



รายงานโครงงาน

ภาคการศึกษาที่ 1 ปีการศึกษา 2566

รายวิชา 523480 โครงงานวิศวกรรมคอมพิวเตอร์

(Computer Engineering Project)

เรื่อง

แดชบอร์ดโรงงาน

Factory Dashboard

ผู้จัดทำ

นางสาวขวัญจิรา พันธเกตุ B6321451

อาจารย์ที่ปรึกษาโครงงาน

อาจารย์ ดร.วิชัย ศรีสุรักษ์

สาขาวิศวกรรมคอมพิวเตอร์

สำนักวิชาวิศวกรรมศาสตร์ มหาวิทยาลัยเทคโนโลยีสุรนารี

## กิตติกรรมประกาศ

โครงการแดชบอร์ดโรงงาน (Factory Dashboard) นี้สำเร็จลุล่วงได้ด้วยความกรุณาอย่างสูงจาก อาจารย์ ดร.วิชัย ศรีสุรักษ์ อาจารย์ที่ปรึกษาโครงการ ที่กรุณาให้คำแนะนำปรึกษา ช่วยจัดหาและอำนวยความสะดวกในการติดตั้งอุปกรณ์ ตลอดจนปรับปรุงแก้ไขข้อบกพร่องต่าง ๆ ด้วยความเอาใจใส่อย่างยิ่ง ผู้จัดทำตระหนักถึงความตั้งใจจริงและความทุ่มเทของอาจารย์ และขอกราบขอบพระคุณเป็นอย่างสูงไว้

ขอกราบขอบพระคุณ คุณพ่อ คุณแม่ และกัลยาณมิตรทุกคน ที่ให้คำปรึกษา เป็นกำลังใจ และสนับสนุนมีส่วนช่วยให้โครงการนี้สำเร็จลุล่วงด้วยดี จึงขอขอบคุณมาในโอกาสนี้ด้วย

ผู้จัดทำหวังว่าโครงการนี้จะเป็นประโยชน์อยู่ไม่น้อย จึงขอมอบส่วนดีทั้งหมดนี้ให้แก่ คณาจารย์ที่ได้ประสิทธิ์ประสาทวิชา ถ่ายทอดความรู้ให้ จนทำให้ผลงานเป็นประโยชน์ และขอมอบเป็นกตัญญูตเวทีแด่บิดามารดา รวมถึงผู้มีพระคุณทุกท่าน ทั้งนี้สำหรับข้อบกพร่องต่าง ๆ ที่อาจจะเกิดขึ้นนั้น ผู้จัดทำขอน้อมรับผิดเพียงผู้เดียว และยินดีที่จะรับคำแนะนำจากทุกท่านที่ได้เข้ามาศึกษาเพื่อเป็นประโยชน์ในการพัฒนาต่อไป

นางสาวขวัญจิรา พันธุ์เกตุ

ผู้จัดทำ

หัวข้อโครงการ : แดชบอร์ดโรงงาน  
 Factory Dashboard

ประเภท : การประยุกต์ใช้ฮาร์ดแวร์

ผู้เสนอโครงการ : นางสาวขวัญจิรา พันธุ์เกตุ รหัสนักศึกษา B6321451

อาจารย์ที่ปรึกษาโครงการ : อาจารย์ ดร.วิชัย ศรีสุรักษ์

ปีการศึกษา : 2566

### บทคัดย่อ

การจัดทำโครงการในครั้งนี้มีวัตถุประสงค์เพื่อศึกษา และออกแบบระบบ โดยนำแนวคิดเทคโนโลยี อินเทอร์เน็ตของสรรพสิ่ง (Internet of thing) หรือ IoT ประยุกต์เข้ากับ PLC (Programmable Logic Controller) เพื่อให้สามารถควบคุมการทำงาน ดูสถานะการทำงานของ Coil และดูข้อมูล Variable memory ใน PLC ผ่าน แพลตฟอร์ม IoT ได้

ผลการศึกษาและการจัดทำโครงการพบว่าเป็นไปตามจุดประสงค์ คือ ได้ศึกษาระบบการทำงานของ IoT ทดลองออกแบบระบบเพื่อให้เกิดความเข้าใจเบื้องต้น ให้สามารถนำไปพัฒนาต่อยอดในด้าน IIoT (Industrial Internet of Things) ได้ในอนาคต และสามารถควบคุมการทำงาน ดูสถานะการทำงานของ Coil ดูข้อมูล Variable memory ใน PLC ผ่าน แพลตฟอร์ม IoT ได้

## สารบัญ

กิตติกรรมประกาศ	ก
บทคัดย่อ	ข
สารบัญ	ค
บทที่ 1 บทนำ	1
ความเป็นมาและความสำคัญ	1
วัตถุประสงค์	1
ตัวชี้วัด และการบรรลุวัตถุประสงค์	1
ขอบเขตของโครงการ	2
ประโยชน์ที่คาดว่าจะได้รับ	2
แผนการดำเนินงาน	2
บทที่ 2 เอกสารและทฤษฎีที่เกี่ยวข้อง	3
IoT (Internet of Thing)	3
IIoT (Industrial Internet of Things)	4
MQTT	5
Modbus TCP/IP	7
Siemens LOGO!8	8
Raspberry Pi	10
Node-RED	11
Ubidots	12
EKI-2525-BE	13
หลักการทำงานในส่วนของการควบคุมมอเตอร์	13
หลักการทำงานของอุปกรณ์ที่เป็น Input และ Output	17
บทที่ 3 ขั้นตอนและวิธีการดำเนินการ	21
กล่าวนำ	21
อุปกรณ์ที่ใช้	21
อุปกรณ์ หรือ ฮาร์ดแวร์ ที่ใช้	21
การติดตั้ง ฮาร์ดแวร์	23

โปรแกรม ซอฟต์แวร์ และ แพลตฟอร์ม ที่ใช้	23
ขั้นตอนการดำเนินการ	24
ติดตั้ง LOGO!Soft Comfort V8.3	24
เขียนโปรแกรม Function Block	24
ติดตั้งระบบปฏิบัติการของ Raspberry Pi	30
เขียนโปรแกรม	31
Ubidosh	39
<b>บทที่ 4 ผลการดำเนินงาน</b>	40
ผลลัพธ์ของโครงการ	40
วิดีโอแสดงผลการดำเนินการ	42
สรุปและอภิปรายผลการทดลอง	43
การพัฒนาต่อยอด	43
<b>บทที่ 5 ปัญหา และข้อเสนอแนะ</b>	44
ปัญหาที่พบ	44
ข้อเสนอแนะ	44
<b>บรรณานุกรม</b>	45

## บทที่ 1

### บทนำ

#### 1.1 ความเป็นมาและความสำคัญ

เนื่องจากปัจจุบันแนวคิดอุตสาหกรรม 4.0 (Industry 4.0) มีอิทธิพลในการสร้างแรงกระตุ้นให้เกิดการเปลี่ยนแปลงด้านนวัตกรรม มีการบูรณาการกระบวนการผลิตเข้ากับเทคโนโลยีการเชื่อมต่อเครือข่ายอินเทอร์เน็ตในรูปแบบ Internet of Things หรือ IoT เป็นเทคโนโลยีที่ช่วยเชื่อมต่อการผลิตเข้ากับเครือข่ายอินเทอร์เน็ต ทำให้ทั้งซัพพลายเชนเชื่อมต่อกันบนโลกดิจิทัลได้แบบไร้รอยต่อ ซึ่งรู้จักกันในชื่อ Industrial Internet of Things (IIoT) คือการนำเครื่องจักร ระบบการวิเคราะห์ขั้นสูง และคนมาทำงานร่วมกันผ่านโครงข่ายของอุปกรณ์ที่เชื่อมต่อกันด้วยเทคโนโลยีการสื่อสาร ส่งผลให้เกิดระบบที่สามารถติดตาม เก็บข้อมูล แลกเปลี่ยนและแสดงผลข้อมูลเชิงลึกที่เป็นประโยชน์ได้ ช่วยเพิ่มประสิทธิภาพของกระบวนการต่างๆ และลดค่าใช้จ่าย

ผู้จัดทำจึงมีแนวคิดที่อยากศึกษา และออกแบบระบบ โดยนำแนวคิด IoT ประยุกต์เข้ากับ PLC (Programmable Logic Controller) ซึ่งเป็น Controller ที่ใช้ในงานอุตสาหกรรมและการควบคุมกระบวนการต่างๆ เช่นการควบคุมเครื่องจักรในโรงงาน, การควบคุมการผลิตในสายงาน, การควบคุมระบบการขนส่ง, และงานอุตสาหกรรมอื่นๆ ที่กระบวนการทำงานอยู่ในรูปแบบอัตโนมัติ เพื่อให้เกิดความเข้าใจและสามารถนำไปพัฒนาต่อยอดในด้าน IIoT ได้ในอนาคต

#### 1.2 วัตถุประสงค์

- 1.2.1. เพื่อศึกษาระบบการทำงานของ IoT ทดลองออกแบบระบบเพื่อให้เกิดความเข้าใจเบื้องต้น ให้สามารถนำไปพัฒนาต่อยอดในด้าน IIoT (Industrial Internet of Things) ได้ในอนาคต
- 1.2.2. เพื่อศึกษาระบบการทำงานของ PLC (Programmable Logic Controller)
- 1.2.3. เพื่อประยุกต์ PLC เข้ากับแนวคิดระบบ IoT ให้สามารถควบคุมการทำงาน ดูสถานะการทำงานของ Coil และดูข้อมูล Variable memory ใน PLC ผ่าน แพลตฟอร์ม IoT ได้

#### 1.3 ตัวชี้วัด และการบรรลุวัตถุประสงค์

- 1.3.1. สามารถออกแบบระบบ IoT ได้
- 1.3.2. สามารถควบคุมการทำงาน ดูสถานะการทำงานของ Coil ใน PLC ผ่าน แพลตฟอร์ม IoT ได้
- 1.3.3. สามารถดูข้อมูล Variable memory ใน PLC ผ่าน แพลตฟอร์ม IoT ได้



## บทที่ 2

### เอกสารและทฤษฎีที่เกี่ยวข้อง

โครงการนี้ได้ทำการศึกษาการทำงานขององค์ประกอบต่างๆ และออกแบบระบบ IoT เพื่อจัดทำแดชบอร์ดโรงงาน เพื่อให้เข้าใจพื้นฐานในการศึกษาและทฤษฎีที่เกี่ยวข้อง ประกอบด้วยรายละเอียด ดังนี้

#### 2.1. IoT (Internet of Thing)

Internet of Things (IoT) คือ "อินเทอร์เน็ตในทุกสิ่ง" หมายถึง การที่อุปกรณ์ต่างๆ สิ่งต่างๆ ได้ถูกเชื่อมโยงทุกสิ่งทุกอย่างสู่โลกอินเทอร์เน็ต ทำให้มนุษย์สามารถสั่งการควบคุมการใช้งานอุปกรณ์ต่างๆ ผ่านทางเครือข่ายอินเทอร์เน็ต เช่น การเปิด-ปิด อุปกรณ์เครื่องใช้ไฟฟ้า (การสั่งการเปิดไฟฟ้าภายในบ้านด้วยการเชื่อมต่ออุปกรณ์ควบคุม เช่น มือถือ ผ่านทางอินเทอร์เน็ต) รถยนต์ โทรศัพท์มือถือ เครื่องมือสื่อสาร เครื่องมือทางการเกษตร อาคาร บ้านเรือน เครื่องใช้ในชีวิตประจำวันต่างๆ ผ่านเครือข่ายอินเทอร์เน็ต เป็นต้น

IoT มีชื่อเรียกอีกอย่างว่า M2M ย่อมาจาก Machine to Machine คือเทคโนโลยีอินเทอร์เน็ตที่เชื่อมต่ออุปกรณ์กับเครื่องมือต่างๆ เข้าไว้ด้วยกัน

เทคโนโลยี IoT มีความจำเป็นต้องทำงานร่วมกับอุปกรณ์ประเภท RFID และ Sensors ซึ่งเปรียบเสมือนการเติมสมองให้กับอุปกรณ์ต่างๆ ที่ขาดไม่คือการเชื่อมต่ออินเทอร์เน็ต เพื่อให้อุปกรณ์สามารถรับส่งข้อมูลถึงกันได้ เทคโนโลยี IoT มีประโยชน์ในหลายด้าน แต่ก็มาพร้อมกับความเสี่ยง เพราะหากระบบรักษาความปลอดภัยของอุปกรณ์ และเครือข่ายอินเทอร์เน็ตไม่ดีพอ ก็อาจทำให้ผู้ไม่ประสงค์ดีเข้ามาขโมยข้อมูลหรือละเมิดความเป็นส่วนตัวของเราได้ ดังนั้นการพัฒนา IoT จึงจำเป็นต้องพัฒนามาตรการ และระบบรักษาความปลอดภัยให้ควบคู่กันไปด้วย

##### 2.1.1. แบ่งกลุ่ม Internet of Things

ปัจจุบันมีการแบ่งกลุ่ม Internet of Things ออกตามตลาดการใช้งานเป็น 2 กลุ่มได้แก่

###### 2.1.1.1. Industrial IoT

คือ แบ่งจาก local network ที่มีหลายเทคโนโลยีที่แตกต่างกันในโครงข่าย Sensor nodes โดยตัวอุปกรณ์ IoT Device ในกลุ่มนี้จะเชื่อมต่อแบบ IP network เพื่อเข้าสู่อินเทอร์เน็ต

###### 2.1.1.2. Commercial IoT

คือ แบ่งจาก local communication ที่เป็น Bluetooth หรือ Ethernet (wired or wireless) โดยตัวอุปกรณ์ IoT Device ในกลุ่มนี้จะสื่อสารภายในกลุ่ม Sensor nodes เดียวกันเท่านั้นหรือเป็นแบบ local devices เพียงอย่างเดียวอาจไม่ได้เชื่อมสู่อินเทอร์เน็ต



## 2.2. IIoT (Industrial Internet of Things)

IIoT หรือ Industrial IoT ย่อมาจาก Industrial Internet of Things คือ การเชื่อมต่อเครื่องจักรและระบบอุตสาหกรรมเข้ากับอินเทอร์เน็ตเพื่อที่จะนำเอาข้อมูลมา ฝ้าดู ประมวลผล และวิเคราะห์เพื่อที่จะเพิ่มประสิทธิภาพ, เพิ่มผลผลิตและลดค่าใช้จ่าย

Things ใน Industrial IoT เป็นเครื่องจักรที่มีองค์ประกอบเป็นเซ็นเซอร์, Actuator, มิเตอร์และคอนโทรลเลอร์ ซึ่งจุดประสงค์ของ IIoT คือ การเอาข้อมูลในกระบวนการผลิต (Utilize Manufacturing Data) มาใช้เพื่อที่จะปรับปรุงกระบวนการผลิตให้ดีขึ้น, มีประสิทธิภาพหรือผลผลิตที่เพิ่มขึ้นและประหยัดค่าใช้จ่ายนั่นเอง

### 2.2.1. โครงสร้างพื้นฐานของ IIoT (IIoT infrastructure)

#### 2.2.1.1. User (ผู้ใช้):

- Data Processing (การประมวลผลข้อมูล): ทำหน้าที่รวบรวมข้อมูลจากอุปกรณ์เซ็นเซอร์และระบบต่างๆ ในระบบ IIoT เพื่อนำมาวิเคราะห์และใช้ในการตัดสินใจ.
- Analytics (การวิเคราะห์): ทำหน้าที่วิเคราะห์ข้อมูลเพื่อนำเสนอข้อมูลที่มีประสิทธิภาพและแนะนำการตัดสินใจที่ถูกต้อง.
- Business Application Integration (การผสมรวมแอปพลิเคชันธุรกิจ): การผสมรวมข้อมูลและการทำงานของระบบ IIoT ในแอปพลิเคชันธุรกิจเพื่อให้ผู้ใช้สามารถใช้ข้อมูลนั้นในการดำเนินธุรกิจของตน.
- Automated Processes Database (ฐานข้อมูลของกระบวนการอัตโนมัติ): ฐานข้อมูลที่ใช้เก็บข้อมูลที่เกี่ยวข้องกับกระบวนการอัตโนมัติที่สามารถให้ข้อมูลเชิงลึกและความเป็นประโยชน์ในการบริหารจัดการ.

#### 2.2.1.2. Cloud (คลาวด์):

- IIoT Platform: เป็นแพลตฟอร์มที่ให้บริการแก่ระบบ IIoT ในรูปแบบคลาวด์ เพื่อการจัดการและประมวลผลข้อมูล.
- On-premises Server (เซิร์ฟเวอร์ภายในองค์กร): เซิร์ฟเวอร์ที่ติดตั้งและดำเนินการภายในองค์กรหรือธุรกิจเพื่อให้มีการควบคุมและการจัดเก็บข้อมูลตามนโยบายขององค์กร.

#### 2.2.1.3. Gateway (เกตเวย์):

- IIoT Gateway: เป็นอุปกรณ์ที่ใช้เป็นตัวกลางในการเชื่อมต่อระหว่างอุปกรณ์ในสิ่งของและระบบเครือข่าย IIoT กับระบบ Cloud หรือ On-premises Server.

- Edge Gateway: เป็นเกตเวย์ที่ติดตั้งในส่วนของ Edge ของระบบ IIoT เพื่อประมวลผลข้อมูล และทำงานใกล้กับแหล่งข้อมูล เพื่อลดการส่งข้อมูลไปยัง Cloud หรือ Server ศูนย์กลาง.

#### 2.2.1.4. Things (สิ่งของ):

- Sensors (เซ็นเซอร์): อุปกรณ์ที่ใช้เพื่อตรวจวัดหรือรับข้อมูลจากระบบและสิ่งแวดล้อม เช่น เซ็นเซอร์อุณหภูมิ, เซ็นเซอร์ความชื้น.
- Actuator (แอตนิวเมต): อุปกรณ์ที่ใช้ในการควบคุมการทำงานของระบบอัตโนมัติ เช่น มอเตอร์, วาล์ว.
- Meter (เครื่องวัด): อุปกรณ์ที่ใช้ในการวัดและบันทึกข้อมูล เช่น เครื่องวัดไฟฟ้า, เครื่องวัดน้ำ.
- Controller (ควบคุม): อุปกรณ์ที่ใช้ในการควบคุมการทำงานของระบบ IIoT, เช่น คอมพิวเตอร์ ตัดสินใจ.

#### 2.2.2. ประโยชน์ของ IIoT

IIoT ช่วยเพิ่มประสิทธิภาพการดำเนินงาน, เพิ่มผลผลิต, ลดของเสีย และลดต้นทุน ประโยชน์สูงสุดอย่างหนึ่งจาก IIoT ที่ใช้ในอุตสาหกรรมการผลิตคือช่วยให้สามารถบำรุงรักษาที่คาดการณ์ล่วงหน้า (Predictive maintenance) ได้ สามารถดูสถานะของเครื่องจักร (Monitor) แบบเรียลไทม์ (Real Time) ที่สร้างจากระบบ IIoT เพื่อคาดการณ์ว่าเมื่อไหร่ที่เครื่องจักรจะต้องได้รับการซ่อมบำรุง ด้วยวิธีนี้ การบำรุงรักษาที่จำเป็นสามารถทำได้ก่อนที่จะเกิดความเสียหาย เพื่อหลีกเลี่ยงการ Downtime ถือว่าเป็นประโยชน์อย่างมากในไลน์การผลิต ซึ่งความเสียหายของเครื่องจักรอาจส่งผลให้กระบวนการผลิตต้องหยุดชะงักและสูญเสียค่าใช้จ่ายสูง

### 2.3. MQTT

MQTT เป็น Protocol ที่ออกแบบมาเพื่อการเชื่อมต่อแบบ M2M (Machine-to-machine) คือ อุปกรณ์ติดต่อหรือสื่อสารกับอุปกรณ์ โดยเป็นส่วนหนึ่งของเทคโนโลยี IoT (Internet of Things) ซึ่งเป็นเทคโนโลยีที่อินเทอร์เน็ตเชื่อมต่อกับอุปกรณ์ต่างๆ

MQTT สำหรับระบบ IoT นั้น การติดต่อสู่ Internet นั้นเป็นสิ่งสำคัญอย่างยิ่ง เพราะอินเทอร์เน็ตทำให้ อุปกรณ์ IoT ต่างๆ สามารถติดต่อสื่อสารและแลกเปลี่ยนข้อมูลกันได้ MQTT (Message Queue Telemetry Transport) ซึ่งพัฒนาต่อมาจาก TCP/IP อีกทีนั้นได้กลายเป็น protocol มาตรฐานสำหรับระบบ IoT และเนื่องจากมันสร้างมาจาก TCP/IP นั้นทำให้ MQTT ประกันว่าข้อมูลที่ส่งกันระหว่างอุปกรณ์ IoT นั้นจะไม่มีภาระหน่วงระหว่างทาง

MQTT ใช้โมเดล publish/subscribe และออกแบบมาเพื่ออุปกรณ์ที่มีความเร็วในการรับและส่งข้อมูลต่ำ (low bandwidth) ซึ่งส่วนมากแล้วอุปกรณ์ของระบบ IoT จะเป็นแบบนั้น จุดประสงค์ของ MQTT ก็คือ เพื่อให้ระบบของเรานั้นมีการส่งหรือรับข้อมูลที่มีประสิทธิภาพมากขึ้น รวมทั้งทำให้อุปกรณ์นั้นใช้พลังงานน้อยลง ซึ่งในระบบ IoT เราต้องการส่งข้อมูลแบบ real-time และไม่ต้องการให้อุปกรณ์ของเราใช้พลังงานเยอะเกินไปโดยไม่จำเป็น ดังนั้น MQTT จึงเหมาะสมกับระบบเหล่านี้

### 2.3.1. TCP/IP (Transmission Control Protocol/Internet Protocol)

ระบบการสื่อสารผ่านอินเทอร์เน็ตระหว่าง 1 อุปกรณ์กับอีก 1 อุปกรณ์ โดยที่จะต้องทำการ connect กันก่อนถึงจะส่งหรือรับข้อมูลหากันได้ อย่างที่ได้บอกไปว่าการทำงานของ TCP/IP นั้นจะรับประกันว่าข้อมูลที่ส่งไปจะไม่สูญหายระหว่างทาง และรับประกันว่าผู้รับจะได้รับข้อมูลที่ส่งไปทั้งหมด (ไม่เหมือนกับ UDP ที่ส่งข้อมูลไปแล้วจะไม่สนใจว่าผู้รับจะได้รับข้อมูลครบหรือไม่) แต่อย่างที่กล่าวไปข้างต้น TCP/IP เป็นการสื่อสารแบบ point-to-point หน้าที่ของมันคือการที่ต้องรับประกันว่าข้อมูลนั้นได้ถูกส่งไปครบถ้วนเท่านั้น TCP/IP อาจจะใช้กับระบบ IoT ได้ แต่ลองคิดดูถ้าเรามีหลายๆอุปกรณ์อยู่ในระบบ แล้วต้องไปไล่ส่งข้อมูลอุปกรณ์ต่ออุปกรณ์ทีละคู่ๆนั้นทำให้เปลืองพลังงานมากแถมยังต้องมาคอย connect กันอีกทำให้ TCP/IP ไม่เหมาะสมกับระบบ IoT ใหญ่ ๆ สักเท่าไรหรอก ดังนั้นเราควรเลือกใช้ protocol ที่เหมาะสมกับระบบและความต้องการจะดีกว่า

### 2.3.2. MQTT(Publish/Subscribe)

MQTT จะมี Broker (Server) และ Clients (Publisher/Subscriber) เป็นหลัก เราจะเรียกการส่งข้อมูลใน MQTT ว่า publish และรับข้อมูลว่า subscribe

- Publish: คือการส่งข้อมูล เราจะต้องบอกด้วยว่าข้อมูลที่เราส่งไปนั้น เราจะส่งไปใน Topic ไหน
- Subscribe คือการรับข้อมูลแต่จะรับข้อมูลเฉพาะที่มาจาก Topic ที่เรา Subscribe เท่านั้น
- Topic คือหัวข้อที่เราสนใจ ซึ่งเอาไว้บ่งบอกว่า เราสนใจที่จะส่งข้อมูลไปยัง topic นี้หรือรอรับข้อมูลสำหรับ topics นี้อยู่ตลอด ตัวอย่าง เช่น home/office/temperature หรือ home/thermostat/temperature เป็นต้น
- Broker คือตัวกลางที่จะรับข้อมูลมาทั้งหมดมาจาก clients (publisher) ไม่ว่าจะเป็น topics อะไรก็ตาม แล้วทำการจัดการส่งข้อมูลไปยัง clients (subscriber) ที่ได้ทำการ subscribe สำหรับ topic ที่ได้รับข้อมูลมานั้นสามารถหา global broker หรือ cloud MQTT broker ได้ในหลาย ๆ เว็บไซต์ และสร้างภายใน network ได้โดยใช้ Mosquitto broker ซึ่งติดตั้งได้บน Raspberry Pi

## 2.4. Modbus TCP/IP



รูปที่ 1. Modbus TCP/IP

(ที่มา : <https://www.evereletronica.com/en/technologies/modbus-tcp-ip/>)

Modbus TCP/IP เป็นโพรโทคอลการสื่อสารที่ใช้ในการแลกเปลี่ยนข้อมูลและคำสั่งควบคุมระหว่างอุปกรณ์และระบบต่าง ๆ ในระบบอุตสาหกรรม โพรโทคอลนี้เป็นหนึ่งในคำสั่ง Modbus ที่ได้รับความนิยมมากที่สุดในปัจจุบัน

Modbus เป็นโพรโทคอลที่ถูกพัฒนาขึ้นในปี 1979 โดย Modicon (ที่ต่อมาเป็นส่วนหนึ่งของ Schneider Electric) เพื่อใช้ในการสื่อสารกับ PLC (Programmable Logic Controller) และอุปกรณ์ต่อพ่วงในระบบการควบคุมและจัดการการผลิต

Modbus TCP/IP นำโครงสร้างของ Modbus RTU ซึ่งใช้ RS-485 ในการสื่อสารมาพัฒนาเป็นระบบเครือข่าย TCP/IP ทำให้สามารถใช้งานผ่านเครือข่าย Ethernet หรืออินเทอร์เน็ตได้. โพรโทคอลนี้ใช้โพรโทคอลการสื่อสารแบบคำสั่ง/ตอบแบบ Master-Slave โดยทำงานผ่านพอร์ต TCP และเบื้องต้นใช้พอร์ต 502.

การใช้ Modbus TCP/IP ทำให้เป็นไปได้ในการควบคุมและโปรแกรมอุปกรณ์แบบต่าง ๆ ในระบบอุตสาหกรรมและโปรเจกต์อัตโนมัติต่าง ๆ อีกทั้งยังมีความยืดหยุ่นและเป็นทางเลือกที่สามารถพัฒนาและปรับแต่งตามความต้องการของแต่ละระบบได้.

Modbus TCP/IP (Modbus-TCP) คือ โพรโทคอล Modbus RTU ที่เชื่อมต่อกับ TCP โดยทำงานบนสถาปัตยกรรมของ Ethernet โครงสร้าง Message ของ Modbus คือ application protocol ที่จะถูกส่งผ่านไปพร้อมกับ TCP/IP (TCP/IP คือ Transmission Control Protocol และ Internet Protocol ซึ่งเป็นตัวกลางที่ใช้ในการส่ง Message ของ Modbus TCP/IP)

<u>Modbus RTU Data Type</u>	<u>Common name</u>	<u>Starting address</u>
Modbus Coils	Bits, binary values, flags	00001
Digital Inputs	Binary inputs	10001
Analog Inputs	Binary inputs	30001
Modbus Registers	Analog values, variables	40001

รูปที่ 2. ตารางข้างบนนี้แสดงชนิดของ Data และ Address ที่ใช้กับ Modbus RTU ซึ่งถูกใช้งานเช่นเดียวกันใน Modbus TCP/IP

(ที่มา : <https://sonicautomation.co.th/สื่อสารผ่าน-modbus-tcp-ip/>)

ตารางข้างล่างนี้แสดงชนิด Function และ Function code ของ Modbus RTU ที่ใช้สำหรับอ่านและเขียนข้อมูลซึ่งเราต้องใส่ไว้ใน Modbus RTU data frame

Function type		Function name	Function code
Bit access	Physical Discrete Inputs	Read Discrete Inputs	2
	Internal Bits or Physical Coils	Read Coils	1
		Write Single Coil	5
		Write Multiple Coils	15
16-bit access	Physical Input Registers	Read Input Registers	4
	Internal Registers or Physical Output Registers	Read Multiple Holding Registers	3
		Write Single Holding Register	6
		Write Multiple Holding Registers	16
		Read/Write Multiple Registers	23
		Mask Write Register	22
		Read FIFO Queue	24

รูปที่ 3. Function และ Function code ของ Modbus RTU  
(ที่มา : <https://sonicautomation.co.th/สื่อสารผ่าน-modbus-tcp-ip/>)

## 2.5. Siemens LOGO!8



รูปที่ 4. รูปตัวอย่าง Siemens LOGO!8  
(ที่มา : <https://shop.ibcon.com/th/product/722490/LOGO8>)

LOGO!8 คือ Smart miniPLC (Programmable Logic Controller) ที่พัฒนาโดยบริษัท Siemens AG ซึ่งเป็นระบบควบคุมและโปรแกรมอัตโนมัติที่ใช้ในงานอุตสาหกรรมขนาดเล็กถึงกลาง มีความสามารถในการควบคุมและตรวจสอบกระบวนการต่าง ๆ โดยมีลักษณะการทำงานที่ง่ายต่อการโปรแกรมและใช้งาน

LOGO!8 ถูกออกแบบมาเพื่อให้การควบคุมและตรวจสอบกระบวนการในอุตสาหกรรมขนาดเล็กเป็นไปอย่างมีประสิทธิภาพ มีลักษณะการทำงานที่น่าเข้าใจและใช้งานง่าย, ทำให้มันเป็นเครื่องมือที่เหมาะสมสำหรับนักวิศวกรและผู้ทำงานทางด้านอิเล็กทรอนิกส์

### 2.5.1.คุณสมบัติของ Siemens LOGO!8

- รองรับ Ethernet LAN สามารถเชื่อมต่อ Network ได้ง่าย
- รองรับ Modbus/TCP Protocol ตัวอุปกรณ์รองรับการเชื่อมต่อ Modbus/TCP Client สามารถเชื่อมต่ออุปกรณ์อื่นๆ หรือ Sensor ที่รองรับ Modbus/TCP ได้ง่าย เช่น Power Meter , Temperature Sensor และยังสามารถรองรับ Modbus/TCP Server สำหรับเปิด ช่องให้อุปกรณ์อื่นๆ อ่านค่าจาก LOGO!8 ได้ด้วย เช่น PLC หรือ SCADA Software
- สามารถใช้เป็น Remote I/O ได้ เนื่องจากอุปกรณ์รองรับ Modbus/TCP ทำให้เราสามารถนำมาใช้งานเป็น Remote I/O สำหรับอ่านข้อมูลและส่งต่อให้ระบบ PLC อื่นๆ ก็ได้เช่นกัน
- รองรับ Web Server ออกแบบหน้าเว็บได้
- รองรับการเขียนโปรแกรมภาษา Ladder และ Function Block เป็นภาษามาตรฐานสำหรับการเขียน PLC ชุดคำสั่งพื้นฐานมีมาครบ ไม่ว่าจะเป็น AND , OR , Counter , Timer , Delay On-Off , Weekly/Yearly Timer และอื่นๆ
- รองรับการแสดงผลสีหน้าจอได้ถึง 3 สี ขาว แดง ส้ม เหมาะสำหรับการเขียนโปรแกรมรูปแบบ แจ้งสถานะต่างๆตามที่เรากำหนด เช่น เมื่อเกิด Alarm ให้แสดงผลสีแดง , เมื่อเหตุการณ์ปกติให้แสดงผลสีขาว
- รองรับการแสดงค่าผ่านจอแสดงผล เราสามารถเขียนโปรแกรม ให้โปรแกรมแสดงค่าสัญญาณ ที่หน้าจอ ได้ทั้งรูปแบบ ค่าตัวเลข ค่ารูปแบบ Graph ช่วยให้ไม่ต้องติดตั้ง Display หรือ HMI เพิ่ม
- รองรับการแสดงผลโปรแกรมผ่านจอแสดงผล เราสามารถเข้าไปตรวจเช็คโปรแกรม หรือการตั้งค่าอื่นๆ เช่น IP Address ผ่านหน้าจออุปกรณ์ได้ทันที โดยไม่จำเป็นต้องใช้คอมพิวเตอร์
- รองรับการตั้งค่า Parameter ผ่านจอแสดงผล โดยไม่ต้องใช้คอมพิวเตอร์ สามารถแก้ไขค่า Parameter ที่ต้องการด้วยการกดปุ่มคำสั่ง บนอุปกรณ์
- รองรับการเชื่อมต่อระหว่าง LOGO!8 ได้สูงสุด 8 ตัว โปรแกรม LOGO! Soft Comfort สามารถทำโปรแกรมเพื่อคุยกันระหว่าง PLC ได้ง่าย
- รองรับ I/O Onboard ที่ 8 Digital Input , 4 Relay Output(10A) อุปกรณ์รองรับ I/O Onboard มากถึง 8 Digital Input รับสัญญาณรูปแบบ Wet Contact ง่ายต่อการ Wiring และ 4 Relay Output

- สามารถขยาย I/O เพิ่มได้สูงสุดถึง 24 Digital Input, 20 Relay Output สามารถต่อ Expansion Module ผ่าน Bus ด้านข้าง LOGO!8 ได้ สูงสุด 24 Digital Input , 20 Relay Output
- ลดจำนวนสายไฟ และอุปกรณ์ ช่วยให้งานระบบลดจำนวนอุปกรณ์และการ Wiring สายไฟ ด้วยจำนวน Amp ของ Relay ที่ให้มากมาถึง 10A ช่วยให้ไม่ต้องซื้อ Relay Module มาติดตั้งเพิ่ม และประหยัดพื้นที่
- รองรับ microSD Card สามารถทำ Application รูปแบบ Data log สำหรับเก็บข้อมูล รูปแบบ CSV file ลง MicroSD Card ได้

## 2.6. Raspberry Pi



รูปที่ 5. รูปตัวอย่าง Raspberry pi 3 b+

(ที่มา : <https://th.element14.com/raspberry-pi/rpi3-modbp/sbc-board-raspberry-pi-3-model/dp/2842228>)

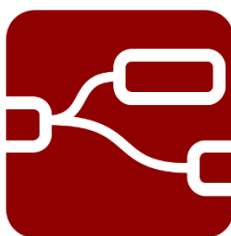
Raspberry Pi เป็นคอมพิวเตอร์ขนาดเล็กพัฒนาขึ้นโดยมูลนิธิ Raspberry Pi ซึ่งเป็นองค์กรการกุศลของสหราชอาณาจักร ที่ทำงานเพื่อนำพลังด้านดิจิทัลเข้าสู่ผู้ใช้งานทั่วโลก ดังนั้นผู้ใช้งานสามารถทำความเข้าใจและสร้างโลกดิจิทัลเพิ่มขึ้นได้โดยง่าย สามารถแก้ปัญหาต่าง ๆ ที่สำคัญได้และเตรียมพร้อมสำหรับงานในอนาคต ซึ่ง Raspberry Pi เป็นคอมพิวเตอร์ที่มีประสิทธิภาพสูง ราคาประหยัด และมีประสิทธิภาพสูงที่ผู้คนใช้เพื่อเรียนรู้ในการแก้ปัญหาและได้รับความสนุกสนาน อีกทั้งมีชุมชนออนไลน์พัฒนาแหล่งข้อมูลฟรี เช่น บทความ, ตัวอย่างโครงการ เพื่อช่วยให้ผู้คนเรียนรู้เกี่ยวกับคอมพิวเตอร์และวิธีการทำสิ่งต่าง ๆ กับคอมพิวเตอร์ ไม่ว่าจะเป็นใช้งานด้านทั่วไป หรือ ทักษะการเขียนโปรแกรม ซึ่งสามารถประหยัดค่าใช้จ่ายในการเรียนรู้โดยเฉพาะการเขียนโปรแกรม

### 2.6.1. คุณสมบัติของ Raspberry Pi (Raspberry Pi3 Model B)

Raspberry Pi สามารถเชื่อมต่อระบบเครือข่ายแบบใช้สายหรือไร้สายได้ ทำให้กลายเป็นอุปกรณ์ Internet of Things โดยสมบูรณ์ ช่วยให้นักวิจัยและผู้สนใจอื่น ๆ สามารถนำไปประยุกต์ใช้เพื่อเชื่อมต่อกับตัวตรวจจับ (Sensor) ในการเก็บข้อมูลตามต้องการ รวมถึงสามารถเชื่อมต่อกับแป้นพิมพ์และเมาส์ได้

ง่ายอีกด้วย โดยระบบปฏิบัติการที่ใช้นั้น คือ Raspbian ซึ่งเป็นระบบปฏิบัติการลินุกซ์เป็นฐานถูกปรับแต่งมาใช้กับ Raspberry Pi โดยเฉพาะ และระบบปฏิบัติการ ติดตั้งผ่าน Micro SD Card สามารถตั้งค่าเป็นเครื่องแม่ข่ายและใช้งานบริการต่าง ๆ เช่น Web Server, FTP Server ได้ เป็นต้น

## 2.7. Node-RED



รูปที่ 6. Node-RED Logo

(ที่มา : <https://nodered.org/about/resources/>)

Node-RED เป็นแพลตฟอร์มการโปรแกรมและการสร้างกราฟิกเพื่อการสื่อสารข้อมูลแบบสตรีมสายทาง (flow-based programming) ซึ่งมีการพัฒนาโดย IBM และถูกเปิดต้นฉบับ (open source) ซึ่งใช้งานได้ฟรี โดยทั่วไปมักนิยมใช้ Node-RED ในการสร้างและจัดการกระบวนการอัตโนมัติ (automation) และการประมวลผลข้อมูลในรูปแบบของกราฟิกแบบโนด (nodes) ซึ่งทำให้ง่ายต่อการเชื่อมต่อและปรับแต่งการทำงานต่าง ๆ โดยไม่ต้องเขียนโปรแกรมให้เป็นเส้นตรง

Node-RED มีความยืดหยุ่นและความสามารถในการเชื่อมต่อกับอุปกรณ์หลากหลายชนิด รวมถึงอุปกรณ์ IoT, เซนเซอร์, กล้อง, การเชื่อมต่อแบบ RESTful API, ฐานข้อมูล, และบริการอื่น ๆ ทำให้สามารถนำมาประยุกต์ใช้ในโครงการที่หลากหลายตามความต้องการของผู้ใช้

โดยโนดใน Node-RED มีหลายประเภทที่ใช้งานต่าง ๆ ได้ เช่น:

- Input Nodes: ใช้ในการรับข้อมูลเข้าสู่ระบบ เช่น การรับข้อมูลจากเซนเซอร์หรือเว็บเซอร์วิส
- Processing Nodes: ใช้ในการประมวลผลข้อมูล เช่น การคำนวณหรือการแปลงข้อมูล
- Output Nodes: ใช้ในการส่งข้อมูลออกจากระบบ เช่น การควบคุมอุปกรณ์ IoT หรือการโพสต์ข้อมูลไปยังเว็บเซอร์วิส

Node-RED มีทางเลือกในการเชื่อมต่อกับหลายพลังงานและแพลตฟอร์มได้, ทำให้มีการใช้งานกว้างขวางในหลากหลายสาขาอุตสาหกรรมและโครงการต่าง ๆ เช่น IoT, การอัตโนมัติ, ระบบควบคุมและตรวจสอบ



## 2.8. Ubidots



รูปที่ 7. Ubidots Logo  
(ที่มา : <https://ubidots.com/>)

Ubidots เป็นแพลตฟอร์ม IoT (Internet of Things) ที่เชื่อมต่ออุปกรณ์ IoT และเซนเซอร์ต่าง ๆ เข้ากับโลกของข้อมูลและการวิเคราะห์ผ่านเว็บแอปพลิเคชัน โดยทำให้ผู้ใช้สามารถเก็บข้อมูลจากอุปกรณ์ IoT, ทำการวิเคราะห์ข้อมูลเหล่านั้น และสร้างแผนภูมิและสถิติเพื่อเข้าใจข้อมูลได้ง่ายขึ้น.

นับเป็นหนึ่งในแพลตฟอร์ม IoT ที่ได้รับความนิยมในการบริหารจัดการข้อมูลที่เกี่ยวข้องกับอุปกรณ์ IoT และการควบคุมอุปกรณ์ได้อย่างมีประสิทธิภาพ โดยเน้นที่การให้บริการในลักษณะของ IoT Application Enablement Platform (AEP) ซึ่งช่วยให้ผู้ใช้สามารถสร้างและบริหารจัดการแอปพลิเคชัน IoT ได้อย่างมีประสิทธิภาพ

### 2.8.1. คุณสมบัติและบริการที่ Ubidots

1. การเชื่อมต่อและการรับข้อมูล (Connectivity and Data Ingestion): ทำให้สามารถเชื่อมต่อกับอุปกรณ์ IoT และเซนเซอร์ต่าง ๆ ผ่านโปรโตคอลต่าง ๆ เพื่อรับข้อมูลเข้าสู่แพลตฟอร์ม
2. การจัดเก็บและจัดการข้อมูล (Data Storage and Management): บริการจัดเก็บข้อมูลอย่างปลอดภัยในรูปแบบที่ง่ายต่อการเข้าถึง และสามารถจัดการข้อมูลได้ตามความต้องการ
3. การวิเคราะห์ข้อมูล (Data Analytics): สามารถวิเคราะห์ข้อมูลที่ถูกเก็บไว้ เพื่อนำเสนอข้อมูลที่มีประสิทธิภาพ และช่วยในการตัดสินใจ
4. การแสดงผลข้อมูล (Data Visualization): สร้างแผนภูมิและรายงานเพื่อให้ผู้ใช้สามารถเข้าใจข้อมูลที่ได้รับในลักษณะที่สวยงามและชัดเจน
5. การควบคุมและการแจ้งเตือน (Control and Notifications): ช่วยในการควบคุมอุปกรณ์ IoT และส่งการแจ้งเตือนถึงผู้ใช้เมื่อเกิดเหตุการณ์ที่สำคัญ

Ubidots เป็นเครื่องมือที่น่าสนใจสำหรับนักพัฒนาระบบ IoT และผู้ที่ต้องการสร้างแอปพลิเคชัน IoT โดยมีการจัดทำอย่างครบวงจรเพื่อการบริหารจัดการข้อมูลและอุปกรณ์ IoT ได้ง่ายและมีประสิทธิภาพ.

## 2.9. EKI-2525-BE



รูปที่ 8. ADVANTECH EKI-2525-BE.

(ที่มา : <https://th.element14.com/advantech/eki-2525-be/enet-module-din-rail-wall-rj45/dp/2822019> )

ADVANTECH EKI-2525-BE เป็น Industrial Ethernet Switch ที่ผลิตโดย Advantech Corporation ซึ่งเป็นบริษัทชั้นนำทางด้านโซลูชันและบริการด้านอุตสาหกรรม IoT (Internet of Things) และอุตสาหกรรมดิจิทัลอื่น ๆ อุปกรณ์ EKI-2525-BE นี้เป็นส่วนหนึ่งของคอลเลกชันของอุปกรณ์เครือข่าย (networking devices) ที่ออกแบบมาเพื่อใช้ในสภาพแวดล้อมอุตสาหกรรม (Industrial Environment) เพื่อให้การเชื่อมต่อเครือข่ายเป็นไปอย่างเสถียรและมีประสิทธิภาพสูง

อุปกรณ์ EKI-2525-BE นี้มีไว้สำหรับการใช้งานในสภาพแวดล้อมอุตสาหกรรมที่ต้องการเครือข่ายเสถียรสูง และการรับส่งข้อมูลที่เร็ว โดยตัวอุปกรณ์นี้ช่วยให้สามารถจัดการและควบคุมการเชื่อมต่อระหว่างอุปกรณ์ต่าง ๆ ในระบบได้อย่างมีประสิทธิภาพและปลอดภัย

## 2.10. หลักการทำงานในส่วนของการควบคุมมอเตอร์

โครงการนี้ผู้จัดทำได้มีการจำลองการทำงานในอุตสาหกรรม โดยใช้มอเตอร์ที่หมุนแทนกระบวนการทำงานของสายการผลิต และติดตั้งที่เป็นโลหะไว้ที่ปลายมอเตอร์ เพื่อให้ Inductive Proximity Sensor ตรวจจับได้ เสมือนเป็นการนับสินค้าที่ไหลผ่านสายพานการผลิต

### 2.10.1. Arduino Micro



รูปที่ 9. Arduino Micro

(ที่มา : <https://www.arduitronics.com/product/341/arduino-micro-free-cable>)

Arduino Micro เป็นบอร์ดไมโครคอนโทรลเลอร์ที่ออกแบบมาเพื่อให้นักพัฒนาสามารถสร้างโปรเจกต์ที่ใช้ไมโครคอนโทรลเลอร์ (microcontroller) ได้ง่ายๆ โดยมีความยืดหยุ่นและสะดวกในการใช้งาน ซึ่งถูกพัฒนาโดยบริษัท Arduino

Arduino Micro มีการเชื่อมต่อหลายแบบเพื่อให้สามารถต่อกับอุปกรณ์ต่างๆ ได้ รวมถึงพอร์ต ดิจิตอล (digital ports), พอร์ตอนาล็อก (analog ports), พอร์ต PWM (Pulse Width Modulation), พอร์ต I2C, พอร์ต SPI, และพอร์ต UART ซึ่งทำให้สามารถเชื่อมต่อกับเซนเซอร์หลายชนิดและอุปกรณ์ อื่นๆ ได้

ผู้ทำได้ใช้ Arduino Micro เพื่อควบคุมการหมุนของมอเตอร์ในทิศทางและความเร็วที่ต้องการ โดยใช้ขา ENABLE, PULSE, และ DIRECTION บน TB6600 และรับค่าความเร็วจากเซนเซอร์ที่เชื่อมต่อกับขา A0 ที่เป็นขออนาล็อก

<pre>// ตั้งค่าขาที่ใช้ควบคุม tb6600 kkkkkk const int EN_PIN = 9; // ขา ENABLE const int PUL_PIN = 10; // ขา PULSE const int DIR_PIN = 11; // ขา DIRECTION // ตั้งค่าความเร็วของมอเตอร์ int motorSpeed = 0; // ความเร็วในรอบต่อนาที (RPM)  // ฟังก์ชันสำหรับเคลื่อนที่มอเตอร์ void moveMotor(bool direction, int steps) {     digitalWrite(DIR_PIN, direction); // ตั้งค่าทิศทางการเคลื่อนที่     for (int i = 0; i &lt; steps; i++)     {         digitalWrite(PUL_PIN, HIGH);         delayMicroseconds(motorSpeed / 2);         digitalWrite(PUL_PIN, LOW);         delayMicroseconds(motorSpeed / 2);     } }  void setup() {     pinMode(EN_PIN, OUTPUT);     pinMode(PUL_PIN, OUTPUT);     pinMode(DIR_PIN, OUTPUT);      // ตั้งค่าเริ่มต้นให้ระบบไม่ทำงาน     digitalWrite(EN_PIN, LOW); // ปลดล็อก tb6600      // เคลื่อนที่มอเตอร์ไปทางซ้าย 10 ก้าน     moveMotor(LOW, 10);     delay(500); }  int ledPin = 13; int AnalogPin = A0; unsigned long prevTime = 0; unsigned long interval = 1000; // เวลาในการหมุนทีละ 1 วินาที (1000 มิลลิวินาที) int oldSensorValue = 0;</pre>	<pre>void loop() {     // อ่านค่าที่รับมาจาก AnalogPin     int sensorValue = analogRead(AnalogPin);      // เมื่อค่าที่รับมาจาก AnalogPin เปลี่ยนแปลงไป     if (sensorValue != oldSensorValue)     {         // อัปเดตค่า oldSensorValue เป็นค่าใหม่         oldSensorValue = sensorValue;          // คำนวณความเร็วใหม่ของมอเตอร์ด้วยค่าที่รับมาจาก AnalogPin เป็นความเร็วของมอเตอร์ใน         หน่วย RPM (รอบต่อนาที).         motorSpeed = map(sensorValue, 0, 1023, 0, 2000); // ความเร็วใหม่จะอยู่ในช่วง 0-2000         RPM          // กำหนดความเร็วใหม่ให้กับมอเตอร์ (ใช้ขา PUL_PIN)         analogWrite(PUL_PIN, motorSpeed);          // เมื่อมีการเปลี่ยนค่าความเร็วใหม่ เช็สถานะการทำงาน         if (motorSpeed &gt; 1300) {             // เปิดการส่งกระแสไปยังขา EN_PIN เมื่อมีการทำงาน             digitalWrite(EN_PIN, LOW);         } else {             // ปิดการส่งกระแสไปยังขา EN_PIN เมื่อไม่มีการทำงาน             digitalWrite(EN_PIN, HIGH);         }     }      // หมุนเป็นวงกลมทางซ้าย     for (int i = 0; i &lt; 10; i++) // 10 ก้าน     {         moveMotor(LOW, 1); // หมุนทางซ้าย 1 ก้าน         delay(2); // ความหน่วงเวลาในการหมุน (ปรับค่าตามความเร็วที่ต้องการ)     } }</pre>
---	--

## 2.10.2. TB6600 โมดูลขับ สเต็ปป์มอเตอร์



รูปที่ 10. TB6600 โมดูลขับ สเต็ปป์มอเตอร์  
(ที่มา : <https://www.mikroelec.com/product/1365/tb6600>)

TB6600 เป็นโมดูลขับสเต็ปป์มอเตอร์ (Stepper Motor Driver Module) ที่ใช้ในการควบคุมการเคลื่อนที่ของมอเตอร์สเต็ปป์ (stepper motor) ซึ่งเป็นประเภทของมอเตอร์ที่เคลื่อนที่ทีละขั้น (step) ตามคำสั่งที่ระบุได้

โมดูลขับ TB6600 มีความสามารถในการควบคุมการหมุนของมอเตอร์สเต็ปป์ให้เป็นละเอียดและแม่นยำ โดยสามารถกำหนดจำนวนขั้นตอนที่มอเตอร์จะหมุนได้ รวมถึงความเร็วในการหมุนของมอเตอร์

### 2.10.2.1 ส่วนประกอบหลักและคุณสมบัติของโมดูล TB6600 ประกอบด้วย:

#### 1. Input Interface (ขาเข้าข้อมูล):

PUL (Pulse): ขานี้ใช้รับสัญญาณ Pulse เพื่อกำหนดการหมุนของมอเตอร์

DIR (Direction): ขานี้ใช้เลือกทิศทางการหมุนของมอเตอร์ (เข้าหรือถอยหลัง)

#### 2. Output Interface (ขาออกข้อมูล):

A+, A-, B+, B- : ขานี้เป็นขาส่งการควบคุมการหมุนของมอเตอร์สเต็ปป์

#### 3. Microstep Resolution (การตั้งค่าขนาดของขั้นตอน):

สามารถตั้งค่าขนาดขั้นตอน (microsteps) ได้ตามที่ต้องการ เพื่อให้มอเตอร์หมุนได้อย่างละเอียด

#### 4. Current Adjustment (การปรับกระแส):

สามารถปรับการตั้งค่ากระแสของมอเตอร์ได้ ซึ่งจะมีผลต่อการทนทานและประสิทธิภาพของมอเตอร์

### 5. Protection Features (คุณสมบัติการป้องกัน):

มีคุณสมบัติการป้องกันต่างๆ เช่น การป้องกันการย้อนกลับ (reverse polarity), การป้องกันไฟตก (voltage drop protection) เป็นต้น.

TB6600 สามารถนำมาใช้ในโปรเจกต์ที่ต้องการควบคุมการหมุนของมอเตอร์สเต็ปป์ให้เป็นละเอียด แม่นยำ และมีความเร็วตามที่ต้องการ เช่น การใช้ในเครื่องจักรอุตสาหกรรม, เครื่อง CNC (Computer Numerical Control), หุ่นยนต์, 3D Printer, และโปรเจกต์ที่ต้องการการควบคุมการเคลื่อนที่อย่างแม่นยำ.

### 2.10.3. Stepper Motor



รูปที่ 11. TB6600 โมดูลขับ สเต็ปมอเตอร์

(ที่มา : <https://www.mikroelec.com/product/1365/tb6600>)

Stepper Motor (มอเตอร์สเต็ปป์) เป็นประเภทหนึ่งของมอเตอร์ไฟฟ้าที่มีการทำงานตามหลักการขั้นหรือขั้นที่กำหนดล่วงหน้า เป็นการเคลื่อนที่ทีละขั้น (step-wise motion) โดยมีการควบคุมการหมุนโดยทำให้มอเตอร์หมุนทีละขั้น ตามตำแหน่งที่กำหนดไว้ล่วงหน้า.

หลักการทำงานของ Stepper Motor คือการสลับการกระทำกระตุ้นทางไฟฟ้าในแต่ละเฟส (phase) ของมอเตอร์ ซึ่งทำให้มอเตอร์หมุนตามขั้นที่กำหนด ตามตำแหน่งขั้นที่ถูกเรียกใช้. การหมุนทีละขั้นทำให้ง่ายต่อการควบคุมและจำกัดความเร็วและตำแหน่งของมอเตอร์ได้อย่างแม่นยำ.

#### 2.10.3.1. ประโยชน์และการนำไปใช้ของ Stepper Motor

1. การควบคุมตำแหน่งแม่นยำ: Stepper Motor สามารถควบคุมตำแหน่งของหน้าที่หลายๆ อย่างอย่างแม่นยำ เช่น การเคลื่อนที่ในระบบ CNC, 3D Printer, หุ่นยนต์, และเครื่องจักรอุตสาหกรรม.
2. ควบคุมความเร็ว: สามารถควบคุมความเร็วของมอเตอร์ได้ตามต้องการ ทำให้เหมาะสำหรับการใช้ในการควบคุมความเร็วในระบบที่ต้องการการปรับแต่งเร็วหรือช้า.
3. การทำงานในสถานะที่ต้องการการควบคุมแม่นยำ: สามารถนำไปใช้ในระบบที่ต้องการควบคุมทิศทางหรือการหมุนแบบแม่นยำ เช่น เคลื่อนที่ตามเส้นทางที่กำหนด.

4. ต้นทุนที่ถูก: Stepper Motor มีต้นทุนที่ถูกเมื่อเทียบกับมอเตอร์ DC หรือมอเตอร์เซอร์โว.
5. ง่ายต่อการใช้งาน: ง่ายต่อการใช้งานและการเขียนโปรแกรมควบคุม เพราะสามารถควบคุมได้ตรงเท่าที่ต้องการตามขั้นที่กำหนด.

โดยสรุป Stepper Motor มีความสามารถในการควบคุมตำแหน่งและทิศทางการหมุนได้อย่างแม่นยำ ซึ่งทำให้มีการนำไปใช้ในหลากหลายแอปพลิเคชันที่ต้องการการควบคุมทิศทาง และตำแหน่งการเคลื่อนที่ที่แม่นยำ และสามารถควบคุมได้ในทุกขณะ

## 2.11. หลักการทำงานของอุปกรณ์ที่เป็น Input และ Output

โครงงานนี้มีการใช้ความรู้เรื่องการเชื่อมต่อและใช้งานอุปกรณ์ในส่วนของการรับข้อมูล (input) และส่วนการส่งข้อมูล (output) ซึ่งประกอบไปด้วย Push Button Switch, LED Indicator Lamp, และ Inductive Proximity Sensor ที่เป็นส่วนสำคัญในการควบคุมและแสดงสถานะของระบบในโครงงานนี้

### 2.11.1. Push Button Switch



รูปที่ 12. 22MM Push Button Switch  
(ที่มา : <https://shorturl.asia/xk5s4>)

Push Button Switch (สวิตช์ปุ่มกด) เป็นอุปกรณ์สวิตช์ที่ใช้ในการเปิด-ปิดวงจรไฟฟ้า เมื่อมีการกด (press) หรือปล่อยปุ่ม (release) ตามตำแหน่งของปุ่มนั้นๆ โดยมีการออกแบบให้มีคุณสมบัติเป็น Normally Open (NO) และ Normally Closed (NC) ซึ่งเป็นการกำหนดสถานะของสวิตช์ก่อนที่จะมีการกดปุ่ม.

#### 1. Normally Open (NO):

- เมื่อปุ่มยังไม่ถูกกด (release), วงจรไม่ทำงาน (off).
- เมื่อกดปุ่ม (press), วงจรทำงาน (on).

#### 2. Normally Closed (NC):

- เมื่อปุ่มยังไม่ถูกกด (release), วงจรทำงาน (on).
- เมื่อกดปุ่ม (press), วงจรไม่ทำงาน (off).

### 2.11.2. LED Indicator Lamp



รูปที่ 13. LED Indicator Lamp

(ที่มา : <https://www.indiamart.com/proddetail/led-indicator-lamp-20907746612.html>)

LED Indicator Lamp (หลอดไฟสัญญาณแสดงผล) ใช้เป็นตัวแสดงผลสถานะหรือสัญญาณต่างๆ ในระบบอิเล็กทรอนิกส์ หลักการทำงานของ LED Indicator Lamp คือการแปลงพลังงานไฟฟ้าเป็นแสงที่สามารถมองเห็นได้ โดยใช้หลักการของไฟฟ้าทฤษฎีไฟฟ้า (electroluminescence) ซึ่งเกิดจากการหลอ่เลี้ยววัสดุพิเศษซึ่งเป็นสารประกอบทางอิเล็กทรอนิกส์ในโครงสร้างของ LED.

ขั้วของ LED ประกอบด้วยขั้วยอด (anode) และขั้วเทียม (cathode) โดยที่ขั้วยอดจะมีสีทองแดงและขั้วเทียมจะมีสีดำหรือสีเงิน การทำงานของ LED Indicator Lamp ได้แบ่งออกเป็นขั้วทำงานแบบ Forward Bias (FB) และ Reverse Bias (RB) ดังนี้:

#### 1. Forward Bias (FB):

- เมื่อกระแสไฟฟ้าไหลผ่านขั้วยอด (anode) และขั้วเทียม (cathode) พาร์ทสีในโครงสร้าง LED จะหลอ่เลี้ยวและปล่อยพลังงานในรูปของแสง.
- พลังงานนี้จะถูกแปลงเป็นแสงที่สามารถมองเห็นได้ตามสีของ LED.

#### 2. Reverse Bias (RB):

- เมื่อไม่มีกระแสไฟฟ้าไหลผ่านขั้วยอด (anode) และขั้วเทียม (cathode) โดยไม่มีการเปิดใช้งาน LED.

เมื่อไฟฟ้าไหลผ่านในทิศทางที่ถูกต้อง (Forward Bias), ไฟ LED จะติดและแสดงสีที่เป็นลักษณะของ LED นั้นๆ ตามการออกแบบของ LED ต่างๆ ที่มีสีและลักษณะการแสดงผลต่างกัน. LED Indicator Lamp จึงเป็นอุปกรณ์ที่มีความสำคัญในการแสดงสถานะและการทำงานของระบบต่างๆ ในการประยุกต์ใช้งานทั้งในงานอุตสาหกรรมและโปรเจกต์อิเล็กทรอนิกส์.

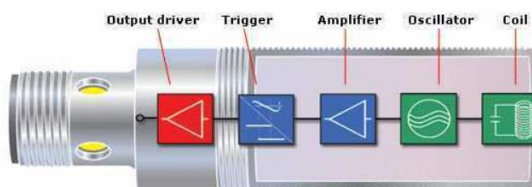
### 2.11.3. Inductive Proximity Sensor



รูปที่ 14. Inductive Proximity Sensor

(ที่มา : <https://www.jaaris.com/product-page/inductive-proximity-sensor>)

Inductive Proximity Sensor จะใช้หลักการเหนี่ยวนำของสนามแม่เหล็กไฟฟ้าในการทำงาน โดยที่มาของสนามแม่เหล็กไฟฟ้านั้น เกิดจากบริเวณส่วนหัวของเซ็นเซอร์ ซึ่งภายในจะมีขดลวด (Coil) ที่คอยทำหน้าที่ปล่อยสนามแม่เหล็กไฟฟ้าความถี่สูงซึ่งขดลวดนั้นจะได้รับสัญญาณไฟฟ้าจากวงจรกำเนิดความถี่ (Oscillator) เพื่อคอยตรวจจับโลหะที่เคลื่อนที่ผ่านเข้ามา และเมื่อชิ้นงานอยู่ในระยะที่เซ็นเซอร์สามารถตรวจจับได้ จะทำให้เกิดการเปลี่ยนแปลงค่าความเหนี่ยวนำ ซึ่งจะทำให้เกิดการหน่วงออสซิลเลท Oscillate หรือ ในบางครั้งอาจถึงจุดการหยุดออสซิลเลท ในขณะที่เกิดการหน่วงหรือการหยุดออสซิลเลทนั้น วงจรขยาย (Amplifier) จะทำหน้าที่ขยายสัญญาณเพื่อส่งต่อไปยัง วงจรทริกเกอร์ (Trigger) ซึ่งวงจรนี้จะมีหน้าที่เปลี่ยนแปลงสถานะของวงจร Output ว่าให้มีการทำงานหรือหยุดการทำงาน



รูปที่ 15. Inductive Proximity Sensor

(ที่มา : <https://mall.factormart.com/inductive-proximity-sensor-working-principle/>)

#### 2.11.3.1. ส่วนประกอบของ Inductive Proximity Sensor

- คอยล์ (Coil): Coil คือ ขดลวดที่มีหน้าที่ปล่อยสนามแม่เหล็กไฟฟ้าความถี่สูง ผ่านบริเวณด้านหน้าของ Inductive Proximity Sensor เพื่อคอยตรวจจับโลหะที่เคลื่อนที่ผ่านเข้ามา



- ออสซิลเลเตอร์ (Oscillator): Oscillator คือ วงจรกำหนดความถี่ จุดส่งสัญญาณไฟฟ้าไปยังส่วน Coil โดยความถี่นี้มีความจำเป็นมากต่อกระบวนการสร้างสนามแม่เหล็กไฟฟ้า
- แอมพลิฟายเออร์ (Amplifier): Amplifier คือ วงจรขยาย ในส่วนนี้จะทำหน้าที่ขยายสัญญาณ ที่เกิดจากการเปลี่ยนแปลงค่าความเหนี่ยวนำที่เกิดขึ้น
- ทริกเกอร์ (Trigger): Trigger หรือ วงจรทริกเกอร์นั้นจะทำหน้าที่ประมวลค่าความเปลี่ยนแปลงของสนามแม่เหล็กไฟฟ้าว่าอยู่ในเกณฑ์ใดต่อจากนั้นก็ส่งการไปยังภาค output ให้ทำการเปลี่ยนแปลงสถานะ
- เอาท์พุท ไดรเวอร์ (Output driver): Output driver เป็นภาคสุดท้ายของวงจร ซึ่งมีส่วนสำคัญในการสร้างสัญญาณเอาท์พุท ให้ได้ตามระดับมาตรฐานที่สามารถใช้งานกับตัวอุปกรณ์ที่มาเชื่อมต่อได้ เช่น คอนโทรลเลอร์หรือรีเลย์ เป็นต้น

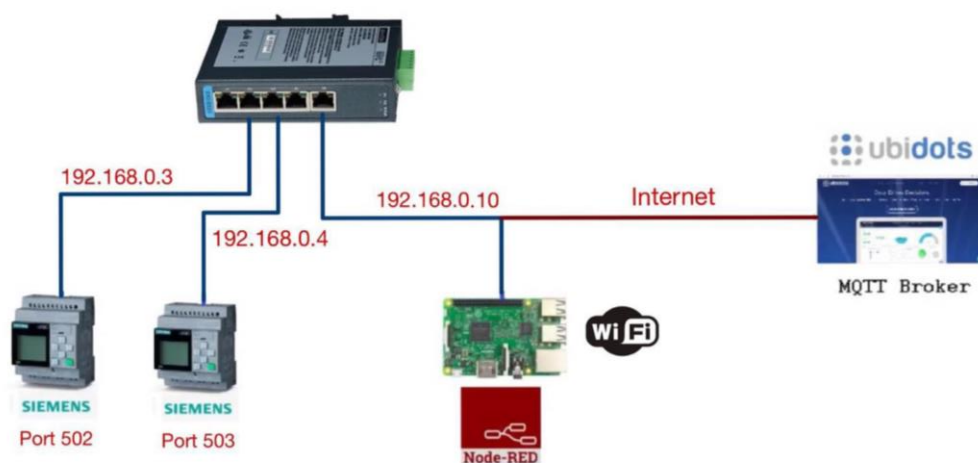
## บทที่ 3

### ขั้นตอนและวิธีการดำเนินการ

#### 3.1. กล่าวนำ

ในขั้นตอนและวิธีการดำเนินการ โครงการแดชบอร์ดโรงงาน(Factory Dashboard) จะประกอบไปด้วย ขั้นตอนหลักๆทั้งหมด 7 ขั้นตอน ดังนี้

##### 3.1.1. ออกแบบระบบ



3.1.2. เขียนโปรแกรม Function Block ให้กับ Siemens LOGO!8

3.1.3. ทำการ Add Server connection เพื่อใช้งาน ModBus Connection

3.1.4. ติดตั้งระบบปฏิบัติการของ Raspberry Pi

3.1.5. ติดตั้ง Node-RED บน Raspberry Pi

3.1.6. เขียน Flow-based programming ใน Node-RED

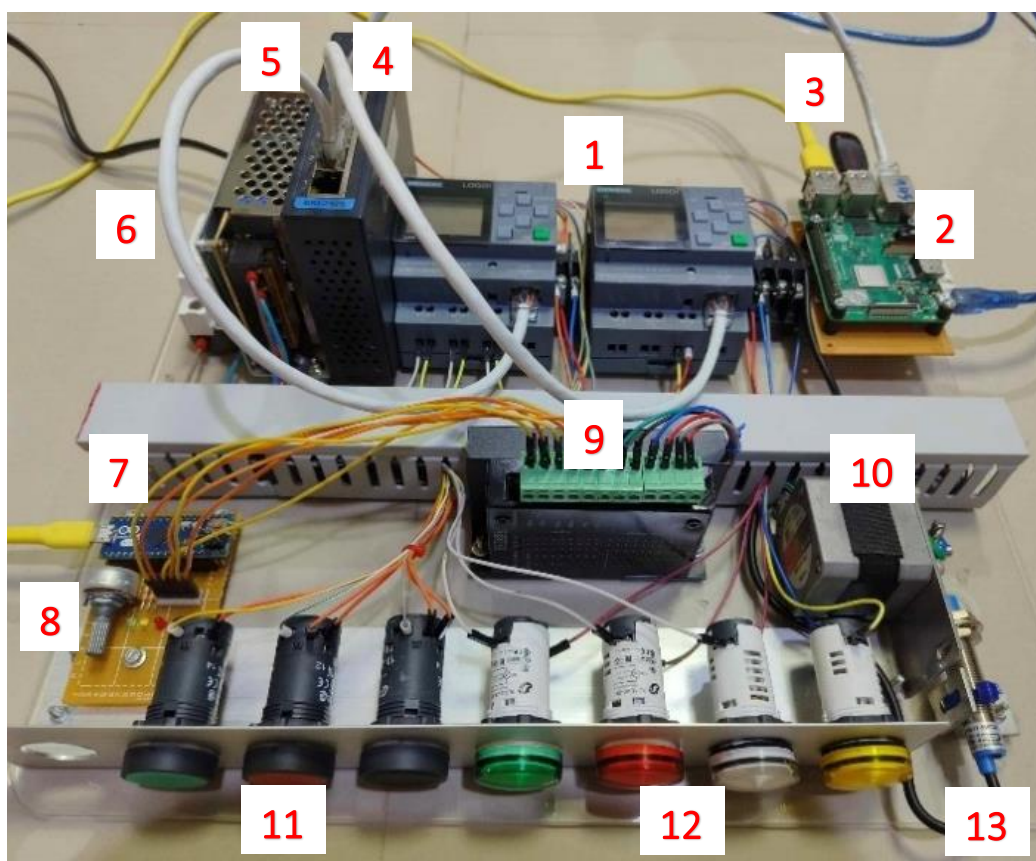
3.1.7. เชื่อมต่อกันระหว่าง Node-RED กับ Ubidots

#### 3.2. อุปกรณ์ที่ใช้

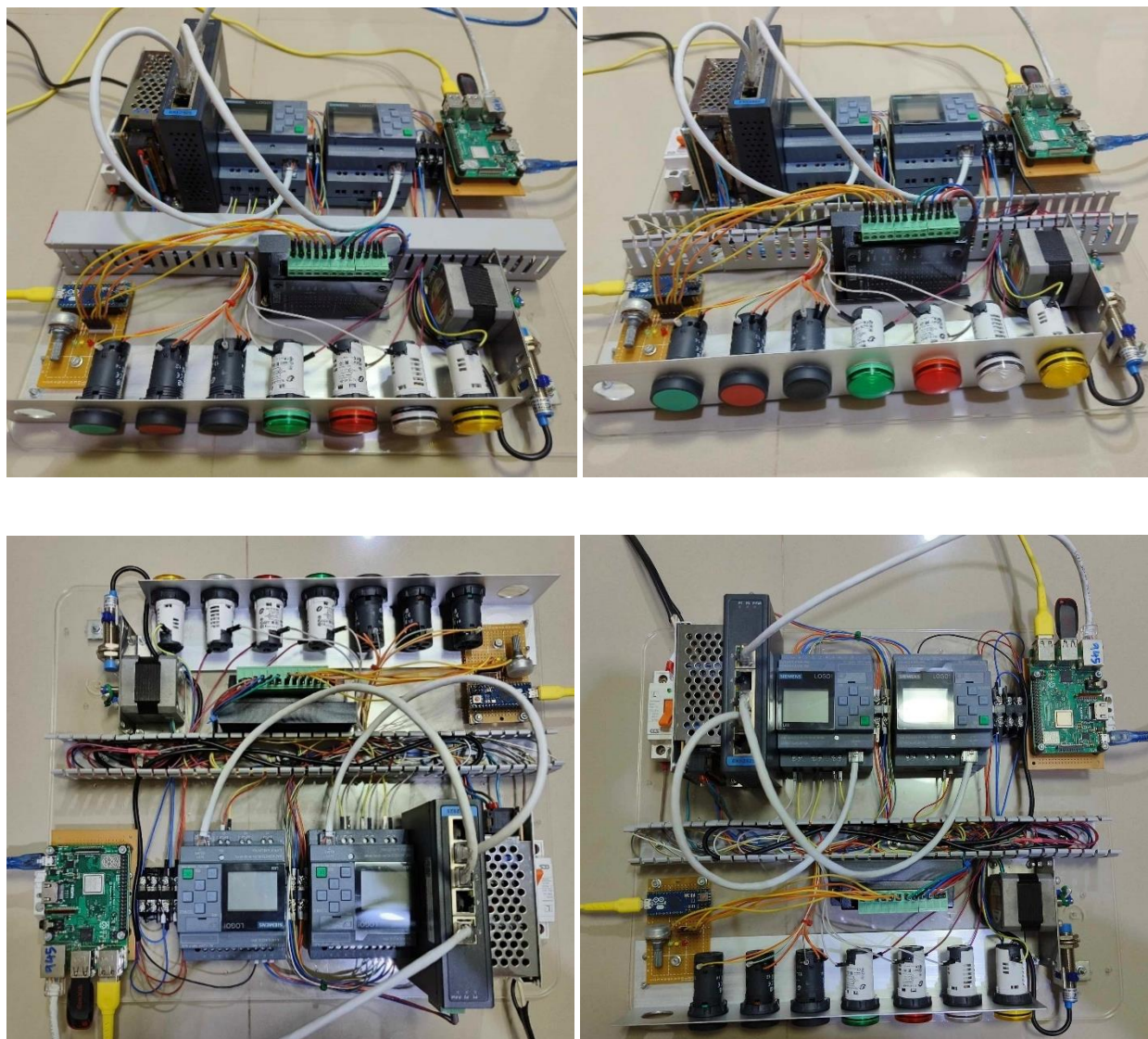
##### 3.2.1. อุปกรณ์ หรือ Hardware ที่ใช้

1. Siemens LOGO!8 จำนวน 2 ตัว
2. Raspberry Pi จำนวน 1 ตัว
3. Micro SD Card จำนวน 1 อัน
4. EKI-2525-BE จำนวน 1 ตัว
5. Switching Power Supply จำนวน 1 ตัว

6. Safety Breaker จำนวน 1 ตัว
7. Arduino Micro จำนวน 1 ตัว
8. Variable Resistor จำนวน 1 ตัว
9. TB6600 โมดูลขับ สเต็ปป์มอเตอร์ จำนวน 1 ตัว
10. Stepper Motor จำนวน 1 ตัว
11. Push Button Switch จำนวน 3 ตัว
12. LED Indicator Lamp จำนวน 4 ตัว
13. Inductive Proximity Sensor จำนวน 1 ตัว
14. สายไฟ Jumper
15. Lan Cable
16. สาย Micro USB Type B to USB 2.0 Type A



### 3.2.2. การติดตั้งฮาร์ดแวร์



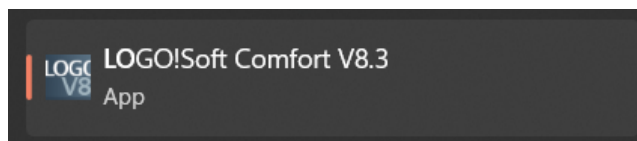
### 3.2.3. โปรแกรม ซอฟต์แวร์ และ แพลตฟอร์ม ที่ใช้

1. LOGO!Soft Comfort V8.3 ใช้ในการเขียนโปรแกรม Function Block ให้กับ LOGO!
2. Raspberry Pi Imager ใช้ในการติดตั้ง OS ให้กับ Raspberry Pi
3. VNC Viewer เป็นโปรแกรมที่ใช้เปิดหน้า Desktop ของ Raspberry Pi
4. Node-RED ใช้ในการเขียน flows เชื่อมต่อระหว่าง PLC กับ Ubidots
5. Ubidots ผู้ให้บริการแพลตฟอร์ม IoT (Internet of Things)
6. Arduino ใช้ในการเขียนโปรแกรม ให้กับ Arduino Micro เพื่อควบคุม Stepper Motor



### 3.3. ขั้นตอนการดำเนินการ

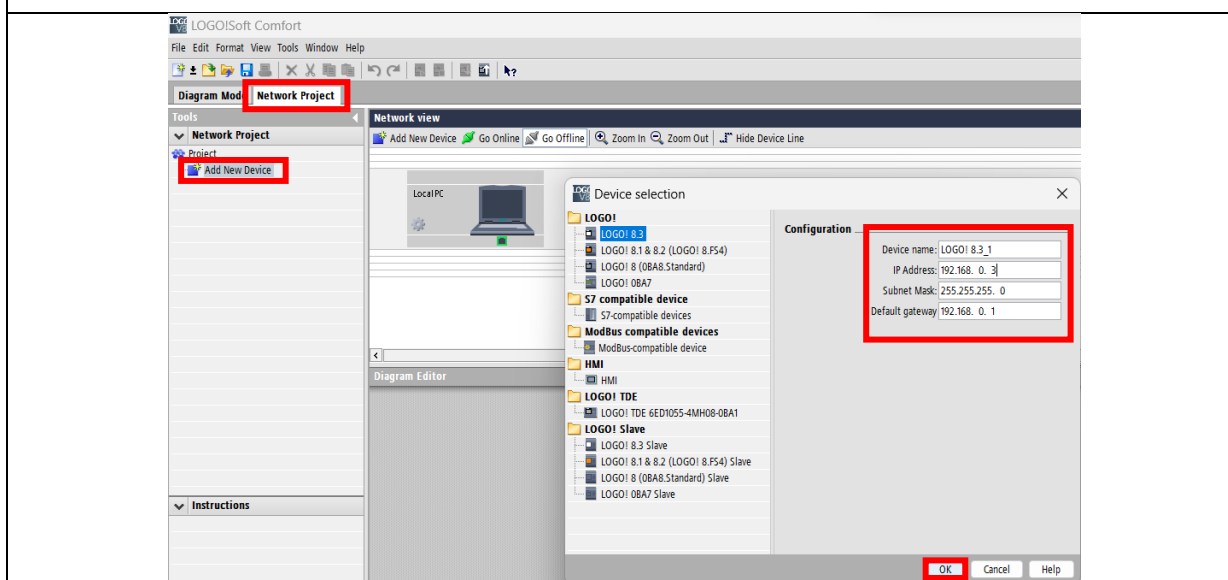
#### 3.3.1. ติดตั้ง LOGO!Soft Comfort V8.3



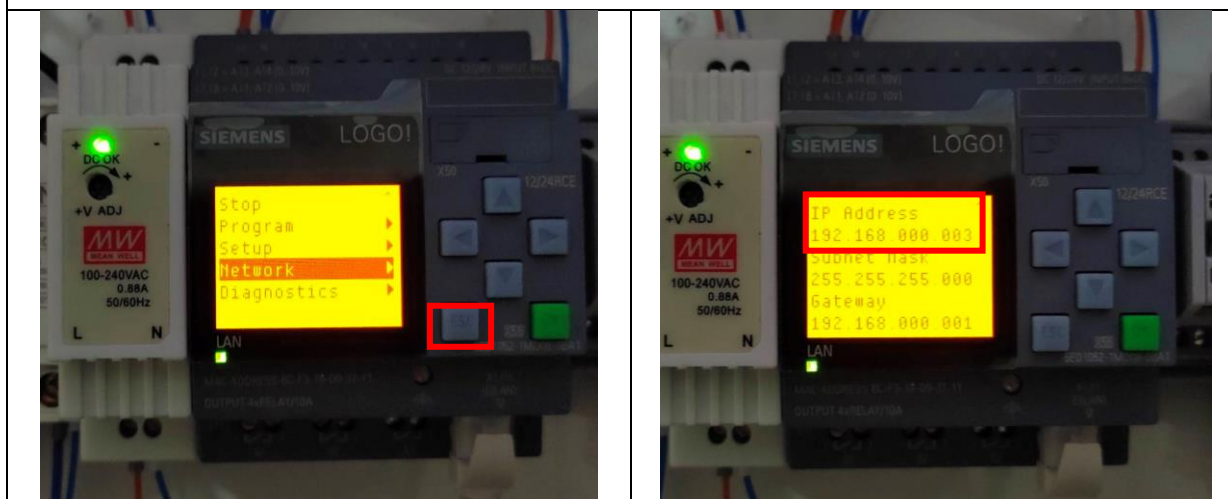
LOGO!Soft Comfort V8.3 เป็นซอฟต์แวร์ที่ใช้ในการโปรแกรม PLC LOGO! จาก Siemens.

#### 3.3.2. เขียนโปรแกรม Function Block

1. เปิดโปรแกรม LOGO!Soft Comfort V8.3 จากนั้นไปที่ Network Project → Add New Device → ตั้งค่า IP Address ให้ตรงกับ IP Address ในเครื่อง PLC

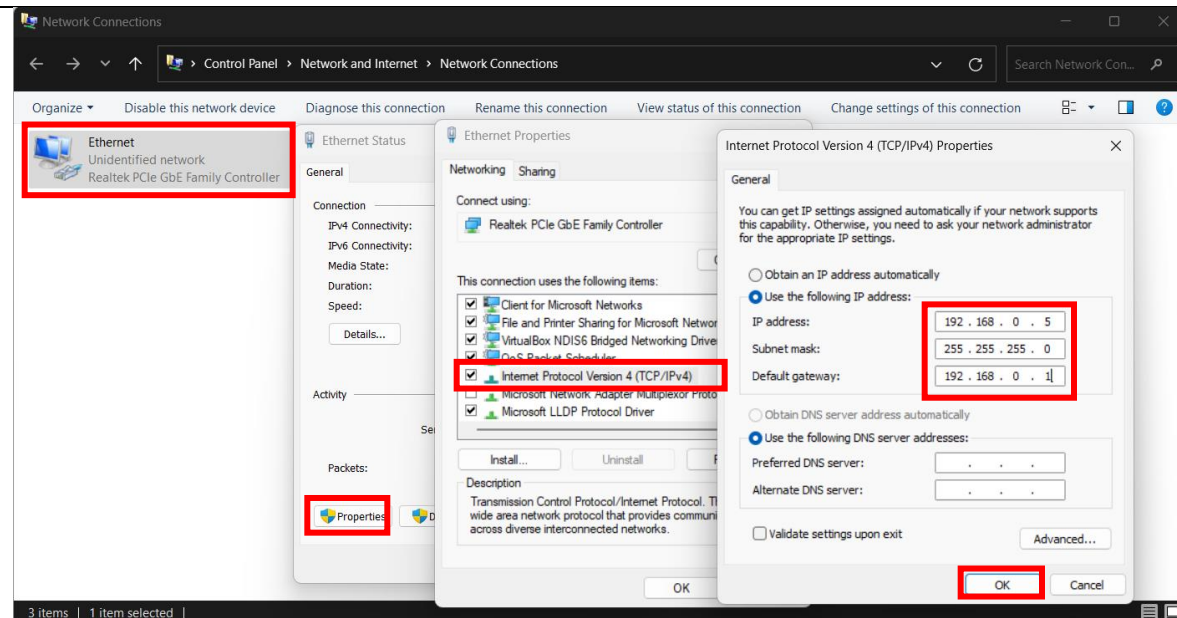


2. เช็ค IP Address เครื่อง PLC ได้โดย กดไปที่ปุ่ม ESC เลื่อนลงมาที่ Network > OK > IP Address

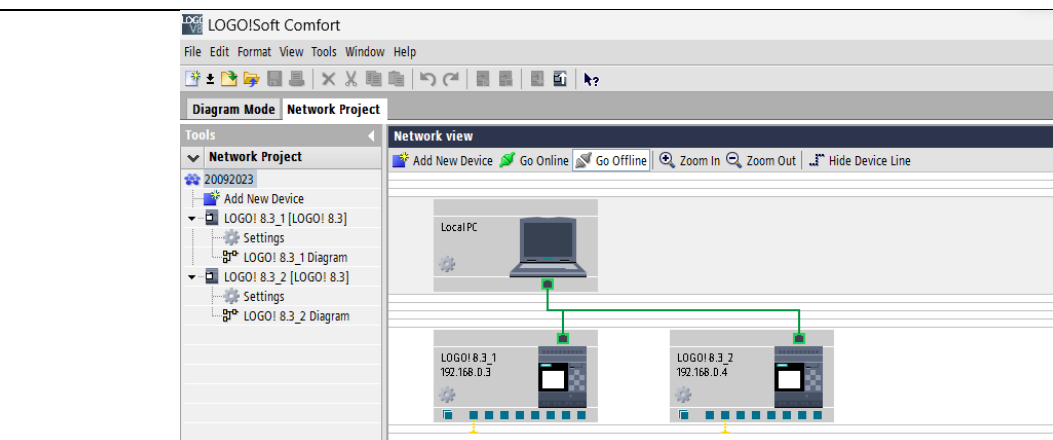


3. กำหนด IP Address ของเครื่องคอมพิวเตอร์ ให้อยู่ในวง LAN เดียวกันกับ PLC

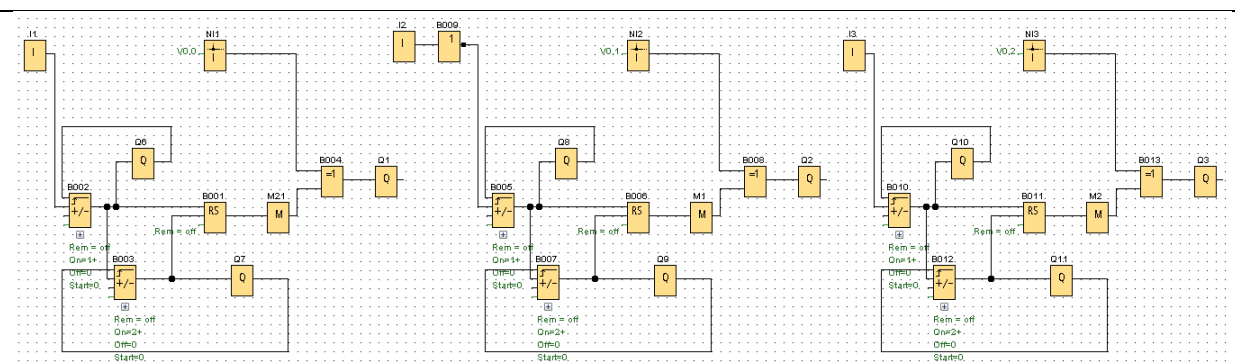
โดยไปที่โปรแกรม View network connections



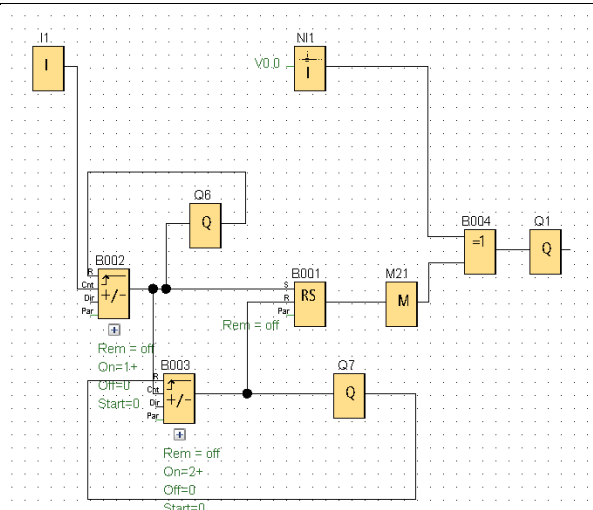
4. ผลลัพธ์ที่ได้ PLC1: IP Address 192.168.0.3 , PLC2: IP Address 192.168.0.4



5. ที่ PLC1 เขียนโปรแกรม Function Block



## 5.1 อธิบาย Function Block

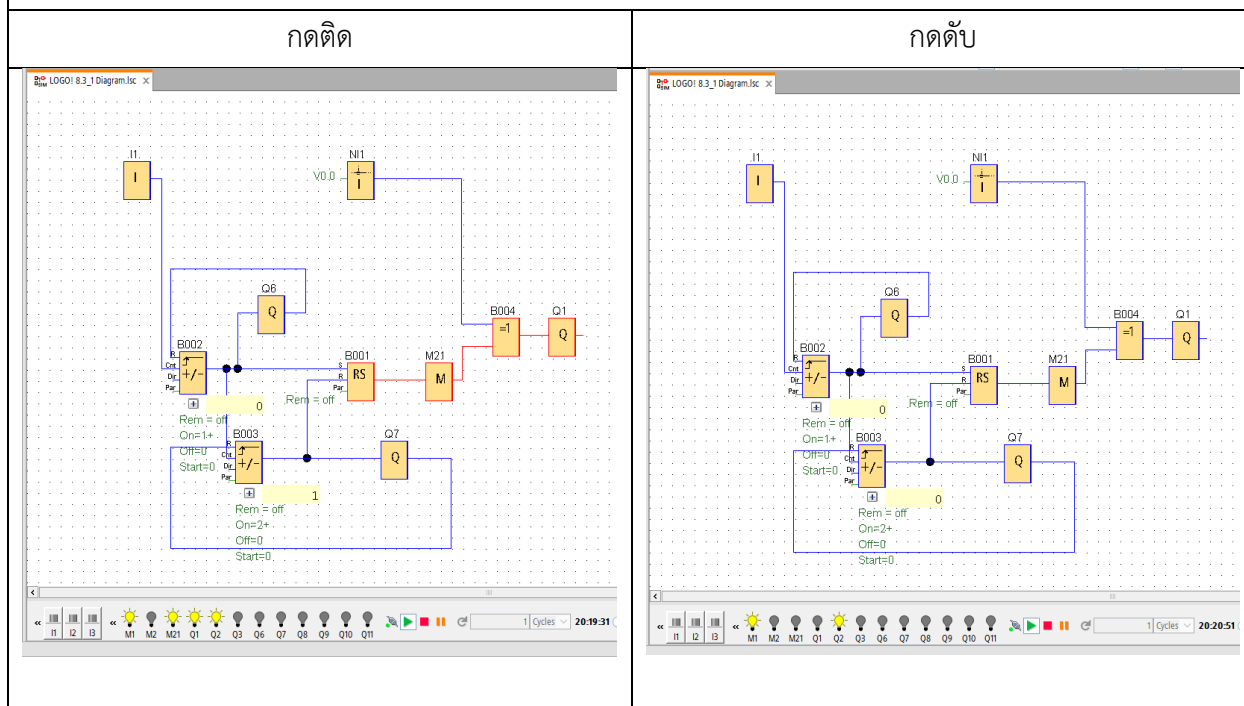


ลักษณะการทำงานของโปรแกรม คือ กดติด กดดับ โดย

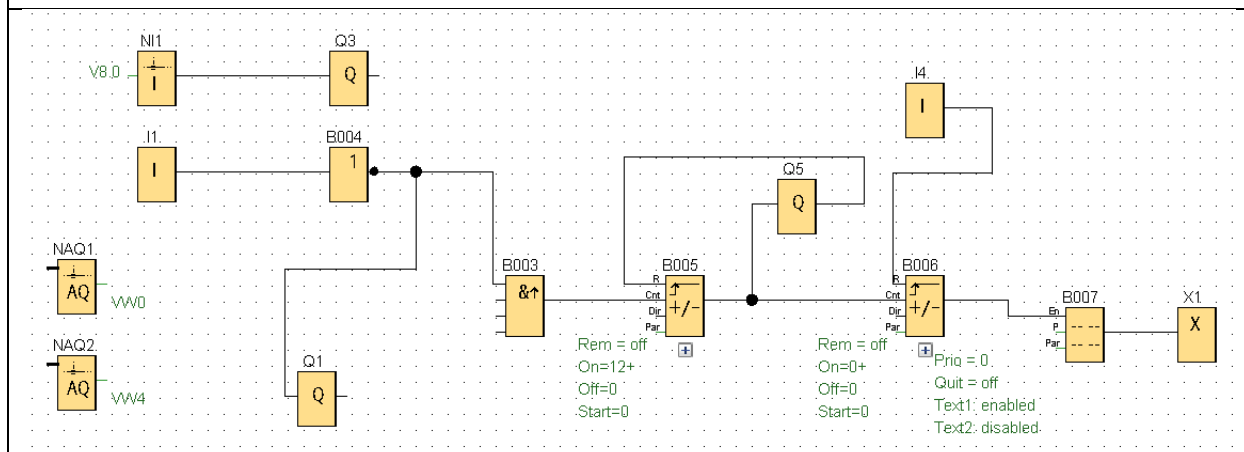
- Input คือ I1 เมื่อปุ่มถูกกดลง จะสร้างการเชื่อมต่อไฟฟ้าระหว่างขั้ว (contacts) เพื่อสร้างสัญญาณ คือ การทำให้สามารถสร้างสัญญาณเมื่อกดแต่จะหยุดสร้างเมื่อปล่อยปุ่ม
- Block B002: Up/Down counter จะมีหน้าที่นับการเปลี่ยนแปลงของสัญญาณขาเข้าและนับจำนวนขึ้นหรือลงตามคำสั่งหรือการกำหนด โดยในที่นี้เรากำหนด  
On = 1+ (Count Up): คือสัญญาณหรือสถานะที่ใช้ในการเริ่มนับจำนวนขึ้น. เมื่อ On ถูกตั้งค่าเป็น "1+" การนับจะเพิ่มค่าขึ้นเมื่อมีสัญญาณ On เข้าสู่บล็อก เมื่อนับครบจำนวนแล้ว จะส่งสัญญาณออกไป และเราได้ทำการนำเอาสัญญาณที่ส่งออกไป มาเป็นสัญญาณ Reset ตัวมันเองด้วย (Q6)
- Block B003: Up/Down counter ที่บล็อกนี้ เราได้ทำการกำหนด On = 2+ หมายถึง เมื่อนับครบ 2 จะทำการส่งสัญญาณออกไป และเราได้ทำการนำเอาสัญญาณที่ส่งออกไป มาเป็นสัญญาณ Reset ตัวมันเองด้วยเช่นกัน (Q7)
- Block B001: Latching Relay (Set, Reset) ใช้ในการควบคุมการสลับสถานะของเปลี่ยนแปลงการเชื่อมต่อไฟฟ้า โดยมีสถานะ "Set" (ตั้ง) และ "Reset" (รีเซ็ต) โดยไม่จำเป็นต้องใช้กระแสไฟฟ้าเพื่อรักษาสถานะ
- Block B004: XOR
- M21: Memory Locations (ตำแหน่งหน่วยความจำ) M21 - M27 ใช้สำหรับการควบคุมความเข้าใจของโปรแกรม, เช่น การเปลี่ยนสถานะของตัวแปร
- NI1: Network Interface หรืออินเตอร์เฟซของเครือข่าย ช่วยในการเชื่อมต่อและการสื่อสารระหว่างอุปกรณ์หรือระบบต่าง ๆ ที่เชื่อมต่อกับ LOGO! 8 ผ่านเครือข่าย เพื่อรับข้อมูลหรือส่งข้อมูล
- Q1: คือหน่วยควบคุมหรือ Relays ที่ใช้เป็นตัวสวิตช์เพื่อควบคุมอุปกรณ์หรือระบบต่างๆ (Output)

เขียนโปรแกรม Function Block ในลักษณะเช่นเดียวกัน ทั้งหมด 3 ชุด

## 5.2 Simulation



## 6. ที่ PLC2 เขียนโปรแกรม Function Block



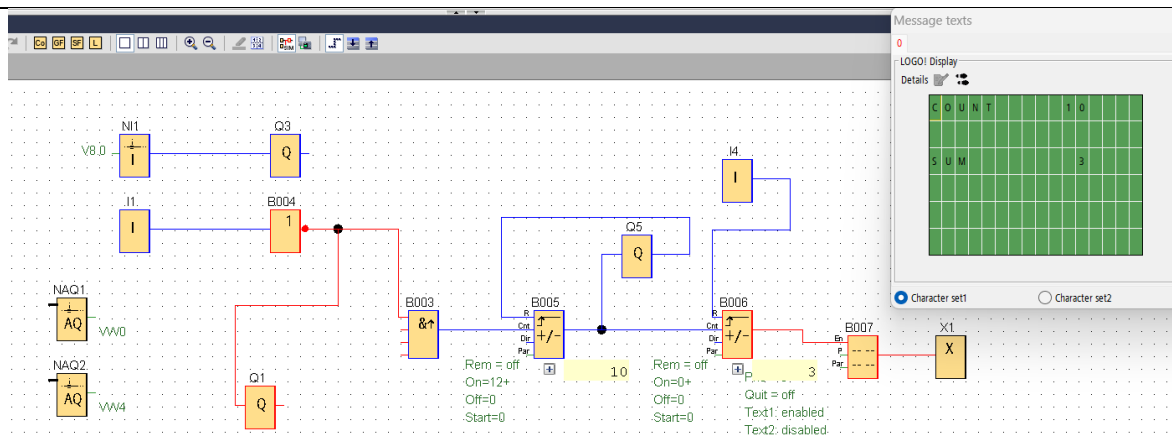
### 6.1 อธิบาย Function Block

- NI1: Network Interface
- I1: Input คือสัญญาณจาก Inductive Proximity Sensor ส่งสัญญาณเป็นไฟลอป (Low), สัญญาณไฟจะมีระดับต่ำเมื่อเซนเซอร์ตรวจจับวัตถุหรือวัตถุอยู่ใกล้เซนเซอร์ นั่นหมายถึงเมื่อเซนเซอร์ตรวจจับวัตถุหรือวัตถุอยู่ใกล้จะเปิดวงจรและส่งสัญญาณไฟลอปออกมา
- Q3: Output ของสัญญาณ Input จาก Network Interface (NI1)
- NAQ1, NAQ2: Network analog output

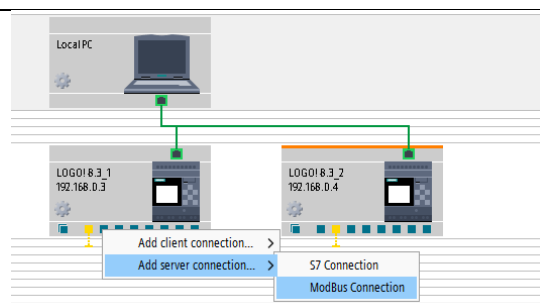


- Block B004: NOT บล็อกตรรกะที่ใช้ในการเปลี่ยนแปลงสถานะของเงื่อนไขหรือค่าตรรกะที่ถูกนำเข้ามาให้เป็นค่าตรงข้าม คือถ้าค่านำเข้าเป็น "1" (จริง) บล็อก NOT จะเปลี่ยนค่าเป็น "0" (เท็จ)
- Block B003: AND (Edge) ใช้ในการตรวจสอบขอบของสัญญาณ ในที่นี้เราตรวจสอบขอบขาขึ้น (Rising Edge) เมื่อเปลี่ยนสถานะจาก 0 เป็น 1 ซึ่งเกิดขึ้นเมื่อเซนเซอร์ตรวจจับวัตถุเจอ จะสร้างขอบขาขึ้น และส่งสัญญาณ "จริง" ไปที่ขานำออก
- Block B005: Up/Down counter ที่บล็อกนี้ เราได้ทำการกำหนด On = 12+ หมายถึง เมื่อนับครบ 12 จะทำการส่งสัญญาณออกไป และเราได้ทำการนำเอาสัญญาณที่ส่งออกไป มาเป็นสัญญาณ Reset ตัวมันเองด้วยเช่นกัน (Q5)
- Block B006: Up/Down counter ที่บล็อกนี้ เราได้ทำการกำหนด On = 0+ หมายถึง นับเพิ่มขึ้นไปเรื่อยๆ ไม่ได้กำหนดจำนวนเพื่อส่งสัญญาณออกไป
- Block B007: Message texts เป็นบล็อกที่ใช้ในการจัดเก็บข้อความหรือข้อความที่สามารถแสดงบนหน้าจอของ LOGO! PLC หรือใช้ในโปรแกรมควบคุมของระบบได้
- Block X1: Open connector การใช้บล็อก "Message Texts" และ "Open Connector Block" ร่วมกันในโปรแกรม PLC จะช่วยในการแสดงข้อความหรือข้อความบนหน้าจอของ LOGO! PLC และจัดการการเชื่อมต่อไฟฟ้าระหว่างอุปกรณ์ต่าง ๆ

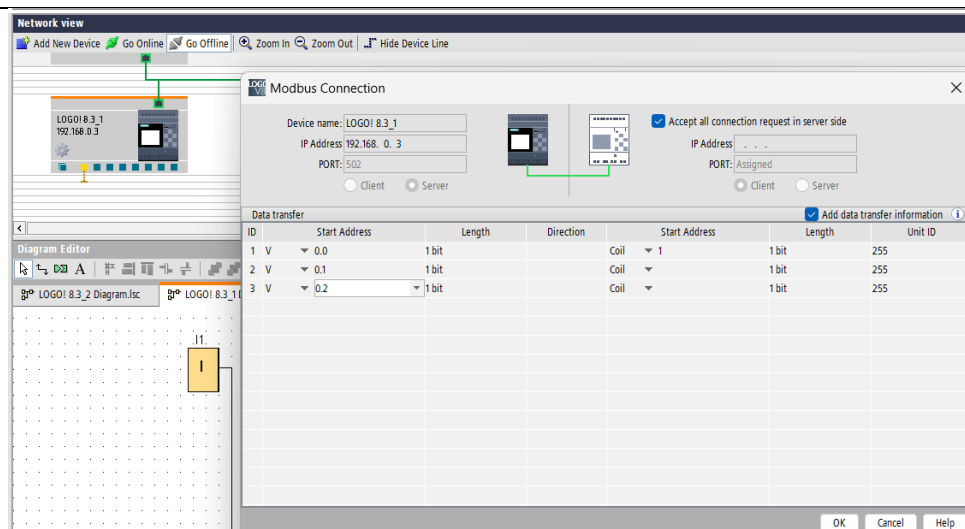
## 6.2 Simulation



## 7. ทำการ Add Server connection เพื่อใช้งาน ModBus Connection (PLC1=port502, PLC2=port503)



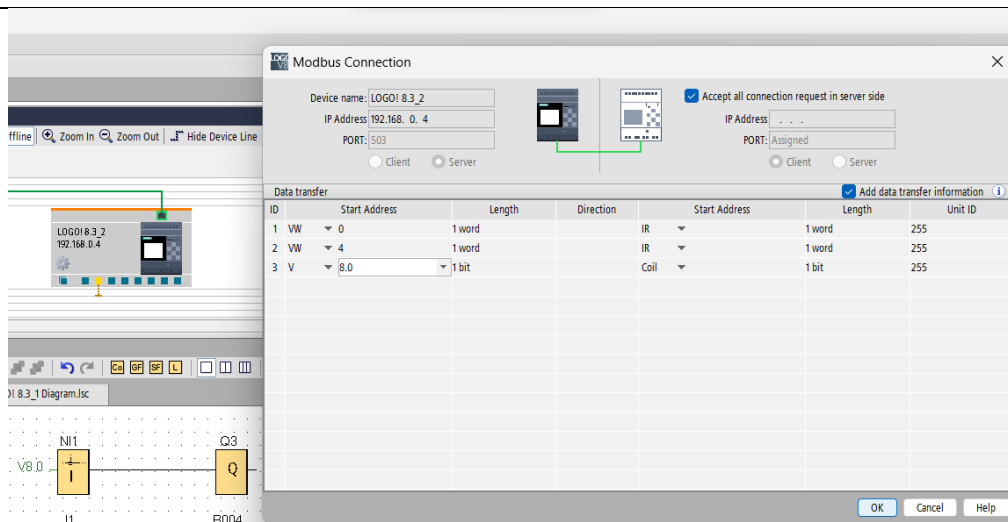
7.1 ที่ LOGO! PLC ตัวที่ 1 ทำการ Add data transfer information (Network input) การเพิ่มข้อมูลสำหรับการถ่ายโอนข้อมูลที่ Port 502 ในคอนเท็กซ์ของ LOGO! PLC ที่มีการเชื่อมต่อผ่านเครือข่ายใช้โปรโตคอล Modbus TCP/IP



7.2 ที่ LOGO! PLC ตัวที่ 2 ใช้ "Variable Table" เพื่อสร้างตัวแปรและกำหนดค่าในโปรแกรม ซึ่งสามารถเก็บค่าตัวเลขหรือข้อมูลอื่น ๆ ได้ตามความต้องการ โดยไปที่ Tools → Parameter VM Mapping

ID	Block	Parameter	Type	Address
1	B005 [Up/Down counter]	Counter	DWord	0
2	B006 [Up/Down counter]	Counter	DWord	4
3				

7.3 2 ที่ LOGO! PLC ตัวที่ 2 Add data transfer information

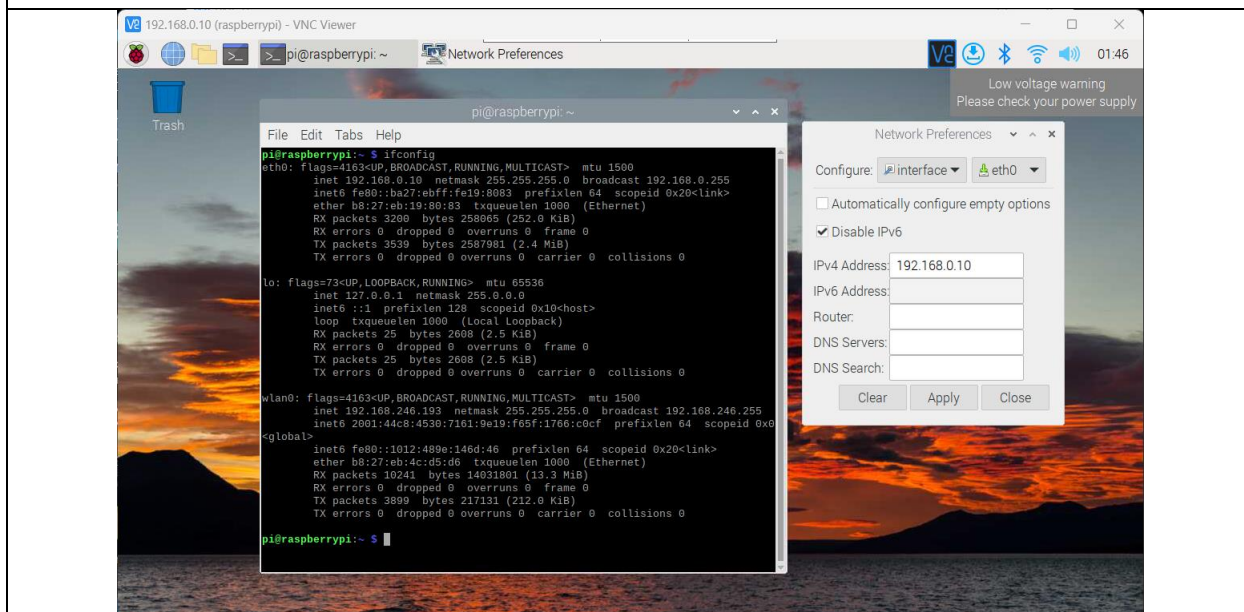


### 3.3.3. ติดตั้งระบบปฏิบัติการของ Raspberry Pi

#### 3.3.3.1. ทำการติดตั้งระบบปฏิบัติการของ Raspberry Pi

- เมื่อติดตั้งสำเร็จ ใช้ VNC Viewer เปิดหน้า Desktop ของ Raspberry Pi

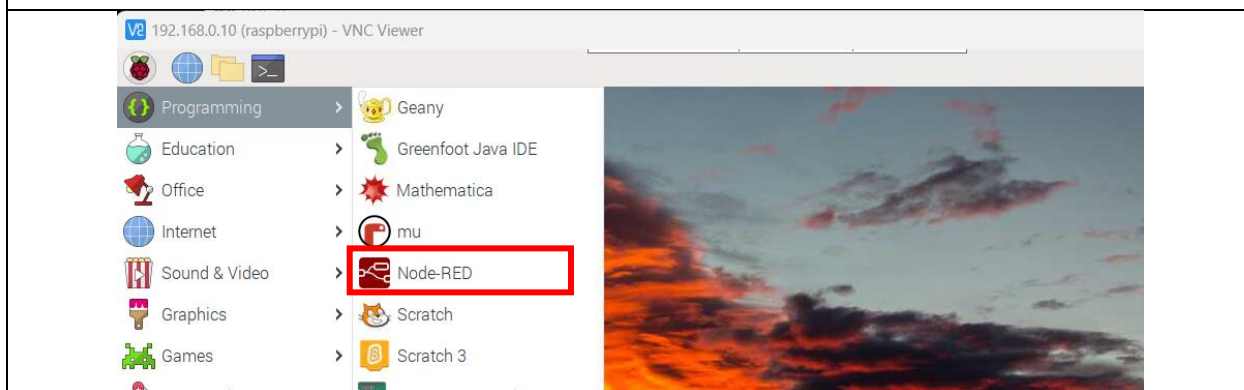
- ทำการกำหนด IPv4 Address ที่ interface eth0 ซึ่งเป็นอินเตอร์เฟซสำหรับการเชื่อมต่อผ่านสาย เราจะกำหนดให้อยู่ในวง LAN เดียวกันทั้งระบบ เพื่อให้ง่ายต่อการ remote control



#### 3.3.3.2. ติดตั้ง Node-RED บน Raspberry Pi

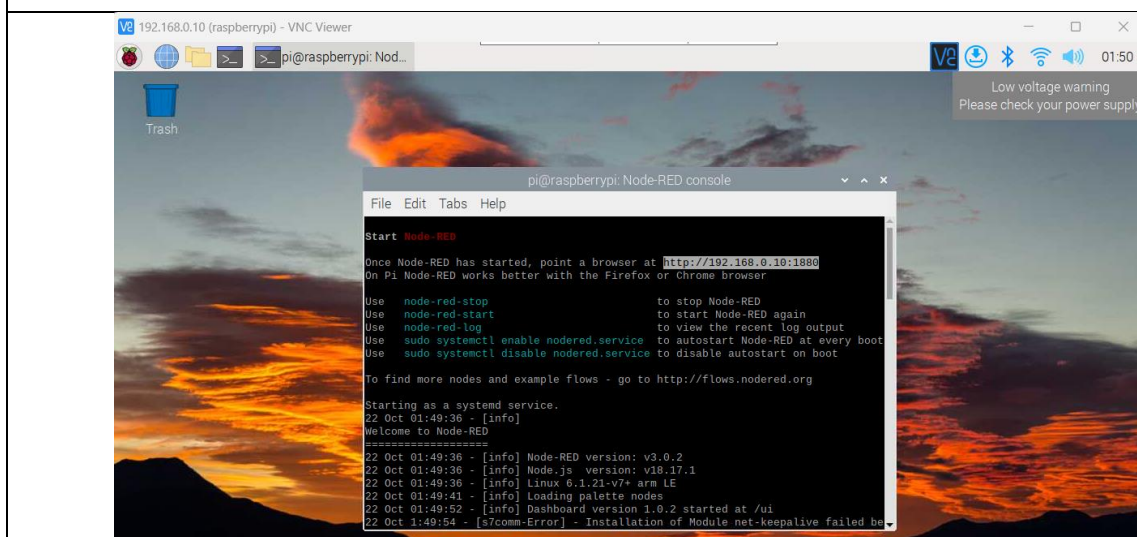
- ติดตั้ง Node-RED (Install/Upgrade) โดยใช้คำสั่งดังนี้

bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)



- หากติดตั้งสำเร็จ จะมีโปรแกรม Node-Red ดังรูป

- Start Node-Red

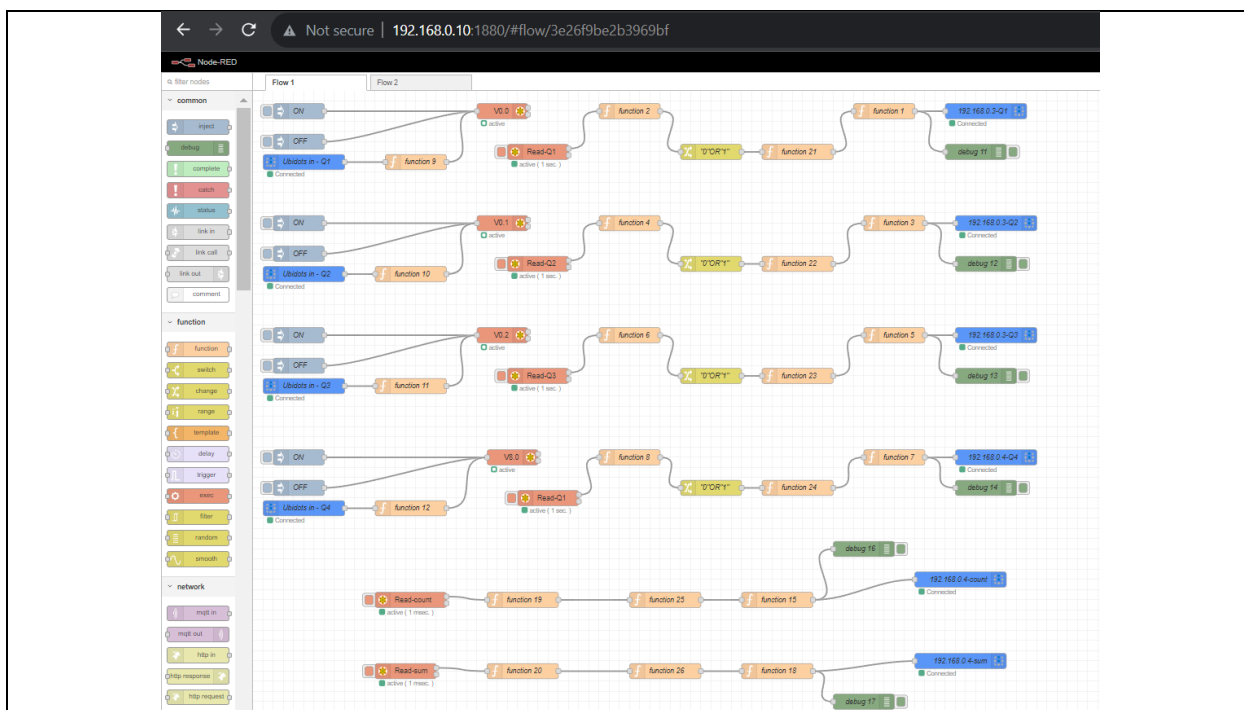


- อีกวิธีคือ การเปิดให้ Node-RED ทำงานอัตโนมัติเมื่อ Raspberry Pi ถูกเปิดหรือรีบูต โดยใช้คำสั่ง `sudo systemctl enable nodered.service`

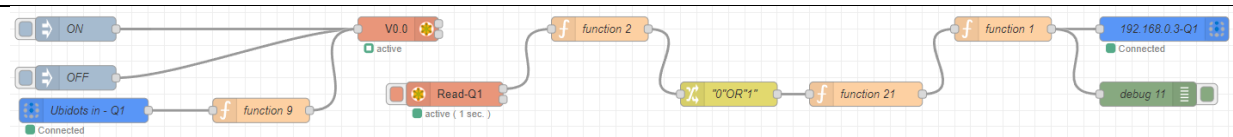
เข้าสู่ Node-RED ผ่านเบราว์เซอร์โดยเข้าไปที่ `http://<Raspberry_Pi_IP_Address>:1880` โดยใช้ที่อยู่ IP ของ Raspberry Pi และพอร์ต 1880 (หากไม่ได้กำหนดพอร์ตอื่น)

### 3.3.3.3. เขียนโปรแกรม

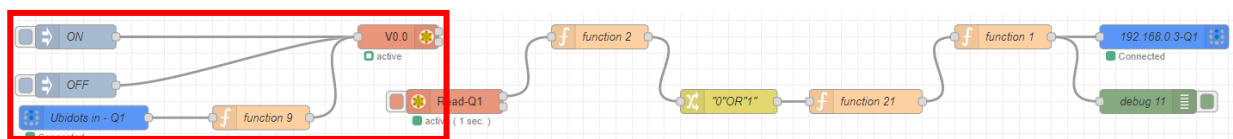
- ที่ Node-Red ทำการ Install `node-red-contrib-modbus` และ `ubidots-nodered`



## 1. อธิบาย Flow-based programming



- ส่วนแรก เป็นการ Write



- ก่อนอื่น เราจะต้องทราบถึง Modbus Address (Address = Address-1)

LOGO! settings

Offline settings Online settings

General  
Hardware type  
I/O settings  
I/O names  
Program password  
Power on  
Message text  
Additional info  
Statistics  
Comment  
Modbus address space

Modbus address space

Address Type	Range	Mapped Modbus Address	Direction	Unit
I	1 - 24	Discrete Input (DI) 1 - 24	R	bit
Q	1 - 20	Coil 8193 - 8212	R/W	bit
M	1 - 64	Coil 8257 - 8320	R/W	bit
V	0.0 - 850.7	Coil 1 - 6808	R/W	bit
AI	1 - 8	Input Register (IR) 1 - 8	R	word
VW	0 - 850	Holding Register (HR) 1 - 425	R/W	word
AQ	1 - 8	Holding Register (HR) 513 - 520	R/W	word
AM	1 - 64	Holding Register (HR) 529 - 592	R/W	word

- เราจะทำกร Write ไปยัง V0.0 (Address = 0) โดยใช้ฟังก์ชัน FC 5 : Force single coil  
การ "Force" หรือ "Force Write" คือการสั่งให้อุปกรณ์ Modbus เขียนค่าลงใน single coil หรือ single register ให้มีค่าที่กำหนดเอง

Edit Modbus-Write node

Delete Cancel Done

Properties

Settings

Name V0.0

Unit-Id 255

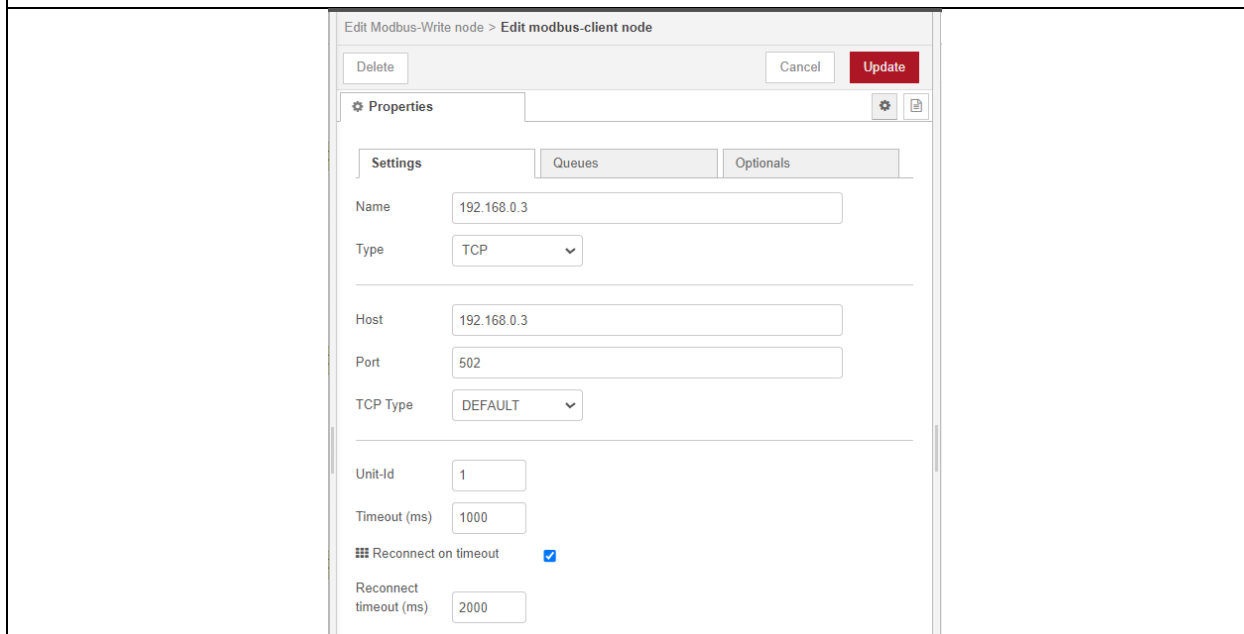
FC FC 5: Force Single Coil

Address 0

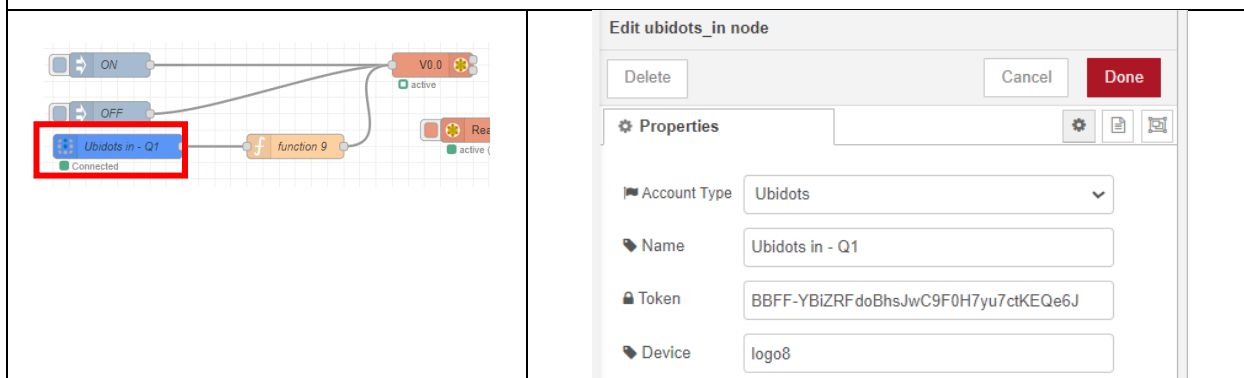
Delay to activate input ☐

Server 192.168.0.3

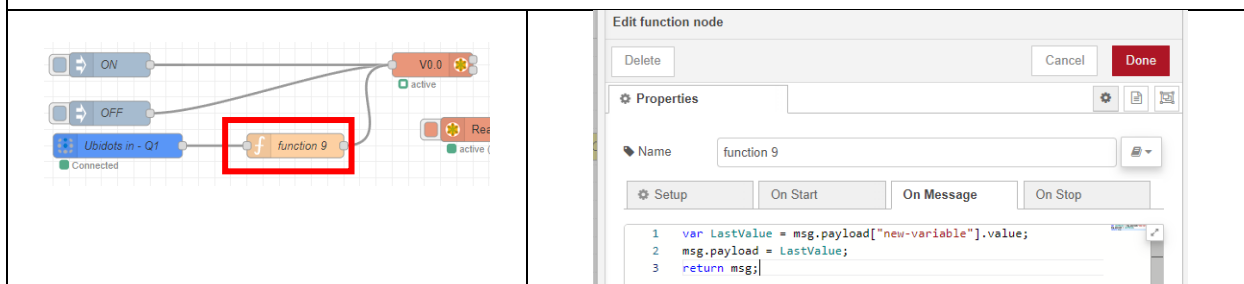
- ในส่วนของ Server กำหนด Type = TCP, Host = IP Address (PLC1), Port = 502 และถ้าเป็น (PLC2) Host = IP Address (PLC2), Port = 503 ตามที่เราเคยกำหนดในขั้นตอน Add Server connection



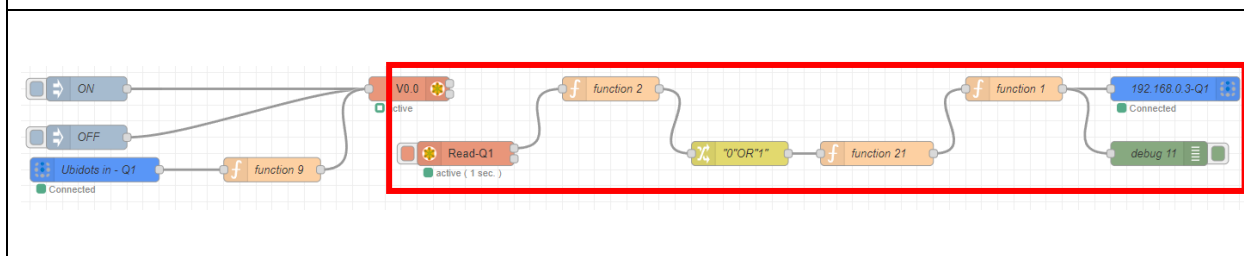
- ใช้บล็อก ubidots-in สำหรับการเชื่อมต่อกับ Ubidots IoT platform



- สร้างตัวแปร LastValue และกำหนดค่าของ LastValue จากข้อมูลใน msg.payload ที่เป็นออบเจกต์ (object) จากนั้นกำหนดให้ msg.payload = LastValue และ return msg



- ส่วนต่อมา เป็นการ Read



- ใช้ Modbus-Read node เพื่อที่จะ Read Coil Status  
ผลลัพธ์ที่ได้ คือ msg.payload ที่เป็น array[8]

**Edit Modbus-Read node**

Delete Cancel Done

**Properties**

**Settings** Options

Name: Read-Q1

Topic: Topic

Unit-Id: 255

FC: FC 1: Read Coil Status

Address: 8192

Quantity: 1

Poll Rate: 1 second(s)

☐ Delay to activate input

Server: 192.168.0.3

```
10/23/2023, 12:52:31 PM node: debug 22
polling : msg.payload : array[8]
> [ true, false, false, false, false,
false, false, false ]
```

- จากนั้นเราได้เขียน function ทำการแยก array โดยในที่นี้เราต้องการ array ตัวแรก นั่นคือ index 0

**Edit function node**

Delete Cancel Done

**Properties**

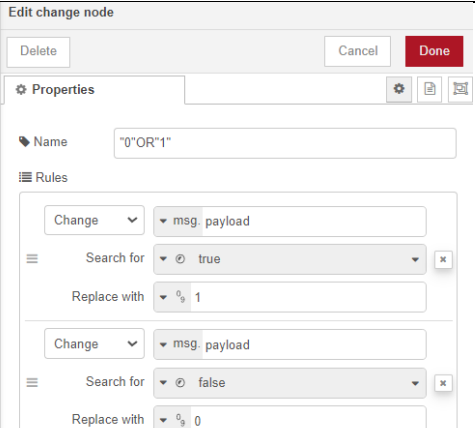
Name: function 2

**Setup** On Start On Message On Stop

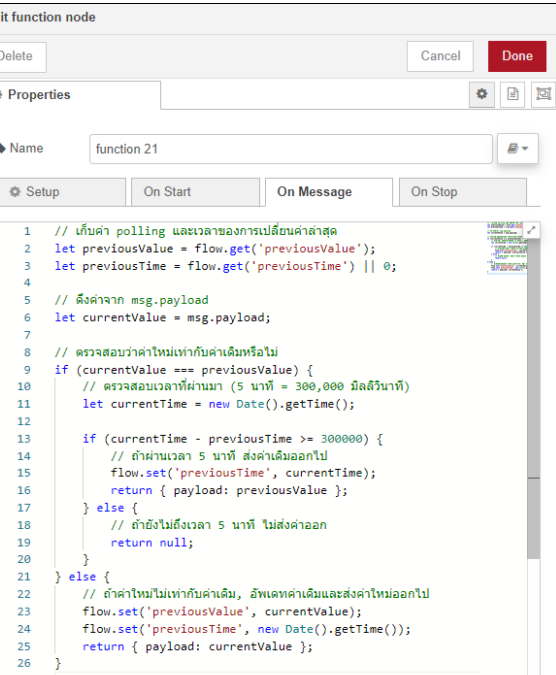
```
1 // ตรวจสอบว่า payload เป็น array และมีค่าใน index 0
2 if (Array.isArray(msg.payload) && msg.payload.length > 0) {
3   // ดึงค่าจาก index 0 ของ array และแปลงเป็น String
4   msg.payload = msg.payload[0].toString();
5 } else {
6   // หาก payload ไม่ใช่ array หรือไม่มีค่าใน index 0 ให้ตั้งค่าเป็นข้อ
7   msg.payload = '';
8 }
9
10 return msg;
11
```

```
10/23/2023, 12:52:31 PM node: debug 23
polling : msg.payload : string[4]
"true"
```

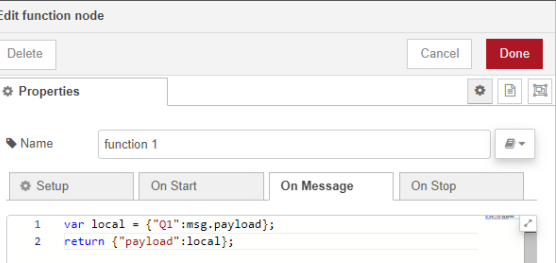
- ใช้ change node เพื่อทำการเปลี่ยน msg.payload ที่เป็น string ให้เป็น number

	<pre>10/23/2023, 12:52:30 PM node: debug 24 polling : msg.payload : number 1</pre>
---	--

- เขียน function ในการเช็คค่าที่เข้ามาใหม่ ว่ามีค่าเท่าเดิมหรือไม่ หากค่าที่เข้ามาใหม่มีค่าเท่าเดิม เรา  
จะให้รอเวลาผ่านไป 5 นาที ถึงจะส่งค่าที่มีค่าเท่าเดิมออกไป แต่หากค่าที่เข้ามามีค่าเปลี่ยนไปจากเดิม  
จะทำการส่งค่าใหม่ออกไปทันที และอัปเดตค่า

	<pre>10/23/2023, 1:47:23 PM node: debug 25 msg.payload : number 1</pre>
--	---

- เขียน function เพื่อส่งค่าที่รับเข้ามา และทำการส่งออกไปในลักษณะที่เป็น Object

	<pre>10/23/2023, 1:47:23 PM node: debug 11 msg.payload : Object ▶ { Q1: 1 }</pre>
---	---



- ใช้ ubidots-out node สำหรับการเชื่อมต่อกับ Ubidots IoT platform

Edit ubidots\_out node

Delete Cancel Done

Properties

Account Type Ubidots

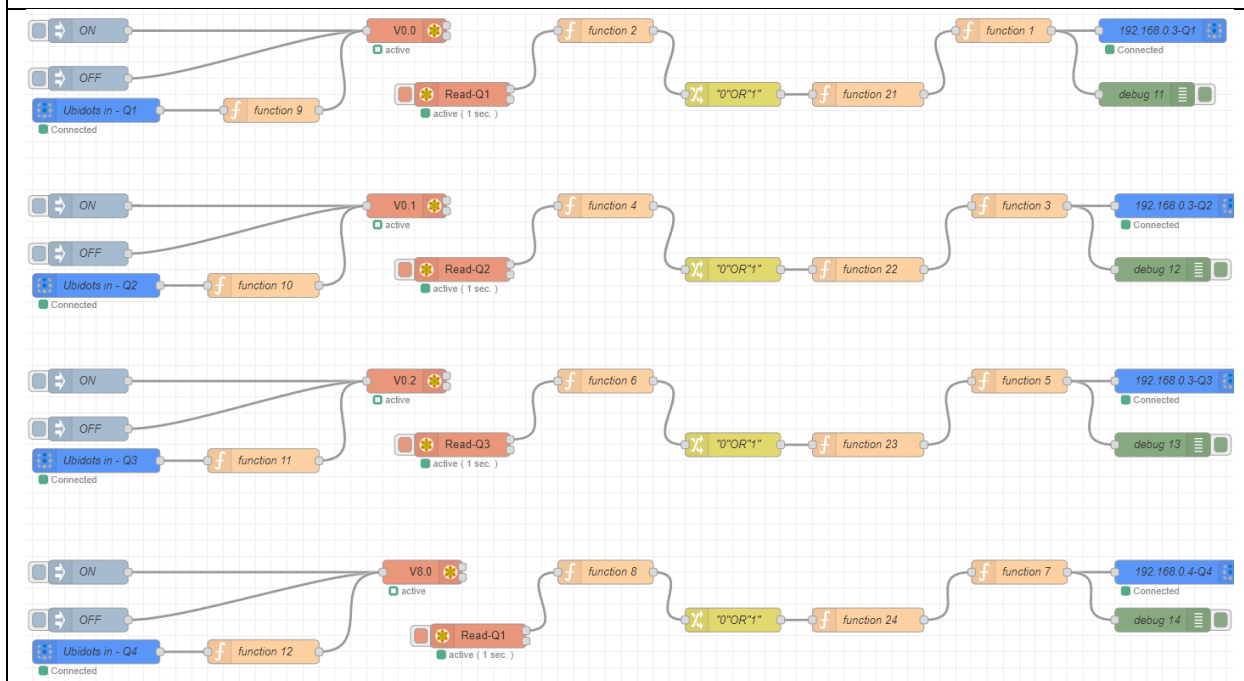
Name 192.168.0.3-Q1

Token BBFF-YBIZRFdoBhsJwC9F0H7yu7ctKEQe6J

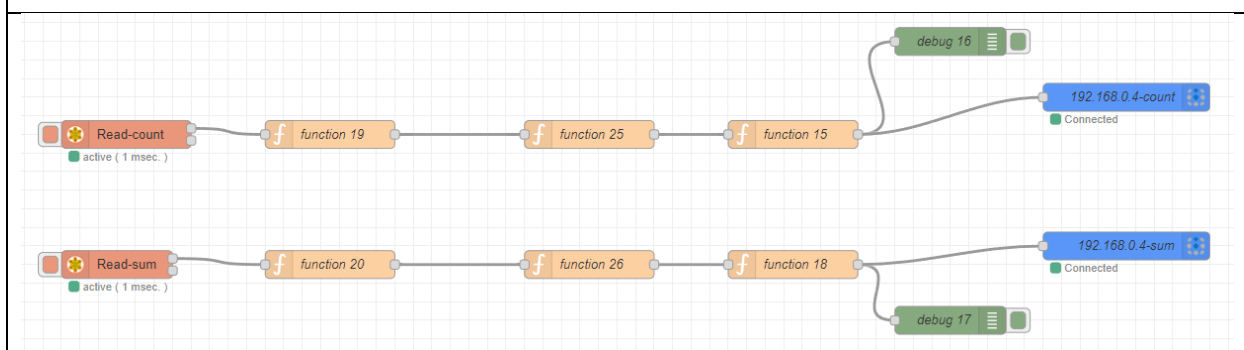
Device Label logo8

☒ Enable secure TLS connection

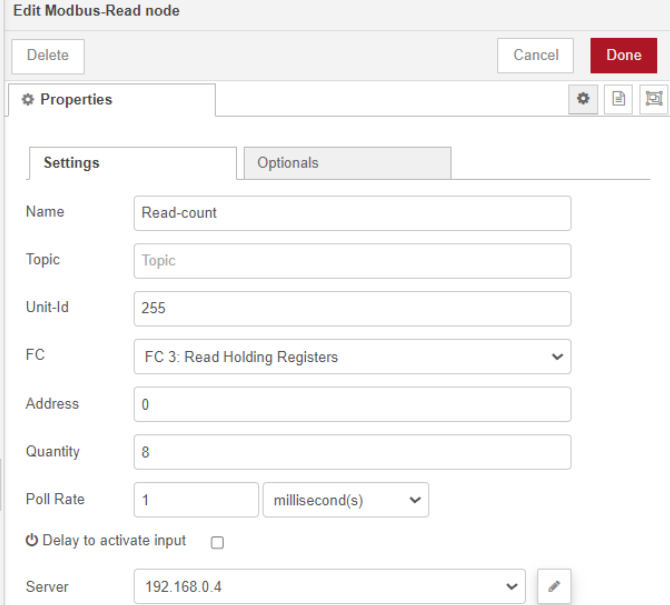
- เขียน Flow-based programming ในลักษณะเดียวกัน จำนวน 4 ชุด



- อธิบายในส่วนของการ Read ค่าที่อยู่ใน Variable Table ซึ่งเป็นค่าที่มาจากการนับ input จากเซ็นเซอร์

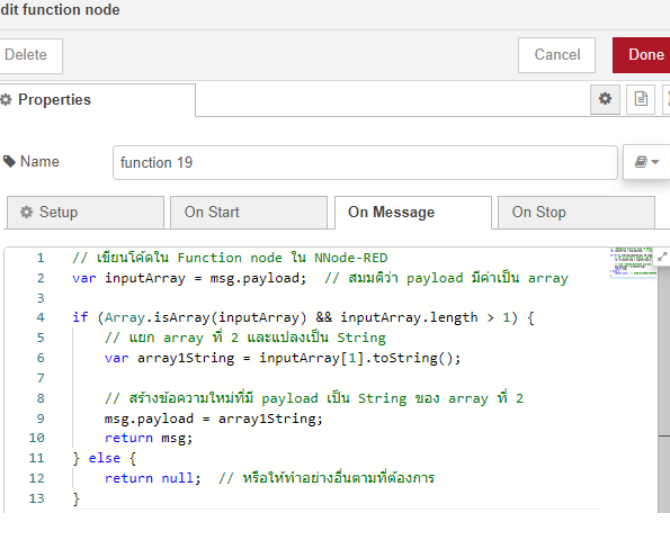


- ที่ Modbus read node เราใช้ function : Read Holding Registers เป็นฟังก์ชันที่ใช้ในการอ่านข้อมูลจาก Holding Registers ในการสื่อสารผ่าน Modbus
- Holding Registers เป็นหนึ่งในหมวดหมู่ของ Registers ใน Modbus protocol ซึ่งใช้ในการเก็บข้อมูลหรือค่าที่สามารถเข้าถึงและอ่านได้จากอุปกรณ์หรือระบบที่ใช้ Modbus
- ผลลัพธ์ที่ได้ คือ msg.payload ที่เป็น array[8]



```
10/23/2023, 1:15:59 PM node: debug 26
polling : msg.payload : array[8]
▶ [ 0, 2, 0, 3, 0, 0, 0, 0 ]
```

- จาก msg.payload array[8] ที่ array index 1 เป็น ตำแหน่งที่เราใช้เก็บค่า cuont จากนั้นเขียน function ทำการแยก array ที่เราสนใจ



```
1 // เขียนโค้ดใน Function node ใน NNode-RED
2 var inputArray = msg.payload; // สมมติว่า payload มีค่าเป็น array
3
4 if (Array.isArray(inputArray) && inputArray.length > 1) {
5   // แยก array ที่ 2 และแปลงเป็น String
6   var array1String = inputArray[1].toString();
7
8   // สร้างข้อความใหม่ที่มี payload เป็น String ของ array ที่ 2
9   msg.payload = array1String;
10  return msg;
11 } else {
12   return null; // หรือให้ทำอย่างอื่นตามที่ต้องการ
13 }
```

```
10/23/2023, 1:15:59 PM node: debug 27
polling : msg.payload : string[1]
"2"
```

- เขียน function ในการแปลงค่า และ เช็คว่าค่าที่เข้ามาใหม่

Delete
Cancel
Done

Properties

Name
function 25

Setup
On Start
On Message
On Stop

```

1 // ดึงค่าจาก msg.payload และแปลงเป็น Number
2 let currentValueCount = parseFloat(msg.payload);
3
4 // เก็บค่าที่ส่งมาไว้ใน flow variable
5 let previousValueCount = flow.get('previousValueCount');
6
7 if (!isNaN(currentValueCount)) {
8   // ตรวจสอบว่าค่าใหม่เท่ากับค่าเดิมหรือไม่
9   if (currentValueCount === previousValueCount) {
10    // ตรวจสอบเวลาที่ผ่านมา (5 นาที = 300,000 มิลลิวินาที)
11    let currentTimeCount = new Date().getTime();
12    let previousTimeCount = flow.get('previousTimeCount') || 0;
13
14    if (currentTimeCount - previousTimeCount >= 300000) {
15      // ถ้าผ่านเวลา 5 นาที ส่งค่าเดิมออกไป
16      flow.set('previousTimeCount', currentTimeCount);
17      return { payload: previousValueCount };
18    } else {
19      // ถ้ายังไม่ถึงเวลา 5 นาที ไม่ส่งค่าออก
20      return null;
21    }
22   } else {
23     // ถ้าค่าใหม่ไม่เท่ากับค่าเดิม, อัปเดตค่าเดิมและส่งค่าใหม่ออกไป
24     flow.set('previousValueCount', currentValueCount);
25     flow.set('previousTimeCount', new Date().getTime());
26     return { payload: currentValueCount };
27   }
28 } else {
29   // ถ้าไม่สามารถแปลงเป็นตัวเลขได้, ไม่ทำอะไร
30   return null;
31 }
32

```

10/23/2023, 1:15:59 PM node: debug 28  
msg.payload : number  
2

- เขียน function เพื่อส่งค่าที่รับเข้ามา และทำการส่งออกไปในลักษณะที่เป็น Object

Delete
Cancel
Done

Properties

Name
function 15

Setup
On Start
On Message
On Stop

```

1 var local = { "count": msg.payload };
2 return { "payload": local };

```

10/23/2023, 1:15:59 PM node: debug 16  
msg.payload : Object  
{ count: 2 }

- ใช้ ubidots-out node สำหรับการเชื่อมต่อกับ Ubidots IoT platform

Delete
Cancel
Done

Properties

Account Type
Ubidots

Name
192.168.0.4-count

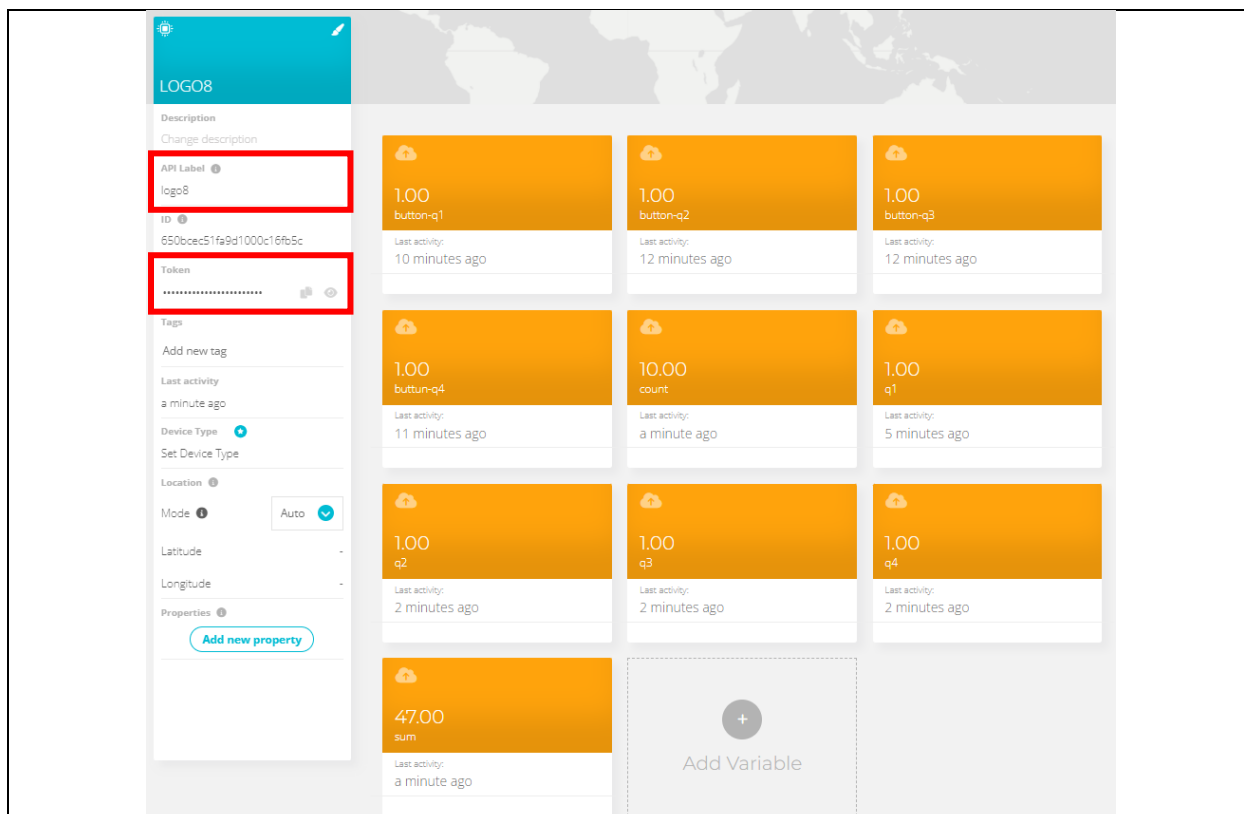
Token
BBFF-YBiZRFdoBhsJwC9F0H7yu7ctKEQe6J

Device Label
logo8

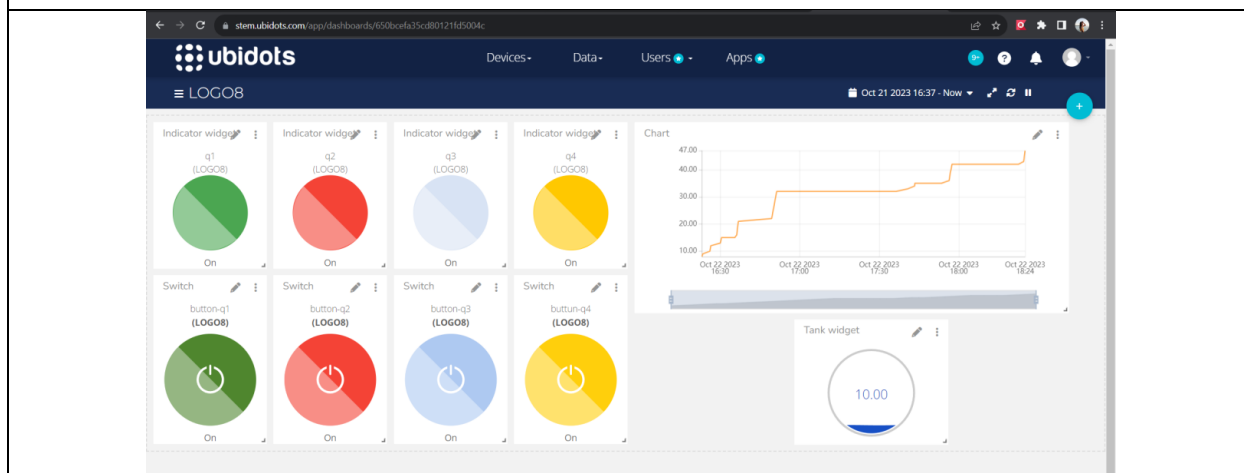
☒ Enable secure TLS connection

### 3.3.4. Ubidosh

ส่วนที่ใช้ในการเชื่อมต่อกันระหว่าง Node-Red กับ Ubidosh คือ API Key และ Token Key



- ทำการสร้างหน้า Dashboards

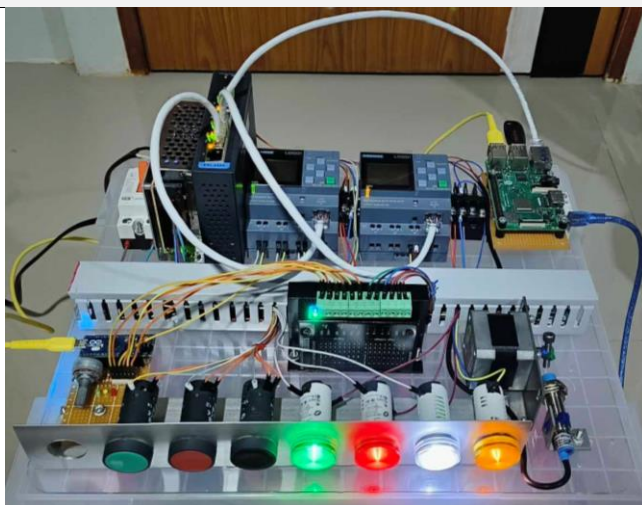
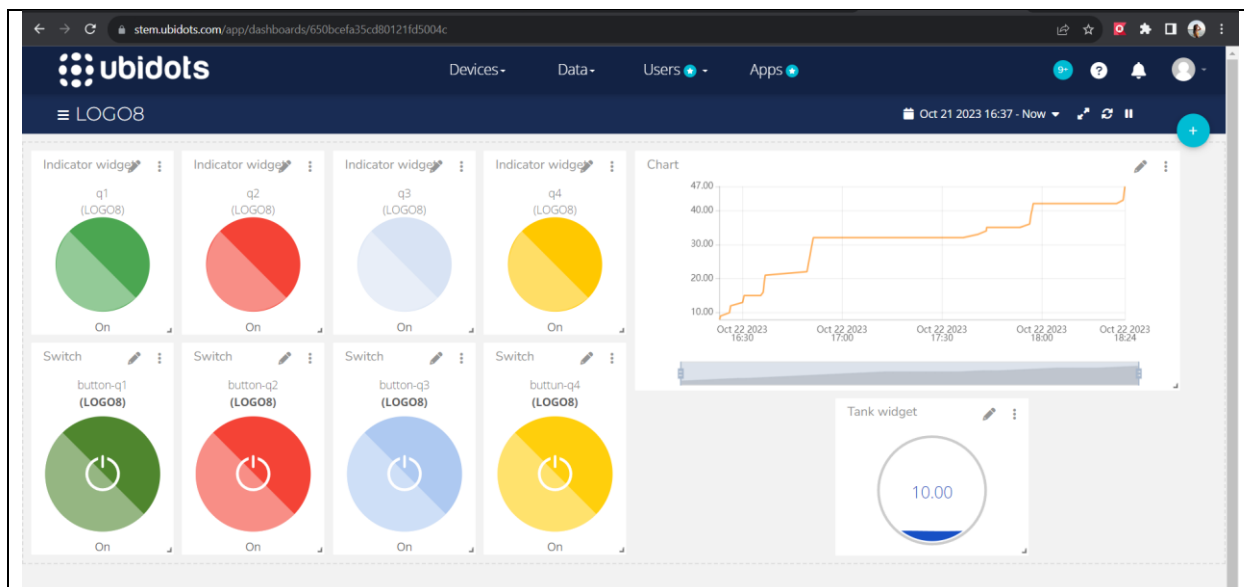


## บทที่ 4

### ผลการดำเนินงาน

#### 4.1. ผลลัพธ์ของโครงการงาน

##### 4.1.1. แสดงผลการดำเนินการ ที่ 1

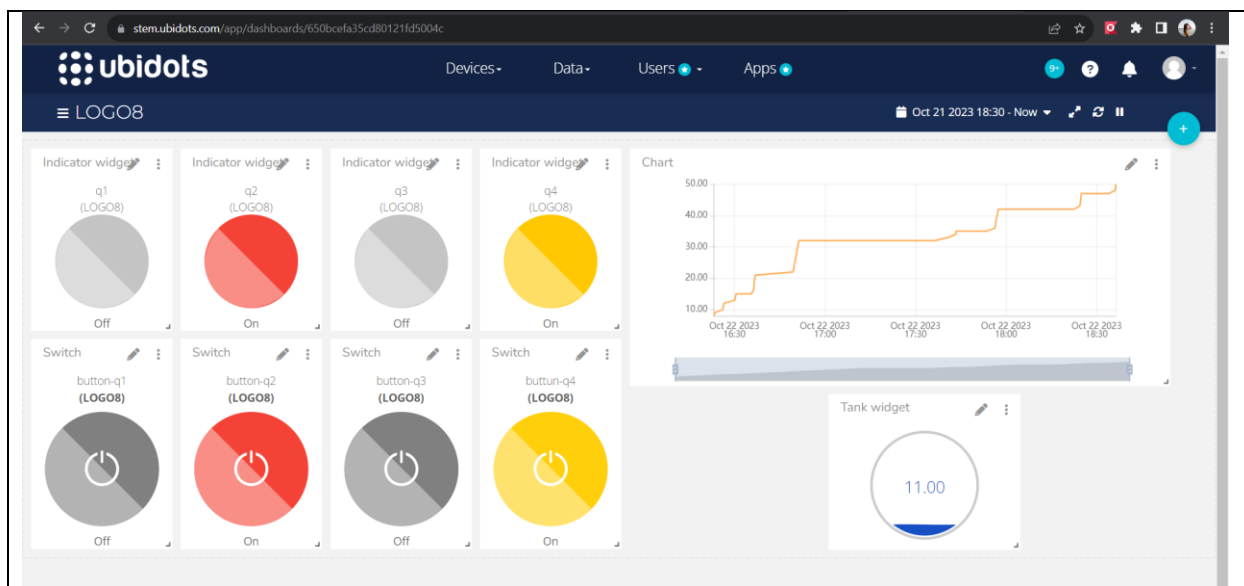


Q1, Q2, Q3 และ Q4  
มีสถานะ ON



LOGO! ตัวที่ 1 แสดงมีสถานะของ Coil  
Q1:On, Q2:On, Q3:On  
LOGO! ตัวที่ 2 แสดงค่าของ Count และ  
Sum  
Count: 10, Sum: 47

### 4.1.2. แสดงผลการดำเนินการ ที่ 2



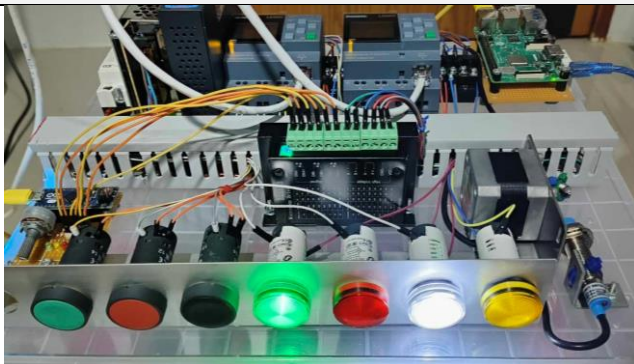
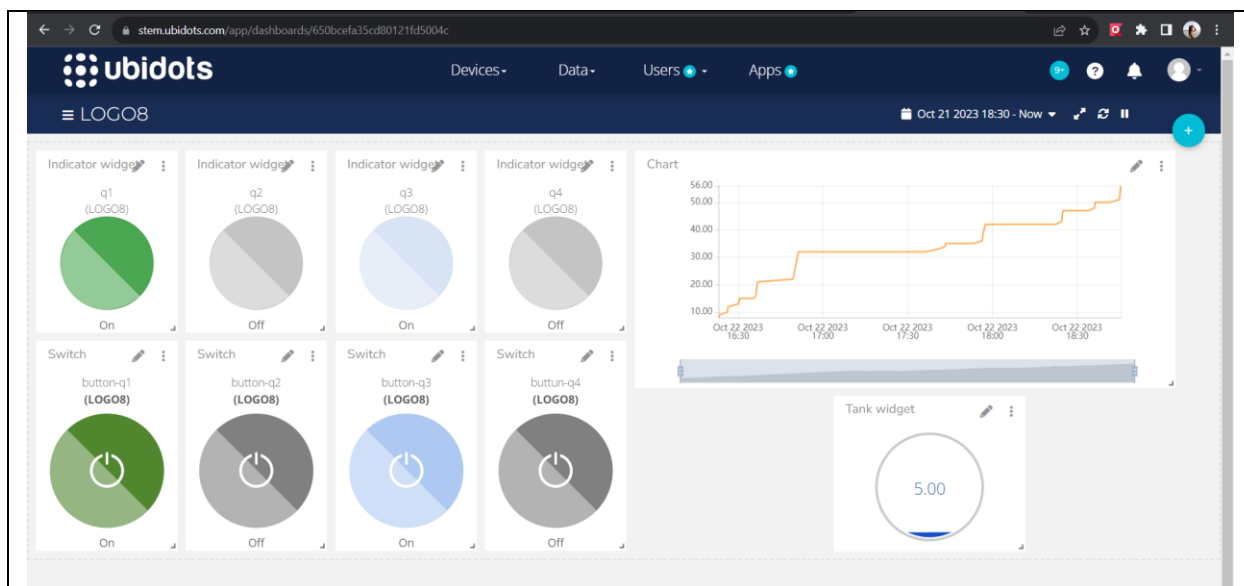
Q1, Q3 มีสถานะ Off  
Q2, Q4 มีสถานะ On



LOGO! ตัวที่ 1 แสดงมีสถานะของ Coil  
Q1:Off, Q2:On, Q3:Off  
LOGO! ตัวที่ 2 แสดงค่าของ Count และ  
Sum  
Count: 11, Sum: 50



### 4.1.3. แสดงผลการดำเนินการ ที่ 3



Q1, Q3 มีสถานะ On  
Q2, Q4 มีสถานะ Off



LOGO! ตัวที่ 1 แสดงมีสถานะของ Coil  
Q1:On, Q2:Off, Q3:On  
LOGO! ตัวที่ 2 แสดงค่าของ Count และ Sum  
Count: 5, Sum: 56

### 4.1.4. วิดีโอแสดงผลการดำเนินการ

<https://www.youtube.com/watch?v=Ka8ClIA3toU>

## 4.2. สรุปและอภิปรายผลการทดลอง

จากผลการดำเนินงานพบว่าการประยุกต์ใช้ IoT ในด้านอุตสาหกรรม ซึ่งคือ IIoT สามารถช่วยอำนวยความสะดวกให้กับผู้ใช้งานได้เป็นอย่างดี สามารถควบคุม และมอนิเตอร์ดูสถานการณ์ทำงาน หรือข้อมูลต่างๆผ่านทางระบบอินเทอร์เน็ต เพื่อเพิ่มประสิทธิภาพให้สามารถควบคุมและติดตามผลได้ตลอดเวลา

## 4.3. การพัฒนาต่อยอด

4.3.1. พัฒนาต่อยอดในด้านการเก็บข้อมูล เช่นข้อมูลเซ็นเซอร์ต่างๆที่เรารับเข้ามา สามารถนำข้อมูลนั้นไปใช้งานต่อได้ ยกตัวอย่างเช่นข้อมูลเซ็นเซอร์นับจำนวนสินค้าที่มาจากกระบวนการผลิต สามารถนำไปวิเคราะห์หาอัตราการผลิต คุณภาพของเครื่องจักรเพื่อเพิ่มประสิทธิภาพการบำรุงรักษาได้



## บทที่ 5

### ปัญหา และข้อเสนอแนะ

#### 5.1. ปัญหาที่พบ

- 5.1.1. ฮาร์ดแวร์บางตัวเป็นรุ่นใหม่ในการใช้งาน จึงใช้เวลาในการศึกษาวิธีใช้งานค่อนข้างนาน เพื่อให้ใช้งานได้อย่างถูกต้อง ไม่ก่อให้เกิดความเสียหาย
- 5.1.2. ใช้เวลาศึกษาเพื่อหาวิธีที่ LOGO! PLC ส่งและรับข้อมูลจากภายนอกในหลายๆวิธี เพื่อหาวิธีที่ดีที่สุด
- 5.1.3. ใช้เวลาศึกษาวิธีที่ LOGO! PLC ส่งและรับข้อมูลจากภายนอกผ่าน Modbus TCP ค่อนข้างนาน
- 5.1.4. ระบบอินเทอร์เน็ตที่ใช้ขณะทำโครงการมีความไม่เสถียร

#### 5.2. ข้อเสนอแนะ

- 5.2.1. การออกแบบระบบที่นำไปใช้งานจริงมีปัจจัยมากมายที่ต้องพิจารณา ทั้งในด้านของโปรแกรมการทำงาน หรือการเลือกใช้อุปกรณ์ ดังนั้นอาจมีการปรับปรุงหรือขยายแนวคิดบางส่วนเพื่อให้ตรงกับความต้องการของโครงการในอนาคตได้
- 5.2.2. ระบบอินเทอร์เน็ตควรใช้ที่มีความเร็วที่สม่ำเสมอ

### บรรณานุกรม

- [1] Sogoodweb: Internet Of Things (IoT) คืออะไร มาหาคำตอบกัน. [ออนไลน์]. เข้าถึงได้จาก:  
[https://blog.sogoodweb.com/Article/Detail/59554/Internet-of-Things-\(IoT\)-](https://blog.sogoodweb.com/Article/Detail/59554/Internet-of-Things-(IoT)-)
- [2] Factomart: Industrial Internet Of Things (IIoT) คืออะไร?. [ออนไลน์]. เข้าถึงได้จาก:  
<https://mall.factomart.com/what-is-industrial-internet-of-things/>
- [3] Mostori: MQTT กับระบบ IoT. [ออนไลน์]. เข้าถึงได้จาก: <https://www.mostori.com/blog93.html>
- [4] Scimath: Raspberry Pi คอมพิวเตอร์ขนาดเล็กสำหรับด้านการศึกษา. [ออนไลน์]. เข้าถึงได้จาก:  
<https://www.scimath.org/article-technology/item/9104-raspberry-pi>
- [5] mall.factomart: หลักการทำงานของ Inductive Proximity Sensor เซ็นเซอร์ตรวจจับโลหะ . [ออนไลน์].  
เข้าถึงได้จาก: <https://mall.factomart.com/inductive-proximity-sensor-working-principle/>
- [6] sonicautomation: การสื่อสารผ่าน Modbus TCP/IP [ออนไลน์]. เข้าถึงได้จาก:  
<https://sonicautomation.co.th/สื่อสารผ่าน-modbus-tcp-ip/>
- [7] SIEMENS Mini PLC Controller LOGO!8 [ออนไลน์]. เข้าถึงได้จาก:  
<https://shop.ibcon.com/th/product/722490/LOGO8>
- [8] Installing and Upgrading Node-RED [ออนไลน์]. เข้าถึงได้จาก:  
<https://nodered.org/docs/getting-started/raspberrypi>