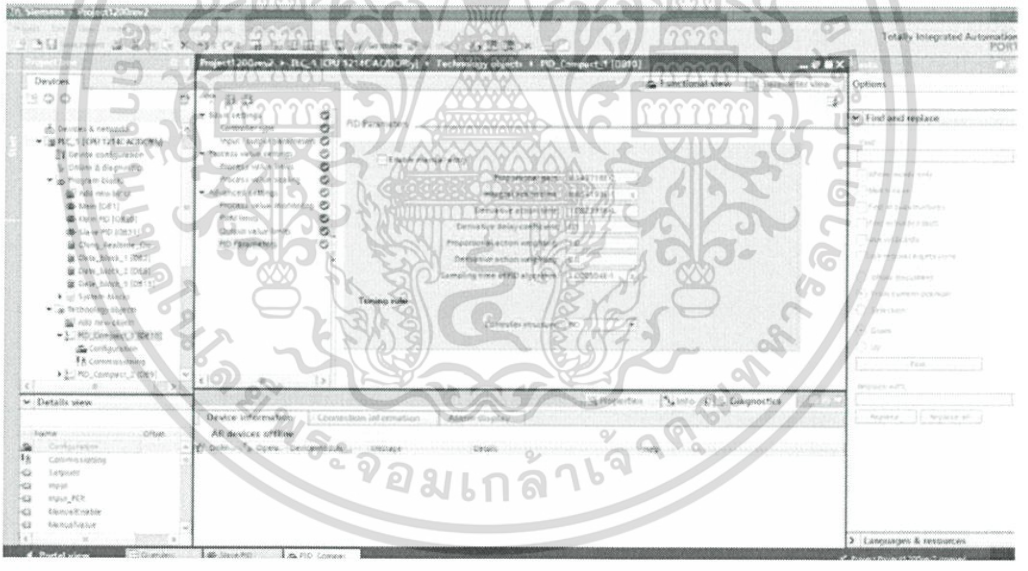
Proportional gain	= 2.114469
Integral action time	= 12.20758 s
Derivative action time	= 3.092197 s
Derivative delay coefficient	= 0.1 s
Proportional action weighting	= 2.555742E-1
Derivative action weighting	= 0.0
Sampling time of PID algorithm	= 2.499896E-1 s

โดยใช้ค่าพารามิเตอร์พีไอดีของคอนโทรลเลอร์ตัวรอง ดังนี้

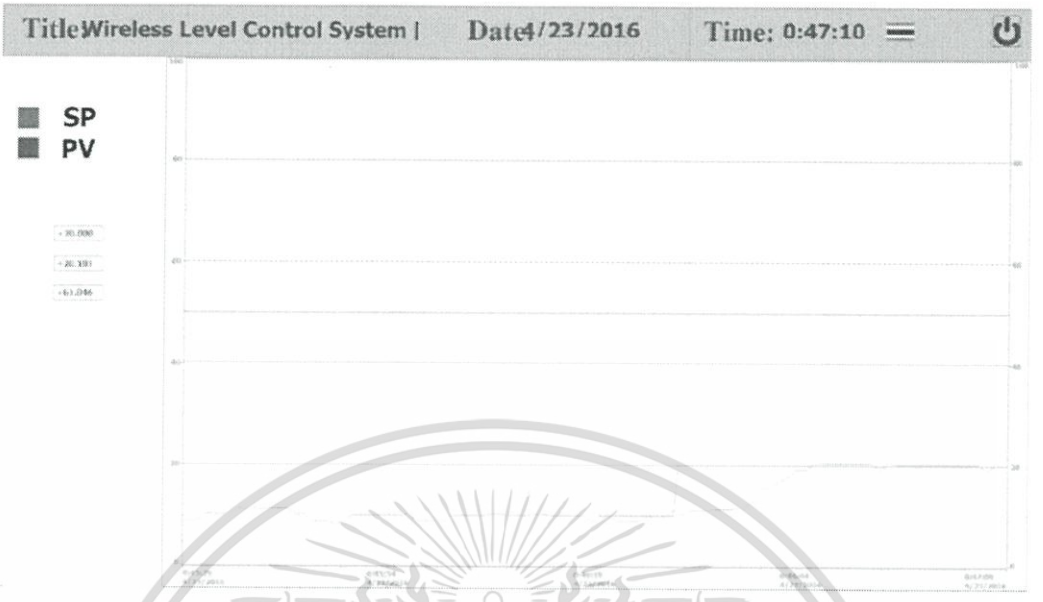
Proportional gain	= 9.149718E-2
Integral action time	= 8.654193E-1 s
Derivative action time	= 1.062393E-1 s
Derivative delay coefficient	= 0.1 s
Proportional action weighting	= 1.0
Derivative action weighting	= 0.0
Sampling time of PID algorithm	= 1.000504E-1 s



รูปที่ 4.37 ค่าพารามิเตอร์พีไอดีของคอนโทรลเลอร์ตัวหลัก

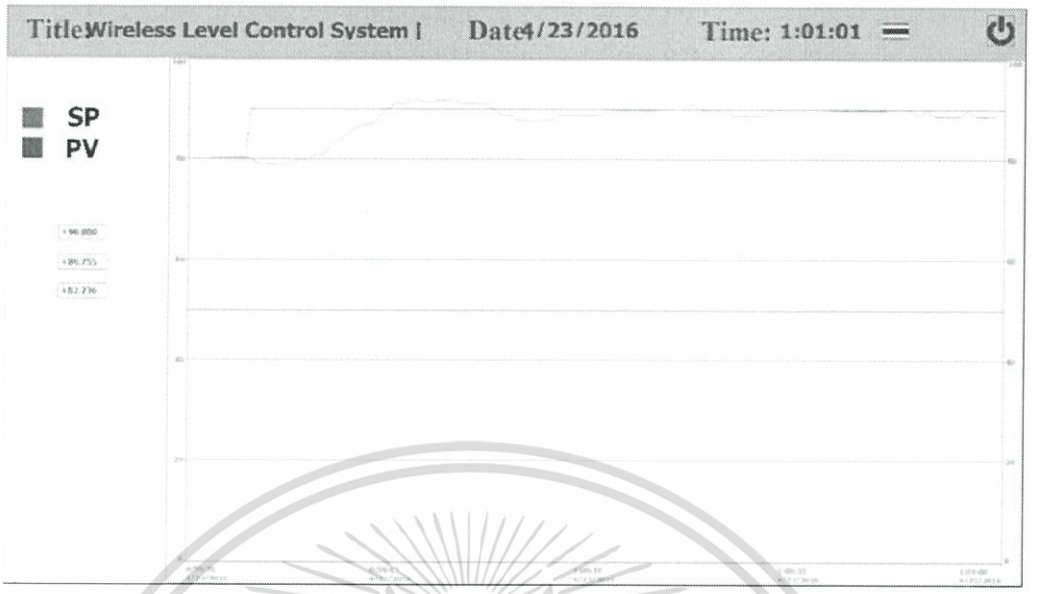


รูปที่ 4.38 ค่าพารามิเตอร์พีไอดีของคอนโทรลเลอร์ตัวรอง



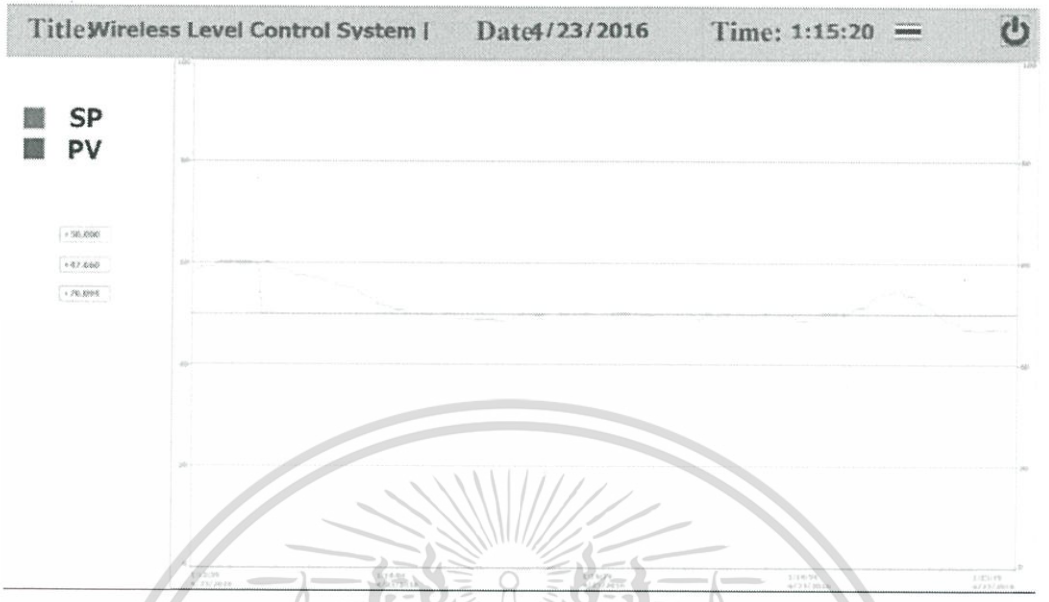
รูปที่ 4.39 กราฟแสดงผลตอบสนองเมื่อเพิ่มค่าเป้าหมายจาก 10 เป็น 20

รูปที่ 4.40 กราฟแสดงผลตอบสนองเมื่อเพิ่มค่าเป้าหมายจาก 40 เป็น 50



รูปที่ 4.41 กราฟแสดงผลตอบสนองเมื่อเพิ่มค่าเป้าหมายจาก 80 เป็น 90

รูปที่ 4.42 กราฟแสดงผลตอบสนองเมื่อลดค่าเป้าหมายจาก 90 เป็น 80



รูปที่ 4.43 กราฟแสดงผลตอบสนองเมื่อลดค่าเป้าหมายจาก 60 เป็น 50

รูปที่ 4.44 กราฟแสดงผลตอบสนองเมื่อลดค่าเป้าหมายจาก 20 เป็น 10

ตารางที่ 4.19 ผลการทดลองเปลี่ยนค่าเป้าหมายของระบบเพื่อหา %Overshoot และ %ess

ค่าเป้าหมาย ก่อนหน้า	ค่าเป้าหมาย ที่ต้องการ	Overshoot	ค่า PV ที่ Steady State	%Overshoot	%ess
0	10	20.331	10	103.31	0
10	20	20.511	20	2.555	0
20	30	-	30	0	0
30	40	41.367	40	3.417	0
40	50	51.367	50	3.182	0
50	60	60.079	60	0.132	0
60	70	71.116	70	1.594	0
70	80	82.073	80	2.591	0
80	90	91.901	90	2.112	0
90	100	-	100	0	0
100	90	86.595	90	3.783	0
90	80	77.768	80	2.79	0
80	70	68.524	70	2.108	0
70	60	58.596	60	2.34	0
60	50	48.546	50	2.908	0
50	40	38.855	40	2.908	0
40	30	28.733	30	4.223	0
30	20	18.826	20	5.87	0
20	10	8.972	10	10.73	0
10	0	-	0	0	0

บทที่ 5

ผลสรุปและข้อเสนอแนะ

จากการทดลองดำเนินงานตามขั้นตอนที่ได้กล่าวมาในส่วนของ Level Transmitter การทดลองต่างๆ มีการพบเจอปัญหาอยู่เป็นระยะๆ ซึ่งต้องใช้ความรู้ และทักษะต่างๆ มาเพื่อแก้ปัญหา โดยในขั้นต้นจะเริ่มแก้ปัญหาจากศึกษาหาข้อมูลทางอินเตอร์เน็ต เพราะในปัจจุบันเทคโนโลยีอินเตอร์เน็ตใช้งานได้อย่างสะดวก และง่ายดาย ทำให้หาข้อมูลได้ไม่ยาก แต่หากหาข้อมูลสิ่งที่ต้องการไม่พบ หรือข้อมูลนั้นยังไม่สามารถแก้ปัญหาได้ ขั้นต่อมาคือการปรึกษาอาจารย์ซึ่งอาจารย์ก็ได้ให้คำปรึกษาเป็นอย่างดี หรือในบางครั้งก็ขอคำแนะนำจากเพื่อนในภาควิชาซึ่งเคยทำการทดลองประเภทนี้มาก่อน เพื่อเป็นแนวทางในการแก้ปัญหา ทำให้ปัญหาต่างๆ สามารถผ่านไปได้

5.1 สรุป

ในส่วนของการสรุปผลการดำเนินงานนั้น แบ่งออกเป็น 3 ส่วนดังนี้

1. Hardware

ในส่วนของการทดลองนั้นขั้นตอนในการออกแบบได้ทำการออกแบบโครงสร้างจากโปรแกรม Solid Works ในเบื้องต้น จากนั้นก็ดำเนินการจัดทำรวมถึงนำชิ้นงานต่างๆ มาประกอบตามแบบที่คาดหวังไว้ และได้ติดตั้งแผงวงจรทั้งหมดอย่างสมบูรณ์ หลังจากนั้นก็ส่งมอบให้กับผู้จัดทำในส่วนของ Level Transmitter, Flow Transmitter, Actuator และโปรแกรมควบคุมทดลองระบบเพื่อทดสอบระบบควบคุมระดับน้ำอัตโนมัติแบบไร้สายต่อไป

2. Transmitter และ Actuator

Pressure Sensor ทำการทดสอบเพื่อดูการทำงานของเซนเซอร์ว่าทำงานได้ และสามารถใช้กับระบบโครงงานนี้ได้ โดยเป็นการทดสอบดูค่าแรงดันที่ได้จากโปรแกรม Arduino และจากมิเตอร์ให้มีค่าที่ใกล้เคียงกันมากๆ และดูค่าเปอร์เซ็นต์ของระดับน้ำในถัง ซึ่งได้ทำการคำนวณเพื่อกำหนดค่าสเกลที่ข้างถังไว้แล้ว มาเปรียบเทียบกับเปอร์เซ็นต์ที่ได้จากโปรแกรมให้มีค่าที่ตรงกัน นำค่าที่ได้ทดสอบได้มาวิเคราะห์เพื่อให้ PLC สั่งการควบคุมระดับน้ำในถังต่อไป ซึ่งจากการทดลองพบว่าในทุกค่าที่นำมาเปรียบเทียบกันนั้น เป็นไปตามที่ได้คาดการณ์ไว้ตรงตามการคำนวณ และมีแนวโน้มไปในทางเดียวกัน แต่อาจมีความคลาดเคลื่อนบ้างเล็กน้อยเท่านั้น

Flow Sensor ทดสอบว่าตัว Sensor สามารถทางานได้ตามปกติและสามารถใช้งานร่วมกับระบบจริงของโครงงานนี้ได้หรือไม่ โดยเป็นการทดสอบดูค่าแรงดันที่วัดได้จาก Flow Sensor แล้วแสดงผ่านหน้าจอ Serial Monitor ของโปรแกรม Arduino ตามค่า Speed Motor ที่สั่งในแต่ละครั้ง นำมาเปรียบเทียบกับค่าที่วัดได้จากมิเตอร์พบว่าในทุกค่าที่นำมาเปรียบเทียบกันนั้น เป็นไปตามที่ได้คาดการณ์และคำนวณไว้ อีกทั้งยังมีแนวโน้มไปในทางเดียวกัน แต่อาจมีความคลาดเคลื่อนบ้าง

เล็กน้อยเท่านั้น ทำให้สรุปได้ว่า Flow Sensor ที่เลือกใช้มีคุณสมบัติตรงกับผลการคำนวณทางทฤษฎี และสามารถใช้งานในระบบนี้ได้จริง

Actuator ซึ่งทำหน้าที่ควบคุมความเร็วของมอเตอร์ปั๊มน้ำ พบว่ามอเตอร์ปั๊มน้ำสามารถทำงานตามคำสั่งจาก Server และทำงานร่วมกับอุปกรณ์อื่นๆ ในระบบได้ โดยหลังจากที่สามารถสร้างระบบควบคุมระดับน้ำแบบไร้สายแล้ว ซึ่งใช้โปรแกรม TIA Portal V13 ในการควบคุม PLC และเขียนโปรแกรมเพื่อใช้งาน Wi-Fi Module ลงบนบอร์ด Arduino ดังนั้น PLC รวมถึงอุปกรณ์ต่างๆ ทั้ง Actuator, Level Transmitter และ Flow Transmitter สามารถส่งข้อมูลถึงกันผ่าน Wi-Fi Module หลังจากการทดลองพบว่าระบบมีเสถียรภาพ โดยสามารถทำให้ระดับน้ำเข้าสู่ Setpoint ที่ต้องการได้หรือสามารถควบคุมระดับน้ำตามต้องการได้

ในส่วนของการเขียนโปรแกรมส่งข้อมูลของ Pressure และ Flow Sensor นั้นจะมีรูปแบบเดียวกันคือ จะกำหนดให้ Wi-Fi Module เป็น Client โดยเริ่มจากการเขียน Library ESP8266 เพื่อใช้สำหรับ Wi-Fi Module จากนั้นทำการเขียนโปรแกรมใน Arduino โดยนำเอาฟังก์ชัน จาก Library ESP8266 ที่เขียนไว้มาใช้งาน ทำการทดสอบส่งข้อมูล โดยในขั้นแรกจะจำลองโปรแกรม Hercules ให้เป็น Server เพื่อรับค่าจาก Wi-Fi Client ซึ่งพบว่าตัว Wi-Fi สามารถส่งค่าออกไปให้กับ Server ได้ และหลังจากที่เขียนโปรแกรมรับค่าให้กับ Wi-Fi Server เรียบร้อยแล้ว ได้ทำการเขียนโปรแกรมส่งค่าของ Wi-Fi Client เพิ่มเติมในบางส่วน เนื่องจากเปลี่ยนตัว Server จากโปรแกรม Hercules มาเป็นโมดูล Wi-Fi จากนั้นทำการทดสอบส่งข้อมูลให้กับ Wi-Fi Server ซึ่งจากการทดสอบพบว่าโมดูล Wi-Fi สามารถรับส่งข้อมูลได้ตามที่ต้องการ

ในส่วนของการเขียนโปรแกรมเพื่อรับข้อมูลนั้นจะกำหนดให้ Wi-Fi Module เป็น Server เพื่อรับค่า ทำการเขียนโปรแกรม Arduino โดยนำฟังก์ชันใน Library ESP8266 ที่ได้เขียนไว้มาใช้งานเช่นเดียวกับการเขียนโปรแกรมเพื่อส่งข้อมูล แต่กำหนดให้เป็น Wi-Fi Server เพื่อรับค่าจาก Wi-Fi Client ซึ่งพบว่า Wi-Fi สามารถรับค่าที่ส่งมาได้ตามที่เขียนโปรแกรมไว้ จากนั้นจึงเขียนคำสั่งเพิ่มเติมกำหนดให้ขา PWM ที่บอร์ด Arduino เป็นขาเอาต์พุต ส่งข้อมูลออกไปให้กับ PLC ทำการวัดค่าแรงดันที่ได้ออกมาจากขา PWM และที่ PLC เปรียบเทียบกับค่าที่แสดงใน Serial Monitor เพื่อตรวจสอบให้ค่าที่ได้ออกมานั้นมีค่าตรงกัน และโปรแกรมกับระบบที่ได้ดำเนินการมาสามารถใช้งานได้จริง ซึ่งพบว่าค่าแรงดันที่ได้เป็นไปที่ได้คำนวณไว้ และโปรแกรมสามารถรับค่าจาก Wi-Fi Client แล้วส่งค่าเอาต์พุตให้กับ PLC ได้ตามที่เขียนโปรแกรมไว้

ในส่วนการทดสอบ Motor Driver BTS7960 43A ทำการทดลองเขียนโปรแกรมควบคุมการทำงานของโมดูลขับมอเตอร์กระแสตรง โดยให้มอเตอร์หมุนไปทางขวา และทางซ้ายด้วยความเร็วเพิ่มขึ้นจาก 0-255 PWM สลับกัน และเขียนโปรแกรมควบคุมความเร็วมอเตอร์ปั๊มน้ำโดยใช้ Duty Cycle 0-100% พบว่า สามารถควบคุมความเร็วมอเตอร์ปั๊มน้ำตามที่ผู้ใช้ต้องการได้ ในโครงงานนี้ได้ทำการเขียน Library ของ Wi-Fi Module ESP8226 ขึ้นมาเองโดยใช้ Software Serial เพื่อใช้งานร่วมกับ Arduino Mega 2560 R3 ได้ง่ายขึ้น

ในส่วนของการเขียนโปรแกรมรับค่า Duty Cycle 0-100% จาก Server มาควบคุมมอเตอร์ปั๊มน้ำ โดย Server ที่ทำการทดลองคือ โปรแกรม Hercules และ PLC ในการทดลองแรกให้ Server เป็น Hercules แล้วทำการกำหนด IP Address และ Port ของคอมพิวเตอร์ที่จะจำลองเป็น Server คือ 192.168.1.28 และ 6000 ตามลำดับ โดย Server จะต้องเชื่อมต่อ Access Point เดียวกันกับโมดูล ESP8266 ซึ่งการรับค่าจาก Server จะต้องทำการแปลงค่าที่ได้รับมาให้อยู่ในรูปแบบจำนวนเต็ม (Integer) ก่อน และเนื่องจากการควบคุมความเร็วมอเตอร์จะต้องใช้ PWM (0-255) แต่ค่าที่รับมาคือ Duty Cycle 0-100% จึงใช้ Function map เพื่อแปลงสเกลค่าจาก 0-100 ให้เป็น 0-255 จึงสามารถสั่งควบคุมความเร็วของมอเตอร์ให้ทำงานตามค่าที่ Server ส่งมาได้ หลังจากนั้นได้ทำการทดลองให้ PLC ทำหน้าที่เป็น Server ซึ่งจะนำโปรแกรมที่ใช้รับค่า Duty Cycle 0-100% จาก Server (Hercules) มาปรับแก้ให้สามารถรับค่าจาก PLC ได้ โดยทำการเปลี่ยน IP Address และ Port ให้เป็นของ PLC คือ 192.168.1.100 และ 2000 ตามลำดับ โดย PLC จะต้องเชื่อมต่อ Access Point เดียวกันกับโมดูล ESP8266 จึงจะทำการเชื่อมต่อกันได้ และเมื่อ PLC ส่งค่า Duty Cycle 0-100% มา ตัวโมดูลจะทำการรับค่า และนำค่าที่ได้ไปแปลงเป็นค่าของ PWM เพื่อทำการสั่งมอเตอร์ปั๊มน้ำต่อไป

3. Software

จากผลการทดลองสรุปได้ว่าพีแอลซี สามารถรับส่งข้อมูลผ่าน WiFi จากเซนเซอร์พร้อมกัน 2 ตัวได้โดยการรับค่าแรงดันไฟฟ้าจากตัว Module WiFi ที่ต่อเข้าขาแอนะล็อกพีแอลซี ได้โดยมีความสูญเสียแรงน้อยมาก นอกจากนีสรุปว่าค่าพารามิเตอร์ของตัวควบคุมหลักที่ P = 2.114469, Ti = 12.20758 s และ Td = 3.092197 s และค่าพารามิเตอร์ตัวควบคุมรองที่ P = 9.149718E-2, Ti = 8.654193E-1 s และ Td = 1.062393E-1 s เป็นค่าที่เหมาะสมสำหรับระบบควบคุมระดับน้ำแบบไร้สายจำลองระบบนี้ เนื่องจากเป็นค่าที่ทำให้ค่าเปอร์เซ็นต์ความผิดพลาดที่สถานะคงตัว หรือ %ess กับค่า %Overshoot มีค่าน้อยใกล้เคียง 0 มากที่สุด

หลังจากที่ได้ออกแบบ และสร้างระบบควบคุมระดับน้ำแบบไร้สายที่มีต้นทุนต่ำกว่าปกติแล้วนั้น ระบบสามารถทำให้ระดับน้ำเข้าสู่ Setpoint ที่ต้องการได้ โดยใช้โปรแกรม TIA Portal V13 เพื่อควบคุม PLC และเขียนโปรแกรมใช้งานโมดูล Wi-Fi ลงบนบอร์ด Arduino เพื่อให้ PLC และอุปกรณ์ต่างๆ สามารถส่งข้อมูลถึงกันผ่านโมดูล Wi-Fi พบว่าระบบมีเสถียรภาพ และสามารถควบคุมระดับน้ำตามต้องการได้

5.2 ปัญหาและอุปสรรค์ในการดำเนินโครงงานนี้

1. เนื่องจากยังขาดทักษะ ประสบการณ์และความชำนาญสำหรับการทำ โครงงานนี้อยู่มาก ทำให้เกิดปัญหาและใช้เวลาค่อนข้างมากในการดำเนินงาน

2. จากการทดลองเขียนโปรแกรมเพื่อส่งค่าให้กับ Server ของ Level และ Flow Transmitter นั้นพบว่าในการเชื่อมต่อโมดูล Wi-Fi ESP8266 เกิดปัญหาอยู่บ่อยครั้ง ทั้งการเชื่อมต่อ

ระหว่าง Wi-Fi Client กับ Wi-Fi Server เชื่อมต่อกันไม่พบ และหลุดการเชื่อมต่อ ทำให้ค่าที่ส่งไปเป็น 0 ซึ่งจะส่งผลต่อระบบทำให้ไม่สามารถควบคุม ระดับน้ำได้ ดังนั้นจึงทำการเขียนโปรแกรมเพิ่มโดยการกำหนดช่วงที่พิจารณาเท่านั้น ซึ่งหากค่าไม่อยู่ในช่วง PLC จะสั่งให้มอเตอร์ทำงานที่ค่า 0 คือไม่ปั๊มน้ำเข้าสู่ระบบ

3. จากการทดลองเขียนโปรแกรมรับค่า และต่อวงจรเพื่อส่งค่าให้กับ PLC ของ Level และ Flow Transmitter นั้นพบว่าในช่วงแรกการเขียนโปรแกรมรับค่านั้น ค่าที่ได้มามีค่าอื่นนอกเหนือจากที่ต้องการส่งมาด้วย จึงต้องเขียนคำสั่งเพิ่มเติมเพื่อตัดเอาค่าที่ไม่ต้องการออก และในการต่อวงจรขยายแรงดัน เพื่อเพิ่มแรงดันก่อนที่จะส่งค่าเข้า PLC นั้น เกิดปัญหาในการต่อวงจรเนื่องจากผู้จัดทำยังขาดประสบการณ์ ทำให้การต่อวงจรเพื่อทดลองในแต่ละครั้งเกิดความล่าช้า

4. จากการทดลองเขียนโปรแกรมรับค่าจาก Server (PLC) เพื่อควบคุมมอเตอร์ปั๊มน้ำ พบว่าตัวโมดูล ESP8266 มักจะหลุดการเชื่อมต่อจาก Server ทำให้ยังค้างค่าเก่าที่ส่งมา ซึ่งอาจเกิดอันตรายได้ เช่น ถ้าระดับน้ำอยู่ที่ 80% แล้วมอเตอร์หลุดการเชื่อมต่อ แล้วค้างค่าก่อนหน้าคือ 100% จะทำให้น้ำล้นได้ ดังนั้นจึงต้องเขียนโปรแกรมเพิ่มคือ เมื่อตัวโมดูล ESP8266 หลุดการเชื่อมต่อ ให้สั่งมอเตอร์ให้หยุดการทำงาน แล้วค่อยเชื่อมต่อกับ Server ใหม่อีกครั้ง

5. โปรแกรม SIMATIC PCS7 ไม่รองรับ PLC Siemens S7-1200 จึงต้องเปลี่ยนจากโปรแกรม SIMATIC PCS7 มาใช้โปรแกรม STEP7 TIA Portal V13 แทน

6. โปรแกรม STEP7 TIA Portal V13 สามารถเชื่อมต่อกับ Module Wi-Fi ได้ทีละตัว จึงได้เพิ่ม Module Wi-Fi เพื่อรับค่าจากตัว Module Wi-Fi ที่ฝั่งระบบจำลอง แล้วต่อเข้ากับขาแอนะล็อกของพีแอลซีโดยตรง

7. ในบางครั้งการเชื่อมต่อมอเตอร์ผ่าน Wi-Fi ไม่ประสบผลสำเร็จ เนื่องจากหมายเลข IP Address ของ Module Wi-Fi มีการเปลี่ยนแปลง จึงต้องตรวจเช็ค IP Address ของ Module Wi-Fi ให้ตรงกันทุกครั้งก่อนการเชื่อมต่อ

5.3 ข้อเสนอแนะ

1. เนื่องจากโมดูล Wi-Fi ที่ใช้ในโครงงานนี้มีราคาถูก ทำให้เกิดความไม่เสถียรอยู่มากในการเชื่อมต่อ ต้องเตรียมโมดูลไว้สำรองจำนวนหนึ่ง เพราะเมื่อเกิดความไม่เสถียรจะต้องใช้เวลาในการรีเซตเพื่อนำกลับมาใช้ใหม่

2. เครื่องมือ และอุปกรณ์ยังมีไม่เพียงพอสำหรับการใช้งาน ทำให้เกิดความไม่สะดวก และล่าช้า เนื่องจากต้องหาอุปกรณ์มาใช้ในการดำเนินงาน

3. ควรศึกษา ค้นคว้าข้อมูลของสิ่งที่จะต้องทำเป็นอย่างดี โดยเฉพาะพื้นฐานการเขียนโปรแกรม Arduino เพื่อให้เกิดความรวดเร็วในการดำเนินงาน และควรมีการวางแผนงานอย่างมีระบบ เพื่อให้โครงงานเสร็จสิ้นตามกำหนดการอีกด้วย

4. โครงงานนี้สามารถประยุกต์ใช้กับระบบอุตสาหกรรมที่มีขนาดใหญ่ได้ แต่จะต้องมีการใช้โมดูล Wi-Fi ที่มีราคาสูงกว่านี้ เพื่อป้องกันไม่ให้ระบบเสียหาย ซึ่งโครงงานนี้จะช่วยให้เกิดความสะดวกในการดำเนินงาน และลดต้นทุนการผลิตได้อีกด้วย

5. เนื่องจากทางผู้จัดทำใช้ PLC Siemens S7-1200 ที่เหมาะกับงานขนาดเล็ก และอาจมีเสถียรภาพไม่สูงนัก หากต้องการนำไปใช้งานในระบบที่ใหญ่ และต้องการเสถียรภาพสูง ควรเลือกใช้พีแอลซีรุ่นที่เหมาะสมในการใช้งานต่อระบบนั้น

เอกสารอ้างอิง

[1] "**Program Solidworks**" เข้าถึงได้จาก: http://solidworkweb.blogspot.com/2009/08/solidwork.html(วันที่สืบค้นข้อมูล : 19 กันยายน 2558)

[2] **"วิธีใช้งาน Solidwork เบื้องต้น"** เข้าถึงได้จาก: https://www.youtube.com/watch?v=AcfLJfHLPHk **(วันที่สืบค้นข้อมูล : 19 กันยายน 2558)**

[3] **"Process and Plant."** เข้าถึงได้จาก: http://www.solidworks.com/sw/industries/7133_ENU_HTML.htm (วันที่สืบค้นข้อมูล : 22กันยายน 2558)

[4] **"SolidWorks เบื้องต้น"** เข้าถึงได้จาก: https://www.youtube.com/watch?v=xLd8Ds0mFeY (วันที่สืบค้นข้อมูล : 25 กันยายน 2558)

[5] **"เขียนแบบ solid works"** เข้าถึงได้จาก: https://www.youtube.com/watch?v=TkYgHyY-xls (วันที่สืบค้นข้อมูล : 12 ตุลาคม 2558)

[6] **"วิธีการแปลงไฟล์ 3D Printer For MakerBot"** เข้าถึงได้จาก: http://www.homeofmaker.com/?p=1062 (วันที่สืบค้นข้อมูล : 20 ตุลาคม 2558)

[7] **"สอนใช้โปรแกรม MakerWare"** เข้าถึงได้จาก: https://www.youtube.com/watch?v=z0CmE0wXZWw (วันที่สืบค้นข้อมูล : 11 ธันวาคม 2558)

[8]**"MakerBot"** เข้าถึงได้จาก: https://eu.makerbot.com/fileadmin/Inhalte/ Support/Manuals/MakerBot_Replicator2X_UserManual_Eng.pdf (วันที่สืบค้นข้อมูล:11 ธันวาคม 2558)

[9] **"การวัดระดับแบบไฮโดรสแตติก"** เข้าถึงได้จาก: http://www.Foodnetworksolution.com /wiki/word /7329/hydrostatic-level- measurement(วันที่สืบค้นข้อมูล: 27 ตุลาคม 2558)

[10] **"Protocol"** เข้าถึงได้จาก: https://sites.google.com/site/maysarm43/phl-ngan-khxng-chan(วันที่สืบค้นข้อมูล: 28 ตุลาคม 2558)

[11] **"ความดันสถิต"** เข้าถึงได้จาก: http://www.foodnetworksolution.com/wiki/word/4318 /static-head(วันที่สืบค้นข้อมูล: 28 ตุลาคม 2558)

[12] **"ESP8266"** เข้าถึงได้จาก: http://www.ioxhop.com/article/(วันที่สืบค้นข้อมูล: 28 ตุลาคม 2558)

[13] **"Arduino"** เข้าถึงได้จาก: http://www.myarduino.net/article/(วันที่สืบค้นข้อมูล: 28 ตุลาคม 2558)

[14] **"Pressure Sensor"** เข้าถึงได้จาก: http://www.ni.com/whitepaper/3639/en/ (วันที่สืบค้นข้อมูล: 28 ตุลาคม 2558)

เอกสารอ้างอิง (ต่อ)

[15] **"ESP8266"** เข้าถึงได้จาก: p://www.ayarafun.com/2014/09/esp8266-at-command-tutorial/(วันที่สืบค้นข้อมูล 29 ตุลาคม 2558)

[16] **"เทคโนโลยี Wi-Fi"** เข้าถึงได้จาก: http://www.vcharkarn.com/blog/34921/4746(วันที่สืบค้นข้อมูล 30 ตุลาคม 2558)

[17] **"สอน-วิธี-ใช้งาน-arduino-wi-fi-module-esp8266"**เข้าถึงได้จาก: http://www.arduinoall.com/article (วันที่สืบค้นข้อมูล 30 ตุลาคม 2558)

[18] **"การใช้ Pressure sensor"** เข้าถึงได้จาก: http//langster1980.blogspot.com/2014/11 how-to-use-pressuresensorwith.html?m=1(วันที่สืบค้นข้อมูล 30 ตุลาคม 2558)

[19] **"เริ่มต้นการเขียนภาษา C++"** เข้าถึงได้จาก: http://www.comscicafe.com/article/1/-cpp-lesson-1-tutortong#.Vwl1P6SLTIU(วันที่สืบค้น 9 กุมพาพันธ์ 2559)

[20] **"พื้นฐานภาษาซี"** เข้าถึงได้จาก: http://kanokwan.sru.ac.th/e-learning/basic.php(วันที่สืบค้น 9 กุมพาพันธ์ 2559)

[21] **"การเขียน Arduino เบื้องต้น"** เข้าถึงได้จาก:http://www.myarduino.net/articlearduino (วันที่สืบค้น 10 กุมพาพันธ์ 2559)

[22] **"โครงสร้างภาษาซี Arduino เบื้องต้น"** เข้าถึงได้จาก: http://www.ioxhop.com/article/arduino(วันที่สืบค้น 10 กุมพาพันธ์ 2559)

[23] **"ออปแอมป์"** เข้าถึงได้จาก: http://www.sukhothaitc.ac.th/faifa/Personal/kraisorn/teaching/OP-A.htm(วันที่สืบค้น 25 กุมพาพันธ์ 2559)

[24] **"วงจรเพิ่มแรงดัน DC"** เข้าถึงได้จาก: http://www.thaiconverter.com/article(วันที่สืบค้น 25 กุมพาพันธ์ 2559)

[25] **"IC LM358"** เข้าถึงได้จาก: http://www.onsemi.com/pub_link/Collateral/LM358D.PDF(วันที่สืบค้น 25 กุมพาพันธ์ 2559)

[26] **"Switching Power Supply"** เข้าถึงได้จาก: http://www.voltextra.com/switching-power -supply/ (วันที่สืบค้น 25 กุมพาพันธ์ 2559)

[27] **"Constrain"** เข้าถึงได้จาก: https://www.arduino.cc/en/Reference/Constrain (วันที่สืบค้น 29 กุมพาพันธ์ 2559)

[28] **"Reading Water Flow Rate from a flow meter"** เข้าถึงได้จาก: http://forum.arduino.cc/index.php/topic,8548.0.html (วันที่สืบค้นข้อมูล: 25 กันยายน 2558)

[29] **"บทความ Arduino"** เข้าถึงได้จาก: http://www.arduinoall.com/article (วันที่สืบค้นข้อมูล:24 กันยายน 2558)

เอกสารอ้างอิง (ต่อ)

[30] **“Water Flow Sensor”**เข้าถึงได้จาก: http://www.seeedstudio.com /wiki/G11/4_WaterFlow_Sensor (วันที่สืบค้นข้อมูล: 25 กันยายน 2558)

{31] **“Hall Effect Sensor”** เข้าถึงได้จาก: http://mcu56.learninginventions.org/page_id= 258(วันที่สืบค้นข้อมูล: 28 กันยายน 2558)

[32] **“Regulator 4.5V-7V to 3.3V”** เข้าถึงได้จาก: http://www. arduinoall.com/product 12/4-5v-7v-to-3-3v-ams1117-3-3v-power-supply-module-ams-1117 (วันที่สืบค้นข้อมูล: 10 กันยายน 2558)

[33] **“วิธีการใช้งาน Arduino ส่ง Data ผ่าน TCP ส่งผลไปยัง Server”** เข้าถึงได้จาก: http://www.thaieasyelec.com/article-wiki//resareview-product-article//wifi-data-logging-using-arduino.html (วันที่สืบค้นข้อมูล: 10 ตุลาคม 2558)

[34] Deitel,Harvey, M. ,Deitel & Paul. (1997). **C++ How to Program** (2nd Edition). Prentice Hall.

[35] “**เทคโนโลยี Wi-Fi”** เข้าถึงได้จาก: http://www.vcharkarn.com/blog/34921/4746 (วันที่สืบค้นข้อมูล: 27 ตุลาคม 2558)

[36] **“Protocol”** เข้าถึงได้จาก: https://sites.google.com/site/maysarm43/phl-ngan-khxng -chan (วันที่สืบค้นข้อมูล: 27 ตุลาคม 2558)

[37] **“โครงสร้างของ Protocol TCP/IP”** เข้าถึงได้จาก: http://itguest.blogspot.com/2011 /04/04-tcpip.html (วันที่สืบค้นข้อมูล: 27 ตุลาคม 2558)

[38] **“รู้จักกับ ESP8266 และรุ่นที่นิยมใช้งาน”** เข้าถึงได้จาก: http://www.ioxhop.com/ article/esp8266-ตอนที่-1-รู้จักกับ-esp-และรุ่นที่นิยมใช้งาน (วันที่สืบค้นข้อมูล: 27 ตุลาคม 2558)

[39] **“มอเตอร์กระแสตรง”** เข้าถึงได้จาก: http://www.adisak51.com/page21.html (วันที่สืบค้นข้อมูล: 28 ตุลาคม 2558)

[40] **“ESP8266 วิธีทดสอบและใช้งาน”** เข้าถึงได้จาก: http://www.ayarafun.com/2014/09/ esp8266-at-command-tutorial/ (วันที่สืบค้นข้อมูล: 29 ตุลาคม 2558)

[41] **“1x BTS7960 H-Bridge DC Motor Drive (6-27V 43A Max) Module”** เข้าถึงได้จาก: http://www.micontechlab.com/product/787/1x - bts7960-full-bridge-dc-motor-drive-6-27v-43a-max-module (วันที่สืบค้นข้อมูล: 29 ตุลาคม 2558)

[42] **“Motor Driver BTS7960 43A”** เข้าถึงได้จาก: http://www.instructables.com/id/ Motor-Driver-BTS7960-43A/ (วันที่สืบค้นข้อมูล: 30 ตุลาคม 2558)

เอกสารอ้างอิง (ต่อ)

[43] **"รูปแบบการเชื่อมต่อของระบบเครือข่ายไร้สาย"** เข้าถึงได้จาก: http://www.wifi.kmutnb.ac .th/StudyWireless.php (วันที่สืบค้นข้อมูล: 5 พฤษภาคม 2559)

[44] Hans Berger. (2554). Automating in STEP 7 Basic with SIMATIC S7-1200. United-State. Wiley

[45] ธีรศิลป์ ทุมวิภาต. (2547). เรียนรู้ PLC ขั้นกลางด้วยตนเอง. กรุงเทพมหานคร. ซีเอ็ดยูเคชั่น

[46] **"Siemens S7-1200 System Manual"** เข้าถึงได้จาก:http://www.generationrobots.com/media/manuel-plc-siemens-s7-en.pdf (วันที่สืบค้นข้อมูล : 21 กันยายน 2558)

[47] **"Technical specification SIMATIC S7-1200"** เข้าถึงได้จาก: http://www.paratrasnet.ro/pdf/automatizari-industriale/S7-1200.pdf (วันที่สืบค้นข้อมูล : 19 ตุลาคม 2558)

[48] **"คุณสมบัติของ PLC Siemens model S7-1214C AC/DC/Relay (6ES7 214-1BG310XB0)"**เข้าถึงได้จาก : http://docs- europe.electrocomponents.com/webdocs/1129/0900766b81129785.pdf (วันที่สืบค้นข้อมูล : 21 ตุลาคม 2558)

[49] **"User Communication"** เข้าถึงได้จาก:https://cache.industry.siemens.com/dl/files/808/67196808/att_108115/v1/net_s7-1200_isoontcp_en.pdf (วันที่สืบค้นข้อมูล : 9 ธันวาคม 2558)

[50] **"PID Control with PID_Compact"** เข้าถึงได้จาก :http://www.infoplc.net/files/descargas/siemens/infoPLC_net_100746401_S71200_PID_Compact_DOKU_v10_en.pdf (วันที่สืบค้นข้อมูล : 10 มกราคม 2559)

[51] **"Analog input specification SIMATIC S7-1200"** เข้าถึงได้จาก: http://www.sahkonumerot.fi/2702074/doc/technicalinfodoc/ (วันที่สืบค้นข้อมูล : 24 มกราคม 2559)

[52] **"Cascade PID Control"** เข้าถึงได้จาก:http://thaicontrol.wordpress.com/2012/02/13/cascade-pid-control/ (วันที่สืบค้นข้อมูล : 13 มีนาคม 2559)

[53] **"Process control"** เข้าถึงได้จาก: http://www.thailandindustry.com/guru/view.php?id=15688 (วันที่สืบค้นข้อมูล : 13 มีนาคม 2559)

[54] **"Fundamentals of cascade control"** เข้าถึงได้จาก:http://www.controleng.com/single-article/fundamentals-of-cascade-control (วันที่สืบค้นข้อมูล : 13 มีนาคม 2559)

ภาคผนวก

ภาคผนวก ก
โปสเตอร์

IC-047

Wireless Level Control System

Ms.Waratchaya Boonchuoi, Ms.Pataraporn Singhaphet, Ms.Thanisa Musikasopon, Ms.Narisara Leatjantuk, Ms.Chanida Sriprab, Ms.Jitraphorn Chusaeng, Mr.Chanathip Chokkanapitak, Mr.Worrathun Thamsuntree, Ms.Parnravee Pornthisarn, Ms.Phattrawadee Sritas, Ms.Pantip Panyasit, Ms.Ramrada Khunnaphap, Ms.Siwan Sasipaisith and Mr.Sarat Pongprasert

Advisor: Prof. Dr. Vanchai Riewruja, Asst.Prof. Thepjit Cheypoca, Asst.Prof. Dr. Wandee Petchmaneelumka and Dr. Sirichai Thammarakwattana

Department of Instrumentation and Control Engineering, Faculty of Engineering

E-mail: 55011089@kmitl.ac.th, 55011046@kmitl.ac.th

Abstract

This project presents a design and implementation of a small plant for study and simulation of closed-loop control systems in the industrial work. Level control system is used for study in this project. Moreover, controlling with wireless is employed in the system. The project divides into two parts; wireless plant networks and controlling method.

First, the plant is built as a water level controlling plant. Transducers used in the model plant consist of two sensors and one motor functioned as flow and level transmitters and actuator. Arduino, a widely used microcontroller in engineering works, is employed to provide establish a communication between transducers and controller over Wi-Fi. As mention earlier, control system based on wireless connectivity can reduce signaling cable requirement, which is helpful in a big plant using wired system.

Second, PLC is used as a controller to control the level of water in the tank to the point where users need. There are various types of controller, however PID controller is selected as the plant control method. A PID controller is a control loop feedback mechanism commonly used in industrial control systems. A PID controller continuously calculates an error value as the difference between a desired setpoint and a measured process variable then attempts to minimize the error over time by adjustment of a control variable. TIA Portal V13 is used to program a control logic and PID controller to make the water reach to the setpoint of the wireless level control system spontaneously.

Introduction

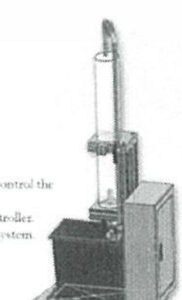
At present, most industrial factory prefer using PLC to control plant from remote location reducing the number of wiring used for connecting relays. Nevertheless, it still need some wiring to connect between some elements such as sensors and actuators. SIEMENS has developed a client module to eliminate the problem by connect via wireless network instead of regular wiring. Due to the fact that client module has high price, so we use Wi-Fi module ESP8266-01 to control water level in the wireless level control system as it has lower cost. Wi-Fi module is used as a medium of a communication between PLC and components. The model plant is designed using SolidWorks and controlled with PLC SIEMENS model S7-1200 via Wi-Fi module ESP8266-01.

Objectives

1. Study of the principle of microcontroller and the communication via Wi-Fi module.
2. Study of the usage of Arduino and TIA Portal v13.
3. Design a wireless control system.

Methodology

1. Build a plant for study.
2. Study about sensor and actuator.
3. Design a circuit board for sensor and motor.
4. Create a library for Wi-Fi ESP8266-01.
5. Program the Arduino boards to send/ receive data and control the motor.
6. Make ladder program to control the plant using PID controller.
7. Make an interface for users for monitor and control the system.
8. Assemble all component of the plant together.
9. Tune PID controller and control the system via Wi-Fi.

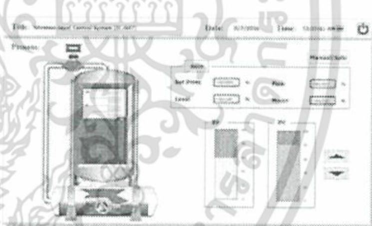


Results

The experimental result shows that the devices in the wireless system can communicate with each other over Wi-Fi by sending the data from pressure sensor to PLC then using TIA Portal to tune PID parameters while the setpoint of the system is 60%. The PID parameters are set as follow:

Proportional gain	=	3.506227E-1
Integral action time	=	14.78867 s
Derivative action time	=	3.986802 s
Derivative delay coefficient	=	0.1
Proportional action weighting	=	5.480434E-1
Derivative action weighting	=	0.0
Sampling time of PID algorithm	=	3.000916E-1 s

Afterward, PLC will send a data to control motor speed and pump the water and control it to the setpoint level. When the setpoint is changed to a difference values, the system still able to keep the stability at the request values.



Conclusion

We simulated an inexpensive water level control system to replace the commonly used in the industrial one and be able to control it to reach the setpoint. By using TIA Portal to program a PLC and using Arduino to connect PLC with the devices over Wi-Fi module, the system is stable and able to be controlled as user desire. Which can be applied to a bigger plant in a control field of work.

References

[1] Deitel, Harvey, M. and Deitel, Paul. (1997). C++ How to Program. 2nd Edition. Hall.
[2] MaysPrentice arm43. "Protocol" [Online]. Available : https://sites.google.com/site/maysarm43/ohl-ngan-khang-chan
[3] Electrocomponents "คุณสมบัติของ PLC Siemens model S7-1214C AC/DC/Relay (6ES7 214-1BG31- 0XB0)." [Online]. Available : http://docseurope.electrocomponents.com/webdocs/1129/0900766b81129785.pdf
[4] Paratrasnet. "คุณสมบัติของ PLC Siemens model S7-1214C AC/DC/Relay (6ES7 214-1BG31- 0XB0)." [Online]. Available : http://www.paratrasnet.ro/pdf/automatizari-industriale/S7-1200.pdf
[5] Sahkonumerot. "ข้อมูล analog datasheet." [Online]. Available : https://www.sahkonumerot.fi/2702074/doc/technicalinfodoc/
[6] Arduino.cc "ตัวอย่าง code ของการอ่านค่า Flow sensor." [Online]. Available : http://forum.arduino.cc/index.php/topic,8548.0.html
[7] Learninginventions. "Hall Effect Sensor." [Online]. Available : http://mouse.learninginventions.org/?page_id=98