

โครงการวิศวกรรมไฟฟ้า
ภาควิชาวิศวกรรมไฟฟ้า
คณะวิศวกรรมศาสตร์
มหาวิทยาลัยเกษตรศาสตร์

เรื่อง
หุ่นยนต์เพื่อการเกษตร
Agriculture Robot

โดย

นายสิรภัทร บุญจันทร์ 5910501127

นายอัครวิทย์ พงศ์วิรัตน์ 5910501178

นายสุทธิพงศ์ สว่าง 5910503341

พ.ศ. 2562

ที่นี่เป็นตัวเพื่อการเกษตร

Agriculture Robot

โดย

นายสิรภัทร บุญจันทร์ 5910501127

นายอัครวิทย์ พงศ์วิรัตน์ 5910501178

นายสุทธิพงษ์ สว่าง 5910503341

โครงการวิศวกรรมไฟฟ้า

ภาควิชาวิศวกรรมไฟฟ้า

คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเกษตรศาสตร์

ตามหลักสูตร

วิศวกรรมศาสตร์บัณฑิต

สาขาวิศวกรรมไฟฟ้า

ได้รับการพิจารณาเห็นชอบโดย

อาจารย์ที่ปรึกษาโครงการวันที่.....เดือน.....พ.ศ.....

(อาจารย์ปัญญา เหล่าอนันต์ธน)

กรรมการ.....วันที่.....เดือน.....พ.ศ.....

(รศ.ดร.พีระยศ แสนโภชน์)

ชื่อ – สกุล นายศิรภัทร บุญจันทร์ ปีการศึกษา 2562
ชื่อ – สกุล นายอัครวิทย์ พงศ์วิรัตน์ ปีการศึกษา 2562
ชื่อ – สกุล นายสุทธิพงศ์ สว่าง ปีการศึกษา 2562

หุ่นยนต์เพื่อการเกษตร

ปริญญาวิศวกรรมศาสตร์บัณฑิต (สาขาวิศวกรรมไฟฟ้า) ภาควิชาวิศวกรรมไฟฟ้า

คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์

บทคัดย่อ

โครงการนี้มีวัตถุประสงค์ในการนำเทคโนโลยีที่คณะผู้จัดทำได้ศึกษาจากสื่อต่างๆ มาประกอบเข้ากับ การเกษตรเพื่อสร้างเป็นหุ่นยนต์เพื่อการเกษตร ณ ปัจจุบัน การนำศาสตร์ของวิศวกรรมมาประกอบกับ การเกษตรนั้นยังไม่เป็นที่นิยมมากนัก อีกทั้งปัจจุบันการใช้ทรัพยากรต่างๆนั้นยังสามารถพัฒนาเพื่อเพิ่ม ประสิทธิภาพไปได้อีก คณะผู้จัดทำได้เล็งเห็นถึงความสามารถที่จะนำศาสตร์ของวิศวกรรมที่ได้ศึกษามา ประกอบเข้ากับการเกษตรเพื่อเพิ่มประสิทธิภาพเกิดเป็น นวัตกรรมใหม่เพื่อการเกษตร โดยหุ่นยนต์เพื่อ การเกษตรนี้จะทำหน้าที่เก็บเกี่ยวและดูแลรักษาต้นไม้ผ่านการควบคุมระยะไกลโดย MQTT Protocol ส่งไป ประมวลผลที่ Micro Controller และส่งไปควบคุมการทำงานของ DC Motor โดยหลักการของ PID Control ซึ่งทั้งหมดนี้จะทำให้สามารถเก็บเกี่ยวผลผลิตและดูแลรักษาได้อย่างแม่นยำและถูกวิธีผ่านการประมวลผลของ Micro Controller ทั้งหมดนี้เพื่อเป็นการนำเทคโนโลยีที่มือyu มาปรับใช้และอำนวยความสะดวกให้กับ บุคคลากรทางการเกษตร

คำสำคัญ MQTT Protocol, DC Motor, Micro Controller, PID Control

เลขที่เอกสารอ้างอิงภาควิชา E5036-PYL-1-2562

Boonchan Siraphat (2019)

Pongvirat Akaravit (2019)

Sawang Suttipong (2019)

Agriculture Robot

Bachelor Degree in Electrical Engineering, Department of Electrical Engineering

Faculty of Engineering, Kasetsart University

Abstract

Purposes of this project is to make a technology that our team had learned assemble with agriculture lead to Agriculture Robot. Nowadays, Agriculture Engineering is not that popular furthermore efficiency of resource that used in agriculture can be improved. Our team see an opportunity to improve agriculture with engineering that we had learned and make it into an innovation for agriculturist. Our robot will help to collect and take care trees remotely control by MQTT Protocol send command to Micro Controller calculate and control DC Motor by using PID control to precise and fast collet and take care trees. In conclusion, we hope to facilitate agriculturist with our engineering.

Keywords: MQTT Protocol, DC Motor, Micro Controller, PID Control

กิตติกรรมประกาศ

วิทยานิพนธ์หัวข้อเรื่องหุ่นยนต์เพื่อการเกษตร (Agriculture Robot) สามารถสำเร็จลุล่วงไปได้ด้วยดีตามความคาดหมายของคณัผู้จัดทำ คณัผู้จัดทำขอกราบขอบพระคุณ อาจารย์ปัญญา เหล่าอนันต์ธนา อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก และ รศ.ดร.พีระยศ แสน่โภชນ์ อาจารย์ที่ปรึกษาวิทยานิพนธ์ร่วม ที่กรุณาให้ความรู้ คำแนะนำต่างๆ ในทุกขั้นตอนการดำเนินงาน รวมไปถึงการจัดหาอุปกรณ์ต่างๆ และการตรวจสอบแก้ไขปัญหาที่เกิดระหว่างการปฏิบัติงานต่างๆ จนสำเร็จลุล่วงไปด้วยดี

มากไปกว่านี้ต้องขอขอบพระคุณทีมงาน อาจารย์ปัญญา เหล่าอนันต์ธนา ที่ได้ให้คำแนะนำ และสนับสนุนในทุกขั้นตอนระหว่างการปฏิบัติงานทำให้ผลการดำเนินงานเป็นไปอย่างราบรื่น และสุดท้ายนี้ขอกราบขอบพระคุณบิดา มารดา และอาจารย์ทุกท่าน ที่ให้คำแนะนำ สนับสนุน และเป็นกำลังใจตลอดการทำวิทยานิพนธ์ฉบับนี้

นายสิรภัทร บุญจันทร์

นายอัครวิทย์ พงศ์วิรัตน์

นายสุทธิพงศ์ สว่าง

ผู้จัดทำ

สารบัญ

สารบัญ.....	VI
สารบัญภาพ	IX
สารบัญตาราง	XI
1. บทนำ	1
1.1 วัตถุประสงค์ของโครงการ	1
1.2 ขอบเขตของโครงการ	1
2. ทฤษฎีที่เกี่ยวข้อง	2
2.1. ข้อมูลเบื้องต้นของหุ่นยนต์เพื่อการเก็บทรัพยากราก	2
2.1.1 โครงสร้างพื้นฐานทางกายภาพของหุ่นยนต์	2
2.2. ข้อมูลเบื้องต้นเกี่ยวกับไมโครคอนโทรลเลอร์ (MICRO CONTROLLER) ^[1]	3
2.2.1. ข้อมูลที่เกี่ยวข้องกับบอร์ด Arduino	3
2.3. ความรู้เบื้องต้นเกี่ยวกับอินเทอร์เน็ตในทุกสิ่ง (INTERNET OF THINGS)	4
2.3.1. ที่มาของการสื่อสารแบบ MQTT Protocol	5
2.3.2. หลักการทำงานของ MQTT ^[9]	5
2.4. การควบคุมมอเตอร์ไฟฟ้ากระแสตรง	6
2.4.1. หลักการทำงานเบื้องต้นของมอเตอร์ไฟฟ้ากระแสตรง	6
2.4.2. หลักการทำงานของอุปกรณ์ที่ใช้ในการควบคุมมอเตอร์ไฟฟ้ากระแสตรง	6
2.4.3. หลักการทำงานของอุปกรณ์เข้ารหัส (Encoder)	7
2.5. ระบบการควบคุม (CONTROL SYSTEM)	7
2.5.1. ระบบควบคุมแบบสัดส่วน-ปริพันธ์-อนุพันธ์ (PID Controller)	8
2.5.2. วิธีการคำนวณสัดส่วน-ปริพันธ์-อนุพันธ์	8
2.5.3. การพิสูจน์สมการ ^[6]	9
3. เครื่องมือที่ใช้ในการทำโครงการ	10
3.1. อุปกรณ์ทางด้าน HARDWARE	10
3.1.1. Arduino Board	10
3.1.2. Ethernet Shield	10
3.1.3. DC Motor	11
3.1.4. Encoder Shield	12
3.1.5. DC Motor Drives ชนิด H-Bridge	12

3.1.6	Encoder	13
3.1.7	Switching Hub	13
3.1.8	Variable Resistor	14
3.1.9	DC Supply	14
3.2.	อุปกรณ์ทางด้าน SOFTWARE.....	15
3.2.1	Arduino IDE	15
3.2.2	MQTT Box.....	15
4.	วิธีการดำเนินโครงการ	16
4.1	ขั้นตอนการใช้งาน ARDUINO [7]	16
4.2	เรียนรู้หลักการทำงานของ MOTOR DRIVE	22
4.3	วิธีการสื่อสารของระบบ MQTT	23
4.4	วิธีการควบคุมหุ่นยนต์	28
	4.4.1 ควบคุมแบบกำหนดเอง.....	28
	4.4.2 ควบคุมแบบอัตโนมัติ	28
5.	ผลการดำเนินโครงการและวิจารณ์	29
5.1	การติดตั้งอุปกรณ์	29
5.1.1	กล่องอุปกรณ์ควบคุม	30
5.1.2	DC Motor	31
5.1.3	แกนอุปกรณ์	31
5.2	การทำงาน	34
5.3	การควบคุมและแสดงผล.....	35
	5.3.1 การหาพิกัด.....	35
5.4	วิจารณ์ผลการดำเนินงานที่ได้	36
6.	สรุปผลการดำเนินงานและข้อเสนอแนะ	37
6.1	การใช้งานและควบคุม CONTROLLER	37
6.2	การควบคุมมอเตอร์ แบบ FEEDBACK (PID-CONTROLLER)	37
6.3	ควบคุมเครื่องจักรจากทางไกลได้	37
6.4	ปัญหาและอุปสรรค	37
6.5	แนวทางการแก้ไขในอนาคต.....	38
6.6	สิ่งที่คณะผู้จัดทำได้เพิ่มเติมในโครงการ	38
7.	บรรณานุกรม	39

ภาคผนวก.....	40
SLAVE1_SPIDERBOTS' CODE.....	40
SLAVE2_SPIDERBOTS' CODE.....	54
SLAVE3_SPIDERBOTS' CODE.....	62
SLAVE4_SPIDERBOTS' CODE.....	70
TOOLS_AGRICULTURE'S CODE	77
ประวัตินิสิต.....	83

สารบัญภาพ

รูปที่ 1 แบบจำลองโครงการ	2
รูปที่ 2 โครงสร้างชิ้นงาน	3
รูปที่ 3 บอร์ด ARDUINO รุ่น MEGA2560	4
รูปที่ 4 องค์ประกอบโดยรวมของระบบ INTERNET OF THINGS	4
รูปที่ 5 หลักการทำงานของ MQTT	5
รูปที่ 6 หลักการทำงานของอุปกรณ์ควบคุมมอเตอร์กระแสตรงชนิด H-BRIDGE	6
รูปที่ 7 อุปกรณ์ ENCODER และกลไกการทำงานภายใน	7
รูปที่ 8 ระบบควบคุมแบบ PID CONTROLLER	8
รูปที่ 9 การคำนวณระยะของสาย	9
รูปที่ 10 ARDUINO MEGA2560	10
รูปที่ 11 ETHERNET SHIELD	10
รูปที่ 12 DC MOTOR	11
รูปที่ 13 DC MOTOR ของอุปกรณ์แกนกลาง	11
รูปที่ 14 ENCODER SHIELD	12
รูปที่ 15 DC MOTOR DRIVES แบบ H-BRIDGE ทั้ง 2 แบบ	12
รูปที่ 16 ENCODER	13
รูปที่ 17 SWITCHING HUB	13
รูปที่ 18 VARIABLE RESISTOR	14
รูปที่ 19 DC SUPPLY	14
รูปที่ 20 ARDUINO IDE	15
รูปที่ 21 MQTT Box	15
รูปที่ 22 หน้าเว็บเพจของ ARDUINO IDE	16
รูปที่ 23 วิธีการ DOWNLOAD ARDUINO IDE	16
รูปที่ 24 ARDUINO IDE APPLICATION	17
รูปที่ 25 การเปิด DEVICE MANAGER	17
รูปที่ 26 DEVICE MANAGER	18
รูปที่ 27 การเชื่อมต่อ ARDUINO	18
รูปที่ 28 การ COMPILE PROGRAM	19
รูปที่ 29 แสดงผลการ COMPILE	20

รูปที่ 30 การ UPLOAD เข้าบอร์ด.....	20
รูปที่ 31 แสดงผลการ UPLOAD	21
รูปที่ 32 MOTOR DRIVE	22
รูปที่ 33 MQTTBox.....	23
รูปที่ 34 สร้าง CLIENT MQTT	24
รูปที่ 35 CLIENT SETTING	24
รูปที่ 36 CLIENT INTERFACE.....	25
รูปที่ 37 TOPIC MQTT	26
รูปที่ 38 PUBLISH MESSAGE.....	26
รูปที่ 39 FLOWCHART การเรียนรู้	27
รูปที่ 40 โครงสร้างของชิ้นงาน	29
รูปที่ 41 โครงสร้างภายในกล่องอุปกรณ์.....	30
รูปที่ 42 อุปกรณ์ต่างๆ	30
รูปที่ 43 ติดตั้งDC MOTOR	31
รูปที่ 44 แกนการเคลื่อนที่ของอุปกรณ์	31
รูปที่ 45 แกนขึ้นลง	32
รูปที่ 46 แกนหมุนอุปกรณ์	32
รูปที่ 47 แกนเบย์ขึ้นหรือกดลง	33
รูปที่ 48 แกนหมุนตัวอุปกรณ์	33
รูปที่ 49 การเคลื่อนที่ของอุปกรณ์.....	34
รูปที่ 50 อุปกรณ์แกนกลาง.....	34
รูปที่ 52 การคำนวณ POSITION.....	35
รูปที่ 53 การส่งค่า SET POINT	36

สารบัญ

ตารางที่ 1 LOGIC การทำงานของMOTOR DRIVE 22

1. บทนำ

ปัจจุบันการเกษตรถือว่าเป็นส่วนสำคัญของประเทศไทย ซึ่งการเก็บเกี่ยวและดูแลรักษาข้าวอาจก่อให้เกิดความเสียหายได้หากทำอย่างผิดวิธี ดังนั้นจึงเกิดเป็นโครงการนี้ขึ้นมาเพื่อที่จะนำเทคโนโลยีต่าง ๆ มาปรับใช้เพื่ออำนวยความสะดวกในการดูแลและเก็บเกี่ยวผลผลิต อีกทั้งยังแม่นยำและเชื่อถือได้ เพราะเป็นการควบคุมผ่าน Micro Controller

1.1 วัตถุประสงค์ของโครงการ

1. เพื่อพัฒนาระบบการเกษตรให้มีประสิทธิภาพมากยิ่งขึ้น
2. เพื่อเพิ่มประสิทธิภาพในการเก็บเกี่ยวผลผลิตและดูแลรักษา
3. อำนวยความสะดวกให้กับเกษตรกรด้วยเทคโนโลยีที่มีอยู่

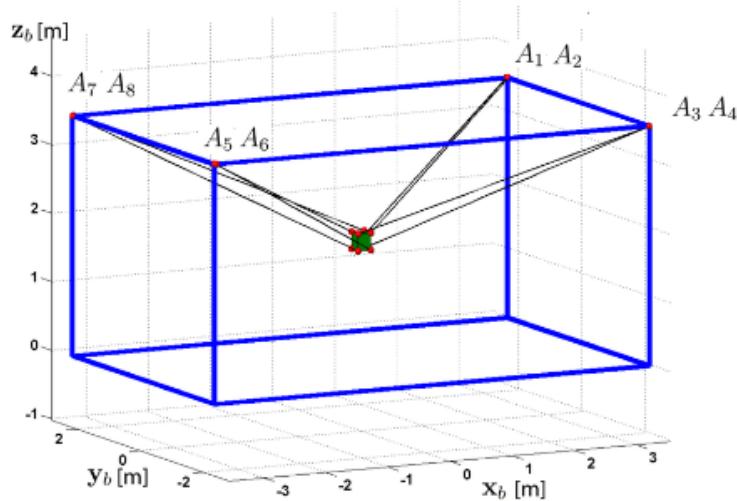
1.2 ขอบเขตของโครงการ

1. สามารถเก็บเกี่ยวผลผลิตในพื้นที่ได้
2. แสดงผลจากข้อมูลที่ป้อนได้อย่างถูกต้อง
3. อุปกรณ์สามารถเคลื่อนที่ได้อย่างราบรื่นและสอดคล้อง

2. ทฤษฎีเกี่ยวข้อง

2.1. ข้อมูลเบื้องต้นของหุ่นยนต์เพื่อการเกษตร

หุ่นยนต์ที่ค่อยควบคุมชุดเครื่องมือทางการเกษตรให้สามารถเคลื่อนที่ได้อย่างอิสระในแนว 3 มิติ มีหลัก planetary ฟังก์ชันการใช้งานเช่น ลดนำ้ ตัดแต่งกิ่ง ฯลฯ คาดว่าจะช่วยพัฒนาการเกษตรของประเทศไทยให้มีประสิทธิภาพมากยิ่งขึ้น โดยหุ่นยนต์นี้ใช้ระบบควบคุมแบบเชิงเส้น (Linear Control) ซึ่งมีกลไกที่ไม่ซับซ้อน สามารถนำไปเรียนรู้และต่อ�อดได้มากขึ้น



รูปที่ 1 แบบจำลองโครงงาน

2.1.1 โครงสร้างพื้นฐานทางกายภาพของหุ่นยนต์

- เสาหลัก มีหน้าที่ในการยึดโครงสร้างทั้งหมดให้มีความมั่นคงและแข็งแรง
- ระบบส่งกำลัง ประกอบด้วย ชุดมอเตอร์ไฟฟ้ากระแสตรง และ เพลาแกนหมุน
- ชุดเครื่องมือทางการเกษตร สามารถปรับเปลี่ยนได้ขึ้นอยู่กับลักษณะงานที่ใช้
- เข็อกสลิ่ม เป็นส่วนที่เชื่อมโยงระหว่างระบบส่งกำลังกับชุดเครื่องมือทางการเกษตร
- แผงควบคุม ภายในประกอบด้วยอุปกรณ์ต่างๆ ที่จำเป็นสำหรับการควบคุมกลไกการทำงาน



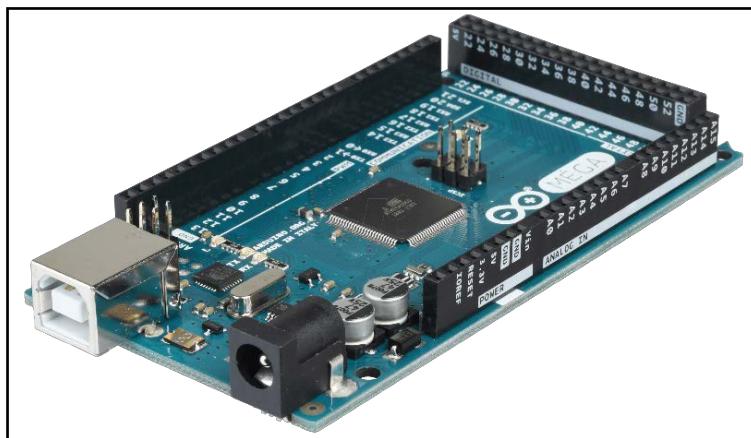
รูปที่ 2 โครงสร้างขั้นงาน

2.2. ข้อมูลเบื้องต้นเกี่ยวกับไมโครคอนโทรลเลอร์ (Micro Controller)^[1]

ไมโครคอนโทรลเลอร์ (Microcontroller) มาจากคำ 2 คำ ไมโคร (Micro) หมายถึง ขนาดเล็กและคำว่า คอนโทรลเลอร์ (Controller) หมายถึง อุปกรณ์ควบคุม ดังนั้นไมโครคอนโทรลเลอร์จึงหมายถึงอุปกรณ์ควบคุม ขนาดเล็ก แต่ภายในอุปกรณ์ควบคุมขนาดเล็กนี้ ได้บรรจุความสามารถหลากหลาย เช่น หน่วยอินพุตเอาต์พุต (I/O) หน่วยความจำ(Memory) ประมวลผลการทำงานต่างๆ โครงงานครั้งนี้ เลือกที่จะใช้คอนโทรลเลอร์ที่ชื่อว่า Arduino

2.2.1. ข้อมูลที่เกี่ยวข้องกับบอร์ด Arduino

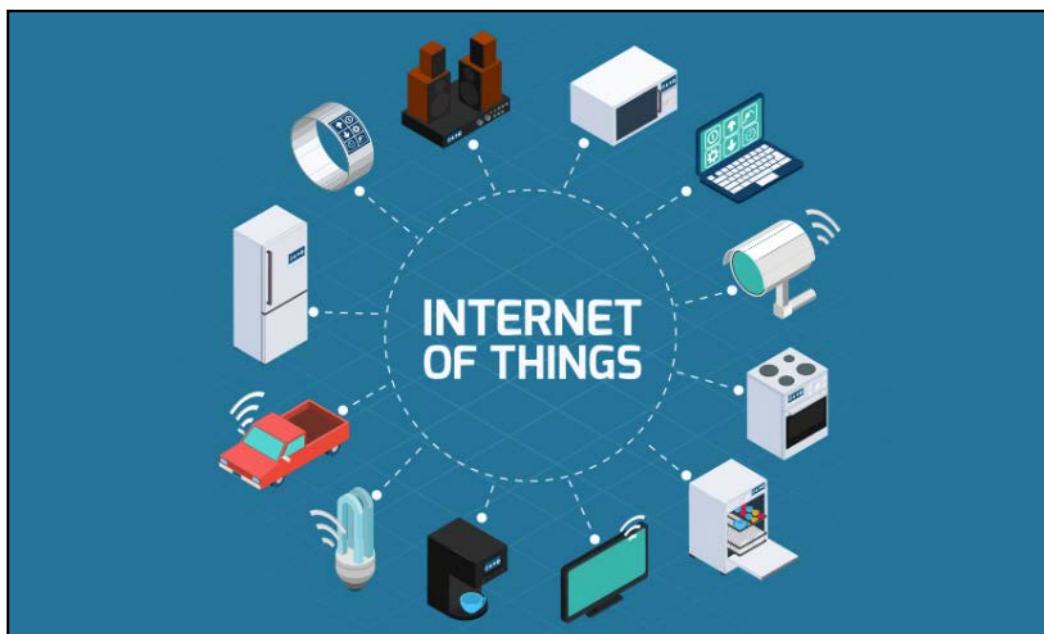
Arduino ได้ออกแบบมาเพื่อให้ใช้งานง่าย ผู้ใช้งานไม่จำเป็นต้องมีความรู้เกี่ยวกับโครงภายในซีพียู โดยรู้ เพียงว่า บอร์ด Arduino ที่เลือกมาใช้งานนั้นมีขาที่ใช้งานอะไรบ้าง มีคุณสมบัติพิเศษอย่างไร เพียงเท่านี้ก็สามารถ ใช้งานได้แล้ว ด้วยประสบการณ์การใช้งานของผู้ใช้งานจำนวนมาก Arduino จึงถูกใช้งานด้านต่างๆ มากมาย การเขียน โปรแกรมควบคุมการทำงานของ Arduino มีความง่ายและยึดหยุ่น สามารถนำไปต่ออยอดและใช้งานในระดับสูงได้ อีกด้วย



รูปที่ 3 บอร์ด Arduino รุ่น Mega2560

2.3. ความรู้เบื้องต้นเกี่ยวกับอินเทอร์เน็ตในทุกสิ่ง (Internet of Things)

อินเทอร์เน็ตในทุกสิ่ง (Internet of Things) หรือที่คนส่วนใหญ่เรียกว่า IOT มีชื่อเรียกอีกอย่างว่า M2M ซึ่งย่อมาจาก Machine to Machine คือ การที่อุปกรณ์ สิ่งต่างๆ ได้ถูกเชื่อมโยงทุกสิ่งทุกอย่างสู่โลกอินเตอร์เน็ต ทำให้มนุษย์สามารถสั่งการควบคุมการใช้งานอุปกรณ์ต่างๆ ผ่านทางเครือข่ายอินเตอร์เน็ตได้ การสื่อสารแบบ IOT มีหลายรูปแบบมาก ซึ่งในโครงงานครั้นนี้เลือกที่จะใช้การสื่อสารแบบ MQTT Protocol



รูปที่ 4 องค์ประกอบโดยรวมของระบบ Internet of Things

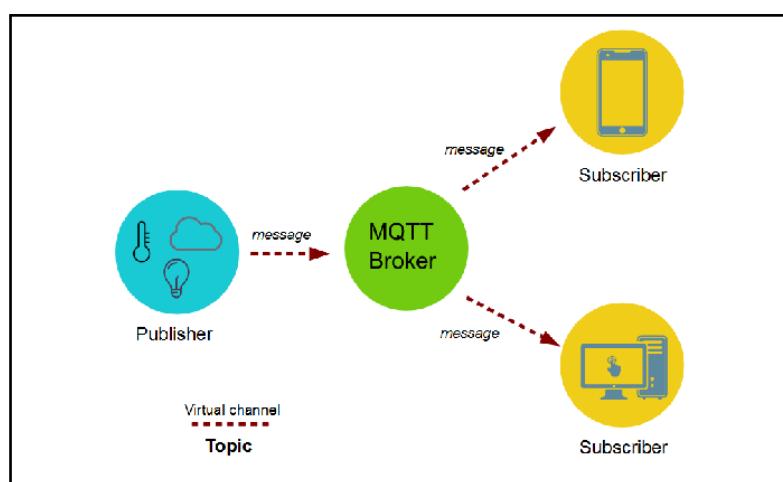
2.3.1. ที่มาของการสื่อสารแบบ MQTT Protocol

MQTT ย่อมาจาก Message Queue Telemetry Transport ออกแบบมาสำหรับการสื่อสารแบบเรียลไทม์แบบไม่จำกัดแพลตฟอร์ม หมายถึง อุปกรณ์ทุกชนิดสามารถสื่อสารผ่านกันได้ MQTT จะเป็น 2 ฝั่งคือ ฝั่งเซิร์ฟเวอร์มักจะเรียกว่า MQTT Broker ส่วนฝั่งผู้ใช้งานจะเรียกว่า MQTT Client ซึ่งแต่ละฝั่งจะมี Username และ Password เป็นของตัวเอง เพื่อที่จะระบุตัวตนในขณะใช้งานได้อย่างถูกต้องและแม่นยำ

2.3.2. หลักการทำงานของ MQTT^[9]

ลักษณะการใช้งาน MQTT อาจจะเปรียบเสมือนการใช้งานในห้องสนทนาระหว่างผู้คนโดยจะมีส่วนประกอบพื้นฐานคือ เส้นทาง (Topic) , คุณภาพข้อมูล (QoS) , การส่งข้อมูล (Publish) , การรับข้อมูล (Subscribe)

- เส้นทาง (Topic) เปรียบเสมือนหัวข้อหรือห้องสนทนาที่จะคุยกัน การคุยกันจะมีเฉพาะอุปกรณ์ที่อยู่ในห้องนั้น (Subscribe) ถึงจะสามารถรับข้อมูลที่มีการส่งในห้องนั้นได้
- คุณภาพข้อมูล (QoS) เปรียบเสมือนลักษณะของการส่งข้อมูล จะมี 3 แบบ คือ
 1. QoS0 ส่งข้อมูลเพียงครั้งเดียว ไม่สนใจว่าผู้รับจะได้รับหรือไม่
 2. QoS1 ส่งข้อมูลเพียงครั้งเดียว จะมีการจดจำข้อมูลที่ล่าสุดที่ส่งไป
 3. QoS2 ส่งข้อมูลหลายๆครั้งจนกว่าปลายทางจะได้รับข้อมูล
- การส่งข้อมูล (Publish) เป็นการส่งข้อมูลไปยัง Topic ที่ได้กำหนดไว้
- การรับข้อมูล (Subscribe) จะรับข้อมูลได้เฉพาะเมื่อมีการเรียก Subscribe ไปยัง Topic ที่กำหนด อาจเปรียบได้กับการเข้าที่เราเข้าร่วมไลน์กลุ่ม เมื่อมีเพื่อนส่งข้อความมาเราจะสามารถกดเข้าไปดูข้อความนั้นได้



รูปที่ 5 หลักการทำงานของ MQTT

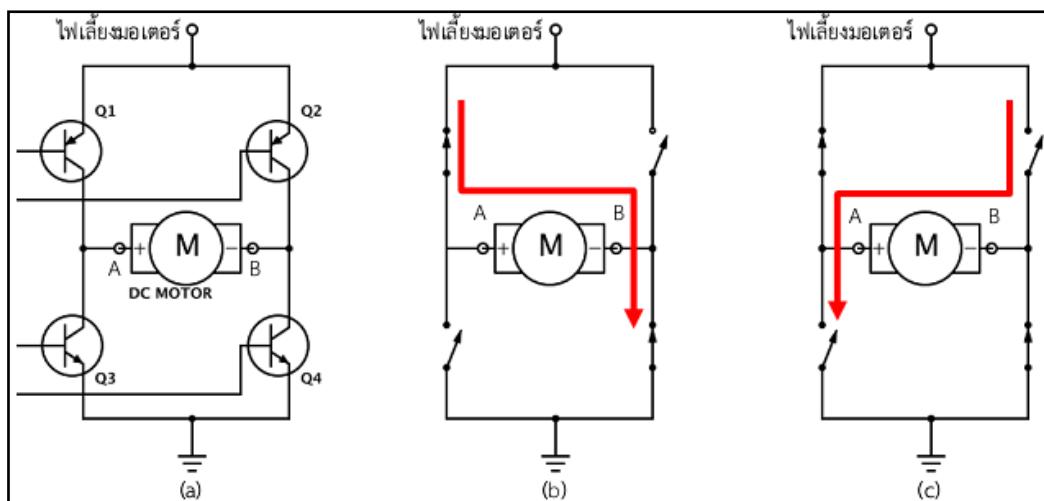
2.4. การควบคุมมอเตอร์ไฟฟ้ากระแสตรง

2.4.1. หลักการทำงานเบื้องต้นของมอเตอร์ไฟฟ้ากระแสตรง

มอเตอร์ไฟฟ้ากระแสตรง (DC Motor) เป็นอุปกรณ์ที่แปลงพลังงานไฟฟ้าให้เป็นพลังกล โครงสร้างภายใน DC motor ประกอบด้วย 2 ส่วนหลัก ได้แก่ แม่เหล็กถาวรและแกนขดลวด นอกจากนี้ยังมีแปรงถ่าน (Brush) ซึ่งเป็นส่วนเชื่อมต่อเพื่อรับพลังงานไฟฟ้าภายนอกไปยังขดลวดของมอเตอร์ เมื่อขดลวดได้รับไฟฟ้ากระแสตรง จะถูกเหนี่ยววนิให้เกิดสนามแม่เหล็กรอบๆ ขดลวด เมื่อเราต่อมอเตอร์กับแหล่งจ่ายไฟกระแสตรงภายนอกมอเตอร์จะเริ่มหมุนในทิศทางหนึ่ง หากเราต่อไฟลับขั้วของแหล่งจ่ายไฟเมอเตอร์จะหมุนในทิศตรงกันข้าม หากต้องการลดความเร็วของมอเตอร์ เราเพียงปรับแรงดันของแหล่งจ่ายไฟให้มีค่าลดลง

2.4.2. หลักการทำงานของอุปกรณ์ที่ใช้ในการควบคุมมอเตอร์ไฟฟ้ากระแสตรง

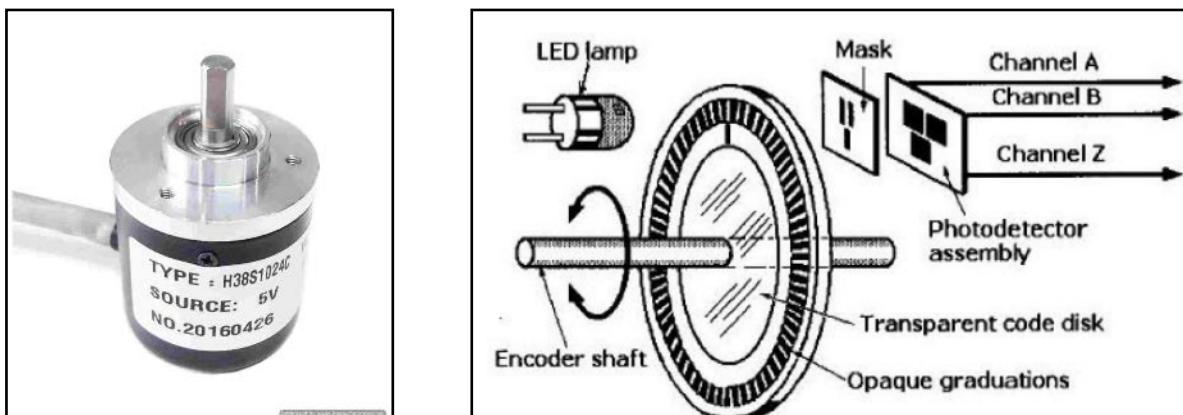
อุปกรณ์ที่ใช้ในการควบคุมการทำงานของมอเตอร์ไฟฟ้านิดที่นิยมใช้กันส่วนใหญ่และเกี่ยวข้องกับโครงงานครั้งนี้ คือ ชนิด H-Bridge ซึ่งจะประกอบด้วยทรานซิสเตอร์หรือมอตเฟส โดยทำหน้าที่เป็นสวิตช์เปิดปิดจำนวน 4 ชุด (Q1-Q4) โดยต่อ กับ DC Motor ดังรูปที่ 4a ซึ่งสามารถควบคุมการทำงานของมอเตอร์ได้ เมื่อส่งสัญญาณควบคุมให้ทรานซิสเตอร์ Q1 และ Q4 ทำงาน และปิดการทำงานของทรานซิสเตอร์ Q2 และ Q3 กระแสจะไหลจากจุด A ไปจุด B ดังรูปที่ 4b จึงทำให้มอเตอร์เริ่มหมุน และเมื่อส่งสัญญาณควบคุมให้ทรานซิสเตอร์ Q2 และ Q3 ทำงาน และปิดการทำงานของทรานซิสเตอร์ Q1 และ Q4 กระแสจะไหลจากจุด B ไปจุด A ดังรูปที่ 4c เป็นผลให้มอเตอร์หมุนกลับทิศ^[3]



รูปที่ 6 หลักการทำงานของอุปกรณ์ควบคุมมอเตอร์กระแสตรงชนิด H-Bridge

2.4.3 หลักการทำงานของอุปกรณ์เข้ารหัส (Encoder)

อุปกรณ์เข้ารหัส คือ เซ็นเซอร์ชนิดหนึ่งที่ทำหน้าที่ในการเข้ารหัส จากระยะทางจากการหมุนรอบตัวเอง และแปลงอุปกรณ์เป็นรหัสในรูปแบบของสัญญาณไฟฟ้า โดยความสามารถนำเอารหัสเหล่านี้มาแปลงกลับ เพื่อหาค่า ต่างๆ ที่เราต้องการได้ ไม่ว่าจะเป็นระยะทางการหมุน องศา การเคลื่อนที่ หรือ ความเร็ว รอบกีดี แล้วนำมาแสดงผล ให้เราได้ทราบค่าผ่านหน้าจอแสดงผล เช่น ถ้าต้องการวัดระยะทาง เราจะต้องต่อเข้ากับตัวนับจำนวน เพื่อแสดงผล เป็นระยะทาง หรือ ถ้าต้องการวัดความเร็วรอบ ก็นำระยะทางที่ได้มาหารด้วยเวลาใน 1 รอบ ส่วนการแสดงผลเป็น ความเร็วของ RPM, RPS โดยอาศัยสัญญาณที่ผ่านการเข้ารหัสแล้วอุปกรณ์เป็นสัญญาณทางไฟฟ้านั้น สามารถ แบ่งรูปแบบของการเข้ารหัสได้อีกหลากหลายรูปแบบ เช่น สัญญาณดิจิตอล ศูนย์กับหนึ่ง ธรรมชาติ หรือ เป็นแบบ Binary Code, BCD Code, Gray Code^[5]



รูปที่ 7 อุปกรณ์ encoder และกลไกการทำงานภายใน

2.5. ระบบการควบคุม (Control System)

การควบคุมระบบหรือสิ่งที่ผู้ออกแบบต้องการ ควบคุม ให้ได้ค่าผลลัพธ์ในรูปแบบของเอาท์พุตที่ต้องการซึ่ง ทำได้โดยการป้อนค่าอินพุตให้กับระบบโดย นิยามศัพท์พื้นฐานของระบบควบคุมมีดังนี้

1. อินพุต (Input) หมายถึง การสัญญาณเข้าที่ต้องการป้อนให้กับระบบรับรู้
2. ระบบ (System) หมายถึง สิ่งที่ต้องการควบคุมซึ่งจะประกอบด้วยชุด ควบคุมกระบวนการ (Process) เอาท์พุต (Output) หมายถึงผลของการทำงานของระบบที่ผ่านการควบคุมซึ่งจะแสดงในรูปแบบ ผลตอบสนอง ทางกล (Mechanical Response) และผลตอบสนองทางไฟฟ้า (Electrical Response)

2.5.1 ระบบควบคุมแบบสัตดส่วน-ปริพันธ์-อนุพันธ์ (PID Controller)

เป็นระบบควบคุมแบบป้อนกลับที่ใช้กันอย่างกว้างขวางซึ่งค่าที่นำไปใช้ในการคำนวณเป็นค่าความผิดพลาดที่นำมาจากความแตกต่างของตัวแปรในกระบวนการและค่าที่ต้องการตัวควบคุมจะพยายามลดค่าผิดพลาดให้เหลือน้อยที่สุดด้วยการปรับค่าสัญญาณขาเข้าของกระบวนการ ^[2]

2.5.2 วิธีการคำนวณสัตดส่วน-ปริพันธ์-อนุพันธ์

วิธีคำนวณของ PID ขึ้นอยู่กับสามตัวแปรคือค่าสัตดส่วน, ปริพันธ์ และ อนุพันธ์ ค่าสัตดส่วนกำหนดจากผลของความผิดพลาดในปัจจุบัน, ค่าปริพันธ์กำหนดจากผลบันทึกฐานของผลรวมความผิดพลาดที่ซึ่งพิ่งผ่านพ้นไป, และค่าอนุพันธ์กำหนดจากผลบันทึกฐานของอัตราการเปลี่ยนแปลงของค่าความผิดพลาด ผลรวมที่เกิดจากการรวมกันของทั้งสามนี้จะใช้ในการปรับกระบวนการ เป็นไปตามสมการ ^[8]

$$u(t) = MV(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{d}{dt} e(t)$$

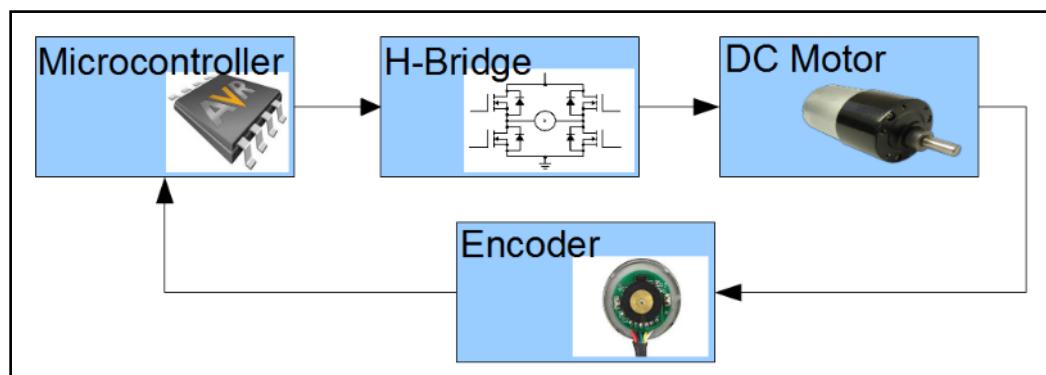
u(t) : เป็นสัญญาณขาออก

e(t) : ค่าความผิดพลาด

K_p : อัตราขยายสัตดส่วนตัวแปรปรับค่าได้

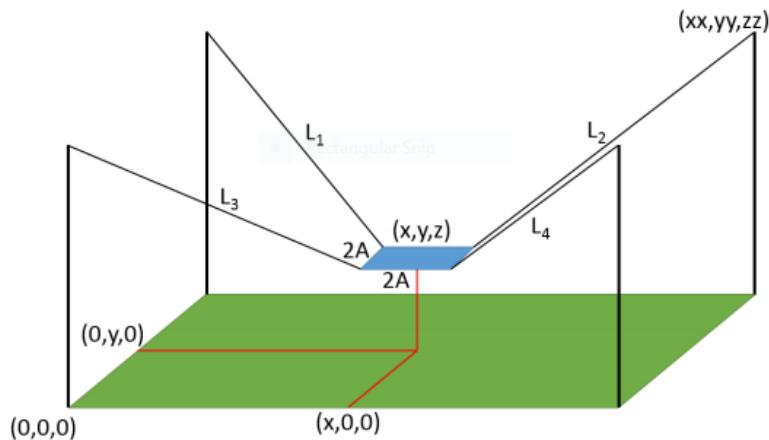
K_i : อัตราขยายปริพันธ์ ปรับค่าได้

K_d : อัตราขยายอนุพันธ์ปรับค่าได้



รูปที่ 8 ระบบควบคุมแบบ PID Controller

2.5.3 การพิสูจน์สมการ^[6]



รูปที่ 9 การคำนวณระยะของสาย

จาก ทฤษฎี สามเหลี่ยมพีทาโกรัส ทำให้สามารถหา ค่า L_1, L_2, L_3, L_4 จาก ตำแหน่ง X, Y, Z ที่ต้องการได้ ตาม สมการดังนี้

$$L_1 = \sqrt{(x - A)^2 + (yy - y - A)^2 + (zz - z)^2}$$

$$L_2 = \sqrt{(xx - x - A)^2 + (yy - y - A)^2 + (zz - z)^2}$$

$$L_3 = \sqrt{(x - A)^2 + (yy - A)^2 + (zz - z)^2}$$

$$L_4 = \sqrt{(xx - x - A)^2 + (yy - A)^2 + (zz - z)^2}$$

โดย ตำแหน่งสีฟ้า คือ จุดที่ติดตั้งเครื่องมือทางการเกษตร

เมื่อกำหนด

L_1 คือ ความยาวของสลิงจากจุดสูงสุดของเสาต้นที่ 1 ไปจนถึงจุดติดตั้งเครื่องมือทางการเกษตร

L_2 คือ ความยาวของสลิงจากจุดสูงสุดของเสาต้นที่ 2 ไปจนถึงจุดติดตั้งเครื่องมือทางการเกษตร

L_3 คือ ความยาวของสลิงจากจุดสูงสุดของเสาต้นที่ 3 ไปจนถึงจุดติดตั้งเครื่องมือทางการเกษตร

L_4 คือ ความยาวของสลิงจากจุดสูงสุดของเสาต้นที่ 4 ไปจนถึงจุดติดตั้งเครื่องมือทางการเกษตร

A คือ ขนาดพื้นที่หน้าตัดของเครื่องมือทางการเกษตร

xx คือ ความกว้างของโครงเหล็ก

yy คือ ความยาวของโครงเหล็ก

zz คือ ความสูงของโครงเหล็ก

X คือ ตำแหน่งพิกัดตามแกน x ของเครื่องมือทางการเกษตร

Y คือ ตำแหน่งพิกัดตามแกน y ของเครื่องมือทางการเกษตร

Z คือ ตำแหน่งพิกัดตามแกน z ของเครื่องมือทางการเกษตร

3. เครื่องมือที่ใช้ในการทำโครงงาน

3.1. อุปกรณ์ทางด้าน Hardware

3.1.1. Arduino Board

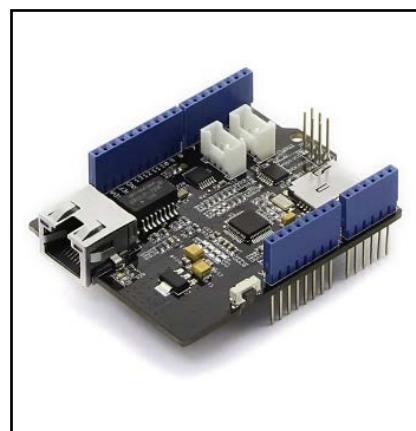
รุ่น Arduino Mega2560 ไฟฟ้าเลี้ยง 5VDC



รูปที่ 10 Arduino MEGA2560

3.1.2. Ethernet Shield

รุ่น W5500 Ethernet Shield ไฟฟ้าเลี้ยง 5VDC



รูปที่ 11 Ethernet Shield

3.1.3. DC Motor

รุ่น MY1016 บริษัท UDOM KLONG THOM ที่สปีดพิกัด 2750 rpm 350 W ใช้ไฟเลี้ยง 24VDC และรุ่น ZGX38REE 101i ที่สปีดพิกัด 50 rpm ใช้ไฟเลี้ยง 12VDC



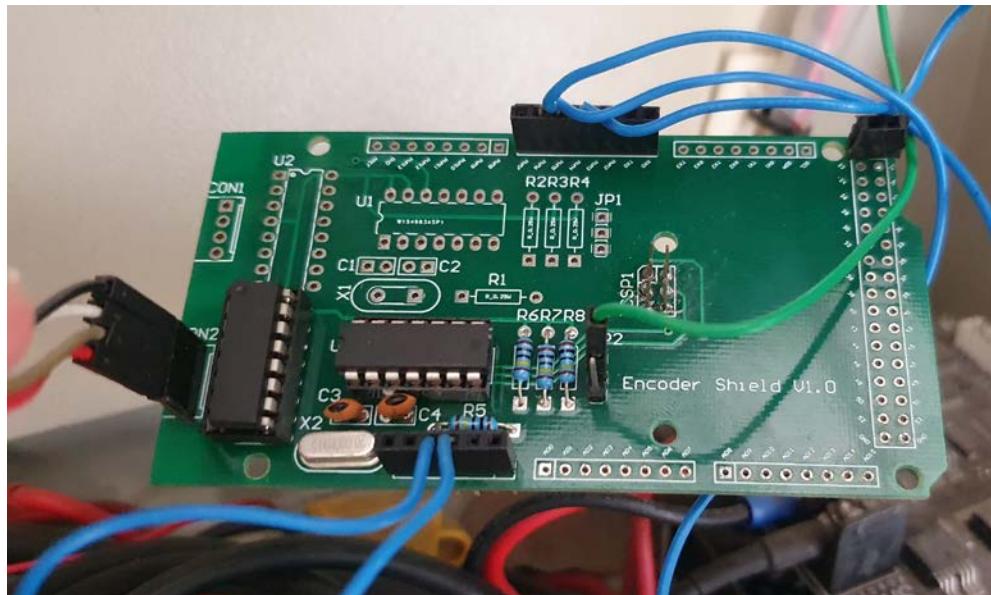
รูปที่ 12 DC Motor



รูปที่ 13 DC Motor ของอุปกรณ์แกนกลาง

3.1.4 Encoder Shield

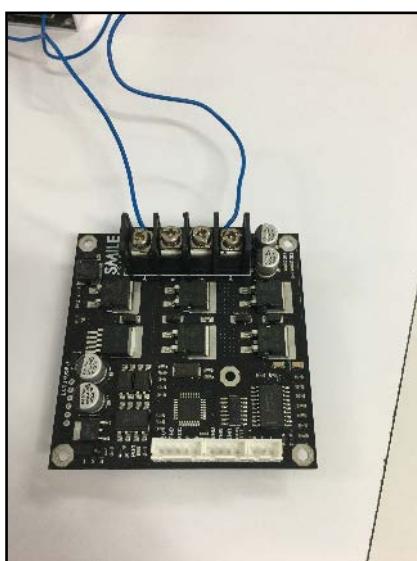
ใช้ติด Top Mount เพื่อรับค่าจากEncoder โดยประกอบตัว2 IC ตัวแรกฝั่งซ้ายคือ AM26LS32 และอีกตัวคือLS7366



รูปที่ 14 Encoder Shield

3.1.5 DC Motor Drives ชนิด H-Bridge

รุ่น EVO24V50 และ EVO24X9 Brushed DC Motor Driver บริษัท Smile Robotics ไฟฟ้าเลี้ยง 12VDC



รูปที่ 15 DC Motor Drives แบบ H-Bridge ทั้ง2แบบ

3.1.6 Encoder

รุ่น M40SB-3600 บริษัท Maxwell ใน 1 รอบของการหมุนสามารถนับได้ 14400 Pulse ใช้ไฟเลี้ยง 5VDC



รูปที่ 16 Encoder

3.1.7 Switching Hub

รุ่น TL-SF1008D บริษัท TP-LINK มี 8 port ที่ความเร็ว 10/100Mbps ใช้ไฟเลี้ยง 9VDC



รูปที่ 17 Switching Hub

3.1.8 Variable Resistor

ใช้ป้อนค่ากลับจากMotor แกนกลางกลับไปยังส่วนควบคุม



รูปที่ 18 Variable Resistor

3.1.9 DC Supply

ใช้เป็นแหล่งจ่ายไฟกับอุปกรณ์ต่างๆ

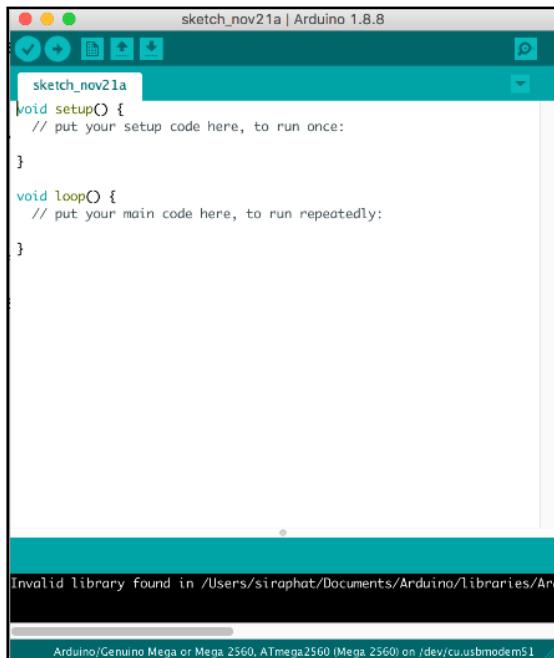


รูปที่ 19 DC Supply

3.2. อุปกรณ์ทางด้าน Software

3.2.1 Arduino IDE

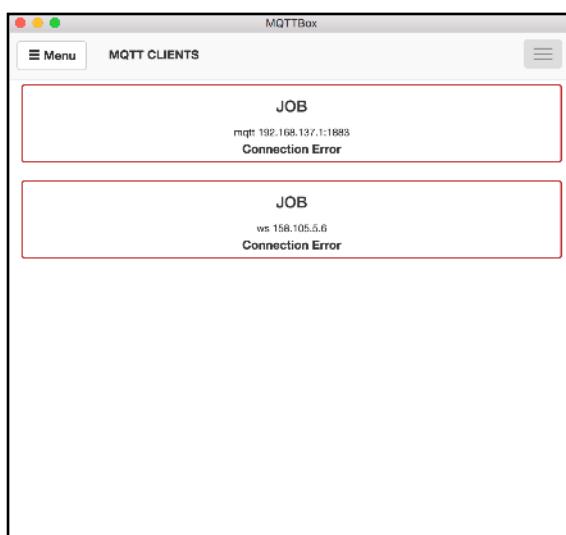
ตัวโปรแกรมมีไว้สำหรับ การเขียนภาษาซี (C Language) เพื่อควบคุมการทำงานของ บอร์ด Arduino



รูปที่ 20 Arduino IDE

3.2.2 MQTT Box

ตัวโปรแกรมมีหน้าที่เหมือนเซิฟเวอร์ให้บอร์ด Arduino หลายๆตัวสามารถสื่อสารผ่านช่องทางนี้ สามารถดูสถานะของบอร์ดแต่ละตัว และยังสามารถแสดงค่าสถานะต่างๆของระบบให้ user ภายนอกได้ผ่านทางหน้าจอ



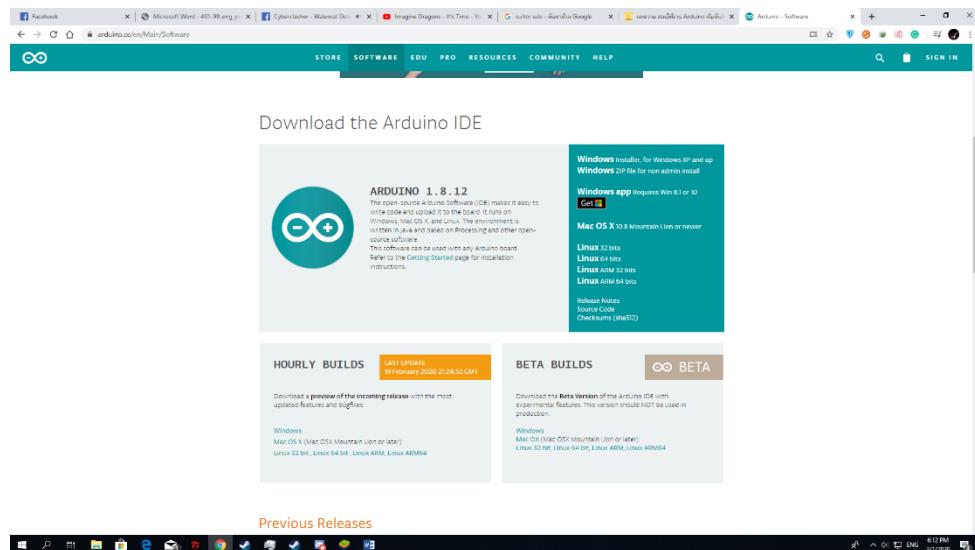
รูปที่ 21 MQTT Box

4. วิธีการดำเนินโครงการ

4.1 ขั้นตอนการใช้งาน Arduino [7]

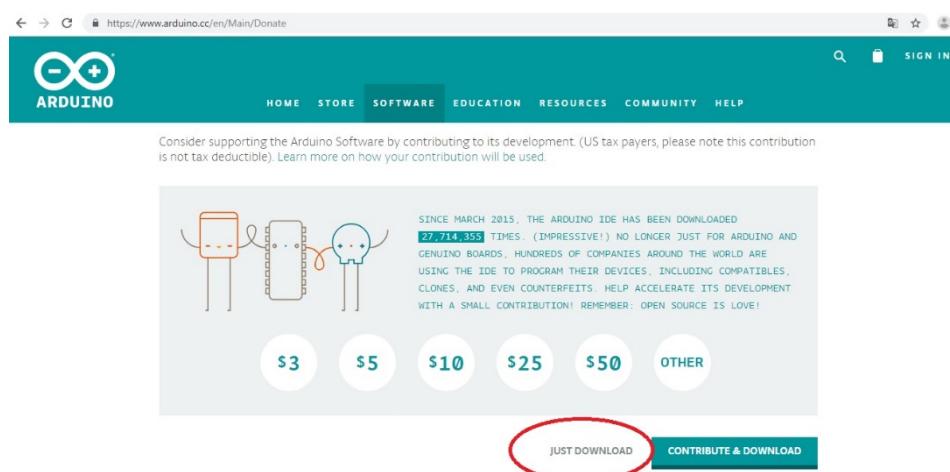
1.ดาวน์โหลด Arduino IDE จาก <https://wwwarduino.cc/en/Main/Software>

2.เลือกระบบปฏิบัติการของเครื่องคอมพิวเตอร์ที่จะใช้ในการเขียนโปรแกรม Arduino



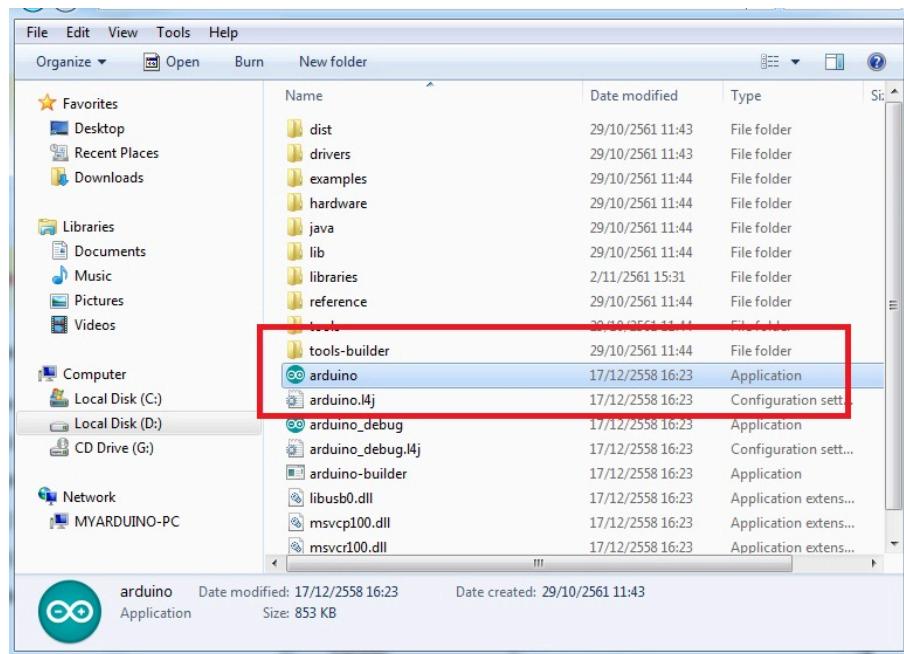
รูปที่ 22 หน้าเว็บเพจของ Arduino IDE

3.กด DOWNLOAD (หากต้องการร่วมบริจาคช่วยการพัฒนา Arduino Software สามารถกด CONTRIBUTE & DOWNLOAD)



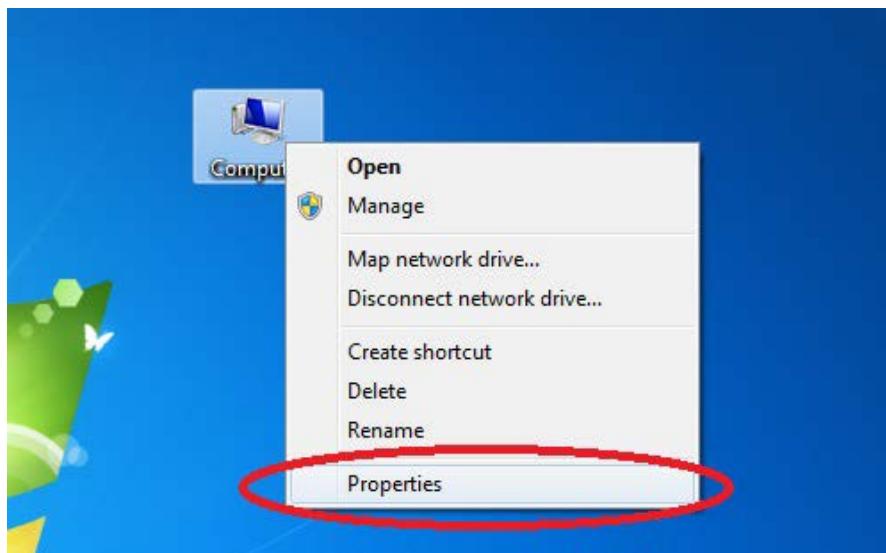
รูปที่ 23 วิธีการ Download Arduino IDE

4. ดับเบิลคลิกที่ไฟล์ arduino.exe เพื่อเปิดโปรแกรม Arduino IDE



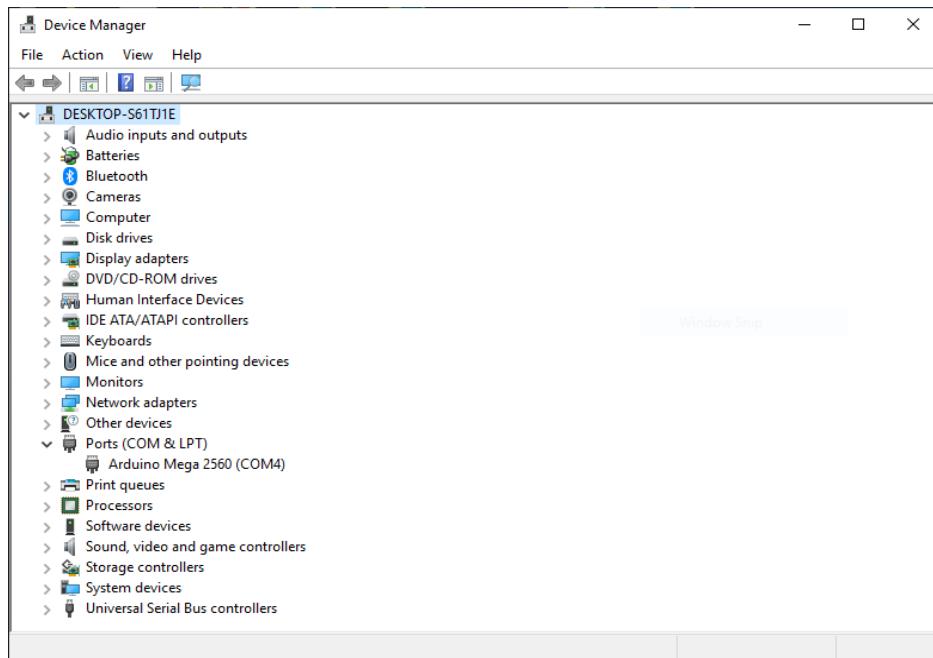
รูปที่ 24 Arduino IDE Application

5. เสียบบอร์ด Arduino Mega 2560 เข้ากับคอมพิวเตอร์ จากนั้นให้คลิกขวาที่ Computer เลือก Properties ไปที่ Device Manager เพื่อดูว่าบอร์ด Arduino MEGA 2560 นั้นต่ออยู่กับ COM Port หมายเลขใด



รูปที่ 25 การเปิด Device manager

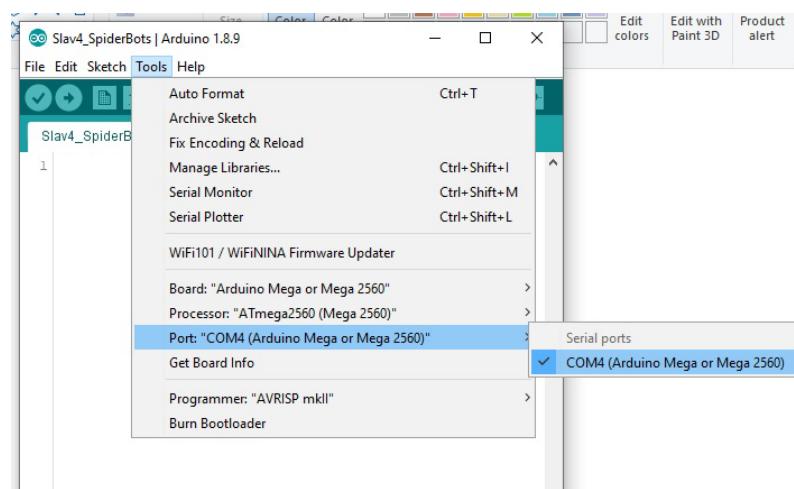
6. ใน Device Manager คลิกที่ Ports เพื่อดูหมายเลข COM Port ที่ Arduino Mega 2560 เชื่อมต่อ ดังภาพ



รูปที่ 26 Device Manager

7. เปิด Arduino IDE ขึ้นมาอีกครั้ง ไปที่เมนู Tools และทำการตั้งค่าบอร์ดและหมายเลขพอร์ตให้ตรงกับที่จะเชื่อมต่อ ซึ่งในตัวอย่างคือ COM4 ดังนี้

7.1 เลือก COM Port ในตรงกับตัวที่จะเชื่อมต่อ ในที่นี่คือ COM4

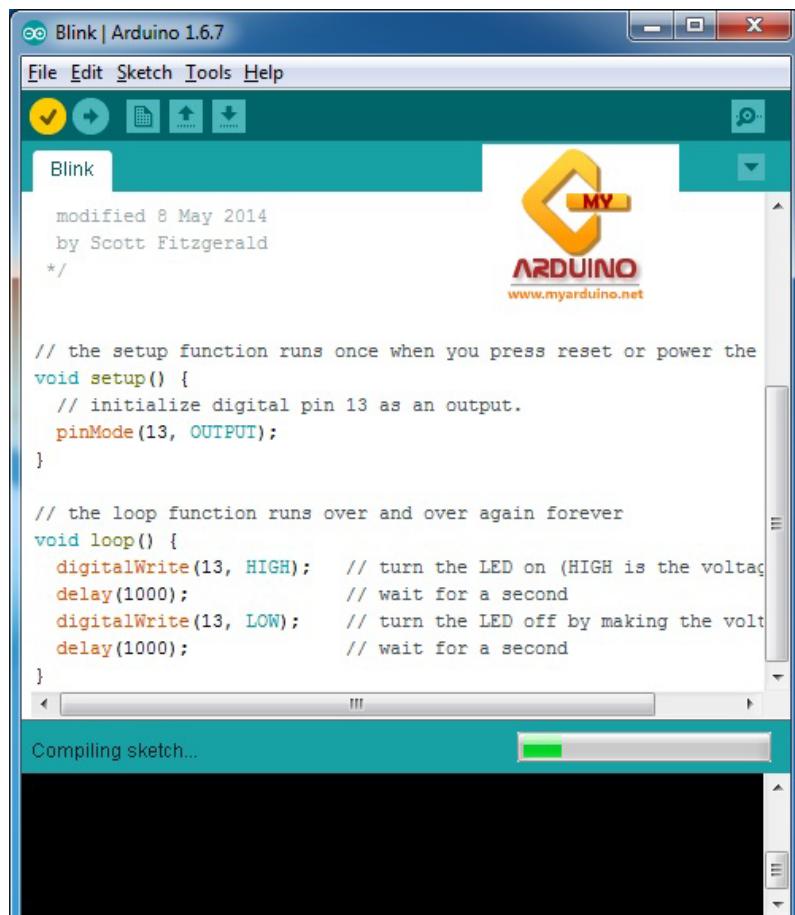


รูปที่ 27 การเชื่อมต่อArduino

7.2 เลือก รุ่น บอร์ด Arduino ที่ต้องการเชื่อมต่อ ในที่นี่คือ Arduino Mega 2560

8. เขียนโปรแกรม

9. กด คอมpile (Compile) โดยคลิกที่ปุ่ม  เพื่อตรวจสอบว่าโค้ดที่เขียนไม่มีข้อผิดพลาด



The screenshot shows the Arduino IDE version 1.6.7. The title bar says "Blink | Arduino 1.6.7". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for file operations like Open, Save, and Upload. The main code editor window displays the "Blink" sketch. The code is as follows:

```
// the setup function runs once when you press reset or power the
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);      // turn the LED on (HIGH is the voltage
                               // level)
    delay(1000);                // wait for a second
    digitalWrite(13, LOW);       // turn the LED off by making the volt
                               // age low
    delay(1000);                // wait for a second
}
```

The status bar at the bottom shows "Compiling sketch..." with a progress bar.

รูปที่ 28 การ Compile Program

10. หากไม่มีข้อผิดพลาด จะปรากฏข้อความว่า “Done compiling” ดังภาพ

```

Blink | Arduino 1.6.7
File Edit Sketch Tools Help
Blink
modified 8 May 2014
by Scott Fitzgerald
*/
// the setup function runs once when you press reset or power the
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);    // turn the LED on (HIGH is the voltage
    delay(1000);              // wait for a second
    digitalWrite(13, LOW);     // turn the LED off by making the volt
    delay(1000);              // wait for a second
}

```

Done compiling.

Sketch uses 1,030 bytes (3%) of program storage space. Maximum is 32,256 bytes. Global variables use 9 bytes (0%) of dynamic memory, leaving 2,039 bytes for local variables.

รูปที่ 29 แสดงผลการ Compile

11. จากนั้นให้ทำการคลิกที่ปุ่ม เพื่อทำการอัพโหลดโค้ดเข้าสู่บอร์ด Arduino Mega 2560

```

Blink | Arduino 1.6.7
File Edit Sketch Tools Help
Blink
modified 8 May 2014
by Scott Fitzgerald
*/
// the setup function runs once when you press reset or power the
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);    // turn the LED on (HIGH is the voltage
    delay(1000);              // wait for a second
    digitalWrite(13, LOW);     // turn the LED off by making the voltage
    delay(1000);              // wait for a second
}

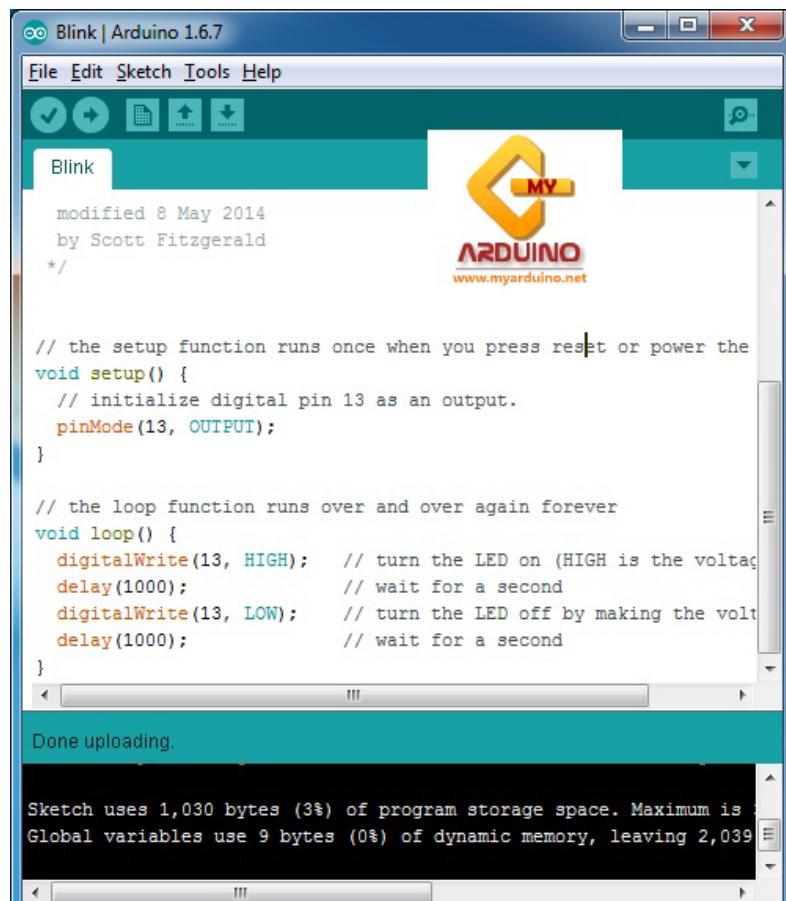
```

Uploading...

Sketch uses 1,030 bytes (3%) of program storage space. Maximum is 32,256 bytes. Global variables use 9 bytes (0%) of dynamic memory, leaving 2,039 bytes for local variables.

รูปที่ 30 การ Upload เข้าบอร์ด

12. หากไม่มีข้อผิดพลาด จะขึ้นคำว่า “Done uploading” ดังภาพ



The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** Blink | Arduino 1.6.7
- Menu Bar:** File Edit Sketch Tools Help
- Sketch Area:** Displays the `Blink` sketch code. The code is as follows:

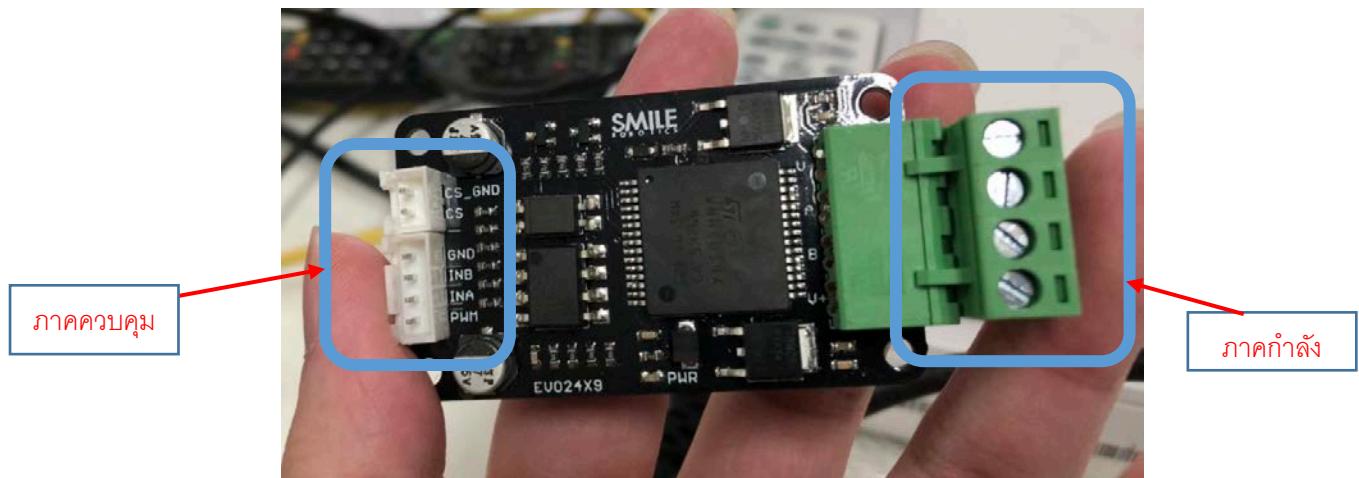
```
// the setup function runs once when you press reset or power the
void setup() {
    // initialize digital pin 13 as an output.
    pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
    digitalWrite(13, HIGH);      // turn the LED on (HIGH is the voltage level)
    delay(1000);                // wait for a second
    digitalWrite(13, LOW);       // turn the LED off by making the voltage level
    delay(1000);                // wait for a second
}
```
- Logo:** MY ARDUINO logo with the URL www.myarduino.net
- Status Bar:** Shows the message "Done uploading."
- Output Window:** Displays memory usage information:

```
Sketch uses 1,030 bytes (3%) of program storage space. Maximum is 32,256 bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2,039 bytes free.
```

รูปที่ 31 แสดงผลการ Upload

4.2 เรียนรู้หลักการทำงานของ Motor Drive



รูปที่ 32 Motor Drive

Motor Drive ชนิด H-bridge จะมีส่วนการทำงานแบ่งเป็น 2 ฝั่ง คือ ภาคควบคุม กับ ภาคกำลัง ภาคควบคุมจะมีการรับข้อมูลทั้งหมด 3 ขา คือ

1. PWM มีหน้าที่ควบคุมแรงดันฟังก์ภาคกำลัง ซึ่งส่งผลกับความเร็วในการหมุนของ motor
2. InA ควบคุมทิศทางการให้力ของกระแสไฟฟ้า ซึ่งส่งผลกับทิศทางในการหมุนของ motor
3. InB ควบคุมทิศทางการให้力ของกระแสไฟฟ้า ซึ่งส่งผลกับทิศทางในการหมุนของ motor

โดย การหมุนของ motor จะมีด้วยกัน 4 แบบ คือ

Logic	In A	In B	ผลการทำงานของ motor
	1	0	หมุนซ้าย
	0	1	หมุนขวา
	1	1	ล็อกการหมุน
	0	0	หมุนได้อย่างอิสระ

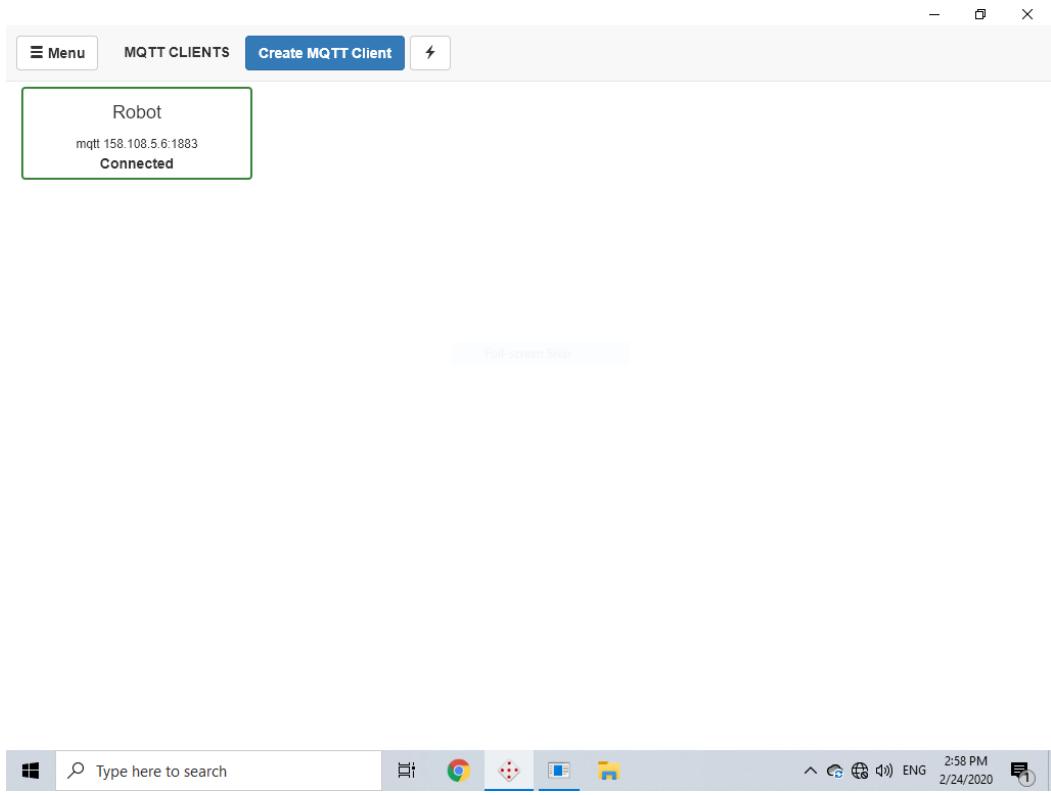
ตารางที่ 1 LOGIC การทำงานของMOTOR DRIVE

4.3 วิธีการสื่อสารของระบบ MQTT

1. Download แอพพลิเคชัน ใน Google Chrome Web store ชื่อว่า MQTT BOX

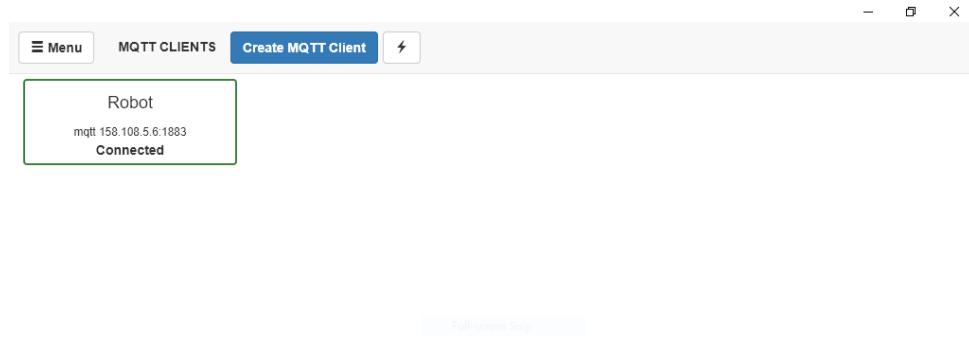
<https://chrome.google.com/webstore/detail/mqttbox/kaajoficamnjijhkeomgfljplicfbkaf?hl=th>

2. เปิดแอพพลิเคชัน MQTT BOX



รูปที่ 33 MQTTBox

3.สร้าง MQTT Client



4.ตั้งค่า MQTT client และ IP Address ของ broker

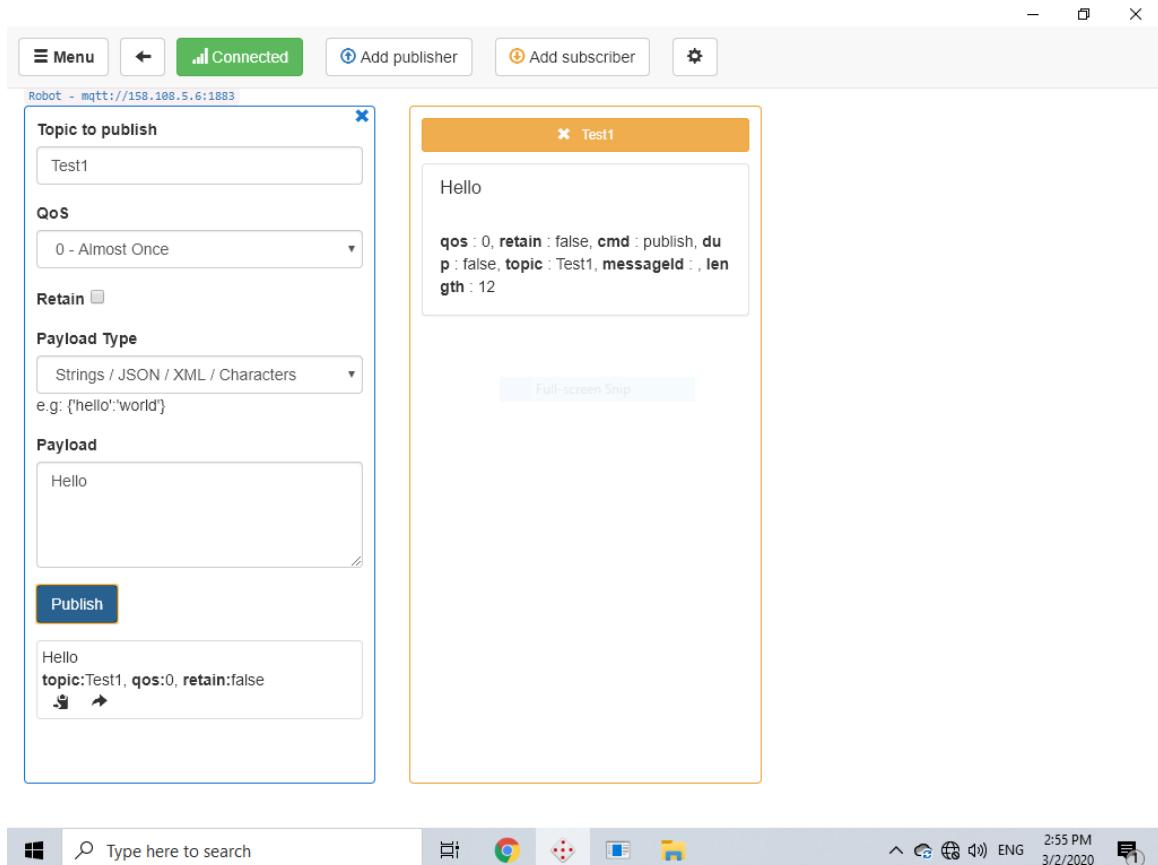
The screenshot shows the "MQTT CLIENT SETTINGS" page. At the top, there are tabs for "Menu", "MQTT CLIENT SETTINGS" (selected), and "Client Settings Help". The main area contains various configuration fields:

MQTT Client Name	MQTT Client Id	Append timestamp to MQTT client id?	Broker is MQTT v3.1.1 compliant?
Robot	3d7f5d36-b574-479e-b3c	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes
Protocol	Host	Clean Session?	Auto connect on app launch?
mqtt / tcp	158.108.5.6:1883	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes
Username	Password	Reschedule Pings?	Queue outgoing QoS zero messages?
Username	Password	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes
Reconnect Period (milliseconds)	Connect Timeout (milliseconds)	KeepAlive (seconds)	
1000	30000	10	
Will - Topic	Will - QoS	Will - Retain	Will - Payload
Will - Topic	0 - Almost Once	<input type="checkbox"/> No	

At the bottom, there are "Save" and "Delete" buttons.

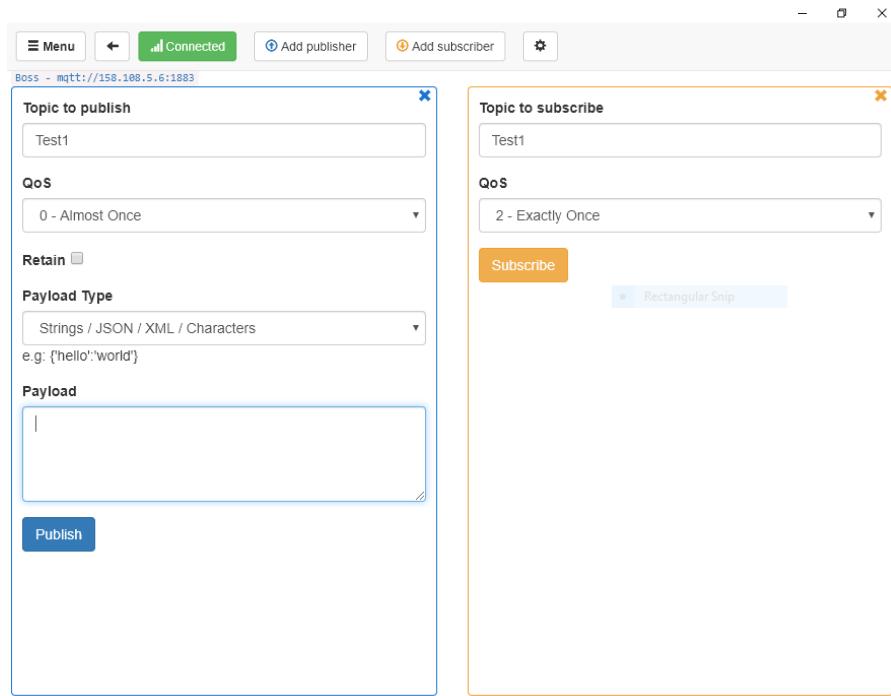
รูปที่ 34 สร้าง Client MQTT

5. กด Add publisher และ Add subscriber เพื่อสร้างช่องทางการส่งและการรับข้อมูล



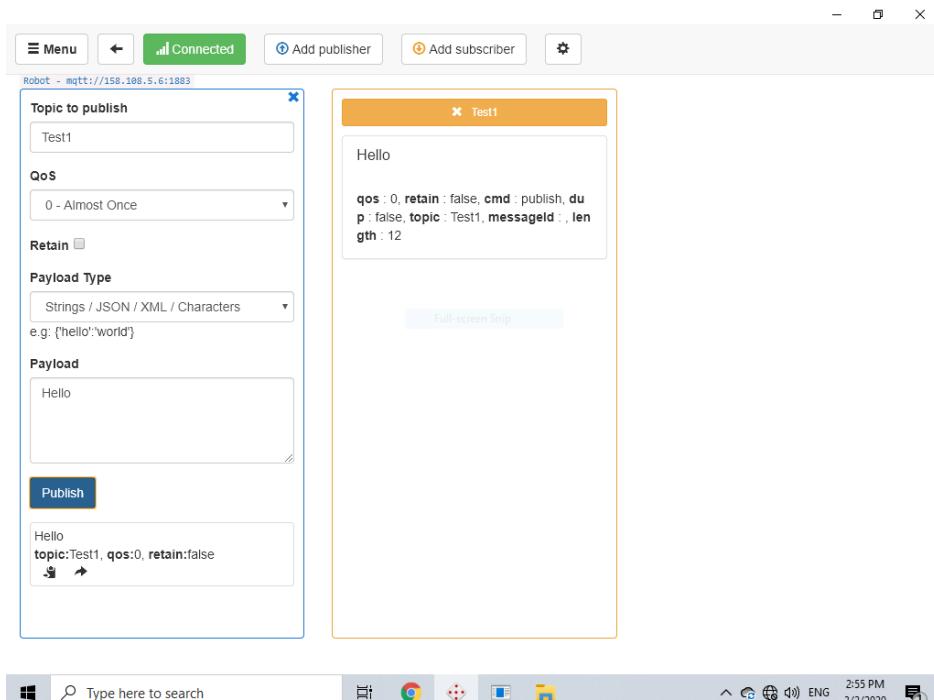
รูปที่ 36 Client Interface

6. สร้างช่อง publisher ชื่อว่า Test1 และ ช่อง subscriber ชื่อ Test1

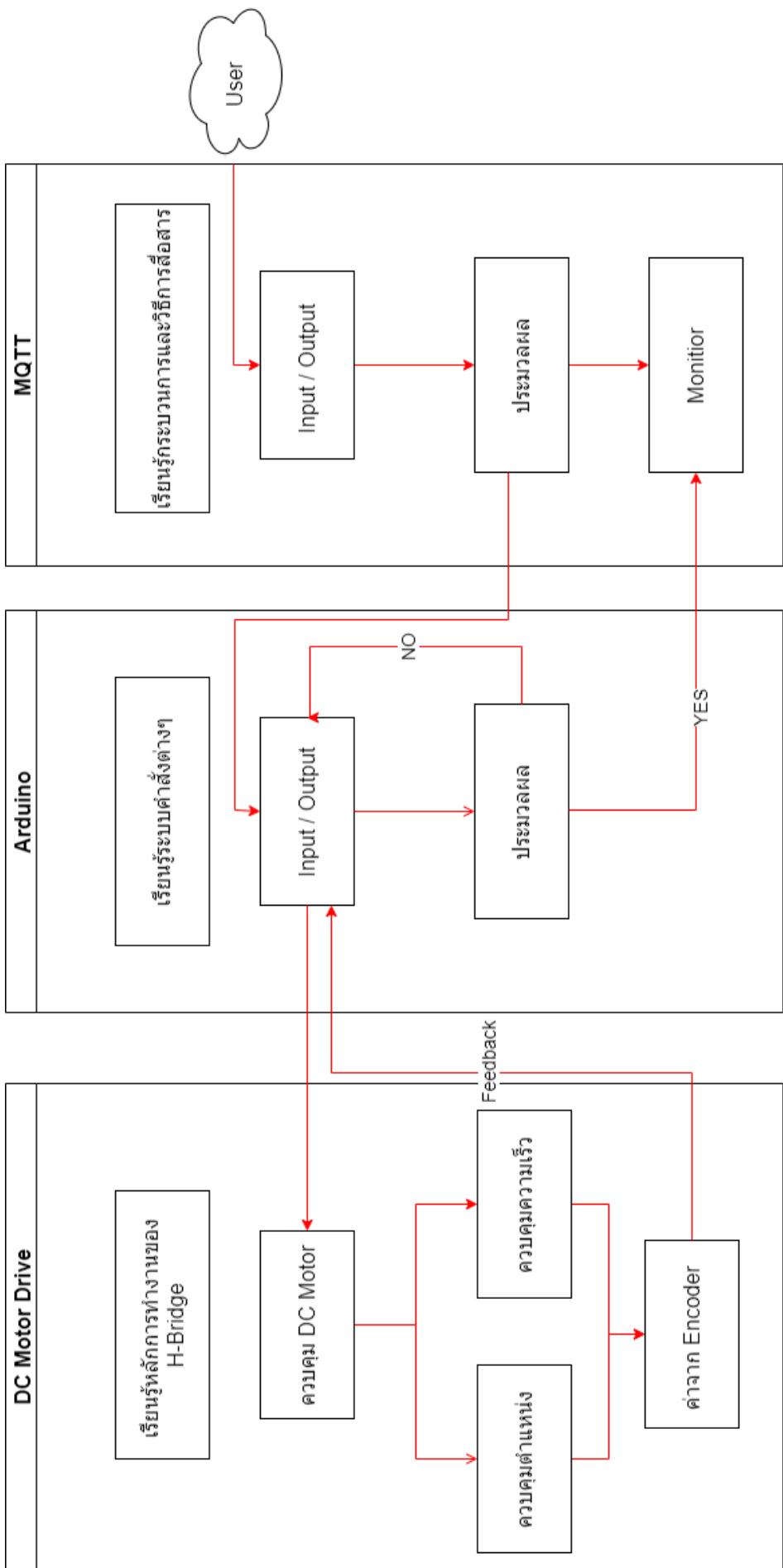


รูปที่ 37 Topic MQTT

7. ทดลองส่งข้อความ “Hello” จะเห็นข้อความที่ส่งเข้ามาทางช่อง subscriber



รูปที่ 38 Publish message



รูปที่ 39 Flowchart ของเรียนรู้

4.4 วิธีการควบคุมหุ่นยนต์

4.4.1 ควบคุมแบบกำหนดเอง

เป็นการควบคุมโดยที่ผู้ใช้งานสามารถกรอกค่าตำแหน่งพิกัดที่ต้องการไปในแกน X Y Z เพื่อหาค่า Setpoint ของมอเตอร์แต่ละตัว ซึ่งจุดเด่นของการควบคุมแบบกำหนดเองมีดังนี้

- สามารถสั่งเกตการณ์เคลื่อนที่ของหุ่นยนต์ได้อย่างชัดเจน
- สามารถรู้ข้อมูลของหุ่นยนต์ที่สามารถเคลื่อนที่ไปได้
- เหมาะสมสำหรับการเรียนรู้และนำไปพัฒนาต่ออยอด

4.4.2 ควบคุมแบบอัตโนมัติ

เป็นการควบคุมให้หุ่นยนต์ทำงานเองโดยไม่ต้องมีบุคคลในการควบคุม โดยต้องมีการเขียนโค้ดให้บอร์ด Master อีก 1 บอร์ด เพื่อที่จะเป็นตัวส่งค่า Setpoint ให้มอเตอร์แต่ละตัว ซึ่งจุดเด่นของการควบคุมแบบอัตโนมัติ มีดังนี้

- สามารถเคลื่อนที่ได้ตลอด 24 ชั่วโมง
- โอกาสความผิดพลาดในการสั่งงานน้อยกว่าแบบสั่งงานผ่านผู้ใช้งาน
- เหมาะสมกับการนำไปใช้งานกับการเกษตรอย่างมาก

5. ผลการดำเนินโครงการและวิจารณ์

5.1 การติดตั้งอุปกรณ์

อุปกรณ์ทั้งหมดประกอบด้วย 3 ส่วนหลัก คือ

1. กล่องอุปกรณ์ควบคุม

2. DC motor

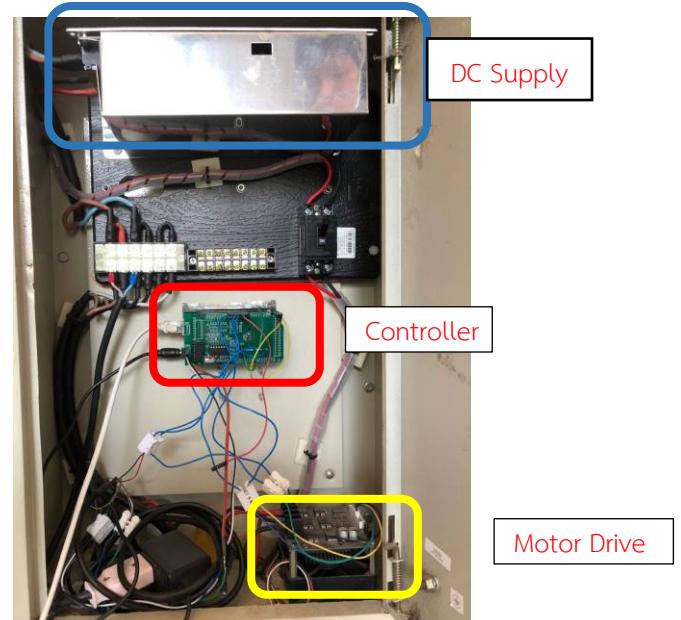
3. แกนอุปกรณ์



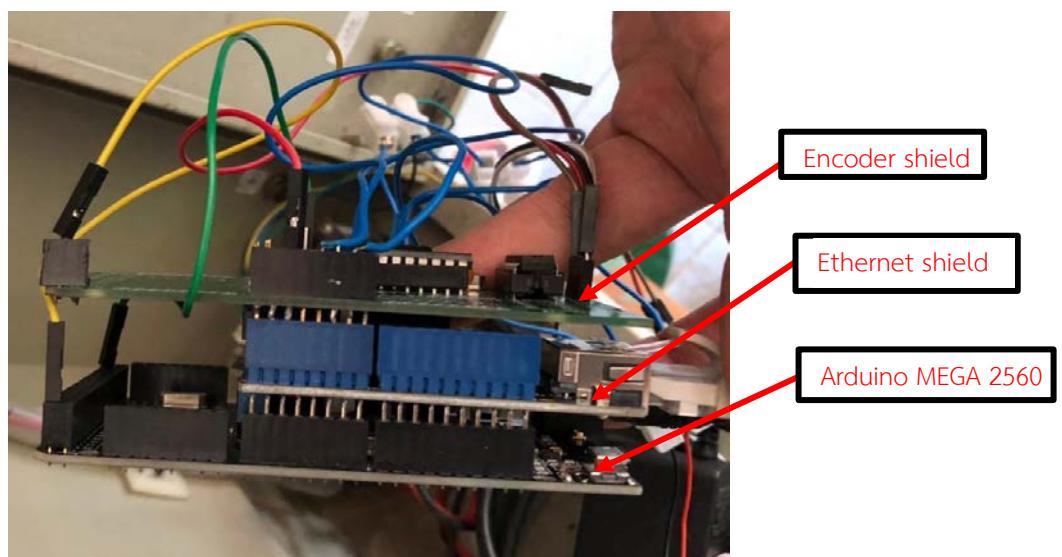
รูปที่ 40 โครงสร้างของชิ้นงาน

5.1.1 กล่องอุปกรณ์ควบคุม

ส่วนประกอบภายในประกอบด้วย แหล่งจ่าย DC, Motor Drive, Arduino, Ethernet shield, Encoder shield และ Circuit breaker



รูปที่ 41 โครงสร้างภายในกล่องอุปกรณ์



รูปที่ 42 อุปกรณ์ต่างๆ

5.1.2 DC Motor

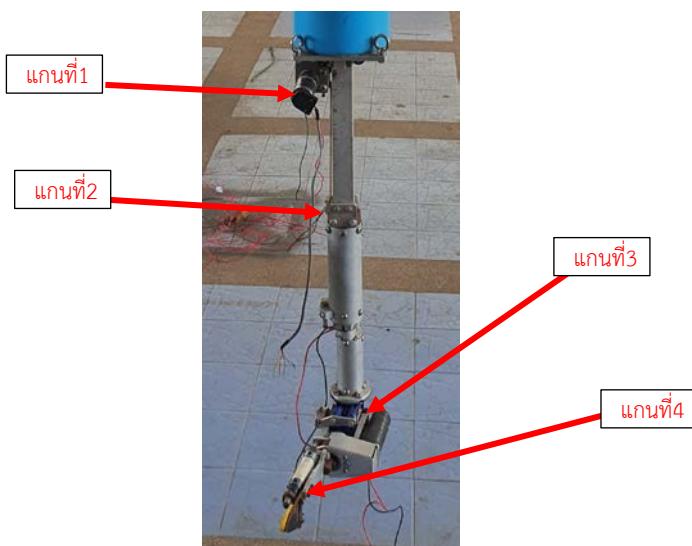
มีการใช้เกียร์ทดเพื่อเพิ่มทอร์คของ Motor



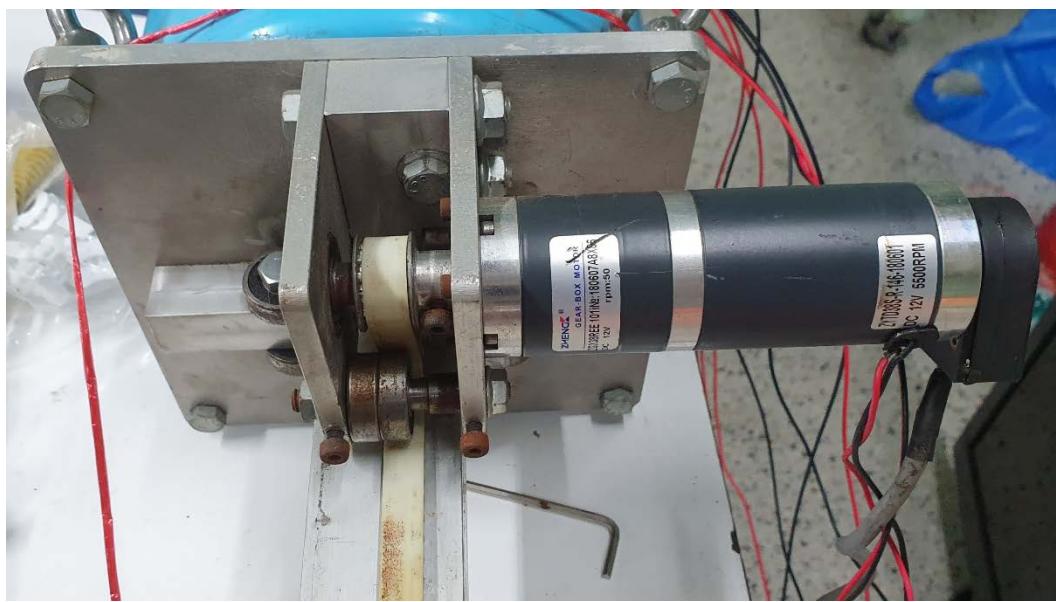
รูปที่ 43 ติดตั้งDC Motor

5.1.3 แกนอุปกรณ์

แกนอุปกรณ์ในโครงงานนี้ที่เลือกมาใช้นั้น ประกอบด้วยการเคลื่อนไหว 4 แกน นั่นคือตัวที่1 เคลื่อนที่ขึ้นลง ตัวที่2 เคลื่อนที่โดยการหมุนแกนอุปกรณ์ ตัวที่3 เคลื่อนที่แบบเบยอุปกรณ์ขึ้น หรือลดลง และตัวสุดท้าย เคลื่อนโดยหมุนตัวอุปกรณ์ไปรอบๆ



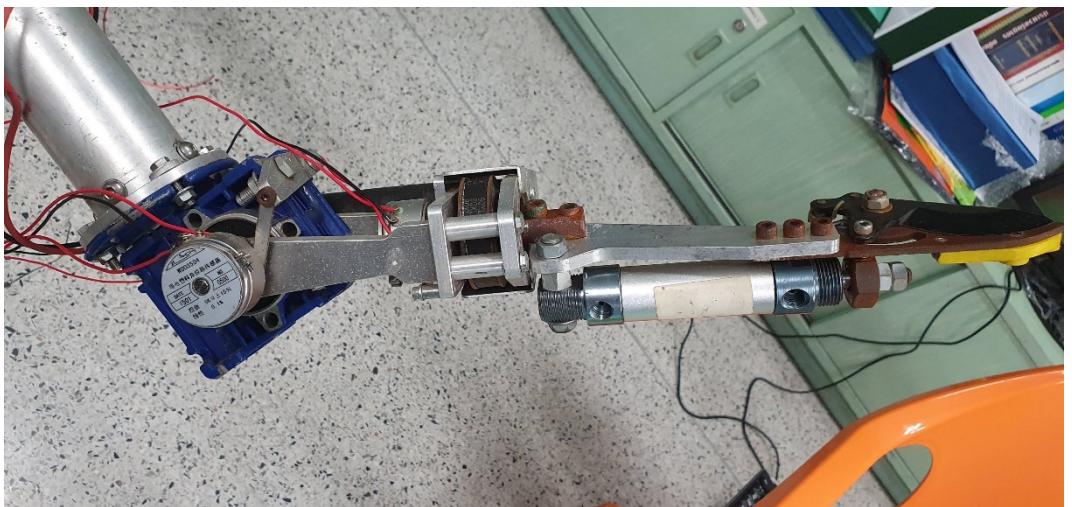
รูปที่ 44 แกนการเคลื่อนที่ของอุปกรณ์



รูปที่ 45 แกนขันล่ง



รูปที่ 46 แกนหมุนอุปกรณ์



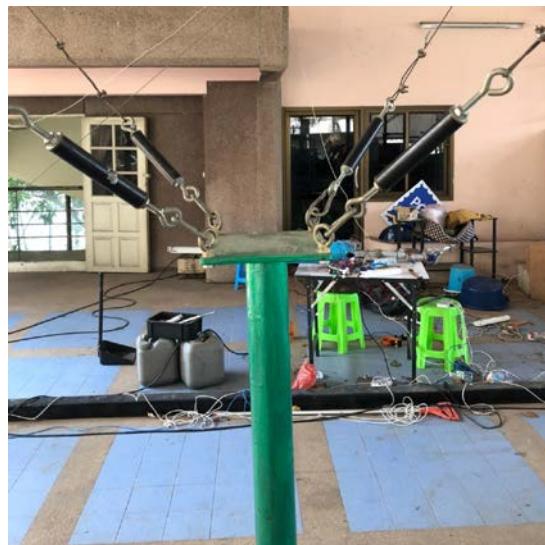
รูปที่ 47 แกนเบยจีนหรือกดลง



รูปที่ 48 แกนหมุนตัวอุปกรณ์

5.2 การทำงาน

User จะสามารถสั่งการผ่านคอมพิวเตอร์หรือโทรศัพท์ได้ โดยเข้า MQTTBox ที่กล่าวไว้ข้างต้นเพื่อสั่งการให้ Motor ทั้ง 4 เสาทำงาน โดยการป้อนข้อมูลของ User จะเป็นการป้อนตำแหน่ง ที่ User ต้องการผ่านการคำนวน ในบทที่ 4 ที่กล่าวมาแล้ว



รูปที่ 49 การเคลื่อนที่ของอุปกรณ์

โดยทั้ง 4 เสา 4 Motor ทำงานกันอย่างอิสระ นั่นคือความสามารถควบคุมได้อย่างอิสระ อีกทั้งแกนกลางนั้นสามารถเปลี่ยนเป็นอุปกรณ์ต่างๆได้ตามความเหมาะสม และสามารถควบคุมได้เช่นเดียวกัน



รูปที่ 50 อุปกรณ์แกนกลาง

5.3 การควบคุมและแสดงผล

การควบคุมจะประกอบด้วยบอร์ด Arduino 6 บอร์ด แบ่งเป็น Master 1 บอร์ด และ Slave 5 บอร์ด ซึ่งการสั่งการทั้งหมดนี้จะเป็นการป้อนข้อมูลให้กับบอร์ด Master คำนวณและส่งชุดคำสั่งต่างๆไปยังบอร์ดอื่นๆให้ทำงานตามที่ต้องการ โดยชุดคำสั่นนี้จะส่งผ่าน MQTT Protocol

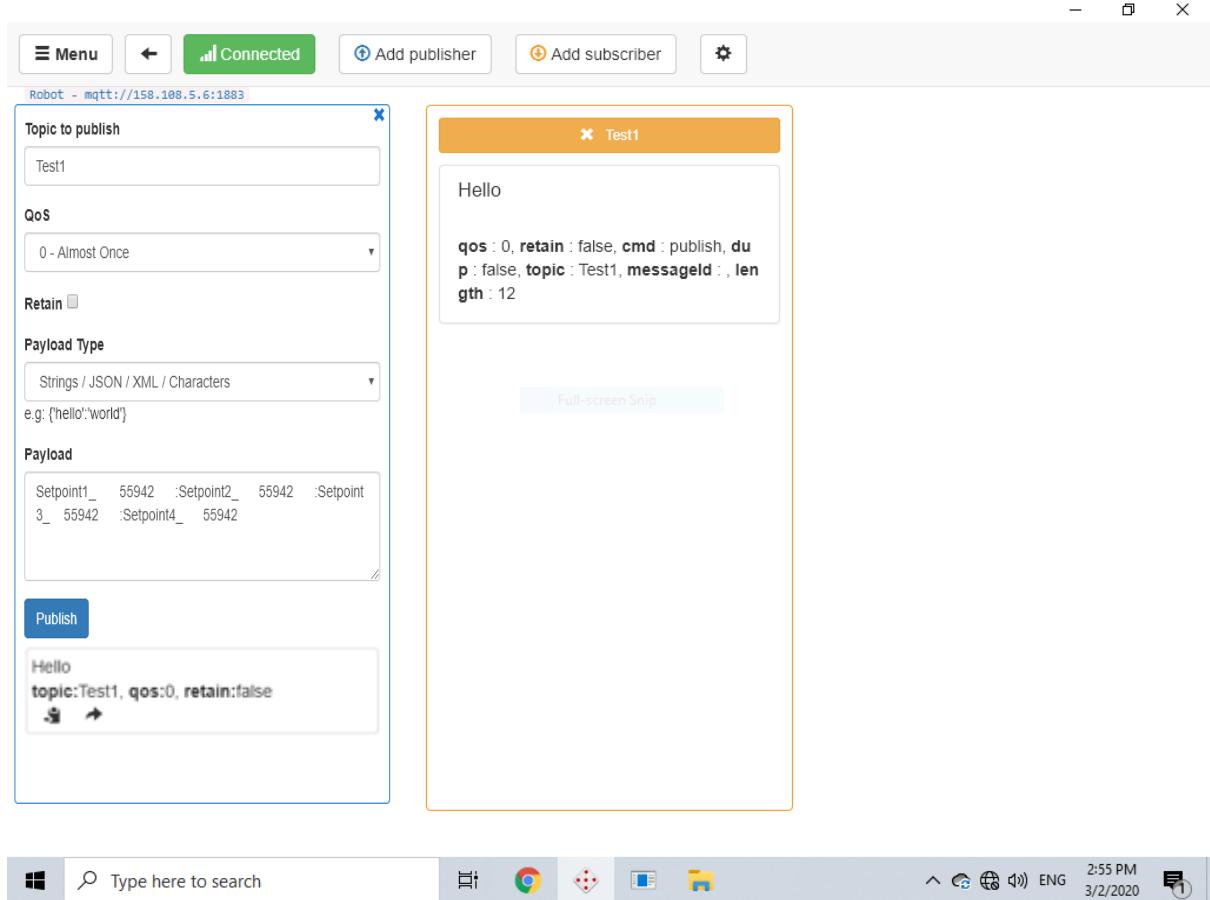
5.3.1 การหาพิกัด

- หากำลัง pulse จากการทดลองใช้ motor ดึงกลิ้งเป็นระยะ 50 เซนติเมตร วัดค่า pulse ของ encoder ออกมาได้ 5889 เทียบัญญัติตรายางค์ 1 เซนติเมตร จะได้ pulse ของ encoder เท่ากับ 117.72
- นำค่า pulse ที่ได้ไปคูณกับค่า L1,L2,L3,L4 จะได้ค่า setpoint1 setpoint2 setpoint3 setpoint4
- นำค่าที่ได้ส่งผ่าน MQTT ดังรูปจะเป็นการส่งพิกัดเป้าหมายให้ motor ทำงาน

ขอบเขตพิกัด				ใส่พิกัดที่ต้องการ		
XX	YY	ZZ	A	X	Y	Z
560	560	400	8.5	280	280	90
						cm
ความยาวสลิง	ความยาวสลิง	ความยาวสลิง	ความยาวสลิง	วัดได้	ค่า pulse	cm
L1	L2	L3	L4		5886	50
493	493	493	493		pulse/1cm	118
				cm		

Setpoint1	58093	:Setpoint2	58093	:Setpoint3	58093	:Setpoint4	58093		
-----------	-------	------------	-------	------------	-------	------------	-------	--	--

รูปที่ 52 การคำนวณ Position



รูปที่ 53 การส่งค่า Set point

เมื่ออุปกรณ์แกนกลางเคลื่อนที่ไปถึงตำแหน่งที่ระบุไว้ Encoder ก็จะส่งค่าตำแหน่งเพื่อแจ้ง และสั่งการให้อุปกรณ์แกนกลางเริ่มทำงาน อุปกรณ์แกนกลางในที่นี้ที่เลือกมาใช้งานจะเป็นการตัดกิ่ง ซึ่งเราสามารถเปลี่ยนอุปกรณ์แกนกลางได้ตามความเหมาะสมของ การใช้งาน

5.4 วิจารณ์ผลการดำเนินงานที่ได้

จากผลการดำเนินงานและทดสอบเคลื่อนที่ประมาณ 50 ครั้ง พบร้าแกนกลางสามารถเคลื่อนที่ 3 มิติได้อย่างราบรื่น มีความคลาดเคลื่อนต่ำกว่า 10 เซนติเมตร และสามารถใช้เครื่องมือทางการเกษตรได้มากไปกว่าพันนั้นยังสามารถสั่งการด้วยระบบอัตโนมัติทางไกลได้อีกด้วย

ผู้ออกแบบประเมินผลการดำเนินงานของโครงงานไว้ ดังนี้

1. อุปกรณ์ดังกล่าวจะสามารถทดแทนแรงงานคนได้
2. อุปกรณ์ดังกล่าวมีความแม่นยำสูงและควบคุมทางไกลได้
3. อุปกรณ์ดังกล่าวมีอายุการใช้งานและความทนทานสูง

6. สรุปผลการดำเนินงานและข้อเสนอแนะ

6.1 การใช้งานและควบคุม Controller

การควบคุม Controller อย่าง Arduino Board นั้น ต้องใช้ภาษาซี ในการเขียนโปรแกรม ซึ่งต้องเข้าใจแต่ละคำสั่ง และฟังก์ชัน ของบอร์ดอย่างลึกซึ้ง ว่าทำงานอย่างไร เพื่อที่จะได้สามารถนำไปควบคุมอุปกรณ์ตัวอื่นที่เชื่อมต่ออยู่ได้ เช่น การควบคุม DC motor ซึ่งต้องควบคุมผ่าน motor drive ชนิด H-Bridge, การรับค่าจาก Encoder ที่ใช้ใน การระบุตำแหน่งที่ DC motor หมุนไปได้, การเชื่อมต่ออินเตอร์เน็ต ผ่าน Ethernet Shield เป็นต้น

6.2 การควบคุมมอเตอร์ แบบ Feedback (PID-Controller)

ต้องมีค่าพารามิเตอร์เหล่านี้ของระบบในใจว่าต้องการมากน้อยขนาดไหน Rise Time ช่วงเวลาข้ามต้องการเร็ว มากหรือไม่หรือยอมรับค่ามากๆได้ roi ได้ Overshoot ยอมให้มีโอเวอร์หรือไม่หรือยอมได้มากน้อยขนาดไหน Settling Time ช่วงเวลาตั้งแต่มีการเปลี่ยนค่าตำแหน่งเก่าไปตำแหน่งใหม่ต้องเร็วขนาดไหน Steady-state Error ยอมให้มีค่า Error ตอนระบบนิ่งแล้วหรือไม่ ยอมได้หรือยอมไม่ได้ ในส่วนของโครงงานครั้งนี้ สิ่งที่สำคัญที่สุด คือ ตำแหน่ง ดังนั้น ในการจูน PID จึงต้องทำให้ค่า Steady-state Error มีค่าเป็น 0 หรือใกล้เคียง 0 จึงสามารถจะทำ ให้เราระบุตำแหน่งของ tool ทางการเกษตรได้อย่างแม่นยำ

6.3 ควบคุมเครื่องจักรจากทางไกลได้

เข้าใจระบบการสื่อสารแบบ MQTT Protocol มา กว่า 1 ชั่วโมง การสื่อสารได้นั้น สิ่งสำคัญคือการระบุ Mac address อย่างเฉพาะเจาะจงให้กับอุปกรณ์แต่ละชนิด และทุกอุปกรณ์ ต้องอยู่ในโครงข่ายเดียวกัน ซึ่งขอบเขตของโครงงานนี้ สามารถส่งข้อมูลผ่านทางโทรศัพท์เพื่อควบคุมการทำงานของหุ่นยนต์ได้

6.4 ปัญหาและอุปสรรค

- ขาดความเข้าใจในเครื่องมือและอุปกรณ์ต่างๆ
- อุปกรณ์บางชนิดเกิดการชำรุดขณะใช้งาน
- สามารถไม่พิ้นฐานความรู้ไม่เท่ากัน
- ทุกอุปกรณ์ทำงานในสภาพแวดล้อมที่ไม่เหมือนกันในอุณหภูมิที่ต้องการ
- ขอบเขตของโครงงานค่อนข้างกว้างทำให้รู้สึกท้อและหมดกำลังใจ

6.5 แนวทางการแก้ไขในอนาคต

- ศึกษาการทำงานของเครื่องมือและอุปกรณ์ต่างๆอย่างละเอียด และจัดทำคู่มือไว้
- ฝึกหาข้อมูลที่น่าเชื่อถือ เช่น Textbook
- ชุดกลไกทางกลบางอย่างมีความซับซ้อนและเก่า ควรเปลี่ยนใหม่ให้เข้าใจง่ายและสอดคล้องกับการทำงาน

6.6 สิ่งที่คณะผู้จัดทำได้เพิ่มเติมในโครงการ

- เขียนโปรแกรมควบคุม (C Language) ชุดกลไกทางกลที่อาจารย์ที่ปรึกษาได้เตรียมไว้
- เขียนโปรแกรมควบคุม (C Language) ชุดเครื่องมือทางการเกษตรที่อาจารย์ที่ปรึกษาได้เตรียมไว้
- เขียนโปรแกรมควบคุม (C Language) ระบบควบคุมอัตโนมัติสำหรับสิ่งงานหุ่นยนต์เพื่อการเกษตร

7. บรรณานุกรม

- [1] ประภาส สุวรรณเพชร. เรียนรู้และลองเล่น Arduino เป็นเบื้องต้น. [On-line]. Available: www.thephylconnect.com/images/Arduin/KruPraphasArduinoBook.pdf [Jan 10, 2019]
- [2] วิกพ ใจแข็ง. ระบบควบคุม. [On-Line]. Available: www.intech.crru.ac.th/research_ind/doc/52_dissemin_ทำ raided 20 พศ 20 วิกพ.pdf [Nov 5, 2019].
- [3] Chaiyaporn Silawatchananai. “การควบคุมมอเตอร์ไฟฟ้ากระแสตรง.” Internet: <http://aimagin.com/blog/motor/?lang=th>, Jul 2, 2014 [Sep 10, 2019].
- [4] John G. Bollinger, Neil A. Duffie. Computer Control of Machines and Processes. Boston, USA: Addison-Wesley, 1989, pp199-210.
- [5] Phannita. “ເອັນໂຄດເດອර് Encoder ດີວ່າໄວ? ມາຫາຄຳຕອບກັນ.” Internet: <https://mall.factoromart.com/what-is-encoder/>, Nov 22, 2018 [Oct 2, 2019].
- [6] M. Newman, “Design and Experimentation of Cable-Driven Platform Stabilization and Control System”, Ph.D dissertation, College of Eng. and Sc., Nebraska Univ., Lincoln, 2017
- [7] “บทความ สอนใช้งาน Arduino เริ่มต้นติดตั้งโปรแกรม Arduino IDE.” Internet: <https://www.myarduino.net/article/74/บทความ-สอนใช้งาน-arduino-เริ่มต้นติดตั้งโปรแกรม-arduino-ide>, [Jan 15, 2020]
- [8] “ระบบควบคุมพีไอดี” Internet: <https://th.wikipedia.org/wiki/ระบบควบคุมพีไอดี>, [Nov 5, 2019].
- [9] “ESP32 เป็นเบื้องต้น: บทที่ 13 เชื่อมต่อ กับ MQTT.” Internet: www.ioxhop.com/article/74/esp32-been-taun-13-cheom-to-kab-mqtt, Jun 26, 2017 [Mar 18, 2019].

ภาคผนวก

```
Master_ Spiderbots' Code {  
  
#include <SPI.h> Serial.begin(57600);  
  
#include <Ethernet.h> client.setServer(server, 1883);  
  
#include <PubSubClient.h> client.setCallback(callback);  
  
EthernetClient ethClient; Ethernet.begin(mac, ip);  
  
PubSubClient client(ethClient); Init_TIMER1_interupt();  
  
delay(1000);  
  
byte mac[] = { 0x1E, 0xD1, 0x11, 0x11, 0x11,  
0x11 }; }  
  
IPAddress ip(158, 108, 5, 15); void loop()  
IPAddress server(158, 108, 5, 6); {  
  
const char* mqtt_server = Step1();  
"158.108.5.6:1883";  
  
const char* Board = "Board_Master"; Step2();  
  
String mData,mData1; Step3();  
  
char mCharOutput[70]; Step4();  
  
String mSent, mX; Step5();  
  
int A = 0, B = 0, C = 0, D = 0; Step6();  
  
void setup() Step7();  
delay(200);  
Step8();
```

```

}

ISR(TIMER1_OVF_vect)      // interrupt          Data_Setpoint();

service routine Timer 01

{

    mData1 = "A_2:B_2:C_2:D_2";

    CheckMQTT();

    TCNT3 = 0;

}

void Init_Input ()          Data_Tool();

{

    pinMode(38, INPUT);

    pinMode(39, INPUT);

    pinMode(40, INPUT);

    pinMode(41, INPUT);

    pinMode(42, INPUT);

    pinMode(43, INPUT);

    pinMode(44, INPUT);

    pinMode(45, INPUT);

}

void Step1()                delay(2000);

{

    mData1 = "Setpoint1_56447:Setpoint2_68493:Setpoint3_41004:Setpoint4_56447";

    Data_Setpoint();

    delay(25000);

    mData1 = "A_1:B_1:C_1:D_1";

    Data_Tool();

    delay(2000);

    Data_StopTool();

}

void Step2()                Data_Setpoint();

{

    mData1 = "Setpoint1_68493:Setpoint2_56447:Setpoint3_56447:Setpoint4_41004";

    Data_Setpoint();

    delay(25000);

    mData1 = "A_2:B_2:C_2:D_2";

    Data_Tool();

    delay(2000);

    Data_StopTool();

}

void Step3()                Data_Setpoint();

{

```

```

mData1 =
    "Setpoint1_56447:Setpoint2_41004:Setpoint3
     _68493:Setpoint4_56447";
}

Data_Setpoint();
delay(25000);

mData1 = "A_2:B_2:C_2:D_2";
Data_Tool();
delay(2000);

Data_StopTool();
}

void Step4()
{
    mData1 =
        "Setpoint1_41004:Setpoint2_56447:Setpoint3
         _56447:Setpoint4_68493";
    Data_Setpoint();
    delay(25000);

    mData1 = "A_1:B_1:C_1:D_1";
    Data_Tool();
    delay(2000);

    Data_StopTool();
}

void Step5()
{
    mData1 =
        "Setpoint1_53614:Setpoint2_66178:Setpoint3
         _37007:Setpoint4_53614";
    Data_Setpoint();
    delay(25000);

    mData1 = "A_2:B_2:C_2:D_2";
    Data_Tool();
    delay(2000);

    Data_StopTool();
}

void Step6()
{
    mData1 =
        "Setpoint1_66178:Setpoint2_53614:Setpoint3
         _53614:Setpoint4_37007";
    Data_Setpoint();
    delay(25000);

    mData1 = "A_1:B_1:C_1:D_1";
}

```

```

Data_Tool();
delay(25000);

delay(2000);

Data_StopTool();
mData1 = "A_1:B_1:C_1:D_1";
}

void Step7()
{
    mData1 =
"Setpoint1_53614:Setpoint2_37007:Setpoint3
_66178:Setpoint4_53614";

Data_Setpoint();
delay(25000);
delay(25000);

mData1 = "A_2:B_2:C_2:D_2";

Data_Tool();
delay(2000);

Data_StopTool();
}

void Step8()
{
    mData1 =
"Setpoint1_37007:Setpoint2_53614:Setpoint3
_53614:Setpoint4_66178";

Data_Setpoint();
}

Data_Tool();
delay(2000);

Data_StopTool();
mData1 = "A_0:B_0:C_0:D_0";
}

```

```

Data_Tool();

}

///////////////////////////////
/////////////////////////////
for (int i = 0; i < length; i++)

{

Serial.print((char)payload[i]);

mData += (char)payload[i];

}

Serial.println();

}

void CheckMQTT()

{

if (!client.connected())

{

reconnect();

}

client.loop();

}

void reconnect()

{

while (!client.connected())

{

if (client.connect("Board_Master"))

{



}

```

```

Serial.println("connected");
A = 2;

//client.subscribe("Test3");
}

client.publish("Test1", "Board_Master is
connection");
else
{
    Serial.print("failed, rc=");
    Serial.print(client.state());
    Serial.println(" try again in 5 seconds");
    //Wait 5 seconds before retrying
    delay(1000);
}
}

void Control_Tool2()
{
    if (digitalRead(40) == LOW)
    {
        B = 1;
    }
    else if (digitalRead(41) == LOW)
    {
        B = 2;
    }
}

void Control_Tool1()
{
    if (digitalRead(38) == LOW)
    {
        A = 1;
    }
    else if (digitalRead(39) == LOW)
    {
        B = 0;
    }
}

```

```

void Control_Tool3()
{
    if (digitalRead(42) == LOW)
    {
        C = 1;
    }
    else if (digitalRead(43) == LOW)
    {
        C = 2;
    }
}

Slave1_Spiderbots' Code
#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>
EthernetClient ethClient;
PubSubClient client(ethClient);
byte mac[] = { 0xDE, 0xED, 0xAA, 0xFE, 0xFE,
0xEF };

void Control_Tool4()
{
    if (digitalRead(44) == LOW)
    {
        D = 1;
    }
    else if (digitalRead(45) == LOW)

```

```

#define T      0.1
int      Mn_max      = 30;
int      Mn_min      = -Mn_max;
const int slaveSelectEnc1 = 53;
double    Timerz = 59200;
float deltaTime ;
double Velocity ;
long currentPos;
long deltaPos ;
float oldTime   = 0;
long oldPos     = 0;
long setpoint   = 0;
double Cn, Mn, Mn1, Mn2, En, En1, En2;
double Kp = 0;
double Ki = 0;
double Kd = 0;
double counter = 0;
double K0 = 0;
double K1 = 0;
double K2 = 0;
double Rn = 0;
double pulse ;
int      mMode = 0;
String  mData;

char mCharOutput[10];
double FactorPulseperCm = 117.72 ;
double XX = 560, YY = 560, ZZ = 400, A =
8.5;
double x = 280, y = 280, z = 90;
int L1_Home = 493;
int L2_Home = 493;
int L3_Home = 493;
int L4_Home = 493;
long L1 = sqrt(sq(x - A) + sq(YY - y - A) +
sq(ZZ - z));
long L2 = sqrt(sq(XX - x - A) + sq(YY - y - A) +
sq(ZZ - z));
long L3 = sqrt(sq(x - A) + sq(y - A) + sq(ZZ -
z));
long L4 = sqrt(sq(XX - x - A) + sq(y - A) +
sq(ZZ - z));
void Init_PWM()
{
    pinMode(IN1, OUTPUT);
    pinMode(IN2, OUTPUT);
    pinMode(PWMz, OUTPUT);
}
void Init_PID_Control()
{

```

```

Mn_max      = 30;                      }

Mn_min      = -Mn_max;                  }

En2         = 0;                       void Init_PosPID()

En1         = 0;                       {

Mn          = 0;                      Rn = setpoint;

Mn1         = 0;                      Kp = 0.1;

Mn2         = 0;                      Ki = 0.003;

counter     = 0;                      Kd = 0.0001;

}                                     }

void Init_VeloPID()                   void Init_PID_Gain()

{

Serial.println(pulse);                K0 = Kp + (Ki * T) + (Kd / T);

if(pulse>0)                         K1 = (-Kp) - (2 * Kd / T);

{

Rn = pulse*0.001;                  K2 = (Kd / T);

Kp = abs(pulse*0.0001);            }

Ki = abs(pulse*0.00005);           ISR(TIMER1_OVF_vect)    {

}

else                                Rez();

{

mCharOutput[i] = PPos[i];          for (int i=0;i<PPos.length();i++)

Rn = pulse*0.001;                  }

Kp = abs(pulse*0.00025);           client.publish("L1",mCharOutput);

Ki = abs(pulse*0.00012);           }

Kd = 0.005;                        TCNT3 = 0;

```

```

}

ISR(TIMER3_OVF_vect)      // interrupt
service routine Timer 03
{
    Cn = currentPos;

    PID_Control_Run();
    TCNT3 = Timerz;
}

void PID_Control_Run()
{
    deltaTime = CalculateDeltaTime();
    deltaPos = CalculateDeltaPos();
    Velocity = ((deltaPos / deltaTime) * 1000 *
60) / 4096.0;
    if ( abs(abs(currentPos) - abs(setpoint)) >=
1000 && mMode == 0 )
    {
        mMode = 0;
        Init_VeloPID();
        Cn = Velocity;
    }
    else
    {
        mMode = 1;
        Init_PosPID();
    }
}

Cn = currentPos;
}

Init_PID_Gain();
En = Rn - Cn;
Mn = Mn1 + (K0 * En) + (K1 * En1) + (K2 *
En2);
if ( abs(En) <= 100 && mMode == 1 )
{
    Mn = 0;
}
PWM();
En2 = En1;
En1 = En;
Mn2 = Mn1;
Mn1 = Mn;
counter++;
}

float CalculateDeltaTime()
{
    float currentTime = millis();
    float deltaTime = currentTime - oldTime;
    oldTime = currentTime;
    return deltaTime;
}

```

```

long CalculateDeltaPos()
{
    currentPos = ((-1)*(readEncoder() * 1024) /
3600.0) + (L1_Home * FactorPulseperCm);
    deltaPos = currentPos - oldPos;
    oldPos = currentPos;
    return deltaPos;
}

void PWM()
{
    double MnTest;
    MnTest = Mn;
    if (Mn > Mn_max)
    {
        if (mMode == 0)
            Mn_max = Mn;
        else
            MnTest = Mn_max;
    }
    else if (Mn < Mn_min && counter >= 10)
    {
        if (mMode == 0)
            Mn_min = Mn;
        else
            MnTest = Mn_min;
    }
    Serial.println("Mn_Test: "+String(MnTest));
    analogWrite(PWMz, abs(MnTest));
    if (Mn > 0)
        dictertion_MotorM1(2);
    else if (Mn < 0)
        dictertion_MotorM1(1);
    else if (Mn == 0)
        dictertion_MotorM1(0);
}

void dictertion_MotorM1(int Dir)
{
    switch (Dir)
    {
        case 0 :
            digitalWrite(IN1, HIGH);
            digitalWrite(IN2, HIGH);
            break;
        case 1 :

```

```

digitalWrite(IN1, HIGH);
digitalWrite(IN2, LOW);
break;
}

case 2 :
digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);
break;
}
}

void setup()
{
Serial.begin(57600);
client.setServer(server, 1883);
client.setCallback(callback);
Ethernet.begin(mac, ip);
InitEncoders();
Serial.println("Encoders Initialized...");
clearEncoderCount();
Serial.println("Encoders Cleared...");
delay(1000);
Init_PWM();
Init_PID_Control();
Init_TIMER1_interuppt();
Init_TIMER3_interuppt();

mData =
"Setpoint1_58035:Setpoint2_58035:Setpoint3
_58035:Setpoint4_58035";
}

void loop()
{
delay(100);
}

void InitEncoders()
{
pinMode(slaveSelectEnc1, OUTPUT);
delay(1);
digitalWrite(slaveSelectEnc1, HIGH);
SPI.begin();
digitalWrite(slaveSelectEnc1, LOW);
delay(1);
SPI.transfer(0x88);
SPI.transfer(0x03);
delay(1);
digitalWrite(slaveSelectEnc1, HIGH);
delay(1);

}

long readEncoder()
{
}

```

```

unsigned int count_1, count_2, count_3,
count_4;

long count_value;

digitalWrite(slaveSelectEnc1, LOW);

SPI.transfer(0x60);

count_1 = SPI.transfer(0x00);

count_2 = SPI.transfer(0x00);

count_3 = SPI.transfer(0x00);

count_4 = SPI.transfer(0x00);

digitalWrite(slaveSelectEnc1, HIGH);

count_value = (count_1 << 8) +
count_2;

count_value = (count_value << 8) +
count_3;

count_value = (count_value << 8) +
count_4;

delayMicroseconds(100);

return count_value;
}

void clearEncoderCount()
{
    digitalWrite(slaveSelectEnc1, LOW
    SPI.transfer(0x98);

    SPI.transfer(0x00);

    SPI.transfer(0x00);
}
}

void callback(char* topic, byte* payload,
unsigned int length)

{
    mData = "";
    for (int i=0;i<length;i++)
    {
        mData += (char)payload[i];
    }
    Init_PID_Control();

    setpoint =
    mqttQueryString("Setpoint1_",mData).toInt();

    pulse = setpoint-currentPos;

    mMode = 0;
}

void reconnect()
{
}

```

```

{
}

if (client.connect("Board1"))

    mstop = mstr1.indexOf(":", mstart);

{
}

client.subscribe("Test2");

client.publish("Test1", "Board1 is

connection");

}

else

{

delay(1000);

}

}

}

String mqttQueryString(String qstr1, String

mstr1)

{
}

int mstart, mstop;

String mresult = " ";

qstr1 = qstr1;

mstart = mstr1.indexOf(qstr1);

if (mstart == -1)

{
}

mresult = "";

return mresult;
}

}

void Init_TIMER1_interuppt()

noInterrupts();

TCCR1A = 0;

TCCR1B = 0;

TCNT1 = 0;

TCCR1B |= (1 << CS12);

TIMSK1 |= (1 << TOIE1);

interrupts();

}

void Init_TIMER3_interuppt()

```

```

{

noInterrupts();

TCCR3A = 0;

TCCR3B = 0;

TCNT3 = Timerz;                                Slave2_Spiderbots' Code

TCCR3B |= (1 << CS32);

TIMSK3 |= (1 << TOIE3);

interrupts();

}

void Rez()

{

if (!client.connected())

{

    noInterrupts();

    reconnect();

    interrupts();

}

}

TCCR3A = 0;

TCCR3B = 0;

TCNT3 = Timerz;                                Slave2_Spiderbots' Code

#include <SPI.h>

#include <Ethernet.h>

#include <PubSubClient.h>

EthernetClient ethClient;

PubSubClient client(ethClient);

byte mac[] = { 0xDE, 0xED, 0xBA, 0xFE, 0xFE, 0xEF };

IPAddress ip(158, 108, 5, 2);

IPAddress server(158, 108, 5, 6);

const char* mqtt_server =

"158.108.5.6:1883";

const char* Board = "Board2";

#define PWMz      6

#define IN1       5

#define IN2       4

#define T        0.1

int Mn_max = 30;

int Mn_min = -Mn_max;

const int slaveSelectEnc1 = 53;

double Timerz = 59200;

```

```

float deltaTime ;
int L1_Home = 493;

double Velocity ;
int L2_Home = 493;

long currentPos;
int L3_Home = 493;

long deltaPos ;
int L4_Home = 493;

float oldTime = 0;
long L1 = sqrt(sq(x - A) + sq(YY - y - A) +
sq(ZZ - z));

long oldPos = 0;
long L2 = sqrt(sq(XX - x - A) + sq(YY - y - A) +
sq(ZZ - z));

long setpoint = 0;
long L3 = sqrt(sq(x - A) + sq(y - A) + sq(ZZ -
z));

double Cn, Mn, Mn1, Mn2, En, En1, En2;
long L4 = sqrt(sq(XX - x - A) + sq(y - A) +
sq(ZZ - z));

double Kp = 0;
void Init_PWM()

double Ki = 0;
double Kd = 0;
double counter = 0;
void Init_PWM()

double K0 = 0;
double K1 = 0;
double K2 = 0;
double Rn = 0;
double pulse ;
void Init_PWM()

int mMode = 0;
void Init_PID_Control()

String mData;
char mCharOutput[10];
double FactorPulseperCm = 117.72 ;
double XX = 560, YY = 560, ZZ = 400, A =
8.5;
double x = 280, y = 280, z = 90;
Mn_max = 30;
Mn_min = -Mn_max;
En2 = 0;
En1 = 0;
Mn = 0;

```

```

Mn1      = 0;                      Kp = 0.1;
Mn2      = 0;                      Ki = 0.003;
counter   = 0;                      Kd = 0.0001;
}

void Init_VeloPID()
{
    Serial.println(pulse);
    if (pulse > 0)
    {
        Rn = pulse * 0.001;
        Kp = abs(pulse * 0.0001);
        Ki = abs(pulse * 0.00005);
    }
    else //pull
    {
        Rn = pulse * 0.001;
        Kp = abs(pulse * 0.00025);
        Ki = abs(pulse * 0.00012);
        Kd = 0.005;
    }
    void Init_PosPID()
    {
        Rn = setpoint;
    }
}

void Init_PID_Gain()
{
    K0 = Kp + (Ki * T) + (Kd / T);
    K1 = (-Kp) - (2 * Kd / T);
    K2 = (Kd / T);
}

ISR(TIMER1_OVF_vect)
{
    Rez();
    //String PPos ="Setpoint1_" + String(X);
    String PPos = String(currentPos);
    for (int i = 0; i < PPos.length(); i++)
    {
        mCharOutput[i] = PPos[i];
    }
    client.publish("L2", mCharOutput);
    TCNT3 = 0;
}

ISR(TIMER3_OVF_vect
{
}

```

```

PID_Control_Run();
Mn = Mn1 + (K0 * En) + (K1 * En1) + (K2 *
En2);
if ( abs(En) <= 100 && mMode == 1 )
{
    Mn = 0;
}
void PID_Control_Run()
{
    deltaTime = CalculateDeltaTime();
    deltaPos = CalculateDeltaPos();
    Velocity = ((deltaPos / deltaTime) * 1000 *
60) / 4096.0;
    if ( abs(abs(currentPos) - abs(setpoint)) >=
1000 && mMode == 0 )
    {
        mMode = 0;
        Init_VeloPID();
        Cn = Velocity;
    }
    else
    {
        mMode = 1;
        Init_PosPID();
        Cn = currentPos;
    }
    Init_PID_Gain();
    En = Rn - Cn;
}

Mn2 = Mn1;
Mn1 = Mn;
counter++;
}

float CalculateDeltaTime()
{
    float currentTime = millis();
    float deltaTime = currentTime - oldTime;
    oldTime = currentTime;
    return deltaTime;
}

long CalculateDeltaPos()
{
    currentPos = ((readEncoder() * 1024) /
3600.0) + (L2_Home * FactorPulseperCm);
}

```

```

deltaPos = currentPos - oldPos;                                }

oldPos = currentPos;                                         }

return deltaPos;                                              Serial.println("Mn_Test: " + String(MnTest) );

}                                                               analogWrite(PWMz, abs(MnTest));

void PWM()                                                       if (Mn > 0)

{                                                               dictertion_MotorM1(1);

double MnTest;                                                 else if (Mn < 0)

MnTest = Mn;                                                   dictertion_MotorM1(2);

if (Mn > Mn_max)                                            else if (Mn == 0)

{                                                               dictertion_MotorM1(0);

if (mMode == 0)                                              }

Mn_max = Mn;                                                 void dictertion_MotorM1(int Dir)

else                                                       {

{                                                               switch (Dir)

MnTest = Mn_max;                                           {

}

case 0 :                                                 digitalWrite(IN1, HIGH);

}

digitalWrite(IN2, HIGH);                                     break;

else if (Mn < Mn_min && counter >= 10)                  case 1 :

{                                                               digitalWrite(IN1, HIGH);

if (mMode == 0)                                            digitalWrite(IN2, LOW);

Mn_min = Mn;                                               break;

else                                                       case 2 :

{                                                               }

MnTest = Mn_min;                                         }

```

```

digitalWrite(IN1, LOW);
digitalWrite(IN2, HIGH);
break;
}

void setup()
{
Serial.begin(57600);

client.setServer(server, 1883);

client.setCallback(callback);

Ethernet.begin(mac, ip);

InitEncoders();

Serial.println("Encoders Initialized...");

clearEncoderCount();

Serial.println("Encoders Cleared...");

delay(1000);

Init_PWM();

Init_PID_Control();

Init_TIMER1_interuppt();

Init_TIMER3_interuppt();

mData =
"Setpoint1_58035:Setpoint2_58035:Setpoint3
_58035:Setpoint4_58035";
}

void loop()
{
delay(100);

}

void InitEncoders()
{
pinMode(slaveSelectEnc1, OUTPUT);

delay(1);

digitalWrite(slaveSelectEnc1, HIGH);

SPI.begin();

digitalWrite(slaveSelectEnc1, LOW);

SPI.transfer(0x88);

SPI.transfer(0x03);

digitalWrite(slaveSelectEnc1, HIGH);

delay(1);

}

long readEncoder()
{
unsigned int count_1, count_2, count_3,
count_4;

long count_value;

digitalWrite(slaveSelectEnc1, LOW

SPI.transfer(0x60);

count_1 = SPI.transfer(0x00);

count_2 = SPI.transfer(0x00);
}

```

```

count_3 = SPI.transfer(0x00);                               digitalWrite(slaveSelectEnc1, HIGH);

count_4 = SPI.transfer(0x00);                               }

digitalWrite(slaveSelectEnc1, HIGH);

count_value = (count_1 << 8) +                         void callback(char* topic, byte* payload,
count_2;                                                 unsigned int length)

count_value = (count_value << 8) +                         mData = "";

count_3;                                                 for (int i=0;i<length;i++)

count_value = (count_value << 8) +                         {
count_4;                                                 mData += (char)payload[i];

delayMicroseconds(100);                                 }

return count_value;
}

void clearEncoderCount()
{
    digitalWrite(slaveSelectEnc1, LOW);
    SPI.transfer(0x98);
    SPI.transfer(0x00);
    SPI.transfer(0x00);
    SPI.transfer(0x00);
    SPI.transfer(0x00);
    SPI.transfer(0x00);
    digitalWrite(slaveSelectEnc1, HIGH);
    delayMicroseconds(100);
    digitalWrite(slaveSelectEnc1, LOW);
    SPI.transfer(0xE0);

    Init_PID_Control();
    setpoint = mqttQueryString("Setpoint2_",
mData).toInt();
    pulse = setpoint - currentPos;
    mMode = 0;
}

void reconnect()
{
    if (client.connect("Board2"))
    {
        client.subscribe("Test2");
        client.publish("Test1", "Board2 is
connection");
    }
}

```

```

        mresult = mstr1.substring(mstart +
qstr1.length(), " " );
    }

    delay(1000);

}

}

mresult = mstr1.substring(mstart +
qstr1.length(), mstop);
}

}

return mresult;

String mqttQueryString(String qstr1, String
mstr1)
{
    int mstart, mstop;
    String mresult = " ";
    qstr1 = qstr1;
    mstart = mstr1.indexOf(qstr1);
    if (mstart == -1)
    {
        mresult = "";
        return mresult;
    }
    mstop = mstr1.indexOf(":", mstart);
    if (mstop == -1)
    {
        mresult = mstr1.substring(mstart +
qstr1.length(), " " );
    }
    else
    {
        mresult = mstr1.substring(mstart +
qstr1.length(), mstop);
    }
}

void Init_TIMER1_interupt()
{
    noInterrupts();
    TCCR1A = 0;
    TCCR1B = 0;
    TCNT1  = 0;
    TCCR1B |= (1 << CS12); // 256 prescaler
    TIMSK1 |= (1 << TOIE1);
    interrupts();
}

void Init_TIMER3_interupt()
{
    noInterrupts();
    TCCR3A = 0;
    TCCR3B = 0;
}

```

```

TCNT3 = Timerz;                                IPAddress server(158, 108, 5, 6);

TCCR3B |= (1 << CS32);                         const char* mqtt_server =
TIMSK3 |= (1 << TOIE3);                        "158.108.5.6:1883";
interrupts();                                     const char* Board = "Board3";
                                                       #define PWMz      6
}
                                                       #define IN1       4
                                                       #define IN2       5
                                                       #define T        0.1
                                                       int      Mn_max     = 30;
void Rez()                                         int      Mn_min     = -Mn_max;
{
if (!client.connected())                           const int slaveSelectEnc1 = 53;
{
noInterrupts();                                 double     Timerz = 59200;
reconnect();                                    float    deltaTime ;
interrupts();                                    double   Velocity ;
}
                                                       long   currentPos;
client.loop();                                    long   deltaPos ;
}

Slave3_Spiderbots' Code
#include <SPI.h>                                float oldTime = 0;
#include <Ethernet.h>                            long oldPos = 0;
#include <PubSubClient.h>                          long setpoint = 0;
EthernetClient ethClient;                         double Cn, Mn, Mn1, Mn2, En, En1, En2;
PubSubClient client(ethClient);                   double Kp = 0;
byte mac[] = { 0xDE, 0xED, 0xCA, 0xFE,           double Ki = 0;
0xFF, 0xEF };                                  double Kd = 0;
IPAddress ip(158, 108, 5, 3);                  double counter = 0;

```

```

double      K0 = 0;                      void Init_PWM()
double      K1 = 0;                      {
double      K2 = 0;                      pinMode(IN1, OUTPUT);
double      Rn = 0;                      pinMode(IN2, OUTPUT);
double      pulse ;                     pinMode(PWMz, OUTPUT);
double      Vv = 0 ;                     }

int       mMode = 0;                    void Init_PID_Control()
String    mData;                      {
char mCharOutput[50];                  Mn_max     = 30;
double FactorPulseperCm = 117.72 ;   Mn_min     = -Mn_max;
double XX = 560, YY = 560, ZZ = 400, A = En2       = 0;
8.5;                                  En1       = 0;
double x = 280, y = 280, z = 90;      Mn        = 0;
int L1_Home = 493;                   Mn1      = 0;
int L2_Home = 493;                   Mn2      = 0;
int L3_Home = 493;                   counter   = 0;
int L4_Home = 493;                   }

long L1 =  sqrt(sq(x - A) + sq(YY - y - A) +
sq(ZZ - z));
long L2 =  sqrt(sq(XX - x - A) + sq(YY - y - A)
+ sq(ZZ - z));
long L3 =  sqrt(sq(x - A) + sq(y - A) + sq(ZZ -
z));
long L4 =  sqrt(sq(XX - x - A) + sq(y - A) +
sq(ZZ - z));

}                                     void Init_VeloPID()
{                                       if (pulse > 0)
{
Rn =  pulse * 0.001;
Kp =  abs(pulse * 0.0001);
Ki =  abs(pulse * 0.00005);
}

```

```

        }

        TCNT3 = 0;

    else

        Rez();

    {

        String PPos = " M " + String(mMode) + " En
        " + String(currentPos) + " Set " +
        String(setpoint) + " Mn " + String(Mn);

        PPos.toCharArray(mCharOutput,50);
        client.publish("L3", mCharOutput);

        Kp = abs(pulse * 0.00025);
        Ki = abs(pulse * 0.00012);
        Kd = 0.005;
    }

}

ISR(TIMER3_OVF_vect)
{
    void Init_PosPID()
    {
        Rn = setpoint;
        Kp = 0.1;
        Ki = 0.003;
        Kd = 0.0001;
    }

    void PID_Control_Run()
    {
        TCNT3 = Timerz;
        deltaTime = CalculateDeltaTime();
        deltaPos = CalculateDeltaPos();
        Velocity = ((deltaPos / deltaTime) * 1000 *
        60) / 4096.0;
        if ( abs(abs(currentPos) - abs(setpoint)) >=
        1000 && mMode == 0 )
        {
            mMode = 0;
            Init_VeloPID();
            Cn = Velocity;
        }
    }
}

```

```

        }

        float currentTime = millis();

    else

        float deltaTime = currentTime - oldTime;

    {

        oldTime = currentTime;

        return deltaTime;

    }

    long CalculateDeltaPos()

    {

        currentPos = (readEncoder()) + (L3_Home *

FactorPulseperCm);

        deltaPos = currentPos - oldPos;

        oldPos = currentPos;

        return deltaPos;

    }

    void PWM()

    {

        double MnTest;

        MnTest = Mn;

        if (Mn > Mn_max)

        {

            if (mMode == 0)

                Mn_max = Mn;

            else

                MnTest = Mn_max;

        }

    }

    float CalculateDeltaTime()

    {

```

```

        }

        digitalWrite(IN1, HIGH);

    }

    else if (Mn < Mn_min && counter >= 10)

        break;

    {

        if (mMode == 0)

            Mn_min = Mn;

        else

            break;

    {

        MnTest = Mn_min;

        digitalWrite(IN1, LOW);

    }

    break;

};

}

if (Mn > 0)

    }

dictertion_MotorM1(1);

else if (Mn < 0)

    dictertion_MotorM1(2);

else if (Mn == 0)

    dictertion_MotorM1(0);

}

void dictertion_MotorM1(int Dir)

{

    switch (Dir)

    {

        case 0 :

            }

            digitalWrite(IN2, HIGH);

        }

        case 1 :

            digitalWrite(IN1, HIGH);

            digitalWrite(IN2, LOW);

        }

        break;

    }

    }

    void setup()

    {

        mData =

        "Setpoint1_58035:Setpoint2_58035:Setpoint3

        _58035:Setpoint4_58035";

        Serial.begin(57600);

        client.setServer(server, 1883);

        client.setCallback(callback);

        Ethernet.begin(mac, ip);

        InitEncoders();

        clearEncoderCount();

    }
}

```

```

delay(100);                                unsigned int count_1, count_2, count_3,
                                         count_4;
Init_PWM();                                 long count_value;
                                         digitalWrite(slaveSelectEnc1, LOW);
                                         SPI.transfer(0x60);
                                         count_1 = SPI.transfer(0x00);
                                         count_2 = SPI.transfer(0x00);
                                         count_3 = SPI.transfer(0x00);
                                         count_4 = SPI.transfer(0x00);
                                         digitalWrite(slaveSelectEnc1, HIGH);
                                         count_value = (count_1 << 8) + count_2;
                                         count_value = (count_value << 8) + count_3;
                                         count_value = (count_value << 8) + count_4;
                                         delayMicroseconds(100);
                                         return count_value;
}

void clearEncoderCount()
{
    digitalWrite(slaveSelectEnc1, LOW);
    SPI.transfer(0x98);
    SPI.transfer(0x00);
    SPI.transfer(0x00);
}

```

```

    SPI.transfer(0x00);                                if (String(topic) == "Test3")

    SPI.transfer(0x00);                                {

digitalWrite(slaveSelectEnc1, HIGH);                Vv = mqttQueryString("Velocity3_",
delayMicroseconds(100);                           mData).toInt();

digitalWrite(slaveSelectEnc1, LOW);                  }

SPI.transfer(0xE0);                                }

digitalWrite(slaveSelectEnc1, HIGH);                  void reconnect()

}                                                 {                                     while (!client.connected())

void callback(char* topic, byte* payload,          {
unsigned int length)                            if (client.connect("Board3"))

{                                                 {

mData = "";                                     client.subscribe("Test2");

for (int i = 0; i < length; i++)                 client.publish("Test1", "Board3 is

{                                                 connection");

mData += (char)payload[i];                         }

}

if (String(topic) == "Test2")                     else

{                                                 {

Init_PID_Control();                            delay(10);

setpoint = mqttQueryString("Setpoint3_",
mData.toInt());                                }

pulse = setpoint - currentPos;                  }

mMode = 0;                                         String mqttQueryString(String qstr1, String

}                                                 mstr1)

```

```

    {
        noInterrupts();
        int mstart, mstop;
        TCCR1A = 0;
        String mresult = " ";
        TCCR1B = 0;
        qstr1 = qstr1;
        TCNT1 = 0;
        mstart = mstr1.indexOf(qstr1);
        TCCR1B |= (1 << CS12);
        if (mstart == -1)
            TIMSK1 |= (1 << TOIE1);  overflow
        {
            interrupt
            interrupts();
        }
        return mresult;
    }

    mstop = mstr1.indexOf(":", mstart);
    void Init_TIMER3_interuppt()
    {
        if (mstop == -1)
            noInterrupts();
        {
            TCCR3A = 0;
            mresult = mstr1.substring(mstart +
            qstr1.length(), " ");
            TCCR3B = 0;
            TCNT3 = Timerz;
            }
            TCCR3B |= (1 << CS32);
            TIMSK3 |= (1 << TOIE3);
            else
            {
                mresult = mstr1.substring(mstart +
                qstr1.length(), mstop);
                void Rez()
                {
                    if (!client.connected())
                    {
                        reconnect();
                    }
                }
            }
        }
    }

    void Init_TIMER1_interuppt() {

```

```

client.loop();

}

Slave4_Spiderbots' Code

#include <SPI.h>

#include <Ethernet.h>

#include <PubSubClient.h>

EthernetClient ethClient;

PubSubClient client(ethClient);

byte mac[] = { 0xDE, 0xED, 0xBA, 0xFE, 0xFF,
0xEF };

IPAddress ip(158, 108, 5, 4);

IPAddress server(158,108,5,6);

const char* mqtt_server =
"158.108.5.6:1883";

const char* Board = "Board4";

#define PWMz 6

#define IN1 4

#define IN2 5

#define T 0.1

int Mn_max = 30;

int Mn_min = -Mn_max;

const int slaveSelectEnc1 = 53;

double Timerz = 59200;

```

```

float deltaTime ;

double Velocity ;

long currentPos;

long deltaPos ;

float oldTime = 0;

long oldPos = 0;

long setpoint = 0;

double Cn, Mn, Mn1,Mn2, En, En1, En2;

double Kp = 0;

double Ki = 0;

double Kd = 0;

double counter = 0;

double K0 = 0;

double K1 = 0;

double K2 = 0;

double Rn = 0;

double pulse ;

int mMode = 0;

String mData;

char mCharOutput[50];

double FactorPulseperCm = 117.72 ;

double XX=560,YY=560,ZZ=400,A=8.5;

double x=280,y=280,z=90;

int L1_Home = 493;

```

```

int L2_Home = 493;                                }

int L3_Home = 493;                                void Init_VeloPID()

int L4_Home = 493;                                {

long L1 = sqrt(sq(x-A)+sq(YY-y-A)+sq(ZZ-z));    Serial.println(pulse);

long L2 = sqrt(sq(XX-x-A)+sq(YY-y-A)+sq(ZZ-      if(pulse>0)  {

z));                                              Rn = pulse*0.001;

long L3 = sqrt(sq(x-A)+sq(y-A)+sq(ZZ-z));        Kp = abs(pulse*0.0001);

long L4 = sqrt(sq(XX-x-A)+sq(y-A)+sq(ZZ-z));      Ki = abs(pulse*0.00005);

void Init_PWM()                                     }

{                                                 else

pinMode(IN1, OUTPUT);                            {

pinMode(IN2, OUTPUT);                            Rn = pulse * 0.001;

pinMode(PWMz, OUTPUT);                          Kp = abs(pulse * 0.00025);

}                                                 Ki = abs(pulse * 0.00012);

void Init_PID_Control()                         Kd = 0.005;

{                                                 }

Mn_max     = 30;                                }

Mn_min     = -Mn_max;                           void Init_PosPID()

En2        = 0;                                 {

En1        = 0;                                 Rn = setpoint;

Mn         = 0;                                 Kp = 0.1;

Mn1        = 0;                                 Ki = 0.003;

Mn2        = 0;                                 Kd = 0.0001;

counter    = 0;                                }

}

```

```

void Init_PID_Gain()
{
    K0 = Kp + (Ki * T) + (Kd / T);
    K1 = (-Kp) - (2 * Kd / T);
    K2 = (Kd / T);
}

ISR(TIMER1_OVF_vect)
{
    TCNT3 = 0;
    Rez();
    String PPos = " M " + String(mMode) + " En
" + String(currentPos) + " Set " +
String(setpoint) + " Mn " + String(Mn);
    PPos.toCharArray(mCharOutput,50);
    client.publish("L4",mCharOutput);
}

ISR(TIMER3_OVF_vect)
{
    PID_Control_Run();
    TCNT3 = Timerz;
}

void PID_Control_Run()
{
    deltaTime = CalculateDeltaTime();
    deltaPos = CalculateDeltaPos();
    Velocity =
((deltaPos/deltaTime)*1000*60)/4096.0;
    if( abs(abs(currentPos)-
abs(setpoint))>=1000 && mMode == 0 )
    {
        mMode = 0;
        Init_VeloPID();
        Cn = Velocity;
    }
    else
    {
        mMode = 1;
        Init_PosPID();
        Cn = currentPos;
    }
    Init_PID_Gain();
    En = Rn - Cn;
    Mn = Mn1 + (K0 * En) + (K1 * En1) + (K2 *
En2) ;
    if( abs(En)<=100 && mMode == 1 )
    {
        Mn=0;
    }
}

```

```

PWM();
{
    En2 = En1;
    En1 = En;
    Mn2 = Mn1;
    Mn1 = Mn;
    counter++;
}

float CalculateDeltaTime()
{
    float currentTime = millis();
    float deltaTime = currentTime - oldTime;
    oldTime = currentTime;
    return deltaTime;
}

long CalculateDeltaPos()
{
    currentPos = ((-1)*readEncoder())+(L4_Home*FactorPulseperCm);
    deltaPos = currentPos - oldPos;
    oldPos = currentPos;
}

return deltaPos;
}

void PWM()
{
    double MnTest;
    MnTest = Mn;
    if (Mn > Mn_max)
    {
        if (mMode == 0)
            Mn_max = Mn;
        else
            MnTest = Mn_max;
    }
    else if (Mn < Mn_min && counter >= 10)
    {
        if (mMode == 0)
            Mn_min = Mn;
        else
        {
            MnTest = Mn_min;
        }
    }
    if(Mn>0)
        dictertion_MotorM1(2);
}

```

```

else if(Mn<0)                                {

    dictortion_MotorM1(1);                      Serial.begin(57600);

    else if(Mn==0)                            client.setServer(server, 1883);

        dictortion_MotorM1(0);                  client.setCallback(callback);

    }                                            Ethernet.begin(mac, ip);

void dictortion_MotorM1(int Dir)               InitEncoders();      Serial.println("Encoders

{                                               Initialized...");

switch (Dir)                                 clearEncoderCount();

{                                                 Serial.println("Encoders Cleared...");

    case 0 :                               delay(1000);

        digitalWrite(IN1, HIGH);            Init_PWM();

        digitalWrite(IN2, HIGH);            Init_PID_Control();

        break;                           Init_TIMER1_interuppt();

    case 1 :                               Init_TIMER3_interuppt();

        digitalWrite(IN1, HIGH);            mData =
        digitalWrite(IN2, LOW);             "Setpoint1_58035:Setpoint2_58035:Setpoint3
                                         _58035:Setpoint4_58035";

        break;                           }

    case 2 :                               void loop()

        digitalWrite(IN1, LOW);           {

        digitalWrite(IN2, HIGH);          delay(100);

        break;                         }

}

void setup()                                void InitEncoders()

{                                              pinMode(slaveSelectEnc1, OUTPUT);
}

```

```

delay(1);

digitalWrite(slaveSelectEnc1,HIGH);

SPI.begin();

digitalWrite(slaveSelectEnc1,LOW);

SPI.transfer(0x88);

SPI.transfer(0x03);

digitalWrite(slaveSelectEnc1,HIGH);

delay(1);

}

long readEncoder()

{

    unsigned int count_1, count_2, count_3,
count_4;

    long count_value;

    digitalWrite(slaveSelectEnc1,LOW);

    SPI.transfer(0x60);

    count_1 = SPI.transfer(0x00);

    count_2 = SPI.transfer(0x00);

    count_3 = SPI.transfer(0x00);

    count_4 = SPI.transfer(0x00);

    digitalWrite(slaveSelectEnc1,HIGH);

    count_value = (count_1 << 8) +
count_2;

    count_value = (count_value << 8) +
count_3;
}
count_value = (count_value << 8) +
count_4;

delayMicroseconds(100);

return count_value;

}

void clearEncoderCount()

{
    digitalWrite(slaveSelectEnc1,LOW);

    SPI.transfer(0x98);

    SPI.transfer(0x00);

    SPI.transfer(0x00);

    SPI.transfer(0x00);

    SPI.transfer(0x00);

    SPI.transfer(0x00);

    digitalWrite(slaveSelectEnc1,HIGH);

    delayMicroseconds(100);

    digitalWrite(slaveSelectEnc1,LOW);

    SPI.transfer(0xE0);

    digitalWrite(slaveSelectEnc1,HIGH);

}

void callback(char* topic, byte* payload,
unsigned int length)

{
    mData = "";

    for (int i=0;i<length;i++)

```

```

{
    mData += (char)payload[i];
}

Init_PID_Control();

setpoint =
mqttQueryString("Setpoint4_",mData).toInt();

pulse = setpoint-currentPos;

mMode = 0;

}

void reconnect()
{
    if (client.connect("Board4"))

        client.subscribe("Test2");

        client.publish("Test1","Board4 is
connection");

    }

    else
    {
        delay(1000);
    }
}

}

String mqttQueryString(String qstr1,String
mstr1)
{
    int mstart,mstop;
    String mresult = " ";
    qstr1=qstr1;
    mstart=mstr1.indexOf(qstr1);

    if (mstart== -1)

    {
        mresult="";
        return mresult;
    }

    mstop=mstr1.indexOf(":",mstart);

    if (mstop== -1)

    {
        mresult=mstr1.substring(mstart+qstr1.length(
)," ");
    }

    else
    {
        mresult=mstr1.substring(mstart+qstr1.length(
),mstop);
    }
}

return mresult;
}

```

```

void Init_TIMER1_interuppt() {
    reconnect();
    noInterrupts();
    TCCR1A = 0;
    } }

TCCR1B = 0;
    client.loop();
    TCNT1  = 0;
    }

TCCR1B |= (1 << CS12);
    Tools_Agriculture's Code

TIMSK1 |= (1 << TOIE1);
    #include <SPI.h>

interrupts();
    #include <Ethernet.h>

}
    #include <PubSubClient.h>

void Init_TIMER3_interuppt()
{
    EthernetClient ethClient;
    PubSubClient client(ethClient);

    noInterrupts();
    TCCR3A = 0;
    TCCR3B = 0;
    byte mac[] = { 0xD1, 0xED, 0xAB, 0xFE, 0xFE,
    0xE8 };
    TCNT3  = Timerz;
    IPAddress ip(158, 108, 5, 102);
    TIMSK3 |= (1 << TOIE3);
    IPAddress server(158, 108, 5, 6);
    interrupts();
    const char* mqtt_server =
    "158.108.5.6:1883";
}

void Rez()
{
    if (!client.connected())
    {
        noInterrupts();
        String mData;
        int A = 0, B = 0, C = 0, D = 0;
        #define INA1 2
    }
}

```

```

#define INB1 3

void loop()

#define INA2 4

#define INB2 5

#define INA3 8

#define INB3 9

#define INA4 10

#define INA4 11

#define INB4 11

#define PWMz 13

#define PWMz 13

void setup()

{

Serial.begin(57600);

client.setServer(server, 1883);

client.setCallback(callback);

Ethernet.begin(mac, ip);

Init_Output();

delay(1000);

}

void loop()

{

Rez();

client.loop();

analogWrite(PWMz, 255);

Control_Tool1();

Control_Tool2();

Control_Tool3();

Control_Tool4();

}

void Control_Tool1()

{

}

void Control_Tool2()

{

}

void Control_Tool3()

{

}

void Control_Tool4()

{

}

void callback(char* topic, byte* payload, unsigned int length)

{

mData = "";

Serial.print("Message arrived [");

Serial.print(topic);

}

```

```

Serial.print("] ");

}

mstop = mstr1.indexOf(":", mstart);

if (mstop == -1)

{

mresult = mstr1.substring(mstart + qstr1.length(), " " );

}

else

{

mresult = mstr1.substring(mstart + qstr1.length(), mstop);

}

return mresult;
}

void Init_Output ()

{

pinMode(INA1, OUTPUT);

pinMode(INB1, OUTPUT);

pinMode(INA2, OUTPUT);

pinMode(INB2, OUTPUT);

pinMode(INA3, OUTPUT);

}

int mstart, mstop;

String mresult = "";

qstr1 = qstr1;

mstart = mstr1.indexOf(qstr1);

if (mstart == -1)

{

mresult = "";

return mresult;
}

```

```

pinMode(INB3, OUTPUT);                                }

pinMode(INA4, OUTPUT);                                void Control_Tool2()

pinMode(INB4, OUTPUT);                                {

pinMode(PWMz, OUTPUT);                                switch (B)

}                                         {                                     case 1:

void Control_Tool1()                                digitalWrite(INA2, 0);

{                                         digitalWrite(INB2, 1);

break;                                         case 2:

switch (A)                                         digitalWrite(INA2, 1);

{                                         digitalWrite(INB2, 0);

case 1:                                         break;

digitalWrite(INA1, 0);                                default:

digitalWrite(INB1, 1);                                digitalWrite(INA2, 0);

break;                                         digitalWrite(INB2, 0);

case 2:                                         break;

digitalWrite(INA1, 1);                                default:

digitalWrite(INB1, 0);                                digitalWrite(INB2, 0);

break;                                         }

default:                                         void Control_Tool3()

digitalWrite(INA1, 0);                                {

digitalWrite(INB1, 0);                                switch (C)

break;                                         {

```

```

case 1:                               digitalWrite(INB4, 0);

digitalWrite(INA3, 0);                 break;

digitalWrite(INB3, 1);                 default:

break;                                digitalWrite(INA4, 0);

case 2:                               digitalWrite(INB4, 0);

digitalWrite(INA3, 1);                 break;

digitalWrite(INB3, 0);                 }

break;                                }

default:                             void Rez()

digitalWrite(INA3, 0);                 {

digitalWrite(INB3, 0);                 if (!client.connected())

break;                                {

}

reconnect();                         }

}

void Control_Tool4()                  }

{

void reconnect()                      {

switch (D)                           {

{

case 1:                            {

digitalWrite(INA4, 0);               if (client.connect("Board_Tool"))

digitalWrite(INB4, 1);               {

break;                                Serial.println("connected");

}

case 2:                            client.subscribe("Test3");

digitalWrite(INA4, 1);               }

```

```
client.publish("Test1", "Board_Tool is
connection");

}

else

{

Serial.print("failed, rc=");

Serial.print(client.state());

Serial.println(" try again in 5 seconds");

//Wait 5 seconds before retrying

delay(5000);

}

}

}
```

ประวัตินิสิต

1. ชื่อ สิรภัทร บุญจันทร์ เลขประจำตัวนิสิต 5910501127

ภาควิชาบริหารธุรกิจ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์

ที่อยู่ปัจจุบัน 13 ซอยฉิมพลี 27 ถนนฉิมพลี แขวงฉิมพลี เขตตลิ่งชัน กรุงเทพมหานคร 10170

โทรศัพท์เคลื่อนที่ 0994483355

E-Mail: siraphat.2541@gmail.com

ระดับการศึกษา:

คณวุฒิการศึกษา	จากโรงเรียน/สถาบัน	ปีการศึกษาที่จบ
มัธยมศึกษาตอนปลาย	โรงเรียนโพธิสารพิทยากร	2558
มัธยมศึกษาตอนต้น	โรงเรียนโพธิสารพิทยากร	2555

2. ชื่อ อัครร่วทธย พงศ์วิรัตน์ เลขประจำตัวนิสิต 5910501178

ภาควิชาบริหารธุรกิจ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์

ที่อยู่ปัจจุบัน 148/423 ซอย 19 ถนนรามคำแหง 190 มีนบุรี กรุงเทพมหานคร 10510

โทรศัพท์เคลื่อนที่ 0877032041

E-Mail: akaravit.p@ku.th

ระดับการศึกษา:

คณวุฒิการศึกษา	จากโรงเรียน/สถาบัน	ปีการศึกษาที่จบ
มัธยมศึกษาตอนปลาย	นวมินทรราชินูทิศ เตรียมอุดมศึกษาน้อมเกล้า	2558
มัธยมศึกษาตอนต้น	นวมินทรราชินูทิศ เตรียมอุดมศึกษาน้อมเกล้า	2555

3. ชื่อ สุทธิพงศ์ สว่าง เลขประจำตัวนิสิต 5910503341

ภาควิชาบริหารธุรกิจ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์

ที่อยู่ปัจจุบัน 323 ถนนไชย 4 เขตลาดพร้าว แขวงลาดพร้าว กรุงเทพมหานคร 10230

โทรศัพท์เคลื่อนที่ 0613855485

E-Mail: nonboss01@hotmail.com

ระดับการศึกษา:

คณวุฒิการศึกษา	จากโรงเรียน/สถาบัน	ปีการศึกษาที่จบ
มัธยมศึกษาตอนปลาย	สาธิตมหาวิทยาลัยรามคำแหง	2558
มัธยมศึกษาตอนต้น	สาธิตมหาวิทยาลัยรามคำแหง	2555