เครื่องควบคุมที่โปรแกรมได้บนคอมพิวเตอร์ส่วนบุคคล
PROGRAMMABLE CONTROLLER ON PC

โดย

นาย เกษมสันต์ เครือธร

นาย สุทธิพร กิตติโชคดำรง

นาย ภารดร จันทร์ฉวี

วัน เดือน ปี 1 ต.ค. 2540
เลขทะเบียน 037147
เลขเรียกหนังสือ T 38340 ก813 ค.

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชา เทคโนโลยีการวัดคุมทางอุตสาหกรรม

คณะ วิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2538

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

037147

ภาควิชา : เทคโนโลยีการวัดคุมทางอุตสาหกรรม

สาขาวิชา : วิศวกรรมการวัดคุมทางอุตสาหกรรม

คณะ : วิศวกรรมศาสตร์

สถาบัน : เทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

เรื่อง : เครื่องควบคุมที่โปรแกรมได้บนคอมพิวเตอร์ส่วนบุคคล
PROGRAMMABLE CONTROLLER ON PC

จัดทำโดย : 1. นาย เกษมสันต์ เครือธร 36.013276

2. นาย สุทธิพร กิตติโชคดำรง 36.013313

3. นาย ภารดร จันทร์ฉวี 36.013296

..... อาจารย์ที่ปรึกษา

(อาจารย์ วิริยะ กองรัตน์)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

บทคัดย่อ

บทนำ

บทที่ 1 : ไมโครคอมพิวเตอร์ IBM & PC หน้า 1

บทที่ 2 : ความรู้เบื้องต้นเกี่ยวกับระบบ
DATA ACQUISITION หน้า 12

บทที่ 3 : การใช้งาน PCL-818 หน้า 18

บทที่ 4 : ระบบควบคุม หน้า 30

บทที่ 5 : ผลการทดลอง หน้า 63

กิตติกรรมประกาศ

หนังสืออ้างอิง

ภาคผนวก : โปรแกรมคอมพิวเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เครื่องควบคุมที่โปรแกรมได้บนคอมพิวเตอร์ส่วนบุคคล

PROGRAMMABLE CONTROLLER ON PC

อ. วิริยะ กองรัตน์

อาจารย์คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

บทคัดย่อ

บทความนี้เสนอเครื่องควบคุมสำหรับกระบวนการทางอุตสาหกรรมที่โปรแกรมได้ ซึ่งเป็นเครื่องควบคุมแบบป้อนกลับที่มีฟังก์ชั่นการควบคุมเป็นแบบสัดส่วนร่วมกับ อินทริกัล และ เดอริเวทีฟ (PROPORTIONAL + INTEGRAL + DERIVATIVE : PID) สามารถทำการควบคุมกระบวนการแบบป้อนกลับได้เป็นจำนวนสามวงรอบ ความสามารถในการโปรแกรมได้ก็เพื่อที่จะทำการจัดวางตำแหน่งของทางเข้าและทางออกของสัญญาณควบคุมให้มีความยืดหยุ่นได้ตามที่ต้องการ มีฟังก์ชั่นทางคณิตศาสตร์สนับสนุนเพื่อการวัดและประมวลผลของตัวแปรในกระบวนการและ สามารถที่จะรีโมตพารามิเตอร์ต่างๆ ให้กับคอมพิวเตอร์สำหรับการควบคุมแบบกระจาย

ABSTRACT

THIS PAPER PRESENTS. PROGRAMMABLE CONTROLLER FOR FEEDBACK PROCESS CONTROLS. THIS SYSTEM CONSIST OF 3 INDEPENDENT CONTROLLERS; EACH CONTROLLER CAN BE WORK AS P, PI, PD AND PID. MATHEMATIC FUNCTION FOR PROCESS VARIABLES PROCESSING AND FLEXIBILITY OF INPUT-OUTPUT ARE MANIPULATED BY PROGRAMMABLE TERMS. HOWEVER OTHER PARAMETER CAN BE REMOTED TO COMPUTER FOR DISTRIBUTION CONTROLS.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทนำ

เครื่องควบคุมสมัยใหม่จะเป็นแบบที่สามารถโปรแกรมได้เพื่อให้การใช้งานเป็นไปอย่างสะดวกสามารถที่จะทำการควบคุมกระบวนการได้อย่างกว้างขวางและมีการสื่อให้ข่าวสารแก่ผู้ปฏิบัติงานได้อย่างครบถ้วนและที่สำคัญความสามารถในการโปรแกรมฟังก์ชั่นทางคณิตศาสตร์อันได้แก่ คูณ,หาร,บวก,ลบ,ถอดรากที่สองและอื่นๆเป็นสิ่งจำเป็นสำหรับการวัดค่าตัวแปรต่างๆที่เป็นฟังก์ชั่นกันในกระบวนการจึงเป็นที่มาของงานวิจัยเรื่องนี้และจะขอกล่าวถึงแนวความคิดและเหตุผลต่างๆในทางปฏิบัติและในทฤษฎีสักเล็กน้อยก่อนที่นำไปสู่เรื่องราวของเครื่องควบคุมที่โปรแกรมได้ต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

สถาปัตยกรรมของ 80486

80486 เป็นไมโครโปรเซสเซอร์ขนาด 32 บิตที่มีส่วนของซีพียูที่สมบูรณ์เหมือนเช่น80386 พร้อมด้วยหน่วยควบคุมและการจัดการหน่วยความจำหน่วยคำนวณตัวเลขโฟลตติ้งพอยต์และหน่วยความจำแคชอยู่ภายใน

อินเทลได้เพิ่มเติมขีดความสามารถให้กับ 80486 โดยที่ 80386 ยังคงเป็นสับเซ็ต โดยทุกคำสั่งของ 80386 จะใช้กับ 80486 ได้ และขยายขีดความสามารถโดยเพิ่มเติมคำสั่งใหม่เพื่อให้ทำการประยุกต์ใหม่ๆได้ โดยเทียบได้กับการนำเอา 80387 เข้ามาไว้ในชิพด้วย การทำงานโดยรวมจะทำได้เร็วกว่า เพราะมีการออกแบบโดยการให้มีหน่วยความจำแคชขนาด 8 กิโลไบต์ การออกแบบ 80486 เป็นเทคนิคการลดวงรอบคำสั่งให้มีความเร็วและประสิทธิภาพเหนือ 80386 ถึงกว่าสองเท่า

การจัดการหน่วยความจำประกอบด้วย การจัดการเซกเมนต์และการกำหนดเพจการกำหนดเซกเมนต์ใน 80486 ได้ดำเนินการให้จัดการแอดเดรสในรูปของลอจิคัลให้มีขนาดใหญ่และสามารถใช้ได้อย่างดีทั้งที่สามารถทำให้เป็นการทำงานแบบมัลติทาสกิ้ง โดยมีการใช้ร่วมกันในเรื่องทรัพยากรบางส่วนได้ ส่วนการจัดการเซ็กเมนต์จะกระทำแบบให้ผู้ใช้มองไม่เห็นโดยความสามารถแยกเซกเมนต์ย่อยที่มีขนาด 4 กิโลไบต์

ขนาดของหน่วยความจำได้รับการจัดแบ่งให้เป็นเซกเมนต์ที่มีขนาดแตกต่างกัน โดยแต่ละเซกเมนต์มีขนาดใหญ่ถึง 4 จิกะไบต์ (2^{32}) แต่ละเซกเมนต์จะมีแอตทริบิวต์ที่บอกถึงการทำงาน ร่วมกัน ซึ่งประกอบด้วย ตำแหน่ง ขนาด ชนิด (สแตก รหัส ข้อมูล) และคุณสมบัติในเรื่อง PROTECTION ยังสนับสนุนการทำงานแบบมัลติทาสกิ้งได้อย่างดีโดยแบ่งโครงสร้างหน่วยความจำรองรับโดยให้แต่ละงานใช้หน่วยความจำได้สูง 16381 เซ็กเมนต์ ถึงขนาด 4 จิกะไบต์ หรือทำให้ใช้หน่วยความจำแบบเทียม (VIRTUAL)ในแต่ละงานได้ถึง 64 เทอราไบต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เซ็ตรีจิสเตอร์

80486 มีรีจิสเตอร์ที่เป็นของ 80386 และของ 80387 รวมกับรีจิสเตอร์เหล่านี้แบ่งออกเป็นหลายกลุ่มดังนี้

รีจิสเตอร์พื้นฐานทางสถาปัตยกรรมของ 80486 ประกอบด้วย

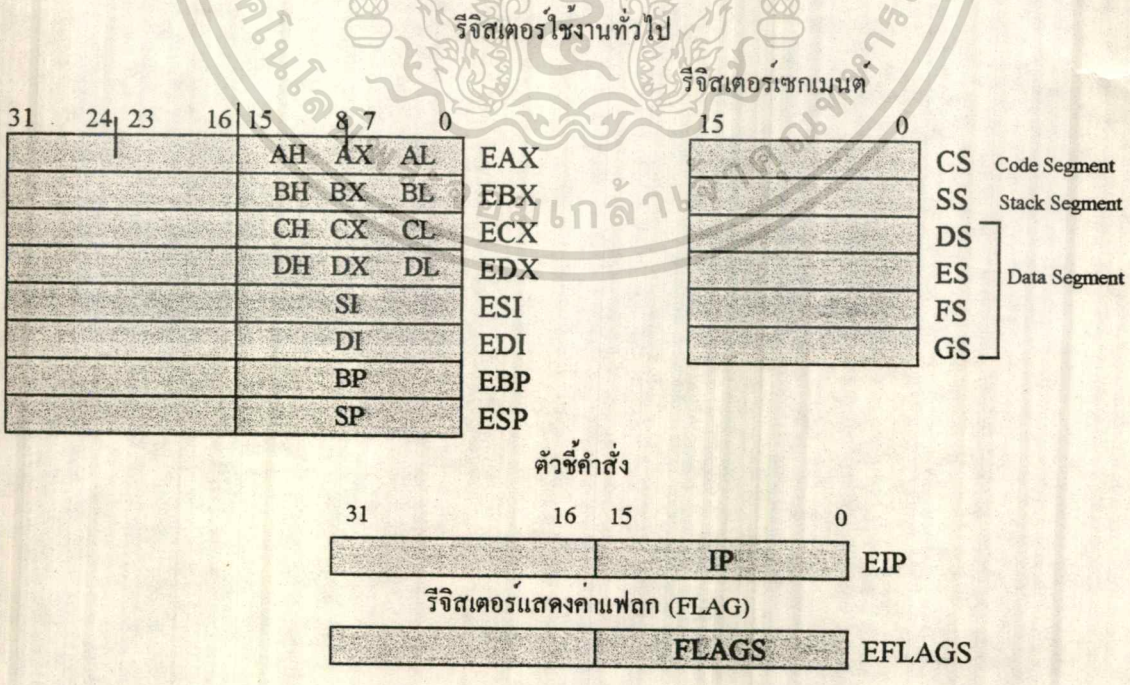
- รีจิสเตอร์ใช้งานทั่วไป (GENERAL PURPOSE REGISTERS)
- ตัวชี้คำสั่ง (INSTRUCTION POINTER)
- รีจิสเตอร์แสดงแฟลก (FLAG REGISTER)
- รีจิสเตอร์กำหนดเซกเมนต์ (SEGMENT REGISTER)

รีจิสเตอร์ที่แบ่งระดับของระบบ ประกอบด้วย

- รีจิสเตอร์ควบคุม (CONTROL REGISTER)
- รีจิสเตอร์แสดงแอดเดรสของระบบ (SYSTEM ADDRESS REGISTER)

รีจิสเตอร์ที่เกี่ยวกับตัวเลขโฟรตติ้งพอยต์ ประกอบด้วย

- รีจิสเตอร์ข้อมูล (DATA REGISTERS)
- แทกเวิร์ด (TAG WORD)
- เวิร์ดแสดงสถานะ (STATUS WORD)
- ตัวชี้คำสั่งและข้อมูล (INSTRUCTION AND DATA POINTERS)
- เวิร์ดควบคุม (CONTROL WORD)
- รีจิสเตอร์ดีบักและทดสอบ (DEBUG AND TEST REGISTERS)



รูปที่ 1 รีจิสเตอร์ที่ใช้งานทั่วไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รีจิสเตอร์พื้นฐาน

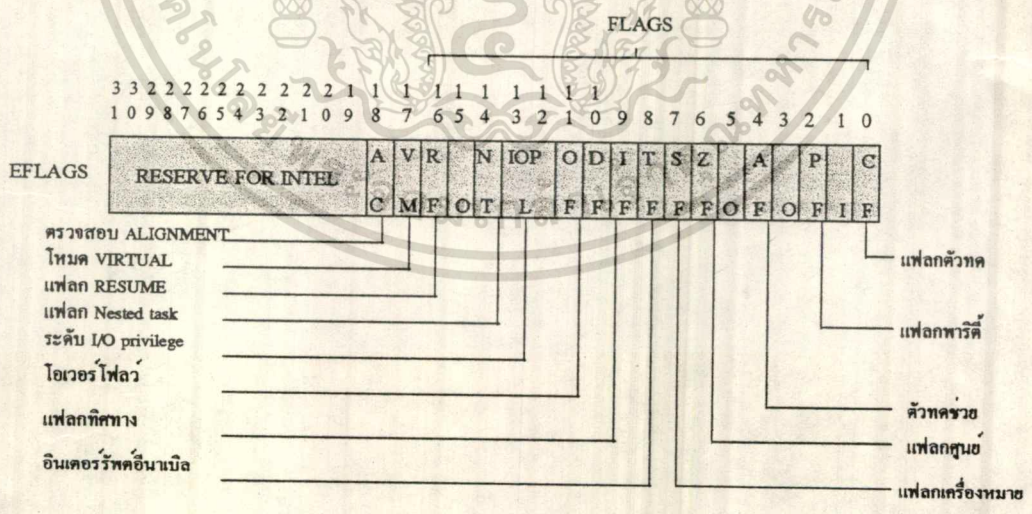
รูปที่ 1 เป็นรีจิสเตอร์พื้นฐานที่ใช้ใน 80486 ข้อมูลที่เก็บในรีจิสเตอร์เหล่านี้เป็นชุดรีจิสเตอร์ที่ใช้ทำงานเฉพาะงานหนึ่งและสามารถโหลดค่ารีจิสเตอร์จากงานอื่นเข้ามาดำเนินการในลักษณะมัลติทาสก์ก็ได้ โดยการสวิตช์ระหว่างงาน

รีจิสเตอร์ที่ใช้งานกันอยู่ทั่วไป มีด้วยกัน 8 รีจิสเตอร์ แต่ละตัวมีขนาด 32 บิตรีจิสเตอร์เหล่านี้มีไว้สำหรับเก็บค่าข้อมูลและแอดเดรส การทำงานกับรีจิสเตอร์ต่างๆยังสามารถกำหนดให้เป็นโอเปอแรนด์ได้โดยเป็นโอเปอแรนด์ตั้งแต่ขนาดตั้งแต่ 1 บิต 8, 16 และ 32 บิต และกำหนดรายละเอียดบิตฟิลด์ตั้งแต่ 1 ถึง 32 บิต ส่วนการกำหนดแอดเดรสโดยใช้รีจิสเตอร์เหล่านี้จะใช้แบบ 16 และ 32 บิตก็ได้ รีจิสเตอร์ 32 บิต มีชื่อEAX, EBX, ECX, EDX, ESI, EDI, EDP, และ ESP

การใช้งานรีจิสเตอร์เหล่านี้จะแยกใช้เฉพาะ 16 บิตล่างก็ได้ ซึ่งจะทำให้เหมือนกับ 8086 โดยตรง โดยใช้ชื่อ AX, BX, CX, DX, SI, DI, BPและ SP โดยเมื่อใช้ 16 บิตล่างและ 16 บิตบนจะไม่มีการเปลี่ยนแปลงใดๆ

หากรีจิสเตอร์ขนาด 8 บิต ก็เรียกได้เหมือน 8086 อีกเช่นกัน คือแยก AX, BX, CX, DX, ออกเป็น AL, BL, CL, DL, และส่วน 8 บิต บนเป็น AH, BH, CH, DH

ตัวชี้คำสั่ง ตัวชี้คำสั่ง หรือ IP ได้เพิ่มเติมจากเดิมเป็น EIP ซึ่งมีความยาว 32 บิต EIP จะเก็บค่าออฟเซ็ตของแอดเดรสที่จะไปเฟตช์คำสั่งต่อมาทำงาน ค่าออฟเซตโดยทั่วไป เปรียบเทียบกับค่า CS โดยส่วนล่าง 16 บิต จะเรือก IP และใช้ในการอ้างแอดเดรสเหมือน 8086



รูปที่ 2 รีจิสเตอร์แฟลก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แฟลก แฟลกของ 80486 มีขนาด 32 บิต โดยเปลี่ยนชื่อมาเป็น EFLAGS ซึ่ง 16 บิตล่างยังคงเหมือนเดิมที่ใช้กับ 80286 โครงสร้างของ EFLAGS แสดงดังรูปที่ 2

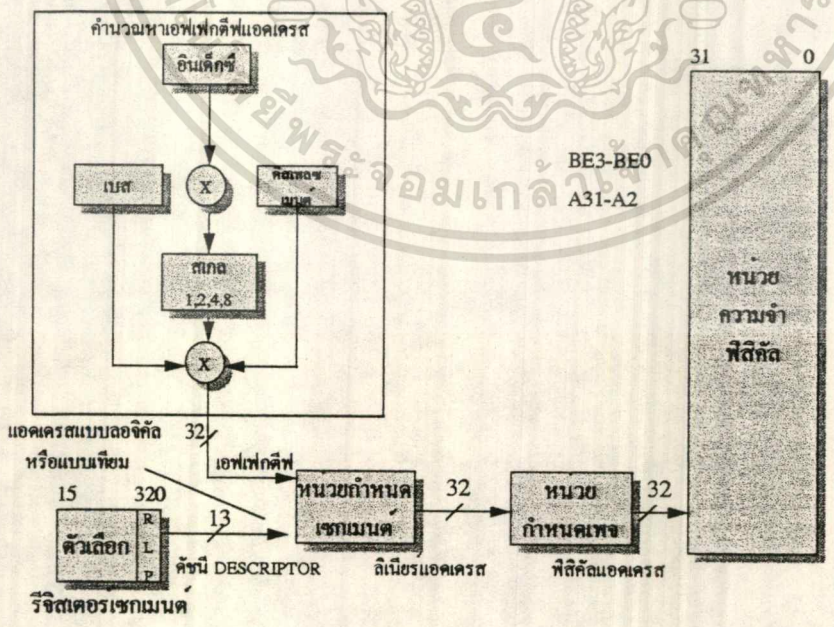
ข้อน่าสังเกตคือ แฟลกใน 16 บิตบนนั้น อินเทลยังสงวนไว้ โดยยังไม่กำหนดบิตการใช้งานใดๆ คงมีเรื่องการ ALIGNMENT ที่เพิ่มให้มา

เซกเมนต์รีจิสเตอร์ เป็นรีจิสเตอร์ที่สำคัญมีขนาดทั้งหมด 6 รีจิสเตอร์ โดยรีจิสเตอร์แต่ละตัวเป็นขนาด 16 บิต ในโหมดปกติที่เรียกว่า REAL MODE การใช้รีจิสเตอร์เหล่านี้ก็คงเหมือนปกติใน 8086 แต่ในโหมดป้องกัน (PROTECTED MODE) จะมีการกำหนดเซกเมนต์ใหม่โดยในแต่ละเซกเมนต์สามารถกำหนดได้จากหนึ่งไบต์ไปจนสูงสุดตามขอบเขตของแอดเดรสทั้งหมดคือ 2^{32} หรือ 4 จิกะไบต์

ใน REAL MODE การกำหนดเซกเมนต์จะกำหนดได้เพียงเซกเมนต์ละ 64 กิโลไบต์โดยการกำหนดค่าใน CS เพื่อเลือกเซกเมนต์รหัสคำสั่ง DS เพื่อเลือกเซกเมนต์ข้อมูลSS เพื่อเลือกเซกเมนต์สแตก

การติดต่อกับหน่วยความจำโดยใช้เซกเมนต์รีจิสเตอร์

เมื่อให้ 80486 ติดต่อกับหน่วยความจำ โดยที่มีแอดเดรสบัสถึง 32 เส้น แต่การใช้งานใน REAL MODE ยังคงเหมือน 8086 การจัดการหน่วยความจำในโหมดป้องกันจะทำให้การติดต่อกับหน่วยความจำได้ในรูปแบบใหม่



รูปที่ 3 การแปลงแอดเดรสของ 80486

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กลไกการแปลแอดเดรสแสดงได้ดังรูปที่ 3

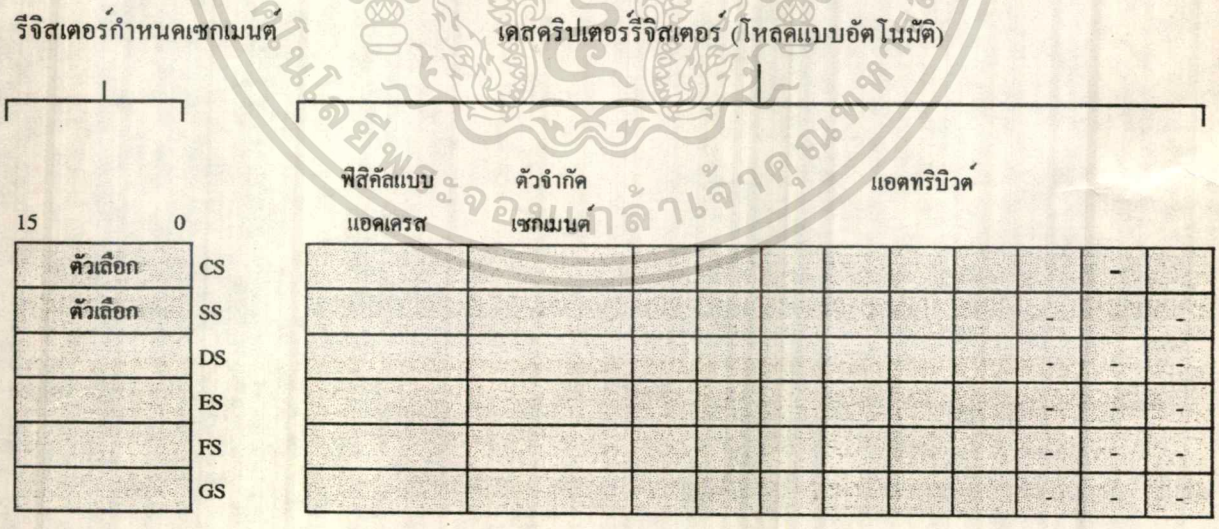
โครงสร้างหน่วยความจำของ 80486 แบ่งโครงสร้างทางการเข้าถึงข้อมูลได้เป็นแบบ 8 บิต 16บิต และ 32 บิต ขอบเขตของการอ้างแอดเดรสเป็นได้ทั้งการเข้าถึงหน่วยความจำ โดยมีขอบเขตของหน่วยความจำเป็นแบบลอจิคัลแอดเดรส ลิเนียร์แอดเดรส และฟิสิคัลแอดเดรส การอ้างอิงแอดเดรสแบบลอจิคัลอาจเรียกอีกอย่างหนึ่งว่าหน่วยความจำแบบเทียม (VIRTUAL MEMORY)

การอ้างแอดเดรสของ 80486 ใช้เซกเมนต์รีจิสเตอร์เดิมเปลี่ยนหน้าที่เป็นตัวเลือก (SELECTOR) และใช้ออฟเซตที่เป็นผลมาจากเบส อินเด็กซ์หรือดิสเพลซเมนต์ ซึ่งมีวิธีการในการกำหนดต่อไป

การจัดการโดยใช้แคชรีจิสเตอร์สำหรับเดสคริปเตอร์เซกเมนต์

ในโหมดป้องกันที่ต้องการติดต่อกับหน่วยความจำ 80486 จะเปลี่ยนเซกเมนต์รีจิสเตอร์ใน REAL MODE ไปเป็นตัวเลือก (SELECTOR) ตัวเลือกจะชี้ไปยังเดสคริปเตอร์ซึ่งแต่เดิมอยู่ในหน่วยความจำ การโหลดข้อมูลนี้จะเกิดขึ้นอย่างอัตโนมัติโดยการนำข้อมูลมาเก็บไว้ในแคชภายในเพื่อใช้ในการทำงานสำหรับคำถามและสร้างค่าแอดเดรสที่จะใช้ต่อไป

โครงสร้างของแคชรีจิสเตอร์ที่เก็บไว้ภายในสำหรับการเก็บเดสคริปเตอร์แสดงดังรูปที่ 4



รูปที่ 4 เซกเมนต์รีจิสเตอร์และส่วนสัมพันธ์กับเดสคริปเตอร์รีจิสเตอร์

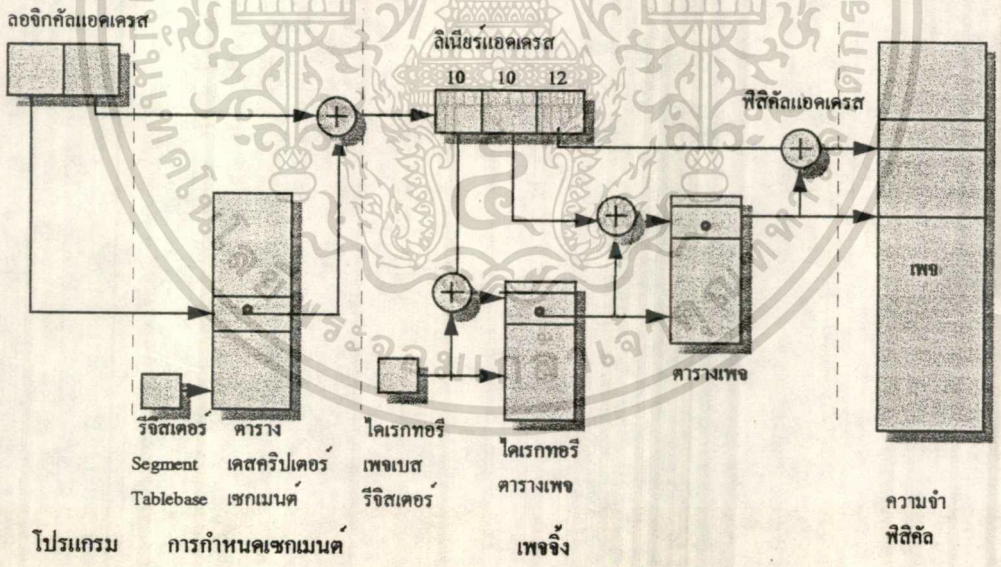
ขนาดของเดสคริปเตอร์รีจิสเตอร์ซึ่งเป็นแคชรีจิสเตอร์อยู่ภายในประกอบด้วยฟิสิคัลเบสแอดเดรสขนาด 32 บิต ตัวจำกัดเซกเมนต์ 32 บิต และแอตทริบิวต์อื่นที่เกี่ยวข้อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การทำงานเริ่มจากการกำหนดค่าลงในเซกเมนต์รีจิสเตอร์ ทันทีที่กำหนดค่าลงในเซกเมนต์รีจิสเตอร์ ก็จะมีการโหลดข้อมูลเดสคริปเตอร์มาใส่ในแคชรีจิสเตอร์ ที่สัมพันธ์กันอย่างอัตโนมัติ การทำงานใน REAL MODE จะปรับปรุงได้เฉพาะในเบสแอดเดรส โดยที่ค่าในตัวจำกัดเซกเมนต์และแอตทริบิวต์จะคงที่ฟิสิคัลเบสแอดเดรสจะเป็นตัวกำหนดเซกเมนต์ใน REAL MODE เมื่อเป็นโหมดป้องกันค่าเบสแอดเดรสค่าในตัวจำกัดเซกเมนต์และแอตทริบิวต์

การกำหนดแอดเดรสโดยมองจากผู้ใช้เป็นหลัก การจัดการกับแอดเดรสขึ้นอยู่กับคำสั่งที่ใช้ การใช้คำสั่งจึงมองเห็นหน่วยความจำเป็นแบบลอจิคัลแอดเดรส ผู้เขียนโปรแกรมจะอ้างอิงแอดเดรสในลักษณะนี้ โดยส่วนของการอ้างแอดเดรสแบบลอจิคัลแอดเดรสนี้จะแบ่งแยกออกโดยประกอบด้วยค่าตัวเลือก(SELECTOR) และออฟเซตSELECTOR จะเป็นตัวกำหนดขอบเขตของแอดเดรสที่ใช้ ส่วนค่าออฟเซตจะเป็นตัวบอกค่าแอดเดรสที่อยู่ในขอบเขตของแอดเดรสที่กำหนดโดยตัวเลือกค่าออฟเซตมีค่า 32 บิต ดังนั้นค่าแอดเดรสที่อ้างอิงได้จึงมีค่าได้สูงสุดถึง 4 จิกะไบต์

หากพิจารณากลไกการทำงานจะพบว่าโปรแกรมจะมีการอ้างอิงแอดเดรสโดยกำหนดค่าตัวเลือก ซึ่งค่านี้จะอ้างอิงเข้าไปเพื่อการกำหนดเซกเมนต์ โดยเทียบกับ SEGMENT TABLE BASE REGISTER ซึ่งก็เป็นการกำหนดเอาตารางเดสคริปเตอร์เพื่อเก็บเข้าไปแคชรีจิสเตอร์ภายใน ค่าที่ได้จากเดสคริปเตอร์นี้จะเป็นตัวกำหนดรายละเอียดร่วมกับค่าออฟเซตที่มาจากโปรแกรม ได้ลิเนียร์แอดเดรสขนาด 32 บิต ดังรูปที่ 5



รูปที่ 5 การกำหนดแอดเดรสของ 80486

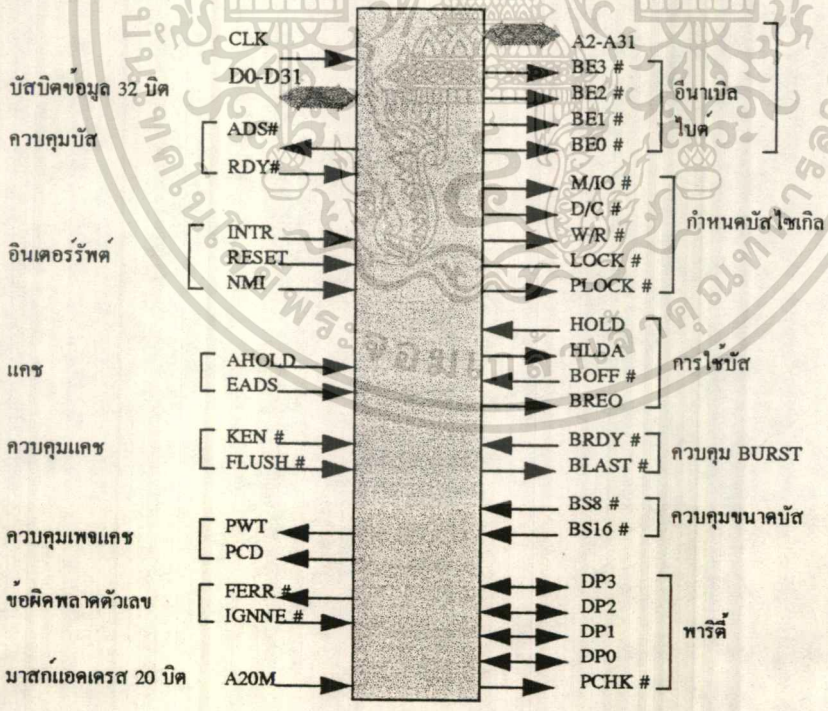
ค่าแอดเดรส 32 บิต แบบลิเนียร์ แอดเดรสนี้จะได้รับการควบคุมการทำงานโดย MMU (หน่วยจัดการหน่วยความจำ) เพื่อทำการกำหนดเพจ ซึ่งจะมีวิธีการกำหนดการอ้างอิงแบบเพจเพื่อกำหนดเข้าหาหน่วยความจำจริงของระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเชื่อมต่อ

80486 เป็นชิพไมโครโปรเซสเซอร์ที่ได้รับการออกแบบให้มีโครงสร้างการเชื่อมต่อคล้ายกับ 80386 ดังนั้นโครงสร้างบัสจึงเหมือนกับ 80386 แต่มีการเพิ่มขีดความสามารถพิเศษทางฮาร์ดแวร์ที่จะทำให้การเชื่อมต่อมีการทำงานฟังก์ชันพิเศษเพิ่มขึ้น เช่นการใช้สัญญาณนาฬิกาความเร็วเท่ากับสัญญาณนาฬิกาของระบบเป็นสัญญาณเบิรสต์ (BURST) เพื่อการนำข้อมูลเข้าใส่ในหน่วยความจำแคชภายใน โครงสร้างการเชื่อมต่อกับหน่วยความจำแคช และโปรเซสเซอร์ทางคณิตศาสตร์ได้ยุบไว้ภายในบางส่วน

80486 ใช้สัญญาณ 1 x CLOCK ซึ่งแตกต่างจาก 80386 ที่ใช้สัญญาณนาฬิกา 2 x CLOCK จะเห็นว่า 80386 ที่ใช้กับสัญญาณ 25 เมกะเฮิรตซ์ จะต้องใช้ความเร็วของสัญญาณนาฬิกาถึง 50 เมกะเฮิรตซ์ 80486 เหมือนกับ 80386 ในเรื่องการแยกบัสระหว่างบัสข้อมูลกับบัสแอดเดรสออกจากกัน บัสข้อมูลมีจำนวน 32 เส้น ส่วนบัสแอดเดรสแยกออกมาเป็นสองส่วน ส่วนหนึ่งมีจำนวน 30 เส้น คือ A_2-A_{31} และอีกส่วนหนึ่งคือ สายสำหรับการอีนาเบิลไบต์ (BEO # -BE3 #) จำนวน 4 เส้น ส่วนของการอ้างแอดเดรสที่ติดต่อกับหน่วยความจำภายนอกเหมือนกับ 80386 ดังนั้นการอ้างแอดเดรสทั้งหมดจึงใช้ 30 บิต และเลือกโหมดการใช้แอดเดรสด้วยสัญญาณ 4 เส้น สายสัญญาณแอดเดรส A_2-A_{31} เป็นสายสัญญาณแบบ 2 ทิศทาง โครงสร้างของสายสัญญาณทั้งหมดของ 80486 แสดงดังรูปที่ 6



รูปที่ 6 การจัดกลุ่มสัญญาณของ 80486

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

80486 มีกลไกในการนำข้อมูลจากภายนอกใส่ในหน่วยความจำแคชภายในด้วยความเร็วสูง ซึ่งสามารถทำได้ด้วยอัตราความเร็วเท่ากับความเร็วของสัญญาณนาฬิกา เพื่อให้เข้าใจการเชื่อมต่อกับภายนอก จึงควรทำความเข้าใจกับสัญญาณต่างๆ ก่อน ดังนี้

CLOCK-CLK เป็นสัญญาณกำหนดจังหวะการทำงานพื้นฐานให้กับ 80486 สัญญาณนี้จะต้องมีเสถียรภาพดี เพราะภายในชิพจะแปลงสัญญาณนี้เป็นสัญญาณแบบสองเฟส และให้ระดับตามมาตรฐานทีทีแอล

บัสแอดเดรส A_2-A_{31} และ BE0# BE3# เป็นสัญญาณกลุ่มแอดเดรส โดยมี BE0#-BE3# แอกทีฟที่ลอจิก "0" เป็นสัญญาณกำหนดแอดเดรสของหน่วยความจำหลัก และพอร์ตอินพุต/เอาต์พุต โดยที่ 80486 สามารถอ้างอิงแอดเดรสโดยตรงได้จากแอดเดรส 00000000H ถึง FFFFFFFFH หรือ 4 กิกะไบต์ สำหรับ BE0#-BE3# เป็นตัวกำหนดตำแหน่งไบต์ โดย

BE3# ใช้บอกไบต์ D_{24} - D_{31}
BE2# ใช้บอกไบต์ D_{16} - D_{23}
BE1# ใช้บอกไบต์ D_8 - D_{15}
BE2# ใช้บอกไบต์ D_0 - D_7

บัสข้อมูล (D_{31} - D_0) เป็นสายสัญญาณแบบสองทิศทาง D_0-D_7 เป็นบิตที่มีค่านัยสำคัญต่ำสุด D_{24} - D_{31} เป็นบิตที่มีค่านัยสำคัญสูงสุด การรับส่งข้อมูลสามารถส่งหรือรับด้วยขนาดข้อมูล 8 หรือ 16 บิตได้ โดยใช้วิธีการควบคุมด้วยขาอินพุต BS8# หรือ BS16#

พาริตี้ (PARITY DP_0 - DP_3) 80486 ให้สัญญาณการตรวจสอบแบบสร้างพาริตี้ 4 เส้น แต่ละเส้นจะคุมข้อมูล 8 บิต สัญญาณพาริตี้ได้แก่ DP_0 - DP_3 การตรวจสอบพาริตี้นี้ตรวจสอบแบบพาริตี้คู่ การสร้างพาริตี้จะสร้างขณะมีการเขียนข้อมูลออกจากซีพียู การรับพาริตี้มาตรวจสอบจะเกิดขึ้นขณะอ่านข้อมูลจากภายนอก การตรวจสอบนี้จะมีสถานะที่แฟลกและสัญญาณประกอบที่จะบอกถึงข้อผิดพลาด

สถานะพาริตี้ (PCHK#) เป็นสัญญาณเอาต์พุต หากการตรวจสอบพาริตี้ข้อผิดพลาดสัญญาณขานี้จะมีค่าเป็น "0" ซึ่งจะนำไปใช้งานอื่นต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สัญญาณเกี่ยวกับวงรอบการทำงานของบัส

เอาต์พุต M/IO#, D/C#, W/R# เป็นสัญญาณบอกปรากฏการณ์ที่เกิดขึ้นที่บัสโดย M/IO# เป็นสัญญาณบอกความแตกต่าง ว่าซีพียูกำลังติดต่อหน่วยความจำหรืออินพุตเอาต์พุต D/C# เป็นสัญญาณบอกความแตกต่างระหว่างข้อมูลกับสัญญาณควบคุม W/R# เป็นสัญญาณบอกว่าเป็นการเขียนหรืออ่าน ข้อสรุปของสัญญาณทั้งสามเป็นดังตารางข้างล่าง

M/IO#	D/C#	W/R#	วงรอบการทำงาน
0	0	0	ตอบรับอินเตอร์รัพต์
0	0	1	HALT/วงรอบพิเศษ
0	1	0	อ่านจาก I/O
0	1	1	เขียน I/O
1	0	0	เฟต
1	0	1	สงวนไว้
1	1	0	อ่านหน่วยความจำ
1	1	1	เขียนหน่วยความจำ

ตารางที่ 1 สถานะต่างๆ ของเอาต์พุต M/IO #, D/C #, W/R # ขณะที่ 80486 ทำงานอย่างใดอย่างหนึ่ง

เอาต์พุต สำหรับล็อกบัส (LOCK#) เป็นสัญญาณที่ใช้บอกว่า 80486 กำลังอยู่ในวงรอบการทำงาน READ-MODIFY-WRITE

ADDRESS HOLD REQUEST (AHOLD) สัญญาณเอาต์พุตที่เพิ่มเข้ามาเพื่อให้ 80486 ทำงานได้ไวขึ้น โดยเป็นสัญญาณที่ 80486 ต้องการอ่านค่าโอเปอแรนด์ที่มีขนาดมากกว่า 32 บิต โอเปอแรนด์นี้ต้องการวงรอบการทำงานมากกว่าหนึ่งไซเกิล 80486 ใช้ PLOCK# สำหรับการเขียนอ่านตัวเลขโฟลตติ้งพอยต์ (64บิต) ตัวตารางอธิบายรายละเอียดเซกเมนต์ (64 บิต) หรือการเติมข้อมูลในแคช (128 บิต)

การควบคุมบัส ประกอบด้วย

สถานะแอดเดรส (ADS#) เป็นสัญญาณแสดงสถานะของแอดเดรสและวงรอบการทำงานของบัส

สัญญาณ NON-BURST READY (RDY#) เป็นสัญญาณอินพุต บอกว่าวงรอบการทำงานของบัสขณะนี้ได้สิ้นสุดแล้ว ในช่วงสัญญาณที่ใช้สำหรับการอ่าน RDY# ใช้เป็นสัญญาณบอกว่าระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภายนอกมีข้อมูลเตรียมพร้อมแล้วอยู่ที่บัสข้อมูล ในช่วงเขียนสัญญาณ RDY# เป็นสัญญาณที่ใช้บอกว่าระบบภายนอกได้รับข้อมูลจากการเขียนเรียบร้อยแล้ว

การควบคุม BURST ประกอบด้วย

BURST READY (BRDY#) เป็นสัญญาณจากภายนอกควบคุมจังหวะการ BURST ในวงรอบของการทำงาน BURST ในจังหวะนี้ระบบภายนอกจะบอก 80486 ว่าสายข้อมูลของระบบมีข้อมูลเตรียมพร้อมไว้ให้แล้ว

BURST LAST (BLAST#) เป็นสัญญาณบอกการสิ้นสุด จังหวะของการถ่ายเทข้อมูล

กลุ่มสัญญาณอินเตอร์รัพต์ ประกอบด้วย

รีเซต (RESET) เป็นสัญญาณรีเซตไมโครโปรเซสเซอร์ 80486

อินเตอร์รัพต์แบบมาสเคเบิล (INTR) เป็นสัญญาณจากภายนอกที่ส่งมาอินเตอร์รัพต์ 80486

นอนมาสเคเบิลอินเตอร์รัพต์ (NMI) สัญญาณอินเตอร์รัพต์แบบนอนมาสเคเบิล

กลุ่มสัญญาณควบคุมการใช้บัส ประกอบด้วย

BUS REQUEST (BREQ) เป็นสัญญาณเอาต์พุตที่สร้างจา 80486 ส่งต่อให้กับลอจิกภายนอกเพื่อใช้ในระบบมัลติโปรเซสเซอร์

BUS HOLD REQUEST (HOLD) เป็นสัญญาณจากภายนอกเพื่อขอใช้บัส

BUS HOLD ACKNOWLEDGE (HLDA) เป็นสัญญาณตอบการใช้บัส เพื่อให้อุปกรณ์ภายนอกควบคุมการใช้บัส

BACK OFF (BOFF#) เป็นสัญญาณจากภายนอกที่ส่งเข้ามายัง 80486 เพื่อให้ซีพียูหยุดการควบคุมบัสในวงรอบที่ติดต่อมา

สัญญาณเกี่ยวกับการทำงานกับหน่วยความจำแคช

ADDRESS HOLD REQUEST (AHOLD) เป็นสัญญาณอินพุต ให้ระบบภายนอกโดยสร้างตัวควบคุมบัส แอกเซสบัสเพื่อทำงานในวงรอบการติดต่อกับแคชภายใน

EXTERNAL ADDRESS VALID (EADS#) เป็นสัญญาณอินพุตที่บอกว่ากำลังส่งค่าแอดเดรสจากภายนอกเข้ามา เป็นแอดเดรสของหน่วยความจำแคชภายใน 80486 ผ่านทางขาแอดเดรสของ 80486

อีนาเบิลแคช (KEN#) เป็นสัญญาณใช้ในการอีนาเบิลหน่วยความจำแคช

CACHE FLUSH (FLUSH#) เป็นสัญญาณอินพุตที่บอกให้ 80486 นำข้อมูลเข้าในหน่วยความจำแคชภายใน

PWT และ PCD เป็นสัญญาณเอาต์พุตที่ทำงานเกี่ยวข้องกับการกำหนดเพจที่จะใช้ในหน่วยความจำแคช

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กลุ่มสัญญาณเกี่ยวกับโปรเซสเซอร์คณิตศาสตร์

FLOATING POINT ERROR (FERR#) เหมือนกับสัญญาณ ERROR# ของ 80386 ที่ใช้ต่อกับ 80387 สำหรับใน 80486 นี้เป็นสัญญาณรายงานข้อผิดพลาดจากการคำนวณทางคณิตศาสตร์

IGNORE NUMBERIC ERROR (IGNNE#) เป็นสัญญาณบอกให 80486 ทำงานต่อไปโดยไม่สนใจข้อผิดพลาดที่เกิดขึ้น

กลุ่มสัญญาณควบคุมขนาดบัส ประกอบด้วย

BS16# เป็นสัญญาณบอกการใช้บัสขนาด 16 บิต

BS8# เป็นสัญญาณบอกการใช้บัสขนาด 8 บิต

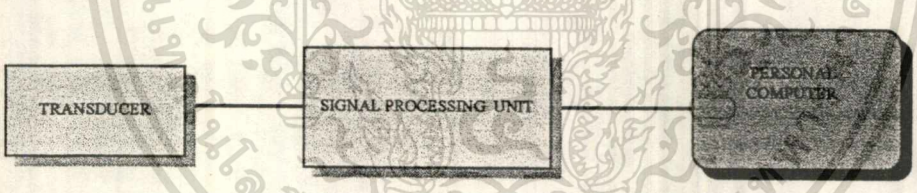
A20M# เป็นสัญญาณอินพุตที่จะบอกให้ 80486 กำหนดกรอบของแอดเดรส 20 บิตก่อนที่จะกระทำการมองหาหน่วยความจำแคชภายใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ความรู้เบื้องต้นเกี่ยวกับระบบ DATA ACQUISITION SYSTEM

ระบบรวบรวมข้อมูล (DATA ACQUISITION SYSTEM) เป็นระบบของเครื่องมือทางอิเลคโทรนิคส์ที่ได้ทำการประยุกต์เอาเครื่องคอมพิวเตอร์มาทำงานร่วมกับอุปกรณ์ตรวจวัด (TRANSDUCER) โดยใช้คอมพิวเตอร์เป็นศูนย์กลางในการเก็บบันทึกข้อมูล (DATA RECORDER) วิเคราะห์ข้อมูล (DATA ANALYSIS) ประมวลผลข้อมูล (DATA PROCESSING) แสดงผลข้อมูล (DATA DISPLAY) และควบคุมระบบการทำงาน (CONTROL) ซึ่งในปัจจุบันมีการใช้กันมากในโรงงานอุตสาหกรรมต่างๆ เช่น โรงกลั่นน้ำมัน โรงงานผลิตกระแสไฟฟ้า ชุมสายโทรศัพท์ โรงงานผลิตชิ้นส่วนอิเลคโทรนิคส์ เป็นต้น เพื่อการควบคุมคุณภาพในการผลิตความรวดเร็วในการตรวจสอบ ความชัดเจนรวดเร็วในการติดตามดูผลการทำงาน (MONITORING) และความรวดเร็วแม่นยำในการควบคุมอุปกรณ์ และเครื่องจักรกลต่างๆ ให้ทำงานตามที่ต้องการ

หลักการทำงานขั้นพื้นฐานของระบบรวบรวมข้อมูล จะประกอบไปด้วยส่วนประกอบหลัก 3 ส่วน คือ ส่วนของตัวตรวจวัด (TRANSDUCER) หน่วยกระบวนการทางสัญญาณทางไฟฟ้า (SIGNAL PROCESSING UNIT) และเครื่องคอมพิวเตอร์ (PERSONAL COMPUTER) ดังแสดงในรูปที่ 1



รูปที่ 1 แสดงหลักการทำงานของระบบรวบรวมข้อมูล

จากรูปที่ 1 แสดงถึงส่วนประกอบหลักของระบบรวบรวมข้อมูลซึ่งจะประกอบไปด้วยอุปกรณ์ตรวจวัด (TRANSDUCER)จะทำหน้าที่รับสัญญาณภายนอกจากจุดที่ต้องการวัดในรูปของอุณหภูมิ (TEMPERATURE) ความดัน (PRESSURE) การไหล (FLOW) แล้วส่งสัญญาณที่วัดได้มายังหน่วยกระบวนการทางสัญญาณไฟฟ้า (SIGNAL PROCESSING UNIT) ทำการปรับระดับสัญญาณ (AMPLIFIER) ลำดับการเลือกสัญญาณ (MULTIPLEXER) และแปลงรูปสัญญาณ (ANALOG TO DIGITAL CONVERTER) ส่งให้เครื่องคอมพิวเตอร์ทำการเก็บข้อมูล (DATA RECORDER) วิเคราะห์ข้อมูล (DATA ANALYSIS) ประมวลผลข้อมูล (DATA PROCESSING) และแสดงผลข้อมูล (DATA DISPLAY) ซึ่งรายละเอียดจะกล่าวในหัวข้อต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลักษณะการออกแบบ

ลักษณะการออกแบบแผงอุปกรณ์ระบบรวบรวมข้อมูล (DATA ACQUISITION CARD) ของ MODEL PCL818 เป็นแผ่นเชื่อมโยงกับคอมพิวเตอร์ จะประกอบไปด้วยวงจรต่างๆ ที่มีคุณสมบัติเฉพาะตัว

ลักษณะการทำงานของระบบ

ระบบรวบรวมข้อมูลอาจจะแบ่งการทำงานออกเป็น 6 ส่วนใหญ่ๆ ตามลำดับขั้นตอนดังต่อไปนี้ คือ

- **ส่วนการตรวจวัด (TRANSDUCER)**

จะทำหน้าที่เป็นอุปกรณ์ตรวจวัดสิ่งต่างๆ ที่ต้องการทราบความเปลี่ยนแปลงซึ่งตัวตรวจวัดจะเปลี่ยนพลังงานความร้อน (TEMPERATURE) ความดัน (PRESSURE) การไหล (FLOW) หรือที่อยู่ในรูปกระแสไฟฟ้า แรงดันไฟฟ้า (CURRENT OR VOLTAGE) ขึ้นอยู่กับชนิดของตัวตรวจวัด เมื่อพลังงานดังกล่าวเกิดการเปลี่ยนแปลงของกระแสไฟฟ้า หรือ แรงดันไฟฟ้าที่ได้จากการตรวจวัดก็จะเกิดการเปลี่ยนแปลงตามอัตราการเปลี่ยนแปลงของไฟฟ้านี้ จะนำไปเป็นข้อมูลของระบบ

- **ส่วนของกระบวนการทางไฟฟ้า (SIGNAL PROCESSING UNIT)**

เป็นส่วนที่จะนำสัญญาณทางไฟฟ้าที่ได้จากอุปกรณ์ตรวจวัดผ่านกระบวนการทางไฟฟ้า ซึ่งในส่วนนี้ยังแบ่งการทำงานออกเป็นส่วนย่อยได้อีก คือ การปรับระดับของสัญญาณทางไฟฟ้า (AMPLIFIER) ลำดับการเลือกสัญญาณ (MULTIPLEXER) แซมเปิ้ลแอนโฮลด์ (SAMPLE AND HOLD) เปลี่ยนรูปสัญญาณ (ANALOG TO DIGITAL CONVERTER) และส่วนเชื่อมโยง (INTERFACE) สัญญาณทางไฟฟ้าที่ออกจากส่วนนี้จะเป็นสัญญาณ DIGITAL ที่พร้อมที่จะส่งให้กับเครื่องคอมพิวเตอร์

- **ส่วนของหน่วยความจำแบบแรม (RAM)**

ย่อมาจากคำว่า RAMDOM ACCESS MEMORY เป็นหน่วยความจำหลักของเครื่องคอมพิวเตอร์ ในขณะใช้งานจะทำหน้าที่เก็บข้อมูลที่ได้จาก SIGNAL PROCESSING UNIT มาทำการวิเคราะห์ประมวลผลและแสดงผลในขณะใช้งานทันที (ON LINE) หรือ จะเก็บข้อมูลไว้ในหน่วยความจำสำรองเพื่อนำไปวิเคราะห์ในภายหลัง (OFF LINE)

- **ส่วนของหน่วยความจำสำรอง**

เป็นส่วนที่จัดเก็บข้อมูลลงในแผ่นดิส (FLOPPY DISK) เพื่อนำไปวิเคราะห์ข้อมูลในภายหลังเท่าที่ต้องการ การเก็บข้อมูลลงในแผ่นดิสอาจจะกำหนดเป็นช่วงเวลาให้เครื่องคอมพิวเตอร์ทำการเก็บข้อมูลไว้เป็นช่วง ๆ เช่น ช่วงละ 1 ชั่วโมงเก็บข้อมูล 1 ครั้ง หรือเก็บข้อมูลทุก ๆ 15 นาที เป็นต้น

- **ส่วนของการวิเคราะห์ข้อมูล (DATA ANALYSIS)**

ส่วนนี้จะทำหน้าที่วิเคราะห์ข้อมูลที่ได้มา ส่วนมากจะวิเคราะห์ออกมาในทางสถิติโดยจะมีโปรแกรมคำสั่ง (COMPUTER SOFTWARE) ควบคุมตามต้องการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **ส่วนของการแสดงผล (DISPLAY)**

ในส่วนนี้สามารถแสดงผลออกมาในลักษณะต่างๆ ได้หลายแบบ เช่น แสดงออกทางจอภาพ (MONITOR) กระดาษกราฟ (CHART RECORDER) เอกซ์-วายพล็อตเตอร์ (X-Y PLOTTER) และอื่น ๆ

องค์ประกอบภายในของระบบ

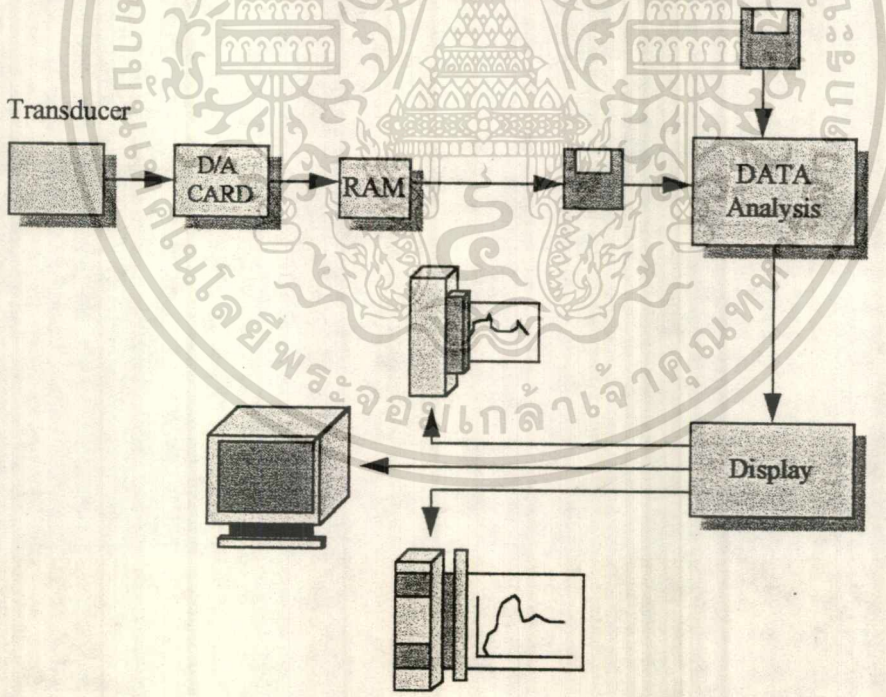
การทำงานของระบบรวบรวมข้อมูลมีองค์ประกอบหลักด้วยกัน 4 ส่วน คือ

1. สัญญาณเข้า (INPUT SIGNAL)

เป็นส่วนของการนำสัญญาณจากอุปกรณ์ตรวจวัดตามจุดต่างๆ ที่เรียกกันว่า ฟิลด์ซิกแนล (FIELD SIGNAL) เข้าสู่ระบบซึ่งสัญญาณเข้าจะมีลักษณะของสัญญาณ 2 แบบด้วยกัน คือ สัญญาณแบบอนาลอก (ANALOG INPUT SIGNAL) และสัญญาณแบบดิจิตอล (DIGITAL INPUT SIGNAL)

2. สัญญาณออก (OUTPUT SIGNAL)

เป็นส่วนของการนำสัญญาณที่ผ่านการวิเคราะห์ประมวลผลจาก เครื่องคอมพิวเตอร์ออกไปควบคุมเครื่องมือหรืออุปกรณ์ตามที่ต้องการ ซึ่งก็มีทั้งแบบอนาลอก และแบบดิจิตอล เช่นกัน



รูปที่ 2 แสดงขั้นตอนการทำงานของระบบรวบรวมข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. ระบบรวบรวมข้อมูลและการควบคุม

(DATA ACQUISIION AND CONTROL SYSTEM)

เป็นส่วนของกระบวนการเชื่อมโยงสัญญาณเข้า ที่จะเปลี่ยนแปลงรูปแบบทางไฟฟ้าเพื่อให้เข้ากับระบบสัญญาณของเครื่องคอมพิวเตอร์ และจะทำการเปลี่ยนแปลงรูปแบบทางไฟฟ้าจากเครื่องคอมพิวเตอร์เป็นสัญญาณออกไปควบคุมเครื่องมือ หรืออุปกรณ์ต่อไป

4. เครื่องคอมพิวเตอร์

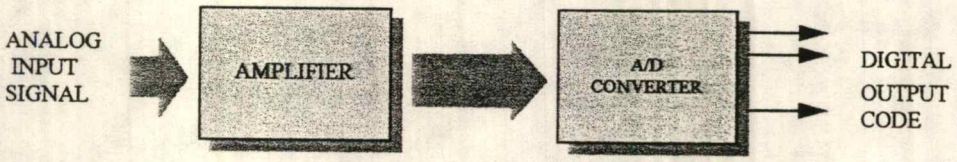
เป็นส่วนของการเก็บข้อมูล,วิเคราะห์,ประมวลผลและควบคุมซึ่งจะเป็นไปตามโปรแกรม (SOFTWARE) ที่กำหนดไว้ให้ทำงานตามวัตถุประสงค์การใช้งาน แผนภูมขององค์ประกอบของระบบดังแสดงในรูปที่ 3

รูปที่ 3 แสดงองค์ประกอบของระบบรวบรวมข้อมูล

ส่วนประกอบภายในของระบบรวบรวมข้อมูล มีส่วนประกอบต่างๆ ดังนี้ คือ

ก. ภาคขยายสัญญาณ (AMPLIFIER)

ทำหน้าที่ขยายสัญญาณจากอุปกรณ์ตรวจวัดจุดต่าง ๆ เข้าสู่ระบบ ซึ่งสัญญาณที่ได้จากอุปกรณ์ตรวจวัดตามจุดต่าง ๆ อาจจะมีความแตกต่างของระดับสัญญาณมาก จึงจำเป็นที่จะต้องขยายสัญญาณ และปรับให้อยู่ในช่วงระดับสัญญาณเดียวกัน ดังรูปที่ 4

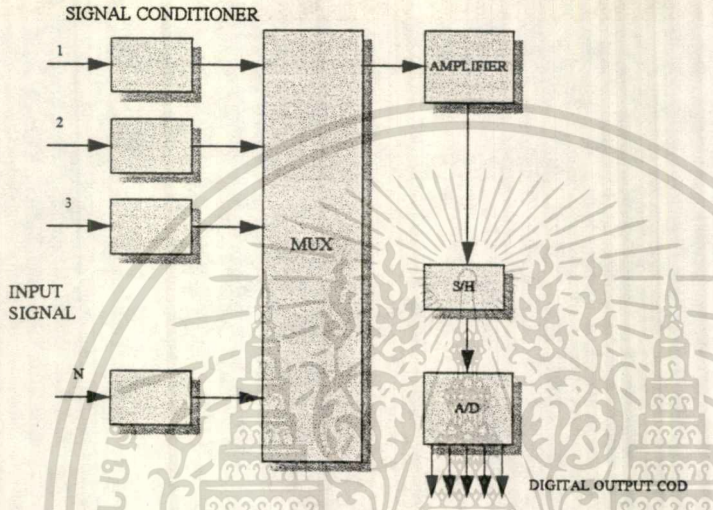


รูปที่ 4 แสดงภาคขยายสัญญาณในระบบรวบรวมข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข. ภาคมัลติเพล็กซ์ (MULTIPLEX)

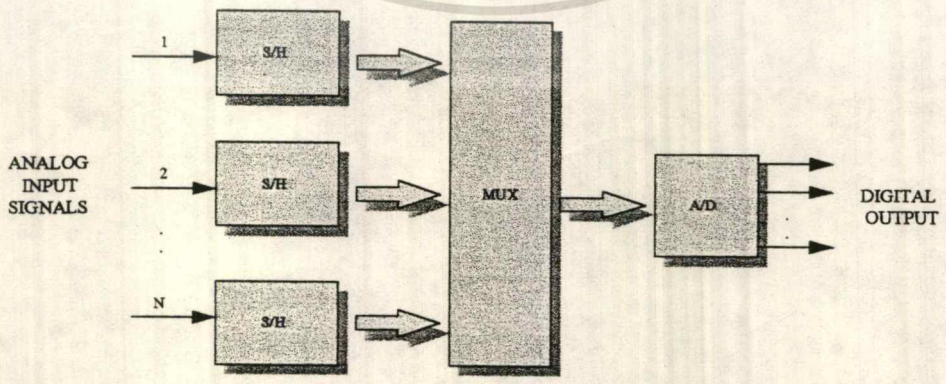
ทำหน้าที่สำคับการเลือกสัญญาณจากจุดต่าง ๆ เข้าสู่ระบบซึ่งสัญญาณที่ต้องการรวบรวมข้อมูลจะมาจากอุปกรณ์ตรวจวัดหลายตัว หลายจัดตามความต้องการของผู้ใช้ การนำสัญญาณเข้าสู่ระบบจึงใช้วิธีลำคับเลือกสัญญาณเข้าครั้งละ 1 สัญญาณหมุนเวียนกันเข้าทุกสัญญาณ โดยมีช่วงเวลาของการหมุนเวียนที่แน่นอน ซึ่งในระบบจะมีการนับเวลาอยู่ด้วย สัญญาณเข้า (INPUT SIGNALS) ของภาคมัลติเพล็กซ์จะมีหลายสัญญาณ แต่สัญญาณออก (OUTPUT SIGNALS) จะมีเพียง 1 สัญญาณเท่านั้น ดังรูปที่ 5



รูปที่ 5 แสดงภาคมัลติเพล็กซ์ในระบบบรวบรวมข้อมูล

ค. ภาคแซมเปิ้ลแอนด์โฮลด์ (SAMPLE AND HOLD)

ใช้อักษรย่อว่า S/H เป็นวงจรที่รับเอาแรงดันไฟฟ้าของสัญญาณเข้าแบบอนาลอกแล้วเก็บไว้ในคาปาซิเตอร์ (CAPACITOR) ในช่วงระยะเวลาสั้น ๆ เพื่อการเก็บตัวอย่างสัญญาณให้กับวงจรในภาคต่อไป ซึ่งวงจรแซมเปิ้ลแอนด์โฮลด์ นี้ อาจจะต่ออยู่ในลักษณะดังรูปที่ 5 แล้วยังสามารถต่อไว้ตามรูปที่ 6 ก็ได้ขึ้นอยู่กับลักษณะการใช้งานที่มีวัตถุประสงค์ต่างกัน



รูปที่ 6 แสดงภาคแซมเปิ้ลแอนด์โฮลด์ในระบบบรวบรวมข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ง. ภาคแปลงรูปสัญญาณอนาลอกเป็นดิจิตอล (ANALOG TO DIGITAL CONVERSION)

ใช้อักษรย่อว่า A/D เป็นภาคที่ทำหน้าที่เปลี่ยนแปลงรูปสัญญาณไฟฟ้าจากสัญญาณแบบอนาลอกเป็นสัญญาณแบบดิจิตอล เนื่องจากในระบบเครื่องคอมพิวเตอร์การรับส่งสัญญาณ หรือข้อมูลต่าง ๆ จะอยู่ในรูปของสัญญาณแบบดิจิตอล ดังนั้นสัญญาณที่ได้จากอุปกรณ์ตรวจวัด ซึ่งเป็นสัญญาณแบบอนาลอกจึงใช้เป็นสัญญาณให้กับเครื่องคอมพิวเตอร์โดยตรงไม่ได้ จึงจำเป็นจะต้องแปลงรูปแบบสัญญาณให้อยู่ในรูปแบบเดียวกันกับรูปแบบสัญญาณของเครื่องคอมพิวเตอร์ ซึ่งจะจับเอาความเปลี่ยนแปลงของระดับสัญญาณ (AMPLITUDE) ของสัญญาณอนาลอกมาเป็นรหัสเลขฐานสองทางดิจิตอล (BINARY CODE) ซึ่งทางสัญญาณเข้ามีสัญญาณแบบอนาลอก 1 สัญญาณ ส่วนสัญญาณออกจะเป็นแบบดิจิตอลเรียงกัน 8 หลัก ซึ่งเรียกหลักของตัวเลขนี้ว่า บิท (BIT) ลักษณะการต่อใช้งานในระบบรวบรวมข้อมูลสามารถทำได้หลายลักษณะ ดังรูปที่ 4, 5 และ รูปที่ 6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3
การใช้งาน PCL-818

บทนำ

PCL-818 เป็นการ์ดที่สามารถทำงานได้หลายหน้าที่ โดยสามารถโดยสามารถโปรแกรมเข้าได้กับเครื่อง IBM PC/XT/AT หรือเครื่องทดแทนอื่นๆ ประสิทธิภาพสูงสุดของการ์ดนี้ และประกอบกับการใช้ซอฟแวร์ที่สนับสนุนเต็มที่สามารถนำไปประยุกต์ได้อย่างกว้างขวางทั้งในอุตสาหกรรมและในห้องทดสอบเช่นการเรียกใช้ข้อมูล การควบคุมการผลิต การทดสอบอัตโนมัติและในอุตสาหกรรมที่ใช้เครื่องจักรแบบอัตโนมัติ

คุณลักษณะ

คุณลักษณะของการ์ด INTERFACE CONTROL นี้ประกอบด้วย

* มี SELECTER SWITCH 16 TERMINAL หรือ 8 ช่องสัญญานแบบอนาลอก
* ใช้ตัวแปลงสัญญานแบบมาตรฐาน 12 BIT ในการแปลงสัญญานอินพุทแบบอนาลอกอัตราการ SAMPLING สูงสุด 100 KHZ ใน DMA MODE
* ตัวควบคุมการนำเข้ามี 2 MODE คือ
  LOCAL MODE สวิทช์แบบเลือกได้ สามารถประยุกต์เข้ากับ PCL-718 ได้
  REMOTE MODE สามารถโปรแกรมได้โดยซอฟแวร์ควบคุมโดยแต่ละช่องสัญญานจะมีย่านของตัวมันเอง ในแต่ละย่าน ที่ถูกกำหนดขึ้นจะถูกเก็บไว้ในแผงของตัว RAM
* SWITCH LOCAL MODE (แบบควบคุมด้วยมือ) หรือการควบคุมที่ใช้ซอฟแวร์ (การควบคุมแบบอัตโนมัติ) ตัวเลือกทางอินพุทจะมีย่านดังนี้
  แบบ BIPOLAR : +/-0.5V , +/-1V , +/-2.5V , +/- 50V , +/-10V
  UNIPOLAR : 0V TO +1V , 0V TO +2V , 0V TO +5V , 0V TO +10V
* การควบคุมสัญญาน ANALOG INPUT CODE จะเก็บไว้ที่บนบอร์ดของ RAM จากคุณสมบัติพิเศษของตัว PCL- 818 สามารถ เปลี่ยนช่องได้อย่างอัตโนมัติไม่มีการโปรแกรมจาก CPU ของ PC
* A/D TRIGGER มี 3 MODE : SOFTWARE TRIGGER , PROGRAMMABLE PACER TRIGGER , EXTERNAL PULSE TRIGGER
* A/D การเปลี่ยนของข้อมูล สามารถเปลี่ยนแปลงได้โดยโปรแกรมควบคุม, ตัวขัดจังหวะ หรือ การแปลง DMA
* ตัวตั้งเวลาและตัวนับเวลาที่โปรแกรมได้ INTEL -8254 ให้ PULSE ที่อัตรา 2.5 MH_z ถึง 0.00023 H_z (71 นาที/ลูก) และ TIME BASE คือ SWITCH ที่เลือกระหว่าง 10 MH_z หรือ 1 MH_z ช่องนับสัญญานแบบ 16 บิท 1 ช่องจะถูกสำรองไว้สำหรับโปรแกรมประยุกต์สำหรับผู้ใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

* 2 ระบบใหญ่ๆขนาด 12 บิท พร้อมกับ D/A ที่ช่องทางออก ย่านของสัญญานที่ทางออกจาก 0V ถึง +5 V (+10) จะถูกสร้างขึ้นโดยแผงขนาด -5V (-10V) อ้างอิง แหล่งจ่าย AC/DC จากภายนอกสามารถนำมาใช้ทำให้เกิดย่านสัญญานออก D/A อื่นๆ
* TTL/DTL มีช่องสัญญานทางเข้า 16 ช่อง และช่องสัญญานทางออก 16 ช่อง

ซอฟท์แวร์สนับสนุน

PCL- 818 ยังช่วยให้สามารถใช้งานฟังก์ชั่นของ SOFTWARE DRIVER ได้ง่าย และมีประสิทธิภาพ สามารถเข้าถึงได้โดยการอ้างถึงตาราง USER DEFINED PARAMETER ด้วยฟังก์ชั่นของ DRIVER เหล่านี้โปรแกรมจะง่ายขึ้นโดยเฉพาะเมื่อต้องการใช้ฟังก์ชั่นที่ซับซ้อนใน PLC-818 เช่นการ INTERRUPT หรือการถ่ายข้อมูล DMA เพื่อเป็นการเป็นการสนับสนุนแก่ผู้ใช้ PCL-818 มากขึ้น โปรแกรมมากมายได้ถูกผลิตขึ้นเพื่อใช้งานกับ PCL-818

คุณลักษณะเฉพาะของการ์ด

สัญญานนำเข้าแบบอนาล็อก (ANALOG INPUT: A/D CONVERTER)

CHANNELS	:	16 SINGLE-ENDED OR 8 DIFFERENTIAL, SWITCH SELECTABLE.
RESOLUTION	:	12 BITS
INPUT RANGE	:	UNIPOLAR : +10V, +5V, +2V, +1V BIPOLAR : +/-10V, +/-5V, +/-2.5V, +/-1V, +/-0.5V สัญญานนำเข้าทั้งหมดเป็นแบบสวิทช์เลือกได้(LOCAL MODE) หรือ SOFTWARE PROGRAMMABLE (REMOTE MODE)
OVERVOLTAGE	:	CONTINUUS +/-30V MAX.
CONVERSION TYPE	:	SUCCESSIVE APPROXIMATION
CONVERSION SPEED	:	100 KH_Z MAX.
ACCURACY	:	+/-(0.01% OF READING) +/-1 BIT
LINEARITY	:	+/-1 BIT
TRIGGER MODE	:	SOFTWARE TRIGGER, ON-BOARD PROGRAMMABLE PACER TRIGGER OR EXTERNAL TRIGGER
EXTERNAL TRIGGER	:	TTL COMPATIBLE, THE LOAD IS 0.4mA MAX. AT 0.5V AND -0.05mA MAX. AT 2.7V.
DATA TRANSFER	:	PROGRAM, INTERRUPT OR DMA.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสาร 037147 ห้ามมิให้นำไปใช้

สัญญานออกแบบอนาล็อก (ANALOG OUTPUT :D/A CONVERTER)

CHANNELS	:	2 CHANNELS
RESOLUTION	:	12 BITS
OUTPUT RANGE	:	0 TO +5V (+10V) WITH ON-BOARD -5V (-10V) REFERENCE. MAX. +10V OR -10V WITH EXTERNAL DC OR AC REFERENCE.
REFERENCE	:	INTERNAL -5V, -10V OR EXTERNAL DC OR AC, +/-10V MAX.
CONVERSION TYPE	:	12 BIT MONOLITHIC MULTIPLYING.
LINEARITY	:	+/-0.5 BIT
OUTPUT DRIVE	:	+/-5mA MAX.
SETTLING TIME	:	5 MICROSECONDS.

สัญญานนำเข้าแบบดิจิตอล (DIGITAL INPUT)

CHANNEL	:	16 BITS.
LEVEL	:	TTL COMPATIBLE
INPUT VOLTAGE	:	LOW ----- 0.8V MAX. HIGH ---- 2.0V MIN.
INPUT LOAD	:	LOW ---- 0.4mA MAX. AT 0.5V. HIGH --- 0.05mA MAX. AT 2.7V.

สัญญานออกแบบดิจิตอล (DIGITAL OUTPUT)

CHANNEL	:	16 BITS
LEVEL	:	TTL COMPATIBLE.
OUTPUT VOLTAGE	:	LOW ---- SINK 8mA AT 0.5V MAX. HIGH ---- SOURCE -0.4mA at 2.4v. MIN.

PROGRAMMABLE TIMER/COUNTER

DEVICE	:	INTEL 8254 OR EQUIVALENT
COUNTERS	:	3 CHANNELS, 16 BIT. 2 CHANNELS ARE PERMANENTLY CONFIGURED AS PROGRAMMABLE PACER, 1 CHANNEL IS FREE FOR USER'S APPLICATION.
INPUT,GATE	:	TTL/DTL/CMOS COMPATIBLE.
TIME BASE	:	PACER (CHANNEL 1 AND 2): 10 MH_Z OR 1 MH_Z

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

		SWITCH SELECTABLE.
		CHANNEL 0: INTERNAL 100 MH_Z OR EXTERNAL CLOCK (10 MH_Z MAX.). THE SELECTION IS CONTROLLED BY THE TIMER/COUNTER ENABLE REGOSTER (BASE + 10).
PACER OUTPUT	:	0.00023 H_Z (71 MINUTES/PULSE) TO 2.5 MH_Z

INTERRUPT CHANNEL

LEVEL	:	IRQ 2 TO 7, SOFTWARE SELECTABLE.
ENABLE	:	VIA INTE BIT OF CONTROL REGISTER (BASE + 9).

DMA CHANNEL

LEVEL	:	1 OR 3, SWITCH SELECTABLE.
ENABLE	:	VIA DMAE BIT OF CONTROL REGISTER (BASE + 9).

GENERAL

POWER CONSUMPTION	:	+5V: 700mA TYPICAL, 1A MAX. +12V: 140mA TYPICAL, 200mA MAX. -12V: 14mA TYPICAL, 20mA MAX.
I/O CONNECTOR	:	20 PIN POST HEADERS FOR I/O CONNECTION. ADAPTOR AVAILABLE TO CONVERT TO 37 PIN D-TYPE CONNECTOR.
I/O BASE ADDRESS	:	REQUIRE 16 CONSECUTIVE ADDRESS LOCATIONS. BASE ADDRESS DEFINABLE BY THE DIP SWITCH SW3 FOR ADDRESS LINE A9-A4. (FACTORY SETTING IS HEX 300).
OPERATING TEMP	:	0 TO +50 DEG. C.
STORAGE TEMP	:	-20 TO +65 DEG. C.
WEIGHT	:	9.8 oz (277.6 gm).

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขาต่างๆใน PCL - 818

ประกอบไปด้วยตัวต่อขนาด 20 ขา 2 ชุด ที่แผ่นด้านหลังและ20ขาอีก 3 ตัว บนแผงต่อโดยตัวต่อเหล่านี้ สามารถต่อไปยังสายแบบชนิดเดียวกันหรือต่อไปยัง D-TYPE แบบ 37ขาโดยผ่านชุด WIRING PCLK - 1050 แผนภาพเหล่านี้แสดงถึงหน้าที่ของตัวต่อแต่ละตัว

Legend :

A/D S	-	ANALOG INPUT (SINGLE-ENDED
A/D H	-	ANALOG INPUT (DIFFERENTIAL)
A/D L	-	ANALOG INPUT LOW (DIFFERENTIAL)
A.GND	-	ANALOG GROUND
D/A	-	ANALOG OUTPUT
D/O	-	DIGITAL OUTPUT
D/I	-	DIGITAL INPUT
D.GND	-	DIGITAL AND POWER SUPPLY GROUND
CLK	-	CLOCK INPUT FOR THE 8254
GATE	-	GATE INPUT FOR THE 8254
OUT	-	SIGNAL OUTPUT OF THE 8254
VREF	-	VOLTAGE REFERENCE
REFIN	-	EXTERNAL VOLTAGE REFERENCE INPUT

CONNECTOR 1 (CN1) - ANALOG INPUT (SINGLE-ENDED CHANNELS)

A/D S0	1	2	A/D S8
A/D S1	3	4	A/D S9
A/D S2	5	6	A/D S10
A/D S3	7	8	A/D S11
A/D S4	9	10	A/D S12
A/D S5	11	12	A/D S13
A/D S6	13	14	A/D S14
A/D S7	15	16	A/D S15
A.GND	17	18	A.GND
A/GMD	19	20	A.GND

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

CONNECTOR 1 (CN1) - ANALOG INPUT (DIFFERENTIAL CHANNELS)

A/D H0	1	2	A/D L0
A/D H1	3	4	A/D L1
A/D H2	5	6	A/D L2
A/D H3	7	8	A/D L3
A/D H4	9	10	A/D L4
A/D H5	11	12	A/D L5
A/D H6	13	14	A/D L6
A/D H7	15	16	A/D L7
A.GND	17	18	A.GND
A/GMD	19	20	A.GND

CONNECTOR 2 (CN2) - ANALOG OUTPUT

D/A 0 REFIN	1	2	D/A 1 REFIN
D/A 0 OUT	3	4	D/A 1 OUT
A.GND	5	6	A.GND
-VREF	7	8	
	9	10	
	11	12	
	13	14	
	15	16	
	17	18	
	19	20	

CONNECTOR 3 (CN3) - DIGITAL OUTPUT

D/O 0	1	2	D/O 1
D/O 2	3	4	D/O 3
D/O 4	5	6	D/O 5
D/O 6	7	8	D/O 7
D/O 8	9	10	D/O 9
D/O 10	11	12	D/O 11
D/O 12	13	14	D/O 13
D/O 14	15	16	D/O 15
D.GND	17	18	D.GND
+ 5V	19	20	+ 12V

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

CONNECTOR 4 (CN4) - DIGITAL INPUT

D/I 0	1	2	D/I 1
D/I 2	3	4	D/I 3
D/I 4	5	6	D/I 5
D/I 6	7	8	D/I 7
D/I 8	9	10	D/I 9
D/I 10	11	12	D/I 11
D/I 12	13	14	D/I 13
D/I 14	15	16	D/I 15
D.GND	17	18	D.GND
+ 5V	19	20	+ 12V

CONNECTOR 5 (CN5) - COUNTER

TRIGO	1	2	
	3	4	
	5	6	
	7	8	CTRO CLK
	9	10	CTRO OUT
	11	12	CTRO GATE
	13	14	CTR2 OUT
	15	16	
D.GND	17	18	D.GND
+ 5 V	19	20	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การติดตั้ง Hard Ware

การติดตั้งการ์ดลงสู่ COMPUTER ทำดังนี้

1. ปิดเครื่องคอมพิวเตอร์ และดับอุปกรณ์ที่ต่ออยู่ในระบบทั้งหมด (เช่น เครื่องพิมพ์และจอภาพ)
2. เอาสาย POWER ออก และสายอื่นๆที่ด้านหลังของเครื่อง และหมุนเครื่องคอมพิวเตอร์ให้ด้านหลังของเครื่องมาอยู่ที่ด้านหน้า
3. เปิดฝาออก (ดูที่คำแนะนำการใช้คอมพิวเตอร์ของท่านถ้าจำเป็น)
4. เลือกช่องต่อที่ด้านหลังของเครื่องที่ยังไม่ได้ใช้
5. ขันสกรูที่ยึดฝา SLOT ออก
6. หยิบการ์ด PLC - 818 ที่มุมอย่างระมัดระวัง ปรับให้ช่องของการ์ดและ Slot ให้ตรงกันและเสียบเข้าไปให้แน่น
7. นำSLOT ใส่เข้าไปแล้วขันสกรูยึด SLOT กับแผง
8. ใส่อุปกรณ์อื่นๆที่จำเป็น แล้วต่อเข้ากับ INTERFACE CARD ขึ้นอยู่กับโปรแกรมที่ใช้
9. ปิดฝาครอบต่อสายเคเบิ้ล ที่ถอดออกในขั้นตอนที่สอง แล้วเปิดเครื่องคอมพิวเตอร์

ตอนนี้การติดตั้ง HARD WARE เสร็จแล้ว หลังจากนั้นก็ติดตั้ง SOFT WARE

แผ่น SOFTWARE

แผ่น DISK ที่บรรจุโปรแกรมการใช้งาน SOFT WARE ประกอบด้วย PCL - 818

โปรแกรมการใช้งานประกอบไปด้วย

1. COMPREHENSIVE I/O DRIVER สำหรับ D/A , A/D DIGITAL I/O และโปรแกรมตัวนับ ที่ DRIVER นี้ยอมให้ใช้ ฟังก์ชั่นมาตรฐาน การทำงานที่เขียนโดยภาษาโปรแกรมมิ่งธรรมดา เพื่อสั่งให้ PCL - 818 ทำงานโดยไม่ต้องเข้าถึงรายละเอียดในการควบคุม ภาษาที่สนับสนุนด้วย SOFT WARE DRIVER ประกอบด้วย BASICA , GW BASIC , QUICKBASIC , MICROSOFT C/C++ และ PASCAL , TURBO C/C++ , BORLAND C/C++ และ TURBO PASCAL.
2. โปรแกรมสาธิต
3. โปรแกรมการปรับแต่ง
4. โปรแกรมทดสอบ

ข้อแนะนำ

ควรทำสำเนาแผ่นต้นฉบับไว้ และเก็บแผ่นต้นฉบับไว้ที่ๆปลอดภัย สามารถใช้คำสั่ง DOS COPY หรือ DISKCOPY เพื่อทำสำเนาไฟล์จากแผ่น DISK ไปยังอีกแผ่นหนึ่ง หรือ ไปยัง HARD DISK

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TURBO C

THE FOLLOWING EXAMPLES SHOW YOU HOW TO COMPILE AND LINK THE INTERFACE FOR DIFFERENT MODES USING TURBO C.

THIS ASSUMES YOU ARE USING A LAB CARD PCL-818 FOR OTHER PC-LAB CARD, YOU SHOULD REPLACE "818" WITH THE APPROPRIATE NUMBER FOR THE CARD YOU ARE USING.

DOS COMMAND LINE

SHELL MODE : TCC-MS FILE.C 818 CS.LIB

COMPACT MODE : TCC-MC FILE.C 818 CC.LIB

MEDIUM MODE : TCC-MM FILE.C 818 CM.LIB

LARGE MODE : TCC-ML FILE.C 818 CL.LIB

การต่อสัญญาน

การต่อสัญญานที่ถูกต้องเป็นขั้นตอนที่สำคัญที่จะทำให้เรามั่นใจในการประยุกต์เข้ากับระบบส่งหรือการรับข้อมูลอย่างถูกต้อง เนื่องจากการเรียกใช้ข้อมูลส่วนใหญ่เกี่ยวกับโวลท์เตจที่เข้ามา/โวลท์เตจที่ออกไปและการไหลของกระแสการต่อสัญญานที่ดีควรหลีกเลี่ยงความเสียหายที่ไม่จำเป็นที่อาจเกิดขึ้นได้กับเครื่อง PC และ อุปกรณ์ HARD WARE ซึ่งในหัวข้อนี้จะกล่าวถึงข้อมูลที่มีประโยชน์ในการต่อสัญญานในรูปแบบต่างๆของการเรียกใช้ข้อมูล

การต่อสัญญานเข้าแบบอนาลอก

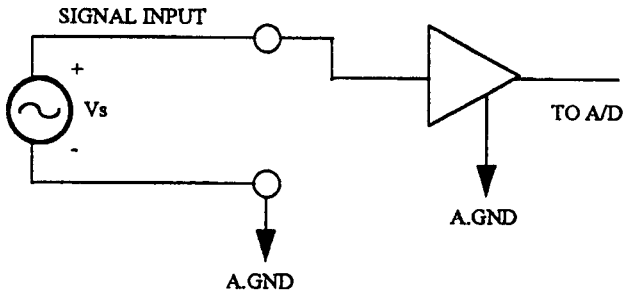
PCL - 818 ช่วยสนับสนุนทั้งการต่อแบบ 16 จุด (แบบ SINGLE ENDED)หรือ 8 จุด แบบ DIFFERENTIAL ช่องสัญญานนำเข้าจะถูกเลือกผ่านSWITCH บน การ์ด ข้อที่แตกต่างสำคัญระหว่าง SINGLE-ENDED และ DIFFERENTIAL INPUT คือ จำนวนสายสัญญาณต่อช่อง

การต่อแบบ SINGLE-ENDED CHANNEL

ข้อกำหนดของ SINGLE-ENDED จะมีเพียงสายสัญญาณหนึ่งเส้นต่อหนึ่งช่องสัญญาณ VOLTAGE ที่วัดได้ คือ VOLTAGE ของสายเส้นนี้เมื่ออ้างอิงกับกราวน์ แหล่งสัญญาณที่ไม่มีกราวน์ เรียกว่า FLOATING SOURCE เป็นการต่อ SINGLE-ENDED แบบง่ายๆ เข้ากับ FLOATING SIGNAL SOURCE

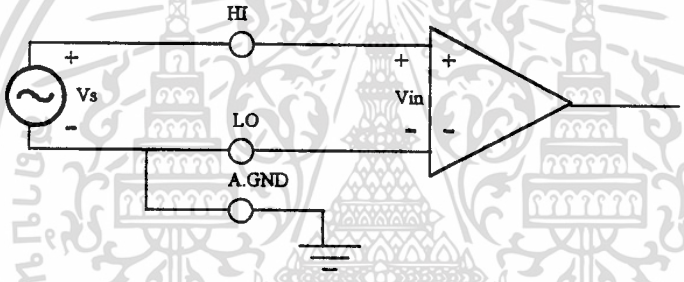
WIRING DIAGRAM มาตรฐาน เป็นดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ข้อกำหนดของ DIFFERENTIAL INPUT

คือ จะมีสายสัญญาณสองเส้นต่อหนึ่งช่องสัญญาณ DIFFERENTIAL INPUT จะตอบสนองเฉพาะความแตกต่างระหว่าง VOLTAGE สูงและVOLTAGE ต่ำ ถ้าแหล่งจ่ายสัญญาณไม่ได้ต่ออยู่กับกราวน์ก็จะเรียกว่า FLOATING SOURCE การต่อจะต้องต่อขาVOLTAGE ต่ำ เข้ากับกราวน์ร่วมสำหรับ FLOATING SIGNAL SOURCE การวัด FLOATING SOURCE ช่องสัญญาณ INPUT ควรต่อดังนี้

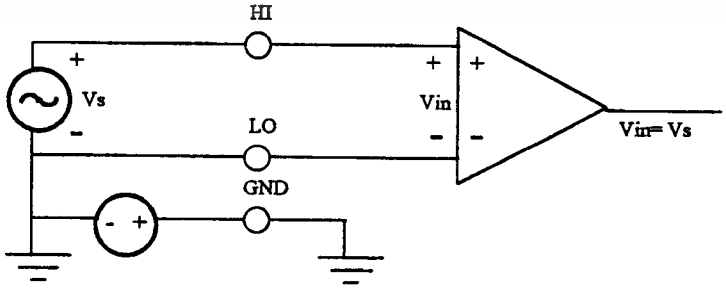


ถ้าแหล่งจ่ายสัญญาณข้างหนึ่งไม่ต่อกับกราวน์ แหล่งสัญญาณกราวน์และ PCL-818 จะไม่เท่ากับ VOLTAGE ที่ต่อผ่านกราวน์กลับไปยังอุปกรณ์และวงจรภายใน

ข้อแตกต่างระหว่างกราวน์VOLTAGEทำให้เกิด COMMON MODE VOLTAGE

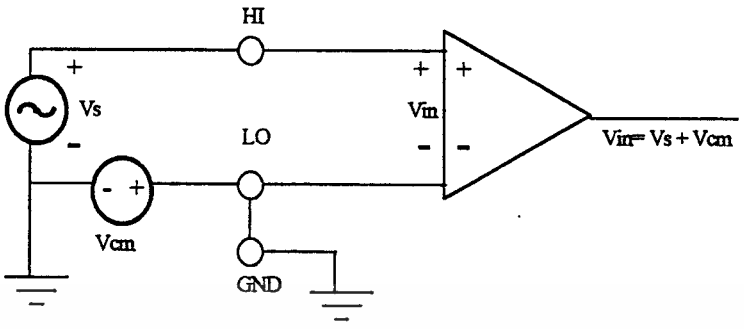
เพื่อหลีกเลี่ยงการเกิด GROUND LOOP NOISE สัญญาณกราวน์ควรที่จะต่อกับด้าน LOW INPUT และทางด้าน LOW INPUT ไม่ควรต่อกับ PCL-818 โดยตรง ในบางกรณี จำเป็นต้องต่อ PCL-818 GROUND เข้ากับ SIGNAL SOURCE GROUND รูปภาพต่อไปนี้เป็นการต่อแบบถูกวิธี และไม่ถูกวิธี

CORRECT CONNECTION :



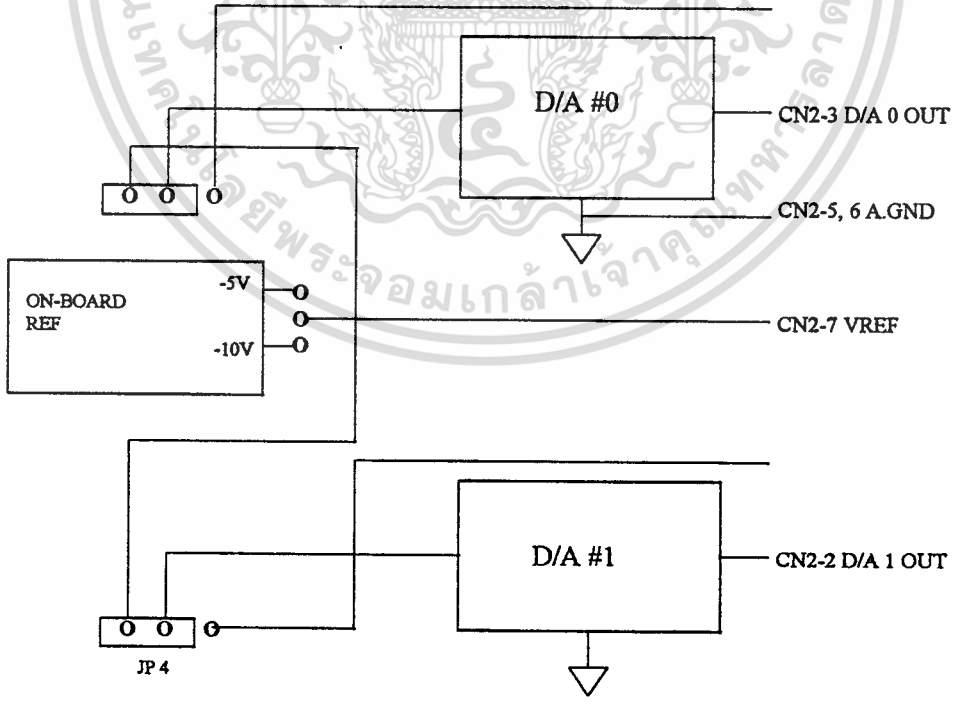
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

INCORRECT CONNECTION :



การต่อ OUTPUT แบบ ANALOG

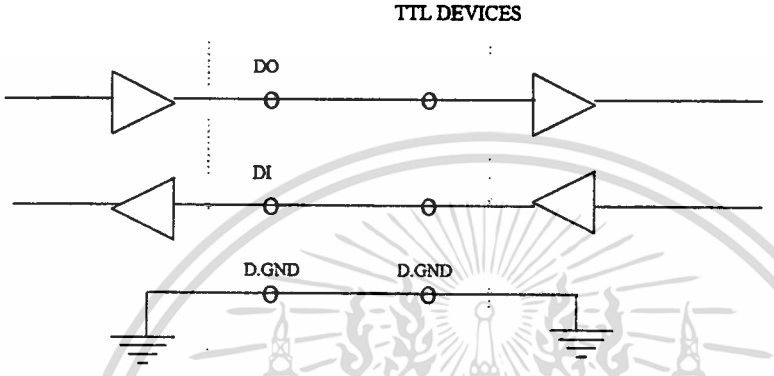
การ์ด PCL-818 มีช่องสัญญาณออกแบบ D/A 2 ช่อง ผู้ใช้งานอาจใช้ตาม PRECISION ที่ได้จัดเตรียมไว้ให้ภายใน คือ -5V(-10V) อ้างอิงเพื่อสร้างสัญญาณออก D/A ที่ 0 ถึง +5V(+10V) ผู้ใช้สามารถสร้างย่าน D/A อื่นๆ ผ่านแหล่งอ้างอิงภายใน ย่าน INPUT อ้างอิงสูงสุด คือ +/-10V และทางออกสูงสุดคือ +/-10V ขั้วต่อ CN2 ของ PCL-818 ใช้สำหรับสัญญาณ D/A ส่วนที่สำคัญในการต่อสัญญาณ D/A เช่น INPUT REFERENCE, D/A OUTPUT และ ANALOG GROUND ดังที่แสดงข้างล่าง และการหน่วงกระแสของ D/A OUTPUT ไม่ควรเกิน 5 mA



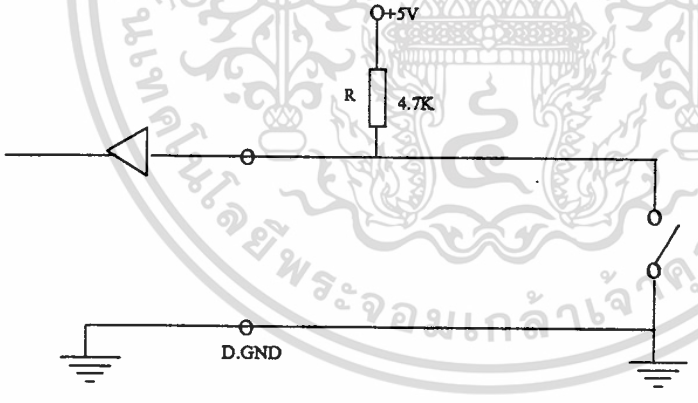
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การต่อสัญญาณแบบ DIGITAL

PCL-818 มีช่องสัญญาณ INPUT 16 ช่อง และช่องสัญญาณ OUTPUT 16 ช่อง แบบDIGITAL ระดับ DIGITAL I/O ใช้ต่อกับอุปกรณ์ TTL ได้ การส่งสัญญาณ หรือ การบีบสัญญาณแบบDIGITAL กับอุปกรณ์ TTL มีลักษณะการต่อ ดังนี้



เพื่อรับสัญญาณ OPEN/SHORT จากสัญญาณ SWITCH หรือ RELAY จะต้องเพิ่ม PULL-UP RESISTOR เพื่อให้เกิดความแน่ใจว่าจะเกิดสัญญาณแบบ HIGH LEVEL เมื่อขณะเปิด



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

ระบบควบคุม

ก่อนที่จะศึกษาหรือทำความเข้าใจเกี่ยวกับระบบควบคุมนั้น เราจะมาทำความรู้จักกับคำจำกัดความต่างๆที่ใช้ในเรื่องของการควบคุมเสียก่อน

"การควบคุม" คือการกระทำใดๆ เพื่อให้เกิดสภาวะทางฟิสิกส์ตามที่เราต้องการ ซึ่ง สภาวะทางฟิสิกส์ที่แสดงออกมาเป็นเพียงสถานะ หรืออาจจะสดงออกมาเป็นตัวเลขเป็นปริมาณค่าต่างๆ

1. องค์ประกอบของการควบคุม

โดยทั่วๆไปแล้วองค์ประกอบหลักในการควบคุมจะประกอบไปด้วย

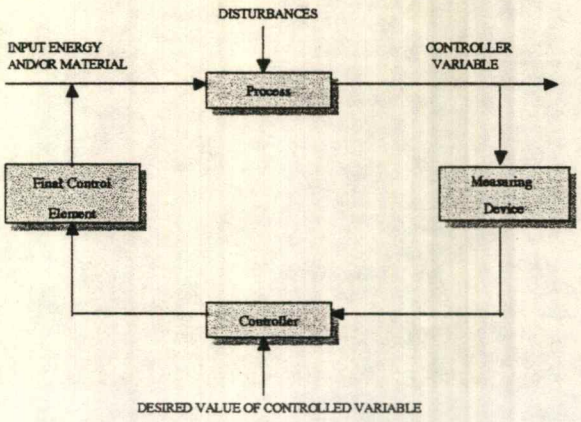
1. ขบวนการ (PROCESS)

ขบวนการ หมายถึงขบวนการทางฟิสิกส์ที่เราต้องการควบคุมให้มีสภาวะตามต้องการ ในขณะที่สภาวะการทำงานและสภาพแวดล้อมอื่นๆอาจเปลี่ยนแปลงตลอดเวลา ซึ่งค่าทางฟิสิกส์ในขบวนการนั้นเราเรียกว่า "ตัวแปรขบวนการ"(PROCESS VARIABLE) ซึ่งอาจจะได้แก่ อุณหภูมิ เป็นต้น นอกจากนั้นยังมีตัวแปรปรับขบวนการ (MANIPULATED VARIABLE) ซึ่งจะเป็นตัวที่เราปรับเพื่อทำการควบคุมค่าปริมาณทางฟิสิกส์ที่เราต้องการควบคุมอีกทีและนอกจากว่าตัวแปรขบวนการนั้น จะขึ้นกับค่าของตัวแปรปรับขบวนการ แล้วยังขึ้นกับค่าของสิ่งรบกวนที่เกิดขึ้นในขบวนการนั้นๆอีกด้วย ซึ่งเราเรียกสิ่งรบกวนขบวนการนี้ว่า "PROCESS DISTURBANCE" ซึ่งหมายถึง การเปลี่ยนแปลงของสภาวะการทำงาน และสภาพแวดล้อมของขบวนการ โดยเราแบ่งสิ่งรบกวนขบวนการได้เป็น 2 ประเภท คือ

(1) SUPPLY DISTURBANCE หรือ SUPPLY LOAD CHANGES เป็นการเปลี่ยนแปลงพลังงาน(วัตถุ)ขาเข้า

(2) DEMAND DISTURBANCE หรือ DEMAND LOAD CHANGES เป็นการเปลี่ยนแปลงพลังงาน(วัตถุ)ขาออก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 1 Control Loop System

2. อุปกรณ์การวัด (MEASURING DEVICES)

เป็นอุปกรณ์ที่ให้สัญญาณขาออกซึ่งมีขนาดสัมพันธ์กับขนาดของตัวแปรทางฟิสิกส์ของสิ่งที่เราต้องการวัด โดยทั่วไปแล้วเราต้องการให้ความสัมพันธ์ดังกล่าวเป็นสมการเชิงเส้น

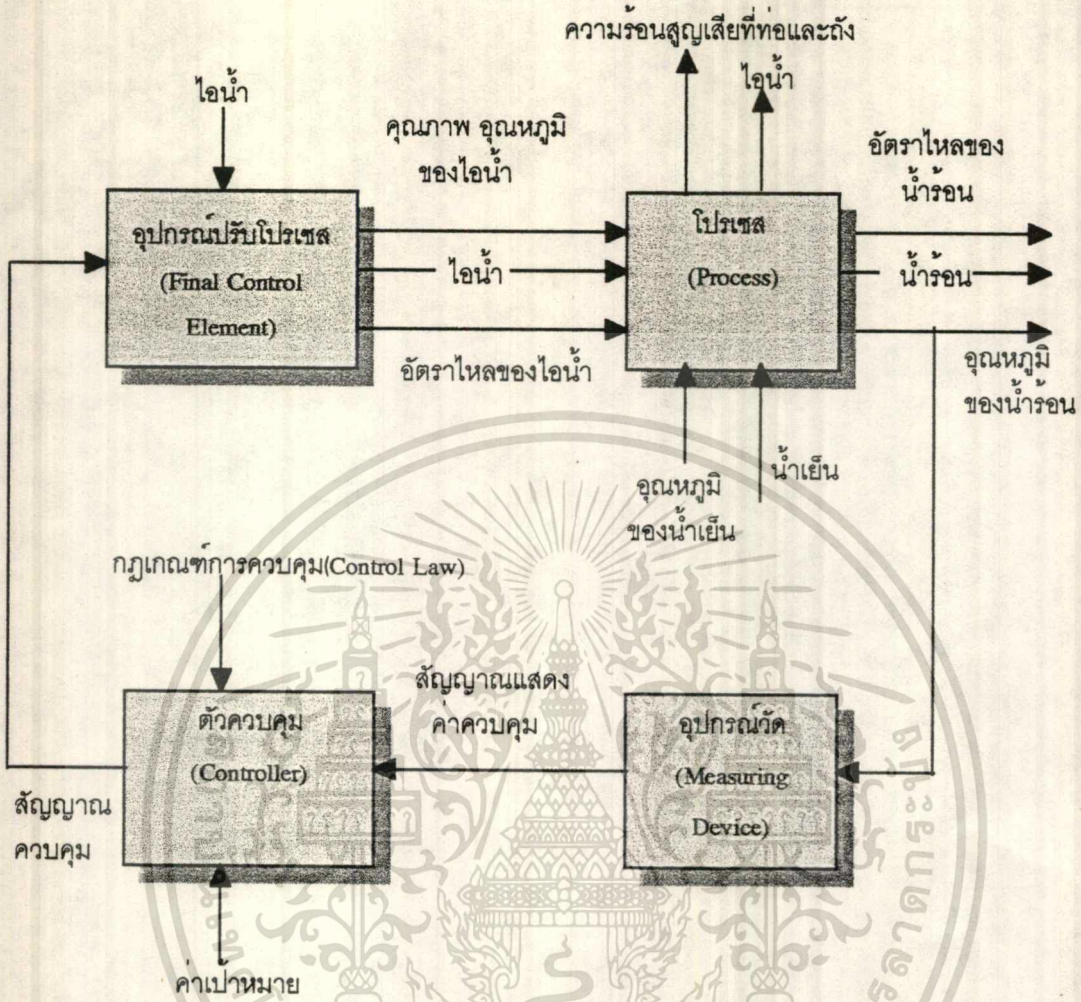
3. ตัวควบคุม (CONTROLLER)

ตัวควบคุม หมายถึง สิ่งที่ทำหน้าที่ออกคำสั่ง หรือกำเนิดสัญญาณควบคุม (CONTROL-SIGNAL) ตามกฎเกณฑ์ของการควบคุม (CONTROL LAW) ที่ถูกกำหนดไว้ล่วงหน้า คำสั่งหรือสัญญาณควบคุมนี้ อาจจะเป็นฟังก์ชันกับเวลาหรือฟังก์ชันกับสัญญาณขาเข้า ที่ได้รับจากอุปกรณ์วัดก็ได้

4. อุปกรณ์ปรับขบวนการ (FINAL CONTROL ELEMENT)

เป็นอุปกรณ์ที่ทำหน้าที่ปรับสภาวะของขบวนการ ด้วยการเปลี่ยนแปลงค่าของตัวแปรปรับขบวนการตามคำสั่งหรือสัญญาณควบคุมที่ได้รับในการควบคุมส่วนใหญ่ คำสั่งหรือสัญญาณควบคุมจะมีขนาดหรือพลังงานน้อยๆ ส่วนตัวแปรปรับขบวนการจะมีขนาดหรือพลังงานมาก พอถึงตอนนี้เราคงกล่าวได้ว่าไม่ว่าการควบคุมจะเป็นแบบ MANUAL หรือ AUTOMATIC ก็ตาม จะประกอบด้วยองค์ประกอบหลักที่สำคัญตามที่กล่าวมาแล้วเพียงแต่ว่าแบบ MANUAL นั้นจะใช้คนในการทำงาน ส่วนแบบ AUTOMATIC นั้น จะใช้เครื่องจักรกลไกในการควบคุม โดยเฉพาะในปัจจุบันนั้นจะใช้เทคโนโลยีทางด้าน อิเลคทรอนิคส์ และคอมพิวเตอร์ มาใช้ในการควบคุมทางอุตสาหกรรมมากขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2 บล็อกไดอะแกรมแสดงองค์ประกอบที่สำคัญในการควบคุม Heat Exchanger พร้อมทั้งตัวแปรทางฟิสิกส์และสัญญาณต่างๆ

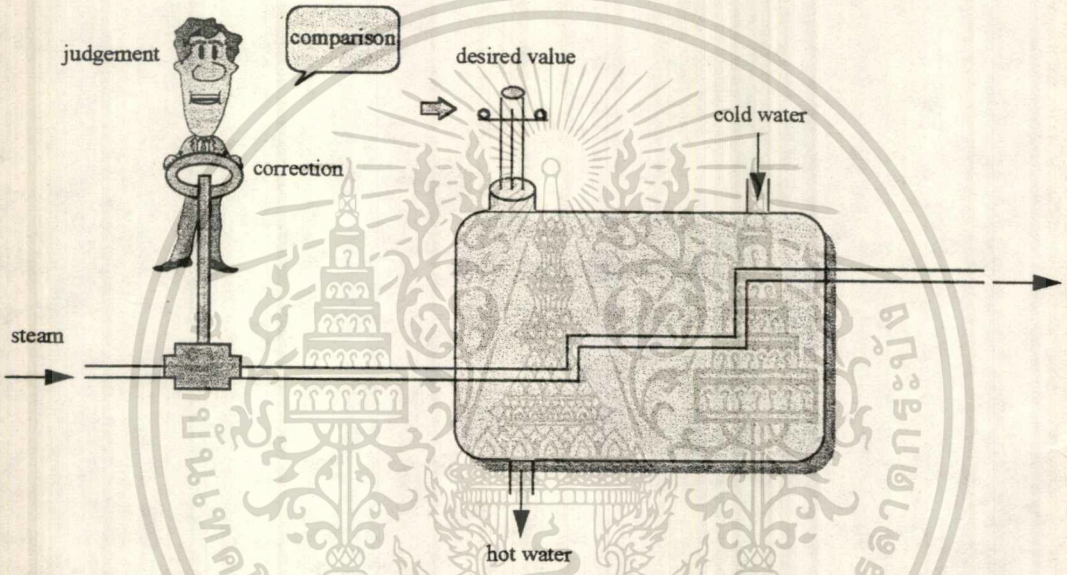
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ชนิดของการควบคุม

การควบคุมขบวนการในปัจจุบันได้รับการพัฒนาไปมาก จากการวัดและควบคุมด้วยมือจนถึงการควบคุมอัตโนมัติ การควบคุมที่สลับซับซ้อน ใช้เครื่องคอมพิวเตอร์เข้ามาช่วย ระบบการควบคุม (CONTROL SYSTEM) สามารถทำได้หลายวิธี โดยหลักการใหญ่ๆ สามารถแบ่งออกได้ดังนี้

2.1 การควบคุมด้วยมือ (MANUAL CONTROL)

การควบคุมแบบนี้จะใช้ประสาทสัมผัสของคนเป็นตัววัดค่าปริมาณต่างๆ ที่เราต้องการควบคุมและก็ใช้คนอีกนั่นแหละที่ทำการปรับกลไกเพื่อให้ได้ค่าที่เราต้องการควบคุมเป็นไปตามที่เราต้องการนั่นเอง



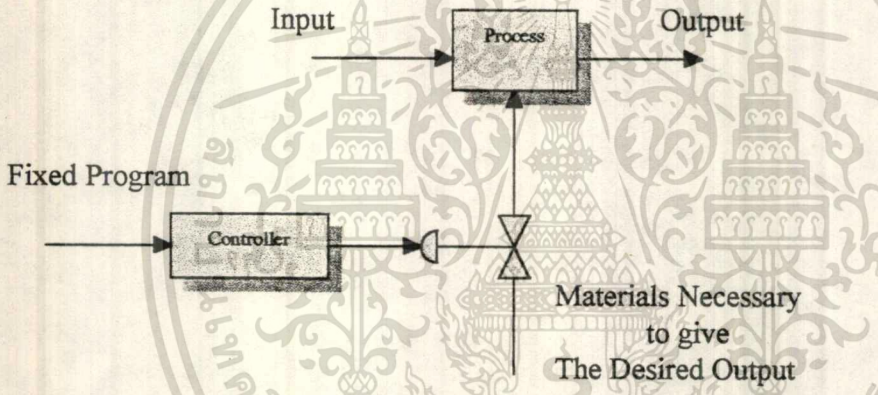
รูปที่ 3 การควบคุมอุณหภูมิด้วยมือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 การควบคุมอัตโนมัติ (AUTOMAIC CONTROL)

การควบคุมอัตโนมัติ หมายถึง การใช้ตัวควบคุม (CONTROLLER) ทำหน้าที่เปรียบเทียบตัดสินใจแทนมนุษย์นั่นเอง นอกจากนั้นการควบคุมแบบอัตโนมัติยังสามารถแบ่งออกได้อีกหลายประเภท ดังนี้

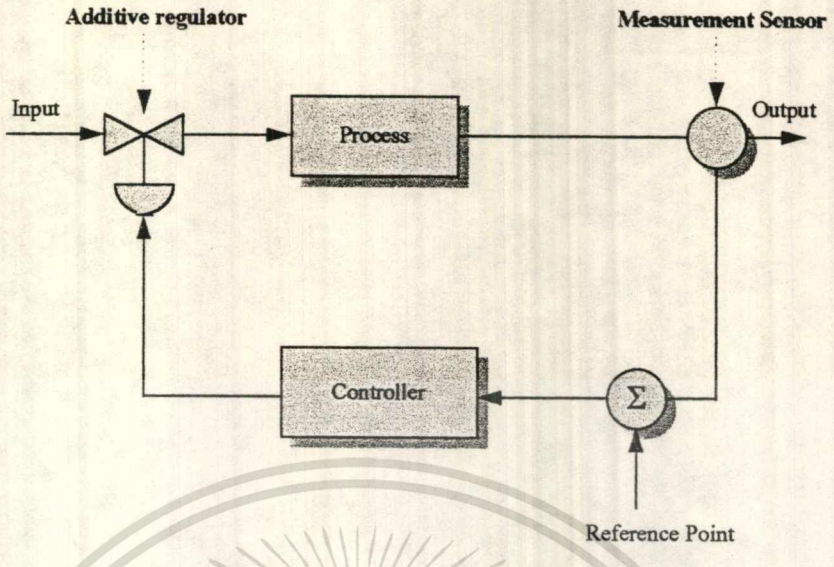
(1) OPEN-LOOP CONTROL เป็นวิธีการควบคุมที่ง่ายที่สุดโดยการใช้คำสั่งที่คาดว่าจะได้เป้าหมายตามที่ต้องการแต่ไม่สามารถจะตรวจสอบได้ว่า เป้าหมายที่ได้เป็นไปตามที่ต้องหรือไม่ เช่น การควบคุมเครื่องซักผ้า เป็นต้น โดยผู้ใช้จะคาดว่าเพื่อให้เสื้อผ้าเหล่านั้นจะสะอาดต้องซักกี่รอบและใช้ผงซักฟอกเท่าไร แล้วตั้งเครื่องตามที่คาดการไว้หลังจากเครื่องทำงานสิ้นสุดแล้ว เสื้อผ้าสะอาดก็นับว่าระบบการควบคุมทำงานสมบูรณ์ หากเสื้อผ้าไม่สะอาดเท่าที่ต้องการซึ่งอาจเนื่องมาจากความสกปรกมีมากผงซักฟองน้อยเกินไป ก็นับว่าระบบควบคุมทำงานไม่สมบูรณ์ ทั้งนี้เพราะเครื่องไม่ได้ตรวจสอบความสกปรกของเสื้อผ้า, ปริมาณผงซักฟอก, ความเย็นของน้ำ และผลหลังจากเสร็จงานแล้วดังกล่าว



รูปที่ 4 Open-Loop Control

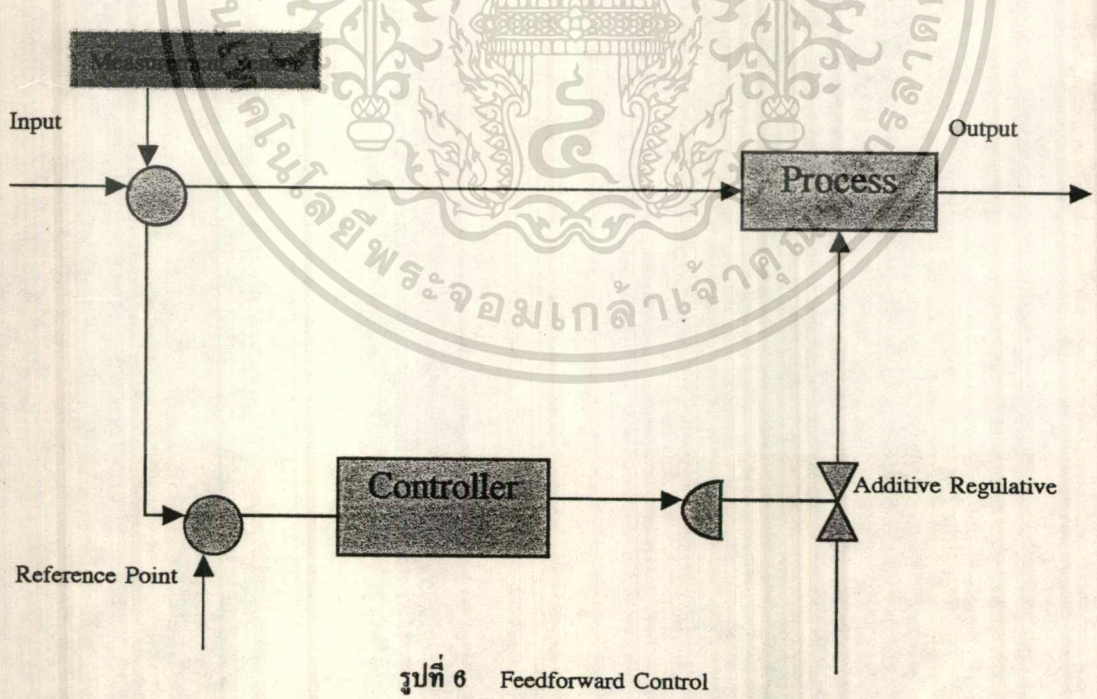
(2) FEEDBACK CONTROL จะมีชุดควบคุมหรือCONTROLLER ที่ทำงานได้รวดเร็วโดยจะทำการวัดค่าต่างๆ ที่ต้องการควบคุม (OUTPUT) และเปรียบเทียบกับค่าที่ต้องการ ซึ่งความแตกต่างของจุดที่ต้องการควบคุม และค่าความต้องการจะถูกปรับแต่งให้ถูกต้องโดยเครื่องควบคุมอัตโนมัติ เช่น PID CONTROLLER ซึ่งเป็นหัวข้อสำคัญของโครงงานนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5 Feedback Control

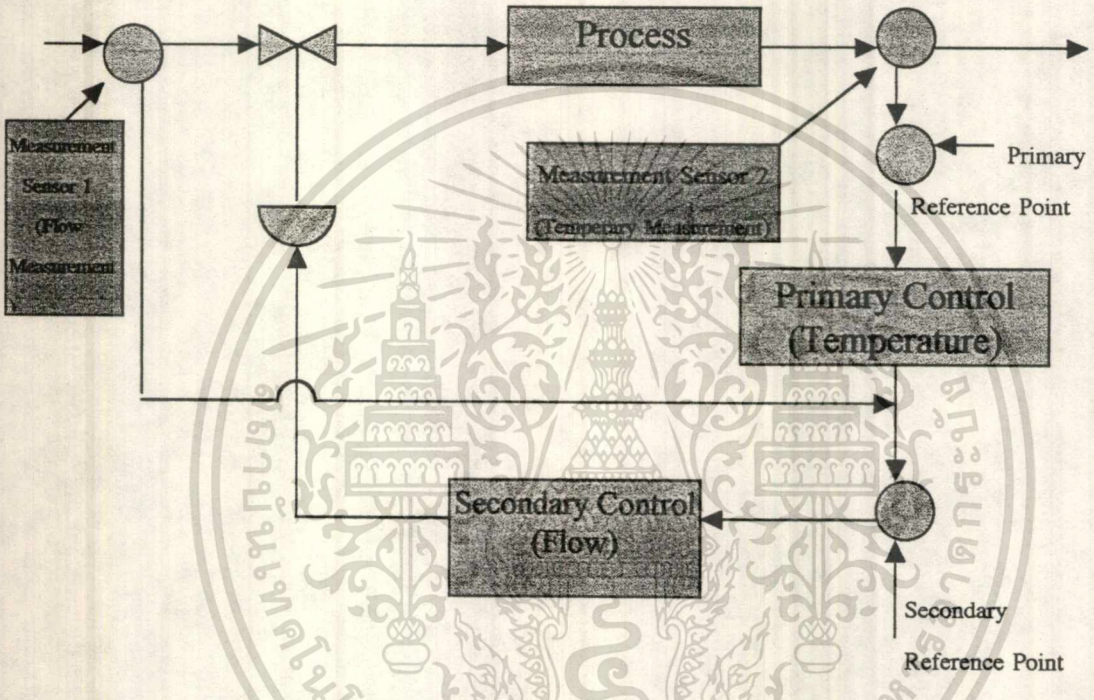
(3) FEEDFORWARD CONTROL เป็นระบบควบคุมที่มีความแตกต่างจาก OPEN-LOOP CONTROL คือมีการวัดสัญญาณ INPUT เปรียบเทียบกับสมมติฐานของคำสั่งควบคุม หากสัญญาณ INPUT ที่เข้ามาแตกต่างจาก REFERENCE มากไปจะสามารถคำนวณและปรับปรุงคำสั่งเพื่อให้ได้ผลลัพธ์ตามที่ตามต้องการ



รูปที่ 6 Feedforward Control

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(4) CASCADE CONTROL เป็นการควบคุมชนิด COMBINE CONTROL ชนิดหนึ่งที่ใช้กันมากในการควบคุมขบวนการ การควบคุมแบบ CASCADE CONTROL นี้จะมี LOOP การควบคุมอยู่ 2 LOOP ใน 2 LOOP นี้จะมีตัวควบคุม คือ PRIMARY CONTROLLER และ SECONDARY CONTROLLER สัญญาณของ PRIMARY CONTROLLER จะเป็นค่าเป้าหมายของ SECONDARY CONTROLLER การควบคุมแบบนี้จะใช้ได้ดีในการควบคุมขบวนการที่มีสิ่งรบกวนต่อขบวนการ (PROCESS DISTURBANCE) ที่มีค่า DEAD TIME และ TIME LAG มากๆ นอกจากนั้นยังใช้กรณีที่ต้องการรักษาความสัมพันธ์ของตัวแปรขบวนการทั้งสองให้คงที่ จะเหมาะสมที่สุด



รูปที่ 7 Cascade Control

หากแบ่งการควบคุมตามลักษณะการทำงาน แบ่งได้ 2 ระดับ คือ

(1) DIGITAL LEVEL CONTROL เป็นระบบที่ง่ายเหมาะสมสำหรับการควบคุมที่ไม่ต้องการความละเอียดมากขึ้น แต่เน้นหนักทางด้านการเรียงการทำงาน (SEQUENCE) นั้นคือ ความรับรู้หรือคำสั่งของเครื่องมี 2 สภาวะคือ สภาวะปิดและสภาวะเปิด หรือ LOGIC "0" และ LOGIC "1" เท่านั้น

(2) ANALOG LEVEL CONTROL เป็นระบบที่สามารถจะรับรู้หรือออกคำสั่งควบคุมได้อย่างต่อเนื่องตลอดช่วง MINIMUM จนถึงช่วง MAXIMUM อีกทั้งยังมีความแม่นยำสูงด้วย

ในโครงการนี้เป็นการศึกษาถึงการ ควบคุมแบบป้อนกลับซึ่งการควบคุมแบบป้อนกลับนี้สามารถแบ่งตามความสัมพันธ์ระหว่าง สัญญาณขาออกกับสัญญาณขาเข้าของตัวควบคุม (CONTROLLER) ความสัมพันธ์ดังกล่าวจะขึ้นอยู่กับกฎเกณฑ์การควบคุม (CONTROL LAW) ที่พนักงาน (OPERATOR) เป็นผู้ "ปรับตั้ง"ไว้ล่วงหน้า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แบบต่างๆ ของการควบคุมแบบป้อนกลับ (FEEDBACK CONTROL MODEL)

แบบของการควบคุมแบบป้อนกลับ ถูกกำหนดโดยความสัมพันธ์ระหว่างสัญญาณขาออกกับสัญญาณขาเข้าของตัวควบคุม(CONTROLLER) ความสัมพันธ์ดังกล่าวจะขึ้นอยู่กับกฎเกณฑ์การควบคุม (CONTROL LAW) ที่พนักงาน (OPERATOR) เป็นผู้ปรับตั้งไว้ล่วงหน้า

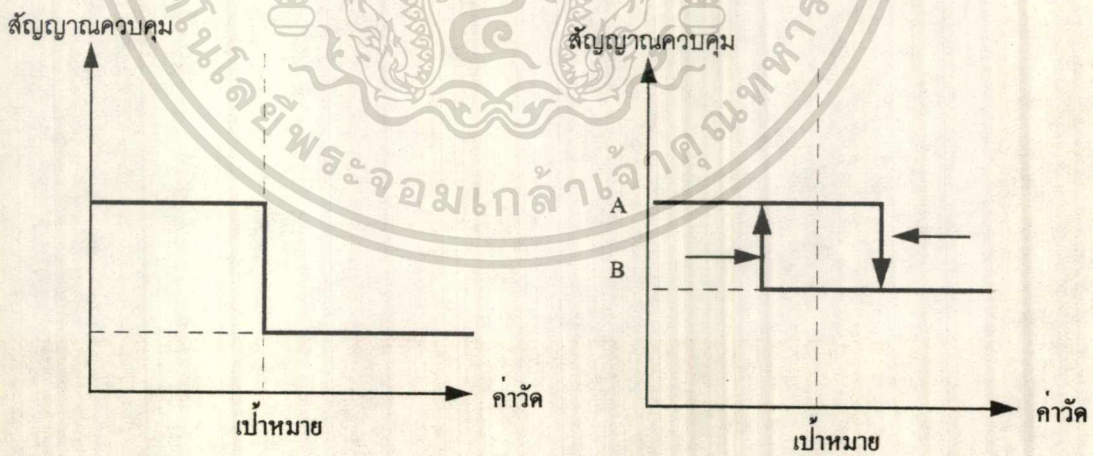
1. ON-OFF CONTROL ACTION

ON-OFF ACTION คือ ACTION ที่จะทำให้ความเกิดสัมพันธ์ของผลต่าง (E) กับสัญญาณควบคุม (CS) ดังนี้

เมื่อ $E > 0$; $CS = A$ ①

เมื่อ $E < 0$; $CS = B$ ②

โดยที่ A และ B คือ ค่าใดๆ 2 ค่าของ CS หรือสัญญาณควบคุม ดังนั้นการควบคุมแบบนี้ CS จะมีค่าเพียงสองค่าเท่านั้น บางครั้งจึงเรียกว่า 2-POSITION ACTION เช่น ให้วาล์วปิดเต็มที่หรือปิดเต็มที่ เป็นต้น จากความสัมพันธ์จะเห็นว่า เมื่อค่าวัด (MV) มีขนาดน้อยกว่าค่าเป้าหมาย (SP) ผลต่าง (E) จะเป็นบวก สัญญาณควบคุมจะมีค่าใดค่าหนึ่ง (Aหรือ B) ซึ่งจะทำให้ PROCESS เปลี่ยนไปในทางที่ค่าวัดจะสูงขึ้น ยกตัวอย่างเช่น เมื่ออุณหภูมิของน้ำร้อนที่ออกจาก HEAT EXCHANGER สูงเกินกว่าเป้าหมาย วาล์วจะถูกควบคุมให้ปิดเมื่ออุณหภูมิของน้ำร้อนลดลง แต่ถ้าลดลงกว่าเป้าหมายก็ให้เปิดวาล์วเพื่อให้อุณหภูมิสูงขึ้นมา

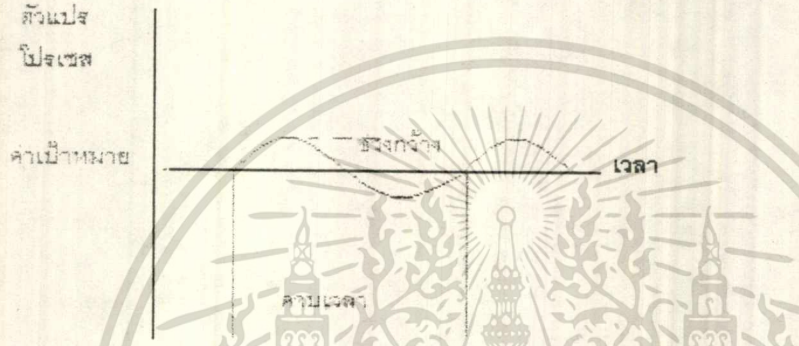


รูปที่ 1 การควบคุมแบบ ON - OFF

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตามปกติการควบคุมแบบ ON-OFF จะมี DEAD BAND ซึ่งเป็นอาณาเขตของค่าเป้าหมาย ซึ่งจะยอมให้ค่าวัดผิดพลาดไปจากค่าเป้าหมายได้บ้าง ก่อนที่จะมีการเปลี่ยนแปลงของสัญญาณควบคุม การควบคุมแบบ ON-OFF เป็นการเปิดปิดพลังงานที่เข้า PROCESS เป็น CYCLE ดังนั้นผลของการควบคุมอาจทำให้ PROCESS เกิดการแกว่ง (CYCLING) ขึ้นได้ ช่วงกว้าง (AMPLITUDE) และค่าเวลา (PERIOD) ในการแกว่งจะขึ้นอยู่กับลักษณะสมบัติของ PROCESS เอง เช่น DEAD-TIME, RESISTANCE, CAPACITANCE

การควบคุมแบบ ON-OFF จะได้ผลต่อเมื่อ PROCESS นั้นมีค่าคงตัวเวลายาวมาก แต่มี DEAD TIME มากเท่านั้น



รูปที่ 2 การแกว่งของผลการควบคุม

2. PROPORTIONAL ACTION

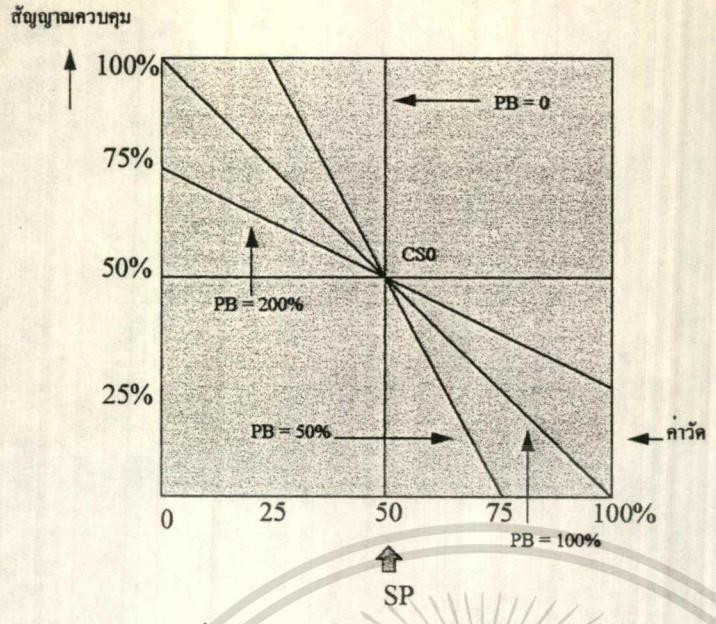
PROPORTIONAL ACTION คือ ACTION ที่ทำให้สัญญาณควบคุมแปรผันโดยตรงกับผลต่าง

K คือ PROPORTIONAL GAIN สัญญาณควบคุมก็คือ การขยายผลต่างด้วยอัตราขยาย K นั่นเอง ปกติในระบบการควบคุมมักจะไม่นิยมใช้ K แต่จะใช้ PROPORTIONAL BAND (PB) แทน โดยที่

$$PB = (1/K) * 100\% \qquad ③$$

PB มีความหมายคือ ผลต่าง (E) จะต้องเปลี่ยนไป PB% จึงจะพอดีทำให้สัญญาณควบคุมเปลี่ยนจาก 0 - 100%

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3 ความหมายของ PROPORTIONAL BAND

จะเห็นได้ว่าเมื่อ PB มีค่าเล็กลงอัตราขยายจะมีค่าสูงขึ้น เมื่อ PB = 0% อัตราขยายจะเป็นอนันต์ การทำงานของ PROPORTIONAL ACTION จะกลายเป็น ON-OFF ถ้าค่าเป้าหมายอยู่ที่ 50% และค่าวัดได้ 50% ผลต่างจะได้เท่ากับ 0 ขณะนั้นสัญญาณควบคุมจะมีค่า CS0 ดังนั้นสัญญาณควบคุมจะถูกกำหนดจากความสัมพันธ์ ดังนี้

$$CS = -KE + CS0 \qquad (4)$$

การที่จำเป็นต้องมี CS0 หรือที่เรียกว่า BIAS นั้นเพื่อให้ผลต่างเท่ากับ 0 สัญญาณควบคุม CS จะได้ไม่เท่ากับ 0 ไปด้วย เพราะจะทำให้ควบคุม PROCESS ไม่ได้

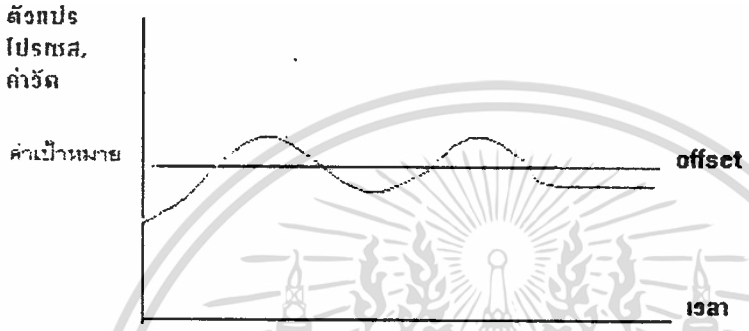
คุณสมบัติของ PROPORTIONAL ACTION

1. เมื่อลด PB ลงไป ทำให้อัตราขยายสูงขึ้น การควบคุมจะไว (SENSITIVE) ขึ้น ทำให้ PROCESS เกิดการแกว่งขึ้นได้
2. เมื่อเพิ่มค่า PB ขึ้น อัตราขยายลดลง จะทำให้ค่าวัดกับค่าเป้าหมายแตกต่างกันมากขึ้น เราเรียกว่าเกิด OFFSET
3. เมื่อ LOAD ของ PROCESS สภาพแวดล้อมเปลี่ยนไป ซึ่งเราเรียกว่า DISTURBANCE ก็จะเกิด OFFSET ได้ง่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. OFFSET

OFFSET เป็นชื่อใช้เรียก ปรากฏการณ์ที่ตัวแปร PROCESS หรือค่าวัดมีค่าไม่เท่ากับค่าเป้าหมาย ทำให้การควบคุมไม่เป็นไปตามที่ต้องการ ในระบบการควบคุมแบบป้อนกลับนี้ เมื่อดูอย่างผิวเผิน OFFSET ไม่น่าจะเกิดขึ้นได้ เพราะตัวแปร PROCESS จะถูกป้อนกลับมาเปรียบเทียบกับค่าเป้าหมายตลอดเวลา อย่างไรก็ตาม OFFSET มักจะเกิดในระบบควบคุมที่ใช้ PROPORTIONAL CONTROL ที่มี PB ใหญ่ และในขณะที่เกิด DISTURBANCE ใน PROCESS ซึ่งได้จากการเปลี่ยนแปลงของ LOAD สภาพแวดล้อม เป็นต้น



รูปที่ 4 OFFSET

วิธีแก้ OFFSET ให้เล็กลง

1. ให้ลด PB ให้เล็กลง
2. เปลี่ยน BIAS ของการควบคุมด้วยมือ (MANUAL RESET)
3. เปลี่ยนค่าเป้าหมายใหม่
4. ใช้การควบคุมที่มี RESET ACTION

4. RESET ACTOPM (INTEGRAL ACTION)

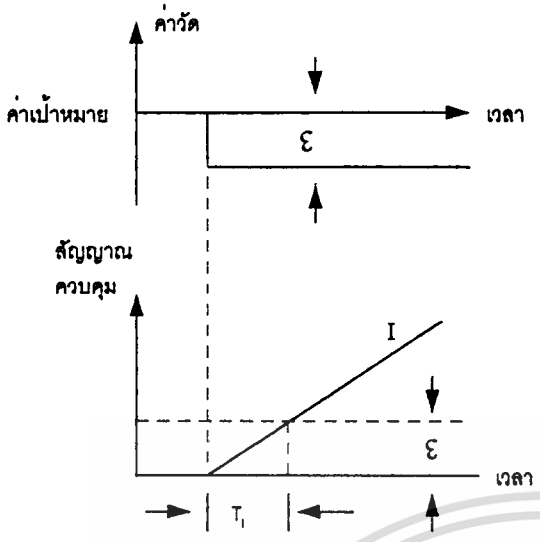
RESET ACTION คือ ACTION ที่ทำให้เกิดความสัมพันธ์ ดังนี้

$$CS = (1/T_I)\int E dt \qquad ⑤$$

โดย T_I คือ RESET TIME มีหน่วยเป็นนาที

การควบคุมแบบนี้สัญญาณควบคุมจะแปรตามค่า INTEGRATE ต่อเวลาของผลต่าง เมื่อผลต่างเปลี่ยนเป็น STEP สัญญาณควบคุมจะค่อยๆ เพิ่มค่าตามเวลา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5 ผลตอบของ RESET ACTION

ตามรูปความหมายของ RESET TIME(T_I) ก็คือเวลาที่ใช้ไปในเวลาเพิ่มค่าจะเพิ่มค่าของสัญญาณควบคุมจนมีขนาดเท่ากับผลต่าง (E) ที่เปลี่ยนแปลงไป ดังนั้นเมื่อT_I มีค่าน้อยเมื่อผลของ RESET ACTION จะมากกว่าเมื่อ T_I มีค่ามาก

5. DERIVATIVE ACTION (RATE ACTION)

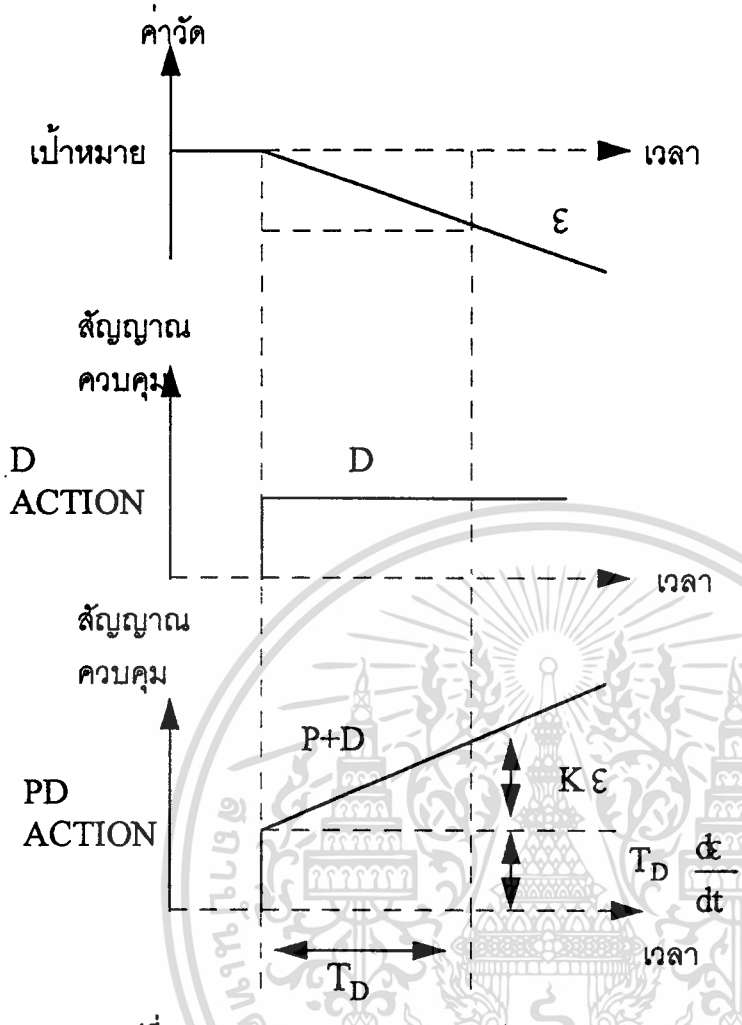
ACTION ที่ทำให้เกิดความสัมพันธ์ของสัญญาณควบคุมกับผลต่างเป็นไปตามสมการสูตร ดังนี้

$$CS = T_D * (dE/dt) \qquad ⑥$$

โดยที่ T_D คือ DERIVATIVE TIME (S)

สัญญาณควบคุมจะแปรผันตาม อัตราการเปลี่ยนแปลงต่อเวลาของผลต่างการควบคุมชนิดนี้มีผลในการหยุดการเปลี่ยนแปลงของตัวแปร PROCESS ที่อาจเปลี่ยนแปรไป เมื่อตัวแปร PROCESS เปลี่ยนไปด้วยอัตราหนึ่ง สัญญาณควบคุมจะถูกผลิตให้แปรผันกับ อัตราการเปลี่ยนแปรนี้ จะถูกสร้างขึ้นไปหยุดการเปลี่ยนแปลงนั้น D ACTION ที่เหมาะสำหรับไปใช้กับ PROCESS ที่มี TIME LAG มากๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6 D ACTION และ PD ACTION

จากรูปจะเห็นว่า T_D จะเท่ากับช่วงเวลาที่ผลต่างเปลี่ยนแปรไปจนมีค่าเท่ากับค่าความชันของการเปลี่ยนแปลงนี้ ถ้า T_D มากจะทำให้ผลของ D ACTION มาก

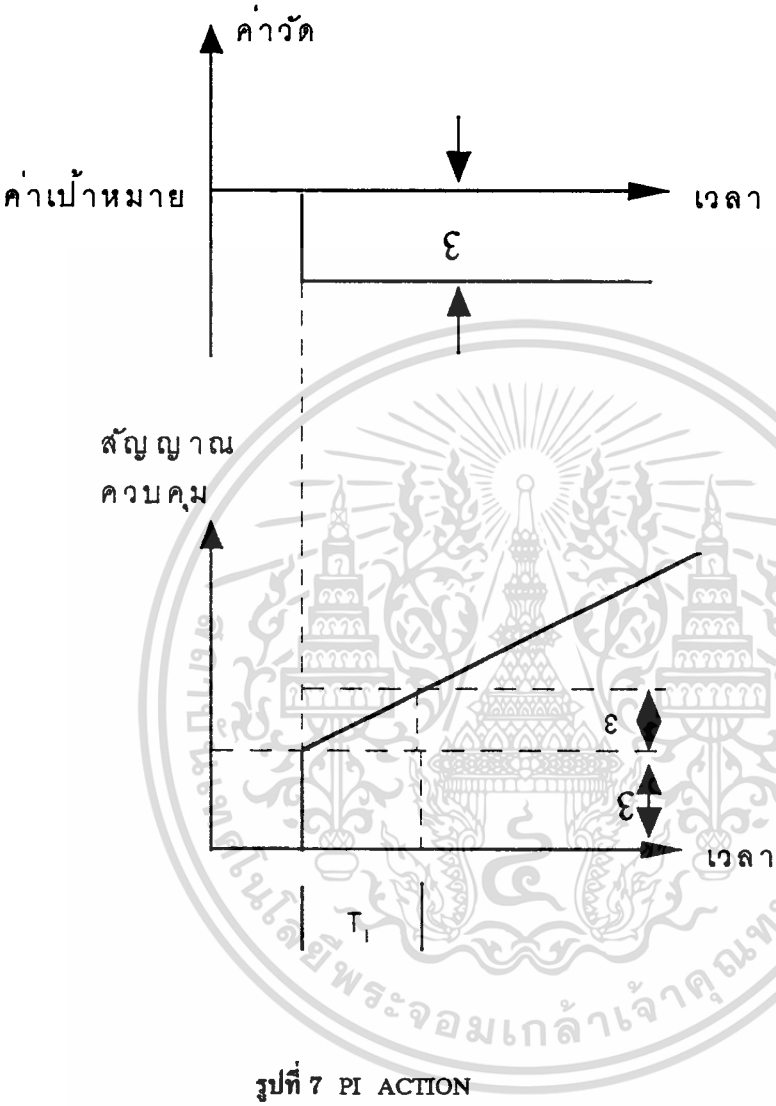
คุณสมบัติของ D ACTION

1. เหมาะสำหรับ PROCESS ที่มี TIME LAG จะช่วยให้การควบคุมถึงจุดที่ต้องการเร็วขึ้น
2. ถ้า T_D มากไปผลของ D จะมากทำให้ระบบทั้งระบบไวขึ้น ขาดเสถียรภาพ
3. ไม่เหมาะสำหรับ PROCESS ที่มี TIME LAG น้อยและตัวแปร PROCESS เปลี่ยนแปรได้ง่าย เช่น ระบบควบคุมการไหล ความดัน เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. PI ACTION

ตามปกติ RESET ACTION จะไม่ใช้โดดๆ แต่จะใช้คู่กับ P เสมอ



รูปที่ 7 PI ACTION

คุณสมบัติ RESET ACTION

1. OFFSET ที่เกิดจาก P ACTION จะถูก RESET ACTION แก้ไขจนหมดไป
2. เมื่อ T_I มีค่าน้อยลงผลของ RESET ACTION จะมากทำให้เกิดการแกว่งขึ้นได้ ซึ่งจะทำให้ระบบขาด เสถียรภาพ (UNSTABLE)

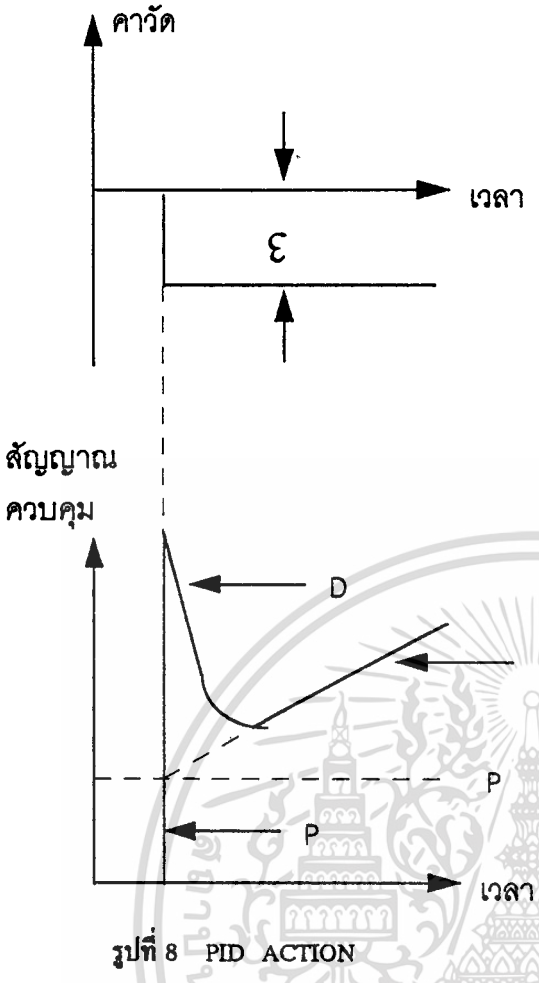
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

7. PROPORTIONAL-PLUS-INTEGRAL-PLUS-DERIVATIVE CONTROLLER

ในทางอุตสาหกรรมเรามักจะเรียกว่า "PROPORTIONAL PLUS RESET PLUS RATE CONTROLLER" ซึ่งการควบคุมแบบนี้จะอาศัยคุณสมบัติของ INTEGRAL ACTION ใน PI CONTROLLER ทำให้ตัวควบคุมตอบสนองต่อความผิดพลาดช้าลง กรณีที่ขบวนการช้าอยู่แล้วเราอาจจำเป็นต้องเพิ่ม DERIVATIVE ACTION (หรือ RATE ACTION) เพื่อลดความช้าของระบบควบคุมซึ่งเป็นคุณสมบัติแบบ PID CONTROLLER โดยที่เรามักจะพบการควบคุมแบบ PID CONTROLLER ในงานควบคุมอุณหภูมิเป็นส่วนใหญ่ ซึ่งผลตอบสนองของ DERIVATIVE ACTION จะไม่มีผลเมื่อสัญญาณขาเข้าเกิดการเปลี่ยนแปลง DERIVATIVE ACTION จะเพิ่มหรือลดสัญญาณขาออก ตามอัตราการเปลี่ยนแปลงของสัญญาณขาเข้า

บางครั้งอาจจะเข้าใจว่า DERIVATIVE ACTION จะปรับสัญญาณขาออกไปต้านทานความผิดพลาดล่วงหน้าก่อนที่ขนาดของความผิดพลาดจะมีค่ามากก็ถือว่าถูกต้องเพราะว่าการเพิ่ม DERIVATIVE ACTION เข้าไปใน PI CONTROLLER จะทำให้ผลตอบสนองของระบบไวต่อสิ่งรบกวนมากขึ้น สัญญาณวัดเปลี่ยนแปลงสู่ค่าเป้าหมายเร็วขึ้น โดยการปรับค่า T_D ให้มากขึ้นสัญญาณวัดยิ่งเปลี่ยนแปลงกลับสู่เป้าหมายได้เร็วขึ้นอย่างไรก็ตามถ้าเราปรับปรุงค่า T_D มากไปสัญญาณวัดอาจเกิดการแกว่งได้ กล่าวโดยสรุปคือ ข้อดีของ PID CONTROLLER จะทำให้ระบบควบคุมตอบสนองต่อสิ่งรบกวน (DISTURBANCE) เร็วขึ้น ทำให้เสถียรภาพของการควบคุมดีกว่า PI CONTROLLER สำหรับขบวนการที่ตอบสนองช้า หรือการควบคุมแบบนี้เหมาะกับขบวนการที่มี CAPACITY LAG โตๆ และ DEAD TIME ขนาดไม่โตนักซึ่งส่วนใหญ่จะเกี่ยวข้องกับการควบคุมอุณหภูมิ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 8 PID ACTION

ในระบบควบคุมที่สัญญาณวัดมีสัญญาณรบกวน (NOISE) เราไม่ควรใช้ DERIVATIVE ACTION แม้ว่าสัญญาณรบกวนจะมีขนาดเล็ก แต่อัตราการเปลี่ยนแปลง (ความถี่) มักจะมีค่าสูง DERIVATIVE ACTION ซึ่งมีขนาดโดยตรงกับอัตราการเปลี่ยนแปลงอาจจะทำให้สัญญาณขาออกผิดพลาดไปอย่างมากมาย เพื่อจะแก้ปัญหา NOISE ในทางปฏิบัติจึงมักจะเพิ่มการกรองความถี่ (FILTERING) ด้วยค่า TIME CONSTANT = TD/N เมื่อ N มีค่าเท่ากับ 3-10 ใน DERIVATIVE ACTION

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คุณสมบัติบางประการของขบวนการ

คุณสมบัติของขบวนการที่เราน่าจะรู้จักเพื่อที่จะเป็นประโยชน์ในการควบคุม ได้แก่

1. DEAD TIME

ซึ่งหมายถึงช่วงเวลานับจากขณะที่สัญญาณควบคุมเปลี่ยนแปลงไปจนกระทั่งสัญญาณวัดของตัวแปรขบวนการเริ่มเปลี่ยนแปลง โดยในระบบควบคุมนั้น DEAD TIME จะเป็นเวลาที่ตัวควบคุมจะต้องคอยเพื่อที่จะทราบผลของการเปลี่ยนแปลงสัญญาณควบคุมที่ตัวควบคุมได้ส่งออกไปแล้ว ยิ่ง DEAD TIME มีค่ามากตัวควบคุมยิ่งต้องคอยนานขึ้น

ในการออกแบบที่ดีนั้นเราจะพยายามลดค่า DEAD TIME โดยการกำหนดจุดวัดและค่าต่างๆ ในขบวนการให้มีผลตอบสนองอย่างรวดเร็วโดยระยะในการส่งสัญญาณนั้นสั้นที่สุด

2. CAPACITY LAG

คือ ความช้าในการตอบสนอง เป็นตัวจำกัดอัตราการเปลี่ยนแปลงสูงสุดของสัญญาณวัด ถ้ามีค่าน้อยช่วงเวลาที่สัญญาณมีค่าเปลี่ยนแปลงจากสภาวะสมดุลย์ (STEADY STATE) เดิมไปสู่สภาวะสมดุลย์ใหม่จะนานขึ้น

CAPACITY LAG จะพบในขบวนการที่มีการกักเก็บพลังงานหรือวัตถุ ซึ่งจะถูกถ่ายเทออกจากขบวนการไปสู่ที่อื่น เราอาจจะเรียก CAPACITY LAG ได้อีกอย่างหนึ่งว่า TRANSFER LAG

สรุปแล้วถ้า CAPACITY LAG ยิ่งมีค่ามากสัญญาณวัดก็ยิ่งจะเปลี่ยนแปลงช้ามากเท่านั้น ตัวควบคุมก็จะสามารถเปลี่ยนแปลงสัญญาณควบคุม ไปต่อต้านการเปลี่ยนแปลงของสัญญาณวัดได้ง่ายยิ่งขึ้นทำให้การควบคุมนั้นสามารถทำได้ง่ายยิ่งขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวควบคุม PID แบบ DIGITAL

เมื่อมีการควบคุมแบบ DIGITAL INPUT และ OUTPUT ของ CONTROLLER จะต้องเป็นสัญญาณ DIGITAL หรือสัญญาณที่สุ่มรับค่าเข้า(SAMPLED) มากกว่าที่จะเป็นสัญญาณANALOG หรือสัญญาณต่อเนื่อง

สัญญาณต่อเนื่องที่ถูกส่งมาจาก TRANSMITTER จะถูกสุ่มรับค่าเข้ามา และแปลงเป็นสัญญาณ DIGITAL ด้วย ANALOG-TO-DIGITAL CONVERTER(ADC) ด้วยระยะเวลาที่คงที่

ALGORITHM ของDIGITAL CONTROL จะถูกนำมาใช้เพื่อคำนวณ OUTPUT ของ CONTROLLER เป็นสัญญาณ DIGITAL ก่อนที่ OUTPUT ของ CONTROLLER จะถูกส่งออกไปยังอุปกรณ์ควบคุมตัวสุดท้าย (เช่น CONTROL VALVE) สัญญาณ DIGITAL จะถูกแปลงไปเป็นสัญญาณต่อเนื่องโดย DIGITAL-TO-ANALOG CONVERTER(DAC)

ความก้าวหน้าของ DIGITAL VERSION จากสมการ PID CONTROLLER ดังสมการที่ ❺ คือการแปลงเทอม INTEGRAL และ DERIVATIVE เป็นแบบ DISCRETE โดยประมาณการ INTEGRAL ด้วยการรวม(SUMMATION) และการ DERIVATIVE โดยหาผลแตกต่างย้อนหลังแบบ FIRST-ORDER ดังสมการ

$$m_n = \bar{m} + Kc \left[e_n + \Delta t/\tau_I \sum e_k + \tau_D/\Delta t(e_n - e_{n-1}) \right] \quad ⑫$$

เมื่อ

Δt = เวลาที่ใช้ในการสุ่มรับค่าเข้ามา(SAMPLING PERIOD) เป็นเวลาที่อยู่ระหว่างการเก็บค่าตัวอย่างที่ตามๆ กันมาของตัวแปรที่ถูกควบคุม

m_n = OUTPUT ของ CONTROLLER ขณะที่มีการสุ่มรับค่าข้อมูลครั้งที่ n

n = 1, 2, ...

e_n = ERROR ครั้งที่ n ของการสุ่มรับค่าข้อมูล

สมการที่ ⑫ นี้ เรียกว่า POSITION FORM ของPID ALGORITHM เมื่อOUTPUT ของ CONTROLLER จริงๆ ถูกคำนวณ

ทางเลือกใหม่ที่ดีกว่า คือ การใช้ VELOCITY FORM ของ ALGORITHM ซึ่งใช้ในการเปลี่ยนแปลง (CHANGE)ของ OUTPUT ของ CONTROLLER ในการคำนวณ

ทำสมการที่ ⑫ สำหรับการสุ่มรับค่าข้อมูลครั้งที่ (n-1)

$$m_{n-1} = \bar{m} + Kc \left[e_{n-1} + (\Delta t/\tau_I)\sum e_k + \tau_D/\Delta t(e_{n-1} - e_{n-2}) \right] \quad ⑬$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การรวม(SUMMATION) จะเริ่มต้นที่ k = 1 นั่นคือ ระบบมีสภาวะคงที่ตามที่ต้องการ เมื่อ k<= 0 หรือ $e_k = 0$ สำหรับ k<= 0

สมการที่ ⑫ กับสมการที่ ⑬

$$\Delta m_n = m_n - m_{n-1} = Kc[(e_n - e_{n-1}) + \Delta t/\tau_1 e_n + \tau_D/\Delta t(e_n - e_{n-1} + e_{n-2})] \quad ⑭$$

และ OUTPUT ของ CONTROLLER คำนวณได้โดย

$$m_n = P_{n-1} + Kc[(e_n - e_{n-1}) + (\Delta t/\tau_1)e_n + \tau_D/\Delta t(e_n - e_{n-1} + e_{n-2})] \quad ⑮$$

$$\Delta m_n = m_n - m_{n-1}$$

$$= Kc[(1 + \Delta t/\tau_1 + \tau_D/\Delta t)e_n + (-1-2\ \tau_D/\Delta t\ )\ e_{n-1} + \tau_D/\Delta t\ \ e_{n-2}] \quad ⑯$$

$$m_n = m_{n-1} + [k_1 e_n + k_2 e_{n-1} + k_3 e_{n-2}] \quad ⑰$$

$$k_1 = Kc(1 + K_i dt + K_d/dt), \quad k_2 = -Kc(1 + 2K_d/dt), \quad k_3 = KcK_d/dt$$

การเปรียบเทียบระหว่าง POSITION และ VELOCITY ALGORITHMS (COMPARISON OF POSITION AND VELOCITY ALGORITHMS)

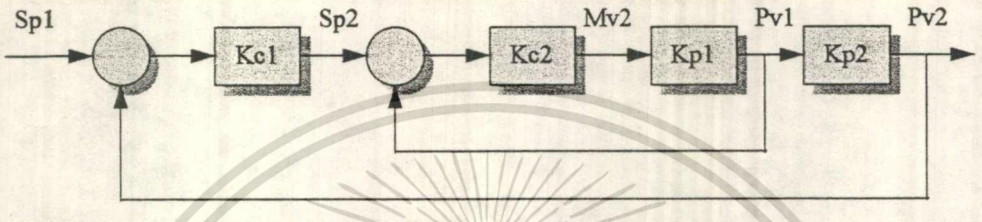
1. POSITION FORM ของ PID CONTROLLER ต้องการค่า m ในขณะที่แน่นอนเลยว่า VELOCITY FORM ไม่ต้องการค่าที่สภาวะคงที่ สำหรับ OUTPUT ของ CONTROLLER อย่างไรก็ดี เป็นการง่ายที่จะเริ่มต้นให้แต่ละ ALGORITHM มีค่าเท่ากัน เมื่อเปลี่ยนระบบควบคุมจาก MANUAL ไปเป็นแบบ AUTOMATIC อันจะเป็น BUMPLESS TRANSFER ในการลด BUMP ของ PROCESS

2. เมื่อ $\bar{m}$ (หรือ m_{n-1} สำหรับ VELOCITY ALGORITHM) ทำสัญญาณที่ส่งไปยังอุปกรณ์ควบคุมตัวสุดท้ายให้เท่ากันได้ง่าย ขณะเปลี่ยน MODE และ VELOCITY FORM หลีกเลี่ยงการคำนวณ $\int e(t)dt$ ได้ ทำให้ VELOCITY FORM มีแนวโน้มในการเกิดปัญหา RESET WIND UP น้อยกว่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การควบคุมแบบ CASCADE (CASCADE CONTROL)

การควบคุมแบบ CASCADE หรือการควบคุมแบบลดหลั่นกันไปเป็นการควบคุมที่มีประโยชน์มากในโรงงานอุตสาหกรรม หลักการทำงาน คือ การต่อ CONTROLLER 2 ตัว เข้าด้วยกัน โดยที่ OUTPUT หรือค่า MANIPULATED VARIABLES ของ CONTROLLER ตัวแรกจะเป็น SET POINT ให้กับ CONTROLLER ตัวที่ 2 ดังรูปที่ 15



รูปที่ 15 การควบคุมแบบ CASCADE

โดยทั่วไปแล้ว CONTROLLER ที่ควบคุมตัวแปรที่ถูกควบคุมตัวแรก จะเรียกว่า MASTER CONTROLLER, OUTER CONTROLLER หรือ PRIMARY CONTROLLER และ CONTROLLER ที่ตัวควบคุมตัวแปรที่ถูกควบคุมตัวที่2 จะเรียกว่า SLAVE CONTROLLER, INNER CONTROLLER หรือ SECONDARY CONTROLLER โดยเทอมของ PRIMARY/SECONDARY จะน่าสนใจกว่า เพราะสามารถเผื่อไว้สำหรับระบบหลายๆ LOOP

ในการออกแบบควบคุมแบบ CASCADE ข้อควรพิจารณาที่สำคัญที่สุด คือ INNER หรือ SECONDARY LOOP ต้องเร็วกว่า OUTER หรือ PRIMARY LOOP ข้อพิจารณานี้เป็นสิ่งที่รู้กันดีทั่วไปและข้อพิจารณานี้สามารถจะขยายไปในหลายๆ LOOP ได้ คือใน ระบบควบคุมที่มี 3 LOOP CASCADE, TERTIARY LOOP จะต้องเร็วกว่า SECONDARY LOOP และ PRIMARY LOOP

โดยปกติแล้ว เมื่อเราประยุกต์ใช้อย่างถูกต้อง การควบคุมแบบ CASCADE จะทำให้ LOOP ทั้งหมดเสถียรมากกว่า และมีการตอบสนองที่เร็วกว่า

มีอยู่ 2 คำถามที่ยังคงอยู่ คือ เราจะทำอย่างไรในการเอาการควบคุมแบบ CASCADE ใส่ลงไปในการปฏิบัติงานแบบ AUTOMATIC และ เราจะ TUNE CONTROLLER อย่างไร คำถาม LOOP ที่อยู่ในสุด จะถูก TUNE ก่อน และถูกเปลี่ยนเป็น AUTOMATIC ขณะที่ LOOP ที่อยู่ข้างนอกยังเป็น MANUAL อยู่ ทำให้เหมือนอย่างนี้เรื่อยไป จน LOOP นอกสุดเป็น AUTOMATIC และสิ่งที่สำคัญที่ต้องระวังคือ ACTION ของ CONTROLLER โดยดูความต้องการของ PROCESS และ ACTION ของ CONTROL VALVE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบ Digital Control

เนื่องจากในโครงงานนี้เป็นการศึกษาการควบคุม ซึ่งใช้ไมโครคอมพิวเตอร์มาทำเป็นตัวควบคุมขบวนการ (CONTROLLER) โดยอาศัยความสามารถของคอมพิวเตอร์ ในการประมวลผลข้อมูลทางคณิตศาสตร์ และความสามารถในการแสดงผลข้อมูลในลักษณะของรูปกราฟิกซึ่งการควบคุมแบบนี้เป็นการควบคุมแบบที่เรียกว่า "DIGITAL CONTROL" การที่เราจะออกแบบระบบควบคุมแบบ DIGITAL CONTROL นั้นจำเป็นจะต้องทราบถึงทฤษฎีการสุ่มข้อมูล (SAMPLE DATA THEORY) ให้เข้าใจเสียก่อนเพราะเป็นส่วนที่สำคัญส่วนแรกของระบบ DIGITAL CONTROL

1. นิยามของ SAMPLE DATA และ SAMPLING PERIOD

ใน ANALOG CONTROLLER สัญญาณ INPUT ของ CONTROLLER จะเป็นสัญญาณในลักษณะข้อมูลต่อเนื่อง (CONTINUOUS DATA SIGNAL) โดยสัญญาณเหล่านี้ จะถูกส่งเข้าไปเพื่อ OPERATE โดยวงจรอิเลคทรอนิคส์ภายใน CONTROLLER ตลอดเวลาการทำงาน แต่ในระบบ DIGITAL CONTROL นั้น ตัว CONTROLLER นั้นไม่สามารถรับสัญญาณข้อมูลต่อเนื่องในแบบของ ANALOG ได้เพราะหลักการทำงานของ DIGITAL CONTROLLER จะมีลำดับขั้น ดังนี้

(1) รับสัญญาณ ANALOG เข้าที่ส่วนอินพุท ณ เวลาใดเวลาหนึ่งที่กำหนดไว้

(2) แปลงสัญญาณ ANALOG นี้ให้เป็นสัญญาณ DIGITAL แล้วนำไปคำนวณ และ CONTROLLER ทำงานตามที่ประมวลผลข้อมูลได้ ซึ่งจะใช้ระยะเวลาช่วงหนึ่ง

(3) นำผลลัพธ์จากข้อ (2) ซึ่งเป็นสัญญาณ DIGITAL มาแปลงเป็นสัญญาณ ANALOG เพื่อส่งเป็นสัญญาณ OUTPUT ที่ออกจากตัว DIGITAL CONTROLLER และก็นำไปควบคุมที่ FINAL CONTROL ELEMENT

(4) เริ่มทำงานตามข้อ 2,3 เช่นนี้เรื่อยไปตลอดการทำงาน

จากหลักการทำงานของ DIGITAL CONTROLLER ที่กล่าวมา เราจะเห็นได้เลยว่า CONTROLLER แบบนี้จะไม่สามารถรับสัญญาณ INPUT ได้ตลอดเวลาเพราะจะต้องใช้ช่วงเวลาหนึ่งหยุดรับสัญญาณเข้า เพื่อไปทำงานในข้อ 2,3 แล้วจึงเริ่มรับสัญญาณเข้ามาใหม่ต่อไปโดยมีช่วงเวลาของการรับสัญญาณ INPUT เท่ากันทุกครั้งตลอดเวลาการทำงาน ซึ่งสัญญาณที่เราได้รับจากการรับเข้ามาในช่วงเวลาหนึ่งนั้นเองที่เราเรียกว่า สัญญาณข้อมูลสุ่ม (SAMPLED DATA SIGNAL OR SAMPLING SIGNAL) และช่วงห่างคงที่ของเวลาที่ใช้ในการรับสัญญาณ INPUT แต่ละครั้งจะเรียกว่า ช่วงการสุ่ม (SAMPLING PERIOD OR SAMPLING INTERVAL) ซึ่งช่วงเวลานี้จะต้องสั้นพอที่จะทำให้ DIGITAL CONTROLLER สามารถรับและนำไปคำนวณได้อย่างไม่เกิดการคลาดเคลื่อน ในการใช้งานจริงๆ ช่วงเวลาที่ใช้ในการสุ่มข้อมูลนี้จะมีค่าเป็น USEC หรืออย่างมากเป็น MSEC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. การออกแบบระบบ DIGITAL CONTROL

การออกแบบระบบ DIGITAL CONTROL นั้น จะมีขั้นตอนหลักๆ อยู่ 2 ขั้นตอน คือ

(1) จะต้องทำให้ขบวนการ (PROCESS OR PLANT) อยู่ในรูปฟอร์มทางคณิตศาสตร์ เพื่อหาค่าคงที่ต่างๆ และวิเคราะห์พฤติกรรมได้

(2) ออกแบบตัว CONTROLLER ให้เหมาะสมกับผลตอบสนองที่ต้องการจากขบวนการภายใต้ระบบควบคุม

การออกแบบระบบ DIGITAL CONTROL จะต้องทำการเปลี่ยนแปลงระบบให้อยู่ในรูปแบบเป็นช่วงๆ (DISCRETE FORM) โดยออกแบบให้อยู่ในรูป ANALOG FORM แล้วจึงเปลี่ยนในรูปของ Z-DOMAIN หรือออกแบบให้อยู่ในรูป Z-DOMAIN โดยตรงเลยก็ได้

2.1 การออกแบบ PID CONTROLLER

การเปลี่ยน PID ให้อยู่ในรูปของ DISCRETE FORM มี 2 วิธี คือ วิธี RECTANGULAR APPROXIMATIONS และวิธี TRAPEZOIDAL APPROXIMATION ในวิธี RECTANGULAR APPROXIMATION การออกแบบจะทำให้อยู่ในรูป ANALOG FORM ก่อนแล้วจึงเปลี่ยนให้อยู่ในรูปของ Z-DOMAIN และสำหรับวิธี TRAPEZOIDAL APPROXIMATION การออกแบบจะทำให้อยู่ในรูปของ Z-DOMAIN โดยตรง โดยใช้เทคนิคการแทนที่ POLE (POLE REPLACEMEN TECHNIQUES)

สมการ PID ทาง ANALOG

$$u(t) = K_P\, e(t) + K_i \int e dt + K_d \left(\frac{de}{dt}\right) \quad \text{..... ①}$$

ซึ่ง K_P, K_i, K_d = ค่าคงที่ของ PID

$u(t)$ = สัญญาณขาออก (Output Signal)

$e(t)$ = สัญญาณผิดพลาด (Error Signal)

2.2 RECTANGULAR APPROXIMATION

ในการเปลี่ยนแปลงรูปแบบของสมการ สมมติช่วงเวลาของการ SAMPLING คือ T จากสมการ (1) เทอมของการอินทิเกรท edt สามารถแทนด้วยการนำสี่เหลี่ยมผืนผ้าเล็กๆ ใต้กราฟมาบวกกัน ซึ่งผลลัพธ์ที่ได้ คือ

$$edt = T \times \Sigma e(i) \text{ ; เมื่อ } i = 1 - n \quad \text{..... ②}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในเทอม Differential de/dt ก็สามารถแทนค่าได้ด้วย

$$de/dt = \frac{e(n) - e(n-1)}{T} \quad \text{.....} ③$$

ซึ่ง e(n) และ e(n-1) คือค่าของสัญญาณผิดพลาด e( ) ที่ช่วงเวลา (n) และ (n-1) ดังนั้นสมการ PID ทาง ANALOG จึงสามารถเปลี่ยนแปลงให้อยู่ในรูป DISCRETE FORM ได้ดังนี้

$$u(n) = K_p\ e(n) + K_i \Sigma ei\ T + \frac{K_d\ [e(n) - e(n-1)]}{T} \quad \text{.....} ④$$

จากสมการที่ 4 แทนค่า (n) ด้วย (n-1)

$$u(n-1) = K_p\ e(n-1) + K_i \sum_{i=1}^{n-1} ei\ T + \frac{K_d\ [e(n) - e(n-1)]}{T} \quad \text{.....} ⑤$$

นำสมการที่ 5 ลบออกจากสมการที่ 4 จะได้

$$u(n) - u(n-1) = K_p\ e(n) - K_p\ e(n-1) + K_i\ [e(n)]T + (K_d/T)\ [e(n) - 2e(n-1) + e(n-2)] \quad ⑥$$

จัดรูปใหม่จะได้ว่า

$$u(n) = u(n-1) + [K_p + K_iT + (K_d/T)]e(n) - [K_p + (K_d/T)]\ e(n-1) + (K_d/T)\ e(n-2) \quad ...⑦$$

$$u(n) = u(n-1)\ K_1\ e(n) + K_2\ e(n-1) + K_3\ e(n-2) \quad \text{.....} ⑧$$

เมื่อ $K_1 = K_p + K_iT + (K_d/T)$

$K_2 = -[K_p + (2K_d/T)]$

$K_3 = K_d/T$

2.3 TRAPEZOIDAL APPROXIMATION

ถ้าต้องการความแม่นยำมากขึ้นกว่าวิธี RECTANGULAR APPROXIMATION ก็ให้ใช้วิธี TRAPEZOIDAL APPROXIMATION โดยการนำพื้นที่สี่เหลี่ยมคางหมูเล็กๆ มาบวกรวมกัน โดยใช้สมการ $\int edt$

จากสมการ PID

$$u(t) = K_p\ e(t) + K_i \int edt + K_d\ de/dt$$

ซึ่งจาก สมการ u(t) TAKE LAPLACE จะได้ว่า

$$U(s) = K_p\ E(s) + K_i\ [E(s)/s] + K_d\ sE(s) \quad \text{.....} ⑨$$

จากนั้นทำการเปลี่ยนให้อยู่ในรูปของ DISCRETE FORM โดยการเปลี่ยน S-DOMAIN เป็น Z-DOMAIN

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$s = (2/T)\frac{(1-Z^{-1})}{(1+Z^{-1})} \quad \text{.....} \quad (10)$$

จากนั้นแทนในสมการของ U(s) จะได้ว่า

$$U(z) - U(z)z^{-2} = K_1 E(z) + K_2 E(z)z^{-1} + K_3 E(z)z^{-2} \quad \text{.....} \quad (11)$$

ซึ่ง $K_1 = K_P + 2K_d/T + K_i T/2$

$K_2 = K_i T - 4K_d/T$

$K_3 = K_i T/2 + 2K_d/T - K_P$

จากนั้นแทน z^{-1} ด้วยเวลาในการสุ่ม 1 ช่วงจะได้ว่า

$$U(n) = U(n-2) + K_1 e(n) + K_2 e(n-1) + K_3 e(n-2) \quad \text{.....} \quad (12)$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. การปรับค่า PID

3.1 ผลตอบสนองของการควบคุมที่ดี

ในการควบคุมแบบป้อนกลับนั้น ตัวควบคุมจะพยายามรักษาให้ค่าตัวแปรของ PROCESS มีค่าเท่ากับค่าเป้าหมายอยู่เสมอ ในกรณีที่เกิด DISTURBANCE ในระบบที่มีการเปลี่ยนค่าเป้าหมาย จะทำให้ตัวแปร PROCESS มีค่าต่างจากค่าเป้าหมายไปขณะหนึ่ง ตัวควบคุมจะพยายามควบคุมให้ตัวแปร PROCESS มีค่าเท่ากับค่าเป้าหมายในที่สุด ลักษณะการนำค่าตัวแปร PROCESS ให้เข้าใกล้ค่าเป้าหมายนี้จะแตกต่างกันตามคุณสมบัติของระบบควบคุมบางระบบควบคุมจะควบคุมให้ตัวแปร PROCESS ส่งเข้าหาค่าเป้าหมายรวดเร็ว แต่บางระบบอาจทำได้ดีกว่า เราสามารถทดสอบความสามารถ ของระบบควบคุมนี้ โดยดูผลตอบสนองของการควบคุม

ระบบควบคุมในรูปถ้าลองเปลี่ยนค่าเป้าหมาย ตัวควบคุมจะพยายามควบคุมให้ตัวแปร PROCESS วิ่งเข้าหาค่าเป้าหมายนั้น ผลตอบสนองของตัวแปร PROCESS จะมีหลายแบบ

ลักษณะสัญญาณ จากผลตอบสนอง

(A) มี OVERSHOOT และมีการแกว่ง

(B) ผลตอบรับเร็ว

(C) ตัวแปร PROCESS มีค่าไม่เท่ากับค่าเป้าหมายแม้เวลาผ่านพ้นไปนานเรียกว่าเกิด OFFSET

(D) ผลการตอบรับช้ามาก

เมื่อตัวแปร PROCESS หยุดนิ่งที่ค่าเป้าหมาย ในขณะนั้น เกิดมี DISTURBANCE เข้ามารบกวน PROCESS (DISTURBANCE ได้แก่ การเปลี่ยนแปลงของสภาพแวดล้อม เป็นต้น)

ผลตอบสนองของ PROCESS จะมีหลายแบบ ดังแสดงในรูป 3.2(ค)

(E) เกิด OFFSET ได้ค่าผิดไปจากค่าเป้าหมายเดิม

(F) เกิดการแกว่งเล็กน้อย ก่อนกลับเข้าสู่ค่าเป้าหมายเดิม

(G) เกิดการแกว่งและขาดเสถียรภาพ

เมื่อพิจารณาผลตอบสนองของการควบคุมชนิดต่างๆเหล่านี้ จึงพอสรุปได้ว่า ระบบการควบคุมที่ดีจะต้องมีคุณสมบัติ ดังนี้

(1) มีเสถียรภาพ ไม่เกิดการแกว่ง (OSCILLATION) เมื่อถูกกระตุ้น

(2) ตอบรับการเปลี่ยนค่าเป้าหมายเดิม

(3) ไม่เกิด OFFSET

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 ความยากง่ายในการควบคุม PROCESS

DEAD TIME เป็นศัตรูตัวร้ายของการควบคุม

ลักษณะสมบัติของ PROCESS นั้นทั่วๆไปจะมี TIME LAG ดังที่ได้กล่าวมาแล้วจากรูปคลื่นผลตอบสนองของ PROCESS ต่อ STEP INPUT (LE) เราสามารถหาค่า DEAD TIME (TE) และค่าคงตัวเวลาโดยประมาณได้

อัตราส่วน LE/TE นี้จะเป็นค่าที่ใช้ประเมินความยากง่ายในการควบคุมและใช้เลือกแบบการควบคุม

1.0 < LE/TE ควบคุมแบบ PID ได้ทั่วๆไป

0.5 < LE/TE < 1.0 การควบคุมแบบ PID ก็ควบคุมได้ยากมาก

LE/TE	แบบการควบคุมที่เหมาะสมสำหรับ Process นั้นๆ
LE/TE < 0.2	On-off, P, PI
0.2 < LE/TE < 1.0	PI, PID
1.0 < LE/TE	FEED FORWARD, COMPUTER CONTROL

ตารางที่ 3.1

สาเหตุของการขาดเสถียรภาพในระบบ

(1) ผลของระบบควบคุมแบบข้างเคียงหรืออื่นๆ

(2) DISTURBANCE เป็น PERIODIC

(3) มี NON-LINEAR ELEMENT ในระบบเอง เช่น BACKFLASH ความฝืด DEAD BAND เป็นต้น

3.3 ผลของ PID ต่อเสถียรภาพของระบบ

ในระบบควบคุมแบบป้อนกลับซึ่งใช้ตัวควบคุมแบบ PID นั้น ถ้าเราทดลองแปรค่า PB, TI, TD จะมีผลตอบสนองของระบบควบคุม ดังนี้

(1) ผลของ P ACTION

เมื่อลดค่า PB ลง ทำให้อัตราขยายสูงขึ้น จะมีผลทำให้

(1.1) OFFSET ลดลง

(1.2) PERIOD ของการแกว่งลดลง

(1.3) อัตราส่วนของช่วงกว้างการแกว่งเพิ่มขึ้น ระบบจะขาดเสถียรภาพมากขึ้น

(2) ผลของ I ACTION

เมื่อให้ P และ D ACTION มีค่าคงที่แล้ว ทดลองลด TI (RESET TIME) จะมีผลทำให้เกิด

(2.1) OFFSET หายไป

(2.2) ผลตอบสนองจะเร็วขึ้น (FAST RESPONSE)

(2.3) อัตราส่วนช่วงกว้างการแกว่งเพิ่มขึ้น ระบบจะขาดเสถียรภาพมากขึ้น

(3) ผลของ D ACTION

เมื่อให้ P และ I ACTION มีค่าคงที่แล้ว ทดลองเพิ่ม T_D (DERIVATIVE TIME)ให้ยาวขึ้น จะทำให้

(3.1) อัตราส่วนช่วงกว้างการแกว่งลดลง ระบบจะมีเสถียรภาพมากขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(3.2) PERIOD ของการแกว่งสั้นลง

ตามปกติการใช้ค่าเวลา T_D ให้ยาวนานมีแนวโน้มที่จะทำให้ระบบมีเสถียรภาพมากขึ้นก็จริง แต่ก็มีจุดอ่อนตรงผลตอบรับต่อ NOISE ได้ง่าย ทำให้ผลตอบสนองของระบบไวเกินไป ผลตอบสนองของการควบคุมแบบต่างๆ ต่อการเปลี่ยนแปลงค่าเป้าหมายและการเกิด DISTURBANCE โดยตั้งค่า PB, T_i , T_D ต่างกัน

3.4 คุณภาพของการควบคุม

ดังที่ได้กล่าวมาแล้ว ระบบควบคุมที่ดีจะต้องมีคุณสมบัติ คือ มีเสถียรภาพผลตอบเร็วและไม่เกิด OFFSET แต่จะใช้อะไรในการตัดสินใจว่า การควบคุมไหนจะดีที่สุดนั้นจะต้องมีเกณฑ์ตัดสินที่มีหลักเกณฑ์

เกณฑ์ในการวัดที่ใช้ตัดสินคุณภาพการควบคุม มีดังนี้

(1) อัตราส่วนช่วงกว้างการแกว่ง (DAMPING RATIO CRITERION)

(2) พื้นที่การควบคุมน้อยสุด (MINIMUM CONTROL AREA CRITERION)

(3) ความสูงของ OVERSHOOT แรก (OVERSHOOT AMPLITUDE CRITERION)

(4) ผลตอบต่อ DISTURBANCE (MINIMUM DISTURBANCE CRITERION)

(5) ผลตอบเชิงความถี่เชิงระบบ (FREQUENCY RESPONSE CRITERION)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ฟังก์ชั่นของตัวแปรในกระบวนการ

การวัดอัตราการไหลเชิงมวลของแก๊สที่ผ่านกระบวนการท่อโดยออริฟิก(ORIFIC)เป็นอุปกรณ์สำหรับกำหนดค่าความแตกต่างของแรงดันมวลของแก๊สจะเป็นฟังก์ชั่นกับอุณหภูมิ,ความดัน,และความหนาแน่นตามสมการ

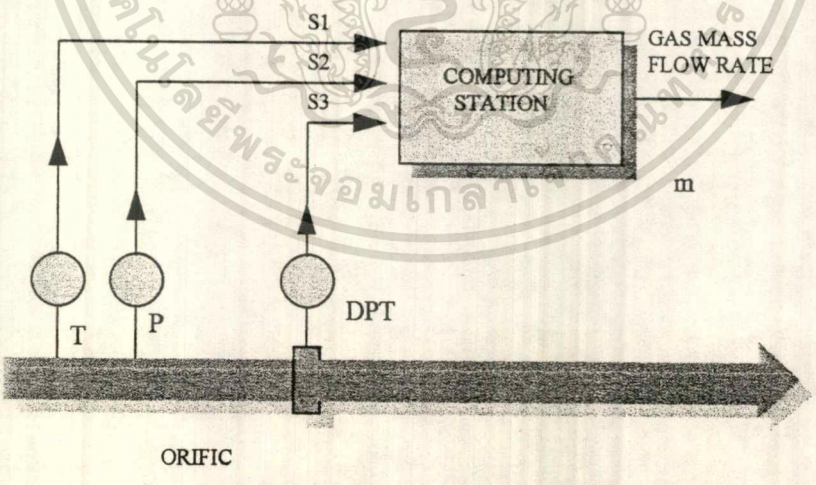
$$\dot{m} = K[h\rho]^{1/2} \qquad (1)$$

และความหนาแน่นของแก๊สในอุดมคติได้จาก

$$\rho = \frac{Mp}{RT} \qquad (2)$$

เมื่อ

$\dot{m}$	มวลของแก๊ส
R	ค่าคงที่ของแก๊สในอุดมคติ
ρ	ความหนาแน่นของแก๊ส
M	น้ำหนักโมเลกุลของแก๊ส
T	อุณหภูมิของแก๊ส
p	แรงดันของแก๊สในท่อ
K	สัมประสิทธิ์ของออริฟิก
h	ความแตกต่างของแรงดันที่ตกคร่อมออริฟิก



รูปที่ 1 แสดงหลักการวัดอัตราไหลเชิงมวลของแก๊ส

จะเห็นว่าอัตราการไหลเชิงมวลของแก๊สจะเป็นค่าจำเพาะของแก๊สนั้น และเป็นตัวแปรที่เป็นสภาวะแวดล้อมความแม่นยำในการวัดจะต้องนำค่าเหล่านี้มาทำการคำนวณในขณะที่แก๊สไหลผ่านท่อเราสามารถที่จะทำการโปรแกรมฟังก์ชั่นการทำงานทางคณิตศาสตร์ของอัตราการไหลโดยวัดตัวแปร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อุณหภูมิ,ความดันความแตกต่างของแรงดันทางขาเข้าของเครื่องควบคุมและส่งผลไปควบคุมที่ทางออกต่อไป

อย่างไรก็ตามฟังก์ชั่นดังกล่าวไม่เป็นเชิงเส้นจะต้องมีกรรมวิธีปรับเป็นเชิงเส้นก่อนที่จะทำการโปรแกรมในส่วนของเครื่องควบคุมต่อไป

ฟังก์ชั่นของเครื่องควบคุม

ฟังก์ชั่นการควบคุมที่เป็นแบบสัดส่วนร่วมกับอินทิกรัลและเดอริเวทีฟในปัจจุบันก็ยังคงถูกนำมาใช้ในการควบคุมกระบวนการในอุตสาหกรรมอยู่โดยกริยาของเครื่องควบคุมที่เป็นความสัมพันธ์กับค่าตัวแปรกระบวนการเมื่อค่าเป้าหมายยังคงที่สามารถแสดงได้ดังสมการต่อไปนี้

$$e(t) = SV - PV(t) \qquad (3)$$

$$m(t) = \bar{m} + K_p e(t) + K_i \int e(t) + K_d \frac{de(t)}{dt} \qquad (4)$$

เมื่อ

SV	ค่าเป้าหมาย
PV(t)	ค่าตัวแปรกระบวนการ
e(t)	ค่าความผิดพลาด
m(t)	สัญญาณทางออกของเครื่องควบคุม
$\bar{m}$	ค่าไบอัสของเครื่องควบคุม
K_p	อัตราขยายสัดส่วน
K_i	อัตราขยายอินทริกัล
K_d	อัตราการขยายเดอริเวทีฟ

จะเห็นว่าอัตราการขยายทั้งสามนั้นไม่มีผลกระทบต่อกันซึ่งทำให้การปรับแต่งเครื่องควบคุมเป็นไปได้ง่าย

โครงสร้างของเครื่องควบคุม

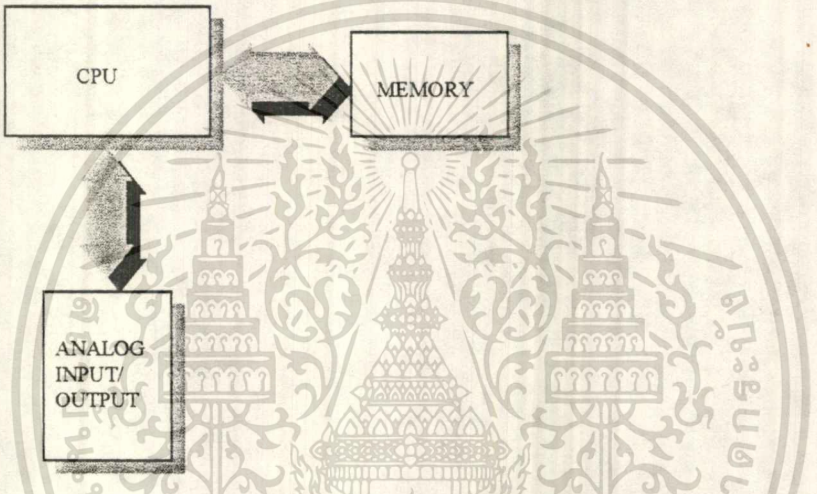
สำหรับหัวข้อนี้จะขอกล่าวแบบกว้างๆ และจะขยายความต่อไปในหัวข้อถัดไปถึงพิกัดต่างๆของเครื่องควบคุมกระบวนการที่โปรแกรมได้ทำการวิจัยนี้โดยเป็นระบบคอมพิวเตอร์ ที่ใช้หน่วยประมวลผลด้วยคอมพิวเตอร์ ประกอบกับ CARD PC818 ซึ่งมีทางเข้าหรืออินพุทที่รับสัญญาณอนาลอกจำนวน 4 ทางเข้าและสัญญาณทางออกแบบอนาลอก 2 ทางออกเช่นเดียวกันมีพารามิเตอร์และแสดงผล(KEYBOARD AND DISPLAY)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

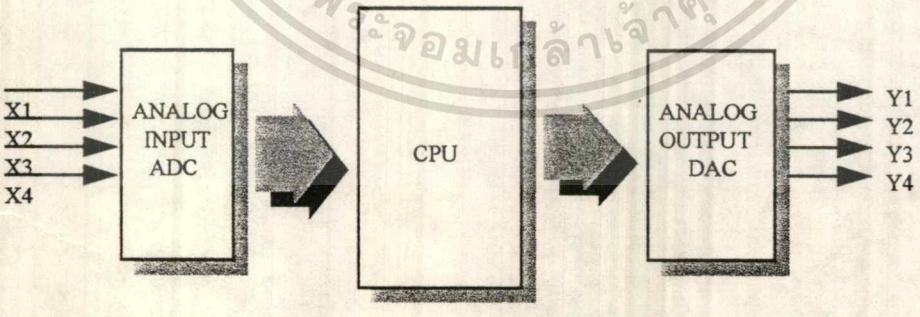
พิกัดและข้อมูลจำเพาะของเครื่องควบคุม

ทางเข้าพิกัด 4 - 20 mA, 1 - 5 Vdc โดยใช้อุปกรณ์แปลงสัญญาณข้อมูลอนาลอกเป็นดิจิตอล A/D ขนาด 8 บิททางออก พิกัด 4 -20 mA,1 - 5 Vdc โดยใช้อุปกรณ์แปลงข้อมูลดิจิตอลเป็นอนาลอก D/A ขนาด 12 บิท วงจรสำหรับหน่วยประมวลผลถูกแสดงในรูปที่ 2 และ รูปที่ 3 แสดงทางเข้าและทางออกแบบอนาลอก

ส่วนการรับพารามิเตอร์เพื่อปรับแต่งจะใช้คีย์บอร์ด และแสดงผลด้วย MONITOR ของ COMPUTER โดยแสดงวงจรดังรูปที่ 4

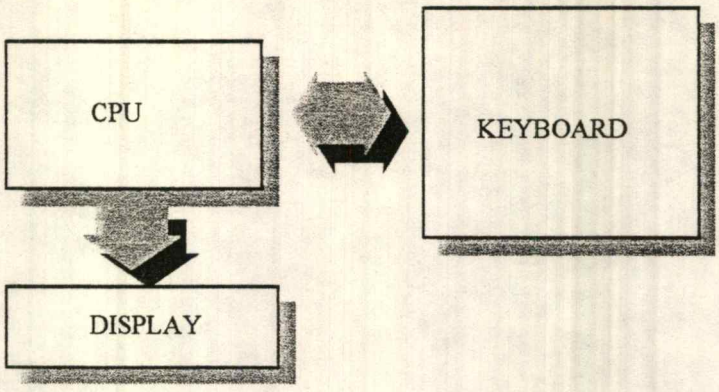


รูปที่ 2 แสดงบล็อคไดอะแกรมของหน่วยประมวลผล



รูปที่ 3 แสดงบล็อคของทางเข้าและทางออก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4 แสดงบล็อคของส่วนคีย์รับพารามิเตอร์เพื่อการปรับแต่งและแสดงผล

การโปรแกรม

การโปรแกรมจะถูกโปรแกรมโดยคอมพิวเตอร์ผ่านทางอุปกรณ์สื่อสาร PCL-818การโปรแกรมสามารถกระทำได้โดยใช้รหัสที่เป็นคำสั่งแก่เครื่องควบคุมดังนี้

คำสั่ง	ตัวแปร	ความหมาย
80H	LD	นำสัญญาณช่องที่กำหนดเข้ามา
81H	PID	เป็นวงรอบการควบคุมเดี่ยวแบบพื้นฐาน PID
82H	CAS	เป็นวงรอบการควบคุมแบบ CASCADE
83H	OUT	นำสัญญาณออกทางช่องที่กำหนด
84H	ADD	บวกสัญญาณสองสัญญาณที่กำหนด
85H	SUB	ลบสัญญาณสองสัญญาณที่กำหนด
86H	MUL	คูณสัญญาณสองสัญญาณที่กำหนด
87H	DIV	หารสัญญาณสองสัญญาณที่กำหนด
88H	SQR	ถอดรากที่สองของสัญญาณที่กำหนด
89H	END	จบการทำงานของโปรแกรม
90H	X1	ช่องทางเข้าที่ X1
91H	X2	ช่องทางออกที่ X2
92H	X3	ช่องทางออกที่ X3
93H	X4	ช่องทางเข้าที่ X4
A0H	Y1	ช่องทางออกที่ Y1
A1H	Y2	ช่องทางออกที่ Y2
A2H	Y3	ช่องทางออกที่ Y3
A3H	Y4	ช่องทางออกที่ Y4
B0H	S1	ค่าเป้าหมายที่ S1

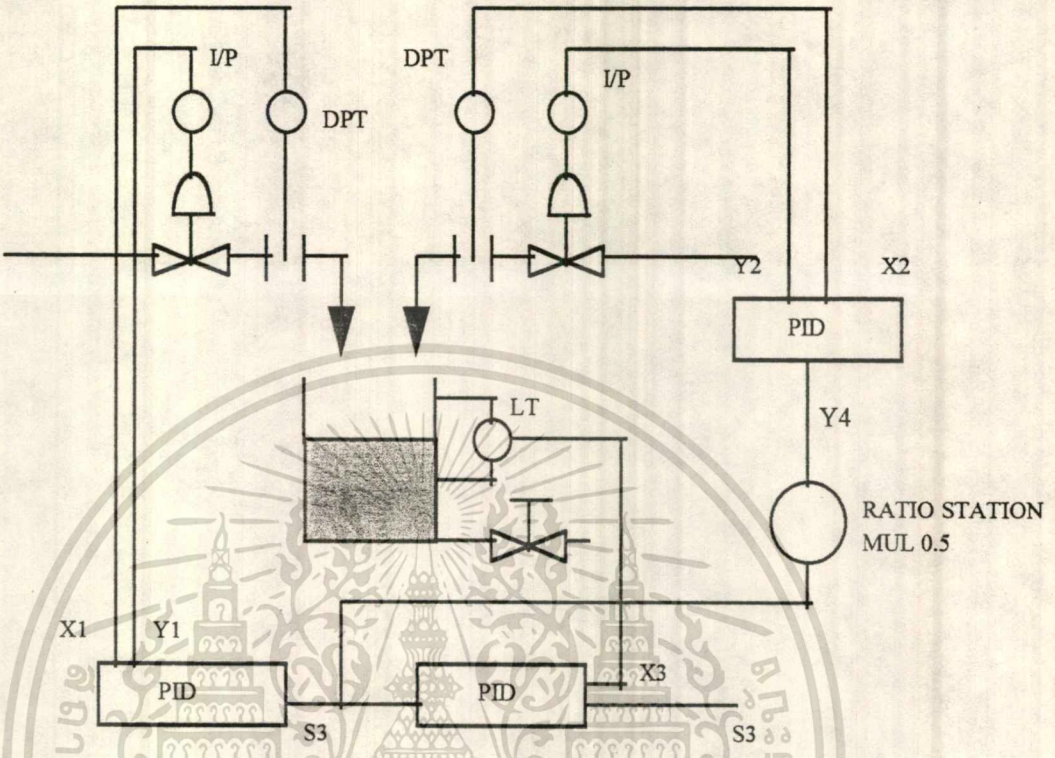
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำสั่ง	ตัวแปร	ความหมาย
B1H	S2	ค่าเป้าหมายที่ S2
B2H	S3	ค่าเป้าหมายที่ S3
B3H	S4	ค่าเป้าหมายที่ S4

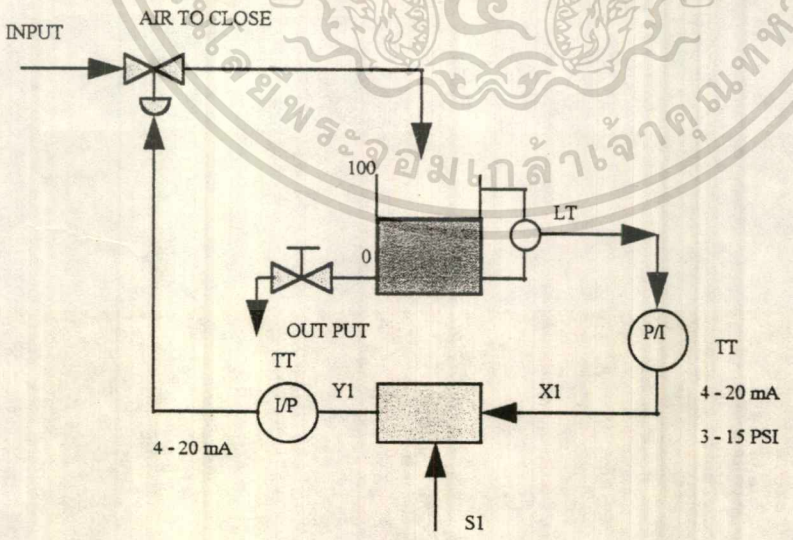
รหัสที่ได้แสดงข้างบนนี้เป็นเพียงส่วนหนึ่งเท่านั้น
ตัวอย่างการโปรแกรมเช่น เมื่อต้องการให้โปรแกรมมีการควบคุม
แบบอัตราส่วนของอัตราการไหล ของวงรอบ SLAVE โดยที่วงรอบของ
ระดับของเหลวเป็น MASTER ของการควบคุมแบบ CASCADE

คำสั่ง	ตัวแปร	ความหมาย
LD	X3	นำสัญญาณช่องที่ X3 เข้ามา
PID	S3	ใช้วงรอบการควบคุมที่ 3 โดย SP เป็น S3
OUT	Y3	นำสัญญาณออกทางช่องที่ Y3
LD	X1	นำสัญญาณช่องที่ X1 เข้ามา
SQR		ถอดรากที่สองสัญญาณช่องที่ X1
PID	Y3	ใช้วงรอบการควบคุมที่ 1 โดย SP เป็น Y3
OUT	Y1	นำสัญญาณออกทางช่องที่ Y1
LD	Y3	นำสัญญาณช่องออกที่ Y3 เข้ามา
MUL	0.50	นำค่า 0.50 มาคูณกับสัญญาณช่องที่ 3
OUT	Y4	นำสัญญาณออกทางช่องที่ Y4
LD	X2	นำสัญญาณช่องที่ X2 เข้ามา
SQR		ถอดรากที่สองสัญญาณช่องที่ X2
PID	Y4	ใช้วงรอบการควบคุมที่ 2 โดย SP เป็น Y4
OUT	Y2	นำสัญญาณออกทางช่องที่ Y2
END		จบการทำงานของโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5 แสดงกระบวนการที่ต้องการควบคุมในตัวอย่าง



รูปแสดง การทดลองใช้ Loop1

S1 คือค่า Set Point ที่ช่อง S1

X1 คือค่า Input ที่ช่อง X1

Y1 คือค่า Output ที่ช่อง Y1

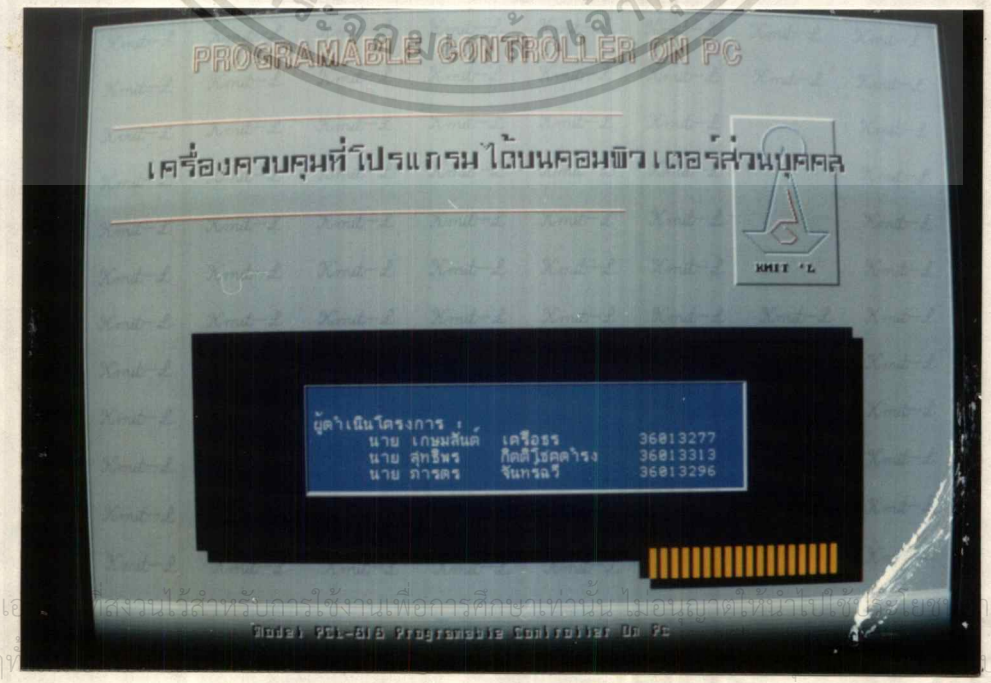
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

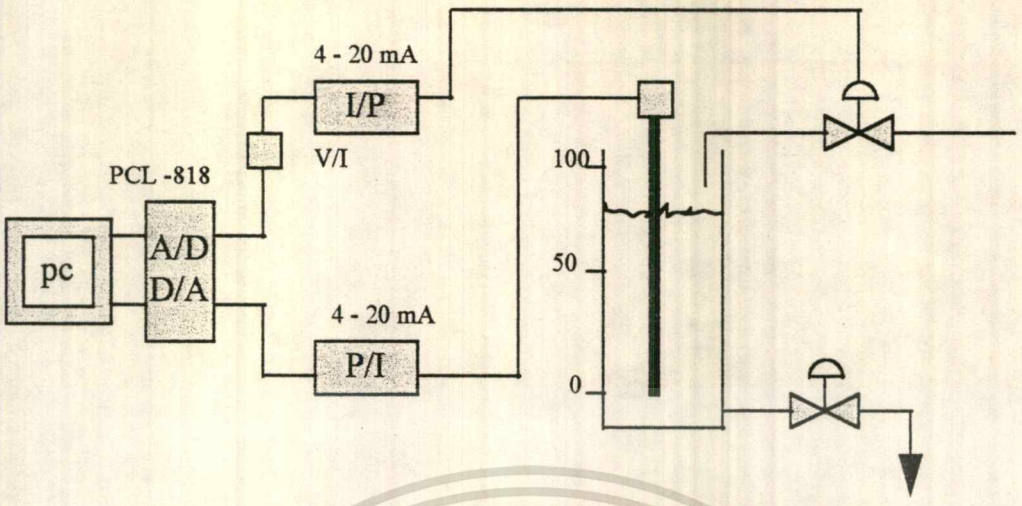
ผลการทดลอง

ในการพัฒนาการใช้โปรแกรมการควบคุมระบบการทำงานของกระบวนการนี้เราได้ใช้ โปรแกรม CTT 3.0 ซึ่งจะอธิบายการใช้งานดังนี้

เริ่มจากหน้า menu หลังจาก run โปรแกรมแล้ว จะเป็นดังนี้ :



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ในการใช้



รูปแสดงการทำงานของการควบคุม 1 Loop

ขั้นตอนการทำงาน

1) จะต้องตั้ง File ขึ้นโดยมีชื่อและจุดก่อน เช่น

C:\md Text.

C:\cd Text.

C:\Text.\

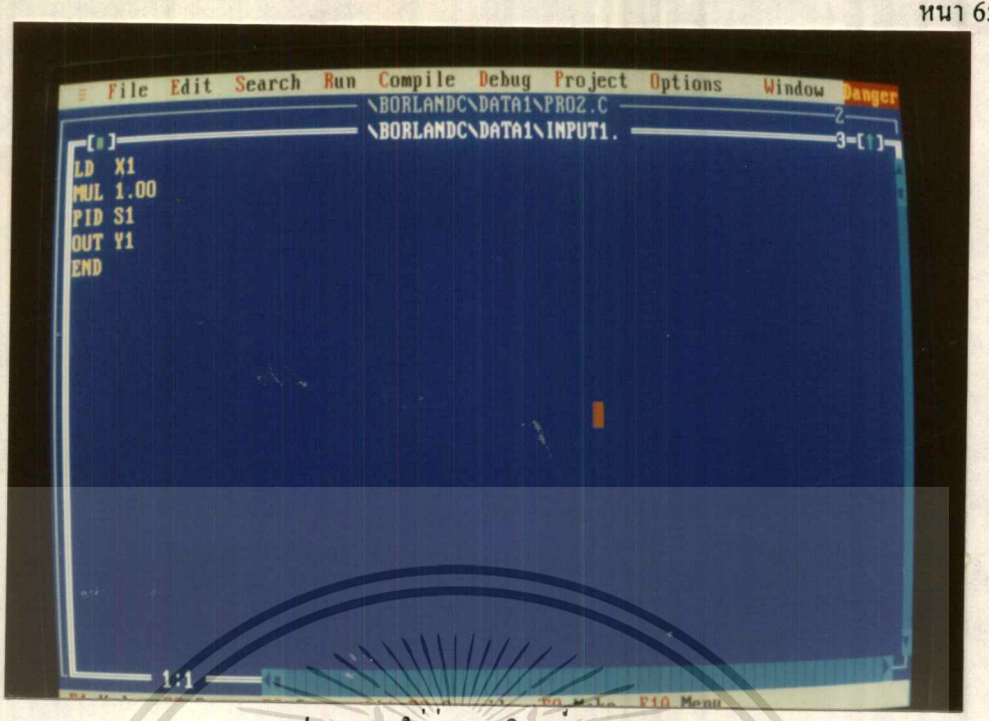
จากนั้นก็เข้าไปใน edit ของ File Text.

C:\edit Text.

2) เมื่อสร้างชื่อ File เรียบร้อยแล้วก็เขียนโปรแกรมให้ทำงานตามหลัก Programmable คือจะต้องเขียนลักษณะ ดังนี้ :

Opcode	Operand	
LD	X1	(รับค่าจากCH1 มาเก็บไว้ที่หน่วยความจำ)
PID	S1	(เข้า PID โดยตั้งค่าsetpoint ไว้และเปรียบเทียบ)
OUT	Y1	(นำสัญญาณออกไปที่ Y1)
END		(จบโปรแกรม plot กราฟ)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดงการใส่ค่าพารามิเตอร์

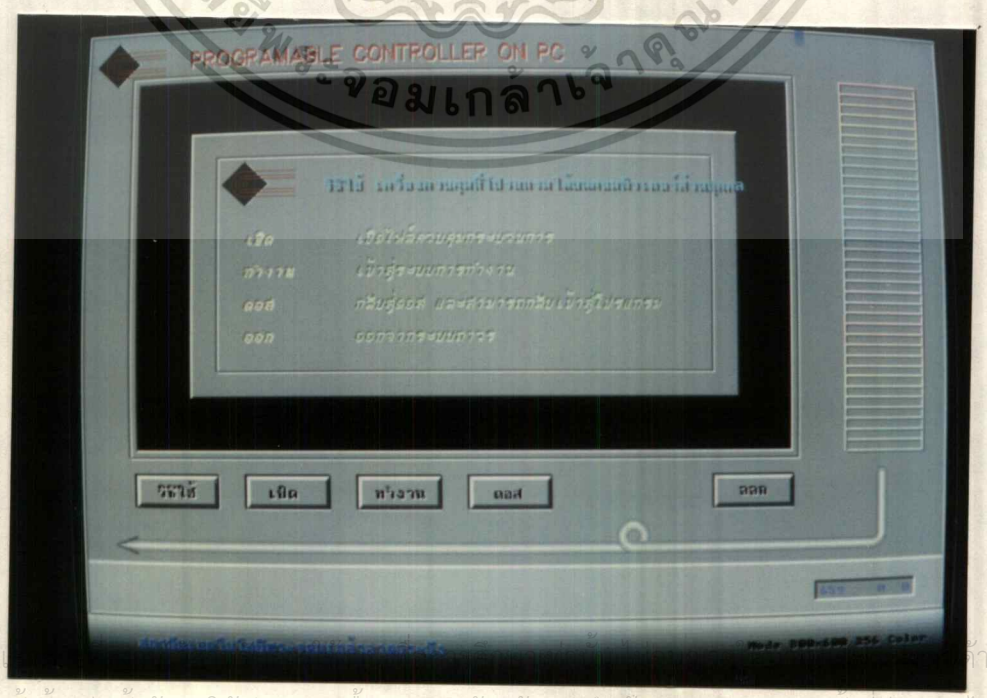
โปรแกรมนี้เป็นโปรแกรมการทำงาน 1 Loop เราสามารถทำเป็นหลาย Loop ได้และจะต้องจบด้วย END เสมอ การทำงานของโปรแกรมจะทำงานทีละบรรทัดและจะ plot กราฟได้ เมื่อมี END อยู่บรรทัดสุดท้าย และจะวนกลับไปทำงานใหม่เมื่อจบEND ทุกครั้ง

3) เมื่อเขียน program เรียบร้อยแล้วก็เข้าไปใน diaplay menu โดยใส่ชื่อไฟล์

C:\pro8

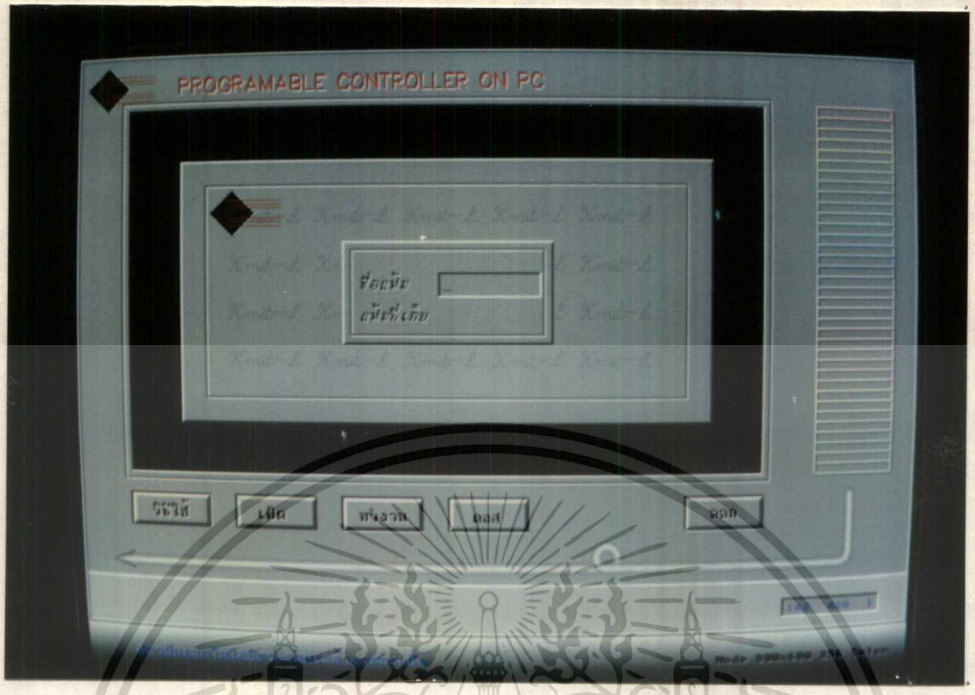
เมื่อใส่ชื่อโปรแกรม โปรแกรมก็จะขึ้นภาพTitle จะมีเสียงเพลงสถาบัน รายชื่อโครงงาน รายชื่อผู้ร่วมงาน รายชื่ออาจารย์ที่ปรึกษา ถ้าต้องการเข้า menu กด Enter

เมื่อกด Enter โปรแกรม ก็จะเข้าไปหน้า menu



เอกสารนี้เป็นลิขสิทธิ์ของ ห้ามมิให้ทำการคัดลอก ทำซ้ำ ดัดแปลง เผยแพร่ หรือกระทำการใดๆ อันเป็นการค้า
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4) เมื่อเข้าหน้าจอ menu แล้ว ก็ค้นหา File โดยใช้ mouse ไปกดที่ เปิด โปรแกรมจะขึ้นภาพดังรูป



เมื่อขึ้นรูปแล้ว ใส่ชื่อFileลงไป แล้ว Enter 2 ครั้ง

ถ้าใส่ชื่อ File ถูก โปรแกรมก็จะ Compile และขึ้น Success

ถ้าใส่ชื่อ File ไม่ถูก จะขึ้น File not Found และ Open Error

เช่น ใส่ชื่อ Text โดยไม่ต้องใช้จุดและนามสกุล

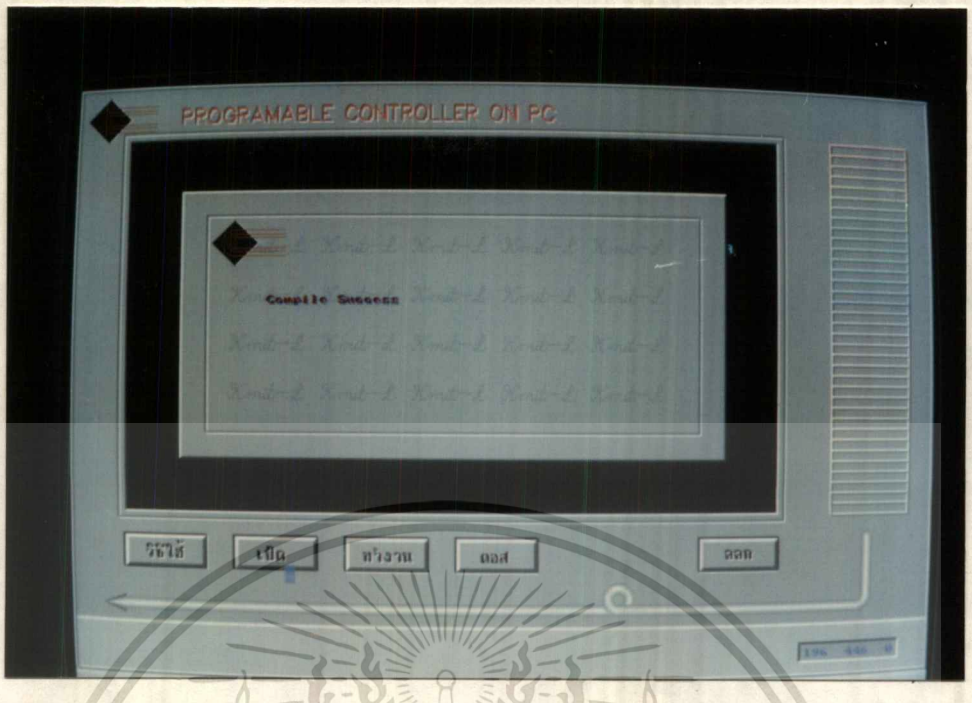
เมื่อใส่ชื่อ File และค้นหาแล้วมีโปรแกรมก็จะCompile โดยสร้างFileจุดHex

ขึ้นมาอีก ซึ่งใน FileจุดHex จะเป็นรหัสOpcodeของภาษาโปรแกรมที่เขียน

เช่น	LD	Y1		80	92	
	PID	S1	Text.	81	B2	Text.hex
	OUT	Y1		83	A2	
	END			89		

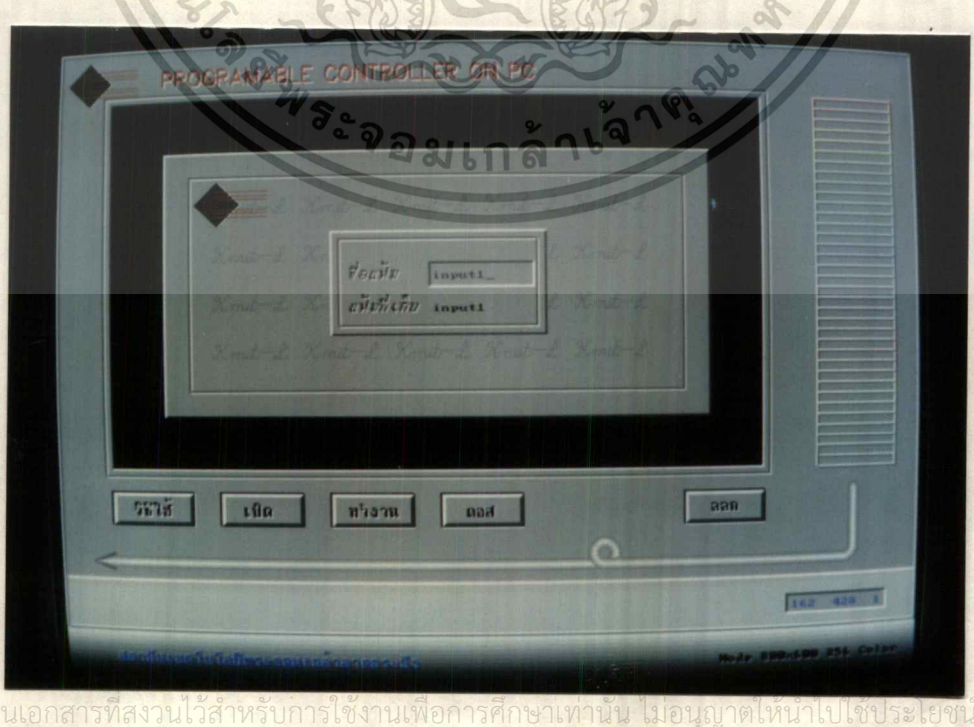
5) เมื่อใส่ชื่อFile เสร็จเรียบร้อยแล้ว ก็กดปุ่มทำงาน จะขึ้นภาพดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เริ่มการทำงานโดยมีฟังก์ชั่นต่างๆ ดังนี้

- หน้า MENU จะมี 5 ฟังก์ชั่น - เปิด คือ เปิดไฟล์ควบคุมกระบวนการ
  - ทำงาน คือ เข้าสู่ระบบการทำงาน
  - คอส คือ กลับสู่คอส และสามารถกลับเข้าสู่โปรแกรม
  - ออก คือ ออกจากระบบถาวร
  - วิธีใช้ คือ บอกวิธีใช้การทำงานของฟังก์ชั่นต่างๆ

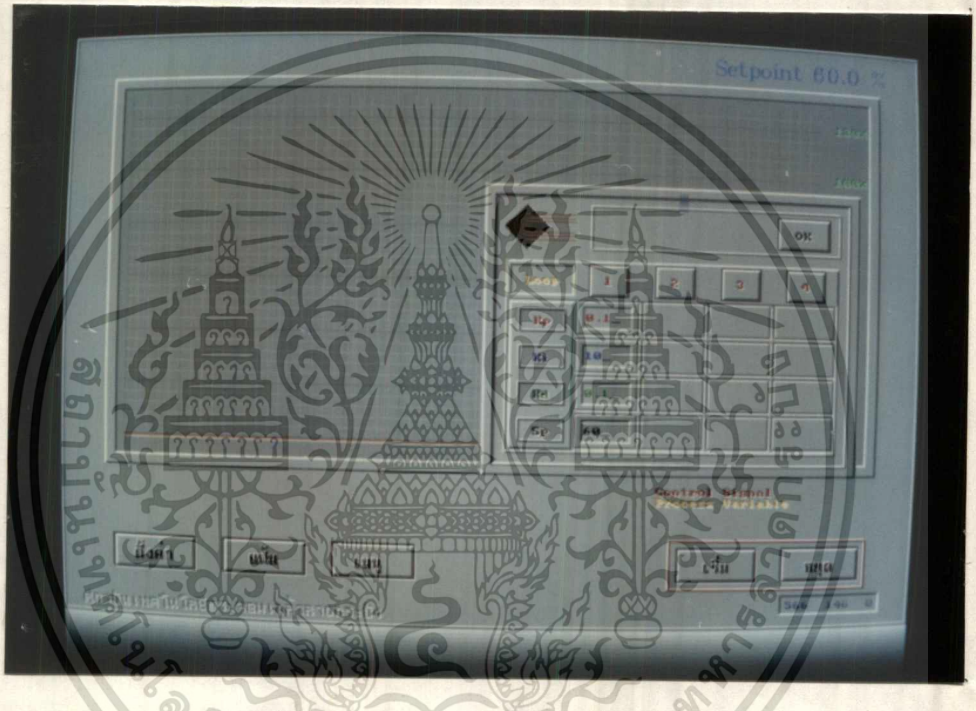


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- หน้า display การทำงานจะมี 5 ฟังก์ชั่น

  - ตั้งค่า คือ ตั้งค่าคงที่ต่างๆซึ่งมี4 Loopโดยตั้งค่า Kp,Ki,Kd และSp
  - แฟ้ม คือ ค้นหาแฟ้มและcompile ไปในตัว
  - เมนู คือ กลับไปหน้าเมนู
  - เริ่ม คือ เริ่มนำสัญญาณ PV เข้ามาและแสดงเป็นกราฟ
  - หยุด คือ หยุดการแสดงผลหน้าจอชั่วคราว

จากนั้นก็กดปุ่มตั้งค่า ก็จะขึ้นภาพดังรูป



ถ้าต้องการ Loop ไหนเป็น Loop หลักก็ใส่ Loop นั้น และ sp หลักก็จะอยู่ที่ Loop นั้น

เช่นใส่ Loop1

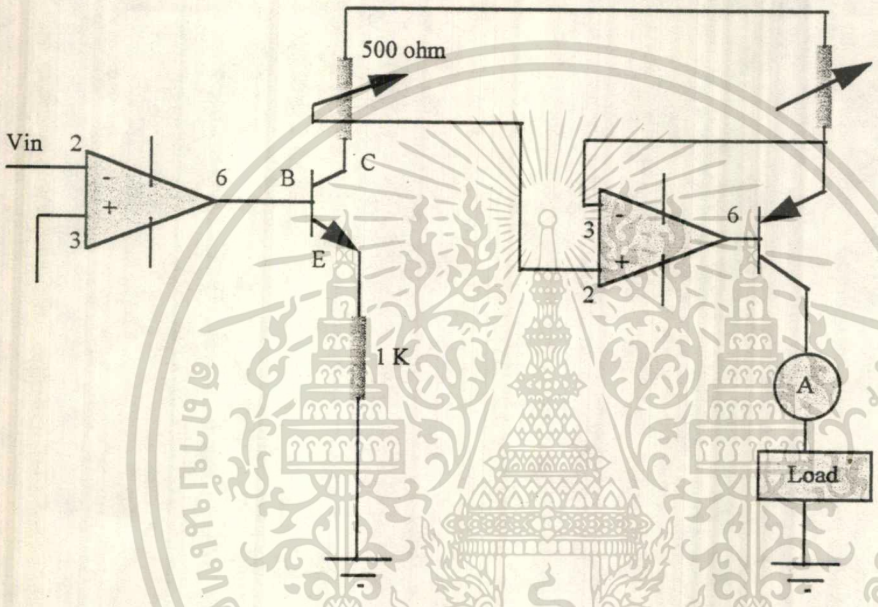
Kp	0.5
Ki	10
Kd	0.5
Sp	50

ค่า set point ก็จะขึ้นที่ Loop 1 เป็นหลัก ในกรณีที่ต่อแบบ Cascade ส่วนมากเราจะมี 2 Loop เราจะใช้ SP ของ Loop1 เป็นหลักและ output ของ Loop1 จะเป็น sv ของ Loop2

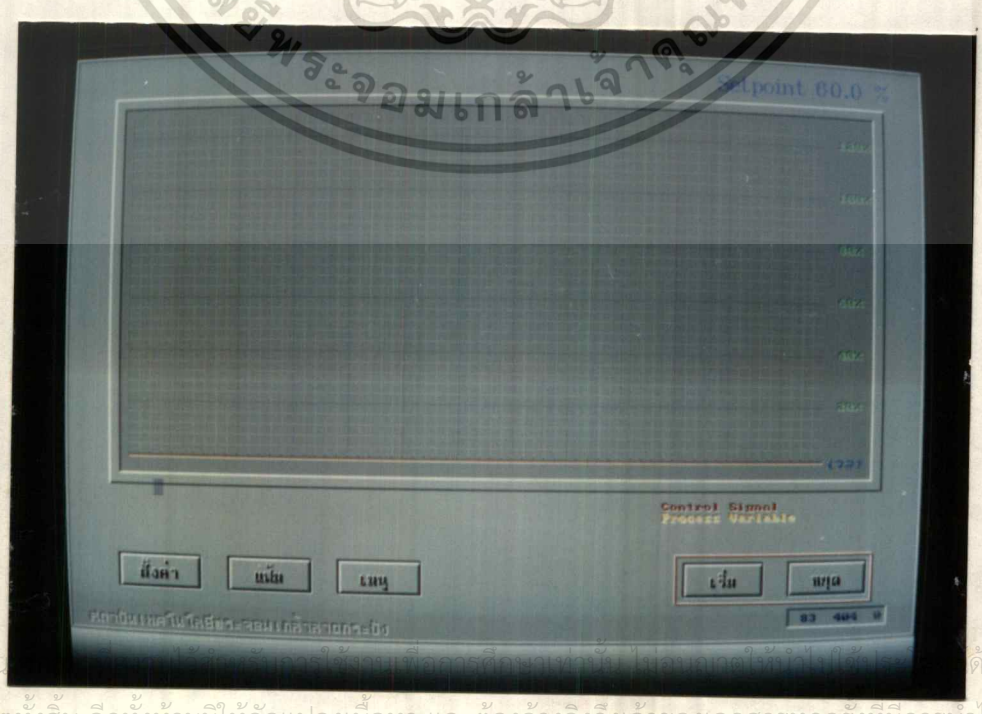
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และในกรณีที่เป็นอิสระต่อกัน เราสามารถเขียนSpได้แบบอิสระ เมื่อตั้งค่าเสร็จเรียบร้อยแล้ว กด **OK** และกดเริ่มทำงาน เครื่องคอมพิวเตอร์ก็จะทำงานส่งสัญญาณแรงดันผ่าน D/A และV/I ไปยังprocess และก็จะรับค่าเข้ามาผ่าน A/D เข้าคอมแล้วมาplotเป็นกราฟ ถ้าเราต้องการปรับค่า เราก็กดปุ่มหยุด โปรแกรมก็จะหยุดทำงานชั่วคราว แต่สัญญาณเข้ามาตลอด จากนั้นก็ไปกดปุ่มตั้งค่าอีกแล้วปรับค่าตามความต้องการแล้วค่อยเริ่มทำงานใหม่

วงจร V to I

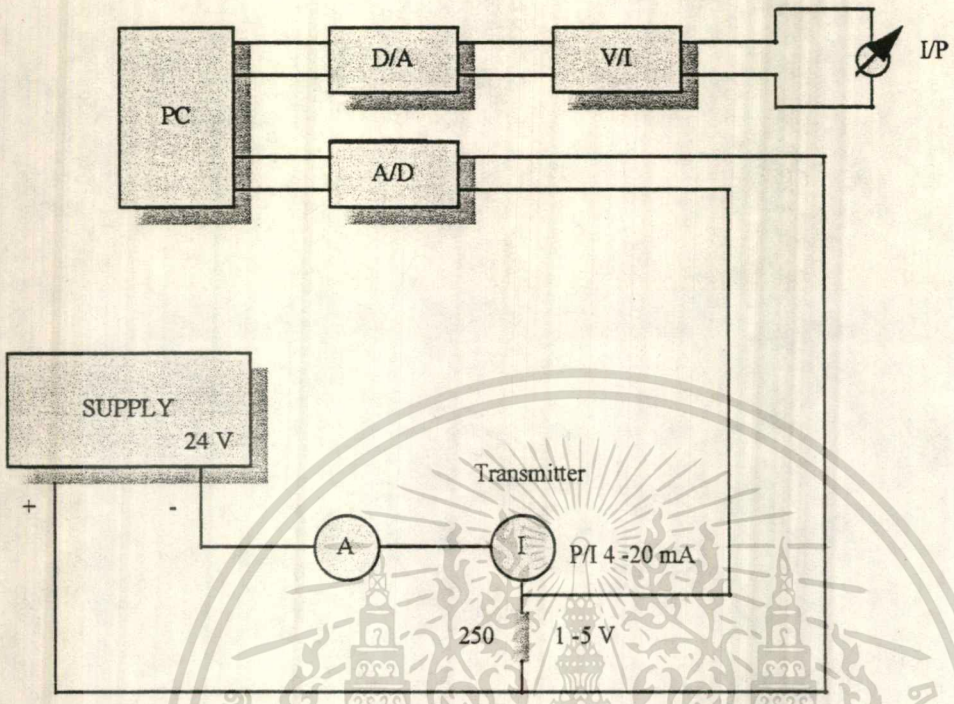


บรรยายการทำงาน



เอกสารนี้เป็น
ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้
ด้านการค้า

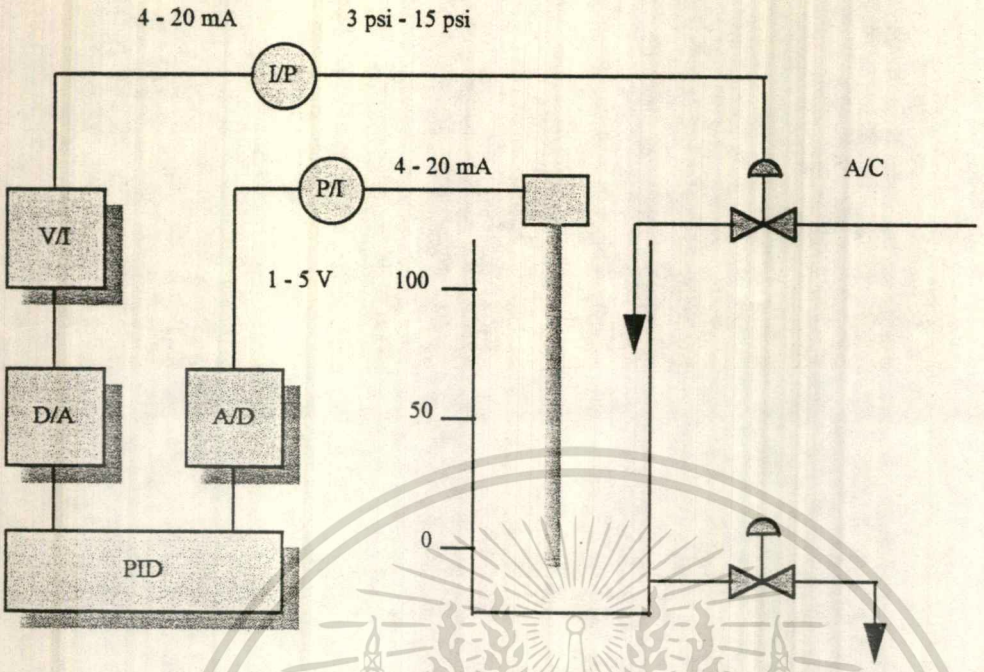
วงจร การต่อ process



Process ที่ใช้ในการทดลองควบคุม

ในส่วนของHardware ตามโครงงานที่ได้ตั้งไว้นั้นเป็นการนำการ์ดPCL-818 ไปใช้งานซึ่งก็คือต้องใช้การ์ด PCL-818 นี้ไปควบคุมงานได้จริงๆ ซึ่งทางกลุ่มพิจารณาแล้วเห็นว่าการcontrol level ในห้องทดลองprocess ซึ่งการทำงานแบบ 1 loop และสามารถcontrolแบบPID และมีตัวcheckคือตัวfeedbackจากอุปกรณ์มายัง computer เพื่อประมวลผลอีกด้วย

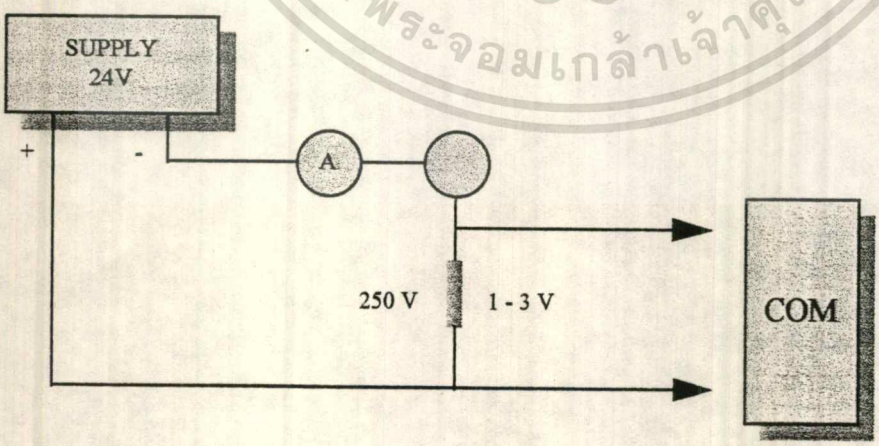
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



จากรูป แสดงถึงกระบวนการควบคุมของโครงงานนี้

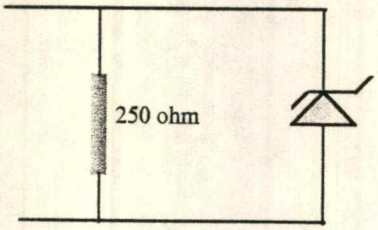
ลำดับขั้นกระบวนการ

เริ่มจากเปิดปั๊มนิวเมติกมาที่processเพื่อที่จะมา control valve แบบ A to C จากนั้นก็เปิดน้ำไหลเข้าสู่Tankน้ำ แล้วลองวัดสัญญาณoutputทาง Transmitter(P/I) ซึ่งจะออกมาเป็น 4-20MA แต่คอมพิวเตอร์จะรับเป็นแรงดัน ดังนั้นจึงใช้ความต้าน 250Ωต่ออนุกรมกับตัวTransmitterและดึงแรงดันไปเข้าคอมพิวเตอร์ดังรูป

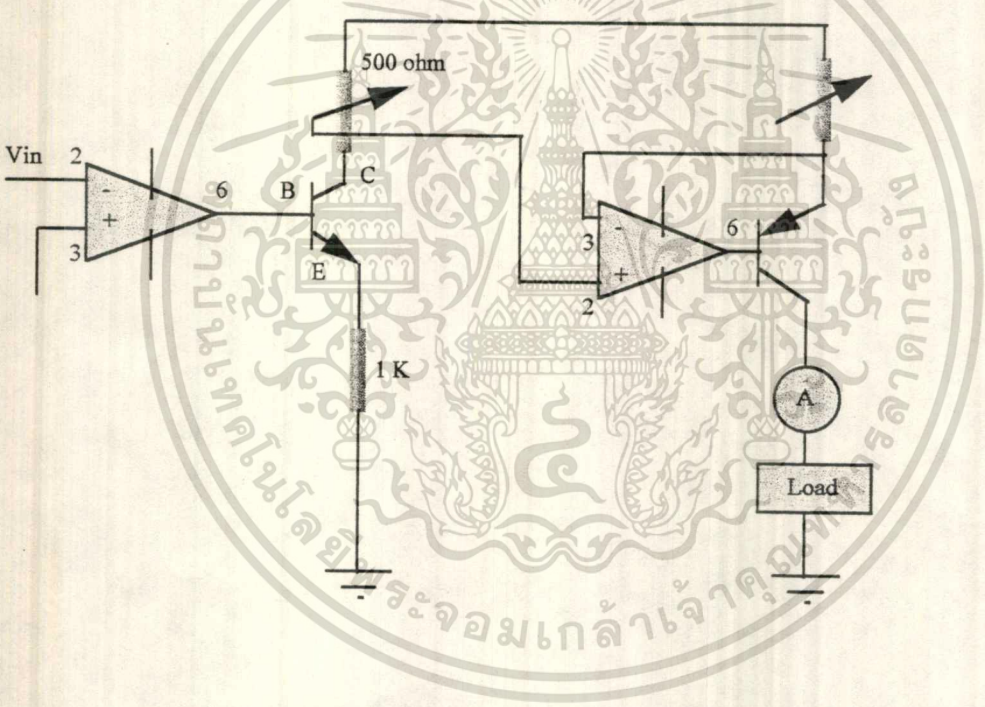


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อลองวัดแล้วระดับน้ำที่ 0% จะมีแรงดันออกมา 1V และที่ระดับน้ำ 50% จะมีแรงดันออกมา 3V และที่ระดับ 100% แรงดันจะเพิ่มขึ้น Card PCL-818 ไม่สามารถทนรับได้เพราะ Ch0 อยู่ในย่าน 0-5Vเท่านั้น ดังนั้นจึงใช้ Zener Diod 5V ต่อขนานกับความต้านทานดังรูป



-การควบคุม valve ตัว Transmitter เป็น I/P จะรับกระแส 4-20MA ดังนั้นจึงต้องใช้วงจรแปลงแรงดัน 1-5 V ให้เป็นกระแสดังรูป



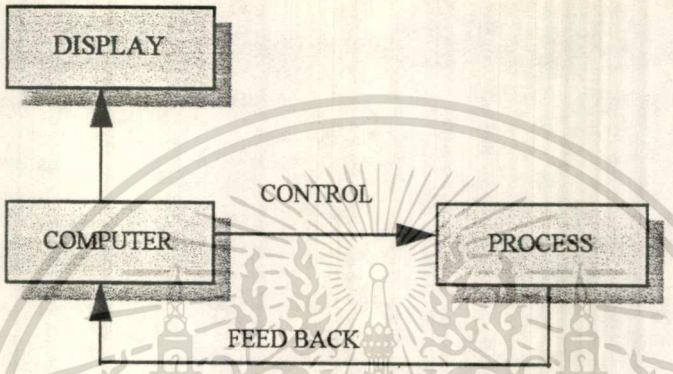
จากนั้นก็ลองให้แรงดันออกมาโดยโปรแกรมทดสอบ Card PCL-818 ให้แรงดันออก 1V outputที่วงจรต้องออก 4MA และ 5V outputต้องออก 20MA เมื่อได้แล้วก็นำไปต่อกับ Transmitter เพื่อลักษณะการควบคุมvalve 1V valveจะต้องไม่เคลื่อนที่เมื่อจ่ายกับ 3V valveจะต้องอยู่ระดับกึ่งกลางและเมื่อจ่าย 5V valveจะเคลื่อนลงมาสุด ถ้าเป็นเช่นนี้แล้วแสดงว่าสามารถ control valve สัมพันธ์กับ inputที่เข้าไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลำดับการควบคุม

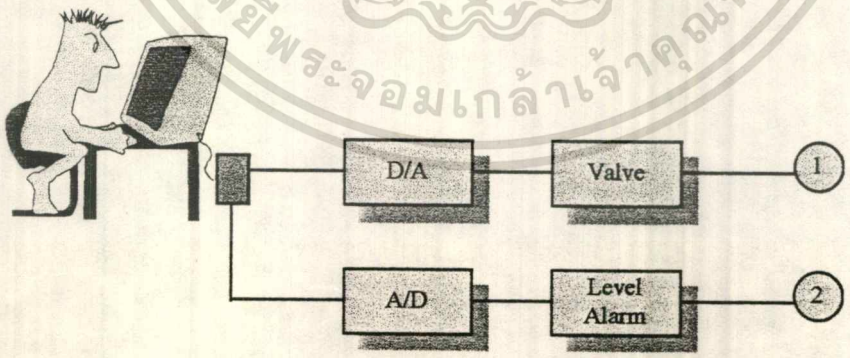
เช็คน้ำในระดับถังให้อยู่ในระดับ 50% ให้อัตราน้ำไหลเข้าเท่ากับไหลออก โดยจ่ายแรงดัน 3V คือให้valve อยู่ระดับกึ่งกลางและปล่อยน้ำออกจนระดับไม่ขยับจะอยู่ที่ 50% จากนั้นก็เริ่มตั้งค่าการควบคุม โดยการเขียนProgrammable ก่อนจากนั้นก็เข้าโปรแกรมเพื่อจะดูหน้าdisplay แล้วก็ตั้งค่า Kp,Ki,Kd และSp ที่ต้องการจะcontrol

ในส่วนของการใช้คอมพิวเตอร์ควบคุมการทำงานนั้นเขียน Block Diagram ได้ดังนี้ :



รูป แสดง Block Diagram โดยรวมของการควบคุม

คอมพิวเตอร์จะส่งสัญญาณควบคุมตามโปรแกรมที่ได้เขียนไว้มาที่อุปกรณ์ต่างๆ ในกระบวนการและมีผลFeedback จากอุปกรณ์นั้นเข้าไปในComputer อีกทีหนึ่งเพื่อวิเคราะห์และประมวลผลโดยการรับค่าและควบคุมจะทำผ่านทางการ์ด PCL-818



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. ในส่วนนี้ เป็นส่วนที่Computer จะส่งสัญญาณออกมาเป็นดิจิตอล 12bit แล้วมาเปลี่ยนเป็นสัญญาณ Analog เพื่อควบคุม Valve A to C โดยผ่าย Transmitter I/P ซึ่งจะเปิดและปิดตามระดับน้ำในTank เพื่อควบคุมให้ได้ระดับน้ำตามความต้องการ

2. ในส่วนนี้เป็นส่วนที่Computer จะรับสัญญาณ Analog จาก level Transmitter ที่ติดตั้งในถังเพื่อ check ว่าระดับน้ำถึงจุด setpointหรือยัง เพื่อนำไปวิเคราะห์ตามโปรแกรมที่ได้ตั้งไว้เพื่อจะได้นำไปดำเนินการในขั้นต่อไป

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

ในการจัดทำปริญญานิพนธ์ฉบับนี้สำเร็จลงได้โดย ได้รับคำแนะนำและความช่วยเหลือจาก อ.วิริยะ กองรัตน์ ซึ่งเป็นอาจารย์ที่ปรึกษารวมทั้งอาจารย์ประจำภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังและทุกๆ ท่านที่ให้ความสะดวกในด้านเครื่องมือ วัสดุอุปกรณ์ ตลอดจนคำแนะนำในด้านต่างๆ

ดังนั้น คณะผู้จัดทำโครงงาน เครื่องควบคุมที่โปรแกรมได้บนคอมพิวเตอร์ส่วนบุคคลนี้ จึงใคร่ขอขอบพระคุณทุกๆ ท่าน ที่มีส่วนช่วยให้ปริญญานิพนธ์นี้สำเร็จลุล่วงไปได้ด้วยดี

เกษมสันต์ เครือธร
สุทธิพร กิตติโชคดำรง
ภารดร จันทร์ฉวี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หนังสืออ้างอิง

ร.ศ. มัณฑนา ปราการสมุทร. การเขียนชุดคำสั่งภาษา C . บริษัท ดวงกมลสมัย จำกัด. มิถุนายน 2534

มนตรี พจนารถลาวัลย์. การเขียนโปรแกรมคอมพิวเตอร์ด้วยเทอร์โบซี. บริษัท ซีเอ็ดยูเคชั่น จำกัด

สมพัฒน์ รุ่งตะวันเรืองศรี. เรียนรู้คอมพิวเตอร์กราฟฟิกส์ 2 มิติ ด้วยภาษา C. บริษัท ซีเอ็ดยูเคชั่น จำกัด

ศิวัฒน์ ศิวะบวร. พรชัย จักรธำรงค์. จิรศักดิ์ ชัยวิริยะกุล. การประยุกต์ใช้งานภาษาซี. บริษัท ซีเอ็ดยูเคชั่น จำกัด

กิตติ ตีรเศรษฐ. การวิเคราะห์ระบบควบคุมเชิงเส้นเล่ม 1. สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. เมษายน 2535

กิตติ ตีรเศรษฐ. พื้นฐานวิศวกรรมระบบควบคุม. สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. มิถุนายน 2536

Carlos A.Smith. Armando B. Corripio. Principle and Practice of Automatic Process Control. John Wiley & Sons.

Curtis D. Johnson. Microprocessor-Based Process Control. Prentice-Hall, Inc.

Peter Hamann. Steve Willings: Mini H-Type Programmable Controller Operation Manual. Omron. 1990.

Omron Tateisi Electronics Co. Omron User's Manual PID Unit. Omron. 1990.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Program MONITOR.C

```
#include <graphics.h>
#include <time.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <ctype.h>
#include <dos.h>
#include <math.h>
#include <stdarg.h>
#include <process.h>
#include <string.h>
#include <io.h>

#include "d: \borlandc\data1\kmit.h"
#include "d: \borlandc\data11\plant40.h"
#include "d: \borlandc\data1\mutifont.h"

#define  LD 0x80
#define  PID 0x81
#define  CAS 0x82
#define  OUT 0x83
#define  ADD 0x84
#define  SUB  0x85
#define  MUL 0x86
#define  DIV 0x87
#define  SQR 0x88
#define  END 0x89
#define  X1 0x90
#define  X2  0x91
#define  X3  0x92
#define  X4  0x93
#define  Y1  0xA0
#define  Y2  0xA1
#define  Y3  0xA2
#define  Y4  0xA3
#define  S1  0xB0
#define  S2  0xB1
#define  S3  0xB2
#define  S4  0xB3

#define  space_line 15
#define  port  0x300

int  ADC(int ch);
void DAC(int ch,int data);
void scan_adc(void);

int check_oprnd(void);
void init_data(void);
void init_pid(void);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void pid(float sp,int ch);
void run_process(void);

unsigned char thaidata[5120];
extern int far _Cdecl Svga256_fdriver[ ] ;

//-------------------------------------------------------------

int  gscanfxy(int xloc,int yloc,char  *fmt,. . .) ; // scanf in Graphics mode
int  gprintf(char  *fmt, . . .) ;
int  ggetche(void) ;
int  gputch(int c) ;  //  Put a character and echoes it to the screen
char  *ggets(char *buffer) ;  // Allows user to enter a string while in graphics
int  gprintfxy(int xloc, int yloc, char *fmt, . . .) ;
int  gscanf(char  *fmt, . . . ) ;

int huge detectEGA(void) ;
void  checkerrors(void) ;
void  svga_mode(void) ;          //     User SuperVGA Mode
void  infor_mouse(void) ;        //     Mouse Information Return X Y B At mx my mb
void  get_mouse_position(int colors) ;
void  check_mouse_process(int colors) ;  //  Check Mouse in process

void  open_file(void) ;
void  compile(void) ;
int  lexical(int  *num_code) ;
int  get_token(FILE *stream,char *token) ;

void loadthai0 (void) ;          //   Serief Narrow Text
void loadthai1 (void) ;          //   Normal Narrow Text
void loadthai2 (void) ;          //   Digital Text
void loadthai3 (void) ;          //   Bold Normal Text
void loadthai4 (void) ;          //   Normal Text
void outthai (int x, int y,int size, unsigned char *_text) ;

void logo_kmit (void) ;
void menu_first(void) ;
void plant_graph (void) ;
void mousr_of_setvalue(void) ;
void mouse_of_graph (void) ;
void load_config(void) ;         //   Load PCI Config at pci.sys files
void outtextxy3d (int x,int y,int color, char *text) ;
void nouttextxy3d (int x,int y,int color, char *text) ;

void start_process_graph (void) ;
void scan_analog_graph (void) ;

//                                  Main System of Process
void white_data(char *save_files,float a2,int mode);

int mx=0, my=0, mb-0 ;           //   Data Of Position  X,Y,B Of Mouse
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
FILE *data_fb, *fp;
int h_,m_,s_=0;
int set_maxchars = 12;          //  Set Maximum Of Char Of Sub_*ggetch( )
float scale_x ;                 //  scale mouse give max x
float scale_y ;                 //  scale mouse give max y
int SETFILES=0;                 //  Set SETFILES=1      is  Save Files
int  SETPRINTER=0 ;

char  SAVE_FILES[20] = "DEFAULT.GRP" ;
char  CODE_FILES[20] = "input1";      // Save and Load Files of The Plant First
char  HEX_FILES[20] = "input3.hex";
int  key_buff=0;                      // Buffer Of Keyboard
int  set_sound=0;                     // Sound Off = 1
int x_cor=0;

struct time ts;

float foperand,AC;
static float X[3], Y[3], //X[ch] is put data from A to D convert
                  //Y[ch]  is output to D to A convert
static float Xo[3], Yo[3];
int flag=0;  /* flag check is X[n] or Y[n] or S[n]*/
static float  Kp[3],Ki[3],Kd[3],Sp[3],Ts;
static float  PV[3],MV[3],er1[3],er2[3];
static float  K1[3],K2[3],K3[3];

//-------------------------------------------------------------

void main( )
{
        load_config( ) ;
        scan_adc( ) ;
        init_data ( ) ;
//        init_pid( ) ;
        svga_mode( ) ;
        logo_kmit( ) ;
        clear_logo( ) ;
        menu_first ( ) ;
        closegraph( ) ;
}

//-------------------------------------------------------------


void scan_adc(void)
{
        int ch;
        float data;
        for(ch=0;ch<=0;ch++)
        {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        data=(5*float)ADC(ch))/4095-1;
        if(data<0) data=0;
        X[ch]=data;
        }
}

int ADC(int ch)
}
        int status, dl, dh;
        outportb(port+2,ch*16+ch);   /* SET SCAN CHANNEL VALUE */
        outportb(port+1,5) ;
        outportb(port,ch);                /* trigger */
        do{
                status=inportb(port+8);
        }while((status & 0x70) !=0x70) ;
        delay(1);
        dl = inportb(port) ;
        dh =  inportb(port+1) ;
        dl = dl/16;
        return  dh*16+dl;
}

void  DAC(int ch, int data)
{
        if(!ch)
        {
                outportb(port+4,data*16) ;
                outportb(port+5,data/16) ;
        }
        else
        {
                outportb(port+6,data*16) ;
                outportb(port+7,data/16) ;
        }
}

void plant_graph(void)
{
  int  i,c,d,e,f;
  char buffer[10] ;
  void *picture, *text ;
        black_g( ) ;
        kmit_box(40,40,760,410,3,1); kmit_box(50,50,750,400,3,2) ;
        setfillstyle(1,16+10);setcolor(16+10);bar(50+3,50+3,750-3,400-3);

loadthai0( ) ;
setcolor(16+11);outthai(705-1,380-1,1,"      ") ;
setcolor(16+8);outthai(705+1,380+1,1,"      ") ;
setcolor(47+8) ;outthai(705,380,1,"         ") ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        nouttextxy3d(550,420,RED,"Control Signal" );
        nouttextxy3d(550,430,YELLOW,"Process Variable");

       for(i=60;i<700;i+=10){
putpixel(i+1,380+1,32-1);
putpixel(i-1,380-1,DARKGRAY);
putpixel(i,380, (16*4)+1);
     }
       for(i=60;i<710;i+=50)
putpixel(i,380,32+7) ;

// ..........Scale line ..........
        setcolor(27);
       for(i=60;i<700;i+=10)          // Vertical Axis
line(i,60,i,377);
        setcolor(25);
       for(i=60;i<710;i+=50)
line(i,60,i,376) ;
        setcolor(27);
       for(i=380;i>60;i-=10)          // Horizontal Axis
line(60,i,700,i);
        setcolor(25);
       for(i=380;i>60;i-=50)          // Horizontal Axis Active

line(60,i,700,i);

//-------------------------------------------------------------
        setcolor(72);
        c=1;
       for(i=380;i>60;i-=50){         // Write Parameter 0% - 100%
        if(c<7){
sprintf(buffer,"%d%",20*(c++)) ;
setcolor(28);
     outtextxy(715+1, (i-5) -50+1,buffer) ;
setcolor(DARKGRAY) ;
     outtextxy(715+1, (i-5) -50-1,buffer);
setcolor(72);
     outtextxy(715,(i-5)-50, buffer) ;
        }
      }
setcolor(32-3);line(60+1,382+1,700+1,382+1) ;
setcolor(DARKGRAY);line(60-1,382-1,700-1,382-1) ;
setcolor((16*4)+1);line(60,382,700,382) ;
//-------------------------------------------------------------
//---------- Same Of The Sub Program Off Graph ----------
//------                PLANT  SIMULATION                -----
//-------------------------------------------------------------

                kmit_box(670,517,770,537,3,2) ;
                        kmit_rec(50,475,50+75,475+30,0,0) ;
                        kmit_box(50,475,50+75,475+30,3,1) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        setcolor(BLACK);RECTANGLE(50,475,50+75,475+30),
                picture=malloc(imagesize(49,419,125,451)) ;
                        putimage(150-1,474-1,picture,COPY_PUT);  // button set va
                        putimage(250-1,474-1,picture,COPY_PUT);  // button savefi
                        putimage(570-1,474-1,picture,COPY_PUT);  // button Start
                        putimage(670-1,474-1,picture,COPY_PUT);  // button Stop
                    kmit_rec(560+3,465+3,670+75+10-3,475+30+10-3,1,LIGHTRED) ;
                free(picture) ;
                  loadthai3( ) ;
                      setcolor(DARKGRAY) ;
     outthai(70+1,490+1,1,"          ") ;
     outthai(175+1,490+1,1,"         ") ;
     outthai(270+1,490+1,1,"         ") ;
     outthai(595+1,490+1,1,"         ") ;
     outthai(695+1,490+1,1,"         ") ;
         setcolor(WHITE) ;
     outthai(70-1,490-1,1,"          ") ;
     outthai(175-1,490-1,1,"         ") ;
      outthai(270+1,490-1,1,"        ") ;
      outthai(595+1,490-1,1,"        ") ;
      outthai(695+1,490-1,1,"        ") ;
         setcolor(BLACK) ;
      outthai(70,490,1,"          ") ;
      outthai(175,490,1,"         ") ;
      outthai(270,490,1,"         ") ;
      outthai(595,490,1,"         ") ;
      outthai(695,490,1,"         ") ;
     loadthai2( );
         setcolor(DARKGRAY) ;
       outthai(20-1,535-1,1,"                                          ") ;
         setcolor(WHITE) ;
       outthai(20+1,535+1,1,"                                          ") ;
         setcolor(LIGHTGRAY) ;
      outthai(20,535,1,"                                             ") ;
   paint_box_may(665,10,755,35,7) ;
   settextstyle(1,  HORIZ_DIR,2) ;
   setcolor(LIGHTBLUE);gprintfxy(600,10,"Setpoint %2.1f %",Sp[0]*20) ;
   settextstyle(0,  HORIZ_DIR,1) ;
         mouse_of_graph( ) ;
}

//--------------------- Check Mouse Informationa For Plant_Graph -----
void mouse_of_graph(voide)
{
int  i,y;
void *text,*picture;

// ------------------------- Begin The Mouse Loop ----------------

    get_mouse_position(104) ;    // Return To "mx my mb"
    mb=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        if(  (my>475)  &&  (my <  (475+30))  )   //  Position Y-Axis
   {
       if(mx>  50  &&  mx <  120)     //   setvalue  Button
  {
   sound_mouse( ) ;
   key_buff=0 ;
    text=malloc(imagesize(40,475,135,515)) ;
        getimage(50+4,475+4,125-5,515-15,text) ;
    picture=malloc(imagesize(40,475,135,515)) ;
        getimage(50-3,475-3,125+3,515+3,picture) ;

      kmit+box(50,475,125,505,3,2) ;
   putimage(50+4,475+4,text,COPY_PUT) ;
     delay(150) ;
   putimage(50-3,475-3,picture,COPY_PUT) ;
     free(picture); free(text) ;

/*  Image of Setvalue  */
 picture=malloc(imagesize(475,240,650,310)) ;
 getimage(475,250,650,310,picture) ;
     kmit_box(385,140,740,410,3,1,) ;
     kmit_36(385+40,140+40) ;
     kmit_rec(385+100,140+20,740-20,140+60,1,LIGHTGRAY) ;
     setcolor(BLACK) ;
     rectangle(385+10,140+10,740-10,410-10) ;
     kmit_box(385+20,140+70,740-20,410-20,3,1) ;
     kmit_box(385+20+2,210+36,740-20-2-,410-20-2,3,2) ;
     kmit_box(385+24,210+36+2,385+24+63-2,410-20-4,3,1) ;
     kmit_box(385+24,210+4,385+24+63-2,140+70+36-2,3,2) ;
     for(y=0;y<=3;y++)
     for(i=0;i<=3;i++){
              kmit_rec(468+4+62*i,246+35*y+2,468+62+62*i,244+35+35*y,2,LIGHTGRAY);
     }
     for (i=o;i<=3;i++) {
          kmit_box((478+62*i)+5,210+4,(468+62-10+62*i),210+36-2,3,1);//dray button
          setcolor(BLACK);
          rectangle((478_62*i) +5,210+4,(468+62-10+62*i),210+36-2);

          kmit_rec(385+24+10,(252+34*i)+4,385+24+62-10,252+24+34*i,1,DARKGRAY);
          }
          nouttextxy3d(478+20,210+16,red,"1");
          nouttextxy3d(478+20+62,210+16,RED, "2");
          nouttextxy3d(478+20+62*2,210+16,RED,"3");
          nouttextxy3d(478+20+62*3,210+16,RED,"4");

          nouttextxy3d(478+13,264,RED,"Kp");
          nouttextxy3d(478+13,264+34,BLUE,"Ki");
          nouttextxy3d(478+13,264+34*2,GREEN,"Kd");
          nouttextxy3d(478+13,264+67*3,BLACK,"Sp");

          nouttextxy3d(478+4,210+17,YELLOW,"Loop");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//draw ok button
          kmit_box(660,165,710,195,3,1);
          setcolor(BLACK);
          rectangle(660,165,710,195);
          nouttextxy3d(677,178,BLACK,"OK");
     free(picture) ;
     tuy_sound2 ( );
     mouse_of_setvalue( );

//                                   END OF Get Setpoint
 }
   else  if(mx> 150 && mx < 220)      //   File
 {
 sound_mouse( ) ;
     text=malloc(imagesize(40,475,135,515));
          getimage(150+4,475+4,225-5,515-15,text);
     picture=malloc(imagesize(40,475,135,515));
          getimage(150-3,475-3,225+3,515+3,picture);

        kmit_box(150,475,225,505,3,2);
 putimage(150+4,475+4,text,COPY_PUT);
      delay(150);
 putimage(150-3,475-3,picture,COPY_PUT);
      free(picture); free(text);
      picture=malloc(imagesize(0,0,200,100));
          getimage(100,350,300,450,picture);
 for(i=0;i<40;i+=1)
       {kmit_box(140-1,390-i,260+i,410+i,3,1);sound(1000+(100*i));}
        nosound( );
          kmit_rec(100+10,350+10,300-10,450-10,0,LIGHTGRAY);
//       Get Save Files
      loadthai0( );
setcolor(DARKGRAY);outthai(120-1,390-1,1,"                    ");
setcolor(WHITE);outthai(120+1,420+1,1,"                    ");
setcolor(241);outthai(120,390,1,"                ");

setcolor(DARKGRAY);outthai(120-1,420-1,1,"                      ");
setcolor(WHITE);outthai(120+1,420+1,1,"                    ");
setcolor(241);outthai(120,420,1,"                ");
 set_maxchars = 11;
       setcolor(131); gscanfxy(200,390,"%s",&SAVE_FILES);putch(007);
       setcolor(216); gprintfxy(200,420,"%s",SAVE_FILES);
       fetch( );
       delay(300);
         for(i=o;i<20;i+=2)
         {
          putimage(100,350,picture,COPY_PUT);
          kmit_box(100+i,350+i,300-i,450-i,3,1);
          sound(300-(200*i));
          nosound( );
          delay(50);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        }
   putimage(100,350,picture,COPY_PUT);free(picture);

        mouse_of_graph( );              //  End Of SAVE_FILES
  }
     else if(mx> 250  &&  mx < 320)     //  MENU
  {
  sound_mouse( );
     text=malloc(imagesize(40,475,135,515)) ;
        getimage(250+4,475+4,325-5,515-15,text) ;
     picture=malloc(imagesize(40,475,135,515));
        getimage(250-3,475-3,325+3,515+3,picture);

       kmit_box(250,475,325,505,3,2);
   putimage(250+4,475+4,text, COPY_PUT);
       delay(150);
   putimage(250-3,475-3,picture,COPY_PUT);
       free(picture); free(text);
          tuy_sound1( ) ;
   setfillstyple(1,BLACK); setcolor(BLACK);
        for(i=0;i<320;i++){
bar(0,0+i,800,10+i);
bar(0,590-i,800,600-i);
        }
        tuy_sound2( );
     menu_first( );
   }
     else if(mx> 570 && mx < 645)           //   Play Start
   {
   sound_mouse( );
       text=malloc(imagesize(40,475,135,515));
         getimage(570+4,475+4,645-5,515-15,text);
       picture=malloc(imagesize(40,475,135,515));
         getimage(570-3,475-3,645+3,515+3,picture);

        kmit_box(570,475,645,505,3,2);
     putimage(570+4,475+4,text,COPY_PUT);
        delay(150);
     putimage(570-3,475-3,picture,COPY-PUT);
       free(picture); free(text);
       run_process( );
    }
    else{ mouse_of_graph( ) ;  }
      }          //              End Any Check Option
      else{   no_mouse( );delay(245);mouse_of_graph ( ); }
  }
  //----------------------------

  void mouse_of_setvalue(void)
  {
  int i;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void *text,*picture;
     get_mouse_position(104);
     mb=0;
    if(   (my>214)   && (my   < 244))   //   Position Y-Axis
    {
    if(nx> 483  &&  mx < 520)              //  loop1   Button
    {
//     ch=0;
    sound_mouse( );
    key_buff=0;
         text=malloc(imagesize(483,214,520,244));
            getimage(483+4,214+4,520-4,244-4,text);
         picture=malloc(imagesize(483-5,214-5,520+5,244+5));
            getimage(483-3,214-3,520+3,244+3,picture);

           kmit_box(483,214,520,244,3,2);
    putimage(483+4,214+4,text,COPY_PUT);
         delay(150);
    putimage(483-3,214-3,picture,COPY_PUT);
         free(picture); free(text);
            delay(300);tuy_sound1( );
            kmit_box(477,252,525,275,3,2);
             set_maxchars=4;
             setcolor(RED);
            gscanfxy(477+4,260,"%f",&Kp[0]);
            delay(300);tuy_sound1( );
              kmit_box(477,252+35,525,275+35,3,2);
            setcolor(BLUE);
            gscanfxy(477+4,260+35,"%f",&Ki[0]);
            delay(300);tuy_sound1( );
              kmit_box(477,252+35*2,525,275+35*2,3,2);
            setcolor(GREEN);
            gscanfxy(477+4,260+35*2,"%f",&Kd[0]);
            delay(300);tuy_sound1( );
              kmit_box(477,252+35*3,525,275+35*3,3,2);
            setcolor(BLACK);
            gscanfxy(477+4,260+35*3,"%f",&Sp[0]);
            Sp[0]=Sp[0]/20;
    mouse_of_setvalue ) ;
    }else
     if(mx> 483+62 && mx < 520+62)          //   loop2   Button
   {
 //  ch=1;
   sound_mouse( );
   key_buff=0;
     text=malloc(imagesize(483+62,214,520+62,244));
         getimage(483+62+4,214+4,520+62-4,244-4,text);
    picture=malloc(imagesize(483+62-5,214-5,520+62+5,244+5));
         getimage(483+62-3,214-3,520+62+3,244+3,picture);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        kmit_box(483+62,214,520+62,244,3,2);
    putimage(483+62+4,214+4,text,COPY_PUT);
      delay(150);
    putimage(483+62-3,214-3,picture, COPY_PUT);
      free(picture); free(text);
                  delay(300);tuy_sound1( );
          kmit_box(477+62,252,525+62,275,3,2);
          set_maxchars=4;
          setcolor(RED);
          gscanfxy(477+4+62,260,"%f",&Kp[1]);
          delay(300);tuy_sound1( );
            kmit_box(477+62,252+35,525+62,275+35,3,2);
          setcolor(BLUE);
          gscanfxy(477+4+62,260+35,"%f",&Ki[1]);
          delay(300);tuy_sound1( );
            kmit_box(477+62,252+35*2,525+62,275+35*2,3,2);
          setcolor(GREEN);
          gscanfxy(477+4+62,260+35*2,"%f",&Kd[1]);
          delay(300);tuy_sound1( );
            kmit_box(477+62,252+35*3,525+62,275+35*3,3,2);
          setcolor(BLACK);
          gxcanfxy(477+4+62,260+35*3,"%f",&Sp[1]);
          sp[1]=Sp[1]/20;
      mouse_of_setvalue( );
   }else
        if(mx> 483+62*2 && mx < 520+62*2)              //  loop3  Button
   {
  //  ch=2
    sound_mouse( ) ;
    key_buff=0;
      text=malloc(imagesize(483+62*2,214,520+62*2,244));
          getimage(483+62*2+4,214+4,520+62*2-4,244-4,text);
      picture=malloc(imagesize(483+62*2-5,214-5,520+62*2+5,244+5));
          getimage(483+62*2-3,214-3,520+62*2+3,244+3,picture);

        kmit_box(483+62*2,214,520+62*2,244,3,2);
  putimage(483+62*2+4,214+4,text,COPY_PUT) ;
      delay(150) ;
  putimage(483+62*2-3,214-3,picture,COPY_PUT) ;
      free(picture) ; free(text) ;
          delay(300) ; tuy_sound1 ( ) ;
        kmit_box(477+62*2,252,525+62*2,275,3,2) ;
        set_maxchars=4 ;
        setcolor(RED) ;
        gscanfxy(477+62*2+4, 260,"%f",&Kp[2]) ;
        delay(300) ; tuy_sound1( ) ;
          kmit_box(477+62*2, 252+35, 525+62*2,275+35,3,2) ;
        setcolor(BLUE) ;
        gscanfxy(477+62*2+4, 260+35,"%f",&KI[2]) ;
        delay(300) ; tuy_sound1( ) ;
          kmit_box(477+62*2, 252+35*2, 525+62*2,275+35*2,3,2) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        setcolor(GREEN) ;
        gscanfxy(477+4+62*2, 260+35*2, "%f", &Kd[2]) ;
        delay(300) ; tuy_sound1( ) ;
          kmit_box(477+62*2, 252+35*3, 525+62*2, 275+35*3,3,2) '
        setcolor(BLACK) ;
        gscanfxy(477+4+62*2, 260+35*3, "%f", &Sp[2]) ;
        Sp[2]=Sp[2]/20;
      mouse_of_setvalue( ) ;
    }else

      if(mx> 483+62*3 && mx < 520+62*3)          //     loop4  Button
    {
   //  ch=3;
     sound_mouse ( );
     key_buff=0 ;
        text=malloc(imagesize (483+62*3,214,520+62*3,244) ) ;
            getimage(483+62*3+4, 214+4, 520+62*3-4, 244-4, text) ;
        picture=malloc(imagesize (483+62*3-5,214-5,520+62*3+5,244+5) ) ;
            getimage(483+62*3-3, 214-3, 520+62*3+3, 244+3, picture) ;

          kmit_box(483+62*3, 214, 520+62*3, 244, 3, 2) ;
     putimage(483+62*3+4, 214+4, text, COPY_PUT) ;
        delay(150) ;
     putimage(483+62*3-3, 214-3, picture, COPY_PUT) ;
        free(picture) ; free(text) ;
                delay(300); tuy_sound1( ) ;
            kmit_box(477+62*3, 252, 525+62*3, 275, 3, 2) ;
            set_maxchars=4 ;
            setcolor(RED) ;
            gscanfxy(477+4+62*3, 260, "&f", &Kp[3]) ;
            delay(300); tuy_sound1 ( ) ;
              kmit_box(477+62*3, 252+35, 525+62*3, 275+35, 3, 2) ;
            setcolor(BLUE) ;
            gscanfxy(477+4+62*3, 260+35, "&f", &Ki[3]) ;
            delay(300); tuy_sound1 ( ) ;
              kmit_box(477+62*3, 252+35*2, 525+62*3, 275+35*2, 3, 2) ;
            setcolor(GREEN) ;
            gscanfxy(477+4+62*3, 260+35*2, "&f", &Kd[3]) ;
            delay(300); tuy_sound1 ( ) ;
              kmit_box(477+62*3, 252+35*3, 525+62*3, 275+35*3, 3, 2) ;
            setcolor(BLACK) ;
            gscanfxy(477+4+62*3, 260+35*3, "&f", &Sp[3]) ;
            Sp[3]=Sp[3]/20 ;
      mouse_of_setvalue ( ) ;
      }else   mouse_of_setvalue ( ) ;
    }else
      if ( (my > 165) && (my < 195) )        // select  OK button
      {    if ( (mx>660) && (mx<710) )
           {
              sound_mouse ( ) ;
              key_buff=0 ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            text=malloc(imagesize(660,165,710,195) ) ;
            getimage(660+4, 165+4, 710-4, 195-4, text) ;
            picture=malloc(imagesize (660-5, 165-5, 710+5, 195+5) ) ;
            getimage(660-3, 165-3, 710+3, 195+3, picture) ;

            kmit_box(660, 165, 710, 195, 3, 2) ;
            putimage(660+4, 165+4, text, COPY_PUT) ;
            delay(150) ;
            putimage(660-3, 165-3, picture, COPY_PUT) ;
            free(picture) ; free(text) ;
            x_cor=0 ;
            plant_graph ( ) ;
            exit (0) ;
      } else mouse_of_setvalue ( ) ;
    }   else
        {
              no_mouse ( ) ; delay(245) ;
              mouse_of_setvalue ( ) ;
        }
}

//-------------------- Check Mouse Information For Plant_Graph -------

void run_process(void)
{
    int  obcode, bufcode, ch=0 ;
    int  i ;
    long  oldpos ;
    float setp;
    if ( (fp=fopen (HEX_FILES, "r" ) ) ==NULL)
    {
        outtetxy3d(100,200,18,"Cannot Open File") ;
        exit (1) ;
     }
     for ( ; ; ) {
        scan_adc( ) ;
        while(fscanf(fp, "%02x", &obcode) !=EOF)
        {
        switch(obcode) {
           case LD    : ch=check_oprnd( ) ;
                        if (! flag)  AC=X[ch] ;
                        else AC=Y[ch] ;
                        break;
           case PID:    ch=check_oprnd( ) ;
                        if(flag==1)
                           setp=Y[ch] ;
                        else if(flag==2)
                                 setp=Sp[ch] ;
                             else break ;
                        oldpos=ftell (fp) ;
                        fseek(fp,oldpose-13, SEEK_SET) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                        fscanf(fp, "%x", &bufcode) ;
                        if(bufcode==LD)
                           ch=check_oprnd( ) ;
                        else{
                           fseek(fp, oldpos-10, SEEK_SET) ;
                           fscanf(fp, "%x", &bufcode) ;
                           if(bufcode==SQR) {
                                fseek(fp, oldpos-15, SEEK_SET) ;
                                ch=check_oprnd( ) ;
                           }else{
                                    fseek(fp, oldpos-23, SEEK_SET) ;
                                    ch=check_oprnd( ) ;
                                }
                           }
                        pid(setp, ch) ;
                        fseek(fp,oldpos,SEEK_SET) ;
                        break ;
          case OUT   : ch=check_oprnd ( ) ;
                        Y[ch] =AC;
                        if(ch<2)  //Y[2], Y[3]  is BUFFER only not OUT to DAC
                          DAC(ch,AC*4095/5+819) ;
                        break ;
          case ADD   : ch\check_oprnd ( ) ;
                        AC=AC+foperand;
                        break ;
          case SUB   : ch=check_oprnd ( ) ;
                        AC=AC-foperand;
                        break ;
          case DIV   : ch=check_oprnd ( ) ;
                        AC=AC/foperand;
                        break ;
          case MUL   : ch=check_oprnd ( ) ;
                        AC=AC*foperand;
                        break ;
          case SQR   : AC=sqrt (AC) ;
                        break ;
          case END   : start_process_graph ( ) ;

           }
        }
        fseek(fp, 0, SEEk_SET) ;
        white_data(SAVE_FILES, X[0], 0) ;
    }
    fclose(fp) ;
}
int check_oprnd(void)
{
    int oprnd, ch ;
    float foprnd ;
    if(fscanf(fp, "%f", &foprnd) ) {
         foperand=foprnd ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            oprnd=(int)foprnd+54;
        }
        else fscanf(fp, "%x", &oprnd) ;
        switch(oprnd)
        {
        case X1:ch=0;flag=0;break;
        case X2:ch=1;flag=0;break;
        case X3:ch=2;flag=0;break;
        case X4:ch=3;flag=0;break;
        case Y1:ch=0;flag=0;break;
        case Y2:ch=1;flag=0;break;
        case Y3:ch=2;flag=0;break;
        case Y4:ch=3;flag=0;break;
        case S1:ch=0;flag=0;break;
        case S2:ch=1;flag=0;break;
        case S3:ch=2;flag=0;break;
        case S4:ch=3;flag=0;break;
        }
        return ch ;
    }

    void init_data(void)
    {
        int ch ;
        Ts=1 ;
        for (ch=0; ch<=0; ch++) {
                Kp[ch]=10 ;
                Ki[ch] =1 ;
                Kd[ch]=10 ;
                Sp[ch]=3 ;
                Xo[ch]=X[ch] ;
                Yo[ch]=0 ;
                K1[ch]=Kp[ch]+Ki[ch]*Ts*100+Kd[ch]/(Ts*100) ;
                K2[ch]= -(Kp[ch]+2*Kd[ch]/(Ts*100) ) ;
                K3[ch]=Kd[ch]/(Ts*100) ;
                er1[ch]=0 ;
                er2[ch]=0 ;
                PV[ch]=0 ;
                MV[ch]=0 ;
        }
    }

    /*void init_pid(void)
    {
        int  ch ;
        for(ch=0; ch<=3; ch++) {
                K1[ch]=Kp[ch]+Ki[ch]*Ts*100+Kd[ch]/(Ts*100) ;
                K2[ch]=-(Kp[ch]+2*Kd[ch]/(Ts*100) ) ;
                K3[ch]=Kd[ch]/(Ts*100) ;
                er1[ch]=0 ;
                er2[ch]=0 ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        PV[ch]=0 ;
        MV[ch]=0 ;
    }
} */

void pid(float sp,int ch)
{
    float er ;
    er=sp*0.8-AC ;
    MV[ch]+=K1[ch]*er+K2[ch]*er1[ch]+K3[ch]*er2[ch] ;
    er2[ch]=er1[ch] ;
    er1[ch]=er ;
    if(MV[ch] > 4)  MV[ch]=4;
    else if (MV[ch]  < 0)  MV[ch]=0 ;
    MV[ch]=4-MV[ch] ;
    AC=MV[ch] ;
}

void start_process_graph(void)
{
void *text, *picture ;
        scan_analog_graph( ) ;
        check_mouse_process(104) ;
        if(  (mx>670)  &&  (mx<745)  &&  (my>475)  && (my < (475+30)) && (mb==1)  )
          {
            sound_mouse ( ) ;
            text=malloc(imagesize(40, 475, 135, 515) ) ;
            getimage (670+4, 475+4, 745-5, 515-15, text) ;
            picture=malloc(imagesize(40,475,135,515) ) ;
            getimage(670-3, 475-3, 745+3, 515+3, picture) ;
            kmit_box(670, 475, 745, 505, 3, 2) ;
            putimage(670+4, 475+4, text, COPY_PUT) ;
            delay(150) ;
            putimage(670-3, 475-3, picture, COPY_PUT) ;
            free(picture) ; free(text) ;
            mouse_of_graph ( );
          }
}

void scan_analog_graph(void)
{
int i,ysp;
int yxo0, yxo1, yxo2, yxo3, yx0 yx1, yx2, yx3 ;
int y0, y1, y2, y3, yo0, yo1, yo2, yo3 ;
//----------              Start plot Graph      ---------------
 if(x_cor>600) {
    X_cor=0 ;
    setfillstyle(1,16+10) ; setcolor(16+10); bar(50+3, 50+3, 710-3, 400-3) ;
//  ..................  Scale line  ........................
        setcolor(27) ;
      for(i=60;i<700;i+=10)              //  Vertical  Axis
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        line(i, 60, i, 377) ;
        setcolor(25) ;
      for(i=60;i<710;i+=50)
        line(i, 60, i, 376) ;
        setcolor(27) ;
      for(i=380;i>60;i-=10)              //  Horizontal  Axis
        line(60, i, 700, i)
        setcolor(25) ;
      for (i=380;i>60;i-=50)             //  Horizontal Axis Active
        line(60, i, 700, i) ;
    }
   nouttextxy3d(450,475,LIGHTRED,"LT:") ;
   setcolor(YELLOW) ;
   gprintfxy(480,475, "%4.1f% ", X[0]*25) ;

   yxo0=380-(62.5*Xo[0]) ;
//       yxo1=380-(62.5*Xo[1) ;
//       yxo2=380-(62.5*Xo[2) ;
//       yxo3=380-(62.5*Xo[3) ;
   yo0=380-(62.5*Yo[0]) ;
//       yo1=380-(50*Yo[1) ;
//       yo2=380-(50*Yo[2) ;
//       yo3=380-(50*Yo[3) ;

  yx0=380-(62.5*X[0]) ;
//       yx1=380-(50*X[1) ;
//       yx2=380-(50*X[2) ;
//       yx3=380-(50*X[3) ;
   y0=380-(62.5*Y[0]) ;
//       y1=380-(50*Y[1) ;
//       y2=380-(50*Y[2) ;
//       y3=380-(50*Y[3) ;

   setcolor(YELLOW) ;
   if(  (y0>60)  ||  (yo0>60)  ) {
        line(64+x_cor,yxo0,65+x_cor+4,yx0) ;
//         line(64+x_cor,yxo1,65+x_cor+4,yx1) ;
//         line(64+x_cor,yxo2,65+x_cor+4,yx2) ;
//         line(64+x_cor,yxo3,65+x_cor+4,yx3) ;
        setcolor(RED) ;
        line(64+x_cor,yo0,65+x_cor+4,y0) ;
//         line(64+x_cor,yo1,65+x_cor+4,y1) ;
//         line(64+x_cor,yo2,65+x_cor+4,y2) ;
//         line(64+x_cor,yo3,65+x_cor+4,y3) ;
    }
    x_cor+=4 ;
    for(i=0;i<=3;i++) {
        Xo[i]=X[i] ;
        Yo[i]=Y[i] ;
    }
 }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void white_data(char *save_files,float a2,int mode)
{
 struct  tm *sys_time;
 time_t  t ;
 int  h,m,s,d ;
 int  checktime ;
t=time('\0]) ;
sys_time = localtime(&t) ;
h=sys_time->tm_hour ;
m=sys_time->tm_min ;
s=sys_time->tm_sec ;
           checktime =  (s%1) ;
        if( (data_fb  =  fopen(save_files, "ab") )==NULL)
           {  printf{"Cannot open file\n");exit(0); }
        //   if( (checktime==1)&&(s_!=s) ) {
           if(s_!=s) {

fprintf(data_fb, "   Time is  %2d:%2d:%2d    LT:    %2.4f    |\n",h,m,sys_time->tm
                if(mode==1)
fprintf(stdprn,   "   Time is  %2d:%2d:%2d    LT:    %2.4f    |\n",h,m,sys_time->tm
           }
       fclose(data_fb) ;
          h_=h ;
          m_=m ;
          s_=s ;
}

//----------------  Check Mouse Information For Plant_Graph  ---------

void info_mouse(void)          // This function is not use it safe that use
{                              // By.  function " menu_first(void)  " Only
void  *box_picture ;
void  *text_picture ;
int  position=0 ;
int y,i ;
get_mouse_position(LIGHTBLUE) ;  //  Return To  "mx my mb"
    if(my>=420  &  my<=450   ||  key_buff  !=0) {
position=1 ;
    if(mx>575 & mx<645)        //    EXIT
        {
        sound_mouse ( ) ;
box_picture=malloc(imagesize(575,420,655,450) ) ;
text_picture=malloc(imagesize(600,425,645,445) ) ;
getimage(575,420,655,450,box_picture) ;
getimage(600,425,645,442,text_picture) ;
  kmit_box(575,420,650,451,3,2) ;
  putimage(600+1,425+1,text_picture,COPY_PUT) ;
  delay(150) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    loadthai3 ( ) ;
          setfillstyle(1,BLACK) ;setcolor(BLACK); bar(100,100,600,350) ;
           kmit_box(200,200,450,250,3,1) ;
           kmit_box(175,175,475,275,3,1) ;
           kmit_box(150,150,500,300,3,1) ;
           kmit_box(125,125,550,325,3,1) ;
           kmit_box(100,100,600,350,3,1) ;
       putimage(575,420,box_picture, COPY_PUT) ;
       free(box_picture); free(text_picture) ;
     setcolor(BLACK); rectangle(100,100,600,350) ;

   kmit_rec(125,125,575,325,0,7,);
kmit_36(150,150);
      setcolor(DARKGRAY);
outthai(250+25-1,190-1,1"                    " ) ;
outthai(250+25-1,210-1,1,"                   " ) ;
outthai(250+25-1,210-1,1,"                   " ) ;
outthai(250+25-1, 256-1,1,"       EXIT    ");
        setcolor(WHITE) ;
outthai(250+25+1,190+1,1,"                   " ) ;
outthai(250+25+1,210+1,1,"                   " ) ;
outthai(250+25+1,256+1,1,"                   " ) ;
    setcolor(BLUE);
outthai(250+25+,190,1,"          EXIT " ) ;
outthai(250+25+,190,1, "                     " ) ;
outthai(250+25+,210,1, "                     " ) ;
     setcoler(RED) ;
outthai(250+25,265,1, "          EXIT " ) ;
     tuy_sound1 ( ) ;
      delay (500) ;
      closegraph ( ) ;
        textmode (3) ;
        textcolor(10+128);
       cprintf("     This PROGRAMABLE CONTROLLER ON PC      \r\n " ) ;
       cprintf("          The PC-LAB CARD PCL-818           \r\n\r\n" ) ;
         tectcolor(LIGHTRED) ;
       cprintf(" Return To Program Again By PRINT \ " EXIT \" at DOS Prompt\r\n"
) ;
       cprintf(" Buy....Buy.... See You Again Next Time\r\n\r\n" ) ;
        system ( " " ) "
         svga_mode ( ) ;
   tuy_sound2 ( ) ;
            menu_first ( ) ;
            } // END IF LOOP DOS SHELL
      else if (mx > 50 & 120 ) // ABOUT
        {
        sound_mouse ( ) ;
box_picture=malloc(imagesize(50, 42, 130, 450) ) ;
text_picture=malloc(imagesize(60, 423, 120, 445) ) ;
getimage(50, 420, 130, 450, box_picture) ;
getimage(60, 424, 120, 445, text_picture) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void white_data(char *save_files,float a2,int mode)
{
  struct  tm *sys_time;
  time_t  t ;
  int  h,m,s,d ;
  int  checktime ;
t=time('\0]) ;
sys_time = localtime(&t) ;
h=sys_time->tm_hour ;
m=sys_time->tm_min ;
s=sys_time->tm_sec ;
            checktime =  (s%1) ;
          if( (data_fb  =  fopen(save_files, "ab") )==NULL)
             {  printf{"Cannot open file\n");exit(0); }
          //   if( (checktime==1)&&(s_!=s) ) {
              if(s_!=s) {

fprintf(data_fb, "   Time is  %2d:%2d:%2d    LT:    %2.4f    |\n",h,m,sys_time->tm
                  if(mode==1)
fprintf(stdprn,   "   Time is  %2d:%2d:%2d    LT:    %2.4f    |\n",h,m,sys_time->tm
              }
        fclose(data_fb) ;
           h_=h ;
           m_=m ;
           s_=s ;
}

//- - - - - - - - - - - - - - - -  Check Mouse Information For Plant_Graph  - - - - - - - - -

void info_mouse(void)          // This function is not use it safe that use
{                              // By.  function " menu_first(void)  "  Only
void  *box_picture ;
void  *text_picture ;
int  position=0 ;
int y,i ;
get_mouse_position(LIGHTBLUE) ;  // Return To  "mx my mb"
     if(my>=420  &  my<=450   ||  key_buff  !=0) {
position=1 ;
     if(mx>575 & mx<645)        //    EXIT
         {
         sound_mouse ( ) ;
box_picture=malloc(imagesize(575,420,655,450) ) ;
text_picture=malloc(imagesize(600,425,645,445) ) ;
getimage(575,420,655,450,box_picture) ;
getimage(600,425,645,442,text_picture) ;
  kmit_box(575,420,650,451,3,2) ;
  putimage(600+1,425+1,text_picture,COPY_PUT) ;
  delay(150) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    kmit_box(50, 420, 125, 451, 3, 2) ;
    putimage(60+1, 424+1, text_picture, COPY_PUT) ;
    delay(150) ;

          loadthai3 ( ) ;
           setfillstyle(1, BLACK) ; setcolor(BLACK); bar(100, 100, 600, 350) ;
             kmit_box(200, 200, 450, 250, 3, 1) ;
             kmit_box(175, 175, 475, 275, 3, 1) ;
             kmit_box(150, 150, 500, 300, 3, 1) ;
             kmit_box(125, 125, 550, 325, 3, 1) ;
             kmit_box(100, 100, 600, 350, 3, 1) ;
             setcolor(BLACK); rectangle(100, 100, 600, 350);
               putimage(50, 420, box_picture, COPY_PUT) ;
  free(box_picture); free(text_picture) ;
    kmit_rec(125, 125, 575, 325, 0, 7) ;
          setcolor(DARKGRAY); outthai(223, 153, 1, "
          setcolor( (16*4) -1+13); outthai(220, 150, 1, "
              kmit_36(150,150) ;
  loadthai0 ( ) ;
          setcolor(DARKGRAY) ;
          outthai(100+50-1, 125+75-1, 1, "      "); outthai(250-1, 125+75-1, 1, "
          outthai(100+50-1, 125+75+30-1, 1, "     "); outthai(250-1,125+75+30-1, 1, "
          outthai(100+50-1, 125+75+60-1, 1, "     "); outthai(250-1,125+75+60-1, 1, "
          outthai(100+50-1, 125+75+90-1, 1, "     "); outthai(250-1,125+75+90-1, 1, "

          setcolor(WHITE) ;
          outthai(100+50+1, 125+75+1, 1, "     "); outthai(250, 125+75+1, 1, "         ") ;
          outthai(100+50+1, 125+75+30+1, 1, "     "); outthai(250, 125+75+30+1, 1, "    " ) ;
          outthai(100+50+1, 125+75+60+1, 1, "     "); outthai(250, 125+75+60+1, 1, "    " ) ;
          outthai(100+50+1, 125+75+90+1, 1, "     "); outthai(250, 125+75+90+1, 1, "      " ) ;

           setcolor( (16*4) -1+6 ) ; outthai ( 100+50,125+75,1,"       " );setcoler((16*4)-1+8);out
           setcolor( (16*4) -1+6 ) ; outthai( 100+50,125+75+30,1,"      " );setcoler((16*4)-1+8);
           setcolor( (16*4) -1+6 ) ; outthai( 100+50,125+75+60,1,"      " );setcoler((16*4)-1+8);
           setcolor( (16*4) -1+6 ) ; outthai( 100+50,125+75+90,1,"      " );setcoler((16*4)-1+8);
          tuy_sound1 ( ) ;
          delay(300) ;
          info_mouse ( ) ;
}          // End IF LOOP ABOUT

        else if(mx>150 & mx<220)   //      OPEN FILE
          {
          sound_mouse( ) ;
box_picture=malloc(imagesize(150, 420, 230, 450) ) ;
text_picture=malloc(imagesize(160, 432, 220, 445) ) ;
getimage(150, 420, 230, 450, box_picture) ;
getimage(160, 432, 220, 445, text_picture) ;
  kmit_box(150, 420, 225, 451, 3, 2) ;
  putimage(160+1, 432+1, text_picture, COPY_PUT) ;
  delay(150) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          loadthai3 ( ) ;
           setfillstyple(1, BLACK); setcolor(BLACK); bar(100, 100, 600, 350);
             kmit_box(200, 200, 450, 250, 3, 1) ;
             kmit_box(175, 175, 475, 275, 3, 1) ;
             kmit_box(150, 150, 500, 300, 3, 1) ;
             kmit_box(125, 125, 550, 325, 3, 1) ;
             kmit_box(100, 100, 600, 350, 3, 1) ;
             setcolor(BLACK); rectangle(100, 100, 600, 350) ;
             putimage(150, 420, box_picture, COPY_PUT);
             free(box_picture); free(text_picture) ;
             kmit_rec(125, 125, 575, 325, 0, 7) ;
             setcolor(26) ;
    settextstyle(5, HORIZ_DIR, 2) ;
    for (y=3;y<7;y++)
          for(i=1;i<6;i++)
            outtextxy(50+82*i, 45*y," Kmit-L ") ;
            kmit_36(150,150) ;
     tuy_sound1( ) ;
     settextstyple(DEFAULT_FONT,  HORIZ_DIR, 1) ;
     open_file ( ) ;
     compile ( ) ;
     info_mouse ( ) ;
    }      // End IF LOOP TEST
          else if(mx>250 & mx<320)     //    RUN
          {
           sound_mouse( ) ;
  box_picture=malloc(imagesize(250, 420, 330, 450) ) ;
  text_picture=malloc(imagesize(260, 432, 320, 445) ) ;
  getimage(250, 420, 330, 450, box_picture) ;
  getimage(260, 432, 320, 445, text_picture) ;
   kmit_box(250, 420, 325, 451, 3, 2) ;
   putimage(260+1, 432+1, text_picture, COPY_PUT) ;
   delay(150) ;

          loadthai3 ( ) ;
            setfillstyle(1, BLACK); setcolor(BLACK); bar(100, 100, 600, 350) ;
              kmit_box(200, 200, 450, 250, 3, 1) ;
              kmit_box(175, 175, 475, 275, 3, 1) ;
              kmit_box(150, 150, 500, 300, 3, 1) ;
              kmit_box(125, 125, 550, 325, 3, 1) ;
              kmit_box(100, 100, 600, 350, 3, 1) ;
              setcolor(BLACK); rectangle(100, 100, 600, 350) ;
                putimage(250, 420, box_picture, COPY_PUT) ;
  free(box_picture); free(text_picture) ;
          kmit_36(150, 150) ;
              kmit_rec(125, 125, 575, 325, 0, 7) ;
    setcolor(WHITE); outthai(250+30+1, 195+1, 1, "                    " ) ;
    setcolor(DARKGRAY); outthai(250+30-1, 195-1, 1, "                    " );
    setcolor(LIGHTGRAY); outthai(250+30, 195, 1, "                    " ) ;
        setcolor(44) ;
  outthai(250+25, 190, 1, "                    ") ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    tuy_sound1 ( ) ;
          delay(300); mb=0 ;
     plant_graph( ) ;
   //plant_first ( ) ;
  }       //  End  IF LOOP RUN
  else
    info_mouse( ) ;
       }            //  End Loop  IF " Y POSITION "
  if(position==0)
    { no_mouse( );delay(245); info_mouse( ); }
  }

  void open_file(void)
  {
  int i;
  void *picture;
  void *text ;
            sound_mouse( );
            key_buff=0 ;
            text=malloc(imagesize(40, 475, 135, 515) ) ;
            getimage(313+4, 210+4, 386-5, 250-15, text) ;
            picture=malloc(imagesize(40, 475, 135, 515) ) ;
            getimage(313-3, 210-3, 386+3, 250+3, picture) ;

            kmit_box(313, 210, 386, 240, 3, 2) ;
            putimage(313+4, 210+4, text, COPY_PUT) ;
            delay(150) ;
            putimage(313-3, 210-3, picture, COPY_PUT) ;
            free(picture); free(text) ;
            picture=malloc(imagesize(0, 0, 200, 100) ) ;
            getimage(250, 175, 450, 275, picture) ;
            for(i=0; i<40; i+=1)
            { kmit_box(290-i, 215-i, 410+i, 235+i, 3, 1); sound(1000+(100*1) ); }
            nosound( ) ;
            kmit_rec(250+10, 175+10, 450-10, 275010, 0, LIGHTGRAY) ;
            kmit_box(345-5, 215-10, 450-12, 215+1, 3, 2) ;
          //     Get Save Files
            loadthai0( ) ;
            setcolor(DARKGRAY); outthai(270-1, 215-1, 1, "          ") ;
            setcolor(WHITE); outthai(270+1, 215+1, 1, "          ") ;
            setcolor(241); outthai(270, 215, 1, "          ") ;

            setcolor(DARKGRAY); outthai(270-1, 24501, 1, "          ") ;
            setcolor(WHITE); outthai(270+1, 245+1, 1, "          ") ;
            setcolor(241); outthai(270, 245, 1, "          ") ;
            set_maxchars=11;
            setcolor(131); gscanfxy(345, 215, "%s", &CODE_FILES);putch(007) ;
            setcolor(216); gprintfxy(345, 245, "%s", CODE_FILES) ;
            getch( ) ;
            delay(300) ;
            for(i=0; i<20; i+=2)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        {

                putimage(250, 175, picture, COPY_PUT) ;
                kmit_box(250+i, 175+i, 450-i, 275-i,3, 1) ;
                sound(3000-(200*i) ) ;
                nosound( );
                delay(50) ;
        }
        putimage(250, 175, picture, COPY_PUT); free(picture) ;
}

void menu_first(void)
{
void *picture ;
void *box_picture ;
void *text_picture ;
int i_bar=0, i=0, c=0 ;
setviewport(0, 0, getmaxx( ), getmaxy( ), 1) ;
hide_mouse( );
setfillstyle(1, LIGHTGRAY); setcolor(LIGHTGRAY); bar(0, 0, getmaxx( ), getmaxy( ));
kmit_box(0, 500, 799, 599, 3, 1) ;
setfillstyle(1, BLACK); setcolor(BLACK) ;
setcolor(WHITE);     rectangle(1-1, 1-1, 798-1, 548-1);//line(2+1, 498-1, 897-1, 498-1)
setcolor(DARKGRAY); rectangle(1+1, 1+1, 798+1, 548+1);//line(2+1, 498+1, 897+1, 498+1)
setcolor(LIGHTGRAY); rectangle(1, 1, 798, 548); // line(2, 498, 897, 498) ;

  for(i=0;i<34;i++)
      kmit_rec(700),50+(10*i), 775, 70+(10*i), 1, 4+i) ;
 pipe  (735, 420, 735, 480, 2); endpipe  (735, 420, 2); c_pipe(725, 465,  4);
 pipe  ( 50, 485, 725, 485,1); arrow (40, 485, 3); c_pice(500, 465, 0);
 pipe  (500, 485, 550, 485,1);

 i=0
          kmit_box(50, 50, 650, 400, 3, 2) ;
          setfillstyle(1, BLACK); setcolor(BLACK); bar(53, 53, 647, 397); kmit_36(30, 30);
setcolor(WHITE) ;     rectangle(50-5+1, 50-5+1,650+5+1, 400+5+1) ;
setcolor(DARKGRAY) ;     rectangle(50-5-1, 50-5-1,650+5-1, 400+5-1) ;
setcolor(LIGHTGRAY) ;     rectangle(50-5, 50-5,650+5, 400+5) ;
          c=15*2; i=0;
          for(i_bar=0; i_bar<=15; i_bar++) {
          setfillstyle(1, c-i_bar); setcolor(c-i_bar) ;
bar(0, 550+i, 800,554+i) ;
i+=4;
     }
for(c=0; c<24; c++) {
     for(i=0;i<398;i++)
putpixel ((i*2)+3, (c*2) +500, WHITE);
     }
kmit_box(670, 517,770, 537, 3, 2) ;
      loadthai3 ( );
          setcolor(WHITE) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
settextstyle(3, HORIZ_DIR, 2); outtextxy(101, 16, "PROGRAMMABLE CONTROLLER ON PC");
setcolor(LIGHTGRAY);
outthai(3, 571, 1, "                                                                    ");
settextstyle(0, HORIZ_DIR, 1); outtextxy(601, 573, " Mode  800x600  256 Color") ;
        setcolor(DARKGRAY) ;
outthai(5,573,1, "                                                                    ") ;
settextstyle(3, HORIZ_DIR, 2); outtextxy( 99, 14, "PROGRAMMABLE CONTROLLER ON PC");
settextstyle(0, HORIZ_DIR, 1); outtextxy(599, 571, " Mode  800x600  256 Color") ;
        setcolor(LIGHTBLUE) ;
outthai(4, 572, 1, "                                                                    ");
setcolor(LIGHTRED);settextstyle(3, HORIZ_DIR, 2); outtextxy(100, 15, "PROGRAMMABLE
    CONTROLLER ON PC");
setcolor(BLACK);settextstyle(0, HORIZ_DIR, 1); outtextxy(600, 572, " Mode  800x600  256
    Color") ;

   //                          Button Check of Program
        kmit_box(51, 421, 124, 449, 3, 1) ;
        setcolor(DARKGRAY); rectangle(50-1, 420-1, 125-1, 450-1) ;
        setcolor(WHITE);rectangle(50+1, 420+1, 125+1, 450+1) ;
        setcolor(BLACK); rectangle(50, 420, 125, 450) ;
picture=malloc(imagesize(40,419,125,451) ) ;
getimage(49, 419, 125, 451, picture) ;
putimage(150-1, 420-1, picture, COPY_PUT); //  button 2
putimage(250-1, 420-1, picture, COPY_PUT); //  button 3
putimage(350-1, 420-1, picture, COPY_PUT); //  button 4
putimage(575-1, 420-1, picture, COPY_PUT); //  button "EXIT"
free(picture) ;
  //      ------------- Print Name of Button ---------
  loadthai3 ( ) ;
        setcolor(DARKGRAY) ;
outthai(69, 432+2, 1, "          ") ;   // Button - HELP -
outthai(169, 432+2, 1, "        ");     // Button  - Test -
outthai(269-3, 432+2, 1, "         ") ;  // Button - WORK -
outthai(369+3, 433+2, 1, "       ") ;     // Dos Shell
outthai(599, 432, 1, "     ") ;          // Button  - EXIT -

        setcolor(WHITE) ;
outthai(71, 434+2, 1, "          ") ;
outthai(171, 434+2, 1, "          ") ;
outthai(271, 434+2, 1, "          ") ;
outthai(371, 433+2, 1, "          ") ;
outthai(601, 434, 1, "          ") ;

        setcolor(BLACK) ;
outthai(70, 433+2, 1, "          ") ;
outthai(170, 433+2, 1, "          ") ;
outthai(270-3, 434+2, 1, "          ") ;
outthai(370+3,433+2, 1, "          ") ;
outthai(600, 434, 1, "          ") ;
//                      Show  About
box_picture=malloc(imagesize(50, 420, 130, 450) ) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
text_picture=malloc(imagesize(60,423, 120, 445) );
getimage(50, 420, 130, 450, box_picture) ;
getimage(60, 424, 120, 445, text_picture) ;
  kmit_box(50, 420, 125, 451, 3, 2) ;

  putimage(60+1, 424+1, text_picture, COPY_PUT) ;
  delay(150) ;

        loadthai3 ( );
         setfillstyle(1, BLACK); setcolor(BLACK);bar(100, 100, 600, 350) ;
           kmit_box(200,200,450,250,3,1) ;
           kmit_box(175,175,475,275,3,1) ;
           kmit_box(150,150,500,300,3,1) ;
           kmit_box(125,125,550,325,3,1) ;
           kmit_box(100,100,600,350,3,1) ;
           setcolor(BLACK); rectangle(100, 100, 600, 350) ;
            PUTIMAGE(50, 420, box_picture, COPY_PUT);
free(box_picture); free(text_picture) ;
 kmit_rec(125,125,575,325,0,7) ;
    setcolor(DARKGRAY) ;outthi(223,153,1,"
    setcolor( ( 16*4 ) -1+13) ; outthai (220,150,1 "
       kmit_36(150,150);
loadthai0 ( ) ;
      setcolor (DARKGRAY) ;
      outthai ( 100+50-1,125+75-1,1, "      " ) ;outthai (250-1,125+75-1,1, "
      outthai ( 100+50-1,125+75+30-1,1, "      " ) ;outthai (250-1,125+75+30-1,1, "
      outthai ( 100+50-1,125+75+60-1,1, "      " ) ;outthai (250-1,125+75+60-1,1, "
      outthai ( 100+50-1,125+75+90-1,1, "      " ) ;outthai (250-1,125+75+90-1,1, "

       setcolor(WHITE) ;
      outthai ( 100+50-1,125+75+1-,1,"      " ) ;outthai (250,125+75+1,1, "
      outthai ( 100+50-1,125+75+30+1,1,"      " ) ;outthai (250,125+75+3+1,1, "
      outthai ( 100+50-1,125+75+60+1,1,"      " ) ;outthai (250+125+75+60+1,1, "
      outthai ( 100+50-1,125+75+90+1,1,"      " ) ;outthai (250,125+75+90+1,1, "


       setcolor( (16*4) -1+6) ; outthai ( 100+50,125+75,1, "      " ) ; setcolor ((16*4) - 1+8)
;out
       setcolor( (16*4) -1+6) ; outthai ( 100+50,125+75+30,1, "      " ) ; setcolor ((16*4) -
1+8);
       setcolor( (16*4) -1+6) ; outthai ( 100+50,125+75+60,1, "      " ) ; setcolor ((16*4) -
1+8);
        setcolor( (16*4) -1+6) ; outthai ( 100+50,125+75+90,1, "      " ) ; setcolor ((16*4) -
1+8);
tuy_sound1 ( ) ;

info_mouse ( ) ;
}
i+=4;
      }
    setfillstyle(1, 18); setcolor(18); bar(100+10, 300+10, 700+10, 500+10);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        bar(500+10, 300+10, 700+10, 525+10) ;
      setfillstyle(1, 228); setcolor(228); bar(100, 300, 700, 500) ;
        bar(500, 300, 700, 525) ;
      setfillstyle(1, 43); setcolor(43) ;
  for(pcb_c=0;pcb_c<18;pcb_c++){
  bar(510+(10*pcb_c), 500, 515(10*pcb_c), 525) ;
        }



     loadthai2 ( );
    setcolor(DARKGRAY); utthai(150-1,570-1,1,"Model PCL-818 Programmable Controller ";
    setcolor(WHITE); outthai(150+1, 570+1, 1, "Model PCL-818 Programmable Controller ";
    setcolor(BLACK); outthai(150, 570, 1, "Model PCL-818 Programmable Controller ";
    setcolor(LIGHTGRAY);

    kmit_box(200, 350, 600, 450, 3, 2) ;
  setfillstyle(1, LIGHTBLUE); bar(200+3, 350+3, 600-3, 450-3) ;
    setviewport(200+3, 350+3, 600-3, 450-3, 1) ;

    loadthai4( ); i=0;

    / / - - - - - - - - - - - - - - - - - - - - Loop Off Display Project Group - - - - - - - - - - - -
  for(; ;)
  {
   if(mouse==1)
    {nosound( ); goto go_out;}
     for(y=120;y>0;y--)
   {
   if(i==109)
      i=0;
    if(mouse==1)
     {nosound( ); break;}
     setcolor(YELLOW); outthai(5, (y-20), 1, "                                                                                    ");
        outthai (5, (y-20)+space_line      ,1, " PROGRAMMABLE CONTROLLER ON PC");
   kmit_song(i, set_sound); i++;
              if(kbhit( )!=0) {
              key_buff= (int)toupper(getch( ) );
              }
   if(key_buff!=0 )
     nosound( ); key_buff=0; goto go_out;}
       bar(0,0,400,200);
          mouse=getmouse_b( );
   }
       for(y=140;y>0;y--)
   {
   if(i==109)
     i=0;
   if(mouse==1)
     {nosound( );goto go_out;}
     setcolor(YELLOW);outthai(5, (y-40),1,"                    PROJECT     ");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    outthai( 5, (y-40)+space_line , 1, "                                    ") ;
    outthai( 5, (y-40)+space_line*2, 1, "                                    ");
      kmit_sond(i,set_sound);i++;
if(kbhit( )!=0)  {
           key_buff=  (int)toupper(getch( ) ) ;
           }
if(key_buff!=0  )
{ nosound( ); key_buff=0;goto go_out; }
mouse=getmouse_b( );
      bar(0,0,400,200);
}
     for(y=150;y>0;y--)
{
if(i==109)
  i=0;
if(mouse==1)
  {nosound( ); goto go_out;}
  setcolor(YELLOW); outthai(  5, (y-50)          ,1,"                  :") ;
      outthai(  55,  (y-50)+space_line     ,1,"          36013277    ");
      outthai(  55,  (y-50)+space_line*2   ,1,"           3601331    ");
      outthai(  55,  (y-50)+space_line*3   ,1,"          36013296    ");

   kmit_song(i, set_sound); i++;
if(kbhit( ) !=0)  {
           key_buff=  (int)toupper(getch( ) );
           }

 if(key_buff!=0  )
 { nosound ( );key_buff=0;goto go_out;}
   mouse=getmouse_b( ) ;
   bar(0, 0, 400, 200) ;
 }
         for(y=120;y>0;y--)
 {
 if(i==109)
   i=0;
 if(mouse==1)
   {nosound( );goto go_out;}
   setcolor(YELLOW);   outthai(  5, (y-20)          ,1, "                  :") ;
        outthai(  5, (y-20) +space_line   ,1, "                          :);
         kmit_sond(i, set_sound);i++;
 if(kbhit( ) !=0)  {
          key_buff=  (int)toupper(getch( ) );
          }

 if(key_buff! =0  )
 { nosound( );key_buff=0;goto go_out;}
       mouse=getmouse_b( );
       bar(0,0,400,200);
          }
        for(y=140;y>0;y--)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
if(i==109)
   i=0;
if(mouse==1)
  {nosound( );goto go_out;}
  setcolor(YELLOW);  outthai(  5, (y-40)                    ,1,"                    ");
     outthai(  5, (y-40) +space_line        ,1,"          Borland C++ Version 3.1");
  setcolor(LIGHTRED);outthai(5, (y-40)+space_line*3    ,1,"                    :)
    kmit_song(i,set_sound); i++;
 if(kbhit( ) ! =0)   {
            key_buff=   (int) toupper(getch ( ) );
            }

 if(key_buff!=0  )
 {  nosound( );key_buff=0;goto go_out;}
     mouse=getmouse_b( );
    bar(0,0,400,200) ;
   }
 if(kbhit ( )!=0)  {
            key_buff=  (int) toupper(getch( ) );
            }

 if(key_buff!=0  )
  {  nosound( );key_buff=0;goto go_out;}
  mouse=getmouse_b( );

    free(picture) ;

  }
  go_out :
 }

 void loadthai0(void)
 {
  int   i=0;
        for(i=0;i<5120;i++)
          thaidata[i]=thaifont0(i) ;
 }

 void loadthai1(void)
 {
  int   i=0;
        for(i=0;i<5120;i++)
          thaidata[i]=thaifont1(i) ;
 }

 void loadthai2(void)
 {
  int   i=0;
        for(i=0;i<5120;i++)
          thaidata[i]=thaifont2(i) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
}

void loadthai3(void)
{
 int   i=0;
       for(i=0;i<5120;i++)
         thaidata[i]=thaifont3(i) ;
}

void loadthai4(void)
{
 int   i=0;
       for(i=0;i<5120;i++)
         thaidata[i]=thaifont4(i) ;
}


/* ---------------------- function display thai character ----------------*/

void outthai (int x, inty, int size, unsigned char *_text)
{
 int i, n, index, plot, da_plot, oldx, oldy;
 register loop, loopy;

   y = y+11;
   x -= 1;
   while(*_text  != '\0')  {
         index = *_text;
         index *= 20;
         x += 1;
         oldy  =  y;
         oldx  =  x;
         y = y - 20;
         for(i=0; i<20; i++)  {
  loopy  =  size;
  do{
     x =  oldx;
     da_plot  =  thaidata[index+i] ;
     plot  =  0x0080 ;

     if( ( plot & da_ploy)!=0)  {
loop  =  size;
do{
    putpixel(x,y,getcolor( ) );  x+=1;  loop  -=1;
}while(loop>0) ;
      }
      if( (plot  &  da_plot) ==0)  {
loop  =  size;
do{
   x+=1;
   loop  -=1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  }while(loop>0);
    }

     for(n=0;n<7;n++) {
 plot = plot  >> 1;
 if((plot & da_plot) !=0) {
    loop = size ;
    do{
       putpixel(x,y,getcolor( ) );
       x+=1;
       loop -=1;
      }while(loop >0) ;
}
if ( (plot & da_plot) = =0) {
     loop =  size ;
     do{
          x++;
          loop -= 1;
         }while(loop>0) ;
 }
         }
        loopy -= 1; y++;
          }while(loopy>0) ;

          }
          _text++; y=oldy;
if( ( ( 212<= *_text) && (*_text <=219) ) || ( (231<=*_text) && (*_text<=236) ) || (*_text
==241)
  x = oldx - 1;
    }
  }

 int gprintfxy(int xloc, int yloc, char *fmt, . . . . )
 {
  va_list argptr;
  char str[140];
  int cnt;
  struct fillsettingstype oldfill;
  char userfillpattern[8];

  va_start(argptr,fmt) ;
  cnt = vsprintf(str, fmt, argptr) ;
  if (str[0] == NULL) return(0) ;

  getfillsettings(&oldfill) ;
  if (oldfill.pattern == USER_FILL);
     getfillpattern(userfillpattern) ;
 setfillstyle(SOLID_FILL,LIGHTGRAY) ;
  bar(xloc,yloc,xloc+textwidth(str),yloc+textheight("H")*5/4) ;
  if (oldfill.pattern == USER_FILL)
     setfillpattern(userfillpattern, oldfill.color) ;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    outtextxy(xloc, yloc, str) ;
    va_end(argptr) ;
    return(cnt);

}
int  gprintf(char  *fmt, ...)
{
    va_list  argptr;
    char str[140];
    int cnt ;
    struct fillsettingstyle oldfill ;
    char userfillpattern[8];
    int xloc, yloc;
    va_start(argptr, fmt) ;
    cnt = vsprintf(str, fmt, argptr);
    if (str[0] == NULL) return(0) ;
        xloc=getx( ); yloc=gety( );
    getfillsettings(&oldfill) ;
  if (oldfill.pattern == USER_FILL);
     getfillpattern*userfillpattern);
  setf.  .      SOLID_FILL, LIGHTGRAY);
  bar(xloc, yloc, xloc+textwidth(str), yloc+textheight("H)*5/4);
  if (oldfill.pattern  ==  USER_FILL)
     setfillpattern(userfillpattern, oldfill.color);
  else
     setfillstyle(oldfill.pattern, oldfill.color);
    outtext(str);
    va_end(argptr) ;
    return(cnt) ;

}
  int  ggetche(void)
  {
        char ch;
        ch  =  getch ( );
        gprintf("%c",ch);
        return (ch);
  }

  int gputch(int c)
  {
    char buffer[2] ;

        sprintf(buffer, "%c",c);
        gprintf(buffer);
        return(c);
  }

  char *ggets(char *buffer)
  {
    int currloc, maxchars, oldcolor;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
struct viewporttype view;
char ch, charbuff[3] ;

buffer[0] = '\0';
currloc = 0;
getviewsettings(&view) ;
   maxchars = set_maxchars;
if (maxchars <-0) return(NULL) ;
gprintfxy(getx( ), gety( ), "_");
while ((ch=getch( ) ) !=13)  {
    if (ch==8)  {
       if (currloc >0)  {
currloc- - ;
if (currloc <=maxchars)  {
  oldcolor = getcolor ( );
  setcolor (getbkcolor ( ) );
sprintf(charbuff, "%c", buffer[currloc]) ;
gprintfxy(getx( ) - textwidth(charbuff), gety( ), "%c",buffer[currloc]) ;
setcolor(oldcolor) ;
moveto(getx( ) - textwidth(charbuff), gety( ) );
          }
        }
      }
      else {
      if (currloc < maxchars)  {
           oldcolor = getcolor( ) ;
           setcolor(getbkcolor ( ) );
           gprintfxy(getx( ), gety( ), "_") ;
           setcolor(oldcolor) ;
           buffer [currloc] = ch ;
           gputch(ch) ;
           currloc++ ;
       }
            else
         putch(0x07);
    }
if (currloc < maxchars)
gprintfxy(getx( ), gety( ), "_") ;
           }
           if (currloc <= maxchars)  {
oldcolor = getcolor( ) ;
setcolor(getbkcolor( ) );
gpritnfxy(getx( ), gety( ), "_") ;
setcolor(oldcolor);
           }
           buffer[currloc] = '\0';
           return(buffer);
 }

int gscanf(char *fmt, . . . )
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    va_list argptr ;
    char str[14] ;
    int cnt;
    va_start(argptr, fmt);
    ggets(str)  ;
    cnt  =  vsscanf(str, fmt, argptr) ;
    va_end(argptr) ;
    return(cnt) ;
}

int  gxcanfxy(int xloc,int yloc, char *fmt,  . . . )
{
    va_list  argptr ;
    char str[140];
    int cnt;
    int oldx,  oldy;

    oldx=xloc;
    oldy=yloc;
    moveto(xloc,yloc) ;
    va_start (argptr, fmt) ;
    ggets (str) ;
    cnt = vsscanf (str, fmt, argptr) ;
    va_end(argptr) ;
    moveto(oldx, oldy) ;
    return(cnt) ;
}

void svga_mode(void)
{
    int gdriver, gmode, y;

    gdriver = installuserdriver("svga256",  detectEGA) ;
    gdriver = DETECT;
    checkerror( );
      initgraph(&gdriver,  &gmode,  " ") ;
    checkerrors( ) ;
}

int huge detectEGA(void)
{
    int driver,  mode,  sugmode =  3;

    detectgraph(&driver,  &mode) ;
    if  ( (driver ==  EGA)  | |  (driver ==  VGA) )
        return sugmode ;
    else
        return grError;
}

void checkerrors(void)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
    int errorcode;

    errorcode = graphresult( );
    if  (errorcode != gr0k)
    {
        printf("Graphics error:  %s\n", grapherrormsg(errorcode) ) ;
        printf("This Program subport Super VGA Mode Only\n") ;
        printf("Press any key to halt:");
        getch( ) ;
        exit (1) ;
    }
}
void get_mouse_position(int colors)
{
void *picture;

//                  Install Mouse
//                  Install Curser Mouse

            picture=malloc(imagesize(0,0,8,16) );
            setfillstyle(1, colors);
for(; ;) {
   mx=(int) (getmouse_x ( )/scale_x) ;     //          Mouse  X
   my=(int) (getmouse_y ( )/scale_y) ;     //           Mouse Y
   mb=(int) (getmouse_b ( ) ) ;          //       Mouse Button
getimage(mx, my, mx+8, my+16, picture);
            setcolor(colors);
bar(mx,my,mx+8,my+16);
gprintfxy(66, 524, "%3d    %3d    %d", mx, my, mb) ;
    //  setfillstyle(1, colors) ;
key_buff=0;
    putimage(mx, my, picture, COPY_PUT) ;

            if(kbhit( ) 1=0)   {
            key_buff=0 ;
            key_buff=  (int) toupper(getch( ) );
            }

            if ( (!mb == 0)  | |  (!key_buff == 0 ))  {  //  Check Press Mouse
               free(picture) ;
               break;
            }
    }
}

void load_config(void)
{
int print_mode;
float load_mouse_, load_mouse_y;
char text [3];
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
FILE *fp;

if( (fp = fopen("pci.sys", "rb"))  != NULL  ) {
   fscanf(fp, "%s\t  %x\n   %s\t  %f\n %s\t  %f\n",&text[1], &print_mode, &text[2],&
load_mode
clrscr( );
SETPRINTER =  print_mode;
scale_x = load_mouse_x/800 ;
scale_y = load_mouse_y/600;
printf("Segment Of PCI Card   \"%x\"\nMaximum Of X Axis    \"%f\"\nMaximum Of Y
 Axis
fclose(fp);
}
else{
     printf("can't open file  PCI.SYS=\n");
     exit (0) ;
    }
}

void nouttextxy3d(int x, int y, int color, char *text)
{
char buffer [80];

   setcolor(DARKGRAY);
   spritf(buffer, "%s",text);
   outtextxy(x-1, y-1, buffer) ;
   setcolor(WHITE);
   sprintf(buffer, "%s", text) ;
   outtextxy(x+1, y+1, buffer) ;
   setcolor(color) ;
   sprintf(buffer, "%s", text);
   outtextxy(x,y,buffer);

}
void outtextxy3d(int x,int y, int color, char *text)
{
char buffer [80];

    setcolor(BLACK);
    sprintf(buffer, "%s",text);
    outtextxy(x-1, y-1, buffer) ;
    setcolor(53) ;
    sprintf(buffer, "%s", text);
    outtextxy(x+1, y+1, buffer);
    setcolor(color) ;
    sprintf(buffer, "%s", text);
    outtextxy(x,y,buffer) ;

}
void check_mouse_process(int colors)
{
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void *Picture_mouse;
int loop;
//                          Install Curser mouse

          for(loop=0; loop<100;loop++) {
            Picture_mouse=malloc(imagesize(0,0,8,16) ) ;
            setfillstyle(1, colors) ;

     mx=(int) (getmouse_x( )/scale_x);          //       Mouse X
     my=(int) (getmouse_y( )/scale_y);          //       Mouse Y
     mb=(int) (getmouse_b( ) ) ;        //       Mouse Button
getimage(mx,my,mx+8,my+16,Picture_Mouse);
           setcolor(colors);
bar(mx,my,ms+8,my+16) ;
gprintfxy(677,524,"%3d   %3d   %d",mx,my,mb) ;
       setfillstyle(1, colors) ;
      putimage(mx,my,Picture_Mouse,COPY_PUT) ;
      free(Picture_Mouse) ;
       if(mb==3) {  // Start loop if mouse
         closegraph( );
          exit (0) ;
         }  // end loop if check mouse
           } // end loop for
}

void compile(void)
{
FILE *fp;
int num_code[1000];                                    // Limit Code Length
int p=0;
int code;

 switch(lexical(num_code) ) {
  case 0:  tuy_sound2 ( ) ;
           outtextxy3d(180,200,RED,"Compile Success");
           strncpy(HEX_FILES, CODE_FILES, 20);
           strcat(HEX_FILES,".HEX") ;
           if ( (fp=fopen(HEX_FILES, "w") ) == NULL) {
             outtextxy3d(180,200,BLUE,"Cannot open file");getch( );
             exit (1) ;
           } else{
           while(0  !=  (cde  = num_code[p++] ) )  {
             switch(code)  {
               case  0xF0  :     fprintf(fp, "%f  ",*(float *)&num_code[p]);
                                 p = p+2;
                                 break;
               case '\n'   :     fprintf(fp, "\n");
                                 break;
               default     :     fprintf(fp, "%X ",code);
                                 break;
             }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            }
            break;
            }
    case 1 :  tuy_sound2( );
              outtextxy3d(180,250,YELLOW, "Open Error");
              break;
    case 2 :  tuy_sound1( );
              outtextxy3d(180,250,GREEN,"Match Error");
              break;
   }
}

//-------------------------------------------------------------------

int lexical(int *num_code)  {

 char *match[ ] = {"LD"   , "PID"   ,"CAS"   ,"OUT"   ,
                   "ADD"   ,"SUB"   ,"MUL"   ,"DIV"   ,"SQR"   ,
                   "END"   ,"X1"   ,"X2"   ,"X3"   ,"X4"   ,
                   "Y1"   ,"Y2"   ,"Y3"   ,"Y4"   ,"S1"   ,
                   "S2"   ,"S3"   ,"S4"   };
  int  code [ ] = {   0x80   ,0x81   ,0x82   ,0x83   ,0x84   ,
                   0x85   ,0x86   ,0x87   ,0x88   ,0x89   ,
                   0x90   ,0x91   ,0x92   ,0x93   ,
                   0xA0   ,0xA1   ,0xA2   ,0xA3   ,
                   0xB0   ,0xB1   ,0xB2   ,0xB3   };
 FILE  *stream;
 int token_num;
 char token[120];
 int i,j,p;
 int check_dot;                          // For check floating point
 char check_num;                         // For check floating point
 int check_count;                        // For check floating point
 int check_alpha;                        // For check floatingpoint

 //-------
  if ( ( stread = flopen9CODE_FILES, "rt") )  == NULL)  {
        ruy_sound1 ( );
   outtextxy3d(180,200,MAGENTA,"File not found.");
   return(1) ;                           // Error!  File not found
   }

 //--------
   do  {
     token_num  = get_token(stream, token);
     p=0;
     for(i=0;i<(token_num  %  10) ; i++)  {
       for(j=0;j<22;j++)   {                          // compare 18 strint (LD, PID, . . .,Y4)
         if(0 ==  stricmp(match[j], &token[p]) )  {  // compare without case sensitive
          *(num_code++)  = code[j] ;                  //Look up Table
          p = p+strlen(&token[p]) +1;                 //Next string
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            break ;
           }
          }
         if(j==22) {                                  //Non match or Float
           check_dot      = 0;
           check_count    = 0;
           check_alpha    = 0;
           do {
              check_num = token[p+(check_count++)] ;
              switch(check_num) {
               case '.' : check_dot++;
                          break;
               case ' 0'  :
               case ' 1'  :
               case ' 2'  :
               case ' 3'  :
               case ' 4'  :
               case ' 5'  :
               case ' 6'  :
               case ' 7'  :
               case ' 8'  :
               case ' 9'  :  break;
               case ' \0' :  break;                   // String Teminator
               default    :  check_alpha++;
              }
             }
             while(check_num != '\0');
             if( (check_alpha == 0)  && (check_dot < 2 ) ) {
              * (num_code++) + 0xF0;
              * ( ( ( float * ) mum_code)++) = atof(&token[p]);
              p = p+strlen(&token[p]+1;
             } else {
              fclose(stream);
              return(2);                              // Error!  Non  match  string
             }
            }
           }
           * (num_code)++ = '\n';                     // 1  Opcode  Terminator
          }
          while(token_num < 4);                       // loop  again  ( Not  EOF ,Math,  string)
          *num_code =  '\0' ;                         //Terminator
    //---------
     fclose(stream);
     return(0);                                       //Success
    }

    //-----------------------------------------------------------
    //Return Value  :      0  if  no token
    //                     1  if  1  token
    //                     2  if  2  token
    //                     3  if  3  token
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//                    4 if  Over                          -> stop
//                  +10 if  EOF                           -> stop
//                  +20 if  string length >  16 char      -> stop

int get_token(FILE *stream ,char *token)  {

char ch;
int string_count;

//------------ start --------------------
 do {                                       //Remove space, Tab and Newline
  ch = fgetc(stream);                       //
 }                                          //
 while( (ch == ' ')   ||                    //
        (ch == '\t')  ||                    //
        (ch == '\n') );                     //
 if(ch == EOF) return(10);                  //
//--------------- token #1 ------------
 string_count = 0;
 do  {
   if( strint_count == 16 )  {              //limit string length
     *token = '\0';                         //Terminator
     return(21);
    } else {
     string_count++;                        //if < 16 get next char
     *(token++) = ch;                       //
     ch = fgetc(stream);                    //
    }
 }
 while( (ch != ' ')  &&                     //Stop if found Space, Tab, Newline a
        (ch != '\t')  &&                    //
        (ch != '\n')  &&                    //
        (ch != EOF ) );                     //
 *(token++)  = '\0' ;                       //Terminator
 switch(ch)  {
   case EOF   : return(11) ;
   case '\n'    : return(1) ;
  }
//-----------------------------------
 do {
   ch  =  fgetc(stream) ;
 }
 while( (ch == ' ')   ||
        (ch == '\t') ) ;
 switch(ch)  {
    case EOF   : return(11) ;
 case '\n' : return(1) ;
}

//------------- token  #2 --------------
 string_count  =  0 ;                       // Like toke  #1
```

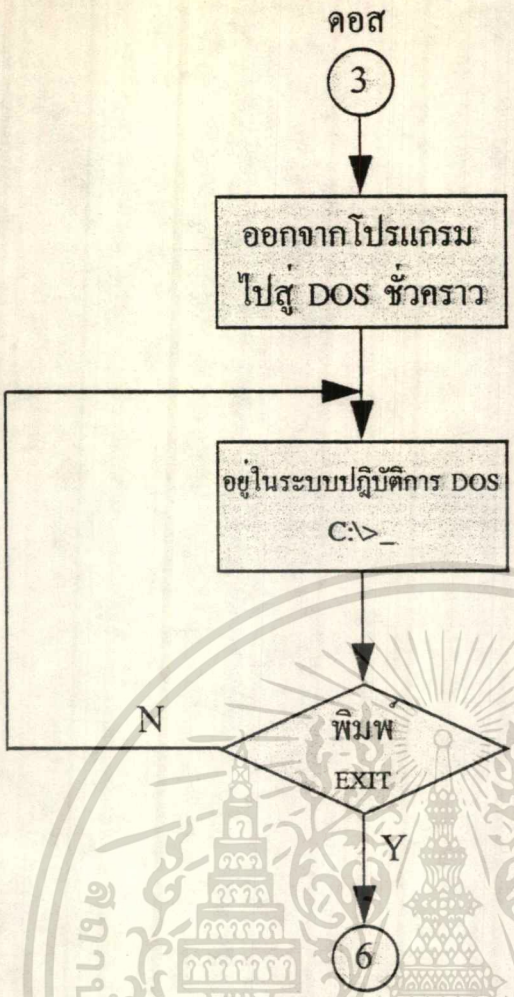
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    do {
      if( string_count == 16 ) {
        *token = '\0' ;
        return(22) ;
      } else {
        string_count++;
        *(token++) = ch;
        ch = fgetc(stream) ;
      }
    }
    while ( (ch != ' ') &&
            (ch != '\t') &&
            (ch != '\n') &&
            (ch != EOF) ) ;
    *(token++) = '\0' ;
    switch(ch) {
       case EOF  : return(12) ;
       case '\n'  : return(2);
    }
   //-------------------------------------
    do {
     ch = fgetc(stream) ;
    }
    while( (ch == ' ')  ||
           (ch == '\t'));
    switch(ch) {
      case EOF  : return(12);
      case '\n'  : return(2);
    }

   //---------- -- token #3 ----------------
    string_count = 0;                          Like token #1
    do {
      if( string_count == 16 ) {
        *token = '\0';
        return(23) ;
      } else {
       string_count++;
       *(token++) = ch;
       ch = fgetc(stream) ;
      }
    }
    while( (ch != ' ') &&
           (ch != '\t') &&
           (ch != '\n') &&
    *(token++) = '\0' ;
    switch(ch) {
      case EOF  : return(13);
      case '\n'  : return(3);
    }
```

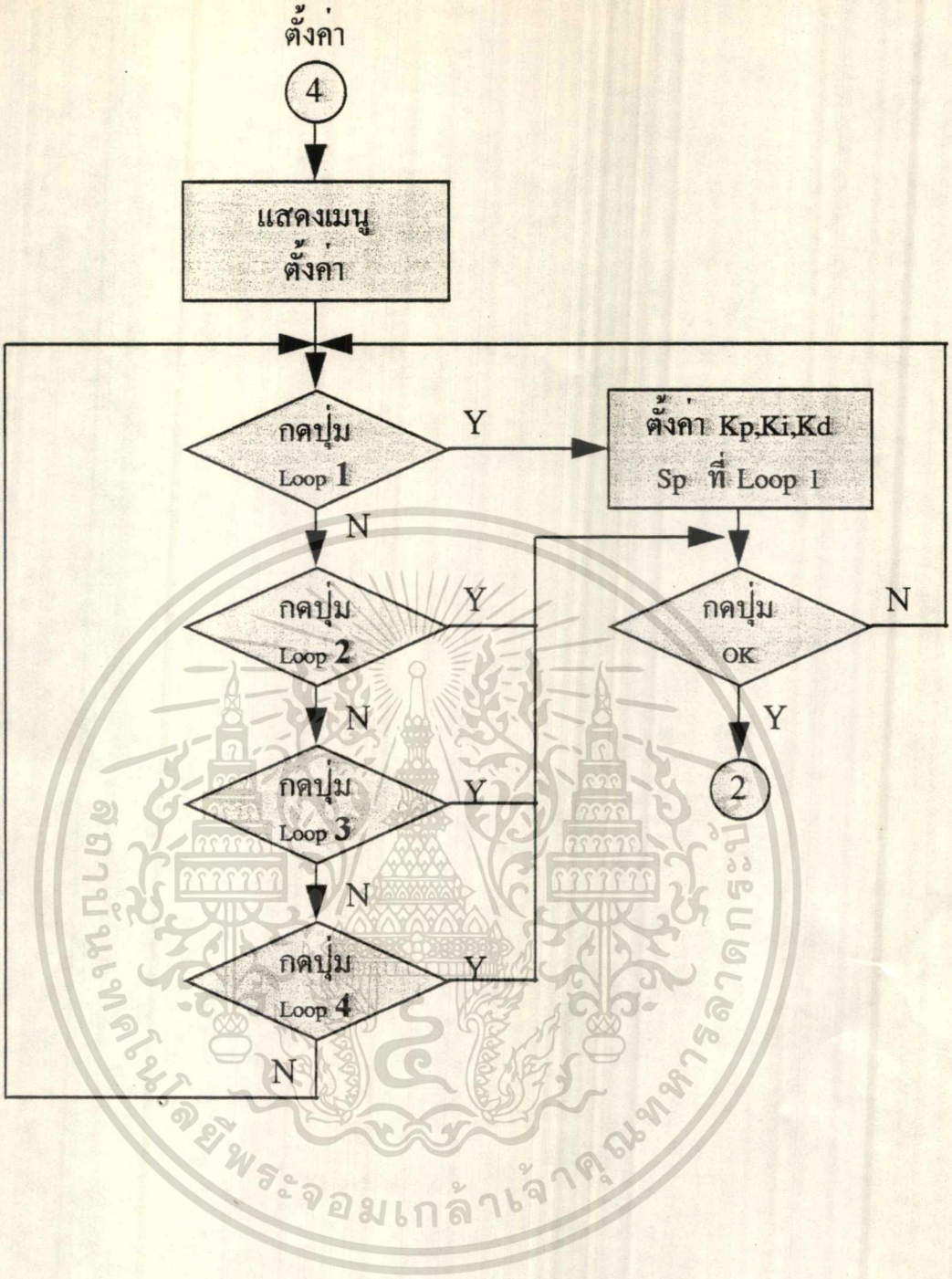
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//-------------------------------------
 do {
   ch = fgetc(stream);
 }
 while( (ch == ' ')  ||
        (ch == ;\t') ) ;
 switch(ch) {
   case EOF   : return(13) ;
   case '\n'    : return(3) ;
  }
//------------ Over ------------------------
return(4) ;

}

//-----------------------------------------
```

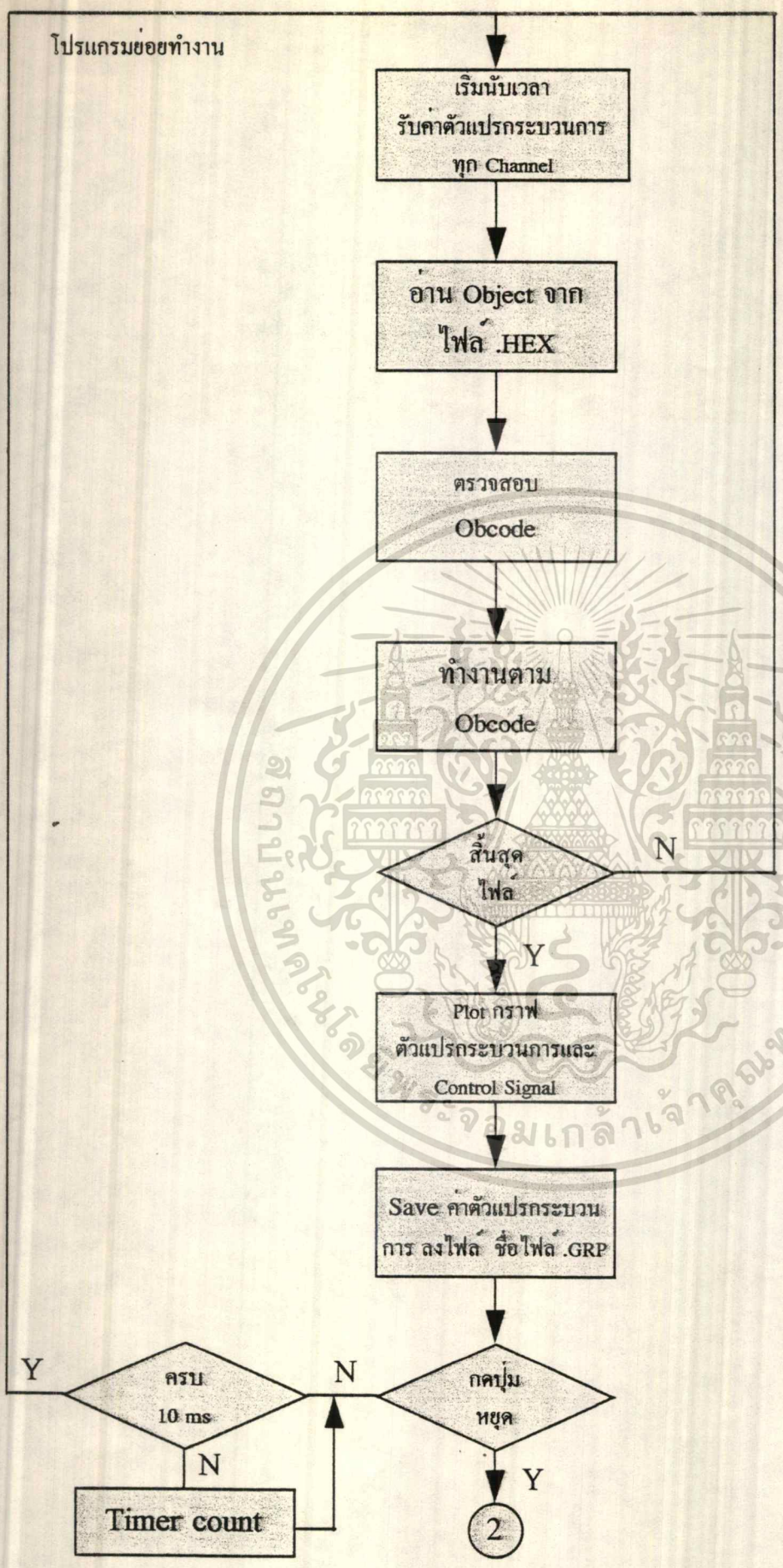
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



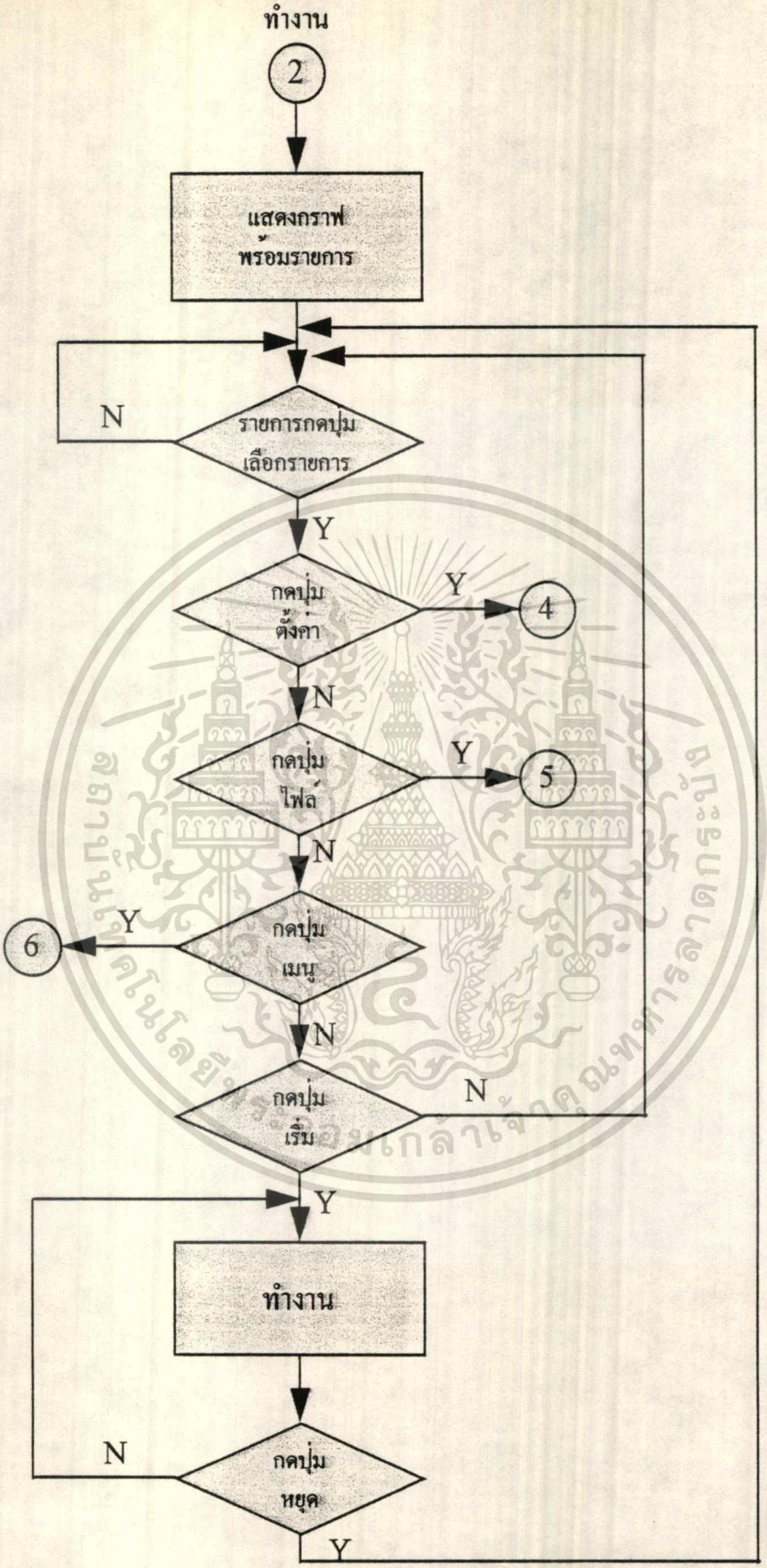
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



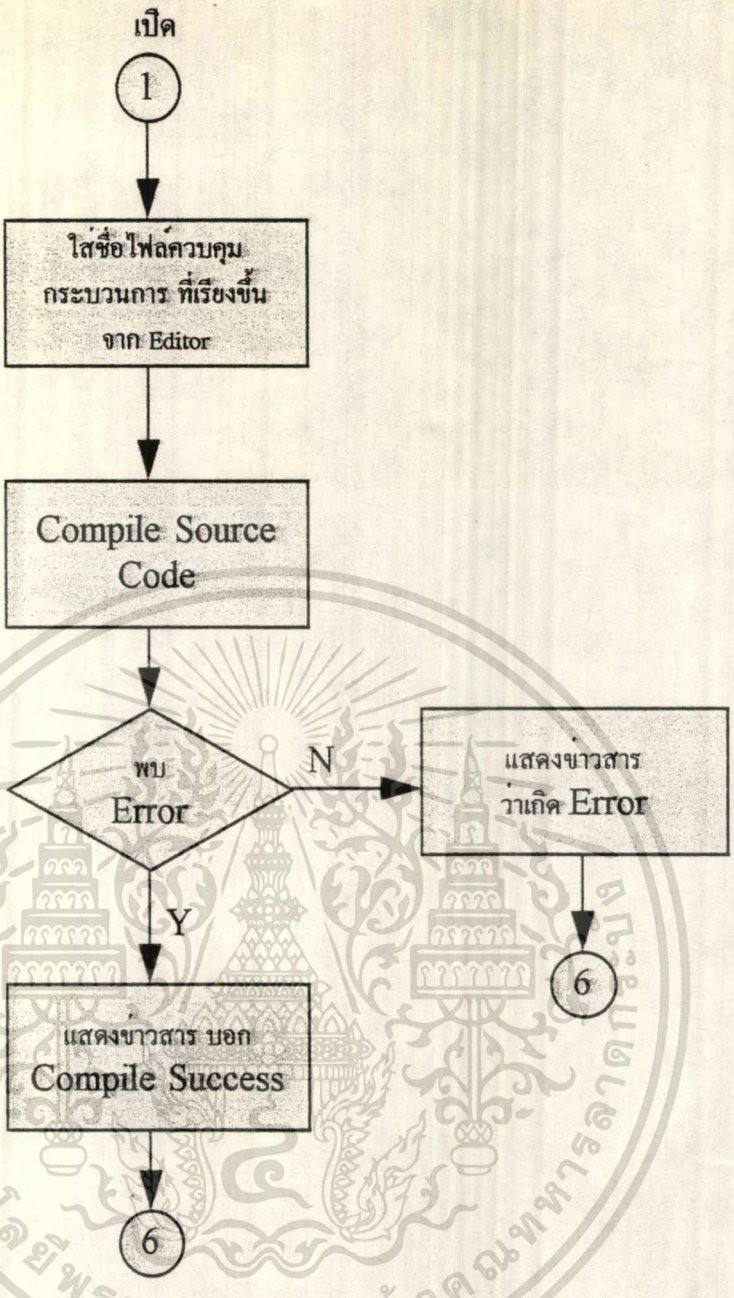
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



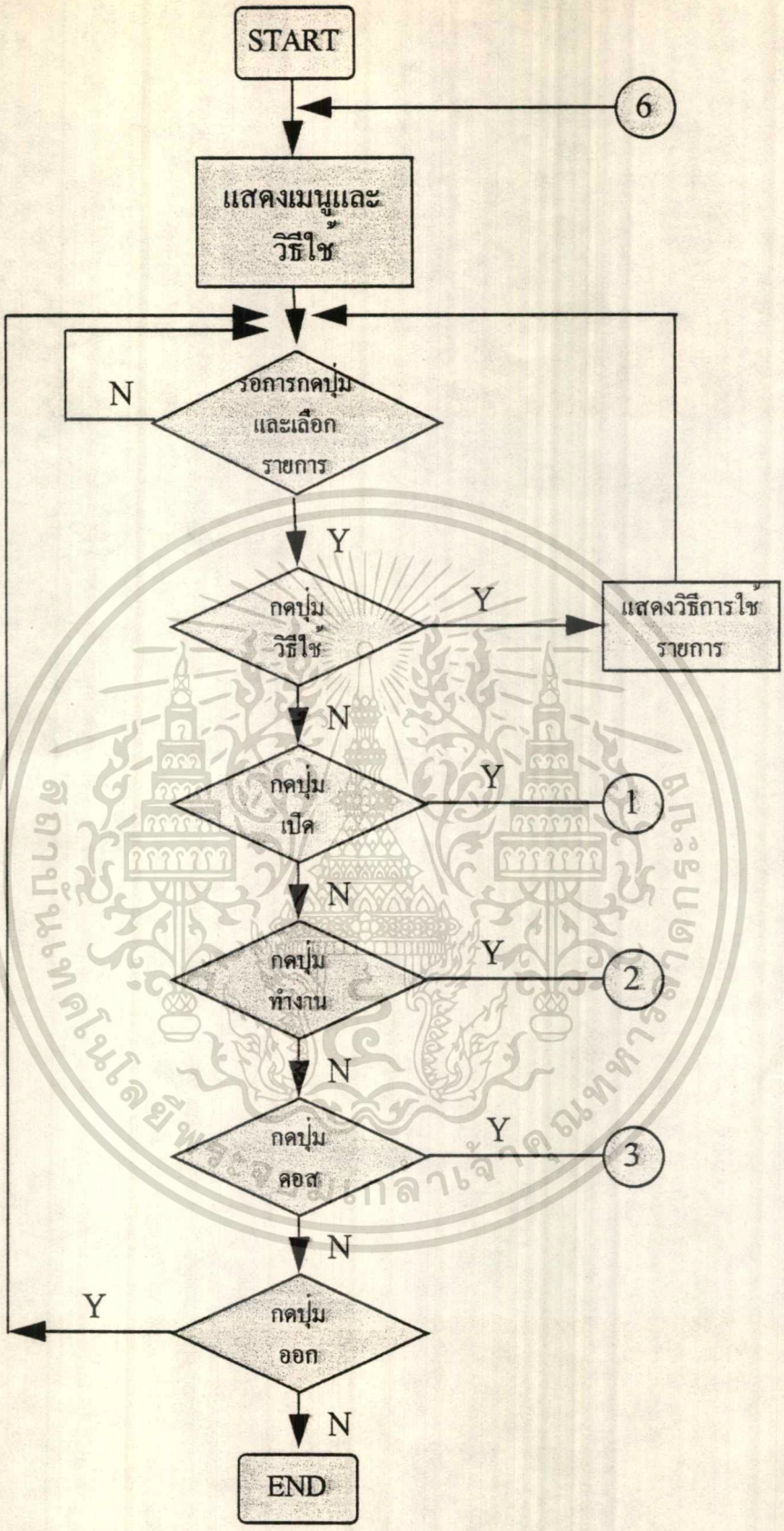
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



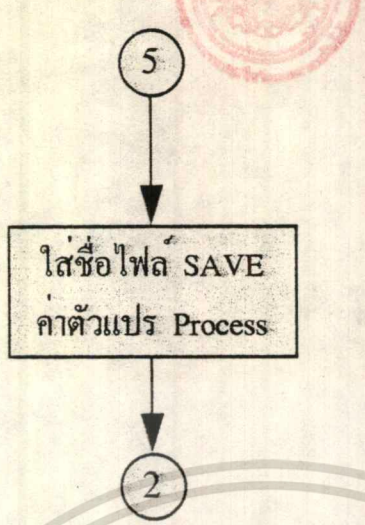
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้