เครื่องควบคุมแบบโปรแกรมซีเควนซ์ขนาด 1 บิท

ONE BIT PROGRAMMABLE LOGIC CONTROLLER

ธวัชชัย เรืองสวัสดิ์

THAWATCHAI RUANGSAWAT

อาจารย์ที่ปรึกษา

ดนุตร วิเศษกุล M.Eng.

ADVISOR

DANOOT VISESKUL M.Eng.

วิทยานิพนธ์สำหรับปริญญาวิศวกรรมศาสตรมหาบัณฑิต

สาขาวิศวกรรมไฟฟ้า

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2529

สารบาญ

บทคัดย่อ

วิทยานิพนธ์ฉบับนี้เสนอวิธีการออกแบบนำเอาไมโครโปรเซสเซอร์ขนาด 1 บิท ซึ่งใช้ในระบบควบคุมแบบง่าย ๆ ที่มีการทำงานแบบเปิดกับปิด มาพัฒนาสร้างเป็นเครื่องพีแอลซี ให้มีฟังก์ชั่นต่าง ๆ สำหรับการควบคุมแบบซีเควนซ์อย่างครบถ้วน ทำให้สามารถนำไปใช้งานที่มีลักษณะการควบคุมเป็นแบบซีเควนซ์ใด ๆ ก็ได้ โดยมีขีดความสามารถและประสิทธิภาพ เทียบเท่าเครื่องพีแอลซีที่ใช้ไมโครโปรเซสเซอร์ขนาด 8 บิททั่ว ๆ ไป จากการทดลองนำเอาพีแอลซีที่ได้ออกแบบขึ้นนี้ไปใช้ในการควบคุมกระบวนการจำลองทางอุตสาหกรรมเกี่ยวกับการผสมของเหลว ปรากฏว่าพีแอลซีสามารถทำการควบคุมได้โดยไม่มีความผิดพลาดเลย

ABSTRACT

The design and development of a simple ON-OFF function' 1 bit microprocessor which constitutes a type of PLC (Programmable Logic Controller) is presented in this thesis. All the functions of sequence control systems can be covered by this PLC. The capability and performance of the designed PLC are equivalent to PLCs based on an 8 bit microprocessor. This PLC can be used to control experimentally a mixing process model without any problem.

บทที่ 1
บทนำ

ในปัจจุบันนี้ ความเจริญก้าวหน้าทางด้านไมโครโปรเซสเซอร์ (Microprocessor) ได้ถูกนำมาพัฒนาสร้างเป็นเครื่องไมโครคอมพิวเตอร์ (Microcomputer) เพื่อใช้ในธุรกิจและในชีวิตประจำวันมากขึ้น เครื่องไมโครคอมพิวเตอร์เป็นเครื่องมือที่ช่วยอำนวยความสะดวกและรวดเร็วให้กับผู้ใช้เป็นอย่างมาก เช่น ใช้ในการเก็บข้อมูลต่าง ๆ เป็นต้น

นอกจากการพัฒนานำเอาไมโครโปรเซสเซอร์มาสร้างเป็นเครื่องไมโครคอมพิวเตอร์แล้ว ยังมีการพัฒนานำเอาไมโครโปรเซสเซอร์มาสร้างเป็นเครื่องควบคุมแบบต่าง ๆ สำหรับใช้ในงานควบคุมกระบวนการทางอุตสาหกรรม แทนการใช้อุปกรณ์ควบคุมแบบเก่า ๆ ที่มีประสิทธิภาพต่ำ

วิทยานิพนธ์ฉบับนี้จะกล่าวถึงการพัฒนานำเอาไมโครโปรเซสเซอร์ขนาด 1 บิท (1 bit Microprocessor) มาออกแบบสร้างเป็นเครื่องควบคุมแบบโปรแกรมซีเควนซ์หรือพีแอลซี (PLC:Programmable Logic Controller) ซึ่งมีขั้นตอนและวิธีการออกแบบสร้างดังได้เสนอไว้ในบทต่าง ๆ ดังนี้

ในบทที่ 2 จะอธิบายถึงวัตถุประสงค์ในการสร้างพีแอลซี และประวัติความเป็นมา บทที่ 3 อธิบายถึงโครงสร้างภายใน การทำงานของหน่วยประมวลผลกลางและคำสั่งต่าง ๆ ของหน่วยประมวลผลกลางขนาด 1 บิท บทที่ 4 อธิบายถึงวิธีการออกแบบวงจรส่วนต่าง ๆ ที่ประกอบกันเป็นพีแอลซี บทที่ 5 จะอธิบายการทำงานของวงจรหลัก (Main Unit) ของพีแอลซี บทที่ 6 จะอธิบายถึงทฤษฎีของพีแอลซีและวิธีการเขียนโปรแกรมคำสั่งต่าง ๆ ที่เป็นเงื่อนไขการทำงาน บทที่ 7 อธิบายถึงวิธีการขยายขีดความสามารถสูงสุด และประสิทธิภาพของพีแอลซีขนาด 1 บิท สำหรับบทที่ 8 จะอธิบายถึงการออกแบบ และวิธีการใช้เครื่องป้อนโปรแกรมในการเขียนโปรแกรมให้กับเครื่องพีแอลซีขนาด 1 บิท ส่วนบทที่ 9 เป็นการยกตัวอย่างนำเอาพีแอลซีไปใช้งาน

ในการควบคุมกระบวนการแบบต่าง ๆ และบทที่ 10 เป็นการสรุปผลของการออกแบบสร้างพีแอลซี และประสิทธิภาพของเครื่องพีแอลซี

บทที่ 2
วัตถุประสงค์และที่มาของพีแอลซี

ในกระบวนการควบคุมแบบ "ON" หรือ "OFF" ที่เป็นซีเควนซ์ต่อเนื่องกันนั้นในสมัยต้น ๆ จะใช้รีเลย์ (Relay) หลาย ๆ ตัวมาต่อร่วมกันเป็นวงจรควบคุมให้เครื่องจักรหรือระบบทำงานตามเงื่อนไขที่ต้องการ แต่วงจรรีเลย์มีข้อเสียหลาย ๆ อย่าง เช่นมีขนาดใหญ่ สิ้นเปลืองพลังงานสูง ราคาแพง และที่สำคัญคือไม่สามารถเปลี่ยนแปลงเงื่อนไขการทำงานของระบบได้ เพราะการเปลี่ยนแปลงเงื่อนไขการทำงานต้องเปลี่ยนแปลงที่วงจรของรีเลย์ จึงเป็นการยุ่งยากและสิ้นเปลืองค่าใช้จ่ายสูง

ต่อมาความเจริญก้าวหน้าทางด้านสารกึ่งตัวนำ (Solid State Device) ได้แพร่หลายมากขึ้น ทำให้มีการเปลี่ยนแปลงระบบควบคุมซีเควนซ์แบบเก่าที่ใช้รีเลย์ มาเป็นพวกเกท (Gate) มากขึ้นและมีข้อดีคือ มีขนาดเล็กลง สิ้นเปลืองพลังงานน้อยลง ราคาถูก แต่ก็ยังไม่สามารถเปลี่ยนแปลงเงื่อนไขของการทำงานได้โดยตรง การเปลี่ยนแปลงเงื่อนไขการทำงาน จะต้องเปลี่ยนแปลงที่วงจร และการตรวจสอบหาจุดเสียยังยุ่งยากอยู่

เมื่อวิวัฒนาการทางด้านไมโครโปรเซสเซอร์ (Microprocessor) เจริญก้าวหน้ามากขึ้นจึงมีการพัฒนานำเอาไมโครโปรเซสเซอร์มาสร้างเป็นพีแอลซี(PLC:Programmable Logic Controller) ซึ่งมีข้อดีกว่าระบบที่ใช้วงจรรีเลย์ หรือวงจรเกท เป็นตัวควบคุม เช่นขนาดของเครื่องเล็กลง การสิ้นเปลืองพลังงานต่ำ การตรวจซ่อมกระทำได้ง่าย และที่สำคัญคือสามารถเปลี่ยนแปลงเงื่อนไขการทำงานได้ง่ายโดยการเปลี่ยนที่โปรแกรมคำสั่ง พีแอลซีชนิดนี้เป็นเครื่องควบคุมซีเควนซ์แบบเอนกประสงค์ คือจะนำไปใช้กับกระบวนการควบคุมแบบ "ON" หรือ "OFF" แบบใด ๆ ก็ได้ เพราะการทำงานขึ้นอยู่กับคำสั่งที่ป้อนเป็นโปรแกรมให้กับเครื่อง ตัวอย่างของการควบคุมแบบซีเควนซ์โดยใช้พีแอลซีเป็นเครื่องควบคุม เช่น กระบวนการผสมของเหลว การขนถ่ายวัสดุ การผลิตชิ้นส่วน ระบบล้างรถอัตโนมัติ หรือระบบการทอแหอวน เป็นต้น

ในปัจจุบันนี้เครื่องพีแอลซี ที่มีการสร้างขึ้นมาสำหรับการใช้งานในระบบอุตสาหกรรมส่วนมากจะใช้ซีพียู (CPU : Central Processing Unit) ขนาด 8 บิทหรือ 16 บิทเป็นหน่วยประมวลผลกลาง และในบางเครื่องสามารถทำฟังก์ชั่นทางคณิตศาสตร์ได้ด้วย แต่ในการควบคุมกระบวนการแบบซีเควนซ์ที่ไม่ซับซ้อนนั้น จะเห็นว่าไม่จำเป็นต้องใช้ฟังก์ชั่นทางคณิตศาสตร์เลย พีแอลซีที่มีขายตามท้องตลาดในปัจจุบันถึงแม้ว่าจะมีการพัฒนาให้มีข้อดีต่าง ๆ มาก แต่ก็ยังมีข้อเสียบางอย่าง เช่น ราคาสูง และจำนวนอินพุทและเอาท์พุทยังไม่สูงมากนัก โดยพอจะจำแนกขนาดของอินพุทและเอาท์พุทของพีแอลซีได้ 3 ขนาดดังนี้คือ

1. พีแอลซีขนาดเล็ก พีแอลซีชนิดนี้จะไม่มีฟังก์ชั่นทางคณิตศาสตร์เลย จะมีเพียงคำสั่งทางลอจิก คำสั่งการตั้งเวลา คำสั่งการนับจำนวน และมีจำนวนของอินพุทเอาท์พุทมาตรฐานคือ 16 อินพุท 8 เอาท์พุท โดยสามารถขยายจำนวนอินพุทเอาท์พุทได้สูงสุดถึง 32 อินพุท 24 เอาท์พุท ซึ่งจะเหมาะกับงานควบคุมขนาดเล็ก ๆ เท่านั้น ราคาของพีแอลซีขนาดเล็กที่มี 16 อินพุท และ 8 เอาท์พุทนี้เริ่มจากราคา 15,000 บาทขึ้นไป

2. พีแอลซีขนาดกลาง พีแอลซีชนิดนี้จะไม่มีฟังก์ชั่นทางคณิตศาสตร์เช่นเดียวกับพีแอลซีขนาดเล็ก มีจำนวนอินพุทเอาท์พุทมาตรฐาน 16 อินพุท 8 เอาท์พุทเหมือนกัน แต่สามารถขยายได้สูงสุดถึง 128 อินพุทและ 128 เอาท์พุท การโปรแกรมคำสั่งสามารถโปรแกรมได้สูงสุดประมาณ 2,000 คำสั่ง โดยราคาเริ่มต้นของพีแอลซีขนาดกลางที่มี 16 อินพุทและ 8 เอาท์พุทนี้เริ่มจากราคา 25,000 บาท และถ้าเพิ่มจำนวนอินพุทเอาท์พุทอีก 16 อินพุท 16 เอาท์พุท จะต้องเพิ่มค่าใช้จ่ายหน่วยละประมาณ 7,000 ถึง 8,000 บาท จะเห็นว่าถ้าขยายจำนวนของอินพุทและเอาท์พุทของเครื่องสูงสุดคือ 128 อินพุท และ 128 เอาท์พุทแล้ว ราคาของพีแอลซีทั้งชุดจะประมาณ 150,000 บาทขึ้นไป

3. พีแอลซีขนาดใหญ่ พีแอลซีชนิดนี้จะมีฟังก์ชั่นทางคณิตศาสตร์ เช่น บวก ลบ คูณ และ หาร เป็นต้น บางครั้งอาจเรียกพีแอลซีชนิดนี้ว่าเป็นพีซี (PC:Programmable Controller) ก็ได้ จำนวนอินพุทและเอาท์พุทของพีแอลซีขนาดใหญ่นี้มีได้สูงสุดถึง 1024 อินพุทและ 1024 เอาท์พุท หน่วยความจำ (Memory) สำหรับเก็บโปรแกรมคำสั่งนี้จะมีสูงถึง 8 กิโลเวิร์ด (Kilo-words) หรือประมาณ 8,000 คำสั่ง ราคาของเครื่องพี-

แอลซีขนาดใหญ่ นี้จะประมาณ 700,000 บาทขึ้นไป

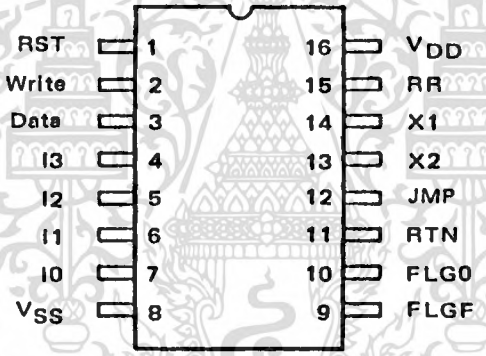
จะเห็นได้ว่าพีแอลซีทั้ง 3 ขนาดที่กล่าวมานี้ราคาต่ำสุดคือ 15,000 บาท และไม่สามารถสร้างขึ้นมาใช้งานเองภายในประเทศไทยได้ ต้องสั่งซื้อจากต่างประเทศเข้ามา

วิทยานิพนธ์ฉบับนี้จะกล่าวถึงการสร้างพีแอลซีโดยใช้หน่วยประมวลผลกลางขนาด 1 บิท (1 bit CPU) ซึ่งมีขนาดของอินพุทและเอาท์พุทสูงเทียบเท่ากับพีแอลซีขนาดใหญ่คือ มีอินพุทเอาท์พุทสูงถึง 1920 อินพุทและ 1280 เอาท์พุท มีหน่วยความจำซึ่งขยายได้สูงถึง 120 กิโลเวิร์ดหรือ 120,000 คำสั่ง พีแอลซีขนาด 1 บิทนี้สามารถจะสร้างขึ้นมาใช้งานเองได้ภายในประเทศ และอุปกรณ์ประกอบต่าง ๆ จะหาซื้อได้ง่าย ต้นทุนในการสร้างต่ำราคาของระบบมาตรฐานที่มี 16 อินพุท และ 8 เอาท์พุทประมาณ 3,500 บาท และราคาของการเพิ่มอินพุทเอาท์พุทประมาณ 400 บาทต่อ 16 อินพุทและ16 เอาท์พุท เมื่อขยายขีดความสามารถสูงสุดแล้วราคาของทั้งชุดจะประมาณ 50,000 บาท ซึ่งเมื่อเทียบกับพีแอลซีขนาดใหญ่แล้วราคายังต่ำกว่ามาก ๆ

บทที่ 3
โครงสร้างและการทำงานของหน่วยประมวลผลกลางขนาด 1 บิท

3.1 โครงสร้างของหน่วยประมวลผลกลางขนาด 1 บิท (1 bit CPU)

หน่วยประมวลผลกลางขนาด 1 บิทที่ใช้ในระบบ (System) เป็นไอซีชิพเดี่ยว (IC single chip) เบอร์ MC14500B ผลิตโดยบริษัทโมโตโรลา วงจรรวมภายในไอซีชิพเดี่ยวนี้เป็นแบบสแตติกส์ซีมอส (Static CMOS) ไอซีชิพเดี่ยวนี้มีทั้งหมด 16 ขา และหน้าที่ของขาแต่ละขาแสดงดังในรูปที่ 3.1



รูปที่ 3.1 โครงสร้างภายนอกของไอซีชิพเดี่ยว

ขาที่ 1 เป็นขารีเซท (Reset) วงจรภายในของหน่วยประมวลผลกลางจะมีลักษณะการทำงานดังนี้ เมื่อขา 1 มีสภาวะเป็น 0 (Low) จะเป็นการเริ่มต้นให้หน่วยประมวลผลกลางทำการประมวลผล (RUN) และเมื่อขา 1 มีสภาวะเป็น 1 (High) ก็จะเป็นการหยุด (Reset) การประมวลผลของหน่วยประมวลผลกลาง

ขาที่ 2 ของหน่วยประมวลผลกลาง มีหน้าที่ส่งสัญญาณอ่าน (Read) ไปยังหน่วยอินพุท (Input) หรือส่งสัญญาณเขียน (Write) ไปยังหน่วยเอาท์พุท (Output)

ขาที่ 3 เป็นขาข้อมูล (Data) แบบ 2 ทิศทาง (Bidirectional) คือใช้รับข้อมูลจากอินพุทเข้ามาประมวลผลยังหน่วยประมวลผลกลาง หรือส่งข้อมูลจากหน่วยประ-

มวลผลกลางออกไปยังเอาท์พุท

ขาที่ 4,5,6 และ 7 เป็นขาคำสั่ง (Instruction) ของหน่วยประมวลผลกลางมีขนาด 4 บิท โดยจะรับรหัส (Code) ขนาด 4 บิทนี้มาทำการถอดรหัส (Decode) ภายในหน่วยประมวลผลกลาง ซึ่งจะได้คำสั่งทั้งหมด 16 คำสั่ง

ขาที่ 8 เป็นขากราวด์ (Ground) ของไอซี ส่วนขาที่ 16 นั้นเป็นขาสำหรับรับไฟ (Supply) เข้ามาเลี้ยงวงจรภายในหน่วยประมวลผลกลาง สามารถใช้ไฟได้กว้างตั้งแต่ 5 ถึง 15 โวลท์ดีซี (5-15Vdc)

ขาที่ 9,10,11 และ 12 เป็นขาแฟลก (Flags) โดยจะให้เอาท์พุทออกมาเป็นช่วง (Pulse) ตามคำสั่งที่รับเข้ามาทางขาคำสั่ง หน่วยประมวลผลกลางจะเป็นหน่วยควบคุมแฟลกต่าง ๆ อีกทีหนึ่ง

ขาที่ 13 และขาที่ 14 เป็นขาควบคุมสัญญาณนาฬิกา (Clock) โดยที่ขา 13 จะรับสัญญาณความถี่จากภายนอกเข้ามา และเมื่อหน่วยประมวลผลกลางเริ่มทำงาน จะส่งผ่านสัญญาณนาฬิกาจากขาที่ 13 มายังขาที่ 14 และจะใช้สัญญาณนาฬิกาจากขาที่ 13 นี้สำหรับการประมวลผลภายในหน่วยประมวลผลกลางด้วย หรือถ้าจะใช้สัญญาณนาฬิกาจากการผลิตโดยหน่วยประมวลผลกลางเองก็สามารถกระทำได้ เพราะภายในหน่วยประมวลผลกลางมีวงจรผลิตความถี่ (Oscillator) ภายใน ความถี่ของสัญญาณนาฬิกาสามารถกำหนดจากค่าความต้านทาน (Resistor) ภายนอก ที่นำมาต่อยังขาที่ 13 และขาที่ 14

ขาที่ 15 เป็นขาที่ใช้แสดงค่าสภาวะรีจิสเตอร์ผลลัพธ์ (Result Register) ซึ่งได้จากการประมวลผลของหน่วยประมวลผลกลาง

3.2 คำสั่ง (Instruction)

คำสั่งทั้งหมดของหน่วยประมวลผลกลางมี 16 คำสั่ง ได้มาจากรหัสขนาด 4 บิท ลักษณะของรหัส และความหมายดังในรูปที่ 3.2

1. คำสั่ง NOPO เมื่อหน่วยประมวลผลกลางได้รับคำสั่ง NOPO จะส่งสัญญาณ 1 ลูกออกไปที่ขา 10 และไม่มีการเปลี่ยนแปลงค่าสภาวะของรีจิสเตอร์ภายในหน่วยประ-

มวลผลกลาง

Instruction Code		Mnemonic	Action
0	0000	NOP0	No Change in Registers.FLG0←⎍
1	0001	LD	Load Result Register.Data→RR
2	0010	LDC	Load Complement $\overline{\text{Data}}$→RR
3	0011	AND	Logical AND. RR.D→RR
4	0100	ANDC	Logical AND Compl. RR.$\overline{D}$→RR
5	0101	OR	Logical OR. RR+D→RR
6	0110	ORC	Logical OR Compl. RR+$\overline{D}$→RR
7	0111	XNOR	Exclusive NOR.If RR=D,RR←1
8	1000	STO	Store. RR→D Pin,Write←1
9	1001	STOC	Store Compl.$\overline{RR}$→D Pin,Write←1
A	1010	IEN	Input Enable.D→IEN Register.
B	1011	OEN	Output Enable.D→OEN Register.
C	1100	JMP	Jump. JMP Flag←⎍
D	1101	RTN	Return. RTN Flag←⎍
E	1110	SKZ	Skip Next Instruction if RR=0
F	1111	NOPF	No Change in Registers.FLGF←⎍

รูปที่ 3.2 คำสั่งและความหมายของคำสั่ง

2. คำสั่ง LD คำสั่งนี้จะเป็นการนำค่าสภาวะจากอินพุทใด ๆ เข้ามาเก็บไว้ในรีจิสเตอร์ผลลัพธ์เพื่อรอการประมวลผลต่อไป

3. คำสั่ง LDC คำสั่งนี้จะเป็นการนำค่าสภาวะตรงข้ามจากอินพุทใด ๆ เข้ามาเก็บไว้ในรีจิสเตอร์ผลลัพธ์เพื่อรอการประมวลผลต่อไป

4. คำสั่ง AND เป็นการนำค่าสภาวะจากอินพุทใด ๆ เข้ามาทำการ AND กับค่าสภาวะที่อยู่ภายในรีจิสเตอร์ผลลัพธ์ และผลลัพธ์ที่ได้จากการ AND กันนี้จะถูกเก็บไว้ในรีจิสเตอร์ผลลัพธ์แทนผลลัพธ์เดิม

5. คำสั่ง ANDC เป็นการนำค่าสภาวะตรงข้ามจากอินพุทใด ๆ เข้ามาทำการ AND กับค่าสภาวะที่อยู่ในรีจิสเตอร์ผลลัพธ์ และผลลัพธ์ที่ได้จากการ AND นี้จะถูกเก็บไว้ในรีจิสเตอร์ผลลัพธ์แทนผลลัพธ์เดิม

6. คำสั่ง OR เป็นการนำค่าสภาวะจากอินพุทใด ๆ เข้ามาทำการ OR กับค่าสภาวะที่อยู่ในรีจิสเตอร์ผลลัพธ์ ผลลัพธ์ที่ได้จากการ OR กันนี้จะถูกเก็บไว้ในรีจิสเตอร์ผลลัพธ์แทนผลลัพธ์เดิม

7. คำสั่ง ORC เป็นการนำค่าสภาวะตรงข้ามจากอินพุทใด ๆ เข้ามาทำการ OR กับค่าสภาวะที่อยู่ในรีจิสเตอร์ผลลัพธ์ และผลลัพธ์ที่ได้จากการ OR นี้จะถูกเก็บไว้ในรีจิสเตอร์ผลลัพธ์แทนผลลัพธ์เดิม

8. คำสั่ง XNOR เป็นการนำค่าสภาวะจากอินพุทใด ๆ เข้ามาทำการ XNOR กับค่าสภาวะที่อยู่ในรีจิสเตอร์ผลลัพธ์ และผลลัพธ์ที่ได้จากการ XNOR นี้ถูกเก็บไว้ในรีจิสเตอร์ผลลัพธ์แทนผลลัพธ์เดิม

9. คำสั่ง STO นี้เป็นการนำค่าสภาวะของผลลัพธ์ที่อยู่ในรีจิสเตอร์ผลลัพธ์ส่งออกไปยังเอาท์พุทใด ๆ และพร้อมกันนั้นก็จะส่งสัญญาณ 1 ลูกออกไปยังขา 2

10. คำสั่ง STOC เป็นการนำค่าสภาวะตรงข้ามของผลลัพธ์ที่อยู่ในรีจิสเตอร์ผลลัพธ์ส่งออกไปยังเอาท์พุทใด ๆ และพร้อมกันนั้นก็จะส่งสัญญาณ 1 ลูกออกไปยังขา 2

11. คำสั่ง IEN นี้เป็นการนำค่าสภาวะจากอินพุทใด ๆ เข้ามาเก็บไว้ในรีจิสเตอร์ IEN (Input Enable Register)

12. คำสั่ง OEN นี้เป็นการนำค่าสภาวะจากอินพุทใด ๆ เข้ามาเก็บไว้ในรีจิสเตอร์ OEN (Output Enable Register)

13. คำสั่ง JMP เมื่อหน่วยประมวลผลกลางได้รับคำสั่ง JMP จะส่งสัญญาณ 1 ลูกออกไปที่ขา 12 และค่าสภาวะของรีจิสเตอร์ภายในหน่วยประมวลผลกลางจะไม่มีการ

เปลี่ยนแปลง .

14. คำสั่ง RTN เมื่อหน่วยประมวลผลกลางได้รับคำสั่ง RTN จะส่งสัญญาณ 1 ลูกออกไปที่ขา 11 และค่าสภาวะของรีจิสเตอร์ภายในหน่วยประมวลผลกลาง จะไม่มีการเปลี่ยนแปลง

15. คำสั่ง SKZ เมื่อหน่วยประมวลผลกลางได้รับคำสั่ง SKZ จะตรวจสอบค่าสภาวะของรีจิสเตอร์ผลลัพธ์ ถ้าค่าสภาวะในรีจิสเตอร์ผลลัพธ์เป็น 0 (Low) ในคำสั่งถัดมาอีก 1 คำสั่ง หน่วยประมวลผลกลางจะไม่รับรู้ (Instructions Ignored)

16. คำสั่ง NOPF เมื่อหน่วยประมวลผลกลางได้รับคำสั่ง NOPF จะส่งสัญญาณ 1 ลูกออกไปที่ขา 9 และค่าสภาวะของรีจิสเตอร์ภายในของหน่วยประมวลผลกลางจะไม่มีการเปลี่ยนแปลง

จากคำสั่งทั้งหมด 16 คำสั่งจะมีอยู่ 4 คำสั่ง คือคำสั่ง NOPO,NOPF,JMP และ RTN เมื่อหน่วยประมวลผลกลางได้รับคำสั่งใดคำสั่งหนึ่งในสี่คำสั่งนี้จะส่งเอาท์พุท 1 ลูกไปยังขาเอาท์พุทของคำสั่งนั้น ๆ และรีจิสเตอร์ภายในหน่วยประมวลผลกลางจะไม่มีการเปลี่ยนแปลงค่าสภาวะ ดังนั้นขาสัญญาณต่าง ๆ นี้จะถูกนำไปใช้เซท (Set) อุปกรณ์ภายนอกตามต้องการได้ดังนี้

1. คำสั่ง JMP จะถูกนำไปใช้เป็นคำสั่ง SET โดยนำสัญญาณที่ได้จากขา 12 (ขา JMP) นี้ไปตั้งค่าของเวลาให้กับตัวตั้งเวลา (Timer) และตั้งค่าของการนับให้กับตัวนับ (Counter)

2. คำสั่ง NOPO สัญญาณจากขา 10 (Flag 0) ซึ่งได้จากคำสั่ง NOPO นี้จะถูกนำไปใช้แสดงการผิด (Error) ของส่วนโปรแกรม (Program) และหยุดการทำงานของหน่วยประมวลผลกลาง

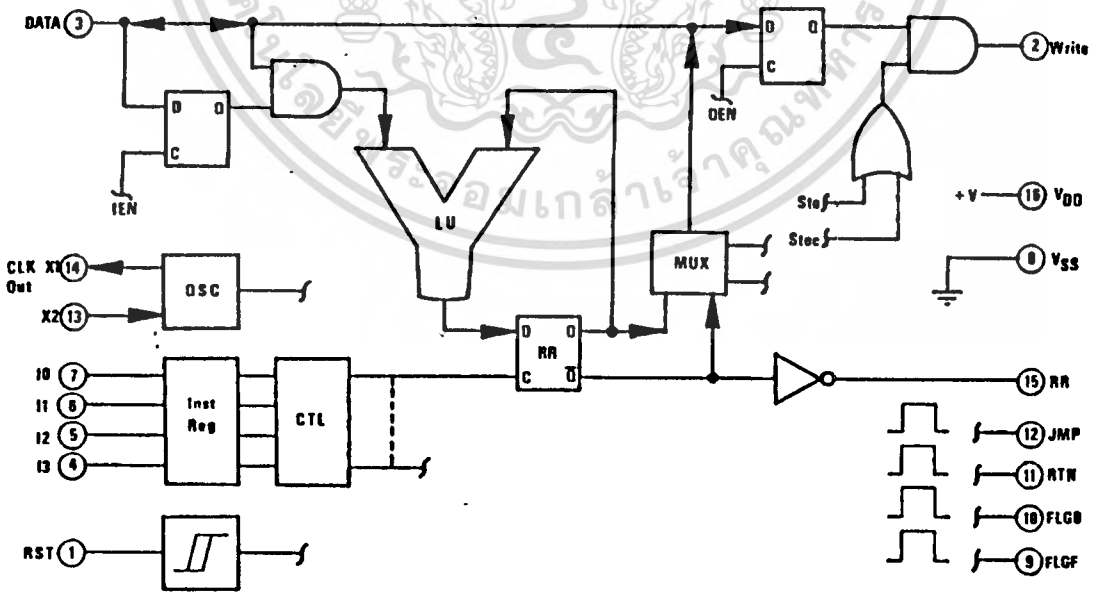
3. คำสั่ง NOPF นี้จะถูกนำไปใช้เป็นคำสั่งสิ้นสุดของโปรแกรม (END) เมื่อหน่วยประมวลผลกลางได้รับคำสั่ง NOPF จะส่งสัญญาณ 1 ลูกออกไปที่ขา 9 และสัญญาณที่ได้จากขา 9 นี้จะถูกนำไปรีเซทโปรแกรมให้กลับไปเริ่มต้นทำงานใหม่

3.3 โครงสร้างภายในและการทำงานของหน่วยประมวลผลกลางขนาด 1 บิท

โครงสร้างภายในของหน่วยประมวลผลกลางขนาด 1 บิท ดังแสดงในรูปที่ 3.3 ซึ่งมีส่วนประกอบและการทำงานดังนี้

การทำงานของหน่วยประมวลผลกลาง ในการกระทำคำสั่ง 1 คำสั่ง จะต้องใช้สัญญาณนาฬิกา 1 ลูกเสมอ เมื่อสัญญาณนาฬิกาที่ขา 14 (X1) เป็น 1 (High) หน่วยประมวลผลกลางจะรับคำสั่งขนาด 4 บิทที่เข้ามาทางขา 4, 5, 6 และ 7 เข้ามาเก็บไว้ในรีจิสเตอร์คำสั่ง (Instruction Register) และจะทำการแปลคำสั่งในรีจิสเตอร์คำสั่งที่ได้นั้นส่งต่อไปยังส่วนควบคุมทางลอจิก (Control Logic)

เมื่อสัญญาณนาฬิกาเป็น 0 (Low) หน่วยประมวลผลกลางจะรับข้อมูลเข้ามาทางขา 3 มายังอินพุทของลอจิกยูนิต (Logic Unit) เพื่อทำการประมวลผลทางด้านลอจิกกับข้อมูลในรีจิสเตอร์ผลลัพธ์ ตามคำสั่งที่อยู่ในรีจิสเตอร์คำสั่ง หรืออาจเป็นการส่งข้อมูลจากรีจิสเตอร์ผลลัพธ์ออกไปทางขา 3 ตามคำสั่งที่อยู่ในรีจิสเตอร์คำสั่งก็ได้ ขึ้นอยู่กับโปรแกรม จะเห็นได้ว่าที่ขา 3 ซึ่งเป็นขาข้อมูลเป็นแบบสองทิศทาง



รูปที่ 3.3 โครงสร้างภายในหน่วยประมวลผลกลาง

ลอจิกยูนิตจะเป็นเสมือนเกท (Gate) ที่มีสองอินพุท และหนึ่งเอาท์พุท โดยที่เอาท์พุทของเกทจะส่งผลไปเก็บไว้ที่รีจิสเตอร์ผลลัพธ์ ส่วนอินพุทของเกทนั้นต่ออยู่กับเอาท์พุทของรีจิสเตอร์ผลลัพธ์หนึ่งอินพุท ส่วนอีกหนึ่งอินพุทจะรับข้อมูลจากขา 3 เข้ามา การทำงานของลอจิกยูนิตนี้จะเป็นเกทชนิดใดก็ขึ้นอยู่กับส่วนควบคุมทางลอจิก

รีจิสเตอร์ IEN ทำหน้าที่เป็นส่วนควบคุมข้อมูลที่เข้ามาทางขา 3 ถ้าค่าสภาวะของรีจิสเตอร์ IEN เป็น 1 (High) ข้อมูลที่เข้ามายังหน่วยประมวลผลกลางจะผ่านเข้ามายังลอจิกยูนิตได้ แต่ถ้าค่าสภาวะของรีจิสเตอร์ IEN เป็น 0 (Low) ข้อมูลจะไม่สามารถผ่านเข้าไปยังส่วนของลอจิกได้

ในการส่งข้อมูลจากรีจิสเตอร์ผลลัพธ์ออกไปทางขา 3 นั้นจะมีสัญญาณควบคุมหรือสัญญาณเขียน (Write) ออกไปที่ขา 2 ด้วย โดยจะมีรีจิสเตอร์ OEN เป็นส่วนควบคุมสัญญาณเขียนอีกทีหนึ่ง ถ้าค่าสภาวะภายในรีจิสเตอร์ OEN เป็น 1 (High) สัญญาณเขียนนี้จะถูกส่งจากภายในหน่วยประมวลผลกลางผ่านออกไปยังขา 2 ได้ และถ้าค่าสภาวะนี้เป็น 0 (Low) สัญญาณเขียนจะไม่สามารถผ่านออกไปได้

รีจิสเตอร์ทั้ง 3 นี้ คือรีจิสเตอร์ IEN,OEN และรีจิสเตอร์ผลลัพธ์นี้ จะมีขนาด 1 บิท เป็นแบบ D Flip-Flop โดยข้อมูลจะเข้าทางขา D และจะมีขา C (Clock) ไว้ใช้สำหรับควบคุมอีกทีหนึ่ง เมื่อสัญญาณควบคุมถูกส่งมาจากส่วนควบคุมลอจิกมาเข้าที่ขา C ของ D Flip-Flop ถ้า D Flip-Flop ได้รับสัญญาณควบคุม ข้อมูลจากขา D จะไปปรากฏทางด้านเอาท์พุทของ D Flip-Flop และค่านี้จะถูกเก็บให้คงสภาวะอยู่จนกว่าจะมีการเปลี่ยนแปลงของสัญญาณควบคุมใหม่

ส่วนเลือกผลลัพธ์ (Multiplex) จากรีจิสเตอร์ผลลัพธ์นี้ จะถูกควบคุมจากส่วนควบคุมทางลอจิกว่าให้เลือกเอาท์พุทจาก Q หรือ $\bar{Q}$ ซึ่งมีค่าสภาวะตรงข้ามกับ Q ของรีจิสเตอร์ผลลัพธ์ เพื่อส่งออกไปที่ขา 3 อีกทีหนึ่ง

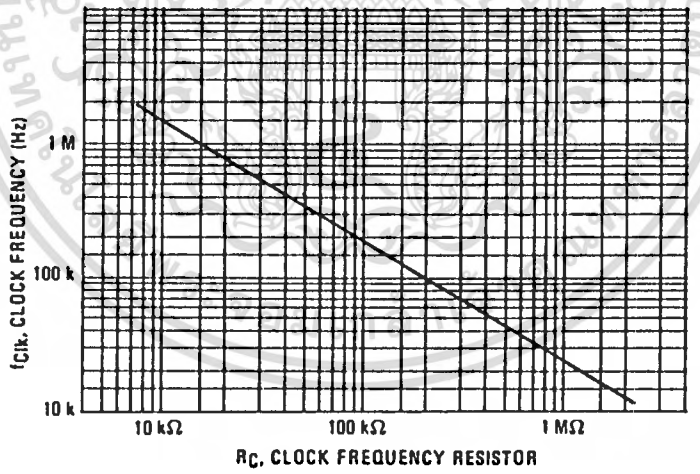
เอาท์พุทที่ปรากฏที่ขา 15 นี้เป็นค่าสภาวะของรีจิสเตอร์ผลลัพธ์ โดยจะเปลี่ยนแปลงตามค่าสภาวะของรีจิสเตอร์ผลลัพธ์

เอาท์พุทที่ได้จากขา 9, 10, 11 และ 12 นี้จะถูกส่งมาจากส่วนควบคุมทางลอจิก

เมื่อส่วนควบคุมทางลอจิกได้รับคำสั่งจากรีจิสเตอร์คำสั่งจะทำการแปลคำสั่ง และจะส่งสัญญาณ 1 ลูกออกมายังขาคำสั่งต่าง ๆ เหล่านี้ตามคำสั่งที่ได้รับมา

ส่วนขา 1 ซึ่งเป็นขารีเซท จะเป็นขารับสัญญาณจากภายนอกเพื่อมาควบคุมการทำงานภายในหน่วยประมวลผลกลาง เมื่อขาที่ 1 มีค่าสภาวะเป็น 0 (Low) หน่วยประมวลผลกลางจะเริ่มทำงาน (RUN) และถ้าเป็น 1 (High) หน่วยประมวลผลกลางจะหยุดการทำงาน (STOP)

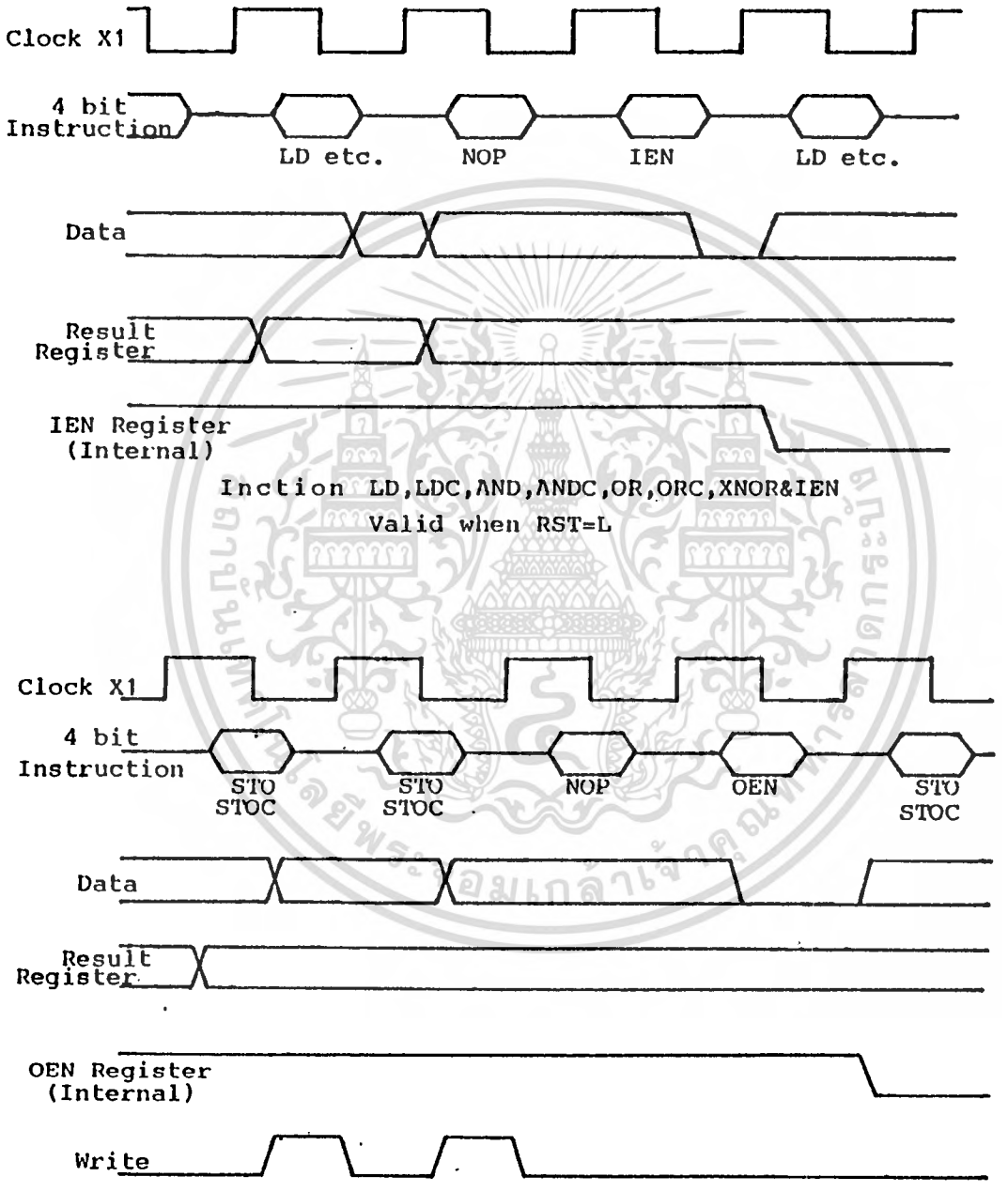
วงจรผลิตความถี่สัญญาณนาฬิกาภายในหน่วยประมวลผลกลาง (CPU) จะเริ่มทำงานเมื่อขารีเซทมีสภาวะเป็น 0 (Low) และความถี่ของสัญญาณนาฬิกา(Clock) นี้จะสามารถกำหนดได้จากค่าความต้านทานที่นำมาต่อกับขา 13 และขา 14 ถ้าค่าความต้านทานสูงความถี่ที่ผลิตจะมีค่าต่ำ และถ้าค่าความต้านทานลดต่ำลง ความถี่ที่ผลิตก็จะสูงขึ้น เป็นส่วนกลับกันระหว่างค่าความต้านทานกับความถี่ที่ผลิต ดังตารางในรูปที่ 3.4

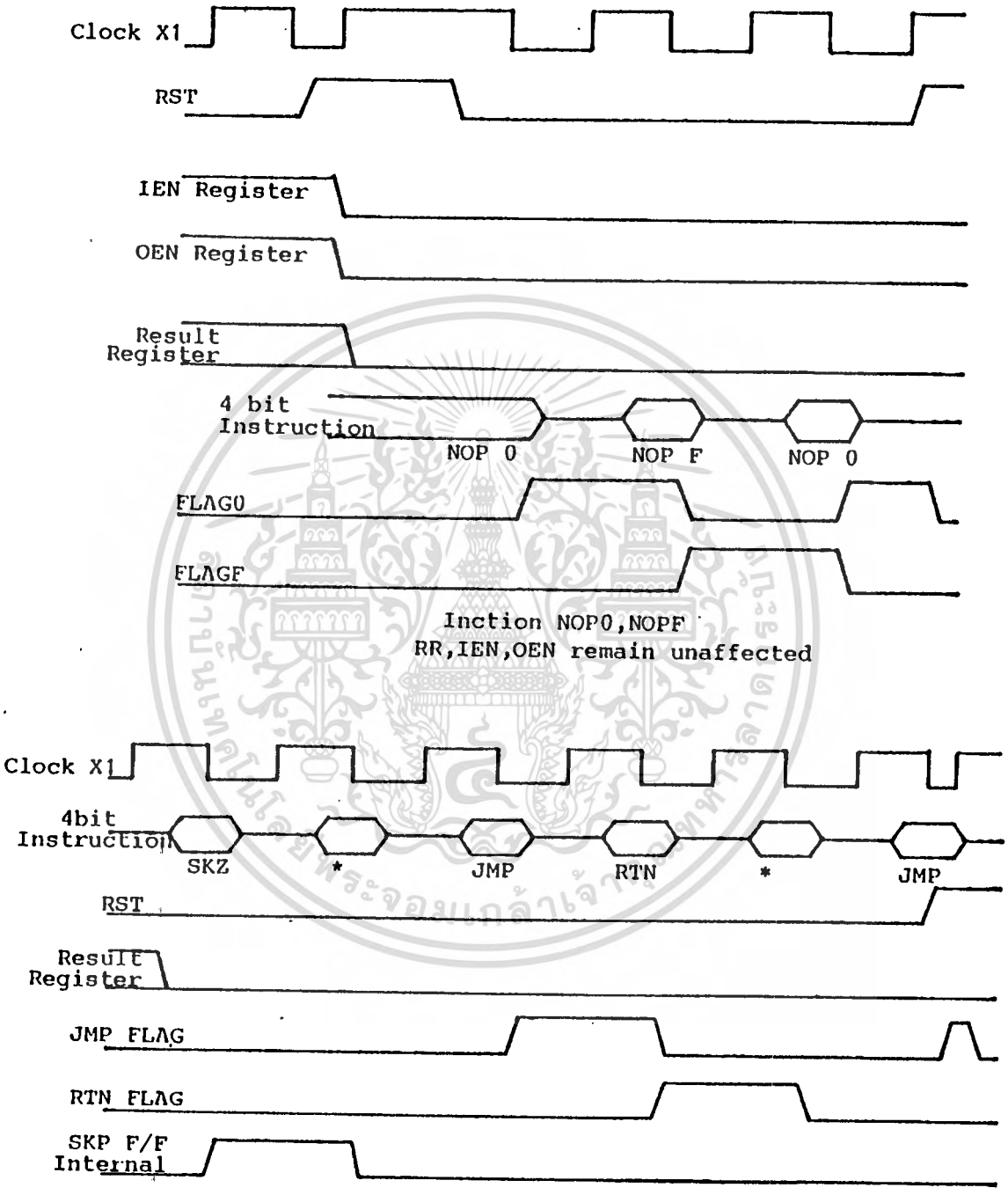


รูปที่ 3.4 ตารางความสัมพันธ์ระหว่างค่าความต้านทานกับความถี่

จากตารางในรูปที่ 3.4 จะเห็นได้ว่าสัญญาณนาฬิกาที่ใช้นั้น สามารถใช้สัญญาณนาฬิกาได้สูงสุดประมาณ 2.5 เมกกะเฮิร์ท (2.5 MHz) หน่วยประมวลผลกลางจะสามารถทำคำสั่งได้ถึง 2.5 ล้านคำสั่งใน 1 วินาที หรือทำคำสั่ง 1 คำสั่งจะใช้เวลาเพียง 400 นาโนเซคกันด์ (400 nS)

3.4 แผนผังเวลา (Timming Diagram) ของหน่วยประมวลผลกลางขนาด 1 บิท





รูปที่ 3.5 Timing Diagram

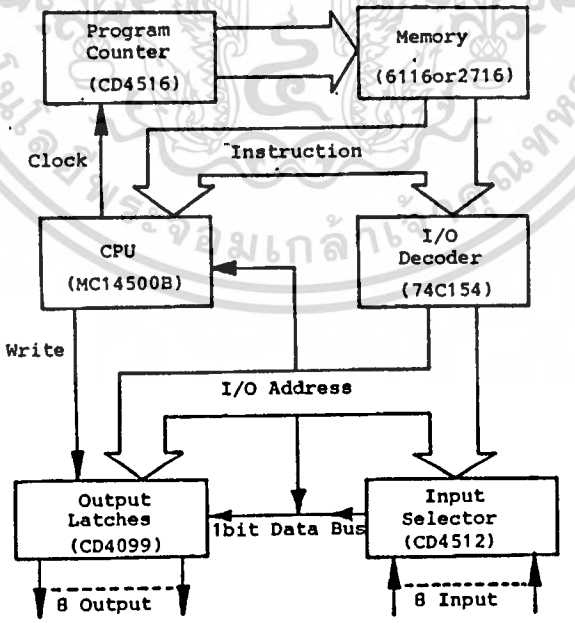
บทที่ 4

การทำงานและการออกแบบพีแอลซีโดยใช้หน่วยประมวลผลกลางขนาด 1 บิท

4.1 การทำงานของพีแอลซีขนาด 1 บิท

ส่วนประกอบของวงจรที่สำคัญของพีแอลซีขนาด 1 บิทประกอบ ด้วยส่วนต่างๆ 6 ส่วนดังนี้

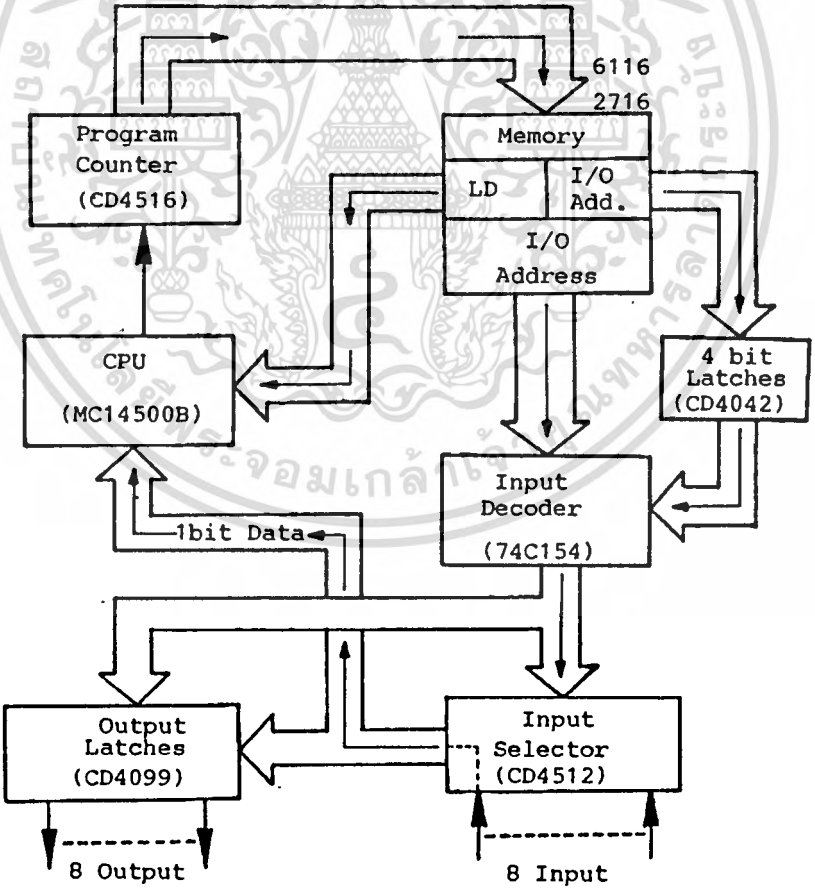
1. หน่วยประมวลผลกลาง (CPU)
2. หน่วยนับโปรแกรม (Program Counter)
3. หน่วยความจำ (Memory)
4. หน่วยแปลตำแหน่งอินพุทและเอาท์พุท (Decoder)
5. อินพุท (Input)
6. เอาท์พุท (Output)



รูปที่ 4.1 แสดงการทำงานของระบบมาตราฐานของพีแอลซีขนาด 1 บิท

จากรูปที่ 4.1 หน่วยนับโปรแกรมจะรับสัญญาณนาฬิกาจากหน่วยประมวลผลกลางมานับ และส่งเอาท์พุทจากการนับไปชี้ตำแหน่งของหน่วยความจำ ข้อมูลที่ได้จากหน่วยความจำขนาด 4 บิทบนจะถูกส่งไปยังหน่วยประมวลผลกลางเพื่อทำการแปลความหมายของคำสั่งและประมวลผลตามคำสั่ง ข้อมูลอีกส่วนหนึ่งจากหน่วยความจำจะถูกนำมาแปลตำแหน่งของอินพุทและเอาท์พุท เพื่อที่จะรับข้อมูลขนาด 1 บิทจากอินพุทใดอินพุทหนึ่งเข้ามายังหน่วยประมวลผลกลาง หรือส่งข้อมูลขนาด 1 บิทออกไปยังส่วนของเอาท์พุทใดเอาท์พุทหนึ่ง

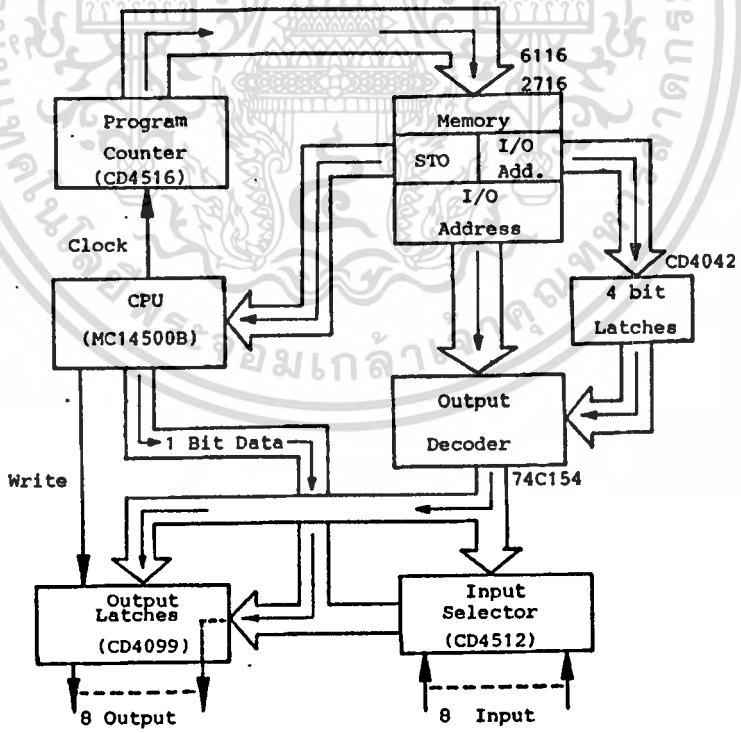
ตัวอย่างการใช้คำสั่ง LD ในการอ่านอินพุทเข้ามายังหน่วยประมวลผลกลางดังในรูปที่ 4.2



รูปที่ 4.2 แผนผังการทำงานในการอ่านอินพุทเข้ามา

การอ่านอินพุทเข้ามาจากรูปที่ 4.2 หน่วยประมวลผลกลางจะส่งสัญญาณนาฬิกาไปยังหน่วยนับโปรแกรมเพื่อทำการนับ หน่วยนับโปรแกรมจะส่งเอาท์พุทไปชี้ตำแหน่งของหน่วยความจำ เมื่อสัญญาณนาฬิกา X1 เป็น 1 (High) ข้อมูลขนาด 4 บิทจากหน่วยความจำจะถูกส่งไปยังหน่วยประมวลผลกลางเพื่อแปลคำสั่ง LD และเมื่อสัญญาณนาฬิกา X1 เป็น 0 (Low) ข้อมูลจากหน่วยความจำจะถูกส่งไปยังหน่วยแปลตำแหน่ง หน่วยแปลตำแหน่งจะเป็นตัวชี้ตำแหน่งของอินพุทตัวใดตัวหนึ่งตามที่กำหนด และค่าสภาวะจากอินพุทนี้จะผ่านตัวเลือกอินพุทเข้าไปยังหน่วยประมวลผลกลาง ซึ่งข้อมูลที่ได้จากตัวเลือกอินพุทนี้จะมีขนาด 1 บิท

ตัวอย่างการใช้คำสั่ง STO ในการส่งข้อมูลจากหน่วยประมวลผลกลางออกไปยังเอาท์พุทดังในรูปที่ 4.3



รูปที่ 4.3 แผนผังการส่งข้อมูลออกไปยังเอาท์พุท

การส่งเอาท์พุทออกไป จากรูปที่ 4.3 หน่วยประมวลผลกลางจะส่งสัญญาณนาฬิกาไปยังหน่วยนับโปรแกรม ซึ่งหน่วยนับโปรแกรมจะไปชี้ตำแหน่งของหน่วยความจำ และเมื่อสัญญาณนาฬิกา X1 เป็น 1(High) ข้อมูลขนาด 4 บิทจากหน่วยความจำจะถูกส่งไปยังหน่วยประมวลผลกลางเพื่อทำการแปลคำสั่ง STO และเมื่อสัญญาณนาฬิกา X1 เป็น 0 (Low) หน่วยประมวลผลกลางจะส่งสัญญาณ Write ไปยังเอาท์พุท เพื่อให้เอาท์พุทพร้อมที่จะส่งข้อมูลผ่านออกไปได้ และพร้อมกันนั้นหน่วยความจำจะส่งตำแหน่งของเอาท์พุทไปยังตัวแปลตำแหน่ง โดยจะส่งตำแหน่งของเอาท์พุทไปยังตัวเลือกเอาท์พุทให้ส่งผ่านข้อมูลจากหน่วยประมวลผลกลางขนาด 1 บิท ออกไปยังตำแหน่งเอาท์พุทที่กำหนดไว้

4.2 หน่วยนับโปรแกรม (Program Counter)

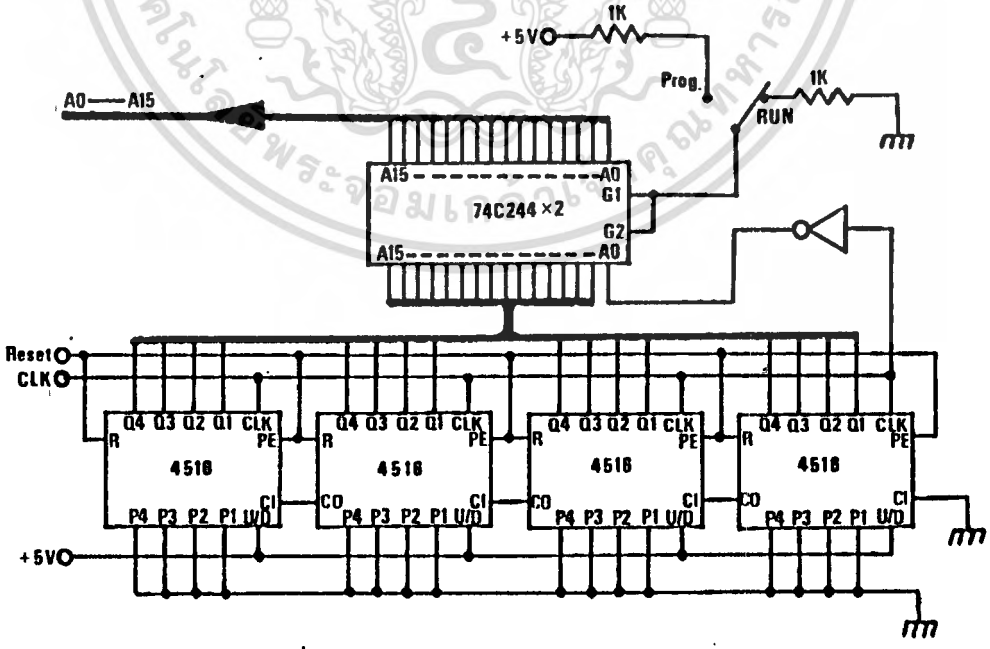
หน้าที่ของหน่วยนับโปรแกรมคือรับสัญญาณนาฬิกาจากหน่วยประมวลผลกลางมาทำการนับ และส่งเอาท์พุทที่ได้จากการนับไปยังขาแอสเดรสของหน่วยความจำเพื่อชี้ตำแหน่งในหน่วยความจำ การนับโปรแกรมนั้นนับแบบขึ้นคือจาก 0 ขึ้นไปเรื่อย ๆ จนถึงค่านับสูงสุดของหน่วยนับโปรแกรม แล้วหน่วยนับจะถูกรีเซทกลับมาเริ่มต้นใหม่ ซึ่งการนับเป็นแบบเลขไบนารี (Binary) คือนับจาก 0 ถึง F (0-15) ซึ่งจะให้เอาท์พุทขนาด 4 บิท และเอาท์พุทขนาด 4 บิทนี้สามารถชี้ตำแหน่งแอสเดรสของหน่วยความจำได้ 16 ตำแหน่ง ถ้าต้องการหน่วยความจำที่สูงกว่านี้ ก็สามารถเพิ่มขนาดของบิทของหน่วยนับโปรแกรมขึ้นไปได้อีกไม่จำกัด การเพิ่มบิทของหน่วยนับโปรแกรมจะต้องดูความเหมาะสมทางด้านความเร็วในการนับว่าเวลาในการนับโปรแกรมตั้งแต่เริ่มต้นจนกระทั่งสิ้นสุด และกลับมาเริ่มใหม่ใช้เวลานานเท่าใด

ในการทำงานของ พีแอลซี จากจุดเริ่มต้นของโปรแกรมจนถึงจุดสิ้นสุดของโปรแกรมนั้นควรจะมีความเร็วในหนึ่งลูปโปรแกรม (Loop Program) น้อยกว่าช่วงเวลาของสวิทช์เปิดปิดซึ่งเป็นอินพุทของพีแอลซีประมาณ 3 เท่าขึ้นไป เพราะว่าเมื่ออินพุทมีการเปลี่ยนแปลง พีแอลซีจะต้องสามารถรับรู้การเปลี่ยนแปลงของอินพุทได้ ถ้าหากว่าความเร็วของการทำโปรแกรมแต่ละครั้งใช้เวลามากกว่าการเปิดปิดของสวิทช์แล้ว เป็นไปได้

ว่าเมื่ออินพุทมีการเปลี่ยนแปลงที่เร็วมาก พีแอลซีอาจจะทำงานไม่ทันกับการเปลี่ยนแปลงของอินพุท จึงทำให้เกิดการผิดพลาดขึ้นได้

จากการทดลองการกดสวิทช์ ช่วงเวลาที่สวิทช์ใช้เวลาในการปิดและเปิดเร็วที่สุดประมาณ 150 มิลิเซคกันด์ (150 mS)โดยเฉลี่ย เพราะฉนั้นเวลาในการทำโปรแกรมสูงสุดในหนึ่งครั้งจะเท่ากับ 50 มิลิเซคกันด์ (50 mS) ถ้าใช้เวลาในการทำโปรแกรมน้อยกว่านี้ความเที่ยงตรงของการทำงานก็จะสูงขึ้น ในการออกแบบนี้จะใช้ความเร็วในการทำโปรแกรมหนึ่งครั้งใช้เวลาน้อยกว่าการปิดเปิดของสวิทช์ประมาณ 5 เท่า ฉนั้นเวลาในการทำโปรแกรมยอมให้ได้สูงสุดเท่ากับ 30 มิลิเซคกันด์ (30 mS)

ในการออกแบบพีแอลซีโดยใช้หน่วยประมวลผลกลางขนาด 1 บิทนี้จะใช้ความถี่ของสัญญาณนาฬิกา 655,360 เฮิรท์ซ (Hz) ซึ่งจะทำคำสั่ง 1 คำสั่งจะใช้เวลา 1.5 ไมโครเซคกันด์ (1.5 uS) ถ้าทำโปรแกรมทั้งหมดในเวลา 30 มิลิเซคกันด์ จะทำคำสั่งได้ทั้งหมด 20,000 คำสั่ง ฉนั้นจะต้องใช้หน่วยนับโปรแกรมทั้งหมด 14 บิท โดยใช้ไอซีเบอร์ CD4516 ซึ่งเป็นแบบ Binary Up/Down Counter ทั้งหมดจำนวน 4 ตัวมาต่อกันดังในรูปที่ 4.4 ซึ่งจะชี้ตำแหน่งของหน่วยความจำได้ถึง 16 Kstep(32 Kbyte)



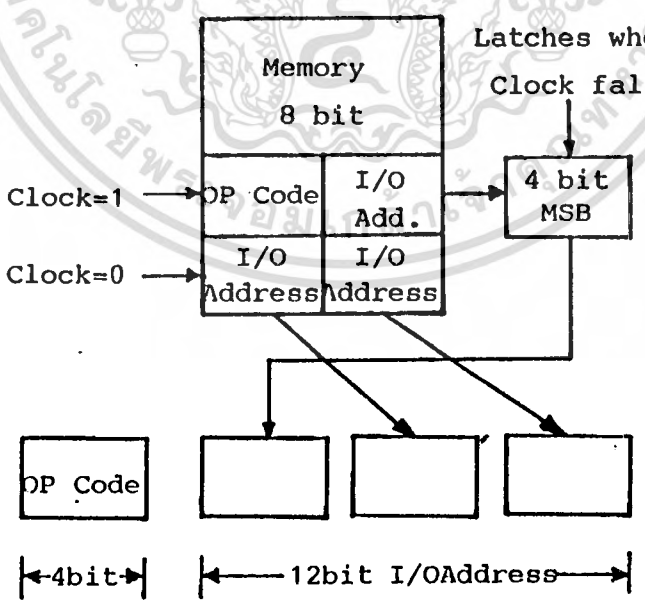
รูปที่ 4.4 หน่วยนับโปรแกรม

4.3 หน่วยความจำ (Memory)

หน่วยความจำของพีแอลซี มีหน้าที่ในการเก็บโปรแกรมการทำงานของระบบไว้ และเมื่อพีแอลซีเริ่มทำงาน หน่วยความจำจะส่งคำสั่งไปยังหน่วยประมวลผลกลางขนาด 4 บิท และส่งตำแหน่งของอินพุทและเอาท์พุทขนาด 12 บิท ไปยังส่วนแปลตำแหน่งอินพุทและเอาท์พุท

ในการออกแบบหน่วยความจำนี้จะใช้หน่วยความจำที่มีความกว้างของข้อมูลขนาด 8 บิท เพื่อให้สัมพันธ์กับเครื่องไมโครคอมพิวเตอร์ขนาด 8 บิททั่ว ๆ ไป เพราะว่าการรับโปรแกรมสามารถจะรับจากเครื่องป้อนโปรแกรม (Program Loader) โดยตรง หรืออาจจะรับโปรแกรมจากไมโครคอมพิวเตอร์ก็ได้

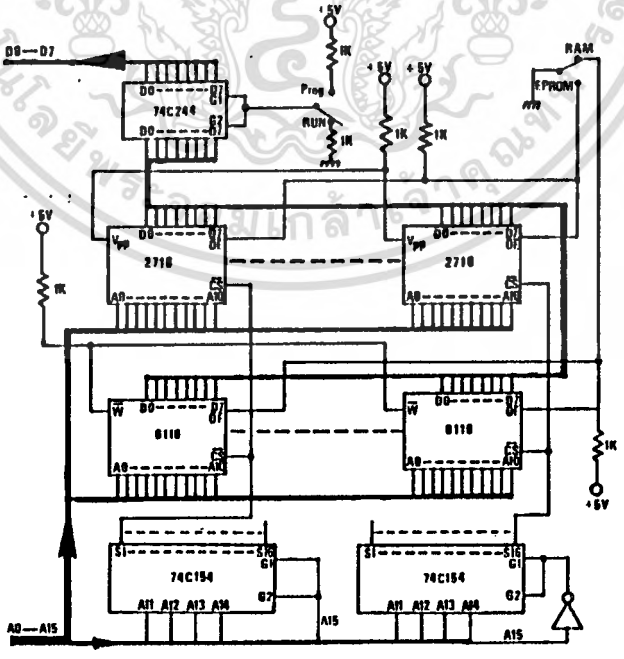
การใช้งานหน่วยความจำใน 1 คำสั่ง การทำงานของพีแอลซีจะประกอบด้วยคำสั่งขนาด 4 บิท และตามด้วยตำแหน่งของอินพุทและเอาท์พุทโดยการนำ 4 บิทที่ต่อจากคำสั่งรวมกับ 8 บิทในตำแหน่งถัดมาซึ่งรวมกันเป็น 12 บิท ซึ่งจะสามารถอ้างถึงตำแหน่งของอินพุทและเอาท์พุทได้สูงสุดถึง 4096 ตำแหน่งดังในรูปที่ 4.5



รูปที่ 4.5 การจัดพื้นที่ของหน่วยความจำ

รูปที่ 4.5 เป็นการจัดลักษณะพื้นที่ของหน่วยความจำ ในการทำงาน 1 คำสั่ง จะต้องใช้พื้นที่ของหน่วยความจำ 2 ตำแหน่ง ๆ ละ 8 บิท เมื่อสัญญาณนาฬิกาเป็น 1 (High) ข้อมูล 4 บิทแรกของหน่วยความจำจะถูกส่งไปยังหน่วยประมวลผลกลาง และข้อมูล 4 บิทหลังจะถูกส่งไปยังวงจรเก็บค่าสภาวะ (Latch) เมื่อสัญญาณนาฬิกาเป็น 0 (Low) ข้อมูล 4 บิทหลังจะถูกเก็บค่าสภาวะไว้เพื่อนำมารวมกับอีก 8 บิทในตำแหน่งถัดมารวมเป็น 12 บิท และทั้ง12 บิทนี้จะถูกส่งไปยังวงจรแปลตำแหน่งอินพุทและเอาท์พุท

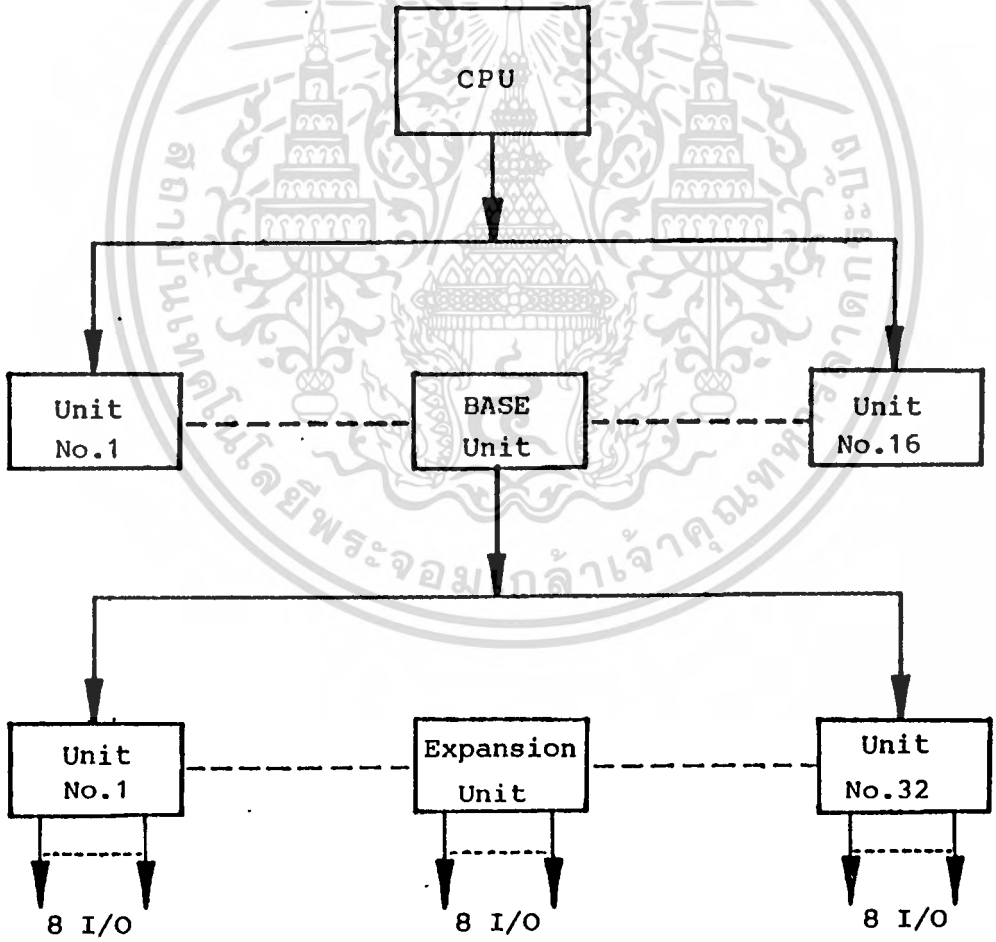
การใช้หน่วยความจำในวงจรจะใช้หน่วยความจำแบบ RAM (Ramdom Access Memory) หรือแบบ ROM (Read Only Memory) ก็ได้ โดยจะใช้ไอซีเบอร์ 6116 ซึ่งเป็น RAM หรือเบอร์ 2716 ซึ่งเป็น ROM ซึ่งเป็นที่นิยมใช้กันมากและหาซื้อได้ง่าย หน่วยความจำชนิดนี้จะมีความจุทั้งหมด 2 กิโลไบท์ (2 Kilo-byte) ถ้าต้องการความจุสูงกว่านี้ก็ได้โดยการเพิ่มไอซีหน่วยความจำเข้าไปภายในวงจร โดยจะเพิ่มได้สูงสุด 16 ตัว ซึ่งจะได้ความจุสูงถึง 32 กิโลไบท์ โดยใน 1 คำสั่งจะใช้หน่วยความจำ 2 ไบท์ เพราะฉนั้นจะเขียนโปรแกรมได้สูงสุดถึง 16,384 คำสั่ง ซึ่งมากพอสำหรับ 4096 อินพุท เอาท์พุทดังในรูปที่ 4.6



รูปที่ 4.6 วงจรของหน่วยความจำ

4.4 หน่วยแปลตำแหน่งอินพุทและเอาท์พุท (Decoder)

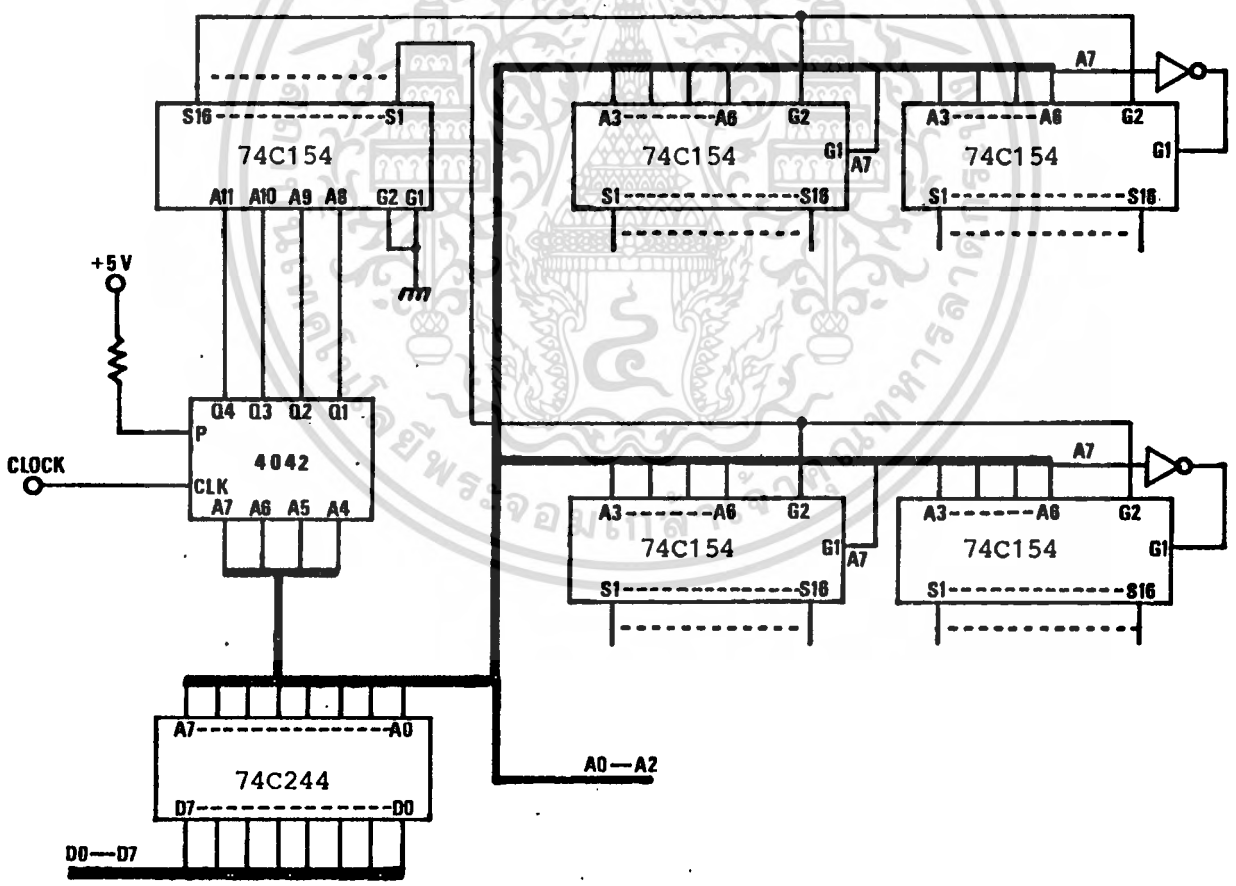
จากหน่วยความจำขนาด 12 บิท ซึ่งเป็นตำแหน่งของอินพุทและเอาท์พุทสามารถแปลเป็นตำแหน่งของอินพุทและเอาท์พุทได้ถึง 4096 ตำแหน่ง โดยการแบ่งตำแหน่งของอินพุทและเอาท์พุทออกเป็น 16 ชุด (16 Base Unit) และใน 1 ชุดนี้จะแบ่งย่อยลงไปเป็น 32 ส่วน (32 Expansion Unit) โดยใน 1 ส่วนจะมีอินพุทเอาท์พุททั้งหมด 8 ตำแหน่ง(8 I/O Unit) ดังในรูปที่ 4.7



รูปที่ 4.7 การแบ่งส่วนของอินพุทและเอาท์พุท

จากรูปที่ 4.7 หน่วยที่เล็กที่สุดคือส่วนของตัวเลือกอินพุทและเอาท์พุท โดยส่วนของตัวเลือกอินพุทหรือเอาท์พุทนี้จะใช้ข้อมูลจากหน่วยความจำ 3 บิทล่างส่งมาเข้าที่ตัวเลือกโดยตรง เพื่อแปลตำแหน่งซึ่งจะแปลตำแหน่งได้ทั้งหมด 8 ตำแหน่ง และจะใช้บิทที่สูงขึ้นมาจำนวน 5 บิทนำมาแปลตำแหน่งได้ 32 ตำแหน่ง ทั้ง 32 ตำแหน่งนี้จะเป็นตัวเลือกส่วนของอินพุทหรือเอาท์พุทส่วนใด ๆ ใน 32 ส่วน และข้อมูลจากหน่วยความจำในบิทสูงที่เหลืออีก 4 บิท จะนำมาแปลตำแหน่งของชุดต่าง ๆ ได้อีก 16 ชุด

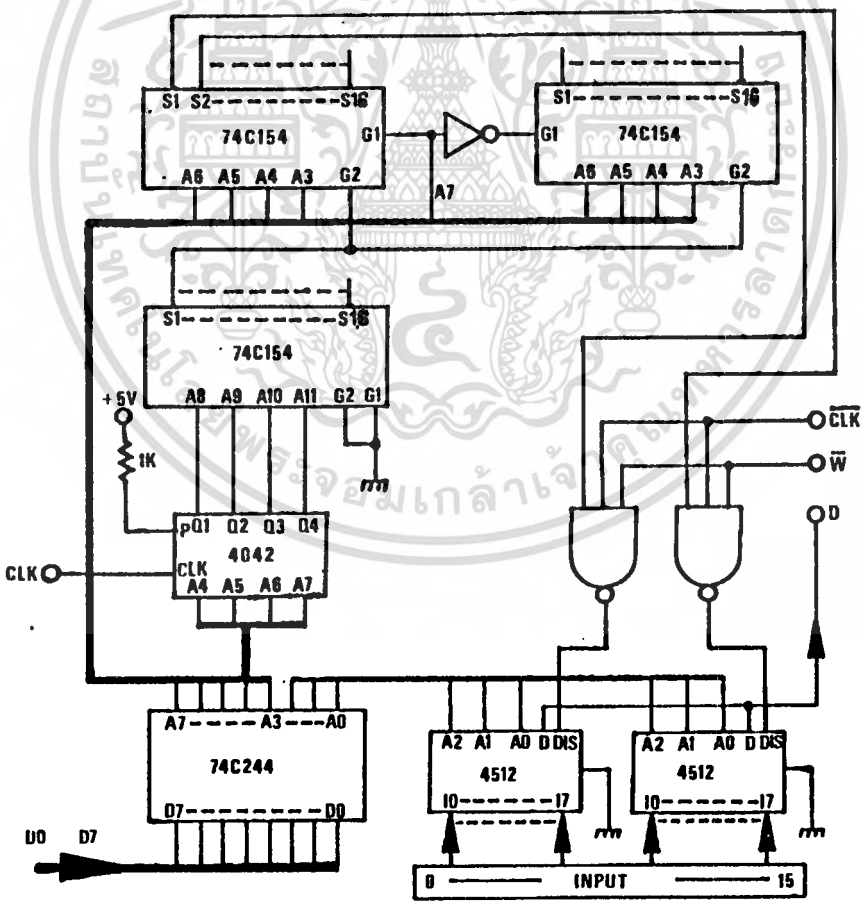
วงจรที่ใช้แปลตำแหน่งของชุดต่าง ๆ และส่วนต่าง ๆ จะใช้ไอซีเบอร์ 74C154 ซึ่งเป็นแบบเข้า 4 ออก 16 ดังในรูปที่ 4.8



รูปที่ 4.8 วงจรแปลตำแหน่งอินพุทและเอาท์พุท

4.5 อินพุท (Input)

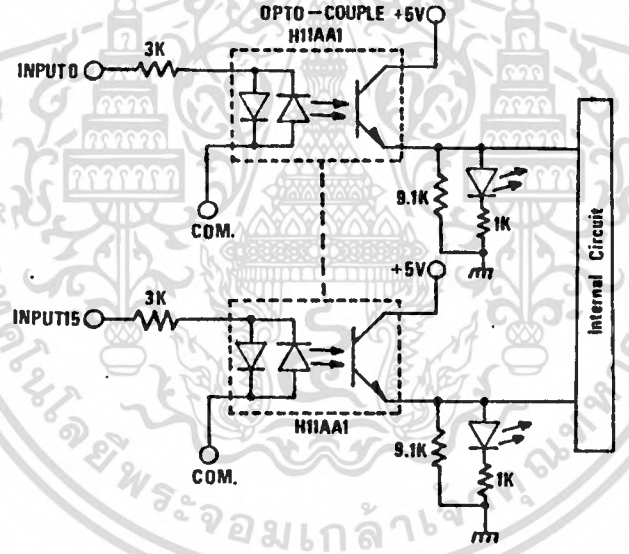
การทำงานของหน่วยประมวลผลกลาง จะรับข้อมูลจากอินพุทเข้ามาทำการประมวลผลได้ครั้งละ 1 บิทเท่านั้นจากอินพุทหลาย ๆ อินพุท เพราะฉนั้นจึงต้องมีหน่วยเลือกอินพุทเข้ามาโดยใช้ไอซีเบอร์ MC4512 ซึ่งเป็นแบบข้อมูลเข้า 8 ข้อมูล แต่เลือกข้อมูลออกทีละ 1 ข้อมูล ซึ่งจะใช้ข้อมูลจากหน่วยความจำบิทต่ำสุด 3 บิทเป็นตัวเลือกอินพุทเข้ามา และมีขาควบคุมอีกหนึ่งขา เพราะเวลาใช้งานจะมีอินพุทมากกว่า 8 อินพุท เช่นเป็น 16 อินพุท จึงต้องมีไอซีเลือกอินพุทถึง 2 ตัว เพราะฉนั้นขาควบคุมนี้จะเป็นส่วนเลือกว่าจะเรียกจากไอซีตัวไหนเข้ามา โดยส่วนควบคุมนี้จะได้มาจากหน่วยแปลอินพุท ดังในรูปที่ 4.9



รูปที่ 4.9 อินพุท

จำนวนของอินพุทใน 1 ชุดจะมีทั้งหมด 120 อินพุท โดยเริ่มจากตำแหน่งที่ 0 ถึงตำแหน่งที่ 119 ในเลขฐานสิบ หรือเริ่มจากตำแหน่งที่ 0 ถึงตำแหน่งที่ 77 ในเลขฐานสิบหก

การแบ่งอินพุท 120 อินพุทจาก 256 อินพุทใน 1 ชุด เพราะว่าตำแหน่งที่ 121 ถึง 256 นั้นใช้เป็นตำแหน่งของเอาท์พุท รีจิสเตอร์ ตัวตั้งเวลาและตัวนับ โดยส่วนต่าง ๆ เหล่านี้จะต้องสามารถเรียกกลับเข้ามาทางอินพุทได้ จึงต้องใช้ตำแหน่งของอินพุทที่เหลือ คือจาก 121 ถึง 256 เป็นตัวรับข้อมูลจากส่วนของเอาท์พุท เข้ามาทำการประมวลผลร่วมกับอินพุทตามเงื่อนไขของการทำงานในโปรแกรม



รูปที่ 4.10 วงจรสำหรับอินพุท

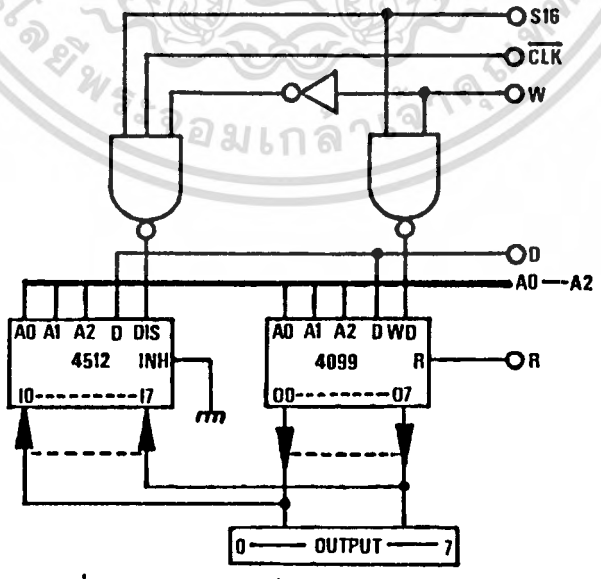
จากรูปที่ 4.10 เป็น Opto Couple ที่ใช้รับค่าสภาวะจากกระบวนการเข้ามาและส่งต่อมาเข้าส่วนของอินพุทของเครื่องควบคุมอีกทีหนึ่ง ทั้งนี้เพราะว่าในกระบวนการทางอุตสาหกรรมนั้น สภาวะ 1 (High) อาจจะไม่ใช่ 5 V. ซึ่งทั่ว ๆ ไปมักจะใช้ไฟ 24 V. เป็นสภาวะ 1 (High) ในการใช้ Opto Couple แปลสัญญาณจากกระบวนการ คือ เมื่ออินพุทของ Opto Couple เริ่มจาก 4.5 V. ถึง 25 V. เอาท์พุทของ Opto Couple จะ "ON" หรือมีสภาวะเป็น 1 (High) และเมื่ออินพุทของ Opto

Couple มีสัญญาณไฟฟ้าต่ำกว่า 4.5 V. เอาท์พุทของ Opto Couple จะ "OFF" หรือมีสภาวะเป็น 0 (Low)

ในการใช้ Opto Couple เป็นตัวกันชนระหว่างกระบวนการกับเครื่องควบคุม จะมีข้อดีคือ เป็นการแยกเครื่องควบคุมออกจากกระบวนการเพื่อป้องกันสัญญาณรบกวนและป้องกันสัญญาณไฟฟ้าสูง ๆ ซึ่งอาจเกิดขึ้นได้ เช่น สายไฟของอินพุทที่ต่อจากกระบวนการมายังเครื่องควบคุมเกิดลัดวงจร (Short Circuit) กับสายไฟความต่างศักย์สูง ๆ เช่น 220 V. ก็จะไม่ทำให้เกิดอันตรายกับเครื่องควบคุมแต่อย่างใด

4.6 เอาท์พุท (Output)

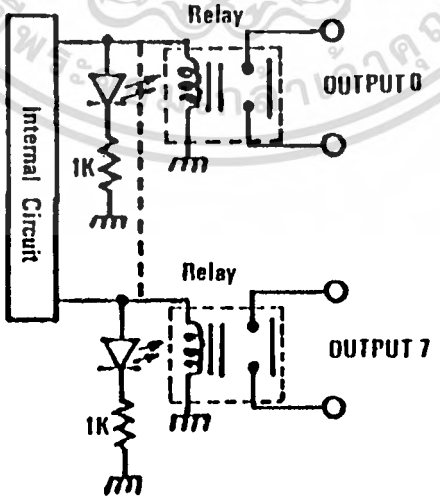
การทำงานของเอาท์พุทจะคล้าย ๆ กับอินพุท คือในส่วนของการแปลตำแหน่งจะเหมือนกัน ยกเว้นทางเดินของข้อมูลจะกลับกัน คือข้อมูลจากหน่วยประมวลผลกลางขนาด 1 บิทจะถูกส่งออกไปยังเอาท์พุทใดเอาท์พุทหนึ่ง และในการส่งออกไปนี้จะมีขาสัญญาณเขียน (Write) เป็นตัวควบคุมอีกทีหนึ่ง เมื่อส่งค่าสภาวะออกไปที่เอาท์พุทแล้วค่าสภาวะจะถูกเก็บให้คงสภาวะอยู่เช่นนั้นจนกว่าจะมีการเปลี่ยนแปลงใหม่หรือถูกลบ (Reset) ซึ่งวงจรการทำงานเป็นดังในรูปที่ 4.11



รูปที่ 4.11 เอาท์พุท (Output)

จากรูปที่ 4.11 จะเห็นว่าเอาท์พุทต่อโดยตรงเข้าทางด้านอินพุท ซึ่งมีตำแหน่งแอสเดรสเดียวกันกับเอาท์พุท เพราะเวลาใช้งานจริงบางครั้งในโปรแกรมจะต้องเรียกเอาท์พุทเข้ามาทำโปรแกรมร่วมกับอินพุทด้วย จำนวนของเอาท์พุทใน 1 ชุดมีทั้งหมด 80 เอาท์พุท โดยเริ่มจากตำแหน่งที่ 120 ถึง 199 ในเลขฐานสิบ หรือจากตำแหน่งที่ 78 ถึง C7 ในเลขฐานสิบหก ซึ่งตำแหน่งของอินพุทที่ใช้สำหรับอ้างถึงเอาท์พุทจะเป็นตำแหน่งเดียวกัน

เอาท์พุทที่ได้จากตัวเลือกเอาท์พุทนี้เมื่อมีสภาวะเป็น 1 (High) จะมีค่าความต่างศักย์ประมาณ 5 V. จึงไม่สามารถนำไปต่อยังโหลด (Load) ได้โดยตรง ทั้งนี้เพราะในกระบวนการอุตสาหกรรมนั้นโหลด (Load) ที่ใช้มักจะเป็นมอเตอร์หรือดวงไฟซึ่งต้องใช้ความต่างศักย์สูง และกินกระแสไฟสูงด้วย จึงต้องนำเอาท์พุทของเครื่องควบคุมส่งไปเข้าคอยล์ (Coil) ของรีเลย์อีกทีหนึ่ง เวลาใช้งานจะใช้หน้าสัมผัสของรีเลย์เป็นเอาท์พุท ซึ่งจะจ่ายกระแสได้สูงและใช้ได้กับไฟตรง (DC) และไฟสลับ (AC) ที่มีความต่างศักย์สูง ๆ ได้ และรีเลย์นี้ยังมีหน้าที่แยกเครื่องควบคุมออกจากกระบวนการด้วยเพื่อป้องกันสัญญาณรบกวน และสัญญาณไฟความต่างศักย์สูง ๆ เข้ามาทำอันตรายกับเครื่องควบคุมดังในรูปที่ 4.12

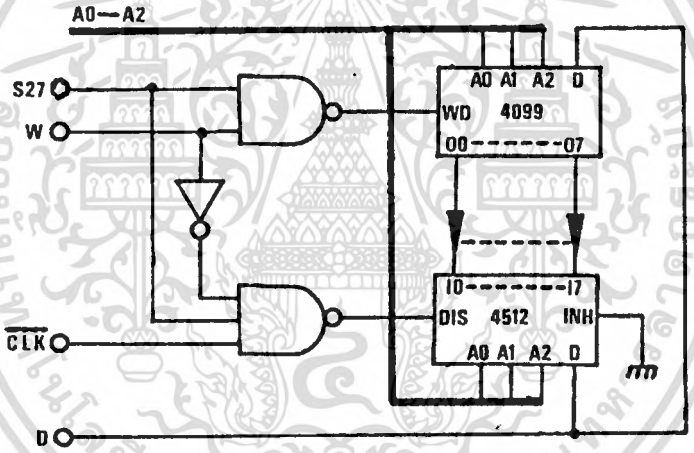


รูปที่ 4.12 วงจรทางด้านเอาท์พุท

4.7 ฟังก์ชั่นพิเศษ (Special Function)

4.7.1 รีจิสเตอร์ (Register)

รีจิสเตอร์มีหน้าที่ในการเก็บค่าสภาวะ หรือเอาท์พุทที่ไม่ได้ใช้งานโดยตรงไว้ชั่วคราวก่อน หรือใช้เก็บค่าสภาวะจากหลาย ๆ อินพุทซึ่งทำการประมวลผลทางลอจิกกันและได้ผลลัพธ์ออกมาแต่ยังไม่ได้ใช้งานในขณะนั้น และ/หรือ อาจจะนำไปใช้งานในส่วนอื่นของโปรแกรม จึงจำเป็นต้องมีที่สำหรับเก็บค่าสภาวะไว้ชั่วคราวก่อน วงจรที่ใช้สำหรับเก็บค่าสภาวะไว้ชั่วคราวนี้เป็นดังรูปที่ 4.13



รูปที่ 4.13 วงจรเก็บค่าสภาวะชั่วคราว (Register)

จากรูปที่ 4.13 วงจรเก็บค่าสภาวะชั่วคราวนี้จะประกอบด้วยอินพุทและเอาท์พุทที่มีตำแหน่งแอสเดรส (Address) ตรงกัน โดยข้อมูลที่ส่งมาจากหน่วยประมวลผลกลางจะถูกนำมาเก็บไว้ที่เอาท์พุท และจากเอาท์พุทจะต่อตรงเข้าทางด้านอินพุท โดยเวลาเรียกมาใช้จะเรียกจากอินพุทเข้ามายังหน่วยประมวลผลกลางได้เลย

ตำแหน่งของรีจิสเตอร์เริ่มจากตำแหน่งที่ 201 ถึง 230 ในเลขฐานสิบ หรือจากตำแหน่งที่ C9 ถึง E6 ในเลขฐานสิบหกซึ่งมีทั้งหมด 30 ตำแหน่งใน 1 ชุด

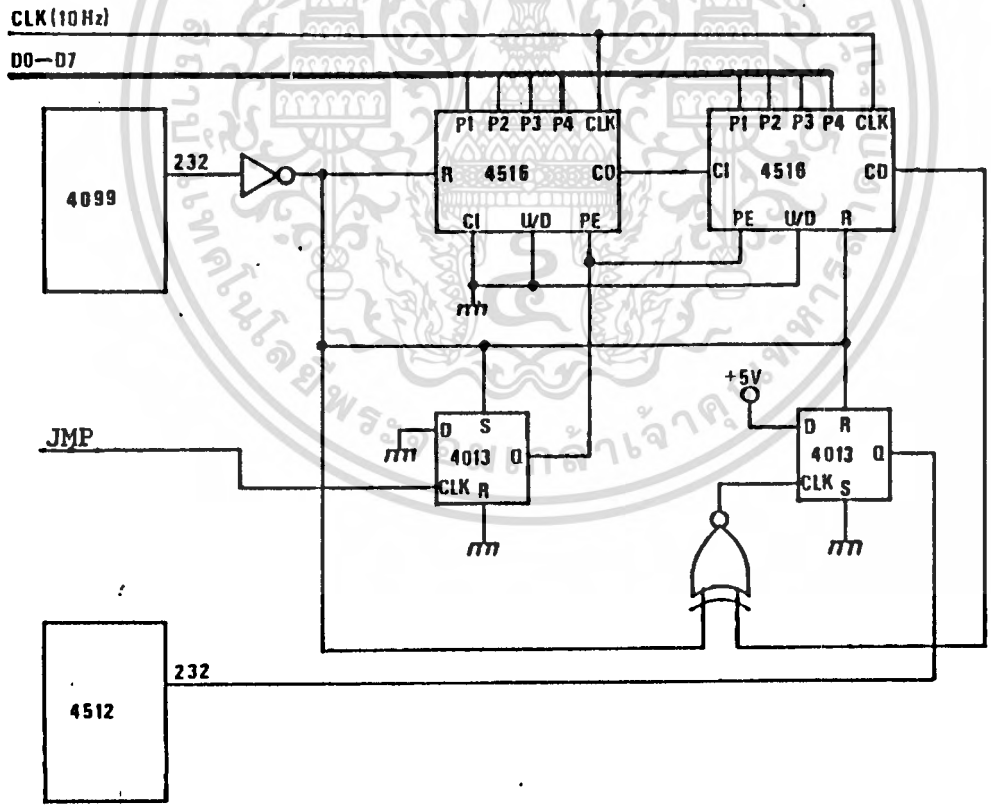
ส่วนรีจิสเตอร์ตำแหน่งที่ 200 (C8) จะมีค่าสภาวะเป็น 1 (High) ใช้สำหรับเวลาต้องการใช้ค่าสภาวะ 1 (High) เพื่อนำไปใช้สำหรับคำสั่ง IEN กับ OEN ตอนเริ่ม

ต้นโปรแกรม หรือเมื่อต้องการใช้ค่าสภาวะ 1 (High) ในส่วนใดของโปรแกรมก็ได้โดยการเรียกจากตำแหน่งที่ 200(C8)

ส่วนตำแหน่งที่ 231 (E7) เป็นตำแหน่งที่เก็บค่าสภาวะของรีจิสเตอร์ผลลัพธ์ เมื่อใดที่ต้องการนำค่าสภาวะจากรีจิสเตอร์ผลลัพธ์ขณะนั้นไปใช้ในโปรแกรม ก็สามารถเรียกได้จากตำแหน่งที่ 231 ได้โดยตรง

4.7.2 หน่วยตั้งเวลา (Timer)

วงจรตั้งเวลาใน 1 ชุดมีทั้งหมด 8 หน่วย การตั้งเวลาสามารถตั้งเวลาได้จาก 0.1 วินาทีถึง 25.5 วินาที โดยการใส่ค่าของการตั้งเวลาไว้ในโปรแกรม และโปรแกรมจะส่งข้อมูลขนาด 8 บิทมาตั้งค่าเวลาที่วงจรตั้งเวลาดังในรูปที่ 4.14



รูปที่ 4.14 วงจรตั้งเวลา

รูปที่ 4.14 เป็นวงจรตั้งเวลาที่ประกอบด้วยไอซีเบอร์ 4516 ซึ่งเป็นวงจรนับ

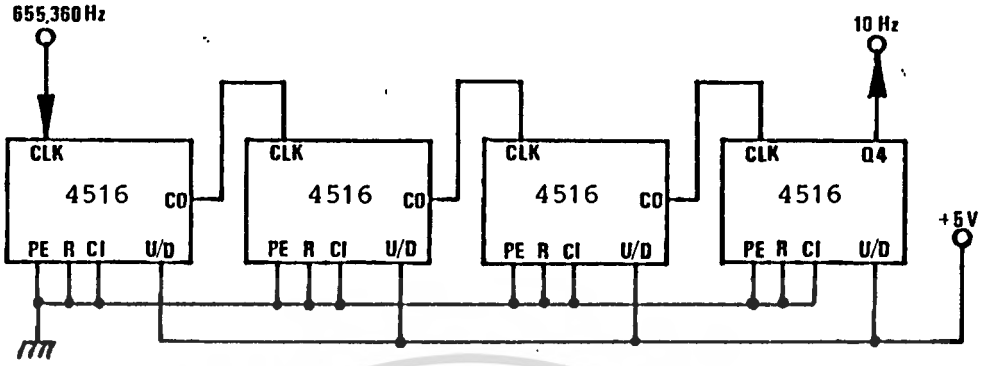
สัญญาณนาฬิกาขนาด 4 บิทสองตัวนำมาต่ออนุกรมกันรวมเป็น 8 บิท สามารถตั้งค่าเวลาได้ 256 ค่าคือ 0 ถึง 255 และค่าสัญญาณนาฬิกามาตรฐานที่นำมาใช้นับนั้นใช้ความถี่ 10 เฮิรท์ซ โดยใช้วงจรหารความถี่ของสัญญาณนาฬิกาจากหน่วยประมวลผลกลางให้เหลือ 10 เฮิรท์ซ และจะใช้ไอซีเบอร์ 4013 ซึ่งเป็น D Flip-Flop เป็นส่วนตั้งค่าเวลาให้กับวงจรนับ 1 ตัว ส่วนอีก 1 ตัวจะเป็นส่วนรับเอาท์พุทจากวงจรนับหลังจากนับครบเวลาแล้วมาเก็บค่าสภาวะเอาไว้เพื่อส่งให้กับอินพุท สำหรับนำไปใช้ในโปรแกรมอีกทีหนึ่ง

เมื่อหน่วยประมวลผลกลางส่งเอาท์พุทเป็น 1 (High) ออกมายังตำแหน่งของวงจรตั้งเวลา ก็จะเป็นการเริ่มตั้งเวลา พร้อมกันนั้นหน่วยประมวลผลกลางก็จะส่งสัญญาณ JMP ออกมา สัญญาณนี้จะใช้เป็นสัญญาณของการตั้งค่าเวลา (SET) ให้กับวงจรนับ โดยสัญญาณ JMP นี้ต่ออยู่กับขา CLK ของ D Flip-Flop เมื่อ D Flip-Flop ได้รับสัญญาณ JMP ก็จะส่งเอาท์พุทไปเข้าขาพรีเซท (Pre-Set) ของวงจรนับ ซึ่งจะเริ่มนับสัญญาณนาฬิกาลงมาจนกระทั่งเป็นศูนย์ วงจรนับจะส่งเอาท์พุทออกมา 1 ลูก D Flip-Flop ก็จะทำการเก็บค่าสภาวะเอาไว้ และเมื่อเอาท์พุทของพีแอลซีเป็น 0 (Low) ในตำแหน่งของวงจรตั้งเวลา วงจรตั้งเวลาจะถูกรีเซท (Reset) ให้กลับไปเริ่มต้นพร้อมที่จะทำงานใหม่

ตำแหน่งของอินพุทและเอาท์พุทของวงจรตั้งเวลา เริ่มจากตำแหน่งที่ 232 ถึง 239 ในเลขฐานสิบ หรือจาก E8 ถึง EF ในเลขฐานสิบหก ซึ่งจะมีทั้งหมด 8 ตัว ถ้าต้องการจำนวนของหน่วยตั้งเวลามากกว่านี้ ก็สามารถใช้ตำแหน่งของอินพุทเอาท์พุทของระบบส่วนหนึ่งมาเป็นตำแหน่งของหน่วยตั้งเวลาได้

4.7.3 สัญญาณนาฬิกามาตรฐาน

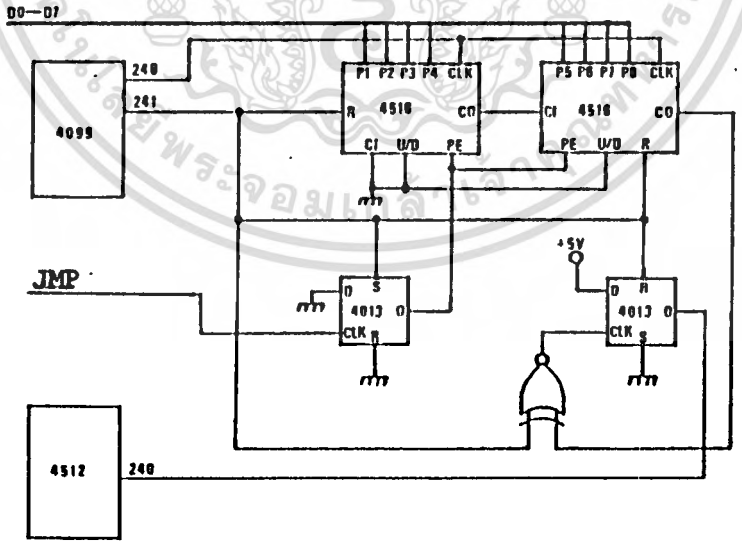
สัญญาณนาฬิกามาตรฐาน 10 เฮิรท์ซนั้นจะใช้วงจรหารความถี่จากสัญญาณนาฬิกา X1 ทั้งหมด 16 บิท คือหารจากความถี่สัญญาณนาฬิกา 655,360 เฮิรท์ซลงมาจนเหลือ 10 เฮิรท์ซ โดยใช้ไอซีเบอร์ CD4516 จำนวน 4 ตัวมาต่อกันดังวงจรในรูปที่ 4.15



รูปที่ 4.15 วงจรหารความถี่

4.7.4 วงจรนับลง (Down Counter)

วงจรนับลงใน 1 ชุดมีทั้งหมด 4 หน่วยโดยค่าของการนับนับได้จาก 1 ถึง 255 ซึ่งการทำงานของวงจรนับเหมือนกับวงจรตั้งเวลา จะต่างกันตรงที่ไม่ต้องใช้สัญญาณนาฬิกา 10 เฮิรท์ซ มาเป็นเวลามาตรฐานให้กับวงจรนับ แต่จะใช้ข้อมูล (Data) จากเอาท์พุทของระบบมาแทน ดังในรูปที่ 4.16



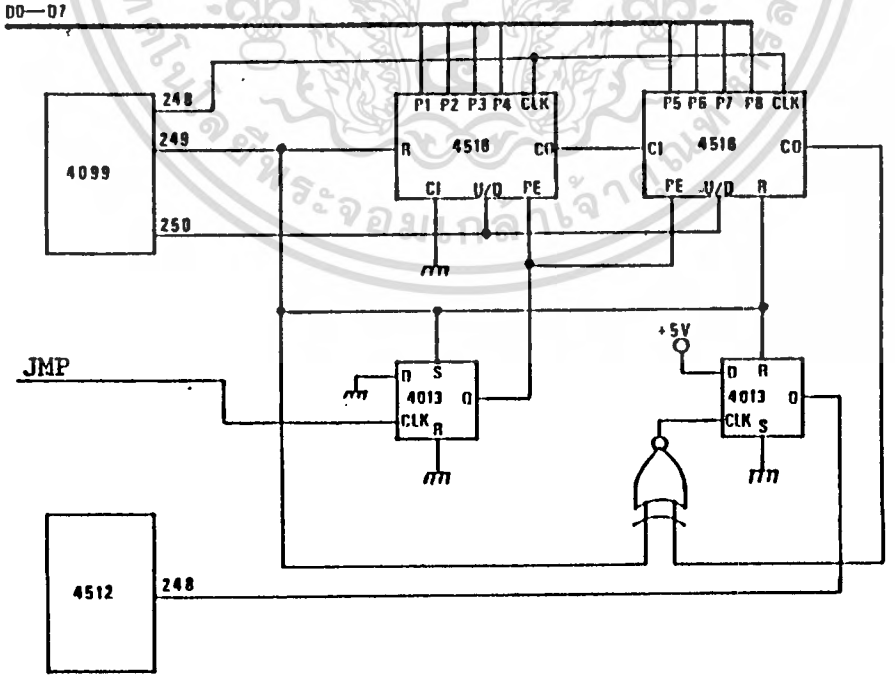
รูปที่ 4.16 วงจรนับลง

จากวงจรนับในรูปที่ 4.16 เอาท์พุทของระบบ 2 เอาท์พุทจะส่งมาให้กับวงจรนับ โดยจะใช้เป็นข้อมูลสำหรับนับ 1 เอาท์พุท และอีก 1 เอาท์พุทจะใช้สำหรับรีเซท (Reset) หลังจากนับครบจำนวนจะได้เอาท์พุทเป็น 1 (High) และถ้ารีเซท (Reset) ให้กับวงจรนับ จะทำให้วงจรนับกลับมาเริ่มต้นพร้อมที่จะนับใหม่

ตำแหน่งของวงจรนับลงจะเริ่มจากตำแหน่งที่ 240 ถึง 247 ในเลขฐานสิบหรือ F0 ถึง F7 ในเลขฐานสิบหก

4.7.5 วงจรนับขึ้นลง (Up/Down Counter)

วงจรนับขึ้นหรือลงใน 1 ชุดมี 2 หน่วยคือเริ่มจากตำแหน่งที่ 248 ถึง 255 ในเลขฐานสิบ หรือจาก F8 ถึง FF ในเลขฐานสิบหก การทำงานจะเหมือนกับวงจรนับลง และสามารถนับค่าได้จาก 1 ถึง 255 จะแตกต่างจากวงจรนับลงตรงที่สามารถเลือกให้นับขึ้นหรือนับลงก็ได้โดยส่งเอาท์พุทของระบบ 1 เอาท์พุทไปเป็นตัวเลือกว่าจะให้นับขึ้นหรือนับลงดังวงจรในรูปที่ 4.17



รูปที่ 4.17 วงจรนับขึ้นลง (Up/Down Counter)

บทที่ 5

การทำงานและการออกแบบวงจรหลัก

วงจรในรูปที่ 5.1 เป็นวงจรหลัก (Main Unit) โดยออกแบบส่วนต่าง ๆ สำรองไว้สำหรับการขยายขีดความสามารถของพีแอลซีจนถึงสูงสุด โดยไม่ต้องแก้ไขหรือดัดแปลงวงจรหลักเลย เมื่อต้องการขยายจำนวนอินพุทเอาท์พุท หรือตัวตั้งเวลา ฯลฯ ก็นำวงจรของส่วนนั้น ๆ มาเสียบเพิ่มเข้ากับวงจรหลักได้เลย

ในวงจรหลักนี้ประกอบด้วยส่วนสำคัญ 4 ส่วน คือ

1. หน่วยนับโปรแกรม (Program Counter)
2. หน่วยความจำ (Memory)
3. หน่วยแปลตำแหน่งอินพุทเอาท์พุท (Decoder)
4. หน่วยประมวลผลกลาง (CPU)

ในการชี้ตำแหน่งต่าง ๆ ของหน่วยความจำนั้นจะใช้โปรแกรมนับเป็นตัวชี้ตำแหน่ง โดยจะใช้ไอซีเบอร์ CD4516 4 ตัว เอาท์พุทที่ได้จากไอซีโปรแกรมนับนี้จะถูกส่งไปเข้าขาแอดเดรส (Address) ของหน่วยความจำ จากไอซี 4 ตัวนี้จะได้จำนวนเอาท์พุททั้งหมด 16 บิท แต่ในการใช้งานจะใช้เอาท์พุทเพียง 15 บิทเท่านั้น เพราะในบิทต่ำสุดจะใช้สัญญาณจาก Clock มาเป็นบิทต่ำสุดรวมกับเอาท์พุทที่ได้จากโปรแกรมนับ 15 บิทเป็น 16 บิท ซึ่งจะชี้ตำแหน่งในหน่วยความจำได้สูงสุดถึง 64 กิโลไบท์ (64 Kilo-byte)

เอาท์พุทจากโปรแกรมนับรวมกับ Clock ทั้งหมด 16 บิทนี้ จะถูกส่งไปเข้ายังอินพุทของไอซี เบอร์ 74C244 ซึ่งเป็นบัฟเฟอร์ (Buffer) แบบ 3 ระดับ (Tri - State) โดยมีขา G1 และ G2 ของไอซีเป็นส่วนควบคุม ถ้า G1 และ G2 เป็น 1 (High) สัญญาณจากอินพุทจะไม่สามารถผ่านออกไปยังเอาท์พุทของบัฟเฟอร์ได้ และที่เอาท์พุทของบัฟเฟอร์จะมีความต้านทานสูง (High Impedance output State) และเมื่อขา G1 และ G2 เป็น 0 (Low) สัญญาณของอินพุทจะถูกส่งออกไปยังเอาท์พุทของบัฟเฟอร์ได้

ในการออกแบบโดยใช้บัฟเฟอร์คั่นกลางระหว่างโปรแกรมนับกับหน่วยความจำเพื่อต้องการที่จะแยกหน่วยความจำออกจากวงจรต่าง ๆ เสมือนหนึ่งว่าหน่วยความจำลอยอยู่ไม่ได้ต่อกับวงจรอื่นเลย โดยจะใช้สำหรับตอนช่วงโปรแกรมคำสั่งต่าง ๆ เข้ามายังหน่วยความจำ จึงจำเป็นต้องให้หน่วยความจำเป็นอิสระ และสามารถควบคุมจากภายนอกได้ และข้อดีอีกอย่างหนึ่งในการใช้บัฟเฟอร์ก็คือ ตัวบัฟเฟอร์นี้สามารถให้กระแสเอาท์พุทได้สูง จึงทำให้นำอินพุทจากไอซีตัวอื่น ๆ หลาย ๆ อินพุทมาต่อกับเอาท์พุทของบัฟเฟอร์ได้โดยตรง

เอาท์พุทจากบัฟเฟอร์นี้มีทั้งหมด 16 บิท คือจาก A0 ถึง A15 และหน่วยความจำที่ใช้คือ 6116 ซึ่งเป็น RAM หรือ 2716 ซึ่งเป็น ROM โดยใน 1 ตัว มีความจุ 2 กิโลไบท์ ซึ่งมีขาแอดเดรสทั้งหมด 11 บิท คือจาก A0 ถึง A10 โดยไอซีหน่วยความจำทั้งหมดนี้ขา A0 ถึง A10 จะถูกต่อขนานถึงกันหมด และเวลาใช้งานไอซีหน่วยความจำจะถูกใช้ทีละ 1 ตัวโดยมีขาชิพซีเลคต์ (CS) เป็นตัวเลือก ถ้าขา CS เป็น 0(low) ไอซีตัวนั้นจะทำงาน และตัวอื่นจะไม่ทำงานหรือถูกลอยเอาไว้ ขา CS นี้จะถูกควบคุมจากไอซีเบอร์ 74C154 ซึ่งทำหน้าที่แปลตำแหน่งจากอินพุท 4 บิทเป็นเอาท์พุท 16 ตำแหน่ง ในการออกแบบนี้จะใช้ไอซี 74C154 2 ตัว ซึ่งจะสามารถแปลตำแหน่งได้ทั้งหมด 32 ตำแหน่งจากอินพุท 5 บิท โดย 4 บิทแรก A11 ถึง A14 นั้นจะต่อเข้ายังขา A, B, C และ D ของไอซีทั้งสองตัว ส่วน A15 นั้นจะต่อเข้ายังขา G1 และ G2 ของไอซีตัวแรกโดยตรง ส่วนขา G1 และ G2 ของไอซีตัวที่สองจะต่ออยู่กับ $\overline{A15}$ ไอซีตัวแรกจะทำหน้าที่เลือก RAM หรือ ROM ตัวที่ 1 ถึง 16 และไอซีตัวที่ 2 จะเลือก RAM หรือ ROM ตัวที่ 17 ถึง 32

ในการใช้งานอาจจะใช้โปรแกรมการทำงานจาก RAM หรือ ROM ก็ได้ โดยจะมีสวิทช์เป็นตัวเลือก สวิทช์นี้จะต่ออยู่ที่ขา OE (Output Enable) ของ RAM หนึ่งขา และของ ROM หนึ่งขา และมีค่าความต้านทานขนาด 1 K ต่ออยู่กับขา OE ไปเข้าไฟ 5v. เมื่อขา OE ของ RAM หรือ ROM ถูกเลือกหมายถึงว่าถูกต่อลงกราวด์อยู่ เช่นขา OE ของ RAM ต่อลงกราวด์ RAM จะทำงานและ ROM จะไม่ทำงานเสมือนหนึ่งว่าลอยอยู่

ขาข้อมูล (Data) D0 ถึง D7 ของ RAM และ ROM จะต่อขนานถึงกันทุกตัวและต่อไปเข้าไอซี 74C244 ซึ่งเป็นบัฟเฟอร์ หรือเป็นการขั้นกลางระหว่างข้อมูลจากหน่วยความ

จำไปยังอินพุทไอซีตัวอื่น ๆ โดยมีขา G1 และ G2 เป็นขาควบคุม ซึ่งขาของ G1 และ G2 นี้จะต่ออยู่กับสวิทช์เพื่อเป็นการเลือกว่าจะทำการโปรแกรม หรือเป็นการทำงาน (RUN) อยู่ ถ้าถูกเลือกให้อยู่ในโหมดของการโปรแกรม ที่ G1 และ G2 จะมีสภาวะเป็น 1 (High) ข้อมูล D0 ถึง D7 จะไม่สามารถผ่านบัฟเฟอร์ออกไปได้ และเมื่อสวิทช์ให้อยู่ในโหมดการ RUN G1 และ G2 จะมีสภาวะเป็น 0 (Low) ข้อมูล D0 ถึง D7 จะถูกส่งผ่านบัฟเฟอร์ออกไปได้ ข้อมูลขนาด 8 บิทในหน่วยความจำนี้ 4 บิทแรกคือ จาก D4 ถึง D7 ในตำแหน่งแรกจะเป็นคำสั่ง และ 4 บิทหลังคือ จาก D0 ถึง D3 จะเป็นตัวชี้ตำแหน่งของอินพุทหรือเอาท์พุท ซึ่งจะส่งไปเก็บไว้ชั่วคราวที่ไอซี เบอร์ 4042 เมื่อสัญญาณนาฬิกาที่ขา 14 หรือ X1 ของหน่วยประมวลผลกลางตกลงมาเป็น 0 (Low) แล้ว ข้อมูล D0 ถึง D7 จากหน่วยความจำจะถูกเปลี่ยนไป 1 ตำแหน่ง ข้อมูลขนาด 8 บิทจาก D0 ถึง D7 นี้ก็จะเป็นตัวชี้ตำแหน่งของอินพุท หรือเอาท์พุทเช่นกัน เมื่อรวมกับ 4 บิทที่ถูกเก็บค่าไว้ในไอซี 4042 ก็จะเป็นตำแหน่งของอินพุทหรือเอาท์พุทขนาด 12 บิท ซึ่งเมื่อนำไปแปลตำแหน่งของอินพุทหรือเอาท์พุทจะได้สูงถึง 4096 ตำแหน่ง

จากข้อมูลที่ใช้แปลตำแหน่งของอินพุทหรือเอาท์พุทขนาด 12 บิทจาก D0 ถึง D11 นี้จะถูกแบ่งออกเป็น 3 ส่วน คือ

ข้อมูล D0 ถึง D3 จำนวน 3 บิทนี้จะถูกส่งไปขา A, B, C หรือขา A0,A1,A2 ของไอซีเลือกอินพุทเบอร์ 4512 หรือไอซีเลือกเอาท์พุทเบอร์ 4099 ข้อมูลขนาด 3 บิทนี้จะแปลตำแหน่งของอินพุทหรือเอาท์พุทได้ทั้งหมด 8 ตำแหน่ง โดยจะเลือกอินพุทหรือเอาท์พุทได้ครั้งละ 1 ตำแหน่ง จากไอซีอินพุทหรือเอาท์พุทตัวใด ๆ ก็ได้ โดยจะมีขา W (Write) ของไอซีเอาท์พุทหรือขา dis (disable) ของไอซีอินพุทเป็นตัวควบคุมอีกทีหนึ่ง

ส่วนข้อมูล D3 ถึง D6 ขนาด 4 บิทนี้จะต่อเข้ายังอินพุท A, B, C และ D ของไอซี เบอร์ 74C154 สำหรับ D7 และ $\overline{D7}$ จะต่อเข้าขา G2 ของไอซีแต่ละตัว

ข้อมูล D8 ถึง D11 จะต่อเข้าขา A,B,C และ D ของไอซี เบอร์ 74C154 อีกตัวหนึ่ง โดยเอาท์พุทของไอซีจาก S1 ถึง S16 นี้จะส่งไปเข้าขา G1 ของไอซี เบอร์ 74C154 ตัวที่รับข้อมูล D3 ถึง D6 ว่าจะเลือกตัวไหนใน 16 ชุด ในการทำงานแต่ละครั้ง

ไอซีที่ใช้แปลตำแหน่งของอินพุทหรือเอาท์พุทเบอร์ 74C154 นี้เป็นส่วนของ Base Unit ส่วนไอซีเบอร์ 4099 หรือ 4512 นั้นเป็นส่วนของ Expansian Unit

ไอซีเบอร์ 74C154 ที่รับข้อมูลทางอินพุทจาก D3 ถึง D6 นี้จะให้เอาท์พุทออกมาสำหรับนำไปเลือกไอซีอินพุทหรือเอาท์พุทตัวใด ๆ โดยนำเอาเอาท์พุทจากไอซี เบอร์ 74C154 ผ่าน Inverter ไปเข้าอินพุทของ NAND Gate ถ้าเป็นการเลือกไอซีอินพุทจะใช้ 3 อินพุท NAND Gate โดย 2 อินพุทที่เหลือของ NAND Gate จะต่ออยู่กับสัญญาณ W (Write) และ CLK (Clock) และถ้าเป็นการเลือกไอซีเอาท์พุทจะใช้ 2 อินพุท NAND Gate โดยอินพุทที่เหลืออีกหนึ่งอินพุทนั้นจะต่ออยู่กับสัญญาณ W (Write) และนำเอาท์พุทที่ได้จาก NAND Gate นี้ต่อไปเข้าขาควบคุมของไอซีอินพุทหรือเอาท์พุท โดยต่อไปเข้าที่ขา dis ของไอซีอินพุท หรือขา W ของไอซีเอาท์พุท

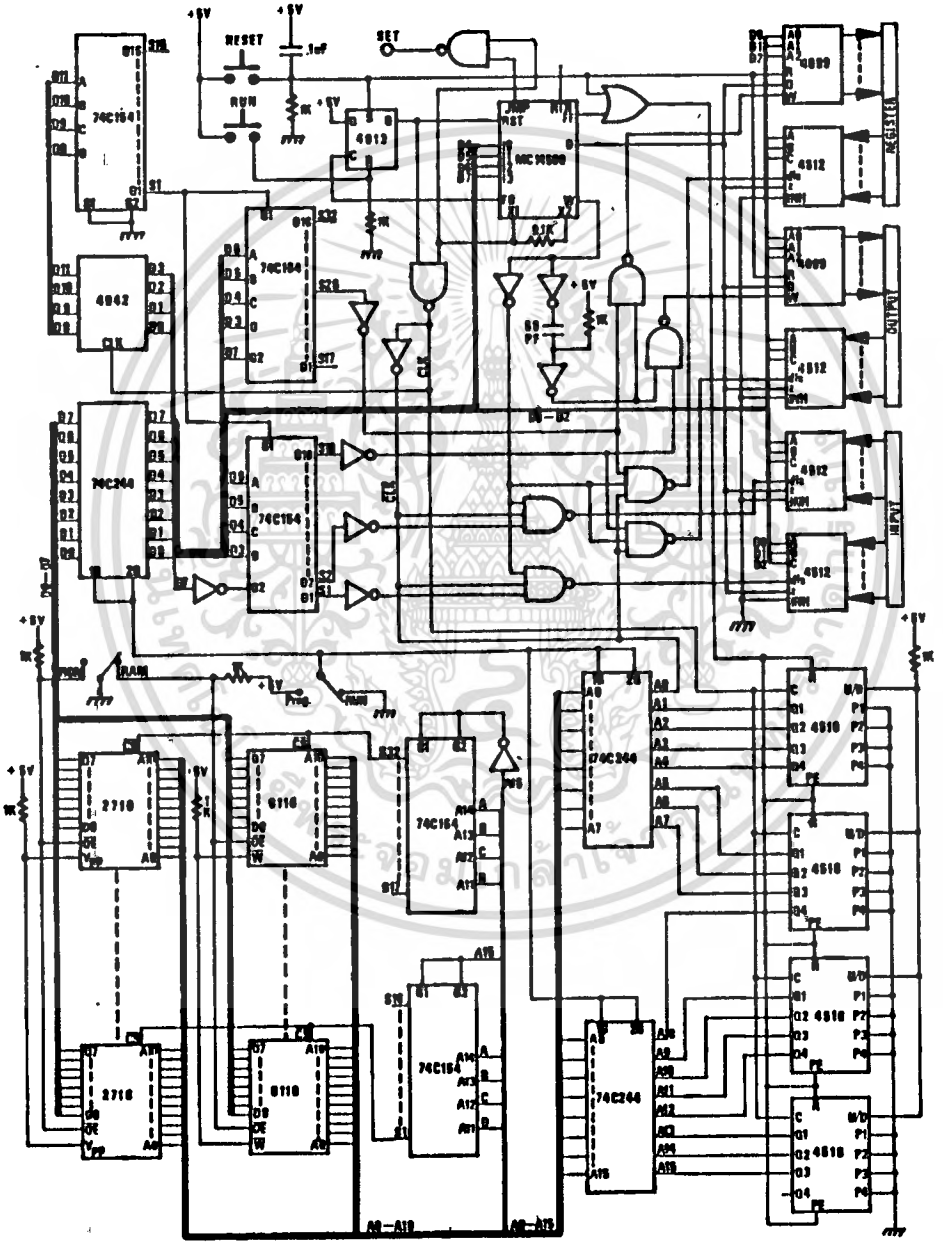
ในการทำงานของพีแอลซีนี้จะมีสวิทช์ของการ RUN คือเป็นการเริ่มต้นให้พีแอลซีทำงาน และสวิทช์ Reset เป็นการทำให้พีแอลซีหยุดทำงาน โดยสวิทช์ RUN จะต่ออยู่ที่ขา R ของ D Flip Flop ส่วนสวิทช์ Reset จะต่ออยู่ที่ขา S ของ D Flip Flop ตัวเดียวกัน เมื่อกดสวิทช์ RUN จะทำให้เอาท์พุท Q ของ D Flip Flop เป็น 0 (Low) ซึ่งขา Q นี้จะต่ออยู่กับขา RST ของหน่วยประมวลผลกลาง เมื่อขา RST ของหน่วยประมวลผลกลางเป็น 0 (Low) หน่วยประมวลผลกลางจะเริ่มทำงาน และเมื่อกดสวิทช์ Reset ที่เอาท์พุท Q ของ D Flip Flop เป็น 1 (High) ที่ขา RST ของหน่วยประมวลผลกลางก็จะเป็น 1 (High) หน่วยประมวลผลกลางก็จะหยุดทำงาน

ที่ขา D (Data) ของหน่วยประมวลผลกลางจะต่ออยู่กับขา D ของไอซีอินพุทหรือเอาท์พุททุกตัวเพื่อรับข้อมูลหรือส่งข้อมูลออกไป

ที่ขา W (Write) จะส่งสัญญาณควบคุมไปยังไอซีเอาท์พุท โดยจะมีค่าความต้านทาน R และค่าของ C ต่อคั่นอยู่เพื่อจะทำให้สัญญาณ W แคบเข้ามาอีก เพราะความกว้างของสัญญาณ W จะเกินไปในส่วนของสัญญาณนาฬิกาในช่วง 1 (High) จึงจำเป็นต้องลดความกว้างของสัญญาณนาฬิกาให้อยู่ในช่วงของสัญญาณนาฬิกา 0 (Low) เท่านั้น

ในการออกแบบวงจรนี้จะมีที่สำหรับเสียบไอซีหน่วยความจำหรือขา Slot สำหรับ

เสียบชุดอินพุท เอาท์พุท รีจิสเตอร์ Timer หรือ Counter ถ้าต้องการเพิ่มส่วนใด ๆ ก็สามารถนำมาเสียบเพิ่มได้เลย



รูปที่ 5.1 วงจรของเครื่องพีแอลซี

บทที่ 6
ทฤษฎีและการเขียนโปรแกรมของพีแอลซี

6.1 ทฤษฎีของเครื่องควบคุมแบบโปรแกรมลอจิก (PLC)

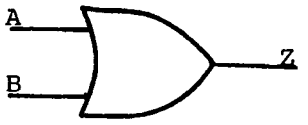
เครื่องควบคุมแบบโปรแกรมลอจิกนี้ ใช้ควบคุมระบบที่มีการทำงานแบบเปิดกับปิด (ON-OFF) ที่ต่อเนื่องหรือซีเควนซ์ การทำงานแบบต่อเนื่องนี้เป็นแบบลอจิก โดยที่ผู้ใช้เขียนโปรแกรมคำสั่งทางลอจิกให้กับเครื่องควบคุม คำสั่งพื้นฐานทั่ว ๆ ไปทางลอจิกของเครื่องควบคุมแบบโปรแกรมนี้มีอยู่ 3 คำสั่งคือ

1. AND เอาท์พุทจะมีผลลัพธ์ หรือสภาวะเป็น 1 ได้ก็ต่อเมื่ออินพุททั้งหมดมีสภาวะเป็น 1

Z = A.B

INPUT A	INPUT B	OUTPUT Z
0	0	0
0	1	0
1	0	0
1	1	1

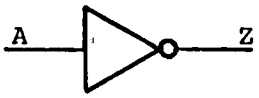
2. OR เอาท์พุทจะมีผลลัพธ์ หรือสภาวะเป็น 1 ได้ก็ต่อเมื่ออินพุทอันใดอันหนึ่งเป็น 1



Z = A + B

INPUT A	INPUTB	OUTPUT Z
0	0	0
0	1	1
1	0	1
1	1	1

3. NOT เอาท์พุทจะมีสภาวะตรงข้ามกับอินพุทเสมอ



Z = A

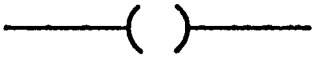
INPUT A	OUTPUT Z
0	1
1	0

ในการเขียนโปรแกรมสำหรับการทำงานของระบบควบคุมแบบซีเควนซ์นั้นมีลำดับขั้นการเขียนดังนี้คือ เขียนระบบการทำงานในรูปของแลดเดอร์ไดอะแกรมก่อน แล้วจึงเขียนคำสั่งของโปรแกรมจากแลดเดอร์ไดอะแกรมอีกทีหนึ่ง การเขียนแลดเดอร์ไดอะแกรมมีลักษณะดังต่อไปนี้

เขียนลักษณะการทำงานของอินพุทในรูปของหน้าหน้าสัมผัส (Contact) ของสวิทช์ซึ่งนำมาต่อกันเป็นวงจร โดยเรียกการเขียนแบบนี้ว่าแลดเดอร์ไดอะแกรม ในการเขียนแลดเดอร์ไดอะแกรมสำหรับเครื่องควบคุมแบบโปรแกรมลอจิกนี้จะถือว่าปกติแล้วอินพุทจะเป็นแบบปกติเปิด (Normally Open:N.O.) อยู่ ในการเขียนแลดเดอร์ไดอะแกรมจะใช้สัญลักษณ์พื้นฐานต่อไปนี้แทนอุปกรณ์ต่าง ๆ ของอินพุทและเอาท์พุท

สัญลักษณ์นี้จะใช้แทนอุปกรณ์ทางด้านอินพุทต่าง ๆ เช่น สวิทช์ไฟฟ้า หน้าสัมผัสของรีเลย์ ตัวตั้งเวลา ตัวนับ หรือรีจิสเตอร์ โดยที่อุปกรณ์ปกติจะมีค่าสภาวะเป็น 0 อยู่

สัญลักษณ์นี้จะใช้แทนอุปกรณ์ทางด้านอินพุทต่าง ๆ เช่น สวิทช์ไฟฟ้า หน้าสัมผัสของรีเลย์ ตัวตั้งเวลา ตัวนับ หรือรีจิสเตอร์ โดยที่อุปกรณ์นั้นปกติจะมีค่าสภาวะเป็น 1 อยู่



สัญญลักษณ์นี้จะใช้แทนอุปกรณ์ทางด้านเอาท์พุทต่าง ๆ เช่นหลอดไฟฟ้า มอเตอร์ไฟฟ้า ขดลวดของรีเลย์ ขดลวดโซลีนอยด์ ตัวตั้งเวลา ตัวนับ หรือรีจิสเตอร์ โดยที่อุปกรณ์เอาท์พุทนั้นปกติจะมีค่าสภาวะเป็น 0 อยู่

สัญญลักษณ์นี้จะใช้แทนอุปกรณ์ทางด้านเอาท์พุทต่าง ๆ เช่น หลอดไฟฟ้า มอเตอร์ไฟฟ้า ขดลวดรีเลย์ ขดลวดโซลีนอยด์ ตัวตั้งเวลาหรือตัวนับ โดยที่ปกติแล้วอุปกรณ์ทางด้านเอาท์พุทจะมีค่าสภาวะเป็น 1 อยู่

ตัวอย่างการเขียนแลดเดอร์ไดอะแกรมแทนลอจิกเกท

1. แลดเดอร์ไดอะแกรมแทนลอจิก AND เกท

2. แลดเดอร์ไดอะแกรมแทนลอจิก OR เกท

3. แลดเดอร์ไดอะแกรมแทนลอจิก NOT เกท

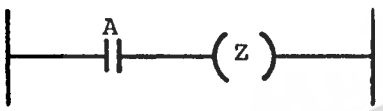
6.2 คำสั่งของเครื่องพีแอลซีโดยใช้หน่วยประมวลผลกลางขนาด 1 บิท

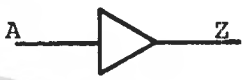
คำสั่งต่าง ๆ ของเครื่องควบคุมแบบโปรแกรมลอจิกมีการทำงานและการเขียนแลดเดอร์ไดอะแกรมแทนลอจิก และการเขียนคำสั่งจากแลดเดอร์ไดอะแกรมดังนี้

6.2.1 คำสั่ง LD และ LDC

คำสั่งนี้ใช้ในการนำค่าสภาวะจากอินพุทเข้ามาเป็นค่าเริ่มต้นของการทำคำสั่งต่อไปภายในหน่วยประมวลผลกลาง

คำสั่ง LD

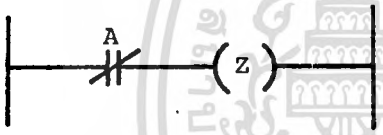




LD A

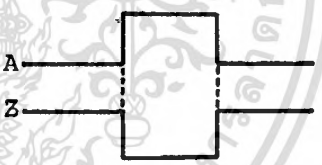
STO Z

คำสั่ง LDC



LDC A

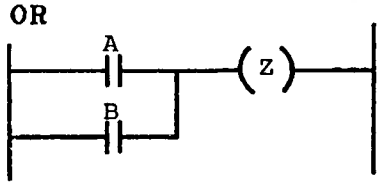
STO Z



6.2.2 คำสั่ง OR และ ORC

คำสั่งนี้ใช้ในการนำค่าสภาวะจากอินพุทเข้ามาทำการ OR กับค่าสภาวะเดิมภายในหน่วยประมวลผลกลาง

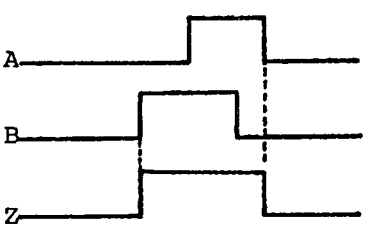
คำสั่ง OR



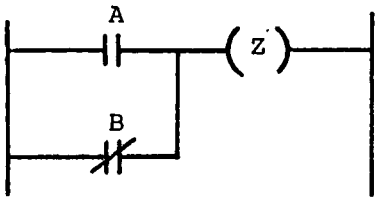
LD A

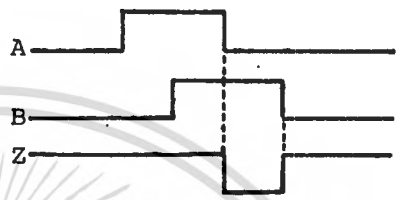
OR B

STO Z



คำสั่ง ORC



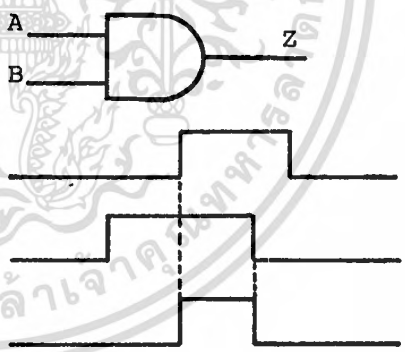



```
LD    A
ORC   B
STO   Z
```

6.2.3 คำสั่ง AND และ ANDC

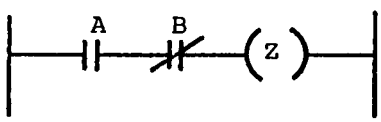
คำสั่งนี้เป็นการนำค่าสภาวะจากอินพุทเข้ามาทำการ AND กับค่าสภาวะเดิมภายในหน่วยประมวลผลกลาง

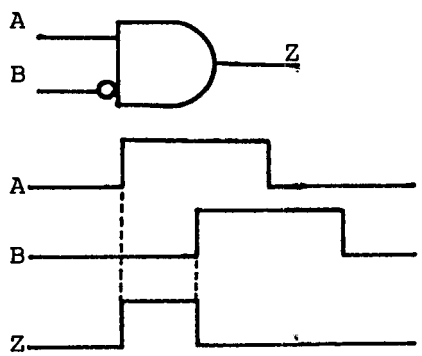
คำสั่ง AND




```
LD    A
AND   B
STO   Z
```

คำสั่ง ANDC



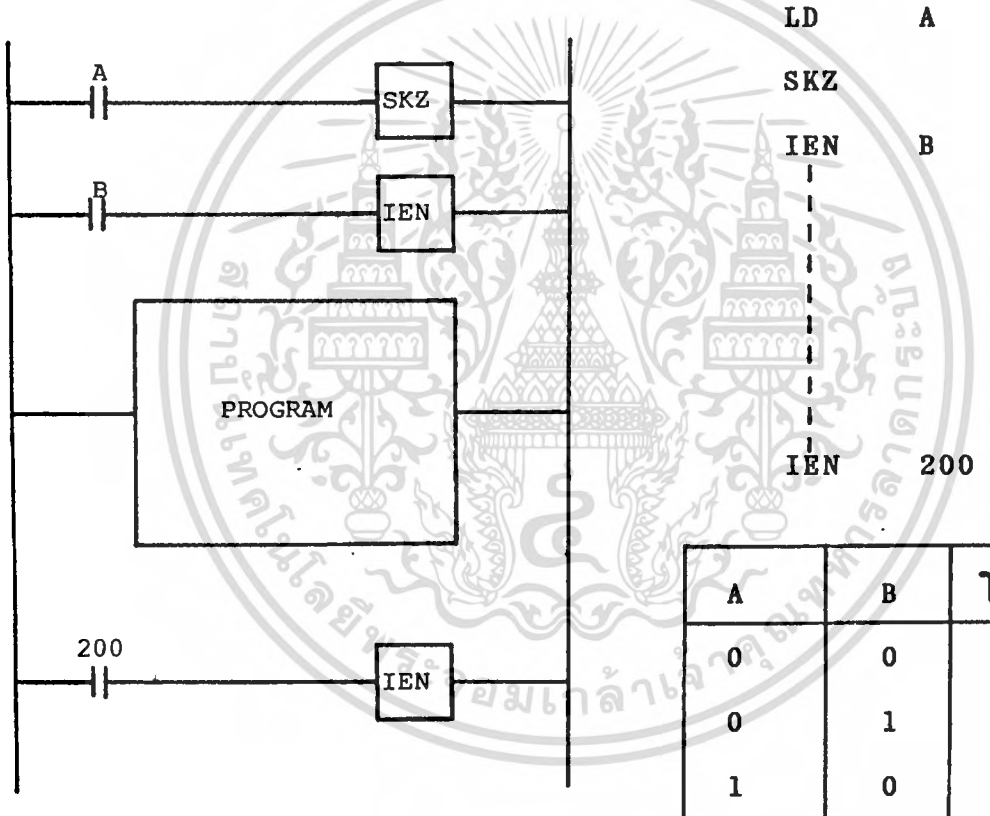



```
LD    A
ANDC  B
STO   Z
```

6.2.4 คำสั่ง SKZ

การใช้คำสั่ง SKZ จะใช้สำหรับข้ามคำสั่ง 1 คำสั่งโดยมีเงื่อนไขดังนี้เมื่อหน่วยประมวลผลกลางได้รับคำสั่ง SKZ หน่วยประมวลผลกลางจะตรวจดูสภาวะภายในรีจิสเตอร์ผลลัพธ์ ถ้าเป็น 0 คำสั่งต่อมาอีก 1 คำสั่งจะถูกข้าม แต่ถ้าเป็น 1 คำสั่งต่อมาจะถูกประมวลผลตามปกติดังตัวอย่างแลดเดอร์ไดอะแกรม และตารางการทำงานดังนี้

ตัวอย่างการใช้คำสั่ง SKZ



A	B	โปรแกรม
0	0	ทำ
0	1	ทำ
1	0	ไม่ทำ
1	1	ทำ

คำสั่ง SKZ จากแลดเดอร์ไดอะแกรม จะเห็นว่าโปรแกรมที่อยู่ภายในคำสั่ง IEN จะทำงานตามปกติยกเว้นเมื่ออินพุท A เป็น 1 และ B เป็น 0 โปรแกรมภายใน IEN จะถูกข้ามไป เงื่อนไขหลักของการทำโปรแกรมคืออินพุท A ส่วนเงื่อนไขรองคืออินพุท B

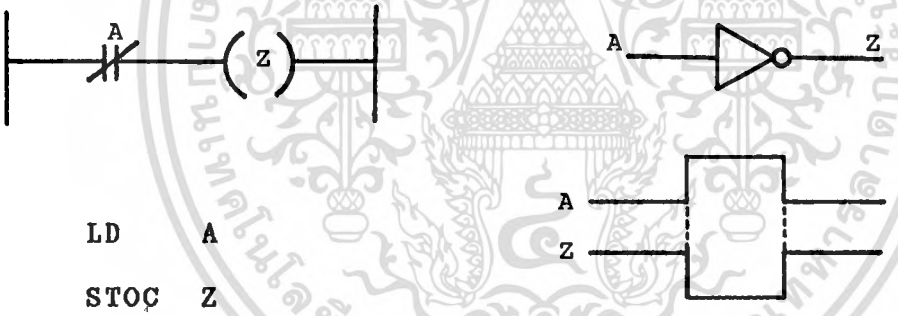
6.2.5 คำสั่ง STO และ STOC

คำสั่งนี้ใช้สำหรับส่งค่าสภาวะของผลลัพธ์จากการประมวลผลของหน่วยประมวล-ผลกลางออกไปยังเอาท์พุท

คำสั่ง STO


```
LD    A
STO   Z
```

คำสั่ง STOC

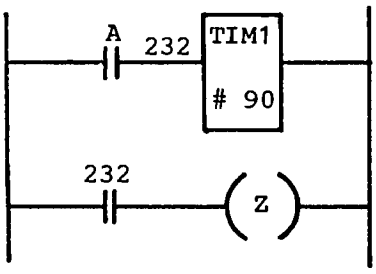


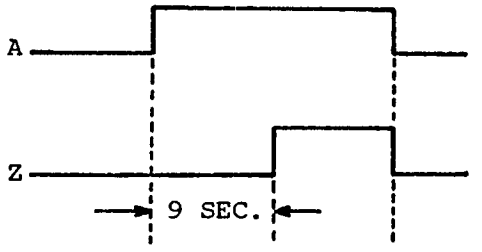

```
LD    A
STOC  Z
```

6.2.6 คำสั่ง Timer

คำสั่งนี้ใช้สำหรับหน่วยเวลาของค่าสภาวะจากหน่วยประมวลผลกลางที่ส่งออกไปยังเอาท์พุท

ตัวอย่างการใช้คำสั่ง Timer





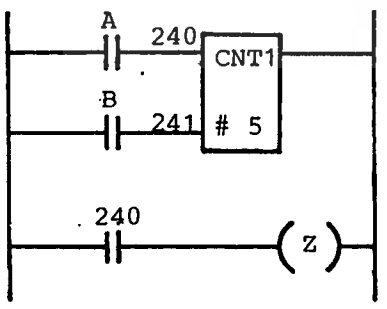
```
LD    A
STO   232 (TIM 1)
SET   90
LD    232 (TIM 1)
STO   Z
```

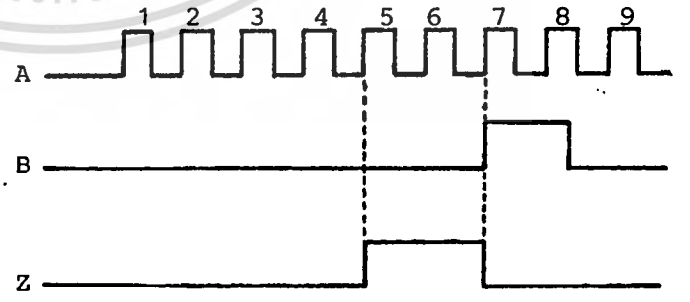
การทำงานของ Timer จะเป็นตัวหน่วงเวลาจากอินพุทสู่เอาท์พุท โดย Timer ตัวที่ 1 นั้นอยู่ที่ตำแหน่งของเอาท์พุทที่กำหนด คือตำแหน่งที่ 232 และการตั้งค่าเวลาใช้คำสั่ง SET โดยตั้งได้จาก 000 ถึง 255 และนำตัวเลขของการตั้งค่าเวลามาคูณกับ 0.1 จะได้เวลาของการหน่วงเป็นวินาที ตัวตั้งเวลามีทั้งหมด 8 ตัวใน 1 ชุด คือจาก Timer 1 ถึง Timer 8 โดยมีตำแหน่งของเอาท์พุทที่เป็น Timer จาก 232 ถึง 239 และเวลาเขียนโปรแกรมจะต้องอ้างถึงตำแหน่งของ Timer จริง ๆ

6.2.7 คำสั่ง Counter

คำสั่งนี้ใช้สำหรับนับจำนวนอินพุทที่เข้ามา และเมื่อนับจำนวนครั้งของอินพุทครบตามที่ตั้งให้ Counter นับแล้ว Counter จะส่งเอาท์พุทออกมา และเมื่อต้องการเริ่มนับใหม่จะต้องรีเซทค่าใน Counter ให้เป็นศูนย์เสียก่อนจึงเริ่มนับใหม่ได้

ตัวอย่างการใช้คำสั่ง Counter





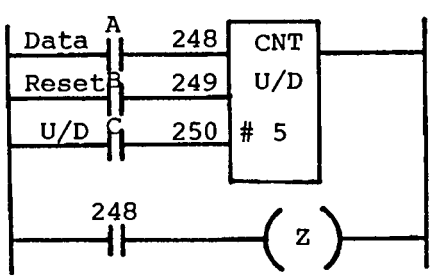
```
LD     A
STO    240 (CNT 1)
LD     B
STO    241 (CNTR 1)
SET    5
LD     240 (CNT 1)
STO    Z
```

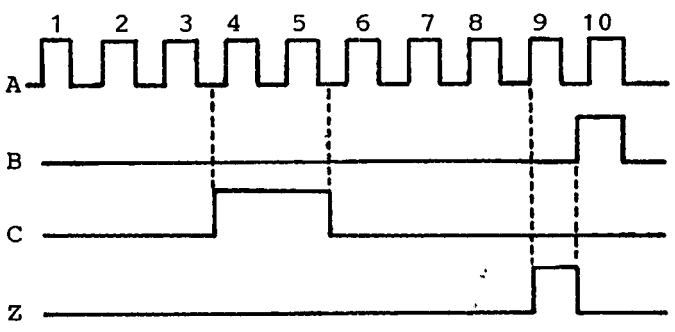
การทำงานของ Counter เมื่อมีอินพุท A เข้ามา 5 ครั้ง จะได้เอาท์พุท Z ออกไป และเมื่อ B เป็น 1 จะเป็นการรีเซทให้ Counter เริ่มทำการนับใหม่ การตั้งจำนวนนับของ Counter สามารถตั้งได้จาก 000 ถึง 255 และตำแหน่งของ Counter จะเริ่มจาก 240 ถึง 247 โดยมีทั้งหมด 4 ตัวใน 1 ชุด

6.2.8 คำสั่ง U/D Counter

คำสั่ง U/D Counter นี้จะมีการลบค่าของการนับออกได้ด้วย คือเมื่ออินพุทที่ใช้สำหรับเลือกการนับเป็น 0 U/D Counter จะนับค่าอินพุท Data ไปเรื่อย ๆ จนครบจำนวนที่ตั้งให้นับ U/D Counter จึงจะให้เอาท์พุทออกมา แต่ถ้าขณะกำลังนับค่าของอินพุทอยู่ อินพุทที่ใช้สำหรับเลือกการนับเป็น 1 จะเป็นการลบจำนวนนับออกทีละ 1 ตามอินพุท Data ที่เข้ามา และเมื่ออินพุทที่ใช้สำหรับเลือกการนับเป็น 0 ก็จะเริ่มนับจากจำนวนนับเดิมขึ้นไปอีกจนกว่าจะครบจำนวนที่ตั้งเอาไว้

ตัวอย่างการใช้คำสั่ง U/D Counter





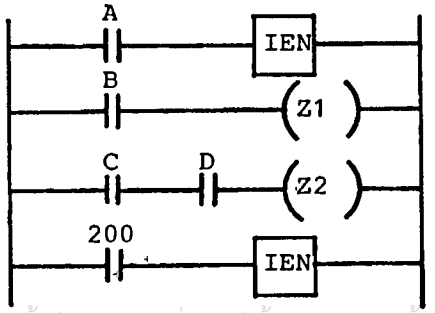
```
LD    A
STO   248 (CNT U/D 1)
LD    B
STO   249 (CNTR U/D 1)
LD    C
STO   250 (U/D)
SET   5
LD    248
STO   Z
```

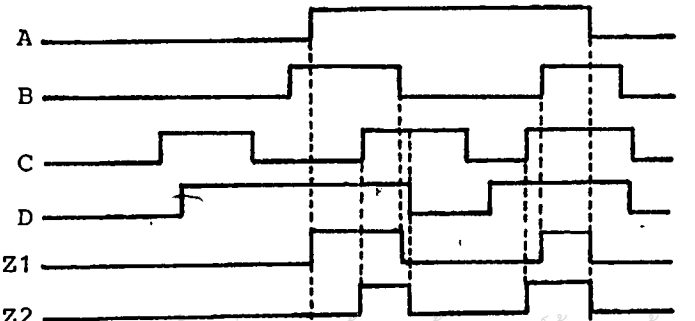
การทำงานของ U/D Counter นั้นเมื่อนับอินพุท A ได้ 3 ครั้งแล้ว ขา U/D มีสภาวะเป็น 1 จึงเป็นการลบค่าของการนับออก. ซึ่งในขณะที่ค่าสภาวะของขา U/D เป็น 1 อยู่นั้นมีอินพุทเข้ามาอีก 2 ครั้ง และขา U/D กลับไปเป็น 0 อีก ค่าการนับใน U/D Counter จะเหลืออยู่ 1 เพราะฉนั้นจะต้องนับอินพุทอีก 4 ครั้งจึงจะเท่ากับจำนวนที่ตั้งไว้ และเมื่อมีสัญญาณรีเซท จึงจะเป็นการเริ่มต้นนับอินพุทใหม่ ตำแหน่งของเอาท์พุทของ U/D Counter ที่มีอยู่ 2 ตัวใน 1 ชุดนั้นอยู่ที่ตำแหน่ง 248 ถึง 253 และสามารถตั้งจำนวนนับได้จาก 000 ถึง 255

6.2.9 คำสั่ง IEN สำหรับข้ามโปรแกรม

เงื่อนไขของการทำโปรแกรม ถ้าอินพุทที่เป็นเงื่อนไขของ IEN เป็น 0 โปรแกรมที่อยู่ระหว่าง IEN จะถูกข้าม แต่ถ้าอินพุทที่เป็นเงื่อนไขของ IEN เป็น 1 โปรแกรมที่อยู่ภายใน IEN จะถูกทำตามคำสั่งเป็นปกติ

ตัวอย่างการใช้คำสั่ง IEN สำหรับการข้ามโปรแกรม





```
IEN   A
LD    B
STO   Z1
LD    C
AND   D
STO   Z2
IEN   200
```

จากแลดเดอร์ไดอะแกรม เมื่อ A เป็น 1 อินพุท B,C,D และเอาท์พุท Z1 และ Z2 จะทำคำสั่งตามปกติ และเมื่อ A เป็น 0 เอาท์พุท Z1 และ Z2 จะไม่มีการเปลี่ยนแปลงสภาวะจากเดิม ในขณะที่อินพุท B,C และ D มีการเปลี่ยนแปลงตามเงื่อนไข เอาท์พุท Z1 กับ Z2 จะไม่เปลี่ยนแปลง และที่ตำแหน่งรีจิสเตอร์ที่ 200 มีสภาวะเป็น 1 อยู่ จะใช้สำหรับการสิ้นสุดเงื่อนไขของ IEN ซึ่งโปรแกรมหลังจาก IEN จะทำคำสั่งตามปกติ การใช้คำสั่ง IEN ในโปรแกรมจะใช้กี่ครั้งก็ได้

6.2.10 คำสั่ง IEN,OEN สำหรับจุดเริ่มต้นของโปรแกรม

ตอนเริ่มต้นของโปรแกรมจะต้องมีคำสั่ง IEN และ OEN เสมอ เพราะว่าตอนเปิดเครื่องควบคุมแบบโปรแกรมลอจิกครั้งแรก ค่าสภาวะของรีจิสเตอร์ IEN อาจจะเป็น 0 หรือ 1 อยู่ก็ได้ จึงต้องทำการเซทให้รีจิสเตอร์ IEN มีค่าสภาวะเป็น 1 เสียก่อนเพื่อที่จะเป็นการยอมให้อินพุทผ่านเข้ามายังหน่วยประมวลผลกลางได้ ส่วนคำสั่ง OEN มีลักษณะเหมือนคำสั่ง IEN คือเป็นทางผ่านของสัญญาณเขียน (Write) ถ้ารีจิสเตอร์ OEN เป็น 1 สัญญาณเขียนจะผ่านออกจากหน่วยประมวลผลกลางได้ แต่ถ้ารีจิสเตอร์ OEN เป็น 0 สัญญาณเขียนจะไม่สามารถผ่านออกไปได้ การตั้งค่าสภาวะของรีจิสเตอร์ IEN และ OEN โดยการเรียกจากรีจิสเตอร์ตำแหน่งที่ 200 ซึ่งมีค่าสภาวะเป็น 1 อยู่ ตัวอย่างของโปรแกรมตอนเริ่มต้น

```
IEN    200
```

```
OPN     200
  |
  |
  |
  |
  |
  |
  |
END
```

6.2.10 คำสั่ง END

คำสั่งนี้ใช้สำหรับสิ้นสุดโปรแกรม เมื่อสิ้นสุดโปรแกรมแล้วไม่มีคำสั่ง END อยู่ เครื่องควบคุมแบบโปรแกรมลอจิกจะไม่สามารถทำงานได้ เพราะว่าเมื่อสิ้นสุดโปรแกรมแล้วจะต้องกลับมาเริ่มต้นทำโปรแกรมที่จุดเริ่มต้นใหม่ ถ้าไม่มีคำสั่ง END เครื่องจะไม่รู้จุดสิ้นสุดของโปรแกรม คำสั่งที่อยู่หลังจุดสิ้นสุดของโปรแกรมอาจเป็นเงื่อนไขที่เครื่องไม่สามารถทำงานได้ และคำสั่ง END นี้จะเป็นการบอกให้หน่วยประมวลผลกลางส่งสัญญาณไปรีเซทให้หน่วยนับโปรแกรมกลับมาเริ่มต้นใหม่

6.3 ตำแหน่งและหน้าที่

ตำแหน่งต่าง ๆ ของอินพุท เอาท์พุท ตัวตั้งเวลา ตัวนับ และรีจิสเตอร์ใน 1 ชุดมีดังนี้

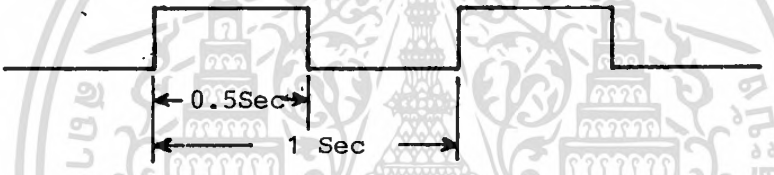
หน้าที่	ตำแหน่ง
อินพุท	0-119
เอาท์พุท	120-199
รีจิสเตอร์	201-230
รีจิสเตอร์พิเศษ	200,231,254,255
ตัวตั้งเวลา	232-239
ตัวนับ	240,242,244,246
ตัวนับขึ้นลง	248,251

รีจิสเตอร์ 200 จะมีค่าสภาวะเป็น 1 อยู่ ใช้สำหรับคำสั่ง IEN,OEN หรือคำสั่งของโปรแกรมที่ต้องการค่าสภาวะ 1 สามารถเรียกจากรีจิสเตอร์ 200 ไปใช้ได้

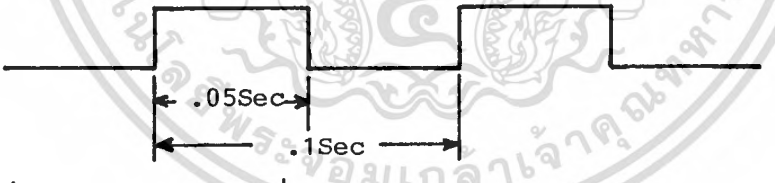
รีจิสเตอร์ 231 เป็นรีจิสเตอร์ที่เก็บค่าสภาวะของรีจิสเตอร์ผลลัพธ์ขณะนั้นไว้ ถ้าต้องการใช้ค่าสภาวะของรีจิสเตอร์ผลลัพธ์ขณะนั้น สามารถเรียกจากรีจิสเตอร์ 231 ไปใช้งานในคำสั่งได้

รีจิสเตอร์ 254 กับ 255 จะให้ความถี่เอาท์พุทออกมา โดยเอาท์พุทของรีจิสเตอร์ 254 จะมีความถี่ 1 เฮิรท์ซ และรีจิสเตอร์ 255 จะมีความถี่ 10 เฮิรท์ซ

เอาท์พุทของรีจิสเตอร์ 254 เป็นดังนี้



เอาท์พุทของรีจิสเตอร์ 255 เป็นดังนี้



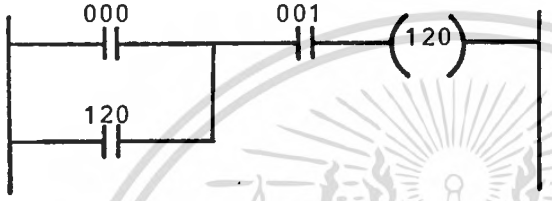
จะเห็นว่าตำแหน่งตัวนับที่ 240 242 244 และ 246 จะเป็นขาข้อมูลที่ส่งให้กับตัวนับนับค่า ส่วนตำแหน่งที่ 241 243 245 และ 247 จะเป็นขาสำหรับรีเซทค่าสภาวะของเอาท์พุทของตัวนับ ตัวนับ 1 ตัวจะมีตำแหน่งใช้สั่งงาน 2 ตำแหน่ง โดยแบ่งเป็นคู่ ๆ ดังนี้คือ 240 กับ 241 242 กับ 243 244 กับ 245 และ 246 กับ 247 รวมตัวนับทั้งหมด 4 ตัวใน 1 ชุด

ส่วนตัวนับแบบขึ้นลงทั้งหมดใน 1 ชุดมี 2 ตัว ใน 1 ตัวจะใช้ 3 ตำแหน่งคือ ตัวที่ 1 จะใช้ตำแหน่งของเอาท์พุทที่ 248 249 และ 250 ส่วนตัวที่ 2 มีตำแหน่งดังนี้คือ 251 252 และ 253 โดยตำแหน่งที่ 248 กับ 251 เป็นขาข้อมูล ส่วนตำแหน่ง

ที่ 249 กับ 252 เป็นตำแหน่งใช้เลือกการนับขึ้นหรือลง สำหรับตำแหน่งที่ 250 กับ 253 เป็นขารีเซท

6.3 ตัวอย่างแลดเดอร์ไดอะแกรมที่ซับซ้อนและการใช้คำสั่ง

1. การใช้คำสั่งสำหรับเซทและรีเซท

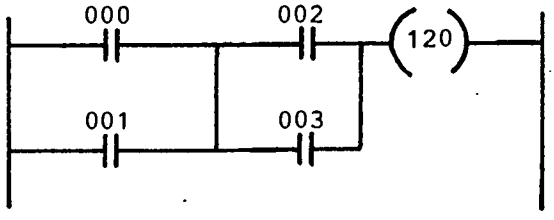



```
LD      000
OR      120
ANDC    001
STO     120
```

การทำงานของคำสั่งชุดนี้คือ เมื่ออินพุท 000 เป็น 1 ที่เอาท์พุท 120 จะเป็น 1 และเมื่ออินพุท 000 กลับมาเป็น 0 เอาท์พุท 120 จะเก็บค่าสภาวะของอินพุท 000 เอาไว้ เพราะฉนั้นเอาท์พุท 120 จะคงสภาวะ 1 อยู่จึงเป็นเงื่อนไขของการเซท และเมื่ออินพุท 001 มีสภาวะเป็น 1 จะเป็นการรีเซทเอาท์พุท 120 ให้มีสภาวะกลับเป็น 0 อีก

2. การใช้รีจิสเตอร์เก็บค่าสภาวะไว้ชั่วคราว

2.1

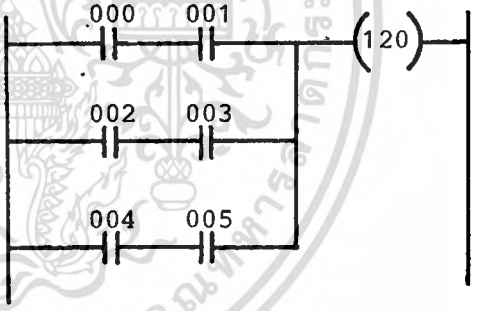


```
LD      000
OR      001
STO     201
LD      002
OR      003
AND     201
STO     120
```

เมื่อนำค่าของอินพุท 000 มา OR กับ 001 ผลลัพธ์ที่ได้นำไปเก็บไว้ที่รีจิสเตอร์ 201 แล้วนำค่าอินพุท 002 มา OR กับ 003 ผลลัพธ์ที่ได้นำไป AND กับค่าในรีจิสเตอร์ 201 แล้ว จึงนำผลลัพธ์สุดท้ายที่ได้ส่งไปยังเอาท์พุทที่ 120

2.2

```
LD      000
AND     001
STO     201
LD      002
AND     003
STO     202
LD      004
AND     005
OR      202
OR      201
STO     120
```

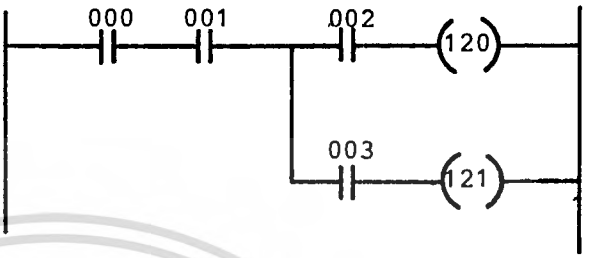


การทำงาน นำค่าของอินพุท 000 กับ 001 มา AND กันได้ผลลัพธ์ออกมาเก็บไว้ที่รีจิสเตอร์ 201 จากนั้นนำค่าอินพุท 002 กับ 003 มา AND กันได้ผลลัพธ์ออกมาเก็บไว้ที่รีจิสเตอร์ 202 จากนั้นนำค่าอินพุท 004 กับ 005 มา AND กัน แล้วนำผลลัพธ์ที่ได้

ไป OR กับรีจิสเตอร์ 202 กับ 201 แล้วนำผลลัพธ์สุดท้ายที่ได้ส่งไปยังเอาท์พุทที่ 120

2.3

```
LD      000
AND     001
STO     201
AND     002
STO     120
LD      201
AND     003
STO     121
```



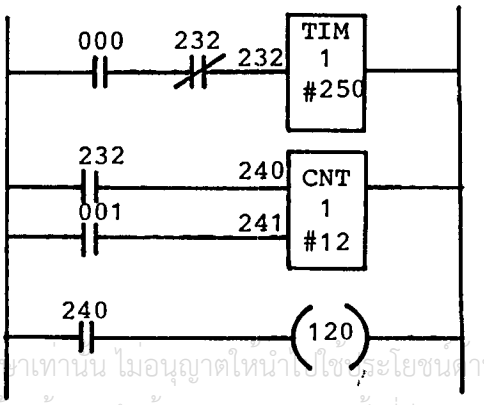
การทำงาน นำค่าของอินพุท 000 มา AND กับอินพุท 001 ผลลัพธ์ที่ได้นำไปเก็บไว้ที่รีจิสเตอร์ 201 และผลลัพธ์เดิมนำไป AND กับอินพุท 002 ได้ผลลัพธ์ส่งไปยังเอาท์พุทที่ 120 หลังจากนั้นนำค่าจากรีจิสเตอร์ 201 มา AND กับ 003 ได้ผลลัพธ์ส่งไปยังเอาท์พุทที่ 121

3. การใช้ตัวตั้งเวลาร่วมกับตัวนับเพื่อขยายการหน่วงเวลาให้ได้นานขึ้น

ตัวตั้งเวลาสามารถตั้งค่าการหน่วงเวลาได้สูงสุด 25.5 วินาที ถ้าต้องการเวลาของการหน่วงนาน ๆ เช่น 300 วินาที ก็สามารถกระทำได้โดยใช้เอาท์พุทของตัวตั้งเวลามาเข้าเป็นอินพุทของตัวนับอีกทีหนึ่ง ถ้าใช้ตัวตั้งเวลา 1 ตัวร่วมกับตัวนับ 1 ตัว จะได้เวลาของการหน่วงสูงสุดถึง 108.4 นาที หรือประมาณ 1 ชั่วโมง 48 นาที

ตัวอย่างการใช้ตัวตั้งเวลาร่วมกับตัวนับ

```
LD      000
ANDC    232     (TIM 1)
STO     232
SET     250
```



```
LD          232    (TIM 1)
STO         240    (CNT 1)
LD          001
STO         241    (CNTR 1)
SET         12
LD          240
STO         120
```

6.4 การเขียนคำสั่งแทนเกทชนิดต่าง ๆ

6.4.1 Non-Inverter

```
LD      A
STO     Z
```

6.4.2 Inverter

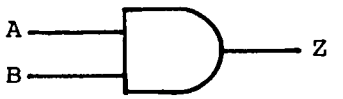
```
LD      A
STOC    Z
```

หรือ

```
LOC     A
STO     Z
```

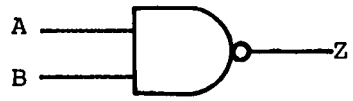
6.4.3 AND

```
LD      A
AND     B
STO     Z
```



6.4.4 NAND

```
LD      A
AND     B
STOC    Z
```



6.4.5 OR

```
LD      A
OR      B
STO     Z
```

6.4.6 NOR

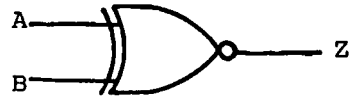
```
LD      A
OR      B
STOC    Z
```

6.4.7 XOR

```
LD      A
XNOR    B
STOC    Z
```

6.4.8 XNOR

```
LD      A
XNOR    B
STO     Z
```



บทที่ 7
การขยายขีดความสามารถสูงสุดของเครื่องพีแอลซีขนาด 1 บิท

7.1 การเพิ่มความถี่สัญญาณนาฬิกาสูงสุด

จากคุณสมบัติของหน่วยประมวลผลกลาง ระบุไว้ว่าความกว้างของสัญญาณนาฬิกา 1 ลูก (1 Clock) จะต้องไม่น้อยกว่า 400 นาโนเซคกันด์ (nS) ซึ่งจะได้ความถี่จากสูตรดังนี้

$$F = 1/T$$

ความถี่ = F (Frequency)

เวลา = T (Time)

ถ้า T = 400 nS

$$F = 2.5 \text{ MHz}$$

ดังนั้นความถี่สัญญาณนาฬิกาสูงสุดที่หน่วยประมวลผลกลางยังสามารถทำงานได้คือ 2.5 เมกกะเฮิรท์ซ (2.5 MHz) แต่จากการทดลองใช้งานจริง จุดที่หน่วยประมวลผลกลางยังสามารถทำงานได้เที่ยงตรงไม่มีการผิดพลาดคือที่ 2 เมกกะเฮิรท์ซ ถ้าใช้ความถี่สัญญาณนาฬิกาสูงกว่านี้ที่ใกล้ ๆ 2.5 เมกกะเฮิรท์ซ การทำงานจะเริ่มไม่คงที่คือมีการผิดพลาดในการส่งสัญญาณควบคุมไปยังอุปกรณ์เอาท์พุท ดังนั้นในการออกแบบความถี่สูงสุดของสัญญาณนาฬิกาที่ใช้คือ 2 เมกกะเฮิรท์ซ ซึ่งจะใช้เวลาในการทำคำสั่ง 1 คำสั่งเท่ากับ 500 นาโนเซคกันด์ ฉนั้นจะสามารถขยายส่วนของอินพุทเอาท์พุท และส่วนของหน่วยความจำขึ้นไปได้อีก

7.2 การเพิ่มอินพุทสูงสุด

รหัสของตำแหน่งอินพุทจากหน่วยความจำมีความกว้างขนาด 12 บิท เมื่อแปลเป็นตำแหน่งของอินพุทจะได้ทั้งหมด 4096 ตำแหน่ง โดยแบ่งอินพุทออกเป็น 16 ชุด ใน 1 ชุดของอินพุทจะมี 256 อินพุท โดยอินพุทส่วนหนึ่งใน 256 อินพุทจะแบ่งเป็นตำแหน่ง

ของการรับข้อมูลจากเอาท์พุทกลับเข้ามา และอีกส่วนหนึ่งเป็นตำแหน่งของรีจิสเตอร์ ตัว-ตั้งเวลาและตัวนับ เพราะฉนั้นใน 1 ชุดจะมีอินพุททั้งหมด 120 อินพุท เมื่อขยายระบบสูงสุดทั้งหมด 16 ชุดจะได้จำนวนอินพุททั้งหมดเท่ากับ 1920 อินพุท

7.3 การเพิ่มเอาท์พุทสูงสุด

จากจำนวนเอาท์พุททั้งหมด 256 เอาท์พุท ใน 1 ชุดจะเป็นตำแหน่งของเอาท์พุทที่ใช้งานจริงเพียง 80 เอาท์พุทเท่านั้น และเมื่อขยายระบบสูงสุด 16 ชุด จะได้เอาท์พุททั้งหมดเท่ากับ 1280 เอาท์พุท

7.4 การเพิ่มรีจิสเตอร์สูงสุด

จากจำนวนตำแหน่งอินพุทเอาท์พุททั้งหมด 256 ตำแหน่งใน 1 ชุด จะแบ่งเป็นจำนวนรีจิสเตอร์ 30 ตัว เมื่อขยายระบบสูงสุดจนครบ 16 ชุด จะได้รีจิสเตอร์ทั้งหมด 480 ตัว

7.5 การเพิ่มตัวตั้งเวลาสูงสุด

ใน 1 ชุดจะมีตัวตั้งเวลาทั้งหมด 8 ตัว เมื่อขยายระบบสูงสุดทั้งหมด 16 ชุด จะได้ตัวตั้งเวลาถึง 128 ตัว

7.6 การเพิ่มตัวนับสูงสุด

ใน 1 ชุดจะมีตัวนับทั้งหมด 4 ตัว และเมื่อขยายระบบสูงสุดทั้งหมด 16 ชุดจะได้ตัวนับรวม 64 ตัว

7.7 การเพิ่มตัวนับขึ้นลงสูงสุด

ใน 1 ชุดจะมีตัวนับขึ้นลงทั้งหมด 2 ตัว และเมื่อขยายระบบสูงสุดครบ 16 ชุด จะได้ตัวนับขึ้นลงทั้งหมด 32 ตัว

จากการแบ่งจำนวนของอินพุท เอาท์พุท รีจิสเตอร์ ตัวตั้งเวลา และตัวนับ จะถือความสำคัญและการใช้งานเป็นหลัก ซึ่งเห็นได้ว่าเมื่อเพิ่มระบบสูงสุดครบ 16 ชุด จำนวนของอินพุทจะมีมากที่สุดคือ 1920 อินพุท ทั้งนี้เพราะในการควบคุมแบบซีเควนซ์นั้น จำนวนอินพุทจะมากกว่าเอาท์พุท เช่น ในการสั่งให้เอาท์พุททำงาน อาจจะต้องเอาเงื่อนไขของอินพุทถึง 3 ตัวหรือมากกว่านี้มาเป็นตัวสั่งให้เอาท์พุททำงาน ดังนั้นจำนวนของอินพุทจึงต้องมีมากกว่าเอาท์พุท และในระบบควบคุมแบบซีเควนซ์นี้ การใช้ตัวนับขึ้นลงจะมีที่ใช้น้อยมาก จึงไม่จำเป็นต้องมีจำนวนของตัวนับขึ้นลงมากนัก และตัวตั้งเวลาหรือตัวนับนี้จะถูกใช้งานน้อยกว่าอินพุทและเอาท์พุทมาก เช่น ในบางโปรแกรมอาจจะไม่มีการใช้ตัวตั้งเวลาหรือตัวนับเลยก็เป็นได้ แต่ในโปรแกรมทุก ๆ โปรแกรมจะต้องมีอินพุทและเอาท์พุท เพราะฉนั้นในการออกแบบเครื่องพีแอลซีจึงต้องมีอินพุท และเอาท์พุทมากกว่าตัวตั้งเวลา หรือตัวนับ

7.8 การเพิ่มหน่วยความจำสูงสุด

จากการใช้สัญญาณนาฬิกาความถี่ 2 เมกกะเฮิรท์ซ ในการทำคำสั่ง 1 คำสั่ง จะใช้เวลา 500 นาโนเซคกันด์ ในการทำโปรแกรม 1 ครั้งจะต้องใช้เวลาในการทำสูงสุดไม่เกิน 30 มิลลิเซคกันด์ เพราะฉนั้นจะทำคำสั่งได้ทั้งหมด 60,000 คำสั่งในเวลา 30 มิลลิเซคกันด์ โดยในการทำคำสั่ง 1 คำสั่ง จะใช้พื้นที่หน่วยความจำ 2 ตำแหน่ง (2 byte) ดังนั้นจะต้องใช้หน่วยความจำถึง 120,000 ตำแหน่ง (120 Kilo-byte) โดยเวลาในการทำโปรแกรมขนาด 120,000 ตำแหน่งใช้เวลา 30 มิลลิเซคกันด์ ถ้าโปรแกรมที่เขียนสั้นกว่านี้ เวลาที่ใช้ในการทำโปรแกรมหนึ่งครั้งก็จะน้อยกว่า 30 มิลลิ-เซคกันด์ จะเห็นว่าจำนวนความจำขนาด 120,000 ตำแหน่งนี้สูงพอสำหรับการเขียนโปรแกรมเมื่อขยายจำนวนของอินพุทหรือเอาท์พุทสูงสุดแล้ว เพราะเวลาใช้งานจริง ๆ จำนวนของหน่วยความจำควรมีมากกว่าอินพุทเอาท์พุทประมาณ 10 เท่าขึ้นไป

บทที่ 8
เครื่องป้อนโปรแกรม

ในการทำงานของเครื่องพีแอลซี จะต้องมีโปรแกรมคำสั่งต่าง ๆ ที่จะเป็นตัวบอกให้พีแอลซีรู้ถึงขั้นตอน และเงื่อนไขของการทำงาน โปรแกรมคำสั่งนี้ผู้ใช้เครื่องจะเป็นคนเขียนลงไปเก็บไว้ในหน่วยความจำของพีแอลซี การเขียนโปรแกรมคำสั่งลงไปเก็บไว้ในหน่วยความจำของพีแอลซีนี้สามารถกระทำได้หลายวิธี เช่น เขียนโปรแกรมในรูปของ OP-Code บนเครื่องคอมพิวเตอร์ขนาด 8 บิทแล้วนำ OP-Code ที่เขียนเสร็จแล้วนี้ส่งไปเก็บไว้ใน EPROM โดยผ่านเครื่องโปรแกรม EPROM แล้วเอา EPROM ที่เก็บคำสั่งและเงื่อนไขต่าง ๆ นี้ไปเสียบลงบนเครื่องพีแอลซี แต่วิธีนี้ค่อนข้างยุ่งยากเพราะเวลาเขียนโปรแกรมจะต้องแปรคำสั่งต่าง ๆ ให้อยู่ในรูป OP-Code เสียก่อน และเวลาตรวจสอบโปรแกรมก็ต้องแปลความหมายของ OP-Code ให้เป็นคำสั่งอีกทีหนึ่ง

เครื่องป้อนโปรแกรม (Program Loader) นี้ถูกออกแบบสำหรับการเขียนโปรแกรมคำสั่งต่าง ๆ สำหรับเครื่องพีแอลซีขนาด 1 บิทเพื่อให้สะดวกต่อการเขียนและเข้าใจ โดยเวลาเขียนโปรแกรมคำสั่งจะเขียนโดยตรงเลย เช่น คำสั่ง LD ก็สามารถกดคีย์คำสั่ง LD ได้ตรง ๆ เลย เครื่องป้อนโปรแกรมนี้จะแปลคีย์คำสั่งเป็นรูปของ OP-Code และส่งไปเก็บไว้ในหน่วยความจำ ทำให้ผู้เขียนโปรแกรมไม่ต้องยุ่งยากในการแปลคำสั่งให้เป็น OP-Code และเวลาตรวจสอบโปรแกรม เครื่องป้อนโปรแกรมจะแปล OP-Code ในหน่วยความจำเป็นคำสั่งต่าง ๆ และส่งผลมาแสดงให้ผู้ใช้ทราบในรูปของคำสั่งตรง ๆ เลย เครื่องป้อนโปรแกรมนี้มีคีย์ใช้งานทั้งหมด 32 คีย์ ดังในรูปที่ 8.1

LD	LDC	IEN	OEN	9	△	▽	SERC
AND	ANDC	STO	STOC	6	7	8	ADD
OR	ORC	SET	END	3	4	5	CLR
XNOR	SKZ	INS	DEL	0	1	2	ENTR

รูปที่ 8.1 ตำแหน่งคำสั่งต่าง ๆ ของคีย์

จากคีย์ทั้งหมด 32 คีย์ จะแบ่งตามหน้าที่ของคีย์ได้ 3 ส่วน ด้วยกันคือ

1. คีย์คำสั่ง (Instruction Key) มีทั้งหมด 14 คีย์ คือ คีย์คำสั่ง LD, LDC, AND, ANDC, OR, ORC, XNOR, SKZ, IEN, OEN, STO, STOC, SET และ END

2. คีย์ตัวเลขมีทั้งหมด 10 คีย์ คือจากเลข 0 ถึงเลข 9

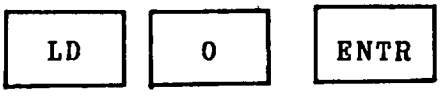
3. คีย์การทำงานพิเศษ (Editor Key) มีทั้งหมด 8 คีย์ คือ INS, DEL, Δ, ∇, SERCH, ADD, CLR และ ENTR

8.1 วิธีการใช้เครื่องป้อนโปรแกรม

เมื่อเริ่มต้นเขียนโปรแกรม ก่อนอื่นจะต้องลบข้อมูลที่อยู่ในหน่วยความจำทั้งหมดให้เป็น 0 ก่อนจึงจะเริ่มต้นเขียนโปรแกรมได้ เพราะว่าถ้าไม่ลบข้อมูลเก่าให้หมดไปนั้นในหน่วยความจำอาจจะมีค่าอื่น ๆ อยู่ และจะทำให้การทำงานของเครื่องพีแอลซีผิดพลาดได้ การลบข้อมูลเก่าในหน่วยความจำให้หมดไปจะทำตามขั้นตอนได้ดังนี้

1. โยกสวิทช์ที่เครื่องพีแอลซีให้อยู่ในตำแหน่ง Program
2. กดคีย์ต่าง ๆ ดังนี้

เมื่อทำขั้นตอนที่ 1 และ 2 เสร็จแล้วจึงเริ่มใส่โปรแกรมเงื่อนไขต่าง ๆ ลงไปในหน่วยความจำ ดังตัวอย่างการใส่คำสั่ง LD อินพุท 000 จะกดคีย์ดังนี้



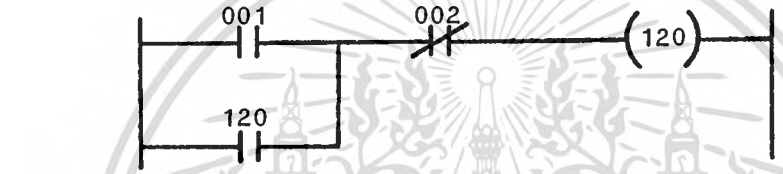
ในคำสั่งแต่ละคำสั่ง เมื่อกดคีย์ต่าง ๆ เรียบร้อยแล้ว ต้องกดคีย์ ENTR เสมอ เพื่อบอกให้เครื่องป้อนโปรแกรมนำคำสั่งนั้น ๆ เข้าไปเก็บไว้ในหน่วยความจำ และเครื่อง

จะเลื่อนไปยังตำแหน่งของหน่วยความจำถัดไปเพื่อรอรับคำสั่งใหม่

เมื่อป้อนโปรแกรมเงื่อนไขของการทำงานต่าง ๆ หมดแล้ว ต้องจบโปรแกรมด้วยคำสั่ง END เสมอ เพื่อบอกให้เครื่องพีแอลซีรู้ว่าโปรแกรมสิ้นสุดแล้ว และเครื่องพีแอลซีจะกลับไปเริ่มต้นทำโปรแกรมคำสั่งแรกสุดใหม่อีก

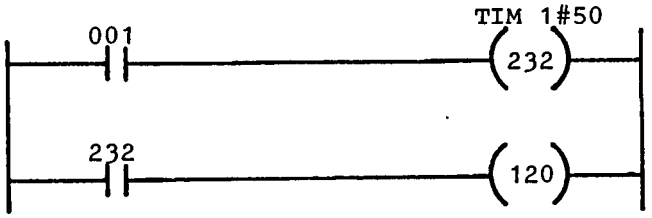
8.2 ตัวอย่างการกดคีย์คำสั่งต่าง ๆ ตามแลดเดอร์ไดอะแกรม

1. การใช้คำสั่ง LD, OR, ANDC และ STO



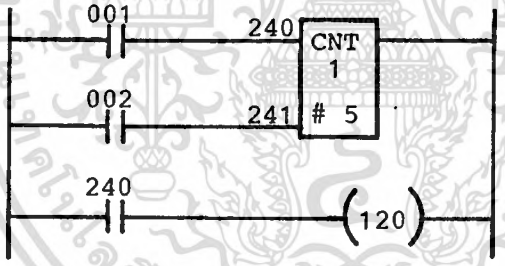
LD	1	ENTR		
OR	1	2	0	ENTR
ANDC	2	ENTR		
STO	1	2	0	ENTR

2. การกดคีย์สำหรับคำสั่งการตั้งเวลา

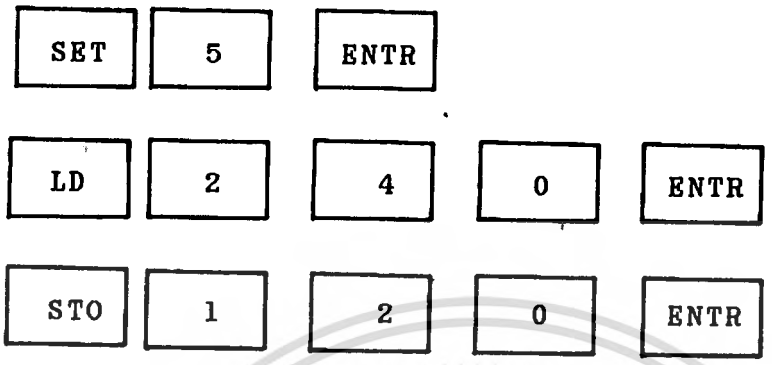


LD	1	ENTR		
STO	2	3	2	ENTR
SET	5	0	ENTR	
LD	2	3	2	ENTR
STO	1	2	0	ENTR

3. การกดคีย์สำหรับคำสั่งการนับ



LD	1	ENTR		
STO	2	4	0	ENTR
LD	2	ENTR		
STO	2	4	1	ENTR



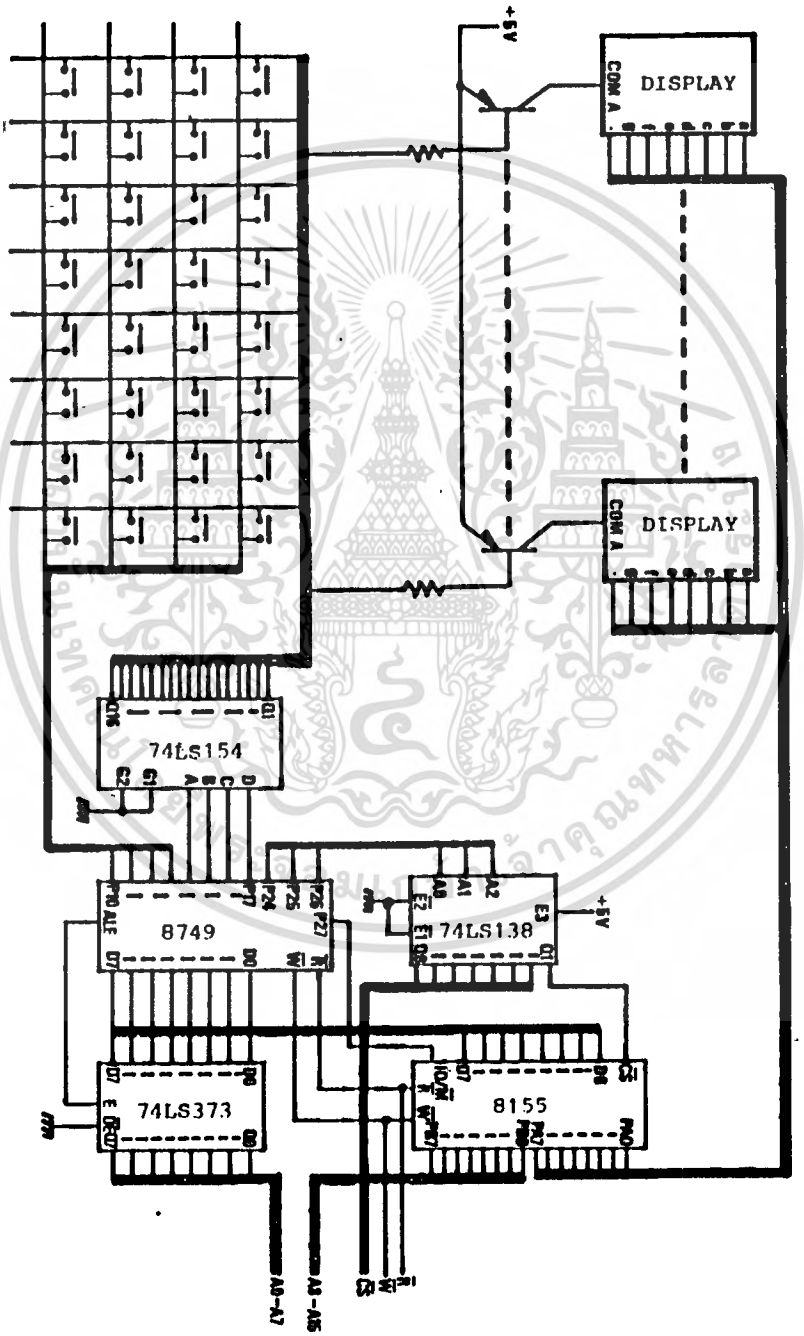
8.3 วงจรของเครื่องป้อนโปรแกรม

เครื่องป้อนโปรแกรมเป็นเสมือนตัวกลางระหว่างผู้เขียนโปรแกรม กับเครื่องพี-แอลซี โดยจะรับคำสั่งจากการกดคีย์และทำการแปลคำสั่งส่งให้กับเครื่องพีแอลซี โดยมีการทำงานเหมือนกับการเปิดตาราง (Look up table) คือเมื่อได้รับสัญญาณการกดคีย์จะนำสัญญาณของคีย์ที่ถูกกดมาเปิดในตารางดูว่าเป็นรหัสอะไร แล้วนำรหัสนั้น ๆ ส่งไปเก็บในหน่วยความจำของเครื่องพีแอลซี หรือถ้าเป็นการตรวจสอบโปรแกรม ก็จะรับรหัสจากหน่วยความจำของเครื่องพีแอลซีมาทำการเปิดตารางดูว่าเป็นคำสั่งอะไรแล้วนำผลในตารางส่งไปแสดงให้ทราบว่าเป็นคำสั่งอะไร

ตัวจัดการในการเปิดตารางและแปลรหัสนี้คือ หน่วยประมวลผลกลางขนาด 8 บิท โดยการออกแบบจะใช้ไอซีชิพเดี่ยว (IC Single Chip) เบอร์ 8749 ซึ่งมีข้อดีตรงที่ตัวมันเองสามารถทำงานได้โดยใช้อุปกรณ์ภายนอกน้อยมาก ทั้งนี้เพราะภายในตัวมันจะมี ROM ขนาด 2 กิโลไบท์ และพอดท์ (Port) ขนาด 8 บิทอยู่ถึง 3 พอดท์ และในขณะทำงานส่งข้อมูลออกไปยังพอดท์ต่าง ๆ หน่วยประมวลผลกลางยังสามารถที่จะอ่านข้อมูลจากพอดท์ต่าง ๆ นั้กลับเข้ามาได้ด้วยซึ่งเป็นข้อดีของไอซีเบอร์นี้ แต่ในการออกแบบเครื่องพีแอลซีไม่ได้ใช้หน่วยประมวลผลกลางขนาด 8 บิทนี้เพราะว่ามีข้อเสียตรงที่การทำคำสั่งแต่ละคำสั่งจะใช้เวลานาน เมื่อนำไปใช้เป็นหน่วยประมวลผลกลางของเครื่องพีแอลซี จะทำให้จำ-นวนอินพุทเอาท์พุท และหน่วยความจำมีน้อย ไม่สามารถขยายให้สูงได้ เพราะเวลาในการทำคำสั่งทั้งหมดในโปรแกรมไม่ควรจะมากกว่า 30 มิลิเซคกันด์ ถ้าใช้เวลาในการทำ

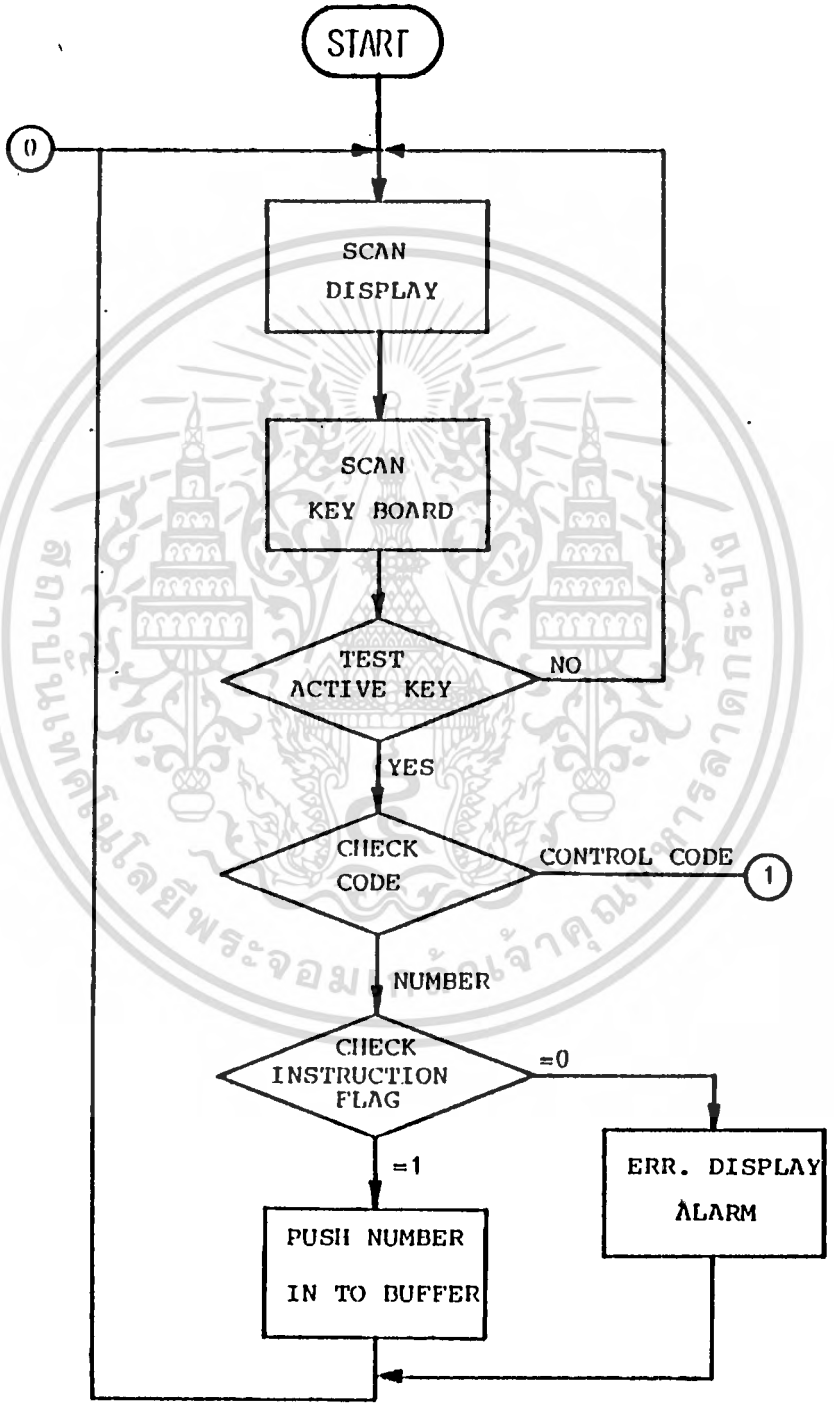
คำสั่งทั้งหมดมากกว่านี้ความผิดพลาดจะมีมาก

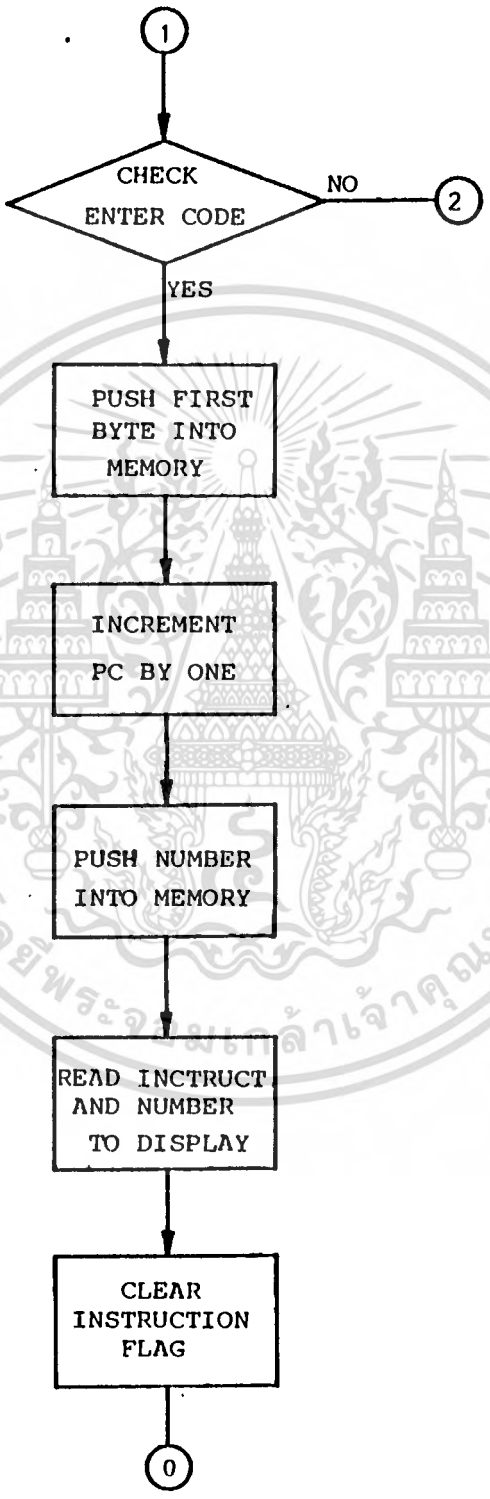
วงจรของเครื่องป้อนโปรแกรมนี้ แสดงได้ในรูปที่ 8.2 และมีการทำงานตามไฟลชาร์ท (Flow Chart) และโปรแกรมจัดการ (Monitor Program)

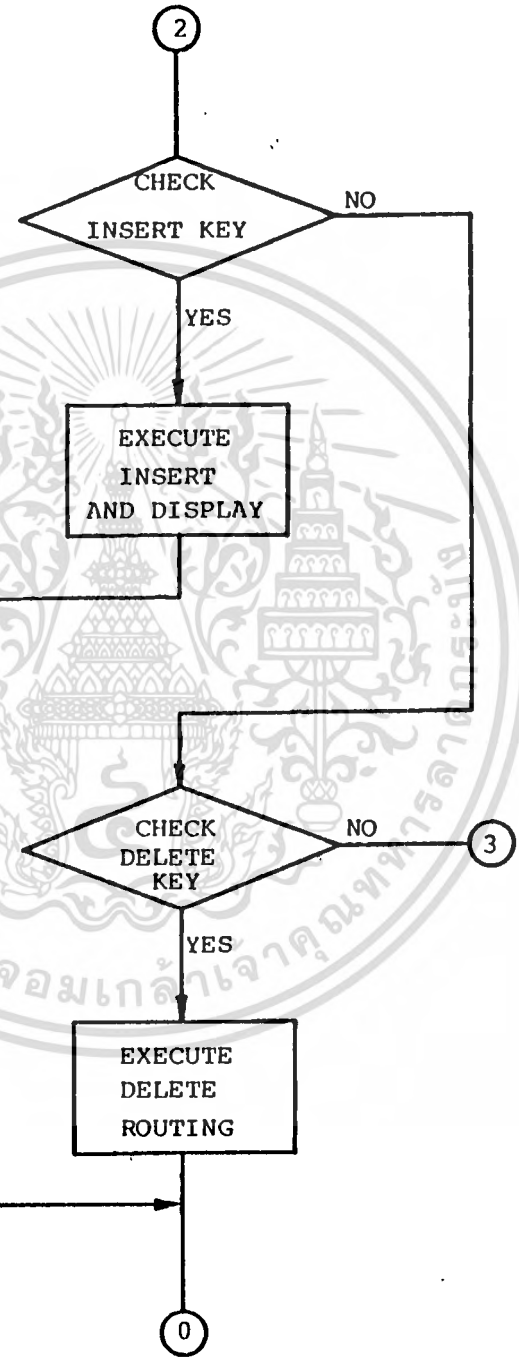


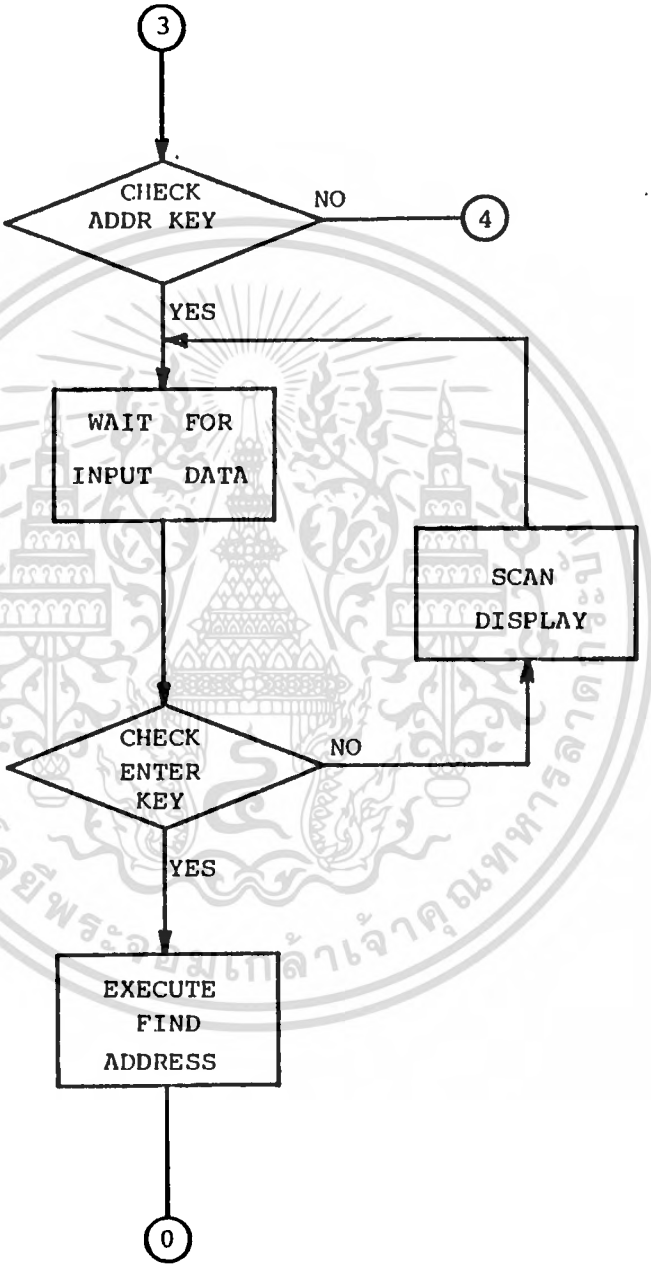
รูปที่ 8.2 วงจรของเครื่องป้อนโปรแกรม

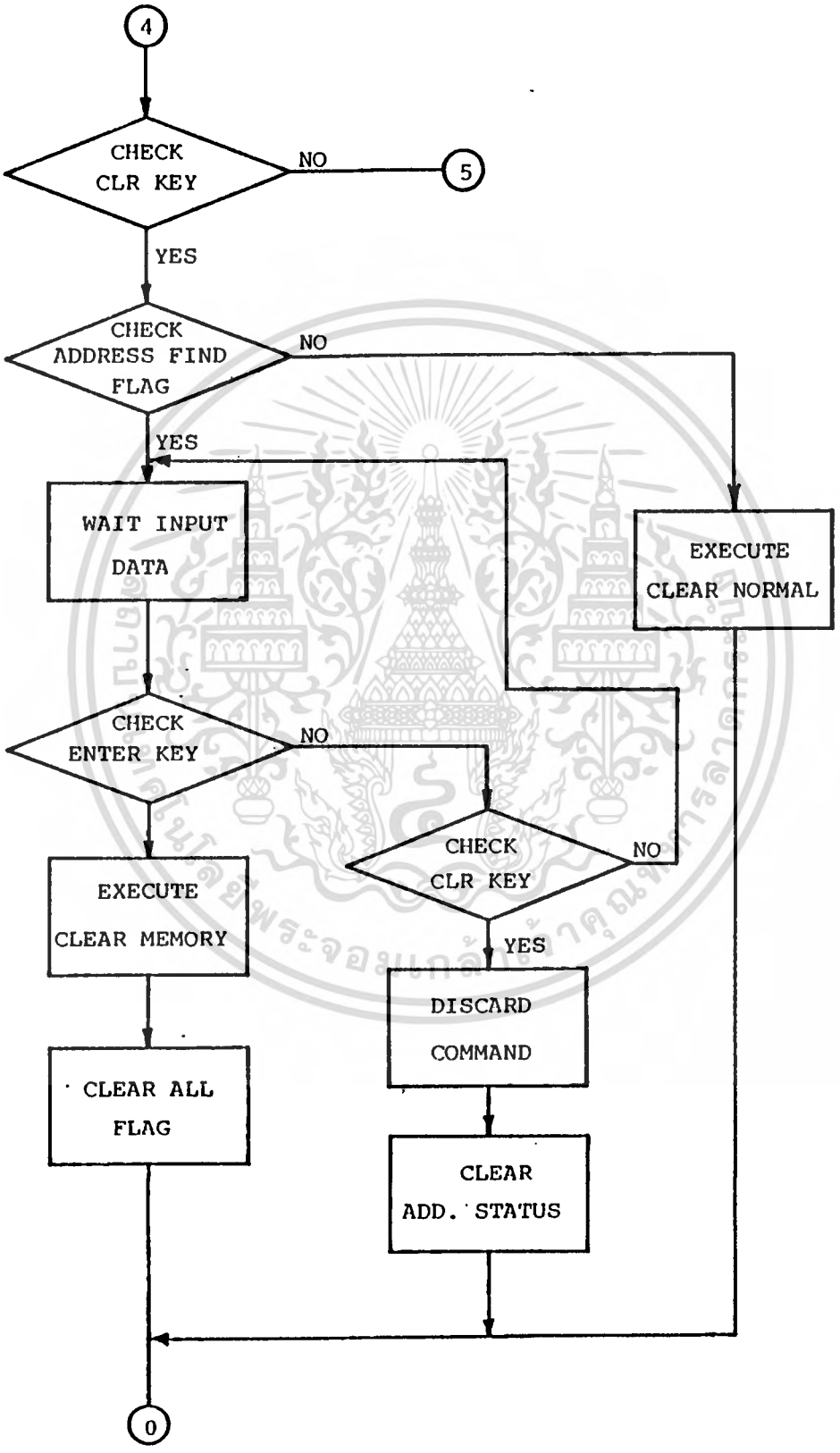
8.4 โฟลชาร์ทแสดงการทำงานของเครื่องป้อนโปรแกรม

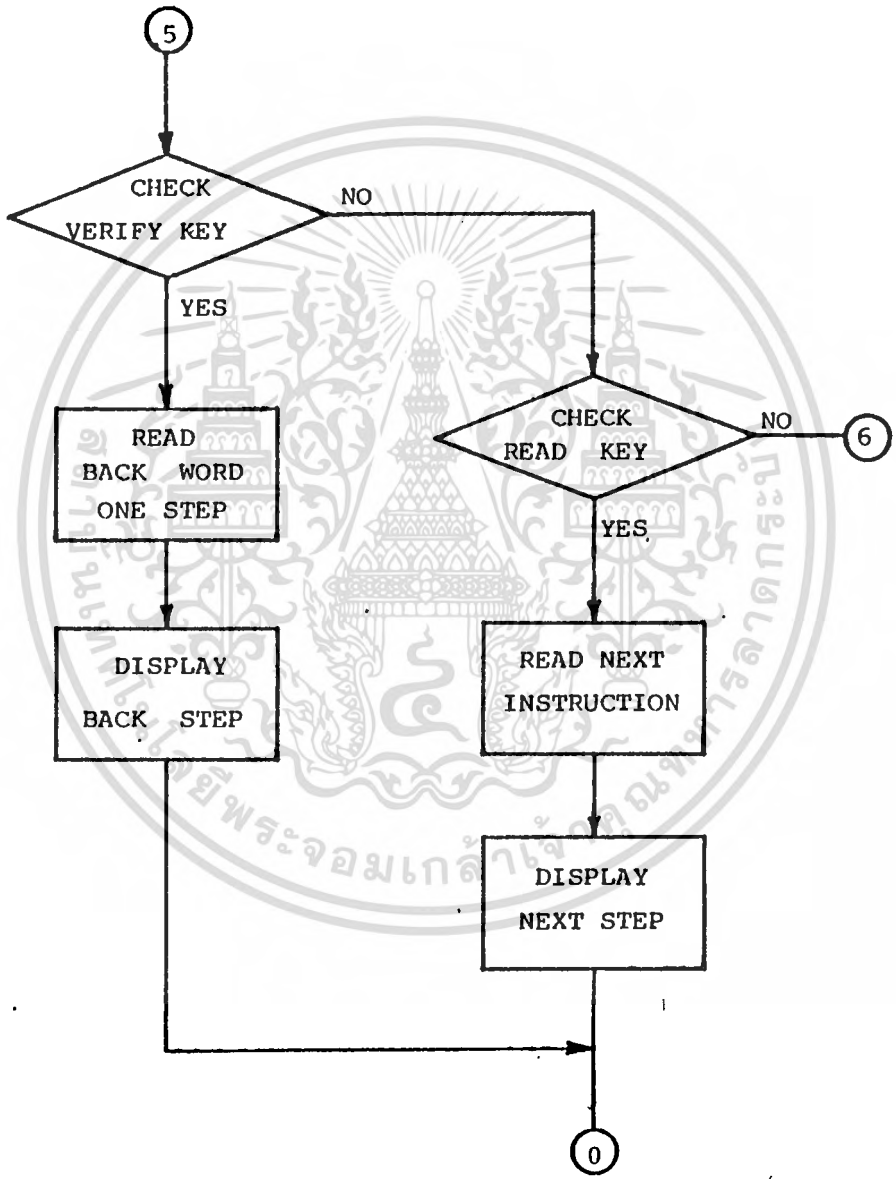


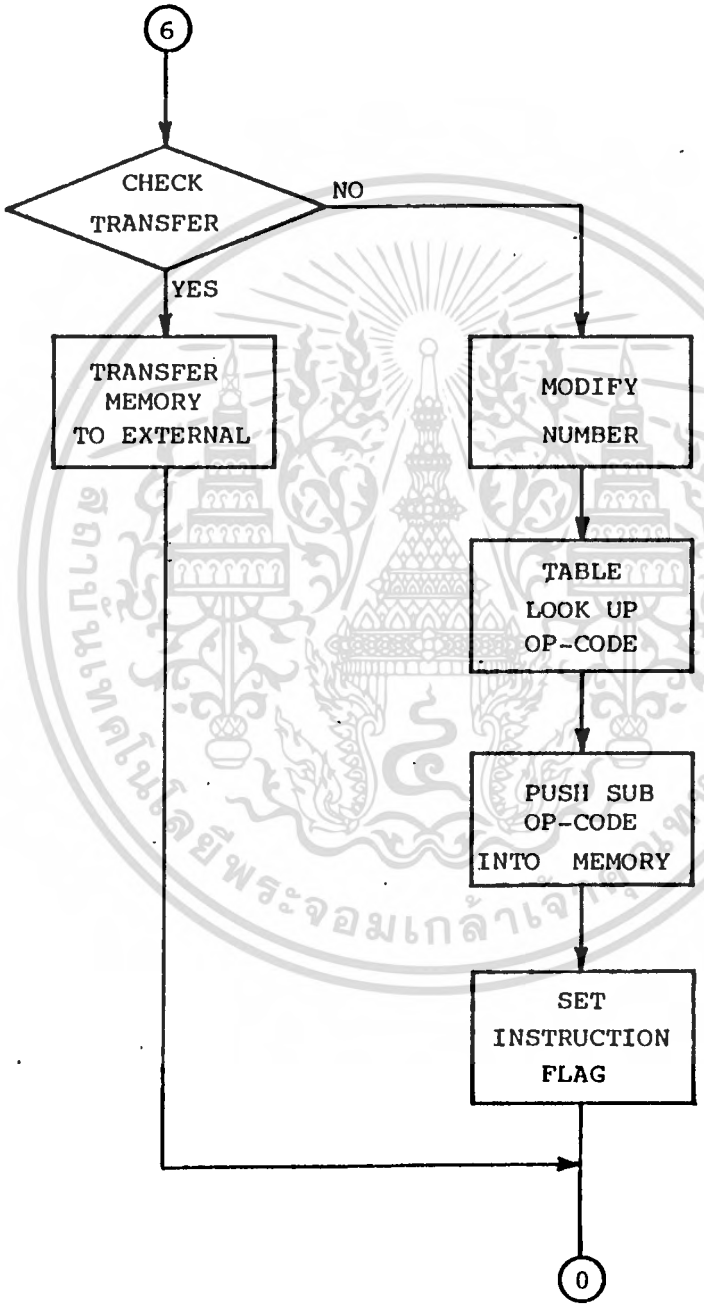












8.5 โปรแกรมจัดการของเครื่องป้อนโปรแกรม

```
                    0000 ************
                    0001 *            *
                    0002 * SET PORT   *
                    0003 *            *
                    0004 ************
0800- 23 10         0005          MOV A,#$10
0802- 3A            0006          OUTL P2,A
0803- B8 00         0007          MOV R0,#00
0805- 23 0F         0008          MOV A,#$0F
0807- 90            0009          MOVX §R0,A
                    0010 *****************
                    0011 *               *
                    0012 * SET REGISTER  *
                    0013 *               *
                    0014 *****************
0808- 34 EC         0015          CALL RDIS1
080A- 34 F7         0016          CALL RDIS2
080C- 54 09         0017          CALL RDIS3
                    0018 ************************
                    0019 *                      *
                    0020 * KEY SCAN & DISPLAY   *
                    0021 *                      *
                    0022 ************************
080E- BF F9         0023 SCAN1    MOV R7,#$F9
0810- B9 3F         0024          MOV R1,#63
0812- CF            0025 SCAN2    DEC R7
0813- F1            0026          MOV A,§R1
0814- B8 01         0027          MOV R0,#01
0816- 90            0028          MOVX §R0,A
0817- C9            0029          DEC R1
0818- FF            0030          MOV A,R7
0819- 47            0031          SWAP A
081A- 39            0032          OUTL P1,A
081B- 09            0033          IN A,P1
081C- AB            0034          MOV R3,A
081D- 53 0F         0035          ANL A,#$0F
081F- 43 F0         0036          ORL A,#$F0
0821- 37            0037          CPL A
0822- 2E            0038          XCH A,R6
0823- FF            0039          MOV A,R7
0824- 53 0F         0040          ANL A,#$0F
0826- 4E            0041          ORL A,R6
0827- C6 0E         0042          JZ SCAN1
0829- FE            0043          MOV A,R6
082A- C6 12         0044          JZ SCAN2
082C- 14 E6         0045          CALL SCAN05
                    0046 ******************
                    0047 *                *
                    0048 * CHECK KEY INS. *
```

```
                        0049 *                    *
                        0050 ********************
082E- 23 F8             0051         MOV A,#$F8
0830- 5B                0052         ANL A,R3
0831- C6 76             0053         JZ LD
0833- 23 F4             0054         MOV A,#$F4
0835- 5B                0055         ANL A,R3
0836- C6 86             0056         JZ AND
0838- 23 F2             0057         MOV A,#$F2
083A- 5B                0058         ANL A,R3
083B- C6 96             0059         JZ OR
083D- 23 F1             0060         MOV A,#$F1
083F- 5B                0061         ANL A,R3
0840- C6 A6             0062         JZ XNOR
0842- 23 E8             0063         MOV A,#$E8
0844- 5B                0064         ANL A,R3
0845- C6 7E             0065         JZ LDC
0847- 23 E4             0066         MOV A,#$E4
0849- 5B                0067         ANL A,R3
084A- C6 8E             0068         JZ ANDC
084C- 23 E2             0069         MOV A,#$E2
084E- 5B                0070         ANL A,R3
084F- C6 9E             0071         JZ ORC
0851- 23 E1             0072         MOV A,#$E1
0853- 5B                0073         ANL A,R3
0854- C6 AE             0074         JZ SKZ
0856- 23 D8             0075         MOV A,#$D8
0858- 5B                0076         ANL A,R3
0859- C6 B6             0077         JZ IEN
085B- 23 D4             0078         MOV A,#$D4
085D- 5B                0079         ANL A,R3
085E- C6 C6             0080         JZ STO
0860- 23 D2             0081         MOV A,#$D2
0862- 5B                0082         ANL A,R3
0863- C6 D6             0083         JZ TCSET
0865- 23 C8             0084         MOV A,#$C8
0867- 5B                0085         ANL A,R3
0868- C6 BE             0086         JZ OEN
086A- 23 C4             0087         MOV A,#$C4
086C- 5B                0088         ANL A,R3
086D- C6 CE             0089         JZ STC
086F- 23 C2             0090         MOV A,#$C2
0871- 5B                0091         ANL A,R3
0872- C6 DE             0092         JZ END
0874- 04 FC             0093         JMP KEY1
                        0094 ********************
                        0095 *                    *
                        0096 * INSTRUCTION SET  *
                        0097 *                    *
                        0098 ********************
0876- 23 10             0099 LD      MOV A,#$10
0878- 54 14             0100         CALL TND
```

```
087A- 23 20    0101          MOV A,#$20
087C- 44 24    0102          JMP TITM
087E- 23 10    0103 LDC      MOV A,#$10
0880- 54 1C    0104          CALL TCD
0882- 23 21    0105          MOV A,#$21
0884- 44 24    0106          JMP TITM
0886- 23 11    0107 AND      MOV A,#$11
0888- 54 14    0108          CALL TND
088A- 23 22    0109          MOV A,#$22
088C- 44 24    0110          JMP TITM
088E- 23 11    0111 ANDC     MOV A,#$11
0890- 54 1C    0112          CALL TCD
0892- 23 23    0113          MOV A,#$23
0894- 44 24    0114          JMP TITM
0896- 23 12    0115 OR       MOV A,#$12
0898- 54 14    0116          CALL TND
089A- 23 24    0117          MOV A,#$24
089C- 44 24    0118          JMP TITM
089E- 23 12    0119 ORC      MOV A,#$12
08A0- 54 1C    0120          CALL TCD
08A2- 23 25    0121          MOV A,#$25
08A4- 44 24    0122          JMP TITM
08A6- 23 13    0123 XNOR     MOV A,#$13
08A8- 54 14    0124          CALL TND
08AA- 23 26    0125          MOV A,#$26
08AC- 44 24    0126          JMP TITM
08AE- 23 13    0127 SKZ      MOV A,#$13
08B0- 54 1C    0128          CALL TCD
08B2- 23 2D    0129          MOV A,#$2D
08B4- 44 24    0130          JMP TITM
08B6- 23 14    0131 IEN      MOV A,#$14
08B8- 54 14    0132          CALL TND
08BA- 23 29    0133          MOV A,#$29
08BC- 44 24    0134          JMP TITM
08BE- 23 14    0135 OEN      MOV A,#$14
08C0- 54 1C    0136          CALL TCD
08C2- 23 2A    0137          MOV A,#$2A
08C4- 44 24    0138          JMP TITM
08C6- 23 15    0139 STO      MOV A,#$15
08C8- 54 14    0140          CALL TND
08CA- 23 27    0141          MOV A,#$27
08CC- 44 24    0142          JMP TITM
08CE- 23 15    0143 STC      MOV A,#$15
08D0- 54 1C    0144          CALL TCD
08D2- 23 28    0145          MOV A,#$28
08D4- 44 24    0146          JMP TITM
08D6- 23 16    0147 TCSET    MOV A,#$16
08D8- 54 14    0148          CALL TND
08DA- 23 2B    0149          MOV A,#$2B
08DC- 44 24    0150          JMP TITM
08DE- 23 16    0151 END      MOV A,#$16
```

```
08E0- 54 1C     0152          CALL TCD
08E2- 23 2E     0153          MOV A,#$2E
08E4- 44 24     0154          JMP TITM
                0155 ***************
                0156 *             *
                0157 * BUAND LIMIT *
                0158 *             *
                0159 ***************
08E6- BF 25     0160 SCAN05 MOV R7,#$25
08E8- CF        0161 SCAN06 DEC R7
08E9- B9 FF     0162          MOV R1,#$FF
08EB- C9        0163 SCAN07 DEC R1
08EC- F9        0164          MOV A,R1
08ED- 96 EB     0165          JNZ SCAN07
08EF- FF        0166          MOV A,R7
08F0- 96 E8     0167          JNZ SCAN06
08F2- 83        0168          RET
08F3- 54 DE     0169 DOWN1  CALL DECPC
08F5- 54 68     0170          CALL SCALL
08F7- 84 CF     0171          JMP SCAN0
08F9- 00        0172          NOP
08FA- 00        0173          NOP
08FB- 00        0174          NOP
                0175 ****************
                0176 *              *
                0177 * CHECK KEY NUM *
                0178 *              *
                0179 ****************
08FC- 23 B8     0180 KEY1     MOV A,#$B8
08FE- 5B        0181          ANL A,R3
08FF- C6 78     0182          JZ N9
0901- 23 B4     0183          MOV A,#$B4
0903- 5B        0184          ANL A,R3
0904- C6 60     0185          JZ N6
0906- 23 B2     0186          MOV A,#$B2
0908- 5B        0187          ANL A,R3
0909- C6 48     0188          JZ N3
090B- 23 B1     0189          MOV A,#$B1
090D- 5B        0190          ANL A,R3
090E- C6 30     0191          JZ N0
0910- 23 A4     0192          MOV A,#$A4
0912- 5B        0193          ANL A,R3
0913- C6 68     0194          JZ N7
0915- 23 A2     0195          MOV A,#$A2
0917- 5B        0196          ANL A,R3
0918- C6 50     0197          JZ N4
091A- 23 A1     0198          MOV A,#$A1
091C- 5B        0199          ANL A,R3
091D- C6 38     0200          JZ N1
091F- 23 94     0201          MOV A,#$94
0921- 5B        0202          ANL A,R3
0922- C6 70     0203          JZ N8
```

```
0924- 23 92     0204          MOV A,#$92
0926- 5B        0205          ANL A,R3
0927- C6 58     0206          JZ N5
0929- 23 6E     0207          MOV A,#$6E
092B- DB        0208          XRL A,R3
092C- C6 40     0209          JZ N2
092E- 24 80     0210          JMP KEY2
                0211 ***************
                0212 *             *
                0213 * NUMBER SET  *
                0214 *             *
                0215 ***************
0930- 23 00     0216 N0       MOV A,#$00
0932- 54 2A     0217          CALL NDIS
0934- 23 30     0218          MOV A,#$30
0936- 44 39     0219          JMP NDU
0938- 23 01     0220 N1       MOV A,#$01
093A- 54 2A     0221          CALL NDIS
093C- 23 31     0222          MOV A,#$31
093E- 44 39     0223          JMP NDU
0940- 23 02     0224 N2       MOV A,#$02
0942- 54 2A     0225          CALL NDIS
0944- 23 32     0226          MOV A,#$32
0946- 44 39     0227          JMP NDU
0948- 23 03     0228 N3       MOV A,#$03
094A- 54 2A     0229          CALL NDIS
094C- 23 33     0230          MOV A,#$33
094E- 44 39     0231          JMP NDU
0950- 23 04     0232 N4       MOV A,#$04
0952- 54 2A     0233          CALL NDIS
0954- 23 34     0234          MOV A,#$34
0956- 44 39     0235          JMP NDU
0958- 23 05     0236 N5       MOV A,#$05
095A- 54 2A     0237          CALL NDIS
095C- 23 35     0238          MOV A,#$35
095E- 44 39     0239          JMP NDU
0960- 23 06     0240 N6       MOV A,#$06
0962- 54 2A     0241          CALL NDIS
0964- 23 36     0242          MOV A,#$36
0966- 44 39     0243          JMP NDU
0968- 23 07     0244 N7       MOV A,#$07
096A- 54 2A     0245          CALL NDIS
096C- 23 37     0246          MOV A,#$37
096E- 44 39     0247          JMP NDU
0970- 23 08     0248 N8       MOV A,#$08
0972- 54 2A     0249          CALL NDIS
0974- 23 38     0250          MOV A,#$38
0976- 44 39     0251          JMP NDU
0978- 23 09     0252 N9       MOV A,#$09
097A- 54 2A     0253          CALL NDIS
097C- 23 39     0254          MOV A,#$39
```

```
097E- 44 39     0255          JMP NDU
                0256 ******************
                0257 *                *
                0258 * CHECK KEY FUN *
                0259 *                *
                0260 ******************
0980- 23 D1     0261 KEY2     MOV A,#$D1
0982- 5B        0262          ANL A,R3
0983- C6 B6     0263          JZ INS
0985- 23 C1     0264          MOV A,#$C1
0987- 5B        0265          ANL A,R3
0988- C6 B4     0266          JZ DEL
098A- 23 A8     0267          MOV A,#$AB
098C- 5B        0268          ANL A,R3
098D- C6 B2     0269          JZ DOWN
098F- 23 98     0270          MOV A,#$98
0991- 5B        0271          ANL A,R3
0992- C6 B0     0272          JZ UP
0994- 23 8B     0273          MOV A,#$8B
0996- 5B        0274          ANL A,R3
0997- C6 AE     0275          JZ SERCH
0999- 23 84     0276          MOV A,#$84
099B- 5B        0277          ANL A,R3
099C- C6 AC     0278          JZ ADD
099E- 23 82     0279          MOV A,#$82
09A0- 5B        0280          ANL A,R3
09A1- C6 AA     0281          JZ CLR
09A3- 23 81     0282          MOV A,#$81
09A5- 5B        0283          ANL A,R3
09A6- C6 A8     0284          JZ ENT
09A8- 44 5A     0285 ENT      JMP ENTER
09AA- A4 1F     0286 CLR      JMP CLR100
09AC- A4 09     0287 ADD      JMP ADDDIS
09AE- A4 5F     0288 SERCH    JMP SERCH1
09B0- 44 EC     0289 UP       JMP UP1
09B2- 04 F3     0290 DOWN     JMP DOWN1
09B4- A4 ED     0291 DEL      JMP DET100
09B6- A4 A3     0292 INS      JMP INS100
                0293 ******************
                0294 *                *
                0295 * CHECK KEY OFF *
                0296 *                *
                0297 ******************
09B8- BF F9     0298 SCAN69   MOV R7,#$F9
09BA- B9 3F     0299          MOV R1,#63
09BC- CF        0300 SCAN01   DEC R7
09BD- F1        0301          MOV A,@R1
09BE- B8 01     0302          MOV R0,#01
09C0- 90        0303          MOVX @R0,A
09C1- C9        0304          DEC R1
09C2- FF        0305          MOV A,R7
09C3- 47        0306          SWAP A
```

```
09C4- 39       0307          OUTL P1,A
09C5- 09       0308          IN A,P1
09C6- 53 0F    0309          ANL A,#$0F
09C8- 43 F0    0310          ORL A,#$F0
09CA- 37       0311          CPL A
09CB- 96 D4    0312          JNZ SCAN02
09CD- FF       0313          MOV A,R7
09CE- 53 0F    0314          ANL A,#$0F
09D0- C6 E8    0315          JZ SCAN03
09D2- 96 BC    0316          JNZ SCAN01
09D4- BF F9    0317 SCAN02   MOV R7,#$F9
09D6- B9 3F    0318          MOV R1,#63
09D8- CF       0319 SCAN04   DEC R7
09D9- F1       0320          MOV A,@R1
09DA- B8 01    0321          MOV R0,#01
09DC- 90       0322          MOVX @R0,A
09DD- C9       0323          DEC R1
09DE- FF       0324          MOV A,R7
09DF- 47       0325          SWAP A
09E0- 39       0326          OUTL P1,A
09E1- FF       0327          MOV A,R7
09E2- 53 0F    0328          ANL A,#$0F
09E4- C6 B8    0329          JZ SCAN69
09E6- 96 D8    0330          JNZ SCAN04
09E8- 14 E6    0331 SCAN03   CALL SCAN05
09EA- 04 0E    0332          JMP SCAN1
               0333 ********************
               0334 *                  *
               0335 * SUB RDIS 1,2,3   *
               0336 *                  *
               0337 ********************
09EC- B8 18    0338 RDIS1    MOV R0,#24
09EE- 23 20    0339          MOV A,#32
09F0- B0 00    0340 DIS1     MOV @R0,#00
09F2- 07       0341          DEC A
09F3- 18       0342          INC R0
09F4- 96 F0    0343          JNZ DIS1
09F6- 83       0344          RET
               0345 ********************
09F7- B8 38    0346 RDIS2    MOV R0,#56
09F9- B0 00    0347          MOV @R0,#00
09FB- 18       0348          INC R0
09FC- B0 00    0349          MOV @R0,#00
09FE- B8 3D    0350          MOV R0,#61
0A00- 23 03    0351          MOV A,#3
0A02- B0 FC    0352 DIS2     MOV @R0,#$FC
0A04- 07       0353          DEC A
0A05- 18       0354          INC R0
0A06- 96 02    0355          JNZ DIS2
0A08- 83       0356          RET
               0357 ********************
0A09- B8 3A    0358 RDIS3    MOV R0,#58
```

```
0A0B- 23 03     0359          MOV A,#3
0A0D- B0 FC     0360 DIS3     MOV @R0,#$FC
0A0F- 07        0361          DEC A
0A10- 18        0362          INC R0
0A11- 96 0D     0363          JNZ DIS3
0A13- 83        0364          RET
                0365 **********************
                0366 *                      *
                0367 * SUBRUTINE TND,TCD,*
                0368 * TITM,NDIS,NDU.     *
                0369 **********************
0A14- E3        0370 TND      MOVP3 A,@A
0A15- B8 38     0371          MOV R0,#56
0A17- A0        0372          MOV @R0,A
0A18- 18        0373          INC R0
0A19- B0 00     0374          MOV @R0,#00
0A1B- 83        0375          RET
0A1C- E3        0376 TCD      MOVP3 A,@A
0A1D- B8 39     0377          MOV R0,#57
0A1F- A0        0378          MOV @R0,A
0A20- C8        0379          DEC R0
0A21- B0 00     0380          MOV @R0,#00
0A23- 83        0381          RET
0A24- E3        0382 TITM     MOVP3 A,@A
0A25- B8 34     0383          MOV R0,#52
0A27- A0        0384          MOV @R0,A
0A28- 84 CF     0385          JMP SCAN0
0A2A- E3        0386 NDIS     MOVP3 A,@A
0A2B- AB        0387          MOV R3,A
0A2C- B8 3E     0388          MOV R0,#62
0A2E- B9 3D     0389          MOV R1,#61
0A30- F0        0390          MOV A,@R0
0A31- A1        0391          MOV @R1,A
0A32- B9 3F     0392          MOV R1,#63
0A34- F1        0393          MOV A,@R1
0A35- A0        0394          MOV @R0,A
0A36- FB        0395          MOV A,R3
0A37- A1        0396          MOV @R1,A
0A38- 83        0397          RET
0A39- E3        0398 NDU      MOVP3 A,@A
0A3A- AB        0399          MOV R3,A
0A3B- B8 36     0400          MOV R0,#54
0A3D- B9 35     0401          MOV R1,#53
0A3F- F0        0402          MOV A,@R0
0A40- A1        0403          MOV @R1,A
0A41- B9 37     0404          MOV R1,#55
0A43- F1        0405          MOV A,@R1
0A44- A0        0406          MOV @R0,A
0A45- FB        0407          MOV A,R3
0A46- A1        0408          MOV @R1,A
0A47- 84 CF     0409          JMP SCAN0
```

```
                        0410 ******************
                        0411 *                *
                        0412 * SUB.ADD 2 BYTE *
                        0413 *                *
                        0414 ******************
                        0415
0A49- FE                0416 ADD3   MOV A,R6
0A4A- CD                0417        DEC R5
0A4B- 69                0418        ADD A,R1
0A4C- AE                0419        MOV R6,A
0A4D- F6 53             0420        JC ADD1
0A4F- FD                0421 ADD2   MOV A,R5
0A50- 96 49             0422        JNZ ADD3
0A52- 83                0423        RET
                        0424 ********************
0A53- 23 00             0425 ADD1   MOV A,#00
0A55- 7F                0426        ADDC A,R7
0A56- AF                0427        MOV R7,A
0A57- 97                0428        CLR C
0A58- 44 4F             0429        JMP ADD2
                        0430 ***************
                        0431 *             *
                        0432 * MAIN ENTER *
                        0433 *             *
                        0434 ***************
0A5A- 54 B1             0435 ENTER  CALL ENTR
0A5C- B4 93             0436        CALL TRAND
0A5E- 54 81             0437        CALL MULTI
0A60- 74 90             0438        CALL WRITE
0A62- 54 D1             0439        CALL INCPC
0A64- 54 6B             0440        CALL SCALL
0A66- 84 D9             0441        JMP SCAND
0A68- 54 81             0442 SCALL  CALL MULTI
0A6A- 74 A9             0443        CALL READ
0A6C- 54 9F             0444        CALL MOVPC
0A6E- 74 CD             0445        CALL HTOD
0A70- B8 2A             0446        MOV R0,#42
0A72- 54 8F             0447        CALL TRAN
0A74- 54 A8             0448        CALL MOVIO
0A76- 74 CD             0449        CALL HTOD
0A78- B8 35             0450        MOV R0,#53
0A7A- 54 8F             0451        CALL TRAN
0A7C- 94 47             0452        CALL DTOCO
0A7E- 94 79             0453        CALL DIS60
0A80- 83                0454        RET
                        0455 ******************
                        0456 *                *
                        0457 * SUB MULTIPLY 2 *
                        0458 *                *
                        0459 ******************
0A81- 54 9F             0460 MULTI  CALL MOVPC
```

```
0A83- B8 2D     0461          MOV R0,#45
0A85- B9 2E     0462          MOV R1,#46
0A87- FD        0463          MOV A,R5
0A88- 97        0464          CLR C
0A89- F7        0465          RLC A
0A8A- A1        0466          MOV @R1,A
0A8B- FC        0467          MOV A,R4
0A8C- F7        0468          RLC A
0A8D- A0        0469          MOV @R0,A
0A8E- 83        0470          RET
                0471 ************
                0472 *          *
                0473 * SUB TRAN *
                0474 *          *
                0475 ************
0A8F- FE        0476 TRAN     MOV A,R6
0A90- 53 0F     0477          ANL A,#$0F
0A92- A0        0478          MOV @R0,A
0A93- 18        0479          INC R0
0A94- FF        0480          MOV A,R7
0A95- 47        0481          SWAP A
0A96- 53 0F     0482          ANL A,#$0F
0A98- A0        0483          MOV @R0,A
0A99- FF        0484          MOV A,R7
0A9A- 53 0F     0485          ANL A,#$0F
0A9C- 18        0486          INC R0
0A9D- A0        0487          MOV @R0,A
0A9E- 83        0488          RET
                0489 ****************
                0490 *              *
                0491 * SUB MOV.PC,IO *
                0492 *              *
                0493 ****************
0A9F- B8 2F     0494 MOVPC    MOV R0,#47
0AA1- B9 30     0495          MOV R1,#48
0AA3- F0        0496          MOV A,@R0
0AA4- AC        0497          MOV R4,A
0AA5- F1        0498          MOV A,@R1
0AA6- AD        0499          MOV R5,A
0AA7- 83        0500          RET
                0501 ****************
0AA8- B8 31     0502 MOVIO    MOV R0,#49
0AAA- B9 32     0503          MOV R1,#50
0AAC- F0        0504          MOV A,@R0
0AAD- AC        0505          MOV R4,A
0AAE- F1        0506          MOV A,@R1
0AAF- AD        0507          MOV R5,A
0AB0- 83        0508          RET
                0509 ************
                0510 *          *
                0511 * SUB ENTR *
                0512 *          *
```

```
                    0513 *************
0AB1- BF 00         0514 ENTR   MOV R7,#$00
0AB3- BE 00         0515        MOV R6,#$00
0AB5- B8 35         0516        MOV R0,#53
0AB7- F0            0517        MOV A,@R0
0AB8- A9            0518        MOV R1,A
0AB9- BD 64         0519        MOV R5,#100
0ABB- 97            0520        CLR C
0ABC- 54 49         0521        CALL ADD3
0ABE- B8 36         0522        MOV R0,#54
0AC0- F0            0523        MOV A,@R0
0AC1- A9            0524        MOV R1,A
0AC2- BD 0A         0525        MOV R5,#10
0AC4- 97            0526        CLR C
0AC5- 54 49         0527        CALL ADD3
0AC7- B8 37         0528        MOV R0,#55
0AC9- F0            0529        MOV A,@R0
0ACA- A9            0530        MOV R1,A
0ACB- BD 01         0531        MOV R5,#1
0ACD- 97            0532        CLR C
0ACE- 54 49         0533        CALL ADD3
0AD0- 83            0534        RET
                    0535 **********************
                    0536 *                      *
                    0537 * SUB INCPC & DECPC *
                    0538 *                      *
                    0539 **********************
0AD1- B8 30         0540 INCPC  MOV R0,#48
0AD3- F0            0541        MOV A,@R0
0AD4- 97            0542        CLR C
0AD5- 17            0543        INC A
0AD6- A0            0544        MOV @R0,A
0AD7- E6 DD         0545        JNC RETNI
0AD9- C8            0546        DEC R0
0ADA- F0            0547        MOV A,@R0
0ADB- 17            0548        INC A
0ADC- A0            0549        MOV @R0,A
0ADD- 83            0550 RETNI  RET
                    0551 **********************
0ADE- B8 30         0552 DECPC  MOV R0,#48
0AE0- F0            0553        MOV A,@R0
0AE1- 97            0554        CLR C
0AE2- A7            0555        CPL C
0AE3- 07            0556        DEC A
0AE4- A0            0557        MOV @R0,A
0AE5- F6 EB         0558        JC RETD
0AE7- C8            0559        DEC R0
0AE8- F0            0560        MOV A,@R0
0AE9- 07            0561        DEC A
0AEA- A0            0562        MOV @R0,A
0AEB- 83            0563 RETD   RET
0AEC- 54 D1         0564 UP1    CALL INCPC
```

```
0AEE- 54 68     0565          CALL SCALL
0AF0- 84 CF     0566          JMP SCANO
                0567 ********
                0568 *       *
                0569 * DATA *
                0570 *       *
                0571 ********
                0572          .OR 300H
                0573          .TA 0B00H
0300- FC 60 DA
0303- F2 66 B6
0306- BF E0     0574          .DA $FC60,$DAF2,$66B6,$BFE0
0308- FE E6     0575          .DA $FEE6
                0576
                0577          .OR 310H
                0578          .TA 0B10H
0310- 80 40 20
0313- 10 08 04
0316- 02 01     0579          .DA $8040,$2010,$0804,$0201
                0580
                0581          .OR 320H
                0582          .TA 0B20H
0320- 10 20 30
0323- 40 50 60
0326- 70 80     0583          .DA $1020,$3040,$5060,$7080
0328- 90 A0 B0
032B- C0 D0 E0
032E- F0 00     0584          .DA $90A0,$B0C0,$D0E0,$F000
                0585          .OR 330H
                0586          .TA 0B30H
0330- 00 01 02
0333- 03 04 05
0336- 06 07     0587          .DA $0001,$0203,$0405,$0607
0338- 08 09 10
033B- 11 12 13
033E- 14 15     0588          .DA $0809,$1011,$1213,$1415
                0589          .OR 340H
                0590          .TA 0B40H
0340- 00 00 00
0343- 16 00 32
0346- 00 48     0591          .DA $0000,$0016,$0032,$0048
0348- 00 64 00
034B- 80 00 96
034E- 01 12     0592          .DA $0064,$0080,$0096,$0112
0350- 01 28 01
0353- 44 01 60
0356- 01 76     0593          .DA $0128,$0144,$0160,$0176
0358- 01 92 02
035B- 08 02 24
035E- 02 40     0594          .DA $0192,$0208,$0224,$0240
                0595          .OR 360H
                0596          .TA 0B60H
```

```
0360- 00 00 02
0363- 56 05 12
0366- 07 68      0597          .DA $0000,$0256,$0512,$0768
0368- 10 24 12
036B- 80 15 36
036E- 17 92      0598          .DA $1024,$1280,$1536,$1792
0370- 20 48      0599          .DA $2048
                 0600          .OR 380H
                 0601          .TA 0B80H
0380- 00 80 80
0383- 40 40 20
0386- 20 10      0602          .DA $0080,$8040,$4020,$2010
0388- 04 04 08
038B- 08 02 FF
038E- 10 02      0603          .DA $0404,$080B,$02FF,$1002
                 0604 ********************
                 0605 *                  *
                 0606 * SUB READ&WRITE *
                 0607 *                  *
                 0608 ********************
0390- B8 2D      0609 WRITE    MOV R0,#45
0392- B9 02      0610          MOV R1,#02
0394- F0         0611          MOV A,§R0
0395- 91         0612          MOVX §R1,A
0396- 23 20      0613          MOV A,#$20
0398- 3A         0614          OUTL P2,A
0399- B8 2E      0615          MOV R0,#46
039B- F0         0616          MOV A,§R0
039C- A9         0617          MOV R1,A
039D- B8 33      0618          MOV R0,#51
039F- F0         0619          MOV A,§R0
03A0- 91         0620          MOVX §R1,A
03A1- 19         0621          INC R1
03A2- C8         0622          DEC R0
03A3- F0         0623          MOV A,§R0
03A4- 91         0624          MOVX §R1,A
03A5- 23 10      0625          MOV A,#$10
03A7- 3A         0626          OUTL P2,A
03A8- 83         0627          RET
                 0628 ********************
03A9- B8 2D      0629 READ     MOV R0,#45
03AB- B9 02      0630          MOV R1,#02
03AD- F0         0631          MOV A,§R0
03AE- 91         0632          MOVX §R1,A
03AF- 23 20      0633          MOV A,#$20
03B1- 3A         0634          OUTL P2,A
03B2- B8 2E      0635          MOV R0,#46
03B4- F0         0636          MOV A,§R0
03B5- A9         0637          MOV R1,A
03B6- 81         0638          MOVX A,§R1
03B7- B8 33      0639          MOV R0,#51
03B9- A0         0640          MOV §R0,A
```

```
03BA- 18          0641          INC R0
03BB- AA          0642          MOV R2,A
03BC- 53 F0       0643          ANL A,#$F0
03BE- A0          0644          MOV §R0,A
03BF- B8 31       0645          MOV R0,#49
03C1- FA          0646          MOV A,R2
03C2- 53 0F       0647          ANL A,#$0F
03C4- A0          0648          MOV §R0,A
03C5- 19          0649          INC R1
03C6- 81          0650          MOVX A,§R1
03C7- 18          0651          INC R0
03C8- A0          0652          MOV §R0,A
03C9- 23 10       0653          MOV A,#$10
03CB- 3A          0654          OUTL P2,A
03CC- 83          0655          RET
                  0656 ********************
                  0657 *                  *
                  0658 * HEX.TO DECIMAL *
                  0659 *                  *
                  0660 ********************
03CD- 23 0F       0661 HTOD     MOV A,#$0F
03CF- B8 0F       0662          MOV R0,#$0F
03D1- B9 30       0663          MOV R1,#$30
03D3- 5D          0664          ANL A,R5
03D4- AA          0665          MOV R2,A
03D5- FA          0666 ZX1      MOV A,R2
03D6- 58          0667          ANL A,R0
03D7- C6 DD       0668          JZ TLP30
03D9- C8          0669          DEC R0
03DA- 19          0670          INC R1
03DB- 64 D5       0671          JMP ZX1
03DD- F9          0672 TLP30    MOV A,R1
03DE- E3          0673          MOVP3 A,§A
03DF- B8 25       0674          MOV R0,#37
03E1- A0          0675          MOV §R0,A
                  0676 ********************
03E2- 23 F0       0677          MOV A,#$F0
03E4- B8 0F       0678          MOV R0,#$0F
03E6- B9 40       0679          MOV R1,#$40
03E8- 5D          0680          ANL A,R5
03E9- 47          0681          SWAP A
03EA- AA          0682          MOV R2,A
03EB- FA          0683 ZX2      MOV A,R2
03EC- 58          0684          ANL A,R0
03ED- C6 F4       0685          JZ TLP40
03EF- C8          0686          DEC R0
03F0- 19          0687          INC R1
03F1- 19          0688          INC R1
03F2- 64 EB       0689          JMP ZX2
03F4- F9          0690 TLP40    MOV A,R1
03F5- E3          0691          MOVP3 A,§A
```

```
03F6- B8 27     0692          MOV R0,#39
03F8- A0        0693          MOV @R0,A
03F9- 19        0694          INC R1
03FA- C8        0695          DEC R0
03FB- F9        0696          MOV A,R1
03FC- E3        0697          MOVP3 A,@A
03FD- A0        0698          MOV @R0,A
                0699 ****************
03FE- 23 0F     0700          MOV A,#$0F
0400- B8 0F     0701          MOV R0,#$0F
0402- B9 60     0702          MOV R1,#$60
0404- 5C        0703          ANL A,R4
0405- AA        0704          MOV R2,A
0406- FA        0705 ZX3      MOV A,R2
0407- 58        0706          ANL A,R0
0408- C6 0F     0707          JZ TLP60
040A- C8        0708          DEC R0
040B- 19        0709          INC R1
040C- 19        0710          INC R1
040D- 84 06     0711          JMP ZX3
040F- F9        0712 TLP60    MOV A,R1
0410- E3        0713          MOVP3 A,@A
0411- B8 29     0714          MOV R0,#41
0413- A0        0715          MOV @R0,A
0414- 19        0716          INC R1
0415- C8        0717          DEC R0
0416- F9        0718          MOV A,R1
0417- E3        0719          MOVP3 A,@A
0418- A0        0720          MOV @R0,A
0419- 97        0721          CLR C
041A- B8 25     0722          MOV R0,#37
041C- B9 26     0723          MOV R1,#38
041E- F0        0724          MOV A,@R0
041F- 61        0725          ADD A,@R1
0420- E6 2B     0726          JNC AX10
0422- AA        0727 AX11     MOV R2,A
0423- 23 00     0728          MOV A,#$00
0425- 19        0729          INC R1
0426- 71        0730          ADDC A,@R1
0427- A1        0731          MOV @R1,A
0428- FA        0732          MOV A,R2
0429- 97        0733          CLR C
042A- C9        0734          DEC R1
042B- 57        0735 AX10     DA A
042C- F6 22     0736          JC AX11
042E- B8 28     0737          MOV R0,#40
0430- 60        0738          ADD A,@R0
0431- E6 3C     0739          JNC AX12
0433- AA        0740 AX13     MOV R2,A
0434- 23 00     0741          MOV A,#$00
0436- 19        0742          INC R1
0437- 71        0743          ADDC A,@R1
```

```
0438- A1          0744          MOV §R1,A
0439- 97          0745          CLR C
043A- C9          0746          DEC R1
043B- FA          0747          MOV A,R2
043C- 57          0748 AX12     DA A
043D- F6 33       0749          JC AX13
043F- AF          0750          MOV R7,A
0440- 19          0751          INC R1
0441- 18          0752          INC R0
0442- F1          0753          MOV A,§R1
0443- 60          0754          ADD A,§R0
0444- 57          0755          DA A
0445- AE          0756          MOV R6,A
0446- 83          0757          RET
                  0758 ******************
                  0759 *                *
                  0760 * SUB D.TO CODE *
                  0761 *                *
                  0762 ******************
0447- B9 2A       0763 DTOCO    MOV R1,#42
0449- 94 69       0764          CALL DIS50
044B- B8 3A       0765          MOV R0,#58
044D- A0          0766          MOV §R0,A
044E- 19          0767          INC R1
044F- 94 69       0768          CALL DIS50
0451- 18          0769          INC R0
0452- A0          0770          MOV §R0,A
0453- 19          0771          INC R1
0454- 94 69       0772          CALL DIS50
0456- 18          0773          INC R0
0457- A0          0774          MOV §R0,A
0458- B9 35       0775          MOV R1,#53
045A- 94 69       0776          CALL DIS50
045C- 18          0777          INC R0
045D- A0          0778          MOV §R0,A
045E- 19          0779          INC R1
045F- 94 69       0780          CALL DIS50
0461- 18          0781          INC R0
0462- A0          0782          MOV §R0,A
0463- 19          0783          INC R1
0464- 94 69       0784          CALL DIS50
0466- 18          0785          INC R0
0467- A0          0786          MOV §R0,A
0468- 83          0787          RET
                  0788 ******************
0469- BA 09       0789 DIS50    MOV R2,#$09
046B- BB F6       0790          MOV R3,#$F6
046D- FB          0791 D80      MOV A,R3
046E- 41          0792          ORL A,§R1
046F- 37          0793          CPL A
0470- C6 76       0794          JZ D90
```

```
0472- CA        0795          DEC R2
0473- 1B        0796          INC R3
0474- 84 6D     0797          JMP D80
0476- FA        0798 D90      MOV A,R2
0477- E3        0799          MOVP3 A,§A
0478- 83        0800          RET
                0801 ***************
                0802 *             *
                0803 * INST.TO DIS.*
                0804 *             *
                0805 ***************
0479- B8 34     0806 DIS60    MOV R0,#52
047B- F0        0807          MOV A,§R0
047C- 47        0808          SWAP A
047D- AD        0809          MOV R5,A
047E- B8 10     0810          MOV R3,#$10
0480- BC 90     0811          MOV R4,#$90
0482- 94 CA     0812          CALL D70
0484- C6 C2     0813          JZ D600
0486- 94 CA     0814          CALL D70
0488- C6 C2     0815          JZ D600
048A- 94 CA     0816          CALL D70
048C- C6 C2     0817          JZ D600
048E- 94 CA     0818          CALL D70
0490- C6 C6     0819          JZ D700
0492- 94 CA     0820          CALL D70
0494- C6 C2     0821          JZ D600
0496- 94 CA     0822          CALL D70
0498- C6 C6     0823          JZ D700
049A- 94 CA     0824          CALL D70
049C- C6 C2     0825          JZ D600
049E- 94 CA     0826          CALL D70
04A0- C6 C6     0827          JZ D700
04A2- 94 CA     0828          CALL D70
04A4- C6 C6     0829          JZ D700
04A6- 94 CA     0830          CALL D70
04A8- C6 C2     0831          JZ D600
04AA- 94 CA     0832          CALL D70
04AC- C6 C6     0833          JZ D700
04AE- 94 CA     0834          CALL D70
04B0- C6 C2     0835          JZ D600
04B2- 94 CA     0836          CALL D70
04B4- C6 C6     0837          JZ D700
04B6- 94 CA     0838          CALL D70
04B8- C6 C2     0839          JZ D600
04BA- 94 CA     0840          CALL D70
04BC- C6 C6     0841          JZ D700
04BE- 94 CA     0842          CALL D70
04C0- C6 C6     0843          JZ D700
                0844 ****************
04C2- FC        0845 D600     MOV A,R4
```

```
04C3- 54 1C     0846          CALL TCD
04C5- 83        0847          RET
04C6- FC        0848 D700     MOV A,R4
04C7- 54 14     0849          CALL TND
04C9- 83        0850          RET
04CA- CC        0851 D70      DEC R4
04CB- CB        0852          DEC R3
04CC- FB        0853          MOV A,R3
04CD- DD        0854          XRL A,R5
04CE- 83        0855          RET
                0856 ******************
                0857 ******************
04CF- D5        0858 SCANO    SEL RB1
04D0- B8 00     0859          MOV R0,#00
04D2- B9 00     0860          MOV R1,#00
04D4- BA 00     0861          MOV R2,#00
04D6- C5        0862          SEL RB0
04D7- 24 B8     0863          JMP SCAN69
                0864 ******************
                0865 *&****************
04D9- B8 18     0866 SCAND    MOV R0,#24
04DB- 23 07     0867          MOV A,#7
04DD- D0        0868          XRL A,§R0
04DE- 96 E2     0869          JNZ DE100
04E0- 94 E4     0870          CALL SCAND1
04E2- 84 CF     0871 DE100    JMP SCANO
                0872 ******************
04E4- 94 FE     0873 SCAND1   CALL DE1
04E6- 54 81     0874 DE20     CALL MULTI
04E8- 74 90     0875          CALL WRITE
04EA- B8 30     0876          MOV R0,#48
04EC- F0        0877          MOV A,§R0
04ED- D3 FF     0878          XRL A,#$FF
04EF- C6 F5     0879          JZ DE30
04F1- 54 D1     0880          CALL INCPC
04F3- 84 E6     0881          JMP DE20
04F5- 94 FE     0882 DE30     CALL DE1
04F7- 54 68     0883          CALL SCALL
04F9- 83        0884          RET
04FA- 00        0885          NOP
04FB- 00        0886          NOP
04FC- 00        0887          NOP
04FD- 00        0888          NOP
                0889 ******************
04FE- B8 37     0890 DE1      MOV R0,#55
0500- 23 13     0891          MOV A,#19
0502- B0 00     0892 DE10     MOV §R0,#00
0504- 07        0893          DEC A
0505- C8        0894          DEC R0
0506- 96 02     0895          JNZ DE10
0508- 83        0896          RET
```

```
                        0897 ****************
                        0898 *               *
                        0899 * FUN. KEY DIS.*
                        0900 *               *
                        0901 ****************
0509- 54 B1             0902 ADDDIS CALL ENTR
050B- B8 2F             0903        MOV RO,#47
050D- FF                0904        MOV A,R7
050E- A0                0905        MOV §RO,A
050F- FE                0906        MOV A,R6
0510- 18                0907        INC RO
0511- A0                0908        MOV §RO,A
0512- 54 68             0909        CALL SCALL
0514- B4 18             0910        CALL SUBDIS
0516- 24 B8             0911        JMP SCAN69
                        0912 ****************
0518- D5                0913 SUBDIS SEL RB1
0519- 23 02             0914        MOV A,#02
051B- 48                0915        ORL A,RO
051C- A8                0916        MOV RO,A
051D- C5                0917        SEL RBO
051E- 83                0918        RET
                        0919 *************
                        0920 *           *
                        0921 * CLEAR FUN.*
                        0922 *           *
                        0923 *************
051F- B4 2E             0924 CLR100 CALL SUBCLR
0521- B8 1A             0925        MOV RO,#26
0523- F0                0926        MOV A,§RO
0524- C6 2A             0927        JZ CLR1
0526- B4 4A             0928        CALL CLR3
0528- 24 B8             0929        JMP SCAN69
052A- B4 3F             0930 CLR1   CALL CLR2
052C- 24 B8             0931        JMP SCAN69
                        0932 ******************
052E- D5                0933 SUBCLR SEL RB1
052F- 23 01             0934        MOV A,#01
0531- 48                0935        ORL A,RO
0532- A8                0936        MOV RO,A
0533- 19                0937        INC R1
0534- 23 03             0938        MOV A,#03
0536- D9                0939        XRL A,R1
0537- 96 3D             0940        JNZ CL10
0539- B9 00             0941        MOV R1,#00
053B- BA FF             0942        MOV R2,#$FF
053D- C5                0943 CL10   SEL RBO
053E- 83                0944        RET
                        0945 ******************
053F- B8 3F             0946 CLR2   MOV RO,#63
0541- 23 03             0947        MOV A,#3
0543- B0 FC             0948 CLR20  MOV §RO,#$FC
```

```
0545- 07        0949          DEC A
0546- C8        0950          DEC R0
0547- 96 43     0951          JNZ CLR20
0549- 83        0952          RET
                0953 ******************
054A- 23 00     0954 CLR3     MOV A,#00
054C- B8 1A     0955          MOV R0,#26
054E- A0        0956          MOV @R0,A
054F- B8 3F     0957 CLR31    MOV R0,#63
0551- 23 06     0958          MOV A,#6
0553- B0 FC     0959 CLR30    MOV @R0,#$FC
0555- 07        0960          DEC A
0556- C8        0961          DEC R0
0557- 96 53     0962          JNZ CLR30
0559- B0 00     0963          MOV @R0,#00
055B- C8        0964          DEC R0
055C- B0 00     0965          MOV @R0,#00
055E- 83        0966          RET
                0967 ***************
                0968 *             *
                0969 * SERCH FUN.  *
                0970 *             *
                0971 ***************
055F- 54 B1     0972 SERCH1   CALL ENTR
0561- B4 93     0973          CALL TRAND
0563- B4 82     0974          CALL TRSE1
0565- 54 D1     0975 SER10    CALL INCPC
0567- 54 81     0976          CALL MULTI
0569- 74 A9     0977          CALL READ
056B- B8 33     0978          MOV R0,#51
056D- B9 1E     0979          MOV R1,#30
056F- F0        0980          MOV A,@R0
0570- D3 00     0981          XRL A,#00
0572- C6 7E     0982          JZ SER20
0574- F0        0983          MOV A,@R0
0575- D1        0984          XRL A,@R1
0576- 96 65     0985          JNZ SER10
0578- C8        0986          DEC R0
0579- 19        0987          INC R1
057A- F0        0988          MOV A,@R0
057B- D1        0989          XRL A,@R1
057C- 96 65     0990          JNZ SER10
057E- 54 68     0991 SER20    CALL SCALL
0580- 24 B8     0992          JMP SCAN69
                0993 ******************
0582- D5        0994 TRSE1    SEL RB1
0583- 23 04     0995          MOV A,#04
0585- 48        0996          ORL A,R0
0586- A8        0997          MOV R0,A
0587- C5        0998          SEL RB0
0588- B8 32     0999          MOV R0,#50
```

```
058A- B9 1F     1000          MOV R1,#31
058C- F0        1001          MOV A,§R0
058D- A1        1002          MOV §R1,A
058E- 18        1003          INC R0
058F- C9        1004          DEC R1
0590- F0        1005          MOV A,§R0
0591- A1        1006          MOV §R1,A
0592- 83        1007          RET
                1008 ****************
0593- B8 31     1009 TRAND    MOV R0,#49
0595- B9 32     1010          MOV R1,#50
0597- FF        1011          MOV A,R7
0598- A0        1012          MOV §R0,A
0599- FE        1013          MOV A,R6
059A- A1        1014          MOV §R1,A
059B- B8 34     1015          MOV R0,#52
059D- F0        1016          MOV A,§R0
059E- 4F        1017          ORL A,R7
059F- B8 33     1018          MOV R0,#51
05A1- A0        1019          MOV §R0,A
05A2- 83        1020          RET
                1021 ***************
                1022 *             *
                1023 * INSERT FUN.*
                1024 *             *
                1025 ***************
05A3- B8 2F     1026 INS100   MOV R0,#47
05A5- B9 20     1027          MOV R1,#32
05A7- B4 D7     1028          CALL INS101
05A9- B8 32     1029          MOV R0,#50
05AB- B9 22     1030          MOV R1,#34
05AD- B4 D7     1031          CALL INS101
05AF- 54 D1     1032 INS103   CALL INCPC
05B1- 54 81     1033          CALL MULTI
05B3- 74 A9     1034          CALL READ
05B5- B4 DE     1035          CALL INS102
05B7- 54 81     1036          CALL MULTI
05B9- 74 90     1037          CALL WRITE
05BB- B8 23     1038          MOV R0,#35
05BD- F0        1039          MOV A,§R0
05BE- D3 00     1040          XRL A,#00
05C0- 96 AF     1041          JNZ INS103
05C2- B8 20     1042          MOV R0,#32
05C4- B9 2F     1043          MOV R1,#47
05C6- B4 D7     1044          CALL INS101
05C8- B8 32     1045          MOV R0,#50
05CA- B0 00     1046          MOV §R0,#00
05CC- 18        1047          INC R0
05CD- B0 00     1048          MOV §R0,#00
05CF- 54 81     1049          CALL MULTI
05D1- 74 90     1050          CALL WRITE
```

```
05D3- 54 68     1051          CALL SCALL
05D5- B4 CF     1052          JMP SCAN0
                1053 ********************
05D7- F0        1054 INS101   MOV A,@R0
05D8- A1        1055          MOV @R1,A
05D9- 18        1056          INC R0
05DA- 19        1057          INC R1
05DB- F0        1058          MOV A,@R0
05DC- A1        1059          MOV @R1,A
05DD- 83        1060          RET
05DE- B8 22     1061 INS102   MOV R0,#34
05E0- B9 32     1062          MOV R1,#50
05E2- 23 02     1063          MOV A,#02
05E4- 20        1064 INS02    XCH A,@R0
05E5- 21        1065          XCH A,@R1
05E6- 20        1066          XCH A,@R0
05E7- 18        1067          INC R0
05E8- 19        1068          INC R1
05E9- 07        1069          DEC A
05EA- 96 E4     1070          JNZ INS02
05EC- 83        1071          RET
                1072 ***************
                1073 *             *
                1074 * DELETE FUN.*
                1075 *             *
                1076 ***************
05ED- B8 2F     1077 DET100   MOV R0,#47
05EF- B9 20     1078          MOV R1,#32
05F1- B4 D7     1079          CALL INS101
05F3- 54 D1     1080 DT101    CALL INCPC
05F5- 54 81     1081          CALL MULTI
05F7- 74 A9     1082          CALL READ
05F9- 54 DE     1083          CALL DECPC
05FB- 54 81     1084          CALL MULTI
05FD- 74 90     1085          CALL WRITE
05FF- B8 33     1086          MOV R0,#51
0601- F0        1087          MOV A,@R0
0602- D3 00     1088          XRL A,#00
0604- C6 0A     1089          JZ DT100
0606- 54 D1     1090          CALL INCPC
0608- A4 F3     1091          JMP DT101
060A- B8 20     1092 DT100    MOV R0,#32
060C- B9 2F     1093          MOV R1,#47
060E- B4 D7     1094          CALL INS101
0610- 54 68     1095          CALL SCALL
0612- 84 CF     1096          JMP SCAN0


SYMBOL TABLE

09AC- ADD
```

```
0A53- ADD1
0A4F- ADD2
0A49- ADD3
0509- ADDDIS
0886- AND
088E- ANDC
042B- AX10
0422- AX11
043C- AX12
0433- AX13
053D- CL10
09AA- CLR
052A- CLR1
051F- CLR100
053F- CLR2
0543- CLR20
054A- CLR3
0553- CLR30
054F- CLR31
04C2- D600
04CA- D70
04C6- D700
046D- D80
0476- D90
04FE- DE1
0502- DE10
04E2- DE100
04E6- DE20
04F5- DE30
0ADE- DECPC
09B4- DEL
05ED- DET100
09F0- DIS1
0A02- DIS2
0A0D- DIS3
0469- DIS50
0479- DIS60
09B2- DOWN
08F3- DOWN1
060A- DT100
05F3- DT101
0447- DTOCO
08DE- END
09A8- ENT
0A5A- ENTER
0AB1- ENTR
03CD- HTOD
08B6- IEN
0AD1- INCPC
09B6- INS
05E4- INS02
```

```
05A3- INS100
05D7- INS101
05DE- INS102
05AF- INS103
08FC- KEY1
0980- KEY2
0876- LD
087E- LDC
0AA8- MOVIO
0A9F- MOVPC
0A81- MULTI
0930- N0
0938- N1
0940- N2
0948- N3
0950- N4
0958- N5
0960- N6
0968- N7
0970- N8
0978- N9
0A2A- NDIS
0A39- NDU
08BE- OEN
0896- OR
089E- ORC
09EC- RDIS1
09F7- RDIS2
0A09- RDIS3
03A9- READ
0AEB- RETD
0ADD- RETNI
0A68- SCALL
04CF- SCAN0
09BC- SCAN01
09D4- SCAN02
09E8- SCAN03
09D8- SCAN04
08E6- SCAN05
08E8- SCAN06
08EB- SCAN07
080E- SCAN1
0812- SCAN2
09B8- SCAN69
04D9- SCAND
04E4- SCAND1
0565- SER10
057E- SER20
09AE- SERCH
055F- SERCH1
08AE- SKZ
```

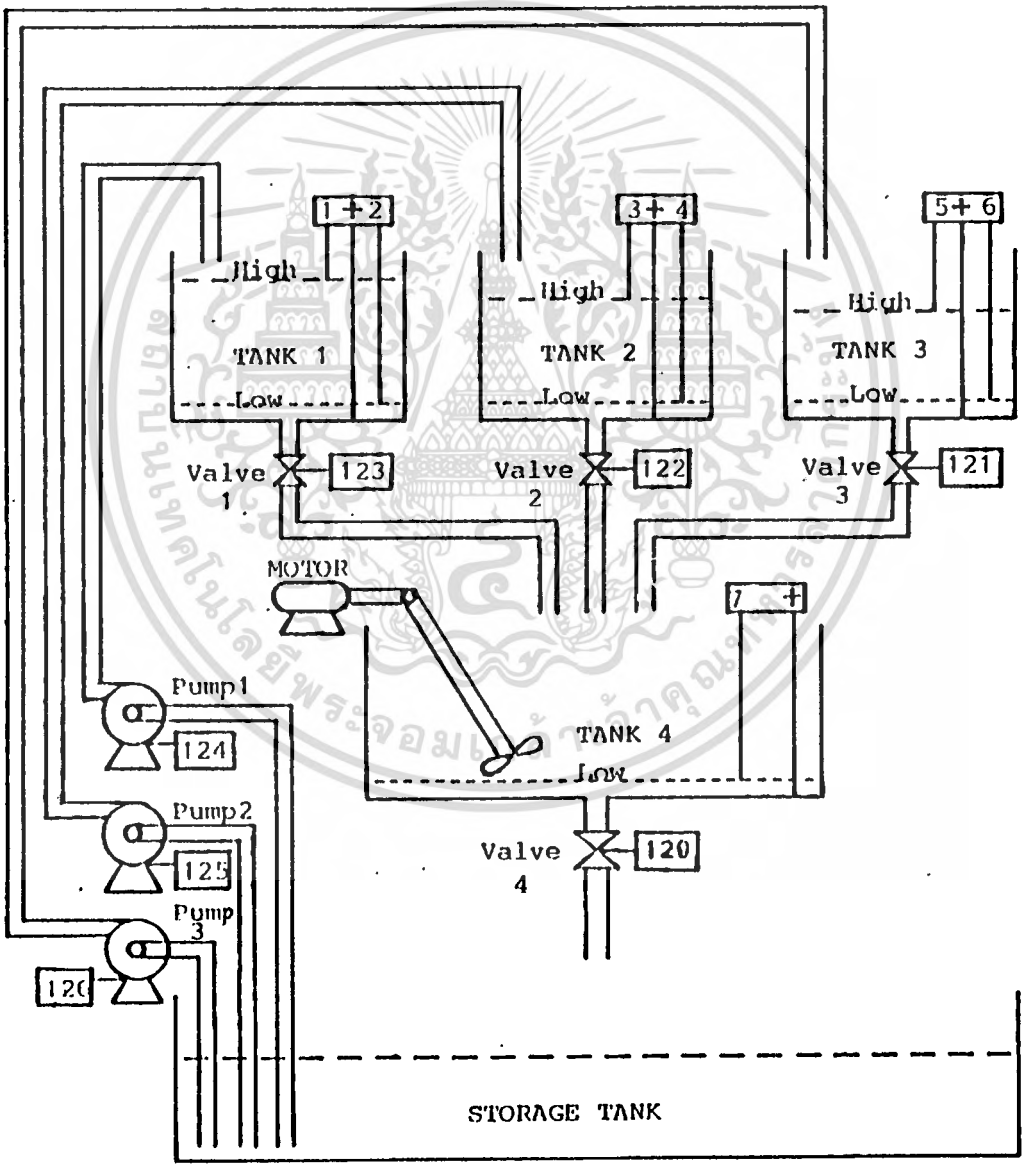
```
08CE- STC
08C6- STO
052E- SUBCLR
0518- SUBDIS
0A1C- TCD
08D6- TCSET
0A24- TITM
03DD- TLP30
03F4- TLP40
040F- TLP60
0A14- TND
0A8F- TRAN
0593- TRAND
0582- TRSE1
09B0- UP
0AEC- UP1
0390- WRITE
08A6- XNOR
03D5- ZX1
03EB- ZX2
0406- ZX3

0000 ERRORS IN ASSEMBLY
```

บทที่ 9

ตัวอย่างการนำไปใช้งาน

9.1 ตัวอย่างที่ 1 กระบวนการผสมของเหลว

ตัวอย่างการนำพีแอลซีขนาด 1 บิทไปควบคุมกระบวนการผสมของเหลว 3 ชนิด โดยมีเงื่อนไขของการผสมกันดังนี้



รูปที่ 9.1 ขบวนการผสมของเหลว

จากรูปที่ 9.1 มีแทงค์ (Tank) อยู่ 4 ใบ โดย T1,T2 และ T3 เป็นแทงค์ที่ใส่ของเหลวไว้ 3 ชนิด ส่วนแทงค์ T4 ใช้ในการผสมของเหลวจากแทงค์ T1,T2 และ T3 เมื่อเริ่มต้นกระบวนการ ปั๊ม P1,P2 และ P3 จะปั๊มของเหลวขึ้นมายังแทงค์ T1,T2 และ T3 จนเต็มแล้วจึงเริ่มกระบวนการผสมโดย

1. ตรวจว่าแทงค์ T1 และ T2 มีของเหลวเต็มแล้วหรือยัง ถ้าไม่มีของเหลวอยู่จะใช้ปั๊ม P1, และ P2 ปั๊มของเหลวขึ้นมาจนเต็มแล้วจึงหยุดปั๊ม

2. เมื่อของเหลว T1 และ T2 เต็มแล้ว S1 และ S2 ซึ่งเป็นโซลิ-นอยด์วาล์ว (Solinoid Valve) จะเปิดให้ของเหลวจากแทงค์ T1 และ T2 ไหลลงมายังแทงค์ T4 พร้อม ๆ กัน

3. เมื่อของเหลวจากแทงค์ T1 หรือ T2 หมด S1 หรือ S2 จะปิดและ P1 หรือ P2 จะปั๊มของเหลวมาเติมให้เต็มเพื่อรอซีเควนซ์ต่อไป

4. เมื่อของเหลวจากแทงค์ T1 และ T2 ถูกปล่อยมายังแทงค์ T4 หมดทั้งสองแทงค์แล้ว มอเตอร์ M ก็จะเริ่มกวนของเหลวในแทงค์ T4 ให้ผสมกันเป็นเวลา 10 วินาที

5. เมื่อของเหลวจากแทงค์ T1 และ T2 ในแทงค์ T4 ถูกกวนครบ 10 วินาทีแล้ว ของเหลวจากแทงค์ T3 จะถูกปล่อยลงมายังแทงค์ T4

6. ในขณะที่ของเหลวจากแทงค์ T3 ถูกปล่อยลงมายังแทงค์ T4 มอเตอร์ M ก็จะยังคงกวนต่อไปเรื่อย ๆ

7. เมื่อของเหลวจากแทงค์ T3 ถูกปล่อยลงมายังแทงค์ T4 หมดแล้ว โซลินอยด์วาล์ว S3 ก็จะปิด และปั๊ม P3 จะปั๊มของเหลวขึ้นมายังแทงค์ T3 จนเต็มเพื่อรอซีเควนซ์ต่อไป และในขณะที่โซลินอยด์วาล์ว S3 ปิด S4 ก็จะเปิดให้ของเหลวในแทงค์ T4 ซึ่งผสมกันเรียบร้อยแล้วถูกนำไปใช้งาน

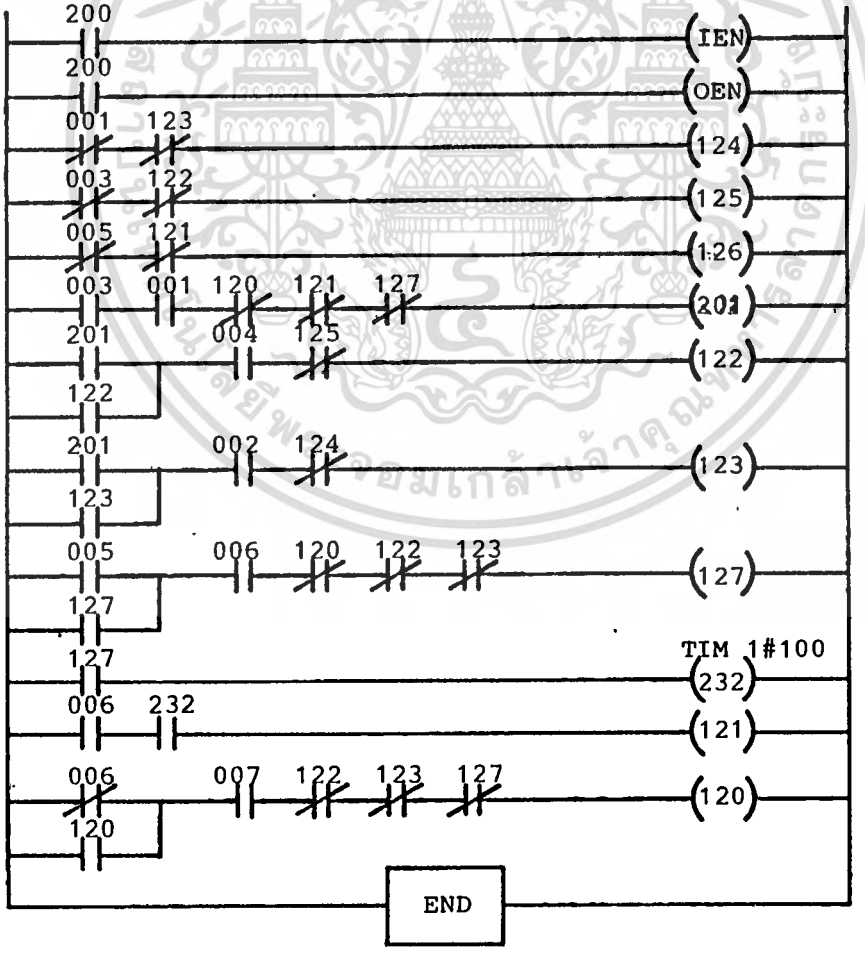
8. เมื่อของเหลวที่ผสมกันในแทงค์ T4 ถูกนำออกไปเพื่อใช้งานหมดแล้ว ก็จะเป็นการเริ่มต้นซีเควนซ์ใหม่โดยแทงค์ T1 และ T2 จะถูกปล่อยมายังแทงค์ T4 ใหม่

ในกระบวนการผสมของเหลวนี้ จะใช้อีเลคโตรด (Electrode) เป็นตัววัด

ระดับของของเหลวว่าเต็มหรือหมดแทงค์หรือยัง โดยแทงค์ T1,T2 และ T3 จะมีอีเลค-ไตรดอยู่แทงค์ละ 3 ตัว โดยเป็น COMMON 1 ตัว และอีก 2 ตัว ใช้สำหรับตรวจระดับเต็มหรือสูงสุด และหมดแทงค์หรือต่ำสุด ส่วนในแทงค์ T4 จะตรวจระดับต่ำสุดหรือหมดแทงค์เพียงอย่างเดียว

ตำแหน่งของอินพุทและเอาท์พุทจากพีแอลซีที่ต่อมายังกระบวนการดังในรูปที่ 9.1 ซึ่งระบุไว้ตามตำแหน่งต่าง ๆ โดยในโปรแกรมการใช้งานจะอ้างถึงตำแหน่งต่าง ๆ จากกระบวนการ

9.11 แลดเดอร์ไดอะแกรมของกระบวนการผสมของเหลว



9.12 โปรแกรมการควบคุมกระบวนการผสมของเหลว

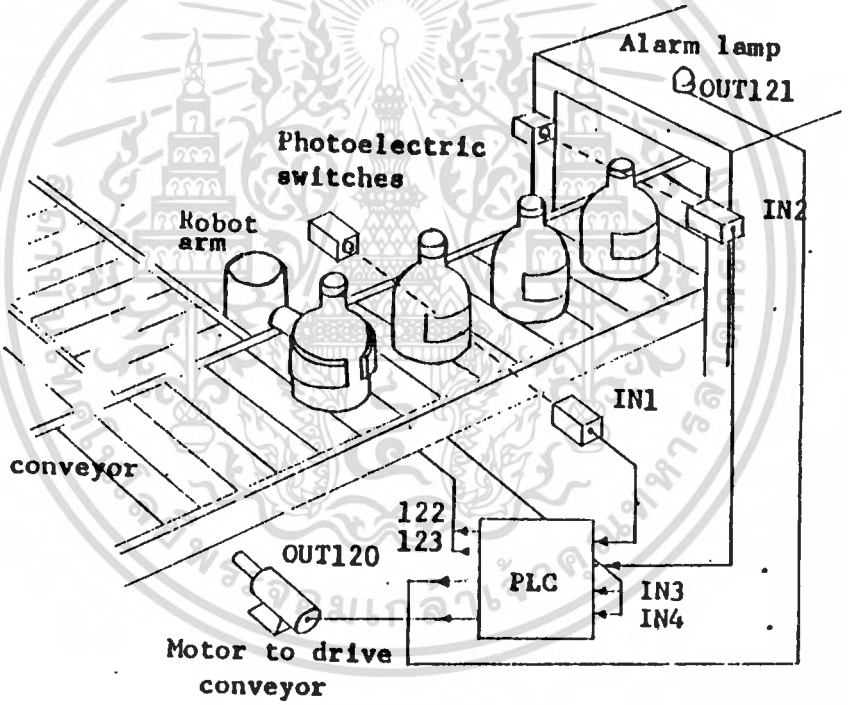
Step	OP-Code	Mnemonic	I/O Address
0	A0 C8	IEN	200
1	B0 C8	OEN	200
2	20 01	LDC	001
3	40 7B	ANDC	123
4	80 7C	STO	124
5	20 03	LDC	003
6	40 7A	ANDC	122
7	80 7D	STO	125
8	20 05	LDC	005
9	40 79	ANDC	121
10	80 7E	STO	126
11	10 03	LD	003
12	30 01	AND	001
13	40 78	ANDC	120
14	40 79	ANDC	121
15	40 7F	ANDC	127
16	80 C9	STO	201
17	10 C9	LD	201
18	S0 7A	OR	122
19	30 04	AND	004
20	40 7D	ANDC	125
21	80 7A	STO	122

Step	OP-Code	Mnemonic	I/O Address
22	10 C9	LD	201
23	50 7B	OR	123
24	30 02	AND	002
25	40 7C	ANDC	124
26	80 7B	STO	123
27	10 05	LD	005
28	50 7F	OR	127
29	30 06	AND	006
30	40 78	ANDC	120
31	40 7A	ANDC	122
32	40 7B	ANDC	123
33	80 7F	STO	127
34	10 7F	LD	127
35	80 E8	STO	232
36	C0 64	SET	100
37	10 06	LD	006
38	30 E8	AND	232
39	80 79	STO	121
40	20 06	LDC	006
41	50 78	OR	120
42	30 07	AND	007
43	40 7A	ANDC	122
44	40 7B	ANDC	123
45	48 7F	ANDC	127

Step	OP-Code	Mnemonic	I/O Address
46	80 78	STO	120
47	FO	END	

9.2 ตัวอย่างที่ 2 กระบวนการตรวจสอบการปิดฉลากข้างขวด

การนำพีแอลซีขนาด 1 บิทไปควบคุมกระบวนการตรวจสอบการปิดฉลากข้างขวด ดังในรูปที่ 9.2



รูปที่ 9.2 กระบวนการตรวจสอบการปิดฉลากข้างขวด

จากรูปที่ 9.2 เมื่อขวดผ่านการปิดฉลากที่ด้านข้างขวดแล้ว จะนำมาบรรจุหีบห่อโดยลำเลียงมาตามสายพาน ก่อนที่จะนำไปบรรจุหีบห่อควรมีการตรวจความเรียบร้อยขั้นสุดท้ายเสียก่อนว่าทุก ๆ ขวดมีการปิดฉลากเรียบร้อยหรือยัง ถ้ามีขวดใดขวดหนึ่งไม่เรียบร้อยก็จะถูกคัดออกและนำกลับไปเข้ากระบวนการปิดฉลากใหม่ โดยมีลำดับขั้นการทำงานดังนี้

1. สวิทช์แสงอินพุทที่ 2 จะตรวจว่าตำแหน่งขวดบนสายพานตรงกับตำแหน่งที่จะ

ตรวจการปิดฉลากหรือยัง เพราะระยะห่างของขวดแต่ละใบจะเท่า ๆ กัน

2. เมื่ออินพุท 2 เป็น Low แสดงว่าขวดอยู่ในตำแหน่งที่จะตรวจฉลาก อินพุท 2 ก็จะถูกอ่านว่าขวดมีการปิดฉลากแล้วหรือยัง

3. ถ้าอินพุท 2 เป็น High แสดงว่าไม่มีการปิดฉลากที่ข้างขวด พีแอลซีจะทำการจำข้อมูลไว้

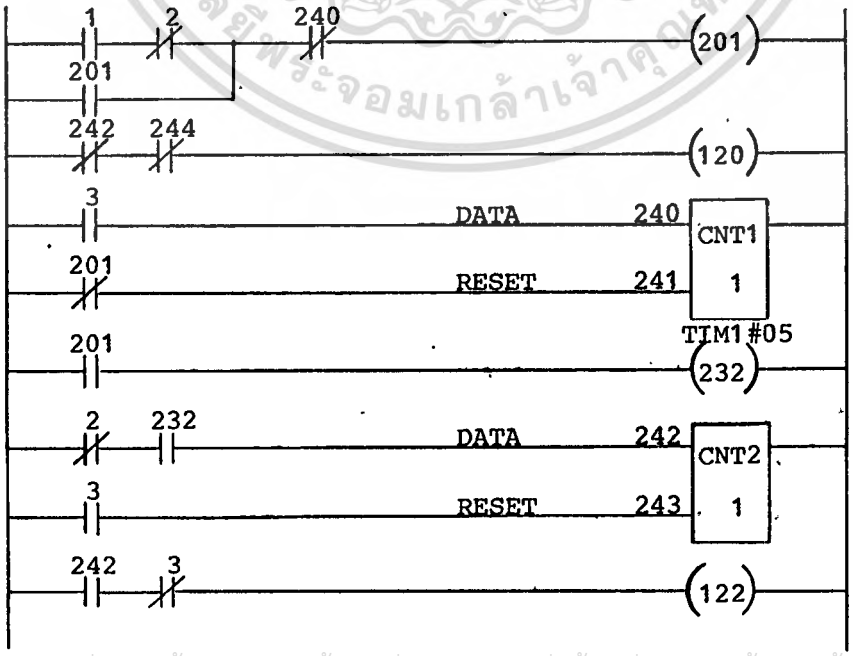
4. เมื่อขวดเลื่อนมายังตำแหน่งของแขนอัตโนมัติ พีแอลซีก็จะส่งเอาท์พุทให้แขนอัตโนมัติทำการหยิบขวดที่ไม่ได้ปิดฉลากออกไปไว้บนสายพานอีกอันหนึ่ง

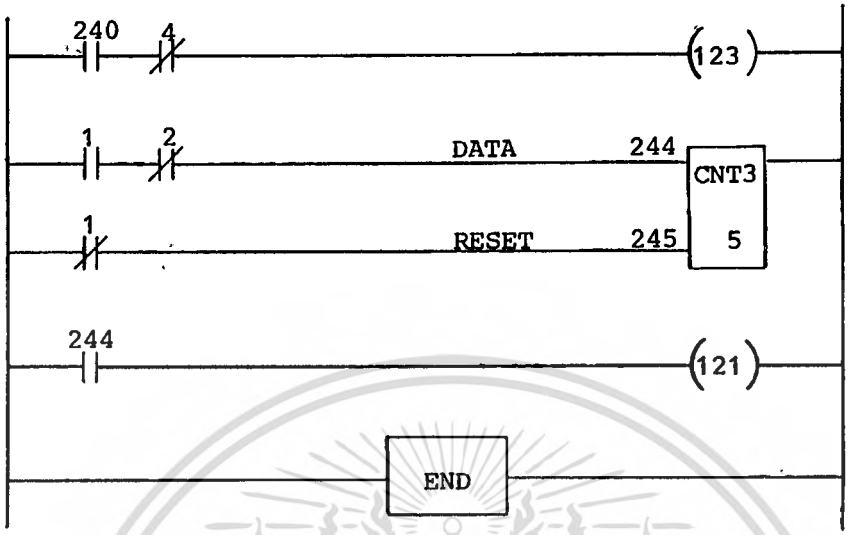
5. ถ้าอินพุท 1 ได้รับสัญญาณว่า ขวดไม่ได้รับการปิดฉลากติดต่อกันครบ 5 ขวด พีแอลซีก็จะสั่งให้สายพานหยุดและส่งเอาท์พุทออกไปยังดวงไฟที่ใช้เตือน (Alarm lamp)

6. เมื่อดวงไฟที่ใช้เตือนติดขึ้น ก็จะต้องมีการตรวจสอบระบบการปิดฉลากเสียก่อนว่ามีการผิดปกติหรือเปล่า ถ้าไม่มีก็เริ่มทำการ Start ระบบใหม่การทำงานก็จะเริ่มจากข้อ 1 ใหม่

การทำงานของพีแอลซีขนาด 1 บิท ในการตรวจการปิดฉลากข้างขวดเป็นตามแลดเดอร์ไดอะแกรมที่ 9.21 และโปรแกรมที่ 9.22 ดังนี้

9.21 แลดเดอร์ไดอะแกรมของกระบวนการตรวจสอบการปิดฉลาก





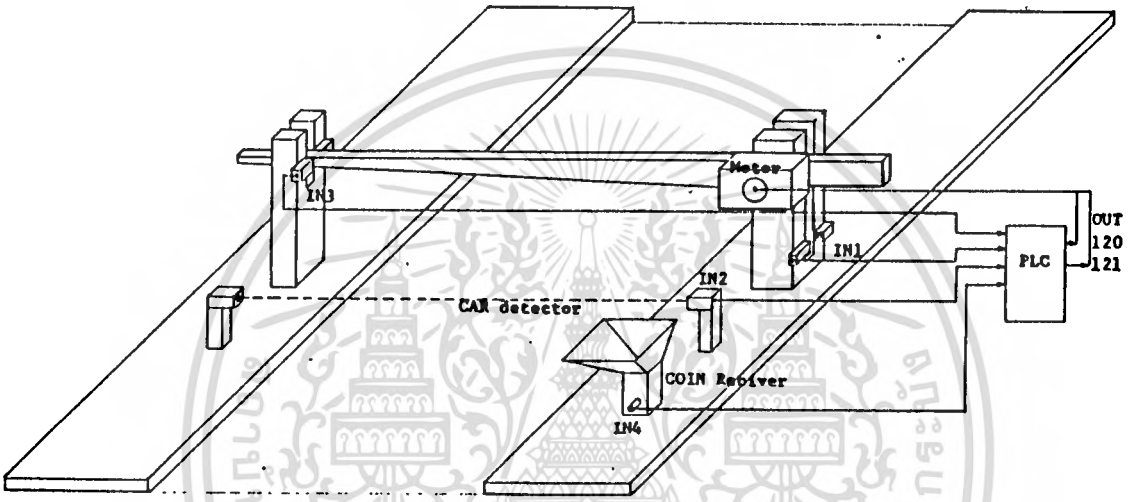
9.22 โปรแกรมควบคุมกระบวนการตรวจสอบการปิดฉลาก

Step	Mnemonic	I/O Address
0	LD	001
1	ANDC	002
2	OR	201
3	ANDC	240
4	STO	201
5	LDC	242
6	ANDC	244
7	STO	120
8	LD	003
9	STO	240
10	LDC	201
11	STO	241
12	SET	001
13	LD	201

14	STO	232
15	SET	005
16	LDC	002
17	AND	232
18	STO	242
19	LD	003
20	STO	243
21	SET	001
22	LD	242
23	ANCD	003
24	STO	122
25	LD	240
26	ANDC	004
27	STO	123
28	LD	001
29	ANDC	002
30	STO	244
31	LDC	001
32	STO	245
33	SET	005
34	LD	244
35	STO	121
36	END	

9.3 ตัวอย่างที่ 3 กระบวนการเก็บเงินอัตโนมัติของระบบทางด่วน

ตัวอย่างที่ 3 เป็นการนำเอาพีแอลซีขนาด 1 บิท ไปควบคุมระบบการเก็บเงินอัตโนมัติของทางด่วนดังในรูปที่ 9.3



รูปที่ 9.3 ระบบการเก็บเงินอัตโนมัติของทางด่วน

จากรูปที่ 9.3 เป็นระบบเก็บเงินอัตโนมัติของทางด่วน ซึ่งมีลำดับขั้นการทำงานดังนี้

1. เมื่อมีรถยนต์วิ่งเข้ามาจอดในช่องทางของด่านเก็บเงิน ซึ่งจะมีที่กั้นไม่ให้รถยนต์ผ่านจะทำให้สวิทช์แสงอินพุท 2 เป็น Low แสดงว่ามีรถมาออยู่

2. สวิทช์อินพุทที่ 4 จะทำการตรวจว่ามีการเสียเงินครบตามที่ตั้งไว้หรือเปล่า เมื่อผู้ที่ต้องการผ่านด่านเก็บเงิน นำรถยนต์เข้ามาจอดจะต้องหยอดเงินใส่ใน Coin Receiver จนครบจำนวนก่อน

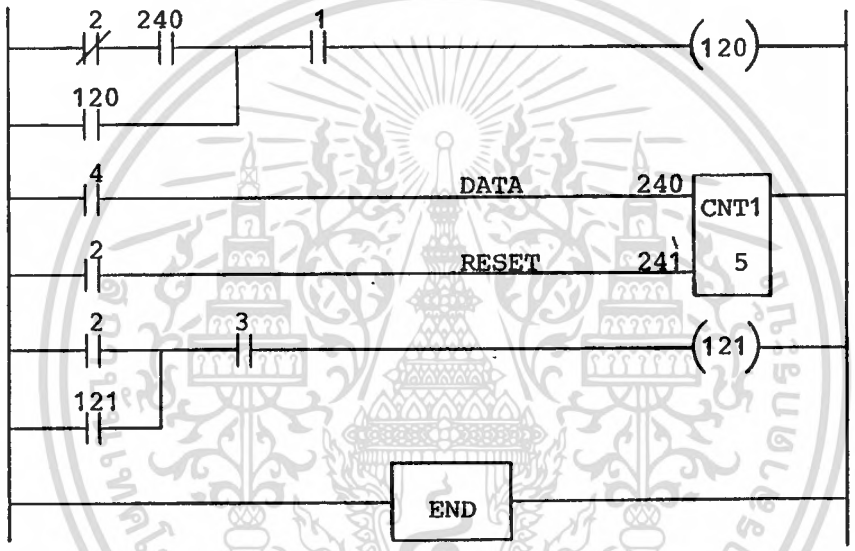
3. เมื่อพีแอลซีตรวจสวิทช์อินพุท 4 ว่ามีการหยอดเงินครบจำนวนแล้วจะทำการเปิดที่กั้นให้รถผ่านไปได้

4. เมื่อรถยนต์ผ่านที่กั้นไปแล้วสวิทช์อินพุท 2 จะมีสภาวะเป็น High พีแอลซีก็

จะส่งเอาท์พุทไปปิดที่กั้นลงมาและทำการรีเซทระบบเพื่อเริ่มต้นทำขั้นตอนใหม่

การทำงานของพีแอลซี ในการควบคุมระบบการเก็บเงินของทางด่วนอัตโนมัติมีการทำงานตามแลดเดอร์ไดอะแกรมที่ 9.31 และโปรแกรมการทำงานที่ 9.32

9.31 แลดเดอร์ไดอะแกรมกระบวนการเก็บเงินอัตโนมัติของระบบทางด่วน



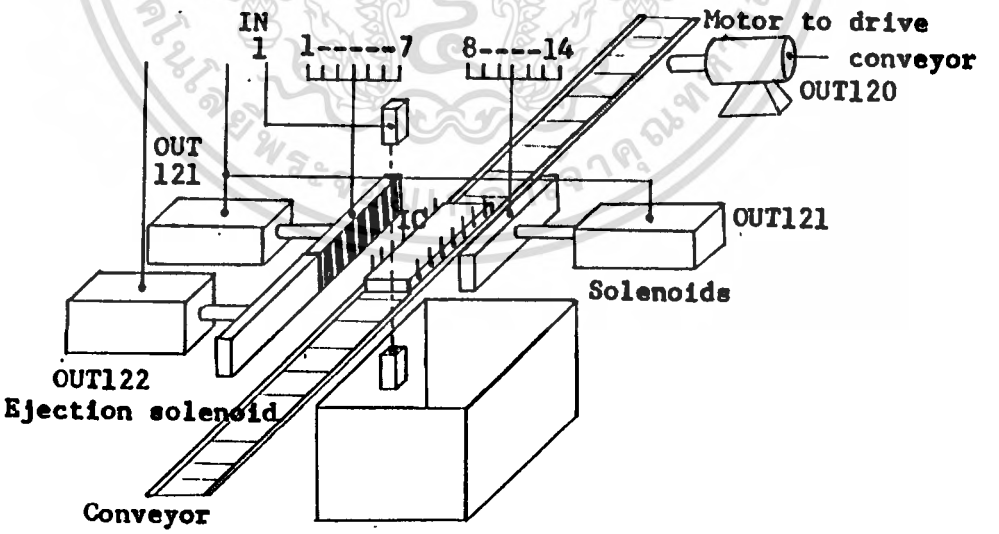
9.32 โปรแกรมการทำงานของกระบวนการเก็บเงินอัตโนมัติระบบทางด่วน

Step	Mnemonic	I/O Address
0	LDC	002
1	AND	240
2	OR	120
3	AND	001
4	STO	120
5	LD	004
6	STO	240
7	LD	002

8	STO	241
9	SET	005
10	LD	002
11	OR	121
12	AND	003
13	STO	121
14	AND	

9.4 ตัวอย่างที่ 4 กระบวนการตรวจสอบไอซีลอจิก

การนำพีแอลซีขนาด 1 บิท ไปใช้ในการตรวจสอบคุณภาพของไอซีลอจิกต่าง ๆ ว่าดีหรือเสียดังในรูปที่ 9.4

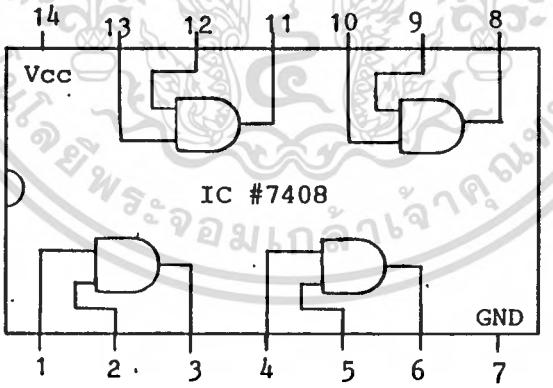


รูปที่ 9.4 ระบบการตรวจสอบไอซีลอจิก

จากรูปที่ 9.4 จะเป็นระบบของการตรวจสอบไอซีลอจิกว่าดีหรือเสีย เมื่อไอซีไหลมาตามสายพานจนถึงจุดที่จะตรวจ โดยจะใช้สวิทช์อินพุท 1 เป็นตัวตรวจว่าไอซีไหลมาถึงจุดที่จะตรวจหรือยัง ถ้าถึงแล้วสวิทช์อินพุท 1 จะมีสภาวะเป็น Low มอเตอร์ที่ขับสายพานก็จะหยุด

Solenoids ก็จะดันแผ่นที่ต่อกับขาไอซีเข้าไปติดกับตัวไอซี ซึ่งตอนนี้ขาของไอซีทุก ๆ ขาจะต่ออยู่กับอินพุทและเอาท์พุทของพีแอลซี พีแอลซีก็จะส่งเอาท์พุทและอ่านอินพุทเข้ามาตามโปรแกรมที่เขียนไว้และตรวจสอบดูว่าไอซีตัวนี้ดีหรือเสีย ถ้าตรวจสอบพบว่าเสียจะถูก Ejection Solenoids ดันทิ้งลงไปในกล่องสำหรับเก็บไอซีที่เสีย การทำงานแบบนี้จะต่อเนื่องไปเรื่อย ๆ

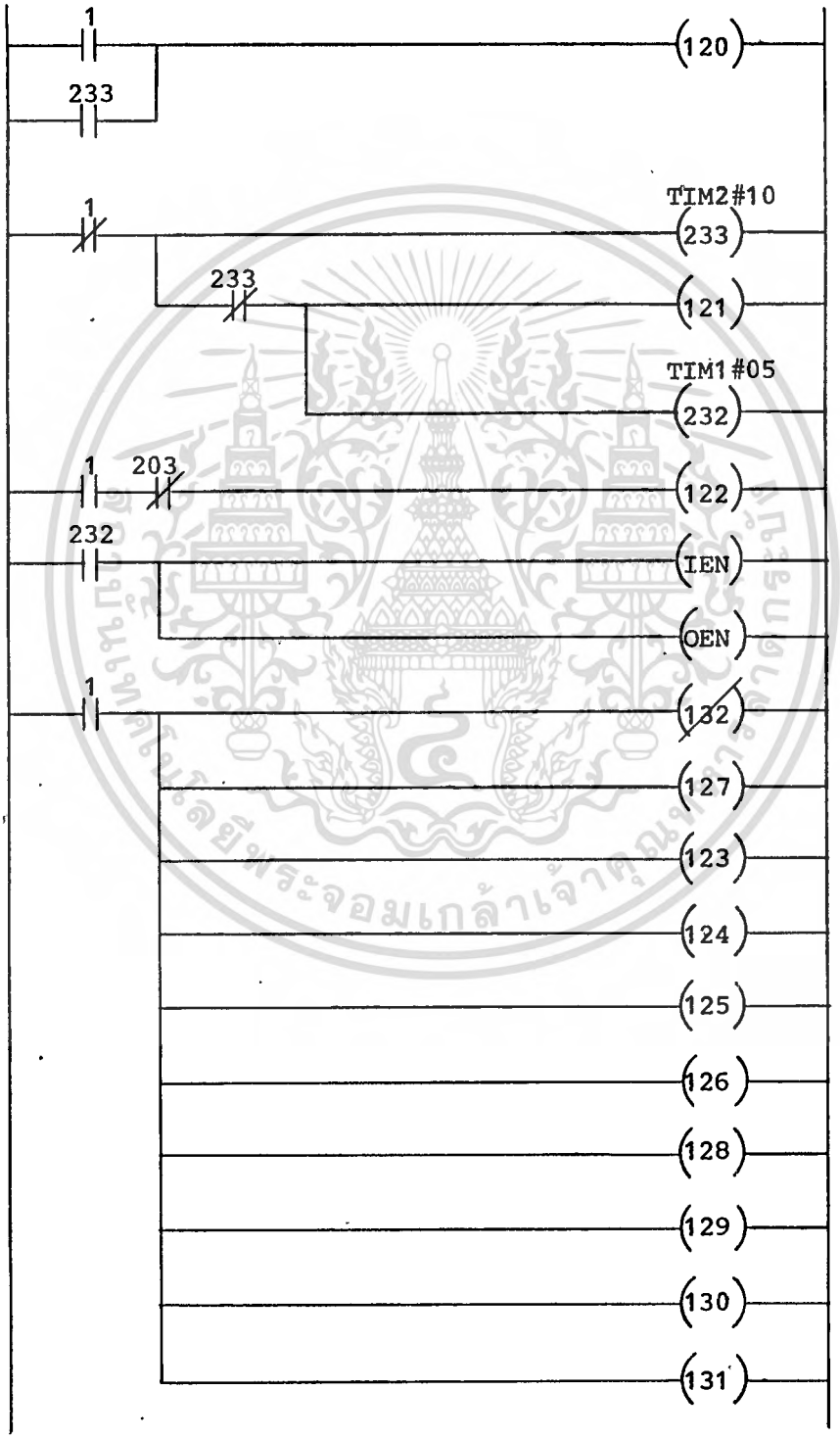
ในตัวอย่างที่ 9.4 นี้จะยกตัวอย่างการตรวจสอบไอซี TTL เบอร์ 7408 ซึ่งเป็น 2 อินพุท AND Gate ซึ่งตำแหน่งและหน้าที่ของขาต่าง ๆ เป็นดังในรูปที่ 9.41

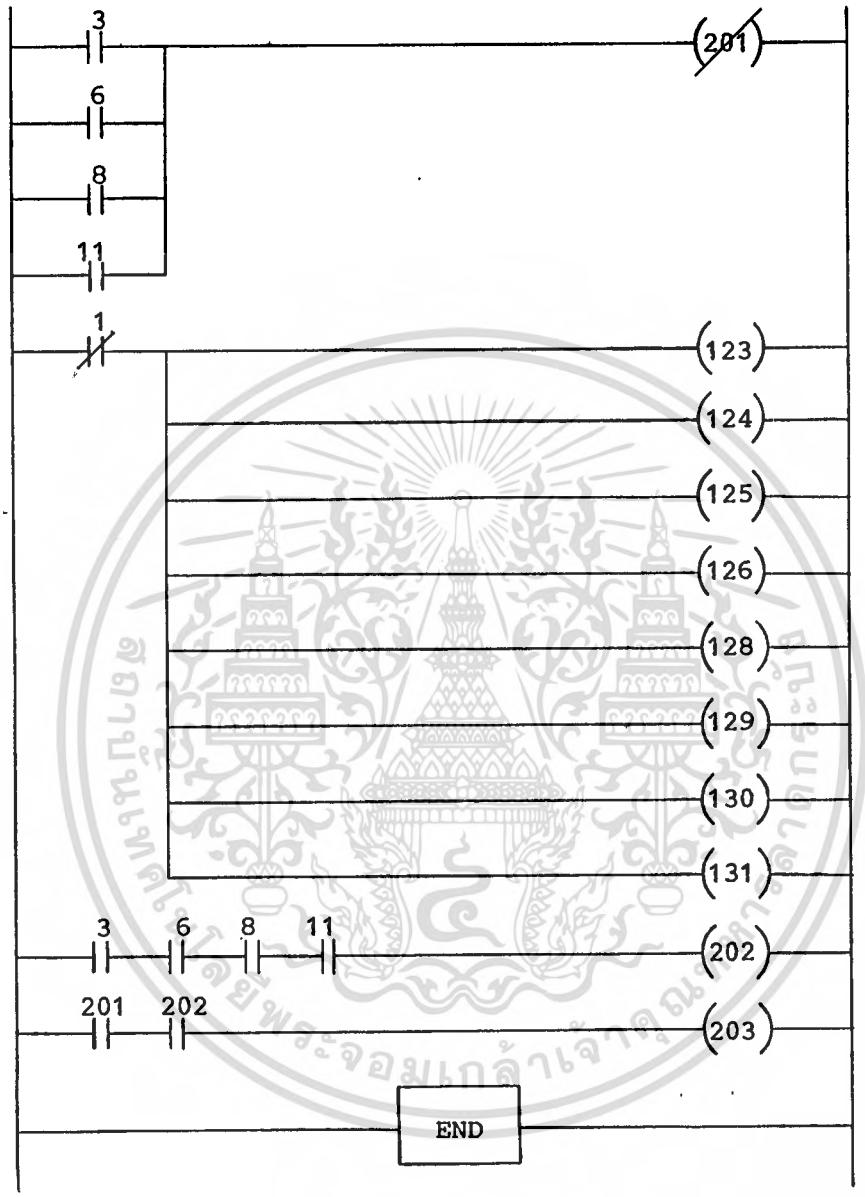


รูปที่ 9.41 AND Gate

การตรวจสอบไอซีเบอร์ 7408 ว่าดีหรือเสียมีลำดับขั้นการทำงานตามแลดเดอร์ไดอะแกรมที่ 9.41 และตามโปรแกรมที่ 9.42

9.41 แลดเดอร์ไดอะแกรมการทำงานของระบบตรวจสอบไอซีลอจิก





9.42 โปรแกรมการทำงานของระบบตรวจสอบไอซีลอจิก

Step	Mnimonic	I/O Address
0	LD	001
1	OR	233
2	STO	120
3	LDC	001
4	STO	233
5	SET	010
6	ANDC	233
7	STO	232
8	SET	005
9	STO	121
10	LD	001
11	ANDC	203
12	STO	122
13	IEN	232
14	OEN	232
15	LD	001
16	STOC	132
17	STO	127
18	STO	123
19	STO	124
20	STO	125
21	STO	126
22	STO	128

23	STO	129
24	STO	130
25	STO	131
26	LD	003
27	OR	006
28	OR	008
29	OR	011
30	STOC	201
31	LDC	001
32	STO	123
33	STO	124
34	STO	125
35	STO	126
36	STO	128
37	STO	129
38	STO	130
39	LD	003
40	AND	006
41	AND	008
42	AND	011
43	STO	202
44	LD	201
45	AND	202
46	STO	203
47	END	

สรุปผลการทดลองของกระบวนการผสมของเหลว

จากแลดเดอร์ไดอะแกรมจะเห็นได้ว่า การทำงานจะเป็นไปตามขั้นตอนได้นั้น จำเป็นจะต้องเขียนแลดเดอร์ไดอะแกรมการทำงานให้ครอบคลุมทั้งหมด เช่น การทำงานของเอาท์พุทใดเอาท์พุทหนึ่งนั้นจำเป็นต้องนำเอาเอาท์พุทอื่น ๆ มาเป็นเงื่อนไขด้วยเพราะว่าเมื่อการทำงานในคำสั่งหนึ่ง ๆ เสร็จแล้ว ในขณะนั้น เครื่องพีแอลซีอาจจะกำลังอ่านโปรแกรมช่วงท้าย ๆ อยู่ จึงเป็นไปได้ที่อาจมีการข้ามการทำงานบางส่วนไปทำงานในซีเควนซ์ตอนท้าย ๆ เลย เพราะฉะนั้นจึงจำเป็นต้องนำเอาท์พุทของซีเควนซ์ก่อนหน้ามาเป็นเงื่อนไขของซีเควนซ์ตอนท้าย ๆ จากการทดลองในโปรแกรมข้างบนนี้กับระบบจริงซึ่งให้ทำซีเควนซ์ไปเรื่อย ๆ เป็นเวลาหลาย ๆ วันนั้นไม่ปรากฏว่ามีการข้ามโปรแกรมเลย เพราะการเขียนโปรแกรมได้มีการป้องกันการข้ามซีเควนซ์ไว้ภายในโปรแกรมอยู่แล้ว ถ้าซีเควนซ์แรก ๆ ยังไม่มีการทำงาน ซีเควนซ์ต่อมาจะไม่มีโอกาสทำงานได้เลย

บทที่ 10
บทสรุป

พีแอลซีขนาด 1 บิทที่สร้างขึ้นมานี้ มีประสิทธิภาพดีไม่แพ้พีแอลซีทั่ว ๆ ไปที่มีขายในตลาด ในขณะที่ราคาต่ำกว่ามาก ๆ ซึ่งจะเห็นได้ว่าพีแอลซีที่มีขายในตลาดทั้งขนาดเล็กและขนาดกลางจะมีอินพุทและเอาท์พุทสูงสุดไม่เกิน 128 อินพุทเอาท์พุท และไม่สามารถสร้างขึ้นมาใช้เองภายในประเทศได้ ต้องสั่งซื้อเข้ามาจากต่างประเทศ แต่พีแอลซีขนาด 1 บิทนี้สามารถออกแบบดัดแปลงให้เหมาะกับงานได้ตามต้องการ เช่น ในกรณีที่ต้องการใช้ตัวตั้งเวลามากกว่าที่กำหนดไว้ก็สามารถดัดแปลงได้ โดยการเพิ่มส่วนของตัวตั้งเวลาเข้าไปทางด้านเอาท์พุทตรงตำแหน่งที่ไม่มีเอาท์พุทจริงอยู่ จากจำนวนตำแหน่งของอินพุทเอาท์-พุททั้งหมด 4096 ตำแหน่งนั้น อาจออกแบบตำแหน่งอินพุทเอาท์พุทใหม่ได้หลายรูปแบบตามต้องการ เช่น อาจแบ่งเป็นอินพุท 2,000 ตำแหน่งและเอาท์พุท 1,000 ตำแหน่ง ส่วนตำแหน่งที่เหลืออีก 1096 ตำแหน่งอาจแบ่งเป็นรีจิสเตอร์ 300 ตำแหน่ง และเป็นตัวตั้ง-เวลาอีก 500 ตำแหน่ง ที่เหลืออีก 296 ตำแหน่งเป็นตัวนับ จะเห็นได้ว่าเครื่องทั่ว ๆ ไปไม่สามารถดัดแปลงส่วนต่าง ๆ ได้เลยเพราะออกแบบมาตายตัว ในบางครั้งนั้นตัวตั้ง-เวลา หรือตัวนับ อาจจะไม่มีการใช้งานเลยก็ได้ในขณะที่พีแอลซีขนาด 1 บิท นี้สามารถเปลี่ยนแปลงอินพุทเอาท์พุท รีจิสเตอร์ ตัวตั้งเวลาและตัวนับได้ตามที่ต้องการ และสามารถสร้างขึ้นมาใช้งานเองภายในโรงงานได้ จึงเป็นการลดต้นทุนการผลิต และสร้างขึ้นมาใช้ให้เหมาะสมกับกระบวนการควบคุมได้ตามต้องการ

ในการออกแบบพีแอลซีขนาด 1 บิทนี้จะแบ่งส่วนของอินพุท เอาท์พุท รีจิสเตอร์ ตัวตั้งเวลา และตัวนับไว้เป็นส่วน ๆ (Card) ถ้าต้องการเพิ่มส่วนใดส่วนหนึ่งมากขึ้นก็สามารถนำส่วนนั้น ๆ มาเสียบเพิ่มได้เลยตามต้องการ

ความเที่ยงตรงในการทำงานระหว่างอินพุทกับเอาท์พุท คือเมื่อสภาวะของอินพุทเกิดขึ้นตามเงื่อนไขที่กำหนดไว้ เอาท์พุทจะถูกหน่วงเวลาไปสูงสุดไม่เกิน 1 มิลิเซคกันด์ โดยโปรแกรมที่ใช้ควบคุมกระบวนการ มีคำสั่ง ประมาณ 2,000 คำสั่ง แต่ถ้าโปรแกรมที่

ใช้สั้น เวลาการเปลี่ยนแปลงของเอาท์พุทจากเงื่อนไขของอินพุทสูงสุดจะน้อยกว่า 1 มิล-ลิเซคกันด์ และความผิดพลาดของตัวตั้งเวลาสูงสุดจะไม่เกิน 0.1 วินาที

กิตติกรรมประกาศ

ขอขอบพระคุณอย่างสูงต่ออาจารย์ ดนุตร วิเศษกุล ในฐานะอาจารย์ที่ปรึกษา ในการให้คำปรึกษาและแนะนำเรื่องการเรียน ตลอดจนการทำวิทยานิพนธ์ตั้งแต่ต้นจนจบการศึกษา พร้อมกันนี้ใคร่ขอขอบพระคุณผู้ช่วยศาสตราจารย์ กิตติ ตีรเศรษฐ ในการให้คำแนะนำที่มีประโยชน์หลาย ๆ อย่างเกี่ยวกับระบบควบคุมทางอุตสาหกรรม

เอกสารอ้างอิง

1. Vern Gregory and Brian Dellande,"MC14500B Industrial Control Unit Handbook", Motorola Semiconductor Product Inc., 1977.
2. C.T. Jones and L.A. Bryan, "PROGRAMABLE CONTROLLERS Concepts and Applications", International Programable Controls Inc., 1983.
3. "Microcontroller Handbook", Intel, 1983.
4. "CMOS Databook", National Semiconductor, 1983.
5. "OPTOELECTRONICS DEVICE DATA", Motorola Inc.,1983.