โครงสร้างและการควบคุมกระบวนการ

PROCESS CONSTRUCTION AND CONTROL

โดย

นายธีรชัย ประทุมเทือง

นางสาวอรพิน อนุรักษ์แดนไทย

อาจารย์ที่ปรึกษา

ผศ.พรสุข รติโรจน์อนันต์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมระบบควบคุม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2546

ปริญญานิพนธ์ปีการศึกษา 4546

ภาควิชาวิศวกรรมระบบควบคุม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง โครงสร้างและการควบคุมกระบวนการ

ผู้จัดทำ

1. นายธีรชัย	ประทุมเทือง	รหัสนักศึกษา	43010185
2. นางสาวอรพิน	อนุรักษ์แดนไทย	รหัสนักศึกษา	43010528

.....อาจารย์ที่ปรึกษา

(ผศ. พรสุข รติโรจน์อนันต์)

โครงสร้างและการควบคุมกระบวนการ

ธีรชัย ประทุมเทือง
อรพิน อนุรักษ์แดนไทย
ผศ.พรสุข รติโรจน์อนันต์ อาจารย์ที่ปรึกษา
ปีการปริกษา 2546

บทคัดย่อ

ปริญญานิพนธ์นี้ได้สร้างระบบควบคุมระดับน้ำ จากอุปกรณ์ที่ภาควิชามีอยู่ และพัฒนาโปรแกรมตัวควบคุมบนคอมพิวเตอร์ ทั้งแบบ PID และ PID แบบอัตโนมัติ ที่สามารถปรับแต่งค่าพารามิเตอร์ได้ จากการทดลองเปรียบเทียบผลของการควบคุม ระหว่างตัวควบคุมบนคอมพิวเตอร์กับการควบคุมโดยใช้เครื่องควบคุม SLCD Indicating Controller พบว่าการใช้ตัวควบคุมบนคอมพิวเตอร์ วิธีของ Dahlin ให้ผลตอบสนองต่อกระบวนการที่สร้างขึ้นดีกว่าวิธีของ Ziegler-Nichols ส่วนการควบคุมโดยใช้เครื่องควบคุม SLCD Indicating Controller เป็นตัวควบคุมพบว่า วิธี Ziegler-Nichols ให้ผลตอบสนองที่ดีกว่า ซึ่งการที่จะได้ผลตอบสนองที่ดีได้นั้น ขึ้นอยู่กับคุณลักษณะของระบบสมการ PID ที่ใช้ในการออกแบบตัวควบคุม รวมไปถึงการปรับแต่งพารามิเตอร์ เพื่อให้ได้ค่าที่เหมาะสมกับกระบวนการนั้นๆมากที่สุด

Abstract

This thesis proposes a construction of level process control model and designs an automatic PID controller parameter program by using Dahlin and Ziegler-Nichols method. Then use that PID parameter to control the water level and compare the controlled result between a SLCD Indicating Controller and a computer programmed controller. The result was revealed that the computer programmed controller best fits with the Dahlin's parameter. In other hand, the SLCD Indicating Controller prefers Ziegler-Nichols' parameter more than Dahlin's parameter. However, the parameters derived from these methods are usually used for normally controlling. In practice, Fine-tuning method is done once again for the best response.

สารบัญ

เรื่อง หน้า

สารบัญรูปภาพ

รูป หน้า

รูป หน้า

สารบัญตาราง

บทที่ 1
บทนำ

ปริญญานิพนธ์นี้สร้างระบบควบคุมระดับน้ำ จากอุปกรณ์ที่ภาควิชามี และทำการออกแบบตัวควบคุม โดยใช้โปรแกรมวิชวลเบสิก (Visual Basic) สร้างตัวควบคุม รวมทั้งปรับแต่ง (Tuning) ค่าพารามิเตอร์ เพื่อให้ได้ตัวควบคุมที่มีประสิทธิภาพ เหมาะกับการใช้งานในระบบขนาดเล็ก ทำให้ประหยัดค่าใช้จ่าย ง่ายต่อการออกแบบตัวควบคุมให้เหมาะสม และคุ้มค่าต่อการใช้งาน

1.1 วัตถุประสงค์ของปริญญานิพนธ์

1. ศึกษาการทำงานของอุปกรณ์วัดและควบคุมทางอุตสาหกรรมเพื่อสร้างระบบควบคุมระดับน้ำ
2. เพื่อศึกษาการรับส่งข้อมูลกับอุปกรณ์ภายนอกของคอมพิวเตอร์โดยผ่านพอร์ตอนุกรม
3. เพื่อศึกษาหลักการและแนวคิดในการเขียนโปรแกรมเพื่อใช้ในการแสดงผลและควบคุมกระบวนการ
4. เพื่อศึกษาหลักการหาทรานสเฟอร์ฟังก์ชันของระบบ
5. เพื่อศึกษาการวิเคราะห์หาค่าพารามิเตอร์ของตัวควบคุม PID แบบอัตโนมัติ โดยใช้คอมพิวเตอร์ส่วนบุคคล
6. เพื่อศึกษาวิเคราะห์และปรับแต่งค่าพารามิเตอร์เพื่อให้ได้ค่าพารามิเตอร์ที่เหมาะสมกับกระบวนการ

1.2 ขอบเขตของปริญญานิพนธ์

1. ศึกษาการทำงานและสร้างระบบควบคุมระดับน้ำ
2. สร้างอุปกรณ์เชื่อมต่อระหว่างคอมพิวเตอร์กับระบบ โดยใช้ ไมโครคอนโทรลเลอร์PIC16F628 ไอซีแปลงสัญญาณดิจิตอลเป็นอนาลอกและอนาลอกเป็นดิจิตอล และวงจรแปลงสัญญาณแรงดันเป็นกระแส
3. ศึกษาการรับส่งข้อมูลแบบอนุกรมระหว่างคอมพิวเตอร์กับระบบ
4. เขียนโปรแกรมเชื่อมต่อคอมพิวเตอร์กับระบบ และออกแบบตัวควบคุมแบบ PID โดยใช้โปรแกรมวิชวลเบสิก
5. เขียนโปรแกรมแสดงผลการทำงานของระบบ คำนวณค่าทรานสเฟอร์ฟังก์ชันของระบบ คำนวณค่าพารามิเตอร์ของตัวควบคุม PID และปรับปรุงค่าพารามิเตอร์ของตัวควบคุม PID เพื่อให้ได้ตัวควบคุมที่มีประสิทธิภาพ

บทที่ 2
ทฤษฎีและหลักการ

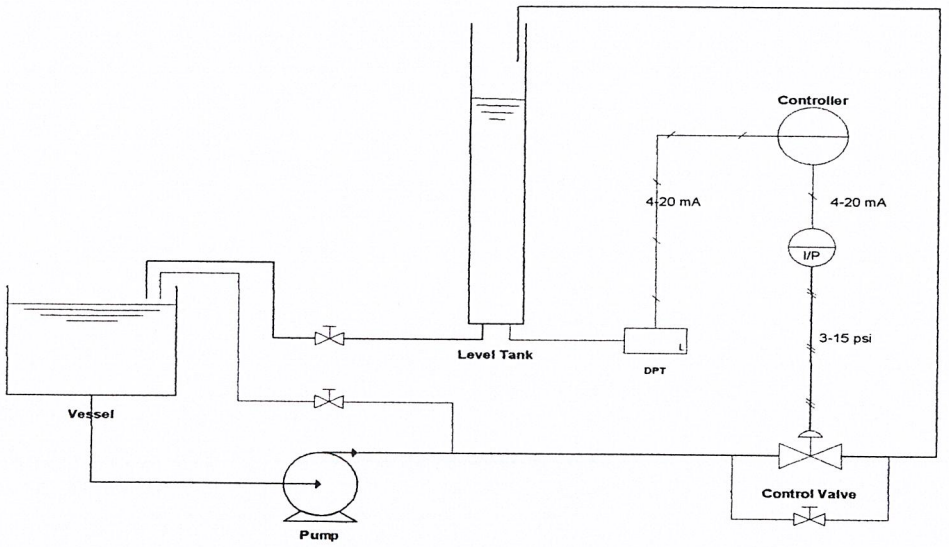
2.1 กล่าวนำ

ระบบควบคุมระดับน้ำที่สร้างขึ้นประกอบด้วย 4 ส่วน คือ

1. ตัวควบคุม(Controller) เป็นเครื่องมือหรืออุปกรณ์ ที่ใช้ในการสร้างสัญญาณควบคุม เพื่อทำหน้าที่ควบคุม ให้ระบบหรือกระบวนการที่ต้องการควบคุมมีเอาท์พุท หรือผลตอบสนองเป็นไปตามต้องการ ตัวควบคุมมีโหมดการควบคุมได้หลายโหมด เช่น ตัวควบคุมแบบเปิด-ปิด(ON-OFF),ตัวควบคุมแบบพี,ตัวควบคุมแบบไอ,ตัวควบคุมแบบดี หรือการใช้ตัวควบคุมหลายๆแบบร่วมกัน เช่น ตัวควบคุมแบบพีไอ ตัวควบคุมแบบพีดี และตัวควบคุมแบบพีไอดี เป็นต้น ซึ่งสัญญาณที่ออกจากตัวควบคุม คือ ตัวแปรปรับกระบวนการ(Manipulated Variable)
2. อุปกรณ์ควบคุมขั้นสุดท้าย(Final Control Element) คือ อุปกรณ์ที่ทำหน้าที่ปรับสภาวะของกระบวนการ ด้วยการเปลี่ยนแปลงค่าตัวแปรปรับกระบวนการหรือสัญญาณควบคุมที่ได้รับจากตัวควบคุม อุปกรณ์ควบคุมขั้นสุดท้ายนั้นมีอยู่ด้วยกันหลายอย่าง เช่น วาล์วควบคุม(Control Valve) อินเวอร์เตอร์(Inverter) และ ตัวขับเคลื่อน(Actuator) เป็นต้น แต่ที่มักพบเห็นกันมากในกระบวนการทางอุตสาหกรรม ได้แก่ วาล์วควบคุม(Control Valve)
3. กระบวนการ (Plant or Process) หมายถึงระบบหรือกระบวนการทางฟิสิกส์ที่ต้องการควบคุมให้สถานะเป็นไปตามต้องการ เช่น กระบวนการเกี่ยวกับการควบคุมระดับของเหลว หรือกระบวนการเกี่ยวกับการควบคุมอุณหภูมิ เป็นต้น ซึ่งสถานะของกระบวนการแสดงด้วยตัวแปรกระบวนการ(Process Variable)
4. อุปกรณ์วัด (Measuring Instruments) หมายถึง อุปกรณ์ซึ่งได้แก่ เซ็นเซอร์(Sensor), ทรานสดิวเซอร์(Transducer), อุปกรณ์ทรานสมิตเตอร์(Transmitter) เครื่องวัดสัญญาณอื่นๆในกระบวนการ เพื่อนำสัญญาณที่ได้ไปใช้เป็นตัวแปรในการควบคุม โดยสัญญาณขาออกของอุปกรณ์วัดทั่วไปจะเป็นสัญญาณมาตรฐานทางอุตสาหกรรม เช่น สัญญาณกระแสไฟฟ้า 4-20 มิลลิแอมป์ (4-20 mA), สัญญาณแรงดันไฟฟ้ากระแสตรง 1-5 โวลต์ (1-5 Vdc) หรือสัญญาณลมขนาด 3-15 ปอนด์/ตารางนิ้ว (3-15 psi หรือ 0.2-1.0 Kg/cm^2) เป็นต้น

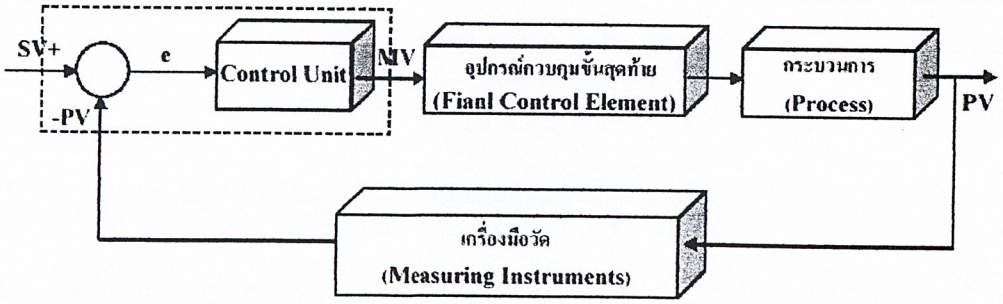
2.2 การทำงานของกระบวนการ

ในส่วนนี้เป็นการแสดงโครงสร้างและการทำงานของกระบวนการควบคุมระดับน้ำ ดังรูปที่ 2.1



รูปที่ 2.1 โครงสร้างของการกระบวนการควบคุมระดับน้ำ

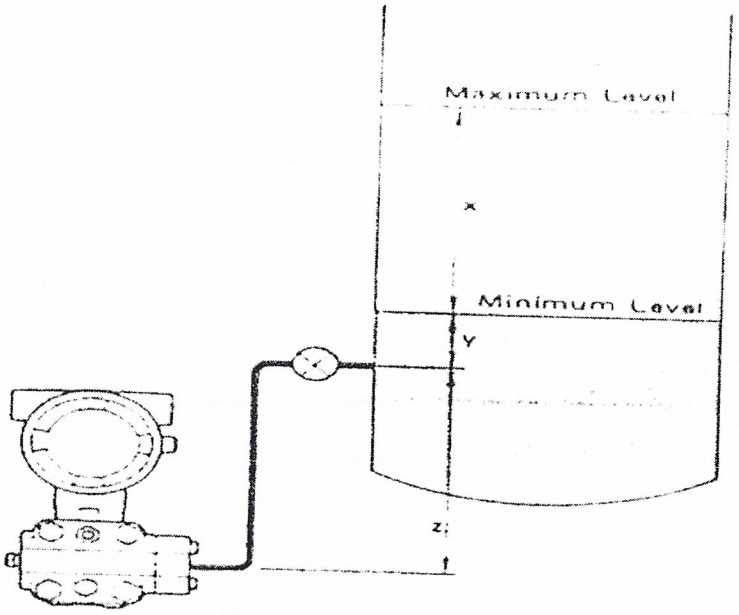
การทำงานเริ่มจาก ปั๊มทำหน้าที่สูบน้ำจากถังเก็บน้ำ(Vessel) ไปยังถังแสดงระดับน้ำ (Level Tank) โดยมี ดิฟเฟอเรนเชียล เพรสเชอร์ ทรานสมิตเตอร์ (Differential Pressure Transmitter) เป็นตัววัดระดับน้ำ และส่งค่าสัญญาณที่วัดได้ไปยังอุปกรณ์ควบคุม (Controller) ซึ่งทำหน้าที่คำนวณ และส่งสัญญาณออกมาควบคุมคอนโทรลวาล์ว เพื่อควบคุมปริมาณน้ำที่ไหลไปยังถังแสดงระดับน้ำ (Level Tank) ให้ได้ระดับตามที่ตั้งไว้ในอุปกรณ์ควบคุม (Controller) ซึ่งสามารถปรับค่าได้ตามที่ต้องการ



รูปที่ 2.2 บล็อกไดอะแกรมแสดงการทำงานของระบบควบคุมแบบป้อนกลับ

2.3 หลักการวัดระดับโดยวิธีวัดความดันดิฟเฟอเรนเชียล

หลักการวัดระดับ โดยวิธีวัดคความดันดิฟเฟอเรนเชียล เป็นที่แพร่หลายในวงการอุตสาหกรรมวิธีหนึ่ง สามารถใช้กับของเหลวที่สกปรก มีความดันหรืออุณหภูมิสูงได้ดี



รูปที่ 2.3 แสดงการวัดระดับแบบดิฟเฟอเรนเชียล

หลักการวัด ตามสมการเบื้องต้น ดังนี้

$$P=h\times SG_m \qquad \text{.....(2.1)}$$

เมื่อ $P=$ ค่าความดัน มีหน่วยเป็น เมตรน้ำ

$h=x+y+z$ คือ ค่าความสูงของของเหลว มีหน่วยเป็น เมตร

$SG_m=$ ค่าความถ่วงจำเพาะของสาร(ไม่มีหน่วย)

พิจารณาจากสมการ ค่าความถ่วงจำเพาะของสารเป็นค่าคงที่ ซึ่งเป็นคุณสมบัติประจำตัวของสารนั้น เช่น น้ำมีความถ่วงจำเพาะเป็น 1, ปรอท มีค่าความถ่วงจำเพาะ 13.6 ดังนั้น ความดันที่เกิดขึ้นจึงมีความสัมพันธ์โดยตรงกับความสูงของของเหลวนั้น ดังสมการใหม่นี้

$$h=\frac{P}{SG_m} \qquad \text{.....(2.2)}$$

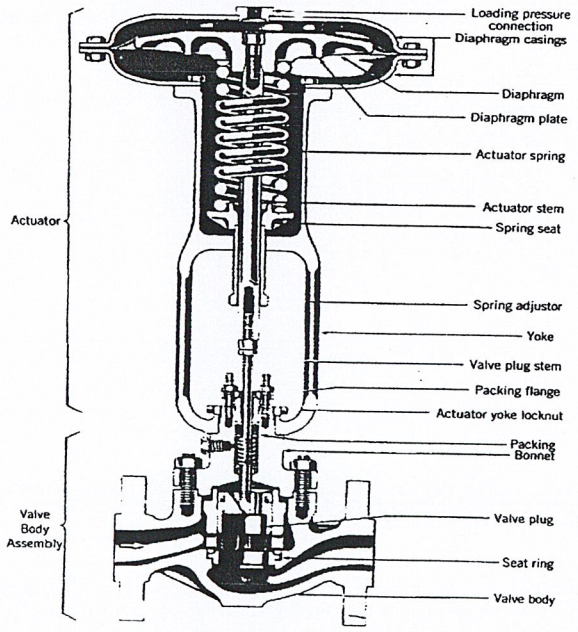
ในกระบวนการนี้ ใช้ดิฟเฟอเรนเชียล เพรสเชอร์ ทรานสมิตเตอร์(Differential Pressure Transmitter) วัดความแตกต่างระหว่างความดันที่ก้นถังแสดงระดับน้ำ (Level Tank) กับความดันบรรยากาศ และส่งสัญญาณออกมาเป็นสัญญาณมาตรฐาน 4-20 มิลลิแอมป์ ไปยังตัวควบคุม (Controller)

2.4 อุปกรณ์พื้นฐานสำหรับการควบคุมกระบวนการ

2.4.1 วาล์วควบคุม (Control Valve)

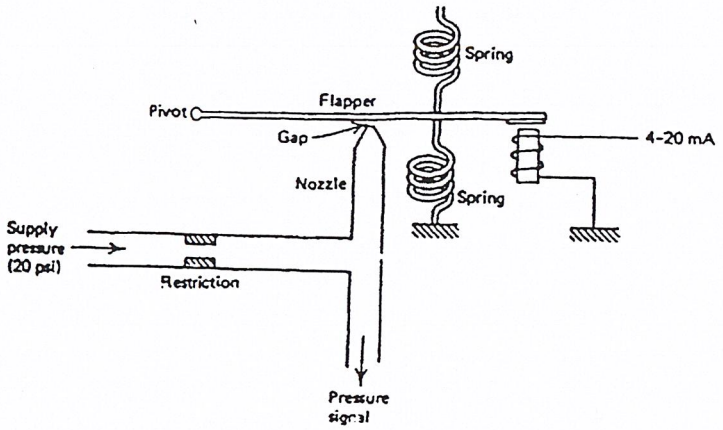
เป็นอุปกรณ์ควบคุมขั้นสุดท้าย (Final Control Element) ที่ทำงานเชื่อมต่อระหว่างตัวควบคุม (Controller) กับกระบวนการ (Process) และยังเป็นตัวชดเชยกระบวนการได้อีกด้วย วาล์วควบคุมประกอบด้วย ตัวกระตุ้นให้เกิดการทำงาน (Power Actuator) และตัววาล์ว (Valve Body) และที่ตัววาล์วมีซีท (Seat) และปลั๊ก (Plug) ตัวกระตุ้นจะควบคุมการเคลื่อนที่ของปลั๊กขึ้นลงในแนวดิ่งให้นั่งที่ซีท ตามสัญญาณควบคุมของตัวควบคุม เป็นการบังคับพื้นที่ส่วนที่ให้ของเหลวไหลผ่าน (Port Area) เพื่อควบคุมอัตราการไหลของของเหลวให้คงที่หรือที่เรียกว่า Throttle the flow

ส่วนของตัวกระตุ้นประกอบด้วย ไดอะแฟรมเคส (Diaphragm case), ไดอะแฟรมเพลท (Diaphragm plate) และสปริง (spring) ทั้งตัวกระตุ้นและตัววาล์ว สามารถออกแบบให้ทำงานแบบทางตรง (Direct-Normally Open) และกลับทาง (Reverse-Normally Closed)



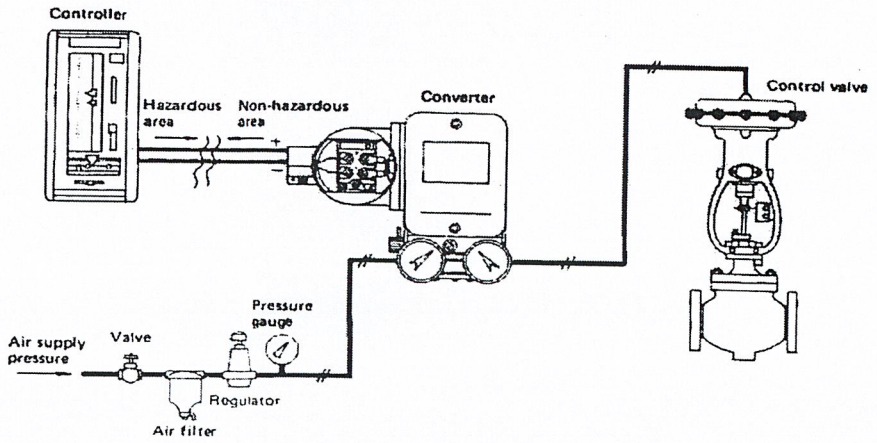
รูปที่ 2.4 โครงสร้างของวาล์วควบคุม

2.4.2 เครื่องเปลี่ยนกระแสเป็นแรงดันลม (Current-to-pressure converter)



รูปที่ 2.5 เครื่องเปลี่ยนกระแสเป็นแรงดันลม

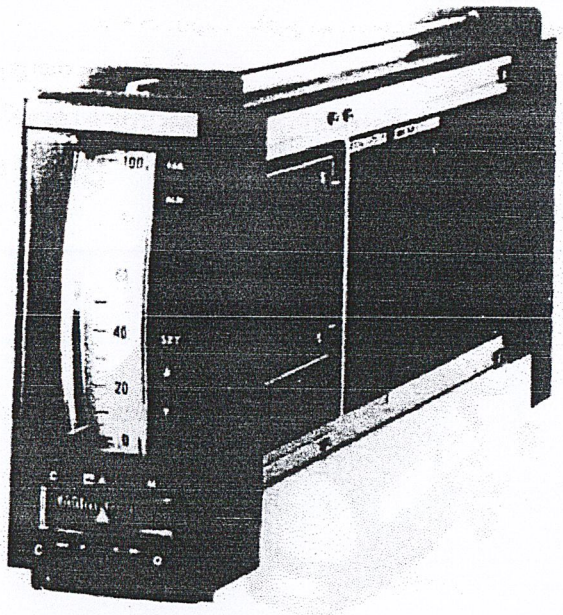
เครื่องเปลี่ยนกระแสเป็นแรงดันลม (I/P converter) เป็นส่วนประกอบที่สำคัญในกระบวนการควบคุม (Process Control) เพราะเมื่อเราต้องการใช้งานสัญญาณกระแสที่มีค่าน้อยๆสะดวก จะให้ค่าเป็นเชิงเส้น โดยแปลง 4-20 mA.(มิลลิแอมป์) ไปเป็น 3-15 psig ซึ่งเครื่องแปลงชนิดนี้มีหลายรูปแบบ แต่ที่เป็นพื้นฐานสำคัญจะประกอบไปด้วย นอซเซิล (Nozzle), แฟลบเปอร์(Flapper) รูปที่ 2.5 จะแสดงวิธีทางพื้นฐานของเครื่องเปลี่ยนกระแสเป็นแรงดันลม สังเกตในรูปภาพกระแสจะผ่านขดลวด และสร้างแรงที่จะขยับแฟลบเปอร์ลงเพื่อปิดช่องว่าง ค่ากระแสมากๆจะสร้างความดันสูง ดังนั้นอุปกรณ์ชนิดนี้ กระทำโดยวิธีทางตรง การปรับสปริงและตำแหน่ง สัมพันธ์กับแกนหมุนที่ติดกับแฟลบเปอร์ ให้ปรับวัดขนาด 4 mA. ให้เท่ากับ 3 psig และ 20 mA. ให้เท่ากับ 15 psig



รูปที่ 2.6 วาล์วควบคุมที่ต่อกับเครื่องเปลี่ยนกระแสเป็นแรงดันลม

2.4.3 ตัวควบคุม (Controller)

ตัวควบคุมทำหน้าที่เป็นอุปกรณ์ในการตัดสินใจให้กับระบบควบคุม โดยการเปรียบเทียบสัญญาณของกระบวนการที่ได้จากอุปกรณ์วัด และส่งสัญญาณซึ่งก็คือตัวแปรที่ถูกควบคุมกับจุดเป้าหมาย จากนั้นก็จะส่งสัญญาณที่เหมาะสมให้กับวาล์วควบคุมหรืออุปกรณ์ควบคุมขั้นสุดท้ายแบบอื่น เพื่อที่จะรักษาไว้ซึ่งค่าของตัวแปรที่ถูกควบคุมกับจุดเป้าหมาย



รูปที่ 2.7 SLCD อินดิเคติ้ง คอนโทรลเลอร์ (SLCD Indicating Controller)

ในโครงงานนี้ใช้ SLCD อินดิเคติ้ง คอนโทรลเลอร์ (SLCD Indicating Controller) เป็นอุปกรณ์ควบคุมซึ่งใช้ไมโครโปรเซสเซอร์เป็นพื้นฐานในการทำงานของเครื่อง โดยมีคุณสมบัติมาตรฐานของเครื่อง (Standard Specification) ดังนี้

สัญญาณเข้า/ออก (Input/Output Signal)

สัญญาณเข้าแบบอนาลอก (Analog Input) : ระดับ 1 ถึง 5 V DC มีจำนวน 4 จุด

สัญญาณออกแบบอนาลอก (Analog Output) : ระดับ 1 ถึง 5 V DC มีจำนวน 2 จุด

: ระดับ 4 ถึง 20 mA มีจำนวน 1 จุด

สถานะของสัญญาณเข้า (Status Input) : เป็นContactหรือสัญญาณแรงเคลื่อนไฟฟ้า (Voltage signal)1 จุด

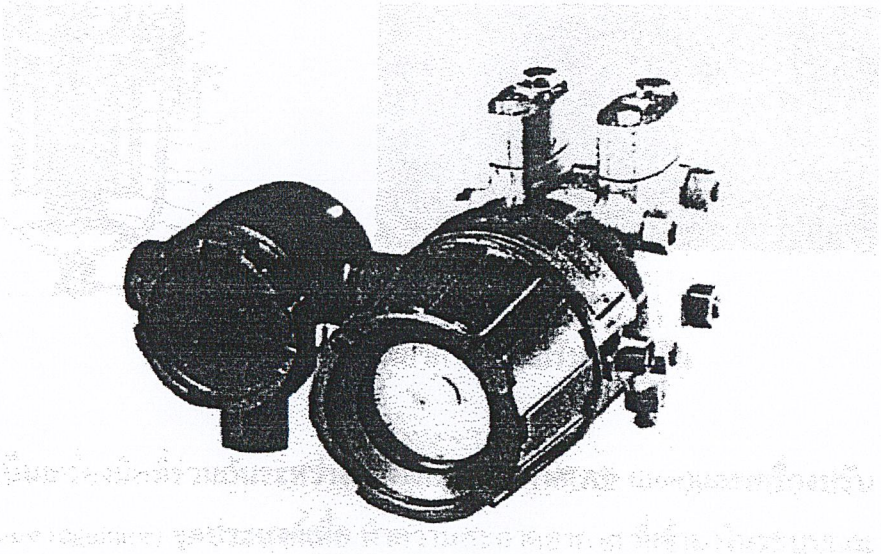
สถานะของสัญญาณออก (Status Output) : Transistor Contact จำนวน 5 จุด

สัญญาณผิดปกติ (Fail Output) : เป็น Transistor Contact 1 จุด Contact เปิดเมื่อมีการผิดปกติของPower Supply หรืออื่นๆ

สัญญาณควบคุมของเครื่องควบคุม มี 2 แบบคือ

- สัญญาณควบคุมแบบกลับ (Reverse action) บางผู้ผลิตอาจเรียกว่า กริยาลดลง (decrease) นั่นหมายความว่า ถ้าตัวควบคุมรับสัญญาณวัด(Manipulate Signal)เข้ามามากขึ้น เป็นผลให้สัญญาณควบคุมลดลง
- สัญญาณควบคุมแบบตรง (Direct action) บางผู้ผลิตอาจเรียกว่า กริยาเพิ่มขึ้น (increase) นั่นหมายความว่า ถ้าตัวควบคุมรับสัญญาณวัด(Manipulate Signal)เข้ามามากขึ้น เป็นผลให้สัญญาณควบคุมเพิ่มขึ้น

2.4.4 ทรานสมิตเตอร์ สำหรับวัดความดันดิฟเฟอเรนเชียล (Differential Pressure Transmitter)



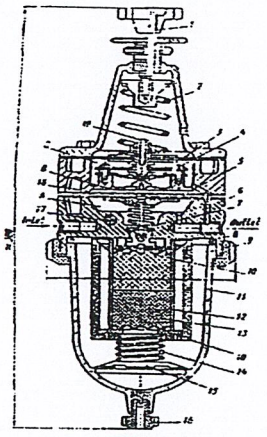
รูปที่ 2.8 แสดงดีพีทรานสมิตเตอร์

ความดันดิฟเฟอเรนเชียล มีที่ประยุกต์ใช้งานอย่างกว้างขวางในอุตสาหกรรม เช่น การวัดอัตราการไหล วัดระดับของเหลว วัดความหนืดหรือวัดค่าความหนาแน่นของของเหลว ในปัจจุบันอุปกรณ์สำหรับวัดความดันดิฟเฟอเรนเชียล พร้อมทรานสมิตเตอร์ได้รับการออกแบบให้อยู่ในแคปซูลเดียวกันมีชื่อเรียก

ย่อๆว่า ดีพีทรานสมิตเตอร์ (dP Transmitter) ในส่วนของทรานสมิตเตอร์ ส่วนใหญ่จะเป็นแบบอิเล็กทรอนิกส์ ให้สัญญาณเอาต์พุต เป็นกระแสหรือในรูปของโวลท์เตจ ซึ่งมีข้อดีคือ สามารถส่งสัญญาณนี้ไปยังเครื่องบันทึก เครื่องควบคุมหรือชี้บอกค่าได้หลายอย่างพร้อมกันในระยะที่ห่างออกไปจากจุดที่วัดค่าได้ ดีพีทรานสมิตเตอร์ ในปัจจุบันได้รับการออกแบบให้เหมาะสมและสะดวกต่อการใช้งานมากขึ้น เช่น ทนต่อสภาพแวดล้อม การเปลี่ยนแปลงของอุณหภูมิ ความชื้นได้ดี ให้ผลการทำงานที่ดี มีความเที่ยงตรงสูง Dead Band ต่ำ ลิเนียริตีดี การเปลี่ยนค่าของความดันสถิต (Static Pressure) มีผลน้อยต่อการวัดความดันดิฟเฟอเรนเชียลค่าเดียวกัน เอาต์พุตของทรานสมิตเตอร์ จะถูกกำหนดให้อยู่ในมาตรฐานเดียวกัน โดยสมาคมผู้ผลิตดังนี้

- เอาต์พุต แบบนิวเมติก 3-15 psi
- เอาต์พุต แบบกระแส 4-20 mA.
- เอาต์พุต แบบโวลเตจ 1-5 Volt

2.4.5 เครื่องกรองอากาศและปรับความดัน (Air filter and Pressure Regulator)



รูปที่ 2.9 เครื่องกรองอากาศและปรับความดัน

เป็นเครื่องมือที่ร่วมกันระหว่างเครื่องกรองอากาศ(Air filter) และเครื่องปรับความดัน (Pressure regulator) จุดประสงค์เพื่อ ทำความสะอาดอากาศ ที่เข้ามาในระบบ และปรับค่าความดันที่ต้องการใช้งานให้ได้ตามต้องการ โดยส่งผ่านไปส่วนแสดงผลในการวัด (Air gage indicating unit)

2.5 การใช้เครื่องคอมพิวเตอร์ควบคุมกระบวนการแบบดิจิตอลโดยตรง

การใช้เครื่องคอมพิวเตอร์ควบคุมกระบวนการแบบดิจิตอลโดยตรง ทำให้เกิดระบบควบคุมใหม่ที่มีคุณสมบัติบางประการต่างจากระบบควบคุมแบบเดิม ดังนี้

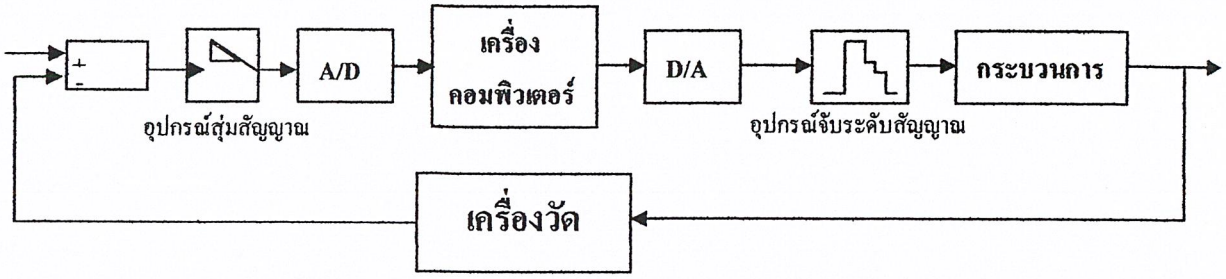
1. เครื่องคอมพิวเตอร์สามารถประมวลผลข้อมูล และใช้ในการคำนวณได้ดีกว่าเครื่องควบคุม หรืออุปกรณ์ควบคุมแบบเดิม ทำให้ระบบควบคุมที่ใช้เครื่องคอมพิวเตอร์สามารถควบคุมกระบวนการขนาดใหญ่ได้ดีกว่าระบบควบคุมแบบเดิม

2. เครื่องคอมพิวเตอร์สามารถเปลี่ยนแปลง แก้ไข หรือปรับปรุงโปรแกรมที่ใช้กำหนดเงื่อนไขในการควบคุมได้ง่ายกว่า ระบบควบคุมแบบเดิมซึ่งยุ่งยากกว่ามากหรือไม่สามารถทำได้เลย

3. เครื่องคอมพิวเตอร์สร้างสัญญาณควบคุม ตามเงื่อนไขที่กำหนด โดยโปรแกรมควบคุม คุณสมบัติของเครื่องควบคุมที่ใช้เครื่องคอมพิวเตอร์ประมวลผล จึงไม่เปลี่ยนแปลงคลาดเคลื่อน ต่างจากระบบควบคุมแบบเดิม ที่ใช้วงจรอิเล็กทรอนิกส์ นิวแมติกส์ หรือเครื่องจักรต่างๆกำหนดเงื่อนไขการควบคุม ซึ่งคุณสมบัติของอุปกรณ์เหล่านี้มักเปลี่ยนแปลงอยู่เสมอ และเกิดสัญญาณรบกวนภายในอุปกรณ์ได้ง่าย ทำให้คุณสมบัติรวมของเครื่องควบคุมเปลี่ยนแปลงตามตลอดเวลา

4. เครื่องคอมพิวเตอร์สามารถคำนวณ และประมวลผลข้อมูล โดยใช้สัญญาณดิจิตอลชนิดไม่ต่อเนื่อง แต่สัญญาณจากกระบวนการที่ต้องการควบคุม เป็นสัญญาณอนาลอกชนิดต่อเนื่อง การแปลงสัญญาณชนิดต่อเนื่อง เป็นสัญญาณชนิดไม่ต่อเนื่องทำให้สัญญาณบางช่วงเวลาขาดหายไป และการแสดงระดับของสัญญาณอนาลอก โดยสัญญาณดิจิตอล และตัวเลขจำนวนรู้จบ จะทำให้ค่าของระดับสัญญาณคลาดเคลื่อนขึ้นด้วยเสมอ ซึ่งต่างจากเครื่องควบคุมแบบเดิมที่อุปกรณ์ต่างๆ ที่ประมวลสัญญาณ โดยใช้สัญญาณอนาลอกชนิดต่อเนื่องโดยตรง จึงไม่มีความคลาดเคลื่อนในการแปลงสัญญาณอนาลอกชนิดต่อเนื่องเป็นสัญญาณดิจิตอลชนิดไม่ต่อเนื่อง เช่นเดียวกับการใช้เครื่องคอมพิวเตอร์ควบคุมกระบวนการแบบดิจิตอลโดยตรง

5. การออกแบบระบบควบคุมชนิดสัญญาณไม่ต่อเนื่อง โดยใช้เครื่องคอมพิวเตอร์เป็นเครื่องควบคุมกระบวนการแบบดิจิตอลโดยตรง มีความซับซ้อนและยุ่งยากกว่าการออกแบบระบบควบคุมชนิดสัญญาณต่อเนื่องมาก การออกแบบระบบควบคุมชนิดสัญญาณไม่ต่อเนื่องจึงออกแบบระบบควบคุมชนิดสัญญาณต่อเนื่องขึ้นเอง แล้วจึงแปลงระบบควบคุมชนิดสัญญาณต่อเนื่องเป็นระบบควบคุมชนิดสัญญาณไม่ต่อเนื่องภายหลัง



รูปที่ 2.10 การใช้เครื่องคอมพิวเตอร์ควบคุมกระบวนการแบบดิจิตอลโดยตรง

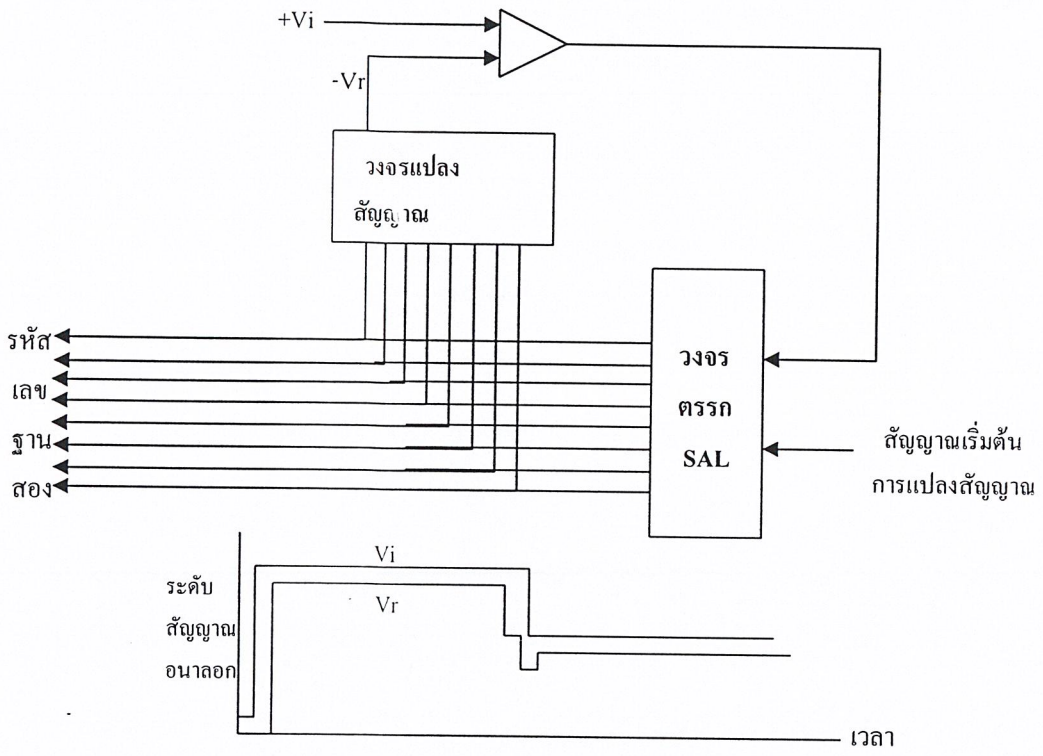
2.5.1 อุปกรณ์แปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล

ปัจจุบัน อุปกรณ์แปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลมีหลายแบบ โดยอุปกรณ์แต่ละแบบจะมีโครงสร้าง วิธีการแปลงสัญญาณ ความคลาดเคลื่อนในการแปลงสัญญาณ ความเร็วในการแปลงสัญญาณ ราคา และคุณสมบัติอื่นๆแตกต่างกัน ให้สามารถเลือกใช้ได้ตามความเหมาะสม อุปกรณ์แปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอล ที่นิยมใช้ในปัจจุบันคือ

1. อุปกรณ์แปลงสัญญาณชนิดวงจรนับ (Counter)
2. อุปกรณ์แปลงสัญญาณชนิดวงจรนับขึ้นและนับลง
3. อุปกรณ์แปลงสัญญาณชนิดประมาณค่าเปรียบเทียบ (Successive Approximation)
4. อุปกรณ์แปลงสัญญาณชนิดขนาน (Parallel)

อุปกรณ์แปลงสัญญาณชนิดประมาณค่าเปรียบเทียบ (Successive Approximation)

อุปกรณ์แปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลชนิดประมาณค่าเปรียบเทียบใช้วงจรตรรกประมาณค่าเปรียบเทียบ (SAL ย่อมาจาก Successive Approximation Logic) เปลี่ยนแปลงค่าของรหัสตัวเลขฐานสองครั้งละ 1 บิต โดยเริ่มจากบิตที่มีค่านัยสำคัญสูงสุด (Most Significant Bit) ก่อน เมื่อเริ่มต้นการแปลงสัญญาณบิตที่มีค่านัยสำคัญสูงสุดจะมีสภาวะ “1” และบิตอื่นๆมีสภาวะ “0” สัญญาณอนาลอก V_r จะมีค่าเท่ากับครึ่งหนึ่งของช่วงการเปลี่ยนแปลงสัญญาณของอุปกรณ์แปลงสัญญาณ ถ้าสัญญาณอนาลอก V_r มีค่ามากกว่า V_i บิตที่มีค่านัยสำคัญสูงสุดจะเปลี่ยนสภาวะเป็น “0” และวงจรตรรกประมาณค่าเปรียบเทียบจะเปลี่ยนแปลงสภาวะของรหัสเลขฐานสองบิตอื่นต่อไป จนถึงบิตสุดท้ายที่มีค่านัยสำคัญต่ำสุด (Least Significant Bit) อุปกรณ์แปลงสัญญาณชนิดค่าเปรียบเทียบทั่วไปสามารถแปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลได้จำนวนตั้งแต่ 8 ถึง 10 บิต อัตราการแปลงสัญญาณประมาณ 100,000 ครั้ง/วินาที



รูปที่ 2.11 อุปกรณ์แปลงสัญญาณอนาลอกเป็นสัญญาณดิจิตอลชนิดประมาณค่าเปรียบเทียบ

2.5.2 อุปกรณ์แปลงสัญญาณดิจิตอลเป็นสัญญาณอนาลอก

อุปกรณ์แปลงสัญญาณดิจิตอลเป็นสัญญาณอนาลอก ทำหน้าที่แปลงข้อมูลรหัสเลขฐานสอง b_{n-1}, $b_{n-2},\ldots b_1,b_0$ ให้เป็นศักย์หรือกระแสไฟฟ้า สามารถแสดงโดยสมการ

$$V_0 = V_r\left(b_{n-1} + \frac{b_{n-2}}{2} + \ldots + \frac{b_1}{2^{n-1}} + \frac{b_0}{2^n}\right) \qquad \ldots\ldots\ldots(2.3)$$

V_r คือศักย์ไฟฟ้าอ้างอิงสำหรับกำหนดค่าสูงสุดของสัญญาณอนาลอก b_{n-1} บิตที่มีนัยสำคัญสูงสุด และ b_0 คือบิตที่มีนัยสำคัญต่ำสุด รูปที่ 2.12 แสดงตัวอย่างการแปลงสัญญาณของอุปกรณ์แปลงสัญญาณดิจิตอลเป็นสัญญาณอนาลอกจำนวน 4 บิต ปัจจุบันอุปกรณ์แปลงสัญญาณดิจิตอลเป็นสัญญาณอนาลอก นิยมใช้ 2 แบบ คือ

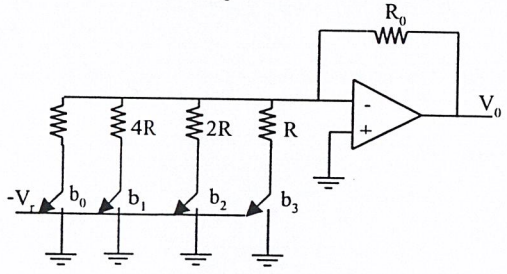
1. อุปกรณ์แปลงสัญญาณชนิดใช้ความต้านทานกำหนดน้ำหนักอัตราขยาย
2. อุปกรณ์แปลงสัญญาณชนิดวงจร R-2R

2.5.2.1 อุปกรณ์แปลงสัญญาณชนิดใช้ความต้านทานกำหนดน้ำหนักอัตราขยาย

อุปกรณ์แปลงสัญญาณดิจิตอลเป็นสัญญาณอนาลอกประกอบด้วยวงจรขยายและความต้านทานไฟฟ้าทางด้านสัญญาณเข้าของวงจรขยาย สำหรับกำหนดอัตราขยายของสัญญาณดิจิตอลแต่ละบิตให้เปลี่ยนแปลงตามสมการ 2.4 เช่น ตัวอย่างอุปกรณ์แปลงสัญญาณจำนวน 4 บิต ดังแสดงในรูปที่ 2.12 ถ้าบิตข้อมูลสภาวะ "1" สวิตซ์ จะต่อกับศักย์ไฟฟ้า V_r และบิตข้อมูลสภาวะ"0" สวิตซ์จะต่อกับศักย์ไฟฟ้า 0 สัญญาณออกของวงจรขยายสามารถคำนวณจากสมการ

$$V_0 = \frac{R_0}{R}\left(b_3 + \frac{b_2}{2} + \frac{b_1}{4} + \frac{b_0}{8}\right)V_r \qquad \text{.....(2.4)}$$

อุปกรณ์แปลงสัญญาณชนิดใช้ความต้านทานกำหนดน้ำหนักอัตราขยาย ไม่ได้รับความนิยม เนื่องจากต้องใช้ความต้านทานไฟฟ้าหลายค่ากำหนดน้ำหนักอัตราขยายของวงจรขยายให้สัญญาณดิจิตอลแต่ละบิต ถ้าอุปกรณ์แปลงสัญญาณขนาดใหญ่ต้องการแปลงข้อมูลจำนวนหลายบิต ค่าความต้านทานไฟฟ้าจะมีความคลาดเคลื่อนและไม่ถูกต้อง ทำให้มีความคลาดเคลื่อนในการแปลงสัญญาณเกิดขึ้น

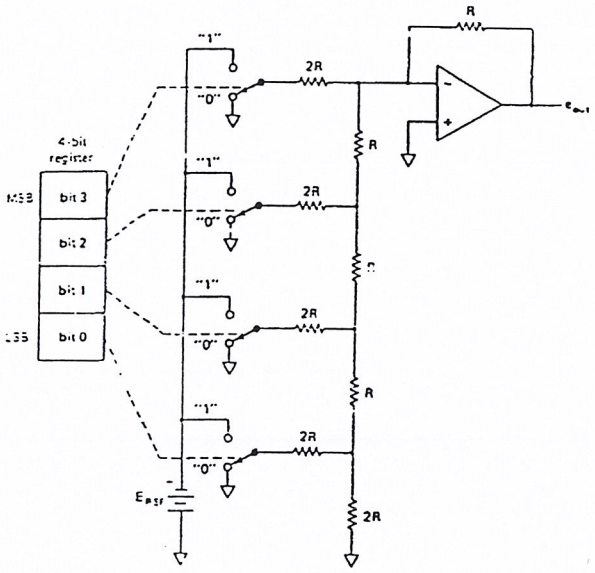


รูปที่ 2.12 อุปกรณ์แปลงสัญญาณชนิดใช้ความต้านทานกำหนดน้ำหนักอัตราขยายจำนวน 3 บิต

2.5.2.2 อุปกรณ์แปลงสัญญาณชนิดวงจร R-2R

อุปกรณ์แปลงสัญญาณดิจิตอลเป็นสัญญาณอนาลอกชนิดวงจร R-2R จะพยายามแก้ไขความคลาดเคลื่อนในการแปลงสัญญาณ ของอุปกรณ์แปลงสัญญาณชนิดใช้ความต้านทานกำหนดน้ำหนักอัตราขยายเพราะต้องการใช้ความต้านทานหลายค่า โดยการลดจำนวนค่าความต้านทานที่ใช้ในวงจร

แปลงสัญญาณให้เหลือเพียง 3 ค่า คือ R, 2R และ 3R เช่นตัวอย่างวงจรแปลงสัญญาณชนิดวงจร R-2R จำนวน 4 บิต ดังรูป



รูปที่ 2.13 อุปกรณ์แปลงสัญญาณชนิดวงจร R-2R จำนวน 4 บิต

สัญญาณออกของวงจรขยายรวม เมื่อบิต b_3,b_2,b_1 และ b_0 มีสภาวะ "1" สามารถหาค่าได้ตามสมการ

$$V_O = \frac{1}{2}\left(b_3 + \frac{b_2}{2} + \frac{b_1}{4} + \frac{b_0}{8}\right)V_r \qquad \text{.....(2.5)}$$

ทำนองเดียวกัน วงจรแปลงสัญญาณจำนวน n บิต สามารถหาค่าสัญญาณออกได้ตามสมการ

$$V_O = \frac{R_0}{R}\left(b_{n-1} + \frac{1}{2}b_{n-2} + \ldots + \frac{1}{2^{n-1}}b_0\right)V_r \qquad \text{.....(2.6)}$$

2.6 ตัวควบคุมแบบป้อนกลับ

2.6.1 ประวัติความเป็นมาของตัวควบคุมแบบป้อนกลับ

การควบคุมป้อนกลับแบบอัตโนมัติมักถูกเข้าใจว่าเป็นวิทยาการสมัยใหม่ แต่ในความเป็นจริงแล้ว ระบบควบคุมแบบป้อนกลับถูกคิดค้นโดยชาวกรีก เมื่อ 250 ปีก่อนคริสตกาล เพื่อนำมาใช้ในการควบคุมระดับน้ำ โดยมีหลักการทำงานเช่นเดียวกับตัวควบคุมระดับน้ำ (Level Regulator)ในสุขภัณฑ์ที่ทศวรรษที่ 30 (ค.ศ. 1930-1939) การควบคุมแบบป้อนกลับมีบทบาทสำคัญในการพัฒนาตัวขยายเชิง

ดำเนินการ(Operation Amplifier) ที่มีอัตราขยายสูง ซึ่งต่อมาได้ถูกนำมาใช้งานอย่างกว้างขวางในอุปกรณ์อิเล็กทรอนิกส์ และตลอดทศวรรษที่ 30 นี้ ตัวควบคุม PID เริ่มมีการวางจำหน่ายในท้องตลาด และบทความทางทฤษฎีเกี่ยวกับการควบคุมระบบได้ถูกตีพิมพ์ครั้งแรกในช่วงทศวรรษนี้ ในช่วงทศวรรษที่ 40 (ค.ศ.1940-1949) ตัวควบคุม PID แบบนิวแมติกส์ก็เป็นที่ยอมรับในงานอุตสาหกรรมอย่างแพร่หลาย และคู่มือวงจรอิเล็กทรอนิกส์ของตัวควบคุมชนิดนี้ได้เข้าสู่ท้องตลาดในช่วงทศวรรษที่ 50 (ค.ศ.1950-1959) ในปลายทศวรรษนี้คอมพิวเตอร์ควบคุมกระบวนการในงานอุตสาหกรรมได้ถูกนำมาใช้ครั้งแรก และตลอด 20 ปีที่ผ่านมานี้ฮาร์ดแวร์ของดิจิตอลคอมพิวเตอร์ก็ได้ถูกนำมาใช้เป็นอุปกรณ์สำคัญในการทำงานของตัวควบคุมกระบวนการด้วยเช่นกัน

2.6.2 ตัวควบคุมแบบเปิด-ปิด (On-Off controller)

ตัวควบคุมแบบเปิด-ปิดเป็นตัวควบคุมที่มีรูปแบบการทำงานง่ายที่สุดและมีราคาถูก นิยมใช้ในการควบคุมกระบวนการที่ไม่ต้องการความเที่ยงตรงสูงนัก และผลของความคลาดเคลื่อนไม่มีผลต่อการควบคุม เช่นการควบคุมอุณหภูมิของตู้เย็นที่ใช้ในบ้านหรือใช้ในงานอุตสาหกรรมที่มีการทำงานไม่ซับซ้อน การควบคุมระดับน้ำบางประเภทและใช้ในการควบคุมระบบความร้อน แต่ข้อเสียคือการควบคุมโดยใช้ตัวควบคุมชนิดนี้อาจมีผลต่อความต่อเนื่องของผลตอบสนองของกระบวนการได้

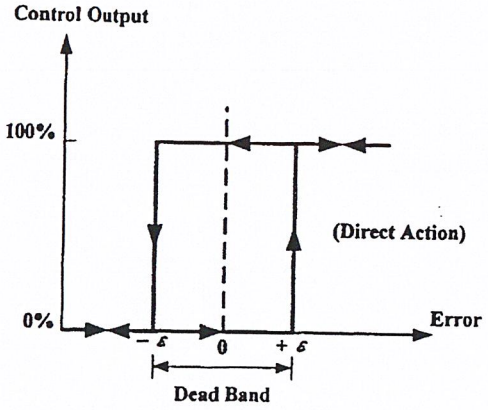
การควบคุมแบบเปิด-ปิดในทางทฤษฎี เอาท์พุทของตัวควบคุมจะเป็นไปได้เพียง 2 ค่าคือ

$$m(t)=\begin{cases} m_{max}, e>0 \\ m_{min}, e\le 0 \end{cases} \qquad \text{.....(2-7)}$$

เมื่อ e คือสัญญาณผิดพลาด (error)

m_{max} และ m_{min} คือค่าในการควบคุมสูงสุดและต่ำสุดตามลำดับ เช่นในตัวควบคุมนิวแมติกส์จะกำหนดให้ m_{max} = 15 psi และ m_{min} = 3 psi หรือ m_{max} = 20 mA และ m_{min} = 4 mA ในอุปกรณ์อิเล็กทรอนิกส์

จากสมการที่ (2-7) จะพบว่าการควบคุมแบบเปิด-ปิด เป็นการควบคุมแบบ 2 ตำแหน่ง หรือแบงแบงคอนโทรล (Bang-Bang Control) หรืออาจพิจารณาว่าเป็นการควบคุมแบบ P ชนิดพิเศษที่มีอัตราการขยายสูงมากก็ได้

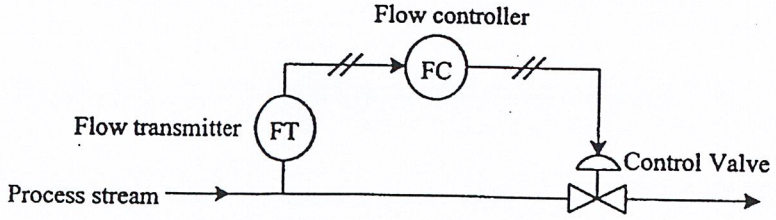


รูปที่ 2.14 การควบคุมแบบเปิด-ปิด

ตัวควบคุมแบบเปิด-ปิด สามารถปรับปรุงให้รวมค่าเดดแบนด์(Dead band) เป็นสัญญาณความผิดพลาดได้ เพื่อลดความไวต่อสัญญาณรบกวนในเครื่องมือวัดหรือเพื่อป้องกันไม่ให้ทำงานบ่อยครั้งจนเกินไป จากรูปที่ 2.14 หากค่าความผิดพลาดเพิ่มขึ้นเกินค่าวิกฤต (Critical Value) หรือ $+\varepsilon$ ค่าเอาท์พุทของตัวควบคุมจึงจะเปลี่ยนจาก m_{min} = 0 % เป็น m_{max} = 100 % เพื่อทำให้ค่าความคลาดเคลื่อนลดต่ำกว่าหรือ $-\varepsilon$ และเมื่อถึงจุดนี้ค่าเอาท์พุทจะเปลี่ยนจาก m_{max} = 100 % เป็น m_{min} = 0 %

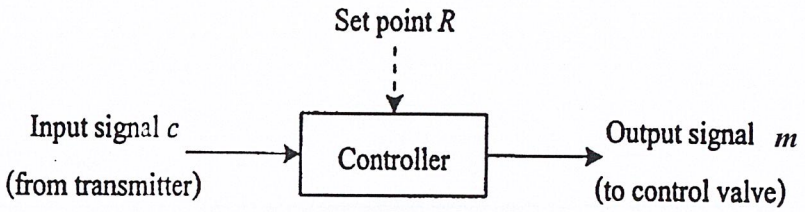
2.6.3 ตัวควบคุมแบบ PID (PID Controller)

การควบคุมป้อนกลับมีพื้นฐานอยู่ 3 แบบที่ใช้งานกันทั่วไปคือ การควบคุมแบบ Proportional (P), การควบคุมแบบ Integral (I) และการควบคุมแบบ Derivative (D) จากรูปที่ 2.15 เมื่อวัดและแปลงอัตราการไหลของไอน้ำเป็นสัญญาณนิวแมติกส์ให้กับตัวควบคุมการไหลแล้ว ตัวควบคุมจะเปรียบเทียบค่าที่วัดได้กับค่าเป้าหมาย (set point) หรือ SP แล้วจึงส่งสัญญาณควบคุมที่เหมาะสมแก่วาล์วควบคุมสัญญาณนิวแมติกส์ให้สัญลักษณ์พื้นฐานคือเส้นขนานตัดเส้นตรง



รูปที่ 2.15 ระบบควบคุมการไหล

ในรูปที่ 2.16 แสดงบล็อกไดอะแกรมของตัวควบคุมแบบป้อนกลับ ค่าเป้าหมายแสดงโดยใช้เส้นประซึ่งถูกกำหนดโดยการป้อนค่าหรือการกำหนดตำแหน่งที่ตัวควบคุมซึ่งเรียกว่า Local set point ตัวควบคุมบางตัวสามารถตั้งค่าเป้าหมายระยะไกลหรือ Remote set point ได้โดยยินยอมให้ตัวควบคุมรับสัญญาณค่าเป้าหมายจากอุปกรณ์ภายนอก เช่นตัวควบคุมอีกตัวหนึ่งหรือเครื่องดิจิตอลคอมพิวเตอร์ อินพุทและเอาท์พุทของตัวควบคุมเป็นสัญญาณแบบต่อเนื่อง ซึ่งเป็นได้ทั้งสัญญาณนิวแมติกส์และสัญญาณไฟฟ้า



รูปที่ 2.16 บล็อกไดอะแกรมของตัวควบคุมแบบป้อนกลับ

2.6.4 การควบคุมแบบป้อนกลับ

วัตถุประสงค์ของการควบคุมแบบป้อนกลับคือการลดสัญญาณผิดพลาด (Error) หรือ $e(t)$ ให้มีค่าเท่ากับศูนย์ ดังสมการ

$$e(t)=R(t)-c(t) \qquad \text{.....(2-8)}$$

เมื่อ $R(t)$ = ค่าเป้าหมาย

$c(t)$ = ผลตอบสนองของกระบวนการหรือค่าสัญญาณจากทรานสมิตเตอร์ (Transmitter)

จากสมการที่ (2-8) จะพบว่าค่าเป้าหมายสามารถเปลี่ยนแปลงตามกาลเวลาได้

2.6.4.1 การควบคุมแบบ Proportional (P) เอาท์พุทของตัวควบคุมจะแปรผันตามสัญญาณผิดพลาดที่เกิดขึ้นดังสมการ

$$m(t)=\overline{m}+K_p e(t) \qquad \text{.....(2-9)}$$

เมื่อ $m(t)$ = ค่าเอาท์พุทของตัวควบคุม

$\overline{m}$ = ค่า Bias

K_p = อัตราขยายของตัวควบคุม โดยปกติจะไม่มีหน่วย

เงื่อนไขของการควบคุมแบบ P

1. อัตราขยายของตัวควบคุม : K_p สามารถปรับค่าได้ เพื่อให้เอาท์พุทของตัวควบคุมสามารถเปลี่ยนแปลงค่าได้ไวตามความต้องการและทำให้ผลตอบสนองของกระบวนการเข้าสู่เป้าหมาย

2. การเลือกค่า K_p สามารถทำให้เอาท์พุทของตัวควบคุมมีค่าเพิ่มขึ้นหรือลดลงได้ ค่า $\overline{m}$ สามารถปรับได้โดยที่เอาท์พุทของตัวควบคุมจะมีค่าเท่ากัน ก็ต่อเมื่อค่าผิดพลาดเท่ากับศูนย์ ซึ่งในสภาวะนี้อาจกล่าวได้ว่าค่าเอาท์พุทของตัวควบคุมรวมไปถึงผลตอบสนองของกระบวนการอยู่ในสภาวะคงที่ (Steady-State) สำหรับตัวควบคุมที่ใช้งานโดยทั่วไปค่า K_p จะไม่มีหน่วยเพราะค่า $m(t)$ และ $e(t)$ มีหน่วยเดียวกัน เช่นในกรณีของอุปกรณ์ไฟฟ้าหรือทางนิวแมติกส์ หรืออาจอยู่ในรูปของตัวเลข 0-100% เพื่อความสะดวกในการทำงานของตัวแสดงกราฟฟิกและซอฟท์แวร์ควบคุมคอมพิวเตอร์ ในอีกกรณีหนึ่งค่าสัญญาณผิดพลาดจะอยู่ในรูปของหน่วยทางวิศวกรรม เช่นหน่วยของอุณหภูมิเป็นองศาเซลเซียส ($^{\circ}$C) หรือหน่วยของความหนาแน่นเป็น mol/L ซึ่งในกรณีนี้ K_p และอัตราขยายที่สภาวะคงที่ (Steady – State Gain) ของอุปกรณ์ในวงรอบควบคุมตัวอื่นๆ เช่นทรานสมิตเตอร์หรือวาล์วควบคุมก็จะมีหน่วยเช่นกัน

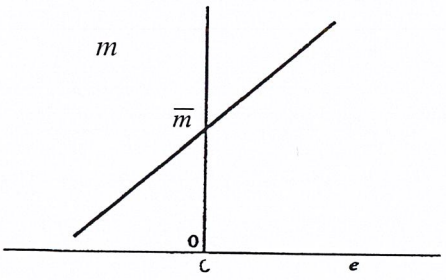
ตัวควบคุมแบบดั้งเดิมบางตัวจะใช้ค่า *Proportional Band* (*PB*) แทนการใช้ค่า K_p ซึ่ง *PB* จะมีหน่วยเป็น % และจะใช้กรณีที่ค่า K_p ไม่มีหน่วยเท่านั้น

$$PB = \frac{100\%}{K_p} \qquad \text{.....(2-10)}$$

ตัวควบคุมแบบ P ในทางอุดมคติจะไม่พิจารณาถึงข้อจำกัดทางกายภาพของเอาท์พุทตัวควบคุม ซึ่งปกติแล้วตัวควบคุมจะเกิดการอิ่มตัวเมื่อถึงขอบเขตทางกายภาพคือ m_{max} และ m_{min}

ในการพิจารณาฟังก์ชันถ่ายโอนของตัวควบคุมแบบ P จะกำหนดให้

$$m'(t) = m(t) - \overline{m} \qquad \text{.....(2-11)}$$



รูปที่ 2.17 การควบคุมแบบ P ในทางอุดมคติ (ความชันของเส้นตรง)

ดังนั้น

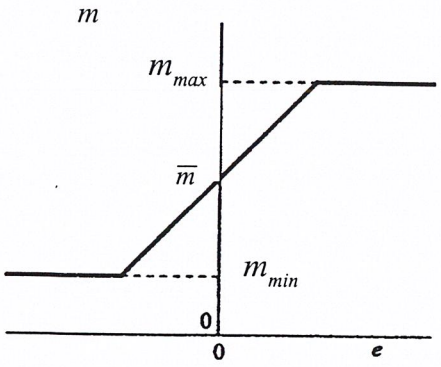
$$m'(t)=K_p e(t) \qquad \text{.....(2-12)}$$

สมการลาปลาซซึ่งเป็นฟังก์ชันถ่ายโอนของตัวควบคุมคือ

$$\frac{M'(s)}{E(s)}=K_p \qquad \text{.....(2-13)}$$

ข้อเสีย ซึ่งเป็นลักษณะเฉพาะของการควบคุมแบบ P คือไม่สามารถกำจัด Steady – State error ที่เกิดขึ้นหลังจากการเปลี่ยนแปลงค่าเป้าหมายหรือการได้รับสัญญาณรบกวนจากโหลด ซึ่งจะเกิดขึ้นกับการควบคุมแบบ P ในทุกๆค่า K_p ในทางทฤษฎีค่าออฟเซตสามารถกำจัดได้โดยการตั้งค่าเป้าหมายหรือค่า $\overline{m}$ ขึ้นใหม่หลังจากเกิดออฟเซต แต่อย่างไรก็ตามวิธีการนี้ไม่ได้เป็นวิธีการที่ดีที่สุดเนื่องจากต้องอาศัยผู้มีความชำนาญเพียงพอและการหาค่าเป้าหมายหรือค่า $\overline{m}$ ใหม่ก็ต้องใช้การลองผิดลองถูกซึ่งอาจเกิดความผิดพลาดขึ้นได้ การแก้ไขปัญหานี้ในทางปฏิบัติจะใช้การทำงานของการควบคุมแบบ Integral (I) เข้ามาช่วยซึ่งจะได้ผลตอบสนองที่ดีขึ้น เนื่องจากการควบคุมแบบ I จะมีการเปลี่ยนแปลงค่าการควบคุมเพื่อลดออฟเซตได้โดยอัตโนมัติ

ในการควบคุมกระบวนการบางอย่างที่ไม่ต้องคำนึงถึงการเกิดค่าออฟเซต การควบคุมแบบ P ก็มีความเหมาะสมในการใช้งาน เนื่องจากมีรูปแบบในการทำงานที่ง่ายและไม่ซับซ้อน เช่นการการควบคุมระดับของเหลวที่ไม่จำเป็นต้องรักษาระดับให้อยู่ในค่าเป้าหมาย แต่ต้องการเพียงควบคุมไม่ให้ของเหลวล้นถังหรือหมดถังเท่านั้นเป็นต้น



รูปที่ 2.18 การควบคุมแบบ P ในทางปฏิบัติ

2.6.4.2 การควบคุมแบบ Integral (I) และการควบคุมแบบ PI

การควบคุมแบบ I หมายถึงการควบคุมแบบรีเซตหรือแบบลอย (Floating) โดยที่ค่าเอาท์พุทของตัวควบคุมจะขึ้นอยู่กับการอินทิเกรตสัญญาณผิดพลาด ดังสมการ

$$m(t)=\overline{m}+\frac{1}{T_i}\int_0^t e(t)dt \qquad \text{.....(2-14)}$$

เมื่อ T_i คือ เวลาอินทิเกรตหรือเวลารีเซตซึ่งมีหน่วยเป็นหน่วยของเวลาและสามารถปรับค่าได้ตามความเหมาะสม

การควบคุมแบบ I เป็นที่นิยมใช้กันอย่างกว้างขวางเพราะสามารถใช้งานได้ดีและสามารถกำจัดออฟเซตได้อย่างมีประสิทธิภาพ จากสมการที่ (2-14) เมื่อกระบวนการเข้าสู่สภาวะคงที่สัญญาณผิดพลาดและเอาท์พุทของตัวควบคุมจะมีค่าคงที่ หรืออาจกล่าวได้ว่าค่า $m(t)$ จะเปลี่ยนแปลงตามเวลาจนกระทั่ง $e(t)=0$ ดังนั้นเมื่อใช้การควบคุมแบบ I ค่า $m(t)$ จะให้ค่าในการควบคุมซึ่งจะทำให้สัญญาณผิดพลาดในสภาวะคงที่มีค่าเท่ากับศูนย์ ซึ่งสัญญาณผิดพลาดนี้จะเกิดขึ้นเมื่อมีการเปลี่ยนแปลงค่าเป้าหมาย หรือได้รับสัญญาณรบกวนจากโหลด และการกำจัดสัญญาณผิดพลาดให้เท่ากับศูนย์จะกระทำได้ก็ต่อเมื่อเอาท์พุทของตัวควบคุม หรืออุปกรณ์ควบคุมตัวสุดท้ายยังไม่ถึงขอบเขตการอิ่มตัว แต่หากเกินช่วงขอบเขตนี้ไปแล้วค่าของกระบวนการจะไม่สามารถกลับสู่ค่าเป้าหมายได้ ซึ่งการอิ่มตัวเกิดจากสัญญาณรบกวนหรือการเปลี่ยนแปลงค่าเป้าหมายเกินช่วงของค่า $m(t)$

วัตถุประสงค์หลักของการควบคุมแบบ I คือ การกำจัดออฟเซต (Offset) แต่ตัวควบคุมแบบ I ไม่นิยมใช้งานเพียงตัวเดียว เพราะจะให้ผลในการควบคุมน้อยมาก จนกว่าจะเกิดสัญญาณผิดพลาดขึ้นมาอย่างต่อเนื่องในบางเวลาเท่านั้น ดังนั้นจึงมักนำการควบคุมแบบ I มาทำงานร่วมกับการควบคุมแบบ P

หรือที่เรียกว่าตัวควบคุมแบบ PI เพราะการควบคุมแบบ P จะให้ผลในการควบคุมแบบทันทีทันใดในขณะตรวจพบสัญญาณผิดพลาด

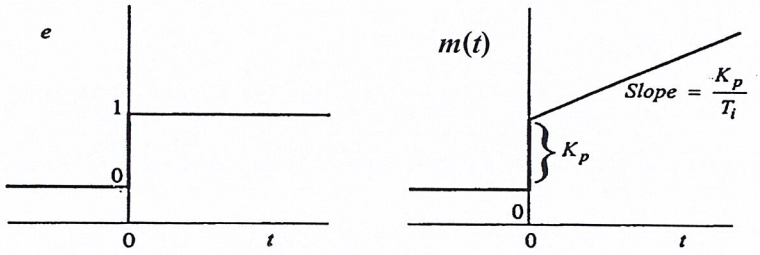
สมการของตัวควบคุมแบบ PI คือ

$$m(t)=\overline{m}+K_p\left[e(t)+\frac{1}{T_i}\int_0^t e(t)dt\right] \qquad \text{.....(2-15)}$$

ฟังก์ชันถ่ายโอนของตัวควบคุมแบบ PI ในรูปของสมการลาปลาซคือ

$$\frac{M'(s)}{E(s)}=K_p\left(1+\frac{1}{T_i s}\right)=K_p\left(\frac{T_i s+1}{T_i s}\right) \qquad \text{.....(2-16)}$$

ในรูปที่ 2.19 แสดงผลตอบสนองของตัวควบคุม PI ต่อการเปลี่ยนแปลงแบบขั้นบันไดของ $e(t)$ ที่เวลา t = 0 เอาท์พุทของตัวควบคุมจะเปลี่ยนแปลงพร้อมกับค่าผิดพลาด โดยเกิดจากการควบคุมแบบ P การควบคุมแบบ I จะทำให้ $m(t)$ เพิ่มขึ้นในลักษณะลาดชัน (Ramp) เมื่อ $t > 0$ ที่ $t=T_i$ เทอม I จึงให้ค่าในการควบคุมเท่ากับเทอม P



รูปที่ 2.19 ผลตอบสนองของตัวควบคุมแบบ PI ต่อการเปลี่ยนแปลงแบบขั้นบันไดของ $e(t)$

ตัวควบคุมบางตัวจะใช้เทอม $\frac{1}{T_i}$ (ครั้ง/นาที) แทนการใช้ T_i (นาที) เช่นถ้า $T_i=0.2$ นาที แล้ว $\frac{1}{T_i}=5$ ครั้ง/นาที เป็นต้น

<u>ข้อเสีย</u> ของการควบคุมแบบ I คือระบบจะมีการตอบสนองแบบแกว่งทำให้ระบบขาดเสถียรภาพ แต่ในบางครั้งก็สามารถยอมรับได้ หากการแกว่งเกิดขึ้นพร้อมกับการตอบสนองของกระบวนการที่รวดเร็ว แต่ข้อเสียของการควบคุมแบบ I ก็สามารถหลีกเลี่ยงได้โดยการปรับใช้ค่าพารามิเตอร์ให้เหมาะสมหรือเพิ่มการควบคุมแบบ D ซึ่งจะสามารถลดผลตอบสนองที่ไม่มีเสถียรภาพลงได้

2.6.4.3 การควบคุมแบบ Derivative (D) การควบคุมแบบ Proportional Plus derivative (PD) และการควบคุมแบบ Proportional plus integral plus derivative (PID)

การควบคุมแบบ D หมายถึงการควบคุมแบบอัตราส่วน (Rate action) หรือการทำนายล่วงหน้า (Pre-act) โดยจะทำนายพฤติกรรมของสัญญาณผิดพลาดที่จะเกิดขึ้นในอนาคต ซึ่งจะพิจารณาจากอัตราการเปลี่ยนแปลงของค่าสัญญาณผิดพลาด เช่นในกรณีที่เตาปฏิกรณ์มีอุณหภูมิเพิ่มขึ้น $10^{\circ}C$ ในเวลา 3 นาที ซึ่งสามารถสรุปได้ว่าเกิดขึ้นเร็วกว่าในกรณีที่อุณหภูมิเพิ่มขึ้น $10^{\circ}C$ ในเวลา 30 นาที และสามารถประเมินประสิทธิภาพในการแก้ไขสภาวะของการเกิดปฏิกิริยา ที่ทำให้อุณหภูมิเพิ่มสูงขึ้นได้ ถ้าการควบคุมเตาปฏิกรณ์ใช้ระบบปรับด้วยมือ (Manual) ผู้ควบคุมที่มีประสบการณ์สามารถประเมินผลตอบสนองที่จะเกิดขึ้น และแก้ไขสถานการณ์ได้อย่างเหมาะสมในเวลาอันรวดเร็วเพื่อทำการลดอุณหภูมิลง แต่การควบคุมด้วยระบบอัตโนมัติตัวควบคุมแบบ P จะตอบสนองต่อการเปลี่ยนแปลงของอุณหภูมิเท่านั้น แต่ไม่สามารถแยกแยะความแตกต่าง ของช่วงเวลาที่เกิดการเปลี่ยนแปลงได้ เช่นเดียวกับตัวควบคุมแบบ I ซึ่งโดยปกติแล้วจะสร้างสัญญาณควบคุมที่แปรผันตามระยะเวลาของสัญญาณรบกวนที่เกิดขึ้น ดังนั้นสัญญาณรบกวนขนาดเล็กที่เกิดขึ้นอย่างช้าๆก็สามารถทำให้ตัวควบคุมสร้างสัญญาณควบคุมที่มีค่ามากขึ้นมาได้

การทำงานของตัวควบคุมแบบ D ในทางอุดมคติมีสมการดังนี้

$$m(t)=\overline{m}+T_d\frac{de}{dt} \qquad \text{.....(2-17)}$$

เมื่อ T_d คือเวลา Derivative มีหน่วยเป็นหน่วยของเวลา

เอาท์พุทของตัวควบคุมมีค่าเท่ากับ $\overline{m}$ เมื่อสัญญาณผิดพลาดมีค่าคงที่หรือเมื่อ $\frac{de}{dt}=0$ การควบคุมแบบ D จะไม่ใช้งานเพียงตัวเดียวแต่จะใช้งานร่วมกับการควบคุมแบบ P หรือ PI

การควบคุมแบบ PD มีฟังก์ชันถ่ายโอนคือ

$$\frac{M'(s)}{E(s)}=K_p(1+T_ds) \qquad \text{.....(2-18)}$$

การควบคุมแบบ D ทำให้กระบวนการมีเสถียรภาพและลดแนวโน้มความไม่มีเสถียรภาพของการควบคุมแบบ I นอกจากนี้ยังนำมาใช้เพื่อปรับปรุงผลตอบสนองทางพลศาสตร์ โดยลดเวลาเข้าสู่สภาวะคงที่ (Setting time) ของกระบวนการลง ซึ่งทำให้กระบวนการเข้าสู่สภาวะคงที่เร็วขึ้น แต่ในกรณีที่เครื่องมือวัดในกระบวนการมีสัญญาณรบกวนความถี่สูงและมีการแกว่งไม่มีแบบแผน การควบคุม

แบบ D จะทำให้ผลตอบสนองของกระบวนการเปลี่ยนแปลงในช่วงกว้างและสัญญาณรบกวนจะถูกขยายแต่สามารถป้องกันได้โดยการกรองความถี่ โดยปกติแล้วการควบคุมแบบ D จะไม่นิยมใช้กับการควบคุมการไหลเนื่องจากวงรอบการควบคุมมีการตอบสนองที่รวดเร็วและเครื่องมือวัดอัตราการไหลมีสัญญาณรบกวนค่อนข้างสูง

การควบคุมแบบ D สามารถทำงานร่วมกับการควบคุมแบบ P และ I หรือที่เรียกว่าตัวควบคุม PID ซึ่งมีสมการคือ

$$m(t)=\overline{m}+K_p\left[e(t)+\frac{1}{T_i}\int_0^t e(t)dt+T_d\frac{de}{dt}\right] \qquad \text{.....(2-19)}$$

ซึ่งมีฟังก์ชันถ่ายโอนคือ

$$\frac{M'(s)}{E(s)}=K_p\left(1+\frac{1}{T_i s}+T_d s\right) \qquad \text{.....(2-20)}$$

แต่อย่างไรก็ตามอุปกรณ์อิเล็คทรอนิคส์และนิวแมติกส์ที่เป็นตัวควบคุมแบบ D ในทางอุดมคติไม่สามารถสร้างขึ้นมาได้จริง เนื่องมาจากข้อจำกัดทางกายภาพ จึงจำเป็นต้องมีการประมาณพฤติกรรมทางอุดมคติของตัวควบคุมในสมการที่ (2-20) ด้วยฟังก์ชันถ่ายโอน

$$\frac{M'(s)}{E(s)}=K_p\left(\frac{T_i s+1}{T_i s}\right)\left(\frac{T_d s+1}{aT_d+1}\right) \qquad \text{.....(2-21)}$$

เมื่อ α เป็นค่าระหว่าง 0.05-0.2

<u>ข้อเสีย</u> อีกประการหนึ่งของตัวควบคุม PID ทางอุดมคติคือเมื่อมีการเปลี่ยนแปลงค่าเป้าหมายอย่างรวดเร็วจะทำให้ค่าจาก D มีค่ามาก และทำให้เกิดค่าควบคุม D พุ่งสูง (Derivative kick) ในอุปกรณ์ควบคุมตัวสุดท้าย การหลีกเลี่ยงความเสียหายที่อาจเกิดขึ้นนั้นสามารถทำได้โดยเปลี่ยนแปลงค่าการควบคุมแบบ D โดยใช้ค่าสัญญาณจากเครื่องมือวัด แทนการใช้ค่าสัญญาณผิดพลาด โดยแทน $\frac{de}{dt}$ ด้วย $-\frac{dc}{dt}$ ในสมการที่ 2-19 ได้

$$m(t)=\overline{m}+K_p\left(e(t)+\frac{1}{T_i}\int_0^t e(t)dt-T_d\frac{dc}{dt}\right) \qquad \text{.....(2-22)}$$

การกำจัดค่าควบคุม D พุ่งสูงเป็นทฤษฎีมาตรฐานที่นิยมใช้กันในตัวควบคุมส่วนใหญ่แต่ในบางทฤษฎีจะกำจัดค่าเป้าหมายในเทอม P และเทอม D เพื่อป้องกันการเกิดค่าควบคุม P พุ่งสูง (Proportional kick) เมื่อมีการเปลี่ยนแปลงค่าเป้าหมาย แต่ทฤษฎีนี้ยังไม่เป็นที่ยอมรับกันอย่างแพร่หลาย

2.6.4.4 สมการ PID สำหรับการคำนวณด้วยคอมพิวเตอร์

จากสมการที่กล่าวมาข้างต้น คอมพิวเตอร์ไม่สามารถนำสมการนี้ไปใช้ในการประมวลผลได้โดยตรงจึงต้องทำการแปลงเป็น สมการดิฟเฟอเรนเชียลต่อเนื่องแบบย่อย (Discrete Differential Equation) เพื่อให้คอมพิวเตอร์สามารถทำการประมวลผลได้ดังนี้

สมการ PID พื้นฐานคือ

$$m(t)=v_0(t)=K_p\left(e(t)+\frac{1}{T}\int_0^t e(t)dt+T_d\frac{de(t)}{dt}\right) \qquad \text{.....(2-23)}$$

จากสมการที่ (2-20) ได้สมการอนุพันธ์อันดับสองคือ

$$\frac{dV_0(t)}{dt}=K_p\left[\frac{de(t)}{dt}+\frac{1}{T_i}\frac{d}{dt}\int_0^t e(t)dt+T_d\frac{d}{dt}\left(\frac{de(t)}{dt}\right)\right]$$

$$=K_p\left[\frac{de(t)}{dt}+\frac{1}{T_i}e(t)+T_d\frac{d^2e(t)}{dt^2}\right]$$

กำหนดให้

$$e(t)=e_n$$

$$\frac{dV_o(t)}{dt}=\frac{\Delta V_o}{\Delta t}=\frac{V_{o_n}-V_{o_{n-1}}}{\Delta t} \qquad \text{.....(2-24)}$$

$$\frac{de(t)}{dt}=\frac{\Delta e}{\Delta t}=\frac{e_n-e_{n-1}}{\Delta t}$$

ดังนั้น

$$\frac{V_{o_n}-V_{o_{n-1}}}{\Delta t}=K_p\left[\frac{e_n-e_{n-1}}{\Delta t}+\frac{1}{T_i}e_n+T_d\frac{d}{dt}\left(\frac{e_n-e_{n-1}}{\Delta t}\right)\right]$$

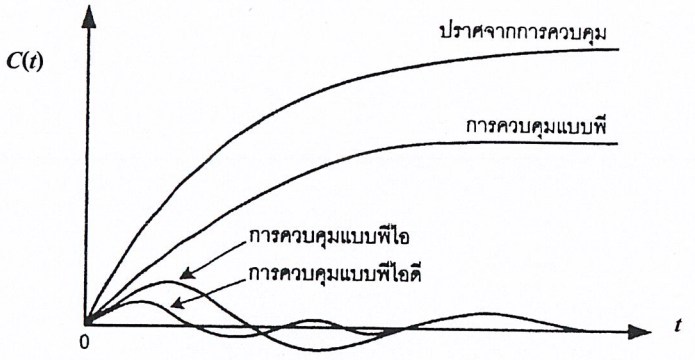
$$=K_p\left[\frac{e_n-e_{n-1}}{\Delta t}+\frac{1}{T_i}e_n+T_d\left(\frac{e_n-e_{n-1}-(e_{n-1}-e_{n-2})}{\Delta t^2}\right)\right] \qquad \text{.....(2-25)}$$

$$V_{o_n} - V_{o_{n-1}} = K_p\left[\left(e_n - e_{n-1}\right) + \frac{1}{T_i}e_n\Delta t + T_d\frac{\left(e_n - 2e_{n-1} + e_{n-2}\right)}{\Delta t}\right]$$

$$V_{o_n} = V_{o_{n-1}} + K_p\left[\left(e_n - e_{n-1}\right) + \frac{\Delta t}{T_i}e_n + \frac{T_d}{\Delta t}\left(e_n - 2e_{n-1} + e_{n-2}\right)\right]$$

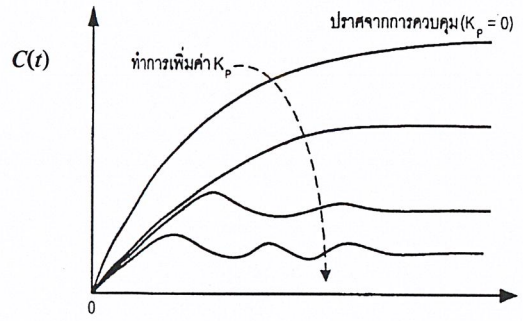
เมื่อ $V_o(t)$ คือสัญญาณเอาท์พุทที่ส่งออกไปควบคุมคอนโทรลวาล์ว

2.6.4.5 ผลตอบสนองของกระบวนการต่อการควบคุมแบบป้อนกลับแต่ละชนิด



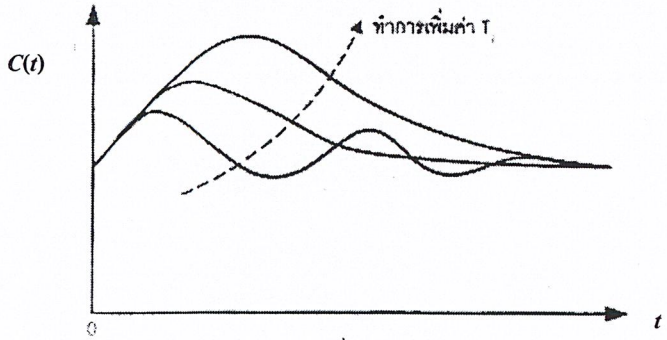
รูปที่ 2.20 ผลตอบสนองของกระบวนการต่อการควบคุมแบบวงเปิด

ในรูปที่ 2.20 แสดงผลตอบสนองของกระบวนการหลังจากมีการเปลี่ยนแปลงค่าเป้าหมายแบบขั้นบันได ผลตอบสนองของกระบวนการแทนด้วย $C(t)$ ซึ่งแสดงการเปลี่ยนแปลงค่าเป้าหมายเริ่มต้น หากไม่มีการควบคุมแบบป้อนกลับกระบวนการจะต้องใช้เวลานานมากในการเข้าสู่สภาวะคงที่ การควบคุมแบบ P จะเร่งการตอบสนองของกระบวนการและลดค่าออฟเซต การควบคุมแบบ PI จะกำจัดค่าออฟเซตแต่ทำให้ผลตอบสนองเกิดการแกว่ง ส่วนการควบคุมแบบ PID จะลดองศาการแกว่งและลดเวลาในการตอบสนองลง จึงสามารถสรุปได้ว่าการใช้ตัวควบคุม P , PI , และ PID จะไม่มีผลตอบสนองของกระบวนการที่มีการแกว่งเสมอไป แต่จะขึ้นอยู่กับการเลือกค่าพารามิเตอร์ของตัวควบคุม K_p, T_i, T_d และส่วนประกอบอื่นๆของกระบวนการ ซึ่งอยู่ในรูปที่ 2.20 แสดงตัวอย่างของผลตอบสนองที่เกิดขึ้นในทางปฏิบัติ

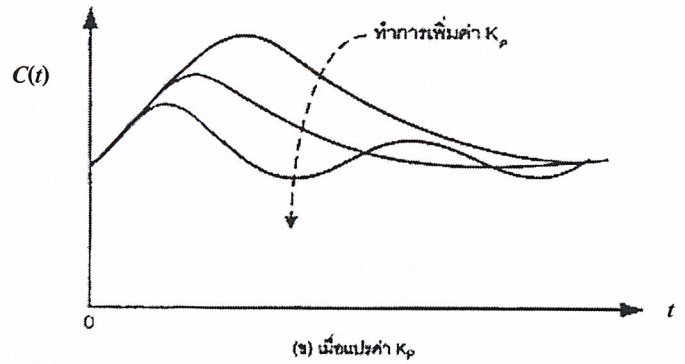


รูปที่ 2.21 ผลตอบสนองของกระบวนการต่อการควบคุมแบบพี เมื่อแปรค่า K_p

รูปที่ 2.21 - 2.23 แสดงผลกระทบเชิงคุณภาพเมื่อมีการเปลี่ยนแปลงค่าพารามิเตอร์ของตัวควบคุม ในรูปที่ 2.21 แสดงผลจากการเปลี่ยนแปลงอัตราขยาย K_p ของกระบวนการ ตามปกติแล้วการเพิ่มอัตราขยายของตัวควบคุมมีแนวโน้มที่จะทำให้การตอบสนองของกระบวนการเร็วขึ้น แต่อย่างไรก็ตามถ้า K_p มีค่ามากเกินไปก็จะทำให้ผลตอบสนองเกิดการแกว่งที่ไม่ต้องการหรืออาจทำให้ระบบขาดเสถียรภาพได้ ดังนั้นการตั้งค่า K_p ให้พอเหมาะจึงให้ผลตอบสนองที่ดี ทั้งในการควบคุมแบบ P และ PI รวมถึงการควบคุมแบบ PID ด้วย



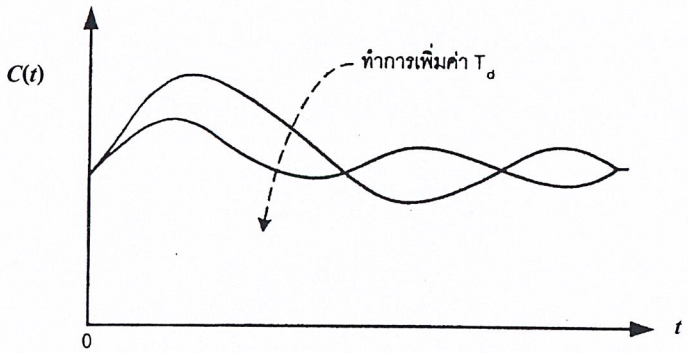
(ก) เมื่อแปรค่า T_i



รูปที่ 2.22 ผลตอบสนองของกระบวนการต่อการควบคุมแบบพีไอ

(ก) เมื่อเปลี่ยนแปลงค่า T_i , (ข) เมื่อแปรค่า K_p

ในรูปที่ 2.22 ก และ 2.22 ข การเพิ่มค่า T_i ทำให้การควบคุมแบบPI และ PID มีผลตอบสนองของกระบวนการที่ช้าลง ในทางทฤษฎีออฟเซตจะถูกกำจัดได้ในทุกค่าของ T_i ซึ่งมีค่าระหว่าง $0 - \infty$ แต่เมื่อมีการเปลี่ยนแปลงโหลดหรือค่าเป้าหมาย T_i ที่มีค่ามากๆจะทำให้การตอบสนองเข้าระบบสู่ค่าเป้าหมายได้ช้าลง



รูปที่ 2.23 ผลตอบสนองของกระบวนการต่อการควบคุมแบบพีไอดี เมื่อมีการเปลี่ยนแปลงค่า T_d

ผลของ T_d ที่มีต่อกระบวนการนั้นสรุปได้ยาก เพราะที่ T_d มีค่าน้อยการเพิ่มค่า T_d จะให้ผลตอบสนองที่ดีขึ้นโดยจะลดเวลาในการตอบสนอง, ลดการเบี่ยงแบนและลดองศาการแกว่ง ดังแสดงในรูปที่ 2.23 แต่อย่างไรก็ตามหาก T_d มีค่ามาก สัญญาณรบกวนของเครื่องมือวัดจะถูกขยายและผลตอบสนองจะเกิดการแกว่ง ดังนั้นจึงควรตั้งค่า T_d ให้เหมาะสมในการใช้งาน

บทที่ 3
การควบคุมและการสร้างตัวควบคุมแบบอัตโนมัติ

ในการวิเคราะห์ และออกแบบตัวควบคุมนั้น จำเป็นต้องศึกษาธรรมชาติ และคุณลักษณะของกระบวนการ รวมถึงอุปกรณ์ต่างๆ ที่ใช้ในการควบคุมกระบวนการทั้งหมด เพื่อใช้ศึกษา และวิเคราะห์ผลกระทบ ที่เกิดขึ้นต่อกระบวนการเมื่อถูกรบกวนจากการเปลี่ยนแปลงของสภาพแวดล้อมภายนอกหรือได้รับการควบคุมจากเครื่องควบคุมหรืออุปกรณ์ควบคุมอื่นๆ ทั้งนี้ก็เพื่อที่จะได้เลือกการควบคุมและค่าพารามิเตอร์ของตัวควบคุมที่เหมาะสม ในอันที่จะได้การควบคุมที่มีประสิทธิภาพมากที่สุด โดยทั่วไป การศึกษาธรรมชาติ และคุณลักษณะ หรือคุณสมบัติต่างๆของกระบวนการที่ใช้กันอยู่ มี 2 วิธี คือ

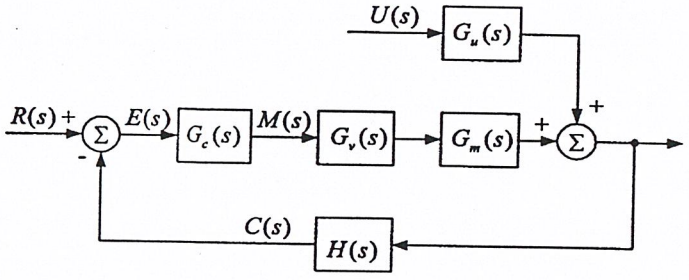
1. การศึกษาคุณลักษณะของกระบวนการ โดยใช้วิธีการทดสอบ วิธีนี้การวิเคราะห์กระบวนการสามารถทำได้โดยใช้เครื่องวัดและเครื่องควบคุม ทดลองป้อนสัญญาณอินพุท และสังเกตผลการเปลี่ยนแปลงของสัญญาณเอาท์พุทของกระบวนการ และอุปกรณ์อื่นๆ ซึ่งผลของสัญญาณเอาท์พุทนี้จะนำไปพยากรณ์ หรือประมาณค่าพารามิเตอร์ของกระบวนการ เพื่ออธิบายคุณลักษณะกระบวนการต่อไป

2. การศึกษาคุณลักษณะของกระบวนการ โดยใช้วิธีการวิเคราะห์ทางคณิตศาสตร์ วิธีนี้การวิเคราะห์คุณลักษณะของอุปกรณ์ และกระบวนการจะถูกแสดงในลักษณะของสมการคณิตศาสตร์ต่างๆ เช่น สมการพีชคณิต สมการอนุพันธ์ (Differential Equation) และสมการต่าง (Difference Equation) และวิเคราะห์คุณสมบัติของกระบวนการจากการหาคำตอบของสมการเหล่านี้

การศึกษาธรรมชาติและคุณลักษณะหรือคุณสมบัติของกระบวนการ โดยใช้วิธีการวิเคราะห์ทางคณิตศาสตร์ เป็นวิธีที่ยุ่งยาก และต้องอาศัยความรู้ทางคณิตศาสตร์ และฟิสิกส์ ซึ่งจำเป็นต้องอาศัยผู้มีความรู้ และความชำนาญ ดังนั้น ในทางปฏิบัติระบบควบคุมของกระบวนการทางอุตสาหกรรมโดยทั่วไป ส่วนใหญ่จะใช้วิธีการศึกษาคุณลักษณะของกระบวนการ โดยวิธีการทดลองป้อนสัญญาณขั้นบันไดให้กระบวนการ เพื่อสังเกตผลที่จะเกิดขึ้นในสภาพความเป็นจริง และนำค่าพารามิเตอร์ หรือค่าคุณลักษณะของกระบวนการที่ได้ไปใช้ในการสังเคราะห์ค่าพารามิเตอร์ของตัวควบคุม เพื่อใช้ในการควบคุมกระบวนการต่อไป

3.1 การวิเคราะห์คุณลักษณะของกระบวนการ

โครงสร้างของระบบควบคุมแบบป้อนกลับโดยทั่วไป แสดงดังรูปที่ 3.1



รูปที่ 3.1 แสดงบล็อกไดอะแกรมของระบบควบคุมแบบป้อนกลับ

เมื่อ $R(s)$ = การแปลงลาปลาซ (Laplace Transform) ของค่าอินพุทอ้างอิงหรือค่าเป้าหมาย

$E(s)$ = การแปลงลาปลาซของค่าความคลาดเคลื่อน (Error)

$M(s)$ = การแปลงลาปลาซของสัญญาณควบคุมจากตัวควบคุม

$C(s)$ = การแปลงลาปลาซของสัญญาณเอาท์พุทของทรานสมิตเตอร์ (Transmitter)

$U(s)$ = การแปลงลาปลาซของสัญญาณรบกวน (Disturbance)

$G_c(s)$ = ฟังก์ชันถ่ายโอนของตัวควบคุม

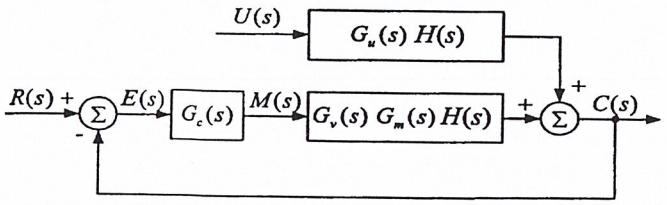
$G_v(s)$ = ฟังก์ชันถ่ายโอนของวาล์วควบคุม (อุปกรณ์ควบคุมสุดท้าย)

$G_m(s)$ = ฟังก์ชันถ่ายโอนของกระบวนการซึ่งอยู่ระหว่างสัญญาณที่ได้จากการควบคุมและสัญญาณการทำงาน

$G_u(s)$= ฟังก์ชันถ่ายโอนของกระบวนการซึ่งอยู่ระหว่างสัญญาณที่ได้จากการควบคุมและสัญญาณรบกวน

$H(s)$= ฟังก์ชันถ่ายโอนของเซนเซอร์-ทรานสมิตเตอร์ (Sensor – Transmitter)

โครงสร้างของระบบควบคุมแบบป้อนกลับในรูปที่ 3.1 สามารถลดรูปเพื่อให้เข้าใจง่าย ดังรูปที่ 3.2



รูปที่ 3.2 แสดงบล็อกไดอะแกรมของระบบควบคุมแบบป้อนกลับหลังการลดรูป

กำหนดให้ การรวมกันของ $G_v(s)G_m(s)H(s)$ เป็นฟังก์ชันถ่ายโอนของกระบวนการ ดังนั้น

$$G(s)=G_v(s)G_m(s)H(s) \qquad \text{.....(3.1)}$$

โดยทั่วไป ฟังก์ชันถ่ายโอนของกระบวนการจะถูกอนุมานให้เป็นรูปแบบของกระบวนการอันดับหนึ่งที่มีการหน่วงเวลา (First-order Lag Plus Dead Time (FOPDT)) เนื่องจากเป็นการอนุมานที่ดีที่สุดสำหรับกระบวนการทางอุตสาหกรรม ซึ่งฟังก์ชันถ่ายโอนของกระบวนการแบบ FOPDT แสดงตามสมการ (3.2)

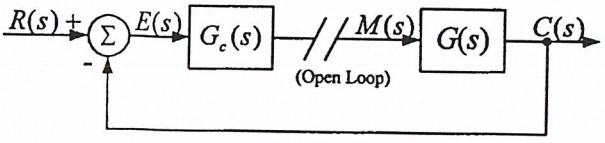
$$G(s) = \frac{Ke^{-t_0 s}}{\tau s+1} \qquad \text{.....(3.2)}$$

โดยที่ K = อัตราการขยายของกระบวนการ (Process Gain)

t_0 = ค่าการหน่วงเวลาของกระบวนการ (Process Dead Time)

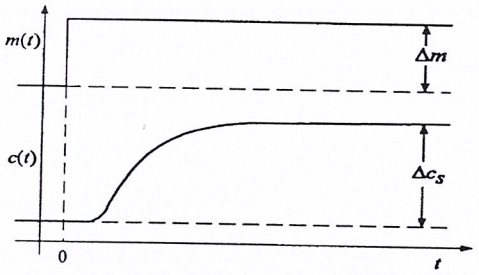
τ = ค่าคงตัวเวลาของกระบวนการ (Process Time Constant)

จากสมการ (3.2) จะเห็นได้ว่าคุณลักษณะของกระบวนการอยู่ในรูปของตัวแปร 3 ตัว ซึ่งสามารถหาได้จากการทดสอบกระบวนการ เนื่องจากคุณลักษณะของกระบวนการเป็นเพียงค่าของกระบวนการเท่านั้น ไม่ใช่ทั้งลูปการควบคุม ดังนั้น ในการทดสอบจะกระทำเฉพาะส่วนของกระบวนการ นั่นคือ ทำการทดสอบแบบ Open – Loop หรือแบบไม่มีการป้อนกลับ และไม่พิจารณาสิ่งรบกวน โดยการป้อนสัญญาณ $M(s)$ แบบ Step ดังรูปที่ 3.3



รูปที่ 3.3 แสดงบล็อกไดอะแกรมสำหรับทดสอบกระบวนการแบบวงเปิด

จากการทดสอบจะได้ผลตอบสนองของกระบวนการที่มีลักษณะเป็นรูปตัว S ดังรูปที่ 3.4 ซึ่งรูปแบบของผลตอบสนองแบบนี้จะครอบคลุมไปถึงกระบวนการระบบอันดับสอง และกระบวนการอันดับสูง (Second and Higher-Order Process) ที่มีอัตราหน่วง (Damping Ratio) มากกว่าหรือเท่ากับหนึ่งได้ด้วย



รูปที่ 3.4 แสดงผลตอบสนองของกระบวนการแบบวงเปิดหลังจากป้อนสัญญาณขั้นบันได

จากรูปที่ 3.3 ฟังก์ชันถ่ายโอนของผลตอบสนองกระบวนการ คือ

$$C(s) = G(s)\,M(s) \qquad \text{.....(3.3)}$$

พิจารณาการเปลี่ยนแปลงค่าสัญญาณควบคุม Δm และกระบวนการตามรูปแบบของ FOPDT ในรูปของการแปลงลาปลาซ จะได้

$$C(s)=\frac{Ke^{-t_0 s}}{\tau s+1}\cdot\frac{\Delta m}{s}$$

$$=K\Delta m e^{-t_0 s}\left[\frac{1}{s}-\frac{\tau}{\tau s+1}\right] \qquad \text{.....(3.4)}$$

แปลงลาปลาซผกผันให้อยู่ในรูปโดเมนเวลา จะได้ว่า

$$\Delta c(t) = K\Delta m \quad u(t-t_0)\left[1-e^{-(t-t_0)/\tau}\right] \qquad \text{.....(3.5)}$$

โดย $\Delta c(t) = c(t) - c(0)$ เป็นการเปลี่ยนแปลงของสัญญาณเอาท์พุทของกระบวนการจากค่าสภาวะเริ่มต้น (Initial Value) ส่วนฟังก์ชัน Unit Step $(u(t-t_0))$ เป็นพจน์ที่กำหนดที่ทำให้ $\Delta c(t) = 0$ ที่เวลา $t \le t_0$

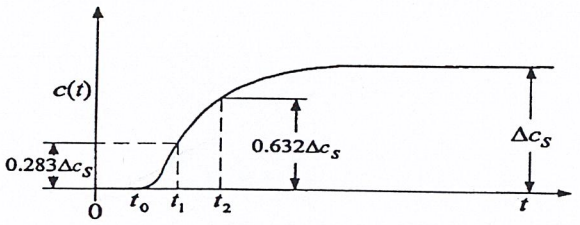
ดังนั้นเมื่อผลตอบสนองของกระบวนการเข้าสู่ที่สภาวะคงที่ (Steady State) $\Delta c(t)$ จะแทนที่ด้วย Δc_s ซึ่งมีค่าเท่ากับ

$$\Delta c_s = \lim_{t\to\infty} \Delta c(t) = K\Delta m \qquad \text{.....(3.6)}$$

ดังนั้นค่าคุณลักษณะของกระบวนการตัวแรกที่สามารถหาได้คืออัตราขยายซึ่งมีค่าเท่ากับ

$$K = \frac{\Delta c_s}{\Delta m} \qquad \text{.....(3.7)}$$

ค่าคุณลักษณะของกระบวนการที่เหลือคือ τ และ t_0 สามารถหาได้โดยการประมาณค่า (Curve Fitting) ดังรูปที่ 3.5



รูปที่ 3.5 แสดงการประมาณค่าพารามิเตอร์ของแบบจำลอง FOPDT

พิจารณาที่เวลา $t = (t_0 + \tau)$ และ $t = (t_0 + \tau/3)$ แทนในสมการ (3.5) จะได้

$$\Delta c(t_0+\tau) = K\Delta m(1-e^{-1}) = 0.632\Delta c_s$$

$$\Delta c(t_0+\frac{\tau}{3}) = K\Delta m(1-e^{-1/3}) = 0.283\Delta c_s \qquad \text{.....(3.8)}$$

กำหนดให้เวลาที่ $t = (t_0 + \tau/3)$ และ $t = (t_0 + \tau)$ เป็น t_1 และ t_2 ตามลำดับ คือ

$$t_1 = t_0 + \frac{\tau}{3} \qquad \text{.....(3.9)}$$

$$t_2 = t_0 + \tau$$

ดังนั้น

$$\boxed{\begin{aligned} t_0 &= t_2 - \tau \\ \tau &= \frac{3}{2}(t_2 - t_1) \end{aligned}} \qquad \text{.....(3.10)}$$

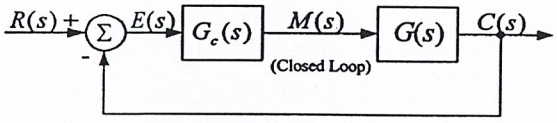
เมื่อ $t_1 =$ เวลาที่ $\Delta c = 0.283\Delta c_s$

$t_2 =$ เวลาที่ $\Delta c = 0.632\Delta c_s$

ค่าคุณลักษณะของกระบวนการทั้ง 3 ค่า คือ K, τ และ t_0 นี้ จะนำไปใช้ในการสังเคราะห์ค่าพารามิเตอร์ของตัวควบคุม PID คือ ค่า K_c, T_i และ T_d ที่จะใช้ปรับตัวควบคุม ดังที่จะได้กล่าวต่อไป

3.2 การสังเคราะห์หาค่าพารามิเตอร์ของตัวควบคุม PID โดยวิธีของ Dahlin

การหาค่าคุณลักษณะของกระบวนการโดยการทดสอบดังที่กล่าวมา จะพิจารณาในลักษณะลูปเปิด ซึ่งต่างจากการสังเคราะห์หาค่าพารามิเตอร์ของตัวควบคุม PID ที่ทำการพิจารณาในลักษณะการควบคุมแบบลูปปิด เนื่องจากต้องพิจารณารวมไปถึงตัวควบคุมด้วย ดังนั้น จากรูปที่ 3.1 ระบบควบคุมลูปปิดเมื่อไม่พิจารณาสิ่งรบกวนจะเป็นดังรูปที่ 3.6



รูปที่ 3.6 แสดงบล็อกไดอะแกรมของระบบแบบลูปปิดเมื่อไม่พิจารณาสิ่งรบกวน

จากรูปที่ 3.6 ฟังก์ชันถ่ายโอนแบบลูปปิด คือ

$$\boxed{\frac{C(s)}{R(s)} = \frac{G_c(s)G(s)}{1 + G_c(s)G(s)}} \qquad \text{.....(3.11)}$$

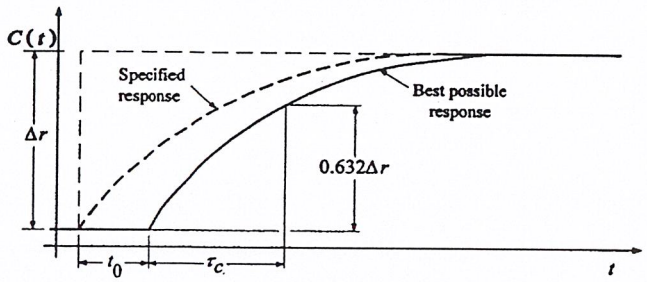
จัดสมการ (3.11) ให้เป็นฟังก์ชันถ่ายโอนของตัวควบคุมได้ คือ

$$\boxed{G_c(s) = \frac{1}{G(s)} \cdot \frac{C(s)/R(s)}{1 - [C(s)/R(s)]}} \qquad \text{.....(3.12)}$$

ฟังก์ชันถ่ายโอนนี้จะใช้สำหรับสังเคราะห์ค่าพารามิเตอร์ของตัวควบคุมต่อไป

พิจารณาผลตอบสนองแบบวงปิดเป็นผลตอบสนองแบบอันดับหนึ่งที่มีค่าหน่วงเวลาเท่ากับ t_0 ที่มีฟังก์ชันถ่ายโอนคือ

$$\frac{C(s)}{R(s)}=\frac{e^{-t_0 s}}{\tau_c s+1} \qquad \text{.....(3.13)}$$



รูปที่ 3.7 แสดงผลตอบสนองแบบปิดที่มีการหน่วงเวลาเป็นเวลา t_0

ดังนั้นจากสมการ (3.2), (3.12) และสมการ (3.13) จะได้ฟังก์ชันถ่ายโอนสำหรับสังเคราะห์ค่าพารามิเตอร์ตัวควบคุมตามกระบวนการแบบ FOPDT คือ

$$G_c(s)=\frac{\tau s+1}{Ke^{-t_0 s}}\cdot\frac{e^{-t_0 s}}{\tau_c s+1-e^{-t_0 s}}$$

หรือ

$$G_c(s)=\frac{\tau s+1}{K}\cdot\frac{1}{\tau_c s+1-e^{-t_0 s}} \qquad \text{.....(3.14)}$$

ถึงแม้ว่าตัวควบคุมชนิดนี้มีความเป็นไปได้ในทางทฤษฎี แต่อย่างไรก็ตามการทำให้เกิดผลในทางปฏิบัตินั้นเป็นไปได้ยาก สาเหตุที่สำคัญที่สุดคือตัวควบคุม PID แบบดั้งเดิมนั้นมีการทำงานในแบบอนาลอกและพจน์ $e^{-t_0 s}$ ไม่สามารถทำให้เกิดผลได้จริงในทางปฏิบัติด้วยอุปกรณ์ทางอนาลอก แต่ตัวควบคุม PID ในปัจจุบันประกอบขึ้นด้วยไมโครโปรเซสเซอร์และดิจิตอลคอมพิวเตอร์จึงทำให้พจน์ของการหน่วงเวลาเกิดผลได้จริง ซึ่งเมื่อมีการทำงานในพจน์นี้จะเรียกว่าพจน์ตัวทำนาย (Predictor) หรือพจน์การชดเชยการหน่วงเวลา (Dead – time compensation)

เทอมเอ็กซ์โพเนนเชียล ($e^{-t_0 s}$) สามารถประมาณค่าได้โดย First – order Pade approximation คือ

$$e^{-t_0 s} = 1 - t_0 s + \frac{1}{2!}(t_0 s)^2 - \frac{1}{3!}(t_0 s)^3 + \ldots\ldots \qquad \ldots\ldots\ldots(3.15)$$

ใช้เฉพาะ 2 เทอมแรก คือ $1 - t_0 s$ (สำหรับกระบวนการอันดับหนึ่ง) แทนในสมการ (3.14) จะได้

$$G_c(s) = \frac{\tau s + 1}{K} \cdot \frac{1}{(\tau_c + t_0)s}$$

$$= \frac{\tau}{K(\tau_c + t_0)} \cdot \left(1 + \frac{1}{\tau s}\right) \qquad \ldots\ldots\ldots(3.16)$$

จากสมการ (3.16) แสดงให้เห็นว่า จากกระบวนการที่เป็นแบบ FOPDT เมื่อนำมาหาฟังก์ชันถ่ายโอนของตัวควบคุมจะได้เป็นตัวควบคุมแบบ PI เท่านั้น โดยมีค่าพารามิเตอร์ คือ

$$K_c = \frac{\tau}{K(\tau_c + t_0)} \quad \text{และ} \quad T_i = \tau \qquad \ldots\ldots\ldots\ldots(3.17)$$

แต่จากการสังเคราะห์หาค่าพารามิเตอร์ของตัวควบคุม PID แบบลูปปิดของ Dahlin ตามกระบวนการ FOPDT สามารถทำเป็นตัวควบคุมแบบ PID ที่มีค่าพารามิเตอร์ของตัวควบคุมคือ

$$K_c = \frac{\tau}{K(\tau_c + t_0)}, T_i = \tau \quad \text{และ} \quad T_d = \frac{t_0}{2} \qquad \ldots\ldots\ldots\ldots(3.18)$$

ข้อแนะนำ สูตรสังเคราะห์หาค่าพารามิเตอร์ของตัวควบคุมของ Dahlin ในโหมด PID ควรใช้เมื่อค่า t_0 มากกว่า $\tau/4$ กรณีนอกเหนือจากนี้ ควรใช้การควบคุมแบบ PI และสำหรับความต้องการการควบคุมให้ค่าพุ่งเกินมีค่าไม่เกิน 5% สำหรับการเปลี่ยนค่าเป้าหมายแนะนำให้ใช้ $\tau_c = t_0$ ดังนั้นที่ความต้องการค่าพุ่งเกิน 5%

$$K_c = \frac{\tau}{K(t_0 + t_0)} = \frac{0.5}{K}\left(\frac{\tau}{t_0}\right) \qquad \ldots\ldots\ldots\ldots(3.19)$$

ตารางหาค่าพารามิเตอร์ของตัวควบคุมของ PID ของ Dahlin แสดงในตารางที่ 3.1

ตารางที่ 3.1 แสดงสูตรสำหรับหาค่าพารามิเตอร์ของตัวควบคุม PID ของ Dahlin

Process	Controller Type	Tuning Parameters
$G(s)=K$	P	$K_c=\frac{1}{K\tau_c}$ tunable
$G(s)=\frac{K}{\tau s+1}$	PI	$K_c=\frac{1}{K\tau_c},T_i=\tau$ tunable
$G(s)=\frac{K}{(\tau_1 s+1)(\tau_2 s+1)}$ $\tau_1>\tau_2$	PID	$K_c=\frac{\tau_1}{K\tau_c},T_i=\tau_1,T_d=\tau_2$ tunable
$G(s)=\frac{Ke^{-t_0 s}}{\tau s+1}$	PID[a]	$K_c=\frac{\tau}{K(t_0+\tau_c)},T_i=\tau,T_d=\frac{t_0}{2}$ tunable

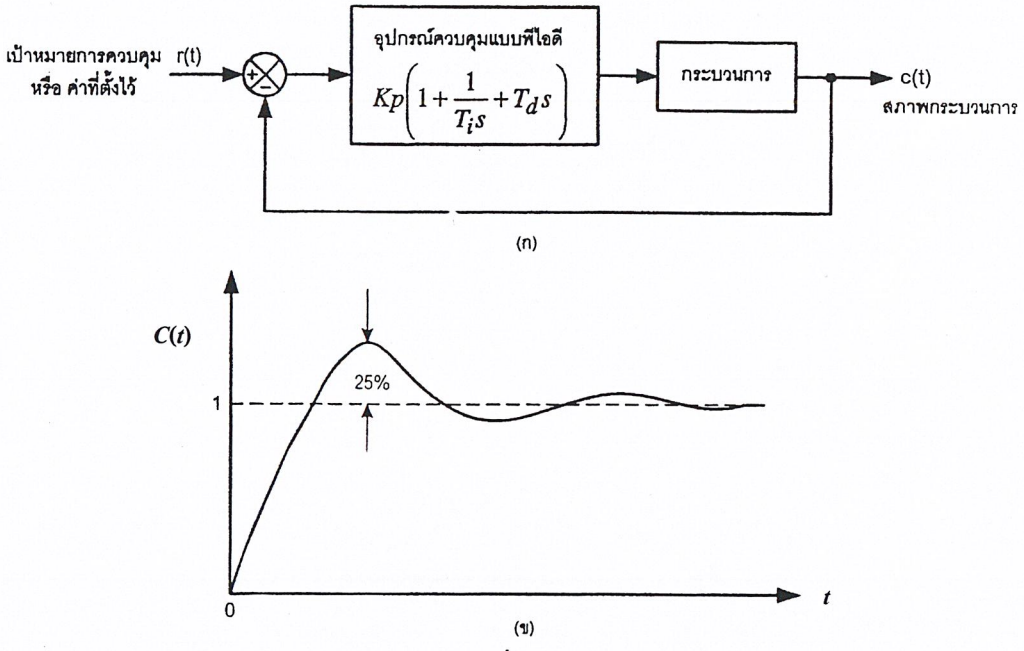
[a] ใช้ได้ทั้งตัวควบคุมแบบ PID และ PI $(T_d=0)$ โดยแบบ PID ใช้ได้เมื่อค่า t_0 มากกว่า $\tau/4$

3.3 การปรับแต่งค่าพารามิเตอร์ของการควบคุมแบบพีไอดี ด้วยวิธี Ziegler-Nichols

วิธี Ziegler-Nichols เป็นวิธีการปรับแต่งค่าพารามิเตอร์ของการควบคุมแบบพีไอดี แบบหนึ่งที่นิยมใช้กันอย่างแพร่หลายอย่างมากในปัจจุบัน รูปแสดงการทำงานแสดงดังรูปที่ 3.8 ก โดยการหาค่าพารามิเตอร์ของกระบวนการซึ่งก็คือ K_p, T_i และ T_d จะขึ้นอยู่กับลักษณะของผลตอบสนองชั่วขณะ (Transient Response) ของกระบวนการที่ถูกควบคุม ซึ่งจะมีอยู่สองวิธี ดังนี้

1. การคำนวณหาค่าพารามิเตอร์ของกระบวนการด้วยวิธี Process Reaction Curve
2. การคำนวณหาค่าพารามิเตอร์ของกระบวนการด้วยวิธี Ultimate Method

โดยทั้งสองวิธีนั้นต่างก็มีจุดมุ่งหมายเดียวกันคือ จะให้ผลตอบสนองของกระบวนการต่อสัญญาณอินพุทแบบสัญญาณระดับหนึ่งหน่วย มีค่าพุ่งเกินสูงสุด (Maximum Overshoot) ไม่เกิน 25 % ดังแสดงในรูปที่ 3.8 ข



รูปที่ 3.8

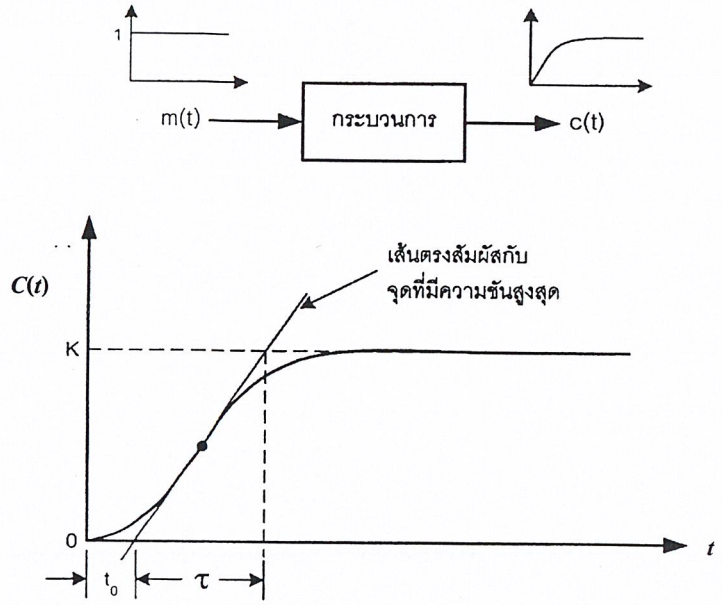
ก. แสดงโครงสร้างของระบบควบคุมแบบ PID

ข. ผลตอบสนองของกระบวนการเมื่อปรับค่าพารามิเตอร์

ด้วยวิธีของ Ziegler-Nichols

3.3.1 การคำนวณหาค่าพารามิเตอร์ของกระบวนการด้วยวิธี Process Reaction Curve

วิธีการคำนวณหาค่าพารามิเตอร์ของกระบวนการการควบคุมแบบพีไอดีด้วยวิธีแบบ Process Reactive Curve นี้จะอาศัยการพิจารณาจากผลตอบสนองของกระบวนการเมื่อป้อนสัญญาณอินพุทที่เป็นสัญญาณระดับหนึ่งหน่วย โดยในกรณีนี้กระบวนการหรือระบบต้องไม่มีตำแหน่งโพลอยู่ที่จุดกำเนิดหรือต้องไม่มีตำแหน่งโพลเชิงซ้อน (Dominant Complex Conjugate Poles) ดังนั้นผลตอบสนองของกระบวนการในเชิงเวลาจะอยู่ในลักษณะตัว S ดังรูปที่ 3.9 ซึ่งวิธีการนี้จะไม่สามารถใช้กับระบบหรือกระบวนการที่ไม่มีผลตอบสนองในลักษณะตัว S เช่นกัน



รูปที่ 3.9 การหาค่าพารามิเตอร์ด้วยวิธี Process Reactive Curve

วิธีการนี้จะแยกการพิจารณาค่าคงที่ได้สองค่า คือค่าเวลาหน่วง (Delay time, t_0) และค่าคงที่เวลา (Time Constant, τ) ซึ่งหาได้โดยการลากเส้นตรงให้สัมผัสกับจุดที่มีความชันสูงสุด และจะได้เส้นตรงเส้นนี้ตัดกับแกนเวลาและแกนเอาท์พุท $c(t)$ ที่จุด $c(t) = K$ ดังนั้นเมื่ออาศัยการประมาณฟังก์ชันด้วยระบบอันดับหนึ่งที่มีการหน่วงเวลา จึงสามารถหาฟังก์ชันถ่ายโอน (Transfer Function) ได้ดังนี้

$$\frac{C(s)}{M(s)} = \frac{Ke^{-t_0 s}}{\tau s + 1} \qquad \text{.....(3.20)}$$

จากวิธี Process Reactive Curve สามารถคำนวณหาค่าพารามิเตอร์ของการควบคุมแต่ละชนิดที่ให้ผลตอบสนองของกระบวนการที่เหมาะสม โดยสรุปได้ดังตารางที่ 3.2 และสามารถสรุปข้อดีและข้อเสียของวิธีดังกล่าวได้ดังต่อไปนี้

ข้อดี ของการใช้วิธี Process Reactive Curve

1. สามารถทำการทดสอบเพียงครั้งเดียว
2. สามารถคำนวณหาค่าพารามิเตอร์ของกระบวนการได้ง่าย
3. ไม่จำเป็นต้องใช้วิธีการลองผิดลองถูก

ข้อเสีย ของการใช้วิธี Process Reactive Curve

1. เนื่องจากการทดสอบกระบวนการกระทำภายใต้สภาวะลูปเปิด (Open Loop) ดังนั้น หากมีการเปลี่ยนแปลงค่าโหลดในขณะที่ทำการทดสอบ ก็จะทำให้ผลการทดสอบที่ได้มีความคลาดเคลื่อนเกิดขึ้น
2. หากเครื่องมือวัดที่ใช้ในการทดสอบมีสัญญาณรบกวน หรือกราฟผลตอบสนองที่ได้จากการทดสอบมีสเกลขนาดเล็กไม่เหมาะสม ก็จะทำให้การหาค่าความชันของกราฟเส้นตรงทำได้ยาก ซึ่งเป็นสาเหตุหนึ่งที่ทำให้เกิดค่าความผิดพลาดขึ้นเช่นกัน
3. วิธีการนี้มีความไวต่อความผิดพลาดจากการสอบเทียบของอุปกรณ์ควบคุม
4. การใช้ค่าพารามิเตอร์ใน ตารางที่ 3.2 มีแนวโน้มที่จะทำให้ผลตอบสนอง ของกระบวนการเกิดการแกว่งได้
5. วิธีการนี้ไม่เหมาะกับกระบวนการควบคุมแบบลูปเปิดที่มีผลตอบสนองแบบแกว่ง เนื่องจากค่าที่ได้จากการทดสอบมักเกิดค่าความผิดพลาดขึ้น

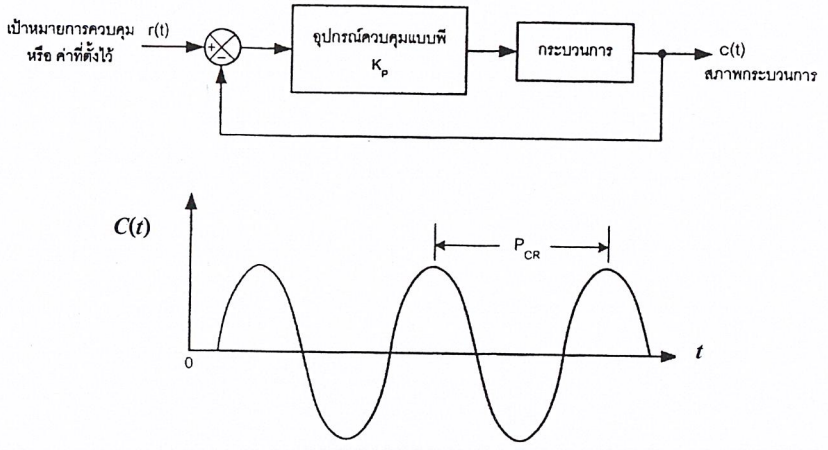
ตารางที่ 3.2 แสดงสูตรสำหรับหาค่าพารามิเตอร์ของตัวควบคุม PID ตามวิธี Process reaction

Controller-Type		Proportional Gain K_c	Integral Time T_i	Derivative Time T_d
Proportional	P	$\frac{1}{K}\left(\frac{\tau}{t_0}\right)$	-	-
Proportional-Integral	PI	$\frac{0.9}{K}\left(\frac{\tau}{t_0}\right)$	$3.33t_0$	-
Proportional-Integral-Derivative	PID	$\frac{1.2}{K}\left(\frac{\tau}{t_0}\right)$	$2.0t_0$	$0.5t_0$

3.3.2 การคำนวณหาค่าพารามิเตอร์ของกระบวนการด้วยวิธี Ultimate Method

วิธีการนี้จะเป็นการคำนวณหาค่าพารามิเตอร์ของกระบวนการควบคุมแบบพีไอดี โดยอาศัยผลตอบสนองของกระบวนการ ซึ่งถูกควบคุมแบบพีต่อสัญญาณอินพุทที่เป็นสัญญาณระดับหนึ่งหน่วย โดยปรับค่าอัตราขยายการควบคุมแบบพีหรือ K_p ไปจนกระทั่งผลตอบสนองเกิดการแกว่งอย่างต่อเนื่อง

(Sustained Oscillations) ดังแสดงในรูปที่ 3.10 ดังนั้นวิธีการลักษณะนี้จึงไม่สามารถกระทำได้ กับกระบวนการ ซึ่งมีผลตอบสนองที่ไม่เกิดการแกว่งอย่างต่อเนื่อง



รูปที่ 3.10 การหาค่าพารามิเตอร์ด้วยวิธี Ultimate Method

จากวิธี Ultimate Method จึงสามารถคำนวณหาค่าพารามิเตอร์ของการควบคุมแต่ละชนิดและนำมาสรุปได้ดังตารางที่ 3.3

ตารางที่ 3.3 แสดงสูตรสำหรับหาค่าพารามิเตอร์ของตัวควบคุม PID ตามวิธี Ultimate Method

Controller-Type		**Proportional Gain** K_c	**Integral Time** T_i	**Derivative Time** T_d
Proportional	P	$K_{cr}/2$	-	-
Proportional-Integral	PI	$K_{cr}/2.2$	$T_{cr}/1.2$	-
Proportional-Integral-Derivative	PID	$K_{cr}/1.7$	$T_{cr}/2$	$T_{cr}/8$

เมื่อ K_{cr} คืออัตราขยายที่ทำให้ผลตอบสนองเกิดการแกว่งอย่างต่อเนื่อง (Critical Gain)

T_{cr} คือคาบเวลาของการแกว่งอย่างต่อเนื่อง (Oscillation Period)

แต่เนื่องจากต้องปรับแต่งค่าพารามิเตอร์ของกระบวนการ ให้มีการทำงานใกล้เคียงกับความไม่เสถียรภาพและยังใช้เวลาค่อนข้างนาน โดยเฉพาะกับกระบวนการที่มีอัตราหน่วงมากๆ เช่น

กระบวนการควบคุมอุณหภูมิ เป็นต้น ดังนั้นด้วยเหตุผลดังกล่าว จึงทำให้วิธีการ Ultimate Method นี้ ไม่ค่อยเป็นที่นิยมใช้กันมากนัก จากวิธีการคำนวณหาค่าพารามิเตอร์ของกระบวนการควบคุมแบบพีไอดี โดยใช้วิธีของ Ziegler-Nichols ทั้งสองวิธีที่ได้กล่าวมาแล้วนั้น ค่าพารามิเตอร์ที่ได้นั้น ไม่สามารถนำไปใช้ได้ทันที ทั้งนี้เนื่องจากค่าพารามิเตอร์ดังกล่าวเป็นค่าที่ได้จากการประมาณให้ใกล้เคียงเท่านั้น รวมทั้งผลตอบสนองของกระบวนการยังมีค่าพุ่งเกินสูงสุดถึง 25% อีกด้วย ดังนั้นในการนำไปใช้งานจึงควรต้องปรับละเอียดค่าพารามิเตอร์ต่างๆเหล่านี้อีก

3.4 PID CONTROL สำหรับ DIGITAL SIGNAL

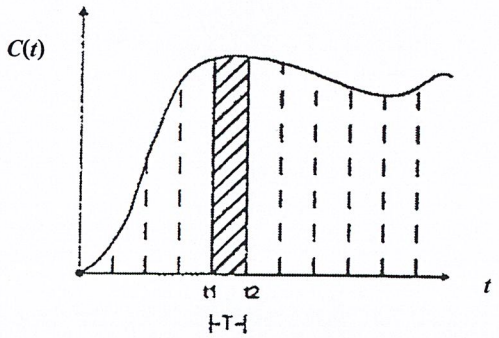
เนื่องจากสัญญาณที่ Controller นำมาคำนวณนี้เป็นสัญญาณดิจิตอล ดังนั้น สมการในการคำนวณ PID เดิม ที่ใช้สำหรับสัญญาณอนาลอกจึงไม่สามารถนำมาใช้ได้โดยตรง สามารถพิจารณาได้ดังนี้

สัญญาณควบคุม(Controller Signal) จาก PID Controller สามารถคำนวณได้จาก

$$E(t) = SP\text{-}PV(t) \qquad \text{.....(3.21)}$$

โดยพิจารณาเป็น 3 ส่วนคือ

$$M(t) = M_p(t) + M_i(t) + M_d(t) \qquad \text{.....(3.22)}$$



T = sampling period

e (t) = Error value

รูปที่ 3.11 ความสัมพันธ์ระหว่าง $e(t)$ กับ t

เมื่อพิจารณาตามรูปที่ 3.11 จะเห็นว่าค่าในส่วนของ Proportional Controller จะมีค่าเท่ากับ

$$M_p(t_2)=K_p*e(t_2) \qquad \text{.....(3.23)}$$

ค่าของ Integral Controller โดยพิจารณาตามรูป จะเห็นว่าค่าในส่วนของ Integral ก็คือพื้นที่ใต้กราฟนั่นเอง เมื่อกำหนด T เป็นคาบในการ sampling มีค่าน้อยมากๆแล้ว จะคิดพื้นที่เป็นสี่เหลี่ยมคางหมูได้นั่นก็คือ

$$M_i(t_2)=K_i*e(t_2)*T+M_i(t_1) \qquad \text{.....(3.24)}$$

ค่าของ Derivative Controller จากรูป ค่าในส่วนของ Derivative ก็คือความชัน (Slope) ของกราฟนั่นเอง

$$M_d(t_2)=K_d*\left[e(t_2)-e(t_1)\right]/T \qquad \text{.....(3.25)}$$

เพราะฉะนั้นเมื่อรวมเข้าด้วยกัน จะได้ว่า สัญญาณควบคุมที่เป็นสัญญาณดิจิตอล (Digital Signal) สามารถคำนวณได้จากสมการ

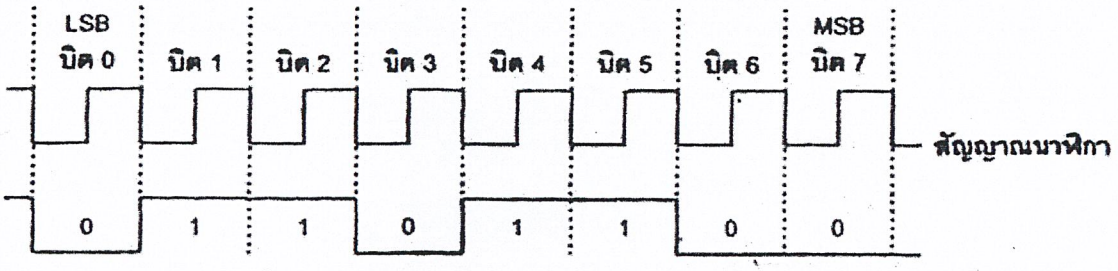
$$M(t_2)=K_p*e(t_2)+K_i*e(t_2)+K_d\left[e(t_2)-e(t_1)\right]/T+M_i(t_1) \qquad \text{..(3.26)}$$

บทที่ 4
การเชื่อมต่อระหว่างคอมพิวเตอร์กับระบบ

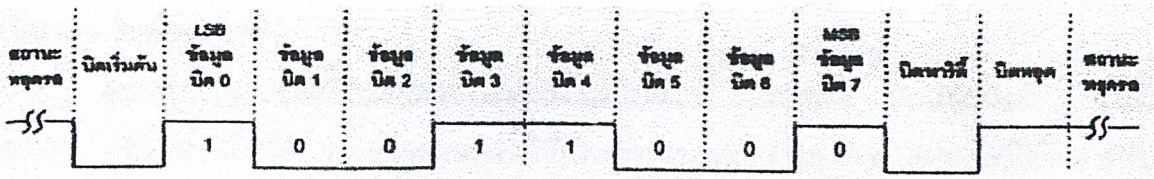
ในปริญญานิพนธ์นี้ ได้พัฒนาการเชื่อมต่อระหว่างคอมพิวเตอร์กับระบบขึ้นมาใหม่แทนการใช้อินเตอร์เฟสแบบเดิม เพื่อความสะดวกในการใช้งานและพัฒนาต่อ โดยอาศัยการรับส่งข้อมูลผ่านพอร์ตอนุกรมตามมาตรฐาน RS-232 เพื่อรับส่งข้อมูลระหว่างคอมพิวเตอร์กับไมโครคอนโทรลเลอร์ตระกูล PIC16F628 ที่ใช้เป็นตัวกลางสำหรับรับส่งข้อมูลระหว่างคอมพิวเตอร์กับระบบ โดยใช้ไอซี PCF8591 เป็นตัวแปลงสัญญาณอนาลอกเป็นดิจิตอล และสัญญาณดิจิตอลเป็นอนาลอก

4.1 การสื่อสารแบบอนุกรม

การสื่อสารแบบอนุกรมนั้นแบ่งออกได้เป็น 2 แบบ คือ การสื่อสารแบบอนุกรมแบบซิงโครนัส และการสื่อสารแบบอนุกรมแบบอะซิงโครนัส การสื่อสารแบบอนุกรมแบบซิงโครนัสจะมีสัญญาณนาฬิการ่วมอยู่กับการรับส่งสัญญาณด้วย ตัวอย่างการส่งข้อมูลแบบอะซิงโครนัส คือ คีย์บอร์ดของคอมพิวเตอร์ ซึ่งสายเส้นหนึ่งจะเป็นของสัญญาณนาฬิกา ส่วนสายอีกเส้นเป็นสายของข้อมูล ดังนั้นการติดต่อกันแบบซิงโครนัสจะต้องใช้สายในการเชื่อมต่ออย่างน้อยที่สุด 3 เส้นคือ สัญญาณนาฬิกา ข้อมูล และกราวด์



รูปที่ 4.1 รูปแบบอย่างง่ายของข้อมูลอนุกรมแบบซิงโครนัส



รูปที่ 4.2 รูปแบบอย่างง่ายของข้อมูลอนุกรมแบบอะซิงโครนัส

4.1.1 การสื่อสารข้อมูลแบบอะซิงโครนัส

การสื่อสารข้อมูลแบบอะซิงโครนัส คือการรับและส่งข้อมูลไปในสายโดยไม่จำเป็นต้องมีสัญญาณนาฬิการ่วมด้วยเหมือนกับการรับส่งข้อมูลแบบซิงโครนัส แต่จะใช้การกำหนดค่าสัญญาณนาฬิกาทั้งภาครับและภาคส่งให้มีค่าเท่ากัน ซึ่งเรียกสัญญาณนาฬิกาที่ใช้ในการกำหนดค่าให้ภาครับภาคส่งว่า อัตราถ่ายทอดข้อมูล หรือ บอดเรต (Baud rate) มีหน่วยเป็น บิตต่อวินาที (Bit Per Second: BPS)

รูปแบบของข้อมูลที่ใช้ในการส่งแบบอะซิงโครนัสประกอบด้วย 4 ส่วนด้วยกัน คือ

1. บิตเริ่มต้น (Start Bit) ซึ่งจะมีขนาด 1 บิต
2. บิตข้อมูลแบบอนุกรม จะมีขนาด 5 ถึง 8 บิต
3. บิตตรวจสอบพาริตี้ (Parity Bit) จะมีขนาด 1 บิต หรือ ไม่มี
4. บิตปิดท้าย (Stop Bit) จะมีขนาด 1,1.5 หรือ 2 บิต

เมื่อไม่มีข้อมูลขา Data จะมีสถานะลอจิกเป็น “1” ซึ่งเรียกว่าสถานะหยุดรอ การเริ่มต้นส่งข้อมูลจะเริ่มจากการให้ขา Data มีลอจิก เป็น “0” ด้วยช่วงระยะเวลา 1 บิต ซึ่งเรียกว่าบิตเริ่มต้นจากนั้นข้อมูลจะถูกส่งออกไป โดยส่งบิตที่มีนัยสำคัญต่ำสุด (LSB) ก่อนแล้วจะตามด้วยพาริตี้บิต เป็นบิตตรวจสอบความผิดพลาด บิตสุดท้ายคือ บิตปิดท้ายจะทำให้ขา Data มีสถานะลอจิกเป็น “1” อีกครั้งด้วยระยะเวลาอย่างน้อย 1, 1.5 หรือ 2 บิต เพื่อแสดงว่าสิ้นสุดข้อมูลแล้ว

อุปกรณ์พิเศษที่ได้รับการออกแบบมาสำหรับการรับส่งข้อมูลอะซิงโครนัส เรียกว่า Universal Asynchronous Receiver/Transmitter หรือ UART อัตราความเร็วในการรับและส่งข้อมูลของการรับและส่งข้อมูลแบบอะซิงโครนัสคือ ค่าบอดเรต ซึ่งก็คือจำนวนบิตต่อวินาทีที่ใช้ในการรับส่งข้อมูล บอดเรตมาตรฐานที่ใช้ในพอร์ตอนุกรม RS-232 ได้แก่ 110, 150, 300, 600, 1200,2400,4800,9600 และ 19200 บิตต่อวินาที และมีค่าเพิ่มขึ้นตามเทคโนโลยีของคอมพิวเตอร์ ซึ่งการรับส่งข้อมูลแบบอนุกรมโดยผ่านโมเด็ม อาจสามารถกำหนดบอดเรตได้ถึง 115200 บิตต่อวินาที เนื่องจากบอดเรตคือจำนวนบิตของข้อมูลที่สามารถถ่ายทอดได้ภายใน 1 วินาที ยกตัวอย่าง ข้อมูลอนุกรมถูกส่งในลักษณะ 8 บิต ไม่มีการตรวจสอบพาริตี้ มีบิตเริ่มต้น 1 บิตและบิตปิดท้าย 1 บิต ความยาวของข้อมูลรับส่งนี้เท่ากับ 10 บิต ถ้าใช้บอดเรต 9600 บิตต่อวินาที ก็จะสามารถรับส่งข้อมูลได้ด้วยความเร็ว 960 ไบต์ต่อวินาที และถ้ามีการใช้พาริตี้ ความเร็วในการรับส่งข้อมูลจะเหลือเป็น 872 ไบต์ต่อวินาที

การตรวจสอบพาริตี้สามารถกำหนดให้เป็นแบบคี่(Odd) หรือแบบคู่ (Even) หรือไม่มีการตรวจสอบพาริตี้ก็ได้ การตรวจสอบพาริตี้เป็นการตรวจสอบจำนวนรวมของบิตที่เป็นลอจิก “1” ภายในข้อมูลที่ส่งไป 1 ไบต์ ว่ามีจำนวนเป็นคู่หรือคี่ โดยรวมพาริตี้บิตเข้าไปด้วย

4.1.2 มาตรฐานพอร์ตอนุกรมแบบ RS-232

มาตรฐานการเชื่อมต่อแบบอนุกรม RS-232 เป็นมาตรฐานอุตสาหกรรมที่ออกแบบมาเพื่อใช้ในการส่งข้อมูลอนุกรมแบบอะซิงโครนัส 2 ทิศทาง โดยมาตรฐาน RS-232 ในอดีตนั้นถูกออกแบบมาเพื่อการส่งผ่านข้อมูลจากคอมพิวเตอร์ไปยังโมเด็มเพียงอย่างเดียว เพื่อที่จะนำข้อมูลจากโมเด็มนี้สื่อสารผ่านสายโทรศัพท์ไปยังคอมพิวเตอร์อีกชุดที่อยู่ห่างไกลกัน โดยคณะกรรมการที่เรียกว่า สมาคมอุตสาหกรรมอิเล็กทรอนิกส์ (Electronic Industrial Association: EIA) ได้วางมาตรฐานที่มีชื่อเรียกกันว่า EIA RS-232 มาตรฐานนี้ในช่วงแรกจะใช้คอนเน็กเตอร์เป็นแบบ DB-25 โดยกำหนดความยาวสูงสุดของสายสัญญาณไว้ที่ 50 ฟุต มีระดับสัญญาณตั้งแต่ -3 ถึง -12V แสดงว่ามีข้อมูล(Mark) และ +3 ถึง +12V แสดงว่าเป็นช่องว่าง (Space)

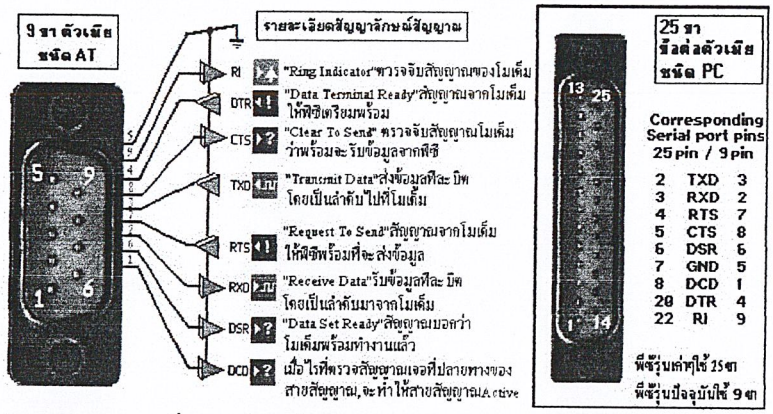
มาตรฐาน RS-232 ได้กำหนดรูปแบบของอุปกรณ์เชื่อมต่อข้อมูล (Data Terminal Equipment: DTE) กับวงจร ข้อมูลปลายทาง (Data Circuit Terminating: DCE) ไว้ว่า อุปกรณ์ DTE จะต้องเป็นอุปกรณ์ที่มีการประมวลผลในตัวเช่น ไมโครคอนโทรลเลอร์ หรือ ไมโครคอมพิวเตอร์ ซึ่งมีความสามารถในการสร้างบิตข้อมูลแบบอนุกรมได้ ส่วนอุปกรณ์ DCE จะทำหน้าที่เป็นเพียงตัวรับข้อมูลที่ส่งมาจาก DTE เท่านั้น โดยการรับส่งข้อมูลระหว่างอุปกรณ์ทั้งสองจะกระทำผ่านมาตรฐาน RS-232

ข้อแตกต่างของอุปกรณ์ DTE และอุปกรณ์ DCE อย่างหนึ่งที่เห็นได้ชัดคือ คอนเน็กเตอร์ของ DTE จะเป็นตัวผู้ ส่วนคอนเน็กเตอร์ของ DCE จะเป็นตัวเมีย ซึ่งพอร์ตอนุกรมของคอมพิวเตอร์ที่ใช้กันอยู่ทั่วไปจะเป็นแบบ DTE ส่วนคอนเน็กเตอร์ที่อยู่ที่โมเด็มจะเป็นแบบ DCE

สำหรับการใช้งานบนคอมพิวเตอร์ พอร์ตอนุกรม RS-232 มักถูกใช้เชื่อมต่อกับโมเด็มหรือเมาส์ โดยสามารถรับส่งข้อมูลได้ที่ความยาวของสายสัญญาณสูงสุดถึง 20 เมตร

4.1.3 คอนเน็กเตอร์สำหรับพอร์ต RS-232 และการเชื่อมต่อ

มาตรฐานการเชื่อมต่อแบบ RS-232 จะใช้คอนเน็กเตอร์แบบ DB-25 ตัวผู้หรือ DB-9 ตัวผู้ ซึ่งคอนเน็กเตอร์แบบ DB-25 จะมีขาต่อใช้งานเพียง 9 เส้นเช่นเดียวกับคอนเน็กเตอร์แบบ DB-9 เนื่องจากขาอื่นๆที่เคยใช้งานในอดีต ปัจจุบันมีการใช้งานไม่มากนัก จึงถูกยกเลิกไป



รูปที่ 4.3 การจัดขาของคอนเน็กเตอร์พอร์ตอนุกรมตามมาตรฐาน RS-232 แบบ DB-25 และ DB-9

รายละเอียดหน้าที่การทำงานในแต่ละขาของพอร์ตอนุกรม RS-232 มีดังนี้

Data Carrier Detect: DCD เรียกว่า Carrier Detect: CD ขานี้จะแอกทีฟ เมื่อมีการส่งสัญญาณพาหะจากอุปกรณ์สื่อสารข้อมูล เช่น โมเด็ม สำหรับการใช้งานปกติ ขานี้จะไม่ได้ถูกใช้งานมากนัก

Receiver Data: RD หรือ RxD ขานี้ใช้เพื่อรับสัญญาณอนุกรมเข้ามายังคอมพิวเตอร์ โดยนำข้อมูลในบัฟเฟอร์ส่งออกข้อมูลส่งออกไป

Transmitter Data:TD (TxD) ใช้ส่งข้อมูลอนุกรมออกจากคอมพิวเตอร์ โดยนำข้อมูลในบัฟเฟอร์ส่งออกข้อมูลส่งออกไป

Data Terminal Ready:DTR เป็นขาสัญญาณที่ส่งออกจากคอมพิวเตอร์ เพื่อให้อุปกรณ์ปลายทางรับรู้ว่าต้องการติดต่อ โดยขา DRT นี้จะต้องเชื่อมต่อกับขา DSR ของอุปกรณ์ปลายDB-25ทาง และขา DTR ของอุปกรณ์ปลายทางจะต้องเชื่อมต่อกับขา DSR ของคอมพิวเตอร์ ถ้าใช้การเชื่อมต่อแบบ Null Modem ซึ่งใช้สายสัญญาณเพียง 3 เส้นจะต้องต่อขา DTR และ DSR ของตัวมันเองเข้าด้วยกัน และต้องต่อกับขา DCD ด้วย ในกรณีที่โปรแกรมสื่อสารที่ใช้มีการตรวจจับสัญญาณพาหะ

Signal Ground: GND ขากราวด์ของระบบ

Data Set Ready: ขานี้จะใช้คู่กับขา DTR เพื่อตรวจสอบการเชื่อมต่อกันระหว่างคอมพิวเตอร์กับอุปกรณ์ปลายทาง ซึ่งขา DSR นี้จะเป็นขาสำหรับรับข้อมูลจากภายนอกซึ่งถูกส่งมาจากขา DTR

Request To Send: RTS	เป็นขาสำหรับร้องขอให้ทางอุปกรณ์ปลายทางส่งข้อมูลกลับมายังคอมพิวเตอร์โดยขาที่รับสัญญาณ RTS ก็คือขา CTS ในกรณีที่เชื่อมต่อแบบ Null Modem 3 สาย จะต้องเชื่อมต่อขา RTS และ CTS ของตัวมันเองเข้าด้วยกัน เพื่อให้การรับและส่งข้อมูลเกิดขึ้นได้ตลอดเวลา
Clear To Send: CTS	ขานี้จะคอยรับสัญญาณจากขา RTS เมื่อรับสัญญาณได้ ข้อมูลที่ขาTxDจะถูกส่งออกไป ดังนั้นขานี้จะถูกใช้ในการตรวจสอบว่าอุปกรณ์ต่อพ่วงพร้อมจะรับข้อมูลหรือไม่
Ring Indicator: RI	ใช้แสดงสถานะสัญญาณเรียกจากสายโทรศัพท์ปกติในการสื่อสารโดยทั่วไป สายนี้จะไม่ถูกใช้งานจะใช้งานก็ต่อเมื่อต่อกับโมเด็ม และโปรแกรมที่มีการตรวจสอบสัญญาณปลายทางนี้เท่านั้น

4.1.4 UART

UART มาจากคำว่า Universal Asynchronous Receiver/Transmitter ซึ่งหมายถึงอุปกรณ์ที่ทำหน้าที่รับส่งข้อมูลแบบอะซิงโครนัสนั่นเอง สำหรับการการสื่อสารแบบอนุกรมบนคอมพิวเตอร์แล้ว UART ถือเป็นหัวใจสำคัญของการสื่อสารแบบอนุกรม

หน้าที่หลักของ UART คือทำหน้าที่แปลงข้อมูลที่อยู่ในรูปแบบขนานจากคอมพิวเตอร์ให้อยู่ในรูปแบบอะซิงโครนัสแล้วส่งออกไป และทำหน้าที่แปลงสัญญาณอนุกรมแบบอะซิงโครนัสที่ป้อนเข้ามายัง UART ให้เป็นแบบขนานก่อนที่จะส่งข้อมูลเข้าคอมพิวเตอร์ ซึ่งนอกจาก UART จะส่งข้อมูลไปยังคอมพิวเตอร์แล้วยังแจ้งข้อมูลอื่นๆให้คอมพิวเตอร์ทราบด้วย เช่น อัตราเร็วในการรับส่งข้อมูล(บอดเรต), รูปแบบของการส่งข้อมูล, ความผิดพลาดที่เกิดขึ้นระหว่างการส่งข้อมูล

ภายใน UART จะมีส่วนของวงจรสร้างบอดเรตแบบโปรแกรมได้ (Programmable buadrate generator) โดยการกำหนดค่าตัวหารให้กับสัญญาณนาฬิกาของ UART โดยตัวหารนี้มีขนาด 16 บิต ดังนั้นจึงสามารกำหนดตัวหารอยู่ในช่วง 1-65,535 UART สามารถรับส่งข้อมูลทั้งแบบฮาล์ฟดูเพล็ก (Half Duplex) และฟูลดูเพล็ก (Full Duplex) โดยการส่งแบบฮาล์ฟดูเพล็ก เป็นการส่งแบบทิศทางเดียว ส่วนการส่งแบบ ฟูลดูเพล็ก เป็นการรับส่งข้อมูลในคราวเดียวกัน

ชนิดของ UART

ในเครื่องคอมพิวเตอร์ทั่วไปมี UART ที่ใช้งานกันอยู่ 2 เบอร์คือ 8250 ซึ่งเป็น UART มาตรฐานที่มีใช้กันยาวนาน UART เบอร์นี้จะมีบัฟเฟอร์สำหรับรับและส่งข้อมูลเป็นตำแหน่งเดียวกันทำให้การรับและส่งข้อมูลจำกัดอยู่เพียง 57.6 กิโลบิตต่อวินาทีเท่านั้น แต่ UART เบอร์นี้ก็ถือว่าเป็นต้นแบบของ UART ที่ใช้ในคอมพิวเตอร์ โดยคอมพิวเตอร์ทุกๆรุ่นจะต้องสนับสนุนการทำงานตามรูปแบบของ UART เบอร์นี้

UART อีกเบอร์หนึ่งคือ 16450 มีความสามารถรับส่งข้อมูลได้ที่ความเร็ว 115,200 บิตต่อวินาที และเพิ่มรีจีสเตอร์สำหรับพักข้อมูลสำหรับ UART นอกจากนี้ยังเพิ่มส่วนของชิพต์รีจีสเตอร์แบบ FIFO (First In First Out) ขนาด 16 บิตเข้าไปทำให้สามารถสนับสนุนความเร็วในการรับส่งข้อมูลที่ 256 กิโลบิตต่อวินาที โดยคอมพิวเตอร์ในปัจจุบันใช้ UART เบอร์นี้หรือใหม่กว่าเช่นเบอร์ TL16C750 ซึ่งมีรีจีสเตอร์ FIFO ขนาด 64 ไบต์ ทำงานได้ที่ระดับแรงดัน +5V และ +3V มีโหมดประหยัดพลังงาน สามารถส่งข้อมูลได้ที่ความเร็ว 1 เมกะบิตต่อวินาที เมื่อใช้สัญญาณนาฬิกา 16 MHz

อย่างไรก็ตามความเร็วในการส่งข้อมูลที่มีมากมายของ UART เบอร์ใหม่ๆก็ไม่ได้ช่วยให้การรับส่งข้อมูลของคอมพิวเตอร์เร็วขึ้น เนื่องจากว่าคอมพิวเตอร์ยังใช้ความถี่ของสัญญาณนาฬิกาในการแปลงข้อมูลเพียง 1.8432 MHz เท่านั้น

4.2 ระบบบัส I^2C และการใช้อุปกรณ์ไอซี PCF8591

4.2.1 ความรู้เบื้องต้นเกี่ยวกับ I^2C

I^2C ย่อมาจาก Inter-IC Communication หมายถึง การติดต่อสื่อสารระหว่างไอซี โดยบัส I^2C ได้รับการพัฒนาขึ้นโดยฟิลลิปส์ (Phillips) ด้วยจุดมุ่งหมายหลักคือ ต้องการให้ไอซีหรือโมดูลสามารถติดต่อ สั่งงาน และควบคุมภายใต้สายสัญญาณเพียง 2 เส้น เส้นหนึ่งคือ สายข้อมูล อีกเส้นหนึ่งคือสายสัญญาณนาฬิกาที่ใช้ในการกำหนดจังหวะการทำงาน การต่อร่วมกันของอุปกรณ์บนบัสทำได้ง่ายมาก เพียงต่อสายสัญญาณนาฬิกาของอุปกรณ์แต่ละตัวขนานหรือพ่วงกันไป ส่วนการกำหนดแอดเดรสหรือตำแหน่งสำหรับติดต่ออุปกรณ์แต่ละตัว จะใช้รหัสข้อมูลและการกำหนดสภาวะลอจิกที่ขาแอดเดรสของอุปกรณ์แต่ละตัว

สายข้อมูลบนบัสมีชื่อเรียกอย่างเป็นทางการว่า สายข้อมูลอนุกรมหรือ SDA (Serial Data Line) ส่วนสายสัญญาณนาฬิกามีชื่อเรียกว่า สายสัญญาณนาฬิกาอนุกรมหรือ SCL (Serial Clock Line)

4.2.2 คุณสมบัติโดยทั่วไปของบัส I^2C

สาย SDA และ SCL เป็นสายสัญญาณ 2 ทิศทาง (Bi Direction Line) ต้องมีการต่อตัวต้านทานพูลอัปกับแรงดัน 5 โวลต์ตลอดเวลา เพื่อให้สายสัญญาณ มีสถานะลอจิกสูงในขณะไม่มีการติดต่อใช้งาน ทั้งยังช่วยในการป้องกันสัญญาณรบกวนที่อาจมีเข้ามาในสายสัญญาณทั้งสอง

อัตราการถ่ายทอดข้อมูลบนบัสสูงถึง 100 กิโลบิตต่อวินาทีในโหมดปกติ และสูงถึง 400 กิโลบิตต่อวินาทีในโหมดความเร็วสูง อุปกรณ์ที่ต่อร่วมอยู่บนบัส จะต้องมีความจุไฟฟ้ารวมที่เกิดขึ้นระหว่างสาย SDA และ SCL ไม่เกิน 400 pF

ข้อเด่นอีกประการหนึ่งคือ สามารถเชื่อมต่ออุปกรณ์ที่ใช้ไฟเลี้ยงไม่เท่ากันให้สามารถติดต่อสื่อสารกันได้

4.2.3 หลักการของบัส I^2C

อุปกรณ์ที่ต่อพ่วงบนบัสสามารถมีได้มากมาย ดังนั้นจึงต้องมีการกำหนดรูปแบบของการติดต่อบนบัส หรือเรียกว่า โปรโตคอล (Protocol) เพื่อที่ผู้ใช้งานได้ทราบว่า ขณะนี้กำลังติดต่ออยู่กับอุปกรณ์ตัวใดอยู่ และอุปกรณ์ตัวใดเป็นตัวรับเป็นตัวส่ง โดยมีข้อตกลงพื้นฐานดังนี้

อุปกรณ์ที่เป็นผู้สร้างข้อมูลหรือตัวส่งข้อมูล เรียกว่า *ตัวส่ง (Transmitter)*

อุปกรณ์ที่เป็นผู้รับข้อมูล เรียกว่า *ตัวรับ(Receiver)* อุปกรณ์บนบัสสามารถเป็นได้ทั้งตัวรับ-ตัวส่ง บางอุปกรณ์ทำหน้าที่เป็นตัวรับอย่างเดียว จะไม่มีอุปกรณ์ใดบนบัสที่มำหน้าที่เป็นตัวส่งอย่างเดียว

อุปกรณ์ที่ทำหน้าที่ควบคุมจังหวะการติดต่อบนบัส เรียกว่า *มาสเตอร์ (Master)*

อุปกรณ์ที่ถูกควบคุมหรืออุปกรณ์ที่ต่อพ่วงเข้าไปบนบัส เรียกว่า *สเลฟ (Slave)*

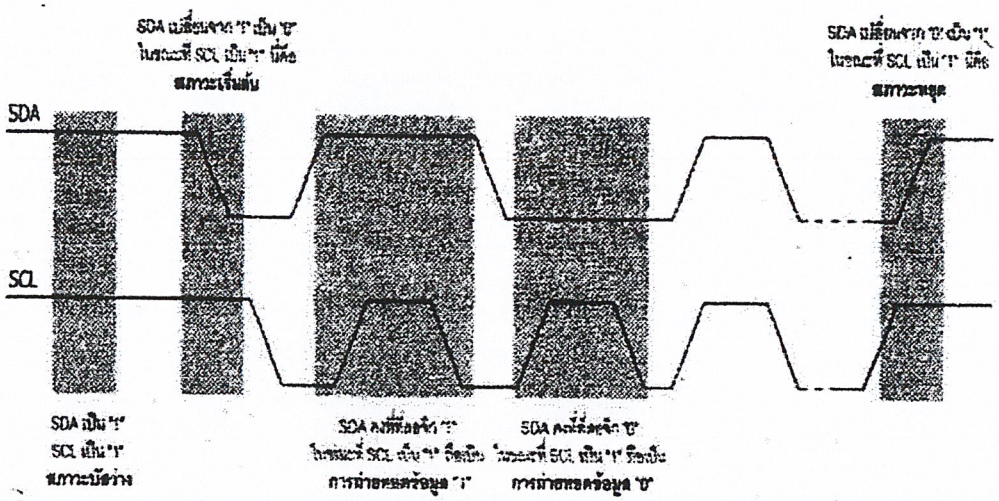
ข้อกำหนด 2 ประการสำคัญของการติดต่อ คือ

1. การถ่ายทอดข้อมูลจะเกิดขึ้นได้เมื่อบัสว่างเท่านั้น
2. ในระหว่างการถ่ายทอดข้อมูล เมื่อใดก็ตามที่สาย SCL มีสถานะเป็นลอจิกสูง สายข้อมูลต้องรักษาข้อมูลไว้ อย่าให้เกิดการเปลี่ยนแปลงขึ้นเด็ดขาด มิฉะนั้นสัญญาณที่เกิดขึ้นจะได้รับการแปลความหมายเป็นสัญญาณควบคุม

4.2.4 สภาวะที่เกิดขึ้นบนบัส I^2C

มีด้วยกัน 5 สภาวะ ดังนี้

1. บัสว่าง สภาวะนี้เกิดขึ้นเมื่อสถานะของลอจิกบนสาย SDA และ SCL เป็นลอจิกสูงทั้งคู่ นั่นหมายความว่าการถ่ายทอดข้อมูลสามารถเริ่มต้นขึ้นได้
2. เริ่มต้นการถ่ายทอดข้อมูล เกิดขึ้นเมื่อสาย SDA มีการเปลี่ยนแปลงระดับลอจิกจากสูงไปต่ำในขณะที่สาย SCL มีสถานะลอจิกสูง เรียกสภาวะนี้ว่า *สภาวะเริ่มต้น (Start)*
3. หยุดการถ่ายทอดข้อมูล เกิดขึ้นเมื่อสาย SDA มีการเปลี่ยนแปลงระดับลอจิกจากต่ำไปสูงในขณะที่สาย SCL มีสถานะลอจิกสูง เรียกสภาวะนี้ว่า *สภาวะหยุด (Stop)*
4. ข้อมูลดำรงอยู่บนบัส สภาวะนี้เกิดขึ้นถัดจากสภาวะเริ่มต้น โดยสถานะลอจิกที่เกิดขึ้นบนสาย SDA ก็คือข้อมูลที่กำลังถ่ายทอด เมื่อสาย SCL เป็นลอจิกสูง สถานะที่สาย SDA ต้องคงที่เพื่อให้อุปกรณ์รับรู้ข้อมูลในจังหวะนั้นว่า เป็น “0” หรือ “1” ข้อมูลอาจเกิดการเปลี่ยนแปลงได้ ในขณะที่ SCL เป็นลอจิกต่ำ แต่เมื่อใดก็ตามที่ต้องการให้เกิดการถ่ายทอดข้อมูลอย่างสมบูรณ์สถานะลอจิกที่ SDA ต้องคงที่ตลอดเวลา ช่วงเวลาที่สาย SCL มีสถานะลอจิกสูง หากเกิดการเปลี่ยนแปลงสถานะในขณะที่สาย SCL มีสถานะลอจิกสูงนั้น อุปกรณ์มาสเตอร์ที่ทำการควบคุมการถ่ายทอดข้อมูลจะแปลความหมายเป็นสภาวะหยุด หรือสภาวะเริ่มต้นได้
5. รับรู้ข้อมูล เกิดขึ้นหลังจากการถ่ายทอดข้อมูลจากตัวส่งมายังตัวรับเกิดขึ้นอย่างสมบูรณ์ โดยตัวส่งจะทำการส่งข้อมูลมา 1 บิต เรียกว่า บิตรับรู้ มีสถานะเป็นลอจิกสูง หลังจากส่งข้อมูลมาครบถ้วน ส่วนอุปกรณ์มาสเตอร์จะทำการส่งสัญญาณรับรู้พิเศษซึ่งสัมพันธ์กับสัญญาณนาฬิกา เพื่อตอบสนองบิตรับรู้ที่ส่งมจากตัวส่ง ทางด้านตัวรับจะส่งบิตรับรู้ที่มีสถานะลอจิกต่ำลงบนบัสอุปกรณ์สเลฟที่ถูกอ้างถึงในการติดต่อหรือกำลังจะติดต่อขณะนั้น ก็จะกำเนิดบิตรับรู้เพื่อตอบสนองให้ทราบว่าได้รับข้อมูลแต่ละไบต์เรียบร้อยแล้ว ในรูปที่ 4.4 เป็นไดอะแกรมเวลาที่แสดงถึงการเกิดสภาวะต่างๆบนบัส ไม่ว่าจะเป็นสภาวะบัสว่าง, เริ่มต้น, ถ่ายทอดข้อมูล และหยุดการถ่ายทอดข้อมูล



รูปที่ 4.4 ไดอะแกรมเวลาที่แสดงสถานะต่างๆบนบัส I^2C

4.2.5 การติดต่ออุปกรณ์ระบบบัส I^2C กับไมโครคอนโทรลเลอร์ PIC16F628

การสร้างสภาวะเริ่มต้น

1. เมื่อต้องการติดต่อกับบัส I^2C สิ่งแรกที่ต้องทำสำหรับไมโครคอนโทรลเลอร์ซึ่งเป็นอุปกรณ์มาสเตอร์คือ การทำให้บัสว่างด้วยการกำหนดขา SCL และ SDA ให้มีลอจิกเป็น 1 ทั้งคู่
2. จากนั้นทำให้ขา SDA มีลอจิก “0” โดย SCL เป็น “1” อยู่
3. กำหนดให้ขา SCL มีลอจิกเป็น “0” ถึงตอนนี้ SDA และ SCL มีสถานะเป็น “0” ทั้งคู่พร้อมที่จะติดต่อแล้ว

การสร้างสภาวะหยุด

1. เมื่อต้องการหยุดส่งข้อมูลจะต้องส่งสภาวะหยุดรอสภาวะออกไป โดยตอนแรกต้องกำหนดให้ขา SCL และ SDA เป็นลอจิก “0” ทั้งคู่
2. กำหนดให้ขา SCL มีลอจิกเป็น “1” โดย SDA ยังคงมีลอจิกเป็น “0” อยู่
3. จากนั้นทำให้ขา SDA มีลอจิกเป็น “1” ซึ่งจะทำให้ระบบบัสเข้าสู่สภาวะบัสว่าง พร้อมที่จะรับส่งข้อมูลต่อไป

<u>การส่งข้อมูลลอจิก “0” และลอจิก “1”</u>

การส่งข้อมูลลอจิก “0”

1. ทำให้ขา SDA เป็น “0” สำหรับการส่งข้อมูลลอจิก “0”
2. ทำให้ขา SCL เป็น “1” สำหรับการป้อนสัญญาณนาฬิกา ในขณะที่ SDA ยังคงเป็น “0” อยู่

3. จากนั้นทำให้ขา SCL กลับมาเป็นลอจิก "0" เหมือนเดิม

การส่งข้อมูลลอจิก "1"

1. ทำให้ขา SDA เป็น "1" สำหรับการส่งข้อมูล "1"
2. ทำให้ขา SCL เป็น "1" สำหรับการป้อนสัญญาณนาฬิกา ในขณะที่ SDA ยังคงเป็น "0" อยู่
3. จากนั้นทำให้ขา SCL กลับมาเป็นลอจิก "0" เหมือนเดิม

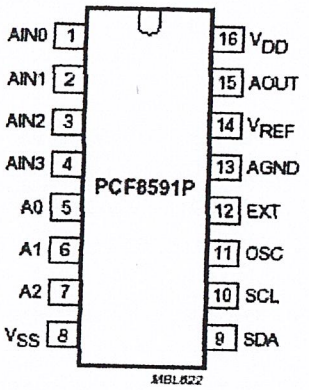
4.3 ข้อมูลเบื้องต้นของ PCF8591

ในการทดลองนี้ใช้ไอซี ADC ที่มีความสามารถสูงเบอร์ PCF8591 เนื่องจากในตัวมันมีวงจร ADC แบบซักเซสซีฟแอปพร็อกซิเมชั่นขนาด 8 บิตสูงถึง 4 ช่อง ทั้งยังมีวงจร DAC อีก 1 ช่องด้วย ระบบการเชื่อมต่อเป็นแบบบัส I^2C ทำให้สามารถใช้สายสัญญาณเพียง 2 เส้น ทั้งยังสามารถต่อพ่วงกันได้สูงสุด 8 ตัว ทำให้ได้วงจร ADC รวมสูงถึง 32 ช่อง และวงจร DAC รวม 4 ช่อง สามารถนำไปประยุกต์ใช้งานได้อย่างกว้างขวางมีรายละเอียดคุณสมบัติทางเทคนิคดังนี้

- ทำงานโดยใช้แหล่งจ่ายไฟชุดเดียว
- ทำงานที่แรงดัน 2.5V-6V
- กินกระแสขณะอยู่ในสภาวะสแตนบายด์ต่ำ
- ติดต่อกับไมโครคอนโทรลเลอร์หรือไมโครคอมพิวเตอร์ผ่านระบบบัส I^2C
- เลือกตำแหน่งแอดเดรสทางฮาร์ดแวร์จากขา A0,A1,A2 ทำให้สามารถต่อพ่วงกันได้สูงสุดถึง 8 ตัว
- อัตราการสุ่มข้อมูล(Sampling) ขึ้นอยู่กับความเร็วของสัญญาณนาฬิกาบนบัส I^2C
- วงจรแปลงสัญญาณอนาลอกเป็นดิจิตอล(ADC)สามารถรับสัญญาณอนาลอกได้ 4 ช่อง ทั้งยังเลือกได้ว่าให้ทำงานแบบแยกช่องหรือทำงานเป็นวงจรดิฟเฟอเรนเชียล
- การอ่านค่าสามารถกำหนดให้เลื่อนช่องอินพุทโดยอัตโนมัติได้
- สัญญาณอนาลอกมีระดับแรงดันตั้งแต่ V_{SS} ไปจนถึง V_{DD}
- วงจรแปลงสัญญาณอนาลอกเป็นดิจิตอลเป็นแบบซักเซสซีฟแอปพร็อกซิเมชั่นขนาด 8 บิต
- มีวงจรแปลงสัญญาณดิจิตอลเป็นอนาลอกขนาด 8 บิต 1 ช่อง

PCF8591 สามารถทำหน้าที่เป็นไอซีแปลงสัญญาณอนาลอกเป็นดิจิตอลขนาด 8 บิต 4 ช่อง และทำหน้าที่เป็นไอซีแปลงสัญญาณดิจิตอลเป็นอนาลอกได้ในคราวเดียวกัน ด้วยการควบคุมผ่านระบบบัส I^2C ทำให้สามารถต่อพ่วงไอซี PCF8591 ได้สูงสุดถึง 8 ตัว รองรับการอ่านค่าสัญญาณอนาลอกอินพุทได้

สูงสุดถึง 32 ช่องADC และสามารถส่งสัญญาณอนาลอกเอาต์พุตสูงสุดได้ถึง 8 ช่องด้วยการกำหนดแอดเดรสจากขา A0, A1 และ A2 การจัดขาของ PCF8591 แสดงดังรูปที่ 4.5 ส่วนรายละเอียดตำแหน่งขาต่างๆมีดังนี้



รูปที่ 4.5 การจัดขาของไอซี ADC/DAC ของ 8 บิตผ่านบัส I^2C เบอร์ PCF8591

ขา AN0-AN3 (ขา1-4) เป็นขาอินพุทสำหรับป้อนสัญญาณอนาลอกที่ต้องการแปลงค่า

ขา A0-A2 (ขา5-7) เป็นขาสำหรับกำหนดข้อมูลแอดเดรสทางฮาร์ดแวร์ ปกติต่อลงกราวด์ แต่ถ้ามีการใช้งาน PCF8591 มากกว่า 1 ตัวต้องกำหนดการต่อขา A0-A2 ของ PCF8591 ให้มาตรงกัน จึงทำให้สามารถต่อใช้งานร่วมกันได้สูงสุด 8 ตัว

ขา V_{ss} (ขา 8) เป็นขาต่อกราวด์

ขา SDA, SCL (ขา9, 10) เป็นขาเชื่อมต่อระบบบัส I^2C

ขา OSC (ขา 11) เป็นขาสำหรับต่อกับสัญญาณนาฬิกาภายนอกเมื่อขา EXT ต่อกับไฟ +5V และจะทำงานเป็นขาเอาต์พุตสัญญาณนาฬิกาถ้าขา EXT ต่อลงกราวด์

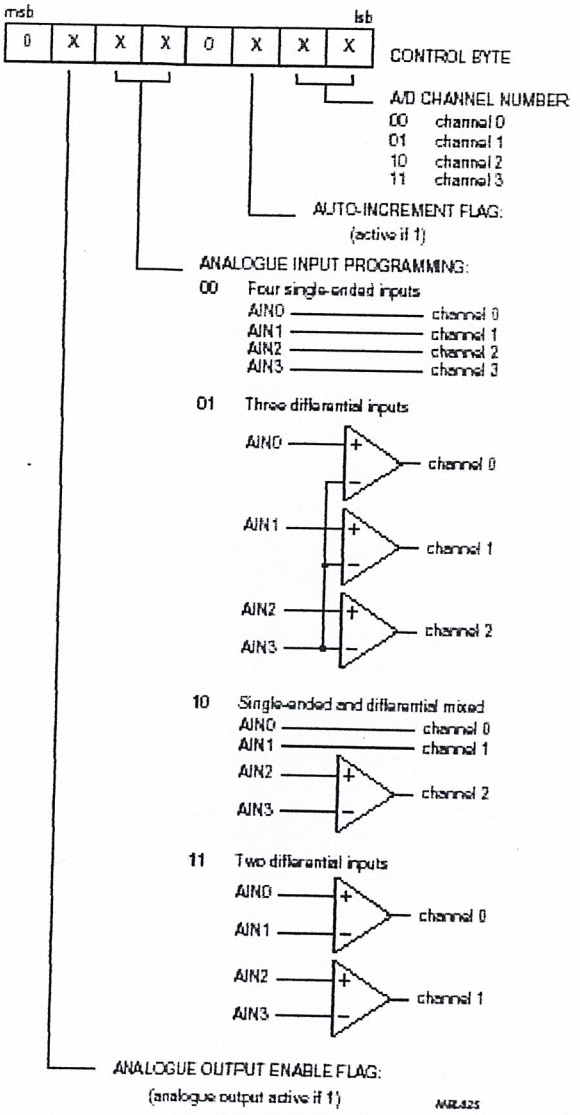
ขา EXT (ขา 12) เป็นขาสำหรับเลือกแหล่งกำเนิดสัญญาณนาฬิกา ถ้าต่อไฟ +5V จะเป็นการเลือกใช้สัญญาณนาฬิกาจากภายนอก โดยต่อสัญญาณนาฬิกาเข้าที่ขา OSC ถ้าต่อขานี้ลงกราวด์ จะเป็นการเลือกใช้สัญญาณนาฬิกาจากภายใน

ขา AGND (ขา13) เป็นขากราวด์ของแรงดันอ้างอิง ปกติต่อลงกราวด์

ขา V_{REF} (ขา 14) เป็นขาสำหรับป้อนแรงดัน ปกติต่อเข้าไฟเลี้ยง +5V

ขา AOUT (ขา 15) เป็นขาเอาท์พุทของวงจรแปลงสัญญาณดิจิตอลเป็นอนาลอก

ขา V_{DD} (ขา 16) เป็นขาต่อไฟเลี้ยง จ่ายได้ตั้งแต่ +2V ถึง +6V ปกติใช้ +5V



รูปที่ 4.6 รายละเอียดข้อมูลควบคุมที่เขียนลงในรีจีสเตอร์ควบคุมภายในไอซี PCF8591

4.3.1 รายละเอียดฟังก์ชันต่างๆของ PCF8591

ตำแหน่งแอดเดรส

ในระบบบัส I^2C การติดต่อกับอุปกรณ์แต่ละตัวต้องระบุแอดเดรสของอุปกรณ์เหล่านั้นอย่างชัดเจน ถ้าเป็นการอ้างอิงแบบ 7 บิต ข้อมูลกำหนดแอดเดรส 4 บิตบนจะเป็นค่าแอดเดรสเฉพาะของอุปกรณ์ตัวนั้นๆที่กำหนดมาจากผู้ผลิต ผู้ใช้งานไม่สามารถเปลี่ยนแปลงได้ สำหรับไอซี PCF8591 จะมีค่าเท่ากับ 1001 (ฐานสอง) ข้อมูล 3 บิตถัดมาจะเป็นค่าของแอดเดรสที่ผู้ใช้งานสามารถกำหนดได้ทางฮาร์ดแวร์เพื่อเลือกใช้ไอซี PCF8591 ที่ต้องการติดต่อด้วยใจกรณีที่มีการต่อใช้งาน PCF8591 มากกว่า 1 ตัว ส่วนบิต LSB ใช้ในการกำหนดว่าต้องการอ่านหรือเขียนข้อมูลกับไอซีตัวนั้นๆ

ข้อมูลควบคุม

หลังจากส่งข้อมูลกำหนดแอดเดรสให้แก่ PCF8591 แล้ว ต้องส่งข้อมูลควบคุมไปด้วยเพื่อกำหนดคุณสมบัติของวงจรแปลงสัญญาณอนาลอกเป็นดิจิตอล และวงจรแปลงสัญญาณดิจิตอลเป็นอนาลอกDACภายใน PCF8591 โดยมีรายละเอียดของข้อมูลในแต่ละบิตดังรูปที่

บิต 6 ของข้อมูลควบคุมใช้สำหรับการเอ็นเนเบิลขาอนาลอกเอาท์พุท เมื่อต้องการเอ็นเนเบิลต้องกำหนดให้ขานี้เป็น “1”

บิต 4 และบิต 5 ของข้อมูลควบคุมใช้สำหรับการกำหนดรูปแบบของสัญญาณอนาลอกอินพุทที่ป้อนให้แก่ PCF8591

บิต 2 ใช้สำหรับเลือกรูปแบบการอ่านข้อมูลจากขาอินพุทอนาลอก ว่าจะเป็นการอ่านจากเพียงอินพุทเดียวหรืออ่านแบบเรียงลำดับทุกอินพุท ถ้าต้องการเลือกให้อ่านเรียงลำดับต้องกำหนดให้บิตนี้เป็น ‘1’

บิต 0 และบิต 1 ใช้สำหรับกำหนดช่องของอินพุทอนาลอกที่ต้องการอ่าน ถ้ากำหนดให้บิต 2 เป็น “1” หลังจากอ่านค่าของบิต “0” และบิต “1” แล้ว ในการอ่านค่าครั้งต่อไปจะเป็นการอ่านค่าอินพุทจากช่องที่ 1

ข้อมูลควบคุมทั้งหมดจะถูกเก็บไว้ในรีจีสเตอร์ควบคุมภายใน PCF8591

เมื่อจ่ายไฟให้แก่ PCF8591 ครั้งแรก บิตต่างๆของข้อมูลภายในรีจีสเตอร์ควบคุมจะเป็น “0”

ออสซิเลเตอร์

วงจรออสซิเลเตอร์ภายใน PCF8591 จะสร้างสัญญาณนาฬิกาสำหรับการแปลงสัญญาณอนาลอกเป็นดิจิตอล เมื่อต้องการใช้วงจรออสซิเลเตอร์ภายใน ขา EXT ต้องต่อลงกราวด์ ถ้าต้องการใช้ออสซิเล

เตอร์จากภายออสซิเลเตอร์นอกขา EXT ต้องต่อเข้ากับไฟบวก และป้อนสัญญาณนาฬิกาเข้าที่ขา OSC ของ PCF8591 โดยความถี่ของสัญญาณนาฬิกาสูงสุดที่ป้อนให้กับออสซิเลเตอร์เท่ากับ 1.25 MHz

4.4 ความรู้เบื้องต้นเกี่ยวกับไมโครคอนโทรเลอร์ PIC16F628

PIC16F628 เป็นไมโครคอนโทรเลอร์ในตระกูล PIC (Peripheral Interface Controller) ของบริษัทไมโครชิป เทคโนโลยี (Microchip Technology) ไมโครคอนโทรเลอร์ในตระกูล PIC มีด้วยกันหลายเบอร์ แต่ละเบอร์ก็จะมีขีดความสามารถแตกต่างกันไป สำหรับในโครงงานนี้ เบอร์ที่มีความเหมาะสมมากที่สุดก็คือ PIC16F628 เนื่องจากมีหน่วยความจำโปรแกรม (Program Memory) แบบแฟลช (Flash) ซึ่งเป็นหน่วยความจำที่สามารถเขียนและลบได้ด้วยสัญญาณไฟฟ้านับพันครั้งสามารถพัฒนาโปรแกรมได้อย่างสะดวก

ไมโครคอนโทรเลอร์ PIC16F628 เป็นไมโครคอนโทรเลอร์ที่จัดอยู่ในไมโครโพรเซสเซอร์ประเภท RISC หรือ Reduced Instruction Set Computer กล่าวคือเป็น ไมโครคอนโทรเลอร์ที่มีคำสั่งน้อยมากคือ 35 คำสั่งพื้นฐานเท่านั้นและทุกคำสั่งจะทำงานเสร็จใน 1 ไซเคิล ทั้งยังทำงานในลักษณะไปป์ไลน์ (Pipe line) เหมือนกับไมโครโพรเซสเซอร์สมัยใหม่ ความเร็วในการทำงานจึงสูงมากเมื่อเทียบกับไมโครคอนโทรเลอร์อื่นที่มีสัญญาณนาฬิกาเท่ากัน

4.4.1 คุณสมบัติทางเทคนิคของ PIC16F628

แบ่งออกเป็น 3 ส่วนคือ หน่วยประมวลผลกลาง (Central Processing Unit: CPU) ส่วนของเพอริเฟอรัล (Peripheral) และคุณสมบัติพิเศษอื่นๆ

1. คุณสมบัติของหน่วยประมวลผลกลางภายใน PIC16F628
   - หน่วยประมวลผลกลางเป็นแบบ RISC
   - มีคำสั่งเพียง 35 คำสั่ง ขนาด 14 บิต
   - ทุกคำสั่งใช้เวลาประมวลผลเพียง 1 ไซเคิลของสัญญาณนาฬิกา ยกเว้นคำสั่งการกระโดดจะใช้เวลา 2 ไซเคิลของสัญญาณนาฬิกา
   - ประมวลผลข้อมูลขนาด 8 บิต
   - มีโหมดการอ้างอิงแอดเดรส 3 โหมดคือ แบบโดยตรง(Direct), ทางอ้อม(Indirect), แบบสัมพันธ์(Relation)
   - มีแหล่งกำเนิดอินเตอร์รัปต์
     1. อินเตอร์รัปต์ภายนอก ที่ขา RB0/INT
     2. จากไทม์เมอร์ TMR0 เมื่อเกิดโอเวอร์โฟลว์

3. ที่พอร์ต B เมื่อเกิดการเปลี่ยนแปลง (RB4-RB7)
4. อินเตอร์รัปต์จาก Computer
5. จาก UART ซึ่งเป็นการเชื่อมต่ออนุกรม
6. จาก CCP (Capture/Compare/PWM)
7. เมื่อไทม์เมอร์ TMR1 เกิดโอเวอร์โฟลว์
8. เมื่อไทม์เมอร์ TMR1 Match

- มีขนาดหน่วยความจำโปรแกรมซึ่งเป็นแบบบแฟลช มีขนาด 2048 เวิร์ด แต่ละเวิร์ดมีขนาด 14 บิต หน่วยความจำอีอีพรอมภายในขนาด 128 ไบต์ และหน่วยความจำแรม 224 ไบต์ซึ่งใช้เป็นรีจีสเตอร์

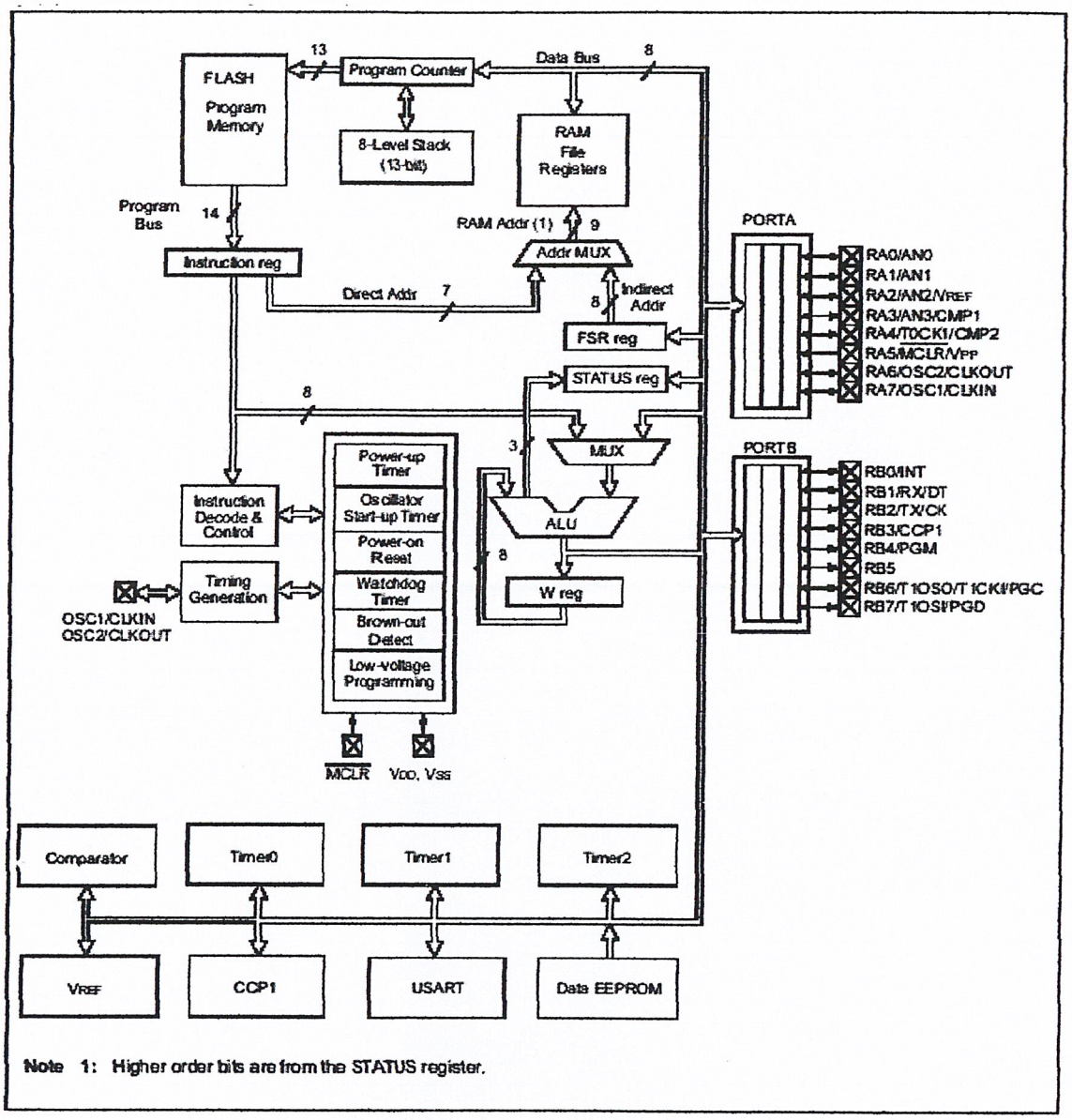
2. คุณสมบัติทางเทคนิคของเพอริเฟอรัล (Peripheral)ใน PIC16F628
   - มีขาอินพุท เอาท์พุท 16 ขา สามารถกำหนดเป็นอินพุท เอาท์พุทได้อิสระ
   - กระแสซิงก์สูงสุด 25mA ต่อขา
   - กระแสซอร์สสูงสุด 20mA ต่อขา
   - มี Timer 3 ตัว
3. คุณสมบัติอื่นๆ
   - สามารถเลือกใช้ออสซิเลเตอร์ทั้งภายในและภายนอก
   - พาวเวอร์ออนรีเซตในตัว (POR)
   - พาวเวอร์อัพ ไทม์เมอร์ (PWRT)
   - บราวน์ เอาท์ รีเซต (BOD)
   - อินเตอร์รัปต์
   - วอตช์ดอกไทม์เมอร์
   - โหมดประหยัดพลังงาน (Sleep Mode)
   - ป้องกันข้อมูลในหน่วยความจำโปรแกรม
   - ไอดี โลเคชั่น
   - สามารถโปรแกรมแบบอนุกรมโดยใช้งานเพียง 2 ขา
   - ย่านไฟเลี้ยง 2.0-5.5 โวลท์

4.4.2 สถาปัตยกรรมของ PIC16F628

ไมโครคอนโทรลเลอร์ PIC16F628 ได้รับการบรรจุหน่วยประมวลผล หน่วยความจำ หน่วยคำนวณทางคณิตศาสตร์ อินพุท เอาท์พุท ไว้และยังมีคุณสมบัติพิเศษอื่นๆที่จำเป็นครบถ้วนสมบูรณ์ ดังแสดงในรูปที่ 4.7 สถาปัตยกรรมของไมโครคอนโทรเลอร์ PIC16F628

PIC16F628 มีการจัดสรรหน่วยความจำดังนี้

หน่วยความจำโปรแกรม เป็นหน่วยความจำแบบแฟลช มีขนาด 2 กิโลเวิร์ด โดย 1 เวิร์ดมีขนาด 14 บิต



Note 1: Higher order bits are from the STATUS register.

Device	Memory		
	FLASH Program	RAM Data	EEPROM Data
PIC16F627	1024 x 14	224 x 8	128 x 8
PIC16F628	2048 x 14	224 x 8	128 x 8
PIC16LF627	1024 x 14	224 x 8	128 x 8
PIC16LF628	2048 x 14	224 x 8	128 x 8

รูปที่ 4.7 สถาปัตยกรรมของไมโครคอนโทรเลอร์ PIC16F6XX

หน่วยความจำอีอีพรอมขนาด 128 ไบต์

หน่วยความจำข้อมูลแรม ทำงานเป็นรีจีสเตอร์ มีขนาด 228 ไบต์

การเข้าถึงหน่วยความจำทั้งหมด ของหน่วยประมวลผลกลาง หรือซีพียู ภายในไมโครคอนโทรลเลอร์นี้ สามารถทำได้ทั้งในลักษณะทางตรง และทางอ้อมและแบบสัมพันธ์ โดยมีรีจีสเตอร์ FSR (File Select Register) ทำหน้าที่ควบคุมการเข้าถึงแบบทางอ้อม

เมื่อไมโครคอนโทรเลอร์ทำงานตามคำสั่งที่กำหนดให้ ข้อมูลของชุดคำสั่งจะถูกนำไปเก็บไว้ในรีจีสเตอร์คำสั่ง (Instruction Register) จากนั้นจะถูกส่งไปยังวงจรถอดรหัสเพื่อทำการควบคุมไทเมอร์ทั้งหมดในตัวไมโครคอนโทรเลอร์ นอกจากนั้นยังส่งไปควบคุมหน่วยคำนวณทางคณิตศาสตร์โดยผ่านวงจรมัลติเพล็กซ์ด้วย

หน่วยคำนวณทางคณิตศาสตร์ (Arithmetic Logic Unit: ALU) มีขนาด 8 บิตสามารถทำการบวก ลบ เลื่อนข้อมูล และประมวลผลทางลอจิก โดยใช้ฟังก์ชันบูลีน ในการทำงาน ALU จะต้องมีรีจีสเตอร์ W ช่วยเสมอ ซึ่งซีพียูจะไม่สามารถเข้าถึง รีจีสเตอร์ W นี้ได้โดยตรง เมื่อ ALU ทำงานจะมีผลต่อบิตทด (Carry Bit) บิตหลัก (Digit Bit) และบิตศูนย์ (Zero Bit) ในรีจีสเตอร์ Status

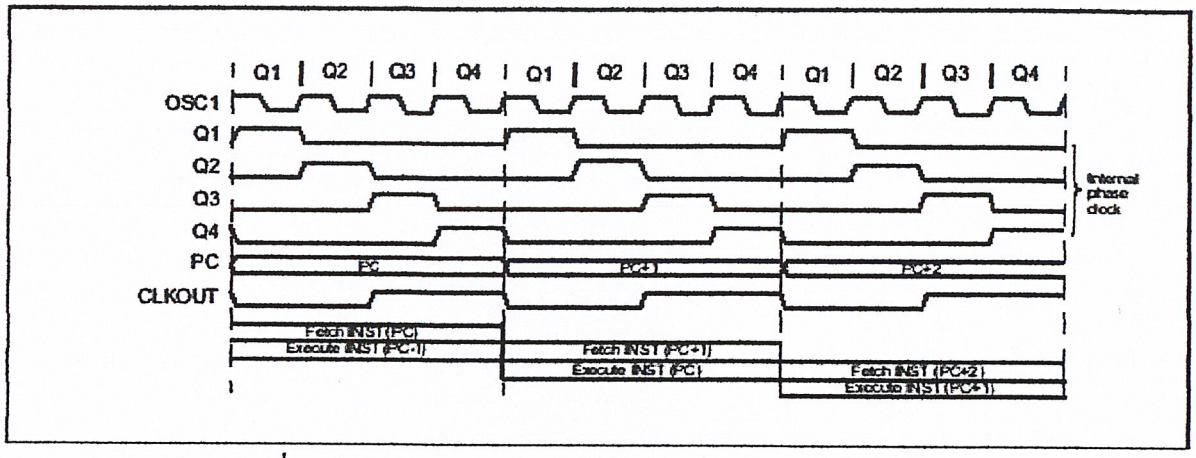
ในส่วนของพอร์ตอินพุทใน PIC16F628 มีด้วยกัน 2 พอร์ต คือ พอร์ต A และ B โดยในพอร์ต A มี 7 บิตและพอร์ต B มี 7 บิต

4.4.3 จังหวะสัญญาณนาฬิกาและไซเคิลการทำงานของ PIC16F628

สัญญาณนาฬิกาอินพุทของ OSC1 จะถูกหารด้วย 4 แล้วแบ่งเป็น 4 ช่วง กำหนดเป็น Q1 Q2 Q3 Q4 โดยโปรแกรมเคาน์เตอร์จะเพิ่มค่าทุกๆสัญญาณ Q1

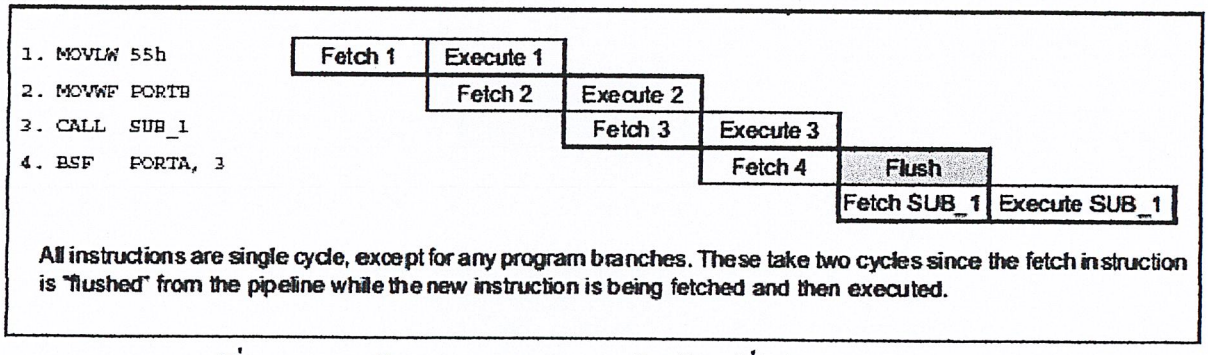
เมื่อโปรแกรมเคาน์เตอร์เพิ่มค่าขึ้น คำสั่งจะถูกเฟตช์จากหน่วยความจำโปรแกรม จากนั้นซีพียูจะทำการแลตช์ข้อมูลคำสั่งนั้นไว้ในรีจีสเตอร์คำสั่ง ที่ช่วงสัญญาณนาฬิกา Q4 คำสั่งจะถูกถอดรหัสและเอ็กซคิวต์ จนเสร็จสิ้นภายในช่วงสัญญาณนาฬิกา Q1-Q4 หรือภายใน 1 ไซเคิลของสัญญาณนาฬิกานั่นเอง

จากรูป 4.8 แสดงให้เห็นความสัมพันธ์ของจังหวะสัญญาณนาฬิกา กับการประมวลผลคำสั่งของไมโครคอนโทรลเลอร์ PIC16F628 จะเห็นว่าในสัญญาณนาฬิกาจะถูกแบ่งออกเป็น 4 ช่วงเพื่อกำหนดการทำงานของซีพียูให้ชัดเจนมากขึ้น



รูปที่ 4.8 ไดอะแกรมเวลาแสดงจังหวะการทำงาน PIC16F628

ไซเคิลการทำงานของ PIC16F628 แบ่งเป็น 2 ไซเคิลคือ เฟตช์และเอ็กซคิวต์ ประกอบด้วย 4 คิวไซเคิล (Q-Cycle: Quadrature Cycle) คือมีลักษณะเป็นไปป์ไลน์ (การทำงานของซีพียูแบบ RISC ที่มีการเหลื่อมกันของขั้นตอนต่างๆ)



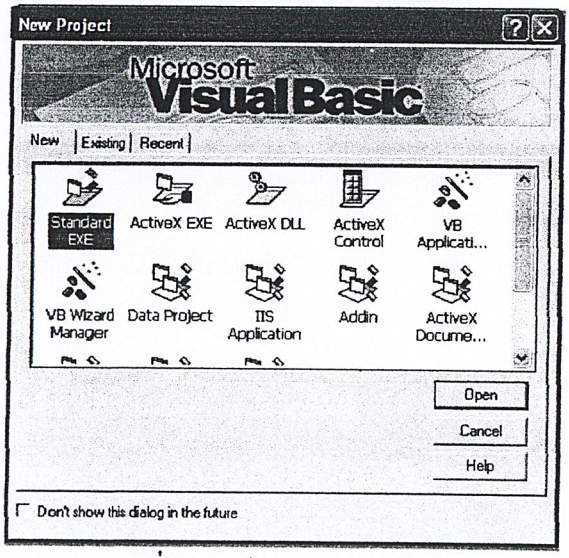
รูปที่ 4.9 แสดงลักษณะการทำงานแบบไปป์ไลน์ที่ใช้ใน PIC16F628

บทที่ 5
การพัฒนาด้านซอฟท์แวร์โดยใช้โปรแกรม Visual Basic 6

สาเหตุที่ใช้โปรแกรม Visual Basic เนื่องจากโปรแกรม Visual Basic นั้นเป็นภาษาคอมพิวเตอร์ที่ได้รับความนิยมนำมาใช้ในการพัฒนาโปรแกรมบน Windows เนื่องจาก เป็นภาษาคอมพิวเตอร์ที่ใช้เทคโนโลยีในลักษณะ Visualize ซึ่งเพียงแต่ เลือก Control ที่เหมาะสม แล้ววาดลงบน Form ก็สามารถสร้างจอภาพที่ใช้สำหรับติดต่อกับผู้ใช้ รวมทั้งการใช้เทคนิคการเขียนโปรแกรมแบบ Even-driven ซึ่งเป็นการเขียนโปรแกรมเพื่อกำหนดขั้นตอนการทำงานให้กับ Control ต่างๆที่สร้างขึ้นตามเหตุการณ์ (Even) ต่างๆที่เกิดขึ้น เช่นการเลื่อนเมาท์ หรือการรับส่งข้อมูลจากคีย์บอร์ด ฯลฯ เป็นต้น ประกอบกับภาษาที่ใช้เขียนโปรแกรมเป็นภาษาเบสิก(Basic) ซึ่งเป็นภาษาคอมพิวเตอร์ที่ผู้ใช้งานคอมพิวเตอร์ส่วนบุคคลส่วนใหญ่คุ้นเคย จึงส่งผลให้การพัฒนาโปรแกรมบนวินโดวส์ (Windows) ด้วยวิชวลเบสิก(Visual Basic) มีขั้นตอนน้อย กระทำได้ง่าย และสะดวกต่อการใช้งาน จึงทำให้ผู้ใช้สามารถเรียนรู้ได้ภายในเวลา 2-3 ชั่วโมง ก็สามารถพัฒนาโปรแกรมบนวินโดวส์ ขึ้นเป็นโปรแกรมแรกได้

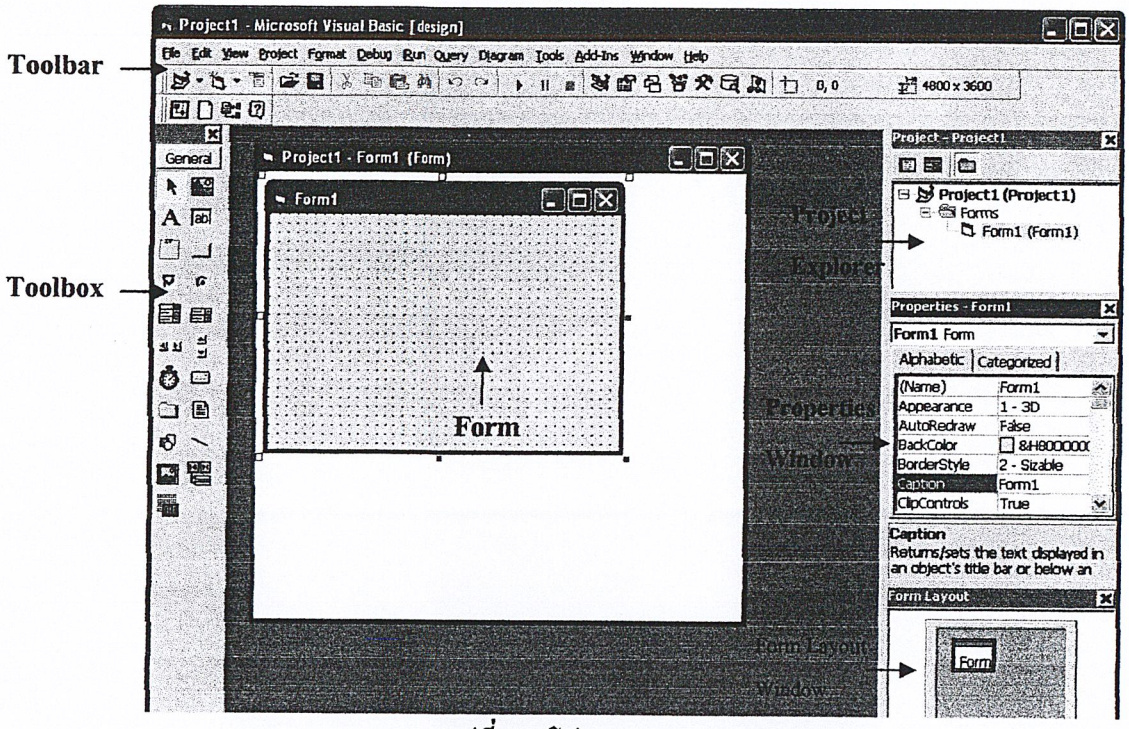
5.1 โปรแกรม Visual Basic

สิ่งแรกเมื่อเข้าสู่โปรแกรม Microsoft Visual Basic 6.0 ได้แก่ จอภาพที่ใช้สำหรับเปิด Project (Project จะเป็นชื่อเรียกแทนระบบงานที่พัฒนาขึ้นด้วย Visual Basic)



รูปที่ 5.1 หน้าต่าง New Project

ในเบื้องต้นให้เลือก Icon "Standard EXE" ใน Tab "New" เพื่อเข้าสู่จอภาพของ Visual Basic ที่ใช้ในการพัฒนาโปรแกรม ดังรูป 5.2



รูปที่ 5.2 โปรแกรม Visual Basic

- **Form** เป็นส่วนที่ใช้สำหรับสร้างจอภาพของโปรแกรมขึ้นใช้งาน
- **Toolbox** เป็นส่วนที่ประกอบด้วย Icon ต่างๆซึ่งนี้ Visual Basic เรียกว่า "Control" ที่จะนำไปใช้งานโดยการ นำลงไปวางบน Form
- **Toolbar** เป็นส่วนที่ประกอบด้วย Icon ต่างๆที่ใช้ในการพัฒนาโปรแกรม
- **Project Explorer Window** เป็นส่วนสำหรับเรียก Form ต่างๆขึ้นมาแก้ไข ในกรณีที่ Project ประกอบด้วย Form มากกว่า 1 Form
- **Properties Window** เป็นจอที่ใช้สำหรับกำหนดคุณสมบัติ (property) ให้กับ Form และ Object ต่างๆที่ปรากฏอยู่ใน Form
- **Form Layout Window** ใช้สำหรับดูตำแหน่งของ Form บนจอภาพ ทำให้จัดตำแหน่งของ Form ได้สะดวกขึ้น

5.2 การสร้างและออกแบบจอภาพ

Visual Basic เป็นภาษาคอมพิวเตอร์ภาษาหนึ่งที่มี Tool สำหรับสร้างจอภาพด้วยเทคนิคแบบ Visual Basic กล่าวคือ ในการสร้างจอภาพด้วย Visual Basic จะเป็นเพียงการนำ Control ต่างๆที่ Visual Basic เตรียมไว้ให้ ไปวาดไว้บน Form ตามความต้องการซึ่งด้วยวิธีนี้การสร้างจอภาพจึงเป็นเรื่องที่ง่าย และใช้เวลาน้อยซึ่งต่างจากโปรแกรมในลักษณะเดิม ที่ต้องใช้เวลาค่อนข้างมาก ในการสร้างแต่ละจอภาพขึ้นใช้งาน

Control ต่างๆใน Toolbox ซึ่งใช้ในการสร้างจอภาพ จะมีชื่อเรียกและการทำงานแตกต่างกันไป การจะเลือก Control ใดมาใช้งานนั้นขึ้นอยู่กับการออกแบบจอภาพของนักพัฒนาโปรแกรม และความเหมาะสมในการใช้งาน

หลังจากที่วาด Control ลงบน Form แล้วขั้นตอนต่อไปในการพัฒนาโปรแกรมด้วย Visual Basic ได้แก่ การกำหนดคุณสมบัติ (ต่อไปนี้จะเรียกว่า "Property") ให้กับแต่ละ Object ที่ปรากฏอยู่บน Form ซึ่งแต่ละ Object จะมี Propertyทั้งที่เหมือนกับ Object อื่นๆ และ Propertyประจำตัว การกำหนด Property ให้กับ Objectจะกระทำตั้งแต่เริ่มวาด Objectลงบน Form หรืออาจเปลี่ยนแปลงโดยการเขียนโปรแกรมก็ได้

<u>การ Run และเลิกงาน Project</u>

ในการ Run Project ที่พัฒนาขึ้นด้วย Visual Basicสามารถ Run ทั้งโดยใช้ Interpreter และการใช้ Complierกล่าวคือ เราสามารถทดลอง Run สิ่งต่างๆที่เราทำขึ้นพร้อมๆกับการแก้ไขโปรแกรมจนกระทั่งเสร็จสมบูรณ์ แล้วจึง Compile โปรแกรมให้อยู่ในรูปของ Executed Program เพื่อนำไปใช้งาน

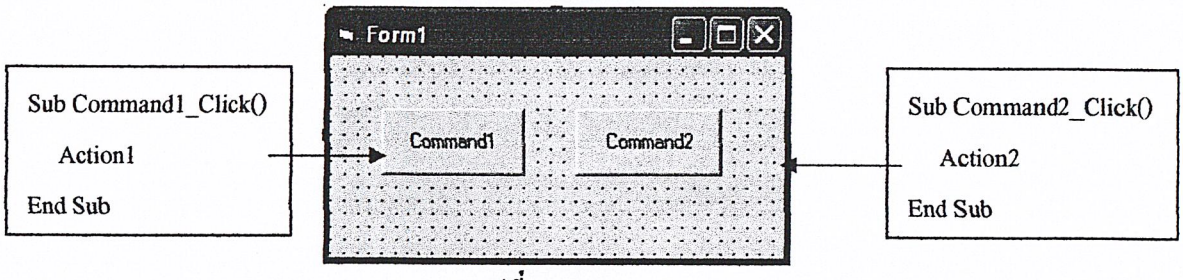
ในการ Run Project ทำได้ 3 วิธี คือ
วิธีที่ 1 โดยการกด F5
วิธีที่ 2 คลิกที่ Icon "Run" ใน Toolbar
วิธีที่ 3 เลือกจากเมนู Run และ Start ตามลำดับ

วิธีการเลิกงาน Project ทำได้ 2 วิธี คือ
วิธีที่1 คลิกที่ Icon"End" ใน Toolbar
วิธีที่2 เลือกจากเมนู Run และ End ตามลำดับ

<u>Procedure และ Function ใน Visual Basic</u>

Visual Basic จะแบ่งโปรแกรมออกเป็นส่วนๆที่เรียกว่า "โมดูล" โดยจะแบ่งโปรแกรมออกป็น "Procedure" และ "Function" แต่เนื่องจากการกำหนดการทำงานของโปรแกรมจะกำหนดตาม Even ที่เกิดขึ้นกับ Object ดังนั้นการแบ่ง Procedure ใน Visual Basic จึงแบ่งตามชื่อ Object และ Even เช่น การเขียนโปรแกรมให้กับ Object สมมุติชื่อ "Command1" ซึ่งเป็นปุ่มบนจอภาพภายใต้ Even กดปุ่ม ก็จะ

เขียนเป็น Procedure หนึ่งกับอีก Object หนึ่งเป็น Object ประเภทเดียวกันแต่คนละชื่อ สมมุติชื่อ "Command2" ภายใต้Even กดปุ่มเหมือนกันก็ต้องเขียนเป็นอีกProcedure หนึ่งถึงแม้จะเป็นEven ที่เกิดกับObject คนละตัว ดังรูป 5.3



รูปที่ 5.3 Form

ชื่อของ Procedure ใน Visual Basic จะอยู่ในรูปแบบดังนี้

[Public I Private] Sub *name* [(*arglist*)]

...

End Sub

โดยที่ Private Sub จะเป็นคำเฉพาะที่ใช้บอกว่าเป็น Procedure

name คือ ชื่อของ Procedure

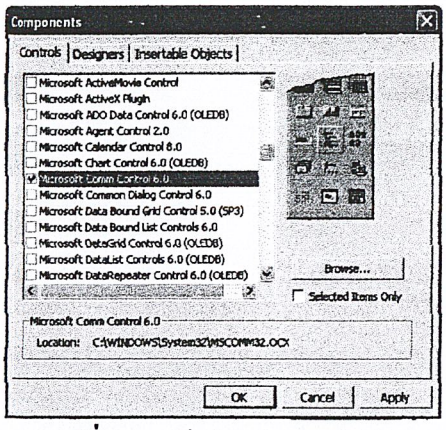
arglist คือ รายชื่อ Argument ที่ใช้ภายใน Procedure นั้น

End Sub คือ เป็นคำเฉพาะที่ใช้สำหรับจบการทำงานของแต่ละ Procedure

คำสั่ง Public ใช้ในกรณีที่ต้องการให้ Procedure นั้นสามารถถูกเรียกใช้โดย Procedure อื่นที่สร้างขึ้นใน Form อื่น ส่วนคำสั่ง Private จะใช้ในกรณีที่ต้องการให้ Procedure นั้นสามารถถูกเรียกใช้ได้เฉพาะ Procedure ที่สร้างขึ้นใน Form เดียวกับ Procedure นั้น

5.3 การเขียนโปรแกรมเพื่อใช้งานพอร์ตอนุกรม

สำหรับการใช้งาน Visual Basic ตั้งแต่เวอร์ชั่น 2 เป็นต้นมาVisual Basic จะมีคัสตอมคอนโทรลสำหรับการสื่อสารอนุกรมผ่านทางพอร์ตอนุกรมของคอมพิวเตอร์มาให้ โดยทำการเลือกเมนู Project →Components แล้วจะปรากฏหน้าต่าง Components ขึ้นมา จากนั้นทำการเลือกไปที่ Microsoft Comm Control 6.0 ดังรูป 5.4



รูปที่ 5.4 หน้าต่าง Components

MSComm จัดเตรียมทางเลือกไว้ 2 ทางเพื่อความสะดวกในการสื่อสารข้อมูล ทางแรกคือ การสื่อสารข้อมูลที่กระตุ้นด้วยเหตุการณ์ (even-driven communications) เป็นรูปแบบที่มีประสิทธิภาพมากสำหรับการตอบสนองแบบทันทีทันใด ส่วนทางเลือกที่ 2 เป็นการตรวจสอบค่าเหตุการณ์ และความผิดพลาดที่เกิดขึ้นด้วยการดูค่าที่เปลี่ยนแปลงภายในคุณสมบัติ CommEven หลังจากให้โปรแกรมทำงานในฟังก์ชันต่างๆไปเรียบร้อยแล้ว ซึ่งวิธีนี้ใช้งานได้ดีในกรณีที่โปรแกรมมีขนาดเล็ก คอนโทรล MSComm 1 ตัว สามารถควบคุมการทำงานของพอร์ตอนุกรมได้ 1 พอร์ตเท่านั้น

คุณสมบัติ (Property) ของคอนโทรล MSComm นั้นมีมากมาย จึงจะขออธิบายเป็นบางส่วนดังนี้

- CommPort

ใช้ในการกำหนดและอ่านค่าพอร์ตอนุกรมที่ติดต่ออยู่ (COM1, COM2, COM3, COM4)

รูปแบบการใช้งาน
Object.CommPort [=value]

โดย Value เป็นค่าของพอร์ตอนุกรม ชนิดของข้อมูลเป็น Integer ค่า Value สามารถกำหนดได้ในช่วง 1-16 (ค่าเริ่มต้นกำหนดไว้ที่ 1)

- Setting

ใช้กำหนดและอ่านค่าอัตราบอดเรต พาริตี้ จำนวนของบิตข้อมูล จำนวนของบิตปิดท้าย

รูปแบบการใช้งาน
Object. Setting [=value]

ค่า Value มีชนิดข้อมูลเป็นแบบ String มีรูปแบบเป็น “BBBB, P, D, S” โดย BBBB เป็นค่าอัตราบอดเรต, P เป็นค่าพาริตี้, D เป็นจำนวนบิตข้อมูล และ S เป็นจำนวนบิตปิดท้าย ปกติแล้วค่านี้ถูกกำหนดไว้เป็น “9600, N, 8, 1”

- PortOpen

ใช้ในการกำหนดและอ่านค่าสถานะของพอร์ตอนุกรม เพื่อเปิดและปิดพอร์ตอนุกรม

รูปแบบการใช้งาน
Object.PortOpen [=value]

ค่า Value มีชนิดข้อมูลเป็นแบบบูลีนคือ True กับ False โดย True หมายถึงการเปิดพอร์ตอนุกรม และ False หมายถึงการปิดพอร์ตอนุกรม สำหรับการปิดพอร์ตนั้นจะมีการเคลียร์บัฟเฟอร์รับข้อมูลและบัฟเฟอร์ส่งข้อมูลด้วย คอนโทรล MSComm จะปิดพอร์ตอนุกรมโดยอัตโนมัติเมื่อออกจากโปรแกรม ถ้าพอร์ตอนุกรมนั้นถูกเปิดไว้แล้ว โปรแกรมก็จะแจ้งความผิดพลาดออกมา

- Input

อ่านค่าและลบค่าขบวนข้อมูลจากบัฟเฟอร์ภาครับ

รูปแบบการใช้งาน
Object.Input

- InBufferCount

ส่งค่าจำนวนตัวอักษรที่อยู่ในบัฟเฟอร์ภาครับ

รูปแบบการใช้งาน
Object.InBufferCount [=value]

คำสั่ง InBufferCount จะแสดงค่าจำนวนตัวอักษรซึ่งรับมาจากภายนอกและยังเก็บอยู่ในบัฟเฟอร์ภาครับ เพื่อให้ผู้ใช้งานอ่านค่าออกไป สำหรับการเคลียร์ค่าบัฟเฟอร์ภาครับทำได้โดยกำหนดให้ InBufferCount มีค่าเป็น 0

- InBufferSize

กำหนดและคืนค่าขนาดขอภาคของบัฟเฟอร์ภาครับในหน่วยเป็นไบต์

รูปแบบการใช้งาน
Object.InBufferSize [=value]

ค่าเริ่มต้นของ InBufferSize จะถูกกำหนดไว้ที่ 1,024 ไบต์

- InputLen

กำหนดค่าและคืนค่าจำนวนของตัวอักษรที่อ่านจากบัฟเฟอร์ภาครับ

รูปแบบการใช้งาน
Object.InputLen [=value]

ค่าเริ่มต้นของคุณสมบัติ InputLen มีค่าเท่ากับ "0" การกำหนดค่าเท่ากับ "0" จะทำให้คำสั่ง Input ของ MSComm อ่านค่าข้อมูลที่อยู่ภายในบัฟเฟอร์ภาครับทั้งหมด

- InputMode

กำหนดค่าและคืนค่าชนิดของข้อมูลที่รับโดยคำสั่ง Input

รูปแบบการใช้งาน
Object.InputMode [=value]

คุณสมบัติ InputMode ใช้กำหนดว่าข้อมูลชนิดไหนที่รับเข้ามาผ่านคำสั่งInput โดยข้อมูลจะเลือกได้ 2 ประเภทคือ

- ComInputModeText สำหรับข้อมูลที่อยู่ในรูปข้อความตัวอักษรตามมาตรฐาน ANSI โดยจะต้องกำหนดค่าเป็น "0" และค่าเริ่มต้นของการรับข้อมูลจะมีค่านี้
- ComInputModeBinary สำหรับข้อมูลอื่นๆซึ่งเก็บในรูปไบนารี่รวมกันอยู่เป็นไบต์ข้อมูล

- Output

ใช้ในการส่งขบวนของข้อมูลไปยังบัฟเฟอร์ส่งข้อมูล

รูปแบบการใช้งาน
Object.Output [=value]

ค่า Value เป็นค่าของตัวอักษรที่เขียนไปยังบัฟเฟอร์ส่งข้อมูล คุณสมบัติ Output สามารถใช้ในการส่งข้อมูลตัวอักษรหรือข้อมูลไบนารี่ก็ได้

- DTREnable

ใช้ในการกำหนดสถานะของขา Data Terminal Ready (DTR) ชนิดข้อมูลเป็นแบบบูลีน

รูปแบบการใช้งาน
Object.DTREnable [=value]

โดย True หมายถึง มีสถานะเป็นลอจิก "1"

False หมายถึง มีสถานะเป็นลอจิก "0"

- RTSEnable

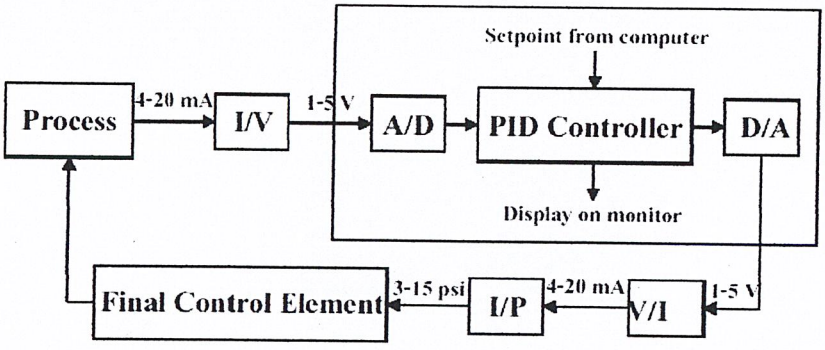
ใช้เพื่อกำหนดสถานะลอจิกให้ขา Request To Send (RTS) ชนิดข้อมูลเป็นแบบบูลีน

รูปแบบการใช้งาน
Object.RTSEnable [=value]

มีลักษณะเช่นเดียวกับ DTREnable

บทที่ 6
การออกแบบตัวควบคุม

การออกแบบตัวควบคุมที่ใช้กับระบบควบคุมระดับน้ำที่ได้สร้างขึ้นนี้ ใช้คอมพิวเตอร์ส่วนบุคคล(PC) ทำหน้าที่ควบคุม ซึ่งออกแบบตัวควบคุมโดยใช้โปรแกรมวิชวลเบสิก ให้สามารถบันทึกผลการทดลอง วิเคราะห์ และปรับค่าพารามิเตอร์ที่เหมาะสมกับกระบวนการได้ สัญญาณตอบสนองที่ได้จากกระบวนการจะถูกส่งมาในลักษณะแรงดันไฟฟ้า 1-5V ผ่านวงจรอินเตอร์เฟสที่สร้างขึ้น แปลงสัญญาณอนาลอกเป็นดิจิตอล และส่งไปยังคอมพิวเตอร์ทางพอร์ตอนุกรม เพื่อทำการวิเคราะห์ และส่งค่าควบคุมกลับมายังวงจรอินเตอร์เฟส ไมโครคอนโทรลเลอร์จะทำหน้าที่รับสัญญาณจากคอมพิวเตอร์ส่วนบุคคลและส่งไปยังไอซี PCF8591 เพื่อทำการแปลงสัญญาณดิจิตอลเป็นอนาลอก 1-5V หลังจากนั้นก็จะใช้วงจร V/I ทำการแปลงสัญญาณที่ได้อีกครั้ง เพื่อให้ได้สัญญาณมาตรฐาน 4-20 mA ก่อนส่งไปยังอุปกรณ์ I/P Converter เพื่อปรับเป็นสัญญาณแรงดันลม 3-15 psi และจ่ายเป็นสัญญาณควบคุมคอนโทรลวาล์ว เพื่อควบคุมกระบวนการ



รูปที่ 6.1 แสดงบล็อกไดอะแกรมของระบบ

ในการทดลองสามารถเลือกใช้งานได้ 3 โหมดคือ

1. SLCD Indicating Controller

ในโหมดนี้จะใช้ตัวควบคุมภายนอก คือ SLCD Indicating Controller เป็นตัวควบคุม โดยคอมพิวเตอร์จะทำหน้าที่เป็นเพียงอุปกรณ์แสดงผล บันทึกผลการทดลอง และสามารถคำนวณทรานเฟอร์ฟังก์ชันได้

2. Manual Control

ในโหมดนี้เป็นโหมดที่ใช้คอมพิวเตอร์เป็นตัวควบคุม โดยแบ่งเป็น 2 โหมดย่อย คือ M Mode และ A Mode ในส่วนของ M Mode คอมพิวเตอร์จะรับค่าจากผู้ใช้งาน ในการเปิด-ปิดวาล์วโดยตรง ผู้ใช้สามารถเปิด-ปิดวาล์วได้ตั้งแต่ 0-100% บนหน้าจอ Plant Graphic ส่วนในโหมด A Mode คอมพิวเตอร์จะทำหน้าที่เป็นตัวคำนวณสัญญาณควบคุมที่จะส่งไปยังคอนโทรลวาล์ว โดยใช้สมการ PID ทั้งนี้ผู้ใช้จะต้องทำการตั้งค่าพารามิเตอร์ *PB, Ti, Td* ให้กับตัวคอนโทรลเลอร์ก่อน และระหว่างการทำงานผู้ใช้สามารถปรับเปลี่ยนโหมดการทำงาน ระหว่าง M Mode และ A Mode ได้หากเกิดการผิดพลาดหรือควบคุมไม่ได้

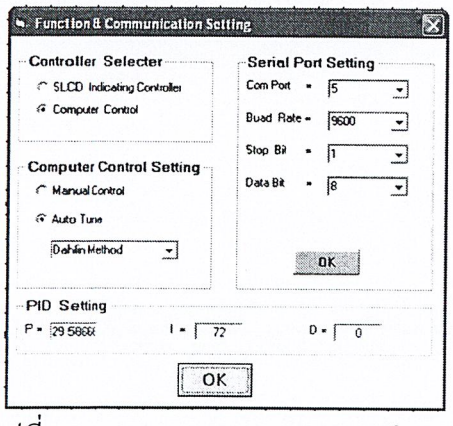
(*หมายเหตุ:* M Mode นิยมใช้ในกรณีเริ่มต้นการทำงานของระบบใหม่ ซึ่งยังไม่ทราบค่าพารามิเตอร์ที่เหมาะสม เพื่อปรับสัญญาณที่ส่งไปควบคุมระบบด้วยมือ หลังจากนั้นก็จะนิยมใช้ A Mode ในการควบคุม)

3. Autotune Mode

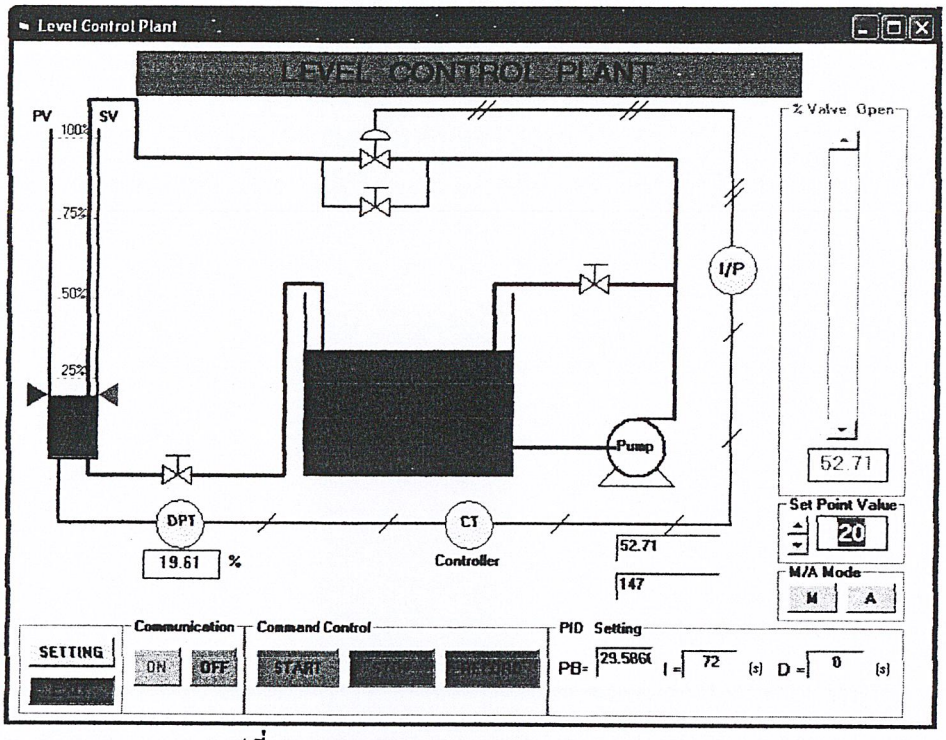
ในโหมดนี้เป็นการคำนวณค่าพารามิเตอร์ *PB, Ti, Td* แบบอัตโนมัติ ให้กับตัวควบคุม PID โดยใช้วิธีของ Ziegler-Nichols หรือวิธีของ Dahlin ซึ่งก่อนใช้งานในโหมดนี้ผู้ใช้ต้องทำการหาพารามิเตอร์ (t_0, K, T) ของทรานเฟอร์ฟังก์ชัน โดยใช้ M Mode ก่อน บนฟอร์ม Graph Recorder ทำการคำนวณค่าพารามิเตอร์ของทรานเฟอร์ฟังก์ชัน ด้วยวิธี 2 Point-Method (Fit 3)

(*หมายเหตุ:* ผู้ใช้งานสามารถปรับค่าพารามิเตอร์ต่างๆของระบบได้ หากค่าพารามิเตอร์ที่ตัวคอนโทรลเลอร์คำนวณให้ไม่เหมาะสมกับระบบ)

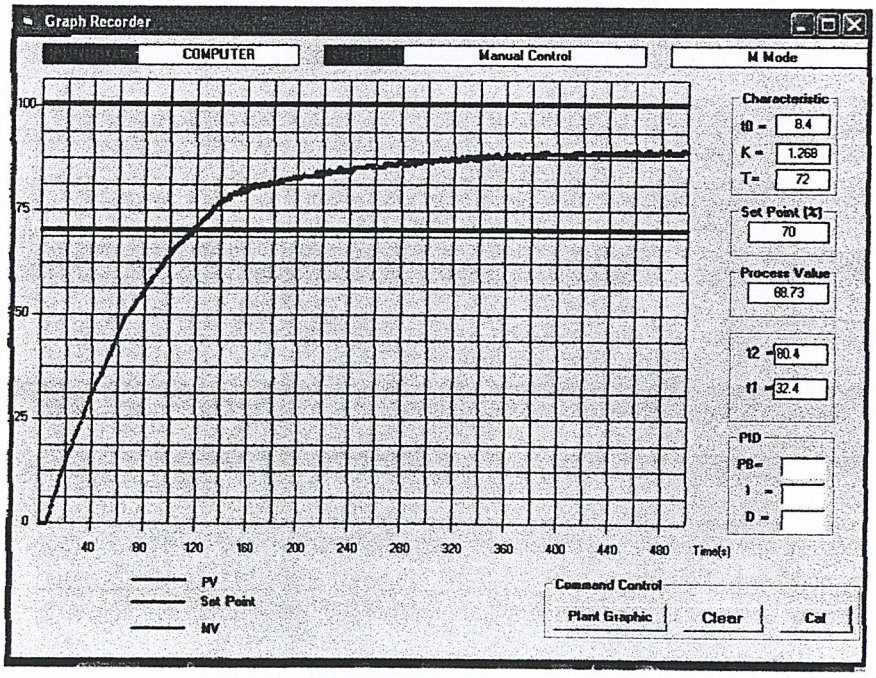
6.1 จอภาพควบคุมการทำงานของการควบคุมระดับน้ำในถัง



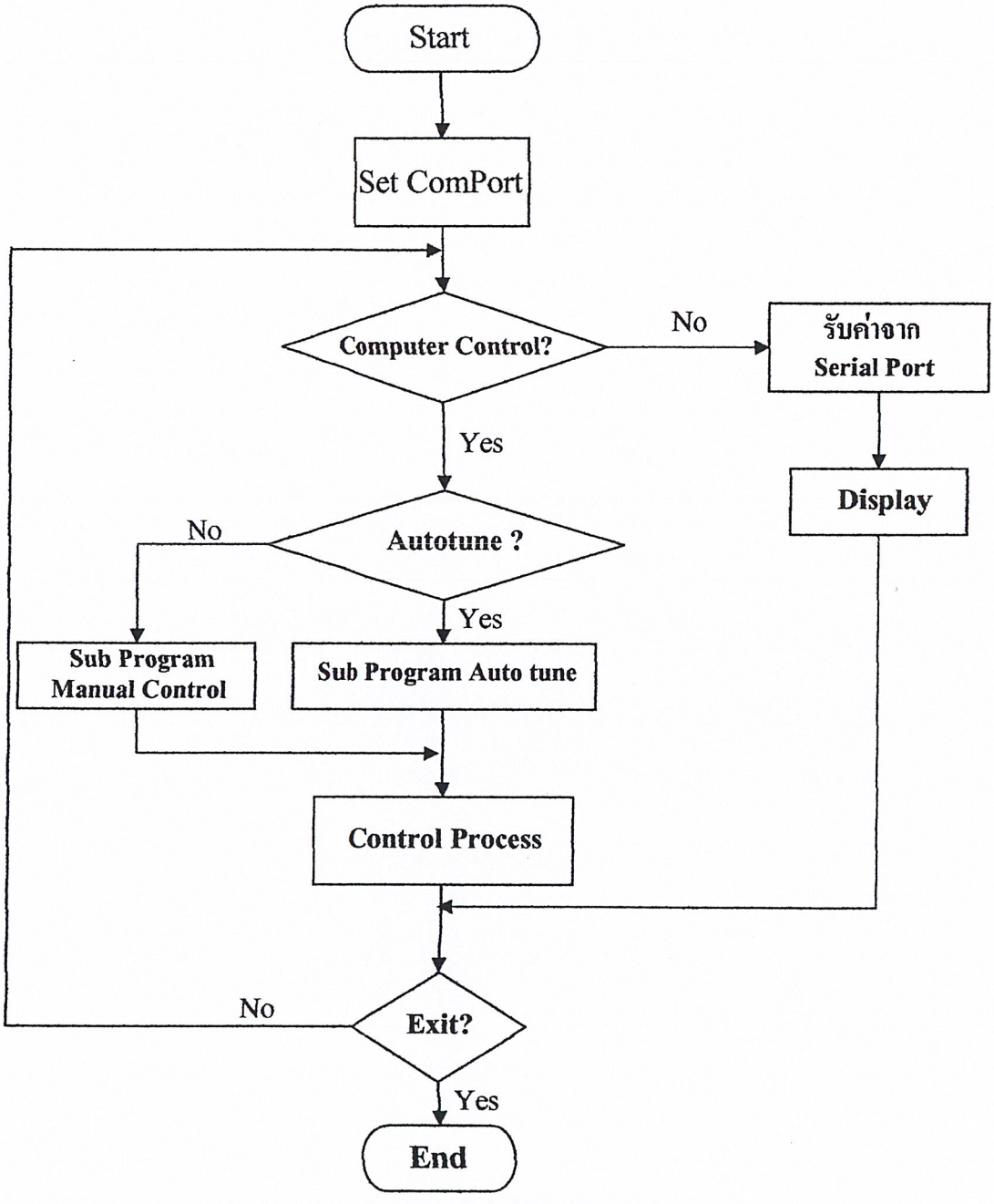
รูปที่ 6.2 จอภาพ Setting ควบคุมการทำงาน



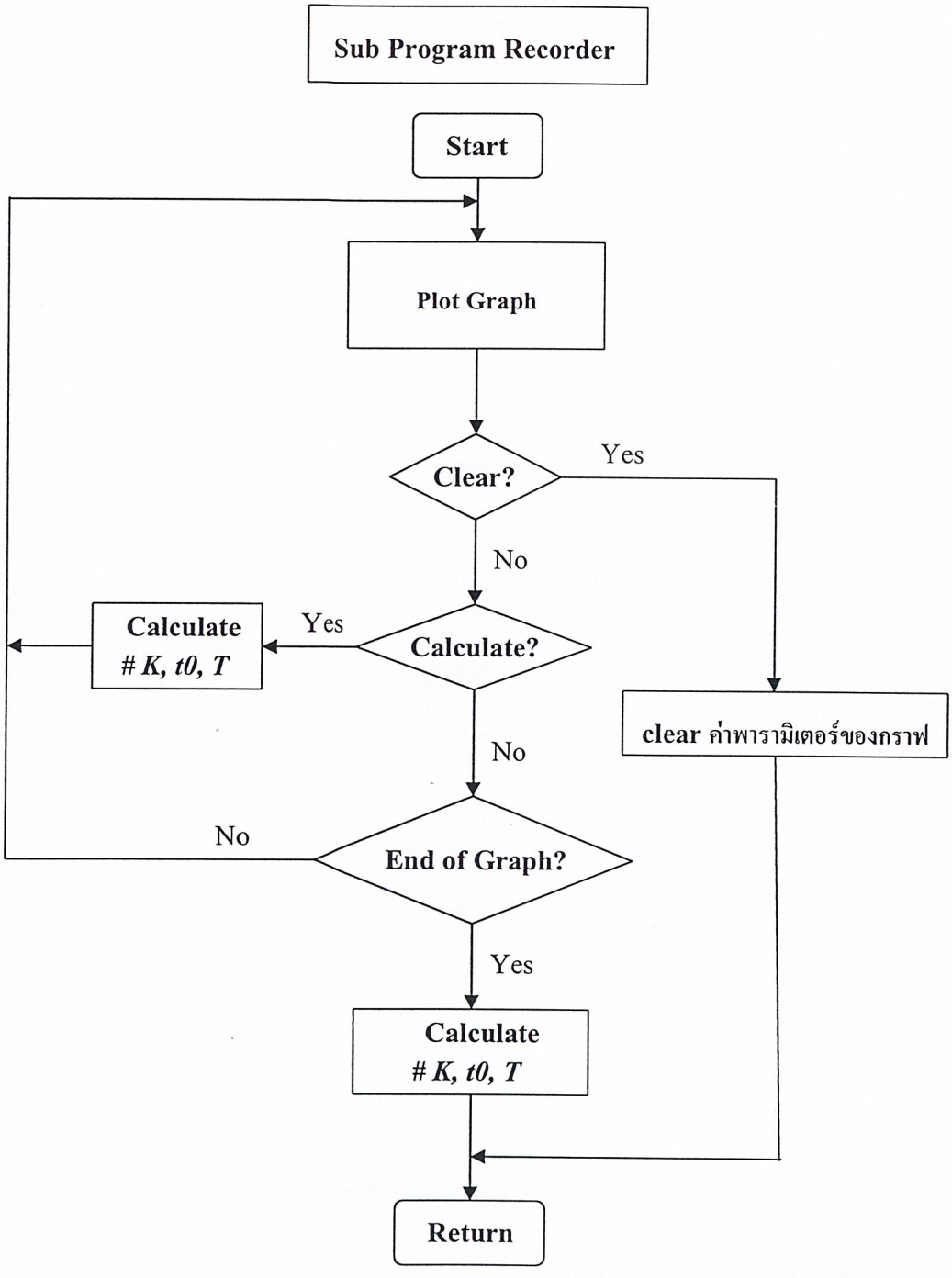
รูปที่ 6.3 จอภาพแสดงการทำงานของ Process



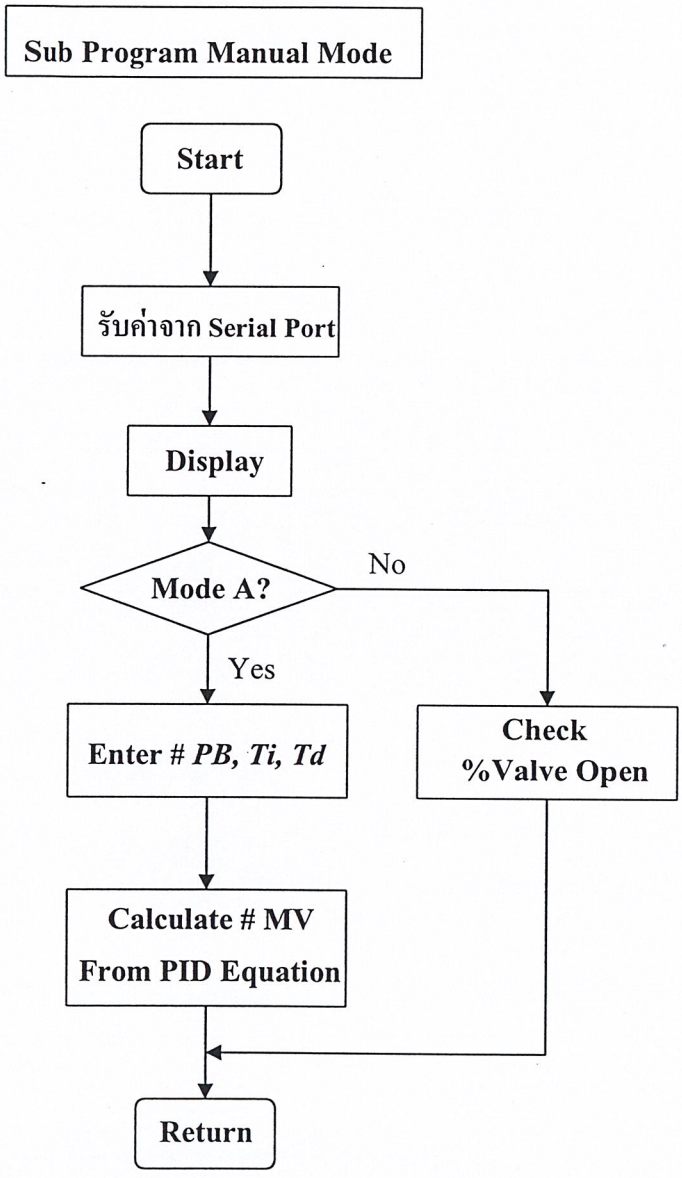
รูปที่ 6.4 จอภาพแสดงการบันทึกผลการทดลองและคำนวณค่าพารามิเตอร์



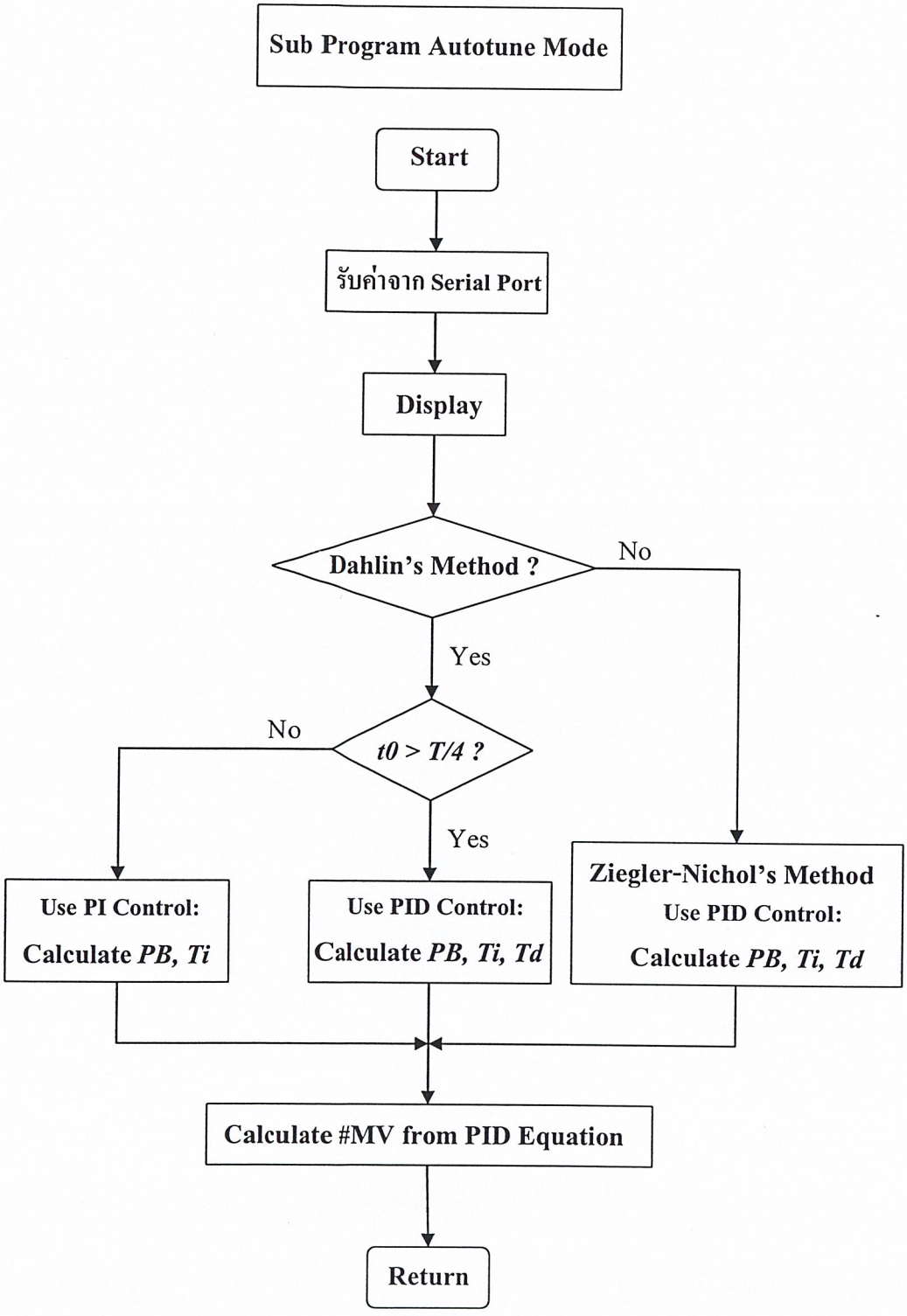
รูปที่ 6.5 แผนผังการควบคุมกระบวนการ



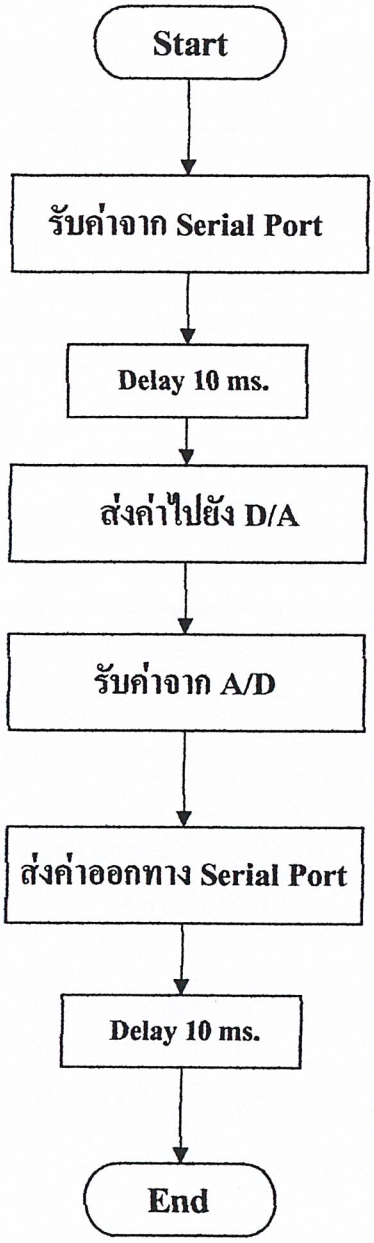
รูปที่ 6.6 แผนผังโปรแกรมย่อย Recorder



รูปที่ 6.7 แผนผังโปรแกรมย่อย Manual Mode



รูปที่ 6.8 แผนผังโปรแกรมย่อย Autotune Mode



รูปที่ 6.9 แผนผังโปรแกรม PIC16F628

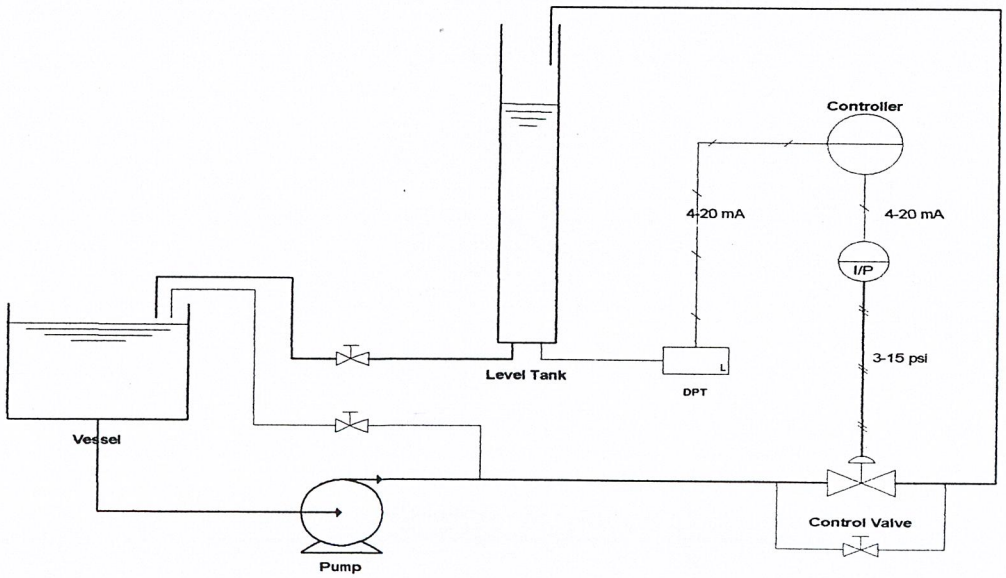
บทที่ 7
การทดลองและผลการทดลอง

7.1 กล่าวนำ

ในบทนี้เป็นการนำคอมพิวเตอร์ส่วนบุคคลมาทำการทดลอง วิเคราะห์ค่าพารามิเตอร์ของตัวควบคุม PID เพื่อนำไปใช้กับกระบวนการควบคุมระดับน้ำ ทำการทดลองควบคุมระดับน้ำที่ค่าเป้าหมายต่างๆ และใส่สิ่งรบกวน (Disturbance) มารบกวนระบบ ทำการทดลองเปรียบเทียบผลการใช้ค่าพารามิเตอร์ของตัวควบคุม โดยใช้วิธีของ Dahlin และZiegler-Nichols ทดลองปรับปรุงค่าพารามิเตอร์จนได้ผลตอบสนองต่อกระบวนการที่ดีที่สุด และเปรียบเทียบผลการทดลอง โดยใช้คอมพิวเตอร์ส่วนบุคคลเป็นตัวควบคุมกับการใช้ SLCD Indicating Controller

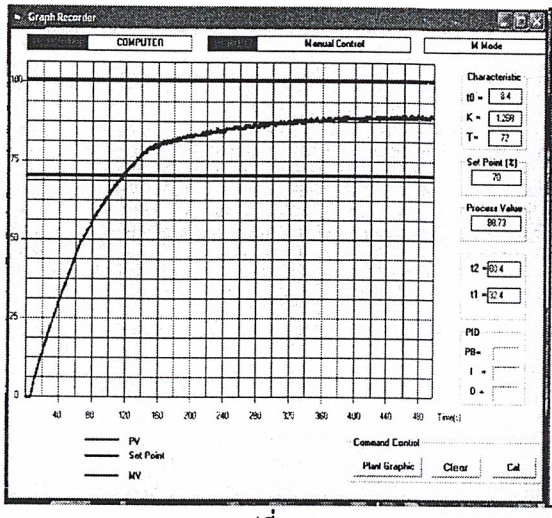
7.2 ผลการทดลองกับกระบวนการควบคุมระดับน้ำ

การทดลองนี้จะทำการวิเคราะห์เพื่อหาค่าพารามิเตอร์ของตัวควบคุมแบบ PID โดยจะทดลองทั้งวิธีของDahlin และZiegler-Nichols กระบวนการควบคุมระดับน้ำที่ใช้ทดลอง มีโครงสร้างแสดงได้ดังรูปที่ 7.1

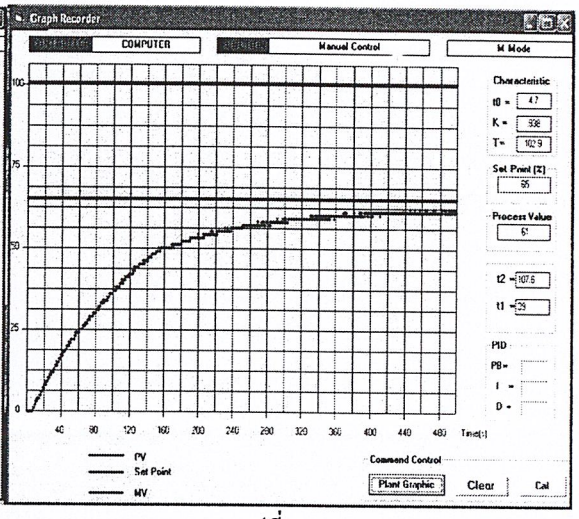


รูปที่ 7.1 โครงสร้างของการกระบวนการควบคุมระดับน้ำ

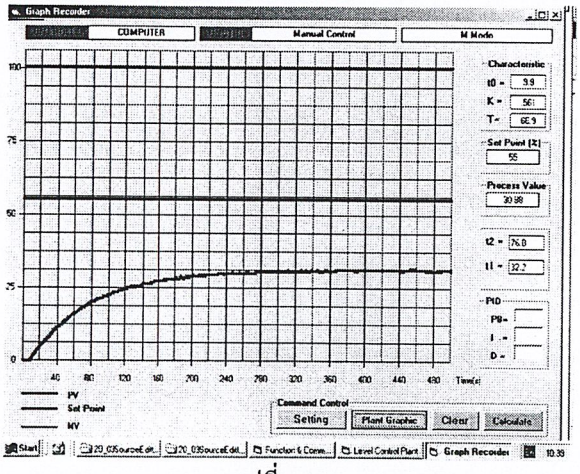
7.2.1 ทดลองหาค่าพารามิเตอร์ของทรานเฟอร์ฟังก์ชันของกระบวนการโดยป้อนสัญญาณระดับที่ค่า 70%, 65% และ55% ได้ผลการทดลองดังรูปที่ 7.2 ก, 7.2 ข และ 7.2 ค ตามลำดับ



รูปที่ 7.2 ก



รูปที่ 7.2 ข

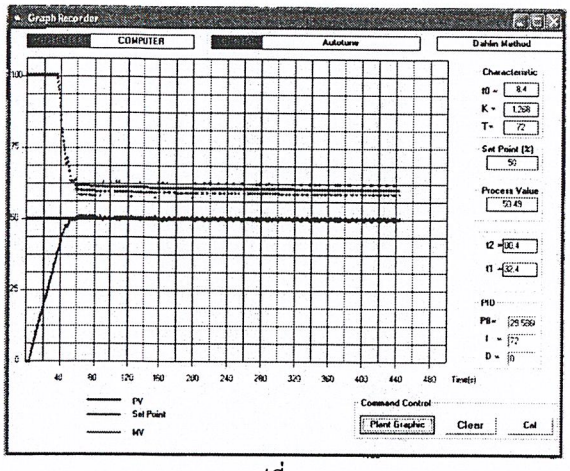


รูปที่ 7.2 ค

จากผลการทดลองที่ได้เลือกใช้ค่าพารามิเตอร์ ของทรานเฟอร์ฟังก์ชัน ที่หาได้จากกระบวนการโดยที่ป้อนสัญญาณระดับที่ 70 % มาใช้ในการทำการทดลองต่อ คือ

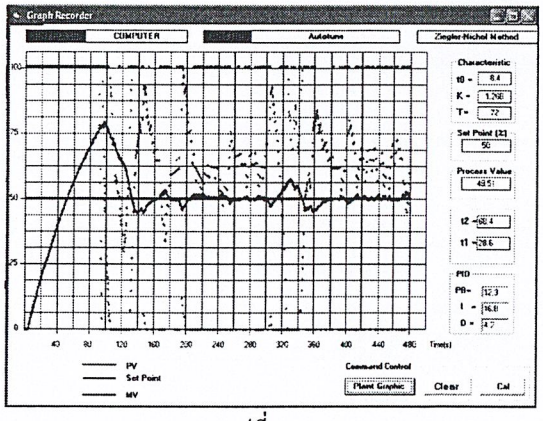
$$t_0 = 8.4\ sec\ ,K = 1.268\ ,\tau = 72\ sec$$

7.2.2 ทดลองควบคุมระดับน้ำที่ค่าเป้าหมาย 50% ด้วยวิธีการของDahlin ได้ผลการทดลองดังรูปที่ 7.3

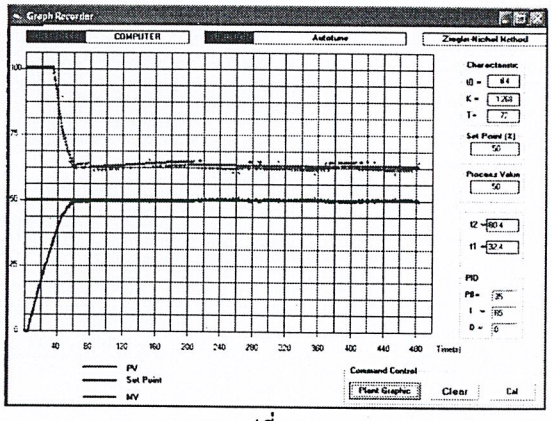


รูปที่ 7.3

7.2.3 ทดลองควบคุมระดับน้ำที่ค่าเป้าหมาย 50% ด้วยวิธีการของZiegler-Nichols ได้ค่าพารามิเตอร์ *PB*=12.3, *Ti*=16.8 และ *Td*=4.2 ผลการทดลองดังรูปที่ 7.4 และทำการปรับปรุงค่าพารามิเตอร์ โดยใช้ *PB*=35, *Ti*=65 และ *Td*=0 ได้ผลการทดลองดังรูปที่ 7.5



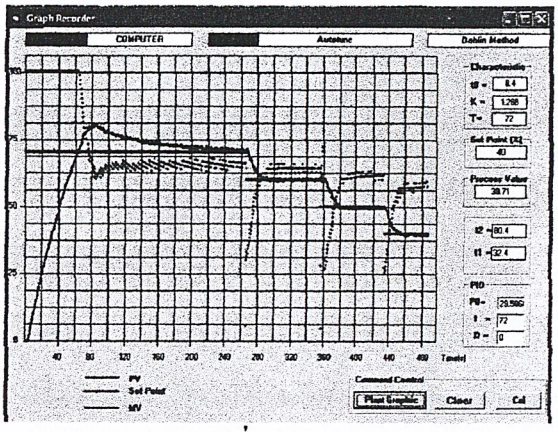
รูปที่ 7.4



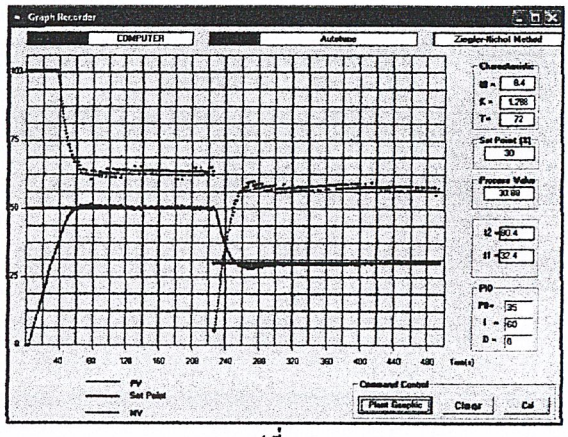
รูปที่ 7.5

7.2.4 ทดลองควบคุมระดับน้ำที่ค่าเป้าหมาย 70% ด้วยวิธีการของDahlin และทำการเปลี่ยนค่าเป้าหมายไปที่ 60%, 50% และที่ 40% ตามลำดับ ผลการทดลองดังรูปที่ 7.6 เปรียบเทียบกับวิธีของ Ziegler-Nicholsที่ค่าเป้าหมาย 50% และทำการเปลี่ยนระดับการควบคุมไปที่ 30% ได้ผลการทดลองดังรูปที่ 7.7

7.2.4 ทดลองควบคุมระดับน้ำที่ค่าเป้าหมาย 70% ด้วยวิธีการของDahlin และทำการเปลี่ยนค่าเป้าหมายไปที่ 60%, 50% และที่ 40% ตามลำดับ ผลการทดลองดังรูปที่ 7.6 เปรียบเทียบกับวิธีของ Ziegler-Nicholsที่ค่าเป้าหมาย 50% และทำการเปลี่ยนระดับการควบคุมไปที่ 30% ได้ผลการทดลองดังรูปที่ 7.7

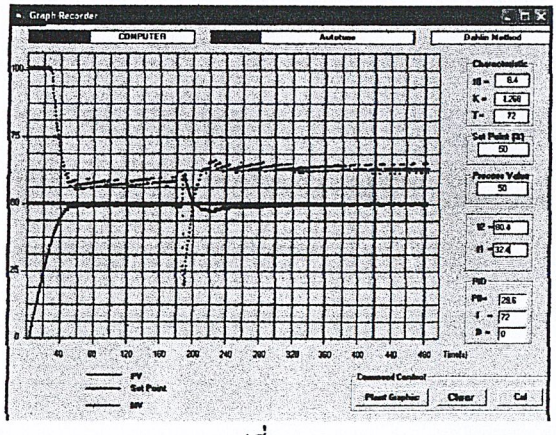


รูปที่ 7.6

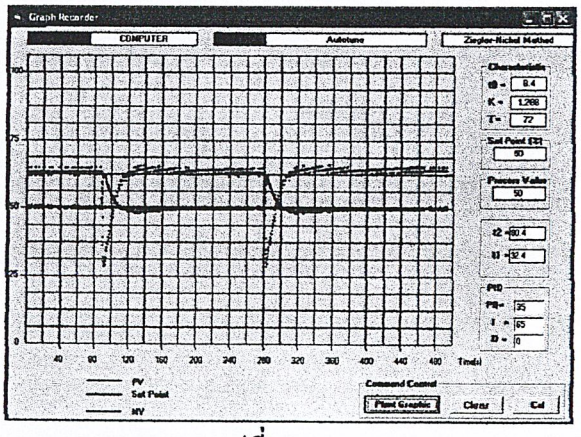


รูปที่ 7.7

7.2.5 ทดลองควบคุมระดับน้ำที่ค่าเป้าหมาย 50% ด้วยวิธีการของ Dahlin แล้วทำการใส่สิ่งรบกวนเป็นปริมาณน้ำขนาดประมาณ 2 ลิตร ผลการทดลองดังรูปที่ 7.8 เปรียบเทียบกับวิธีของ Ziegler-Nichols ผลการทดลองดังรูปที่ 7.9

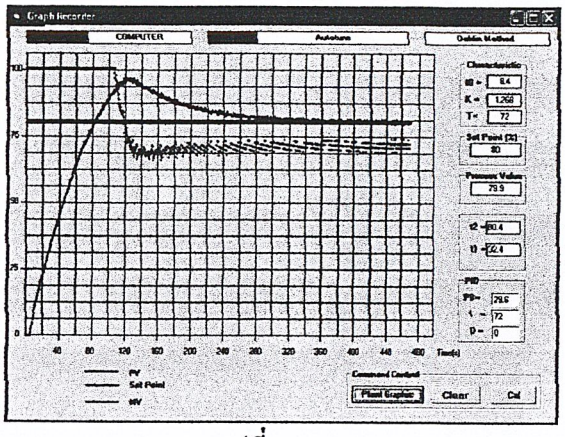


รูปที่ 7.8

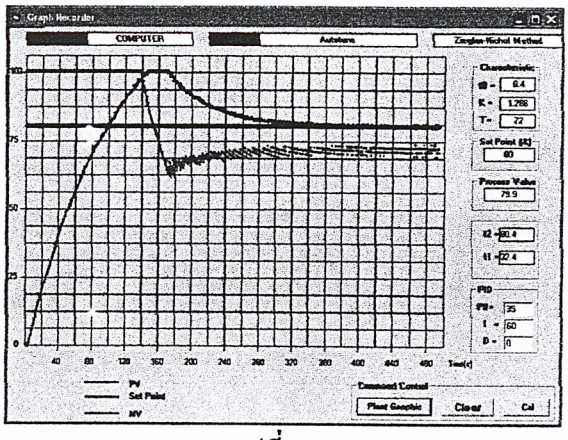


รูปที่ 7.9

7.2.6 ทดลองควบคุมระดับน้ำที่ค่าเป้าหมายสูงสุดที่ 80% โดยวิธีของ Dahlin ผลการทดลองดังรูปที่ 7.10 และ วิธีของ Ziegler-Nichols ผลการทดลองดังรูปที่ 7.11

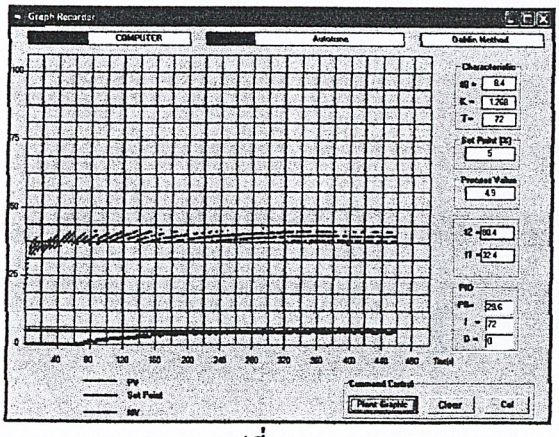


รูปที่ 7.10

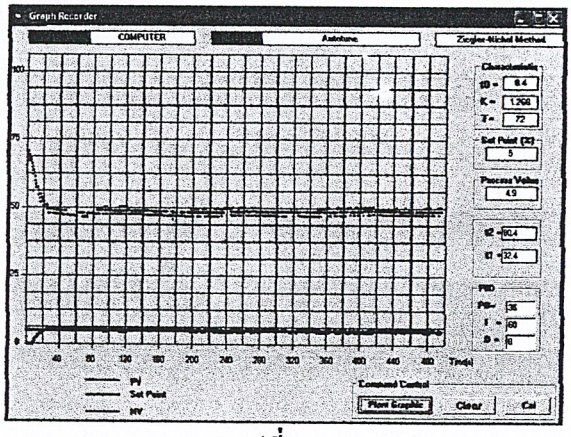


รูปที่ 7.11

7.2.7 ทดลองควบคุมระดับน้ำที่ค่าเป้าหมายต่ำสุดที่ 5% โดยวิธีของ Dahlin ผลการทดลองดังรูปที่ 7.12 และ วิธีของ Ziegler-Nichols ผลการทดลองดังรูปที่ 7.13

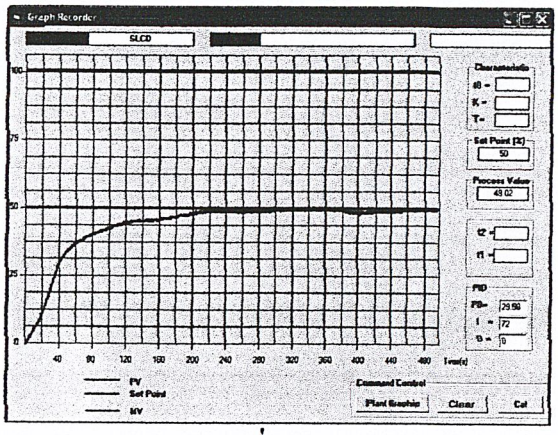


รูปที่ 7.12

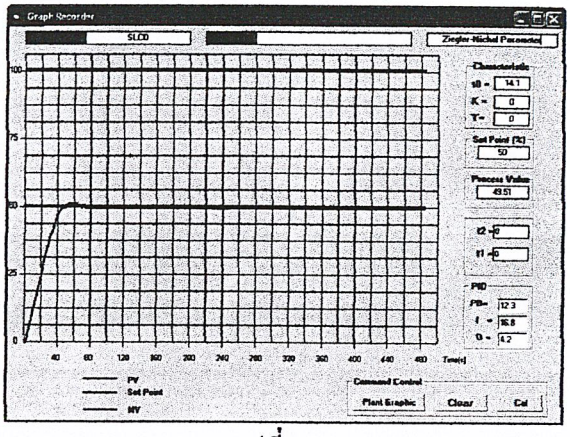


รูปที่ 7.13

7.2.8 ทดลองควบคุมระดับน้ำที่ค่าเป้าหมาย 50% โดยใช้ SLCD Indicating Controller ด้วยวิธีการของDahlinผลการทดลองดังรูปที่7.14และวิธีการของZiegler-Nichols ผลการทดลองดังรูปที่ 7.15



รูปที่ 7.14



รูปที่ 7.15

ตารางที่ 7.1 แสดงการเปรียบเทียบค่าพารามิเตอร์ของตัวควบคุมโดยวิธี Dahlin กับวิธีของ Ziegler-Nichols (ก่อนการปรับปรุงค่าพารามิเตอร์)

	Dahlin	Ziegler-Nichols
Proportional Band : PB	29.59	12.33
Integral Time : Ti	72.0	16.80
Derivative Time : Td	0	4.20

ตารางที่ 7.2 แสดงการเปรียบเทียบค่าพารามิเตอร์ของตัวควบคุมโดยวิธี Dahlin กับวิธีของ Ziegler-Nichols (หลังจากทำการปรับปรุงค่าพารามิเตอร์เรียบร้อยแล้ว)

	Dahlin	Ziegler-Nichols
Proportional Band : PB	29.59	35
Integral Time : Ti	72.0	60
Derivative Time : Td	0	0

บทที่ 8
บทวิจารณ์และสรุป

8.1 สรุปผลการดำเนินงาน

ปริญญานิพนธ์ฉบับนี้ได้ทำการศึกษาและสร้างระบบควบคุมระดับน้ำ วงจรอินเตอร์เฟส ที่ใช้ในการติดต่อระหว่างคอมพิวเตอร์กับระบบ โดยอาศัยการแปลงสัญญาณดิจิตอลกับสัญญาณอนาลอก และใช้ไมโครคอนโทรลเลอร์ PIC16F628 เป็นตัวควบคุมการทำงาน ในส่วนตัวควบคุมใช้ภาษาวิชวลเบสิกเขียนโปรแกรมเพื่อทำการหาค่าพารามิเตอร์ของตัวควบคุมพีไอดี เพื่อนำค่าพารามิเตอร์ของตัวควบคุมที่เหมาะสมมาใช้ในการควบคุมระดับน้ำ ซึ่งจากผลการทดลองสามารถสรุปได้ดังนี้

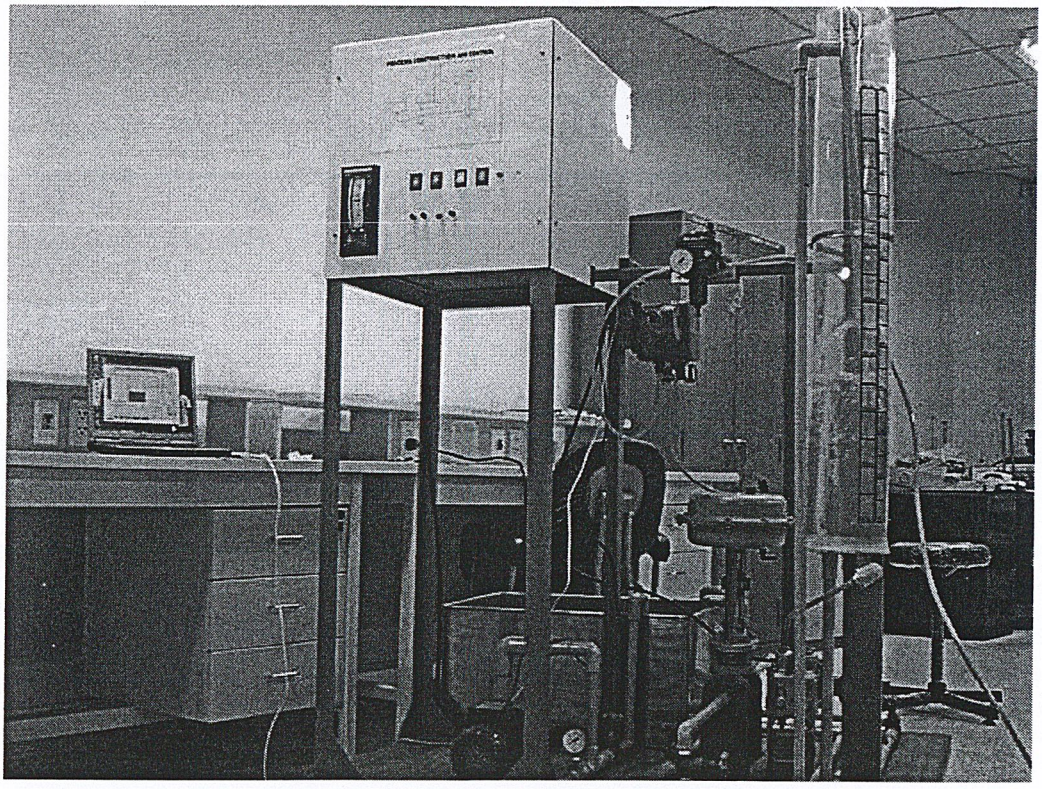
1. จากการควบคุมระดับน้ำที่ระดับต่างๆ โดยยังมิได้มีการปรับปรุงค่าพารามิเตอร์ วิธีของDahlin ให้ผลตอบสนองที่ดีกว่าวิธีของZiegler-Nichols เนื่องจากค่าพารามิเตอร์ที่ได้จากตารางของ Dahlin ได้ตัวควบคุมเป็นแบบพีไอ ส่วนตารางของZiegler-Nichols ได้ตัวควบคุมเป็นแบบพีไอดี ซึ่งค่าดี (Derivative Time: Td) ส่งผลให้ ตัวควบคุมเกิดผลตอบสนองมากต่อการเปลี่ยนแปลงค่าอินพุทเพียงเล็กน้อย ไม่เหมาะกับระบบที่อินพุทมีการเปลี่ยนแปลงอย่างรวดเร็ว อย่างระบบควบคุมระดับน้ำที่มีภาชนะบรรจุน้ำขนาดเล็ก อย่างที่ได้สร้างขึ้นในปริญญานิพนธ์นี้ เนื่องจากอาจจะทำให้เกิดความเสียหายของคอนโทรลวาล์วได้
2. ในบางครั้งค่าพารามิเตอร์ที่ได้จากตาราง อาจไม่ใช่ค่าที่เหมาะสมที่สุดในการใช้งานจริงกับระบบแต่ละระบบ ดังนั้นอาจมีการปรับปรุงค่าพารามิเตอร์เพื่อให้ได้ตัวควบคุมที่ดีที่สุดกับระบบแต่ละระบบ ซึ่งในการทดลอง ได้ทำการปรับปรุงค่าพารามิเตอร์ของวิธี Ziegler-Nichols จนได้ค่าพารามิเตอร์ที่เหมาะสมกับระบบ
3. จากการทดลองปรับค่าเป้าหมายของระดับน้ำเป็นระดับอื่นๆพบว่า การควบคุมทั้ง 2 วิธีสามารถเข้าสู่ค่าเป้าหมายใหม่ได้ ไม่ว่าจะเพิ่ม หรือลดระดับค่าเป้าหมาย
4. จากการทดลองใส่สัญญาณรบกวนเป็นน้ำปริมาตร 2 ลิตรลงในถังควบคุมระดับน้ำ ในขณะที่ระบบเข้าสู่สภาวะคงตัวแล้ว พบว่าการควบคุมทั้ง 2 วิธีสามารถควบคุมระดับน้ำให้เข้าสู่ระดับค่าเป้าหมายได้ดังเดิม
5. จากการทดลองควบคุมระดับน้ำพบว่า สามารถควบคุมระดับน้ำได้ตั้งแต่ 5%-80% ของระดับน้ำในถังควบคุมระดับน้ำเหมือนกันทั้งวิธีของ Dahlin และวิธีของ Ziegler-Nichols

6. การทดลองโดยใช้เครื่องควบคุม SLCD Indicating Controller ให้ผลตอบสนองที่ดีกว่าตัวควบคุมที่สร้างขึ้นเมื่อใช้พารามิเตอร์จากวิธีของZiegler-Nichols ซึ่งต่างจากผลการทดลองโดยใช้คอนโทรลเลอร์ที่สร้างขึ้น เนื่องจากสมการพีไอดีที่ใช้ในการคำนวณต่างกัน
7. Overshoot ของผลตอบสนองที่เกิดขึ้น สามารถปรับได้ตามความต้องการของผู้ใช้ โดยการปรับค่าพารามิเตอร์ *PB, Ti, Td*

ภาคผนวก ก

คู่มือการใช้งาน

PROCESS CONSTRUCTION AND CONTROL



การใช้งาน การใช้งานระบบควบคุมระดับน้ำนี้สามารถใช้งานได้ 2 แบบ คือ แบบแรก ใช้ SLCD Indicating Controller เป็นตัวควบคุม และแบบที่สอง คือใช้คอมพิวเตอร์เป็นตัวควบคุม

ในการใช้คอมพิวเตอร์เป็นตัวควบคุม ผู้ใช้สามารถเรียกใช้งานได้จาก ไฟล์ PCC.EXE โดยมีฟังก์ชันการใช้งานหลักดังนี้

1. Manual Control
2. AutoTune

โดยใน Manual Control นั้น เป็นการทำงานเลียนแบบ SLCD Indicating Controller ผู้ใช้จะต้องเป็นผู้กำหนดค่าพารามิเตอร์ต่างๆให้กับตัวควบคุม ซึ่งการทำงานแบ่งออกเป็น 2 โหมดย่อย คือ M Mode และ A Mode

M Mode เป็นการทำงานโดยนำค่า เปอร์เซ็นการเปิดวาล์ว ส่งออกไปควบคุมการเปิดวาล์วโดยตรง ไม่มีการคำนวณค่า MV ใช้งานในกรณีที่Start up ระบบขึ้นมาใหม่ และผู้ใช้งานต้องการปรับวาล์วเอง

A Mode เป็นการทำงานโดยนำค่า พารามิเตอร์ PB Ti Td ที่ผู้ใช้งานเป็นผู้ใส่ให้กับโปรแกรม มาคำนวณค่า MV จากสมการ PID เพื่อส่งออกไปปรับวาล์ว ทั้งนี้ผู้ใช้จะต้องเลือกค่าพารามิเตอร์ของตัวควบคุม PID ให้เหมาะสมกับกระบวนการ จึงจะได้ผลการควบคุมที่ดี ซึ่งอาจจะต้องอาศัยประสบการณ์หรือการลองผิดลองถูกก็ได้

ในส่วน AutoTune นั้น ก่อนใช้งานในโหมดนี้ผู้ใช้จะต้องทำการหาค่าพารามิเตอร์ของทรานสเฟอร์ฟังก์ชัน(***K, t0, T***)ด้วยวิธีการทดสอบมาก่อน ใน M Mode โดยการป้อนสัญญาณ Unit Step ให้กับระบบ และทำการบันทึกผลการทดลอง และคำนวณค่าพารามิเตอร์ของทรานสเฟอร์ฟังก์ชัน(***K, t0, T***) โดยใช้โปรแกรมอัตโนมัติที่ฟอร์ม RECORDER หลังจากนั้นเปลี่ยนฟังก์ชันการทำงานมาเป็น AutoTune และเลือกวิธีที่จะใช้คำนวณพารามิเตอร์ PB,Ti,Td แบบอัตโนมัติ ระหว่าง Dahlin และ Ziegler-Nicholsเพื่อทำการใช้งานหลังจากนั้นโปรแกรมจะคำนวณค่าพารามิเตอร์ PB,Ti,Td ให้ ซึ่งค่าพารามิเตอร์ที่โปรแกรมคำนวณให้นี้ สามารถทำการปรับค่าได้เพื่อให้ได้ผลการควบคุมที่ดีที่สุด

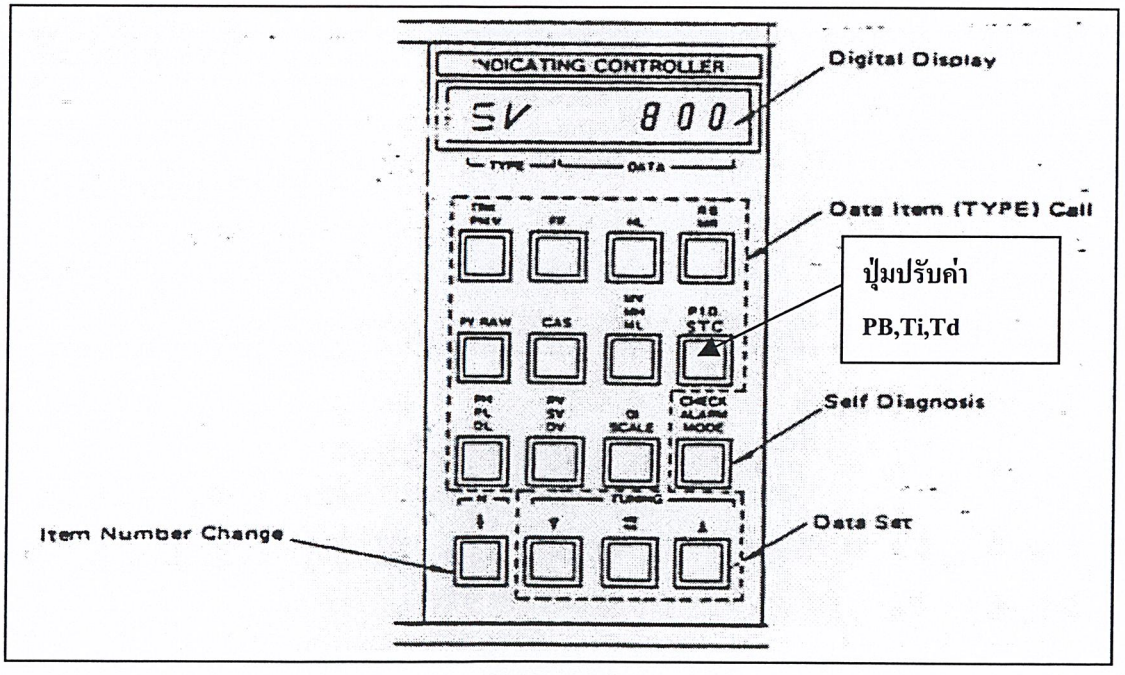
A. การใช้งานโดยใช้ SLCD Indicating Controller เป็นอุปกรณ์ควบคุม

1. เปิดสวิตซ์ที่ตู้คอนโทรลดังตาราง

POWER	SLCD	SUPPLY	PUMP	CONTROLLER SELECTER
ON	**ON**	**ON**	**ON**	**SLCD**

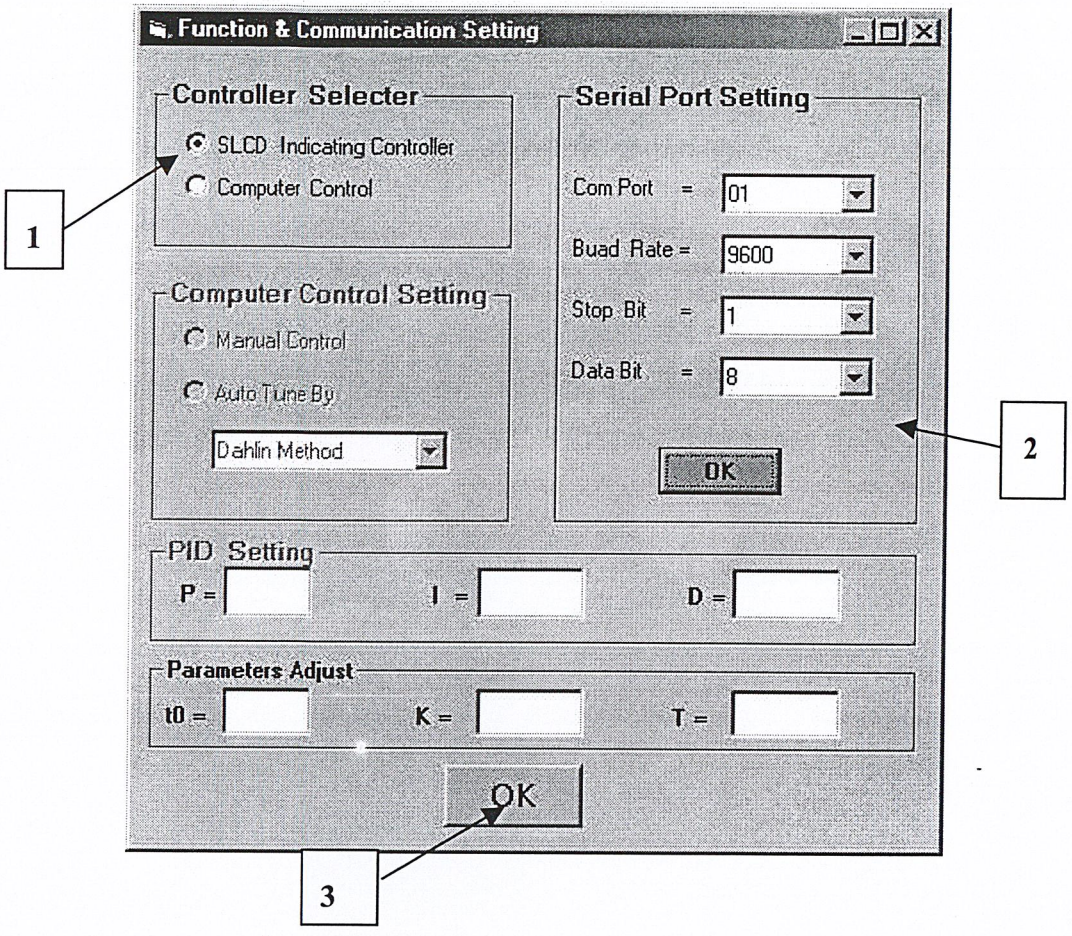
2. ตั้งค่าพารามิเตอร์ PID ที่ Panel ของ **SLCD Indicating Controller** ดังรูป

กดที่ปุ่มปรับค่า PB,Ti,Td เมื่อกดปุ่ม 3 ครั้งหัวข้อของข้อมูลจะปรากฏบนdigital Display ดังนี้



3. ตั้งค่า Set point ที่ SLCD เช่น Set point = 30 %

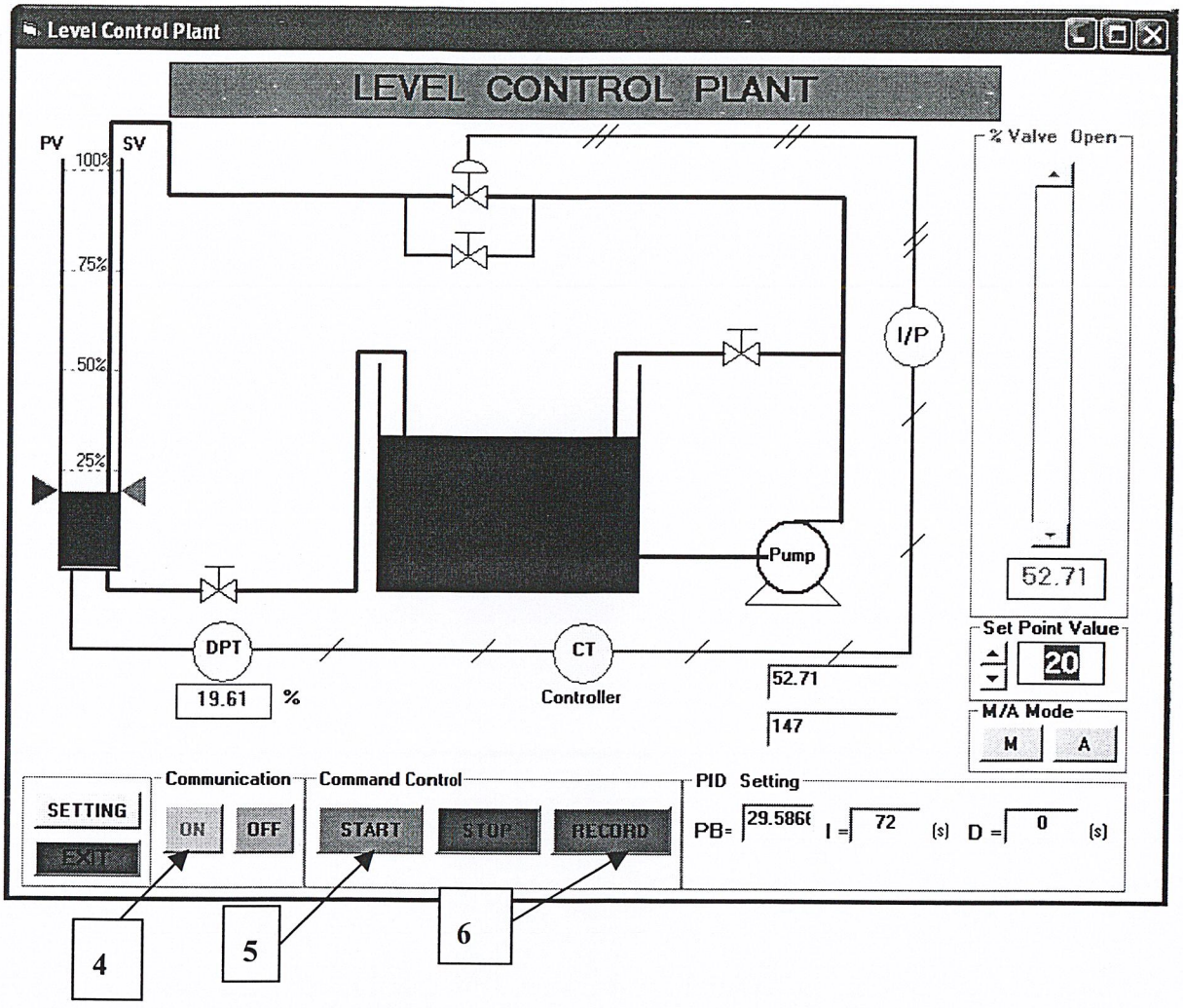
4. เปิดโปรแกรมที่คอมพิวเตอร์เพื่อบันทึกผลการทดลอง ดังรูป



4.1 เลือกตัวควบคุมเป็น **SLCD Indicating Controller**

4.2 ตั้งค่า Serial Port แล้วคลิกปุ่ม OK

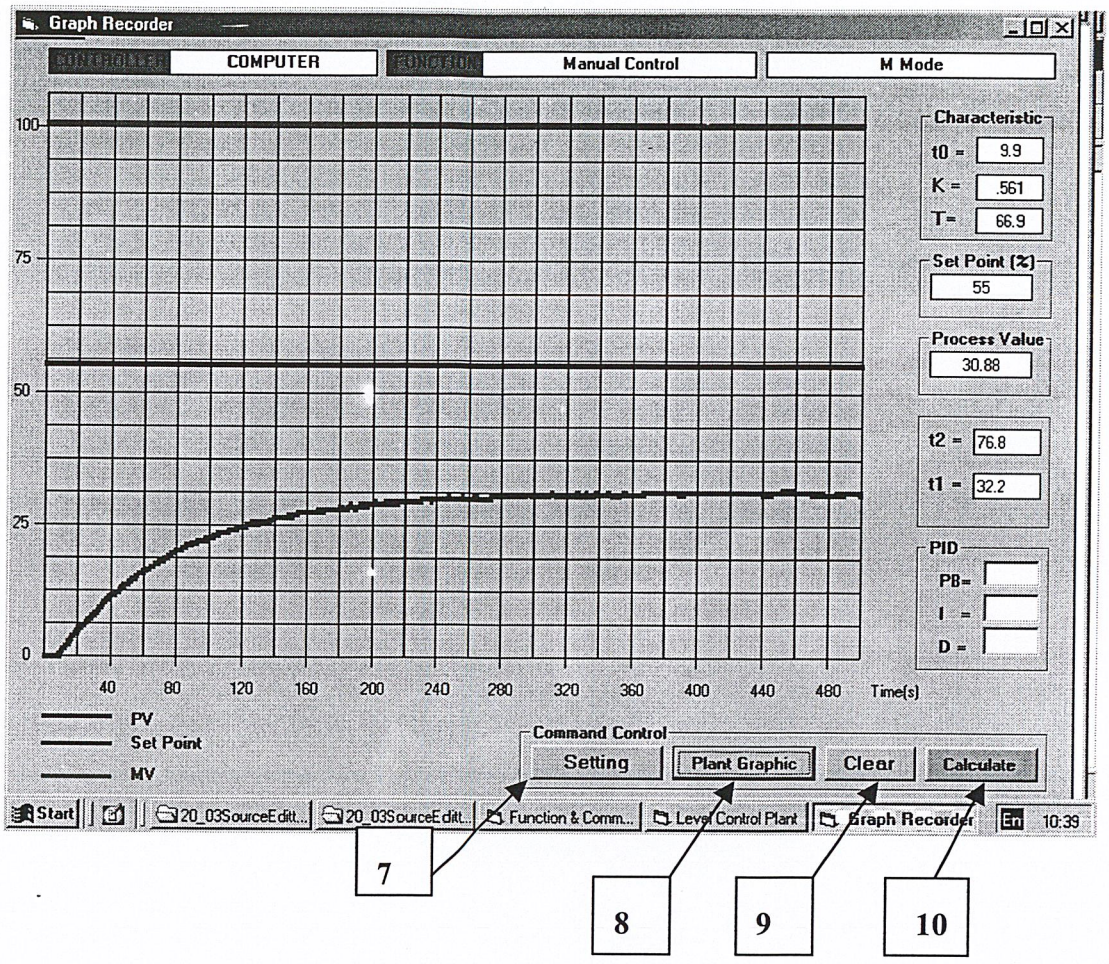
4.3 คลิก OK เมื่อตั้งค่าเสร็จ จะได้หน้าต่างถัดไป



4.4 คลิก ON เพื่อเปิดพอร์ท รับ-ส่งข้อมูล

4.5 คลิก START เพื่อ รับค่า และแสดงภาพการทำงานของระบบ

4.6 คลิก RECORD เมื่อต้องการบันทึกผลการทดลอง



4.7 คลิกปุ่ม Setting เพื่อกลับไปยัง ฟอร์ม Setting

4.8 คลิกปุ่ม Plant Graphic เพื่อกลับไปยัง ฟอร์ม Plant Graphic

4.9 คลิกปุ่ม Clear เพื่อ Clear กราฟบนฟอร์ม

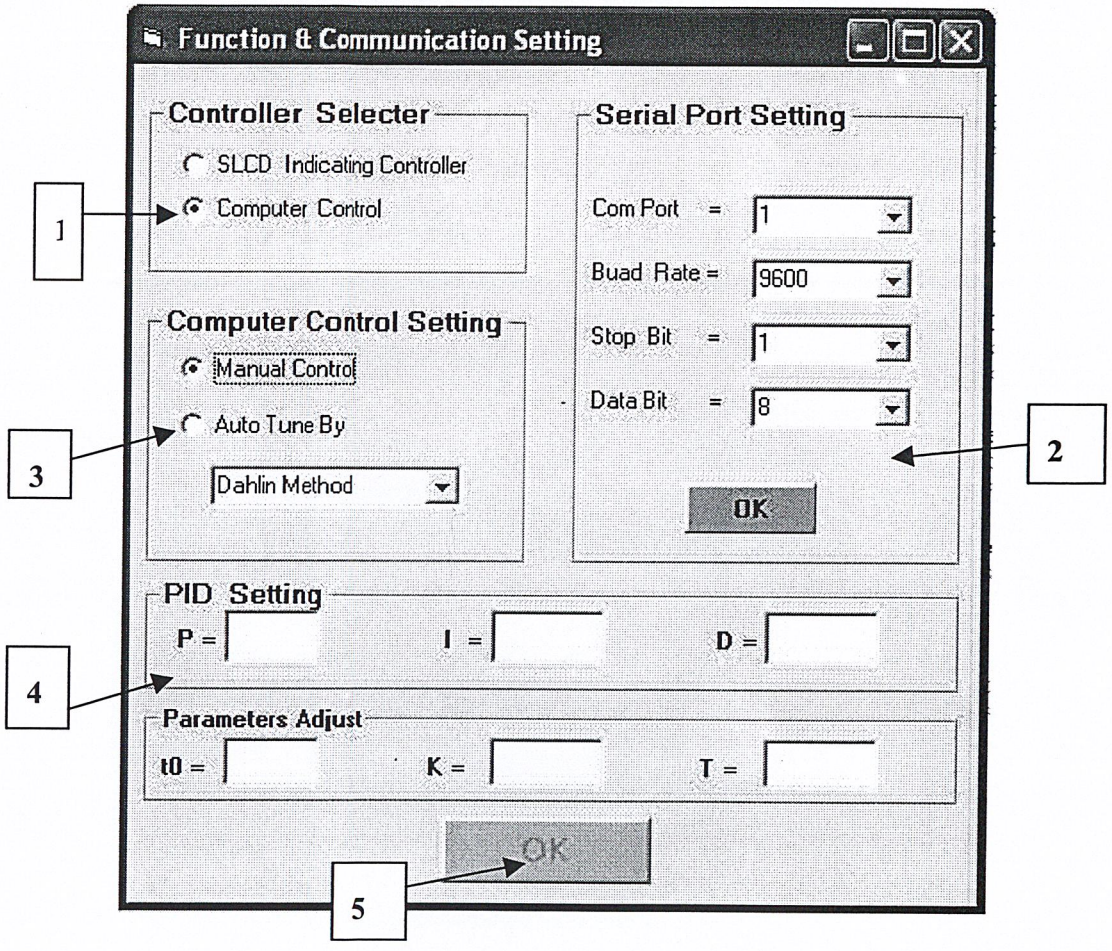
4.10 คลิกปุ่ม Calculate หากต้องการคำนวณ ค่า ***K, t0, T***

B. การใช้งานโดยใช้ Computer เป็นอุปกรณ์ควบคุม

1. เปิดสวิตซ์ที่ตู้คอนโทรลดังตาราง

POWER	SLCD	SUPPLY	PUMP	CONTROLLER SELECTER
ON	**OFF**	**ON**	**ON**	**COMPUTER**

2. เปิดโปรแกรมที่คอมพิวเตอร์เพื่อทำการทดลอง ดังรูป



2.1 เลือกตัวควบคุม เป็น Computer Control

2.2 ตั้งค่า Serial Port แล้วคลิกปุ่ม OK

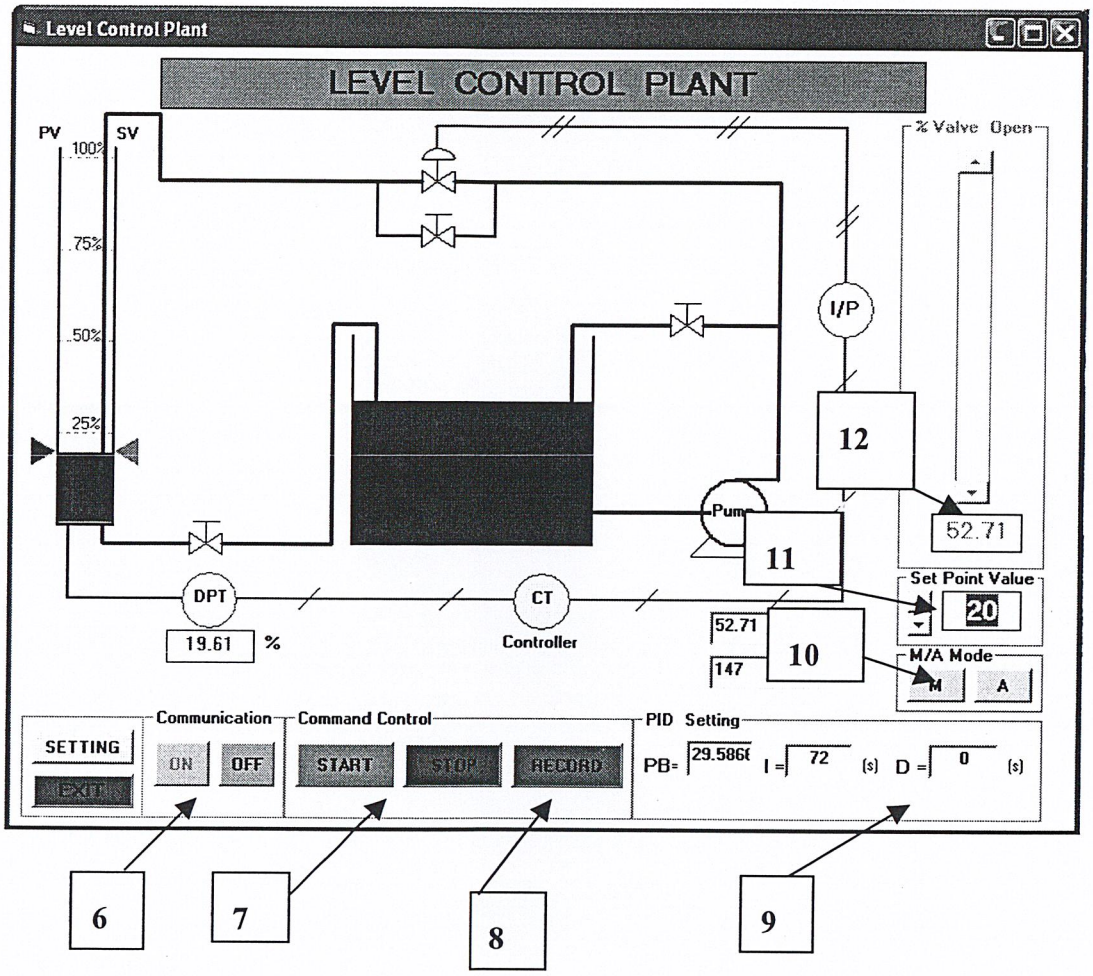
2.3 เลือกโหมดการควบคุม

(หมายเหตุ : ในกรณีที่ต้องการใช้งานในโหมด AutoTune ต้องทำการหาพารามิเตอร์ของทรานสเฟอร์ฟังก์ชันก่อนใน Manual Control โดยใช้ M Mode เพื่อนำค่าพารามิเตอร์มาใช้งาน)

2.4 ตั้งค่า พารามิเตอร์ต่างๆที่ต้องการ

(หมายเหตุ: ในกรณีที่ใช้งานโหมด A Mode ต้องกำหนดค่า *PB, Ti, Td* หรือ ในกรณีที่เรามีค่า t_0, K, T เก็บไว้ ก็สามารถนำมาใส่ให้โปรแกรมคำนวณค่า *PB, Ti, Td* ได้)

2.5 คลิก OK เมื่อตั้งค่าเสร็จ



2.6 คลิก ON เพื่อเปิดพอร์ท รับ-ส่ง ข้อมูล

2.7 คลิก START ,STOPเพื่อ รับ-หยุดรับ ค่า และแสดงภาพการทำงานของระบบ

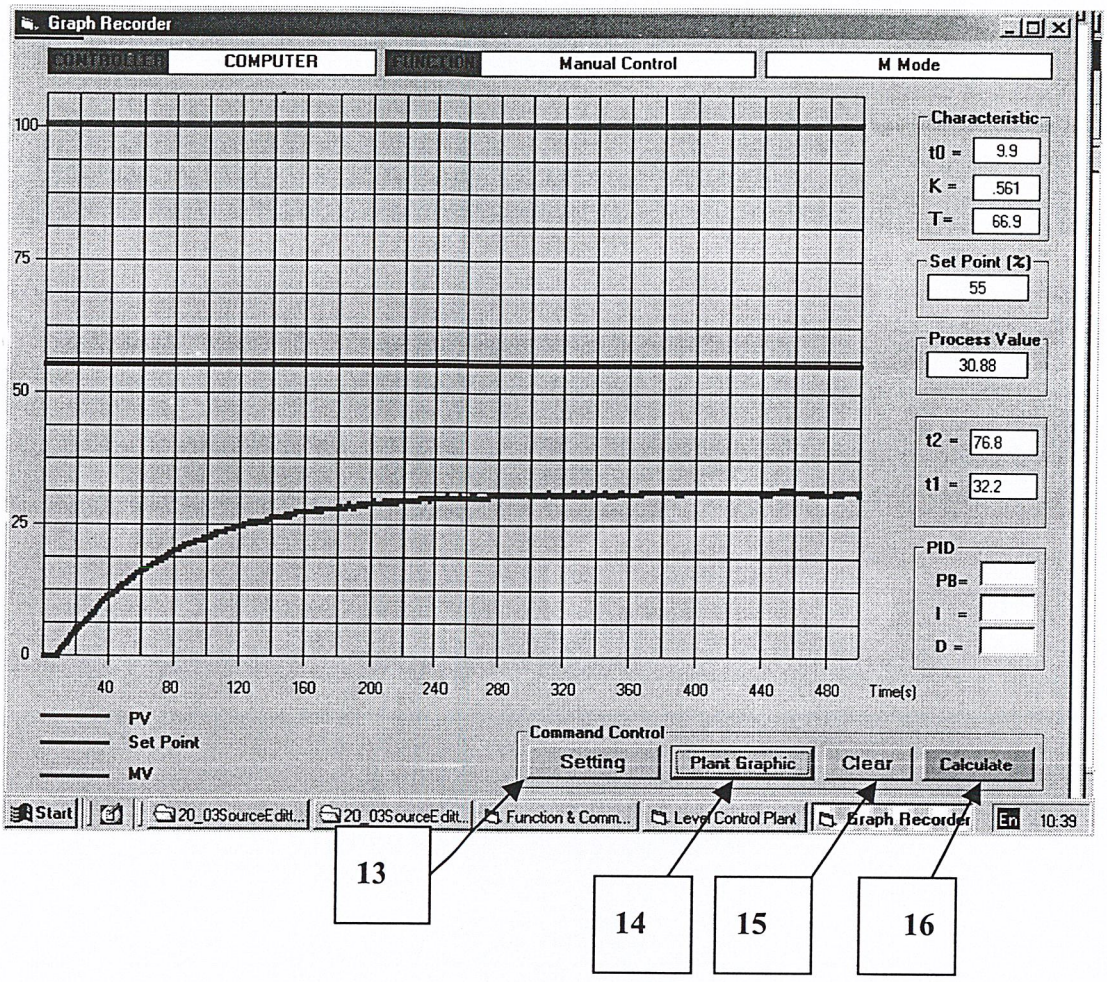
2.8 คลิก RECORD เมื่อต้องการบันทึกผลการทดลอง

2.9 ปรับค่า *PB, Ti, Td*

2.10 เปลี่ยนโหมดระหว่าง M Mode และ A Mode ใน Manual Control

2.11 ตั้งค่า Set Point

2.12 ค่าเปอร์เซ็นการเปิดวาล์ว



2.13 คลิกปุ่ม Setting เพื่อกลับไปยัง ฟอร์ม Setting

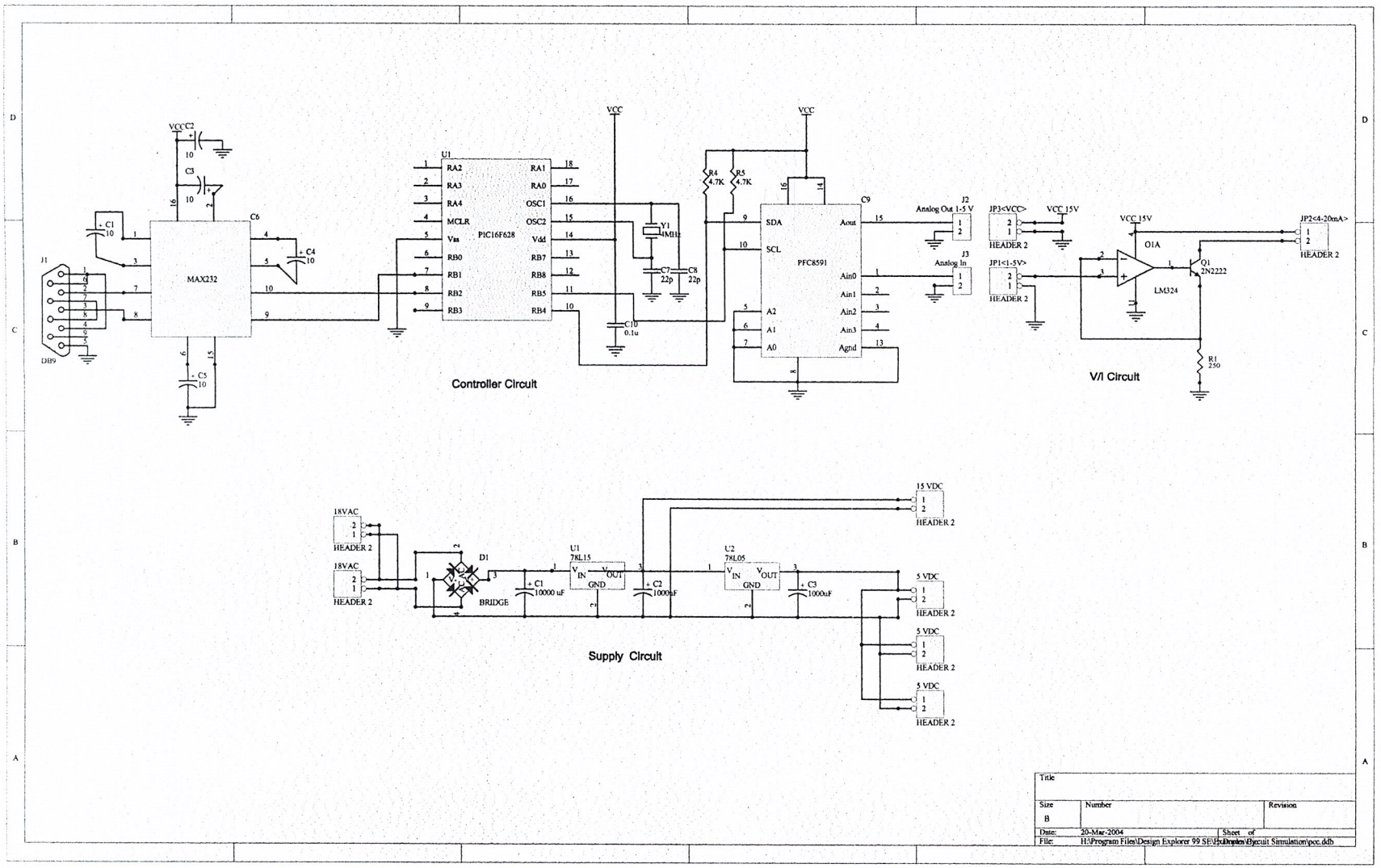
2.14 คลิกปุ่ม Plant Graphic เพื่อกลับไปยัง ฟอร์ม Plant Graphic

2.15 คลิกปุ่ม Clear เพื่อ Clear กราฟบนฟอร์ม

2.16 คลิกปุ่ม Calculate หากต้องการคำนวณ ค่า ***K, t0, T***

(หมายเหตุ: ในการเปลี่ยนโหมดการทำงานทุกครั้งต้องทำการ STOPเพื่อ หยุดรับ-ส่ง ข้อมูล ก่อนทุกครั้ง)

ภาคผนวก ข


VCC
C2
10
C3
10
C1
10
C6
C4
10
J1
DB9
MAX232
C5
10
U1
RA2
RA3
RA4
MCLR
Vss
RB0
RB1
RB2
RB3
RA1
RA0
OSC1
OSC2
Vdd
RB7
RB8
RB5
RB4
PIC16F628
Y1
4MHz
C7
22p
C8
22p
C10
0.1u
R4
4.7K
R5
4.7K
C9
SDA
SCL
A2
A1
A0
Aout
Ain0
Ain1
Ain2
Ain3
Agnd
PFC8591
J2
Analog Out 1-5 V
J3
Analog In
Controller Circuit
JP3<VCC>
VCC_15V
HEADER 2
JP1<1-5V>
HEADER 2
O1A
LM324
Q1
2N2222
R1
250
JP2<4-20mA>
HEADER 2
V/I Circuit
18VAC
HEADER 2
18VAC
HEADER 2
D1
BRIDGE
U1
7KL15
VIN
VOUT
GND
C1
10000 uF
C2
1000uF
U2
7KL05
VIN
VOUT
GND
C3
1000uF
15 VDC
HEADER 2
5 VDC
HEADER 2
5 VDC
HEADER 2
5 VDC
HEADER 2
Supply Circuit
Title
Size
B
Number
Revision
Date:
20-Mar-2004
Sheet of
File:
H:\Program Files\Design Explorer 99 SE\Examples\Pycuit Simulation\pcc.ddb

PIC Source Code

```
'*****************************************************************
'* Name    : UNTITLED.BAS                                        *
'* Author  : [set under view...options]                           *
'* Notice  : Copyright (c) 2004 [set under view...options]       *
'*         : All Rights Reserved                                 *
'* Date    : 10/2/2004                                           *
'* Version : 1.0                                                 *
'* Notes   :                                                     *
'*         :                                                     *
'*****************************************************************
DEFINE OSC 4

INCLUDE        "bs2defs.bas"
        DA VAR WORD
        AD VAR WORD

LOOP:  SERIN PORTB.1,T9600,["A"],DA
        I2CWRITE PORTB.4,PORTB.5,%10010000,$40
        PAUSE 10
        I2CWRITE PORTB.4,PORTB.5,%10010000,[DA]
        PAUSE 10

        I2CREAD PORTB.4,PORTB.5,%10010001,[AD]
        PAUSE 10
        SEROUT PORTB.2,T9600,[AD]

GoTo    LOOP
End
```

Visual Basic Source Code

Module1.bas

```
Public CT, SV1, Auto As Integer
Public P, I, D, Ti, Td, Mi, Mi1, Dt1, Dt2, Dt3, Dt4 As Double
Public DataIn, Data, Data1, MV, SV, K, Kc, Kp, Ki, Kd As Double
Public t0, t, t1, t2, step(3300), count1, count2, num1, num2, firm1, firm2 As Double
Public c1, e, e0, e1, e2, c2, x, y, y2, y3, B As Double
Public valveOpen, valveOut, Out As Variant
Public ManualMode, clear As Boolean
Sub Main()
   Load Setting
   Setting.Show
End Sub
```

Set.frm

```
Private Sub Option4_Click()       ' Auto Tuning
   Combo1.Enabled = True
   CT = 2
   Recorder.Mode = "Autotune"
End Sub
Private Sub Combo1_Click()
   t0 = Text1
   K = Text2
   t = Text3
   B = t / 4
  If Combo1 = "Dahlin Method" Then
     Auto = 1
     Recorder.By = "Dahlin Method"
'------------------ Cal PID ------------------
     If t0 > B Then
        Td = t0 / 2
     Else
```

```
            Td = 0
        End If
            Kc = t / (K * t0 * 2)
            Ti = t
            P = 100 / Kc
            I = Ti
            D = Td
            setP = Round(P, 2)
            setI = I
            setD = D
'-----------------------------------------
    Else
        Auto = 2
        Recorder.By = "Ziegler-Nichol Method"
'------------------------Cal PID ----------------
            Kc = (1.2 * t) / (t0 * K)
            Td = 0.5 * t0
            Ti = 2 * t0
            P = 100 / Kc
            I = Ti
            D = Td
            setP = P
            setI = I
            setD = D
'-----------------------------------------------
    End If
End Sub
Private Sub Form_Load()
    Frame2.Enabled = False
    Frame4.Enabled = False
    OK.Enabled = False
End Sub
```

```
Private Sub obComControl_Click()
    Frame2.Enabled = True
    Option4.Enabled = True
    obManControl.Enabled = True
    Combo1.Enabled = True
    Recorder.Status.Text = "COMPUTER"
End Sub
Private Sub obManControl_Click()
    Frame4.Enabled = True
    plantGraphic.Frame1.Enabled = True
    plantGraphic.Frame2.Enabled = True
    plantGraphic.Frame6.Enabled = True
    setP.BackColor = &H80000009
    setI.BackColor = &H80000009
    setD.BackColor = &H80000009
    CT = 1
    Recorder.Mode = "Manual Control"
End Sub
Private Sub obSLCD_Click()
    Frame2.Enabled = False
    Frame4.Enabled = False
    Option4.Enabled = False
    obManControl.Enabled = False
    Combo1.Enabled = False
    plantGraphic.Frame2.Enabled = False
    plantGraphic.Frame5.Enabled = False
    plantGraphic.Frame6.Enabled = False
    Recorder.Status.Text = "SLCD"
    CT = 0
End Sub
Private Sub OK_Click()
    plantGraphic.Show
```

```
    If CT = 1 Then
        plantGraphic.Frame1.Enabled = True
        plantGraphic.VScroll1.Enabled = True
        plantGraphic.txtVOpen.Enabled = True
    Else
        plantGraphic.Frame1.Enabled = False
        plantGraphic.VScroll1.Enabled = False
        plantGraphic.txtVOpen.Enabled = False
    End If
End Sub

Private Sub portOK_Click()
    plantGraphic.MSComm1.CommPort = setCom
    OK.Enabled = True
    portOK.Enabled = False
End Sub
Private Sub setD_Change()
    D = setD
    plantGraphic.DText5.Text = D
End Sub
Private Sub setI_Change()
    I = setI
    plantGraphic.IText4.Text = I
End Sub
Private Sub setP_Change()
    P = setP
    plantGraphic.PText3.Text = P
End Sub
Private Sub Text1_Change()
    t0 = Text1
    Recorder.txtt0 = t0
End Sub
```

```
Private Sub Text2_Change()
   K = Text2
   Recorder.txtK = K
End Sub

Private Sub Text3_Change()
   t = Text3
   Recorder.txtT = t
End Sub
```

plantGraphic.frm

```
Private Sub plantGraphic_Load()
   PText3.Text = P
   IText4.Text = I
   DText5.Text = D
   ManualMode = True
   Frame4.Enabled = False
End Sub
Private Sub cmnEXIT_Click()
   cmnEXIT.BackColor = &HFF&
End
End Sub
Private Sub Command8_Click()   'command Setting
   Setting.Show
End Sub
Private Sub CommA_Click()
   ManualMode = False
   Recorder.By = "A Mode(PID)"
End Sub
Private Sub CommM_Click()
   ManualMode = True
```

```
    Recorder.By = "M Mode"
End Sub
Private Sub commRecord_Click()
    Recorder.Show
    Recorder.Timer1.Enabled = True
End Sub
'----------------------------------------start----------------------------------------
Private Sub commStart_Click()
    Line8.BorderColor = &HFF0000
    Line9.BorderColor = &HFF0000
    Line19.BorderColor = &HFF0000
    Line20.BorderColor = &HFF0000
    Line21.BorderColor = &HFF0000
    Line22(0).BorderColor = &HFF0000
    Line23(0).BorderColor = &HFF0000
    Line24(0).BorderColor = &HFF0000
    Line25(0).BorderColor = &HFF0000
    Line26.BorderColor = &HFF0000
    Line27.BorderColor = &HFF0000
    Line32.BorderColor = &HFF0000
    Line33.BorderColor = &HFF0000
    Line34.BorderColor = &HFF0000
    Line35.BorderColor = &HFF0000
    Line38.BorderColor = &HFF0000
    Line39.BorderColor = &HFF0000
    Line40.BorderColor = &HFF0000
    Line41.BorderColor = &HFF0000
    Line42.BorderColor = &HFF0000
    If CT = 0 Then
        '++++++++++++++++++++SLCD Indicating
Controller++++++++++++++++++++++++++
        CT0R.Enabled = True
```

```
        CT0S.Enabled = True
    ElseIf CT = 1 Then
        '++++++++++++++++++++++Manual
Control++++++++++++++++++++++++++++++++++
        CT1R.Enabled = True
        CT1S.Enabled = True
    Else
        '++++++++++++++++++++Auto Tune by Ziegler-Nichol or
Dahlin+++++++++++++++
        Mi1 = ReceiveTxt
        MV0 = ReceiveTxt
        MV = ReceiveTxt
        CT2R.Enabled = True
        CT2S.Enabled = True

    End If
End Sub
'-------------------------------------Stop-----------------------------------
Private Sub commStop_Click()
    Recorder.Timer1.Enabled = False
    CT0R.Enabled = False
    CT0S.Enabled = False
    CT1R.Enabled = False
    CT1S.Enabled = False
    CT2R.Enabled = False
    CT2S.Enabled = False

    Line8.BorderColor = &H80000008
    Line9.BorderColor = &H80000008
    Line19.BorderColor = &H80000008
    Line20.BorderColor = &H80000008
    Line21.BorderColor = &H80000008
```

```
    Line22(0).BorderColor = &H80000008
    Line23(0).BorderColor = &H80000008
    Line24(0).BorderColor = &H80000008
    Line25(0).BorderColor = &H80000008
    Line26.BorderColor = &H80000008
    Line27.BorderColor = &H80000008
    Line32.BorderColor = &H80000008
    Line33.BorderColor = &H80000008
    Line34.BorderColor = &H80000008
    Line35.BorderColor = &H80000008
    Line38.BorderColor = &H80000008
    Line39.BorderColor = &H80000008
    Line40.BorderColor = &H80000008
    Line41.BorderColor = &H80000008
    Line42.BorderColor = &H80000008
End Sub

'[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[[ CT0R_Timer ]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
Private Sub CT0R_Timer()
   On Error Resume Next
  '    ::::::::::::          Receive  Module        ::::::::::::
   Data = MSComm1.Input
   Data1 = Asc(Data)
   DataIn = Round(((((Data1 / 255) * 100) - 20) / 80 * 100), 3)
   'ReceiveTxt.Text = Data1

'jjjjjjjjjjjjjjjjjjjjjjjjjj Display  Graphgic  jjjjjjjjjjjjjjjjjjjjjj
   Dt1 = Round(DataIn, 2)
   txtDPT.Text = Dt1
   If Dt1 < 0 Then
   Dt1 = 0
   Shape1.Height = 0
```

```
    Else
    Shape1.Height = 40 * DataIn
    End If

    Shape1.Top = 1200 + 40 * (100 - Dt1)
    Image2.Top = 1050 + 40 * (100 - Dt1)
    Recorder.Text5.Text = Round(DataIn, 2)
      If DataIn > 95 Then
        Beep
        Text2.BackColor = &HFF&
      ElseIf DataIn < 5 Then
        Beep
        Text2.BackColor = &HFF&
      Else
        Text2.BackColor = &HFF00&
      End If
End Sub
Private Sub CT0S_Timer()
   On Error Resume Next
   PersenOut = ((((valveOpen) / 100) * 80) + 20) / 100 * 255
   AscOut = Round(PersenOut, 0)
   MSComm1.Output = "A" & Chr(AscOut)
   'txtVOpen.Text = ""
End Sub

'{{{{{{{{{{{{{{{{    CT1R_Timer     }}}}}}}}}}}}}}}}}}}}}}}
Private Sub CT1R_Timer()
 On Error Resume Next
   '   ::::::::::::        Receive  Module      ::::::::::::::
   Data = MSComm1.Input
   Data1 = Asc(Data)
   DataIn = Round(((((Data1 / 255) * 100) - 20) / 80 * 100), 3)
```

```
'jjjjjjjjjjjjjjjjjjjjjjjjjjj Display Graphgic jjjjjjjjjjjjjjjjjjjjj
 Dt1 = Round(DataIn, 2)
 txtDPT.Text = Dt1
 If Dt1 < 0 Then
    Dt1 = 0
    Shape1.Height = 0
 Else
    Shape1.Height = 40 * DataIn
 End If
 Shape1.Top = 1200 + 40 * (100 - Dt1)
 Image2.Top = 1050 + 40 * (100 - Dt1)
 Recorder.Text5.Text = Round(DataIn, 2)
    If DataIn > 95 Then
       Beep
       Text2.BackColor = &HFF&
    ElseIf DataIn < 5 Then
       Beep
       Text2.BackColor = &HFF&
    Else
       Text2.BackColor = &HFF00&
    End If
'<<<<<<<<<<<<< Calculate valveOut Value  >>>>>>>>>>>>>>>>>
If ManualMode = True Then
   ' Manual Open
   valveOut = (100 - VScroll1)       '  Change to Direct Action
   txtVOpen.Text = VScroll1
Else
'---------- PUT PID CAL----------
   Kp = 100 / P
   Ki = Kp / I
   Kd = Kp * D
```

```
        PText3 = Round(P, 2)
        IText4 = Round(I, 2)
        DText5 = Round(D, 2)
        Recorder.Text3 = Round(P, 2)
        Recorder.Text4 = Round(I, 2)
        Recorder.Text6 = Round(D, 2)
        '////////////////////////////  PID EQ  ////////////////////
        e2 = S√ - DataIn                ' Cal  Error
        Mi = (Ki * e2 * 0.2) + Mi1          ' Cal Intrigal Manipulate Value
        MV0 = (Kp * e2) + (Ki * e2 * 0.2) + ((Kd * (e2 - e1)) / 0.2) + Mi1    'PID Eq
        Mi1 = Mi
        e1 = e2
    '---------------- Valve  Protection  ----------------------
        ReceiveTxt.Text = Round(MV0, 2)
        MV = 100 - MV0          '   Change to Direct Action
        If MV > 100 Then        '   Protect 0 < %O/P <100
          MV = 100
        ElseIf MV < 0 Then
          MV = 0
        Else
        End If
    '-----------------------------------------------------------
        valveOut = MV             'sent out
        txtVOpen.Text = Round((100 - MV), 2)
    End If
End Sub

'[[[[[[[[[[[[[[          CT1S_Timer          ]]]]]]]]]]]]]
Private Sub CT1S_Timer()
    On Error Resume Next
    PersenOut = ((((valveOut) / 100) * 80) + 20) / 100 * 255
    AscOut = Round(PersenOut, 0)
```

```
    MSComm1.Output = "A" & Chr(AscOut)
    oo = AscOut
End Sub

'[[[[[[[[[[[[[[          CT2R_Timer          ]]]]]]]]]]]]
Private Sub CT2R_Timer()
 On Error Resume Next
     '   ::::::::::::          Receive  Module       ::::::::::::::
    count2 = count2 + 1
    Data = MSComm1.Input
    Data1 = Asc(Data)
    DataIn = Round(((((Data1 / 255) * 100) - 20) / 80 * 100), 3)
    '---------------------- Display  Graphgic-----------------------
    Dt1 = Round(DataIn, 2)
    txtDPT.Text = Dt1
    If Dt1 < 0 Then
    Dt1 = 0
    Shape1.Height = 0
    Else
    Shape1.Height = 40 * DataIn
    End If

    Shape1.Top = 1200 + 40 * (100 - Dt1)
    Image2.Top = 1050 + 40 * (100 - Dt1)
    Recorder.Text5.Text = Round(DataIn, 2)
      If DataIn > 95 Then
         Beep
         Text2.BackColor = &HFF&
      ElseIf DataIn < 5 Then
         Beep
         Text2.BackColor = &HFF&
      Else
```

```
    Text2.BackColor = &HFF00&
  End If
      Kp = 100 / P
      Ki = Kp / I
      Kd = Kp * D
  '--------------------Display------------------------------
  PText3 = Round(P, 1)
  IText4 = I
  DText5 = D
  Recorder.Text3 = Round(P, 1)
  Recorder.Text4 = I
  Recorder.Text6 = D

'-------------------- PID EQ -------------------------------
  e2 = SV - DataIn              ' Cal  Error
  Mi = (Ki * e2 * 0.2) + Mi1    ' Cal Intrigal Manipulate Value
  MV0 = (Kp * e2) + (Ki * e2 * 0.2) + ((Kd * (e2 - e1)) / 0.2) + Mi1    'PID Eq
  Mi1 = Mi
  e1 = e2

'------------------- Valve Protection --------------------
  ReceiveTxt.Text = Round(MV0, 2)
  MV = 100 - MV0        '  Change to Direct Action

  If MV > 100 Then      '  Protect 0 < %O/P <100
    MV = 100
  ElseIf MV < 0 Then
    MV = 0
  Else
  End If

'------------------------- sent out ----------------------------
```

```
        valveOut = MV
        txtVOpen.Text = Round((100 - MV), 2)
        If count2 = 2490 Then
           Recorder.Timer1.Enabled = False
           plantGraphic.CT2S.Enabled = False
           plantGraphic.CT2R.Enabled = False
        End If
    End Sub

    '[[[[[[[[[[[[[[          CT2S_Timer          ]]]]]]]]]]]]
    Private Sub CT2S_Timer()
       On Error Resume Next
       PersenOut = ((((valveOut) / 100) * 80) + 20) / 100 * 255
       AscOut = Round(PersenOut, 0)
       MSComm1.Output = "A" & Chr(AscOut)
       oo = AscOut
    End Sub

    '----------------------------  On Off Port ----------------------------
    Private Sub OffCommu_Click()
       MSComm1.PortOpen = False
       OffCommu.Enabled = False
       OnCommu.Enabled = True
       Setting.portOK.Enabled = True
    End Sub
    Private Sub OnCommu_Click()
       OffCommu.Enabled = True
       MSComm1.PortOpen = True
       Frame4.Enabled = True
       OnCommu.Enabled = False
    End Sub
```

```
'<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< Recieve and Send to
Tank,txtShow>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
Private Sub IText4_Change()
   I = IText4
End Sub
Private Sub PText3_Change()
   P = PText3
End Sub
Private Sub DText5_Change()
   D = DText5
End Sub
Private Sub SV_Change()
   Image1.Top = 1060 + 40 * (100 - SV)
   VScroll3 = SV
   Recorder.Text1 = SV
End Sub
Private Sub txtVOpen_Change()
   valveOpen = txtVOpen
   VScroll1 = txtVOpen
End Sub
Private Sub VScroll1_Change()
   valveOpen = VScroll1
   txtVOpen.Text = VScroll1
End Sub
Private Sub VScroll3_Change()
   Recorder.Text1 = VScroll3
   SV.Text = VScroll3
   SV = VScroll3
   Image1.Top = 1060 + 40 * (100 - SV)
End Sub
```

Recorder.frm

```
Private Sub cmdCal_Click()
    Gain = step(count1) / SV
    K = Round(Gain, 3)
    c2 = step(count1) * 0.632
    c2 = Round(c2, 0)
    c1 = step(count1) * 0.283
    c1 = Round(c1, 0)
  For num1 = 1 To count1
   If (Round((step(num1)), 0)) = c1 And firm1 <> 2 Then
      t1 = num1 * 0.2
      firm1 = 2
      txtt1.Text = t1
    End If

    If (Round((step(num1)), 0)) = c2 And firm2 <> 2 Then
      t2 = num1 * 0.2
      firm2 = 2
      txtt2.Text = t2
    End If
  Next num1
    t = (t2 - t1) * 1.5
    t = Round(t, 2)
    t0 = t2 - t
    txtt0.Text = t0
    txtK.Text = K
    txtT.Text = t
End Sub
Private Sub cmdClear_Click()
  count1 = 0
  count2 = 0
```

```
    x = 0
    Timer1.Enabled = False
    Cls
End Sub
Private Sub Command1_Click()
    plantGraphic.Show
End Sub
Private Sub Command2_Click()
Setting.Show
End Sub
Private Sub Timer1_Timer()
    'Static step(2600) As Double
    count1 = count1 + 1
    pvin = DataIn
        If pvin < 0 Then
            pvin = 0
        End If
    SV = plantGraphic.SV.Text
    y2 = SV * 58
    step(count1) = pvin
    y = step(count1) * 58
    y3 = (100 - MV) * 58
    PSet (360 + x, 6720 - y2), QBColor(9)   ' SP
    PSet (360 + x, 6720 - y3), QBColor(13)  ' MV
    PSet (360 + x, 6720 - y), QBColor(12)   ' PV
    ScaleMode = 1
    DrawWidth = 3

    x = x + 3.6
'************ Fit3 ***********************
    If count1 = 2500 Then
        plantGraphic.CT1S.Enabled = False
```

```
plantGraphic.CT1R.Enabled = False
plantGraphic.CT2S.Enabled = False
plantGraphic.CT2R.Enabled = False
   Gain = step(count1) / SV
   K = Round(Gain, 3)
   c2 = (step(count1 - 1)) * 0.632
   c2 = Round(c2, 0)
   c1 = (step(count1 - 1)) * 0.283
   c1 = Round(c1, 0)

For num1 = 1 To count1
   If (Round((step(num1)), 0)) = c1 And firm1 <> 2 Then
      t1 = num1 * 0.2
      firm1 = 2
   txtt1.Text = t1
   Else
   End If
   If (Round((step(num1)), 0)) = c2 And firm2 <> 2 Then
      t2 = num1 * 0.2
      firm2 = 2
      txtt2.Text = t2
   Else
   End If
Next num1
   t = ((t2 - t1) * 3) / 2
   t0 = (t2 - t)
   t0 = Round(t0, 2)
   t = Round(t, 2)
'------------------------- Display ------------------
   txtt0.Text = t0
   txtK.Text = K
   txtT.Text = t
```

```
        Setting.Text1 = t0
        Setting.Text2 = K
        Setting.Text3 = t
    '-------------------------------------------------------
      Timer1.Enabled = False
    Else
    End If
End Sub

Private Sub Command2_()
Setting.Show
End Sub

Private Sub txtt0_Change()
    txtt0 = t0
End Sub
```

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้เสร็จสิ้นลงด้วยดี ก็เพราะด้วยความอนุเคราะห์ของบุคคลหลายท่าน ผู้จัดทำขอขอบพระคุณบิดา มารดา ของผู้จัดทำที่ให้การอุปการะและเป็นกำลังใจให้ผู้จัดทำเสมอมา

ขอขอบคุณคณาจารย์ทุกท่าน ที่ได้ประสิทธ์ ประสาท วิชาความรู้ และให้คำแนะนำแก่ผู้จัดทำตลอดมา

ขอขอบคุณ ผศ.พรสุข รติโรจน์อนันต์ อาจารย์ที่ปรึกษาที่ได้กรุณาให้ความรู้ คำแนะนำ และเป็นกำลังใจแก่ผู้จัดทำมาโดยตลอด

ขอขอบคุณ อ.สว่าง เลิศถิรสุนทร ที่ได้ให้คำแนะนำในการทำปริญญานิพนธ์

ขอขอบคุณ ดร.เกียรติศักดิ์ คมวัชระ ที่ได้ให้คำแนะนำในการใช้งานวงจรต่างๆแก่ผู้จัดทำ

ขอขอบคุณ คุณอดิศัย จีนลอย Assistance Manager แผนกCustomer Support Center และพี่ๆ ที่บริษัท โยโกกาวา(ประเทศไทย) จำกัด ที่ให้คำปรึกษาและให้ความช่วยเหลือในการหาข้อมูลเกี่ยวกับเครื่องมือวัดและควบคุมต่างๆ

ขอขอบคุณ คุณสิทธิชัย โลจน์รัศมีกุล ที่ให้คำปรึกษาและให้ความช่วยเหลือในการทำปริญญานิพนธ์

ขอขอบคุณ คุณสุทธิรักษ์ ชุ่มจิต ที่ให้คำปรึกษาและให้ความช่วยเหลือในการใช้งานไมโครคอนโทรลเลอร์ PIC

คณะผู้จัดทำ

บรรณานุกรม

1. กิตติ ภักดีวัฒนะกุล, จำลอง ครูอุตสาหะ, "Visual Basic 6 ฉบับโปรแกรมเมอร์", หจก.ไทยเจริญการพิมพ์, 2542.
2. สมศักดิ์ กีรติวุฒิเศรษฐ์, "หลักการและการใช้งานเครื่องมือวัดในอุตสาหกรรม", สมาคมส่งเสริมเทคโนโลยี (ไทย-ญี่ปุ่น), 2542.
3. วรพจน์ กรแก้ววัฒนกุล, ชัยวัฒน์ ลิ้มพรจิตวิไล, "เรียนรู้และปฏิบัติการไมโครคอนโทรลเลอร์ MCS-51 ฉบับ AT89C5x ของ Atmel ", บริษัท อินโนเวตีฟ เอ็กซ์เพอริเมนต์ จำจัด, กรุงเทพฯ.
4. สุพดี พนาดำรง, สุรวุฒิ แพรศรี, "การควบคุมกระบวนการ 1", ปริญญานิพนธ์คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง สาขาวิศวกรรมระบบกรรมคุม, 2544.
5. การะเกด จันทสังข์, สิทธิชัย โลจน์รัศมีกุล, "การควบคุมกระบวนการ", ปริญญานิพนธ์คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง สาขาวิศวกรรมระบบกรรมคุม, 2545.
6. พิชิต ไวทย์เลิศสุกล, สารัติ วัฒนปกรณ์, สุทธิรักษ์ ชุ่มจิต, "เครื่องบันทึกและแสดงผลอุณหภูมิไร้สาย", ปริญญานิพนธ์คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง สาขาอิเล็กทรอนิกส์, 2545.
7. Carlos A. Smith, Armando B. Corripio, "Principles and Practice of Automatic Process Control.", 2nd ed., John Wiley & Sons, Inc., 1997.
8. Katsuhiko Okata, "Modern Control Engineering." , 3rd ed., Prentice-Hall, Inc., New Jersey, 1997.