

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ระบบควบคุมอุณหภูมิแบบหลายจุด

MULTIPOINT TEMPERATURE CONTROLLER



ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2548

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MULTIPOINT TEMPERATURE CONTROLLER



A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR
THE DEGREE OF BACHELOR IN DEPARTMENT OF INFORMATION ENGINEERING
FACULTY OF ENGINEERING
KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2005

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ใบเสนอปริญญาบัตร

หัวข้อปริญญาบัตร

ระบบควบคุมอุณหภูมิแบบหลายจุด

TITLE

MULTIPOINT TEMPERATURE CONTROLLER

นักศึกษา

นายวีระเดช ชูมา

รหัสนักศึกษา 46015683

อาจารย์ผู้ควบคุมปริญญาบัตร

ผศ. คล้าย สุขเจริญผล

ภาควิชา

วิศวกรรมสารสนเทศ

ปีการศึกษา

2548

ปริญญาบัตรฉบับนี้ได้รับการอนุมัติให้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิศวกรรมศาสตรบัณฑิต คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหาร
ลาดกระบัง

(ผศ. คล้าย สุขเจริญผล)

อาจารย์ผู้ควบคุมปริญญาบัตร

ลิขสิทธิ์ของคณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อปริญญานิพนธ์ ระบบควบคุมอุณหภูมิแบบหลายจุด

ชื่อนักศึกษา นายวีระเดช ฐูมา รหัสนักศึกษา 46015683

อาจารย์ที่ปรึกษา ผศ. คลชัย สุขเจริญผล

ระดับการศึกษา ปริญญาตรี วิศวกรรมศาสตรบัณฑิต
สาขาวิศวกรรมสารสนเทศ

ภาควิชา วิศวกรรมสารสนเทศ

ปีการศึกษา 2548

บทคัดย่อ

ปริญญานิพนธ์นี้กล่าวถึงการออกแบบ และการสร้างระบบควบคุมอุณหภูมิแบบหลายจุด จัดทำขึ้นเพื่อเพิ่มความสามารถในการควบคุมอุณหภูมิในระบบต่างๆ ใกล้เคียงความต้องการของมนุษย์มากที่สุด โดยเป็นระบบจะเป็นการควบคุมแบบPID เพื่อเพิ่มประสิทธิภาพของระบบ โดยใช้ไมโครคอนโทรลเลอร์ตระกูล F²MC-8L ในการควบคุมค่าต่างๆ ลักษณะของการควบคุมเป็นการตรวจสอบอุณหภูมิ จากเซนเซอร์ แล้วนำค่าที่ได้มาเป็นอินพุท ที่ใช้ควบคุม ฮีทเตอร์ พัดลม หรือตัวจ่ายอุณหภูมิ โดยจะมีเซนเซอร์และสามารถควบคุมอุณหภูมิได้หลายจุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Thesis Title MULTIPOINT TEMPERATURE CONTROLLER
Student Mr. Weeradet Thuma ID. 46015683
Advisor Asst. Prof. Dolchai Sookcharoenphol
Graduate Level Bachelor Degree of Information Engineering
Department Information Engineering
Academic Year 2005

ABSTRACT

This project is about Multipoint Temperature Controller for increase controlling system ability. It guesses the result for increase efficiency of system. Microcontroller F²MC-8L Series is the controlling center that can control temperature with PID-control system form sensor and use that result to be input to control heater and fan. It has mean and multi point of temperature that points of temperature have multi of sensor

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ได้จัดทำขึ้นเป็นผลสำเร็จ ทางผู้จัดทำต้องขอขอบพระคุณท่าน
บูรพาจารย์ทั้งหลาย ผู้เขียนเอกสารและตำราอ้างอิงต่างๆ โดยเฉพาะอาจารย์ที่ปรึกษา
ผู้ช่วยศาสตราจารย์ คล้าย สุขเจริญผล เป็นอย่างสูง ที่ช่วยถ่ายทอดความรู้ ให้ข้อคิดข้อเสนอแนะและ
ให้โอกาส ในการพัฒนาปริญญานิพนธ์ พร้อมทั้งขอขอบคุณเพื่อน เจดน์ ออสวัสดิ์ ที่เป็นที่ปรึกษา
ในทุกๆด้าน รวมถึงผู้ช่วยให้ปริญญานิพนธ์ฉบับนี้สำเร็จได้ บนอุปสรรคทั้งการเรียนและการทำงาน
ของผู้จัดทำ และขอขอบคุณเพื่อนๆและพี่ๆที่บริษัท ไอทีวี จำกัด (มหาชน) ที่ให้ได้โอกาสและ
กำลังใจในการมาศึกษาต่อ ผู้จัดทำต้องขอขอบพระคุณ ภาควิชาวิศวกรรมสารสนเทศ คณะ
วิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่ให้โอกาสหลายครั้ง
กับผู้จัดทำได้มีโอกาสเข้ามาศึกษา ณ สถาบันแห่งนี้

ท้ายสุดขอขอบคุณอาจารย์ทุกท่านที่ไม่ได้กล่าวถึงในที่นี้ที่กรุณา ประสิทธิ์ประสาทวิชา
ความรู้ รวมถึงแนวทางการคิดและแนวปฏิบัติให้แก่ผู้จัดทำ จนทำให้ปริญญานิพนธ์ฉบับนี้
สำเร็จผลตามเป้าหมาย

คุณความดี ใดๆ ที่เกิดจากปริญญานิพนธ์ฉบับนี้ขอมอบให้แก่ บิลา มารดา ครูบาอาจารย์ที่
ได้ประสิทธิ์วิชาความรู้ให้

นายวีระเดช ชูมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เรื่อง	หน้าที่
บทคัดย่อภาษาไทย	ก
บทคัดย่อภาษาอังกฤษ	ข
กิตติกรรมประกาศ	ค
สารบัญ	ง
สารบัญรูป	จ
บทที่ 1 บทนำ	1
1.1 แนวคิดและที่มาของปัญหา	1
1.2 จุดประสงค์	1
1.3 ขอบเขตของโครงการ	2
1.4 ผลที่คาดว่าจะได้รับ	2
1.5 วิธีการดำเนินงาน	2
บทที่ 2 ทฤษฎีที่เกี่ยวข้อง	3
2.1 PID Control	3
2.1.1 การควบคุมแบบ PID	3
2.1.2 ระบบการควบคุมแบบต่างๆ	4
2.1.3 การควบคุม ON-OFF Control	4
2.1.4 การควบคุมแบบ Proportional (P)	5
2.1.5 การควบคุมแบบ Integral (I)	7
2.1.6 การควบคุมแบบ Derivative (D)	10
2.2 การปรับค่าในระบบควบคุมแบบ PID	12
2.2.1 ผลตอบสนองของการควบคุมที่ดี	12
2.2.2 ความยากง่ายในการควบคุมของโปรเซส	14
2.2.3 ผลของ PID ต่อเสถียรภาพของระบบ	14
2.2.4 คุณภาพของการควบคุม	15
2.2.5 วิธีการปรับค่า PID	16
2.2.6 การรบกวนซึ่งกันและกันของค่า PID	18
2.2.7 วิธีการตั้งค่า PID (Dial setting) ในตัวควบคุมที่มีการรบกวน	20
กัน	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3 MODBUS PROTOCOL	21
2.3.1 ลำดับชั้นของ MODBUS ในแบบจำลอง OSI	21
2.3.2 หมายเลขคำสั่งหรือ Function Code	26
2.3.3 โพรโทคอล MODBUS Serial Line	29
2.3.4 โหมดการรับส่งข้อมูลอนุกรม	32
2.3.5 รูปแบบการเชื่อมต่อบัสข้อมูล	36
สรุป Modbus Protocol	38
บทที่ 3 การออกแบบโครงงาน	39
การออกแบบฮาร์ดแวร์และซอฟต์แวร์	40
3.1 ส่วนควบคุม	40
3.2 ส่วนติดต่อพอร์ตอนุกรม	41
3.3 เซ็นเซอร์	41
3.4 การออกแบบการทำงานในส่วนของซอฟต์แวร์	44
3.5 การออกแบบตัวควบคุม PID	46
บทที่ 4 การทดลองและทดสอบระบบ	48
4.1 การทดลองการทำงาน	48
4.2 ผลการทดลองการทำงานของระบบควบคุมอุณหภูมิแบบหลายจุด	51
บทที่ 5 สรุปการพัฒนาโครงงาน	53
5.1 สรุปการพัฒนาโครงงาน	53
5.2 ปัญหาที่เกิดขึ้น	53
5.3 แนวทางในการพัฒนาต่อ	54
ภาคผนวก	55

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

ภาพและคำอธิบาย	หน้าที่
รูปที่ 2.1 แสดง Loop การควบคุมที่สมบูรณ์แบบ	3
รูปที่ 2.2 การทำงานของ ON-OFF Control	4
รูปที่ 2.3 แสดงบล็อกไดอะแกรมของการควบคุมแบบ P	5
รูปที่ 2.4 กราฟแสดงการเกิดค่า Offset ในระบบการควบคุมแบบ P	6
รูปที่ 2.5 กราฟแสดงผลตอบสนองของการควบคุมแบบ P	6
รูปที่ 2.6 แสดงบล็อกไดอะแกรมของการควบคุมแบบ I	7
รูปที่ 2.7 แสดงการเปลี่ยนแปลงของการควบคุมแบบ I	7
รูปที่ 2.8 แสดงบล็อกไดอะแกรมของการควบคุมแบบ PI ในแต่ละแบบ	8
รูปที่ 2.9 แสดงกราฟการเปลี่ยนแปลงของการควบคุมแบบ PI	9
รูปที่ 2.10 แสดงบล็อกไดอะแกรมของการควบคุมแบบ D	10
รูปที่ 2.11 แสดงบล็อกไดอะแกรมของการควบคุมแบบ PID	11
รูปที่ 2.12 แสดงกราฟการเปลี่ยนแปลงของการควบคุมแบบ PID	12
รูปที่ 2.13 แสดงกราฟผลตอบของระบบควบคุม	13
รูปที่ 2.14 แสดงรูปกราฟที่มีพื้นที่การควบคุมน้อยที่สุด	16
รูปที่ 2.15 แสดงลักษณะของรูปคลื่น โปรเซสในการปรับค่า PID ในวิธีที่ 1	17
รูปที่ 2.16 แสดงการหาความเวลาในการแกว่งของการปรับค่า PID ในวิธีที่ 2	18
รูปที่ 2.17 แสดงบล็อก ไดอะแกรมของตัวควบคุมแบบดีที่สุด	19
รูปที่ 2.18 แสดงบล็อกไดอะแกรมของตัวควบคุมที่มีขายทั่วๆ ไป	19
รูปที่ 2.19 กราฟแสดงการหาค่า μ	20
รูปที่ 2.3.1 MODBUS คับแบบจำลอง OSI	22
รูปที่ 2.3.2 เฟรมข้อมูล ADU และ PDU	23
รูปที่ 2.3.3 การรับส่งข้อมูลระหว่าง ไคลเอนท์กับเซิร์ฟเวอร์	24
รูปที่ 2.3.5 การเข้าถึงและอ่านค่าข้อมูลชนิดต่างๆ	27
รูปที่ 2.3.6 ช่วงค่าแอดเดรสของฟังก์ชันทั้งแบบ Public และ User Defined	28
รูปที่ 2.3.7 ลำดับชั้นของ MODBUS Serial Line	30
รูปที่ 2.3.8 การติดต่อสเลฟแบบ unicast และ broadcast	31
รูปที่ 2.3.9 เฟรมข้อมูลของ MODBUS Serial Line	32

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 2.3.10	แผนผังเวลาการรับส่งเฟรมข้อมูลระหว่างมาสเตอร์กับสเลฟ	33
รูปที่ 2.3.11	เฟรมข้อมูลในโหมด RTU	34
รูปที่ 2.3.12	การส่งข้อมูล 1 ไบต์ในโหมด RTU	35
รูปที่ 2.3.13	การกำหนดช่วงระยะห่างทางเวลาระหว่างเฟรมข้อมูลแต่ละเฟรม	36
รูปที่ 2.3.14	เฟรมข้อมูลในโหมด ASCII	37
รูปที่ 2.3.15	การส่งข้อมูล 1 ตัวอักษรในโหมด ASCII	38
รูปที่ 2.3.16	IEA/TIA-485	39
รูปที่ 3.19	IEA/TIA-485	40
รูปที่ 3.1	ถึงโครงสร้างของระบบควบคุมอุณหภูมิแบบหลายจุด	41
รูปที่ 3.2	วงจรของระบบควบคุมอุณหภูมิแบบหลายจุด	42
รูปที่ 3.3	ลายวงจรของระบบควบคุมอุณหภูมิแบบหลายจุด	44
รูปที่ 3.4	บอร์ดระบบควบคุมอุณหภูมิแบบหลายจุด	44
รูปที่ 3.5	ผังการทำงานของระบบควบคุมอุณหภูมิแบบหลายจุด	45
รูปที่ 3.6	ค่า Error ที่มีการเปลี่ยนแบบเป็น Step	47
รูปที่ 3.7	ค่า Error ที่มีการเปลี่ยนแบบเป็น ramp	47
รูปที่ 4.1	บล็อกไดอะแกรมของระบบควบคุมอุณหภูมิแบบหลายจุด	48
รูปที่ 4.2	กราฟอุณหภูมิที่ Setpoint 35 องศาโดยตั้งค่า $KP = 25$	49
รูปที่ 4.3	กราฟอุณหภูมิที่ Setpoint 35 องศาโดยตั้งค่า $KP = 10$	49
รูปที่ 4.4	กราฟอุณหภูมิที่ Setpoint 40 องศาโดยตั้งค่า $KP = 20$ $KI = 1$	50
รูปที่ 4.5	รูปกราฟแสดงถึงกราฟของ Setpoint และ Output แบบเปอร์เซ็นต์	51
รูปที่ 4.6	รูปกราฟแสดงผลการทำงานของระบบควบคุมอุณหภูมิแบบหลายจุด	51

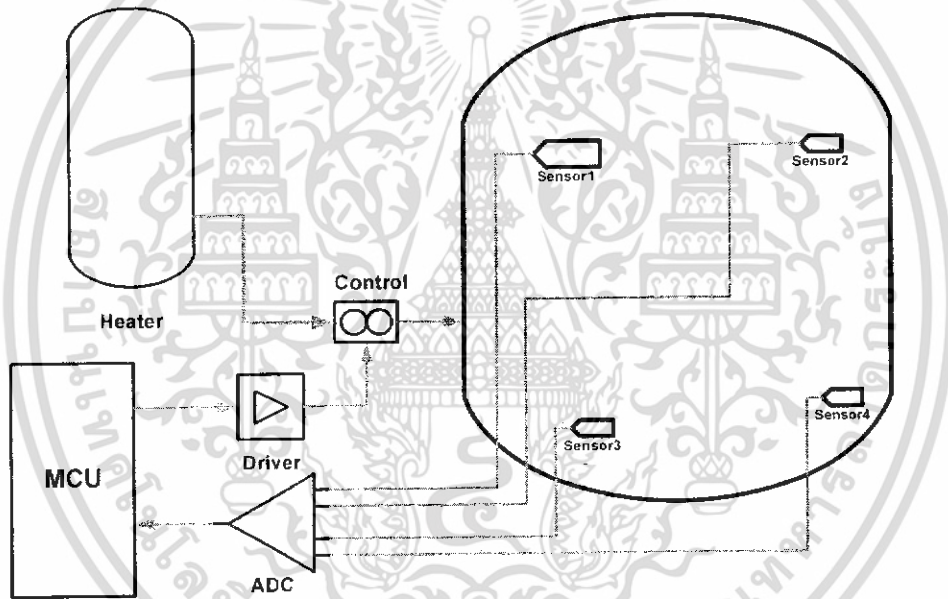
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

แนวคิดและที่มาของปัญหา

1.1 แนวคิดและที่มาของปัญหา

เนื่องด้วยระบบที่ต้องการควบคุมอุณหภูมิในปัจจุบันที่ใช้กันอยู่เช่น การปรับอากาศในอาคาร การอบผลผลิตทางการเกษตร การควบคุมอุณหภูมิในอุตสาหกรรมต่างๆ ที่ต้องการการควบคุมอุณหภูมิในบริเวณกว้าง และต้องใช้อุปกรณ์วัดและจ่ายอุณหภูมิในหลายจุด ยังไม่มีการควบคุมการจ่ายอุณหภูมิที่ดีพอทำให้เกิดผลเสียเช่น อุณหภูมิในห้องที่ต้องการควบคุมอุณหภูมิเย็นหรือร้อนเกินไปในบางช่วง หรือสูญเสียพลังงานในการจ่ายอุณหภูมิมากเกินไปโดยไม่จำเป็น



1.2 จุดประสงค์

- 1.2.1 เพื่อพัฒนาระบบการควบคุมอุณหภูมิ
- 1.2.2 เพื่อประหยัดพลังงานที่ต้องใช้ในการควบคุมอุณหภูมิ
- 1.2.3 ลดความยุ่งยากในการควบคุมอุณหภูมิที่ใช้ระบบ Manual

1.3 ขอบเขตของโครงการ

- 1.3.1 ระบบจะทำการวัดอุณหภูมิในจุดต่างๆเพื่อนำมาประมวลผลในการควบคุมตัวจ่ายอุณหภูมิตามจุดต่างๆ
- 1.3.2 ใช้ PID-control system ในการควบคุม
- 1.3.3 ใช้ ไมโครคอนโทรลเลอร์ที่มีประสิทธิภาพสูงในการทำงานเพื่อลดข้อผิดพลาดของไมโครคอนโทรลเลอร์ในการทำงานในพื้นที่ ที่มีอุณหภูมิสูง
- 1.3.4 สามารถปรับแต่งและแสดงผลการทำงานผ่านคอมพิวเตอร์ได้(Virtual instruments)

1.4 ผลที่คาดว่าจะได้รับ

- 1.4.1 ระบบที่พัฒนาขึ้น จะสามารถนำไปใช้ในอุตสาหกรรม และในอาคารที่ต้องการควบคุมอุณหภูมิแบบอัตโนมัติได้
- 1.4.2 สามารถประหยัดพลังงานที่ต้องใช้ในการควบคุมอุณหภูมิได้

1.5 วิธีการดำเนินงาน

- 1.5.1 โครงการนี้เริ่มต้นด้วยการศึกษาทฤษฎี ต่างๆเกี่ยวกับงาน ซึ่งมีเรื่องหลักคือ PID และ Modbus Protocol ที่มีรายละเอียดในบทที่ 2 เรื่องการออกแบบงานต่างๆ ในบทที่ 3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

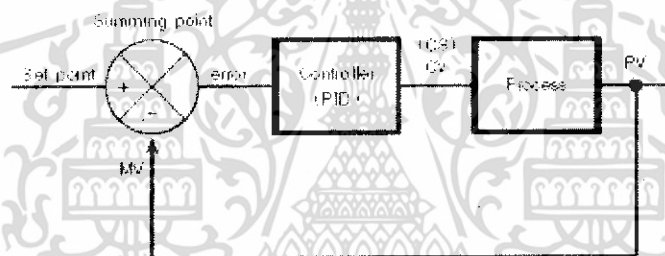
บทที่ 2

ทฤษฎีที่เกี่ยวข้อง

2.1 PID Control

2.1.1 การควบคุมแบบ PID

จากการที่ระบบควบคุมจะต้องแม่นยำ แม่นอนและต้องประหยัดจึงทำให้มีการคิดระบบควบคุมต่างๆ เพื่อใช้ในงานควบคุมให้ได้ผลตามที่ต้องการ การควบคุมแบบPID เป็นการควบคุมแบบหนึ่งที่จะต้องตอบสนองต่อความต้องการ และระบบการควบคุมดังกล่าว ใน Loop ของการควบคุมจะประกอบด้วยส่วนต่างๆดังรูป



รูปที่ 2.1 แสดง Loop การควบคุมที่สมบูรณ์แบบ

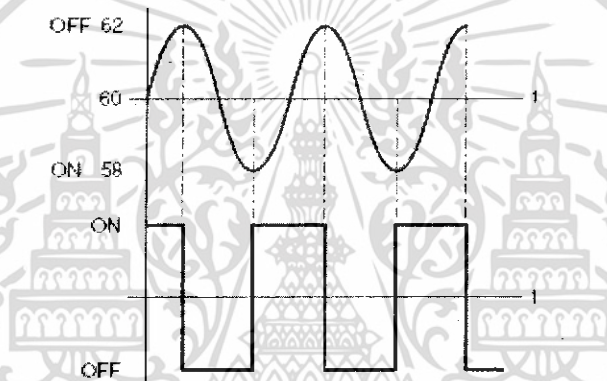
คือจุดรวม (Summing point) วงจรควบคุม(Controller) และตัวดำเนินการ (Process) การทำงานของ Loop การควบคุมจะเริ่มจากการป้อนค่าเป้าหมาย (Set point) ที่ต้องการเข้ามายัง Summing point ทางด้านบวก ส่วนทางด้านลบของ Summing point จะรับค่าจากเอาต์พุต Process หรือค่า MV สัญญาณทั้ง 2 ที่เข้ามายัง Summing point จะถูกหักล้างออกมาเป็นค่า Error ทางด้านเอาต์พุตเพื่อส่งต่อไปให้กับอินพุตของ Controller ให้ทำการควบคุมค่าแรงดันที่ออกทางเอาต์พุตให้กับอินพุตของ Process ส่วนทางด้านเอาต์พุตของ Process ก็จะเปลี่ยนแปลงค่าไปเรื่อยๆเพื่อนำไปหักล้างกับค่า Set point

การทำงานจะวนรูปแบบนี้ไปเรื่อยๆจนกระทั่งค่า Set point และค่า MV เท่ากันเมื่อนำมาหักล้างกัน Error ก็ จะเหลือศูนย์การทำงานของลูปควบคุมก็จะหยุดทันที นั่นหมายความว่า Process ทำงานได้ตรงตามค่า Set point ที่ต้องการพอดี หลังจากนั้นถ้ามีการเปลี่ยนแปลงค่า Set point เกิดขึ้นการทำงานของลูปควบคุมก็จะเข้าหาจุด Set point ไปเรื่อยๆ

2.1.2 ระบบการควบคุมแบบต่างๆ

โดยทั่วไปในระบบควบคุมการทำงานของ Controller จากในรูปที่ 1 ซึ่งเป็นส่วนสำคัญที่ต้องนำมาใช้ควบคุมการทำงานทั้งหมดของระบบ โดยได้แบ่งชนิดของระบบ ออกเป็น 4 ชนิด ได้แก่

- ON OFF control
- Proportional control (P-Control)
- Intergral หรือ Reset control (I-Control)
- Derivative control (D-Control)



รูปที่ 2.2 การทำงานของ ON-OFF Control

2.1.3 การควบคุม ON-OFF Control

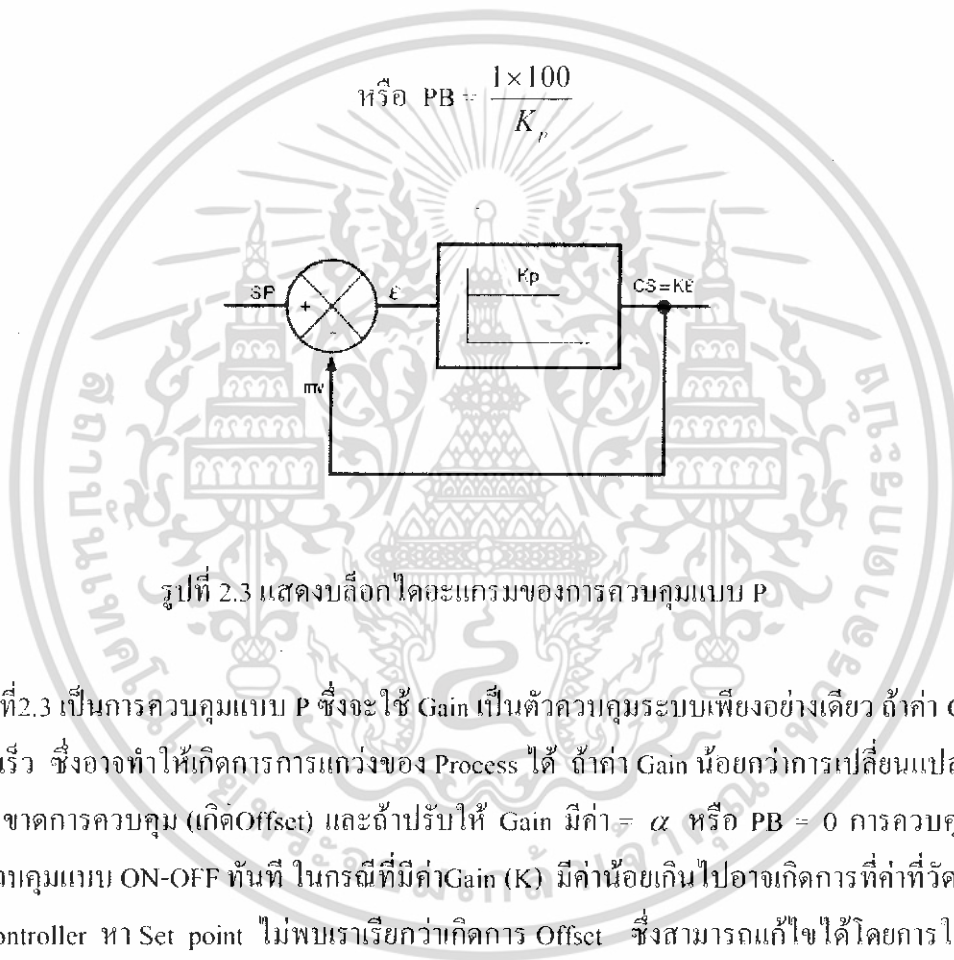
เป็นลักษณะการควบคุมแบบวิธีการปิด-เปิดธรรมดาเหมือนการปิดเปิดสวิตซ์ไฟฟ้าตัวอย่างเช่น เทอร์โมสแตสในเตาเรีดหรือการปิด-เปิด วาล์ว จากรูปที่ 2 จะสังเกตเห็นว่าการทำงาน ON-OFF Control นั้นเอาต์พุตจะไม่คงที่จะมีการเปลี่ยนแปลงตลอดทำให้ไม่สามารถจะใช้ในการควบคุมบางอย่างได้ ปกติการควบคุมแบบ ON-OFF จะมี Dead band ซึ่งหมายความว่าเอาต์พุตของ Controller ที่ออกมาจะมีการแกว่งอยู่ในย่านที่ Controller ยอมได้ เช่น จากรูปจะเห็นว่าอุณหภูมิจะเปลี่ยนแปลงอยู่ระหว่าง 58 องศาเซลเซียส ถึง 62°C ถ้ากำหนด Set point ที่ค่า 60°C การทำงานของ ON-OFF Control จะมี Dead band เท่ากับ $62 - 58 = 4^{\circ}\text{C}$ หรือจะพูดได้ว่า Controller เกิดการแกว่งระหว่าง 62°C ถึง 58°C ซึ่งการแกว่งนี้ จะขึ้นกับค่า Dead time ของ Process เอง สรุปว่าการควบคุมแบบ ON-Off จะใช้ได้ดีกับ Process ที่มี Dead time สั้นแต่มีค่าคงตัวของเวลายาว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.4 การควบคุมแบบ Proportional (P)

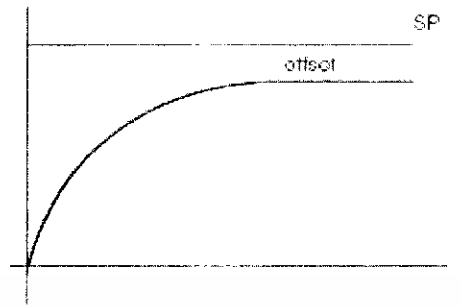
การควบคุมแบบ P เป็นการควบคุมโดยการกำหนดค่า Gain ของ Controller หรือเรียกได้ว่าเป็นการกำหนดอัตราส่วนของเข้าที่พุดต่ออินพุต หรืออาจจะกำหนดออกมาอยู่ในรูปของ Proportional Gain หรือ Proportional Band ก็ได้ โดยมีสมการหาค่า PB ดังนี้

$$\text{Proportional Band} = \frac{1 \times 100}{\text{Proportional Gain}}$$



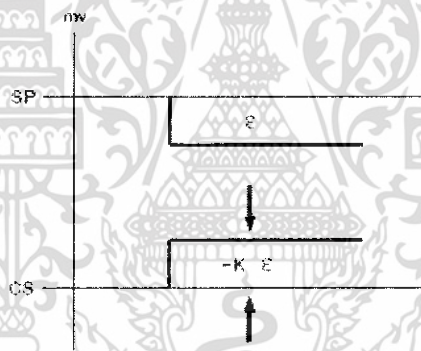
รูปที่ 2.3 แสดงบล็อกไดอะแกรมของการควบคุมแบบ P

จากรูปที่ 2.3 เป็นการควบคุมแบบ P ซึ่งจะใช้ Gain เป็นตัวควบคุมระบบเพียงอย่างเดียว ถ้าค่า Gain มากการเปลี่ยนแปลงจะเร็ว ซึ่งอาจทำให้เกิดการแกว่งของ Process ได้ ถ้าค่า Gain น้อยกว่าการเปลี่ยนแปลงของค่าอาจทำให้ Process ขาดการควบคุม (เกิด Offset) และถ้าปรับให้ Gain มีค่า = ∞ หรือ $PB = 0$ การควบคุมแบบ P จะกลายเป็นการควบคุมแบบ ON-OFF ทั้งนี้ ในกรณีที่มีค่า Gain (K) มีค่าน้อยเกินไปอาจเกิดการที่ค่าที่วัดได้ไม่เท่ากับค่า SP หรือ Controller หา Set point ไม่พบเราเรียกว่าเกิดการ Offset ซึ่งสามารถแก้ไขได้โดยการให้ค่า Bias ที่เหมาะสม



รูปที่ 2.4 กราฟแสดงการเกิดค่า Offset ในระบบการควบคุมแบบ P

ส่วนในรูปที่ 2.4 เป็นกราฟแสดงการเกิด Offset จะเห็นว่าค่าเอาต์พุตไม่สามารถหาค่า Set point (SP) ได้ การเปลี่ยนแปลงของการควบคุมแบบ P จะมีผลตอบสนองออกมา เป็นกราฟดังรูปที่ 2.5

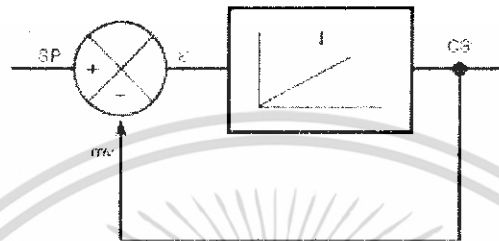


รูปที่ 2.5 กราฟแสดงผลตอบสนองของการควบคุมแบบ P

จะเห็นว่าลักษณะของกราฟจะเป็นเส้นตรงทั้งหมด ค่า SP และค่า CS จะมีลักษณะรูปกราฟเหมือนกันแต่จะแตกต่างกันที่เฟส เนื่องจากการควบคุมแบบ P จะไม่มีค่าเวลาเกี้ยวข้อง เมื่ออินพุตเปลี่ยนเอาต์พุตก็เปลี่ยนตามทันที

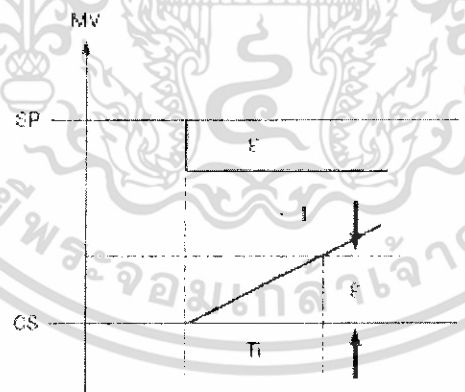
2.1.5 การควบคุมแบบ Integral (I)

การควบคุมแบบ I เป็นการควบคุมแบบอินทิเกรตสัญญาณ ดังในรูป



รูปที่ 2.6 แสดงบล็อกไดอะแกรมของการควบคุมแบบ I

ลักษณะของสัญญาณที่ผ่านการควบคุมแบบ I จะลาดชัน คือจะค่อยๆ ไล่สูงขึ้นจนถึงจุด Set Point การควบคุมแบบ I จะให้ผลตอบสนองต่อการควบคุมได้ดีกว่าการควบคุมแบบ P เนื่องจากการควบคุมแบบ I เมื่ออินพุตเปลี่ยนแปลงค่าๆ จะไม่เปลี่ยนโดยทันที แต่จะค่อยๆ เปลี่ยนไปดังรูปที่ 7



รูปที่ 2.7 แสดงการเปลี่ยนแปลงของการควบคุมแบบ I

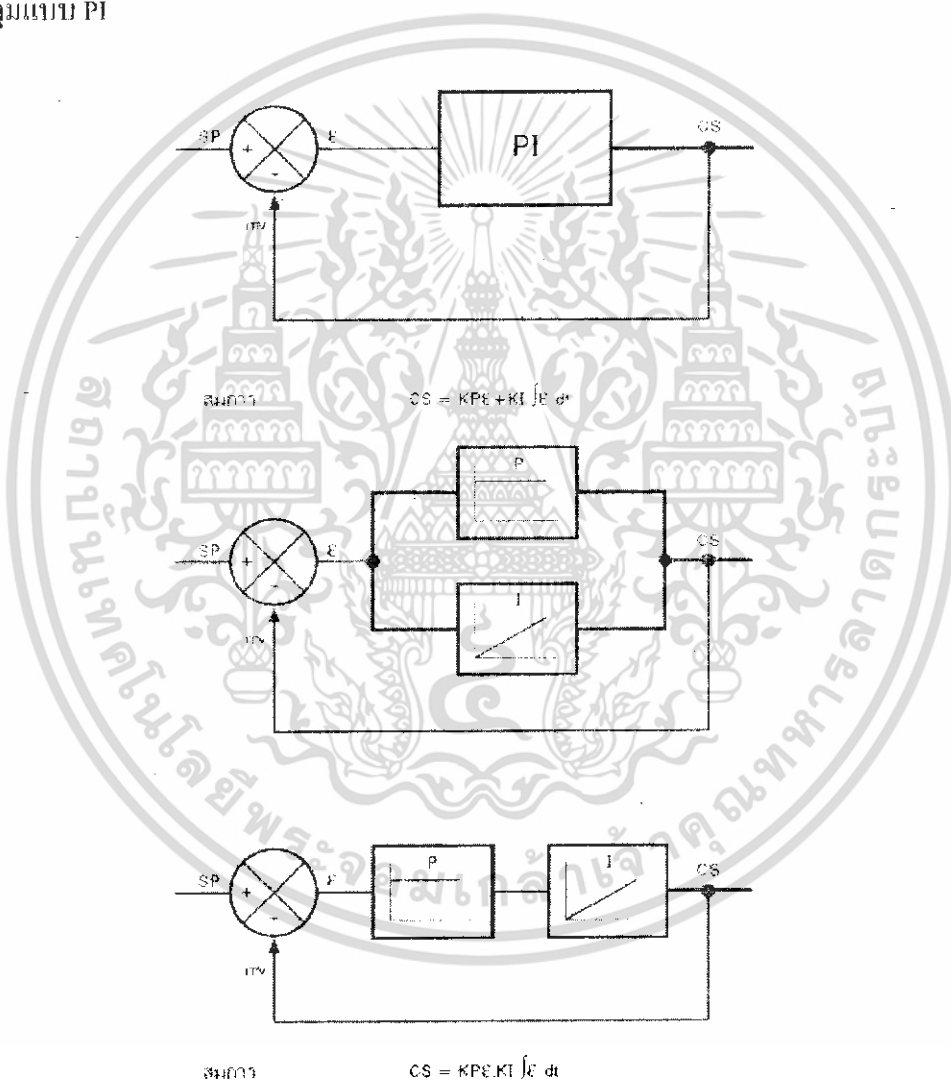
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะเห็นว่าค่าเอาต์พุต หรือ CS จะค่อยๆ ไล่สูงขึ้นจนถึงค่าเป้าหมาย (SP) ในช่วงระยะเวลา T_i ซึ่งเป็นตัวกำหนดค่าเวลาของการควบคุมแบบ I ซึ่งสามารถเปลี่ยนแปลงได้ การควบคุมแบบ I มีลักษณะของสมการคำนวณดังนี้

$$CS = \frac{1}{T_i} \int \sum dt$$

โดยปกติการควบคุมแบบ I จะไม่ใช่ตัวเดียวแต่จะใช้ควบคู่กับแบบ P โดยเรียกว่าเป็น PI Control ดังรูปที่ 2.

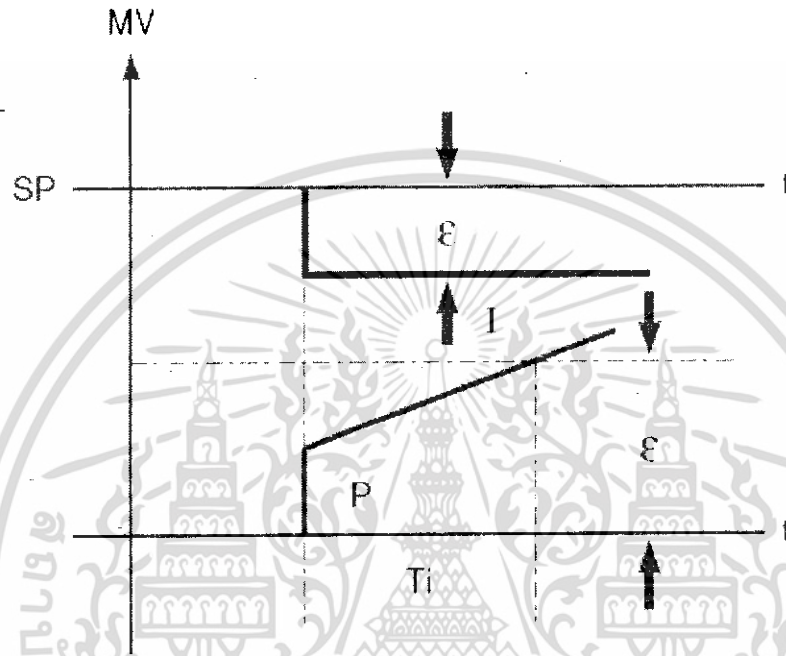
8 เป็นการควบคุมแบบ PI



รูปที่ 2.8 แสดงบล็อกไดอะแกรมของการควบคุมแบบ PI ในแต่ละแบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ซึ่งมีวิธีใช้งานอยู่ 2 แบบ แบบแรกเป็นการต่อแบบขนานโดยนำค่า P กับ I มาบวกกัน ส่วนแบบที่ 2 เป็นการต่อแบบอนุกรมโดยนำค่า P กับ I มาคูณกัน การต่อแบบทั้ง 2 แบบนี้จะช่วยให้การควบคุมมีประสิทธิภาพเพิ่มขึ้นกว่าการควบคุมแบบใดแบบหนึ่งเพียงอย่างเดียว



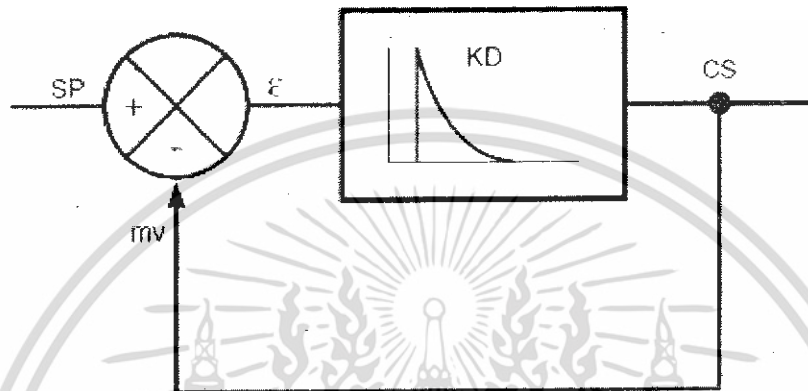
รูปที่ 2. 9 กราฟการเปลี่ยนแปลงของการควบคุมแบบ PI

ในรูปที่ 2.9 แสดงการเปลี่ยนแปลงของการควบคุมแบบ PI จะเห็นว่าค่าสัญญาณเอาต์พุตเริ่มต้นจะขึ้นเป็นเส้นตรงซึ่งเป็นผลของการควบคุมแบบ P จากนั้นเอาต์พุตก็จะเป็นเส้นลาดชัน ซึ่งเป็นผลของการควบคุมแบบ I นั้นเอง จุดเด่นของการควบคุมแบบ PI คือจะช่วยให้การควบคุมเอาต์พุตเข้าหาเป้าหมายได้เร็ว สรุปแล้วการเพิ่มการควบคุมแบบ I เข้าไปในการควบคุมแบบ P นั้น (PI Control) ก็เพื่อแก้ค่า Offset อันเกิดจาก P Control แต่ถ้ากำหนดค่า T_i น้อยเกินไปอาจทำให้ระบบเกิดการขาดเสถียรภาพ หรือเกิดออสซิลเลต (Oscilate)

2.1.6 การควบคุมแบบ Derivative (D)

สมการ

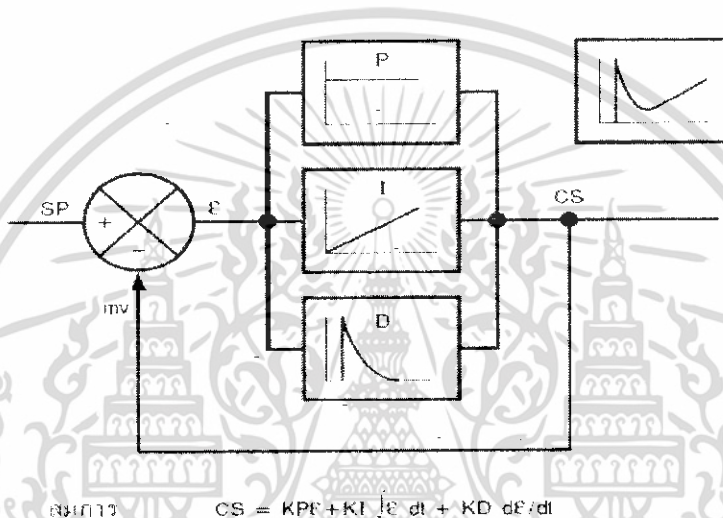
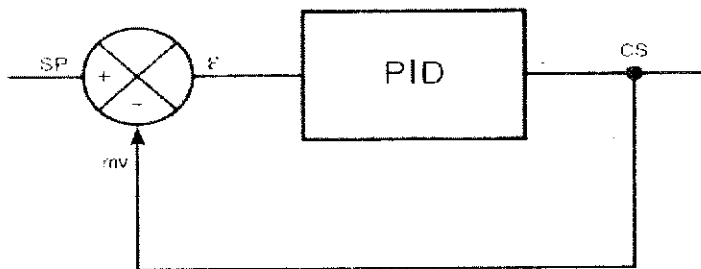
$$CS = KD \frac{d\varepsilon}{dt}$$



รูปที่ 2.10 แสดงบล็อกไดอะแกรมของการควบคุมแบบ D

การควบคุมแบบ (D) เป็นการควบคุมแบบดิฟференเชียล ลักษณะของสัญญาณเอาต์พุตเมื่อผ่านการควบคุมแบบ D แล้วจะเป็นลักษณะของการดิฟференเชียล คือช่วงเริ่มต้นสัญญาณจะขึ้นไปหาค่าสูงสุดก่อนแล้วจะค่อยๆลดลงเป็นทางลาดจนถึงค่าเป้าหมายที่ตั้งไว้

การควบคุมแบบ D นี้จะไม่ใช้ตัวเดียวแต่ จะใช้ร่วมกับแบบ P เป็น PD Control หรือใช้รวมกันเป็น PID Control ดังรูปที่ 2.11

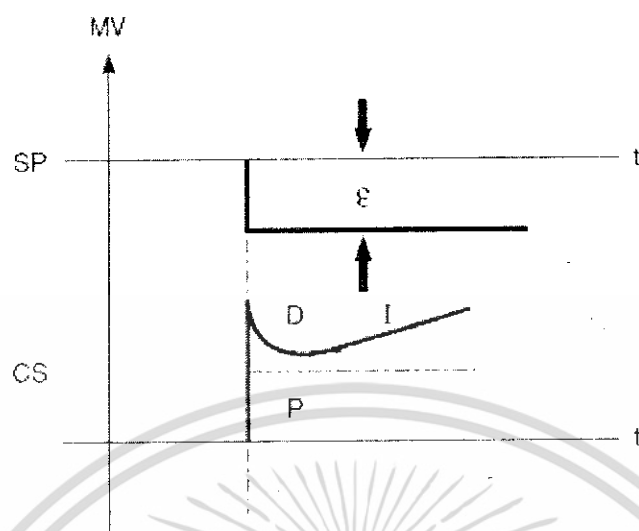


สมการ $CS = KPF + KI \int E dt + KD dE/dt$

รูปที่ 2.11 แสดงบล็อกไดอะแกรมของการควบคุมแบบ PID

ซึ่งใช้การควบคุมแบบ P, แบบ I, และแบบ D มาต่อขนานกันทุกตัว ซึ่งจะช่วยให้ระบบการควบคุมเข้าหาเป้าหมายได้เร็วและดีกว่าการควบคุมแบบอื่นๆ เช่นเมื่อมีการเปลี่ยนแปลงค่าเป้าหมาย (SP) หรือ Process เปลี่ยนไป การควบคุมแบบ D จะทำหน้าที่ปรับค่าเพื่อให้เข้าสู่เป้าหมายได้อย่างรวดเร็ว ส่วนแบบ PI จะทำให้การเข้าสู่เป้าหมายไม่เกิดการแกว่ง

การควบคุมแบบ PID จะใช้กับค่าเป้าหมาย (SP) ที่เปลี่ยนแปลงอยู่ตลอดเวลา หรือจะใช้กับการควบคุมตัว Process ที่มีความเร็วต่อการตอบสนองช้าแต่มีค่า Dead time มากๆเช่น Temperature หรือตัว Process ที่มีการเปลี่ยนแปลงตลอดเวลา หรือการเกิดการรบกวนจากภายนอกของ Process ได้ง่าย ซึ่งลักษณะของสัญญาณเอาต์พุต (CS) จะมีการเปลี่ยนแปลงไปตามรูปที่ 2.12



รูปที่ 2.12 แสดงกราฟการเปลี่ยนแปลงของการควบคุมแบบ PID

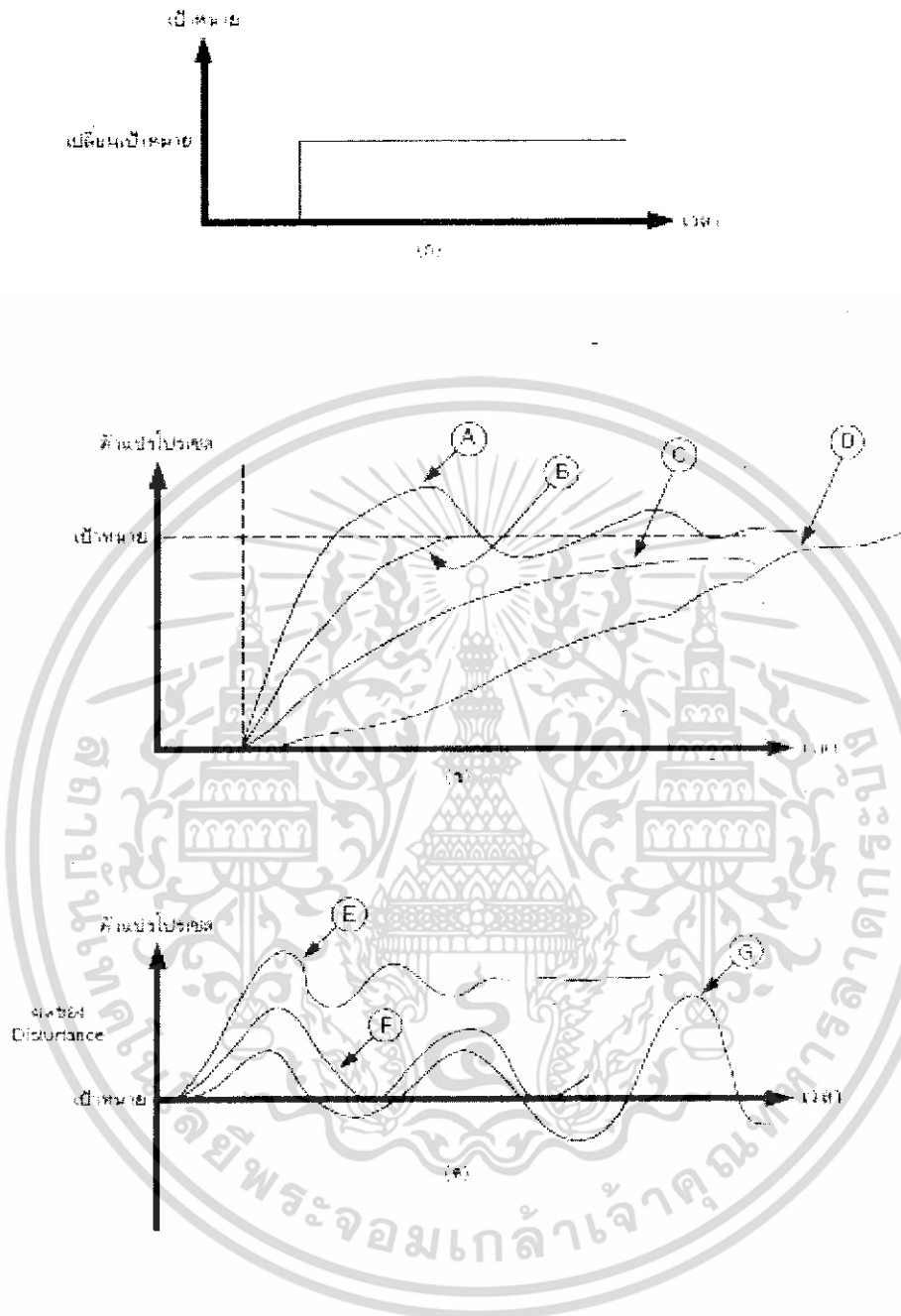
จะเห็นว่าสัญลักษณ์ของสัญญาณเอาต์พุต (CS) จะมีการผสมกันของแบบ P, แบบ I, และแบบ D รวมกันอยู่
อย่างเห็นได้ชัด

2.2 การปรับค่าในระบบควบคุมแบบ PID

2.2.1 ผลตอบของการควบคุมที่ดี

ในระบบควบคุมที่ออกแบบมาดีจะพยายามรักษาให้ค่าตัวแปร โพรเซสมีค่าเท่ากับค่าเป้าหมายอยู่เสมอ
ในกรณีที่เกิด Disturbance ในระบบหรือมีการเปลี่ยนค่าเป้าหมายจะทำให้ตัวแปร โพรเซสมีค่าต่างจากค่าเป้าหมายไป
ขณะหนึ่ง จากนั้นตัวควบคุมจะพยายามควบคุมให้ตัวแปร โพรเซสมีค่าเท่ากับเป้าหมายนี้ในที่สุด ลักษณะการนำ
ค่าตัวแปร โพรเซสให้เข้าใกล้ค่าเป้าหมายนี้ จะแตกต่างกันตามคุณสมบัติของระบบควบคุม บางระบบควบคุมอาจ
ควบคุมให้ตัวแปร โพรเซสส่งเข้าหาเป้าหมายได้เร็ว แต่บางระบบอาจทำได้ดีกว่า เราสามารถทดสอบความสามารถ
ของระบบควบคุมนี้โดยดูที่ผลการตอบของการควบคุมตามในรูปที่ 2.13

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.13 แสดงกราฟผลตอบของระบบควบคุม

จากระบบควบคุมแบบ PID ถ้าลองเปลี่ยนค่าเป้าหมายตามรูปที่ 2.13 (ก) ตัวควบคุมจะพยายามควบคุมให้ตัวแปรโปรเซสวิ่งเข้าหาเป้าหมายนั้น ผลตอบของตัวแปรโปรเซสจะมีหลายแบบดังแสดงในรูปที่ 2.13 (ข) ซึ่งสามารถอธิบายแต่ละจุดได้ดังนี้

- จุด A มี Overshoot และการแกว่ง เข้าหาเป้าหมายได้ไม่ดี
- จุด B มีการตอบสนองเร็ว โปรเซสเข้าหาเป้าหมายได้เร็ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- จุด C ตัวแปรโพรเซสมีค่าไม่เท่ากับค่าเป้าหมาย แม้เวลาจะผ่านไปนานเรียกว่าเกิด offset
- จุด D มีการตอบรับช้ามาก โพรเซสเข้าหาเป้าหมายได้ช้า

เมื่อตัวแปรโพรเซสหยุดนิ่ง ที่ค่าเป้าหมาย และในขณะนั้นเกิดมี Disturbance เข้ามารบกวนในโพรเซส (Disturbance ได้แก่ การเปลี่ยนแปลงของโหลดหรือการเปลี่ยนแปลงของสภาพแวดล้อม เป็นต้น) ผลของโพรเซสมีหลายแบบดังแสดงในรูปที่ 2.13(ค)

- จุด E เกิด offset ได้ค่าผิดจากเป้าหมายเดิม
- จุด F เกิดการแกว่งเล็กน้อยก่อนกลับสู่ค่าเป้าหมายเดิม
- จุด G เกิดการแกว่งและขาดเสถียรภาพ

เมื่อพิจารณาผลตอบของการควบคุมชนิดต่างๆ เหล่านี้จึงจะพอสรุปได้ว่าระบบการควบคุมที่ดีจะต้องมีคุณสมบัติดังนี้ คือ มีเสถียรภาพไม่เกิดการแกว่ง (Oscillation) เมื่อถูกการกระตุ้นและการตอบรับการเปลี่ยนค่าเป้าหมาย Disturbance ได้รวดเร็วไม่เกิด offset

2.2.2 ความยากง่ายในการควบคุมของโพรเซส

ค่า Dead Time เป็นศัตรูตัวร้ายของตัวควบคุม ลักษณะสมบัติของโพรเซสต่างๆ ไป มักจะมีค่า Dead Time ปรากฏให้เห็นจากรูปคลื่นของผลโพรเซสต่อ Step input เราสามารถหาค่า dead time (LE) และค่าคงตัวเวลา (TE) โดยประมาณได้ ซึ่งอัตราส่วน LE/TE นี้จะเป็นค่าที่ใช้ประเมินความยากง่ายในการควบคุม และใช้เลือกแบบการควบคุมที่เหมาะสมกับโพรเซสตามในตารางที่ 2.2.2

LE/TE	แบบการควบคุมที่เหมาะสม
$LE/TE < 0.2$	ON-OFF, P, PI
$0.2 < LE/TE < 1.0$	PI, PID
$1.0 < LE/TE$	Feed Forward, Computer Control

ตารางที่ 2.2.2 แสดงการควบคุมที่เหมาะสมกับโพรเซส

2.2.3 ผลของ PID ต่อเสถียรภาพของระบบ

ในระบบควบคุมแบบป้อนกลับ ซึ่งใช้ตัวควบคุมแบบ PID นั้น ถ้าทดลองเปลี่ยนค่า PB , T_i และ T_d จะมีผลต่อผลตอบของระบบการควบคุมดังนี้

ผลของการควบคุมแบบ P เมื่อลดค่า PB ลง ทำให้อัตราขยายสูงขึ้นจะมีผลทำให้ค่า offset ลดลง รูปคลื่นช่วงเวลาของการแกว่งเล็กน้อย และอัตราส่วนของช่วงกว้างการแกว่งเพิ่มขึ้น ระบบขาดเสถียรภาพมากขึ้น

ผลการควบคุมแบบ I เมื่อให้การควบคุมแบบ P และ D คงที่แล้วทดลองลดค่า T_i จะมีผลทำให้ค่า offset หายไป ผลตอบจะเร็วขึ้น และอัตราส่วนของช่วงกว้างการแกว่งเพิ่มขึ้นระบบขาดเสถียรภาพมากขึ้น

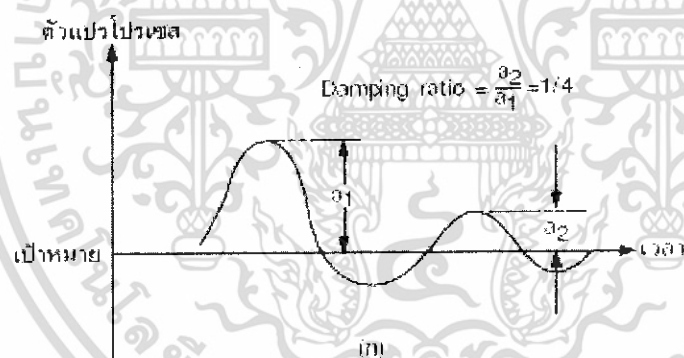
ผลการควบคุมแบบ D เมื่อให้การควบคุมแบบ P และ I คงที่แล้ว ทดลองเพิ่มค่า T_D ให้ยาวขึ้นจะมีผลทำให้ อัตราส่วนของช่วงกว้างการแกว่งลดลงระบบมีเสถียรภาพมากขึ้น และรูปคลื่นช่วงเวลาของการแกว่งสั้นลง

ตามปกติการใช้ค่าเวลา T_D ให้ยาวมีแนวโน้มที่จะให้ระบบมีเสถียรภาพก็จริง แต่ก็มีจุดอ่อนตรงตอบรับต่อ นอยส์ (Noise) ได้ง่าย ทำให้ผลตอบของระบบไวเกินไป

2.2.4 คุณภาพของการควบคุม

ดังที่ได้กล่าวมาแล้วว่า ระบบควบคุมที่ดีจะต้องมีคุณสมบัติ คือมีเสถียรภาพผลตอบเร็ว และไม่เกิด Offset แต่จะใช้อะไรในการตัดสินว่าการควบคุมไหนจะดีที่สุดที่สุดนั้น จะต้องมีเกณฑ์ตัดสินคุณภาพการควบคุม ซึ่งจะเห็นว่าการควบคุมโปรเซส อัตราส่วนช่วงกว้างการแกว่ง (Damping ratio criterion) เป็นที่นิยมใช้มากที่สุด เพราะสามารถวัดผลได้ง่าย การควบคุมที่ดีจะให้อัตราส่วนช่วงกว้างการแกว่งประมาณ 25% หรือ 1 ต่อ 4 ตามรูปที่ 2.14

(ก)



(ข)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

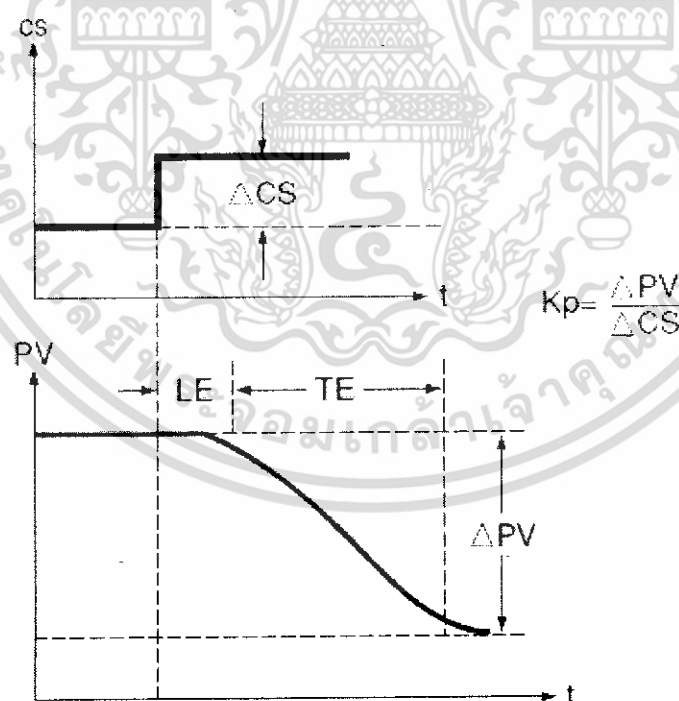
จากรูปที่ 2.14 (ข) เมื่อโปรเซสแกว่งเข้าหาเป้าหมาย จะเห็นว่าพื้นที่การควบคุมที่ดีจะต้องแกว่งเข้าหาเป้าหมายประมาณ 4 ลูกคลื่น เพื่อให้ระบบควบคุมสามารถเข้าหาเป้าหมายได้เร็วและไม่เกิด offset ถ้าหากพื้นที่การควบคุมแกว่งมากกว่านี้จะทำให้ระบบขาดเสถียรภาพใช้งานได้ไม่ดีเท่าที่ควร

2.2.5 วิธีการปรับค่า PID

ได้มีนักคณิตศาสตร์คิดวิธีที่จะหาทางตั้งค่า PID เมื่อให้ได้การควบคุมที่มีคุณภาพที่ดีที่สุดหลายวิธีด้วยกัน แต่ในที่นี้ขอยกตัวอย่างเพียงแค่ 2 วิธีเท่านั้น ซึ่งในทางปฏิบัติเป็นวิธีที่นิยมใช้กันมาก การปรับค่า PID เพื่อให้ได้ค่า Damping ratio เท่ากับ 25% สามารถทำได้ 2 วิธี ดังนี้

วิธีที่ 1

1. ให้เปลี่ยนระบบควบคุมเป็นวงรอบเปิด (Open Loop)
2. หา Process Characteristic โดยเปลี่ยนค่าสัญญาณควบคุมไป ΔCS
3. แล้วบันทึกการปฏิกิริยาของตัวแปรโปรเซส
4. หา Process gain (K_p), dead time (LE) และค่าคงตัวเวลา (TE) จาก Process Characteristic ในรูปที่ 2.15 และคำนวณหา K_p ตามสูตร ดังนี้



รูปที่ 2.15 แสดงลักษณะของรูปคลื่นโปรเซสในการปรับค่า PID ในวิธีที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

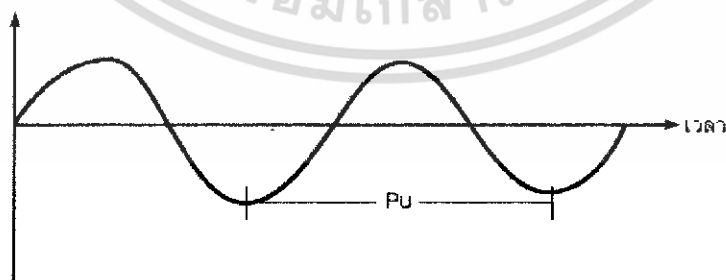
5. นำค่า K_p , LE , TE ที่หาได้ไปคำนวณหา PB , T_I และ T_D จากตารางที่ 2

ชนิดการควบคุม	$PB(\%)$	T_I (นาที)	$T_D(\%)$
P	$100 K_p \cdot LE/TE$		0
PI	$110 K_p \cdot LE/TE$	$3.3 LE$	0
PID	$83 K_p \cdot LE/TE$	$2 LE$	$0.5 LE$

ตารางที่ 2.2.3 แสดงการคำนวณหาค่า PB , T_I และ T_D ในการปรับค่า PID

วิธีที่ 2

1. ให้ระบบควบคุมเป็นแบบวงรอบปิด (Closed loop)
2. ตั้ง T_I สูงสุด (max) และ T_D ต่ำสุด (0 หรือ min) ใช้การควบคุมแบบ P อย่างเดียว
3. ครั้งแรกตั้งค่า PB ไว้ที่ค่าสูงสุด แล้วลดค่า PB ลงมา ลองเปลี่ยนค่าเป้าหมายเพื่อดูผลตอบแทนและลดค่า PB ให้ต่ำลงเรื่อยๆ จนถึงค่าที่เมื่อเปลี่ยนค่าเป้าหมายไปเล็กน้อย จะทำให้โปรเซสเกิดการแกว่งต่อเนื่องไปตลอดค่า PB ในขณะนั้นเรียกว่า PB_u (Ultimate proportional band)
4. หาคาบเวลาในการแกว่ง ให้เท่ากับ P_u (ตามในรูปที่ 2.16)

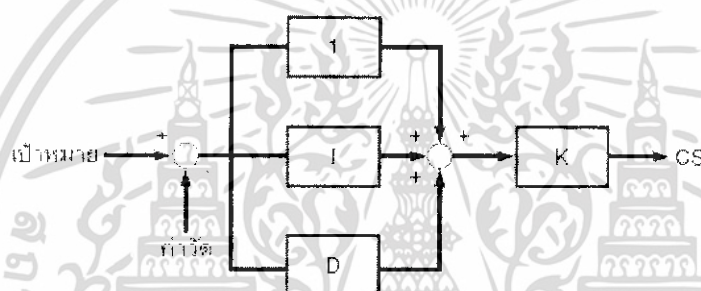


รูปที่ 2.16 แสดงการหาคาบเวลาในการแกว่งของการปรับค่า PID ในวิธีที่ 2

5. นำค่า PBU และ PU ที่หาได้ไปใช้คำนวณหาค่า PB, T_I และ T_D จากตารางที่ 3

ชนิดการควบคุม	PB	T_I	T_D
P	$2 PBU$	∞	0
PI	$2.2 PBU$	$0.83 PU$	0
PID	$1.7 PBU$	$0.5 PU$	$0.125 PU$

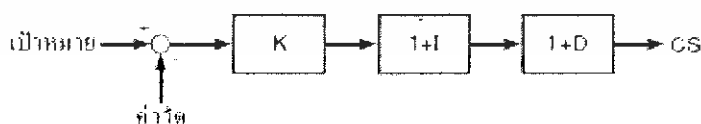
ตารางที่ 2.2.4 แสดงการคำนวณหาค่า PB, T_I และ T_D ในการปรับค่า PID ในวิธีที่ 2



รูปที่ 2.17 แสดงบล็อกไดอะแกรมของตัวควบคุมแบบดีที่สุด

2.2.6 การรบกวนซึ่งกันและกันของค่า PID

จากรูปที่ 2.17 แสดงบล็อกไดอะแกรมของตัวควบคุมแบบอุดมคติ ซึ่งค่า PID จะสามารถตั้งได้อย่างอิสระไม่มีการรบกวนซึ่งกันและกัน ตัวควบคุมแบบนี้มักจะมีราคาแพง โดยทั่วไปตัวควบคุมที่มีขายในท้องตลาดมักจะตัดแปลงวงจรให้ง่ายลง โดยมีบล็อกไดอะแกรมดังรูปที่ 2.18



รูปที่ 2.18 แสดงบล็อกไดอะแกรมของตัวควบคุมที่มีขายทั่วไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะเห็นว่า I และ D ก่อน โทรล ไม่เป็นอย่างอุดมคติ และจะมีการรบกวนซึ่งกันและกัน (Mutual interfere) ซึ่งได้แก่การเปลี่ยนค่า T_1 จะมีผลทำให้ PB และ T_D เปลี่ยนแปลงไป หรือถ้าเปลี่ยนค่า T_D ก็จะมีผลทำให้ PB และ T_1 เปลี่ยนไปเช่นกัน

ดังนั้นเมื่อทำการปรับค่า PID ที่เหมาะสมได้แล้ว ให้ทำการคำนวณหาค่า PB , T_1 และ T_D จากวิธีการปรับค่า PID ทั้ง 2 วิธีข้างต้น ซึ่งจริงๆ แล้วค่าเหล่านี้ยังนำมาตั้งที่ตัวควบคุมไม่ได้ ก่อนอื่นต้องกำหนดให้ PB , T_1 , T_D เป็นค่าตั้งที่ตัวควบคุม (Dial Setting) และให้ PB' , T_1' , T_D' เป็นค่าที่คำนวณได้ (Effective value) ทั้งสองค่าจะมีความสัมพันธ์กันตามสูตรดังนี้

$$PB = \mu PB'$$

$$T_1 = \frac{T_1'}{\mu}$$

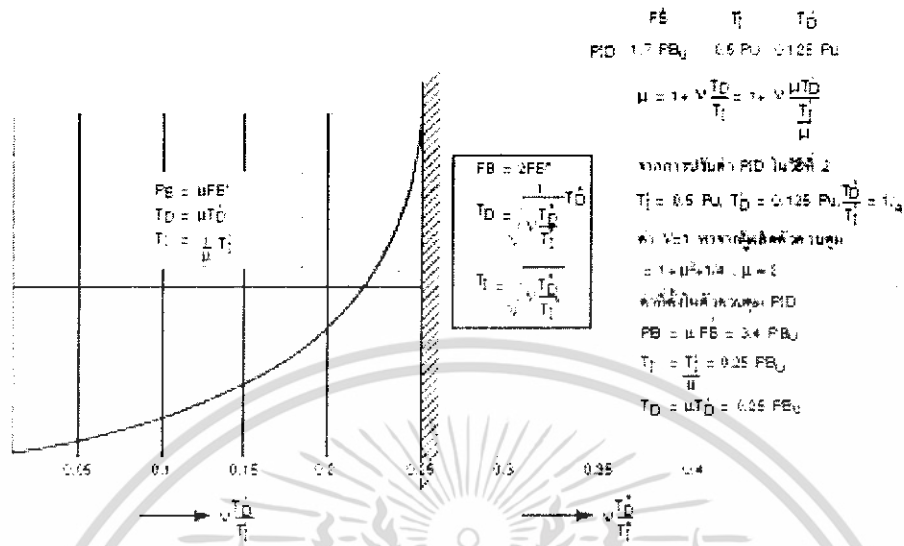
$$T_D = \mu T_D'$$

โดยที่ μ คือสัมประสิทธิ์การรบกวนซึ่งกันและกัน (Mutual Interference Coefficient)

ค่า μ นี้จะมีค่าไม่คงที่แน่นอน จะขึ้นกับอัตราส่วนของ T_1 กับ T_D และขึ้นกับชนิดของตัวควบคุมด้วย μ นี้สัมพันธ์กับค่า

$$V = \frac{T_D'}{T_1'}$$

(V คือค่าคงตัวค่าหนึ่งซึ่งบริษัทผู้ผลิตตัวควบคุมจะต้องกำหนดให้) ซึ่งแสดงตามกราฟใน รูปที่ 2.19



รูปที่ 2.19 กราฟแสดงการหาค่า μ

ต้องมีค่าน้อยกว่า 0.25 ถ้ามีค่ามากกว่าจะหาความสัมพันธ์ไม่ได้

2.2.7 วิธีการตั้งค่า PID (Dial Setting) ในตัวควบคุมที่มีการรบกวนกัน

1. กำหนดค่า PB' , T_I' , T_D' ที่ได้จาก Optimum tuning method
2. กำหนดค่า $V \frac{T_D'}{T_I'}$ โดยที่ V หาจากผู้ผลิตตัวควบคุม
3. หา μ จากกราฟในรูปที่ 7
4. กำหนดค่า PB , T_I , T_D จากสูตรข้างบน
5. นำค่า PB , T_I , T_D ที่คำนวณได้ไปตั้งที่ตัวควบคุม

สรุปแล้วรายละเอียดที่ต้องใช้เกี่ยวกับการปรับค่า PID จะประกอบด้วย 3 อย่าง คือ ค่า PB , T_I , T_D ซึ่งเป็นค่าที่ได้จากการคำนวณหรือเป็นค่าที่ได้จากการทดลองสุ่มตัวอย่างก็ได้ เพื่อที่จะนำค่าเหล่านี้ไปใช้ปรับแต่งในระบบควบคุมแบบ PID ให้ได้ผลตอบของการควบคุมที่มีประสิทธิภาพมากที่สุดปกติการปรับค่า PID ให้ได้ผลดีนั้นจะต้องทำการทดลองปรับหลายๆ ครั้งจนกว่าจะได้ค่า PB , T_I , T_D ที่เหมาะสมพร้อมที่จะนำไปใช้งานจริงได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3 MODBUS PROTOCOL

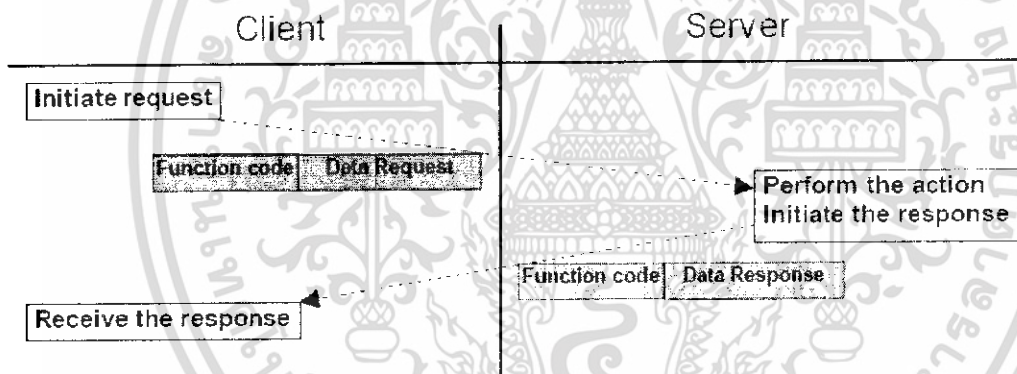
MODBUS เป็นโพรโตคอลที่นิยมใช้กันมากในภาคอุตสาหกรรม โดยเฉพาะในระบบควบคุมอัตโนมัติ ด้วยโครงสร้างที่ได้รับการออกแบบให้ใช้งานง่าย สามารถนำไปใช้กับระบบรับส่งข้อมูลได้หลายประเภท เช่น ระบบรับส่งข้อมูลแบบอนุกรม หรือเครือข่าย LAN ทำให้ได้รับความนิยมใช้งานอย่างกว้างขวาง โดยจะสังเกตเห็นได้ว่า อุปกรณ์ที่สามารถติดต่อสื่อสารได้ในภาคอุตสาหกรรม ไม่ว่าจะเป็นอุปกรณ์ควบคุมอัตโนมัติ เครื่องมือวัด ชนิดต่างๆ ส่วนใหญ่จะสามารถสื่อสารด้วยโพรโตคอล MODBUS ได้ทั้งนั้น หรือแม้กระทั่งในงานออกแบบระบบ ไมโครคอนโทรลเลอร์ ที่มีพอร์ตสื่อสารข้อมูลอนุกรมให้อยู่แล้ว ก็สามารถเขียนโปรแกรมควบคุมพอร์ตให้สื่อสารข้อมูลด้วยโพรโตคอล MODBUS ได้ ซึ่งนั่นหมายความว่า เราสามารถออกแบบระบบ ไมโครคอนโทรลเลอร์ของเราเอง ให้สามารถติดต่อพูดคุยกับอุปกรณ์มาตรฐานที่มีขายอยู่ในท้องตลาดได้ ทำให้เป็นการเพิ่มระดับขีดความสามารถในงานออกแบบระบบ ไมโครคอนโทรลเลอร์ให้สูงขึ้นไปอีก และหากได้ทำความเข้าใจกับโพรโตคอลตัวนี้แล้ว จะพบว่าไม่ยากเลยที่จะทดลองพัฒนาขึ้นมาใช้งานเอง

2.3.1 ลำดับชั้นของ MODBUS ในแบบจำลอง OSI

MODBUS เป็นโพรโตคอลรับส่งข้อมูลในลำดับชั้นแอปพลิเคชัน หรือ application layer เป็นลำดับชั้นที่ 7 ซึ่งจัดเป็นลำดับชั้นสูงสุดในแบบจำลอง OSI การที่ MODBUS ถูกออกแบบให้อยู่ในลำดับชั้นสูงสุดนั้น มีข้อดีโดยทำให้สามารถนำไปใช้กับระบบนำส่งข้อมูลหรือเครือข่ายแบบต่างๆ ได้หลากหลายชนิด เนื่องจากความแตกต่างเหล่านี้เป็นเรื่องของลำดับชั้นที่อยู่ถัดลงมา โดยเฉพาะในลำดับชั้นกายภาพ หรือ Physical Layer ซึ่งเป็นตัวกำหนดลักษณะรูปแบบทางกายภาพของตัวกลางนำสัญญาณข้อมูล ซึ่งไม่มีผลต่อลำดับชั้นที่อยู่สูงกว่า

ดำเนินการกับข้อมูลที่อยู่ในส่วนที่สอง PDU มีความเป็นอิสระ ไม่ขึ้นอยู่กับลักษณะทางกายภาพของบัสหรือเครือข่าย ซึ่งเมื่อนำไปใช้จริงบนบัสหรือเครือข่ายชนิดใด ก็จะเพิ่มส่วนประกอบสำคัญอีก 2 ส่วนลงไป ได้แก่ ข้อมูลแอดเดรส (additional address) และอีกส่วนหนึ่งคือข้อมูลที่ได้จากระบวนการเข้ารหัสเพื่อใช้ตรวจสอบความถูกต้องของข้อมูลที่ได้รับ (error check) เรียก PDU ที่เพิ่มส่วนประกอบ 2 ส่วนนี้ว่า Application Data Unit (ADU)

ในระบบโปรโตคอล MODBUS โคลเอนท์จะเป็นตัวเริ่มต้นการสื่อสาร โดยสร้างเฟรมข้อมูล ADU ส่งไปยังเซิร์ฟเวอร์ เพื่อแจ้งให้ทราบว่าต้องการทำอะไร ตามหมายเลขฟังก์ชันที่ระบุใน ADU ส่วนข้อมูลที่ระบุได้จากหมายเลขฟังก์ชันนั้น ใช้ประกอบการทำงานตามฟังก์ชัน เช่น เก็บพารามิเตอร์บอกตำแหน่งแอดเดรสของข้อมูลที่จะทำการอ่านด้วยฟังก์ชันอ่านข้อมูล ข้อมูลมีทั้งแบบลอจิก (0) หรือ (1) และแบบรีจิสเตอร์หรืออนุภาค (เช่น ค่า 134.25) รวมทั้งข้อมูลรหัสตัวอักษร ซึ่งก็จัดเป็นข้อมูลแบบรีจิสเตอร์ประเภทหนึ่ง

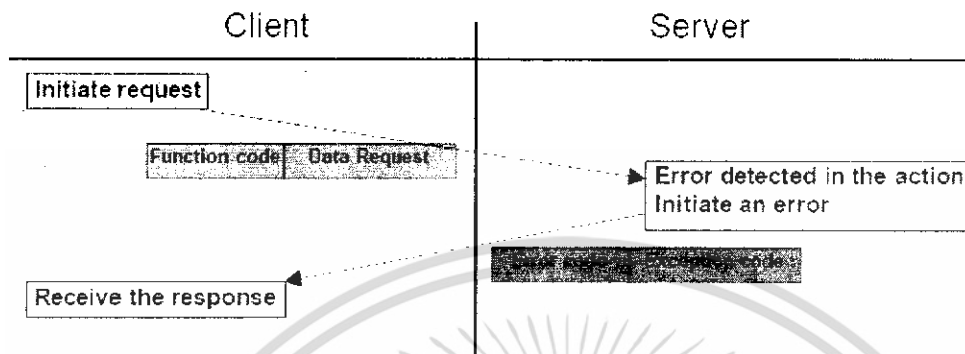


รูปที่ 2.3.3 การรับส่งข้อมูลระหว่างโคลเอนท์กับเซิร์ฟเวอร์

สำหรับในบางฟังก์ชัน อาจไม่ปรากฏฟิลด์ข้อมูล หรืออีกนัยหนึ่งก็คือ เฟรมข้อมูลสำหรับฟังก์ชันนั้นๆ ไม่ต้องอาศัยข้อมูลประกอบเพิ่มเติมแต่อย่างใด

รูปที่ 2.3.3 แสดงขั้นตอนการรับส่งเฟรมข้อมูลระหว่างโคลเอนท์กับเซิร์ฟเวอร์ เริ่มต้นโดย โคลเอนท์ส่งเฟรมข้อมูลระบุหมายเลขฟังก์ชัน ประเภท ตำแหน่ง และจำนวนข้อมูลที่ต้องการ (data request) ไปที่เซิร์ฟเวอร์ เมื่อเซิร์ฟเวอร์ได้รับเฟรมข้อมูลถูกต้องครบถ้วน และสามารถประมวลผลฟังก์ชันตามที่ระบุได้เสร็จเรียบร้อยแล้ว ก็จะส่งเฟรมข้อมูลที่บรรจุข้อมูลที่ต้องการ (data response) กลับมาให้โคลเอนท์ตัวนั้น โดยที่ส่วนแรกของเฟรมข้อมูลจะระบุหมายเลขฟังก์ชันเดิมที่ได้รับจากโคลเอนท์ แต่หากเกิดความผิดพลาดขึ้นมา ซึ่งอาจจะเป็นการที่เฟรมข้อมูลที่

เซิร์ฟเวอร์ได้รับมีความผิดพลาด หรือเฟรมข้อมูลถูกต้อง แต่เซิร์ฟเวอร์ไม่สามารถทำงานตามฟังก์ชันที่ระบุได้ เซิร์ฟเวอร์ก็จะส่งเฟรมข้อมูลที่บรรจุข้อมูลหมายเลขความผิดพลาดหรือ Exception code ไป



รูปที่ 2.3.4 เกิดความผิดพลาดในการรับส่งข้อมูล

ให้กับโคลอนท์แทน เพื่อแจ้งให้ทราบว่าเกิดปัญหาอะไร ในกรณีนี้ ส่วนแรกของเฟรมข้อมูลที่เซิร์ฟเวอร์ส่งกลับมานั้น เรียกว่า Exception function code ที่เป็นหมายเลขฟังก์ชันเดิม แต่รับค่าบิตบนสุด (MSB) ให้เป็น 1 เนื่องจากในอดีต การใช้งานโปรโตคอล MODBUS เริ่มต้นในระบบสื่อสารข้อมูลแบบอนุกรม ที่มีการกำหนดให้ขนาดของเฟรมข้อมูลในการสื่อสารซึ่งก็คือ ADU มีความยาวไม่เกิน 256 ไบต์ และเมื่อหักเอาส่วนที่เป็น address ขนาด 1 ไบต์ และรหัสตรวจสอบความผิดพลาด หรือ error check ขนาด 2 ไบต์ออก ก็จะเหลือเป็นเฟรมข้อมูล PDU ขนาด 253 ไบต์ และขีดค่านี้เป็นมาตรฐาน จากนั้น เมื่อมีการนำเอาโปรโตคอล MODBUS ไปใช้กับการสื่อสารในระบบอื่นๆ เช่น TCP/IP ค่าความยาวของ PDU ค่านี้ ก็ถูกนำไปใช้อ้างอิงด้วยเช่นกัน ระบบการจัดแบ่งกลุ่มข้อมูลที่เข้าร่วมกับการสื่อสารด้วยโปรโตคอล

Primary tables	Object type	Type of access	Comments
Discretes Input	Single bit	Read-Only	This type of data can be provided by an I/O system.
Coils	Single bit	Read-Write	This type of data can be alterable by an application program.
Input Registers	16-bit word	Read-Only	This type of data can be provided by an I/O system.
Holding Registers	16-bit word	Read-Write	This type of data can be alterable by an application program.

ตารางที่ 2.3.1 แสดงกลุ่มข้อมูลชนิดต่างๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MODBUS มีลักษณะดังในรูปที่ 5 แบ่งออกเป็น 4 กลุ่ม ดังนี้

1. Discrete Input เป็นกลุ่มข้อมูลแบบบิต ใช้อ่านอย่างเดียว แก้ไขไม่ได้ เปรียบเสมือนกับหน่วยอินพุตที่เชื่อมต่อกับค่าสถานะจากวงจรอินพุตที่เป็นฮาร์ดแวร์ภายนอก รวมทั้งอาจใช้เป็นพารามิเตอร์แสดงสถานะต่างๆ ของระบบ ดังเช่น ภายในตัว PLC จะมีหน่วยความจำพิเศษแบบบิตที่ใช้บอกค่าสถานะต่างๆ เช่น ฟัลส์สัญญาณนาฬิกา, ฟัลส์ที่มีค่าเป็น 1 เฉพาะในลูปรการทำงานแรก, บิตแสดงโหมดการทำงานของ PLC เป็นต้น ซึ่งบิตพิเศษเหล่านี้ ผู้ใช้งานไม่สามารถเปลี่ยนแปลงแก้ไขได้ ทำได้เพียงอ่านค่าเท่านั้น

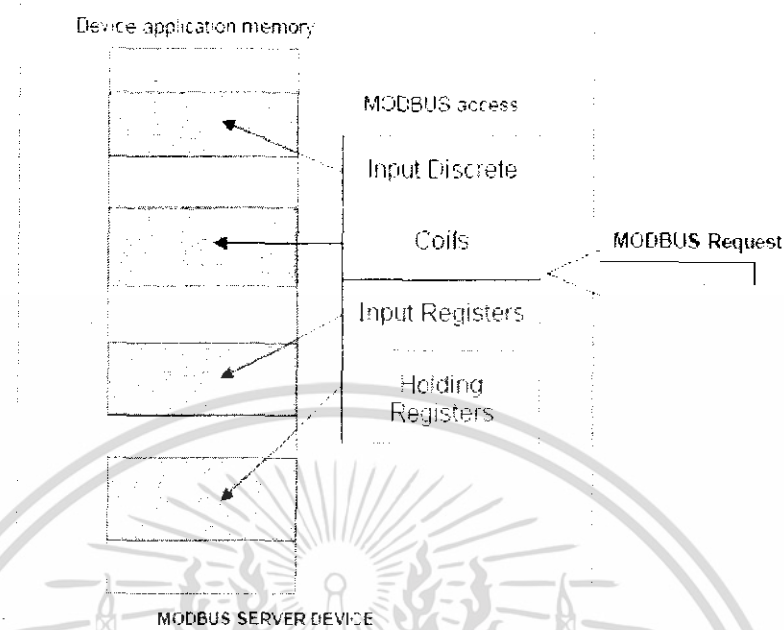
2. Coils เป็นกลุ่มข้อมูลแบบบิตเหมือนกัน แต่สามารถอ่านและเขียนได้ ใช้เป็นประโยชน์สำหรับการประมวลผลฟังก์ชันต่างๆ เช่น ใช้เก็บค่าตัวแปรแบบบิตชั่วคราวในขณะประมวลผล

3. Input Registers เป็นกลุ่มข้อมูลแบบรีจิสเตอร์ที่อ่านได้อย่างเดียว เปรียบเสมือนกับหน่วยความจำที่เชื่อมต่อกับค่าแอนาลอกจากอุปกรณ์ฮาร์ดแวร์ภายนอก เช่น วงจรแอนาลอกอินพุต

4. Holding Registers เป็นกลุ่มข้อมูลแบบรีจิสเตอร์ที่สามารถอ่านและเขียนได้ เช่นเดียวกับ Coils คือ ใช้ประโยชน์ในการประมวลผลฟังก์ชันต่างๆ

ภายในเฟรมข้อมูล MODBUS PDU ข้อมูลถูกอ้างอิงตำแหน่งด้วยหมายเลขตั้งแต่ 0 ถึง 65535 โดยข้อมูลภายในกลุ่มข้อมูลแต่ละกลุ่มจะถูกกำหนดหมายเลขอ้างอิงเริ่มต้นที่เลข 1 เหมือนกันทุกกลุ่ม ค่าข้อมูลในตำแหน่งที่ถูกระบุถึง จะถูกอ่านและส่งต่อไปยัง Device application ซึ่งเป็นแอปพลิเคชัน หรือการประมวลผลที่เกิดขึ้นจากฟังก์ชันที่ถูกระบุหมายเลขในเฟรมข้อมูลนั้นๆ ซึ่งแอปพลิเคชันจะอ่านเขียนข้อมูลกับกลุ่มข้อมูลอย่างไรนั้น ขึ้นอยู่กับโครงสร้างการทำงานของฮาร์ดแวร์แต่ละประเภท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.3.5 การเข้าถึงและอ่านค่าข้อมูลชนิดต่างๆ

2.3.2 หมายเลขคำสั่งหรือ Function Code

การประมวลผลหรือทำงานในฟังก์ชันต่างๆเกิดขึ้นได้โดยที่ระบบจะไปทำอ่านคำสั่งตามตำแหน่งที่ระบุใน หมายเลขฟังก์ชัน หลักการก็คือ ดูว่าฟังก์ชันที่ต้องการให้ประมวลผลมีหมายเลขใด จากนั้นก็จะไปค้นดูในตาราง อ้างอิงว่าฟังก์ชันหมายเลขที่ระบุนี้ จะสั่งให้ระบบทำงานอะไร เช่น ฟังก์ชันหมายเลข 02 สั่งให้ระบบอ่านข้อมูลจาก หน่วยความจำข้อมูลอินพุตแบบบิต

รูปที่ 2.3.1 แสดงการจัดแบ่งช่วงค่าหมายเลขที่ใช้อ้างอิงฟังก์ชัน โดยแบ่งออกเป็น 2 ประเภท ได้แก่ ฟังก์ชันแบบ Public function code ซึ่งเป็นฟังก์ชันมาตรฐานของโพรโตคอล MODBUS ที่อุปกรณ์ทุกตัวที่สื่อสารด้วย MODBUS ทุกตัวจะต้องรู้จักและเรียกใช้งานได้ ส่วนฟังก์ชันที่ผู้ผลิตอุปกรณ์สามารถออกแบบเองได้ เรียกว่า user-defined function code หมายเลขเหล่านี้ ทั้ง Public และ user-defined จะต้องไม่ซ้ำกันเลย และมีค่าอยู่ในช่วงค่าที่กำหนดให้เท่านั้น เช่น ผู้ผลิตจะต้องเลือกค่าหมายเลขภายในช่วง 65 ถึง 71 และ 100 ถึง 109 สำหรับใช้ระบุตำแหน่ง อ้างอิงของฟังก์ชันแบบ user-defined ที่พัฒนาขึ้นมาใช้งานกับผลิตภัณฑ์ของตน ส่วนฟังก์ชันแบบ Public นั้น ได้มีการระบุค่าตำแหน่งเอาไว้ตายตัวแน่นอนแล้ว ผู้ผลิตควรยึดตามค่าที่กำหนดไว้เหมือนกัน ไม่ควรเปลี่ยนแปลง เพราะอาจเกิดปัญหาเมื่อต่อใช้งานร่วมกับผลิตภัณฑ์ของผู้ผลิตรายอื่น ทำให้ไม่สามารถทำงานร่วมกันได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

127	PUBLIC function codes
110	User Defined Function codes
100	PUBLIC function codes
72	User Defined Function codes
65	PUBLIC function codes

รูปที่ 2.3.6 ช่วงค่าแอดเดรสของฟังก์ชันทั้งแบบ Public และ User Defined

ฟังก์ชันแบบ Public สำหรับโปรโตคอล MODBUS แบ่งออกเป็นกลุ่มประเภทต่างๆ ดังแสดงในรูปที่ 8 สังเกตเห็นว่า มี 2 กลุ่มใหญ่ คือ กลุ่มฟังก์ชันการเข้าถึงข้อมูล (Data Access) และกลุ่มฟังก์ชันตรวจสอบระบบ (Diagnostics) และฟังก์ชันอีก 1 ฟังก์ชัน ที่ใช้ในการอินเตอร์เฟสระบบ

กลุ่มฟังก์ชันการเข้าถึงข้อมูลประกอบด้วยฟังก์ชันอ่านเขียนข้อมูลบนกลุ่มข้อมูลทั้ง 4 ชนิด รวมทั้งฟังก์ชันอ่านเขียนไฟล์ข้อมูลด้วย ส่วนกลุ่มฟังก์ชันตรวจสอบระบบใช้สำหรับตรวจสอบการทำงานของระบบ เช่น ฟังก์ชันหมายเลข 07 คือฟังก์ชัน แปลความหมายรหัสความผิดพลาดจากในการรับส่งข้อมูล ที่ทำหน้าที่ตรวจดูว่าความผิดพลาดที่เกิดขึ้นนั้นคือความผิดพลาดอะไร หรือฟังก์ชันตรวจสอบบันทึกเหตุการณ์ต่างๆ ที่เกิดขึ้น ในกระบวนการสื่อสารภายในช่วงระยะเวลาหนึ่ง ผลลัพธ์ที่ทุกตัวที่ถูกออกแบบให้สามารถสื่อสารด้วยโปรโตคอล MODBUS จะต้องสามารถประมวลผลฟังก์ชันเหล่านี้ได้ทุกฟังก์ชัน ส่วนที่จะมีฟังก์ชันหรือความสามารถพิเศษอื่นๆ เพิ่มเติมอย่างไร นั้น ขึ้นอยู่กับผู้ผลิตแต่ละรายที่ต้องการพัฒนาให้ผลลัพธ์ของตนมีจุดเด่นจุดขายเหนือกว่าคู่แข่งอย่างไร

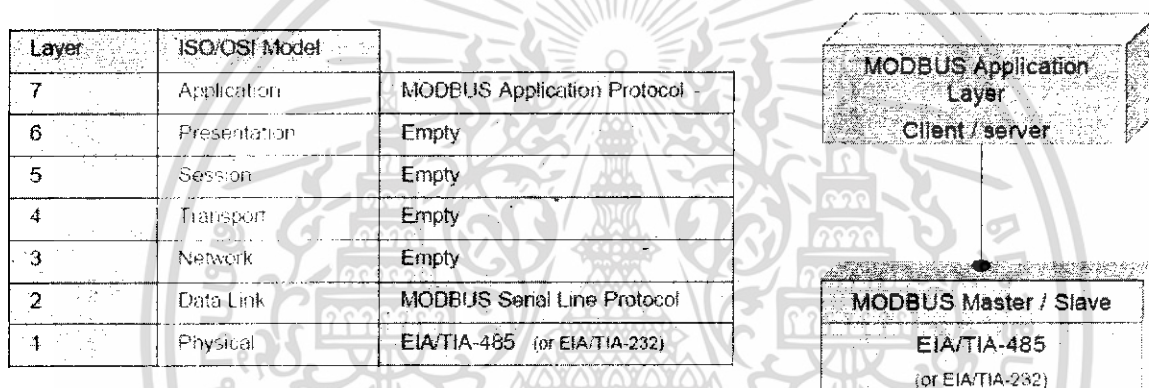
				Function Codes		
				code	Sub Code	(HEX)
Data Access	Bit access	Physical Discrete Inputs	Read Discrete Inputs	02		02
		Internal Bits Or Physical coils	Read Coils	01		01
			Write Single Coil	05		05
				Write Multiple Coils	15	
	16 bits access	Physical Input Registers	Read Input Register	04		04
			Read Holding Registers	03		03
		Internal Registers Or Physical Output Registers	Write Single Register	06		06
			Write Multiple Registers	16		10
			Read/Write Multiple Registers	23		17
			Mask Write Register	22		16
			Read FIFO queue	24		18
	File record access		Read File record	20	6	14
			Write File record	21	6	15
	Diagnostics		Read Exception status	07		07
			Diagnostic	08	00-18	
			Get Com event counter	11		0B
			Get Com Event Log	12		0C
			Report Slave ID	17		11
			Read device Identification	43	14	2B
	Other		Encapsulated Interface Transport	43		2B

รูปที่ 2.3.7 ฟังก์ชัน Public ชนิดต่างๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.3 โพรโทคอล MODBUS Serial Line

MODBUS ในช่วงแรก ถูกพัฒนาขึ้นมาใช้งานกับระบบสื่อสารข้อมูลแบบอนุกรม ได้แก่ EIA/TIA-232 และ EIA/TIA-485 หรือที่เรียกกันว่า RS-232 และ RS-485 ตามลำดับ สาเหตุเป็นเพราะว่า ในสมัยก่อน ระบบสื่อสารข้อมูลแบบอนุกรมเป็นที่นิยมใช้ในงานอุตสาหกรรม แม้กระทั่งในปัจจุบัน RS-485 ก็ยังเป็นที่ยอมรับกันอยู่ เพราะต้นทุนต่ำ ใช้สายสัญญาณเพียงแค่ 2 เส้น อีกทั้งยังสามารถรับส่งข้อมูลฯ ได้ระยะทางไกลถึง 1.2 กิโลเมตร ดังนั้น โครงสร้างของการสื่อสารด้วยโพรโทคอล MODBUS จึงต้องมีตัวกลางที่ทำหน้าที่เชื่อมประสานระหว่างลำดับชั้นบนคือแอปพลิเคชันโพรโทคอล กับลำดับชั้นล่าง คือฮาร์ดแวร์ที่ทำหน้าที่ดำเนินการทางกายภาพในการรับส่งข้อมูล ก็คือวงจรสื่อสารข้อมูลแบบอนุกรม ตัวกลางที่วันนี้ เรียกว่า โพรโทคอล MODBUS Serial



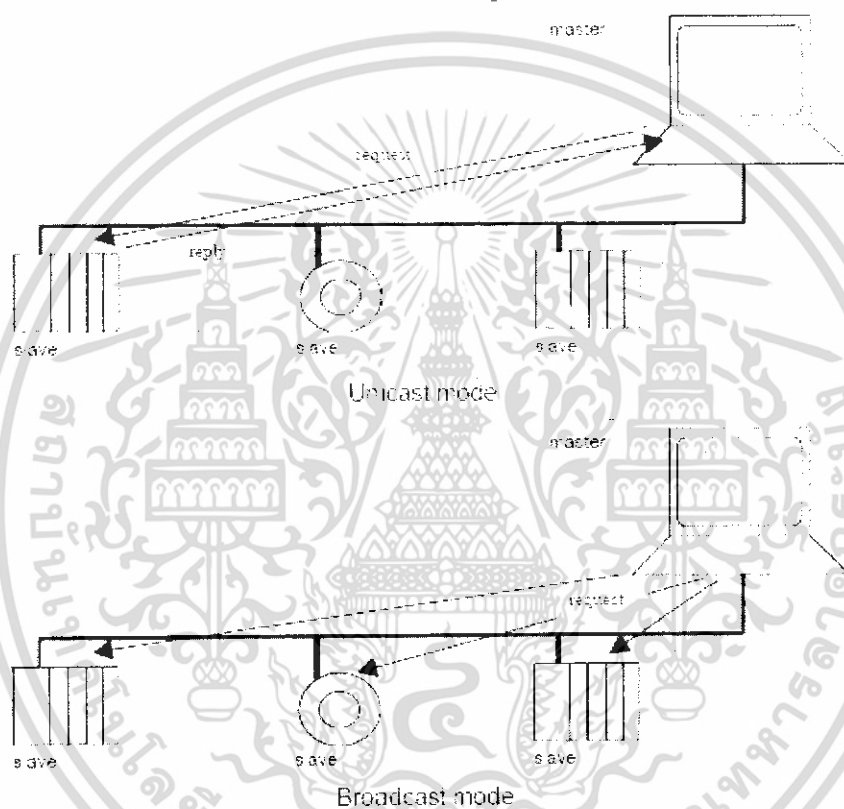
รูปที่ 2.3.8 ลำดับชั้นของ MODBUS Serial Line

Line ใช้เชื่อมประสานกับระบบสื่อสารข้อมูลแบบอนุกรม จัดอยู่ในลำดับชั้นที่ 2 ของแบบจำลอง OSI คือ Data Link Layer ทำหน้าที่แปลงข้อมูลจากให้อยู่ในสภาพพร้อมส่ง และแปลงกลับข้อมูลที่รับเข้ามาเพื่อส่งไปให้ลำดับชั้นบน ซึ่งก็คือการแปลงเฟรมข้อมูลระหว่าง PDU กับ ADU นั่นเอง โพรโทคอล MODBUS Serial Line มีรูปแบบการทำงานเป็นแบบมาสเตอร์-สเลฟ โดยที่ระบบบัสข้อมูลหนึ่งๆ จะมี Master เพียง 1 ตัว ทำหน้าที่ควบคุมจัดการการสื่อสารของสมาชิกทุกตัวภายในระบบบัสข้อมูล ที่เรียกว่าสเลฟ สามารถมีจำนวนสเลฟได้มากที่สุดไม่เกิน 247 ตัว ระบบสื่อสารแบบนี้จะเริ่มต้นด้วยมาสเตอร์เสมอ สเลฟจะไม่สามารถเริ่มต้นการสื่อสารได้ และจะส่งข้อมูลออกมาที่บัสข้อมูลก็ต่อเมื่อได้รับการติดต่อร้องขอจากมาสเตอร์เท่านั้น และสเลฟแต่ละตัวก็ไม่สามารถสื่อสารกันเองได้ ในขณะเวลาหนึ่ง มาสเตอร์จะสามารถควบคุมการสื่อสารได้เพียง 1 กระบวนการเท่านั้น ไม่สามารถควบคุมการสื่อสารพร้อมๆ กันหลายกระบวนการได้ ดังนั้น การสื่อสารภายในบัสข้อมูลจึงเป็นแบบครั้งต่อครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มาสเตอร์สามารถติดต่อกับสเลฟได้ด้วยโหมดการติดต่อ 2 โหมด คือ

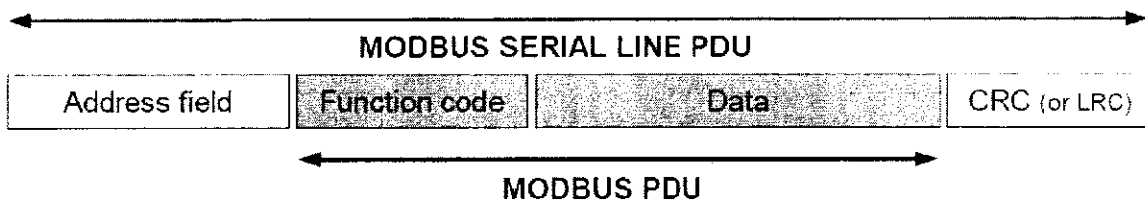
1. โหมด unicast มาสเตอร์เริ่มต้นด้วยการประกาศหมายเลขของสเลฟตัวที่ต้องการติดต่อดำลงไปในบัส จากนั้น สเลฟตัวที่ถูกระบุถึง จะส่งเฟรมข้อมูล reply เพื่อตอบกลับไปยังมาสเตอร์ ดังนั้น ในโหมดนี้ การสื่อสาร 1 ครั้ง จึงประกอบด้วยเฟรมข้อมูล 2 เฟรม คือเฟรมแรกที่มาสเตอร์ส่งออกมา และเฟรมข้อมูลที่สเลฟตอบกลับ ไปยังมาสเตอร์



รูปที่ 2.3.9 การติดต่อสเลฟแบบ unicast และ broadcast

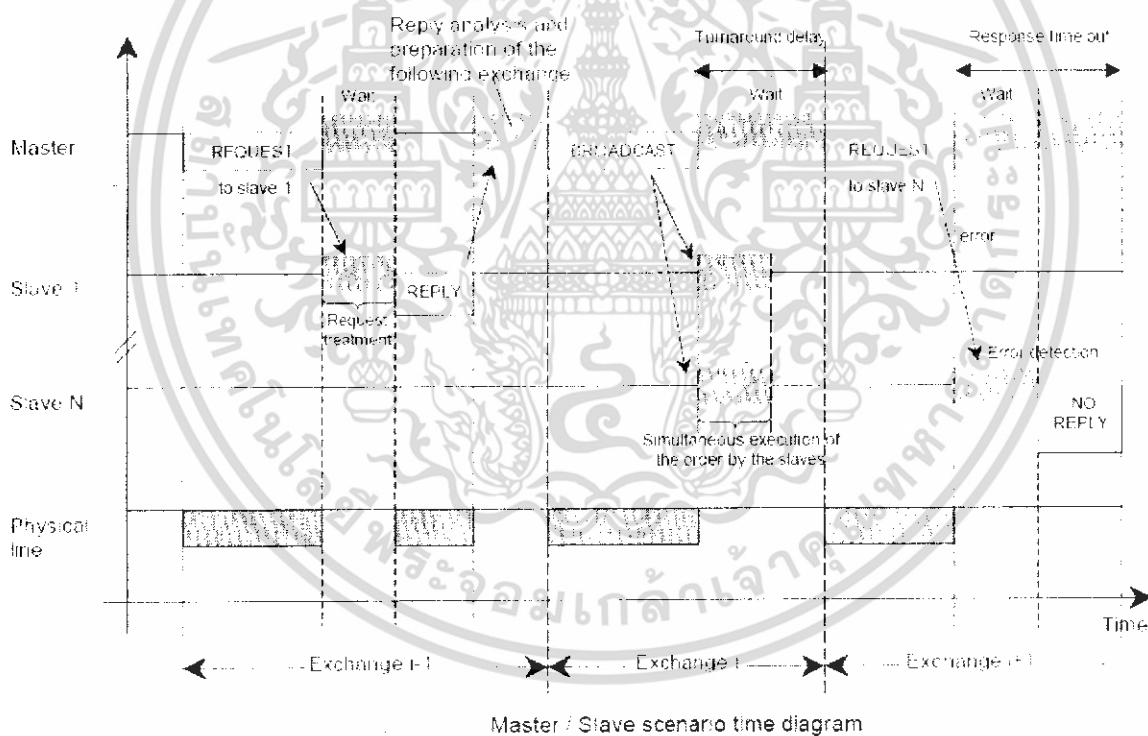
2. โหมด broadcast เป็นการติดต่อจากมาสเตอร์ไปที่สเลฟทุกตัวในบัส ก็คือการติดต่อแบบกระจายสัญญาณนั่นเอง สเลฟทุกตัวจะรับข้อมูลจากมาสเตอร์ แต่ไม่มีการตอบกลับ ดังนั้น โหมดนี้จึงถูกใช้ในการเขียนข้อมูลบางประเภทลงไปในสเลฟทุกตัว เช่น ค่าพารามิเตอร์เริ่มต้นในการทำงานของสเลฟ หรือ คำสั่งกำหนดค่าเริ่มต้นเมื่อระบบเริ่มทำงาน (Initialization Command) หมายเลขแอดเดรสที่ใช้ในเฟรมข้อมูลในโหมด broadcast คือ หมายเลข 0 ส่วนหมายเลขแอดเดรสตั้งแต่ 248 จนถึง 255 ถูกสงวนเอาไว้ไม่ให้ใช้งาน คาดว่าคงเพื่อเอาไว้สำหรับการปรับปรุงพัฒนาโปรโตคอลในอนาคต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.3.10 เฟรมข้อมูลของ MODBUS Serial Line

เฟรมข้อมูลของโปรโตคอล MODBUS Serial Line มีองค์ประกอบดังแสดงในรูปที่ 2.3.10 จะสังเกตเห็นได้ว่า MODBUS PDU ก็คือ PDU ที่กล่าวถึงในช่วงแรก ไม่ว่าจะนำไปใช้กับระบบสื่อสารแบบอนุกรม แบบ TCP/IP หรือแบบอื่นๆ ก็ไม่แตกต่างกัน ส่วนที่แตกต่างก็คือ ADU ซึ่งในที่นี้ สำหรับ MODBUS Serial Line เรียกว่า MODBUS Serial Line PDU เพิ่มเติมส่วนการระบุแอดเดรสของสเลฟ และรหัสตรวจสอบความถูกต้องของข้อมูลแบบ CRC หรือ LRC



รูปที่ 2.3.11 แผนผังเวลาแสดงตัวอย่างการรับส่งเฟรมข้อมูลระหว่างมาสเตอร์กับสเลฟ

รูปที่ 2.3.11 แสดงแผนผังเวลาในการเปลี่ยนแปลงระดับสัญญาณบนบัสข้อมูล เริ่มต้นด้วยมาสเตอร์ส่งเฟรมข้อมูลออกมาที่บัสข้อมูลเพื่อติดต่อกับสเลฟหมายเลข 1 ด้วยโหมด unicast จากนั้น มาสเตอร์จะรอให้สเลฟแต่ละตัว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีกรนำไปใช้

ทำการเปรียบเทียบแอดเดรสของตัวเองว่าตรงกับแอดเดรสที่ระบุในเฟรมข้อมูล คือ 1 หรือไม่ ซึ่งในที่สุด สเตลหมายเลข 1 จะรับรู้ว่ามีแอดเดรสส่งเฟรมข้อมูลมาให้ตัวเอง และรับเอาข้อมูลภายในเฟรมนั้นเข้ามาประมวลผลตามหมายเลขฟังก์ชันและข้อมูลประกอบภายในเฟรมข้อมูล เมื่อเสร็จเรียบร้อยแล้ว จึงส่งเฟรมข้อมูลตอบกลับลงมาในบัสข้อมูลเพื่อให้แอดเดรสอ่านไปอีกที จากนั้น แอดเดรสส่งเฟรมข้อมูลออกมาให้กับสเตลทุกตัว ด้วยโหมด broadcast ทำให้สเตลทุกตัวรับเฟรมข้อมูลนั้นเอาไว้ โดยไม่มีการตอบกลับแต่อย่างใด ปิดท้ายด้วยการส่งเฟรมข้อมูลจากแอดเดรสไปที่สเตลหมายเลข N แต่คราวนี้ เกิดปัญหาในขณะที่สเตลหมายเลข N กำลังส่งเฟรมตอบกลับมายังแอดเดรส ทำให้แอดเดรสรอรับข้อมูลตอบกลับจนหมดเวลา ก็จะรู้ว่าเกิดความผิดพลาดในการสื่อสาร

2.3.4 โหมดการรับส่งข้อมูลอนุกรม

การรับส่งข้อมูลอนุกรมด้วยโปรโตคอล MODBUS สามารถเลือกได้ 2 โหมด คือ โหมด RTU และโหมด ASCII ซึ่งทั้ง 2 โหมดนี้ มีความแตกต่างกันที่การกำหนดรูปแบบของชุดข้อมูลภายในเฟรม จะเลือกโหมดใดก็ได้ แต่มีเงื่อนไขว่า อุปกรณ์ทุกตัวที่ต่อรวมอยู่ในบัสหรือเครือข่ายเดียวกัน จะต้องถูกปรับตั้งให้เลือกใช้โหมดเดียวกันทั้งหมด

1. โหมด RTU

Slave Address	Function Code	Data	CRC
1 byte	1 byte	0 up to 252 byte(s)	2 bytes CRC Low, CRC Hi

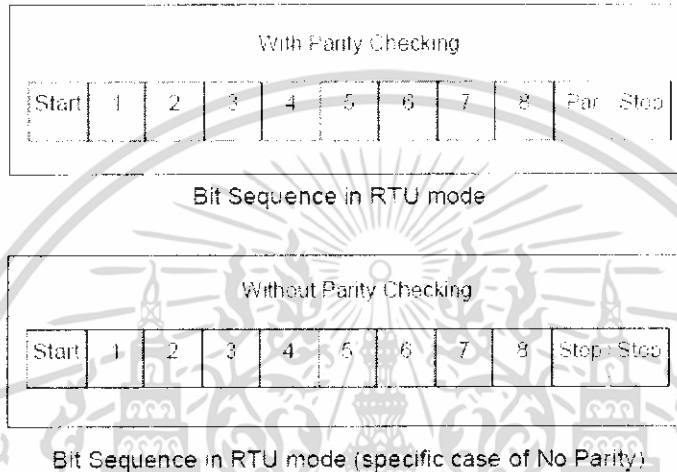
RTU Message Frame

รูปที่ 2.3.12 เฟรมข้อมูลในโหมด RTU

เฟรมข้อมูลในโหมด RTU ประกอบด้วยข้อมูลแสดงตำแหน่งแอดเดรสของสเตล 1 ไบต์, หมายเลขฟังก์ชัน 1 ไบต์, ข้อมูลที่ทำการรับส่งจำนวนมากสุดไม่เกิน 252 ไบต์, และรหัสตรวจสอบความถูกต้องของข้อมูลแบบ CRC (Cyclical Redundancy Checking) ขนาด 2 ไบต์ ค่า CRC นี้เป็นค่าที่คำนวณมาจากข้อมูลทุกไบต์ ไม่รวม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บิต start , stop , และ parity check สำหรับสูตรที่ใช้ในการคำนวณนั้น ผู้เขียนจะขอไม่กล่าวถึง เพียงแต่ให้ทราบว่า มีประโยชน์อย่างไร โดยที่สเลฟตัวที่ส่งข้อมูลออกมาจะสร้างรหัส CRC แล้วส่งตามท้ายไบต์ข้อมูลออกมา หลังจากนั้น เมื่อมาสเตอร์ได้รับเฟรมข้อมูล และถอดข้อมูลออกจากเฟรมแล้ว จะทำการคำนวณค่า CRC ตามสูตรเดียวกันกับสเลฟ เพื่อทำการเปรียบเทียบค่า CRC ทั้ง 2 ค่า ว่าตรงกันหรือไม่ หากไม่ตรงกัน แสดงว่า เกิดความผิดพลาดในการรับส่งข้อมูล

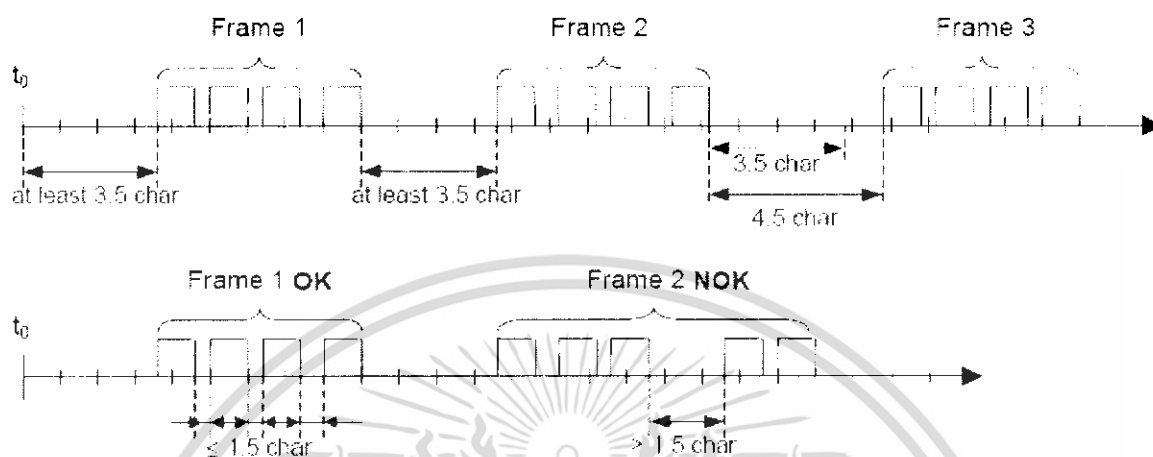


รูปที่ 2.3.13 การส่งข้อมูล 1 ไบต์ในโหมด RTU

ในโหมด RTU การส่งข้อมูล 1 ไบต์ ไม่ว่าจะเป็นส่วนใดภายในเฟรม จะต้องทำการส่งบิตข้อมูลรวม 11 บิต คือ บิตเริ่มต้น (Start) 1 บิต , บิตข้อมูล 8 บิตซึ่งก็คือ 1 ไบต์ , บิตตรวจสอบ parity ของข้อมูล 1 บิต , และบิตหยุด (Stop) 1 บิต หรือหากเลือกแบบไม่มีบิต parity ก็จะเป็นบิต stop แทน รวม 2 บิต สำหรับการกำหนดให้มีบิต parity นั้น สามารถเลือกเป็นแบบคู่ (even parity) หรือคี่ (odd parity) ก็ได้ และหากต้องการออกแบบให้สอดคล้องกับอุปกรณ์ที่มีใช้กันทั่วไปมากที่สุด ควรเลือกแบบคู่ โดยที่สามารถปรับเปลี่ยนเป็นแบบคี่ หรือไม่มีการตรวจสอบ parity (no parity) ได้ด้วย

รูปที่ 2.3.13 แสดงช่วงเวลาที่เหมาะสมในกระบวนการส่งเฟรมข้อมูลออกมาในบัสข้อมูล เมื่อส่งเฟรมข้อมูลออกไป 1 เฟรมแล้ว จะต้องรอเวลาอย่างน้อยเท่ากับเวลาที่ใช้ส่งข้อมูลจำนวน 3.5 ตัวอักษร จึงจะสามารถส่งเฟรมข้อมูลต่อไปได้ และภายในเฟรมแต่ละเฟรม ซึ่งประกอบด้วยชุดบิตข้อมูลจำนวนหลายชุด ก็จะอยู่ห่างกันไม่เกิน 1.5 ตัวอักษร วัตถุประสงค์ในการกำหนดช่วงเวลาระหว่างเฟรมข้อมูล และชุดบิตข้อมูลภายในเฟรม ก็เพื่อให้อุปกรณ์ต่างๆ ไม่ว่าจะมาเป็นมาสเตอร์หรือสเลฟ สามารถรับรู้ถึงจุดเริ่มต้นและจุดสิ้นสุดของเฟรมข้อมูลแต่ละเฟรมได้

และสามารถตรวจสอบได้ว่า การรับส่งข้อมูลในขณะนั้น เกิดความผิดพลาดขึ้นมาหรือไม่ โดยตรวจสอบกับช่วงระยะห่างของเวลาที่การจะเป็น กับค่าที่วัดได้จริง



รูปที่ 2.3.14 ระยะห่างทางเวลาระหว่างเฟรมข้อมูลแต่ละเฟรม และระหว่างชุดข้อมูลแต่ละชุดภายใน เฟรมเดียวกัน

2. โหมด ASCII

การรับส่งข้อมูลในโหมด ASCII มีความแตกต่างจากโหมด RTU ตรงที่ ในโหมด RTU ข้อมูลที่จะส่งขนาด 1 ไบต์ นำมารวมกับบิตประกอบต่างๆ ก็สามารถส่งออกไปได้เลย แต่สำหรับโหมด ASCII จะมองข้อมูล 1 ไบต์นั้น ออกเป็นตัวอักษร 2 ตัว เช่น ค่า 0x5B ซึ่งเป็นเลขฐานสิบหก ก็จะถูกมองเป็นตัวอักษร '5' และตัวอักษร 'B' จากนั้นก็จะทำการค้นหารหัส ASCII ของตัวอักษรทั้ง 2 ตัวนั้น ซึ่งได้แก่ 0x35 สำหรับ '5' และ 0x42 สำหรับ 'B' แล้วทำการส่งรหัส ASCII ทั้ง 2 ค่านี้ออกไป ซึ่งจะได้ผลเท่ากับการส่งค่า 0x5B ซึ่งเป็นข้อมูลขนาด 1 ไบต์ ในโหมด RTU

จะเห็นได้ว่าการส่งข้อมูลในโหมด ASCII จะต้องทำงานมากกว่าการส่งข้อมูลในโหมด RTU ซึ่งทำให้อัตราเร็วในการสื่อสารมีค่าต่ำกว่า โหมด ASCII สาเหตุที่เป็นแบบนี้ก็เพราะว่า โหมด ASCII ได้ถูกออกแบบมาสำหรับอุปกรณ์ที่ไม่มีความสามารถในการกำหนดช่วงระยะห่างทางเวลาในการส่งเฟรมข้อมูล อย่างเช่นในโหมด RTU ที่อุปกรณ์

Start	Address	Function	Data	LRC	End
1 char	2 chars	2 chars	0 up to 2x252 charis)	2 chars	2 chars CR,LF

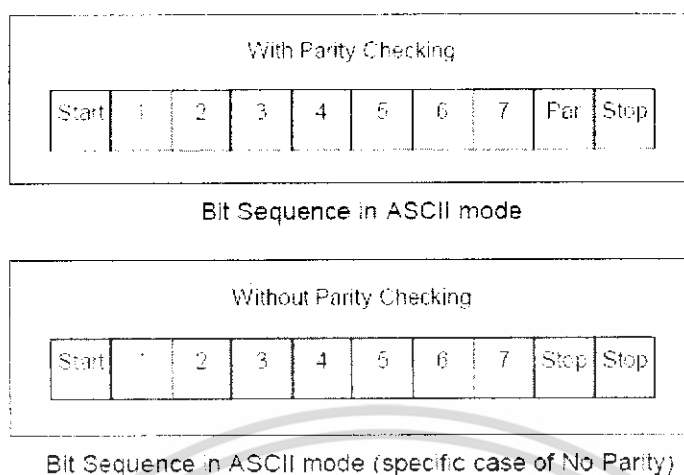
ASCII Message Frame

รูปที่ 2.3.15 เฟรมข้อมูลในโหมด ASCII

สามารถกำหนดได้ว่าจะส่งเฟรมข้อมูลแต่ละเฟรมออกมาด้วยเวลาห่างกันเท่าใด และอุปกรณ์ที่รองรับข้อมูลก็
ต้องสามารถตรวจจับแยกแยะได้ด้วยว่าเฟรมข้อมูลแต่ละเฟรมที่รับเข้ามานั้น มีระยะเวลาห่างกันภายในช่วงเวลาที่
กำหนดหรือไม่ เพื่อให้สามารถตรวจสอบหาจุดเริ่มต้นและจุดสิ้นสุดของเฟรมข้อมูลแต่ละเฟรมได้ แต่ในความเป็นจริง
ยังมีอุปกรณ์อีกหลายชนิด ที่ไม่มีความสามารถพิเศษแบบนี้ จึงต้องใช้วิธีอื่นที่จะช่วยให้สามารถรับรู้
จุดเริ่มต้นและจุดสิ้นสุดของเฟรมข้อมูลได้ ได้แก่ โหมด ASCII ซึ่งในโหมดนี้ จะเริ่มต้นเฟรมข้อมูลด้วยการส่งรหัส
ASCII ที่กำหนดให้หมายถึงจุดเริ่มต้น คือ 0x3A ซึ่งตรงกับตัวอักษร ‘:’ ตามด้วยแอดเดรสของสากล หมายเลข
ฟังก์ชัน, ข้อมูล, รหัสตรวจสอบ LRC และรหัส ASCII 2 ตัว ที่กำหนดให้หมายถึงจุดสิ้นสุด คือ รหัส 0x0D และ
0x0A คือรหัส CR (Carriage Return) และ LF (Line Feed) ตามลำดับ (ดูรูปที่ 2.3.5) โดยขณะที่ใส่ข้อมูลว่างจาก
การรับส่งข้อมูล อุปกรณ์ทุกตัวจะคอยตรวจจับข้อมูลในบัสว่ามีการส่งรหัส ASCII ของ ‘:’ ออกมาหรือไม่ หรือมี ก็
จะรับรู้ว่าจะขณะนี้ ได้มีการเริ่มต้นส่งเฟรมข้อมูลออกมาแล้ว ก็จะเข้าสู่กระบวนการรับข้อมูลต่อไป

รูปที่ 2.3.16 แสดงชุดของบิตข้อมูลที่ส่งทั้งหมดในการส่งข้อมูลแต่ละตัวอักษร จะเห็นได้ว่าหน่วยของชุด
ข้อมูลในโหมด ASCII คือตัวอักษร ไม่เหมือนในโหมด RTU ที่มีหน่วยเป็นไบต์ เพราะโหมด ASCII เป็นการส่ง
ข้อมูลในรูปแบบของรหัส ASCII ของตัวอักษร ซึ่งสามารถกำหนดได้ด้วยบิตข้อมูลจำนวน 7 บิต ไม่ต้องใช้ถึง 8 บิต
ดังนั้น บิตที่ต้องส่งต่อการส่งรหัส ASCII 1 ตัว ได้แก่ บิต Start 1 บิต, บิตข้อมูลรหัส ASCII 7 บิต, บิตตรวจสอบ
parity 1 บิต, และบิต Stop 1 บิต รวมทั้งหมดเท่ากับ 10 บิต และเช่นเดียวกับโหมด RTU ก็ สามารถเลือกประเภท
ของบิตตรวจสอบ parity ได้ ว่าเป็นแบบคู่, คี่, หรือไม่มีบิตตรวจสอบ ซึ่งจะเปลี่ยนเป็นบิต Stop แทน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



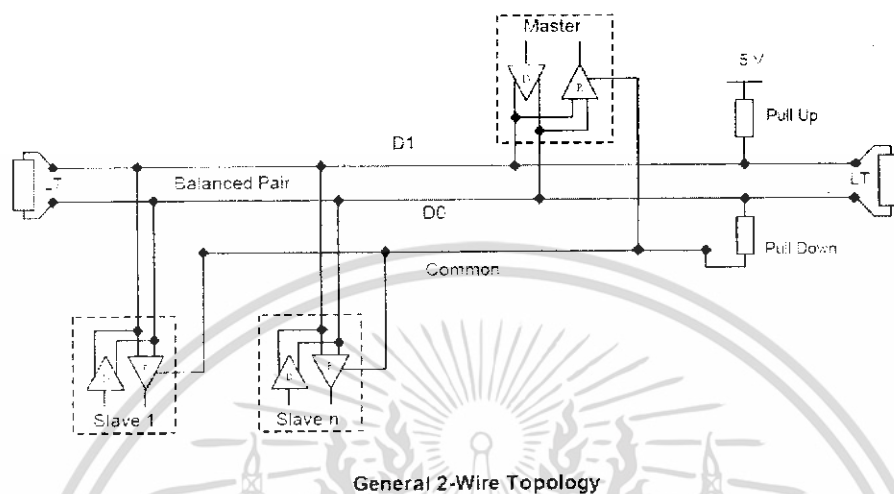
รูปที่ 2.3.16 การส่งข้อมูล 1 ตัวอักษรในโหมด ASCII

เนื่องจากการส่งข้อมูลในโหมด ASCII เป็นการแปลงข้อมูลจากเลขฐานสิบหกเป็นรหัส ASCII ของตัวอักษรที่แสดงค่าเลขฐานสิบหก ดังนั้น รหัส ASCII ที่ปรากฏในบัสข้อมูล นอกเหนือไปจากรหัสเริ่มต้นและรหัสสิ้นสุดแล้ว จะเป็นรหัส ASCII ของ ตัวอักษรตั้งแต่ '0' ถึง '9' และ 'A' ถึง 'F' เท่านั้น ถึงแม้โหมด ASCII จะไม่ต้องกำหนดช่วงระยะห่างทางเวลาของเฟรมข้อมูลแต่ละเฟรม แต่อุปกรณ์ยังคงสามารถตรวจจับช่วงระยะห่างทางเวลาระหว่างการส่งข้อมูลรหัส ASCII แต่ละตัวได้ ซึ่งหากเว้นช่วงห่างกันนานเกินไป แสดงว่าเกิดความผิดพลาดในการสื่อสาร โดยปกติ จะกำหนดค่าเวลานี้ไว้ที่ 1 วินาที เรียกว่าเวลานี้ว่า time-out period หากเปรียบเทียบระหว่างเฟรมข้อมูลในโหมด ASCII กับโหมด RTU จะพบว่า การส่งข้อมูลในโหมด ASCII นั้น หากต้องการส่งไบต์ข้อมูลให้ได้เท่ากับโหมด RTU จะต้องส่งข้อมูลรหัส ASCII ออกไปเป็นจำนวน 2 เท่าของจำนวนไบต์ข้อมูล เช่นในโหมด RTU เฟรมข้อมูล 1 เฟรม สามารถส่งข้อมูลได้มากที่สุด 252 ไบต์ ซึ่งหากเป็นโหมด ASCII จะต้องส่งข้อมูลตัวอักษรออกไปทั้งหมด 2x252 เท่ากับ 504 ตัวอักษร และเพื่อให้มาตรฐานขนาดของเฟรมข้อมูลของทั้ง 2 โหมดมีขนาดเท่ากัน จึงกำหนดให้ค่า 504 เป็นค่าจำนวนตัวอักษรมากที่สุดในการส่งเฟรมข้อมูลด้วยโหมด ASCII

2.3.5 รูปแบบการเชื่อมต่อบัสข้อมูล

การเชื่อมต่อหรือเดินสายไฟฟ้าสำหรับใช้ทำหน้าที่เป็นบัสรับส่งสัญญาณข้อมูลภายในเครือข่าย สามารถกระทำได้ 2 วิธี คือ การเดินสายสัญญาณแบบ 2 เส้น และแบบ 4 เส้น โดยใช้มาตรฐาน EIA/TIA-485 เป็นตัวรับส่งข้อมูลในลำดับชั้นกายภาพ (Physical Layer) หรือระดับฮาร์ดแวร์ ซึ่งมีข้อดีคือ สามารถใช้รับส่งข้อมูลได้ระยะทางไกลถึง 1.2 กิโลเมตร โดยอาศัยหลักการขั้วกระแสไฟฟ้าระหว่างคู่สายสัญญาณที่มีแรงดันแตกต่างกันตามค่า

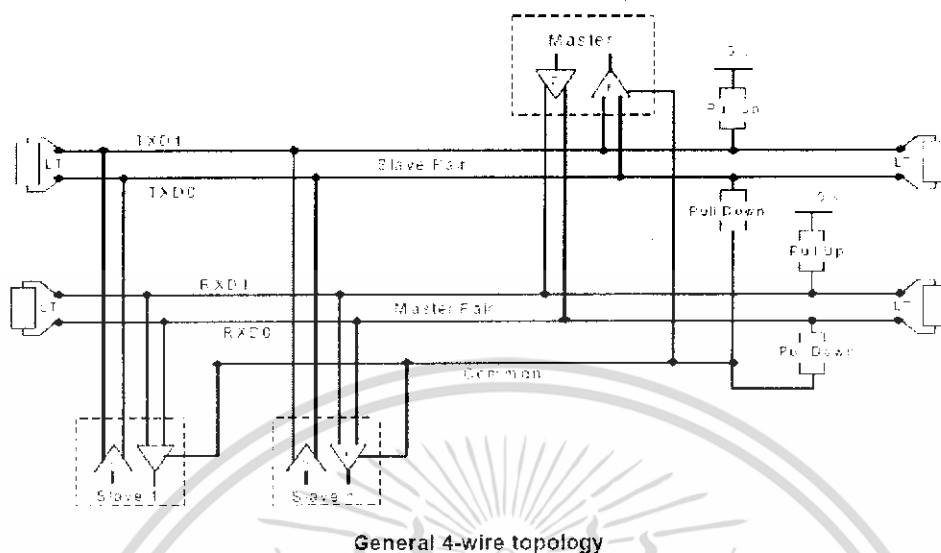
สถานะของข้อมูลที่ต้องการส่ง การเดินสายสัญญาณแบบ 2 เส้น "ได้รับความนิยมมากกว่า" เนื่องจาก ใช้สายสัญญาณที่มีจำนวนเส้นน้อย



รูปที่ 2.3.17 IEA/TIA-485

ตัวนำไฟฟ้าภายในน้อยกว่า ทำให้ประหยัดปริมาตรภายในรางเดินสายไฟ และการเชื่อมต่อสายสัญญาณเข้ากับอุปกรณ์ก็ทำได้ง่ายกว่า เพราะมีสายสัญญาณที่ต้องเชื่อมต่อเพียงแค่ 2 เส้นเท่านั้น ทำหน้าที่ทั้งส่งและรับข้อมูล รูปที่ 18 แสดงการเดินสายสัญญาณระหว่างมาสเตอร์และสเลฟ โดยมีตัวความต้านทานปิดหัวท้ายของสายสัญญาณ เรียกว่า LT (Line Termination) ทำหน้าที่ป้องกันผลกระทบการคลื่นสัญญาณสะท้อนภายในคู่สายสัญญาณ ส่วนตัวความต้านทานอีก 2 ตัว คือ Pull Up และ Pull Down ทำหน้าที่ช่วยดึงระดับของสถานะของสายสัญญาณให้มีค่าที่ควรจะเป็น ในขณะที่ไคว่าง เพราะมิฉะนั้นแล้ว สายสัญญาณอาจมีสถานะเป็น High Impedance ซึ่งจะแปลความหมายในการสื่อสารไม่ได้ และจะทำให้ระบบไม่ทำงาน หรือทำงานผิดพลาด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



General 4-wire topology

รูปที่ 3.20 IEC 611A-485

จะเห็นได้ว่า นอกจากสายสัญญาณ D0 และ D1 แล้ว การเดินสายสัญญาณ ยังต้องมีการเดินสาย common อีก 1 เส้น เพื่อใช้เป็นจุดอ้างอิงระดับสัญญาณของอุปกรณ์ทุกตัวภายในระบบบัสข้อมูลเดียวกัน ซึ่งการเดินสายสัญญาณแบบ 4 สาย ก็จำเป็นต้องใช้สาย common ด้วยเช่นกัน ความแตกต่างก็คือ การเดินสายสัญญาณแบบ 4 เส้น ใช้สายสัญญาณรับส่งข้อมูล 2 คู่ยกกัน คู่หนึ่งสำหรับส่งข้อมูล อีกคู่สำหรับรับข้อมูล

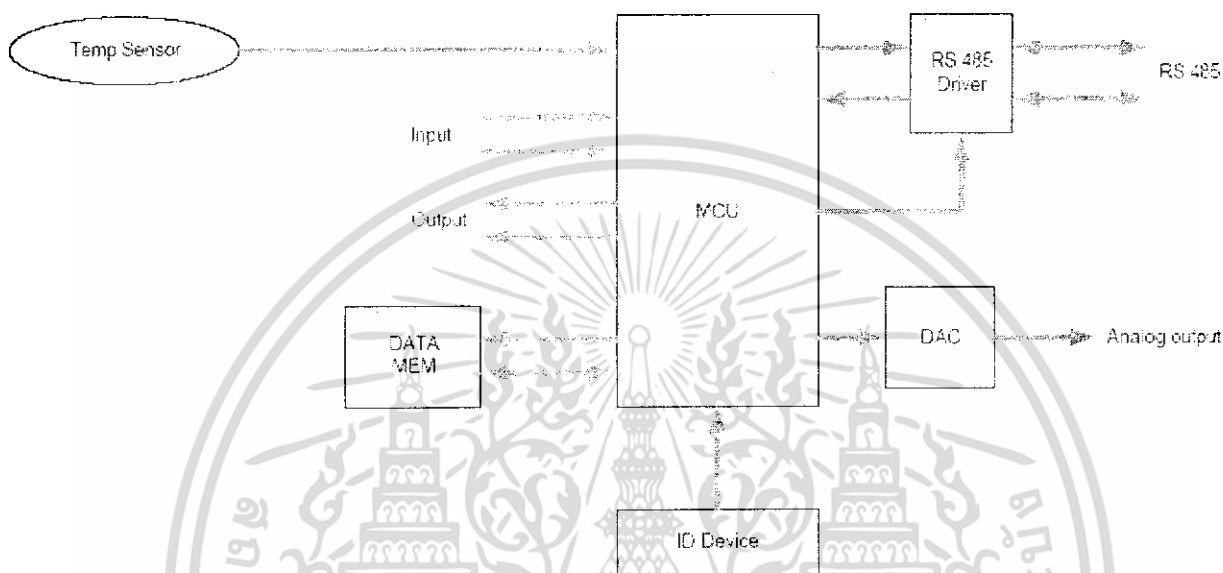
สรุป MODBUS PROTOCOL

จากที่ได้กล่าวมาทั้งหมด คงสามารถมองเห็นภาพการทำงานของกระบวนการรับส่งข้อมูลด้วยโปรโตคอล MODBUS ได้ดีพอสมควร ซึ่งจะเห็นได้ว่า MODBUS เป็นโปรโตคอลที่ไม่ซับซ้อน มีความยืดหยุ่นสูง โดยสามารถนำไปใช้กับระบบสื่อสารข้อมูลได้หลายชนิด ซึ่งก็คือคุณสมบัติของความเป็นระบบเปิด และเปิดโอกาสให้ผู้ผลิตผลิตภัณฑ์ที่สามารถสื่อสารด้วย MODBUS สามารถพัฒนาฟังก์ชันพิเศษต่างๆ ขึ้นมา เพื่อเป็นจุดขายที่แตกต่างไปจากผลิตภัณฑ์ของกลุ่มคู่แข่งได้อีกด้วย และด้วยจุดเด่นต่างเหล่านี้ ทำให้ MODBUS ยังคงเป็นโปรโตคอลยอดนิยมที่ยังไม่ล้าสมัย ดังเช่นโปรโตคอลในภาคอุตสาหกรรมบางตัว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

การออกแบบโครงงาน



รูปที่ 3.1 โครงสร้างของระบบควบคุมอุณหภูมิแบบหลายจุด

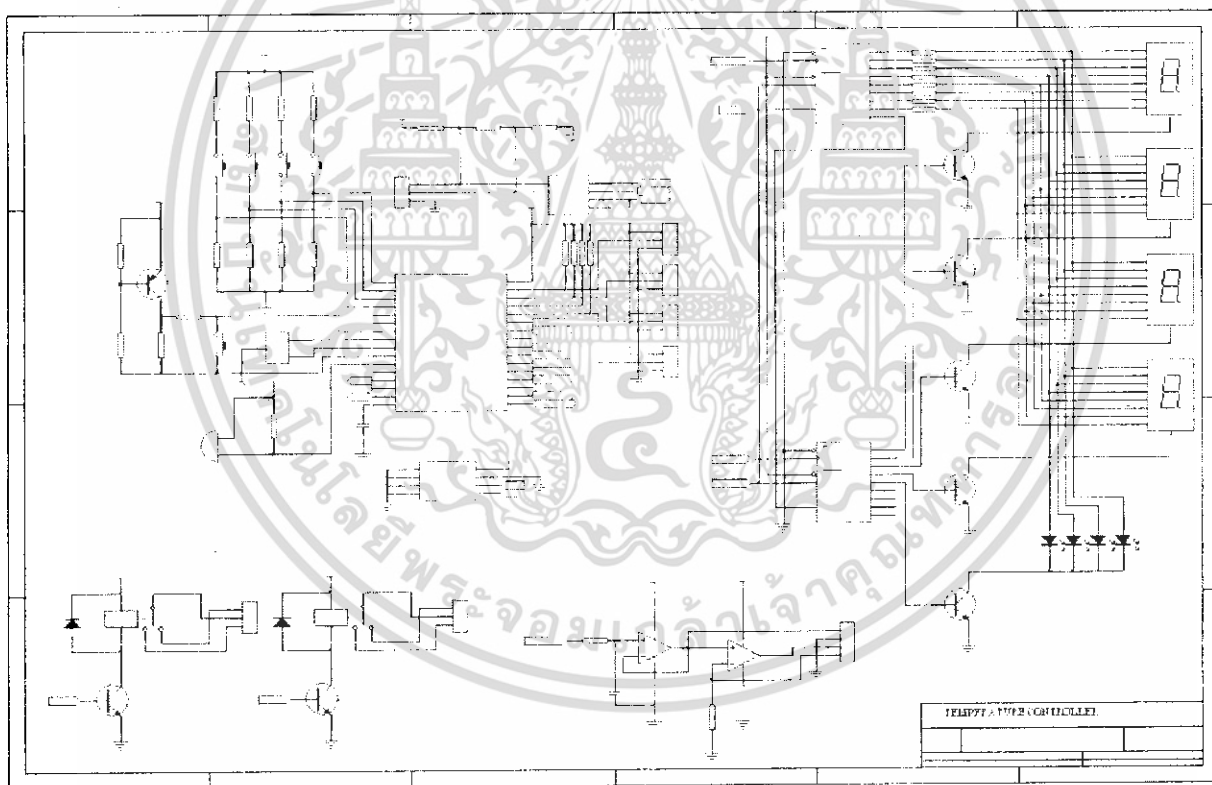
จุดประสงค์หลักของการควบคุม คือ ต้องการให้เอาต์พุตของระบบเข้าสู่ค่าอ้างอิง (setpoint) และไม่ส่งผลต่อค่าพุ่งเกิน (overshoot) และค่าผิดพลาดที่สภาวะคงตัวสูงเกินไป โดยในปริยฐานิพนธ์นี้ต้องการควบคุมอุปกรณ์ตัวที่จ่ายอุณหภูมิ เพื่อให้ได้อุณหภูมิบริเวณที่ต้องการควบคุมอุณหภูมิคงที่ตามค่าอ้างอิง การออกแบบระบบควบคุมอุณหภูมิแบบหลายจุดมีดังนี้

การออกแบบฮาร์ดแวร์ และ ซอฟต์แวร์

3.1 ส่วนควบคุม

ใช้ไมโครคอนโทรลเลอร์ตระกูล F^{MC}-8L เบอร์ MB89N202 ของบริษัท Fujitsu เนื่องจากเป็นไมโครคอนโทรลเลอร์ที่มีประสิทธิภาพสูงและใช้งานแพร่หลายในวงการอุตสาหกรรม โดยวงจรของระบบควบคุมอุณหภูมิแบบหลายจุดแสดงในรูปที่ 3.2 ลายวงจรแสดงในรูปที่ 3.3 ภาพของบอร์ดจริงแสดงในรูป 3.4

โดยระบบควบคุมอุณหภูมิจะใช้ระบบวงปิด (Closed loop control) แบบ PID ซึ่งสามารถกำหนดค่าผ่านระบบคอมพิวเตอร์ โดยมีความละเอียดในการวัด ± 0.5 องศา แสดงผลด้วยเซเว่นเซกเมนต์พร้อมหลอดไฟแสดงผลการทำงาน สามารถทำงานแบบตัวเดียว (stand alone) และสามารถเชื่อมต่อแบบระบบเน็ตเวิร์ก โดยสามารถควบคุมและแสดงผลได้ผ่านคอมพิวเตอร์



รูปที่ 3.2 วงจรของระบบควบคุมอุณหภูมิแบบหลายจุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 ส่วนติดต่อพอร์ตอนุกรม

พอร์ตสื่อสารข้อมูลแบบอนุกรม เป็นพอร์ตภายในของไมโครคอนโทรลเลอร์ ซึ่งติดต่อได้ด้วยซอฟต์แวร์โดยใช้ไอซี 75176 เป็นการสื่อสารผ่านพอร์ต RS485 โดยใช้ MODBUS PROTOCOL โดยในปฏิญญาฉบับนี้จะใช้ MODBUS RTU ในการติดต่อสื่อสารระหว่างบอร์ดควบคุมอุณหภูมิกับเครื่องคอมพิวเตอร์ การทำงานของพอร์ตอนุกรมจะใช้การทำงานโดยการอินเทอร์พท์ผ่านพอร์ตอนุกรม โดยปกติจะบอร์ดคอนโทรลเลอร์จะทำการอ่านค่าอุณหภูมิแสดงผลและควบคุมการจ่ายอุณหภูมิ เมื่อมีการอินเทอร์พท์ผ่านพอร์ตอนุกรม ก็จะกระโดดไปทำงานในส่วนอินเทอร์พท์นี้ และหยุดการทำงานเมื่อจบกระบวนการทำงาน ไปทำงานในโปรแกรมหลักต่อไป

3.3 เซ็นเซอร์

ใช้การต่อแบบ 1 wire bus (DS1820) โดยต่อสายสัญญาณทั้งหมดเข้าที่ขาเพียงขาเดียว จะต้องมีการ पुलล์อัพสายสัญญาณด้วยตัวต้านทาน 4700 โอห์ม การจ่ายไฟ 5 โวลต์ จะเป็นการจ่ายมาจากจุดเดียวกันหรือคนละจุดก็ได้

ในการติดตั้งเซ็นเซอร์ จะต้องมีการติดตั้งให้เหมาะสมกับสภาพแวดล้อมนั้นๆ เพื่อให้อุณหภูมิมีค่าใกล้เคียงกันในแต่ละจุดและทั่วถึง

ส่วนของซอฟต์แวร์ การทำงานในส่วนที่เป็นการอ่านหมายเลขประจำตัว (ROM CODE) ของตัวเซ็นเซอร์โดยมีคำสั่งใช้งานอยู่ 5 รูปแบบ โดยอยู่ในรูปของเลขฐาน 16 โดยมีคำสั่งใช้งานอยู่ 2 แบบคือ 33h (Read Rom) และ 55h (Match Rom)

การทำงาน Read Rom

1. อุปกรณ์มาสเตอร์ ส่งสัญญาณ รีเซต โดยการส่ง 0 เป็นเวลา 480-960 ไมโครวินาที
2. อุปกรณ์สเลฟ ตอบรับ โดยการส่ง 0 เป็นเวลา 60-120 ไมโครวินาที
3. อุปกรณ์มาสเตอร์ ส่งคำสั่ง 33h ออกไป
4. อุปกรณ์สเลฟ จะตอบค่า Rom ออกมาให้เป็นเลข 8 ไบต์ ไบต์แรกเป็น ตระกูล 6 ไบต์ต่อมาเป็นหมายเลขประจำตัว และไบต์สุดท้ายเป็นไบต์ตรวจสอบความถูกต้องตามลำดับ

การทำงานของ Math Rom

1. อุปกรณ์มาสเตอร์ ส่งสัญญาณ รีเซต โดยการส่ง 0 เป็นเวลา 480-960 ไมโครวินาที
2. อุปกรณ์สเลฟ ตอบรับ โดยการส่ง 0 เป็นเวลา 60-120 ไมโครวินาที
3. อุปกรณ์มาสเตอร์ ส่งคำสั่ง 55h ออกไป

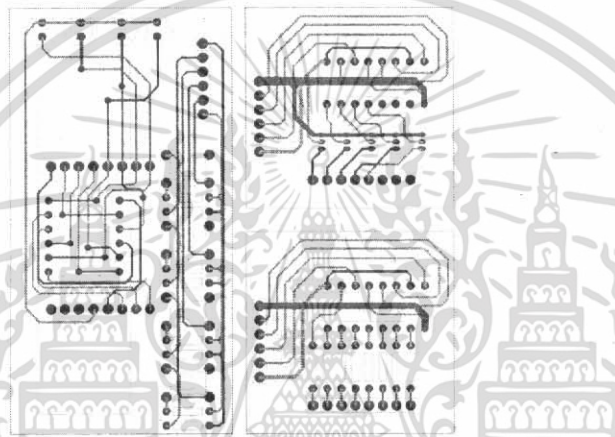
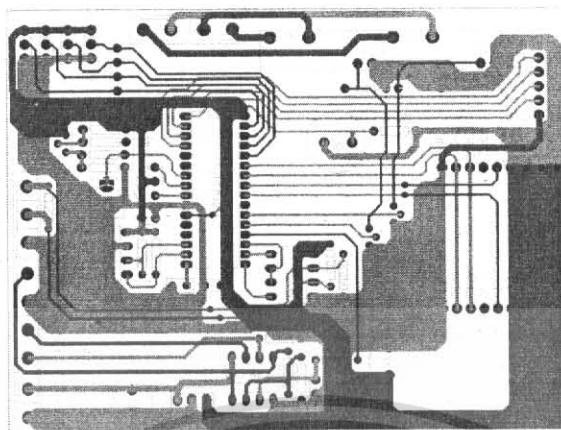
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. อุปกรณ์มาตรฐาน ส่งไป 0 ไป เปรียบเทียบกับค่าที่อยู่ในอุปกรณ์สเลฟ ถ้าถูกก็จะส่งไปต่อต่อไป แต่ถ้าผิดก็จะกลับไปเริ่มกระบวนการใหม่

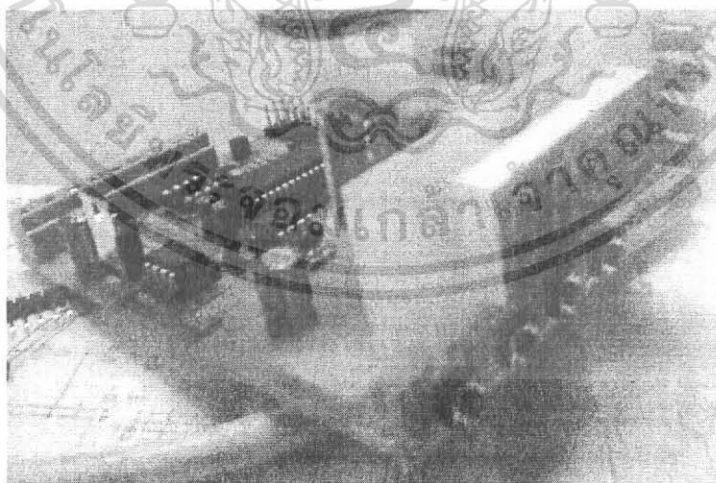
การทำงานของซอฟต์แวร์อีกส่วนหนึ่งคือการทำงานอ่านค่าอุณหภูมิ มีคำสั่งที่ใช้งานคือ 44h คือการแปลงค่าอุณหภูมิมาเป็นข้อมูลดิจิทัล และ BEh คือการอ่านค่าที่ถูกเก็บเอาไว้ในสแควนแพด มีรูปแบบการทำงาน คือ

1. อุปกรณ์มาตรฐาน ส่งสัญญาณ รีเซต
2. อุปกรณ์สเลฟ ตอบรับ
4. อุปกรณ์มาตรฐาน ส่งคำสั่ง 55h ออกไปสั่งให้อุปกรณ์ที่มีหมายเลขตรงกับที่สั่งทำงาน
5. อุปกรณ์มาตรฐาน ส่งคำสั่ง 44h ออกไปแปลงค่าอุณหภูมิเป็นดิจิทัลแล้วเก็บค่าไว้ในสแควนแพด
5. หน่วงเวลารอกการเปลี่ยนแปลงค่าอุณหภูมิเป็น ดิจิตอล อย่างน้อย 200 นาโนวินาที
6. อุปกรณ์มาตรฐาน ส่งสัญญาณ รีเซต
7. อุปกรณ์สเลฟตอบรับ
8. อุปกรณ์มาตรฐาน ส่งคำสั่ง 55h ออกไป
9. อุปกรณ์มาตรฐาน ส่งคำสั่ง BEh ออกไปอ่านค่าในสแควนแพด
10. เลื่อนบิตไปทางขวา 1 ครั้ง เพื่อคัดทสนิยมออก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.3 ลายวงจรของระบบควบคุมอุณหภูมิแบบหลายจุด



รูปที่ 3.4 บอร์ดระบบควบคุมอุณหภูมิแบบหลายจุด

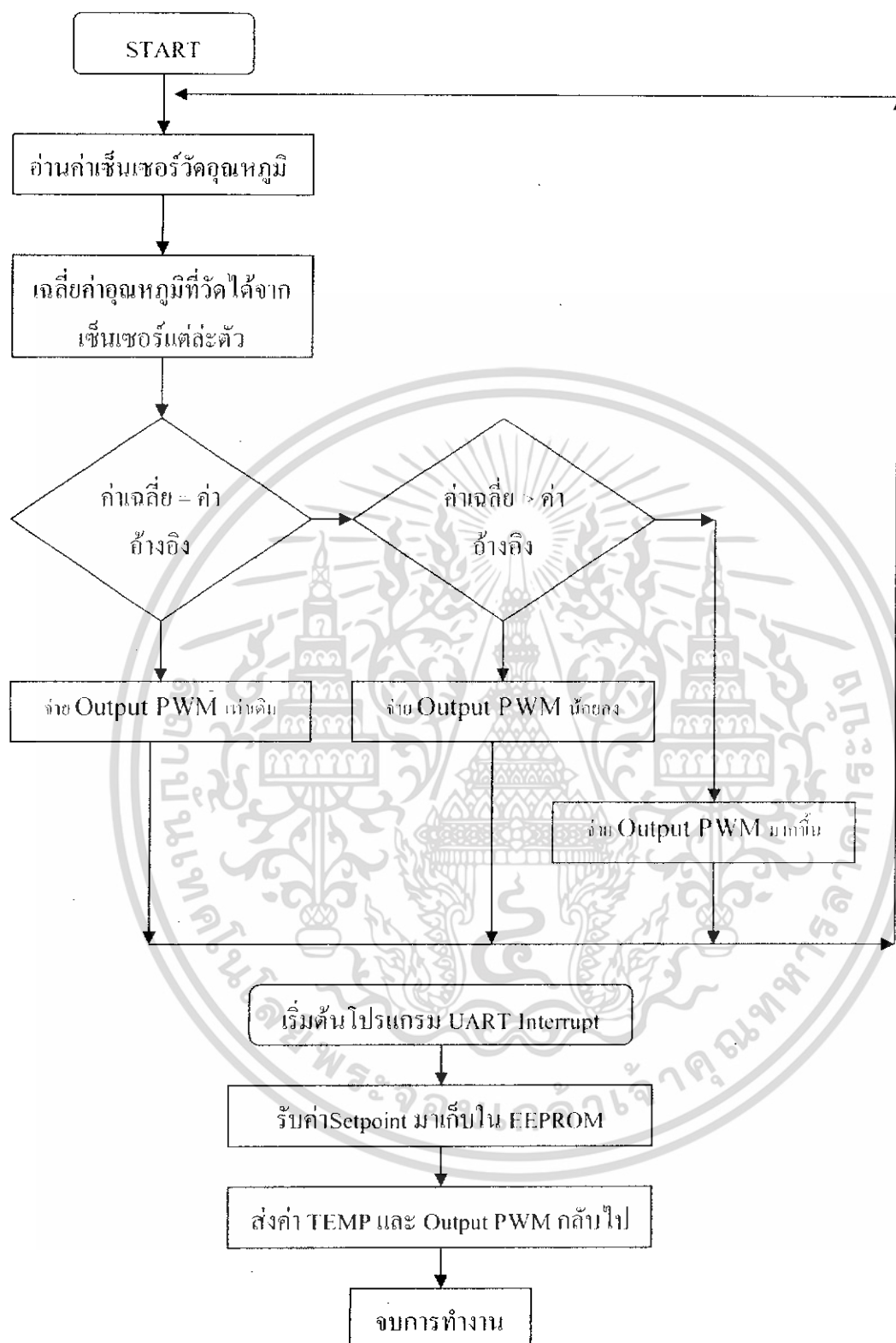
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.4 การออกแบบการทำงานในส่วนของซอฟต์แวร์

การทำงานของโปรแกรม จะเป็นการควบคุมอุณหภูมิให้ได้เท่ากับค่าที่ต้องการ โดยมีรูปแบบการทำงานคือ

1. ต้องมีการกำหนดค่าที่ต้องการลงไปเป็นค่าอุณหภูมิ 2 หลักทางเครื่องคอมพิวเตอร์เพื่อเป็นค่าอ้างอิง
2. ส่งข้อมูลที่กำหนดไว้ผ่าน โปรแกรม Modbus Commander ซึ่งจะต้องกำหนดค่า Address ของบอร์ดควบคุมอุณหภูมิแต่ละตัวด้วยว่าจะส่งไปที่ตัวใด
3. รับข้อมูลผ่านพอร์ตอนุกรมมาที่ตัวบอร์ด และ โปรแกรมอินเทอร์พรีทจะงานตามคำสั่งที่ส่งมาทาง Mosbusว่าจะให้รับค่าอ้างอิงหรือจะให้ส่งค่าออกไปแสดงผลที่คอมพิวเตอร์
4. โปรแกรมที่ตัวบอร์ดจะทำงานตามปกติคือการอ่านค่าอุณหภูมิและแสดงผลไปที่เซเว่นเซ็กเมนต์และควบคุมอุณหภูมิจนกว่าจะมีการอินเทอร์พรีทเข้ามาถึงจะกระโดดเข้าไปทำงานตามส่วนอินเทอร์พรีท
5. ให้มีการอ่านค่าอุณหภูมิ 1 รอบก่อน เพื่อดูว่าอุณหภูมิตอนนี้เท่าไรนำค่าอุณหภูมิที่อ่านมาได้ลบกับค่าอ้างอิง จะได้ค่าความผิดพลาดออกมา
6. นำค่าที่วัดได้และค่าอ้างอิงมาเปรียบเทียบกันว่าควรเพิ่มหรือลดอุณหภูมิและทำการเพิ่มหรือลดอุณหภูมิตามที่เปรียบเทียบได้
7. ระบบจะทำงานวนอยู่อย่างนี้ตลอดเพื่อทำการปรับค่าอุณหภูมิจนกว่าจะจนกว่าค่าอุณหภูมิจะเข้าอุณหภูมิจะเข้าใกล้ค่าอ้างอิง

แสดงผังการทำงานของระบบได้ในรูปที่ 3.5



รูปที่ 3.5 ฟังก์การทำงานของระบบควบคุมอุณหภูมิแบบหลายจุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.5 การออกแบบตัวควบคุม PID

การออกแบบตัวควบคุม คือการกำหนดค่า K_p , K_i , K_d (หรือเรียกว่า GAIN)

ในทางการควบคุมงาน โดยใช้ Microcontroller นั้นโดยทั่วไปการควบคุมนั้นจะไม่ออกแบบให้รับค่า K_p , K_i , K_d ได้โดยตรงแต่นิยมใช้ค่าที่เรียกว่า proportional band (PB or X_p), integral time, T_i (Reset Time) และ derivative time, T_d โดยค่าเหล่านี้มีความสัมพันธ์กับ K_p , K_i , K_d ดังนี้

- Proportional-band, $X_p = 100\% / K_p$
- Integral time (reset time), $T_i = K_p / K_i$
- Derivative time, $T_d = K_d / K_p$

จากสมการของ PID Controller

$$u = K_p e + K_i \int e dt + K_d de/dt$$

ซึ่งในกรณีนี้สามารถเขียนใหม่ได้

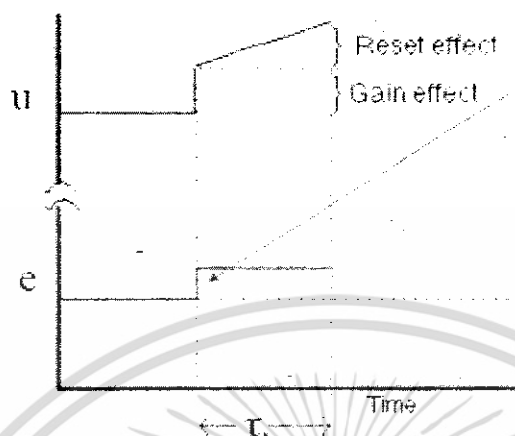
$$u = K_p (e + 1/T_i \int e dt + T_d de/dt)$$

- ความหมายของ Proportional-band

$$X_p = 100\% / K_p \Rightarrow u(t) = K_p e(t) = (100\% / X_p) e(t) \\ u = (100\% / X_p) e$$

หาก $X_p = e$ จะทำให้ u เปลี่ยนไป 100% ดังนั้นค่า Proportional band จึงสื่อถึงค่าปริมาณของ Error ที่จะทำตัวควบคุมส่งสัญญาณไปควบคุมระบบเพิ่มขึ้น 100%

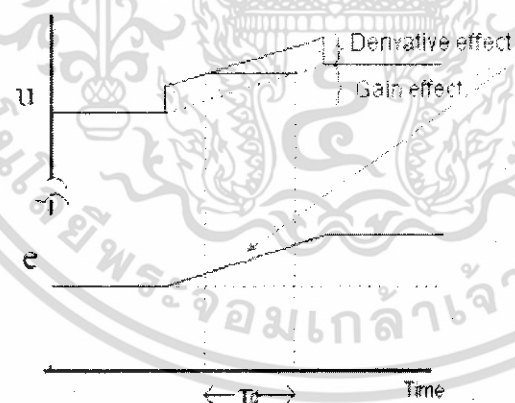
- ความหมายของ integral time (reset time)



รูปที่ 3.6 ค่า Error ที่มีการเปลี่ยนแปลงเป็น Step

หาก Error มีค่าเปลี่ยนไปแบบ step ตามรูปที่ 3.6 จะทำให้ u เปลี่ยนไปซึ่งเกิดจากผลของ P (Gain effect) ค่า T_r คือค่าของการ integrate ที่ทำให้สัดส่วนของ reset effect มีค่าเท่ากับ Gain effect

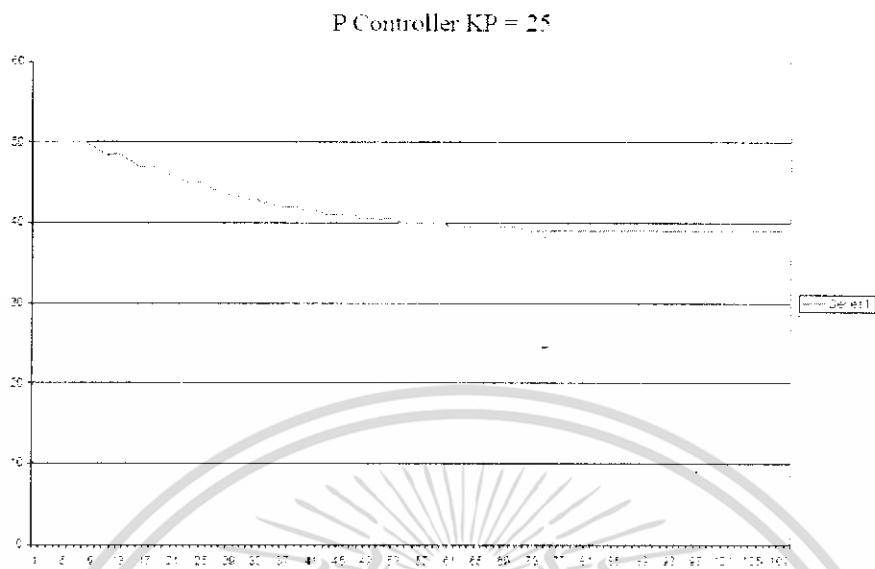
- ความหมายของ Derivative time



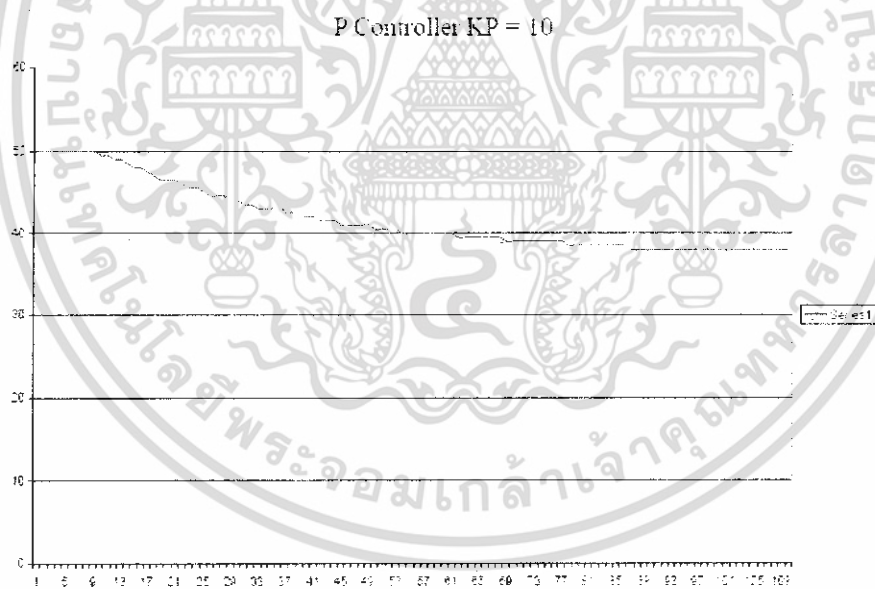
รูปที่ 3.7 ค่า Error ที่มีการเปลี่ยนแปลงเป็น ramp

หาก Error มีค่าเปลี่ยนไปแบบ ramp ตามรูปที่ 3.7 จะทำให้ u เปลี่ยนไปซึ่งเกิดจากผลของ P (Gain effect) และผลของ D (Derivative effect) ค่า T_d คือค่าเวลาที่สัดส่วนของ Gain effect จะต้องใช้ในการปรับตัวให้มีค่าเท่ากับ Derivative effect

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

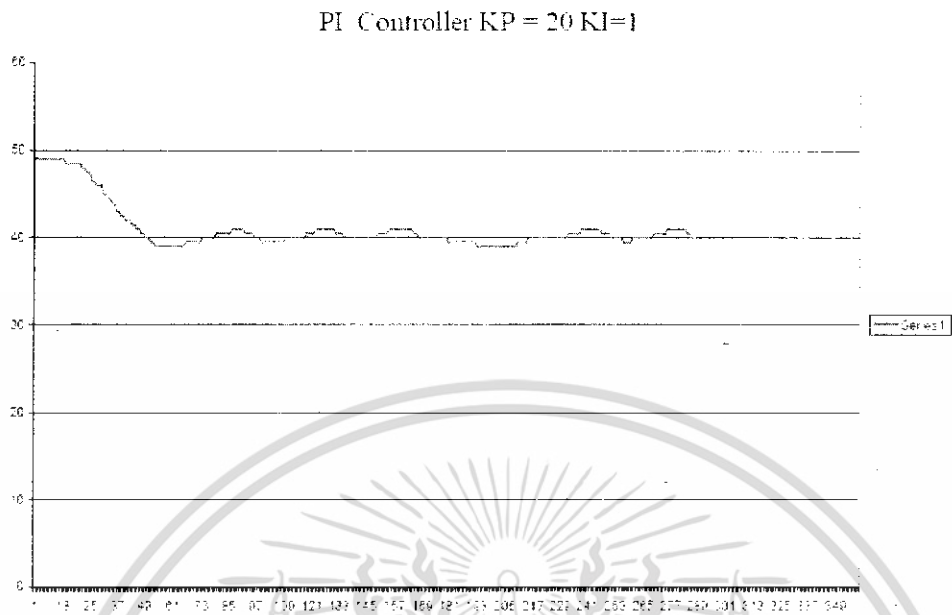


รูปที่ 4.2 กราฟอุณหภูมิที่ Setpoint 35 องศาโดยตั้งค่า $K_P = 25$



รูปที่ 4.3 กราฟอุณหภูมิที่ Setpoint 35 องศาโดยตั้งค่า $K_P = 10$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

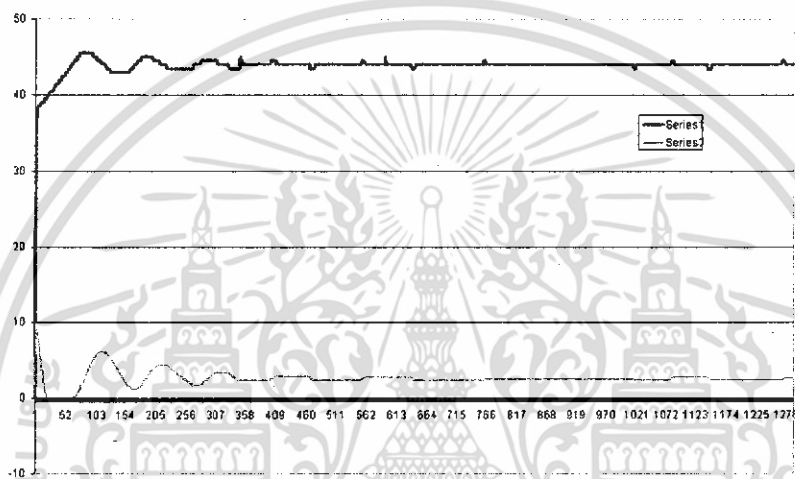


รูปที่ 4.4 กราฟพลหภูมิที่ Setpoint 40 องศาโดยตั้งค่า $K_P = 20$ $K_I = 1$

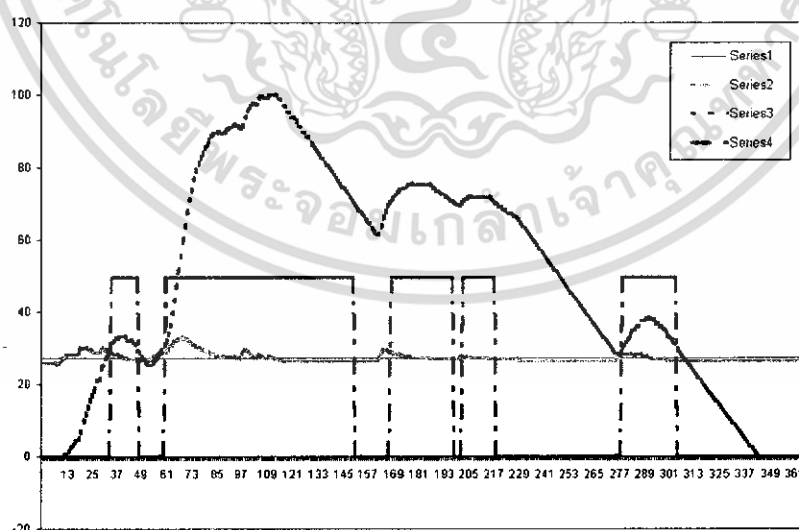
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 ผลการทดลองการทำงานของระบบควบคุมอุณหภูมิแบบหลายจุด

ผลการทดลองของระบบได้จากโปรแกรมของบอร์ดระบบควบคุมอุณหภูมิแบบหลายจุดจะทำการอ่านค่าอุณหภูมิจากเซ็นเซอร์แต่ละตัวและนำค่านั้นมาเฉลี่ยกันแล้วนำมาเปรียบเทียบกับค่า Set point ตามสมการของ PID แล้วนำมาเป็น Output ในการควบคุม โดยตั้งค่า Setpoint ไว้ที่ 27 องศาและใช้การควบคุมแบบ On/Off แล้วนำค่าต่างๆที่ได้มาพล็อตเป็นกราฟจะได้กราฟดังรูปที่ 4.5 และรูปที่ 4.6



รูปที่ 4.5 รูปกราฟแสดงถึงการเข้าถึง Setpoint และ Output แบบเปอร์เซ็น



รูปที่ 4.6 รูปกราฟแสดงผลการทำงานของระบบควบคุมอุณหภูมิแบบหลายจุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่ 4.6 เส้นตรงที่ 27 องศาจะคือค่าของ Set point และค่าอุณหภูมิจริงจะคอยลู่เข้าสู่ค่า Set point ตามหลักการของ PID ส่วนเส้นกราฟด้านล่างที่มีการเปลี่ยนแปลงแบบ Step จะเป็นการควบคุม Output แบบ On/Off ที่ใช้งานอยู่ และเส้นกราฟด้านบนจะเป็นการควบคุม Output แบบเปอร์เซ็นต์ เมื่ออุณหภูมิสูงขึ้นพัดลมจะทำงาน และจะหยุดทำงานเมื่ออุณหภูมิต่ำกว่า Set point ตามที่กำหนดไว้ในสมการ และจะทำงานวนลูปแบบนี้ต่อไป



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น. อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

สรุปการพัฒนาโครงการ

5.1 สรุปการพัฒนาโครงการ

ระบบที่นำมาใช้มีการออกแบบอย่างมีแบบแผนเพื่อให้เหมาะสมกับวัตถุประสงค์ของการนำไปใช้งาน และมีความยืดหยุ่นพอที่จะปรับเปลี่ยนให้มีการใช้งานได้หลายรูปแบบ และได้ออกแบบให้ใช้งานร่วมกับระบบอื่นๆ ได้ หรือสามารถนำมาต่อแบบเนตเวิร์กเพื่อการควบคุมที่ใหญ่ขึ้นได้ และสามารถสรุปได้ดังนี้

- สามารถควบคุมอุณหภูมิได้ละเอียดและประหยัดพลังงานที่ใช้
- มีเซ็นเซอร์หลายจุดสามารถดูได้ว่าจุดไหนอุณหภูมิมีเท่าใดเพื่อสามารถปรับอุณหภูมิจุดนั้นได้อย่างถูกต้อง
- สามารถปรับเปลี่ยนอุปกรณ์แต่ละตัวได้โดยง่าย เมื่อต้องการเปลี่ยนสภาพแวดล้อมในการควบคุม
- สามารถเพื่ออุปกรณ์เพื่อขยายระบบได้โดยง่าย

การเลือกใช้ระบบ Modbus ก็เพื่อให้ลดการใช้สายสัญญาณและจำนวนพอร์ทและใช้งานกับอุปกรณ์อื่นได้ถ้ามีการใช้งานในโรงงาน แต่อาจจะมีปัญหาถ้าจำเป็นต้องต่อสายในระยะทางไกลจำเป็นต้องใช้สายที่มีคุณภาพดีมาก

5.2 ปัญหาที่เกิดขึ้น

- ปัญหาในการคิดอัลกอริทึมสำหรับใช้งานเนื่องจากมีความหลากหลายในการใช้งานอัลกอริทึมที่ใช้และไม่ตอบสนองความต้องการได้ จึงต้องทำการศึกษาใหม่
- การใช้ UART INTERRUPT เป็นปัญหาพอสมควรในการออกแบบซอฟต์แวร์ เนื่องจากอินเทอร์รัปต์ผ่านพอร์ทอนุกรมในรูปแบบของ Modbus Protocol ซึ่งในช่วงแรกค่าที่อ่านได้จาก Modbus อ่านมาผิดพลาด ซึ่งจำเป็นต้องการศึกษา code คำสั่งและ Frame ข้อมูลของ Modbus อย่างละเอียด
- การใช้สายสัญญาณไม่ดีพอจึงไม่สามารถต่อสายสัญญาณออกไปได้ไกล ถ้าใช้สายสัญญาณทั่วไปอาจจะสามารถต่อออกไปได้ไกลเพียง 15 เมตรเท่านั้นซึ่งก็เพียงพอในการใช้งานกับระบบที่ไม่ใหญ่มาก
- การใช้เครื่องปรับอากาศขนาดเล็กเกินไปอาจจะทำให้ถึงค่า Set point ได้ช้าลง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

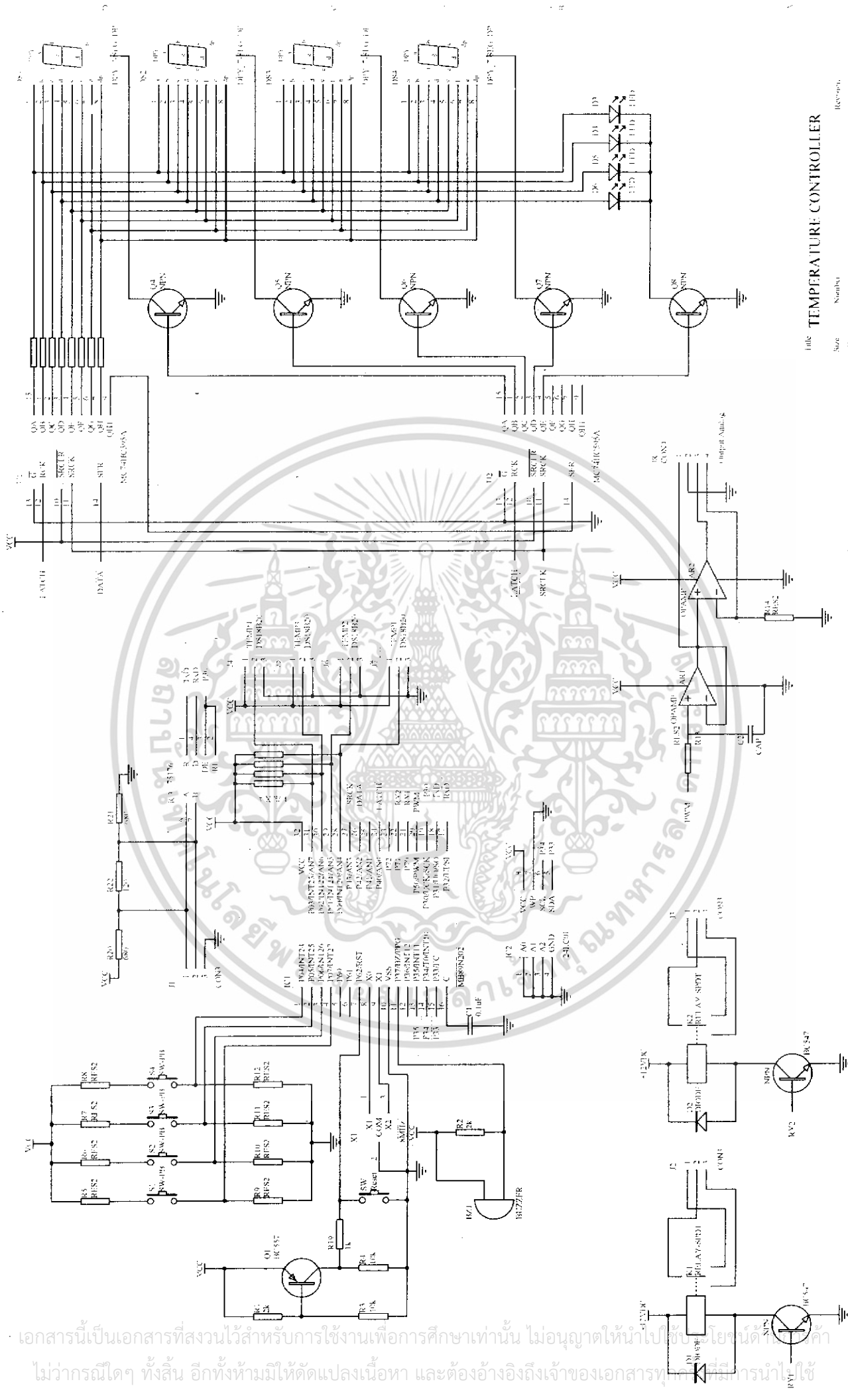
5.3 แนวทางในการพัฒนาต่อ

ระบบควบคุมอุณหภูมิแบบหลายจุดนี้ถูกออกแบบและได้ทดลองในสถานการณ์ที่จำลองขึ้นมา ซึ่งสามารถพัฒนาไปใช้ได้กับ การปรับอากาศในอาคาร การอบผลผลิตทางการเกษตร การควบคุมอุณหภูมิในอุตสาหกรรมต่างๆ ที่ต้องการการควบคุมอุณหภูมิในบริเวณกว้าง และต้องใช้อุปกรณ์วัดและจ่ายอุณหภูมิในหลายจุด ตัวอย่างเช่นการอบผลผลิตทางการเกษตร อาจนำ Output ที่ได้ไปควบคุมการจ่ายก๊าซ LPG เพื่อจ่ายเข้าไปในตู้อบ และวัดค่าของอุณหภูมิในแต่ละจุดออกมาเพื่อควบคุมการจ่าย ก๊าซ LPG เพื่อให้อุณหภูมิคงที่และได้ผลผลิตทางการเกษตรที่ผ่านการอบที่มีคุณภาพที่ควบคุมได้และใกล้เคียงกันออกมาก ทำให้มีผลผลิตที่มีคุณภาพมากขึ้นและสามารถประหยัดการใช้พลังงานได้

รวมถึงการควบคุมอุณหภูมิในห้อง Server หรือห้องที่มีอุปกรณ์คอมพิวเตอร์และอุปกรณ์สื่อสารซึ่งจะเกิดความร้อนมากในเวลาทำงาน และต้องการควบคุมอุณหภูมิที่เหมาะสมในการทำงานถ้าไม่มีการควบคุมก็อาจจะมี การทำความเย็นจนอุณหภูมิต่ำเกินไป จนสูญเสียพลังงานโดยเปล่าประโยชน์ และพัฒนาให้ควบคุม Output ได้หลายตัว

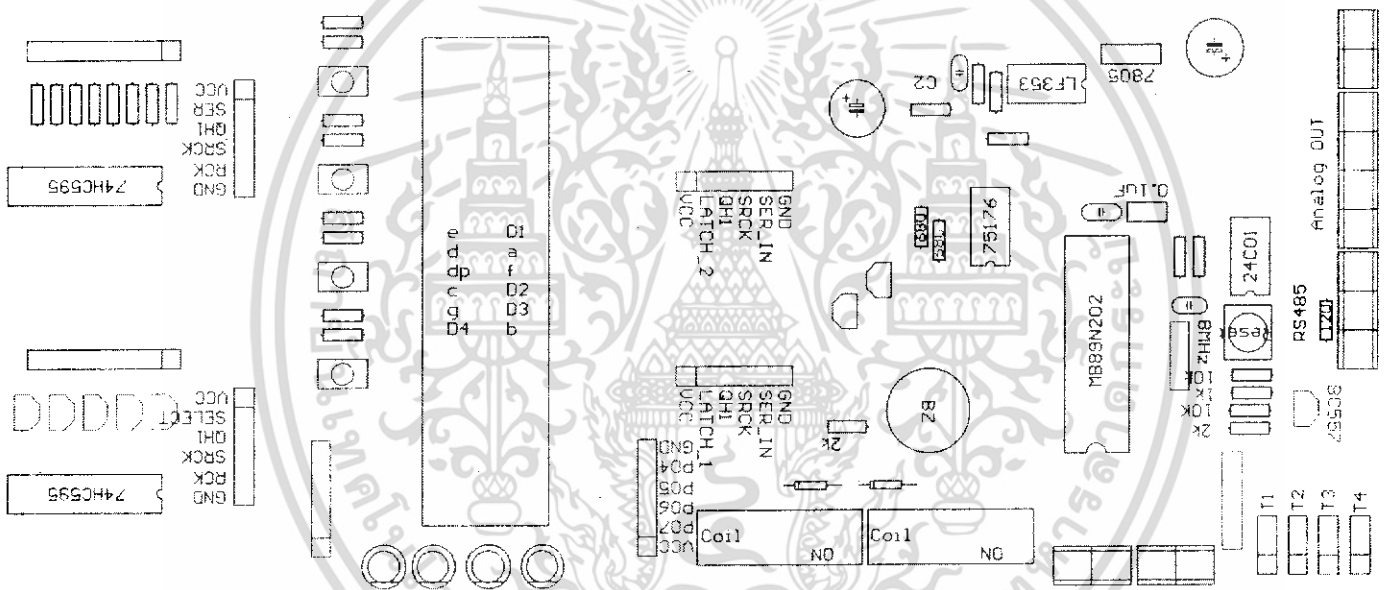


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



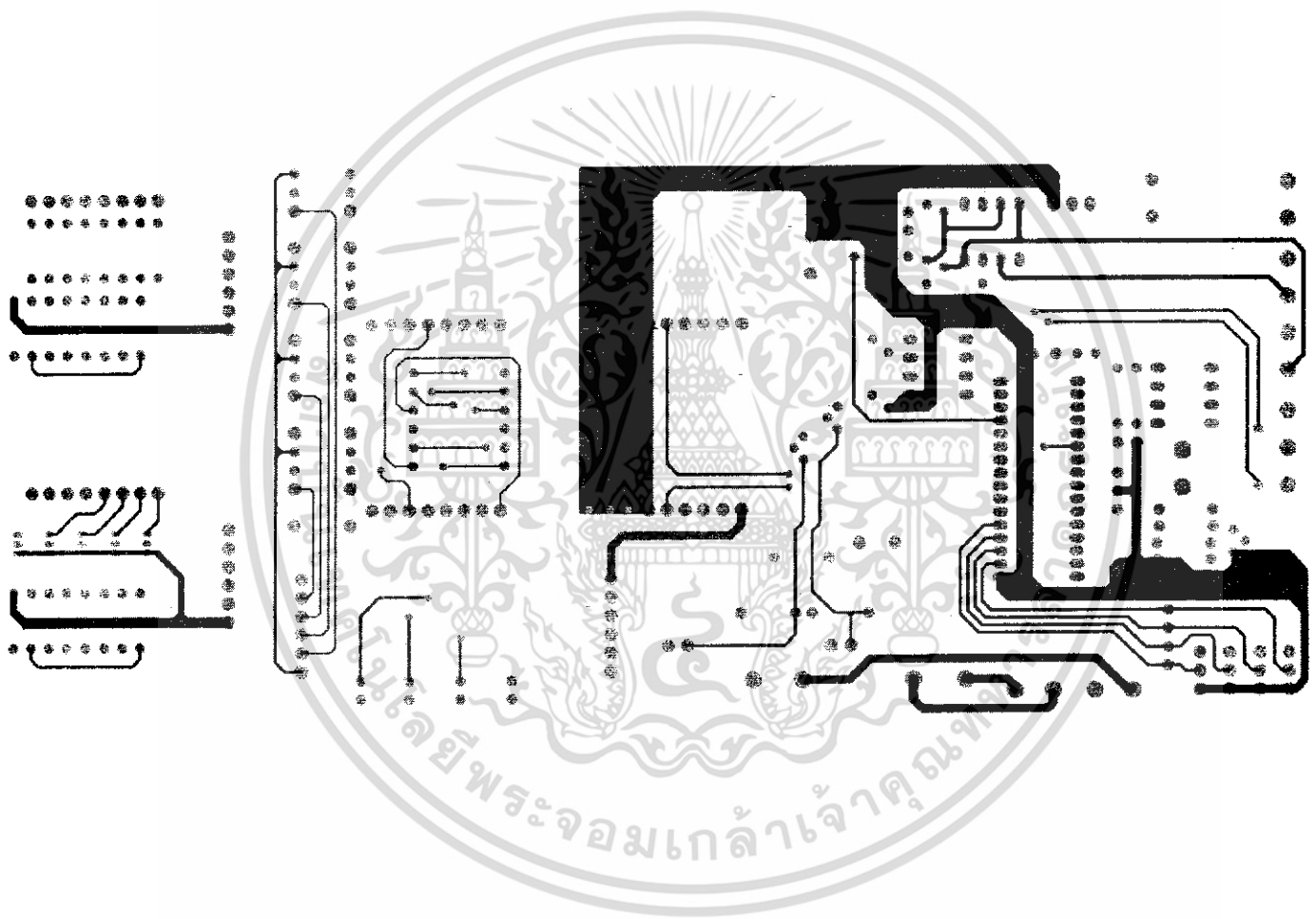
Title: TEMPERATURE CONTROLLER
 Rev: 0001
 Size: 4
 Date: 25/06/2025
 File: 1-PROJECT-DIY-BAK-Cooking-2B

เอกสารนี้เป็นเอกสารที่ สงวนลิขสิทธิ์ไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์อื่นใด
 ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารที่ทำการนำใช้



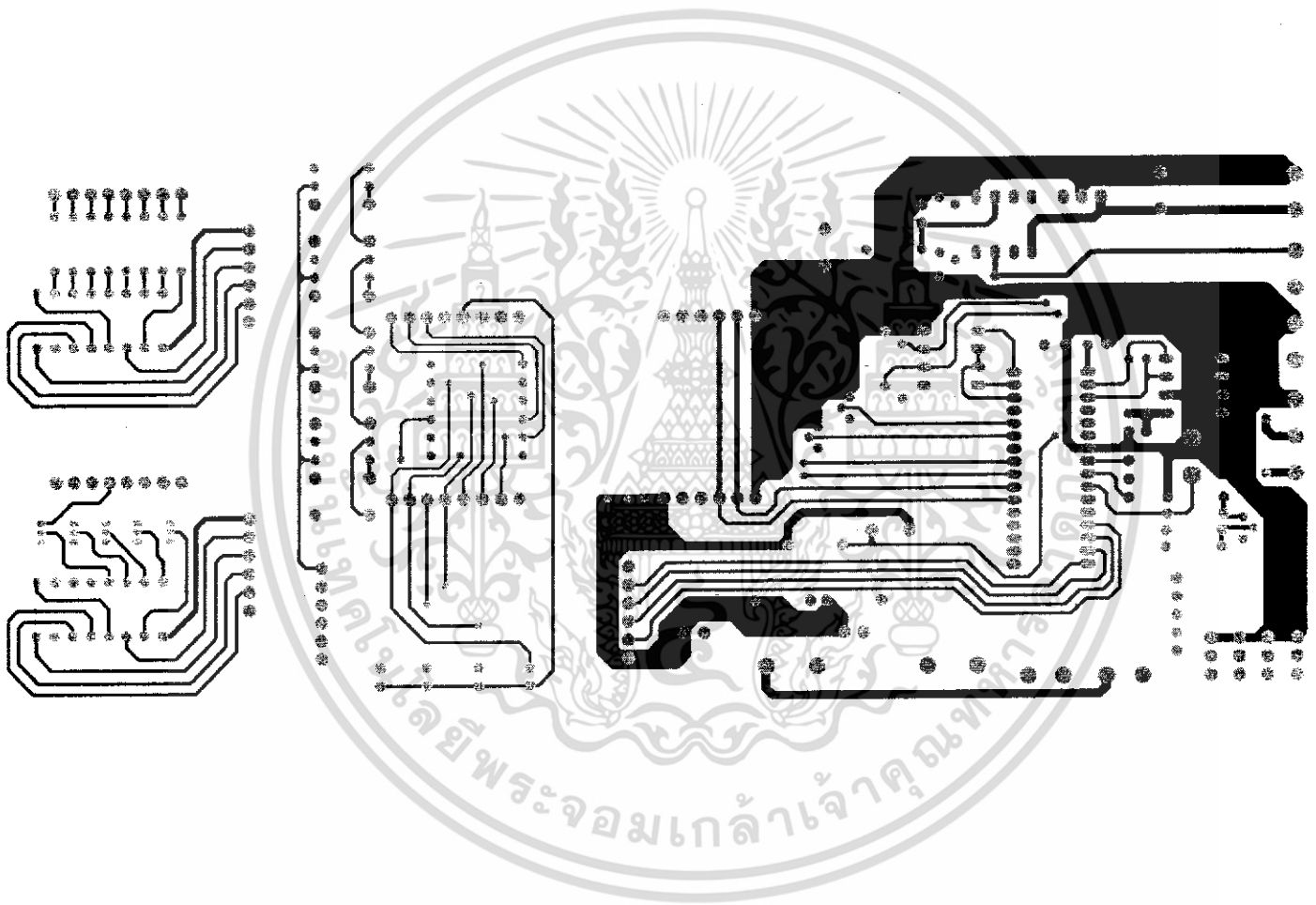
รูปการลงอุปกรณ์ของบอร์ด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปลายวงจรด้าน TOPLAYER

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปลายวงจรด้าน BOTTOMLAYER

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้