ตัวควบคุมหุ่นยนต์ด้วยพีแอลซี: กรณีศึกษาของหุ่นยนต์ประเภทสการา

PLC-Based Robot Controller: A Case Study of SCARA Robot Control

นายเจ้อหลิน แซ่เฉิน

นายเมฆวัฒ อภิรามเมธา

นางสาวศิรัญญา ศรีท่าพระ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมอัตโนมัติ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2560

ตัวควบคุมหุ่นยนต์ด้วยพีแอลซี: กรณีศึกษาของหุ่นยนต์ประเภทสการา

PLC-Based Robot Controller: A Case Study of SCARA Robot Control

นายเจ๋อหลิน แซ่เฉิน

นายเมฆวัฒ อภิรามเมธา

นางสาวศิรัญญา ศรีท่าพระ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมอัตโนมัติ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2560

PLC-Based Robot Controller: A Case Study of SCARA Robot Control

Mr. Ze Lin Chen
Mr. Mekawat Apirammeta
Miss Sirunya Srithaphra

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THR DEGREE OF
BECHELOR OF ENGINEERING IN AUTOMATION ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG
ACADEMIC YEAR 2017

ภาควิชาวิศวกรรมการวัดคุมและควบคุม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

หัวข้อปริญญานิพนธ์ ตัวควบคุมหุ่นยนต์ด้วยพีแอลซี: กรณีศึกษาของหุ่นยนต์ประเภทสการา

PLC-Based Robot Controller: A Case Study of SCARA Robot Control

นักศึกษาผู้จัดทำ

นายเจ๋อหลิน	แซ่เฉิน	รหัสประจำตัว 57010225
นายเมฆวัฒ	อภิรามเมธา	รหัสประจำตัว 57011022
นางสาวศิรัญญา	ศรีท่าพระ	รหัสประจำตัว 57011240

ปริญญา วิศวกรรมศาสตรบัณฑิต

สาขาวิชา วิศวกรรมอัตโนมัติ

ปีการศึกษา 2560

อาจารย์ผู้ควบคุมปริญญานิพนธ์	ลายมือชื่อ
รศ.ประภาษ อุคคกิมาพันธุ์	
รศ.ดร.อัมพวัน จุลเสรีวงศ์	

หัวข้อปริญญานิพนธ์	ตัวควบคุมหุ่นยนต์ด้วยพีแอลซี: กรณีศึกษาของหุ่นยนต์ประเภทสการา		
	PLC-Based Robot Controller: A Case Study of SCARA Robot Control		
นักศึกษาผู้จัดทำ	นายเจ๋อหลิน	แซ่เฉิน	**รหัสประจำตัว** 57010225
	นายเมฆวัฒ	อภิรามเมธา	**รหัสประจำตัว** 57011022
	นางสาวศิรัญญา	ศรีท่าพระ	**รหัสประจำตัว** 57011240
อาจารย์ที่ปรึกษา	รศ.ประภาษ	อุคคกิมาพันธุ์	
	รศ.ดร.อัมพวัน	จุลเสรีวงศ์	
ปีการศึกษา	2560		

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้มีวัตถุประสงค์เพื่อใช้พีแอลซี (PLC) แทนตัวควบคุมหุ่นยนต์เพื่อนำหุ่นยนต์อุตสาหกรรมที่เคยใช้งานแล้วนำกลับมาใช้ใหม่ โดยใช้หุ่นยนต์ประเภท SCARA แบบสี่ข้อต่อและสองท่อนแขน ที่มีความยาวแขน 650 มิลลิเมตร และยกน้ำหนักสูงสุด 3 กิโลกรัม ในการทดสอบความสามารถของตัวควบคุมด้วยพีแอลซีที่นำเสนอ ในการควบคุมแขนหุ่นยนต์ที่ศึกษามีการสร้างชุดฟังก์ชันบล็อคของพีแอลซี ยี่ห้อมิตซูบิชิ รุ่น MELSEC Q-Series เพื่อควบคุมห้าโหมดการทำงานได้แก่ การควบคุมการทำงานแบบ Jog การควบคุมตำแหน่งอัตโนมัติ การเลือกเหตุการณ์ M-code การควบคุมการเคลื่อนที่แบบพิกัด และการควบคุมเส้นทางการเคลื่อนที่ ในส่วนการวางแผนการเคลื่อนที่ของแขนหุ่นยนต์และฐานข้อมูลที่ใช้ในการกำหนดเส้นทางการเคลื่อนที่ถูกสร้างขึ้นด้วยโปรแกรม RobotStudio และโปรแกรม Microsoft SQL Server ร่วมกับ WW InTouch ตามลำดับ นอกจากนี้ ยังมีการสร้างส่วนที่ติดต่อกับผู้ใช้งานด้วยโปรแกรม GT Designer 3 ในกรณีศึกษาการควบคุมหุ่นยนต์ได้ใช้เซอร์โวมอเตอร์ยี่ห้อมิตซูบิชิ สี่ชุด รุ่น HG-KR Series เพื่อควบคุมการเคลื่อนที่ของหุ่นยนต์ตามคำสั่งที่ได้รับจากพีแอลซี และผลการทดลองยืนยันได้ว่าตัวควบคุมที่นำเสนอสามารถทำงานได้อย่างถูกต้อง

Thesis Title	PLC-Based Robot Controller: A Case Study of SCARA Robot Control
Authors	Mr. Ze Lin Chen
	Mr. Mekawat Apirammeta
	Miss Sirunya Srithaphra
Thesis Advisors	Assoc.Prof. Prapart Ukakimaparn
	Assoc.Prof.Dr. Amphawan Julsereewong
Year	2017

ABSTRACT

This thesis aims at utilizing a programmable logic controller (PLC) to replace a robot controller for reusing a discarded industrial robot. An existing four-joint and two-link SCARA arm having arm length of 650 mm and maximum pay load of 3 kg is used to verify the workability of the proposed PLC-based robot controller. The function blocks of the PLC branded Mitsubishi MELSEC Q-Series for five operating modes (jog operation, automatic position control, M-code event selection, coordinate movement control, and trajectory-path movement control) are created for the controlled robot arm. The trajectory paths for designed motion planning are built by using the RobotStudio. The motion planning database is created by utilizing the Microsoft SQL server and WW InTouch. In addition, the graphical user interface is also implemented by employing GT Designer 3. Four servo motors branded Mitsubishi HG-KR Series are employed as the final control elements. Experimental results are confirmed that the proposed controller can operate correctly.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี เนื่องด้วยความเมตตาอนุเคราะห์และความช่วยเหลือดูแลใส่ใจเป็นอย่างดีจากหลาย ๆ ฝ่าย โดยเฉพาะอย่างยิ่ง รศ.ประภาษ อุคคกิมาพันธุ์ และ รศ.ดร.อัมพวัน จุลเสรีวงศ์ รวมไปถึงอาจารย์ประจำสาขาวิศวกรรมอัตโนมัติ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังทุก ๆ ท่าน ที่คอยให้คำแนะนำ ติดตามถามไถ่ความเป็นไปของปริญญานิพนธ์ และความเมตตาตลอดการดำเนินปริญญานิพนธ์จนสำเร็จลุล่วงไปได้ด้วยดี ขอขอบคุณในความเอื้อเฟื้อด้านอุปกรณ์และเครื่องมือต่าง ๆ จาก บริษัท มิตซูบิชิ อีเล็คทริค แฟคทอรี่ ออโตเมชั่น (ประเทศไทย) จำกัด (MITSUBISHI ELECTRIC FACTORY AUTOMATION (THAILAND) CO., LTD.) ที่ได้สนับสนุนอุปกรณ์ในการทำปริญญานิพนธ์นี้ ขอขอบคุณนาย นนท์ปวิธ ก้อนทอง ที่ได้ให้การสนันสนุนในด้านความรู้ด้านเทคนิคตลอดการทำปริญญานิพนธ์ นอกจากนั้นทางผู้จัดทำขอขอบคุณในความช่วยเหลือจากบุคคลภายนอกต่าง ๆ ไม่ว่าจะเป็น ญาติ ๆ เพื่อน ๆ พี่ ๆ น้อง ๆ และรวมไปถึงศิษย์เก่าของสถาบันฯ ที่ได้ให้การสนับสนุนตลอดมา

สุดท้ายนี้คุณค่าและและประโยชน์ใด ๆ ที่พึงมีจากการทำปริญญานิพนธ์ฉบับนี้ขอยกให้แก่ บิดา มารดา ครูบาอาจารย์ และผู้มีพระคุณทุกๆท่านสืบต่อไป

คณะผู้จัดทำ

สารบัญ

หน้า

บทคัดย่อภาษาไทย I

บทคัดย่อภาษาอังกฤษ II

กิตติกรรมประกาศ III

สารบัญ IV

สารบัญตาราง VIII

สารบัญรูป IX

รายการสัญลักษณ์ XV

บทที่ 1 บทนำ 1

1.1 ที่มาและความสำคัญ 1

1.2 วัตถุประสงค์ของปริญญานิพนธ์ 1

1.3 ขอบเขตของปริญญานิพนธ์ 2

1.4 ขั้นตอนการศึกษา 2

1.5 ประโยชน์ที่คาดว่าจะได้รับ 5

บทที่ 2 ทฤษฎีและหลักการที่เกี่ยวข้อง 6

2.1 กล่าวนำ 6

2.2 หุ่นยนต์อุตสาหกรรม 6

2.3 การควบคุมหุ่นยนต์อุตสาหกรรม 9

2.3.1 ตัวควบคุมหุ่นยนต์ 9

2.3.2 วิถีการเคลื่อนที่ 10

2.4 พีแอลซี 10

2.5 ส่วนติดต่อกับผู้ใช้งาน 13

2.6 เซอร์โวมอเตอร์ (Servo Motor) 14

2.6.1 ประเภทของ Servo Motor 15

2.6.2 โครงสร้างของ Servo Motor 16

2.6.3 หลักการทำงานของ Servo Motor 18

2.6.4 โครงสร้างของระบบควบคุม Servo Motor 18

2.6.5 องค์ประกอบในการทำงานของ Servo Motor 19

สารบัญ (ต่อ)

หน้า

2.7 ฐานข้อมูล 21
2.7.1 ระบบจัดการฐานข้อมูล 21
2.7.2 การออกแบบฐานข้อมูล 21
2.7.3 การออกแบบฐานข้อมูลเชิงสัมพันธ์ 22
2.7.4 การออกแบบฐานข้อมูลในระดับตรรกะ 22

บทที่ 3 ตัวควบคุมหุ่นยนต์ที่นำเสนอ 24
3.1 กล่าวนำ 24
3.2 สรีระหุ่นยนต์ที่ศึกษา 24
3.3 สมการทางคณิตศาสตร์ของหุ่นยนต์ที่ศึกษา 25
3.3.1 สมการ Inverse Kinematics 25
3.3.2 สมการ Forward Kinematics 28
3.4 ตัวควบคุมหุ่นยนต์ที่ศึกษาด้วยพีแอลซี 28
3.4.1 แนวคิดสำหรับการควบคุมหุ่นยนต์ที่ศึกษา 28
3.4.2 โครงสร้างทางฮาร์ดแวร์ของระบบควบคุม 30
3.4.3 การสื่อสารข้อมูลระหว่างอุปกรณ์ในระบบควบคุม 33
3.5 การเขียนโปรแกรมสำหรับตัวควบคุมหุ่นยนต์ด้วยพีแอลซี 35
3.5.1 ภาพรวมของโปรแกรมในการควบคุมหุ่นยนต์ที่ศึกษา 35
3.5.2 โปรแกรมควบคุมการทำงานของหุ่นยนต์ 37
3.5.3 การสร้าง Trajectory Path และโปรแกรมการถ่ายโอนข้อมูล Trajectory Path 37
3.5.4 พารามิเตอร์ที่สำคัญในการควบคุม 38
3.5.5 ภาพรวมของลำดับการใช้งานหุ่นยนต์ 42
3.6 การตั้งค่าการสื่อสารระหว่าง PLC กับโมดูลพิเศษ ด้วย GX Works2 44
3.7 การตั้งค่าโมดูลพิเศษในส่วน Simple Motion 44
3.7.1 เลือก Servo Amplifier ตามที่ใช้ในแต่ละแกน 44
3.7.2 การตั้งค่า Parameter ในหมวดต่าง ๆ 46
3.8 การเขียนโปรแกรมเริ่มการทำงานของ Simple Motion ด้วย GX Works2 50
3.9 ฟังก์ชันบล็อคสมการการคำนวณ Kinematics ของหุ่นยนต์ด้วย GX Works2 52

สารบัญ (ต่อ)

หน้า

3.9.1 การทดลองการหาความยาว Link ของหุ่นยนต์ 52
3.9.2 ฟังก์ชันบล็อคสำหรับสมการคำนวณ Kinematics ของหุ่นยนต์ 53
3.10 การสร้าง Programing Pendant จาก HMI ด้วย GT Designer3 55
3.11 ฟังก์ชันบล็อคสำหรับการทำงานแบบ Jog Operation ด้วย GX Works2 [12] 59
3.11.1 การเขียนฟังก์ชันบล็อคสำหรับควบคุมการทำงานแบบ Jog Operation 61
3.11.2 การทดลองการหาอัตราทดที่ Joint ของหุ่นยนต์ 63
3.12 ฟังก์ชันบล็อคการทำงานแบบ Automatic Position Control ด้วย GX Works2 65
3.12.1 แผนผังฟังก์ชันบล็อคควบคุมการทำงานแบบ Automatic Position Control 66
3.12.2 การตั้งค่าการทำงานแบบ Automatic Position Control 67
3.12.3 การเขียนฟังก์ชันบล็อคควบคุมการทำงานแบบ Automatic Position Control 72
3.12.4 การทดลองการหา Software Stroke Limit 73
3.13 ฟังก์ชันบล็อคการทำงานโหมด M-code Event Selection ด้วย GX Works2 73
3.14 ฟังก์ชันบล็อคการทำงานแบบ Coordinate Movement Control ด้วย GX Works2 74
3.15 ฟังก์ชันบล็อคควบคุมการทำงานแบบ Trajectory-path Movement Control 75
3.16 การสร้าง Trajectory Path หุ่นยนต์ด้วย RobotStudio 76
3.16.1 การสร้าง Trajectory Path 76
3.16.2 การกำหนดค่า Resolution 81
3.16.3 การคำนวณหาค่า Resolution 82
3.17 การสร้างและจัดเก็บข้อมูลของระบบฐานข้อมูล 83
3.18 การสร้างระบบจัดการข้อมูล Trajectory Path 85
3.18.1 ขั้นตอนการสร้างระบบจัดการข้อมูล Trajectory Path 85
3.18.2 การเขียนโปรแกรมการ Query ข้อมูล Trajectory Path จากระบบฐานข้อมูล 87
3.18.3 การเขียนโปรแกรมสำหรับส่งข้อมูล Trajectory Path ให้ PLC 88

บทที่ 4 การทดสอบตัวควบคุมหุ่นยนต์ที่นำเสนอ 90
4.1 กล่าวนำ 90
4.2 การดำเนินงาน 90
4.2.1 การติดตั้งอุปกรณ์ทาง Hardware 90

สารบัญ (ต่อ)

หน้า

4.2.2 การทำงานของ Input 95
4.2.3 การทำงานของ Output 99
4.2.4 การทำงานของ Simple Motion 101
4.2.5 การทำงานแบบ Jog Operation 102
4.2.6 การทำงานแบบ Automatic Position Control 106
4.2.7 การทำงานแบบ M-code Event Selection 107
4.2.8 การทำงานแบบ Coordinate Movement Control 108
4.2.9 การใช้โปรแกรม WW InTouch ในการ Query ข้อมูลจากระบบฐานข้อมูล 109
4.2.10 การใช้โปรแกรม WW InTouch ในการส่งข้อมูลให้ PLC 110
4.2.11 การทำงานแบบTrajectory-path Movement Control 111

บทที่ 5 สรุปผลและข้อเสนอแนะ 114
5.1 สรุปผล 114
5.2 ปัญหาที่พบและแนวทางแก้ไข 114
5.3 ข้อเสนอแนะ 115

เอกสารอ้างอิง 117

สารบัญตาราง

ตารางที่ หน้า

1.1 แผนการดำเนินงาน....3

3.1 Input Parameter....38

3.2 Output Parameter....39

3.3 Motion Parameter....39

3.4 Positioning Data Parameter....41

3.5 ผลการทดลองหาอัตราทดของแกนที่ 1....63

3.6 ผลการทดลองหาอัตราทดของแกนที่ 2....63

3.7 ผลการทดลองหาอัตราทดของแกนที่ 3....64

3.8 ผลการทดลองหาอัตราทดของแกนที่ 4....64

3.9 รายละเอียด Positioning Data Parameter....67

3.10 ผลการทดลองหา Software Stroke Limit....73

3.11 โค้ดคำสั่งการ Query....89

4.1 ตรวจสอบความถูกต้องของการเชื่อมต่อวงจร....94

4.2 ตรวจสอบความถูกต้องในการทำงานของ Limit Switch....99

4.3 ตรวจสอบความถูกต้องในการทำงานของ Output (Y45) 100

4.4 ตรวจสอบความพร้อมใช้งานของ Servo Amplifier และ 102

4.5 ผลการทดสอบการเคลื่อนที่แบบ Jog Operation....105

4.6 ผลการทำงานตาม Event ที่กำหนด....108

สารบัญรูป

รูปที่ หน้า

2.1 การเปรียบเทียบ SCARA Robot กับการเคลื่อนไหวของร่างกาย 6
2.2 ส่วนประกอบของตัวหุ่นยนต์ 8
2.3 ตัวอย่าง Programming Pendant 8
2.4 ตัวอย่าง Robot Controller 9
2.5 ตัวอย่าง Servo Amplifier 9
2.6 Mitsubishi Programmable Logic Control รุ่น Q03UDECPU 11
2.7 โครงสร้างของ PLC 11
2.8 Human Machine Interface (HMI) 13
2.9 ตัวอย่าง Servo Motor 15
2.10 ประเภทของ Servo Motor 15
2.11 โครงสร้างของ AC Servo Motor 16
2.12 วัสดุที่นำมาสร้างแม่เหล็กถาวร 17
2.13 โครงสร้างและการทำงานของ AC Servo Motor 18
2.14 โครงสร้างของระบบควบคุม Servo Motor 19
2.15 ตัวอย่างตัวควบคุม Servo Motor 19
2.16 ตัวอย่างเซอร์โวแอมปลิฟายเออร์ (Servo Amplifier) 20
2.17 ตัวอย่าง Servo Motor 20
3.1 SCARA Robot รุ่น MELFA RH-P2 ค.ศ. 1987 24
3.2 SCARA Robot Schematic 25
3.3 Inverse Kinematics Quadrant1 25
3.4 Inverse Kinematics Quadrant2 27
3.5 Forward Kinematics 28
3.6 การทำงานรูปแบบต่าง ๆ ของหุ่นยนต์ 29
3.7 Overall System Architecture 31
3.8 Block Diagram of Data Transfer 33
3.9 การสร้าง Trajectory Path 34

สารบัญรูป (ต่อ)

รูปที่ หน้า

3.10 องค์ประกอบโปรแกรมพื้นฐานสำหรับควบคุม Servo Motor 36
3.11 การถ่ายโอนข้อมูล Trajectory Path 37
3.12 ลำดับการทำงานของโปรแกรมหลัก 43
3.13 การสร้าง I/O Assignment 44
3.14 การเข้าโปรแกรม Simple Motion Module Setting 45
3.15 การตั้งค่า Simple Motion Module กับ Servo Amplifier 45
3.16 การกำหนดรุ่น Servo Amplifier 46
3.17 Servo Amplifier Parameter และ Servo Parameter 46
3.18 หน้าต่างตั้งค่า Servo Amplifier Parameter 47
3.19 หน้าต่าง Compute Basic Parameter1 47
3.20 การตั้งค่า Servo Parameter 49
3.21 Servo Motor Rotation 50
3.22 แผนผังการเริ่มการทำงานของ Servo Amplifier 51
3.23 ผลการหาความยาว Link1 52
3.24 การหาความยาว Link2 53
3.25 ผลการหาความยาว Link2 53
3.26 ตัวอย่าง Stucture Text สมการคำนวณ Kinematics 54
3.27 Inverse Kinematics Function Block 54
3.28 Forward Kinematics Function Block 54
3.29 หน้าต่าง Overview 55
3.30 หน้าต่างโหมด Adjustment 56
3.31 หน้าต่างการตั้งค่า Stock Limit 56
3.32 หน้าต่างโหมด Position 57
3.33 หน้าต่างโหมดการเคลื่อนที่แบบระบุพิกัด 57
3.34 หน้าต่างใช้งาน Position Run 58
3.35 หน้าต่าง Error Message 58

สารบัญรูป (ต่อ)

รูปที่ หน้า

3.36 หน้าต่าง Alarm Log 59
3.37 การทำงานแบบ Jog Operation 59
3.38 กราฟการทำงาน Jog Operation 60
3.39 แผนผังการเขียนฟังก์ชันบล็อคสำหรับควบคุมการทำงานแบบ Jog Operation 61
3.40 Jog Operation Function Block for Axis1 62
3.41 Jog Operation Function Block for Axis2 62
3.42 Jog Operation Function Block for Axis3 62
3.43 Jog Operation Function Block for Axis4 63
3.44 แผนผังการเขียนฟังก์ชันบล็อคสำหรับควบคุมการทำงานแบบ Automatic Position Control... 66
3.45 หน้าต่างตั้งค่าข้อมูลสำหรับ Automatic Position Control 67
3.46 การทำงานแบบ Position Complete 68
3.47 การทำงานแบบ Position Complete 69
3.48 การทำงานแบบ Location หรือ Continuous Path Control 69
3.49 การควบคุมแบบ ABS Linear 70
3.50 การควบคุมแบบ INC Linear 71
3.51 การควบคุมแบบ LOOP และ LEND 71
3.52 การตั้งค่า Axis to be interpolated 72
3.53 Automatic Position Operation Function Block for Axis1 72
3.54 Automatic Position Operation Function Block for Axis2 72
3.55 Automatic Position Operation Function Block for Axis3 72
3.56 Automatic Position Operation Function Block for Axis4 73
3.57 การเขียนโปรแกรมควบคุมแบบ M-code Event Selection 74
3.58 กระบวนการของฟังก์ชันบล็อกควบคุมการทำงานแบบ Coordinate Movement Control 75
3.59 การสร้างฟังก์ชันบล็อคควบคุมการทำงานแบบ Trajectory-path Movement Control 76
3.60 ตัวอย่างหน้าต่างโปรแกรม RobotStudio 77

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

3.61 ตัวอย่างหุ่นยนต์ที่ถูกจำลองขึ้นมาด้วยโปรแกรม RobotStudio 77
3.62 การนำรูปทรงสามมิติเข้าโปรแกรม RobotStudio 78
3.63 ตัวอย่างรูปทรงสามมิติที่สร้างด้วยโปรแกรม SolidWork 78
3.64 การสร้างเส้นทางการเดินให้แก่หุ่นยนต์ 79
3.65 รูปแบบของภาษา RAPID 79
3.66 ข้อมูลภาษา RAPID 80
3.67 ข้อมูลพิกัดการเคลื่อนที่จากโปรแกรม Microsoft Excel 80
3.68 หน้าต่างการสร้าง Trajectory Path 81
3.69 การกำหนดค่า Resolution 81
3.70 ความสัมพันธ์ระหว่างค่าความผิดพลาดกับค่า Resolution แบบเส้นตรง 82
3.71 Resolution ของรูปวงกลม 83
3.72 ภาพขยายของรูปที่ 3.71 83
3.73 ขั้นตอนการสร้างระบบฐานข้อมูล 84
3.74 ตัวอย่างตารางที่ใช้เก็บข้อมูล Trajectory Path 84
3.75 ขั้นตอนการสร้างระบบจัดการข้อมูล Trajectory Path 85
3.76 หน้าต่าง Overview ของระบบจัดการข้อมูล Trajectory Path 85
3.77 หน้าต่าง Control Setting ของระบบจัดการข้อมูล Trajectory Path 86
3.78 หน้าต่าง Path Select ของระบบจัดการข้อมูล Trajectory Path 86
3.79 การทำงานของโปรแกรมการ Query ข้อมูล Trajectory Path จากระบบฐานข้อมูล 87
3.80 การเชื่อมต่อระหว่างระบบฐานข้อมูลกับระบบจัดการข้อมูล Trajectory Path 88
3.81 การทำงานของโปรแกรมสำหรับส่งข้อมูล Trajectory Path ให้ PLC 88
4.1 การติดตั้งระบบควบคุมหุ่นยนต์ 90
4.2 วงจรไฟฟ้าภาคกำลังส่วนที่ 1 91
4.3 วงจรไฟฟ้าภาคกำลังส่วนที่ 2 91
4.4 วงจรการต่อสายของ PLC 92
4.5 วงจร Limit Switch ที่ต่อเข้ากับ Input ของ PLC 92

สารบัญรูป (ต่อ)

รูปที่ **หน้า**

4.6 วงจร Motor Brake ที่ต่อเข้ากับ Output ของ PLC 93

4.7 วงจร Servo Amplifier และ Servo Motor ของแกนที่ 1 และ 2 93

4.8 วงจร Servo Amplifier และ Servo Motor ของแกนที่ 3 และ 4 94

4.9 พื้นที่การทำงานของหุ่นยนต์ (Top View) 95

4.10 พื้นที่การทำงานของหุ่นยนต์ (Front View) 95

4.11 ทดสอบ Input กรณีที่ Limit Switch ยังไม่ถูกตรวจจับได้ตั้งแต่แกนที่ 1 ถึง 3 ตามลำดับ 96

4.12 ทดสอบ Input จาก Lower Limit Switch ของแกนที่ 1 96

4.13 ทดสอบ Input จาก Upper Limit Switch ของแกนที่ 1 97

4.14 ทดสอบ Input จาก Lower Limit Switch ของแกนที่ 2 97

4.15 ทดสอบ Input จาก Upper Limit Switch ของแกนที่ 2 98

4.16 ทดสอบ Input จาก Lower Limit Switch ของแกนที่ 3 98

4.17 ทดสอบ Input จาก Upper Limit Switch ของแกนที่ 3 99

4.18 ผลการทดสอบ Output เมื่อ Y45 เป็น OFF 100

4.19 ผลการทดสอบ Output เมื่อ Y45 เป็น ON 100

4.20 ผลการทดสอบความพร้อมใช้งานของ Servo เมื่อคำสั่งเป็น OFF 101

4.21 ผลการทดสอบความพร้อมใช้งานของ Servo เมื่อคำสั่งเป็น ON 101

4.22 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 1 แบบ Reverse 102

4.23 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 1 แบบ Forward 103

4.24 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 2 แบบ Reverse 103

4.25 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 2 แบบ Forward 104

4.26 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 3 แบบ Forward 104

4.27 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 3 แบบ Reverse 105

4.28 แสดงผลการทดลองการเคลื่อนที่แบบ Position Control ของแกนที่ 1 และแกนที่ 2 106

4.29 แสดงผลการทดลองการเคลื่อนที่แบบ Position Control ของแกนที่ 3 107

4.30 ตัวอย่างการตั้งค่า Positioning Data ของการทำงาน Event 107

4.31 ผลการเคลื่อนที่แบบระบุพิกัด X และ Y 108

สารบัญรูป (ต่อ)

รูปที่ หน้า

4.32 ผลการเชื่อมต่อระหว่าง WW InTouch กับระบบฐานข้อมูล 109
4.33 ผลการ Query ข้อมูลจากระบบฐานข้อมูล 110
4.34 ข้อมูลที่ถูกเก็บไว้ใน Buffer Memory ของ PLC ก่อนส่งให้ Simple Motion 111
4.35 ข้อมูลที่ถูกเก็บไว้ใน Buffer Memory ของ PLC หลังส่งให้ Simple Motion 111
4.36 การเคลื่อนที่ตาม Trajectory Path เป็นรูปสามเหลี่ยม 112
4.37 การเคลื่อนที่ตาม Trajectory Path เป็นรูปวงกลม 112
4.38 การเคลื่อนที่ตาม Trajectory Path เป็นรูปสี่เหลี่ยม 113

รายการสัญลักษณ์

a1	หมายถึง	ระยะความยาว Link1
a2	หมายถึง	ระยะความยาว Link2
Xc	หมายถึง	พิกัดจุด X ของ End Effect
Yc	หมายถึง	พิกัดจุด Y ของ End Effect
Rc	หมายถึง	ระยะความยาวอ้างอิงจากฐานของหุ่นยนต์ ถึง End Effect
θ_1	หมายถึง	มุมของ Link1
θ_2	หมายถึง	มุมของ Link2

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญ

การพัฒนาอย่างต่อเนื่องในทางอุตสาหกรรมทำให้เกิดความก้าวหน้าทางเทคโนโลยี ส่งผลให้อุปกรณ์ที่ใช้ในปัจจุบันมีการพัฒนาขึ้นอย่างก้าวกระโดด และมีความแม่นยำสูง อีกทั้งในปัจจุบันมีการนำหุ่นยนต์เข้ามาใช้ในอุตสาหกรรมมากขึ้น และการใช้หุ่นยนต์อุตสาหกรรมในอนาคตมีแนวโน้มเพิ่มมากขึ้น ซึ่งการควบคุมหุ่นยนต์แต่ละรุ่นจะมีตัวควบคุมหุ่นยนต์ (Robot Controller) ที่มีลักษณะอันเป็นเอกลักษณ์เฉพาะตามแต่ละผู้ผลิต ในกรณีที่ตัวควบคุมหุ่นยนต์ เกิดชำรุดเสียหาย การซ่อมแซมต้องอาศัยผู้เชี่ยวชาญของแต่ละผู้ผลิต หรือในกรณีที่ตัวควบคุมหุ่นยนต์นั้นถูกใช้มานาน แต่ในปัจจุบันผู้ผลิตได้ยกเลิกการผลิตแล้ว ทำให้ไม่สามารถใช้หุ่นยนต์ที่มีอยู่ได้ การซื้อหุ่นยนต์ และตัวควบคุมใหม่อาจเป็นค่าใช้จ่ายที่สูงสำหรับผู้ใช้งานหรือผู้ประกอบการขนาดกลาง และขนาดเล็ก เพื่อลดข้อจำกัดดังกล่าว จึงมีแนวคิดที่จะสร้างทางเลือกในการใช้งานหุ่นยนต์อุตสาหกรรม โดยการนำเอาหุ่นยนต์ที่มีใช้อยู่แล้วหรือหุ่นยนต์เก่าที่เลิกใช้แล้ว นำกลับมาใช้ใหม่ (Reuse) ร่วมกับตัวควบคุมหุ่นยนต์ที่สร้างขึ้นโดยใช้ตัวควบคุมอุตสาหกรรมที่เรียกว่า พีแอลซี (Programmable Logic Controller: PLC) เนื่องจากพีแอลซีมีภาษาที่ใช้ในการเขียนโปรแกรมเป็นภาษาที่มีมาตรฐาน ดังเช่น มาตรฐาน IEC61131-3 ดังนั้นการใช้งานพีแอลซีจึงมีความยืดหยุ่น ในการสับเปลี่ยนแทนที่ (Repplacement) ในกรณีที่อุปกรณ์ทางฮาร์ดแวร์ของพีแอลซีเกิดการชำรุดเสียหาย นอกจากนี้ในปัจจุบันมีพีแอลซีจากผู้ผลิตที่หลากหลาย ทำให้มีความหลากหลายของผลิตภัณฑ์ให้ผู้ใช้งานเลือกใช้

1.2 วัตถุประสงค์ของปริญญานิพนธ์

ปริญญานิพนธ์นี้ได้นำเสนอการนำ PLC มาประยุกต์ใช้เป็น Robot Controller สำหรับควบคุมหุ่นยนต์อุตสาหกรรมเก่าประเภท SCARA ที่ไม่สามารถใช้งานได้ให้สามารถนำกลับมาใช้ใหม่ได้ โดยการสร้างฟังก์ชันบล็อค สำหรับควบคุมการทำงานของหุ่นยนต์อุตสาหกรรม โดยใช้ภาษาแลดเดอร์ (Ladder) ร่วมกับภาษาสตรัคเจอร์เทค (Structure Text) ในการเขียนโปรแกรม ซึ่งภาษา Structure Text จะใช้สำหรับเขียนโปรแกรมกำหนดสมการคณิตศาสตร์ที่เป็นสมการที่เกี่ยวข้องกับการเคลื่อนที่ Kinematics และในส่วนของภาษา Ladder นั้น จะใช้สำหรับการควบคุมลำดับการทำงานของหุ่นยนต์อุตสาหกรรมที่ศึกษา

1.3 ขอบเขตของปริญญานิพนธ์

1. ศึกษาหุ่นยนต์ประเภท SCARA ที่มีจำนวน Joint และ Link เท่ากับ 4 และ 2 ตามลำดับ

2. สามารถควบคุมการเคลื่อนที่ของหุ่นยนต์ด้วยความเร็วรอบของมอเตอร์สูงสุด 5,000 รอบต่อนาที

3. สมการการวางแผนการเคลื่อนที่ (Motion Planning) ได้มีการสร้างวิถีการเคลื่อนที่ (Trajectory Path) ของหุ่นยนต์ด้วยโปรแกรม RobotStudio

4. นำข้อมูล Trajectory Path ที่สร้างจากโปรแกรม RobotStudio มาจัดเก็บที่เครื่องคอมพิวเตอร์ Server ที่มีการติดตั้งโปรแกรม Microsoft SQL Server

5. ใช้โปรแกรม WW InTouch เป็นส่วนติดต่อกับผู้ใช้งาน และมีการแสดงผลการควบคุมการเคลื่อนที่ของหุ่นยนต์ นอกจากนี้ยังใช้คำสั่ง Query ของโปรแกรม WW InTouch ในการเลือกกำหนดให้สัมพันธ์กับรูปแบบการเคลื่อนที่ของหุ่นยนต์ที่ต้องการ เช่น วงกลม สามเหลี่ยม และสี่เหลี่ยม พร้อมทั้งมีการส่งต่อข้อมูล Trajectory Path ที่เกี่ยวข้องให้กับ PLC

6. นำ PLC ของ Mitsubishi รุ่น Q03UDECPU มาเขียน Function Blocks ควบคุม Servo motor ให้มีรูปแบบทำงาน 5 รูปแบบ ได้แก่ Jog Operation, Automatic Position Control, M-code Event Selection, Coordinate Movement Control, และ Trajectory-path Movement Control

7. จำลองการเคลื่อนที่ของ SCARA Robot ในลักษณะของแขนขวา ซึ่งอยู่ในช่วง Quadrant ที่ 1 และ 2

1.4 ขั้นตอนการศึกษา

รายละเอียดขั้นตอนการดำเนินงานแสดงดังตารางที่ 1.1

ตารางที่ 1.1 แผนการดำเนินงาน

งาน	รายละเอียด	ช่วงเวลาดำเนินงาน	ระยะเวลาในการดำเนินงาน (สัปดาห์)															
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	ศึกษาหุ่นยนต์ที่ใช้และอัลกอริทึมในการควบคุมหุ่นยนต์	1 สัปดาห์																
2	ศึกษาการใช้งาน Servo Amplifier เพื่อใช้งานร่วมกับ PLC ผ่านโปรแกรม GX Work2	2 สัปดาห์																
3	เขียนโปรแกรม PLC สำหรับควบคุมการทำงานของหุ่นยนต์	6 สัปดาห์																
4	นำโปรแกรม PLC ที่เขียนมาทำการทดสอบเพื่อหาค่าพารามิเตอร์ที่จำเป็นสำหรับการควบคุม	1 สัปดาห์																
5	ศึกษาสมการ Kinematic สำหรับหุ่นยนต์ประเภท SCARA	1 สัปดาห์																
6	ทดลองหาค่าความยาว Link1 และ Link2 ของหุ่นยนต์ และนำสมการ Kinematic ที่ศึกษามาเขียนโปรแกรมคำนวณ Kinematic	2 สัปดาห์																

ตารางที่ 1.1 (ต่อ)

งาน	รายละเอียด	ช่วงเวลาดำเนินงาน	ระยะเวลาในการดำเนินงาน (สัปดาห์)															
			1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
7	ศึกษาการใช้งานโปรแกรม Robotstudio และสร้าง Trajectory Path ให้หุ่นยนต์	1 สัปดาห์																
8	ศึกษาการใช้งานโปรแกรม Microsoft SQL Server, สร้างระบบฐานข้อมูล	1 สัปดาห์																
9	ออกแบบและสร้างระบบจัดการข้อมูล Trajectory Path, ศึกษาวิธีการสร้างการเชื่อมต่อระหว่างระบบจัดการข้อมูล Trajectory Path กับระบบฐานข้อมูล และการเชื่อมต่อระหว่างระบบจัดการข้อมูล Trajectory Path กับ PLC	2 สัปดาห์																
10	เขียนโปรแกรมการ Query ข้อมูลจากระบบฐานข้อมูลและโปรแกรมส่งข้อมูลให้ PLC	2 สัปดาห์																
11	ทดสอบการทำงานของหุ่นยนต์และแก้ไขส่วนผิดพลาด	3 สัปดาห์																

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1.ทำให้ PLC สามารถทำงานเป็น Robot Controller ได้ เพื่อเป็นตัวเลือกในการควบคุมหุ่นยนต์

2.สามารถนำหุ่นยนต์เก่าให้กลับมาใช้งานได้ใหม่ โดยมีตัวควบคุมใหม่

3.ทราบขีดความสามารถในการทำงานของ PLC สำหรับการควบคุมหุ่นยนต์

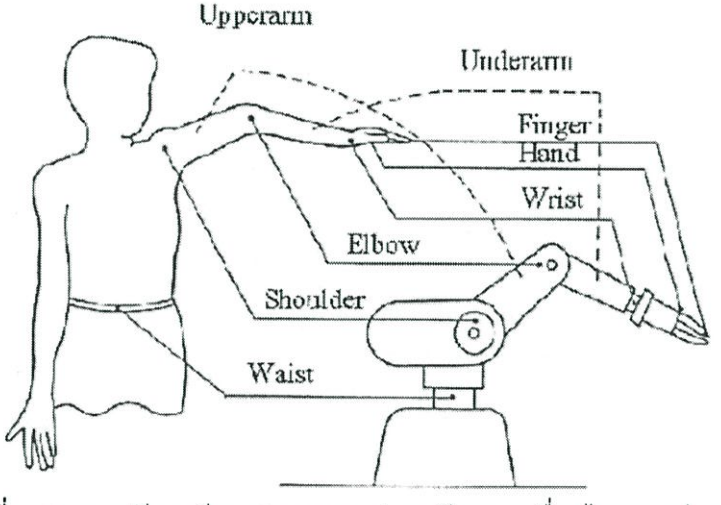
บทที่ 2
ทฤษฎีและหลักการที่เกี่ยวข้อง

2.1 กล่าวนำ

ในบทนี้จะเป็นการกล่าวถึงทฤษฎี และหลักการที่เกี่ยวข้องเช่น หุ่นยนต์อุตสาหกรรม สมการ Inverse Kinematics และสมการ Forward Kinematics รวมถึงอุปกรณ์ที่ใช้ในการควบคุมหุ่นยนต์ที่ใช้เป็นกรณีศึกษาทั้งในด้านฮาร์ดแวร์ และด้านซอฟต์แวร์

2.2 หุ่นยนต์อุตสาหกรรม [1] – [6]

หุ่นยนต์อุตสาหกรรม (Industrial Robot) ได้ถูกออกแบบมาจากการเคลื่อนไหวพื้นฐานในช่วงแขนของมนุษย์ เพื่อให้สามารถทำงานแทนมนุษย์ ดังรูปที่ 2.1 ประสิทธิภาพของหุ่นยนต์นั้นเกิดขึ้นจากความแม่นยำในการเคลื่อนที่ของหุ่นยนต์ ส่วนใหญ่ใช้ในงานที่ต้องทำซ้ำ ๆ ต่อเนื่องตลอดเวลา งานอันตราย หรืองานที่ยากเกินมนุษย์จะรับไหว นอกจากนั้นยังทำให้งานที่ได้ออกมามีคุณภาพเที่ยงตรง



รูปที่ 2.1 การเปรียบเทียบ SCARA Robot กับการเคลื่อนไหวของร่างกาย

โครงสร้างของหุ่นยนต์ (Robot Structure) สำหรับระบบอุตสาหกรรมมีหลากหลายรูปแบบ เพื่อให้สะดวกในการใช้งาน และเพื่อความเหมาะสมของกระบวนการต่าง ๆ จึงต้องมีลักษณะที่ต่างกันออกไปตามแต่กระบวนการผลิต สภาพแวดล้อม เนื้อที่ และปัจจัยต่าง ๆ ที่ส่งผลให้วิธีการในการควบคุมนั้นแตกต่างกันออกไป แต่จะมีส่วนประกอบหลักทั้งหมด 3 ส่วนได้แก่

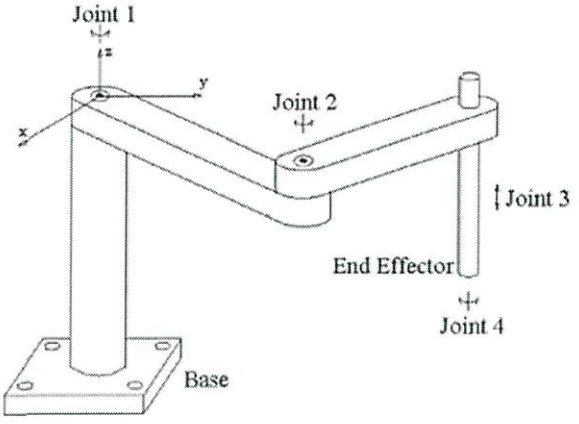
1. ตัวหุ่นยนต์ (Robot Body) เป็นส่วนที่สำคัญในการพิจารณาเลือกหุ่นยนต์มาใช้ในกระบวนการผลิตที่ต้องคำนึงถึงลักษณะของงาน พื้นที่ และสิ่งแวดล้อม เนื่องจากตัวหุ่นยนต์นั้นเป็นส่วนที่ต้องนำมาใช้ในกระบวนการทำงาน ลักษณะงานที่แตกต่างกันก็จะเป็นตัวบ่งบอกในเรื่องของขนาด และโครงสร้างของตัวหุ่นยนต์ ซึ่งประกอบไปด้วย

- ฐาน (Base) เป็นส่วนที่ใช้ยึดตัวหุ่นยนต์กับบริเวณที่ตั้ง ส่วนประกอบส่วนนี้จำเป็นที่จะต้องมีความแข็งแรง และทนทาน เพื่อรองรับน้ำหนักของหุ่นยนต์ และแรงเหวี่ยงที่เกิดจากการเคลื่อนที่ขอหุ่นยนต์ นอกจากนั้นยังใช้เป็นจุดอ้างอิง เพื่อสร้างเฟรมในการเคลื่อนที่ให้แก่หุ่นยนต์

- ท่อนแขน (Link) เป็นส่วนที่ใช้ในการเชื่อมต่อระหว่างข้อต่อ ส่วนใหญ่มักจะมีลักษณะตรงยาว และต้องมีความแข็งแรงทนทาน สามารถรองรับน้ำหนักได้สูง เพื่อรองรับน้ำหนักของตัวหุ่นยนต์ในส่วนต่อ ๆ ไป รวมไปถึงน้ำหนักของวัตถุที่หุ่นยนต์หยิบ หรือจับเอาไว้

- ข้อต่อ (Joint) คือ จุดหมุน หรือเลื่อนของหุ่นยนต์มีอยู่หลายประเภท แต่ที่นิยมใช้กันมากที่สุดได้แก่ประเภทข้อต่อแบบหมุน (Revolute Joint) และข้อต่อแบบเลื่อน (Prismatic Joint) สำหรับข้อต่อแบบหมุน ท่อนแขนทั้งสองท่อนจะถูกยึดติดกันที่จุดหมุนซึ่งอยู่บนท่อนแขน โดยแต่ละท่อนสามารถหมุนได้รอบจุดหมุนนี้ นอกจากนี้ผู้ใช้งานสามารถบอกตำแหน่งของสองท่อนแขนที่สัมพันธ์กันด้วยมุมที่ท่อนแขนหมุนไป ส่วนข้อต่อแบบเลื่อนนั้น ท่อนแขนทั้งสองท่อนจะถูกยึดติดกันในลักษณะเดียวกันกับเสาอากาศวิทยุรถยนต์ที่ยืดหดได้ โดยท่อนแขนแต่ละท่อนสามารถเลื่อนเข้าออกได้ในหนึ่งทิศทาง ผู้ใช้งานสามารถระบุตำแหน่งที่สัมพันธ์กันของสองท่อนแขนได้จากระยะเลื่อนเข้าออกดังกล่าว จะเห็นได้ว่าข้อต่อแบบหมุน และข้อต่อแบบเลื่อนมีพื้นที่ในการเคลื่อนที่ที่แตกต่างกัน จึงทำให้เกิดเป็นตัวแปร หรือพารามิเตอร์ที่ต้องใช้ในการคำนวณหา Trajectory Path ให้แก่หุ่นยนต์ ซึ่งในแต่ละข้อต่อจะมีมอเตอร์ที่ใช้ในการควบคุมการเคลื่อนที่ของข้อต่ออยู่ภายใน โดยมอเตอร์แต่ละตัวนี้จะรับค่าคำสั่งในการเคลื่อนที่มาจากตัวควบคุมที่ส่งเข้าไปยัง Servo Amplifier และทำการขยายสัญญาณไฟฟ้าจนมาเป็นการเคลื่อนที่ทางกลของมอเตอร์

การนับข้อต่อของหุ่นยนต์จะเริ่มนับตั้งแต่ส่วนที่ติดกับฐานของหุ่นยนต์จนมาถึงปลายแขนของหุ่นยนต์ ดังแสดงในรูปที่ 2.2 ซึ่งเป็นตัวอย่างตัวหุ่นยนต์ที่มีจำนวน Joint และ Link เท่ากับ 4 และ 2 ตามลำดับ



รูปที่ 2.2 ส่วนประกอบของตัวหุ่นยนต์

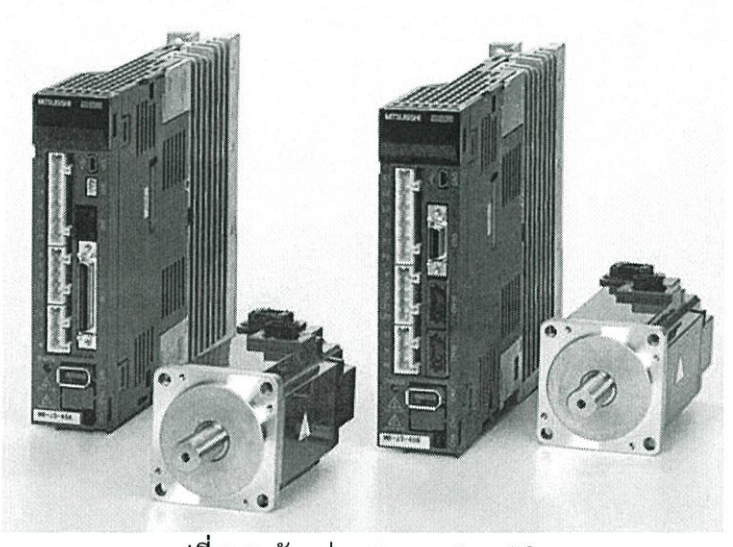
2. แป้นสอนตำแหน่ง (Programming Pendant) ใช้ติดต่อสื่อสารระหว่างมนุษย์กับหุ่นยนต์ ซึ่งได้มีการพัฒนาอย่างต่อเนื่อง นับตั้งแต่ระยะแรกที่หุ่นยนต์เริ่มมีบทบาทในโรงงานอุตสาหกรรม เช่น โรงงานผลิตรถยนต์ โดยงานของหุ่นยนต์จะเน้นการหยิบจับ วาง หรือประกอบชิ้นงานเป็นหลัก ดังนั้น เพื่อความสะดวกในการควบคุมหุ่นยนต์ จึงมีการคิดค้นอุปกรณ์สอน และบันทึกค่าตำแหน่งของปลายแขนหุ่นยนต์ซึ่งเรียกว่าแป้นสอนตำแหน่ง ดังตัวอย่างแสดงในรูปที่ 2.3 อุปกรณ์ในส่วนนี้จะเป็นส่วนที่อินเตอร์เฟสกับผู้ใช้งานติดอยู่บริเวณใกล้เคียงกับหุ่นยนต์ โดยผู้ใช้งานสามารถป้อนเป็นค่าตัวเลขของตำแหน่ง หรือควบคุมการเคลื่อนที่ของแต่ละข้อต่อผ่านปุ่มบนแป้นควบคุม จนปลายแขนถึงตำแหน่งที่ต้องการแล้วสั่งบันทึกค่ามุมของแต่ละข้อต่อนั้นไว้ หลังจากนั้นผู้ใช้งานสามารถสั่งการให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่งที่เคยสอน และบันทึกไว้ได้ การควบคุมประเภทนี้เหมาะกับการใช้งานประเภทหยิบแล้ววาง (Pick-and-Place Task)

รูปที่ 2.3 ตัวอย่าง Programming Pendant

3. Controller & Amplifier Set ส่วนนี้จะเป็นส่วนที่ใช้ในการควบคุมการเคลื่อนที่ของมอเตอร์ในแต่ละข้อต่อ สามารถควบคุมได้อย่างรวดเร็ว มีความละเอียดแม่นยำสูง แต่จะมีความละเอียดอ่อนต่อสัญญาณต่าง ๆ ที่มารบกวนทำให้ข้อมูลตำแหน่ง หรือค่าพารามิเตอร์ต่าง ๆ ที่ส่งไป

ควบคุมหุ่นยนต์เกิดการผิดเพี้ยนจากสัญญาณรบกวน ทำให้ประสิทธิภาพในการทำงานลดลง ดังตัวอย่างแสดงในรูปที่ 2.4-2.5

รูปที่ 2.4 ตัวอย่าง Robot Controller



รูปที่ 2.5 ตัวอย่าง Servo Amplifier

2.3 การควบคุมหุ่นยนต์อุตสาหกรรม

ในส่วนนี้จะกล่าวถึงอุปกรณ์ที่ใช้ควบคุมการเคลื่อนไหวของหุ่นยนต์ และการสร้างวิถีการเคลื่อนที่ (Trajectory Path) ให้กับหุ่นยนต์

2.3.1 ตัวควบคุมหุ่นยนต์

ตัวควบคุมหุ่นยนต์ (Robot Controller) เป็นตัวส่งสัญญาณควบคุม (Signal Command) ไปยังตัว Driver ซึ่งตัว Driver จะทำหน้าที่ขยายสัญญาณ และส่งผ่านสัญญาณส่งไปที่ Motor ทำให้ Motor หมุนด้วยความเร็วไปยังตำแหน่งที่ต้องการตามคำสั่งที่มาจาก Controller และสามารถจำแนกการทำงานออกได้ทั้งหมด 2 ส่วน ได้แก่ Motion Control และ Kinematics

1. Motion Control คือ การควบคุมการเคลื่อนที่ของมอเตอร์ โดยการส่งสัญญาณในการเคลื่อนที่ให้แก่ Servo Amplifier เพื่อขยายสัญญาณ และนำไปควบคุมมอเตอร์ให้เคลื่อนที่ไปยังตำแหน่งที่กำหนด

2. จลนศาสตร์ (Kinematics) เป็นการศึกษาถึงลักษณะการเคลื่อนที่ของวัตถุ โดยมุ่งความสนใจไปที่ การหาการขจัด ความเร็ว และความเร่งของอนุภาค โดยไม่มีการพิจารณาแรงกระทำบนวัตถุ โดยจะพิจารณาลักษณะของหุ่นยนต์ ซึ่งใช้ในการคำนวณว่าปลายแขนของหุ่นยนต์อยู่ที่ตำแหน่งใด การคำนวณดังกล่าวจะต้องกำหนดให้ท่อนแขนแต่ละท่อนมีเฟรมเป็นของตัวเอง (Frame) โดยเฟรมในที่นี้จะประกอบไปด้วยจุดกำเนิด (Origin) และเวกเตอร์ (Vector) โดยที่แขนจะไม่มีการขยับออกจากเฟรม

- จลนศาสตร์ทางตรง (Forward Kinematics) จลนศาสตร์ทางตรงของหุ่นยนต์จะสนใจเกี่ยวกับการเรียงตัว ตำแหน่ง ทิศทาง และการเปลี่ยนแปลงของสิ่งต่าง ๆ เหล่านี้ของโครงสร้างโดยไม่คำนึงถึงแรงต้นเหตุ จลนศาสตร์ทางตรงของหุ่นยนต์ว่าด้วยการคำนวณตำแหน่งพิกัดที่ถือว่าหยุดนิ่งของจุดบนหุ่นยนต์ที่กำหนดให้ใช้ในการแก้ไขปัญหาเกี่ยวกับจลนศาสตร์ที่เกี่ยวกับการเคลื่อนที่ ซึ่งเกี่ยวข้องกับความสัมพันธ์ระหว่างข้อต่อของหุ่นยนต์แต่ละตัวกับตำแหน่ง และทิศทางของเครื่องมือหรือ (End-Effector) ตัวแปรร่วม คือ มุมระหว่างการเชื่อมโยงในข้อต่อ

- จลนศาสตร์ผกผัน (Inverse Kinematics) จะตรงข้ามกับสมการ Forward Kinematics กล่าวคือ จะเป็นการคำนวณจากค่าพิกัดคาร์ทีเซียนของปลายหุ่นยนต์ให้กลายเป็นค่ามุมของแต่ละข้อต่อ ซึ่งการคำนวณจลนศาสตร์ผกผันของหุ่นยนต์ต้องคำนึงถึงสองเรื่องหลัก เรื่องแรก คือ พารามิเตอร์ของข้อต่อที่ต้องการ เรื่องที่สอง คือ พารามิเตอร์คำตอบที่เป็นไปได้อาจมีหลายชุด

โดยสมการทั้งสองนี้จะเป็นส่วนที่ใช้ในการสร้าง Trajectory Path ให้กับหุ่นยนต์

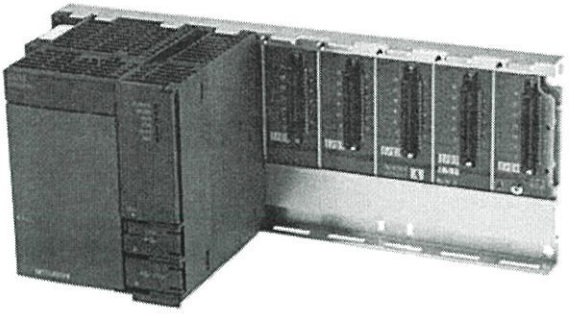
2.3.2 วิถีการเคลื่อนที่

วิถีการเคลื่อนที่ (Trajectory Path) คือ เส้นทางที่กำหนดให้วัตถุเคลื่อนที่ไปตามที่ต้องการ สามารถสร้างขึ้นได้ด้วยการสร้างพิกัดเป้าหมายหลาย ๆ จุดบน Trajectory Path ที่ต้องการแล้วจึงนำมาเรียงต่อกันจนเกิดเป็น Trajectory Path ของหุ่นยนต์

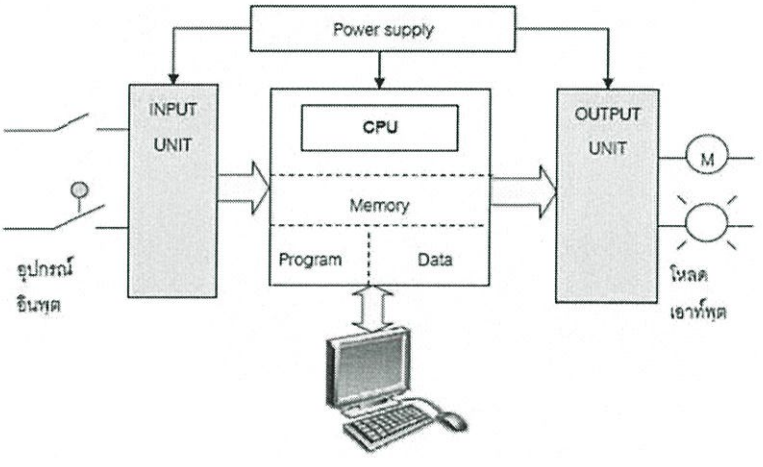
2.4 พีแอลซี [7]

พีแอลซี (Programmable Logic Controller: PLC) เป็นอุปกรณ์ที่ใช้ในการควบคุมการทำงานของระบบการผลิตในอุตสาหกรรมที่มีความทนทาน อุปกรณ์นี้ถูกคิดค้นเพื่อใช้ควบคุมการทำงานของเครื่องจักร ทดแทนวงจรรีเลย์แบบเก่า ซึ่งรีเลย์แบบเก่าจะมีข้อเสียในส่วนของการเดินสายไฟฟ้า และการเปลี่ยนแปลงเงื่อนไขในการควบคุมที่มีความยุ่งยาก เมื่อใช้ไปเป็นเวลานานจะทำให้หน้าสำผัสของรีเลย์เกิดการเสื่อมสะภาพลง โดยภายในจะประกอบไปด้วยระบบ Microprocessor

เป็นส่วนหลักในการสั่งการ วิเคราะห์ และประมวลผลในการทำงาน เปรียบเทียบได้กับสมองของ PLC จะมีส่วนที่เป็น Input ที่ใช้ต่อกับเครื่องมือวัด หรือสวิตช์ต่าง ๆ และส่วนที่เป็น Output จะใช้ต่อออกไปควบคุมการทำงานของอุปกรณ์ หรือเครื่องจักรเป้าหมายที่ต้องการจะควบคุม ผู้ใช้งานสามารถออกแบบการควบคุม หรือการทำงานได้โดยการเขียนคำสั่งป้อนเข้าไปใน PLC มากไปกว่านั้นยังสามารถใช้งานควบคู่ไปกับอุปกรณ์อื่น ๆ ได้อีก เช่น เครื่องพิมพ์ (Printer) เครื่องอ่านบาร์โค๊ด (Barcode Reader) นอกจากนั้นในปัจจุบัน PLC ยังสามารถทำงานได้แบบเดี่ยว (Standalone) และทำงานร่วมกับตัวอื่น ๆ (Network) เพื่อเพิ่มประสิทธิภาพในการควบคุมการทำงานของระบบ ทำให้ PLC มีการใช้งานที่ยืดหยุ่นมากยิ่งขึ้น แสดงดังรูปที่ 2.6



รูปที่ 2.6 Mitsubishi Programmable Logic Control รุ่น Q03UDECPU



รูปที่ 2.7 โครงสร้างของ PLC

รูปที่ 2.7 แสดงถึงโครงสร้างภายในของ PLC แต่ละส่วนที่จะทำงานร่วมกันเป็นระบบควบคุมเรียกว่า PLC ซึ่งประกอบด้วยส่วนสำคัญต่าง ๆ 5 ส่วน ดังนี้

1. หน่วยประมวลผลกลาง (Central Processing Unit: CPU) ซีพียู หรือหน่วยประมวลผลกลางทำหน้าที่ประมวลผลการทำงานตามคำสั่งของส่วนต่าง ๆ ตามที่ได้รับมา ผลจากการประมวลผลจะถูกส่งออกไปยังส่วนต่าง ๆ ตามที่ได้ทำการระบุไว้ในคำสั่งนั้น ๆ ซึ่งซีพียูจะใช้เวลาในการประมวลผลช้า หรือเร็วจะขึ้นอยู่กับขนาดของซีพียู และขนาดของโปรแกรมที่ได้เขียนป้อนไว้ในซีพียูของ PLC

2. หน่วยความจำ (Memory Unit) หน่วยความจำเป็นอุปกรณ์ที่ใช้สำหรับการเก็บโปรแกรมควบคุม และข้อมูลต่าง ๆ ในตัว PLC ที่สั่งการ ตัว PLC กรณีที่สั่ง Run PLC จะนำเอาโปรแกรมควบคุม และข้อมูลในหน่วยความจำมาประมวลผลการทำงาน ซึ่งหน่วยความจำที่ใช้งานอยู่ใน PLC มีด้วยกัน 2 แบบ คือ

- หน่วยความจำชั่วคราว (Read Only Memory: ROM) ใช้สำหรับจัดเก็บซอฟต์แวร์ของระบบ (System Software) และเป็นชุดโปรแกรมสำรอง และข้อมูล (Backup Program and Data) เพื่อป้องกันกรณีที่โปรแกรม และข้อมูลในหน่วยความจำชั่วคราวหายไป ซึ่งผู้ใช้งานสามารถถ่ายโอนโปรแกรม และข้อมูลเข้าไปเก็บไว้ที่หน่วยความจำชั่วคราวใหม่ได้

3. ภาคอินพุต (Input Unit) ภาคอินพุตของ PLC ทำหน้าที่รับสัญญาณอินพุตเข้ามาแปลงสัญญาณ และส่งเข้าไปภายใน PLC อุปกรณ์ที่สามารถนำมาต่อเข้ากับภาคอินพุต เช่น ตัวควบคุมอุณหภูมิ เซนเซอร์ชนิดใช้แสง ลิมิตสวิตช์ พร็อกซิมิตี้สวิตช์ รีเลย์ หรือเอ็นโค้ดเดอร์ เป็นต้น

วงจรภาคอินพุตแบ่งออกเป็น 2 แบบ คือ

- ดิจิตอลอินพุต (Digital Input Type) หมายถึงอินพุตที่รับรู้สัญญาณได้เพียงแค่ “ON” หรือ “OFF” เท่านั้น ตามโครงสร้างสามารถจำแนกอินพุตได้ 2 แบบ คือ วงจรอินพุตไฟฟ้ากระแสตรง (DC Input) และวงจรอินพุตไฟฟ้ากระแสสลับ (AC Input)

- อนาลอกอินพุต (Analog Input Type) เป็นอินพุตที่สามารถรับสัญญาณที่บอกเป็นปริมาณที่เปลี่ยนแปลงค่าได้เช่น 0-10 โวลต์ -10 ถึง 10 โวลต์ 1-5 โวลต์ หรือ 4-20 มิลลิแอมป์ ซึ่งเป็นสัญญาณมาตรฐานที่กำหนดไว้ใช้ในอุตสาหกรรม

4. ภาคเอาต์พุต (Output Unit) ภาคเอาต์พุตของ PLC ทำหน้าที่ในการนำส่งสัญญาณออกไปขับ โหลดชนิดต่าง ๆ ตามเงื่อนไขที่ได้โปรแกรมเอาไว้ อุปกรณ์ที่สามารถนำมาต่อเข้ากับภาคเอาต์พุต เช่น หลอกไฟ วาล์ว ตัวควบคุมมอเตอร์ ตัวควบคุมอุณหภูมิ คอนแทคเตอร์ เป็นต้น

ชนิดของเอาต์พุตของ PLC มี 2 ลักษณะ คือ

- ดิจิตอลเอาต์พุต (Digital Output Type) สัญญาณของเอาต์พุตจะมีลักษณะเป็นบูลีน คือ 0 กับ 1 หรือก็ คือ สัญญาณ “ON” และ “OFF” เพียงเท่านั้น

- อนาลอกเอาต์พุต (Analog Output Type) ค่าที่ส่งออกไปจัดเป็นค่าสัญญาณมาตรฐานเหมือนภาคอินพุตแบบอนาลอก คือ สัญญาณ 0-10 โวลต์ 1-5 โวลต์ หรือ 4-20 มิลลิแอมป์

5. ภาคแหล่งจ่ายพลังงาน (Power Supply Unit) ภาคแหล่งจ่ายพลังงานทำหน้าที่จ่ายพลังงานให้กับอุปกรณ์ภายใน PLC ได้แก่อุปกรณ์ไอซีเป็นต้น นอกจากนี้ยังสามารถจ่ายพลังงานไปเลี้ยงวงจรในส่วนที่นำเข้ามาต่อกับ PLC ในภาคอินพุตและภาคเอาต์พุต

PLC แต่ละยี่ห้อจะใช้ภาษาในการเขียนโปรแกรมเพื่อความคุมการทำงานเพื่อสั่งให้ PLC ทำงานตามความต้องการที่แตกต่างกันไป ซึ่งตามมาตรฐาน IEC 61131-3 ได้แบ่งมาตรฐานภาษาที่ใช้ในการเขียนแบ่งออกเป็น 5 แบบ ได้แก่

- Sequential Flow Chart Language
- Structure Text Language
- Function Block Diagram Language
- Instruction List Language
- Ladder Diagram

2.5 ส่วนติดต่อกับผู้ใช้งาน [8]

ส่วนติดต่อกับผู้ใช้งาน (Human Machine Interface: HMI) คือ การใช้งานร่วมกันระหว่าง PLC Programming กับเครื่องคอมพิวเตอร์ จึงเรียกว่า HMI (Human Machine Interface) โดยนำคอมพิวเตอร์มาเป็นอุปกรณ์ที่ใช้ในการติดต่อระหว่างผู้ใช้งานกับเครื่องจักร เพื่อควบคุม และเป็นจอแสดงผล HMI รวมไปถึง SCADA เกิดจากความต้องการของผู้ใช้งานที่ต้องการเข้าไปควบคุมระบบที่ PLC เป็นตัวควบคุมอยู่ โดย HMI นั้น จะเป็นการนำข้อมูลจาก PLC ส่งผ่านโครงข่ายของการสื่อสารแบบต่าง ๆ และทำการรวบรวมข้อมูลในรูปแบบต่าง ๆ เข้าด้วยกัน อีกทั้งสามารถสั่งการได้โดยผู้เชี่ยวชาญ แสดงในรูปที่ 2.8

รูปที่ 2.8 Human Machine Interface (HMI)

คุณสมบัติของ HMI

1. การสื่อสาร (Communication)

HMI สามารถสื่อสารข้อมูลกับอุปกรณ์อื่น ๆ ในลักษณะแบบดิจิตอล โดยมีรูปแบบของสัญญาณให้เลือกหลากหลายแบบ และสามารถสื่อสารข้อมูลกับอุปกรณ์ต่าง ๆ หลายหลายยี่ห้อได้อย่างมี ประสิทธิภาพ สามารถต่อได้ทั้งอุปกรณ์ PLC, Meter, Controller และอีกมากมายตามการ

ใช้งาน ประเภทต่าง ๆ โดยอุปกรณ์ HMI เพียงตัวเดียวสามารถควบคุม และอ่านค่าอุปกรณ์ฮาร์ดแวร์ตัวอื่น ๆ ที่เชื่อมต่ออยู่ได้อย่างมีประสิทธิภาพง่ายดาย ผ่านการเชื่อมต่อทางเครือข่าย Internet, Lan หรือ Wireless

2. การเก็บค่า (Collection)

HMI สามารถเก็บข้อมูลกระบวนการผลิตต่าง ๆ ในรูปแบบไฟล์ Excel รวมไปถึงการเข้าถึงข้อมูล (Data Logger) ผ่านทาง Web Browser ได้อย่างง่ายดายทำให้สะดวกในการทราบข้อมูล แม้ไม่ได้อยู่ที่หน้าไลน์ผลิตงาน

3. การเชื่อมต่อ (Connection)

- สามารถอำนวยความสะดวกให้กับผู้ใช้งานในการอ่านค่า หรือควบคุมกระบวนการผลิตจากระยะไกล โดยการเชื่อมต่อผ่านมือถือหรือแท็บเล็ต

- ใช้เว็บบราวเซอร์มาตรฐานตัวใดก็ได้ในการดูค่าหรือควบคุม โดยหน้าจอแสดงผลโชว์หน้าตาเสมือนว่าอยู่ตรงหน้า

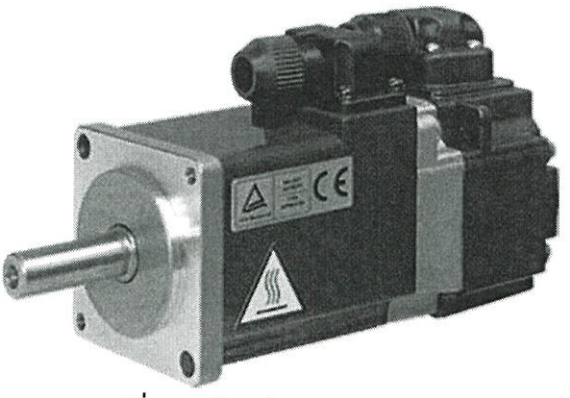
- สามารถส่งข้อความ SMS หรือ email แจ้งเตือนให้กับบุคคลที่เกี่ยวข้อง

- สามารถดูค่าที่หน้าจอ, ค่าที่บันทึกไว้ในMemory Card หรือควบคุมแก้ไขเปลี่ยนค่าได้แม้ไม่ได้อยู่ที่หน้างาน

2.6 เซอร์โวมอเตอร์ (Servo Motor) [10]

Servo Motor เป็นมอเตอร์ที่มีการควบคุมการเคลื่อนที่ (State) ไม่ว่าจะเป็นระยะ ความเร็ว มุมการหมุน โดยใช้การควบคุมแบบป้อนกลับ (Feedback control) เป็นอุปกรณ์ที่สามารถควบคุมเครื่องจักรกล หรือระบบการทำงาน ให้เป็นไปตามความต้องการ เช่น ควบคุมความเร็ว (Speed), ควบคุมแรงบิด (Torque), ควบคุมตำแหน่ง (Position), ระยะทางในการเคลื่อนที่ (Position Control) ของตัวมอเตอร์ได้ ซึ่งมอเตอร์ทั่วไปไม่สามารถควบคุมในลักษณะงานเบื้องต้นได้ โดยให้ผลลัพธ์ตามความต้องการที่มีความแม่นยำสูง แสดงดังรูปที่ 2.9

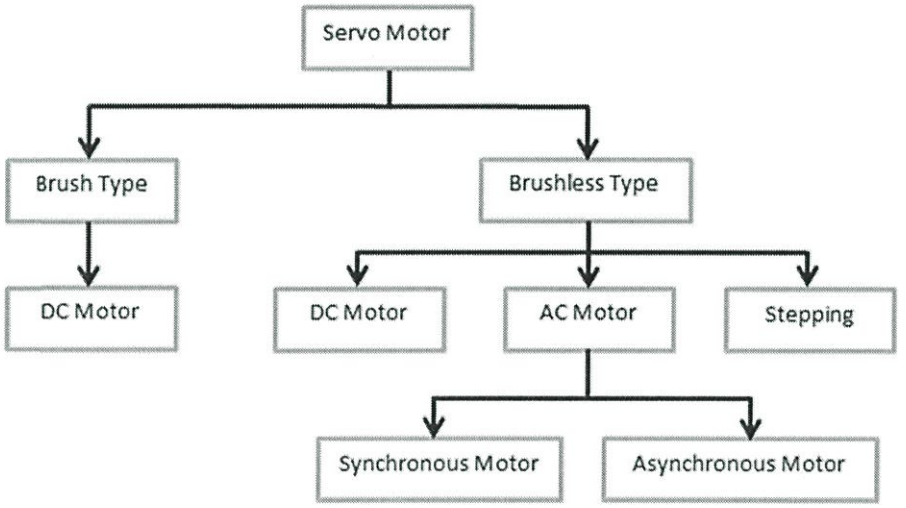
ขนาดของ Servo Motor จะมีหน่วยในการบอกขนาดเป็นวัตต์ (Watt) Servo Motor จะมีขนาดตั้งแต่ 50W-15kWทำให้ผู้ใช้งานมีความหลากหลายในการใช้งาน



รูปที่ 2.9 ตัวอย่าง Servo Motor

2.6.1 ประเภทของ Servo Motor

โดยทั่วไปจะมี Servo Motor ทั้งหมด 2 ประเภท ได้แก่ DC และ AC Servo Motor ในระบบอุตสาหกรรมจะพบว่า DC Servo Motor มีการใช้งานกับเครื่องจักรกลอุตสาหกรรมมากกว่า AC Servo Motor เนื่องจากช่วงที่ผ่านมาการควบคุมกระแส กระแสสูง ๆ นั้นจะต้องใช้ เอสซีอาร์ (Silicon Control Rectifier: SCRs) แต่ปัจจุบันทรานซิสเตอร์ได้พัฒนาขีดความสามารถให้ตัดต่อกระแสสูง และสามารถใช้งานที่ความถี่สูงขึ้นได้ จึงทำให้ระบบควบคุมทางเอซี และระบบเซอร์โวได้ถูกนำมาใช้งานมากขึ้น ซึ่งสามารถแยกประเภทของเซอร์โวได้ ดังรูปที่ 2.10



รูปที่ 2.10 ประเภทของ Servo Motor

1. Servo Motor ชนิดที่มีแปรงถ่าน

Servo Motor ชนิดนี้ที่สเตเตอร์จะเป็นแม่เหล็กถาวร ส่วนโรเตอร์ยังใช้แปรงถ่าน และคอมมิวเตอร์เรียงกระแสเข้าสู่ขดลวดอาร์เมเจอร์ เหมือนกับดีซีมอเตอร์ทั่วไป

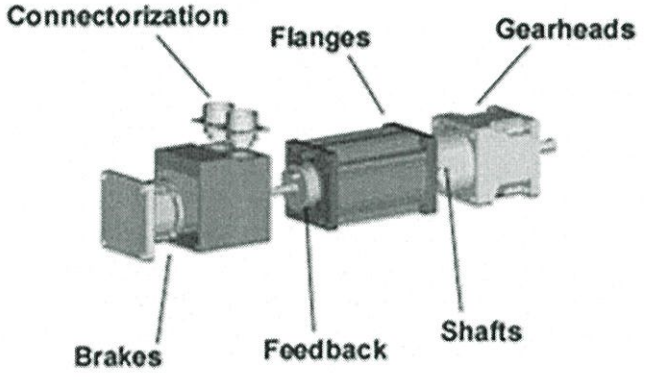
2. Servo Motor ชนิดที่ไม่มีแปรงถ่าน

Servo Motor ในกลุ่มนี้ประกอบด้วยดีซีเซอร์โว (DC Brushless Servo) ซึ่งโรเตอร์ทำด้วยแม่เหล็กถาวร เอซีเซอร์โว (AC Servo) ซึ่งมีทั้งแบบซิงโครนัสเซอร์โวมอเตอร์ (Synchronous Motor) และอะซิงโครนัสเซอร์โวมอเตอร์ (Asynchronous Motor) การนำอินดัคชั่นมอเตอร์ (Induction Motor) มาใช้ทำเป็นระบบขับเคลื่อน Servo Motor และ สเตปปิ้งเซอร์โวมอเตอร์ (Stepping Motor)

2.6.2 โครงสร้างของ Servo Motor

ข้อจำกัดอย่างหนึ่งของระบบควบคุมเซอร์โว คือ การใช้งานจะต้องเป็นแบบ Closed loop เท่านั้น การใช้งานระบบควบคุมเซอร์โวไม่สามารถเลือกควบคุมเป็นแบบ Open loop ได้ เหมือนกับระบบขับเคลื่อนเอซี (AC Drives) การตอบสนองของระบบเซอร์โว เช่น อัตราเร่ง แรงบิด และตำแหน่งที่ควบคุม จะไม่เป็นไปตามวัตถุประสงค์หากไม่มีสัญญาณป้อนกลับไปยังชุดขับเคลื่อนเซอร์โว

การควบคุมการทำงานในระบบนี้ อุปกรณ์ป้อนกลับหรือเอ็นโค๊ดเตอร์ (Encoder) จะมีบทบาทความสำคัญอย่างยิ่ง ในทางปฏิบัติ Servo Motor และ Encoder ถูกออกแบบ และผลิตสร้างขึ้นมาคู่กันในลักษณะเป็นแพ็คเกจ (Package) ซึ่งมี Encoder ติดอยู่ที่ส่วนท้ายของมอเตอร์ ดังรูปที่ 2.11



รูปที่ 2.11 โครงสร้างของ AC Servo Motor

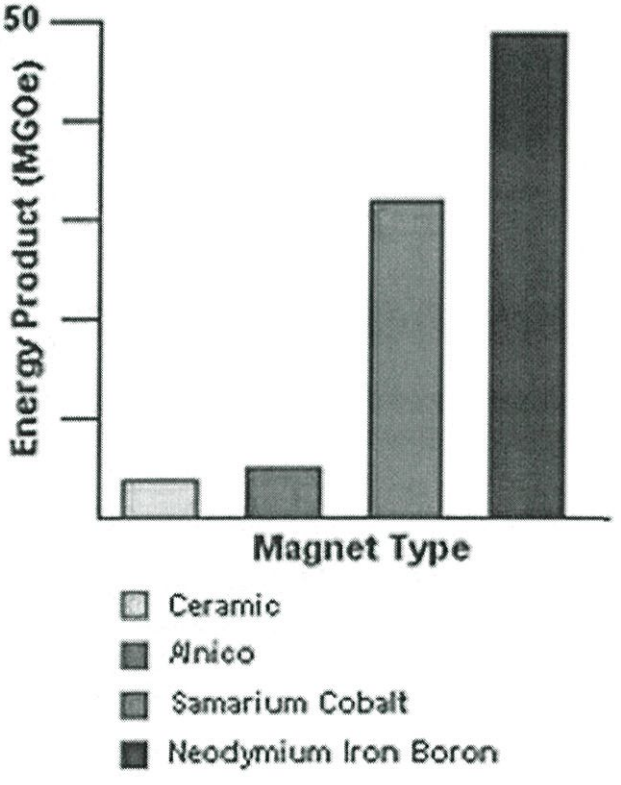
โครงสร้างของ AC servo Motor

- Gearheads = เกียร์สำหรับลดความเร็วรอบเพื่อเพิ่มแรงบิด
- Shafts = เพลาของมอเตอร์
- Flanges = หน้าแปลนสำหรับติดตั้งมอเตอร์
- Feedback = อุปกรณ์ป้อนกลับ หรือ encoder
- Connectorization = ขั้วต่อสายไฟเข้ามอเตอร์ และขั้วต่อสายสำหรับEncoder
- Brakes = ชุดเบรก

โครงสร้างของ AC Servo Motor จะคล้ายกับมอเตอร์ 3 เฟสทั่ว ๆ ไป ซึ่งจะประกอบด้วย 2 ส่วนที่สำคัญ คือ สเตเตอร์และโรเตอร์ โดยสเตเตอร์จะประกอบด้วยขดลวด 3 ชุด ขดลวดภายในจะต่อเป็นแบบสตาร์ (Star หรือ WYE) และมีสายต่อมาที่ขั้วต่อสายด้านนอก 3 เส้น (จุดนิวทรอลจะอยู่ด้านใน) ส่วนโรเตอร์ทำด้วยแม่เหล็กถาวร (Permanent Magnet) ไม่มีขดลวดพัน ไม่มีคอมมิวเตเตอร์ และไม่มีแปรงถ่าน (Brushless)

โครงสร้างที่ไม่มีขดลวดพัน และไม่มีแปรงถ่าน จะทำให้ประสิทธิภาพของมอเตอร์สูงขึ้น เพราะไม่เกิดการสูญเสียในขดลวดทองแดง ไม่ต้องบำรุงรักษาแปรงถ่าน ไม่เกิดประกายไฟเนื่องจากการเรียงกระแสจากแปรงถ่านผ่านคอมมิวเตอร์ไปยังขดลวดทองแดงที่พันอยู่ในตัวโรเตอร์

สำหรับวัสดุที่นำมาสร้างแม่เหล็กถาวรนี้จะแตกต่างกันไป โดยขึ้นอยู่กับราคา และเทคโนโลยีของบริษัทผู้ผลิตนั้น ๆ ซึ่งมีตั้งแต่ชนิดที่ราคาถูกเช่น เซรามิก (เฟอไรต์) จนถึงการใช้วัสดุที่มีราคาแพงอย่างเช่น ซามาเรียม โคบอลต์หรือนีโอไดเมียม เป็นต้น ซึ่งปัจจุบัน AC Servo Motor ส่วนใหญ่จะใช้วัสดุสารแม่เหล็กแบบ นีโอไดเมียม เนื่องจากมีคุณสมบัติความเป็นแม่เหล็ก และความเหมาะสมเรื่องราคาดีกว่า เมื่อเปรียบเทียบกับวัสดุสารแม่เหล็กแบบอื่น ๆ แสดงดังรูปที่ 2.12

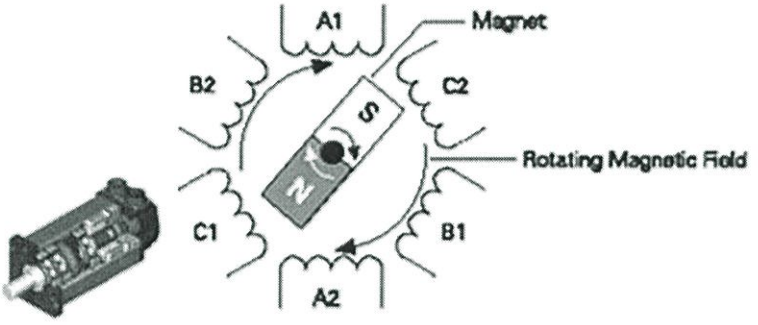


รูปที่ 2.12 วัสดุที่นำมาสร้างแม่เหล็กถาวร

2.6.3 หลักการทำงานของ Servo Motor

การทำงานของ Servo Motor ชนิดนี้จะคล้ายกับการทำงานของซิงโครนัสมอเตอร์ 3 เฟส กล่าวคือ เมื่อมีการควบคุมให้คอนโทรลเลอร์จ่ายกระแสไฟฟ้าเข้าไปยังขดลวดที่สเตเตอร์ แกนเหล็กของสเตเตอร์จะกลายเป็นแม่เหล็กไฟฟ้า และหมุนเคลื่อนที่ด้วยความเร็วที่แปรผันตามความถี่ ซึ่งเรียกว่า ความเร็วซิงโครนัส (Synchronous Speed) หรือความเร็วสนามแม่เหล็กหมุน และจะดูดให้โรเตอร์ซึ่งเป็นแม่เหล็กถาวรหมุนเคลื่อนที่ตาม

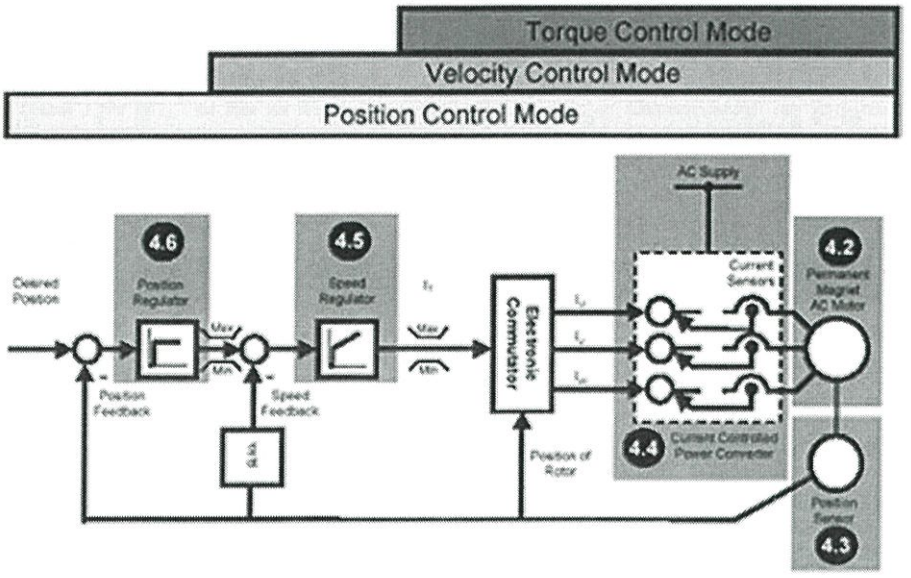
จากลักษณะโครงสร้างของโรเตอร์ และหลักการทำงานที่เหมือนกับซิงโครนัสมอเตอร์ซึ่งเป็นมอเตอร์แบบเอซี แต่ไม่มีแปรงถ่าน ไม่มีซี่คอมมิวเตอรเตอร์ จึงทำให้มอเตอร์ชนิดนี้มีชื่อเรียกแตกต่างกันออกไป เช่น เรียกทับศัพท์ว่า Permanent Magnet Synchronous Motor (PMSM) ซึ่งหมายถึงซิงโครนัสมอเตอร์ที่ไม่มีแปรงถ่าน บ้างก็เรียกว่าเอซีเซอร์โวมอเตอร์ (AC Servo motor) หรือบ้างก็เรียกว่า AC Brushless หรือ Brushless Motor แสดงดังรูปที่ 2.13



รูปที่ 2.13 โครงสร้างและการทำงานของ AC Servo Motor

2.6.4 โครงสร้างของระบบควบคุม Servo Motor

ลักษณะของระบบควบคุม Servo Motor จะเป็นระบบควบคุมแบบลูปปิด (Closed loop control) ซึ่งประกอบด้วย 3 โหมดการควบคุม คือ โหมดการควบคุมแรงบิด (Torque Control Mode) โหมดการควบคุมอัตราเร่ง (Velocity Control Mode) และโหมดการควบคุมตำแหน่ง (Position Control Mode) โดยมีองค์ประกอบที่สำคัญ ๆ ดังรูปที่ 2.14



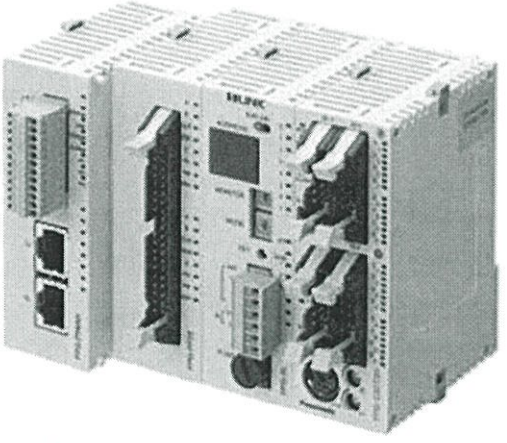
รูปที่ 2.14 โครงสร้างของระบบควบคุม Servo Motor

1. Servo Motor (แสดงในรูปที่ 2.14 ตำแหน่งที่ 4.2)
2. ชุดควบคุมการขับเคลื่อนเซอร์โว (Servo Drive, Servo Amplifier หรือ Servo Controller) (แสดงในรูปที่ 2.14 ตำแหน่งที่ 4.4, 4.5, 4.6)
3. อุปกรณ์ป้อนกลับ (แสดงในรูปที่ 2.14 ตำแหน่งที่ 4.3)

2.6.5 องค์ประกอบในการทำงานของ Servo Motor

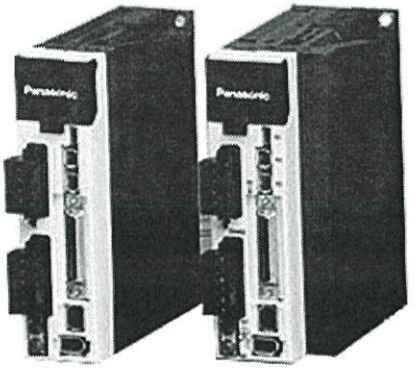
การทำงานเพียงตัว Servo Motor เพียงอย่างเดียวนั้นไม่สามารถทำงานได้ การที่จะให้ Servo Motor จะควบคุมลักษณะที่กล่าวมาข้างต้นนั้นต้องมีองค์ประกอบดังนี้

1. ตัวควบคุม Servo Motor (Controller) คือ อุปกรณ์ที่มีหน้าที่รับคำสั่งจากผู้ใช้งานว่าต้องการให้ Servo Motor นั้นเคลื่อนที่ด้วยความเร็ว และระยะทางที่ต้องการ หน้าที่ตรงจุดนี้ Controller จะเป็นตัวกำหนดให้กับตัว Servo Motor แสดงดังรูปที่ 2.15



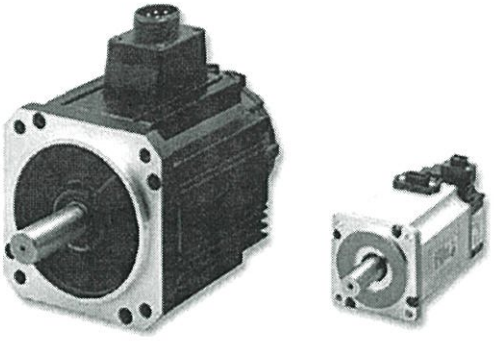
รูปที่ 2.15 ตัวอย่างตัวควบคุม Servo Motor

2. เซอร์โวแอมปลิฟายเออร์ (Servo Amplifier) คือ อุปกรณ์ที่รับสัญญาณมาจาก Controller และสั่งการให้กับตัว Servo Motor เคลื่อนที่ตามที่ Controller สั่งการมา แต่ Controller จะไม่สั่งการควบคุมไปที่ Servo Motor โดยตรง เนื่องจาก Servo Amplifier จะเป็นตัวที่ปรับตั้งค่าของตัว Servo Motor ให้สามารถทำงานตามรูปแบบของการควบคุมไม่ว่าจะเป็นการควบคุม ความเร็ว (Speed Control), แรงบิด(Torque) และตำแหน่ง (Position Control) ตัว Servo Amplifier จะเป็นตัวกำหนดค่าตัวแปร หรือพารามิเตอร์ต่าง ๆ ให้กับตัว Servo Motor ให้ทำงานได้อย่างถูกต้อง และแม่นยำ แสดงดังรูปที่ 2.16



รูปที่ 2.16 ตัวอย่างเซอร์โวแอมปลิฟายเออร์ (Servo Amplifier)

3. Servo Motor มีหน้าที่ขับเคลื่อนอุปกรณ์ของเครื่องจักรกล หรือระบบของการทำงานนั้น ๆ ให้เป็นไปตามรูปแบบที่ได้รับคำสั่งจากตัว Servo Driver พร้อมกับส่งสัญญาณป้อนกลับให้กับตัว Servo Driver ว่าตอนนี้ Servo Motor เคลื่อนที่ด้วย ความเร็วเท่าไหร่และระยะทางในการเคลื่อนที่เป็นระยะทางเท่าไหร่แล้ว ด้วยสัญญาณของตัว Encoder ที่อยู่ภายในตัว Servo Motor ทำให้การเคลื่อนที่ของ Servo Motor นั้นมีความแม่นยำสูง แสดงดังรูปที่ 2.17



รูปที่ 2.17 ตัวอย่าง Servo Motor

2.7 ฐานข้อมูล [11]

ฐานข้อมูลประกอบด้วยกลุ่มการจัดการข้อมูลสำหรับผู้ใช้งานหนึ่งคน หรือหลาย ๆ คน โดยทั่วไปมักอยู่ในรูปแบบดิจิตอล วิธีการแบ่งชนิดของฐานข้อมูลมีหลายรูปแบบ ซึ่งรูปแบบหนึ่ง คือ แบ่งตามชนิดของเนื้อหา เช่น บรรณานุกรม เอกสารตัวอักษร สถิติ โดยฐานข้อมูลดิจิตอลจะถูกจัดการโดยใช้ระบบจัดการฐานข้อมูลซึ่งเก็บเนื้อหาฐานข้อมูล โดยอนุญาตให้สร้าง ดูแลรักษา ค้นหา และการเข้าถึงในรูปแบบอื่น ๆ

2.7.1 ระบบจัดการฐานข้อมูล

ระบบจัดการฐานข้อมูล (DBMS) ประกอบด้วยซอฟต์แวร์ที่ใช้ในการจัดการฐานข้อมูล, จัดเตรียมพื้นที่ในการเก็บ, การเข้าถึง, ระบบรักษาความปลอดภัย, สำรองข้อมูล และสิ่งอำนวยความสะดวกอื่น ๆ ระบบจัดการฐานข้อมูลสามารถแบ่งหมวดหมู่ได้ตามแบบจำลองฐานข้อมูล (Database Model) ที่สนับสนุน อาทิเช่น แบบจำลองเชิงสัมพันธ์ (Relational Model) หรือฐานข้อมูล XML เป็นต้น แบ่งตามประเภทของคอมพิวเตอร์ที่สนับสนุน อาทิเช่น Server Cluster หรือโทรศัพท์พกพา เป็นต้น แบ่งตามประเภทของภาษาสอบถามที่ใช้ในการเข้าถึงฐานข้อมูล อาทิเช่น ภาษาสอบถามเชิงโครงสร้าง หรือ XQuery แบ่งตามประสิทธิภาพในการ Trade-offs อาทิเช่น ขนาดที่ใหญ่ที่สุด หรือความเร็วสูงสุด และอื่น ๆ เป็นต้น ในบาง DBMS จะครอบคลุมมากกว่าหนึ่งหมวดหมู่ เช่น สนับสนุนภาษาสอบถามได้หลาย ๆ ภาษา ยกตัวอย่างเช่น ใน DBMS ที่นิยมใช้การอย่างแพร่หลาย MySQL, PostgreSQL, Microsoft Access, SQL Server, FileMaker, Oracle, Sybase, dBASE, Clipper, FoxPro และอื่น ๆ ในทุก ๆ ซอฟต์แวร์ฐานข้อมูลจะมี Open Database Connectivity (ODBC) Driver มาให้ด้วย เพื่ออนุญาตให้ฐานข้อมูลสามารถทำงานร่วมกับฐานข้อมูลแบบอื่น ๆ ได้

2.7.2 การออกแบบฐานข้อมูล

การออกแบบฐานข้อมูล (Designing Databases) มีความสำคัญต่อการจัดการระบบฐานข้อมูล (DBMS) ทั้งนี้เนื่องจากข้อมูลที่อยู่ภายในฐานข้อมูลจะต้องศึกษาถึงความสัมพันธ์ของข้อมูล โครงสร้างของข้อมูลสำหรับการเข้าถึงข้อมูล และกระบวนการที่โปรแกรมจะประยุกต์เพื่อเรียกใช้ฐานข้อมูล ดังนั้น ผู้ใช้งานสามารถแบ่งวิธีการสร้างฐานข้อมูลได้ 3 ประเภท

1. รูปแบบข้อมูลแบบลำดับขั้น หรือโครงสร้างแบบลำดับขั้น (Hierarchical Data Model) วิธีการสร้างฐานข้อมูลแบบลำดับขั้นถูกพัฒนาโดยบริษัท ไอบีเอ็ม จำกัด ในปี 1980 ได้รับความนิยมมาก ในการพัฒนาฐานข้อมูลบนเครื่องคอมพิวเตอร์ขนาดใหญ่ และขนาดกลาง โดยที่โครงสร้างข้อมูลจะสร้างรูปแบบเหมือนต้นไม้ โดยความสัมพันธ์เป็นแบบหนึ่งต่อหลาย (One- to -Many)

2. รูปแบบข้อมูลแบบเครือข่าย (Network Data Model) ฐานข้อมูลแบบเครือข่ายมีความคล้ายคลึงกับฐานข้อมูลแบบลำดับชั้น ต่างกันที่โครงสร้างแบบเครือข่ายอาจจะมีการติดต่อหลายต่อหนึ่ง (Many-to-one) หรือหลายต่อหลาย (Many-to-many) กล่าวคือ ลูก (Child) อาจมีพ่อแม่ (Parent) มากกว่าหนึ่ง สำหรับตัวอย่างฐานข้อมูลแบบเครือข่ายให้ลองพิจารณาการจัดการข้อมูลของห้องสมุด ซึ่งรายการจะประกอบด้วย ชื่อเรื่อง ผู้แต่ง สำนักพิมพ์ ที่อยู่ ประเภท

3. รูปแบบความสัมพันธ์ข้อมูล (Relational Model) เป็นลักษณะการออกแบบฐานข้อมูลโดยจัดข้อมูลให้อยู่ในรูปของตารางที่มีระบบคล้ายแฟ้ม โดยที่ข้อมูลแต่ละแถว (Row) ของตารางจะแทนเรคอร์ด (Record) ส่วนข้อมูลในแนวดิ่งจะแทนคอลัมน์ (Column) ซึ่งเป็นขอบเขตของข้อมูล (Field) โดยที่ตารางแต่ละตารางที่สร้างขึ้นจะเป็นอิสระ ดังนั้น ผู้ออกแบบฐานข้อมูลจะต้องมีการวางแผนถึงตารางข้อมูลที่จำเป็นต้องใช้ เช่น ระบบฐานข้อมูลบริษัทแห่งหนึ่งประกอบด้วย ตารางประวัติพนักงาน ตารางแผนก และตารางข้อมูลโครงการ แสดงประวัติพนักงาน ตารางแผนก และตารางข้อมูลโครงการ

2.7.3 การออกแบบฐานข้อมูลเชิงสัมพันธ์

การออกแบบฐานข้อมูลในองค์กรขนาดเล็กเพื่อตอบสนองความต้องการของผู้ใช้งานอาจเป็นเรื่องที่ไม่ยุ่งยากนัก เนื่องจากระบบ และขั้นตอนการทำงานภายในองค์กรไม่ซับซ้อนปริมาณข้อมูลที่มีก็ไม่มาก และจำนวนผู้ใช้งานฐานข้อมูลก็มีเพียงไม่กี่คน แต่ทว่าในองค์กรขนาดใหญ่ ซึ่งมีระบบ และขั้นตอนการทำงานที่ซับซ้อนรวมทั้งมีปริมาณข้อมูล และผู้ใช้งานจำนวนมากการออกแบบฐานข้อมูลจะเป็นเรื่องที่มีความละเอียดซับซ้อน และต้องใช้เวลาในการดำเนินการนานพอควรทีเดียว ทั้งนี้ฐานข้อมูลที่ได้รับการออกแบบอย่างเหมาะสมจะสามารถตอบสนองต่อความต้องการของผู้ใช้งานภายในหน่วยงานต่าง ๆ ขององค์กรได้จะทำให้การดำเนินงานขององค์กรมีประสิทธิภาพดียิ่งขึ้น เป็นผลตอบแทนที่คุ้มค่าต่อการลงทุนเพื่อพัฒนาระบบฐานข้อมูลภายในองค์กรทั้งนี้ การออกแบบฐานข้อมูลที่นำซอฟต์แวร์ระบบจัดการฐานข้อมูลมาช่วยในการดำเนินการสามารถจำแนกหลักในการดำเนินการได้ 6 ขั้นตอน คือ

1.การรวบรวม และวิเคราะห์ความต้องการในการใช้ข้อมูล

2.การเลือกระบบจัดการฐานข้อมูล

3.การออกแบบฐานข้อมูลในระดับแนวคิด

4.การนำฐานข้อมูลที่ออกแบบในระดับแนวคิดเข้าสู่ระบบจัดการฐานข้อมูล

5.การออกแบบฐานข้อมูลในระดับกายภาพ

6.การนำฐานข้อมูลไปใช้ และการประเมินผล

2.7.4 การออกแบบฐานข้อมูลในระดับตรรกะ

การออกแบบฐานข้อมูลในระดับตรรกะ หรือในระดับแนวความคิด เป็นขั้นตอนการออกแบบความสัมพันธ์ระหว่างข้อมูลในระบบโดยใช้แบบจำลองข้อมูลเชิงสัมพันธ์ ซึ่งอธิบายโดยใช้

แผนภาพแสดงความสัมพันธ์ระหว่างข้อมูล (E-R Diagram) จากแผนภาพ E-R Diagram นำมาสร้างเป็นตารางข้อมูล (Mapping E-R Diagram to Relation) และใช้ทฤษฎีการ Normalization เพื่อเป็นการรับประกันว่าข้อมูลมีความซ้ำซ้อนกันน้อยที่สุด ซึ่งการออกแบบเชิงตรรกะนี้จะบอกถึงรายละเอียดของ Relation, Attribute และ Entity

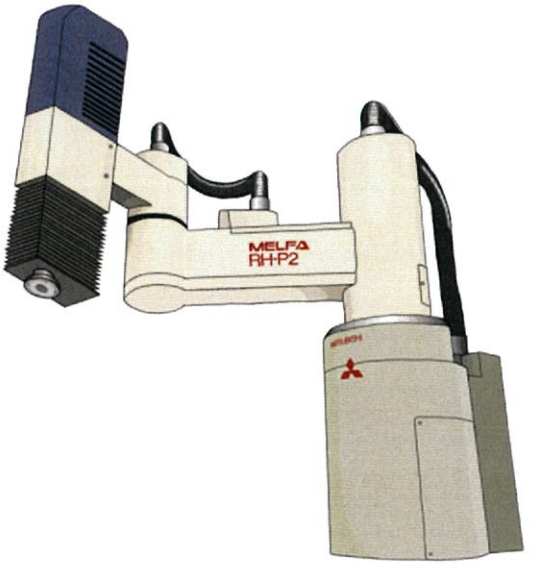
บทที่ 3
ตัวควบคุมหุ่นยนต์ที่นำเสนอ

3.1 กล่าวนำ

สำหรับบทนี้จะกล่าวถึงรายละเอียดในการดำเนินงานขั้นตอนต่าง ๆ ซึ่งจะมีการอธิบายเนื้อหาตั้งแต่การศึกษาโครงสร้าง และสมการทางคณิตศาสตร์เพื่อนำข้อมูลไปตั้งค่าให้กับ PLC รุ่น MELSEC Q-Series จนไปถึงการสร้าง Function Blocks ให้หุ่นยนต์สามารถทำงานได้ 5 รูปแบบ ได้แก่ Jog Operation, Automatic Position Control, M-code Event Selection, Coordinate Movement Control และ Trajectory-path Movement Control

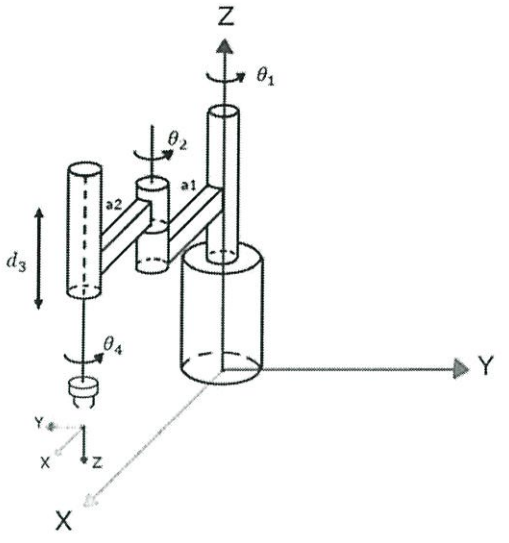
3.2 สรีระหุ่นยนต์ที่ศึกษา

หุ่นยนต์ที่ได้ทำการศึกษานั้น คือ แขนกลหรือแขนหุ่นยนต์ (Robot Arm) รุ่น MELFA RH-P2 ปี ค.ศ.1987 ที่เป็นหุ่นยนต์ประเภท SCARA ซึ่งประกอบไปด้วยจำนวน Joint และ Link เท่ากับ 4 และ 2 ตามลำดับ มีความยาวแขนอยู่ที่ 650 mm. มี Maximum Payload ที่ 3 kg และมีการทำงานในลักษณะแขนขวาใน Quadrant ที่ 1 และ 2 โดยได้ทำการติด Servo Motor ไว้ในแต่ละ Joint ของหุ่นยนต์ แสดงดังรูปที่ 3.1



รูปที่ 3.1 SCARA Robot รุ่น MELFA RH-P2 ค.ศ. 1987

3.3 สมการทางคณิตศาสตร์ของหุ่นยนต์ที่ศึกษา

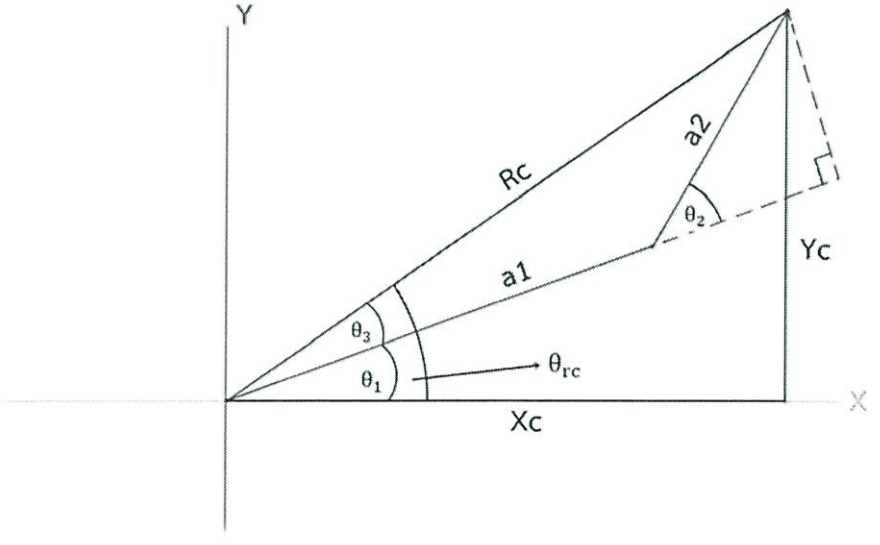


รูปที่ 3.2 SCARA Robot Schematic

จากรูปที่ 3.2 แสดงถึง การอ้างอิงแกน X, Y และ Z ของหุ่นยนต์

3.3.1 สมการ Inverse Kinematics

1) Quadrant1



รูปที่ 3.3 Inverse Kinematics Quadrant1

จากรูปที่ 3.3 สามารถแสดงความสัมพันธ์ได้ดังนี้

$$R_c^2 = X_c^2 + Y_c^2 \qquad (3.1)$$

ซึ่งภายในแต่ละแนวแกนจะประกอบไปด้วยความยาวของ Link และมุมที่หุ่นยนต์เคลื่อนที่

$$R_c^2 = (a_1 + a_2\cos\theta_2)^2 + (a_2\sin\theta_2)^2 \quad (3.2)$$

สามารถจัดรูปสมการได้ดังต่อไปนี้

$$R_c^2 = a_1^2 + 2a_1a_2\cos\theta_2 + a_2^2\cos^2\theta_1 + a_2^2\sin^2\theta_2 \quad (3.3)$$

$$R_c^2 = a_1^2 + 2a_1a_2\cos\theta_2 + a_2^2(\cos^2\theta_1 + \sin^2\theta_2) \quad (3.4)$$

$$R_c^2 = a_1^2 + 2a_1a_2\cos\theta_2 + a_2^2 \quad (3.5)$$

นำสมการที่ (3.1) มาเทียบกับสมการที่ (3.2) จะได้ว่า

$$X_c^2 + Y_c^2 = a_1^2 + 2a_1a_2\cos\theta_2 + a_2^2 \quad (3.6)$$

จากสมการที่ (3.6) สามารถแก้สมการได้ดังต่อไปนี้

$$\cos\theta_2 = \frac{X_c^2+Y_c^2-(a_1^2+a_2^2)}{2a_1a_2} \quad (3.7)$$

$$\theta_2 = \cos^{-1}\left(\frac{X_c^2+Y_c^2-(a_1^2+a_2^2)}{2a_1a_2}\right) \quad (3.8)$$

สมการต่อไปนี้จะเป็นสมการที่ใช้ในการหา θ_{rc}

$$\cos\theta_{rc} = \frac{X_c}{R_c} \quad (3.9)$$

$$\theta_{rc} = \cos^{-1}\left(\frac{X_c}{R_c}\right) \quad (3.10)$$

สมการต่อไปนี้จะเป็นสมการที่ใช้ในการหา θ_3

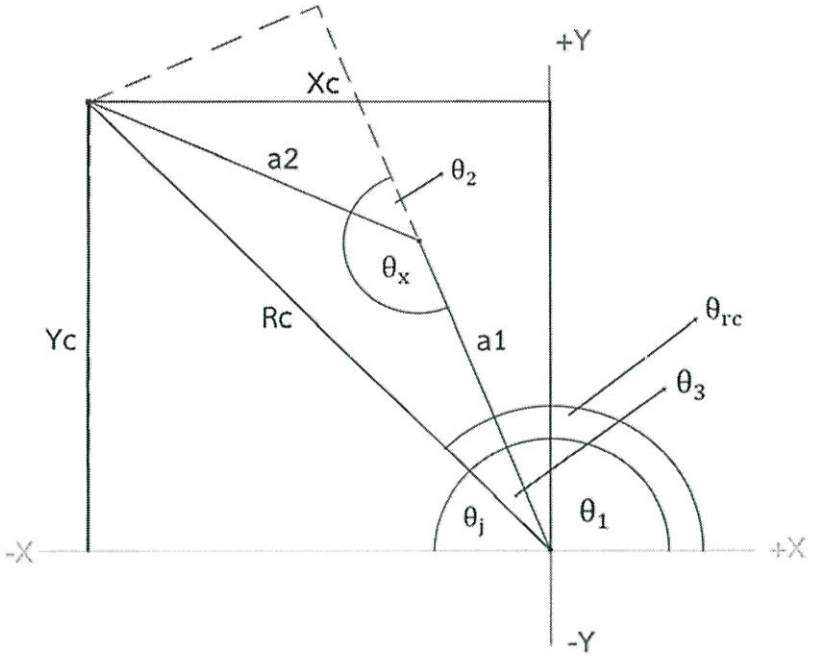
$$\cos\theta_3 = \frac{a_1+a_2\cos\theta_2}{R_c} \quad (3.11)$$

$$\theta_3 = \cos^{-1}\left(\frac{a_1+a_2\cos\theta_2}{R_c}\right) \quad (3.12)$$

สมการต่อไปนี้จะเป็นสมการที่ใช้ในการหามุมที่ Link ที่ 1 กระทำกับแกน X

$$\theta_1 = \theta_{rc} - \theta_3 \quad (3.13)$$

2) Quadrant2



รูปที่ 3.4 Inverse Kinematics Quadrant2

จากรูปที่ 3.4 สามารถหาความสัมพันธ์ได้ ดังนี้

ในการหาองศาในการเคลื่อนที่ของหุ่นยนต์ใน Quadrant2 จะใช้สมการเดียวกับการหาค่าใน Quadrant1 นั่นคือสมการที่ (3.8)

$$\theta_2 = \cos^{-1}\left(\frac{X_c^2+Y_c^2-(a_1^2+a_2^2)}{2a_1a_2}\right) \quad (3.8)$$

เพื่อที่จะหาค่า θ_j จำเป็นจะต้องใช้สมการดังต่อไปนี้

$$\sin\theta_j = \frac{Y_c}{R_c} \quad (3.15)$$

$$\theta_j = \sin^{-1}\left(\frac{Y_c}{R_c}\right) \quad (3.16)$$

สมการต่อไปนี้จะเป็นสมการที่ใช้ในการหา θ_{rc} ใน Quadrant2 เพื่อใช้ในการหาค่า θ_1

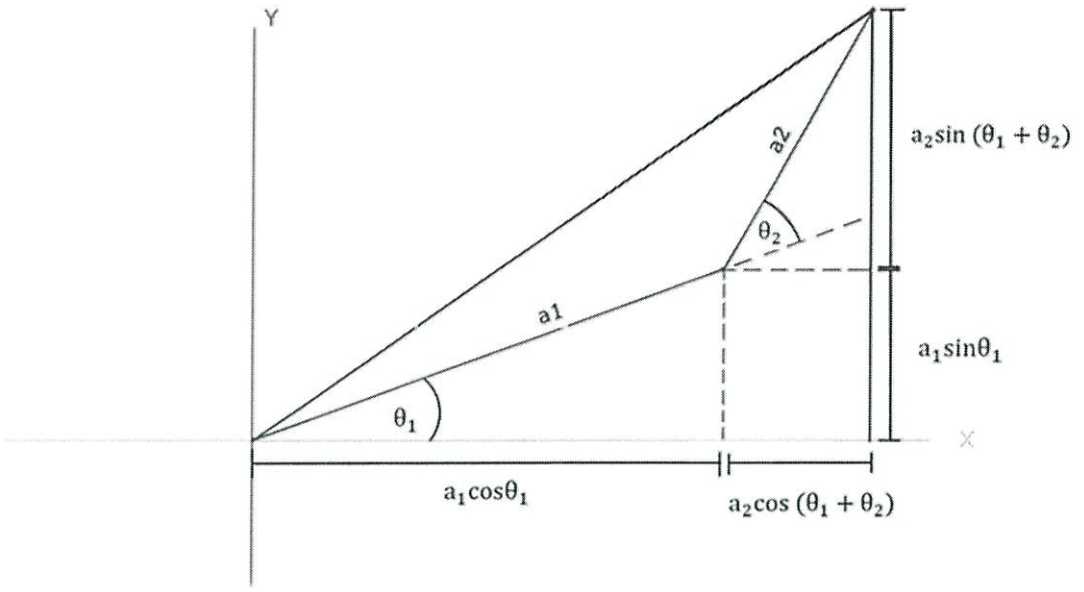
$$\theta_{rc} = \pi - \theta_j \quad (3.17)$$

$$a_2^2 = a_1^2 + R_c^2 - 2a_1R_c\cos\theta_3 \quad (3.18)$$

$$\theta_{rc} = \cos^{-1}\left(\frac{a_1^2+R_c^2-a_2^2}{2a_1R_c}\right) \quad (3.19)$$

$$\theta_1 = \theta_{rc} - \theta_3 \quad (3.13)$$

3.3.2 สมการ Forward Kinematics



รูปที่ 3.5 Forward Kinematics

จากรูปที่ 3.5 จะเป็นการแปลงระยะขจัด (R_c) ให้อยู่ในรูปของพิกัด ซึ่งจะได้สมการดังนี้

$$X_c = a_1\cos\theta_1 + a_2\cos(\theta_1 + \theta_2) \quad (3.21)$$

$$Y_c = a_1\sin\theta_1 + a_2\sin(\theta_1 + \theta_2) \quad (3.22)$$

3.4 ตัวควบคุมหุ่นยนต์ที่ศึกษาด้วยพีแอลซี

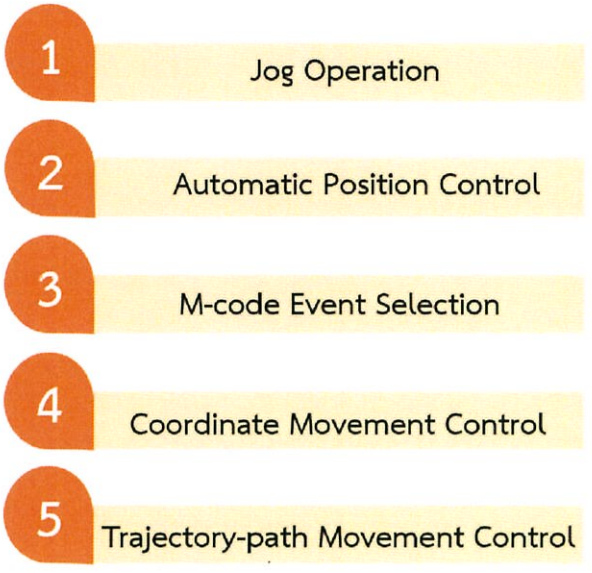
3.4.1 แนวคิดสำหรับการควบคุมหุ่นยนต์ที่ศึกษา

สำหรับการควบคุมหุ่นยนต์ด้วยพีแอลซีนั้นได้ถูกแบ่งโหมดการทำงานเป็น 5 รูปแบบ ซึ่งประกอบไปด้วยรูปแบบการทำงาน ดังแสดงในรูปที่ 3.6 โดยมีรายละเอียด ดังนี้

1. Jog Operation

การทำงานแบบ Jog Operation หมายถึง วิธีควบคุมด้วยมือ (Manual Control) นั่นคือ การควบคุมโดยไม่ใช้ Positioning Data (การระบุตำแหน่ง) ในการทำให้หุ่นยนต์เคลื่อนที่ ซึ่งโหมดการทำงานนี้มักใช้งานสำหรับการ Set up ความพร้อมของหุ่นยนต์ก่อนเริ่มทำงาน การใช้หาอัตราทดหรือ Reduction Gear Ratio หรือแม้กระทั่งใช้สำหรับกำหนดตำแหน่งอ้างอิงของการทำงานรูปแบบอื่น ๆ เช่น Automatic Position Control, M-code Event Selection,

Movement Control และ Trajectory-path Movement Control เนื่องจากการทำงานรูปแบบอื่น ๆ นั้นเป็นการเคลื่อนที่ที่อาศัยการระบุตำแหน่งดังนั้นจึงจำเป็นต้องมีตำแหน่งอ้างอิง ซึ่งโดยก่อนเริ่มทำงานผู้ใช้งานต้องมีการกำหนดตำแหน่งอ้างอิงโดยใช้โหมดการทำงานรูปแบบ Jog Operation



รูปที่ 3.6 การทำงานรูปแบบต่าง ๆ ของหุ่นยนต์

2. Automatic Position Control

การทำงานรูปแบบ Automatic Position Control หมายถึง การควบคุมการเคลื่อนที่ของหุ่นยนต์ด้วยการระบุตำแหน่งโดยใช้ข้อมูลตำแหน่งองศาเป็นตัวกำหนด ซึ่งข้อมูลการระบุตำแหน่งนี้จะเป็นส่วนนึงของ Positioning Data อีกทั้งรูปแบบการทำงานโหมดนี้จะเป็นพื้นฐานของการทำงานรูปแบบ M-code Event Selection, Coordinate Movement Control และ Trajectory-path Movement Control เนื่องจากโปรแกรมดังกล่าวนั้นต้องใช้ฟังก์ชันบล็อกของ Automatic Position Control ควบคู่ไปด้วย

3. M-code Event Selection

การทำงานรูปแบบ M-code Event Selection เป็นรูปแบบการทำงานเสริมของการทำงานแบบ Automatic Position Control, Coordinate Movement Control และ Trajectory-path Movement Control เพื่อเป็นการเพิ่มฟังก์ชันการทำงานให้หุ่นยนต์สามารถทำงานได้มากขึ้น โดยการทำงานรูปแบบ M-code Event Selection จะสามารถทำงานได้เมื่อมีการเปิดการทำงานของ M-code Event Selection ในการทำงานแบบใดแบบนึงของการทำงานที่กล่าวข้างต้น คือ ในระหว่างทำงานดังกล่าวมีการกำหนดให้ทำงานในรูปแบบ M-code Event Selection หุ่นยนต์จะเปลี่ยนมาทำงานในโหมด M-code Event Selection ทันที และจะทำงานจนกว่าจะทำงานเสร็จสิ้นแล้วจึงกลับไปทำงานรูปแบบเดิมต่อ โดยลักษณะการทำงานของ M-code Event Selection นั้นเป็นการ

แล้วจึงกลับไปทำงานรูปแบบเดิมต่อ โดยลักษณะการทำงานของ M-code Event Selection นั้นเป็นการกำหนด Event ที่จะให้หุ่นยนต์ทำงานว่าในแต่ละ Event จะให้หุ่นยนต์ทำงานอย่างไร เช่น Event No.1 คือให้หุ่นยนต์หนีบของ และ Event No.2 คือให้หุ่นยนต์ปล่อยของที่หนีบ เป็นต้น

4. Coordinate Movement Control

การทำงานรูปแบบ Coordinate Movement Control คือ การระบุการเคลื่อนที่ของหุ่นยนต์ด้วยพิกัด X และ Y แทนการระบุด้วยองศา (Automatic Position Control) และการกำหนดให้หุ่นยนต์ เคลื่อนที่ไปทางแกน X หรือแกน Y ทีละหน่วยตามที่กำหนด

5. Trajectory-path Movement Control

การทำงานรูปแบบ Trajectory-path Movement Control คือ การนำข้อมูล Trajectory Path ที่สร้างจากโปรแกรม RobotStudio เป็นตัวกำหนดการเคลื่อนที่ของหุ่นยนต์

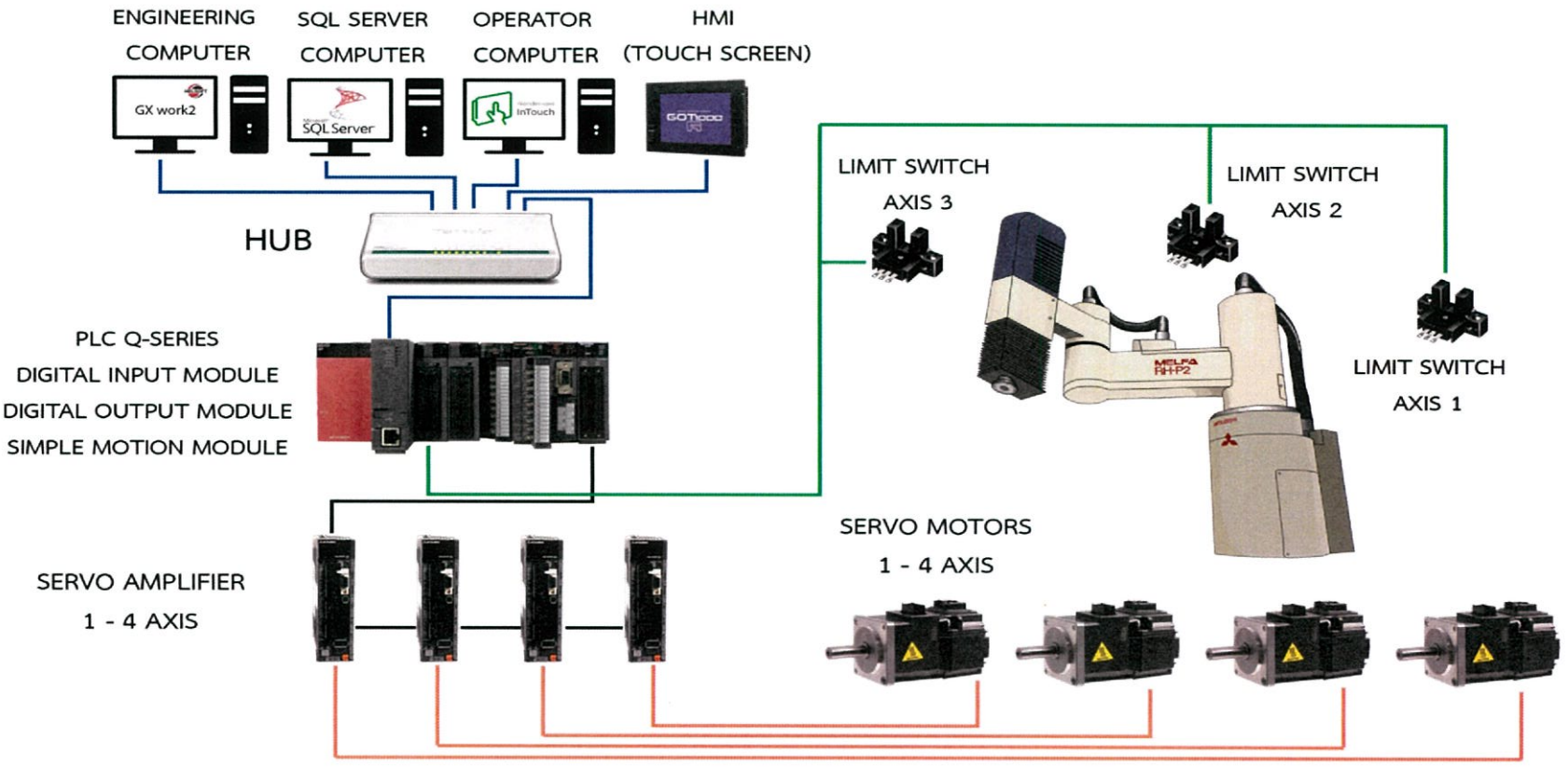
3.4.2 โครงสร้างทางฮาร์ดแวร์ของระบบควบคุม

โครงสร้างทางฮาร์ดแวร์ของระบบควบคุมนั้น แสดงดังรูปที่ 3.7 โดยโครงสร้างทางฮาร์ดแวร์ในภาพรวมที่ใช้ในระบบ ซึ่งประกอบได้ด้วย อุปกรณ์แต่ละส่วนที่ทำหน้าที่ ดังนี้

1) คอมพิวเตอร์ แบ่งเป็น 3 ส่วนคือ ส่วนที่เป็นโปรแกรมควบคุมประมวลผล, ส่วนที่เป็นระบบจัดการข้อมูล Trajectory Path และส่วนระบบฐานข้อมูล โดยที่ทั้งสามส่วนนี้สามารถเป็นคอมพิวเตอร์เครื่องเดียวกันหรือคนละเครื่องก็ได้

2) HMI ทำหน้าที่เป็นอินเตอร์เฟสให้กับผู้ใช้งาน เพื่อควบคุมการทำงานของหุ่นยนต์หน้างาน ทำหน้าที่เปรียบเสมือนอุปกรณ์ Programing Pendant

3) PLC ทำหน้าที่เป็นตัว Robot Controller ที่ทำการควบคุมการทำงานของหุ่นยนต์โดยสั่งการไปที่ Simple Motion Module หลังจากนั้น Simple Motion Module จะส่งสัญญาณผ่านสาย Fiber Optic ไปที่ Servo Amplifier ให้ Servo Amplifier ควบคุม Servo Motor ที่ต่ออยู่ที่ Joint ต่าง ๆ ของหุ่นยนต์เพื่อให้หุ่นยนต์เคลื่อนที่ตามคำสั่งจาก PLC ที่มีสมการ Kinematics ของหุ่นยนต์ให้สามารถควบคุมการเคลื่อนที่ให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่งต่าง ๆ ได้



รูปที่ 3.1 Overall System Architecture

สำหรับการสื่อสารระหว่าง PLC, Engineering Computer,ระบบจัดการข้อมูล Trajectory Path, ระบบฐานข้อมูล และ HMI นั้นจะสื่อสารผ่าน Ethernet โดยมี Hub เป็นตัวกลางในการสื่อสาร แต่การส่งสัญญาณทางดิจิตอลจาก Limit Switch และ Motor Brake ที่ติดตั้งที่ตัวหุ่นยนต์ จะส่งสัญญาณหรือรับคำสั่งต่าง ๆ จาก PLC โดยผ่านโมดูล Digital Input และ Digital Output เพื่อให้สามารถควบคุมหุ่นยนต์ให้มีประสิทธิภาพมากขึ้น

อุปกรณ์ที่ใช้สำหรับควบคุม SCARA Robot ด้วย PLC

1. PLC Q series slot Configuration
   - CPU Module: Q03UDECPU
   - Slot 0: QX42 (Digital Input Module)
   - Slot 1: QY42P (Digital Output Module)
   - Slot 2: Q64AD (Analog Input Module) Not Really Use
   - Slot 3: Q64DAN (Analog Output Module) Not Really Use
   - Slot 4: QJ1C24N (Communication Module) Not Really Use
   - Slot 5: QD77MS4 (Simple Motion Module)
2. Servo Amplifier [15]
   - MR J4-B Series จำนวน 4 ตัว
3. Servo Motor [16]
   - HG-KR43 จำนวน 2 ตัว
   - HG-KR053B จำนวน 1 ตัว
   - HG-KR053 จำนวน 1 ตัว
4. Graphic Terminal Operation [17]
   - GOT1000 GT14xx-Q Series จำนวน 1 ตัว
5. Limit Switch
   - PM-T53B จำนวน 6 ตัว
6. Motor Brake จำนวน 1 ตัว
7. Hub จำนวน 1 ตัว
8. Engineering PC
9. คอมพิวเตอร์สำหรับจัดการข้อมูล Trajectory Path
10. คอมพิวเตอร์สำหรับระบบฐานข้อมูล
11. หุ่นยนต์

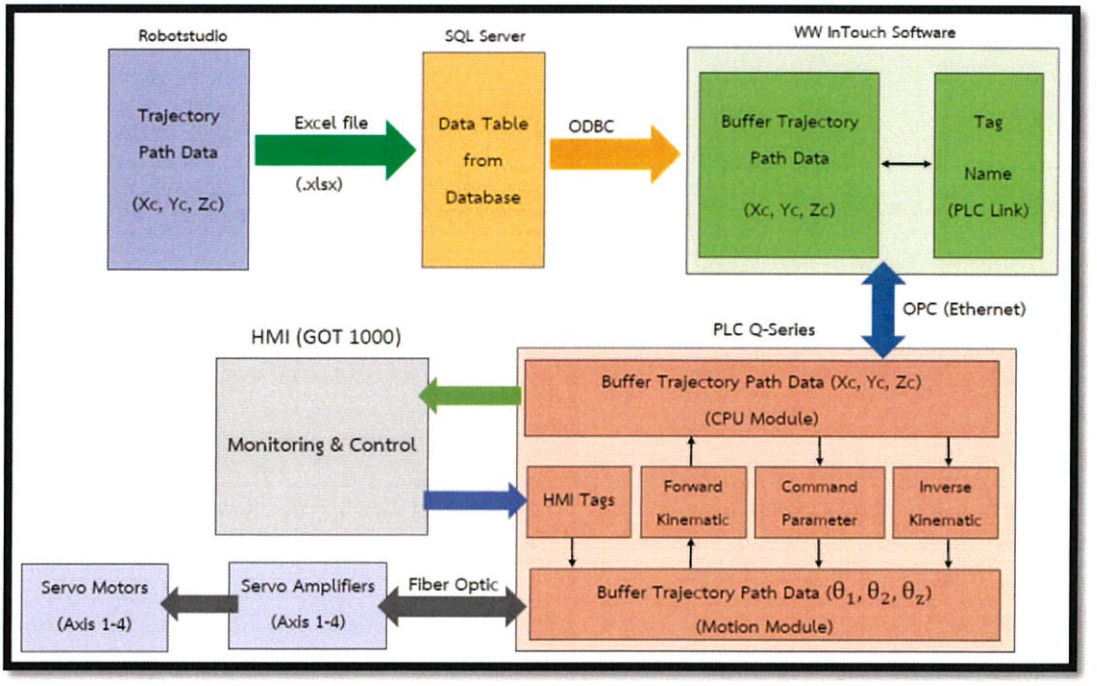
- MELFA RH-P2 ปี ค.ศ.1987

3.4.3 การสื่อสารข้อมูลระหว่างอุปกรณ์ในระบบควบคุม

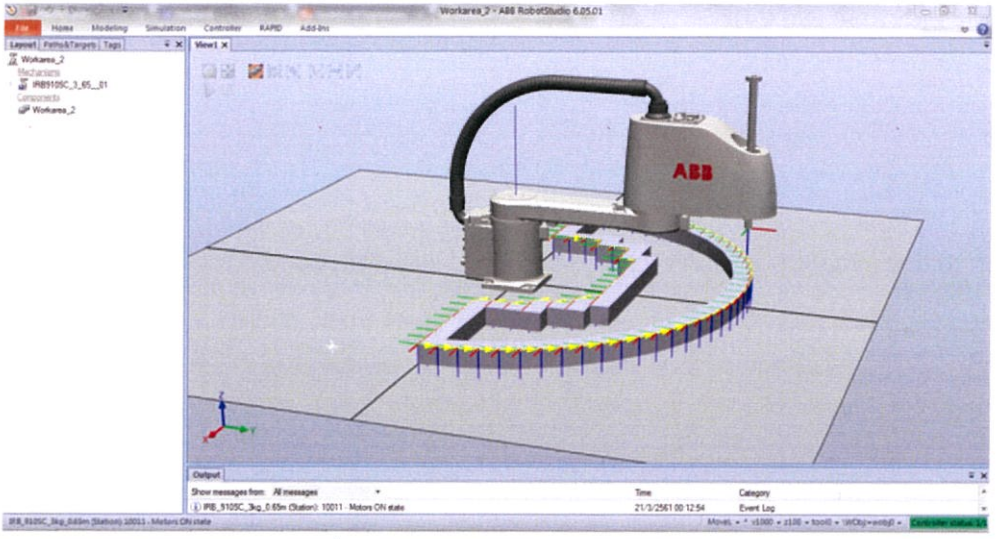
การสื่อสารข้อมูลระหว่างอุปกรณ์ในระบบควบคุม แสดงด้วยแผนภาพตามรูปที่ 3.8 ซึ่งจะแสดงถึงการสื่อสารข้อมูลระหว่างอุปกรณ์ในระบบควบคุม ซึ่งมีลักษณะ ดังนี้

1) สร้างวิถีการเคลื่อนที่ของหุ่นยนต์ หรือ Trajectory Path

การสร้างเส้นทางการเดินหรือ Trajectory Path ของหุ่นยนต์เป็นการออกแบบว่าจะหุ่นยนต์ทำงานรูปแบบใด แล้วในการทำงานนั้นหุ่นยนต์ต้องเคลื่อนที่ไปยังตำแหน่งใดของ Work Area บ้าง ซึ่งข้อมูลที่ได้จะออกมาในรูปแบบเส้นทางเดินแล้วนำเส้นทางเดินดังกล่าวมาสร้างเป็นจุดพิกัด X, Y, Z ที่เรียงต่อกันเป็นเส้นทางการเดิน แล้วส่งออกมาในรูปไฟล์ .xlsx โดยส่วนนี้จะใช้โปรแกรม RobotStudio รูปที่ 3.9



รูปที่ 3.8 Block Diagram of Data Transfer



รูปที่ 3.9 การสร้าง Trajectory Path

2) การเก็บข้อมูลในระบบฐานข้อมูล

เมื่อได้พิกัด X, Y, Z ที่ส่งมาในรูปไฟล์ .xlsx แล้วจะจัดเก็บข้อมูลดังกล่าวไว้ในระบบฐานข้อมูล เนื่องจากหน่วยความจำใน CPU ของ PLC ที่เป็นตัวสั่งการหุ่นยนต์นั้นมีน้อย จึงจำเป็นต้องมีการเก็บสำรองข้อมูล Trajectory Path ของหุ่นยนต์ไว้ในระบบฐานข้อมูล อีกทั้งการเก็บข้อมูลไว้ในระบบฐานจะช่วยให้ผู้ใช้งานสามารถเรียกใช้ข้อมูลดังกล่าวได้ทุกครั้งที่ต้องการใช้งานโดยไม่ต้องสร้างข้อมูลใหม่ โดยการเรียกใช้ข้อมูลนั้นจะเรียกใช้ผ่านระบบจัดการข้อมูล Trajectory Path ซึ่งโปรแกรมระบบฐานข้อมูลที่ได้ทำการเลือกใช้คือ Microsoft SQL server

3) การรับ-ส่งข้อมูลระหว่างระบบจัดการข้อมูล Trajectory Path กับระบบฐานข้อมูล

การรับส่งข้อมูลระหว่างระบบจัดการข้อมูล Trajectory Path กับระบบฐานข้อมูลนั้น จะมีโปรโตคอลที่เป็นตัวกลางการเชื่อมต่อนั้นคือ “ODBC” โดยเลือกใช้โปรแกรม ODBC Data Sources มาเป็นตัวกลางที่ทำหน้าที่ในการรับและส่งข้อมูล

4) ระบบจัดการข้อมูล Trajectory Path

หน้าที่ของระบบจัดการข้อมูล Trajectory Path คือ เชื่อมต่อกับระบบฐานข้อมูลเพื่อ Query ข้อมูลมาเก็บไว้ใน Buffer Memory ของระบบจัดการข้อมูล Trajectory Path และส่งข้อมูลพิกัดให้แก่ PLC เนื่องจาก PLC ไม่สามารถเชื่อมต่อกับระบบฐานข้อมูลได้โดยตรง ดังนั้นในการเรียกใช้ข้อมูลจากระบบฐานข้อมูลเพื่อส่งไปให้หุ่นยนต์ทำงานตามที่ต้องการ จึงจำเป็นต้องมีระบบจัดการข้อมูล Trajectory Path เป็นตัวกลางในการส่งข้อมูล โดยโปรแกรมระบบจัดการข้อมูล Trajectory Path ที่เลือกใช้คือ WW InTouch

5) การรับ-ส่งข้อมูลระหว่าง PLC กับระบบจัดการข้อมูล Trajectory Path

เช่นเดียวกับการรับ-ส่งข้อมูลระหว่างระบบจัดการข้อมูล Trajectory Path กับระบบฐานข้อมูล การรับส่งข้อมูลระหว่าง PLC กับระบบจัดการข้อมูล Trajectory Path นั้นก็จำเป็นต้องมีตัวกลางในการสื่อสารนั่นคือ DASMTEthernet ที่ทำหน้าที่เป็น OPC สำหรับการเชื่อมต่อสื่อสารระหว่าง PLC กับระบบจัดการข้อมูล Trajectory Path โดยโปรแกรมที่เลือกใช้คือ System Management Console ซึ่งเป็นโปรแกรมสำหรับ WW InTouch

6) โปรแกรม PLC เพื่อควบคุมการทำงานของหุ่นยนต์

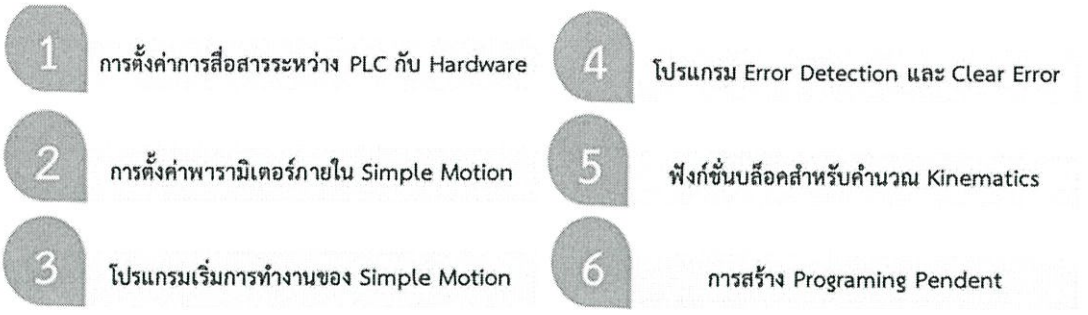
ข้อมูล Trajectory Path ที่ระบบจัดการข้อมูล Trajectory Path ทำการ Query ขึ้นมาจะถูกส่งต่อมาที่ PLC ที่มีโปรแกรมสำหรับควบคุมการทำงานของหุ่นยนต์ โดยเก็บข้อมูลไว้ที่ Buffer Memory ของ PLC จากนั้นจะคำนวณ Kinematics ของหุ่นยนต์เพื่อให้ได้ข้อมูลในรูปขององศา (θ) และทำการส่งข้อมูลไปให้ Buffer Memory ของ Simple Motion Module เพื่อไปควบคุมหุ่นยนต์ให้เคลื่อนที่ตาม Trajectory Path สำหรับการเขียนโปรแกรม PLC เพื่อควบคุมการทำงานของหุ่นยนต์นั้นได้ทำการเลือกใช้โปรแกรม GX Works2

3.5 การเขียนโปรแกรมสำหรับตัวควบคุมหุ่นยนต์ด้วยพีแอลซี

การเขียนโปรแกรมสำหรับโครงานตัวควบคุมหุ่นยนต์ด้วย PLC กรณีศึกษาของหุ่นยนต์ประเภท SCARA นั้นในที่นี้จะแบ่งเป็น 3 ส่วนใหญ่ ๆ คือ โปรแกรมพื้นฐานสำหรับควบคุม Servo Motor, โปรแกรมควบคุมการทำงานของหุ่นยนต์ และสุดท้ายคือโปรแกรมการถ่ายโอนข้อมูล Trajectory Path และการสร้าง Trajectory Path ซึ่งรายละเอียดทั้งสามส่วนสามารถแสดงได้ดังนี้

3.5.1 ภาพรวมของโปรแกรมในการควบคุมหุ่นยนต์ที่ศึกษา

ภาพรวมของโปรแกรมในการควบคุมหุ่นยนต์ที่ศึกษา หมายถึง การเขียนโปรแกรมพื้นฐานสำหรับควบคุม Servo Motor ที่ควบคุม Joint ต่าง ๆ ของหุ่นยนต์ที่อิสระต่อกันให้สามารถควบคุมการเคลื่อนที่ของหุ่นยนต์ได้และการตั้งค่าการใช้งานทั้ง PLC และโมดูลพิเศษในส่วน Simple Motion Module อีกทั้งยังรวมถึงการเขียนโปรแกรมคำนวณ Kinematics และการสร้างอุปกรณ์ที่ช่วยในการควบคุมการเคลื่อนที่ของหุ่นยนต์ที่เรียกว่า Programing Pendant จาก HMI อีกด้วย ซึ่งโปรแกรมพื้นฐานสำหรับควบคุม Servo Motor นั้นจะประกอบไปด้วย โปรแกรมที่แสดงดังรูปที่ 3.10



รูปที่ 3.10 องค์ประกอบโปรแกรมพื้นฐานสำหรับควบคุม Servo Motor

จากรูปที่ 3.10 แสดงถึงโปรแกรมที่จัดเป็นโปรแกรมพื้นฐานสำหรับการควบคุม Servo Motor ซึ่งมีองค์ประกอบ ดังนี้

1) การตั้งค่าการสื่อสารระหว่าง PLC กับโมดูลพิเศษ หมายถึง การตั้งค่าให้โมดูลพิเศษต่าง ๆ ที่นำมาเชื่อมต่อกับ PLC นั้นสามารถทำงานร่วมกับ Software หรือโปรแกรมที่เขียนภายใน PLC ได้

2) การตั้งค่าโมดูลพิเศษในส่วน Simple Motion นอกจากตั้งค่าการเชื่อมต่อกับโมดูลพิเศษแล้วต้องมีการตั้งค่าให้กับพารามิเตอร์ต่าง ๆ ภายในโมดูลพิเศษ Simple Motion เพื่อการควบคุมการทำงาน Servo motor ที่ติดที่ Joint ของหุ่นยนต์ให้สามารถทำงานตามโปรแกรมที่เขียนได้

3) โปรแกรมเริ่มการทำงานของ Simple Motion หมายถึง การเปิดการทำงานของ Servo หรือการ Start servo นั่นเอง นอกจากนี้โปรแกรมส่วนนี้ยังรวมไปถึงการปลดล็อค Motor brake ของแกนที่ 3 ของหุ่นยนต์เพื่อเป็นการเตรียมพร้อมสำหรับการควบคุมหุ่นยนต์

4) โปรแกรม Error Detection และ Clear Error เป็นโหมดป้องกัน เมื่อมีการตรวจจับ Error ได้ จะมีการแจ้งเตือน เพื่อที่จะให้สามารถหุ่นยนต์กลับมาทำงานได้อีกครั้งต้อง Clear Error ที่แจ้งเตือนก่อนเสมอ จึงจะสามารถควบคุมหุ่นยนต์ได้ใหม่ เพราะในขณะที่เกิด Error ผู้ใช้จะไม่สามารถควบคุมหุ่นยนต์ได้หากไม่มีการ Clear Error เสียก่อน

5) ฟังก์ชันบล็อคสำหรับคำนวณ Kinematics เนื่องจากตำแหน่งที่ได้จากระบบจัดการข้อมูล Trajectory Path อยู่ในรูปพิกัด X, Y, Z แต่การสั่งการมอเตอร์ให้หุ่นยนต์เคลื่อนที่นั้นสั่งด้วยตำแหน่งในรูปองศา (θ) ดังนั้นในโปรแกรมจึงจำเป็นต้องมีสมการคำนวณ Kinematics ของหุ่นยนต์เพื่อให้ได้พิกัดการเคลื่อนที่ที่อยู่ในรูปองศา

6) การสร้าง Programing Pendent สำหรับ Programing Pendent นี้เป็นอุปกรณ์ที่สำหรับทำให้การควบคุมหุ่นยนต์นั้นสะดวกขึ้น และโดยเฉพาะการปรับตั้งค่าต่าง ๆ ไม่ว่าจะช่วงการ

ทดสอบการทำงานของหุ่นยนต์หรือแม้กระทั่งการตั้งค่าเริ่มต้นหรือที่เรียกว่าจุด Home หรือจุดอ้างอิงของหุ่นยนต์

สำหรับโปรแกรมในส่วนของโปรแกรมพื้นฐานสำหรับควบคุม Servo Motor ทั้ง 6 ที่กล่าวข้างต้นนั้นจะถูกกล่าวอย่างละเอียดถึงการทำงานของโปรแกรมและการสร้างโปรแกรมหรือการเขียนโปรแกรมอีกทีในหัวข้อที่ 3.6-3.10 ตามลำดับ โดยโปรแกรมทั้งหมดจะเขียนด้วยโปรแกรม GX Works2 ยกเว้นการสร้าง Programing Pendent ที่จะใช้โปรแกรม GT Designer3

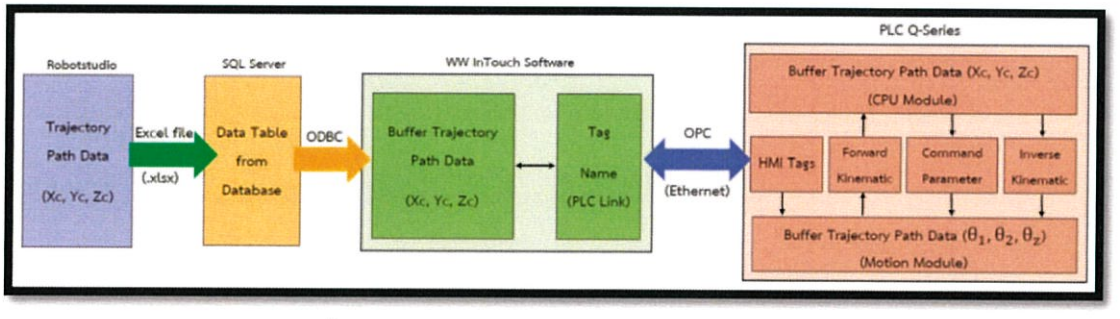
3.5.2 โปรแกรมควบคุมการทำงานของหุ่นยนต์

โปรแกรมส่วนนี้หมายถึงโปรแกรมสำหรับควบคุมการทำงานทั้ง 5 รูปแบบของหุ่นยนต์ที่ได้กล่าวไว้ในหัวข้อที่ 3.4.1 ซึ่งจะเขียนในรูปฟังก์ชันบล็อค โดยรายละเอียดการทำงานและการเขียนฟังก์ชันบล็อค ทั้ง 5 รูปแบบดังกล่าวจะกล่าวในหัวข้อที่ 3.11-3.15 ตามลำดับ และฟังก์ชันบล็อคทั้ง 5 นี้จะเขียนด้วยโปรแกรม GX Work2

3.5.3 การสร้าง Trajectory Path และโปรแกรมการถ่ายโอนข้อมูล Trajectory Path

โปรแกรมในส่วนนี้จะเป็นโปรแกรมในส่วนของการสร้าง Trajectory Path และการถ่ายโอนข้อมูล Trajectory Path ซึ่งเป็นข้อมูลที่การกำหนดการทำงานในโหมด Trajectory-path Movement Control โดยมีการแบ่งเป็นส่วน ๆ แสดงดังรูปที่ 3.11

จากรูปที่ 3.11 แสดงถึงการถ่ายโอนข้อมูล Trajectory Path ไปสู่ PLC ซึ่งนำไปสู่การเขียนโปรแกรม ดังนี้



รูปที่ 3.11 การถ่ายโอนข้อมูล Trajectory Path

1) การสร้าง Trajectory Path ด้วยโปรแกรม RobotStudio รายละเอียดโปรแกรมส่วนนี้จะกล่าวในหัวข้อที่ 3.16

2) การสร้างระบบฐานข้อมูลเพื่อจัดเก็บข้อมูล Trajectory Path ด้วยโปรแกรม Microsoft SQL Server ในหัวข้อที่ 3.17

3) การสร้างระบบจัดการข้อมูล Trajectory Path เชื่อมต่อกับฐานข้อมูลฐาน เพื่อทำการ Query ข้อมูล Trajectory Path ผ่าน ODBC แล้วส่งต่อข้อมูลดังกล่าวไปให้ PLC ผ่าน OPC โดยข้อมูลที่ส่งจะอยู่ในรูปพิกัด X, Y, Z เพื่อให้ PLC ประมวลผล และจัดการข้อมูลในรูปองศา เพื่อนำไปสั่งการควบคุมการทำงานของหุ่นยนต์ต่อไป การทำงานในส่วนนี้นั้นจำเป็นต้องใช้โปรแกรมหลายโปรแกรมมาก และมีรายละเอียดมากมายที่จะกล่าวในหัวข้อที่ 3.18

3.5.4 พารามิเตอร์ที่สำคัญในการควบคุม

พารามิเตอร์ที่สำคัญในการควบคุม แสดงดังตารางที่ 3.1-3.4

1). Input Parameter โดยรูปแบบการระบุ Device Address ของ Input Parameter สำหรับ PLC นั้นจะอ้างอิงตามการกำหนดของผู้ผลิต

ตารางที่ 3.1 Input Parameter

Item	Device Address	Data Type
Start PLC	XC0	Bit
Axis1 M-code ON	XC4	Bit
Axis2 M-code ON	XC5	Bit
Axis3 M-code ON	XC6	Bit
Axis4 M-code ON	XC7	Bit
Axis 1 Error detection	XC8	Bit
Axis 2 Error detection	XC9	Bit
Axis 3 Error detection	XCA	Bit
Axis 4 Error detection	XCB	Bit
Axis1 BUSY	XCC	Bit
Axis2 BUSY	XCD	Bit
Axis3 BUSY	XCE	Bit
Axis4 BUSY	XCF	Bit

2). Output Parameter โดยรูปแบบการระบุ Device Address ของ Output Parameter สำหรับ PLC นั้นจะอ้างอิงตามการกำหนดของผู้ผลิต

ตารางที่ 3.2 Output Parameter

Item	Device Address	Data Type
PLC Ready	YC0	Bit
All Axis servo ON	YC1	Bit
Axis1 Forward run JOG Start	YC8	Bit
Axis1 Reverse run JOG Start	YC9	Bit
Axis2 Forward run JOG Start	YCA	Bit
Axis2 Reverse run JOG Start	YCB	Bit
Axis3 Forward run JOG Start	YCC	Bit
Axis3 Reverse run JOG Start	YCD	Bit
Axis4 Forward run JOG Start	YCE	Bit
Axis4 Reverse run JOG Start	YCF	Bit
Axis1 Positioning Start	YD0	Bit
Axis2 Positioning Start	YD1	Bit
Axis3 Positioning Start	YD2	Bit
Axis4 Positioning Start	YD3	Bit

3). Motion Parameter โดยรูปแบบการระบุ Device Address ของ Motion Parameter สำหรับ PLC นั้นจะอ้างอิงตามการกำหนดของผู้ผลิตและรุ่นของ Simple Motion Module ที่ใช้

ตารางที่ 3.3 Motion Parameter

Item	Buffer Memory Address	Data type
Axis1 Hardware Stroke Limit Upper	U0C\G1928.0	Bit
Axis1 Hardware Stroke Limit Lower	U0C\G1928.1	Bit
Axis2 Hardware Stroke Limit Upper	U0C\G1928.4	Bit
Axis2 Hardware Stroke Limit Lower	U0C\G1928.5	Bit
Axis3 Hardware Stroke Limit Upper	U0C\G1928.8	Bit
Axis3 Hardware Stroke Limit Lower	U0C\G1928.9	Bit
Axis4 Hardware Stroke Limit Upper	U0C\G1928.C	Bit

ตารางที่ 3.3 (ต่อ)

Item	Buffer Memory Address	Data type
Axis4 Hardware Stroke Limit Lower	U0C\G1928.D	Bit
Axis1 error reset	U0C\G1502	Word
Axis2 error reset	U0C\G1602	Word
Axis3 error reset	U0C\G1702	Word
Axis4 error reset	U0C\G1802	Word
Axis1 Positioning start No.	U0C\G1500	Word
Axis2 Positioning start No.	U0C\G1600	Word
Axis3 Positioning start No.	U0C\G1700	Word
Axis4 Positioning start No.	U0C\G1800	Word
Axis1 M-code OFF request	U0C\G1504	Word
Axis2 M-code OFF request	U0C\G1504	Word
Axis3 M-code OFF request	U0C\G1504	Word
Axis4 M-code OFF request	U0C\G1504	Word
Axis1 Valid M-code	U0C\G808	Word
Axis2 Valid M-code	U0C\G908	Word
Axis3 Valid M-code	U0C\G1008	Word
Axis4 Valid M-code	U0C\G1108	Word
Axis1 JOG Speed	U0C\G1518	Double Word
	U0C\G1519	
Axis2 JOG Speed	U0C\G1618	Double Word
	U0C\G1619	
Axis3 JOG Speed	U0C\G1718	Double Word
	U0C\G1719	
Axis4 JOG Speed	U0C\G1818	Double Word
	U0C\G1819	
Axis1 Machine feed value	U0C\G802	Double Word
	U0C\G803	
Axis2 Machine feed value	U0C\G902	Double Word
	U0C\G903	

ตารางที่ 3.3 (ต่อ)

Item	Buffer Memory Address	Data type
Axis3 Machine feed value	U0C\G1002	Double Word
	U0C\G1003	
Axis4 Machine feed value	U0C\G1102	Double Word
	U0C\G1103	

4). Positioning Data Parameter โดยรูปแบบการระบุ Device Address ของ Positioning Data Parameter สำหรับ PLC นั้นจะอ้างอิงตามการกำหนดของผู้ผลิตและรุ่นของ Simple Motion Module ที่ใช้

ตารางที่ 3.4 Positioning Data Parameter

Item	Buffer Memory Address
Operation Pattern	U0C\G2000
Control Method	
Acceleration time No.	
Deceleration time No.	
Axis to be interpolated	
M-code	U0C\G2001
Dwell time	U0C\G2002
Command speed	U0C\G2004
	U0C\G2005
Positioning Address	U0C\G2006
	U0C\G2007
Arc Address	U0C\G2008
	U0C\G2009

จากตารางที่ 3.4 เป็นเพียงแค่ตัวอย่างในการอธิบายถึงความสัมพันธ์ของตัวแปร Positioning Data Parameter ของแกนที่ 1 Position No. 1 เท่านั้น แต่สำหรับพารามิเตอร์ Position อื่น แกนอื่น ๆ นั้นสามารถคำนวณ Address ได้จากสมการดังนี้

กำหนดให้ Command คือ Address ของ Operation Pattern, Control Method, Acceleration time No., Deceleration time No. และ Axis to be interpolated

Speed คือ Address ของ Command speed

Address คือ Address ของ Positioning Address

M-code คือ Address ของ M-code

Dwell time คือ Address ของ Dwell time

Arc Address คือ Address ของ Arc Address

$$Command = \{[(Axis-1) \times 6000] + 2000\} + [(Position\_No.-1) \times 10] \quad (3.23)$$

$$Speed = Command + 4 \quad (3.24)$$

$$Address = Command + 6 \quad (3.25)$$

$$M\text{-}code = Command + 1 \quad (3.26)$$

$$Dwell\ time = Command + 2 \quad (3.27)$$

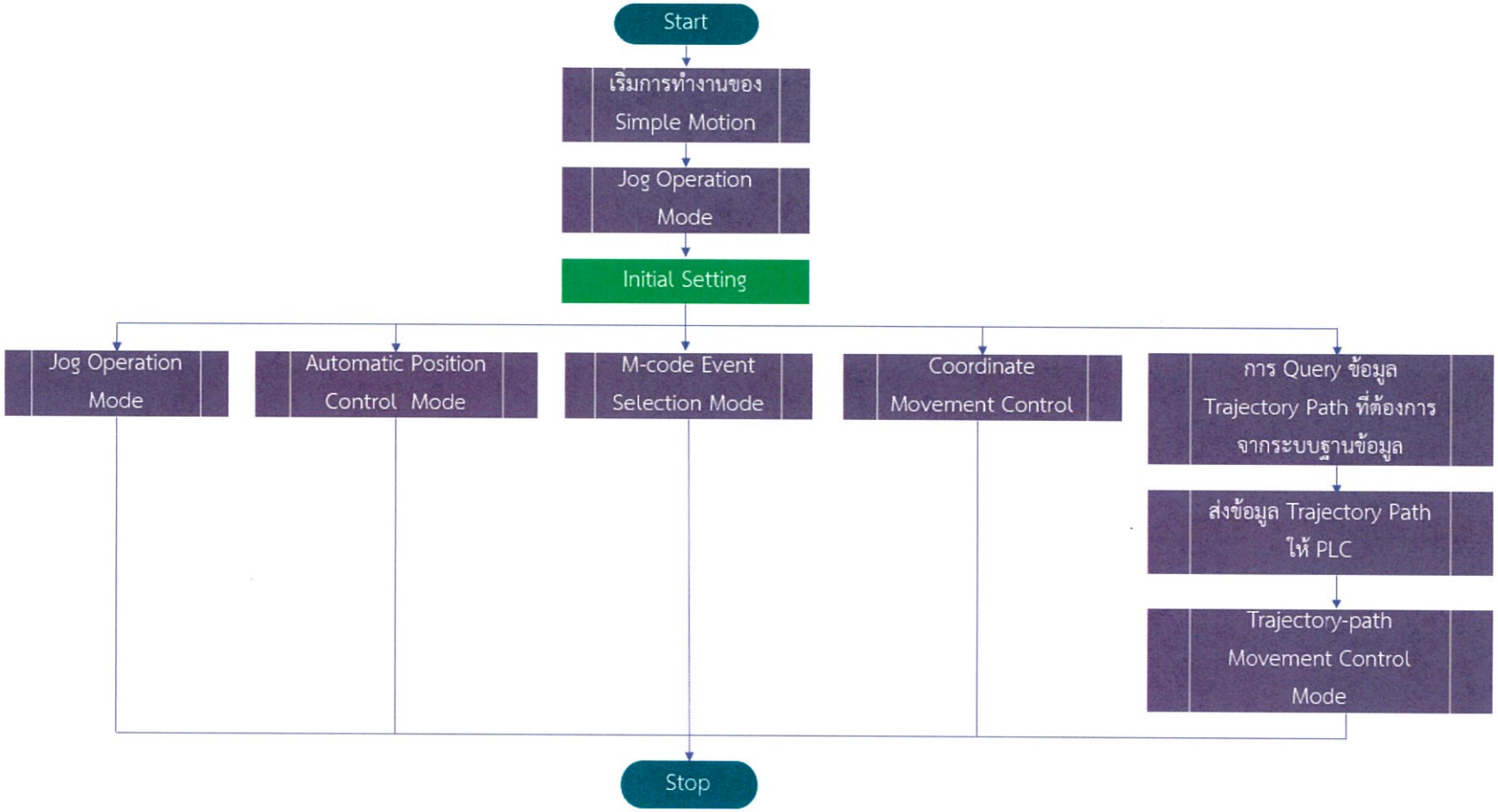
$$Arc\ Address = Command + 8 \quad (3.28)$$

3.5.5 ภาพรวมของลำดับการใช้งานหุ่นยนต์

ภาพรวมของลำดับการใช้งานหุ่นยนต์ หมายถึง ขั้นตอนหรือกระบวนที่ผู้ใช้งานต้องทำสำหรับการใช้งานหรือควบคุมหุ่นยนต์ โดยมีลำดับขั้นตอนแสดงดังรูปที่ 3.12

จากรูปที่ 3.12 แสดงถึงแผนผังการขั้นตอนการทำงานของการใช้งานของหุ่นยนต์โดยมีขั้นตอน ดังนี้

ขั้นแรก ผู้ใช้งานจะต้องเริ่มการทำงานของ Simple Motion โดยการกดปุ่ม Start ที่หน้าจอ HMI เพื่อเรียกให้งานโปรแกรมเริ่มการทำงานของ Simple Motion เป็นการเปิดการทำงานของ Servo Motor ทุกตัว จากนั้น ผู้ใช้งานจะต้องเรียกใช้งานโหมด Jog Operation เพื่อเคลื่อนแขนหุ่นยนต์แต่ละแกนไปยังตำแหน่งที่จะเป็นตำแหน่งอ้างอิงหรือตำแหน่งที่ต้องการให้เป็นตำแหน่ง 0 องศา แล้วทำการ Setting Initial เมื่อทำการเปิดการทำงาน Servo Motor ทุกตัวและทำการตั้งค่าตำแหน่งอ้างอิงแล้วผู้ใช้งานสามารถเรียกใช้งานโหมดต่าง ๆ ได้ ไม่ว่าจะเป็น Jog Operation, Automatic Position Control, M-code Event Selection, Coordinate Movement Control หรือ Trajectory-path Movement Control แต่หากผู้ใช้งานต้องการใช้งานโหมด Trajectory-path Movement Control ผู้ใช้งานต้องมีการเลือก Path ที่ต้องการทำงานก่อนโดยการเรียกใช้โปรแกรมการ Query ข้อมูล Trajectory Path ที่ต้องการจากระบบฐานข้อมูล และโปรแกรมส่งข้อมูล Trajectory Path ให้ PLC ก่อน จากนั้นจึงทำการเรียกใช้งานโปรแกรม Trajectory-path Movement Control



รูปที่ 3.2 ลำดับการทำงานของโปรแกรมหลัก

3.6 การตั้งค่าการสื่อสารระหว่าง PLC กับโมดูลพิเศษ ด้วย GX Works2

โมดูลพิเศษ หมายถึง อุปกรณ์หรือโมดูลต่าง ๆ ที่ไม่ได้มาพร้อม PLC แต่นำมาเสริมเพื่อเพิ่มความสามารถให้แก่ PLC ให้ทำงานได้หลากหลายขึ้น ดังนั้น เมื่อมีการนำอุปกรณ์ดังกล่าวมาต่อเพิ่มต้องมีการตั้งค่าทาง Software ให้สามารถเชื่อมต่อกันได้ โดยหมายถึงการตั้งค่า I/O Assignment ที่จะเป็นตัวกำหนด Address ของแต่ละโมดูลเพื่อเขียนโปรแกรมควบคุมการทำงาน

I/O Assignment(*1)

No.	Slot	Type	Model Name	Points	Start XY
0	PLC	PLC			
1	0(*-0)	Input	QX42	64Points	0000
2	1(*-1)	Output	QY42P	64Points	0040
3	2(*-2)	Intelligent	Q64AD	16Points	0080
4	3(*-3)	Intelligent	Q64DAN	16Points	0090
5	4(*-4)	Intelligent	QJ71C24N	32Points	00A0
6	5(*-5)	Intelligent	QD77MS4	32Points	00C0
7	6(*-6)				

รูปที่ 3.13 การสร้าง I/O Assignment

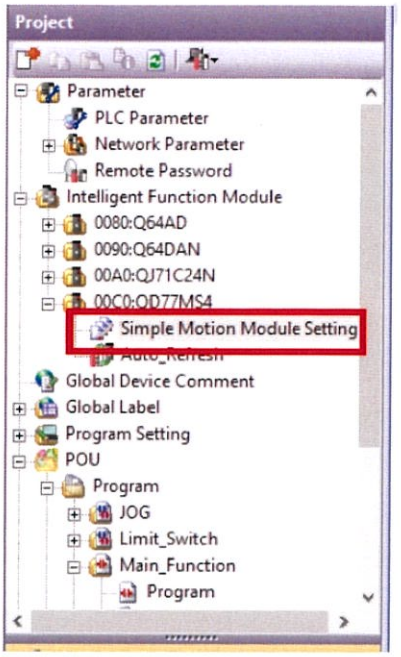
จากรูปที่ 3.13 เป็นตัวอย่างการตั้งค่า I/O Assignment ของอุปกรณ์ที่ใช้เชื่อมต่อกับ PLC โดยจำเป็นต้องเรียงโมดูลตาม Slot ที่จัดเรียงตามฮาร์ดแวร์จริง ซึ่งจะเห็นว่าโมดูล Digital Input นั้นอยู่ Slot 0 Address จะเริ่มต้นที่ X000 ส่วนโมดูล Digital Output นั้นจะอยู่ที่ Slot 1 Address เริ่มต้นที่ Y040 และสุดท้ายโมดูล Simple Motion อยู่ Slot 5 Address เริ่มต้นที่ X0C0 และ Y0C0 เป็นต้น

3.7 การตั้งค่าโมดูลพิเศษในส่วน Simple Motion

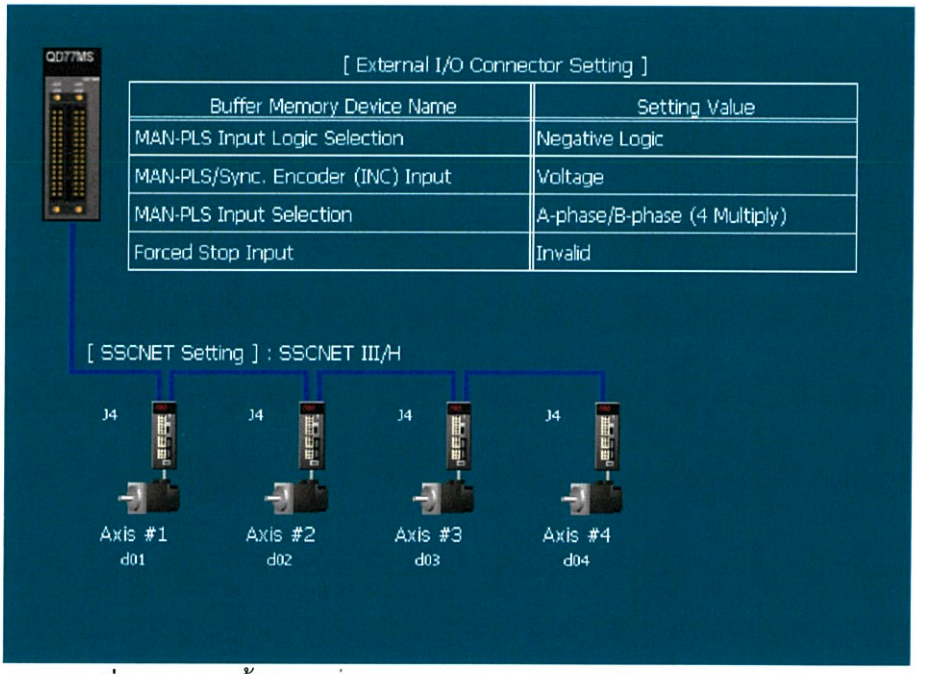
ในส่วนนี้จะอธิบายเกี่ยวกับการตั้งค่า Configuration ของโมดูล Simple Motion รุ่น QD77MS4 ซึ่งเป็นส่วนสำคัญที่จะกำหนดรูปแบบการทำงานของ Servo Amplifier เพื่อให้ทำงานร่วมกับโปรแกรมที่ออกแบบให้ไปควบคุมหุ่นยนต์ โดยการตั้งค่า Configuration นี้จะอยู่ในโปรแกรม Simple Motion Module Setting รูปที่ 3.14

3.7.1 เลือก Servo Amplifier ตามที่ใช้ในแต่ละแกน

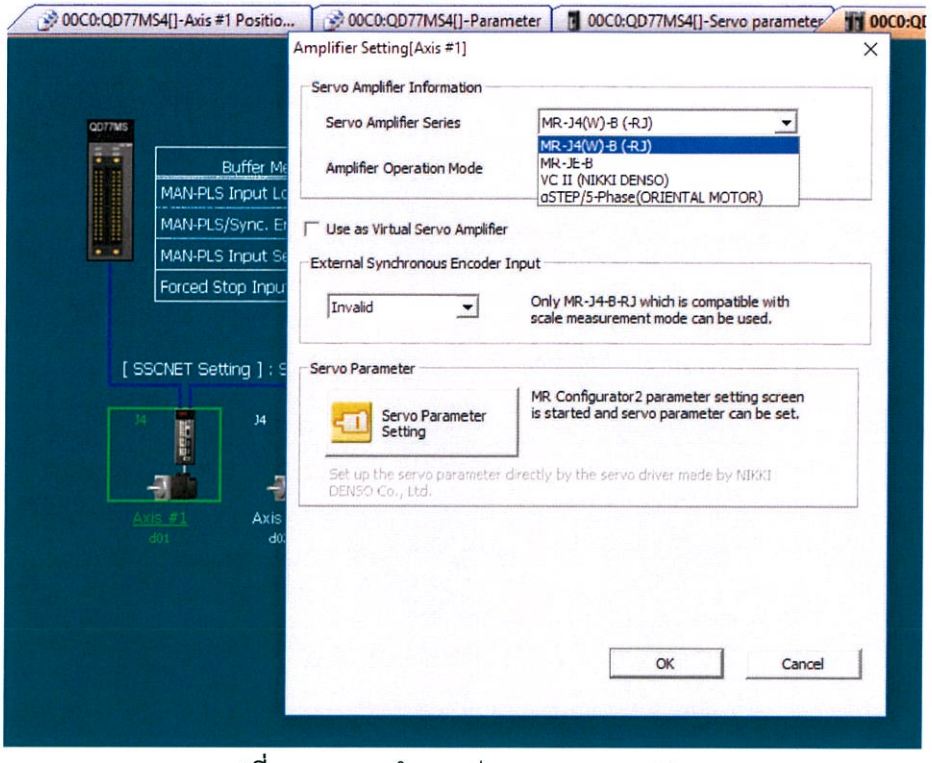
จากรูปที่ 3.14 เป็นการกำหนดจำนวนโมดูล Servo Amplifier ที่ใช้งานและรุ่นของ Servo Amplifier ที่ใช้งาน โดย Servo Amplifier ที่ใช้นี้สามารถกำหนดให้มี Servo Amplifier ได้มากสุด 4 ตัวต่อ Simple Motion 1 โมดูล และหุ่นยนต์ประเภท SCARA นั้นประกอบไปด้วย 4 แกน ดังนั้น ต้องกำหนดให้มี Servo Amplifier 4 ตัว และในปริญญานิพนธ์นี้ใช้ Servo Amplifier รุ่น MR-J4-B สำหรับการกำหนดจำนวนโมดูลและรุ่นที่ใช้แสดงดังรูปที่ 3.15-3.16



รูปที่ 3.14 การเข้าโปรแกรม Simple Motion Module Setting



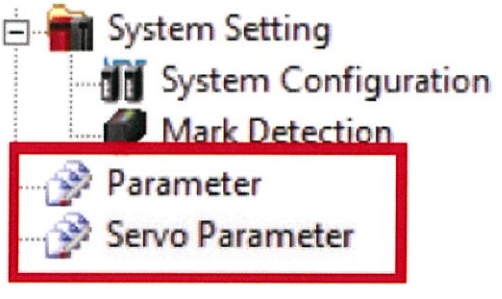
รูปที่ 3.15 การตั้งค่า Simple Motion Module กับ Servo Amplifier



รูปที่ 3.16 การกำหนดรุ่น Servo Amplifier

3.7.2 การตั้งค่า Parameter ในหมวดต่าง ๆ

Parameter ของปริญญานิพนธ์นี้จะมีอยู่ 2 ส่วนหลักคือ Parameter และ Servo Parameter ดังรูปที่รูปที่ 3.17

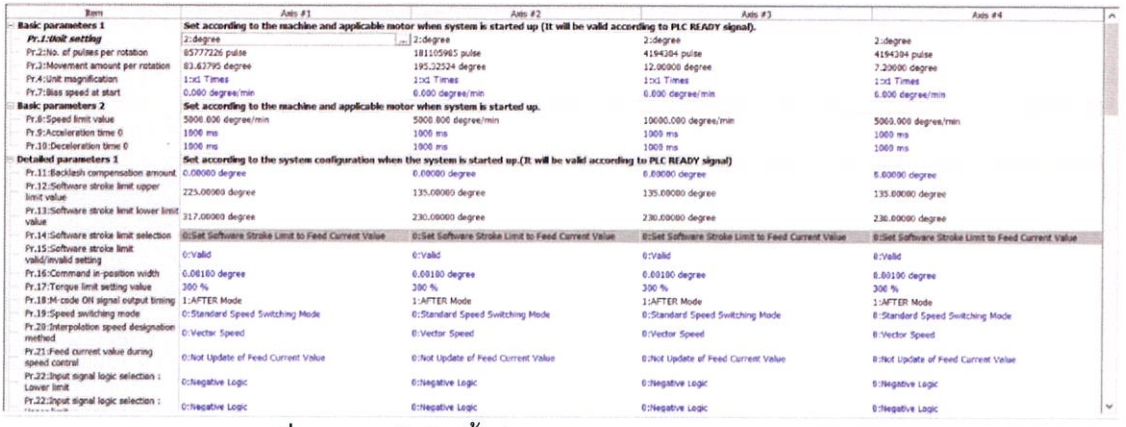


รูปที่ 3.17 Servo Amplifier Parameter และ Servo Parameter

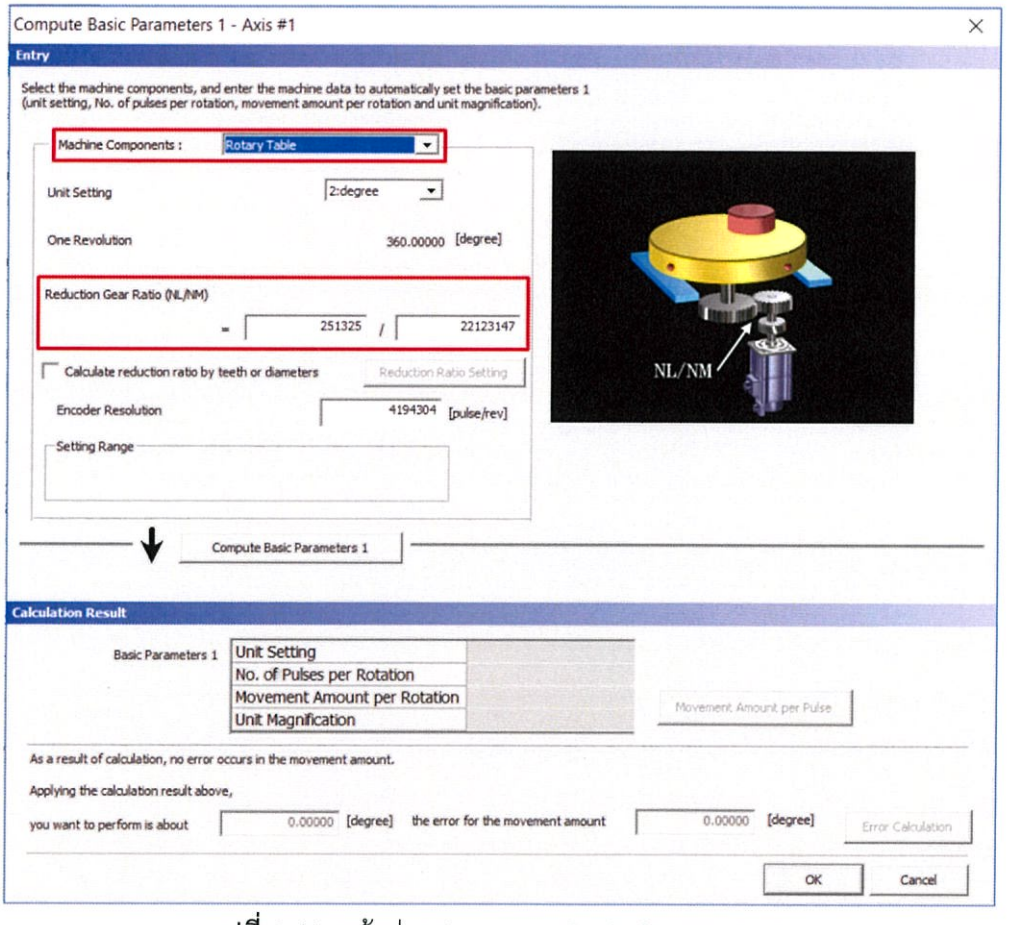
Parameter คือค่าการตั้งค่าสำหรับ Simple Motion Module เมื่อเข้าไปในส่วนนี้จะพบกับการตั้งค่าส่วนย่อยอีกมากมาย ซึ่งการตั้งค่าที่จำเป็นต้องใช้ได้แก่

3.7.2.1 Basic Parameter

เป็นค่าพารามิเตอร์ที่เกี่ยวข้องกับ Servo Motor ที่เราใช้งาน แสดงดังรูปที่ 3.18 ซึ่งเกี่ยวกับหน่วย ลักษณะการเคลื่อนที่ และความเร็วการเคลื่อนที่ โดยมีค่าพารามิเตอร์สำคัญที่ควรทราบในปริญญานิพนธ์นี้ได้แก่



รูปที่ 3.18 หน้าต่างตั้งค่า Servo Amplifier Parameter



รูปที่ 3.19 หน้าต่าง Compute Basic Parameter1

1) Machine Components หมายถึง การกำหนดรูปแบบการเคลื่อนที่ของ Servo Motor ให้สอดคล้องกับหุ่นยนต์ที่ใช้ในงาน ในปริญญานิพนธ์นี้เลือกใช้ Rotary Table หน่วยที่ถูกใช้ในการคำนวณการเคลื่อนที่คือ Degree หรือ หน่วยองศา รูปที่ 3.19

2) Reduction Gear Ratio หรือ อัตราทด หมายถึง อัตราส่วนของอุปกรณ์ตัวรับกำลัง (NL) ต่อ อัตราส่วนของอุปกรณ์ตัวส่งกำลัง (NM) รูปที่ 3.19

3) Speed limit value หมายถึง การกำหนดความเร็วสูงสุดของการเคลื่อนของ Servo Motor

3.7.2.2 Detailed Parameter

Detailed Parameter เป็นค่าพารามิเตอร์ที่เกี่ยวข้องกับการจำกัดการทำงานต่าง ๆ เพื่อความปลอดภัยในการทำงานของ Servo Motor โดยมีค่าพารามิเตอร์สำคัญที่ควรทราบในปริญญานิพนธ์นี้ได้แก่

1) Software stoke limit upper limit value คือค่าจำกัดสูงสุดที่ Servo Motor จะเคลื่อนที่ไปถึง

2) Software stoke limit lower limit value คือค่าจำกัดต่ำสุดที่ Servo Motor จะเคลื่อนที่ไปถึง

3) M-code ON Signal output timing คือคำสั่งที่กำหนดการใช้งานของ Event ซึ่งมี 2 โหมด ได้แก่ WITH MODE คือ Event จะทำงานขณะที่ Servo Motor กำลังดำเนินการทำงานใน Position นั้น ๆ อยู่ และ AFTER MODE คือ Event จะทำงานหลังจากที่ Servo Motor ดำเนินงานใน Position นั้นเสร็จสิ้นแล้ว

4) External signal selection คือ กำหนดการรับค่าสัญญาณต่าง ๆ มี 3 โหมด ได้แก่ External Input Signal of QD77MS, External Input Signal of Servo Amplifier และ Use Buffer Memory of QD77MS

5) JOG Speed limit value คือ การกำหนดค่าความเร็วสูงสุดของ Servo Motor ในโหมดของ JOG

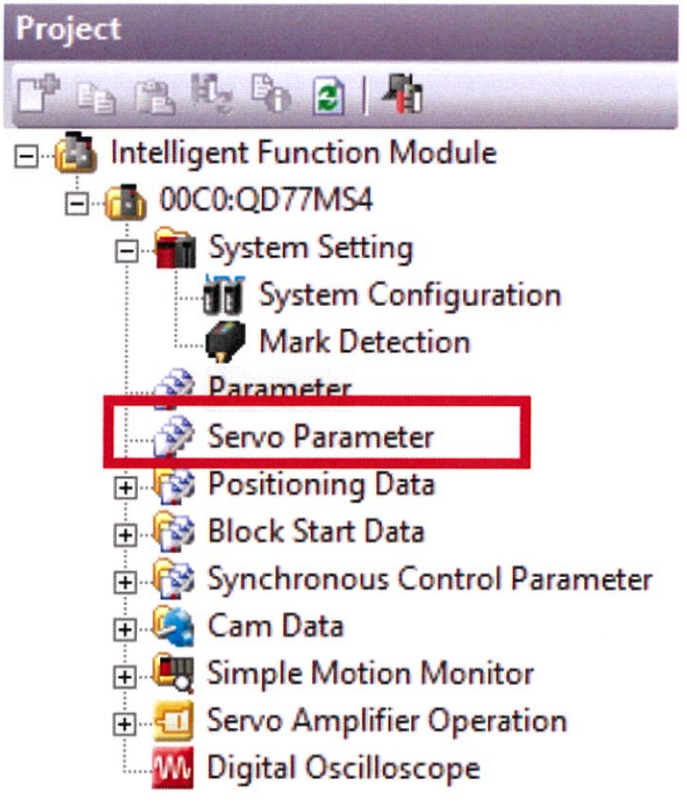
3.7.2.3 HPR basic Parameter หรือ HPR detailed parameter

HPR basic Parameter หรือ HPR detailed parameter คือ พารามิเตอร์ที่เกี่ยวกับการกลับสู่ตำแหน่งเริ่มต้นของการทำ Position Mode โดยมีค่าพารามิเตอร์สำคัญที่ใช้ในปริญญานิพนธ์นี้ได้แก่ HPR Method คือคำสั่งที่เราจะกำหนดว่า Servo Motor ที่ทำงานตามคำสั่งแบบไหน โดยมี Proximity Dog Method คือการมีอุปกรณ์ตรวจจับ Proximity คอยตรวจจับและสั่งการเคลื่อนที่ให้กับ Servo Motor และ Data Set Method คือการเคลื่อนที่ของ Servo Motor โดย

ไม่มีอุปกรณ์ตรวจจับ Proximity แต่จะทำการเคลื่อนที่ของ Servo Motor ผ่านการทำงานโหมด JOG Operation

3.7.2.4 Servo Parameter

Servo Parameter คือ Parameter บนตัว Servo Amplifier สำหรับลูปควบคุม Servo Motor ซึ่งในปริญญานิพนธ์นี้จะกล่าวแค่บางส่วนที่ใช้งานคือ Forced stop input และการแก้ไขทิศทางการหมุนของ Servo Motor มีพารามิเตอร์ที่ใช้งาน แสดงดังรูปที่ 3.20-3.21



รูปที่ 3.20 การตั้งค่า Servo Parameter

1) PARAMETER PA04 คือ พารามิเตอร์ที่เกี่ยวกับ Forced stop ว่าจะทำการปิด หรือ เปิด

2) PARAMETER PA14 คือ พารามิเตอร์ที่เลือกทิศทางการหมุนของ Servo Motor ค่า 0 คือ หมุนทวนเข็มนาฬิกา CCW (Forward) และค่า 1 คือหมุนตามเข็มนาฬิกา CW (Reverse) [12]

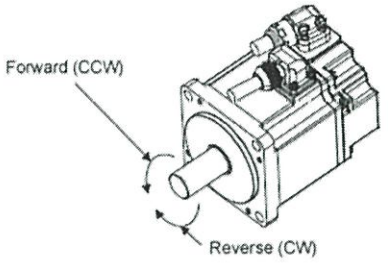
PARAMETER PA14

(MR-J4-B)

*** Rotation direction selection/Moving direction selection:**
This is used to select a rotation direction or moving direction.

Setting value	Servo motor rotation direction/linear servo motor moving direction	
	Positioning address increase	Positioning address decrease
0	CCW or positive direction	CW or negative direction
1	CW or negative direction	CCW or positive direction

The following shows the servo motor rotation directions.

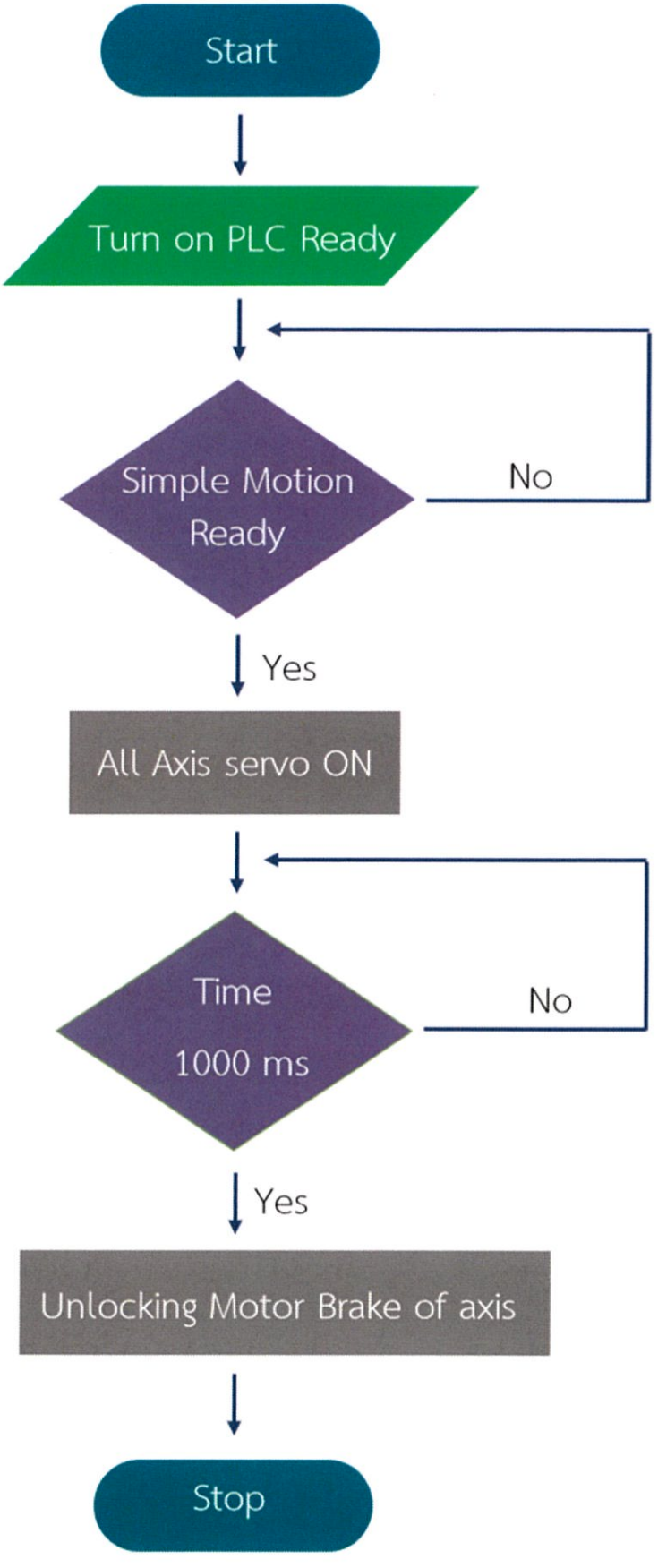


รูปที่ 3.21 Servo Motor Rotation

3.8 การเขียนโปรแกรมเริ่มการทำงานของ Simple Motion ด้วย GX Works2

โปรแกรมเริ่มการทำงานของ Simple Motion จัดเป็นโปรแกรมที่สำคัญมากสำหรับการเขียนโปรแกรมควบคุมการทำงานของหุ่นยนต์ เนื่องจากเป็นขั้นตอนแรกสำหรับการเริ่มการควบคุมหุ่นยนต์ เพราะว่าโปรแกรมเริ่มทำงานของ Servo Motor จะมีการตรวจความพร้อมของ Simple Motion Module และ PLC ว่าพร้อมหรือไม่ จากนั้นจึงสั่งเริ่มการทำงานของ Servo และทำการควบคุมหุ่นยนต์ในรูปแบบการทำงานต่าง ๆ โดยมีขั้นตอนการทำงาน ที่แสดงดังแผนผังการดำเนินงาน ดังรูปที่ 3.22

จากรูปที่ 3.22 แสดงถึงขั้นตอนในการเริ่มต้นการทำงานของหุ่นยนต์ โดยเริ่มจากการรับคำสั่งจากภายนอกให้เปิดการทำงานของพารามิเตอร์ PLC Ready จากนั้นจะตรวจสอบความพร้อมในการทำงานของ Simple Motion Module หากโมดูลไม่พร้อมก็จะทำการตรวจสอบไปเรื่อย ๆ โดยความพร้อมของ Simple Motion Module จะเป็นการตรวจสอบภายในของโมดูลเอง เมื่อคำสั่งเปิดการทำงานยังคงอยู่ แต่เมื่อ Simple Motion มีความพร้อมเรียบร้อยแล้ว จะทำการสั่งเปิดการทำงาน Servo Motor ทุกตัว เมื่อจบกระบวนการนั้นจะมีการตรวจสอบเวลาว่าครบ 1000 ms หรือยัง หากยังไม่ครบก็จะรอไปเรื่อย ๆ จนกว่าจะครบ และเมื่อครบ 1000 ms แล้วจะสั่งปลด Motor Brake ของแกนที่ 3 ออก เป็นการจบกระบวนการทำงาน



รูปที่ 3.22 แผนผังการเริ่มการทำงานของ Servo Amplifier

3.9 ฟังก์ชันบล็อคสมการการคำนวณ Kinematics ของหุ่นยนต์ด้วย GX Works2

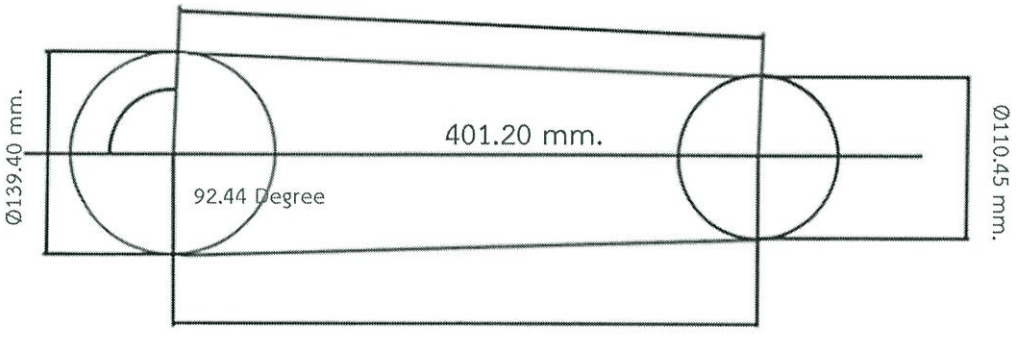
3.9.1 การทดลองการหาความยาว Link ของหุ่นยนต์

เนื่องจากหุ่นยนต์ที่ทำการศึกษานั้นเป็นหุ่นยนต์เก่าที่ถูกยกเลิกการผลิตมานานแล้ว จึงไม่มีข้อมูลในส่วนรายละเอียดคุณสมบัติของหุ่นยนต์ เช่น ความยาวของ Link1 และ Link2 ซึ่งเป็นพารามิเตอร์ที่สำคัญที่ใช้ในการคำนวณ Kinematics ของหุ่นยนต์เพื่อใช้ในการควบคุมการทำงานแบบ Coordinate Movement Control ดังนั้นจึงจำเป็นต้องใช้หลักทางคณิตศาสตร์เพื่อช่วยในการหาความยาว Link1 และ Link2 ดังนี้

1) การหาความยาว Link1

การหาความยาว Link1 นั้นสามารถทำได้โดยการวัดความยาวรอบ Joint1, Joint2 และ ความยาวรอบ Joint1 และ Joint2 จากนั้นนำความยาวที่วัดได้วาดรูปลงในโปรแกรม Solidwork แล้วทำการหาเส้นความยาวของ Link1 ผลการทดลองแสดงดังรูปที่ 3.23

- ความยาวรอบ Joint1 เท่ากับ 346 mm.
- ความยาวรอบ Joint2 เท่ากับ 438 mm.
- ความยาวรอบ Joint1 และ Joint2 เท่ากับ 1198 mm.



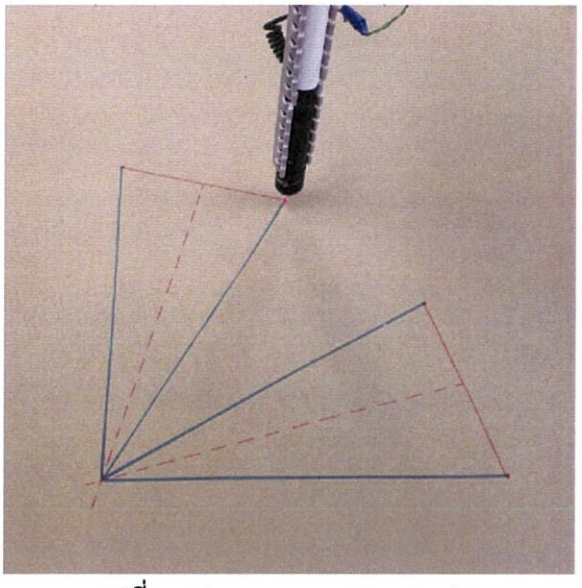
รูปที่ 3.23 ผลการหาความยาว Link1

จากรูปที่ 3.23 แสดงถึงผลการให้โปรแกรม SolidWork เพื่อช่วยในการความยาวของ Link1 โดยพบว่าได้ค่า 401.20 mm. หมายถึงว่า Link1 จะมีความยาวประมาณ 400 mm.

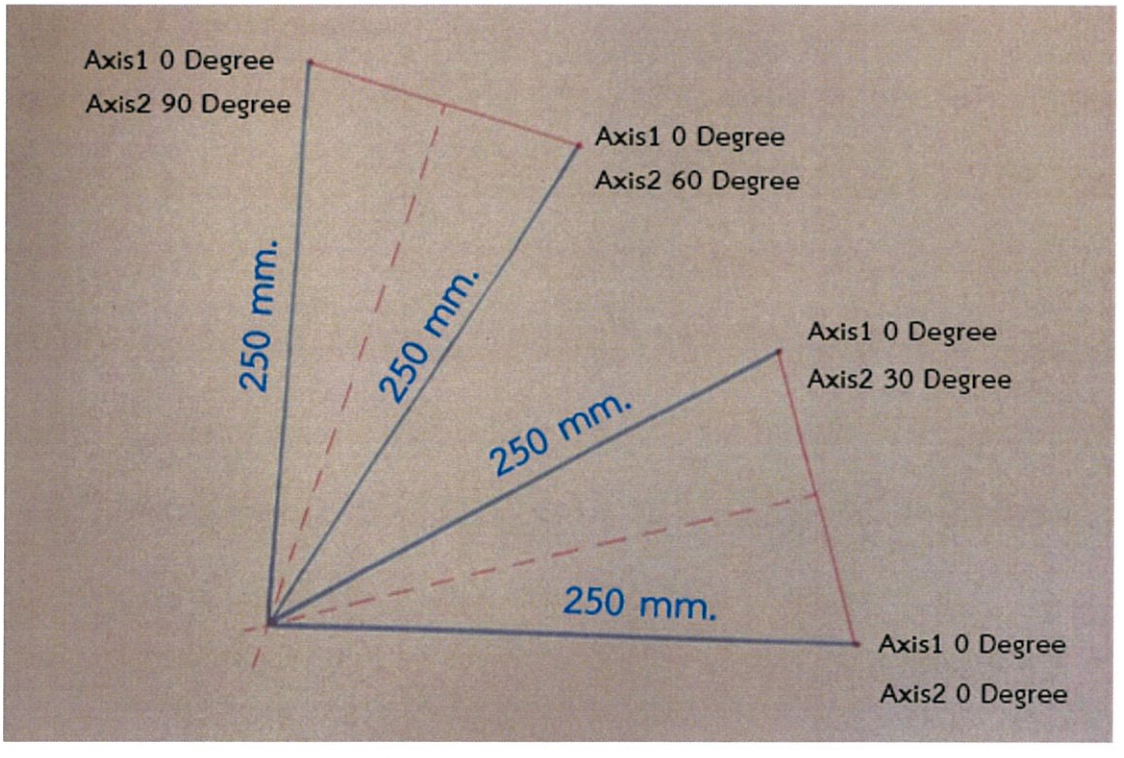
2) การหาความยาว Link2

การหาความยาว Link2 นั้นหาได้โดยการทำเครื่องหมายการเคลื่อนที่ของแกนที่ 2 ที่ระยะ 0 องศา, 30 องศา, 60 องศา และ 90 องศา จากนั้นลากเส้นเชื่อมระหว่างระยะ 0 องศากับ 30 องศา และเชื่อมระหว่างระยะ 60 องศากับ 90 องศา แล้วสร้างเส้นตั้งฉากกับเส้นเชื่อมทั้งสอง เพื่อหาจุดตัดของเส้นตั้งฉากนั้น โดยระยะระหว่างจุดตัดของเส้นตั้งฉากถึงจุดต่าง ๆ ที่หุ่นยนต์เคลื่อนที่ไปได้นั้นก็คือความยาวของ Link2 ซึ่งหลังจากทำการทดองนั้นได้ความยาว Link2 เท่ากับ 250 mm. แสดงดังรูปที่ 3.24 และ 3.25

นั้นก็คือความยาวของ Link2 ซึ่งหลังจากทำการทดองนั้นได้ความยาว Link2 เท่ากับ 250 mm. แสดงดังรูปที่ 3.24 และ 3.25



รูปที่ 3.24 การหาความยาว Link2



รูปที่ 3.25 ผลการหาความยาว Link2

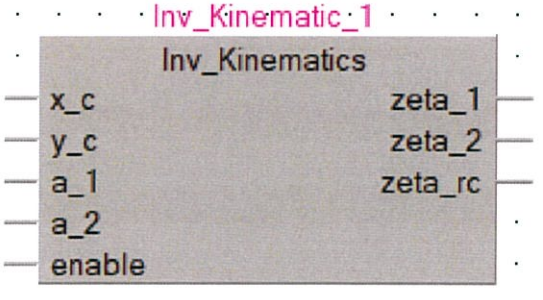
3.9.2 ฟังก์ชันบล็อคสำหรับสมการคำนวณ Kinematics ของหุ่นยนต์

สำหรับฟังก์ชันบล็อคสำหรับสมการคำนวณ Kinematics ของหุ่นยนต์นั้นสามารถนำสมการของ Inverse Kinematics และ Forward Kinematics ที่กล่าวข้อที่ 3.3.1 และ 3.3.2 มาเขียน

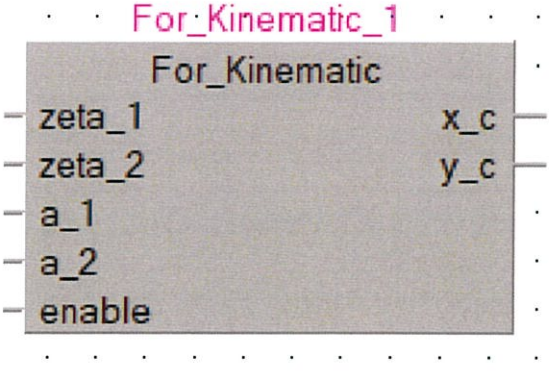
โปรแกรมคำนวณได้เลย เพื่อเป็นการง่ายสำหรับการเขียนโปรแกรมนั้นได้เลือกใช้ภาษา Structure Text เป็นภาษาหลักในการเขียนโปรแกรมคำนวณ โดยภายในโปรแกรมต้องมีการกำหนดพารามิเตอร์ต่าง ๆ ที่จำเป็นสำหรับการคำนวณด้วย เช่น ต้องมีส่วนที่ระบุความยาวของ Link1 และLink2, Work Area ที่สามารถทำงานได้ และพิกัด x, y, z หรือองศาที่ต้องการแปลงตัวอย่างการใช้ Stucture Text สำหรับสร้างฟังก์ชันบล็อคสมการคำนวณ Kinematics ของหุ่นยนต์และตัวอย่างฟังก์ชันบล็อคสมการคำนวณ Kinematics ของหุ่นยนต์ แสดงดังรูปที่ 3.26-3.28

```
(* invert kinematic function block *)
(*Automation Engineering KMITL *)
(* Trial version 15 december 2560*)

rc:=sqrt(x_c*x_c+y_c*y_c);     (* rc    = sqrt(x_c^2 + y_c^2) *)
rc_min:=a_1-a_2;               (*inner keep out area can not use this area  *)
rc_max:=a_1+a_2;               (*out of operating area cannot use area *)

IF (rc >=rc_min) AND (rc<=rc_max) AND (x_c>=zero) THEN  ;  (*  case left hand side robot arm and a_1-link longer than a_2-
link
                                                            In this case can not solve o(0,0) (x_c mut bee mothan or equal a_1 - a_
2)
                                                            *)

j_1:=  ((x_c*x_c+y_c*y_c)-(a_1*a_1+a_2*a_2))/(2.0*(a_1*a_2)),
ACOS(enable , j_1 , zeta_2 );  (* zeta_2 = acos( [ (x_c^2 + y_c^2)-(a_1^2 + a_2^2) ]/2*(a_1*a_2) ) *)
ACOS(enable,x_c/rc,zeta_rc);    (* zeta_rc = acos(x_c/rc)  *)
COS(enable,zeta_2,j_2);         (* j_2   =  cos(zeta_2*)
j_3:=(a_2*j_2+a_1)/rc;          (* j_3   = ( a_1+ a_2*cos(zeta_2))/rc *)
```

รูปที่ 3.26 ตัวอย่าง Stucture Text สมการคำนวณ Kinematics



รูปที่ 3.27 Inverse Kinematics Function Block



รูปที่ 3.28 Forward Kinematics Function Block

3.10 การสร้าง Programing Pendant จาก HMI ด้วย GT Designer3

ในหัวข้อนี้จะนำเสนอการใช้อุปกรณ์ GOT 1000 มาจำลองเป็น Programing Pendant ด้วยการเขียน HMI ผ่านโปรแกรม GT Designer3 ซึ่งส่วนประกอบสำคัญที่ต้องใช้ในการควบคุมหุ่นยนต์สามารถแบ่งได้ดังนี้

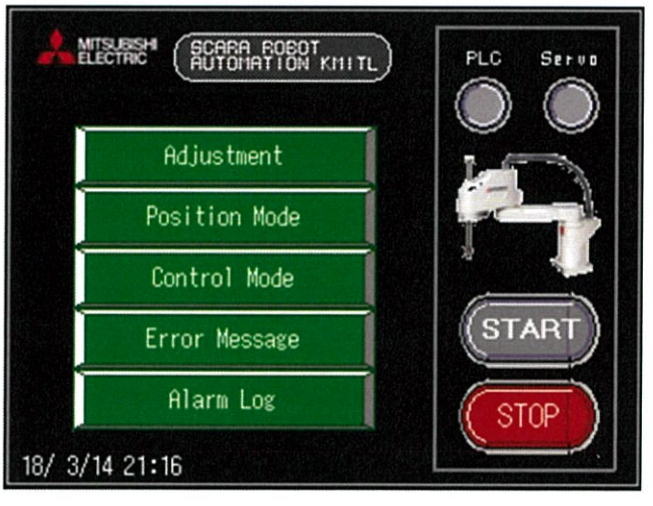
1. Adjustment คือ การควบคุมหุ่นยนต์ในโหมด Jog Operation

2. Position Mode คือ การควบคุมหุ่นยนต์ในโหมดการทำงาน Automatic Position Control

3. Control Mode คือ การควบคุมหุ่นยนต์ในโหมดการทำงานแบบ Coordinate Movement Control

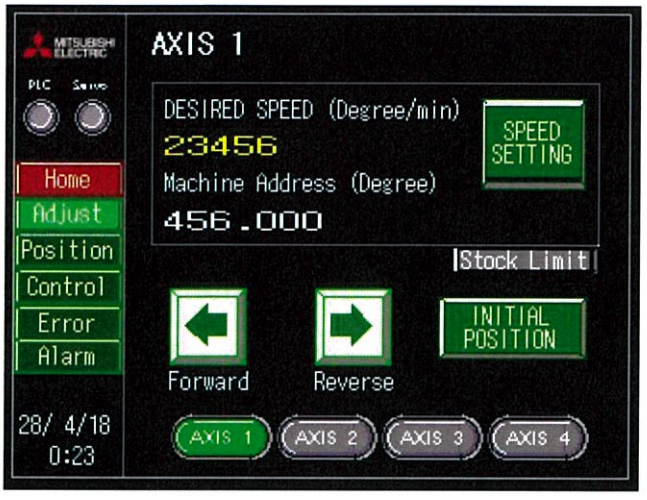
4. Error Message คือ การแจ้งเตือนข้อความเมื่อหุ่นยนต์ทำงานเกิดความผิดพลาด

5. Alarm Log คือ โหมดเก็บข้อมูลสถานะการทำงานล้มเหลวของหุ่นยนต์



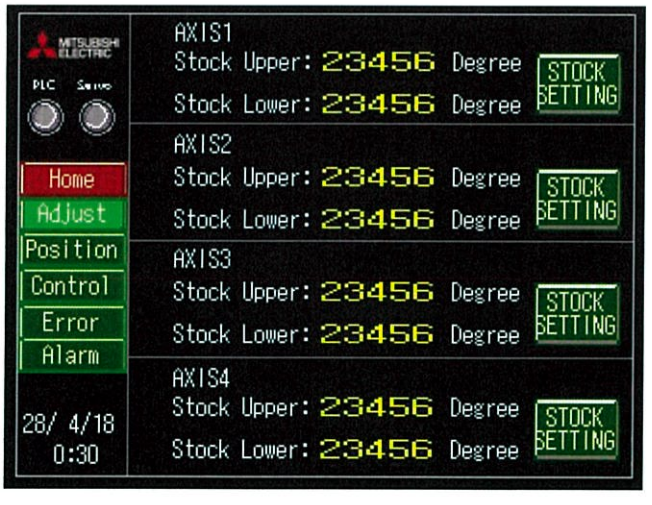
รูปที่ 3.29 หน้าต่าง Overview

จากรูปที่ 3.29 แสดงถึงหน้า Overview ของหุ่นยนต์ การเริ่มทำงานต้องกดปุ่ม Start หรือหากต้องการหยุดการทำงานต้องกดปุ่ม Stop นอกจากนี้ยังมีการแสดงไฟสถานะของ PLC และ Servo หากไฟติดทั้ง 2 สถานะ นั่นหมายถึงหุ่นยนต์พร้อมสำหรับการทำงาน จากนั้นก็เลือกโหมดการทำงานของหุ่นยนต์ตามที่ต้องการ



รูปที่ 3.30 หน้าต่างโหมด Adjustment

จากรูปที่ 3.30 แสดงถึงส่วนประกอบของโหมด Adjustment เช่น ควบคุมการทำงานแบบ Jog Operation ในหน้านี้ผู้ใช้งานสามารถปรับความเร็วการเคลื่อนที่ของหุ่นยนต์ได้ อีกทั้งยังสามารถดูสถานะปัจจุบันได้ว่าปัจจุบันหุ่นยนต์อยู่ที่ตำแหน่งใด นอกจากนี้ ผู้ใช้งานยังสามารถปรับตั้งตำแหน่งอ้างอิงหรือจุด Home ได้จากหน้านี้อีกด้วย โดยโหมด Adjustment นี้ผู้ใช้งานสามารถเลือกควบคุมหรือดูสถานะได้ทั้ง 4 แกน โดยเลือกแกนที่ด้านล่างของหน้าจอ

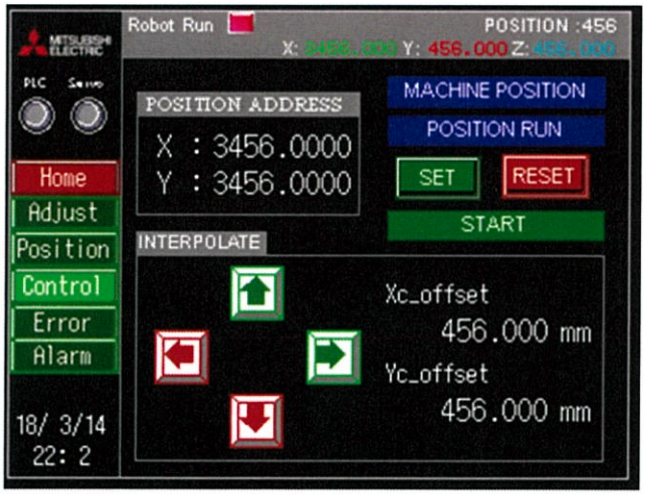


รูปที่ 3.31 หน้าต่างการตั้งค่า Stock Limit

จากรูปที่ 3.31 คือการกำหนดค่าระยะต่ำสุดและสูงสุดของ Software Stroke limit

รูปที่ 3.32 หน้าต่างโหมด Position

จากรูปที่ 3.32 คือการใช้งานโหมด Automatic Position Control เป็นหน้าต่างการตั้งค่า Positioning Data ต่าง ๆ หลังจากที่ตั้งค่าแล้วสามรถทดสอบการเคลื่อนที่โดยการกดปุ่ม Position Run เพื่อดำเนินการทำงานแบบ Automatic Position Control ตามที่ตั้งค่าไว้ ซึ่งการกำหนด Position ที่ต้องการเคลื่อนที่นั้นสามารถระบุหรือจัดการได้ในหน้าต่างที่แสดงในรูปที่ 3.34



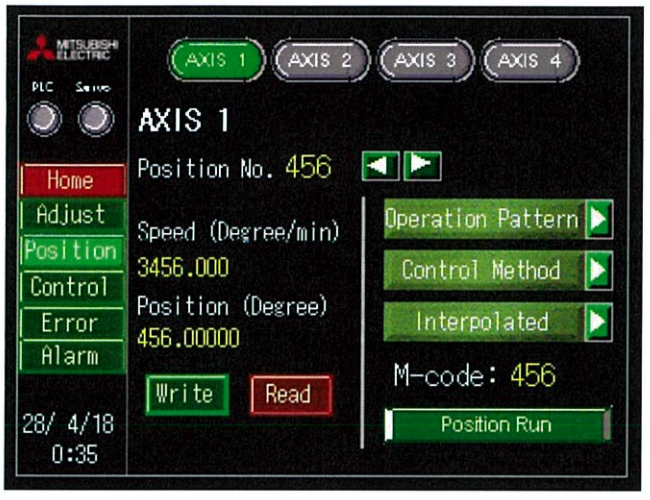
รูปที่ 3.33 หน้าต่างโหมดการเคลื่อนที่แบบระบุพิกัด

สำหรับรูปที่ 3.33 คือการใช้งานโหมด Coordinate Movement Control ซึ่งเป็นหน้าต่างที่มีให้เลือกการใช้งาน 2 แบบ ได้แก่

1.การระบุพิกัด X และ Y ที่ต้องการเคลื่อนที่ เพื่อให้หุ่นยนต์เคลื่อนที่ไปยังพิกัดปลายทาง

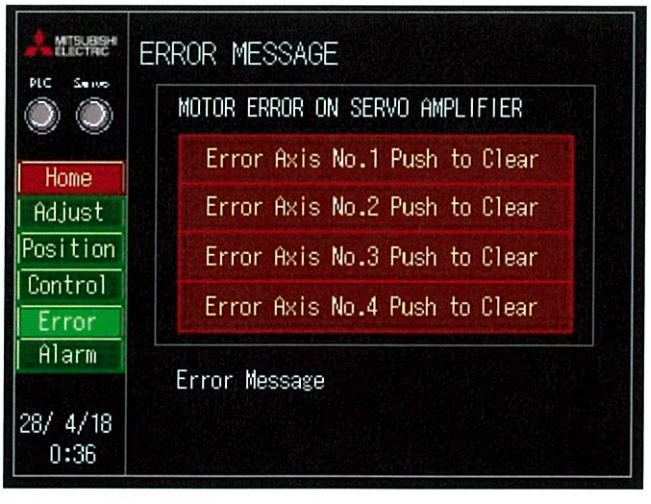
2.การกดทิศทางเพื่อให้หุ่นยนต์เคลื่อนที่ไปยังพิกัด X และ Y ที่กำหนด โดยที่การใส่ offset คือการกำหนดระยะทางการเคลื่อนที่ต่อหนึ่งหน่วย

ในโหมดการทำงานแบบ Coordinate Movement Control ทุกการเคลื่อนที่จะมีการบันทึกไว้ใน Positioning Data ซึ่งสามารถเรียกการเคลื่อนที่ซ้ำได้ ด้วยการกดปุ่ม Position Run อีกทั้งยังมีแถบสีเทาที่แสดงตำแหน่งปัจจุบันของหุ่นยนต์ และสถานะการทำงานของหุ่นยนต์



รูปที่ 3.34 หน้าต่างใช้งาน Position Run

ในรูปที่ 3.34 แสดงหน้าต่าง Position Run จากที่กล่าวมาข้างต้นในโหมดการทำงานแบบ Automatic Position Control (รูปที่ 3.32) และโหมดการทำงานแบบ Coordinate Movement Control (รูปที่ 3.33) สำหรับโหมดการทำงานแบบ Automatic Position Control นั้นเมื่อตั้งค่าเสร็จแล้ว สามารถทดสอบการเคลื่อนที่ด้วยการระบุหมายเลข Position No. แล้วสั่ง Start และในส่วนของโหมดการทำงานแบบ Coordinate Movement Control สามารถตั้งค่าเพิ่มเติมในหน้าต่างนี้ นอกจากนี้ในหน้าต่างนี้ยังสามารถแสดงตำแหน่งปัจจุบันของหุ่นยนต์ได้ทั้ง 2 แบบไม่ว่าจะในแบบองศาหรือตำแหน่งตามพิกัด X และ Y



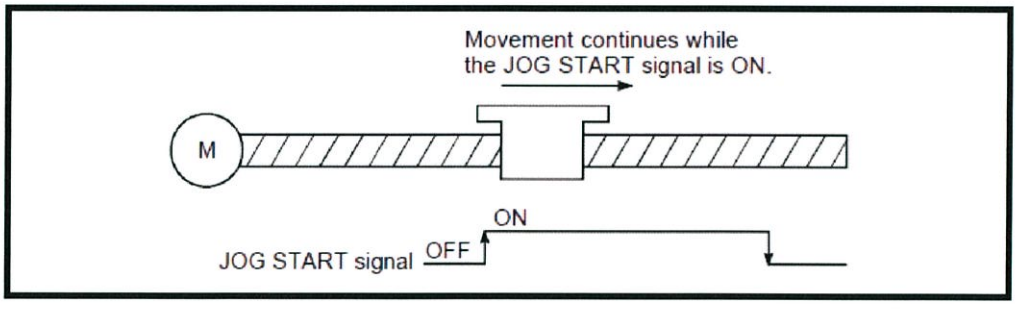
รูปที่ 3.35 หน้าต่าง Error Message

สำหรับในหน้าต่าง Error Message (รูปที่ 3.35) เมื่อหุ่นยนต์มีการทำงานผิดพลาดจนทำให้เคลื่อนที่ต่อไม่ได้ สามารถตรวจความผิดพลาดได้ที่หน้าต่างนี้ โดยที่ Error Message โดยจะแสดงข้อความว่าแกนใดเกิดความผิดพลาด เพื่อให้ผู้ใช้แก้ไขปัญหา หลังจากแก้ไขเสร็จ ต้องทำการกดปุ่ม Error Axis Push to Clear เพื่อเป็นการทำให้แกนของหุ่นยนต์นั้นกลับสู่สถานะพร้อมทำงาน

รูปที่ 3.36 หน้าต่าง Alarm Log

สำหรับในหน้าต่าง Alarm Log (รูปที่ 3.36) นั้นคือการเก็บประวัติการทำงานที่ผิดพลาดของหุ่นยนต์ เพื่อให้ผู้ใช้สามารถตรวจสอบย้อนหลังได้

3.11 ฟังก์ชันบล็อคสำหรับการทำงานแบบ Jog Operation ด้วย GX Works2 [12]

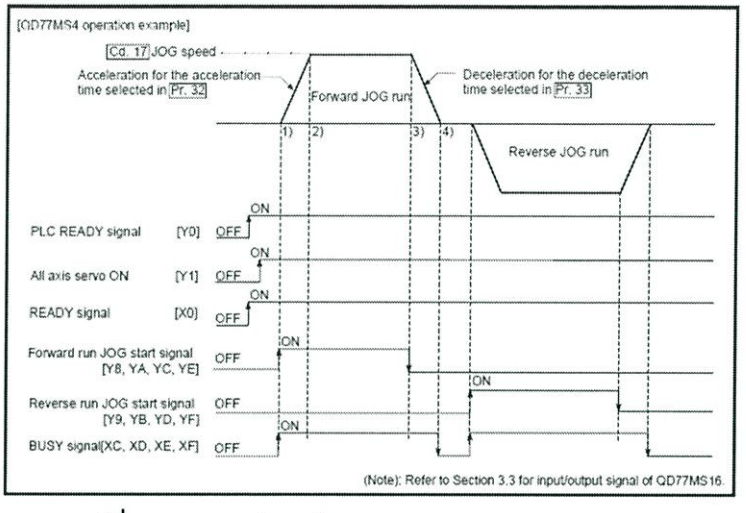


รูปที่ 3.37 การทำงานแบบ Jog Operation

จากรูป 3.37 อธิบายถึง Jog Operation ซึ่งเป็นวิธีควบคุมด้วยมือ (Manual Control) วิธีหนึ่ง หมายถึงการควบคุมโดยไม่ใช้ Positioning Data (การระบุตำแหน่ง) ในการทำให้หุ่นยนต์เคลื่อนที่ แต่หุ่นยนต์จะเคลื่อนที่เมื่อสัญญาณ Jog Start ทำงาน โดยที่หุ่นยนต์ต้องเคลื่อนที่ภายในช่วงที่จำกัดไว้ด้วย Limit Switch และหุ่นยนต์จะหยุดเคลื่อนที่เมื่อสัญญาณ Jog Start นั้นหยุดทำงานหรือเกินช่วงที่จำกัดไว้ (เกิน Limit Switch)

การทำงานแบบ Jog Operation ของหุ่นยนต์มีการเคลื่อนที่อยู่ 2 ทิศทางคือ Forward (เคลื่อนที่ไปข้างหน้า) และ Reverse (เคลื่อนที่ถอยหลัง) ซึ่งหุ่นยนต์จะสามารถเคลื่อนที่ในทิศทางใดทิศทางหนึ่งเท่านั้นไม่สามารถทำงานพร้อมกันได้

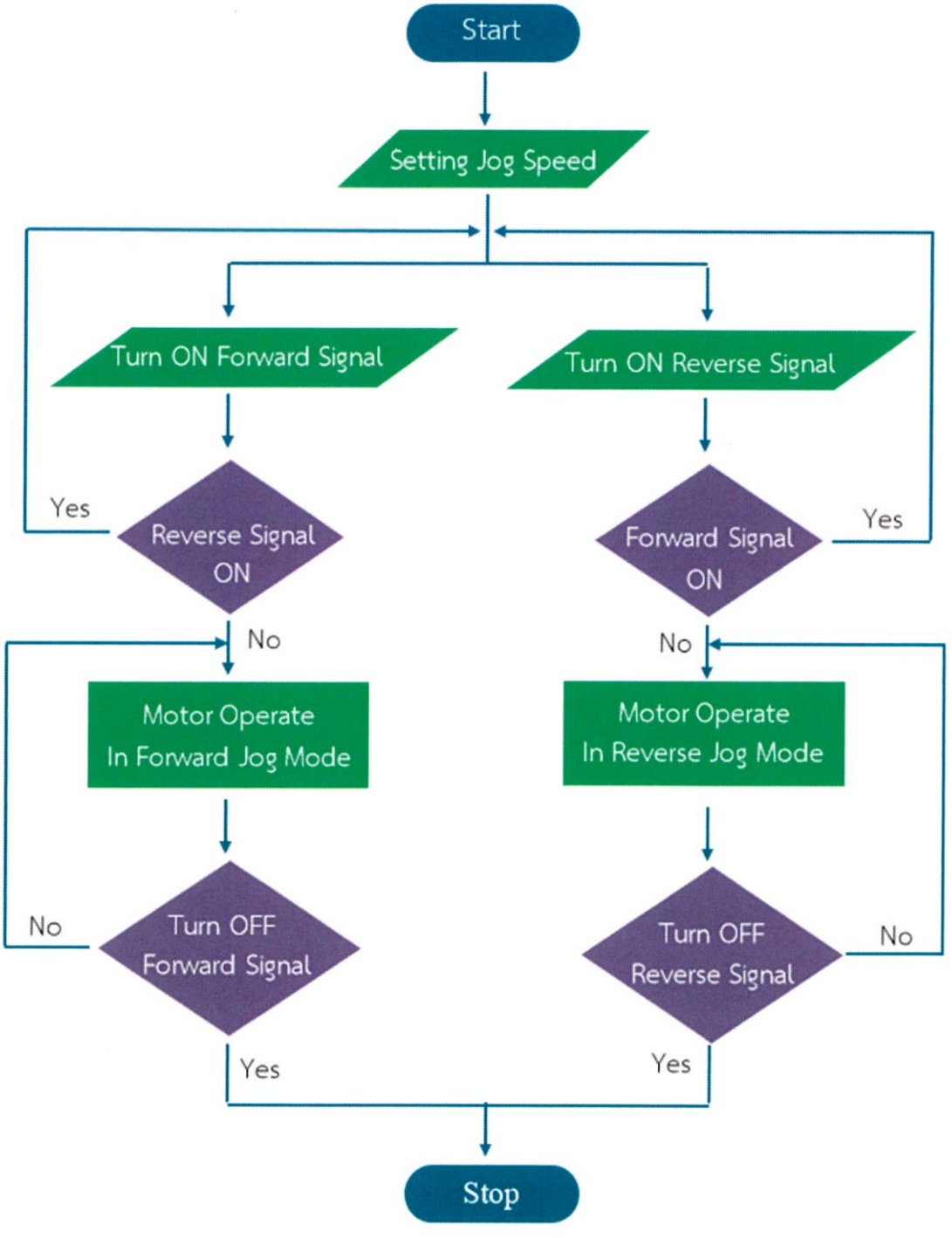
เมื่อมีคำสั่งให้หุ่นยนต์เคลื่อนที่ไปในทาง Forward หุ่นยนต์ก็จะเคลื่อนที่ไปในทิศทาง Forward จนกว่าสัญญาณ Forward จะไม่ทำงาน สำหรับทาง Reverse ก็เช่นเดียวกัน และในขณะที่หุ่นยนต์กำลังเคลื่อนที่ไปในทิศทาง Forward หรือ Reverse ในขณะที่หุ่นยนต์เคลื่อนที่อยู่ สัญญาณ Busy จะเป็น ON และจะ OFF เมื่อหยุดการเคลื่อนที่ แสดงตามรูปที่ 3.38



รูปที่ 3.38 กราฟการทำงาน Jog Operation

3.11.1 การเขียนฟังก์ชันบล็อคสำหรับควบคุมการทำงานแบบ Jog Operation

แผนผังการเขียนฟังก์ชันบล็อคสำหรับควบคุมการทำงานแบบ Jog Operation



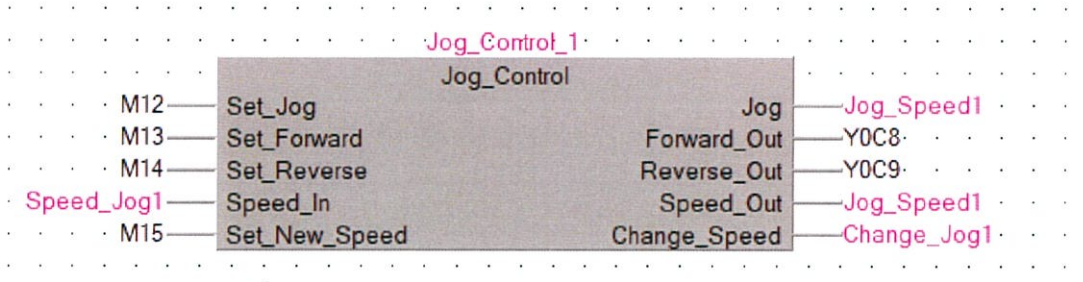
รูปที่ 3.39 แผนผังการเขียนฟังก์ชันบล็อคสำหรับควบคุมการทำงานแบบ Jog Operation

จากรูปที่ 3.39 จะเป็นการแสดงหลักการทำงานในส่วนของ Jog Operation โดยจะเริ่มจากการกำหนดค่าความเร็วในการเคลื่อนที่ของมอเตอร์ เมื่อส่งค่าความเร็วในการเคลื่อนที่ให้กับมอเตอร์แล้ว

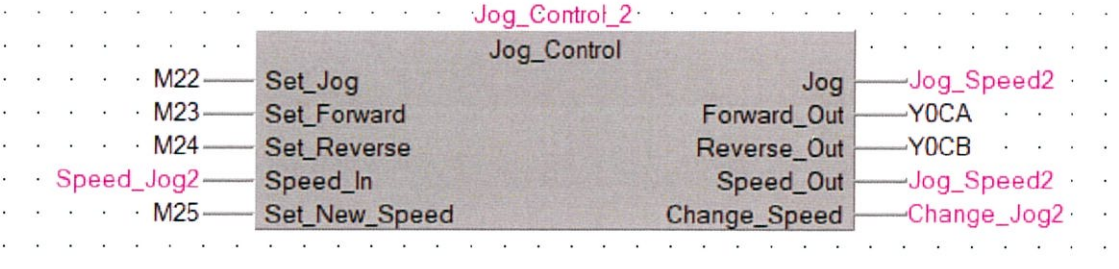
จากนั้นจะรับคำสั่งว่ามีการให้เคลื่อนที่ไปในทิศทาง Forward หรือ Reverse หากได้รับคำสั่งให้เคลื่อนที่ไปในทิศทาง Forward จะทำการตรวจสอบว่ามีคำสั่ง Reverse หรือไม่ หากมีมอเตอร์จะยังคงไม่ทำงานแต่จะกลับไปตรวจสอบอีกครั้งว่ามีคำสั่งจากสัญญาณใด หากไม่มีคำสั่ง Reverse มอเตอร์จะดำเนินงานในโหมด Jog Operation ในทิศทาง Forward ไปเรื่อย ๆ จนกว่าสัญญาณสั่งให้เคลื่อนที่ในทิศทาง Forward จะหายไปเป็นการจบการทำงาน

สำหรับการเคลื่อนที่ในทิศทาง Reverse ก็เช่นกัน หากได้รับคำสั่งให้เคลื่อนที่ไปในทิศทาง Reverse จะทำการตรวจสอบว่ามีคำสั่ง Forward หรือไม่ หากมีมอเตอร์จะยังคงไม่ทำงานแต่จะกลับไปตรวจสอบอีกครั้งว่ามีคำสั่งจากสัญญาณใด หากไม่มีคำสั่ง Forward มอเตอร์จะดำเนินงานในโหมด Jog Operation ในทิศทาง Reverse ไปเรื่อย ๆ จนกว่าสัญญาณสั่งให้เคลื่อนที่ในทิศทาง Forward จะหายไปเป็นการจบการทำงาน

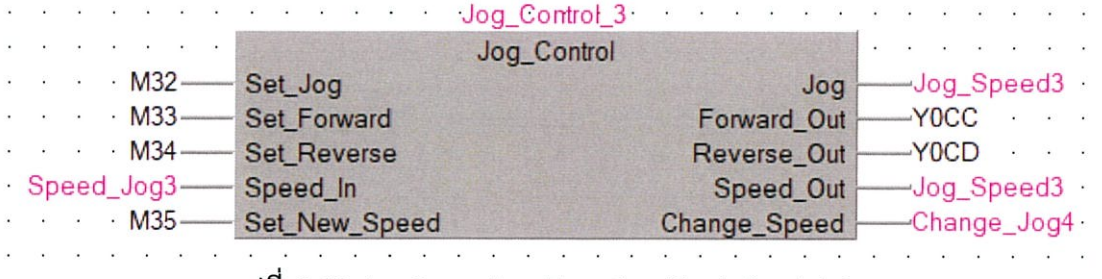
ตัวอย่าง ฟังก์ชันบล็อกสำหรับการควบคุมการทำงานแบบ Jog Operation ของแต่ละแกน แสดงดังรูปที่ 3.40-3.43



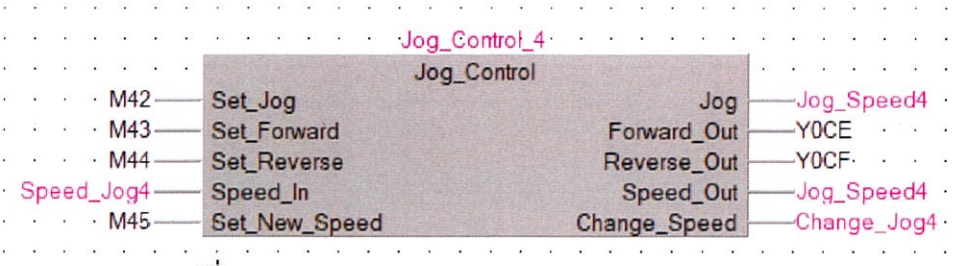
รูปที่ 3.40 Jog Operation Function Block for Axis1



รูปที่ 3.41 Jog Operation Function Block for Axis2



รูปที่ 3.42 Jog Operation Function Block for Axis3



รูปที่ 3.43 Jog Operation Function Block for Axis4

3.11.2 การทดลองการหาอัตราทดที่ Joint ของหุ่นยนต์

การหาอัตราทดนั้นสามารถหาได้โดยการใช้การทำงานแบบ Jog Operation ที่ปรับความเร็วการเคลื่อนที่ไว้สูง ๆ จากนั้นสังเกตว่าการที่หุ่นยนต์เคลื่อนที่จริงของหุ่นยนต์นั้นเป็นสัดส่วนกับค่า Machine Feed Value เท่าไหร่

1) การหาอัตราทดแกนที่ 1 โดยสังเกตจากการที่หุ่นยนต์เคลื่อนจาก 0 - 180 องศา ได้ผลดังตารางที่ 3.5

ตารางที่ 3.5 ผลการทดลองหาอัตราทดของแกนที่ 1

Reduction Gear Ratio 1:1		Reduction Gear Ratio 1:78		Error (%)
Machine Feed Value	Reduction Gear Ratio(Calculate)	Machine Feed Value	Machine Movement	
1001.1255		6.98250		
-12937.94871	77.439	-171.46809	178.45059	-0.8608
1059.48514	77.7635	7.16169	178.62978	-0.7612
-12952.12861	77.84229	-171.57074	178.73243	-0.7042
Error Average				-0.7754

2) การหาอัตราทดแกนที่ 2 โดยสังเกตจากการที่หุ่นยนต์เคลื่อนจาก 0-180 องศา ได้ผลดังตารางที่ 3.6

ตารางที่ 3.6 ผลการทดลองหาอัตราทดของแกนที่ 2

Reduction Gear Ratio 1:1		Reduction Gear Ratio 1:80		Error (%)
Machine Feed Value	Reduction Gear Ratio(Calculate)	Machine Feed Value	Machine Movement	
-2077.50536		-42.03710		
12525.77506	81.1293	140.54164	182.57874	1.4326
-2077.18790	81.12757	-41.03534	181.57698	0.8761
12396.18511	80.40763	140.68582	181.42166	0.9562
Error Average				1.0883

3) การหาอัตราทดแกนที่ 3 โดยสังเกตจากการที่หุ่นยนต์เคลื่อนจาก0-360 องศา ได้ผลดังตารางที่ 3.7

ตารางที่ 3.7 ผลการทดลองหาอัตราทดของแกนที่ 3

Reduction Gear Ratio 1:1		Reduction Gear Ratio 1:15		Error (%)
Machine Feed Value	Reduction Gear Ratio(Calculate)	Machine Feed Value	Machine Movement	
-617.11078		359.99179		
4884.22227	15.28148	1.62222	1.62222	-0.45289
-370.52723	14.59653	359.96666	-0.03333	-0.45988
5216.76125	15.52025	1.82962	1.82962	-0.51749
Error Average				-0.47675

4) การหาอัตราทดแกนที่ 4 โดยสังเกตจากการที่หุ่นยนต์เคลื่อนจาก 0-360 องศา ได้ผลดังตารางที่ 3.8

ตารางที่ 3.8 ผลการทดลองหาอัตราทดของแกนที่ 4

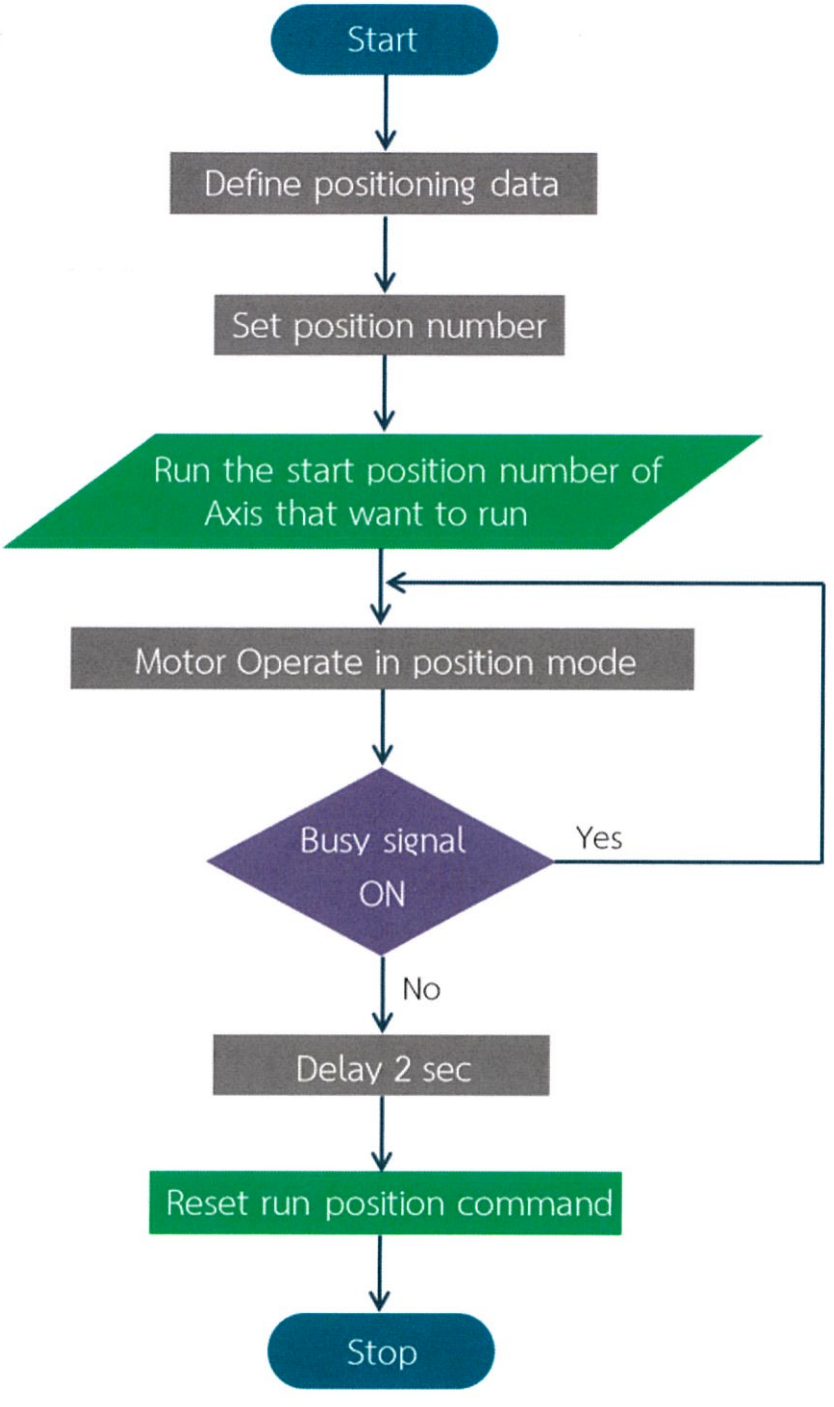
Reduction Gear Ratio 1:1		Reduction Gear Ratio 1:50		Error (%)
achine Feed Value	Reduction Gear Ratio(Calculate)	Machine Feed Value	Machine Movement	
-15822.62603		2212.39265		
2226.24963	50.13577	2571.88895	359.4	-1.6667
20275.12533	50.15766	2932.89636	361.007	0.2797
38172.99059	49.71629	3291.87413	358.97	-0.2861
Error Average				0.5577

3.12 ฟังก์ชันบล็อคการทำงานแบบ Automatic Position Control ด้วย GX Works2

การควบคุมการทำงานรูปแบบ Automatic Position Control หมายถึง การสั่งให้หุ่นยนต์เคลื่อนที่ไปในตำแหน่งต่าง ๆ ด้วยการระบุตำแหน่งโดยใช้ข้อมูลองศาเป็นตัวกำหนด ซึ่งข้อมูลการระบุตำแหน่งนี้จะเรียกว่า Positioning Data

จากรูปที่ 3.44 อธิบายถึงลำดับในการทำงานของ Automatic Position Control การทำงานในโหมดนี้จะเริ่มจากการกำหนด Positioning Data ให้แก่ Servo Amplifier และกำหนดหมายเลขหรือ Position No. เริ่มต้นสำหรับการเริ่มการทำงานในรูปแบบ Automatic Position Control จากนั่นเมื่อได้รับคำสั่งเริ่มการทำงาน มอเตอร์จะดำเนินการทำงานในโหมด Automatic Position Control เรื่อย ๆ จนกว่าสัญญาณ Busy Signal จะไม่ทำงาน แล้วจะถ่วงเวลาไว้ 2 วินาทีจากนั้นจะล้างคำสั่งการทำงานรูปแบบ Automatic Position Control เป็นการจบกระบวนการเช่น ให้เริ่มการทำงานจาก Position No.1 จากนั้นมอเตอร์จะทำการเคลื่อนที่ไปในตำแหน่งที่กำหนด ตาม Positioning Data จนกว่าจะเจอคำสั่ง END จาก Positioning Data และในระหว่างการเคลื่อนที่จะส่งสัญญาณ Busy signal ออกมา เมื่อการเคลื่อนที่สิ้นสุดลง และสัญญาณ Busy signal หยุดทำงานโปรแกรมจะหน่วงเวลาไว้ 2 วินาที ก่อนจะรีเซ็ตค่าคำสั่งให้ทำงานแบบ Automatic Position Control เป็นการเสร็จสิ้นการทำงาน

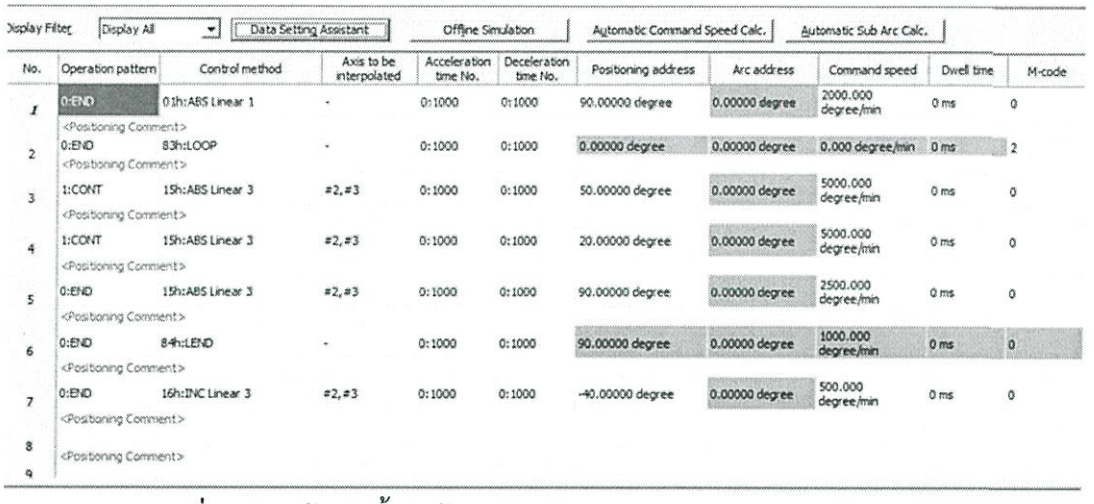
3.12.1 แผนผังฟังก์ชันบล็อคควบคุมการทำงานแบบ Automatic Position Control



รูปที่ 3.44 แผนผังฟังก์ชันบล็อคควบคุมการทำงานแบบ Automatic Position Control

3.12.2 การตั้งค่าการทำงานแบบ Automatic Position Control

สำหรับการทำงานในโหมดการทำงานแบบ Automatic Position Control ต้องมีการตั้งค่าพารามิเตอร์ต่าง ๆ ที่เป็นข้อมูลสำหรับการทำงานแบบ Automatic Position Control ของแต่ละแกนก่อน โดยพารามิเตอร์เหล่านี้ เรียกว่า Positioning Data ซึ่งสามารถกำหนดจุดการเคลื่อนที่ของแต่ละแกนได้ทั้งหมด 600 จุด หรือ 600 Position



รูปที่ 3.45 หน้าต่างตั้งค่าข้อมูลสำหรับ Automatic Position Control

จากรูปที่ 3.45 แสดงถึงตัวอย่างตั้งค่าข้อมูลการเคลื่อนในแต่ละจุด โดยมีรายละเอียดการตั้งค่าแต่ละพารามิเตอร์ แสดงดังตารางที่ 3.9

ตารางที่ 3.9 รายละเอียด Positioning Data Parameter

Positioning Data Parameter	คำอธิบาย
Operation Pattern	รูปแบบการทำงานของมอเตอร์
Control Method	วิธีควบคุมหรือวิธีสั่งการทำงานของมอเตอร์
Axis to Interpolate	แกนที่จะมาทำงานร่วม
Acceleration time No.	อัตราเร่งที่ใช้ในโหมด Automatic Position Control
Deceleration time No.	อัตราหน่วงที่ใช้ในโหมด Automatic Position Control
Positioning address	ตำแหน่งเป้าหมายของ Automatic Position Control ซึ่งขึ้นอยู่กับการตั้งค่า Control Method
Arc address	เป็นค่าที่ต้องตั้งในกรณีที่ต้องการเคลื่อนที่เป็นเส้นโค้ง

ตารางที่ 3.9 (ต่อ)

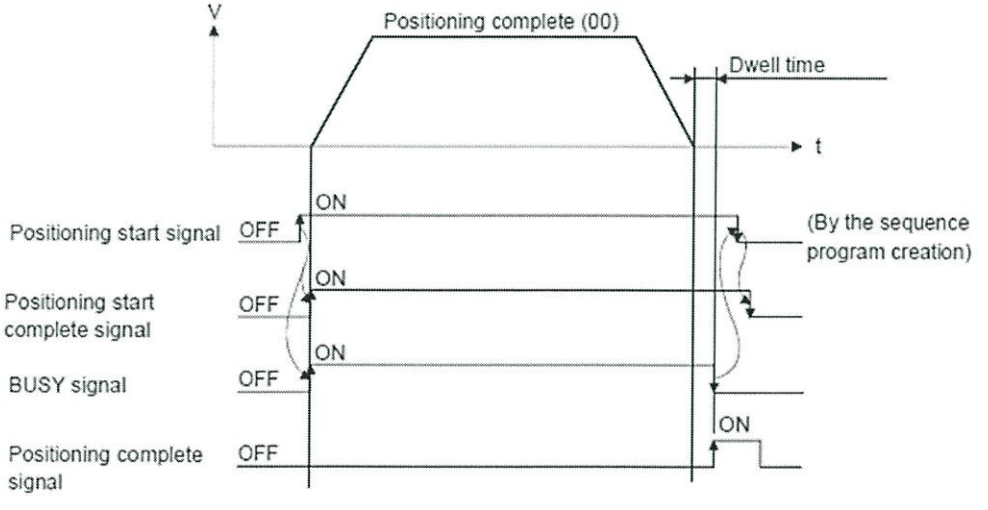
Positioning Data Parameter	คำอธิบาย
Command Speed	ความเร็วที่ใช้ในการเคลื่อนที่ของ Position นั้น ๆ
Dwell time	เวลาที่หยุดรอ
M-code	แบ่งออกเป็น 2 กรณี 1) กรณีที่ Control Method เป็น Loop ค่าที่ใส่ใน M-code หมายถึง จำนวนรอบการทำงาน 2) กรณีที่ Control Method ไม่ใช่ Loop ค่าที่ใส่ใน M-code หมายถึงการเลือก Event ที่ต้องการจะทำงานใน Position นั้น ๆ หรือการเปิดการทำงานโหมด M-code Event Selection

1) Operation Pattern [12]

Operation Pattern หมายถึง การกำหนดรูปแบบการทำงาน ซึ่งประกอบไปด้วย 3 รูปแบบ ดังนี้

1.1) END (00)

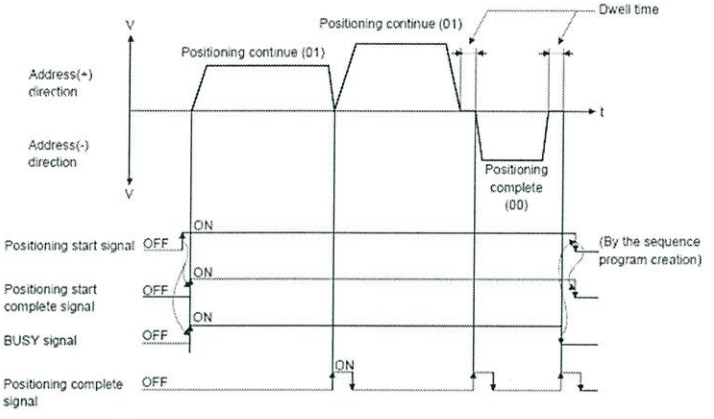
รูปแบบ END หมายถึง รูปแบบการทำงานแบบเสร็จสิ้น (Position Complete) คือ การเสร็จสิ้นการควบคุมแบบ Position Control ซึ่งอาจจะเป็นข้อมูลการเคลื่อนที่ลำดับสุดท้ายของรูปแบบการทำงานแบบต่อเนื่อง (Continuous) และแบบพาธต่อเนื่อง (Continuous Path) เมื่อใดก็ตามเมื่อมอเตอร์เคลื่อนตาม Positioning Data ที่กำหนดแล้วพบ Position ที่มีรูปแบบการทำงานแบบ END จะถือเป็น Position สุดท้ายที่จะดำเนินการเคลื่อนที่หลังจากดำเนินการตาม Positioning data เสร็จสิ้นแล้วมอเตอร์จะหยุดทำงาน เป็นการจบการทำงานแบบ Automatic Position Control แล้วรอคำสั่งเริ่มการทำงานใหม่อีกครั้ง แสดงดังรูปที่ 3.46



รูปที่ 3.46 การทำงานแบบ Position Complete

1.2) CONS (01)

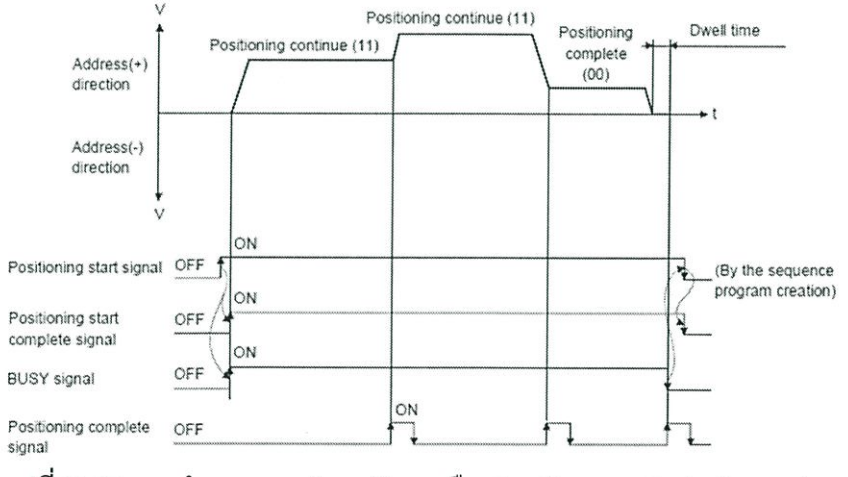
รูปแบบ CONS หมายถึง รูปแบบการทำงานแบบต่อเนื่อง (Continuous Positioning Control) คือ เมื่อจบการทำงานของ Position แรก (ไม่จำเป็นต้องเป็น Position No.1) จะลดความเร็วของมอเตอร์จนเหลือ 0 แล้วหลังจากนั้นจะดำเนินการตาม Position ถัดไปเรื่อย ๆ จนกว่าจะเจอ Position ที่เป็นรูปแบบ END ซึ่งทุก ๆ ครั้งที่จบการทำงาน Position ใด ๆ จะลดความเร็วมอเตอร์ให้เป็น 0 ก่อนที่จะทำงาน Position ถัดไปเสมอ แสดงดังรูปที่ 3.47



รูปที่ 3.47 การทำงานแบบ Position Complete

1.3) LOCA [11]

รูปแบบ LOCA หมายถึง รูปแบบการทำงานแบบพาธต่อเนื่อง (Location หรือ Continuous Path Control) รูปแบบการทำงานแบบนี้มีลักษณะการทำงานที่คล้ายกับการทำงานแบบต่อเนื่องมาก คือ จะทำงาน Position ถัดไปเรื่อย ๆ จนว่าจะเจอรูปแบบการทำงานแบบ END แต่ต่างกันที่แบบพาธต่อเนื่องนั้นความเร็วการเคลื่อนที่จะต่อจาก Position ก่อนหน้า คือ ไม่มีการหยุดการทำงานก่อนที่จะทำงานใน Position ถัดไปแต่การทำงานแบบ CONS จะความเร็วจะถูกลดเป็น 0 ก่อน แสดงดังรูปที่ 3.48



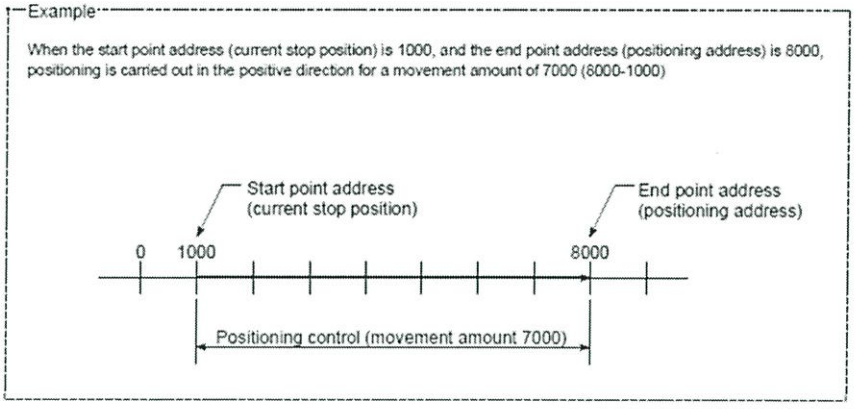
รูปที่ 3.48 การทำงานแบบ Location หรือ Continuous Path Control

2) Control Method [12]

Control Method หมายถึง วิธีการควบคุมการทำงานแบบ Automatic Position Control ซึ่งมีวิธีการควบคุมหลัก ๆ ดังนี้

2.1) ABS Linear

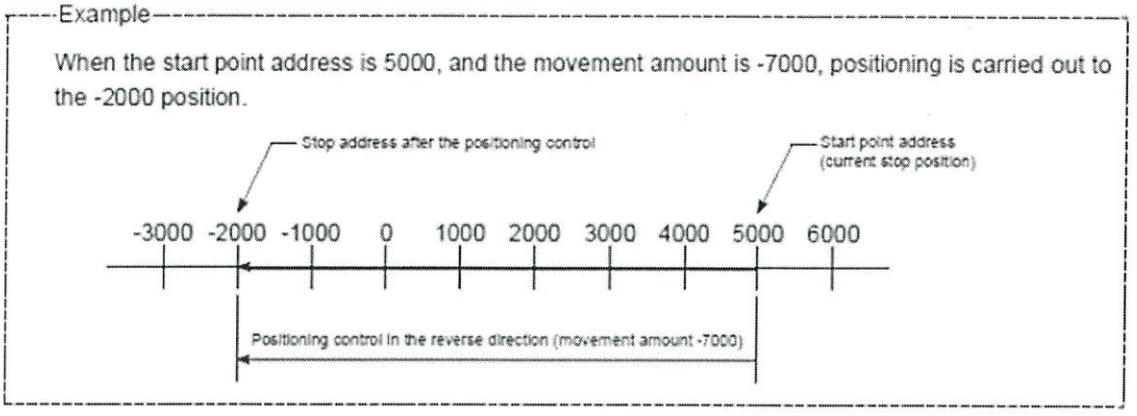
การควบคุมแบบ ABS Linear หมายถึง ลักษณะการควบคุมที่มอเตอร์จะหมุนไปยังตำแหน่งที่ระบุใน Positioning Address ไม่ว่าตอนแรกนั้นมอเตอร์จะอยู่ที่ตำแหน่งใดก็ตาม เมื่อมีการสั่งควบคุมแบบ ABS Linear มอเตอร์จะเคลื่อนที่ไปยังตำแหน่งที่ระบุ สำหรับการทำงานแบบ ABS นี้จะมีทั้งแบบ 1 แกนหรือการ Interpolated หลายแกน เช่น ABS Linear1 จะหมายถึงการที่มอเตอร์เพียงตัวเดียวหมุนหรือเคลื่อนที่ไปยังตำแหน่งที่กำหนด แต่หาก ABS Linear2 จะหมายถึงมอเตอร์ 2 ตัวที่ Interpolated กันจะเคลื่อนที่ไปยังตำแหน่งที่กำหนดของแต่ละตัวพร้อม ๆ กัน ดังรูปที่ 3.49



รูปที่ 3.49 การควบคุมแบบ ABS Linear

2.2) INC Linear

การควบคุมแบบ INC Linear หมายถึง ลักษณะการควบคุมการเคลื่อนที่เป็นการที่ให้มอเตอร์หมุนในทิศทางไปข้างหน้าหรือถอยหลังตาม Positioning Address ที่กำหนด เช่น หากกำหนด Positioning Address เป็น -7000 องศา นั่นคือการให้มอเตอร์หมุนไปข้างหน้าอีก -7000 องศา ถ้าปัจจุบันมอเตอร์อยู่ที่ตำแหน่งที่ 5000 องศา เมื่อเคลื่อนที่ตามวิธีการนี้สุดท้ายมอเตอร์จะอยู่ที่ตำแหน่ง -2000 องศา ซึ่งต่างจากการควบคุมแบบ ABS Linear ที่ไม่ว่ามอเตอร์จะอยู่ตำแหน่งไหนก็ตามจะเคลื่อนที่ไปที่ตำแหน่ง -7000 องศาถ้า Positioning Address ของ ABS Linear คือ -7000 องศา เช่นเดียวกับ ABS Linear วิธีการควบคุมแบบ INC Linear นี้จะมีทั้ง 1 แกนและการ Interpolated แบบหลายแกน ซึ่งแสดงวิธีการควบคุมแบบ INC Linear ดังรูปที่ 3.50



รูปที่ 3.50 การควบคุมแบบ INC Linear

2.3) LOOP และ LEND

การควบคุมแบบ LOOP และ LEND หมายถึง ลักษณะการควบคุมการเคลื่อนที่ที่เป็นคำสั่งการทำซ้ำ Position ในวงการทำงานเริ่มตั้งแต่ Position ที่มีการควบคุมแบบ Loop ไปจนถึง Position ที่มีการควบคุมแบบ LEND ที่เป็นคำสั่งจบวงการทำงน โดยภายในวงการทำงานแต่ละวงนั้นจะต้องมีการควบคุมแบบอื่น ๆ เช่น แบบ ABS หรือ INC หรืออาจจะมีคำสั่งหลาย ๆ รูปแบบก็ได้ จำนวนรอบที่จะทำซ้ำนั้นจะถูกกำหนดไว้ในพารามิเตอร์ M-code ของ Position ที่สั่งควบคุมแบบ Loop ตัวอย่างการควบคุมแบบ Loop แสดงดังรูปที่ 3.51

Positioning data No.	Operation pattern	Control system	Conditions	Operation
1	Continuous control	ABS2		Executed in the order of the positioning data No. 1 → 2 → 3 → 4 → 5 → 2 → 3 → 4 → 5 → 6. (The operation patterns of the positioning data Nos. 2 and 5 are ignored.)
2	Positioning complete	LOOP	Number of loop cycles: 2	
3	Continuous path control	ABS2		
4	Continuous control	ABS2		
5	Positioning complete	LEND		
6	Positioning complete	ABS2		

รูปที่ 3.51 การควบคุมแบบ LOOP และ LEND

3) Axis to be interpolated [12]

Axis to be interpolated หมายถึง แกนที่จะมาทำงานร่วมกัน เช่น การ Interpolated 2 แกน ในการควบคุมแบบ ABS Linear 2 โดยที่แกนที่ 1 เป็นแกนหลัก และแกนที่ 2 เป็นแกนตามหรือแกนที่ทำงานร่วม ในการทำงานแบบ Automatic Position Control แกนที่ 1 กับแกนที่ 2 จะทำงานประสานกันด้วย Position ที่ถูกกำหนดไว้ ใน Positioning Address ของแต่ละแกน สำหรับ QD77MS4 นั่นจะต้องตั้งแกนที่จะเป็นแกนตามแค่การ Interpolated แบบสองแกนเท่านั้น หากมากกว่าจะโมดูลจะกำหนดให้เอง โดยมีการกำหนดแกนตามดังรูปที่ 3.52

Da.5 Axis to be interpolated QD77MS2 QD77MS4

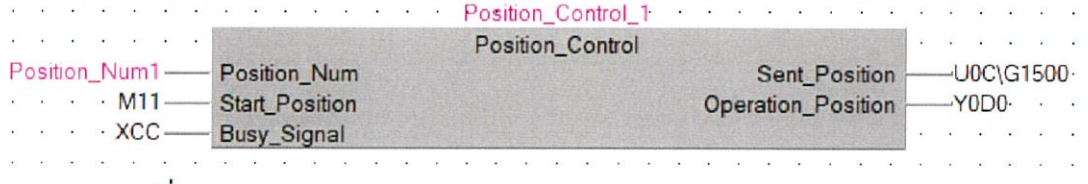
Set the target axis (partner axis) for operations under the 2-axis interpolation control.
0: Selects the axis 1 as the target axis (partner axis).
1: Selects the axis 2 as the target axis (partner axis).
2: Selects the axis 3 as the target axis (partner axis).
3: Selects the axis 4 as the target axis (partner axis).

รูปที่ 3.52 การตั้งค่า Axis to be interpolated

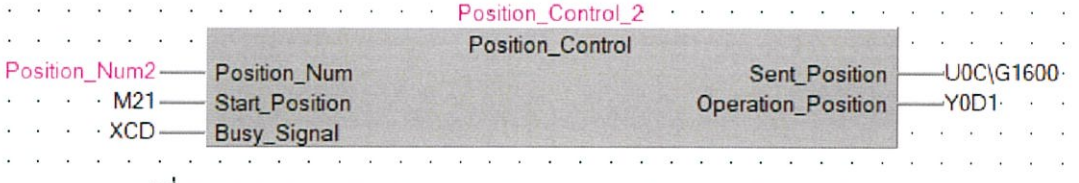
จากรูปที่ 3.52 เป็นการตั้งค่าแกนตามที่จะมาทำงานร่วม นั่นคือ หากตั้งค่าให้พารามิเตอร์ Axis to be interpolated มีค่าเท่ากับ 0, 1, 2 และ 3 แกนที่จะมาทำงานร่วมคือแกนที่ 1, 2, 3 และ 4 ตามลำดับ

3.12.3 การเขียนฟังก์ชันบล็อคควบคุมการทำงานแบบ Automatic Position Control

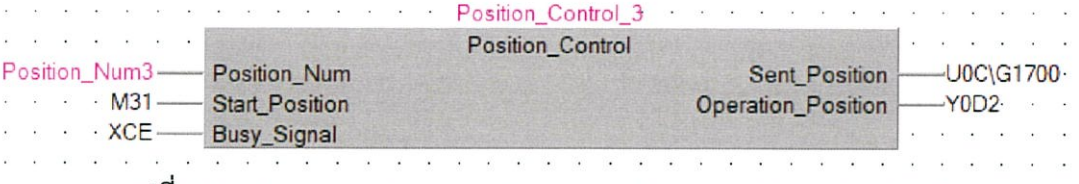
การเขียนฟังก์ชันบล็อคควบคุมการทำงานแบบ Automatic Position Control นั้น สามารถดำเนินการได้ตามแผนผังการทำงานแบบ Automatic Position Control ในหัวข้อที่ 3.11.1 รูปที่ 3.44 โดยตัวอย่างฟังก์ชันบล็อคสำหรับการควบคุมการทำงานแบบ Automatic Position Control ของแต่ละแกน แสดงดังรูปที่ 3.53-3.56



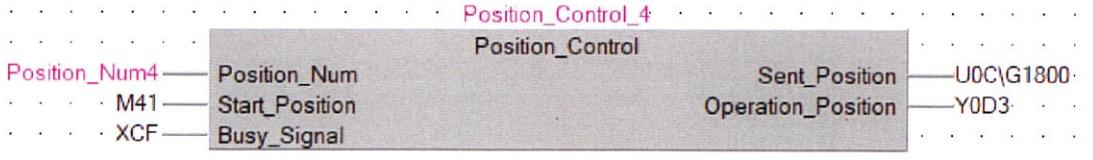
รูปที่ 3.53 Automatic Position Operation Function Block for Axis1



รูปที่ 3.54 Automatic Position Operation Function Block for Axis2



รูปที่ 3.55 Automatic Position Operation Function Block for Axis3



รูปที่ 3.56 Automatic Position Operation Function Block for Axis4

3.12.4 การทดลองการหา Software Stroke Limit

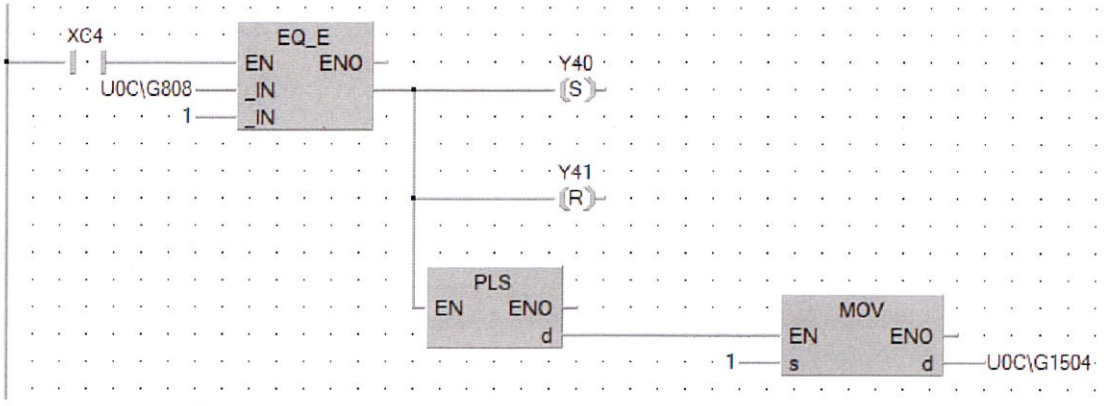
สำหรับการทดลองเพื่อหา Software Stroke Limit นั้นก็จะใช้การเคลื่อนที่แบบ Jog Operation และ Automatic Position Control ในการเคลื่อนที่หาว่าระยะต่ำสุดและสูงสุดที่แต่ละแกนของหุ่นยนต์สามารถทำงานได้นั้นอยู่ในช่วงใด โดยเริ่มจากการตั้งตำแหน่งอ้างอิงให้แก่หุ่นยนต์ เช่น กำหนดว่าจุดไหนคือตำแหน่ง 0 องศา โดยการสั่งให้หุ่นยนต์เคลื่อนที่ไปตาม Position No.9001 ด้วยฟังก์ชั่น Automatic Position Control จากนั้นเคลื่อนหุ่นยนต์ด้วย Jog Operation ไปในทิศทาง Forward และ reverse เพื่อหาว่าตำแหน่งที่หุ่นยนต์เคลื่อนที่ไปได้ต่ำสุดและสูงสุดคือตำแหน่งใด ตำแหน่งนั้นคือระยะต่ำสุดและสูงสุดของการทำงานตามลำดับ จากนั้นนำผลจากการทดลองดังกล่าวไปตั้งค่าพารามิเตอร์ ผลการทดลองแสดงดังตารางที่ 3.10

ตารางที่ 3.10 ผลการทดลองหา Software Stroke Limit

	Axis1	Axis2	Axis3	Axis4
Stroke Upper Limit	225.0	135.0	180.0	359.0
Stroke Lower Limit	317.0	230.0	0.0	0.0

3.13 ฟังก์ชันบล็อคการทำงานโหมด M-code Event Selection ด้วย GX Works2

การดำเนินงานของโหมด M-code Event Selection คือ เมื่อมีคำสั่งเปิดการทำงานจาก Automatic Position Control ให้มีการใช้งานในโหมด M-code Event Selection จากการตั้งค่าในส่วนของพารามิเตอร์ M-code ตามตารางที่ 3.10 ในส่วนของ M-code กรณีที่ 2 จะทำงานแบบ WITH MODE หรือ AFTER MODE นั้นขึ้นรูปกับการตั้งค่า M-code ON Signal output timing ในข้อที่3.7.2.2 ข้อที่ 3 ตัวอย่าง การเขียนโปรแกรมการทำงานโหมด Event แสดงดังรูปที่ 3.57



รูปที่ 3.57 การเขียนโปรแกรมควบคุมแบบ M-code Event Selection

ตัวแปรสำคัญที่ควรทราบ

1) ตัวแปร XC4 คือ สัญญาณ Digital ที่เข้ามาเพื่อเริ่มการทำงานของ Event

2) ตัวแปร U0C\G808 คือ Event ที่ต้องการทำงาน ซึ่งค่า Default จะมีค่าเท่ากับ 0

3) ตัวแปร U0C\G1504 คือคำสั่งปิดใช้งาน M-code

ลักษณะการทำงาน

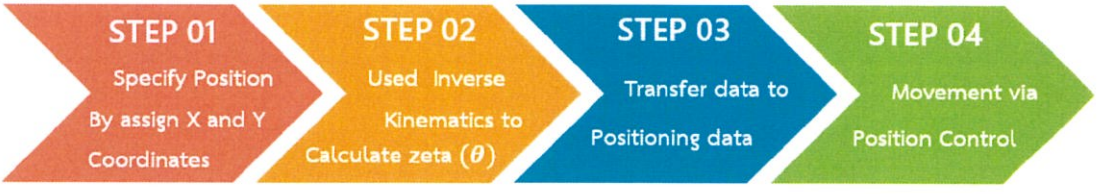
เมื่อมีสัญญาณ Digital เริ่มการทำงานในโหมด M-code Event Selection จากตัวแปร XC4 Function Block EQ_E จะเปรียบเทียบหา Event ที่ต้องการทำงาน ว่าใน ตัวแปร U0C\G808 มีค่าเท่ากับเท่ากับค่าที่กำหนดหรือไม่ หากเท่าจะดำเนินการต่อไป เช่น จากรูปที่ 3.57 เป็นการทำงานของ Event1 เมื่อตัวแปร U0C\G808 มีค่าเท่ากับ 1 โดยจะสั่งให้ Y40 ทำงานและ Y41 หยุดทำงาน จากนั้น Function Block PLS ส่งค่า Pulse ไป Reset คำสั่งการทำงานโหมด Event โดยการ Set U0C\G1504 ให้เป็น 1 เพื่อปิดคำสั่งโหมด M-code Event Selection ถือเป็นการเสร็จสิ้นการทำงาน Position นั้น แล้วดำเนินงานให้ Automatic Position Control ในตำแหน่งต่อไปทำงาน

3.14 ฟังก์ชันบล็อคการทำงานแบบ Coordinate Movement Control ด้วย GX Works2

ฟังก์ชันบล็อคควบคุมการทำงานแบบ Coordinate Movement Control นั้นเป็นฟังก์ชันบล็อคที่นำ ฟังก์ชันบล็อคคำนวณ Kinematics มาคำนวณและป้อนคำสั่ง สั่งให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่งต่าง ๆ ใน Work Area ด้วยการระบุตำแหน่งเป็นพิกัด X และ Y แทนการระบุด้วยองศา นอกจากนี้ยังหมายถึงการสั่งให้หุ่นยนต์เคลื่อนที่ไปทางแกน X หรือแกน Y ทีละหน่วยตามที่กำหนด ฟังก์ชันบล็อคควบคุมการทำงานแบบ Coordinate Movement Control นั้นจัดเป็นโปรแกรมที่สำคัญมากสำหรับใช้ในช่วงทดสอบการเคลื่อนที่ของหุ่นยนต์ว่าสามารถเคลื่อนที่ไปยังตำแหน่งต่าง ๆ ใน Work Area ได้ตามคำสั่งหรือไม่ ก่อนที่จะให้หุ่นยนต์เคลื่อนที่ไปตาม Trajectory Path หรือโหมด

Trajectory-path Movement Control อีกทั้งยังเป็นโปรแกรมสามารถตรวจสอบความถูกต้องของสมการ Kinematics ได้อีกด้วยเนื่องจากสามารถวัดระยะหรือสังเกตพิกัดจริงที่หุ่นยนต์เคลื่อนที่ไปได้ว่าตรงกับที่สั่งการหรือตรงกับที่อ่านค่าได้หรือไม่

กระบวนการทำงานของฟังก์ชันบล็อคควบคุมการทำงานแบบ Coordinate Movement Control แสดงดังนี้



รูปที่ 3.58 กระบวนการของฟังก์ชันบล็อกควบคุมการทำงานแบบ Coordinate Movement Control

จากรูปที่ 3.58 อธิบายถึงกระบวนการทำงานของฟังก์ชันบล็อกควบคุมการทำงานแบบ Coordinate Movement Control ได้ ดังนี้

ขั้นตอนที่ 1 ระบุพิกัด X และ Y ที่ต้องการให้หุ่นยนต์เคลื่อนที่

ขั้นตอนที่ 2 นำพิกัดที่ระบุมาเข้าสมการ Inverse Kinematics เพื่อคำนวณออกมาเป็น θ_1 และ θ_2

ขั้นตอนที่ 3 นำข้อมูลที่ได้จัดเรียงเข้า Positioning Data ของ Motion Buffer Memory

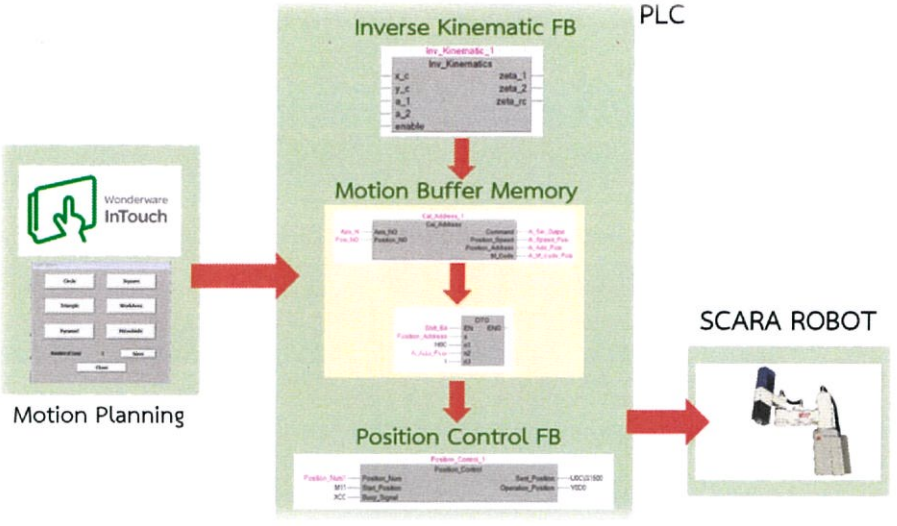
ขั้นตอนที่ 4 สั่งการให้หุ่นยนต์เคลื่อนที่ด้วยโปรแกรม Automatic Position Control

3.15 ฟังก์ชันบล็อคควบคุมการทำงานแบบ Trajectory-path Movement Control

จากรูปที่ 3.59 แสดงถึงกระบวนการสร้างฟังก์ชันบล็อคควบคุมการทำงานแบบ Trajectory-path Movement Controlซึ่งมีกระบวนการสร้างดังนี้

1) รับข้อมูล Trajectory Path จากระบบจัดการข้อมูล Trajectory Path จากนั้นนำข้อมูล Trajectory Path ที่ได้รับมาคำนวณผ่านฟังก์ชันบล็อค Inverse Kinematics ที่สร้างขึ้นเพื่อแปลงข้อมูลรูปพิกัด X และ Y ให้เป็นข้อมูลในรูปองศา (θ) เพื่อนำข้อมูลที่ได้ส่งให้กับ Positioning Data สำหรับการควบคุมต่อไป

2) สำหรับการส่งข้อมูลให้แก่ Positioning Data นั้นจำเป็นต้องมีการคำนวณ Address ของข้อมูลที่จะส่งไปก่อนด้วยการสร้างฟังก์ชันบล็อค Calculate Address โดยใช้สมการที่ 3.23-3.28 แล้วจึงนำข้อมูลที่ผ่านการคำนวณจากฟังก์ชันบล็อค Inverse Kinematics ส่งเข้าไปใน Positioning Data ตาม Address ที่คำนวณได้จากฟังก์ชันบล็อค Calculate Address



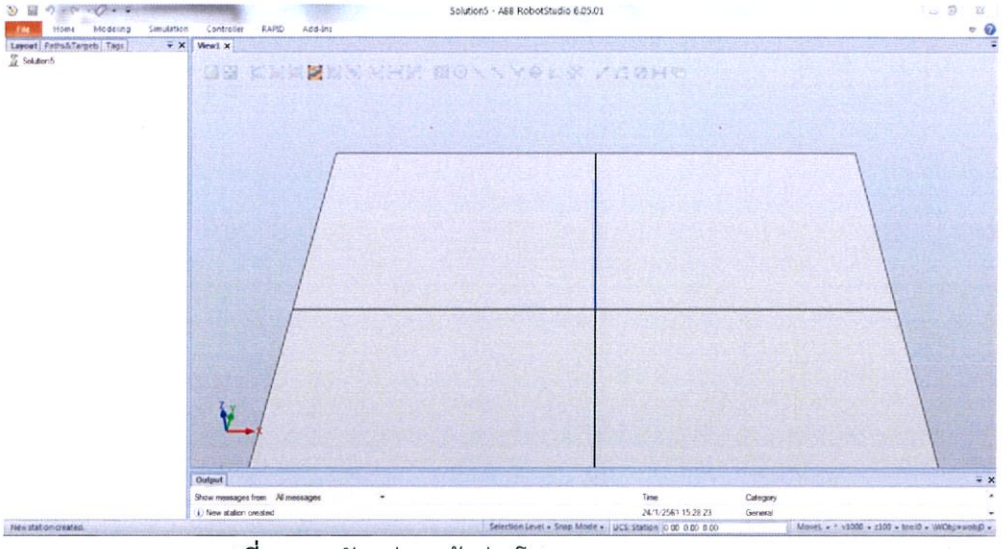
รูปที่ 3.59 การสร้างฟังก์ชันบล็อคควบคุมการทำงานแบบ Trajectory-path Movement Control

3) เมื่อข้อมูลทั้งหมดถูกส่งเข้าไปใน Positioning Data เรียบร้อยแล้วผู้ใช้งานสามารถสั่งให้หุ่นยนต์เคลื่อนที่ตาม Trajectory Path ได้จากการสั่งผ่านฟังก์ชันบล็อค Automatic Position Control ที่สร้างขึ้นหุ่นยนต์ก็จะสามารถเคลื่อนที่ตาม Trajectory Path ที่สร้างขึ้นได้

3.16 การสร้าง Trajectory Path หุ่นยนต์ด้วย RobotStudio

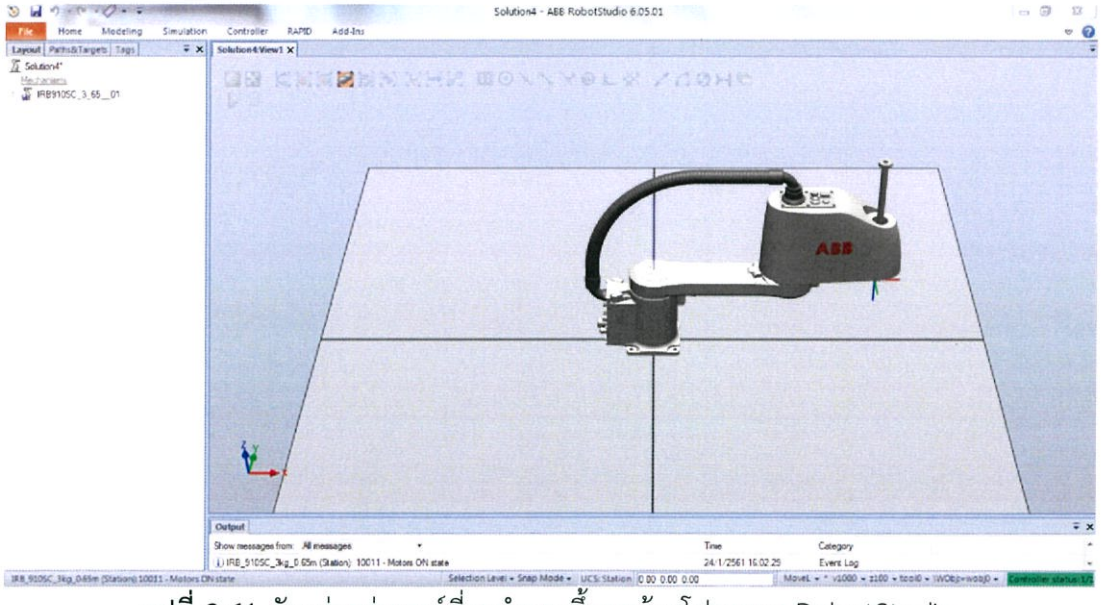
3.16.1 การสร้าง Trajectory Path

ในหัวข้อนี้จะกล่าวถึงการสร้าง Trajectory Pathให้แก่หุ่นยนต์ ซึ่งการเคลื่อนที่ของหุ่นยนต์แต่ละประเภทจะไม่เหมือนกันขึ้นอยู่กับประเภทของงาน ในปริญญานิพนธ์นี้เราจะสร้างเส้นทางการเดินให้แก่หุ่นยนต์ประเภท SCARA ด้วยโปรแกรม RobotStudio โดยมีหน้าต่างโปรแกรมแสดงดังรูปที่ 3.60



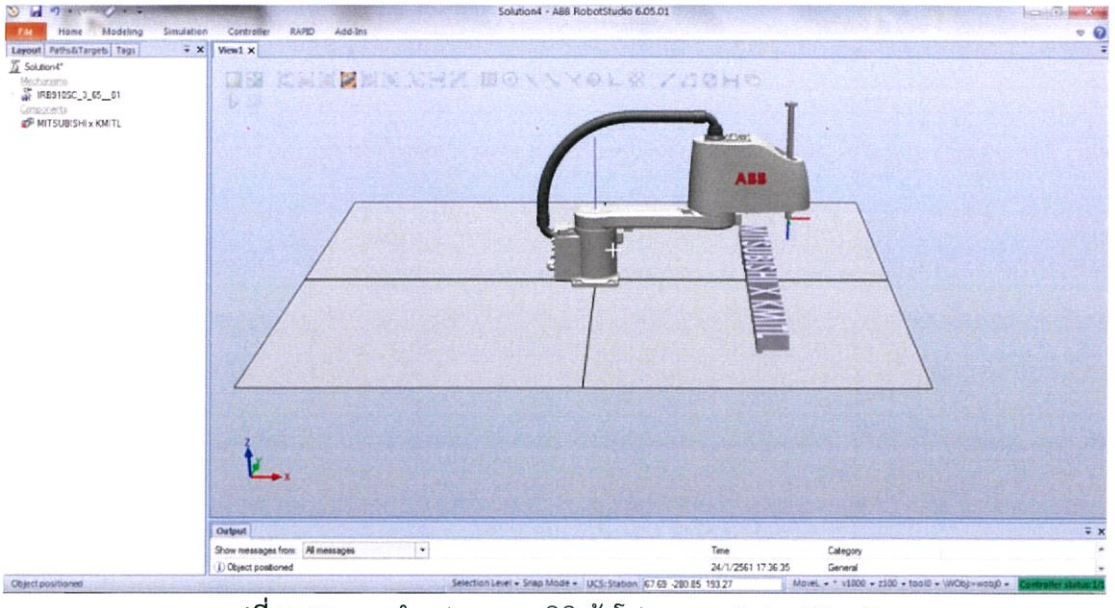
รูปที่ 3.60 ตัวอย่างหน้าต่างโปรแกรม RobotStudio

ตัวโปรแกรม RobotStudio สามารถจำลองหุ่นยนต์ที่ใช้งานเพื่อนำมาสร้างเส้นทางในการเคลื่อนที่ของหุ่นยนต์ในรูปพิกัดที่เรียงต่อกัน แสดงดังรูปที่ 3.61

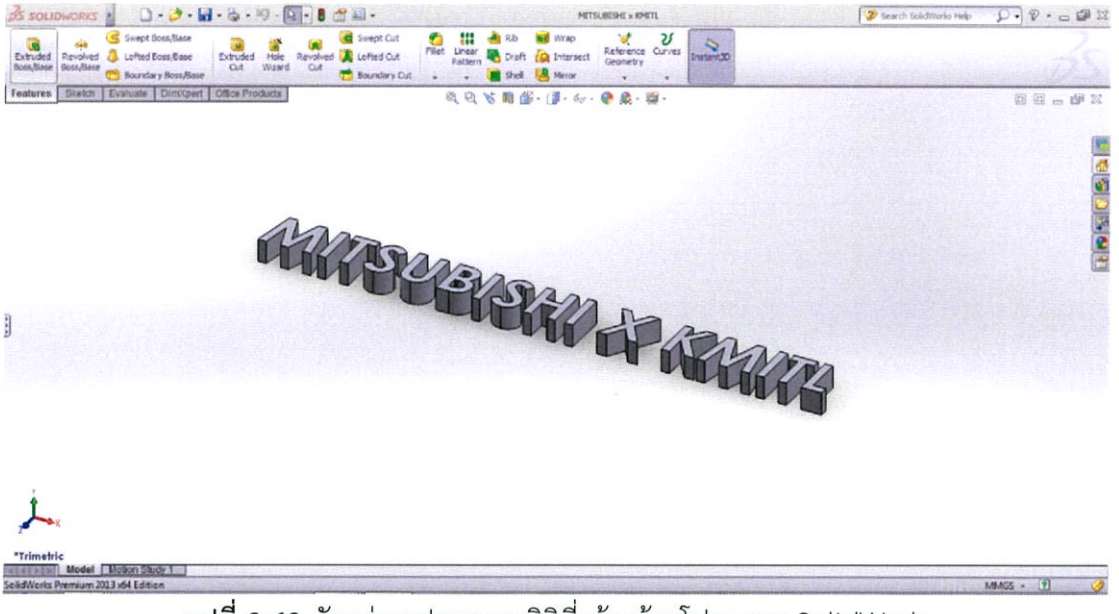


รูปที่ 3.61 ตัวอย่างหุ่นยนต์ที่ถูกจำลองขึ้นมาด้วยโปรแกรม RobotStudio

การสร้างวิถีการเคลื่อนที่สามารถสร้างจากรูปทรงสามมิติ ซึ่งการสร้างรูปทรงสามมิตินี้สามารถสร้างมาจากโปรแกรมต่าง ๆ แล้วนำมาลงในโปรแกรม RobotStudio ที่แสดงในรูปที่ 3.62 แต่ในปริญญานิพนธ์นี้จะเป็นการสร้างรูปทรงสามมิติด้วยโปรแกรม SolidWork รูปที่ 3.63

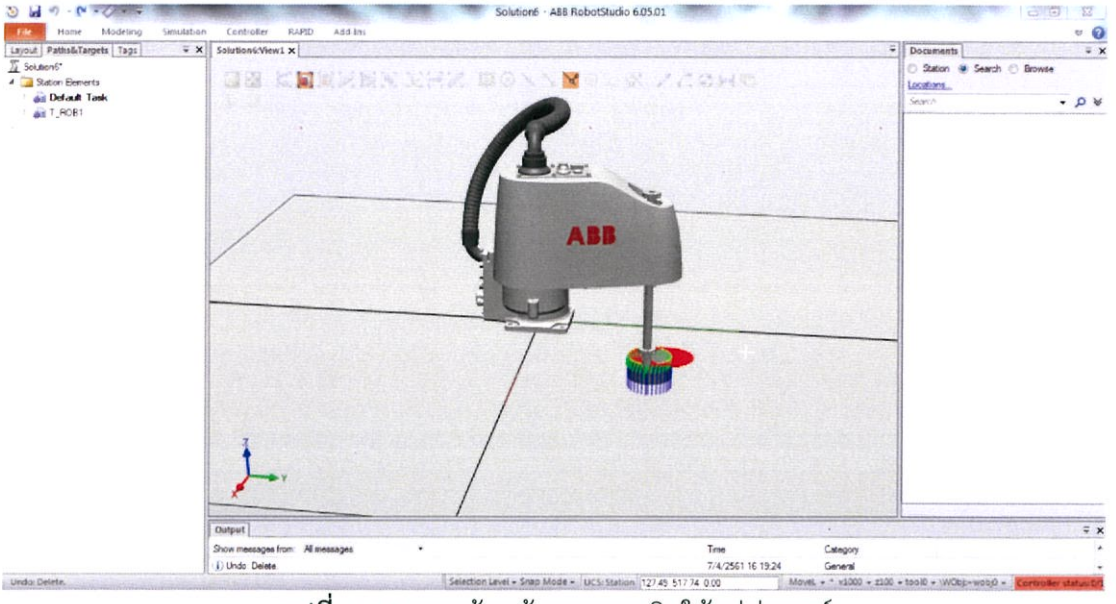


รูปที่ 3.62 การนำรูปทรงสามมิติเข้าโปรแกรม RobotStudio



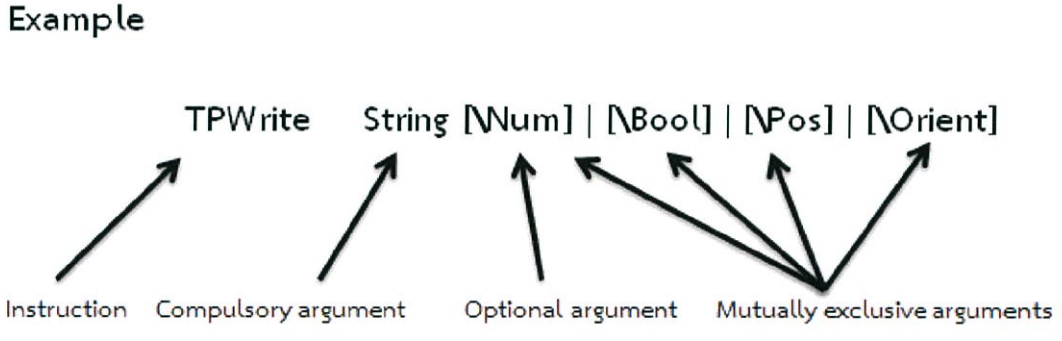
รูปที่ 3.63 ตัวอย่างรูปทรงสามมิติที่สร้างด้วยโปรแกรม SolidWork

การสร้างวิถีการเคลื่อนที่ให้แก่หุ่นยนต์สามารถทำได้หลากหลายรูปแบบ แต่ในส่วนของปริญญานิพนธ์นี้จะใช้วิธีการกำหนดให้หุ่นยนต์เลื่อนที่ไปในแต่ละจุดเรียงต่อกัน โดยโปรแกรม RobotStudio ซึ่งมีความสามารถในการสร้างวิถีการเดินได้ง่ายและรวดเร็ว เหมือนที่แสดงในรูปที่ 3.64



รูปที่ 3.64 การสร้างเส้นทางการเดินให้แก่หุ่นยนต์

หลังจากที่ได้ทำการสร้างวิถีการเคลื่อนที่ต้องการครบแล้ว ขั้นต่อไปจะเป็นการรวมพิกัดที่ต้องการเดินเข้าด้วยกัน หรือการ Synchronize ข้อมูล พร้อมทั้งแปลงข้อมูลออกมาในรูปแบบของภาษา RAPID ที่มีพิกัดในการเคลื่อนที่ต้องการ ซึ่งเป็นข้อมูลที่ใช้กับหุ่นยนต์ของบริษัท เอบีบี โดยภาษา RAPID จะมีรูปแบบตามรูปที่ 3.65



รูปที่ 3.65 รูปแบบของภาษา RAPID

View1 | IRB_910SC_3kg_0.65m (Station) ×

T_ROB1/Module1 ×

```
MODULE Module1
    CONST robtarget Target_10:=[[0,650,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_20:=[[0,600,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_30:=[[0,550,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_40:=[[0,500,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_50:=[[0,450,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_60:=[[0,400,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_70:=[[0,350,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_80:=[[0,300,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_90:=[[50,300,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_100:=[[100,300,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_110:=[[150,300,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_120:=[[150,250,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_130:=[[200,250,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_140:=[[250,250,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_150:=[[250,200,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_160:=[[250,150,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_170:=[[300,150,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_180:=[[300,100,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_190:=[[300,50,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_200:=[[300,0,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_210:=[[300,-50,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_220:=[[300,-100,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_230:=[[300,-150,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_240:=[[300,-200,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_250:=[[250,-200,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_260:=[[250,-250,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_270:=[[200,-250,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_280:=[[150,-250,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_290:=[[150,-300,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_300:=[[100,-300,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_310:=[[50,-300,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Target_320:=[[0,-300,50],[0,-0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
```

รูปที่ 3.66 ข้อมูลภาษา RAPID

ข้อมูลที่ได้ออกมาจะถูกเก็บไว้เป็นไฟล์ที่แสดงข้อมูล robtarget ดังรูปที่ 3.66 ซึ่งจะแสดงพิกัด X, Y, Z ที่จุดปลายของวิถีการเคลื่อนที่ ไฟล์ที่ได้จะสามารถเปิดได้โดยโปรแกรม Notepad หรือโปรแกรมอื่น ๆ โดยจะไม่มีการแบ่งคอลัมน์และแถวไว้ ดังนั้นเมื่อต้องการใช้งานข้อมูลภาษา RAPID จะต้องนำข้อมูลลงมาใส่ใน Microsoft Excel โดยการนำข้อมูลเข้ามาเลือกจากข้อความ และกำหนดแบ่งคอลัมน์ จะได้พิกัดในการเคลื่อนที่เพื่อส่งต่อไปให้กับระบบฐานข้อมูล ตามรูปที่ 3.67

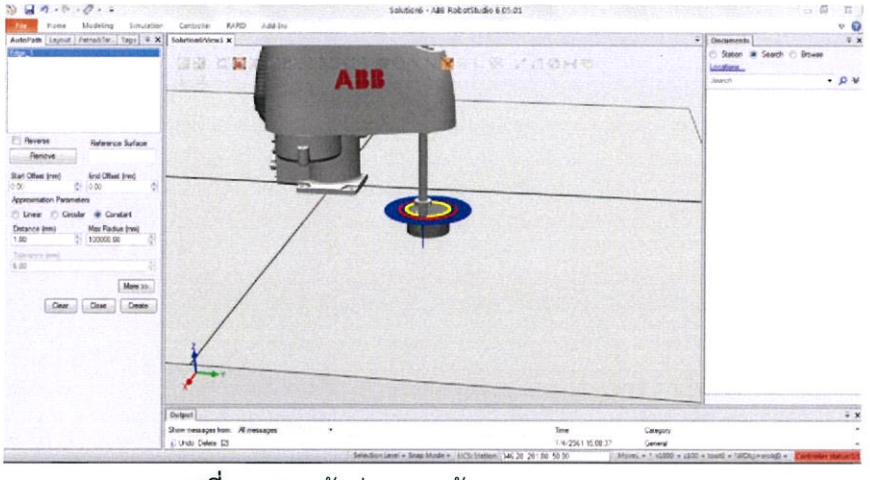
จากรูปที่ 3.67 ข้อมูลที่ได้จะออกมาอยู่ในรูปของพิกัด X, Y, Z แล้วจะทำการส่งค่าพิกัดนี้เข้าสู่ฐานข้อมูลเพื่อใช้ในการเก็บข้อมูลวิถีการเคลื่อนที่ก่อนจะส่งให้ระบบจัดการข้อมูล Trajectory Path เรียงคิวเพื่อส่งข้อมูลให้แก่ PLC

	A	B	C
1	x	y	z
2	426.2433	-110.546856	60
3	420.6168	-105.967506	60
4	410.6237	-105.594866	60
5	400.6303	-105.232399	60
6	390.6341	-104.962597	60
7	396.703	-103.520995	60
8	406.5125	-101.581809	60
9	416.2908	-99.4874419	60
10	425.8319	-97.2013779	60
11	419.2746	-92.4343922	60
12	409.5238	-90.2160129	60
13	399.7636	-88.0393396	60
14	389.9536	-86.1011369	60
15	397.4623	-85.6614029	60
16	407.4577	-85.3600047	60
17	417.4509	-84.9914339	60
18	426.2433	-83.4654919	60
19	419.5758	-80.414699	60
20	409.5844	-80.828337	60

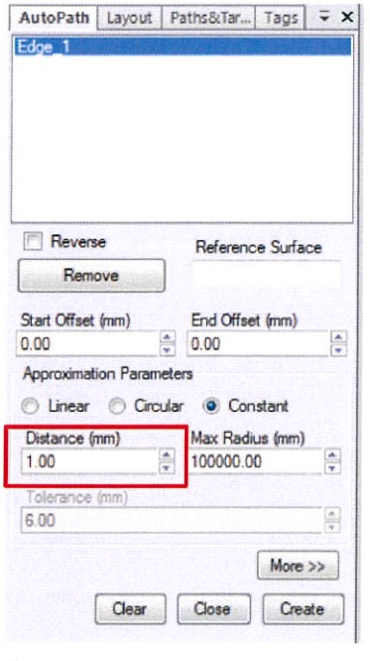
รูปที่ 3.67 ข้อมูลพิกัดการเคลื่อนที่จากโปรแกรม Microsoft Excel

3.16.2 การกำหนดค่า Resolution

การกำหนดค่าความละเอียด หรือ Resolution คือ การกำหนดความละเอียดของเส้นทาง Trajectory Path ที่สร้างว่าจากพิกัดจุดนึงไปยังอีกจุดนึง จะมีระยะทางเท่าไหร่ โดยระยะที่กำหนดนี้โปรแกรมจะนำไปคำนวณว่าใน Trajectory Path ที่สร้างนั้นจะมีพิกัดเรียงต่อกันกี่จุด หากระยะจากพิกัดจุดนึงไปยังอีกจุดนึงน้อย จำนวนพิกัดก็จะมาก และในทางกลับกัน หากระยะทางมากจำนวนพิกัดก็จะน้อย ซึ่งสำหรับโปรแกรม RobotStudio ที่ใช้นั้นสามารถกำหนดโดยการกำหนดค่า Distance ในส่วนของการสร้าง Trajectory Path ดังรูปที่ 3.68-3.69



รูปที่ 3.68 หน้าต่างการสร้าง Trajectory Path



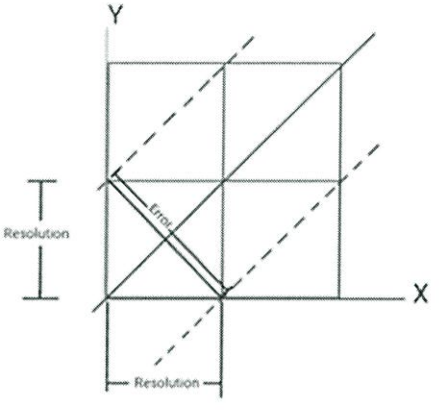
รูปที่ 3.69 การกำหนดค่า Resolution

3.16.3 การคำนวณหาค่า Resolution

การกำหนดค่า Resolution นั้นจะมักถูกคำนวณจากค่าความผิดพลาด (Error) ที่ยอมรับได้ ในที่นี้จะยกตัวอย่างจากคำนวณเพียง 2 รูปแบบ ดังนี้

1) Trajectory Path แบบเส้นตรง

สำหรับ Trajectory Path แบบเส้นตรงนั้นค่า Resolution ที่กำหนดนั้น หมายถึง ระยะขจัดที่เคลื่อนที่นั้นเท่ากับค่า Resolution ที่กำหนด ดังนั้น ค่า Resolution แบบเส้นตรงนั้น การกำหนดค่า Resolution คือ การกำหนดระยะทางตามแนวแกน X หรือ Y ที่มากสุดนั้นมีระยะทางเท่าไหร่ เช่น หากไม่มีการเคลื่อนที่ตามแนวแกน Y มีแค่การเคลื่อนที่ตามแนวแกน X ระยะทางตามแนวแกน X ก็จะมีค่าเท่ากับค่า Resolution ที่กำหนด ดังนั้นค่าความผิดที่เกิดขึ้นนั้นสามารถแสดงได้ดังรูปที่ 3.70



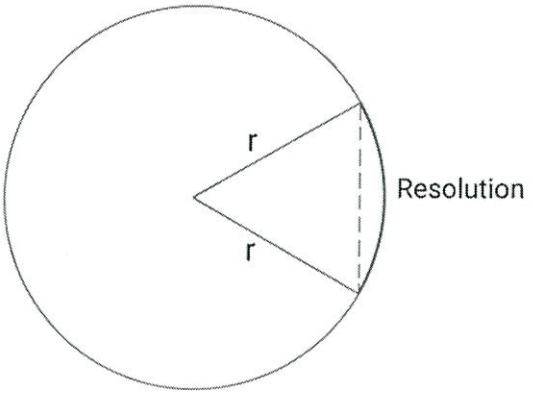
รูปที่ 3.70 ความสัมพันธ์ระหว่างค่าความผิดพลาดกับค่า Resolution แบบเส้นตรง

จากรูปที่ 3.70 การคำนวณหาค่า Resolution ของเส้นจากค่าความผิดพลาดที่ยอมรับได้นั้น สามารถหาได้จากสมการนี้

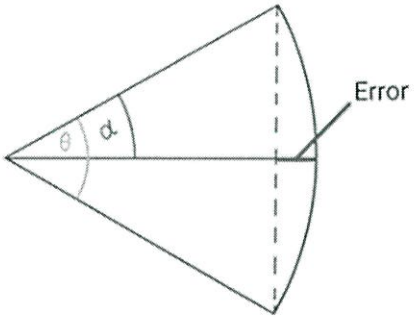
$$\text{Resolution} = \sqrt{\frac{\text{Error}^2}{2}} \qquad (3.29)$$

2) Trajectory Path แบบวงกลม

สำหรับ Trajectory Path แบบวงกลมนั้นค่า Resolution ที่กำหนดนั้น หมายถึง ระยะทางที่เคลื่อนที่ตามเส้นรอบวงของวงกลมจากจุดนึงไปยังอีกจุดนึง ดังนั้น ค่าความผิดของ คือ ระยะเส้นตั้งฉาก ณ จุดกึ่งจากของเส้นคอร์ดถึงเส้นรอบวงของวงกลม โดยเส้นคอร์ดนี้เกิดจากการลากเส้นตรงระหว่างจุดต่อจุดที่เคลื่อนที่ตาม Resolution แสดงดังรูปที่ 3.71-3.72



รูปที่ 3.71 Resolution ของรูปวงกลม



รูปที่ 3.72 ภาพขยายของรูปที่ 3.71

จากรูปที่ 3.72 สามารถแสดงความสัมพันธ์

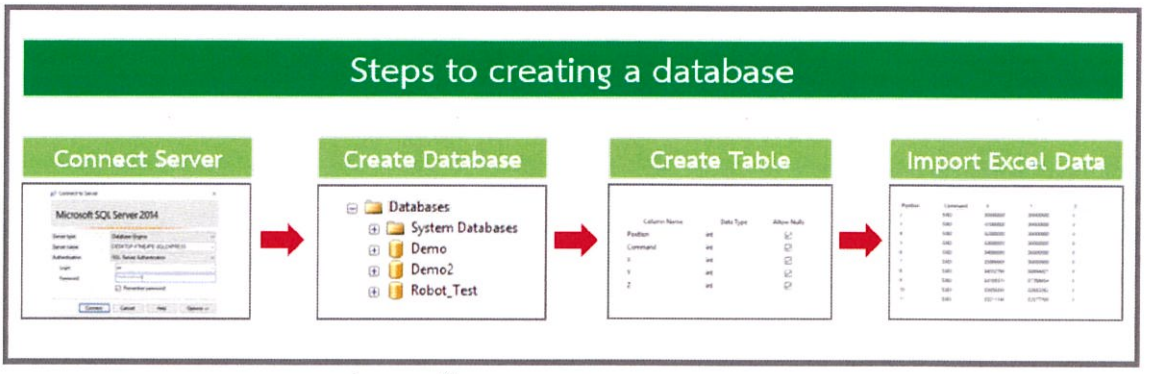
$$Error = r - (r*(\cos\alpha)) \tag{3.30}$$

โดยความสัมพันธ์ระหว่าง α และ θ คือ $2\alpha = \theta$ ซึ่ง θ เป็นมุมที่เกิดจากการเคลื่อนที่ตาม Resolution จากจุดนึงไปอีกจุดนึง ดังนั้น

$$Resolution = r\,\theta \tag{3.31}$$

3.17 การสร้างและจัดเก็บข้อมูลของระบบฐานข้อมูล

การสร้างระบบฐานข้อมูลเพื่อจัดเก็บ Trajectory Path ที่สร้างจาก RobotStudio โดยใช้โปรแกรม Microsoft SQL Server มีขั้นตอนแสดงดังรูปที่ 3.73



รูปที่ 3.73 ขั้นตอนการสร้างระบบฐานข้อมูล

จากรูปที่ 3.73 แสดงขั้นตอนการสร้างระบบฐานข้อมูล ที่มีรายละเอียดการสร้าง ดังนี้

1) เชื่อมต่อกับเซิร์ฟเวอร์ คือการเชื่อมต่อกับเซิร์ฟเวอร์ที่จัดเก็บฐานข้อมูล โดยในที่นี้คือคอมพิวเตอร์ ข้อมูลการเชื่อมต่อกับเซิร์ฟเวอร์นอกจากจะใช้ในการเชื่อมต่อเข้าไปจัดการฐานข้อมูลแล้ว ยังใช้ในการเชื่อมต่อเซิร์ฟเวอร์เพื่อให้สามารถ Query ข้อมูลผ่านระบบจัดการข้อมูล Trajectory Path ซึ่งจะกล่าวในหัวข้อที่ 3.21

2) สร้างฐานข้อมูล เป็นสร้างและตั้งชื่อฐานข้อมูลที่จะใช้ในการเก็บข้อมูล Trajectory Path

3) สร้างตาราง จะเป็นการสร้างตารางที่ไว้เก็บข้อมูล ซึ่งภายในจะประกอบไปด้วยหลายคอลัมน์

4) นำเข้าข้อมูล Trajectory Path ที่อยู่ในรูปไฟล์ .xlsx และข้อมูลต่าง ๆ ที่ใช้สำหรับการควบคุมการทำงานของหุ่นยนต์มาใส่ในตารางที่สร้างขึ้น

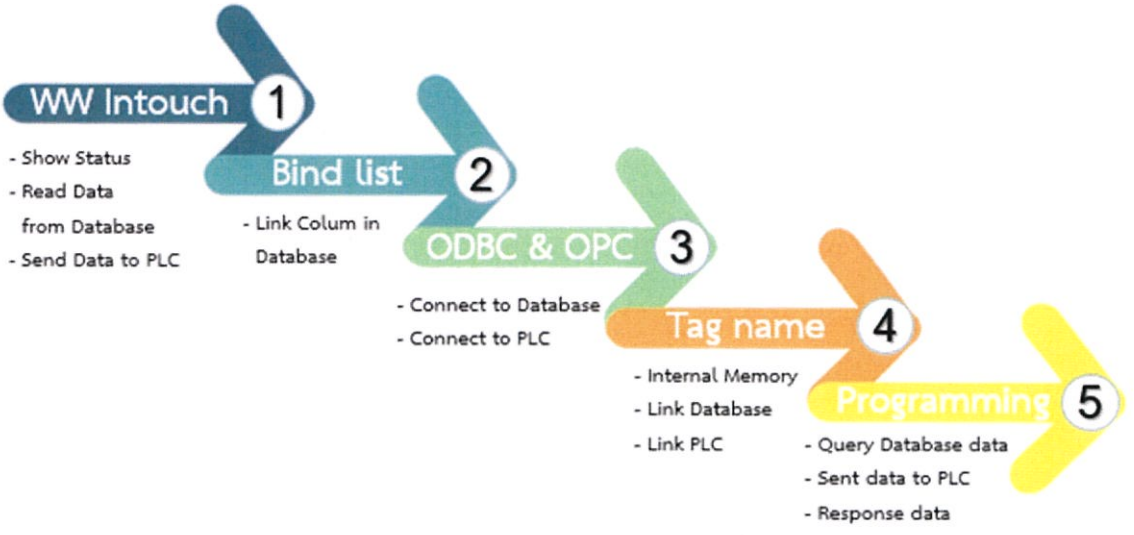
ตัวอย่างตารางที่ใช้เก็บข้อมูล Trajectory Path แสดงดังรูปที่ 3.74

Position	Command	X	Y	Z
2	5383	30000000	30000000	0
3	5383	31000000	30000000	0
4	5383	32000000	30000000	0
5	5383	33000000	30000000	0
6	5383	34000000	30000000	0
7	5383	35000000	30000000	0
8	5383	34552786	30894427	0
9	5383	34105573	31788854	0
10	5383	33658359	32683282	0
11	5383	33211146	33577709	0

รูปที่ 3.74 ตัวอย่างตารางที่ใช้เก็บข้อมูล Trajectory Path

3.18 การสร้างระบบจัดการข้อมูล Trajectory Path

3.18.1 ขั้นตอนการสร้างระบบจัดการข้อมูล Trajectory Path

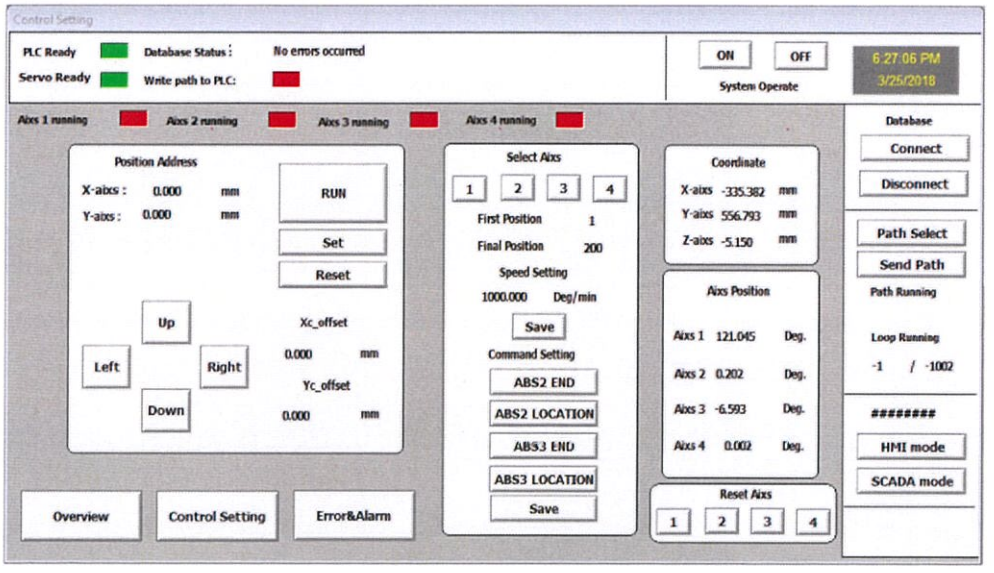


รูปที่ 3.75 ขั้นตอนการสร้างระบบจัดการข้อมูล Trajectory Path

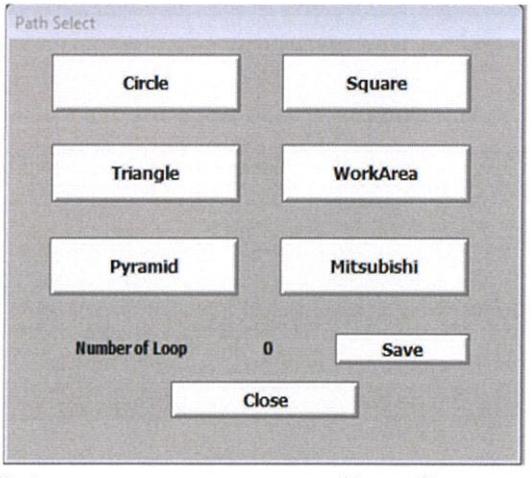
จากรูปที่ 3.75 แสดงถึงขั้นตอนการสร้างระบบจัดการข้อมูล Trajectory Path ซึ่งจะมีทั้งหมด 5 ขั้นตอนด้วยกัน ดังนี้

ขั้นตอนที่ 1 คือการสร้างหน้าต่างระบบจัดการข้อมูล Trajectory Path ให้มีแอพพลิเคชันในการแสดงค่าสถานะต่าง ๆ ที่จำเป็นต่อการควบคุมและการ monitoring รวมไปถึงในส่วนของการรับข้อมูลจากระบบฐานข้อมูลและการส่งข้อมูลไปยัง PLC ตัวอย่างหน้าต่างระบบจัดการข้อมูล Trajectory Path แสดงดังรูปที่ 3.76-3.78

รูปที่ 3.76 หน้าต่าง Overview ของระบบจัดการข้อมูล Trajectory Path



รูปที่ 3.77 หน้าต่าง Control Setting ของระบบจัดการข้อมูล Trajectory Path



รูปที่ 3.78 หน้าต่าง Path Select ของระบบจัดการข้อมูล Trajectory Path

ขั้นตอนที่ 2 จะเป็นการสร้าง Blind list ที่เชื่อมต่อกับ Tag name ประเภทที่เชื่อมต่อกับระบบฐานข้อมูล ซึ่งทำหน้าเปรียบเสมือน pointer ที่กำหนดคอลัมน์ที่ต้องการเรียกข้อมูลจากระบบฐานข้อมูล

ขั้นตอนที่ 3 สร้างสื่อกลางในการสื่อสารระหว่างระบบจัดการข้อมูล Trajectory Path, ระบบฐานข้อมูล และ PLC โดยสื่อกลางระหว่างระบบฐานข้อมูลและระบบจัดการข้อมูล Trajectory Path จะถูกเรียกว่า ODBC และสื่อกลางระหว่างระบบจัดการข้อมูล Trajectory Path และ PLC จะถูกเรียกว่า OPC ในขั้นตอนนี้ต้องทำการตั้งค่าพารามิเตอร์ต่าง ๆ ที่จำเป็นต่อการเชื่อมต่อสื่อสาร

ขั้นตอนที่ 4 สร้าง Tag name ให้แก่อุปกรณ์ในระบบจัดการข้อมูล Trajectory Path เพื่อให้ระบบจัดการข้อมูล Trajectory Path สามารถรับข้อมูลมาแสดง ซึ่งจะมี Tag ทั้งหมด 3 ประเภท ดังนี้

- Internal Memory คือ Tag name ที่ถูกกำหนดขึ้นเพื่อเก็บข้อมูลหรือค่าภายในระบบจัดการข้อมูล Trajectory Path อาจใช้เพื่อการคำนวณ การกำหนดค่า หรือ การเขียนคำสั่งที่ใช้แค่ภายในระบบจัดการข้อมูล Trajectory Path

- เชื่อมต่อกับระบบฐานข้อมูล จะเป็นการสร้าง Tag name ที่ใช้ในการส่งข้อมูลระหว่างระบบฐานข้อมูลกับระบบจัดการข้อมูล Trajectory Path

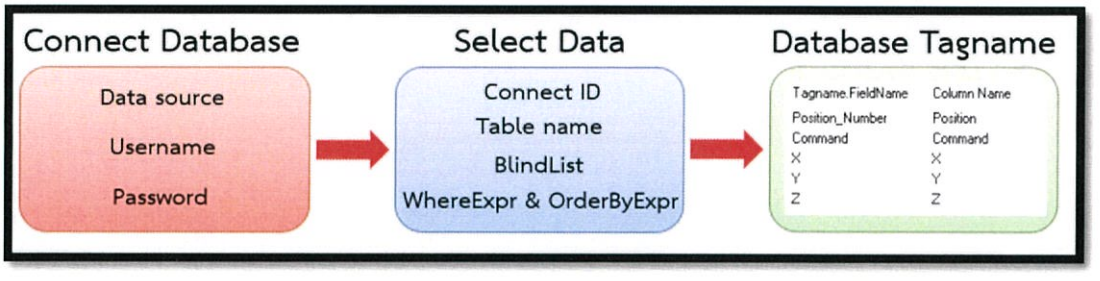
- เชื่อมต่อ PLC จะเป็น Tag name สำหรับเชื่อมต่อและรับส่งข้อมูลระหว่าง PLC กับระบบจัดการข้อมูล Trajectory Path

ขั้นตอนที่ 5 คือการเขียนโปรแกรมโดยโปรแกรมสำหรับระบบจัดการข้อมูล Trajectory Path จะแบ่งเป็น 2 ส่วนด้วยกัน ดังนี้

- โปรแกรมการ Query ข้อมูล Trajectory Path จากระบบฐานข้อมูล

- โปรแกรมสำหรับส่งข้อมูล Trajectory Path ให้ PLC

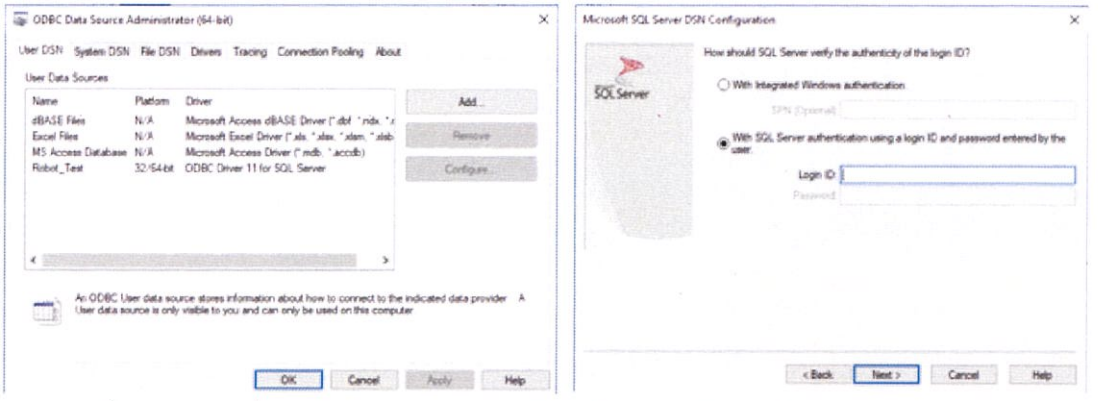
3.18.2 การเขียนโปรแกรมการ Query ข้อมูล Trajectory Path จากระบบฐานข้อมูล



รูปที่ 3.79 การทำงานของโปรแกรมการ Query ข้อมูล Trajectory Path จากระบบฐานข้อมูล

จากรูปที่ 3.79 แสดงถึงการทำงานของโปรแกรม Query ข้อมูล Trajectory Path จากระบบฐานข้อมูล โดยมีการทำงานดังนี้

1). เชื่อมต่อข้อมูลกับระบบฐานข้อมูล ซึ่งในส่วนนี้จะเป็นการเชื่อมต่อกับ ODBC โดยการใส่ข้อมูลที่ตั้งค่าไว้ใน ODBC (รูปที่ 3.80) เช่น ชื่อของ User Data Sources, Username และ Password สำหรับโค้ดคำสั่งของการเชื่อมต่อกับระบบฐานข้อมูล คือ โค้ดคำสั่ง ConnectString แสดงดังตารางที่ 3.11



รูปที่ 3.80 การเชื่อมต่อระหว่างระบบฐานข้อมูลกับระบบจัดการข้อมูล Trajectory Path

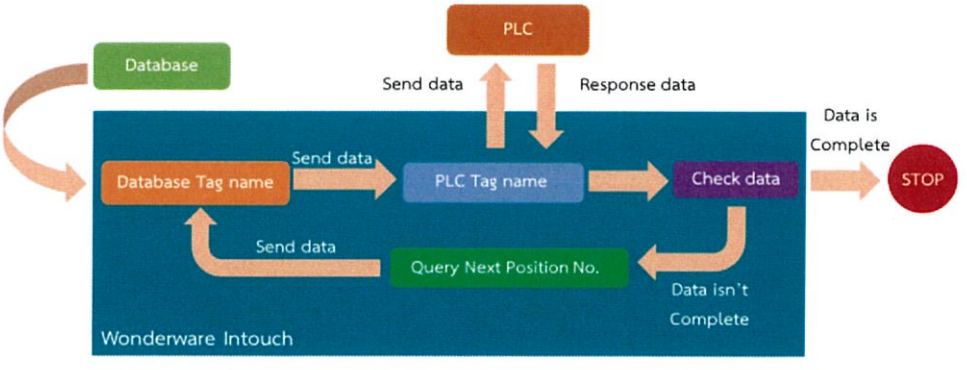
2). การใช้คำสั่ง Select สำหรับ Query ข้อมูลจากระบบฐานข้อมูล ซึ่งต้องมีการระบุข้อมูล ดังนี้

- Connect ID หมายถึง การบอกให้ทำการเชื่อมต่อกับระบบฐานข้อมูล
- Table Name หมายถึง ชื่อของตารางที่ต้องการ Query ข้อมูล
- BlindList หมายถึง ตัวแปรที่ชี้ไปยังคอลัมน์ของตารางในระบบฐานข้อมูลที่ต้องการ Query ข้อมูล
- WhereExpr หมายถึง การระบุตำแหน่งหรือข้อมูลที่ต้องการ Query ข้อมูลนี้อาจจะมีหรือไม่มีก็ได้
- OrderByExpr หมายถึง จะให้ข้อมูลที่ทำการ Query มีการเรียงลำดับด้วยข้อมูลของคอลัมน์ไหน เช่นเดียวกันกับ WhereExpr ข้อมูลส่วนนี้จะมีหรือไม่มีก็ได้

โค้ดคำสั่งของการ Select ข้อมูล คือ คำสั่ง Select และ First ดังตารางที่ 3.11

3). หลังจากที่ Query ข้อมูลแล้ว ข้อมูลดังกล่าวจะถูกเก็บที่ Tag Name ของระบบจัดการข้อมูล Trajectory Path ที่สร้างไว้สำหรับเชื่อมต่อกับระบบฐานข้อมูล นั่นคือ database Tag name

3.18.3 การเขียนโปรแกรมสำหรับส่งข้อมูล Trajectory Path ให้ PLC



รูปที่ 3.81 การทำงานของโปรแกรมสำหรับส่งข้อมูล Trajectory Path ให้ PLC

จากรูปที่ 3.81 แสดงถึงกระบวนการทำงานของโปรแกรมสำหรับส่งข้อมูล Trajectory Path ให้ PLC โดยมีลักษณะการทำงาน คือ หลังจากที่มีการ Query ข้อมูลจากระบบฐานข้อมูล ข้อมูลTrajectory Path ที่เก็บไว้ใน database Tag name จะถูกส่งให้กับ PLC Tag name เป็นการส่งข้อมูลให้กับ PLC เมื่อ PLC ได้รับข้อมูลจะมีการตอบรับข้อมูลกลับมา ต่อจากนั้นระบบจัดการข้อมูล Trajectory Path จะทำการตรวจเช็คว่าข้อมูลที่ทำการ Query มานั้นได้ส่งให้ PLC ครบหรือไม่ หากยังไม่ครบถ้วนก็จะทำการ Query ข้อมูล Position No. ถัดไปแล้วทำการส่งให้ PLC อีกครั้ง ระบบจะทำตามกระบวนการข้างต้นไปเรื่อย ๆ จนกว่าจะตรวจพบว่าข้อมูลที่ส่งให้กับ PLC ครบถ้วนแล้วเป็นการเสร็จสิ้นการทำงาน โดยโค้ดที่ใช้สำหรับ Query ข้อมูลลำดับถัดไปนั้นได้ให้คำสั่ง Next ดังตารางที่ 3.11

ตารางที่ 3.11 โค้ดคำสั่งการ Query

คำสั่ง	โค้ดคำสั่ง
ConnectString	ConnectString = "Data Source=xxx;User ID=xxx;Password=xxx;"
Select	ResultCode=SQLSelect (*ConnectionID, TableName, BindList, WhereExpr, OrderByExpr)*
First	ResultCode=SQLFirst(ConnectionID*)*
Next	ResultCode=SQLNext(ConnectionID)

บทที่ 4
การทดสอบตัวควบคุมหุ่นยนต์ที่นำเสนอ

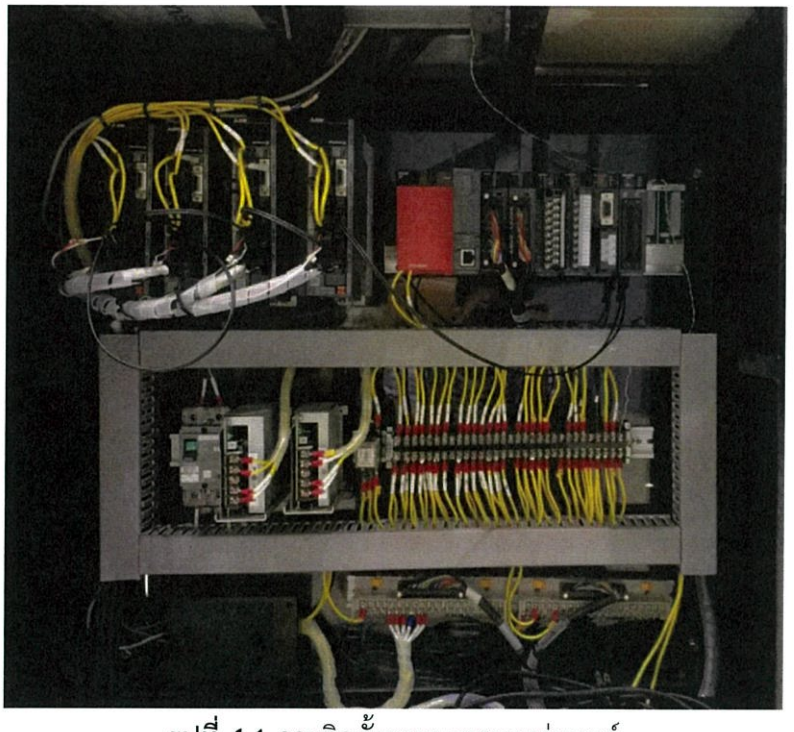
4.1 กล่าวนำ

ในบทที่ 4 นี้จะเป็นการนำเสนอผลการทดสอบตัวควบคุมหุ่นยนต์ที่นำเสนอที่ได้มีการกล่าวถึงการดำเนินงานสำหรับตัวควบคุมหุ่นยนต์ที่นำเสนอมาในบทที่ 3 โดยจะเป็นการทดลอบวัดผลการควบคุมหุ่นยนต์ในการควบคุมรูปแบบต่าง ๆ ที่ได้มีการจัดทำขึ้น เพื่อทดสอบความถูกต้องแม่นยำในการเคลื่อนที่ของหุ่นยนต์ว่าสอดคล้องกับข้อมูลในการเคลื่อนที่ ที่ได้ป้อนคำสั่งให้แก่หุ่นยนต์หรือไม่

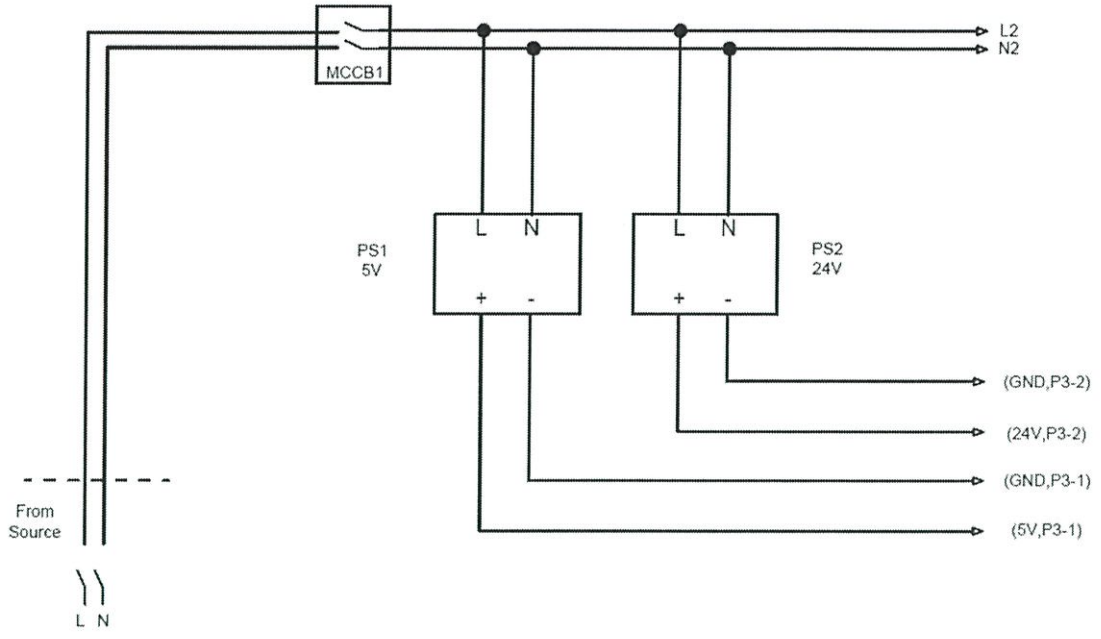
4.2 การดำเนินงาน

4.2.1 การติดตั้งอุปกรณ์ทาง Hardware

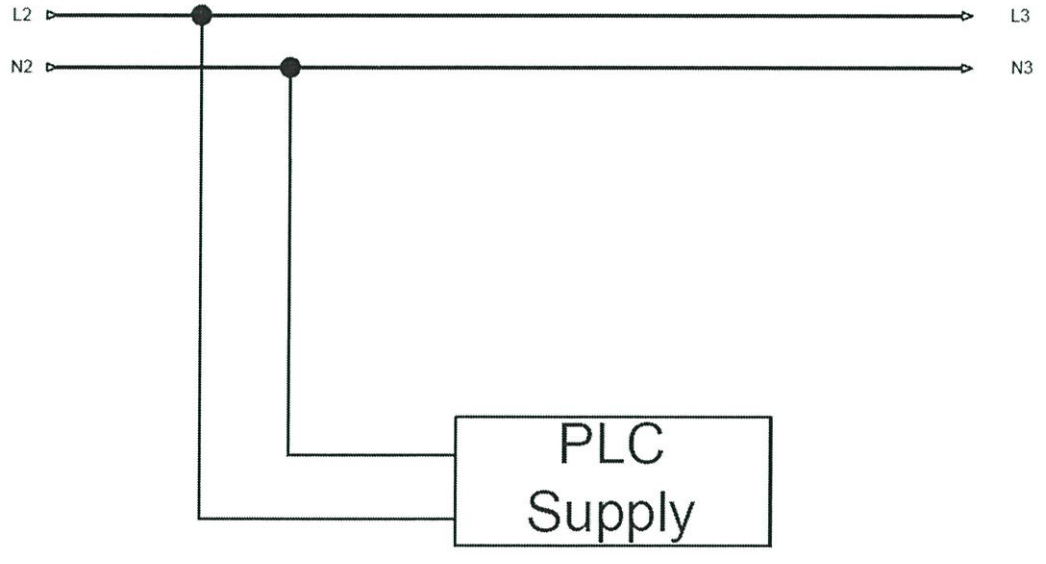
จากรูปที่ 4.1 เป็นการติดตั้งทาง Hardware ซึ่งใช้วงจรในรูปที่ 4.2-4.8 และมีทำการตรวจสอบความถูกต้องในการเชื่อมต่อของวงจรจากตารางที่ 4.1



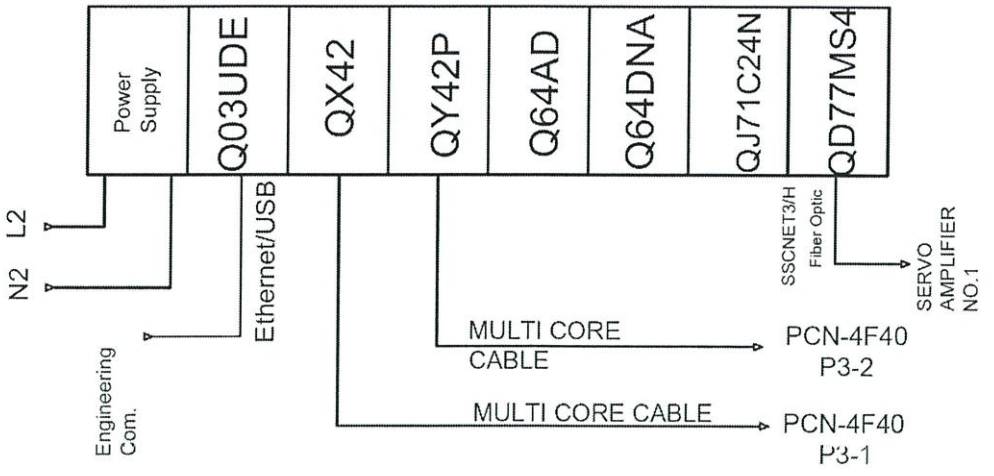
รูปที่ 4.1 การติดตั้งระบบควบคุมหุ่นยนต์



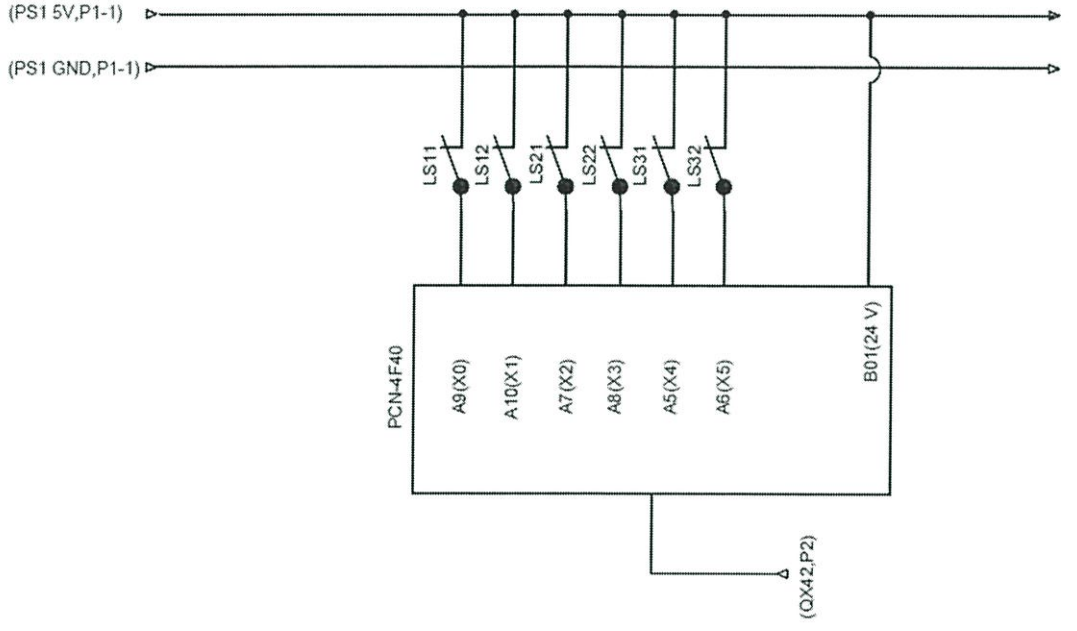
รูปที่ 4.2 วงจรไฟฟ้าภาคกำลังส่วนที่ 1



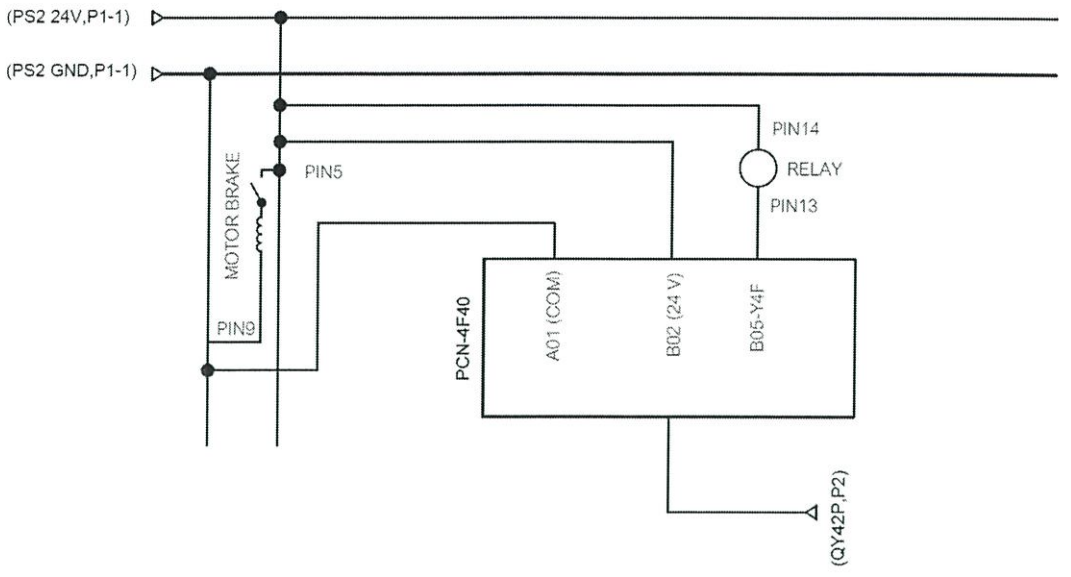
รูปที่ 4.3 วงจรไฟฟ้าภาคกำลังส่วนที่ 2



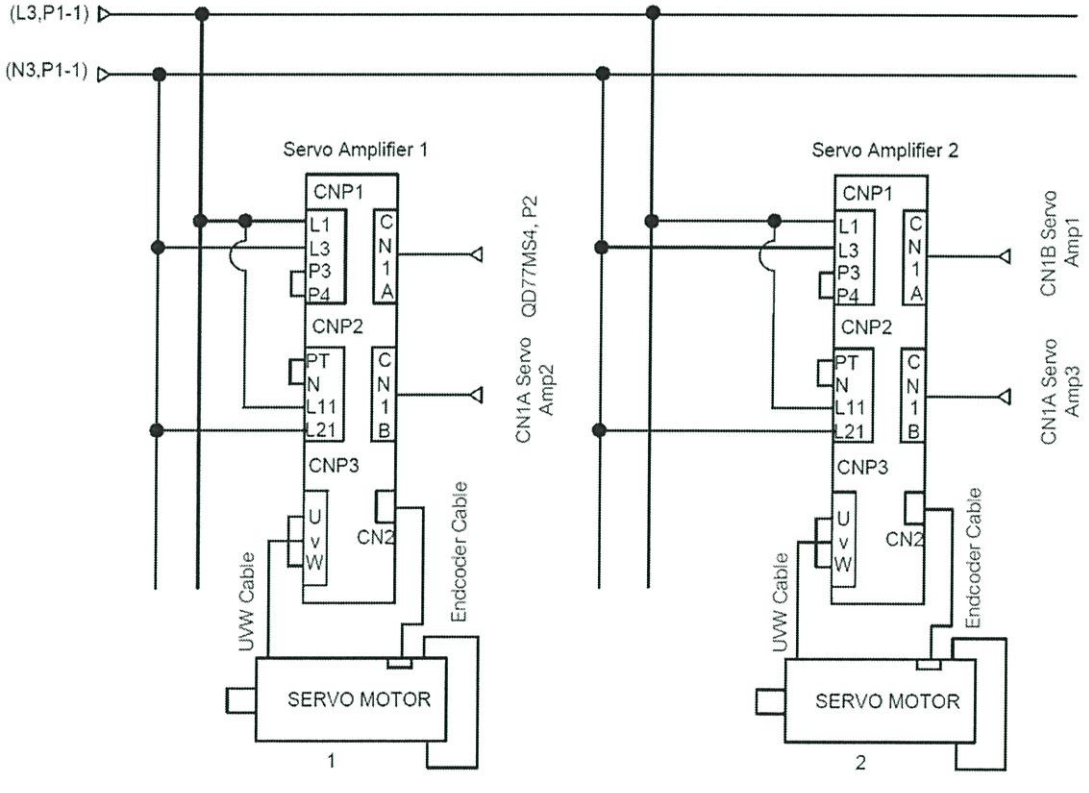
รูปที่ 4.4 วงจรการต่อสายของ PLC



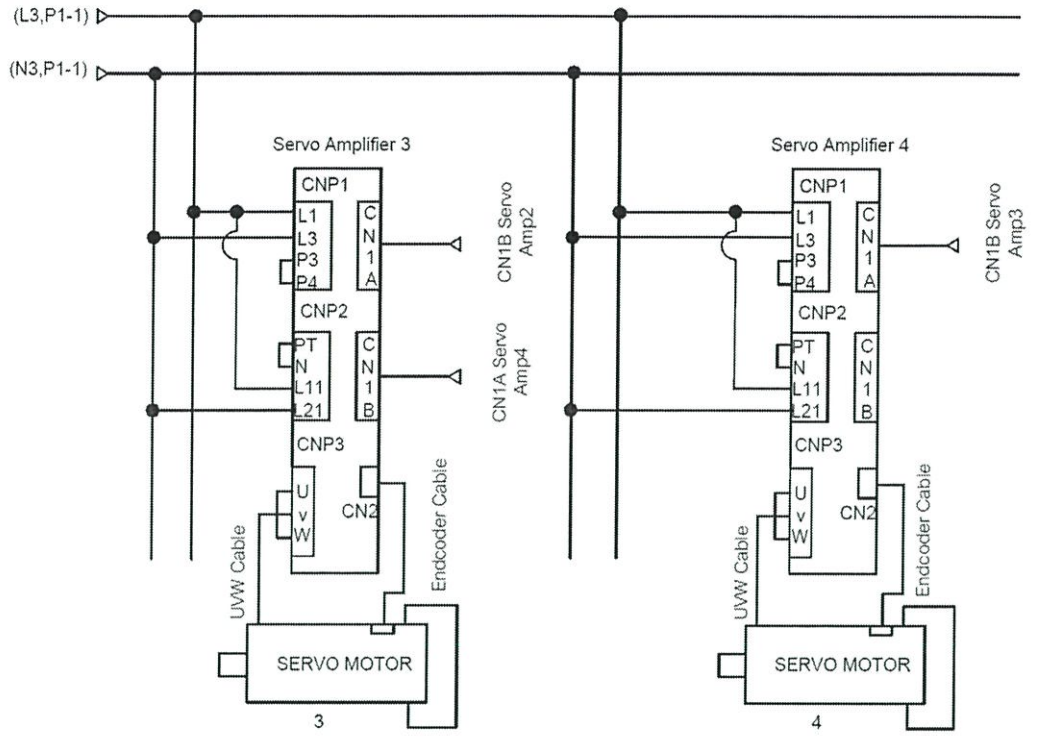
รูปที่ 4.5 วงจร Limit Switch ที่ต่อเข้ากับ Input ของ PLC



รูปที่ 4.6 วงจร Motor Brake ที่ต่อเข้ากับ Output ของ PLC



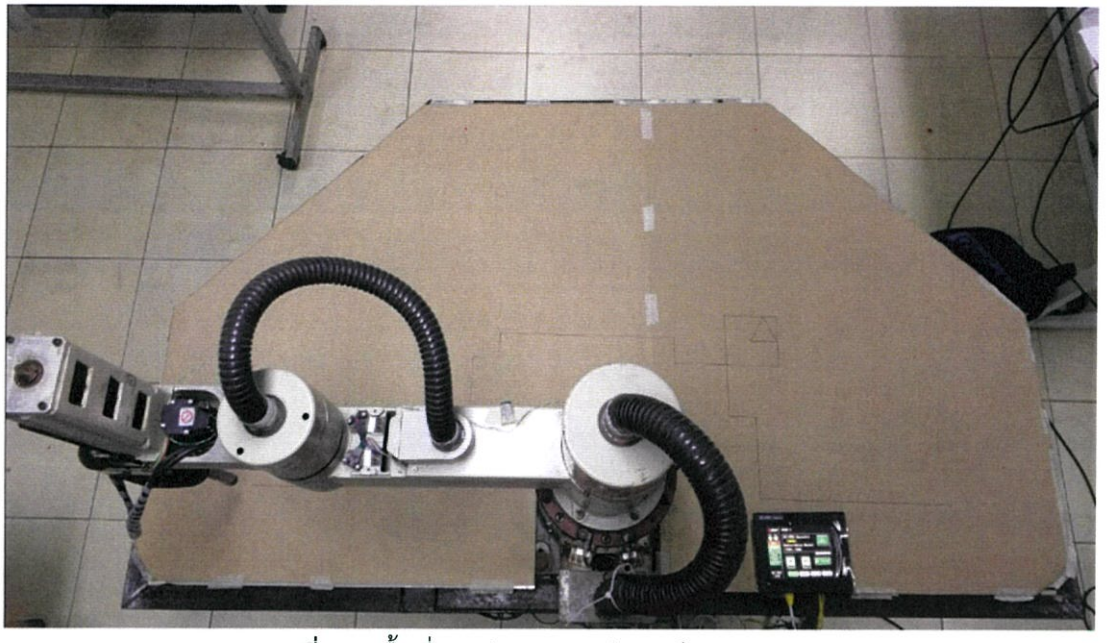
รูปที่ 4.7 วงจร Servo Amplifier และ Servo Motor ของแกนที่ 1 และ 2



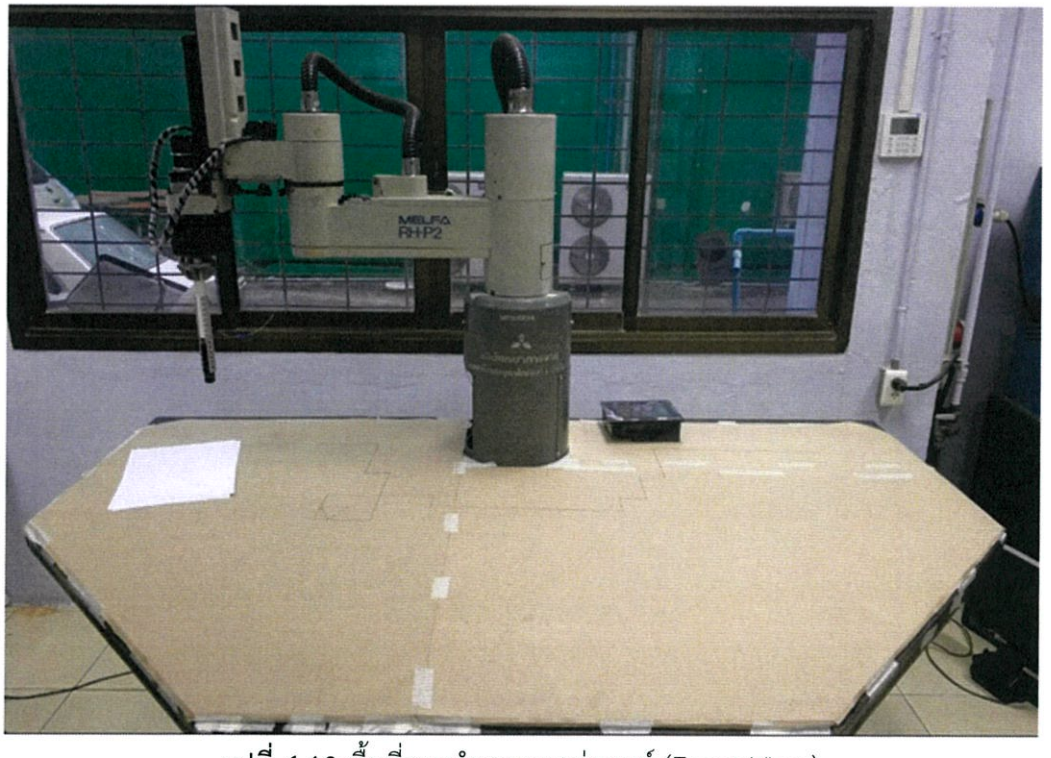
รูปที่ 4.8 วงจร Servo Amplifier และ Servo Motor ของแกนที่ 3 และ 4

ตารางที่ 4.1 ตรวจสอบความถูกต้องของการเชื่อมต่อวงจร

ลำดับ	รายละเอียด	ผลลัพธ์
1	ไฟ 220 โวลต์ เชื่อมต่อเข้ากับ ซัพพลาย 24 โวลต์ และ ซัพพลาย 5 โวลต์	ถูกต้อง
2	ไฟ 220 โวลต์ เชื่อมต่อเข้ากับ ซัพพลาย PLC	ถูกต้อง
3	วงจร Limit Switch เชื่อมต่อเข้ากับ Input ของ PLC	ถูกต้อง
4	วงจร Motor Brake เชื่อมต่อเข้ากับ Output ของ PLC	ถูกต้อง
5	วงจร Servo Amplifier และ Servo Motor แกนที่ 1 และ 2 เชื่อมต่อดังรูปที่ 4.7 และสามารถทำงานได้	ถูกต้อง
6	วงจร Servo Amplifier และ Servo Motor แกนที่ 3 และ 4 เชื่อมต่อดังรูปที่ 4.8 และสามารถทำงานได้	ถูกต้อง



รูปที่ 4.9 พื้นที่การทำงานของหุ่นยนต์ (Top View)

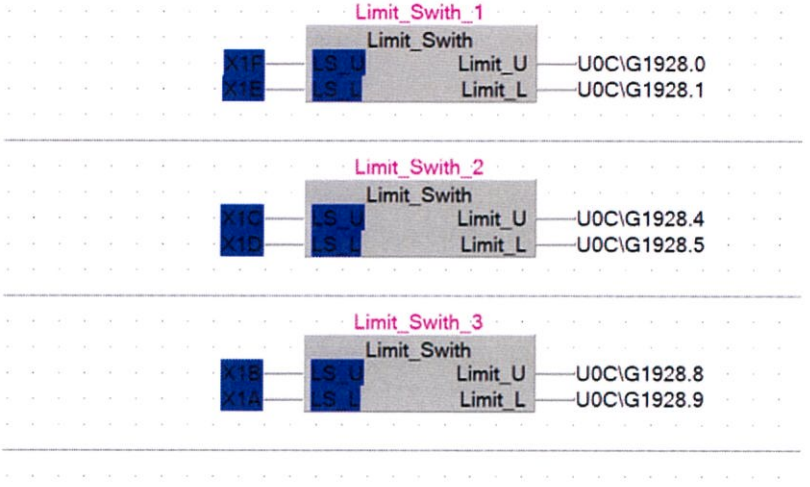


รูปที่ 4.10 พื้นที่การทำงานของหุ่นยนต์ (Front View)

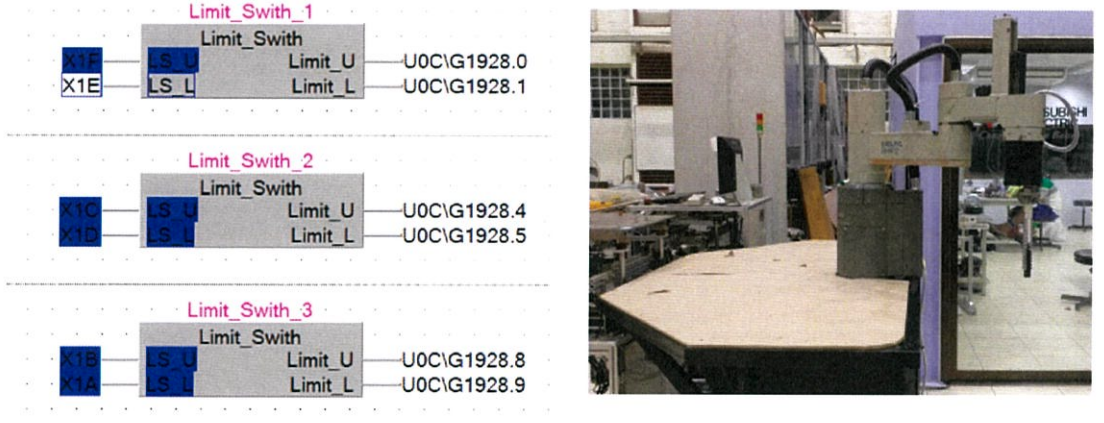
4.2.2 การทำงานของ Input

ผลการทดสอบการทำงานของ Limit Switch เมื่อทำการเชื่อมต่อเข้ากับ PLC เมื่อแกนใดแกนหนึ่งของหุ่นยนต์มีการเคลื่อนที่จน Limit Switch มีสัญญาณเข้าแกนนั้น ๆ จะหยุดการ

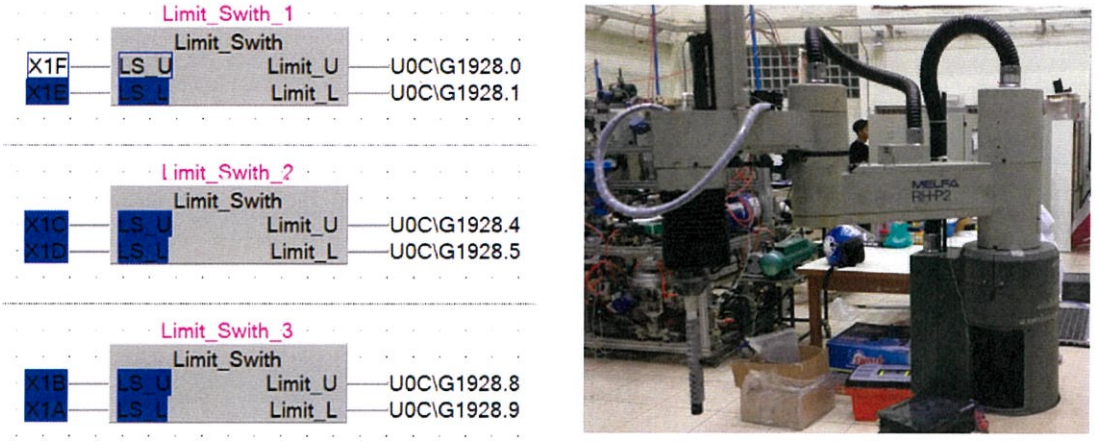
เคลื่อนที่ทันทีได้จริง ซึ่งการทำงานของ Limit Switch ดูได้เมื่อ Limit Switch ไม่ถูกตรวจจับดัง Function Blocks ในรูปที่ 4.12-4.17 ซึ่งมี Tag Name LS_U คือ Upper Limit Switch และ LS_L คือ Lower Limit Switch ทั้งสองจะเป็นสีน้ำเงิน แต่หาก Tag Name ตัวใดตัวนึงมีสัญญาณเข้า Tag Name นั้นจะเป็นสีขาว



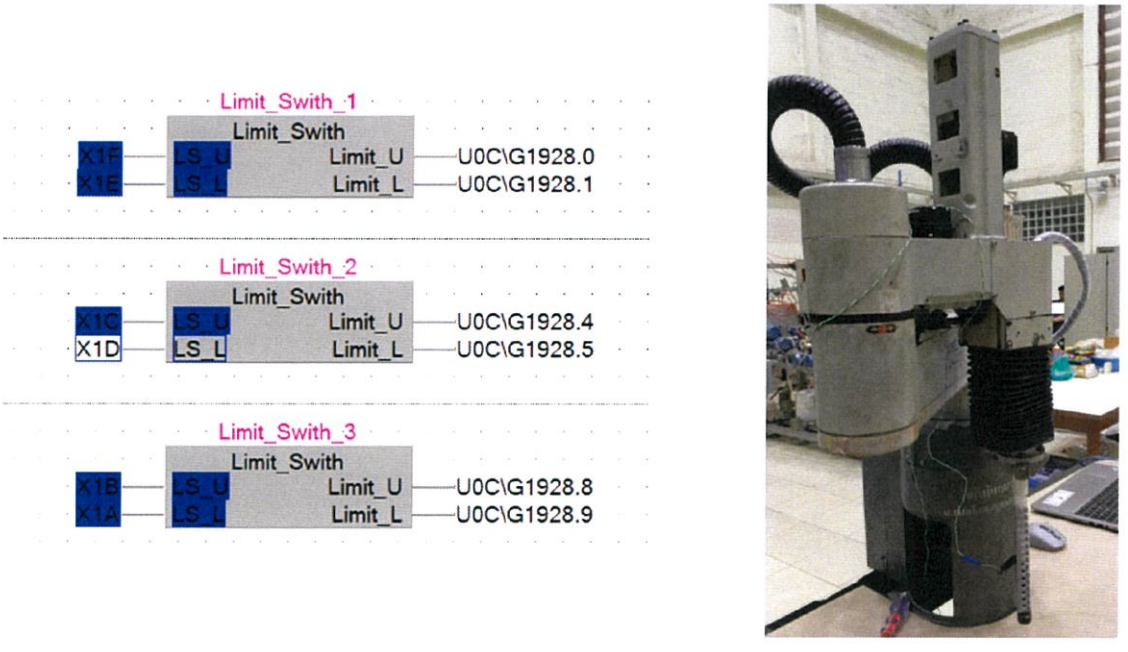
รูปที่ 4.11 ทดสอบ Input กรณีที่ Limit Switch ยังไม่ถูกตรวจจับได้ตั้งแต่แกนที่ 1 ถึง 3 ตามลำดับ



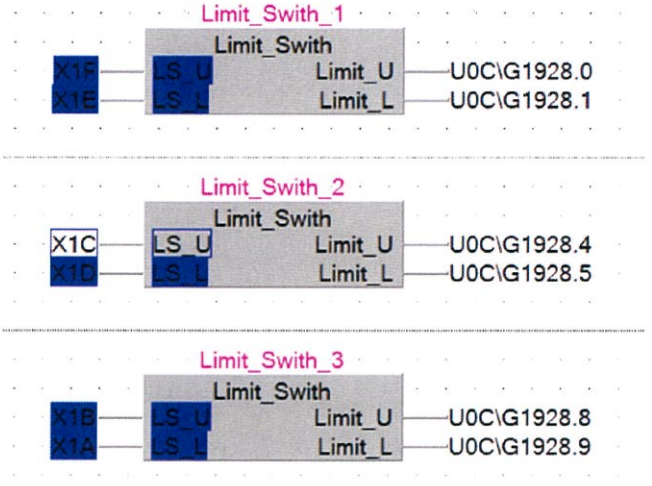
รูปที่ 4.12 ทดสอบ Input จาก Lower Limit Switch ของแกนที่ 1



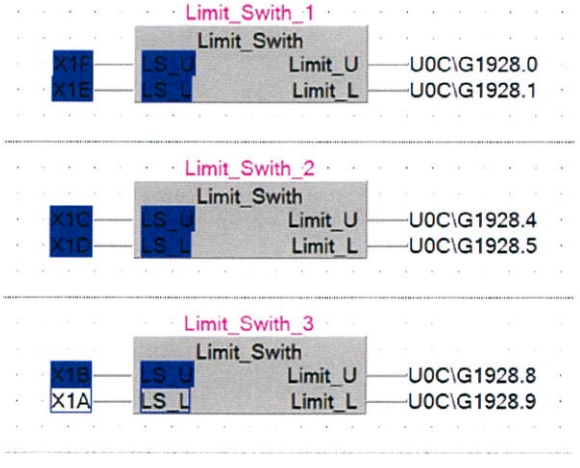
รูปที่ 4.13 ทดสอบ Input จาก Upper Limit Switch ของแกนที่ 1

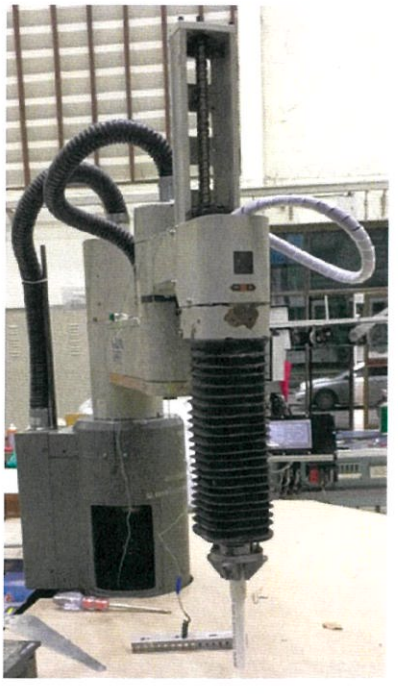


รูปที่ 4.14 ทดสอบ Input จาก Lower Limit Switch ของแกนที่ 2

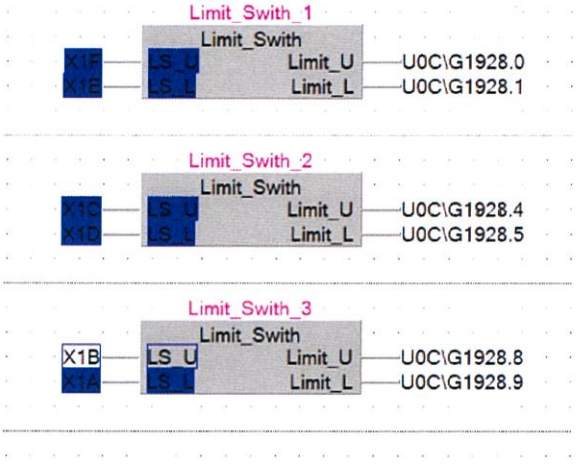


รูปที่ 4.15 ทดสอบ Input จาก Upper Limit Switch ของแกนที่ 2





รูปที่ 4.16 ทดสอบ Input จาก Lower Limit Switch ของแกนที่ 3



รูปที่ 4.17 ทดสอบ Input จาก Upper Limit Switch ของแกนที่ 3

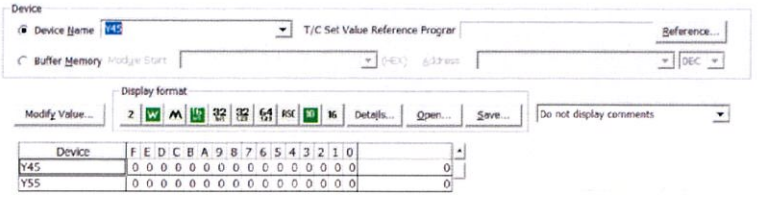
จากรูปที่ 4.12 – 4.17 ทดสอบการทำงานของ Limit Switch ด้วยการควบคุมหุ่นยนต์ทำให้ Limit Switch ทำงานสามารถสรุปเป็นตารางที่ 4.2 ได้ดังนี้

ตารางที่ 4.2 ตรวจสอบความถูกต้องในการทำงานของ Limit Switch

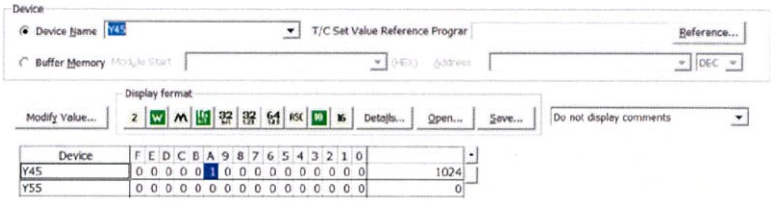
ลำดับ	รายละเอียด	Lower Switch	Upper Switch
1	ทดสอบ Input จาก Lower Limit Switch ของแกนที่ 1	ทำงาน	-
2	ทดสอบ Input จาก Upper Limit Switch ของแกนที่ 1	-	ทำงาน
3	ทดสอบ Input จาก Lower Limit Switch ของแกนที่ 2	ทำงาน	-
4	ทดสอบ Input จาก Upper Limit Switch ของแกนที่ 2	-	ทำงาน
5	ทดสอบ Input จาก Lower Limit Switch ของแกนที่ 3	ทำงาน	-
6	ทดสอบ Input จาก Upper Limit Switch ของแกนที่ 3	-	ทำงาน

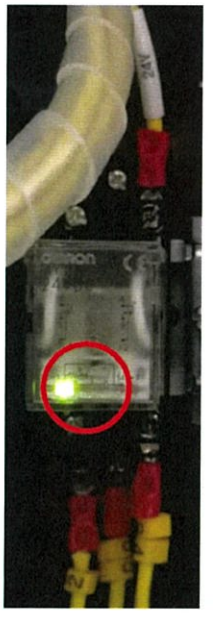
4.2.3 การทำงานของ Output

จากการทดสอบโดยการต่อ Output ไปยัง Relay เพื่อสั่งงานให้ Motor Brake ปลดเบรค เมื่อไม่มีการสั่ง Output (Y45) ทำงาน Relay จะไม่ทำงานเพื่อปลดเบรคตามรูปที่ 4.18 แต่เมื่อมีการสั่ง Output (Y45) ให้ทำงาน Relay จะทำงานสังเกตได้ว่ามีไฟขึ้นในวงกลมสีแดงและปลดเบรคทันทีตามรูปที่ 4.19 โดยมีการสรุปการทำงานตามตารางที่ 4.3



รูปที่ 4.18 ผลการทดสอบ Output เมื่อ Y45 เป็น OFF





รูปที่ 4.19 ผลการทดสอบ Output เมื่อ Y45 เป็น ON

ตารางที่ 4.3 ตรวจสอบความถูกต้องในการทำงานของ Output (Y45)

ลำดับ	รายละเอียด	ผลการทำงานของ Relay
1	สั่งปิด Output (Y45)ให้หยุดทำงาน	ปิด
2	สั่งเปิด Output (Y45)ให้ทำงาน	เปิด

4.2.4 การทำงานของ Simple Motion

การทดสอบการทำงานของ Simple Motion สามารถทดสอบจากการเปิดการทำงานของ Servo ด้วยการกดปุ่ม Start ที่หน้าจอ HMI และสังเกตผลการทำงานหรือความพร้อมใช้งานของ Servo คือ เมื่อหลอดไฟ PLC READY และ Servo READY จากหน้าต่าง Servo Monitor ใน โปรแกรม Simple Motion Module Setting หรือที่หน้าจอ HMI ทำงานแสดงสามารถเชื่อมต่อ PLC กับ Simple Motion Module ได้ แต่หากหลอดไฟไม่ทำงานหลังจากกดปุ่ม Start แสดงว่าไม่สามารถเชื่อมต่อได้ แสดงดังรูปที่ 4.20-4.21 และตารางที่ 4.4 ผลการทำงานของ Simple Motion

รูปที่ 4.20 ผลการทดสอบความพร้อมใช้งานของ Servo เมื่อคำสั่งเป็น OFF

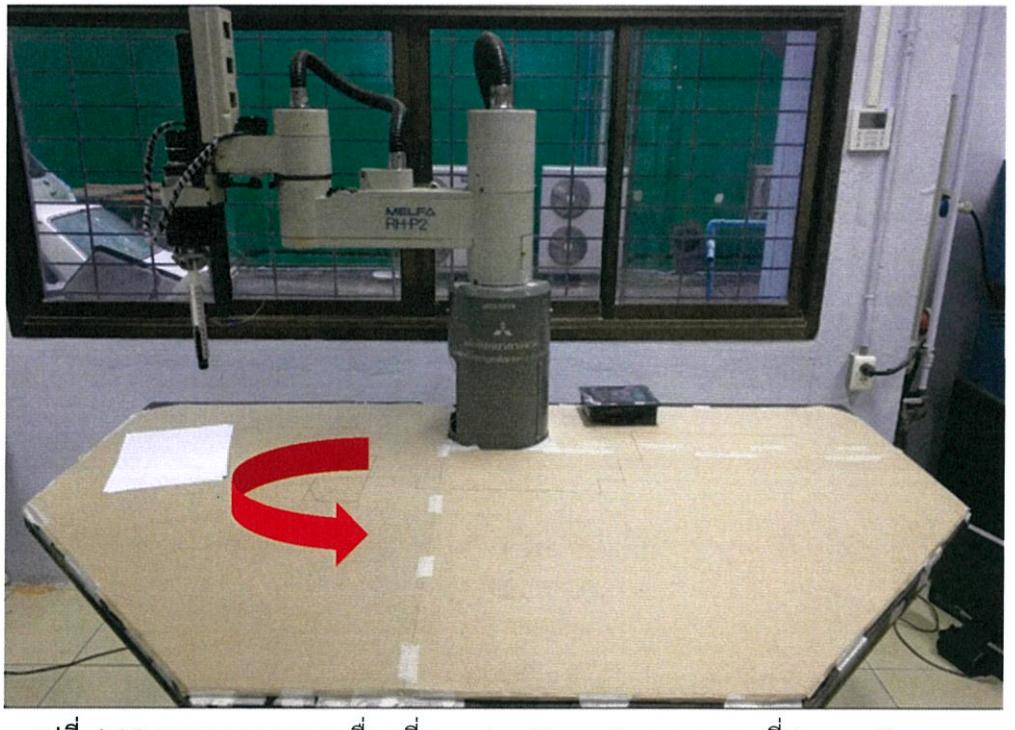
รูปที่ 4.21 ผลการทดสอบความพร้อมใช้งานของ Servo เมื่อคำสั่งเป็น ON

ตารางที่ 4.4 ตรวจสอบความพร้อมใช้งานของ Servo Amaplifier และ PLC

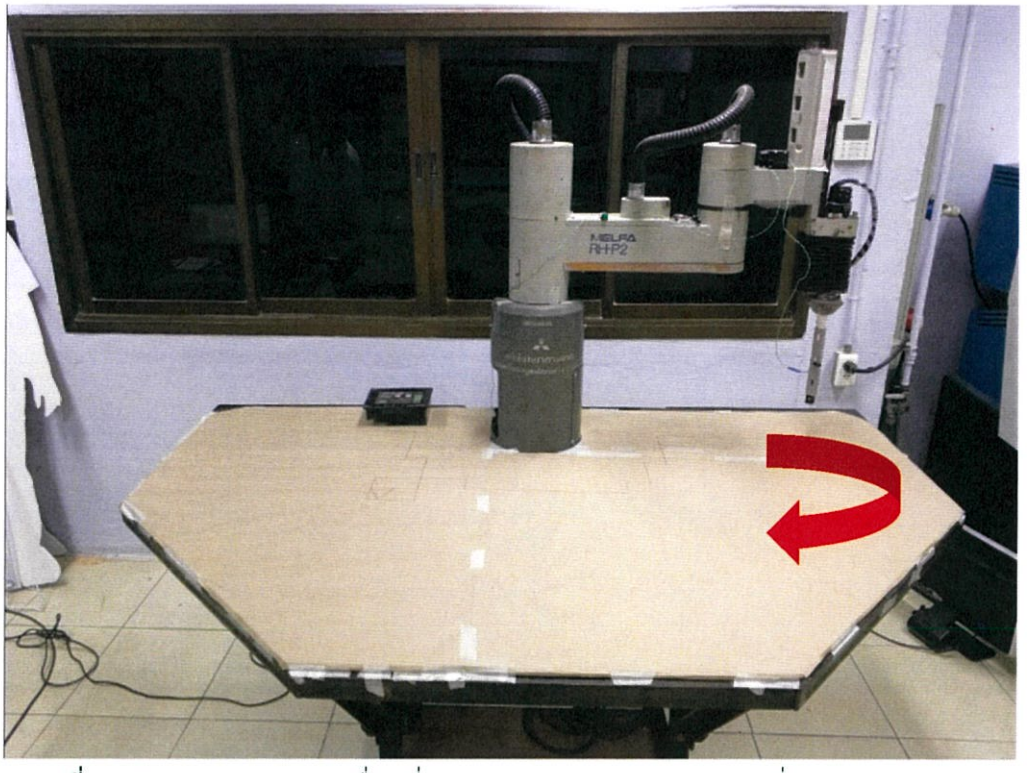
ลำดับ	รายละเอียด	ผลลัพธ์
1	กดปุ่ม START บนหน้าจอ HMI ดูความพร้อมใช้งาน Servo และ PLC	ทำงาน
2	กดปุ่ม STOP บนหน้าจอ HMI ดูความพร้อมใช้งาน Servo และ PLC	ไม่ทำงาน

4.2.5 การทำงานแบบ Jog Operation

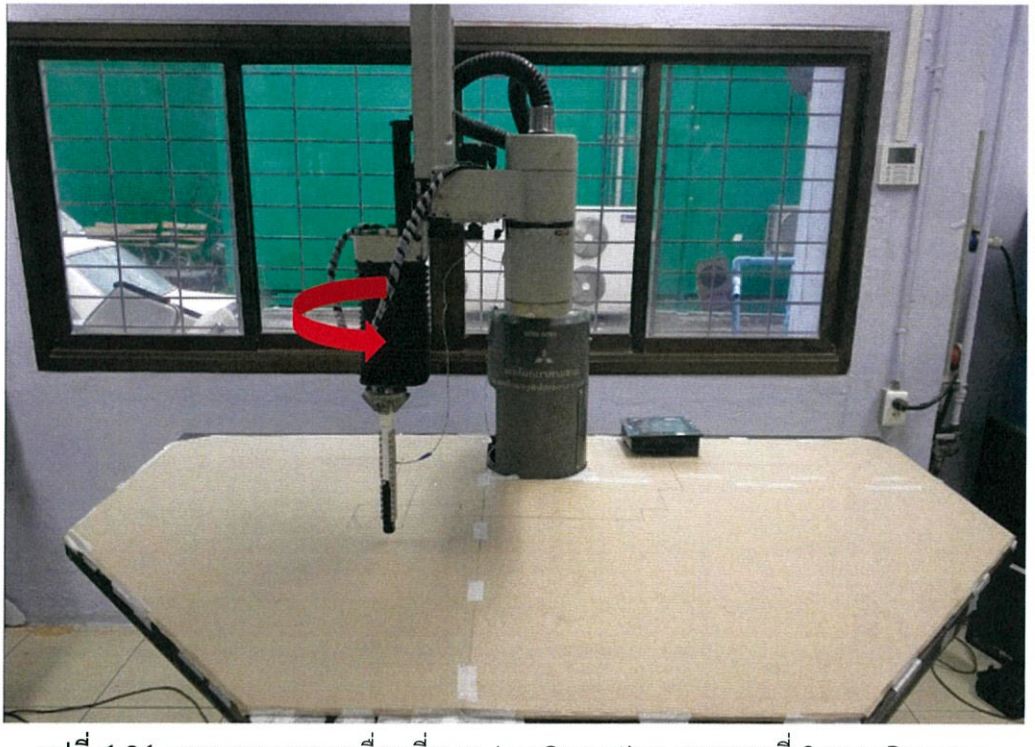
ทำการทดสอบการทำงานแบบ Jog Operation โดยสั่งให้หุ่นยนต์เคลื่อนที่ทีละแกนในทิศทาง Forward และ Reverse ผลที่ได้จากการทดลองแสดงดังตารางที่ 4.5 และรูปที่ 4.22-4.27 โดยการทดลองนี้เริ่มจากการทดลองเคลื่อนที่ Forward และ Reverse ที่ละแกนจนครบ 3 ครั้งตั้งแต่แกนที่ 1 ถึงแกนที่ 4



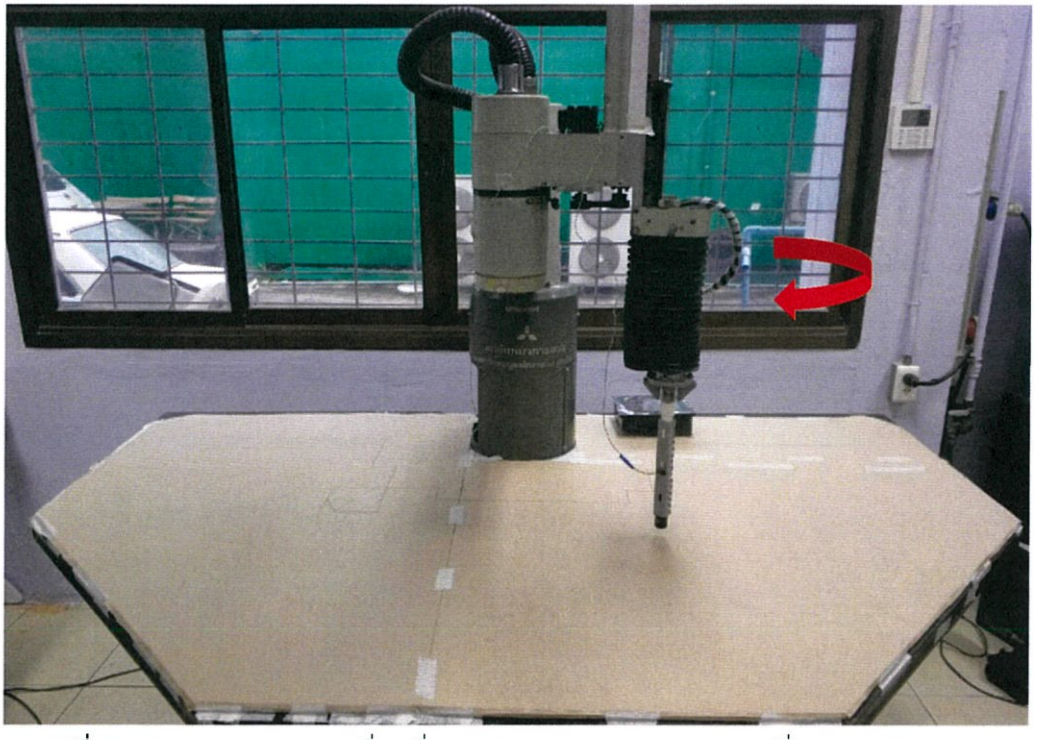
รูปที่ 4.22 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 1 แบบ Reverse



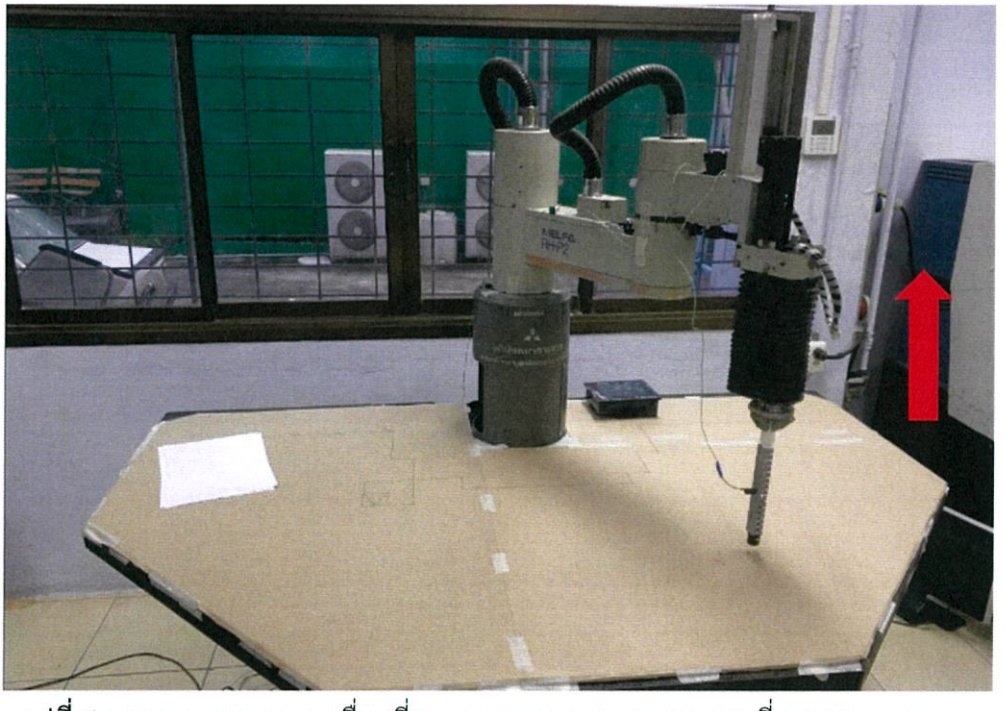
รูปที่ 4.23 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 1 แบบ Forward



รูปที่ 4.24 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 2 แบบ Reverse



รูปที่ 4.25 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 2 แบบ Forward



รูปที่ 4.26 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 3 แบบ Forward

รูปที่ 4.27 การทดลองการเคลื่อนที่แบบ Jog Operation ของแกนที่ 3 แบบ Reverse

ตารางที่ 4.5 ผลการทดสอบการเคลื่อนที่แบบ Jog Operation

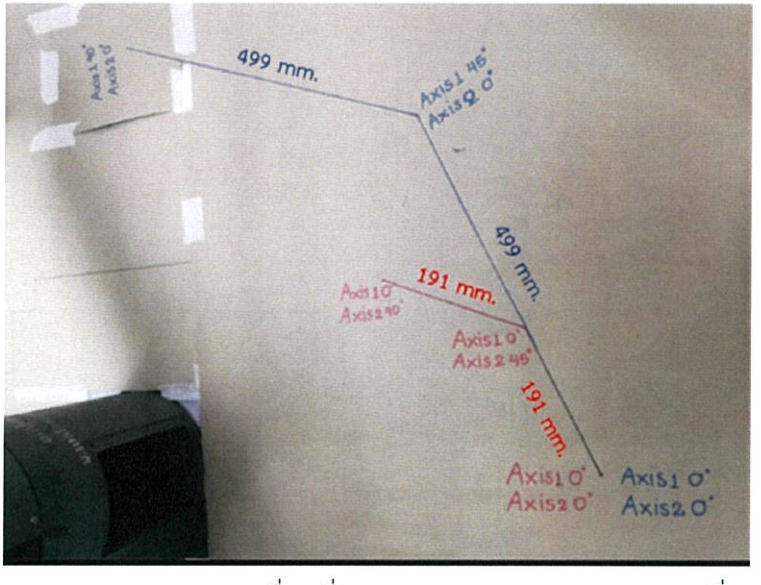
Axis	ทิศทางการเคลื่อนที่	ครั้งที่ 1	ครั้งที่ 2	ครั้งที่ 3
1	Forward Direction	✓	✓	✓
	Reverse Direction	✓	✓	✓
2	Forward Direction	✓	✓	✓
	Reverse Direction	✓	✓	✓
3	Forward Direction	✓	✓	✓
	Reverse Direction	✓	✓	✓
4	Forward Direction	✓	✓	✓
	Reverse Direction	✓	✓	✓

4.2.6 การทำงานแบบ Automatic Position Control

การทดสอบการทำงานแบบ Automatic Position Control นั้นสามารถทดลองโดยการตั้งค่าการเคลื่อนที่ใน Positioning data จากนั้นสังเกตว่าหุ่นยนต์สามารถเคลื่อนที่ได้ตาม Positioning data ที่ตั้งได้หรือไม่ โดยสามารถทดสอบความแม่นได้ดังนี้

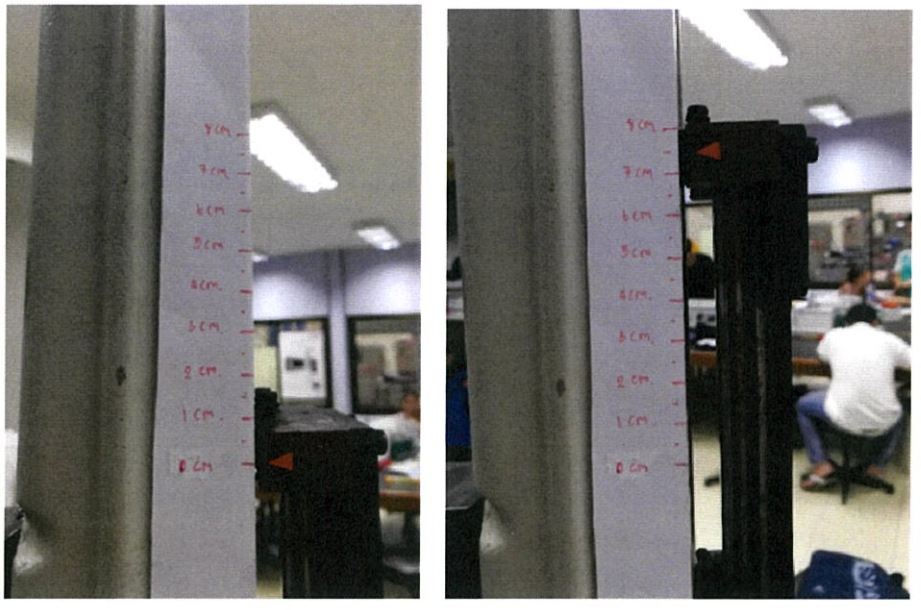
1) สำหรับแกนที่ 1 และแกนที่ 2 นั้นสามารถทดสอบได้ด้วยการเคลื่อนที่ไปที่ตำแหน่ง 0 องศา, 45 องศา และ 90 องศา และทำเครื่องหมายของแต่ละจุด จากนั้นวัดค่าว่าระยะทางจาก 0 องศาถึง 45 องศานั้นเท่ากับระยะทางจาก 45 องศาถึง 90 องศาเท่ากันหรือไม่ หากเท่ากัน แสดงว่าการเคลื่อนที่ Position Control นั้นถูกต้องตามหลักการทางคณิตศาสตร์

หลังจากที่ทำการทดสอบและพบว่าทั้งแกนที่ 1 และแกนที่ 2 สามารถเคลื่อนได้ถูกต้องตามมุมที่กำหนดจริง แสดงผลดังรูปที่ 4.28



รูปที่ 4.28 แสดงผลการทดลองการเคลื่อนที่แบบ Position Control ของแกนที่ 1 และแกนที่ 2

2) สำหรับแกนที่ 3 ทดสอบโดยทำมาตราส่วนการเคลื่อนที่ไว้ที่แกนที่ 3 จากนั้นสั่งให้หุ่นยนต์เคลื่อนที่แล้วสังเกตว่าหุ่นยนต์สามารถเคลื่อนที่ไปที่ระยะ 0 ซม. ไปยัง 7.5 ซม. จริง แสดงดังรูปที่ 4.29 (ก) และ 4.29 (ข)



(ก) (ข)

รูปที่ 4.29 แสดงผลการทดลองการเคลื่อนที่แบบ Position Control ของแกนที่ 3

4.2.7 การทำงานแบบ M-code Event Selection

จำลองเหตุการณ์การเคลื่อนที่แบบต่อเนื่อง คือ เริ่มแรกหุ่นยนต์จะอยู่ที่ตำแหน่ง 0 องศา จากนั้นจะเคลื่อนที่ไปที่ตำแหน่ง 45 องศา เมื่อถึงตำแหน่ง 45 องศา ทำงาน Event ที่ 1 คือ Y40 จะทำงานเป็นการจำลองการหนีบสิ่งของ เมื่อจบการทำงานหุ่นยนต์จะเคลื่อนที่ไปยังตำแหน่ง 90 องศา ต่อไป และเมื่อถึงตำแหน่ง 90 องศา ทำงาน Event ที่ 2 คือ Y41 จะทำงานเป็นการจำลองการปล่อยสิ่งของที่หนีบอยู่ หลังจากที่ปล่อยเสร็จแล้ว หุ่นยนต์จะเคลื่อนที่ไปยังตำแหน่ง 0 องศา เป็นการจบการทำงาน

การตั้งค่า Positioning Data สำหรับการเคลื่อนงานตามสถานการณ์จำลองข้างต้น สามารถแสดงดังรูปที่ 4.30 และผลการทดสอบการทำงานสามารถแสดงได้ตามตารางที่ 4.6

No.	Operation pattern	Control method	Axis to be interpolated	Acceleration time No.	Deceleration time No.	Positioning address	Arc address	Command speed	Dwell time	M-code
10	<Positioning Comment>									
11	1:CONT	01h:ABS Linear 1	-	0:1000	0:1000	0.00000 degree	0.00000 degree	1000.000 degre...	0 ms	0
	<Positioning Comment>									
12	1:CONT	01h:ABS Linear 1	-	0:1000	0:1000	45.00000 degree	0.00000 degree	1000.000 degre...	0 ms	1
	<Positioning Comment>									
13	1:CONT	01h:ABS Linear 1	-	0:1000	0:1000	90.00000 degree	0.00000 degree	1000.000 degre...	0 ms	2
	<Positioning Comment>									
14	0:END	01h:ABS Linear 1	-	0:1000	0:1000	0.00000 degree	0.00000 degree	1000.000 degre...	0 ms	0
	<Positioning Comment>									
15	<Positioning Comment>									

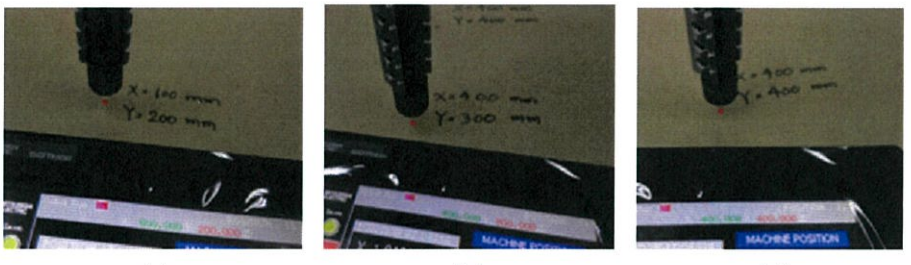
รูปที่ 4.30 ตัวอย่างการตั้งค่า Positioning Data ของการทำงาน Event

ตารางที่ 4.6 ผลการทำงานตาม Event ที่กำหนด

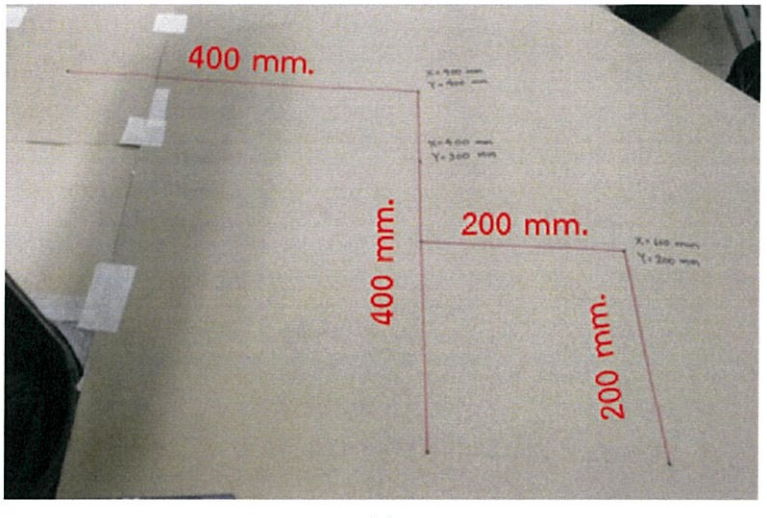
Position No.	ตำแหน่งของหุ่นยนต์ (องศา)	Y40	Y41
11	0	ไม่ทำงาน	ไม่ทำงาน
12	45	ทำงาน	ไม่ทำงาน
13	90	ไม่ทำงาน	ทำงาน
14	0	ไม่ทำงาน	ไม่ทำงาน

4.2.8 การทำงานแบบ Coordinate Movement Control

การทดลองการเคลื่อนที่แบบระบุพิกัด X และ Y สามารถทำได้โดยการกำหนดจุดที่ต้องการให้หุ่นยนต์เคลื่อนที่ไปใน Work Area จากนั้นสั่งการให้หุ่นยนต์เคลื่อนที่ไปยังตำแหน่ง ๆ ต่าง ๆ แล้วทำการความความยาวว่าหุ่นยนต์หาสามารถเคลื่อนที่ไปยังตำแหน่งต่าง ๆ ได้จริงหรือไม่ ผลการทดลองแสดงดังรูปที่ 4.31 (ก)–(ง) ซึ่งเป็นภาพขยายของรูปที่ 4.9



(ก) (ข) (ค)

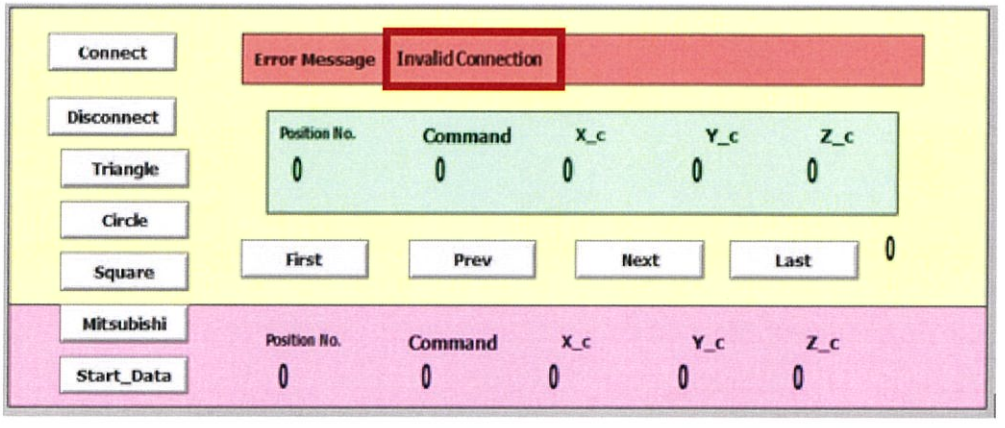


(ง)

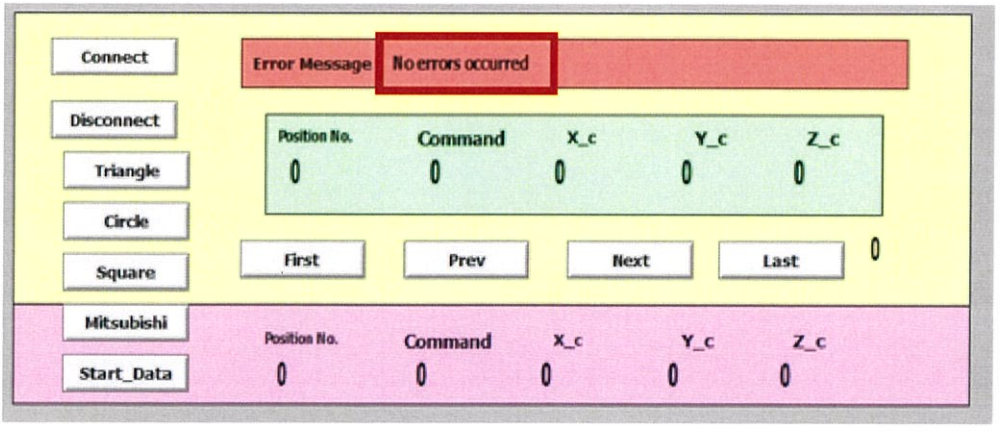
รูปที่ 4.31 ผลการเคลื่อนที่แบบระบุพิกัด X และ Y

4.2.9 การใช้โปรแกรม WW InTouch ในการ Query ข้อมูลจากระบบฐานข้อมูล

ทดสอบการใช้โปรแกรม WW InTouch ในการ Query ข้อมูลจากระบบฐานข้อมูล ขั้นตอนแรกสำหรับการทดสอบการใช้โปรแกรม WW InTouch ในการ Query ข้อมูลจากระบบฐานข้อมูล ต้องมีการทดสอบการเชื่อมต่อระหว่าง WW InTouch กับระบบฐานข้อมูล ผลการทดสอบการเชื่อมต่อระหว่าง WW InTouch กับระบบฐานข้อมูล แสดงดังรูปที่ 4.32



(ก)

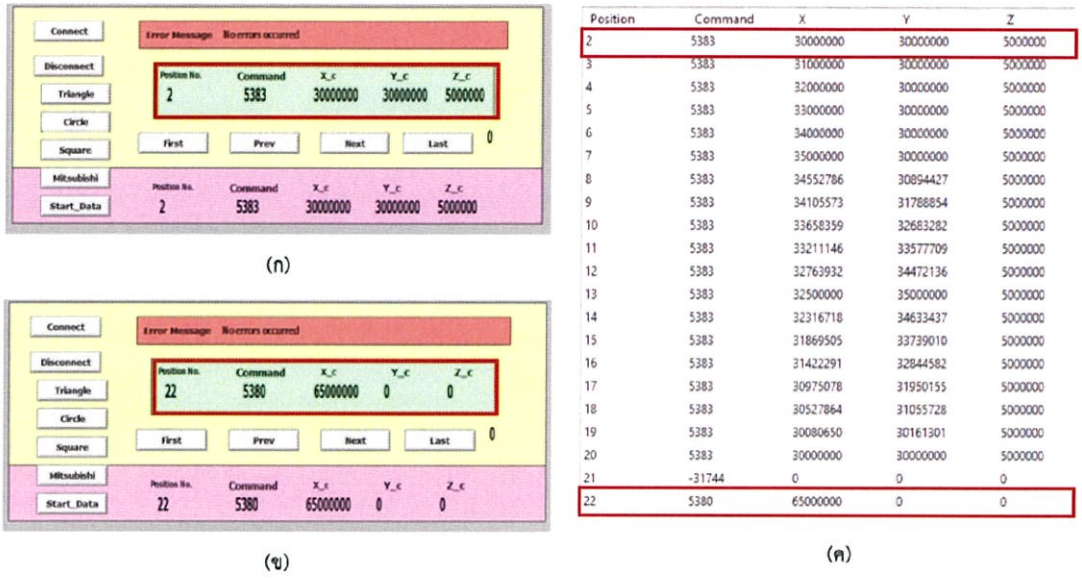


(ข)

รูปที่ 4.32 ผลการเชื่อมต่อระหว่าง WW InTouch กับระบบฐานข้อมูล

จากรูปที่ 4.32 (ก) ข้อความใน Error Message แสดงค่า Invalid Connection หมายถึง การเชื่อมต่อระหว่าง WW InTouch กับระบบฐานข้อมูลนั้นไม่สำเร็จ หรือไม่มีการเชื่อมต่อกัน และจากรูปที่ 4.32 (ข) ข้อความใน Error Message แสดงค่า No errors occurred หมายถึง ในขั้นตอนนี้ไม่มี Error เกิดขึ้น

เมื่อ WW InTouch สามารถเชื่อมต่อกับระบบฐานข้อมูลได้ ผู้ใช้งานจึงสามารถทำการทดสอบการ Query ข้อมูลจากระบบฐานข้อมูลได้ โดยการเลือกข้อมูลส่วนแรกจากตารางในระบบฐานข้อมูลจากนั้นจึงเลือกข้อมูลถัดไปเรื่อย ๆ ว่าข้อมูลที่แสดงใน WW InTouch นั้นตรงกับระบบฐานข้อมูลหรือไม่ ผลการทดสอบการใช้โปรแกรม WW InTouch ในการ Query ข้อมูลจากระบบฐานข้อมูล แสดงดังรูปที่ 4.33

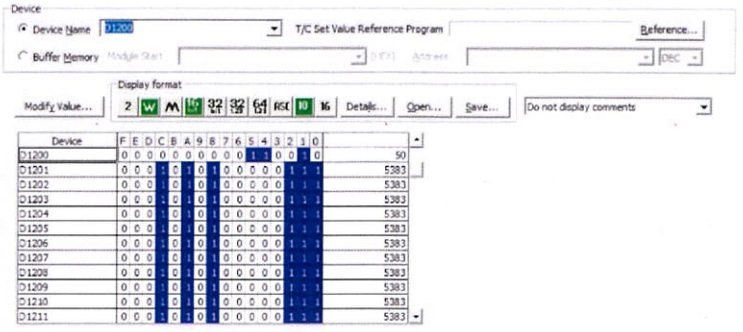


รูปที่ 4.33 ผลการ Query ข้อมูลจากระบบฐานข้อมูล

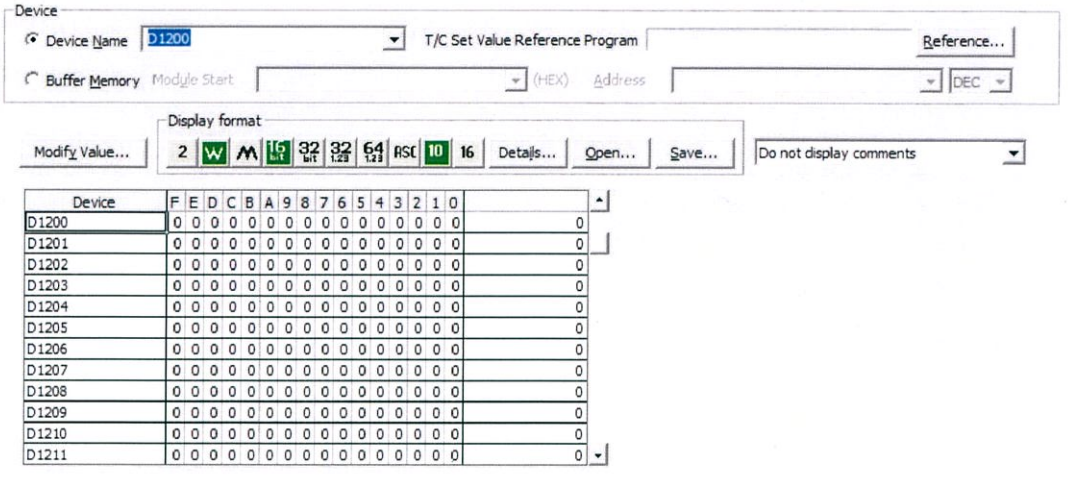
จากรูปที่ 4.33 (ก) เป็นการ Query ข้อมูลของ Triangle Path ของ Position No.2 และจากรูปที่ 4.33 (ข) เป็นการ Query ข้อมูลของ Triangle Path ของ Position No.22 ซึ่งทั้งสองข้อมูลนั้นตรงกับข้อมูลในระบบฐานข้อมูลที่แสดงในรูปที่ 4.33 (ค) จริง

4.2.10 การใช้โปรแกรม WW InTouch ในการส่งข้อมูลให้ PLC

การทดสอบการใช้โปรแกรม WW InTouch ในการส่งข้อมูลให้ PLC ทำได้โดยการ Query ข้อมูลจาก WW InTouch แล้วส่งให้ PLC นำไปเก็บไว้ใน Buffer Memory ของ PLC ก่อนจากนั้นจึงส่งข้อมูลดังกล่าวไปให้ Memory ของ Simple Motion Module อีกที ซึ่งผลการทดสอบแสดงดังรูปที่ 4.34



รูปที่ 4.34 ข้อมูลที่ถูกเก็บไว้ใน Buffer Memory ของ PLC ก่อนส่งให้ Simple Motion

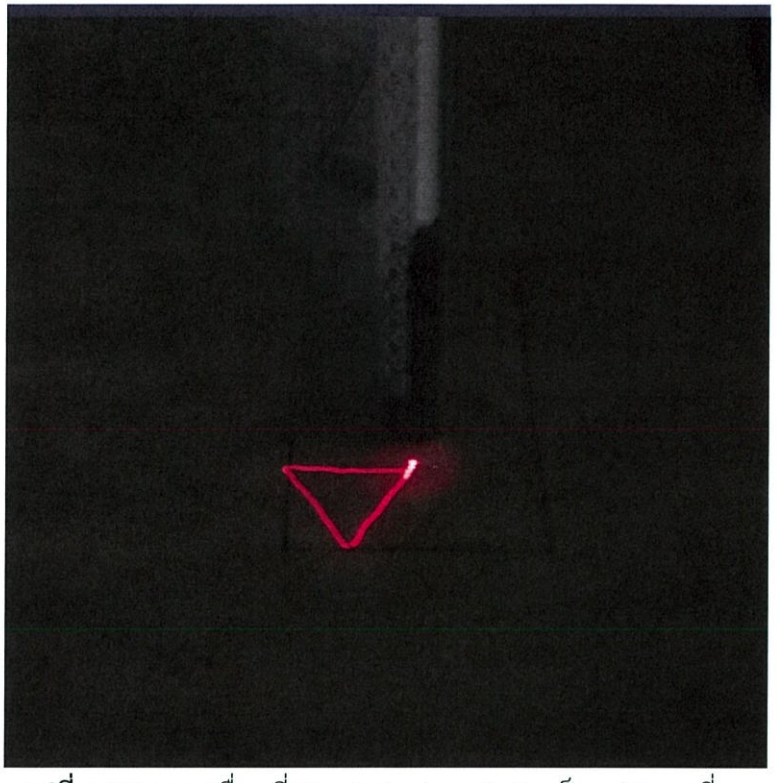


รูปที่ 4.35 ข้อมูลที่ถูกเก็บไว้ใน Buffer Memory ของ PLC หลังส่งให้ Simple Motion

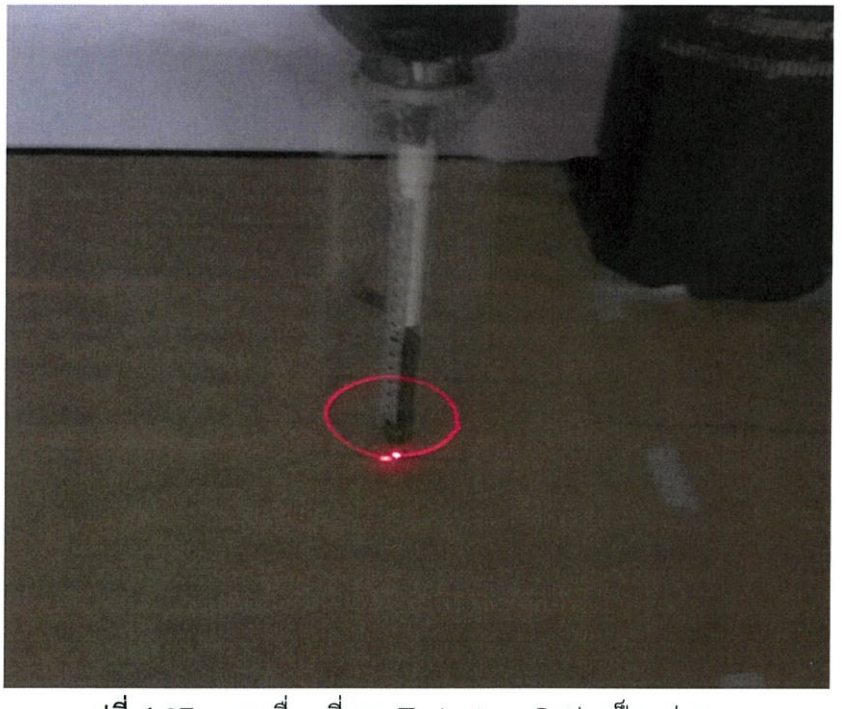
เมื่อข้อมูลที่ถูกส่งจาก WW InTouch ถูกส่งเข้ามาใน Buffer Memory ของ PLC ข้อมูลดังกล่าวจะถูกจัดเรียงตามลำดับ แสดงดังรูปที่ 4.34 และเมื่อข้อมูลนั้นถูกส่งไปที่ Memory ของ Simple Motion ข้อมูลใน Buffer Memory ของ PLC ก็จะถูกลบออก ตามรูปที่ 4.35

4.2.11 การทำงานแบบTrajectory-path Movement Control

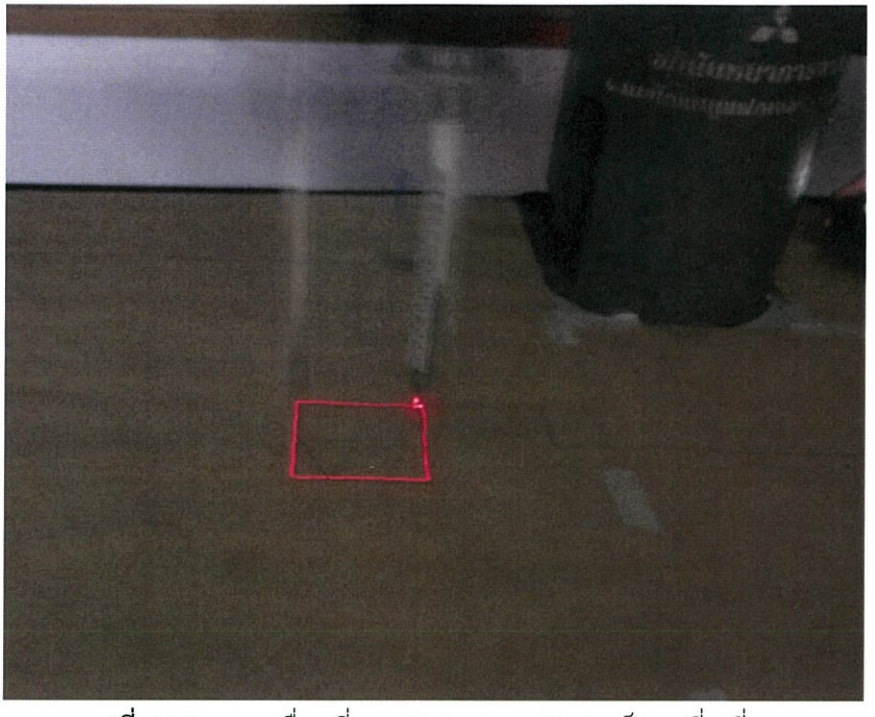
เป็นการนำ Trajectory Path ต่าง ๆ ที่สร้างจากโปรแกรม RobotStudio มาสั่งให้หุ่นยนต์เคลื่อนที่แล้วบันทึกผลโดยการถ่ายภาพด้วยเทคนิค Speed Shutter แสดงผลดังรูปที่ 4.36-4.38



รูปที่ 4.36 การเคลื่อนที่ตาม Trajectory Path เป็นรูปสามเหลี่ยม



รูปที่ 4.37 การเคลื่อนที่ตาม Trajectory Path เป็นรูปวงกลม



รูปที่ 4.38 การเคลื่อนที่ตาม Trajectory Path เป็นรูปสี่เหลี่ยม

บทที่ 5
สรุปผลและข้อเสนอแนะ

5.1 สรุปผล

จากการทดลองในการควบคุมหุ่นยนต์พบว่าสามารถทำการควบคุมหุ่นยนต์ได้ทั้งหมด 5 รูปแบบที่แตกต่างกันซึ่งจะขึ้นอยู่กับชนิดของคำสั่งที่ป้อนให้แก่ PLC ใช้ในการควบคุมได้แก่

1) การเคลื่อนที่แบบ Jog Operation โดยหุ่นยนต์จะทำการเคลื่อนที่ตามคำสั่งที่ได้รับจากอุปกรณ์ Programming Pendant ที่ผู้ใช้ได้ป้อนให้แก่หุ่นยนต์ว่าจะควบคุมแต่ละแกนไปทาง Forward Direction หรือ Reverse Direction

2) การเคลื่อนที่แบบ Position Control คือ ผู้ใช้จะป้อนคำสั่งที่มีค่าเป็น มุม (องศา) และ Positioning Data ต่าง ๆ ในแต่ละแกนเพื่อใช้ในการควบคุมการเคลื่อนที่ของหุ่นยนต์ โดยที่ผู้ใช้สามารถตั้งค่าให้หุ่นยนต์เคลื่อนที่ทีละแกนหรือทีละหลายแกนก็ได้ขึ้นอยู่กับ Positioning Data ที่ตั้งไว้

3) การเคลื่อนที่แบบมี Event คือ การให้หุ่นยนต์เคลื่อนที่ตาม Positioning Data ที่กำหนดไว้แต่ในระหว่างที่เคลื่อนที่นั้นหุ่นยนต์อาจมีการทำงานอื่นเสริมด้วยระหว่างทาง เช่น มีการหยิบจับสิ่งของ มีการเปิดการทำงานของอุปกรณ์ภายนอก และอื่น ๆ ขึ้นอยู่กับ Event ที่สร้างไว้

4) การเคลื่อนที่แบบระบุพิกัด ในการควบคุมรูปแบบนี้ผู้ใช้จะทำการป้อนพิกัดที่ต้องการให้ End Effect เคลื่อนที่ไป โดยอาจมีการป้อนพิกัดผ่านอุปกรณ์ Programming Pendant

5) การเคลื่อนที่ตาม Trajectory Path คือ หุ่นยนต์สามารถเคลื่อนตาม Trajectory Path ที่สร้างจากโปรแกรม RobotStudio ที่ถูกจัดเก็บไว้ในระบบฐานข้อมูลเพื่อการร้องขอข้อมูลจากระบบจัดการข้อมูล Trajectory Path แล้วส่งข้อมูลให้ PLC ทำการควบคุมการเคลื่อนที่ของหุ่นยนต์ให้สามารถเคลื่อนที่ตาม Trajectory Path ที่สร้างขึ้นได้

นอกจากนั้นผู้ใช้ยังสามารถเรียกดูสถานะการทำงานของหุ่นยนต์ผ่านระบบจัดการข้อมูล Trajectory Path และผ่านอุปกรณ์ Programming Pendant ได้ซึ่งประกอบไปด้วยหลายหลายส่วน เช่น Axis Error, Run, Stop, สถานะการส่งข้อมูล Trajectory Path และอื่น ๆ

5.2 ปัญหาที่พบและแนวทางแก้ไข

1) Software Stoke limit

Software Stoke limit ไม่สามารถทำงานได้เมื่อป้อนคำสั่งของการเคลื่อนเกิน 180 องศา ทางผู้จัดทำปริญญานิพนธ์จึงได้นำเดินการแก้ไขโดยการเพิ่มเขียนโปรแกรมและ ปรับอัตราทดในการ

ควบคุมให้แก่มอเตอร์ที่ใช้ในการควบคุมหุ่นยนต์ เพื่อให้หุ่นยนต์ สามารถทำงานได้ตามที่ต้องการ

2) หุ่นยนต์ที่นำมาดำเนินงานในปริญญานิพนธ์

หุ่นยนต์ที่นำมาดำเนินงานในปริญญานิพนธ์ เป็นหุ่นยนต์รุ่นเก่าที่ถูกยกเลิกการผลิตไปแล้ว ทำให้ไม่สามารถทราบข้อมูลบางอย่างที่จำเป็นต่อการควบคุมหุ่นยนต์ได้ เช่น ระยะความยาวของ Link1 และ Link2 ซึ่งจำเป็นต่อการใช้ในการควบคุมการเคลื่อนที่แบบระบุพิกัดและการเคลื่อนที่ตาม Trajectory Path ผู้จัดทำจึงได้ทำการให้เทคนิคทางคณิตศาสตร์เพื่อหาข้อมูลดังกล่าว

3) การส่งข้อมูลจากจัดการข้อมูล Trajectory Path

การส่งข้อมูลจากจัดการข้อมูล Trajectory Path ไปยัง PLC หากมีจำนวน ข้อมูลที่มากจะต้องใช้เวลาในการส่ง เนื่องจาก WW InTouch เป็น ผลิตภัณฑ์คนละบริษัทกับ PLC ทำให้การติดต่อสื่อสารระหว่างอุปกรณ์จึงเกิด ความล่าช้า

4) Driver OPC

Driver OPC ที่เป็นประเภท DASMTEthernet ในโปรแกรม SMC มีความไม่เสถียรบน Windows 10 หากมีการอัพเดท Windows บางครั้งจะทำให้มอง ไม่เห็น Driver

5.3 ข้อเสนอแนะ

เนื่องจากภาษาที่ใช้ในการเขียนโปรแกรมสำหรับตัวควบคุมหุ่นยนต์ที่มีการสร้างเป็นฟังก์ชันบล็อคนั้น ได้ใช้ภาษามาตรฐาน ดังเช่น มาตรฐาน IEC 61131-3 ดังนั้น จึงสามารถนำไปเป็นแนวทางในการพัฒนาต่อได้ เช่น การนำฟังก์ชันบล็อคที่สร้างขึ้นไปประยุกต์ใช้กับ PLC รุ่นอื่นสำหรับการควบคุมหุ่นยนต์อุตสาหกรรม หรือการนำฟังก์ชันบล็อคที่สร้างขึ้นไปประยุกต์ใช้ควบคุมหุ่นยนต์อุตสาหกรรมประเภทอื่น ๆ หรือหุ่นยนต์ประเภทเดียวกัน แต่ขนาดต่างกัน

นอกจากนี้ปริญญานิพนธ์นี้มีการใช้โปรแกรมที่หลากหลายโปรแกรม จึงเป็นผลที่อาจก่อให้เกิดความซับซ้อนทางด้านการเชื่อมต่อระหว่างโปรแกรม ยกตัวอย่างเช่น โปรแกรม RobotStudio ไม่สามารถส่งข้อมูลไปยังระบบจัดการข้อมูล Trajectory Path ได้โดยตรง จึงต้องมีการนำข้อมูลจากโปรแกรม RobotStudio ออกมาสังเคราะห์ด้วยโปรแกรม Excel และเรียงข้อมูลในโปรแกรมระบบฐานข้อมูล ซึ่งในขั้นตอนนี้ไม่ได้มีการใช้โปรแกรมอื่นในการช่วยสังเคราะห์ ซึ่งอาจทำให้เกิดความผิดพลาดของข้อมูลได้ในกรณีที่ข้อมูลมีปริมาณที่เยอะ จึงไม่เหมาะสมกับชิ้นงานที่มีขนาดใหญ่หรือชิ้นงานที่มีความละเอียดสูง และเช่นเดียวกันกับการส่งข้อมูลระหว่างระบบจัดการข้อมูล Trajectory Path และระบบ PLC นั้นก็มีการดำเนินการที่ค่อนข้างช้า จึงไม่เหมาะสมกับกับชิ้นงานที่มีขนาดใหญ่หรือชิ้นงานที่มีความละเอียดสูง เพราะทำให้ข้อมูลนั้นที่มีขนาดใหญ่ แต่หากผู้ใช้งานนั้นมีการใช้โปรแกรมอื่นในการช่วยสังเคราะห์ข้อมูล Trajectory Path สำหรับจัดเรียงในระบบฐานข้อมูล ก็อาจจะเป็นการช่วยลดข้อผิดพลาดดังกล่าวนี้ได้หรือได้มีการใช้โปรแกรมจัดการข้อมูล Trajectory

Path ของผู้ผลิตเดียวกันกับ PLC อาจจะช่วยลดเวลาในการส่งข้อมูลได้เช่นกัน ส่วนวิธีแก้ปัญหาสำหรับกรณีที่ OPC ไม่พบ DASMTEthernet Driver สามารถแก้ไขได้ด้วยการติดตั้ง Operation Integration, Mitsubishi Electric – MELSEC Driver ในโปรแกรม SMC

เอกสารอ้างอิง

[1] นิรนาม. **"แขนกลอุตสาหกรรม."** [Online].

เข้าถึงได้จาก : http://www.tpa.or.th/writer. 2558

[2] AppliCAD Co., Ltd. **"แขนกลอุตสาหกรรม."** [Online].

เข้าถึงได้จาก : https://www.applicadthai.com/articles. 2561

[3] MODERN MANUFACTURING **"พื้นฐานหุ่นยนต์ในงานอุตสาหกรรม ฉบับผู้ใช้งาน (ตอนที่ 2)."** [Online].

เข้าถึงได้จาก : https://www.mmthailand.com. 2560

[4] สารานุกรมไทยสำหรับเยาวชน **"การติดต่อสื่อสารระหว่างมนุษย์กับหุ่นยนต์และวิทยาการของการพัฒนาหุ่นยนต์ในอนาคต."** [Online].

เข้าถึงได้จาก : http://kanchanapisek.or.th. 2561

[5] มหาวิทยาลัยเทคโนโลยีราชมงคล **"การเคลื่อนที่ตามแนวเส้นตรงอย่างต่อเนื่อง."** [Online].

เข้าถึงได้จาก : http://www.atom.rmutphysics.com. 2561

[6] อรรถวิทย์ **"แขนกล."** [PDF].

[7] ADVANCE ELECTRONIC TRAINING CENTER **"PLC คือ อะไร."** [Online].

เข้าถึงได้จาก : http://www.advance-electronic.com. 2561

[8] ENERGYSCOPE **"HMI Programming."** [Online].

เข้าถึงได้จาก : http://www.energyscopethai.com/hmi-programming/. 2561

[9] ผศ. ปฏิพัทธ์ หงส์สุวรรณ **"HMI Programming."** [Online].

เข้าถึงได้จาก : http://www.cncroom.com/forum/index.php?topic=3870.0. 2557

[10] Allicano' **"เซอร์โวมอเตอร์."** [Online].

เข้าถึงได้จาก : http://chanaphinp.blogspot.com. 2559

[11] วิกิพีเดีย สารานุกรมเสรี **"ฐานข้อมูล."** [Online].

เข้าถึงได้จาก : https://th.wikipedia.org. 2560

[12] MITSUBISHI."MELSEC -Q QD77MS Simple Motion Module User's Manual." [PDF].

[13] WW FactorySuite." WW FactorySuite SQL Access Manager User's Guide." [PDF].

[14] รศ.ดร. ประเสริฐ คณาวัฒนไชย."**playlist สอน Microsoft SQL Server 2012.**" [Online]. แหล่งที่มา : https://www.youtube.com/watch?v=lQdjbBrm38s. 2561

[15] MITSUBISHI."**Mitsubishi Servo Amplifier MELSERVO-J4 Series MR-J4-A-RJ.**" [PDF].

[16] MITSUBISHI."**General-Purpose AC Servo.**" [PDF].

[17] MITSUBISHI."**GOT1000 Series Handbook Ver.D.**" [PDF].