ชุดฝึกไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุม

MICROPROCESSOR 1 BIT FOR THE CONTROLED WORK

นายกะรอซะ มะนุง

นายบุญส่ง บุญล้อม

นายสำเริง บุญมาก

วิทยานิพนธ์สำหรับปริญญาอุตสาหกรรมศาสตร์บัณฑิต

สาขาเทคโนโลยีอิเล็กทรอนิกส์

ภาควิชาเทคนิคอุตสาหกรรม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

ปีการศึกษา 2532

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ปีการศึกษา 2532

ชุดฝึกไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุม

โดย

นายกะรอชะ มะนูง

นายบุญส่ง บุญล้อม

นายสำเริง บุญมาก

อาจารย์ที่ปรึกษา

ผศ. นิกร สุขุตมตันติ

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

026900

ปริญญานิพนธ์ปีการศึกษา 2532

ภาควิชา เทคนิคอุตสาหกรรม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง

เรื่อง ชุดฝึกไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุม

ผู้จัดทำ

1. นายกะรอชะ มะบุง 29.6601
2. นายบุญส่ง บุญล้อม 29.6609
3. นายสำเริง บุญมาก 29.6619

.....อาจารย์ที่ปรึกษา

(ผศ. นิกร สุขุคมทันติ)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชุดฝึกไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุม

นายกะรอชะ มะนูง
นายบุญส่ง บุญล้อม
นายสำเริง บุญมาก
ผศ.นิกร สุขุตมตันติ อาจารย์ที่ปรึกษา
ปีการศึกษา 2532

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้ แสดงถึงไอซี MC14500 ไมโครโปรเซสเซอร์ (Microprocessor) 1 บิท สำหรับงานควบคุม มีคำสั่งที่จะควบคุมระบบการทำงาน 16 คำสั่ง มันสามารถที่จะทำงานได้จากออสซิลเลเตอร์ (Oscillator) ที่มีความถี่ตั้งแต่ไฟตรงจนถึง 1 MHz มันจะรับรหัสคำสั่งจากหน่วยความจำ (Memory) 4 บิท ซึ่งก็มีคำสั่งได้เพียง 16 คำสั่ง คำสั่งเหล่านี้จะได้รับการอ่านเข้ามายังรีจิสเตอร์รับคำสั่ง (Instruction Register) แล้วตีความหมายเพื่อกระทำตามและบัสข้อมูล (Data Bus) จะมีเพียงบิทเดียว ซึ่งนำข้อมูลมาประมวลผล ระบบไมโครโปรเซสเซอร์ 1 บิท จึงเหมาะที่จะใช้กับงานทั่ว ๆ ไป โดยเฉพาะระบบควบคุมอัตโนมัติในงานอุตสาหกรรม จุดมุ่งหมายของชุดฝึกไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุม เพื่อที่จะได้ใช้เป็นชุดฝึกโปรแกรมที่สลับซับซ้อนขึ้น หรือจะใช้เป็นชุดทดลองโปรแกรมการทำงานในระบบ PLC (Programmable Logic Controller) ตามหลักการแทนวงจรรีเลย์ที่นิยมเขียนเป็นขั้นบันไดนั่นได เมื่อทดลองได้ดีแล้วจึงนำมาสร้างเป็นระบบโดยให้โปรแกรมอยู่ใน ROM และชุดฝึกไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุม นับว่าเป็นประโยชน์โดยตรงกับนักศึกษา จะได้พัฒนาเพิ่มเติมเพื่อออกแบบสำหรับการควบคุมอุปกรณ์บางอย่างโดยเฉพาะต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MICROPROCESSOR 1 BIT FOR THE CONTROLED WORK

Mr. Karasa Manung

Mr. Boonsong Boonlom

Mr. Somrerng Boonmak

Assistant Professor Nikorn Sukutamatanti Advisor

1989

Abstact

This thesis shows IC MC 14500 Microprocessor 1 bit for the controled work. Instruction will control the system of working to have sixteen instructions. IC MC 14500 can work by using frequency from Oscillator to have frequency since the direct current until 1 MHz and will receive the instruction code from memory until 4 bit which has the instruction only 16 instructions. These instructions will received reading into Instruction Register and translate meaning for activity follows and Data Bus will have 1 bit which data is brought to come processing. The system of Microprocessor 1 bit popularizes to use the work,about the system of the industrial control spreadly in the present. Purpose of Microprocessor 1 bit for the controled work for bring to use be program to complex higher and will use be the program to work in the system of PLC (programmable Logic Controller) follow the principle replaces Relay and Digital to popular write is step when the experiment is good to would to would take to build be the system by gives program into ROM. Microprocessor 1 bit for the controled work be the direct advantage to the students. That will improve for design controls equiment soneting partly continue.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

		หน้า
บทที่ 1	บทนำ	
	1.1 ไมโครคอนโทรลเลอร์	1
	1.2 ไมโครโปรเซสเซอร์ 1 บิท	3
บทที่ 2	ทฤษฎีหรือหลักการ	
	2.1 การอินเตอร์เฟสกับแป้นกดข้อมูล	10
	2.1.1 เบานส์	10
	2.1.2 ROLLOVER	12
	2.1.3 แป้นกดข้อมูลชนิดไม่เข้ารหัส	13
	2.1.4 แป้นกดข้อมูลชนิดเข้ารหัส	15
	2.1.5 ตัวเข้ารหัสแป้นกดข้อมูล	15
	2.1.6 แป้นกดข้อมูลรหัส ASCII	19
	2.2 การอินเตอร์เฟสเพื่อแสดงผลด้วยแอลอีดี	21
	2.2.1 แอลอีดี แบบที่เปล่งแสงจุดเดียว	21
	2.2.2 แอลอีดี แบบเปล่งแสงเจ็ดส่วน	22
	2.2.3 แอลอีดี แบบเปล่งแสงหลายจุด	24
	2.3 การใช้งานไอซี 555	27
	2.3.1 การใช้เป็นวงจรโมโนสเตเบิล	27
	2.3.1.1 วงจรอะสเตเบิล	28
	2.3.1.2 วงจรหาพัลส์ที่ขาดหายไป	29
	2.3.2 การใช้เป็นวงจรตั้งเวลาสองตัวที่แยกจากกัน	30
	2.3.2.1 การตั้งเวลาเป็นลำดับ	30
	2.3.2.2 คีย์ออสซิลเลเตอร์	30
	2.3.2.3 การหารความถี่	31
	2.3.2.4 วงจรกำเนิดสัญญาณความถี่คงที่แต่ปรับความกว้างของพัลส์	32
	2.3.2.5 การมอดูเลททางความกว้างของพัลส์	33
	2.3.2.6 การมอดูเลททางตำแหน่งของพัลส์	34

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

2.3.3 ใช้ไอซี 556 ในวงจรวัดความเร็วของรถยนต์ 35

2.3.3.1 การทำงานของไอซีที่ไม่ใช่เป็นตัวตั้งเวลา 35

2.4 ไอซีซีมอส 36

2.4.1 ลักษณะของมอสฟิลด์เอฟเฟคทรานซิสเตอร์ 36

2.4.2 ลักษณะของวงจรซีมอสที่ใช้เป็นเกท 41

2.4.3 ชนิดของไอซีซีมอส 44

2.4.4 การใช้งานไอซีซีมอส 44

2.4.5 บทสรุปไอซีซีมอส 45

2.5 โปรแกรมเคาน์เตอร์ 46

2.5.1 วงจรนับชนิดริปเปิ้ล 47

2.5.2 วงจรหาร 48

2.5.3 วงจรนับ-หารด้วยเอ็น 49

2.5.4 การถอดรหัสวงจรนับ 51

2.5.5 วงจรนับเดินหน้า-ถอยหลัง 51

2.5.6 วงจรนับชนิดวงแหวน 52

2.5.7 วงจรนับชนิดซิงโครนัส 53

2.5.8 สรุปวงจรนับ-หาร 53

2.6 หน่วยความจำ 54

2.6.1 ตระกูล ROM 56

2.6.2.1 Mask - Programmed ROM 57

2.6.2.2 Programmable ROM (PROM) 58

2.6.2.3 Reprogrammable ROM 60

2.6.3 ตระกูล RAM 61

2.6.5 ชนิดของ RAM 62

2.6.5.1 RAM ชนิดสเตติค 62

2.6.5.2 RAM ชนิดไดนามิค 63

2.6.5.3 การรีเฟรช์ RAM ชนิดไดนามิค 65

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

	หน้า
2.6.6 ระบบหน่วยความจำ	68
2.7 โครงสร้างของอินพุท-เอาท์พุท	71
2.7.1 อินพุทด้วยตัวเลือกข้อมูล MC 14512	71
2.7.2 เอาท์พุทด้วยวงจรแลทช์ MC 14590	71
2.7.3 I/Oพอร์ท	72
2.7.3.1 I/O พอร์ท แบบขนาน	73
2.7.3.2 I/O พอร์ท แบบอนุกรม	76
2.7.4 สัญญาณควบคุมการทำงานของพอร์ทและการกำหนดแอดเดรส	76
2.7.4.1 Isolated I/O	76
2.7.4.2 Memery mapped I/O	77
2.7.4.3 การสร้างสัญญาณเลือกอุปกรณ์โดยใช้ตัวถอดรหัส	78
2.7.5 วิธีโพลลิ่งหรือโปรแกรม I/O	79
บทที่ 3 การคำนวณและการสร้าง	
3.1 ระบบไมโครคอมพิวเตอร์สำหรับการพัฒนา	82
3.2 โครงสร้างของชุดพัฒนาระบบ	82
3.3 วงจรภาคแสดงผล	84
3.4 วงจรรับคีย์บอร์ดและแสดงค่าที่กดคีย์	86
3.5 โปรแกรมเคาวน์เตอร์กับหน่วยความจำ	88
3.6 วงจรไมโครโปรเซสเซอร์	91
3.7 วงจรอินพุท-เอาท์พุทพอร์ท	92
3.8 เมื่อต่อเป็นระบบ	94
3.9 นำระบบมาสร้างเป็นโปรแกรมเมเบิ้ลคอนโทรเลอร์ (PLC)	97
บทที่ 4 การทดลอง และผลการทดลอง	
4.1 โครงสร้างระบบพื้นฐาน	98
4.2 วงจรไมโครโปรเซสเซอร์สำหรับการทดลอง	101
4.3 การทดลองคำสั่งของไมโครโปรเซสเซอร์	102
4.4 การทดลองทำ AND – OR – XNOR	104
4.5 จะทำไอซี MC 145000ให้เป็นเกทธรรมดา	104

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

4.6 การทำเป็นวงจรเทคนิคต่าง ๆ 105

4.7 การใช้คำสั่ง LDC 107

4.8 การใช้คำสั่ง ANDC 108

4.9 การใช้ OR และ ORC 109

4.10 การใช้หน่วยความจำเก็บข้อมูลชั่วคราว 109

4.11 คำสั่ง XNOR 111

4.12 คำสั่ง STOC 111

4.13 การใช้คำสั่ง IEN และ OEN 113

4.14 คำสั่ง OEN 114

4.15 การทดสอบโปรแกรม 116

4.16 การโปรแกรมเกี่ยวกับการตั้งเวลา 116

4.17 การกำหนดช่วงเวลาแบบดิจิตอลที่โปรแกรมได้ 119

4.18 การเขียนโปรแกรมแบบ IF.....THEN 120

4.19 การเขียนโปรแกรมแบบ WHITE 123

บทที่ 5 บทวิจารณ์และสรุป 124

ภาคผนวก

DATA SHEET MC 145000 B 127

DATA SHEET MC 14495 133

DATA SHEET MC 14175 B 136

DATA SHEET MC 14419 140

DATA SHEET SCL 45168 144

DATA SHEET SCL 4071B, SCL 4072B, SCL 4075B 148

DATA SHEET SCL 4049 UB, SCL 4050B 151

DATA SHEET SCL 4013 B 153

DATA SHEET SCL 4011B, SCL 4012B, SCL 4023B 155

DATA SHEET SCL 4011 UB 157

DATA SHEET CDN 6116 A 160

DATA SHEET HCC/HCF 4099 B 165

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

DATA SHEET HCC/HCF 4503 B 172

กิตติกรรมประกาศ 179

หนังสืออ้างอิง 180



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

	หน้า
บทที่ 1 บทนำ	
ระบบไมโครโปรเซสเซอร์	2
การใช้งานไมโครโปรเซสเซอร์ในขบวนการอุตสาหกรรม	3
การจัดขาไอซี MC14500 ซึ่งเป็นไอซี 1 บิท ไมโครโปรเซสเซอร์	4
โครงสร้างของไมโครโปรเซสเซอร์	4
บทที่ 2 ทฤษฎีหรือหลักการ	
โครงสร้างภายในของ MC 14500	8
โครงสร้างของระบบที่จะต่อออกไปใช้งาน	9
เบกนส์ที่เกิดจากการกดปุ่ม	11
วงจรฟลิบฟล็อบทำหน้าที่ดีเบานส์	11
ผังงานโปรแกรมดีเบานส์แป้นกดข้อมูล	12
HEX KEYBOARD	13
WALKING ONES KEYBOARD DECODE	14
แป้นกดข้อมูลชนิด 64 ปุ่ม	15
STATIC ENCODER	16
SCANNING ENCODER	17
ROM และแลทช์	18
NEC KEYBOARD ENCODER	18
8279 KEYBOARD DISPLAY CONTROLLER	19
ASCII Serial Keyboard Interface	20
วงจรขับแอลอีดีหนึ่งตัว	21
LED Interface	22
Seven Segment LED	23
Seven Segment Charactors	23
Multiplexing LED's (SCAN DIGIT AND DATA)	24
Drivers for scanning	24
7 x 5 Dot matrix LED	25

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

	หน้า
Dot matrix ROM	26
Dot matrix charactors	27
วงจรโมโนสเตเบิ้ล	27
วงจรอะสเตเบิ้ลมัลติไวเบรเตอร์	28
การหาความถี่จากกราฟ	29
วงจรตั้งเวลาสองตัวที่แยกจากกัน	30
คีย์ออสซิลเลเตอร์	31
สัญญาณของการหารความถี่	32
วงจรโมโนสเตเบิ้ล	33
วงจรมอดูเลทหาความกว้างของพัลส์	34
รูปลักษณะของสัญญาณที่สำคัญ	34
การนำไอซีมามอดูเลทเข้าที่ขั้วแรงดันควบคุม	35
วงจรวัดความเร็วรอบ	35
วงจรป้องกันชไมยที่ตั้งเวลาได้	36
โครงสร้างของเอ็นมอส	37
วงจรมอสเฟทที่ใช้เป็นวงจรอินเวอร์เตอร์	39
โครงสร้างของมอสเฟทชนิด พี แชนเนล	39
สัญญลักษณ์ของพีมอส	40
วงจรอินเวอร์เตอร์ที่ใช้ซีมอส	41
โครงสร้างของซีมอส NAND เกท	41
วงจรซีมอส NOR เกท	42
วงจร TG และลักษณะสมมูลย์ด้วยรีเลย์	43
สัญญลักษณ์ของ Transmission gate	43
ซีมอสในกลุ่มเอและกลุ่มบี	44
วงจรนับ 4040 ที่ใช้เป็นโปรแกรมเคาวน์เตอร์	46
โครงสร้างของโปรแกรมเคาน์เตอร์ที่ใช้เอาท์พุทถึง 12 บิท	46
วงจรนับชนิดริปเปิ้ล	47

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้วนการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

	หน้า
รูปสัญญาณคลื่นที่เอาท์พุทของวงจรนับชนิดริปเปิ้ล	48
การใช้ฟลิบฟลอบหารความถี่	48
ใช้ฟลิบฟล้อบ 4 ตัว สามารถหารความถี่ได้ด้วย 16	49
วงจรนับ-หาร MOD – 6	50
สัญญาณการทำงานของวงจร MOD – 6	50
วงจรนับเมื่อนับถึง 1100 แล้วมันจะหยุดนับ	50
วงจรถอดรหัสวงจรนับไบนารี่ 2 บิท	51
วงจรถอดรหัสจากตัวเลขไบนารี่เป็นเอาท์พุทภาคแสดง 7 ส่วน	51
โครงสร้างของวงจรนับเดินหน้า-ถอยหลังชนิด 4 บิท	52
วงจรนับชนิดซิงโครนัส	53
การจัดขาของ EPROM	54
โครงสร้างข้อมูลใน ROM	54
การใช้คำสั่งยาว 2 ไบท์	55
โครงสร้างภายในของ ROM	57
วงจรภายในของภาควงจรเข้ารหัสของ ROM	58
วงจรภายในบางส่วนของ PROM เบอร์ 82 เอส 23/123	59
โครงสร้างของ MOSFET	60
เซลพื้นฐานของ RAM ชนิดสเตติค	62
เซลหน่วยความจำของ RAM ชนิดไดนามิคแบบทรานซิสเตอร์ 3 ตัว	64
วงจรควบคุมการรีเฟรชแบบซิงโครนัสที่จังหวะ T_4 ของแมชีลไซเคิล	67
วงจรควบคุมการรีเฟรชในขณะใช้งาน ROM	68
ระบบหน่วยความจำขนาด 8 บิท ที่ประกอบจาก EPROM(2708)	70
การจัดขาของไอซี MC 14512	71
การจัดขาและแผนผังภายในของ MC 14599	72
โครงสร้างภายในของ 8212 อินพุทและเอ้าท์พุท พอร์ท	74
พอร์ทเบอร์ 8255	75
การถอดรหัสแอดเดรสสำหรับ 8255	77

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

		หน้า
	Memory mapped I/O	78
	การสร้างสัญญาณเพื่อเลือกอุปกรณ์ 16 ตัว	79
	ระบบวงจรที่ใช้วิธีโพลลิ่ง	80
	แผนภูมิแสดงวิธีการโพลลิ่ง	81
บทที่ 3	การคำนวณและการสร้าง	
	โครงสร้างของชุดพัฒนาระบบ	83
	ไดอะแกรมรายละเอียดของไอซีเบอร์ 4495	84
	วงจรของส่วนแสดงผลแอดเดรสและข้อมูลในหน่วยความจำ	85
	แสดงรายละเอียดของไอซีเบอร์ 4419	86
	วงจรการรับอินพุทจากคีย์บอร์ด	87
	แสดงรายละเอียดของไอซีเบอร์ 4175 และ 4503	88
	แสดงรายละเอียดของไอซีเบอร์ 4516	89
	วงจรเชื่อมโยงหน่วยความจำกับโปรแกรมเคาวน์เตอร์	90
	การต่อวงจรของแรม เบอร์ 6116 เพื่อใช้แทนเบอร์ 2112	91
	ส่วนของไมโครโปรเซสเซอร์และวงจรประกอบ	92
	รายละเอียดของไอซีส่วนอินพุท-เอาท์พุทเบอร์ 4512 และ 4099	93
	วงจรอินพุท-เอาท์พุทที่สมบูรณ์	94
	วงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์ MC14500สำหรับใช้งาน	95
	ค่าความต้านทานของสัญญาณนาฬิกาแปรค่ากับ R_x	95
	แสดงค่าตำแหน่งอินพุท-เอาท์พุทของระบบ	96
บทที่ 4	การทดลองและผลการทดลอง	
	ระบบไมโครโปรเซสเซอร์	98
	ระบบไมโครคอมพิวเตอร์ที่ใช้ MC 14500 เป็นซีพียู	99
	วงจรพื้นฐานไมโครโปรเซสเซอร์ไอซี MC 14500	101
	โครงสร้างของไมโครโปรเซสเซอร์ MC 14500	102
	การต่อไอซี MC 14500 ให้เป็นวงจรทดลองของฮาร์ดแวร์ลอจิก	105
	วงจรและโปรแกรมที่เขียนขึ้นแสดงการทำงานอย่างง่าย ๆ	107

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

วงจรตัวอย่างของการใช้คำสั่ง ANDC 108

การใช้ลอจิก OR และตัวอย่างโปรแกรม 109

วงจรและโปรแกรมของการใช้คำสั่ง XNOR 111

วงจรและโปรแกรมของคำสั่ง STOC เมื่อมีอินเวอร์เตอร์ทางเอาท์พุท 112

ลอจิกที่มีทางเลือก 114

วงจรตั้งเวลาทางฮาร์ดแวร์และวงจรซีมอสโมโนสเตเบิ้ล 117

ตัวอย่างของแผนผังการตั้งเวลา turn on-turn off 118

ตัวอย่างการต่อวงจรนับ 119

หลักการการใช้คำสั่ง OEN 121

แผนผังการใช้คำสั่ง OEN 122

แผนผังและโปรแกรมของตัวอย่างการใช้ IF...then 122

แผนผังการเขียนโปรแกรมแบบ WHITE 123

บทที่ 5 บทวิจารณ์และสรุป

ระบบไมโครคอมพิวเตอร์ไอซี MC 14500 125

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

	หน้า
บทที่ 1 บทนำ	
ตารางแสดง คำสั่งต่าง ๆ ทั้ง 16 คำสั่งของ MC 14500	5
บทที่ 2 ทฤษฎีหรือหรักการ	
ตารางแสดงคำสั่งและรหัสจำนวน 16 คำสั่งของ MC 14500	9
ตารางรหัส ASCII	20
ตารางแสดงการทำงานของ MC 14512	71
ตารางแสดง การเลือกวิธีการทำงานของ 8255	75
บทที่ 4 การทดลองและผลการทดลอง	
แสดงตารางรายละเอียดของ MC 14500	100
บทที่ 5 บทสรุปและวิจารณ์	
ตาราง ชุดคำสั่งทั้ง 16 คำสั่งของไอซี MC 14500	124

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

บทนำ

1. หลักการพื้นฐานเกี่ยวกับไมโครโปรเซสเซอร์

1.1 ไมโครคอนโทรลเลอร์

ไมโครโปรเซสเซอร์เป็นไอซีที่ใช้ในการควบคุมระบบ และเป็นหัวใจของระบบไมโครโปรเซสเซอร์ก็คือ หน่วยประมวลผลกลางของคอมพิวเตอร์นั่นเอง ไอซีไมโครโปรเซสเซอร์มีขนาดจำนวนขาได้ตั้งแต่ 16 ขา จนถึงมากกว่า 100 ขา ไอซีไมโครโปรเซสเซอร์แต่ละตัวได้ออกแบบมาเพื่อจุดมุ่งหมายของการใช้งานต่างกัน เช่น ไมโครโปรเซสเซอร์ที่มีขนาดจำนวนบิทกว้าง ๆ เช่น 16 บิท 32 บิท มักจะเน้นการใช้งานทางด้านการคำนวณและการประมวลผลข้อมูล แต่สำหรับไมโครโปรเซสเซอร์ที่จะเน้นนี้เป็นเพียงไมโครโปรเซสเซอร์บิทเดียว ซึ่งจะมีจุดเน้นการใช้งานทางการควบคุมในอุตสาหกรรม ซึ่งก็เป็นการทำงานทางลอจิกมากกว่าทางคณิตศาสตร์ ไมโครโปรเซสเซอร์ตัวนี้ จึงมีชื่อว่า "ไมโครคอนโทรลเลอร์"

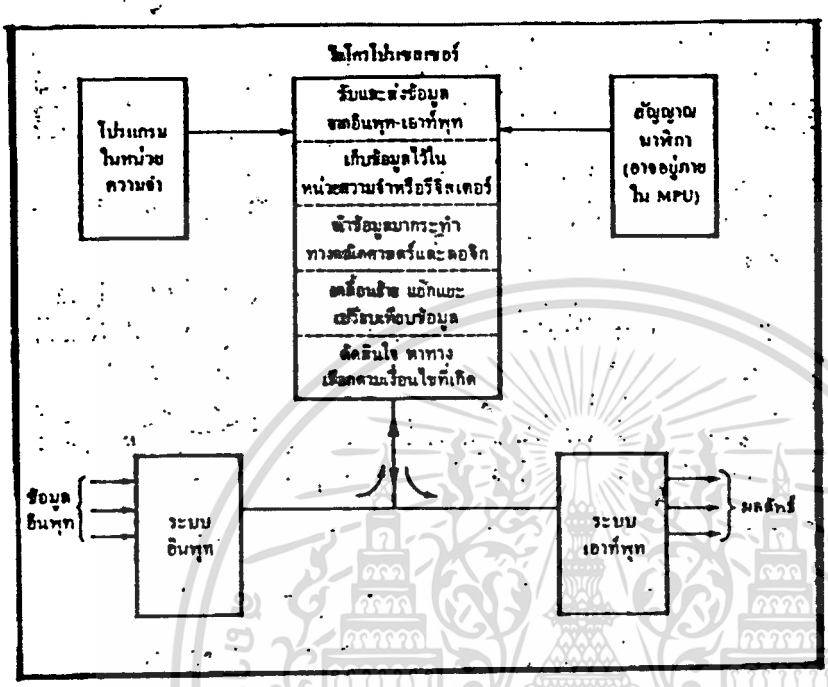
การทำงานของไมโครโปรเซสเซอร์ ส่วนใหญ่จะต้องทำงานภายใต้ของคำสั่งที่มีผู้โปรแกรมแล้วนำเอามาเก็บไว้ในหน่วยความจำ หลักการนี้เป็นหลักการที่ยึดถือมาตั้งแต่ครั้งเริ่มต้นสร้างดิจิตอล-คอมพิวเตอร์ โดยเป็นแนวความคิดมาจาก "ฟอน นอยมานน์" ซึ่งสรุปว่าให้นำเอาคำสั่งและข้อมูลที่จะประมวลผล เก็บรวมกันไว้ในหน่วยความจำแล้วให้หน่วยประมวลผลกลางเรียกมาตีความหมายและกระทำตาม

โครงสร้างของระบบไมโครโปรเซสเซอร์ที่ใช้งานในระบบควบคุมที่จะกล่าวถึงในที่นี้ เขียนเป็นแผนผังของระบบได้ดังรูป ซึ่งหน่วยหลักของระบบคือ ไมโครโปรเซสเซอร์ (MPU : Microprocrssor Unit) หน่วยนี้จะนำคำสั่งจากหน่วยความจำมาตีความหมายและกระทำตาม จังหวะการทำงานของMPU นี้ จะได้รับการกำหนดจังหวะด้วยสัญญาณนาฬิกา

ระบบไมโครโปรเซสเซอร์ที่กล่าวถึงนี้จึงเหมาะที่จะใช้ในงานทั่วไป โดยเฉพาะระบบควบคุมอัตโนมัติในอุตสาหกรรมไมโครโปรเซสเซอร์จะรับข้อมูลจากอินพุท ซึ่งอาจจะมาจากระบบรับความรู้สึกหรือเซนเซอร์อื่นใดก็ได้ ข้อมูลจะรับเข้ามายังหน่วยไมโครโปรเซสเซอร์โดยจะเก็บไว้ในหน่วยความจำ ในรีจิสเตอร์ หรือนำมาประมวลผลให้เกิดการกระทำทางลอจิกทางคณิตศาสตร์ การเปรียบเทียบ แล้วส่งผลลัพธ์ไปไว้ในรีจิสเตอร์ในหน่วยความจำหรือส่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต่อไปยังระบบเอาท์พุทก็ได้ ส่วนของผลลัพธ์ที่ได้จากระบบอาจจะเป็นวงจรควบคุมรีเลย์ วงจรควบคุมการปิดเปิดอุปกรณ์อื่น ๆ หรือแม้แต่เป็นวงจรอิเล็กทรอนิกส์เชื่อมต่ออื่น ๆ อีก



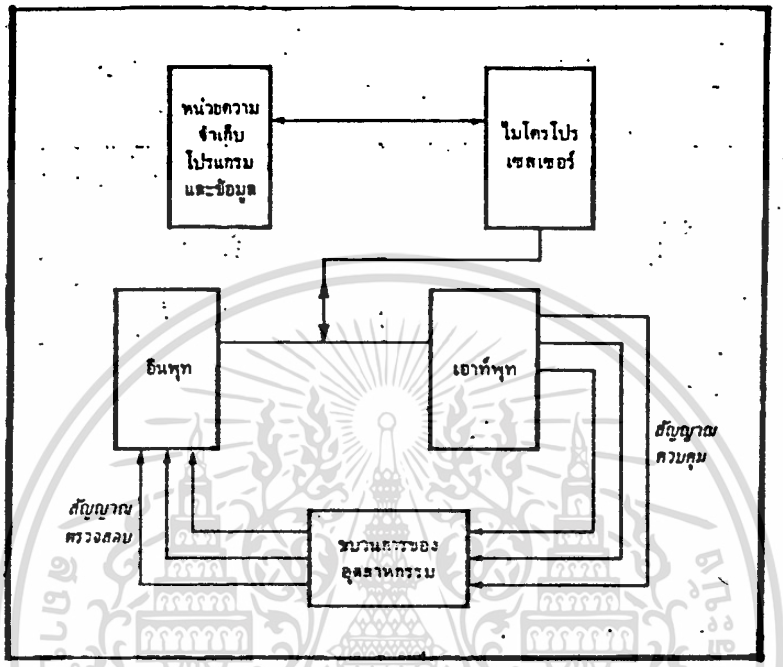
รูปแสดง ระบบไมโครโปรเซสเซอร์

จุดเด่นของการใช้ไมโครโปรเซสเซอร์ในกรณีนี้ที่เห็นได้ชัดคือ ไม่ว่าจะควบคุมหรือเชื่อมต่อกับอินพุท-เอาท์พุทอย่างไร ส่วนของไมโครโปรเซสเซอร์และหน่วยความจำก็ยังคงเหมือนเดิม ไมโครโปรเซสเซอร์จึงทำให้ระบบทั้งหมดมีความคล่องตัวและสามารถดัดแปลงแก้ไขขบวนการการควบคุมได้ง่าย เพราะเป็นการแก้ไขเฉพาะในส่วนของโปรแกรมซึ่งเป็นซอฟท์แวร์เท่านั้น เป็นผลให้ต้นทุนของระบบถูกลงไปได้มาก

อนึ่งระบบการทำงานของวงจรทั้งหมดมีผลจากการเขียนโปรแกรมในหน่วยความจำอยู่มาก ดังนั้นในระบบที่มีความยุ่งยากมาก ๆ จะเป็นผลที่เกี่ยวข้องกับโปรแกรม แต่วงจรไมโครโปรเซสเซอร์ยังคงเหมือนเดิม

การใช้ระบบไมโครโปรเซสเซอร์นี้ยังสามารถใช้ในระบบขบวนการทางอุตสาหกรรมได้มาก และสามารถตั้งระบบให้เป็นระบบแบบป้อนกลับได้โดยง่าย ดังรูป เป็นตัวอย่างการใช้ไมโครโปรเซสเซอร์ควบคุมขบวนการของอุตสาหกรรมต่าง ๆ ในระบบอัตโนมัติ เช่น ควบคุมความร้อน ความเย็น ความดัน ควบคุมระดับของเหลว ควบคุมอัตราการไหล หรือแม้แต่สร้างระบบให้ทำงานตามลำดับ ที่เรียกว่า programable sequence controller

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



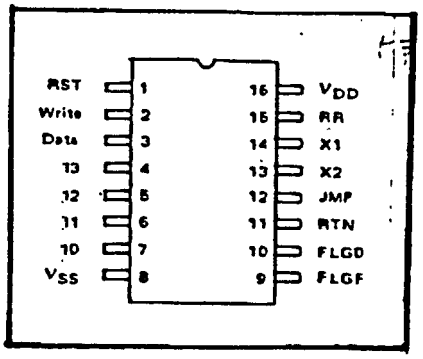
รูปแสดง การใช้งานไมโครโปรเซสเซอร์ในขบวนการอุตสาหกรรม

1.2 ไมโครโปรเซสเซอร์ 1 บิท

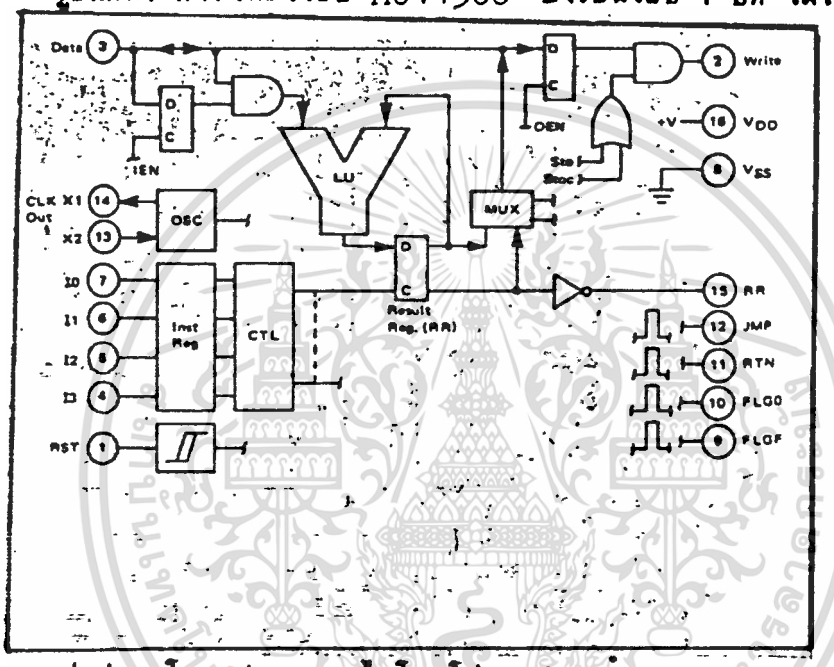
MC14500 เป็นไมโครโปรเซสเซอร์ที่ทำงานด้วยบัสข้อมูลเพียงบิทเดียว เป็นไอซีไมโครโปรเซสเซอร์ที่มีขาเพียง 16 ขา ขนาดของไอซีเท่ากับไอซีเกทตัวหนึ่งเท่านั้น มีคำสั่งที่จะควบคุมการทำงานเพียง 16 คำสั่ง และเป็นไมโครโปรเซสเซอร์ที่เข้าใจได้ง่าย ใช้งานง่าย เป็นไอซีซีมอส (CMOS) ใช้กำลังต่ำมาก และเนื่องจากเป็นซีมอสจึงทำให้ใช้งานกับแหล่งจ่ายไฟเลี้ยงได้ตั้งแต่ 3 ถึง 15 โวลท์เหมือนซีมอสทั่วไป และที่แตกต่างกับไมโครโปรเซสเซอร์อื่นคือ มันสามารถทำงานได้จากออสซิลเลเตอร์ที่ความถี่ตั้งแต่ไฟตรงจนถึง 1 เมกกะเฮตร์

ความแตกต่างระหว่างไมโครโปรเซสเซอร์ 1 บิท กับไมโครโปรเซสเซอร์อื่น เช่น 8 บิท 16 บิท ที่เห็นได้ชัดคือ ไมโครโปรเซสเซอร์ 1 บิท จะนำข้อมูลทีละบิทมาประมวลผล การประมวลผลจะเป็นไปด้วยคำสั่งง่าย ๆ ซึ่งเกี่ยวกับการกระทำทางลอจิก โดยมีไมโครโปรแกรมอยู่ภายในไมโครโปรเซสเซอร์และควบคุมการทำงานดังกล่าวโครงสร้างของไมโครโปรเซสเซอร์ และการจัดขาไอซีแสดงได้ดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง การจัดขาไอซี MC14500 ซึ่งเป็นไอซี 1 บิท ไมโคาโปรเซสเซอร์



รูปแสดง โครงสร้างของไมโครโปรเซสเซอร์ MC14500

ไมโครโปรเซสเซอร์เบอร์นี้มีโครงสร้างทางฮาร์ดแวร์ที่ค่อนข้างจะเข้าใจได้ง่าย กล่าวคือมันจะรับรหัสคำสั่งจากหน่วยความจำ 4 บิท ซึ่งก็มีคำสั่งได้เพียง 16 คำสั่ง คำสั่งเหล่านี้จะได้รับการอ่านเข้ามายังที่ Inst Reg หรือรีจิสเตอร์รับคำสั่งแล้วตีความหมายเพื่อกระทำตาม สิ่งที่น่าสังเกตคือบัสข้อมูลจะมีเพียงบิทเดียว ซึ่งนำข้อมูลเข้ามาประมวลผล

รูปแบบของคำสั่งทั้ง 16 คำสั่งมีรายละเอียดดังตาราง

สิ่งที่น่าสังเกตสำหรับสถาปัตยกรรมของไมโครโปรเซสเซอร์เบอร์นี้คือ มีรีจิสเตอร์หลักเพียง RR ซึ่งเป็นรีจิสเตอร์ 1 บิท ส่วนการทำงานจะกระทำได้เพียงคำสั่งทางลอจิก คือ AND OR NOT EXNOR ซึ่งเราสามารถผสมให้เกิดการทำงานทางลอจิกได้ อย่างไรก็ดีจะเห็นความแตกต่างกับไมโครโปรเซสเซอร์ตัวอื่นได้ชัดเจนคือ การทำงานหลายคำสั่งให้ผลลัพธ์เป็นพัลส์ที่ออกจากขาของไมโครโปรเซสเซอร์ เช่น JMP RTN เป็นต้น ซึ่งโครงสร้างการควบคุมโปรแกรมนี้ จะเป็นฮาร์ดแวร์ภายนอกทำการควบคุม หรือแม้แต่โปรแกรม

เอกสารนี้มีไว้สำหรับการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เคาน์เตอร์ซึ่งก็ต้องมีฮาร์ดแวร์วงจรนับภายนอกทำการควบคุมการทำงาน

สิ่งที่น่าสังเกตอีกประการหนึ่งคือ รีจิสเตอร์ 1 บิทอีก 2 ตัวคือ IEN และOEN เป็นรีจิสเตอร์ที่มีลักษณะเป็นเพียง D ฟลิบฟลอบแลทช์เท่านั้น IEN มีชื่อว่าอินพุทอีนาเบิ้ล เป็นรีจิสเตอร์ที่จะมีผลในการอีนาเบิ้ลเส้นทางจากบัสข้อมูลมากระทำกับ LU (Logic Unit) ส่วน OEN คือส่วนของรีจิสเตอร์ที่จะอีนาเบิ้ลบัสทางเอ้าท์พุทเพื่อจะส่งพัลส์ Write ออกไปหรือไม่ โครงสร้างการทำงานของรีจิสเตอร์ทั้งสองตัวนี้มีความสำคัญมากและจะต้องเข้าใจให้ดี

ตารางแสดง คำสั่งต่าง ๆ ทั้ง 16 คำสั่งของ MC 14500

รหัสคำสั่ง			ความหมาย
เลขฐานสิบหก	เลขไบนารี	นิวโมนิคส์	
0	0000	NOPO	No Operation Zero ทุกรีจิสเตอร์ไม่มีการเปลี่ยนแปลง ส่วนแฟลก ๑ ซึ่งเป็นเอาท์พุท จะส่งพัลส์ออกไป 1 ถูก พัลส์จะเริ่มเมื่อขอบสัญญาณนาฬิกา X_1 เปลี่ยนจาก"1" เป็น"0"
1	0001	LD	LOAD ค่าของ RR (Result Register) จะได้รับการโหลดด้วยข้อมูลจากบัสข้อมูล
2	0010	LDC	LOAD COMPLEMENT เป็นการโหลดค่าจากบัสข้อมูลมาเก็บใน RR โดยมีการคอมพลีเมนท์ข้อมูลก่อนนำมาเก็บ
3	0011	AND	AND เป็นการนำเอาข้อมูลจาก RR มาทำการ AND กับข้อมูลในบัสข้อมูล ผลลัพธ์ที่ได้เก็บไว้ที่ RR
4	0100	ANDC	AND COMPLEMENT เป็นการนำเอาข้อมูลจากบัสข้อมูลมาทำการคอมพลีเมนท์ แล้ว AND กับ RR ผลลัพธ์ที่ได้เก็บไว้ที่ RR

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รหัสคำสั่ง			ความหมาย
เลขฐานสิบหก	เลขไบนารี	นิวโมนิคส์	
5	0101	OR	OR เป็นการนำเอาข้อมูลจากบัสข้อมูลมาทำการ OR กับข้อมูลใน RR ผลลัพธ์ที่ได้เก็บไว้ที่ RR
6	0110	ORC	OR COMPLEMENT เป็นการนำเอาข้อมูลจากบัสข้อมูลมาทำการคอมพลีเมนท์แล้ว OR กับค่าใน RR ผลลัพธ์เก็บไว้ที่ RR
7	0111	XNOR	EXCLUSIVE NOR เป็นการกระทำ EX-NOR ระหว่างค่าในบัสข้อมูลกับ RR ผลลัพธ์เก็บไว้ที่ RR หรือกล่าวได้ว่าผลลัพธ์ที่ RR จะมีค่าเป็น"1" เมื่อข้อมูลใน RR เดิม กับในบัสข้อมูลเหมือนกันและผลลัพธ์ใน RR จะเป็น"0" ถ้าข้อมูลใน RR เดิมและบัสข้อมูลต่างกัน
8	1000	STO	STORE เป็นการส่งข้อมูลใน RR เข้าไปยังบัสข้อมูลและมีการส่งพัลธ์บวกออกทางขา Write โดยพัลธ์ที่เริ่มขึ้นเมื่อสัญญาณที่ X_1 เปลี่ยนสถานะเป็น"0" ค่าใน RR จะไม่มีการเปลี่ยนแปลง
9	1001	STOC	STORE COMPLEMENT เหมือนกับคำสั่ง STO แตกต่างกันที่ข้อมูลจะได้รับการคอมพลีเมนท์ก่อน
A	1010	IEN	INPUT ENABLE เป็นการโหลดข้อมูลจากบัสข้อมูลมาไว้ยังรีจิสเตอร์ IEN
B	1011	OEN	OUTPUT ENABLE เป็นการโหลดข้อมูลจากบัสข้อมูลไว้ยังรีจิสเตอร์ OEN

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

รหัสคำสั่ง			ความหมาย
เลขฐานสิบหก	เลขไบนารี	นิวโมนิคส์	
C	1100	JMP	JUMP จะมีพัลส์ได้รับการส่งออกมาที่ขา JMP พัลส์นี้จะเริ่มเมื่อสัญญาณที่ X_1 เปลี่ยนสถานะจาก "1" เป็น "0"
D	1101	RTN	RETURN จะมีพัลส์ออกมาที่ขา RTN หนึ่งพัลส์โดยเริ่มต้นเมื่อสัญญาณที่ X_1 เปลี่ยนสถานะจาก "1" เป็น "0"
E	1110	SKZ	Skip if Zero ถ้าหากว่า RR มีค่าเป็น "0" ระหว่างการกระทำตามคำสั่งนี้ จะทำให้เกิดการข้ามการทำงานคำสั่งถัดไป (อ่านคำสั่งถัดไปแต่ไม่กระทำอะไร)
F	1111	NOPF	No Operation F จะไม่มีการเปลี่ยนค่าในรีจิสเตอร์ใดและจะมีพัลส์ส่งออกทางขา FLAGF 1 พัลส์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

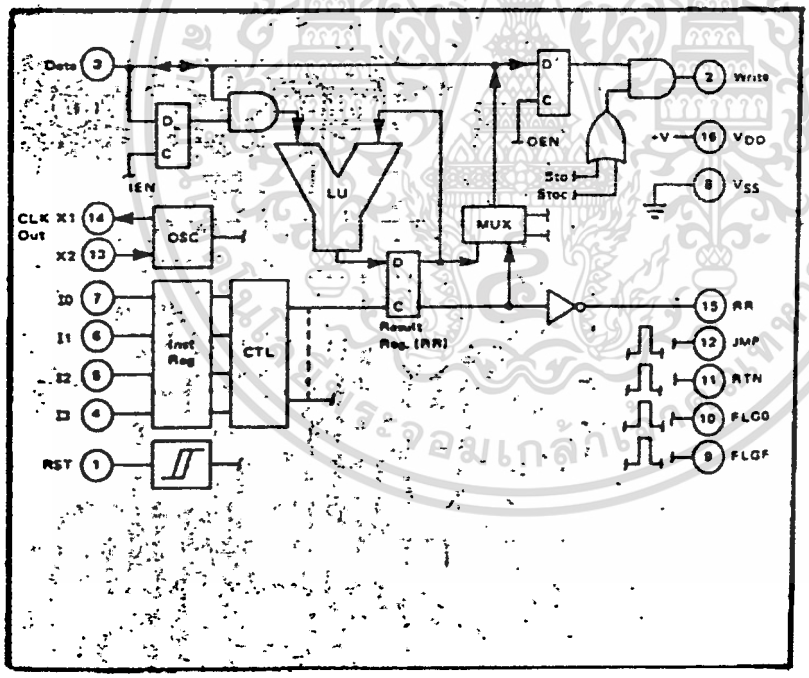
บทที่ 2

ทฤษฎีหรือหลักการ

2. ฮาร์ดแวร์ของ MC 14500

ในบทนำ ได้กล่าวถึงโครงสร้างของไอซีไมโครคอนโทรลเลอร์ ซึ่งเป็นไมโครโปรเซสเซอร์ขนาด 1 บิท ซึ่งจะเห็นโครงสร้างของไมโครโปรเซสเซอร์ทำงานแบบทีละขั้น โดยมีคำสั่งที่จะรับเข้าขนาด 1 บิท และติดต่อกับอินพุท หรือเอาท์พุทได้ทีละบิท การประมวลผลก็ทำได้ด้วยคำสั่งพื้นฐาน

โครงสร้างของไมโครโปรเซสเซอร์ตัวนี้ จึงเป็นโครงสร้างที่เหมาะแก่การทดลอง โดยผู้ทดลองสามารถเข้าใจการทำงานตั้งแต่พื้นฐาน ทั้งทางฮาร์ดแวร์และซอฟแวร์ ในที่นี้ขอทบทวนโครงสร้างภายใน MC 14500 อีกครั้งหนึ่ง ดังรูปที่แสดง



รูปแสดง โครงสร้างภายในของ MC 14500

จากโครงสร้างภายในนี้เองจะเห็นว่าการรับคำสั่งจะรับออปโค้ด 4 บิท ซึ่งก็จะได้รับคำสั่งทั้งสิ้นเพียง 16 คำสั่งเท่านั้น ดังตารางที่แสดง

สิ่งที่น่าสังเกตคือ คำสั่งหลัก ๆ จะกระทำกับ Result Register เช่น AND ก็หมายถึงนำข้อมูลจากบัสข้อมูลเข้ามา AND กับ Result Register แล้วแลทช์เข้ากับผลลัพธ์ไว้ที่รีจิสเตอร์เดิมนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางแสดงคำสั่งและรหัสจำนวน 16 คำสั่งของ MC [illegible]

Instruction Code		Mnemonic	Action
#0	0000	NOPO	No change in registers, RR→RR, FLGO→ ⎍
#1	0001	LD	Load Result Reg. 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 → Data Pin, Write ← 1
#9	1001	STOC	Store Compl. $\overline{RR}$ → Data Pin, Write ← 1
#A	1010	IEN	Input Enable, D → IEN Reg.
#B	1011	OEN	Output Enable, D → OEN Reg.
#C	1100	JMP	Jump, JMP Flag ← ⎍
#D	1101	RTN	Return, RTN Flag ← ⎍, Skip next inst.
#E	1110	SKZ	Skip next instruction if RR = 0
#F	1111	NOPF	No change in Registers RR → RR, FLGF ← ⎍

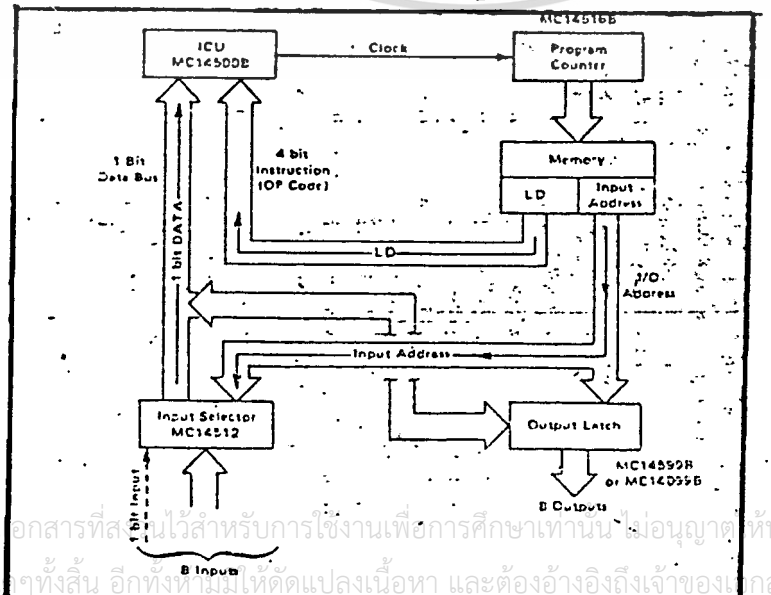
เมื่อเป็นเช่นนี้ โครงสร้างของโปรแกรมจะต้องอยู่ภายในหน่วยความจำภายนอก โดยมีส่วนของโปรแกรมเคาวน์เตอร์เป็นตัวชี้งโครไนซ์ ให้คำสั่งส่งผ่านเข้ามา และโครงสร้างการติดต่อกับอินพุท-เอาท์พุท บิทใดจะใช้หลักการของคำสั่งที่ประกอบเป็นดังนี้

4 บิท	โอเปอร์แรนด์	โอเปอร์แรนด์
ออปโค้ด	บอกอินพุท	บอกเอาท์พุท

เช่น ถ้าต้องการ LD จากบิท 5 ของอินพุทเข้ามายังรีจิสเตอร์ผลลัพธ์ภายในซีพียูก็ใช้คำสั่ง

LD	บอกบิทอินพุท	ดิสเอเบิ้ล

โครงสร้างการติดต่อกับอินพุทเอาท์พุทบิทใดจะใช้หลักการของตัวเลือกข้อมูลซึ่งเป็นฮาร์ดแวร์ โดยมีโครงสร้างของระบบเป็นดังรูปที่แสดง



รูปแสดง โครงสร้างของระบบที่จะต่อออกไปใช้งาน

026900

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ดูจากแผนผังของการทำงานในรูปที่แสดง จะเห็นว่าโปรแกรมที่เก็บไว้ในหน่วยความจำจะได้รับการกำหนดแอดเดรสผ่านทางโปรแกรมเคาน์เตอร์ ตัว MC 14500 นี้จะรับข้อมูลจากอินพุทมาประมวลผล โดยข้อมูลอินพุทนี้จะรับเข้ามาได้ทีละบิท แต่จะเป็นบิทใดก็จะขึ้นอยู่กับตัวเลือกที่เรียกว่า ROM แอดเดรสหรือตัวเลือกแอดเดรส ในทำนองเดียวกันเมื่อข้อมูลที่ผ่านการประมวลผลแล้ว และต้องการที่จะส่งออกทางเอาท์พุทก็จะส่งผ่านออกทางบัสข้อมูลเช่นกัน โดยมีตัวเลือกแอดเดรสเป็นตัวบอกว่าจะให้ข้อมูลออกทางบิทใด

วงจรที่จะกล่าวถึงนี้เป็นวงจรที่จะสร้างขึ้นง่าย ๆ เพื่อใช้งานโดยยึดหลักให้ MC14500 เป็นระบบควบคุมลอจิกสำหรับการทำงานเป็น PC-Programmable Controller สำหรับงานควบคุมในอุตสาหกรรมโดยใช้โปรแกรมควบคุมอยู่ใน ROM ซึ่งการทำงานทั้งหมดในที่นี้จะมีอินพุท 8 บิท และเอาท์พุท 8 บิท แต่ขยายเอาท์พุทอีก 8 บิทได้การต่อวงจรบางบิทของเอาท์พุท เราอาจป้อนกลับมาเป็นอินพุทของระบบเพื่อให้เป็นเสมือนที่เก็บข้อมูลชั่วคราวได้

2.1 <u>การอินเตอร์เฟสกับแป้นกดข้อมูล</u>

แป้นกดข้อมูลเป็นอุปกรณ์ที่สำคัญอย่างหนึ่ง ที่ใช้นำข้อมูลที่เป็นตัวเลขและตัวอักษรรวมทั้งรหัสควบคุมต่าง ๆ เข้าสู่เครื่องคอมพิวเตอร์

ลักษณะของแป้นข้อมูลจะประกอบด้วยปุ่มกด (KEY) ซึ่งเป็นตัวอักษรและเป็นตัวเลขนำมาจัดเรียงกันในรูปของเมทริกส์ (MATRIX) เครื่องคอมพิวเตอร์จะรับรู้การกดปุ่มเหล่านี้ได้จะต้องมีการเข้ารหัส (ENCODED) เพื่อแทนตัวอักษรและตัวเลขบนแป้นกด ซึ่งอาจจะใช้วิธีการทางซอฟแวร์ก็ได้ โดยทั่วไปแป้นกดข้อมูล จะแบ่งเป็นแป้นกดข้อมูลแบบเข้ารหัส (ENCODED KEYBOARD) กับแป้นกดข้อมูลแบบไม่มีการเข้ารหัส (NON DECODED KEYBOARD) ซึ่งจะกล่าวในลำดับต่อไป

2.1.1 <u>เบานส์</u>

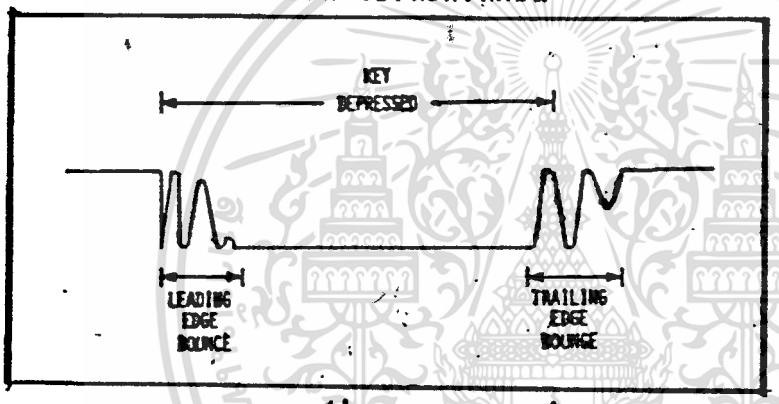
เป็นตัวที่ทำให้เกิดปัญหาเป็นอย่างมาก ซึ่งจะเกิดขึ้นชั่วขณะหนึ่งประมาณ 10 mS ทุกครั้งที่มีการสัมผัส (CLOSE) หรือจากกัน (OPEN) ของหน้าสัมผัส (CONTACT) ซึ่งเป็นกลไกของสวิทช์ที่ใช้ทำเป็นปุ่มกด เบานส์จะเกิดขึ้นทุกครั้งที่มีการกดปุ่มและปล่อยนิ้วมือจากปุ่มกด เบานส์จะมีรูปคลื่นตามที่รูปแสดงเบานส์ยังเป็นตัวที่ทำให้เกิดปัญหาเป็นอย่างมากสำหรับเครื่องคอมพิวเตอร์ ที่จะทำการตรวจแยกว่าสัญญาณที่เข้ามานั้นเบานส์หรือแป้นรหัสสัญญาณที่เกิดจากการกดปุ่ม ดังนั้นเพื่อให้เครื่องคอมพิวเตอร์ได้รับสัญญาณที่ถูกต้อง จึงจำต้องทำการแก้ปัญหาเรื่องเบานส์ก่อนที่จะส่งสัญญาณไฟฟ้าซึ่งเป็นรหัสที่ถูกต้องแก่เครื่องคอมพิวเตอร์

เอกสารนี้เป็นการชี้แจงไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

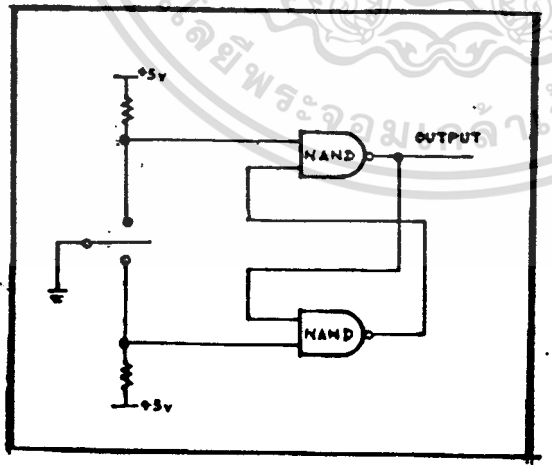
เพื่อใช้งานขั้นต่อไป

การแก้ปัญหาเบาน์ส์เรียกว่าดีเบาน์ส์ (debounce) โดยทั่วไปสามารถทำได้ 2 วิธีคือ

ก) ใช้วิธีการทางฮาร์ดแวร์ โดยการต่ออุปกรณ์เป็นวงจร เพื่อกำจัดเบาน์ส์ให้หมดไป วิธีง่าย ๆ ได้แก่ การต่อวงจรกรอง RC (RC filter) ที่บริเวณหน้าสัมผัสของปุ่มกดทุกปุ่ม หรือไม่ก็ใช้วิธีต่อวงจรฟลิบฟล็อบ อย่างเช่นในรูปที่แสดงก็ได้ วิธีนี้ค่อนข้างยุ่งยากเพราะต้องมีวงจรฮาร์ดแวร์เพิ่มขึ้น เหมาะสำหรับแป้นกดข้อมูลที่มีปุ่มกดไม่กี่ปุ่ม แต่ไม่เหมาะที่จะใช้กับแป้นกดข้อมูลที่มีปุ่มกดหลายปุ่ม เพราะจะต้องมีวงจรลักษณะนี้เพิ่มขึ้นเท่ากับจำนวนปุ่ม ซึ่งทำให้เปลืองค่าใช้จ่ายและใช้เนื้อที่เพิ่มขึ้น



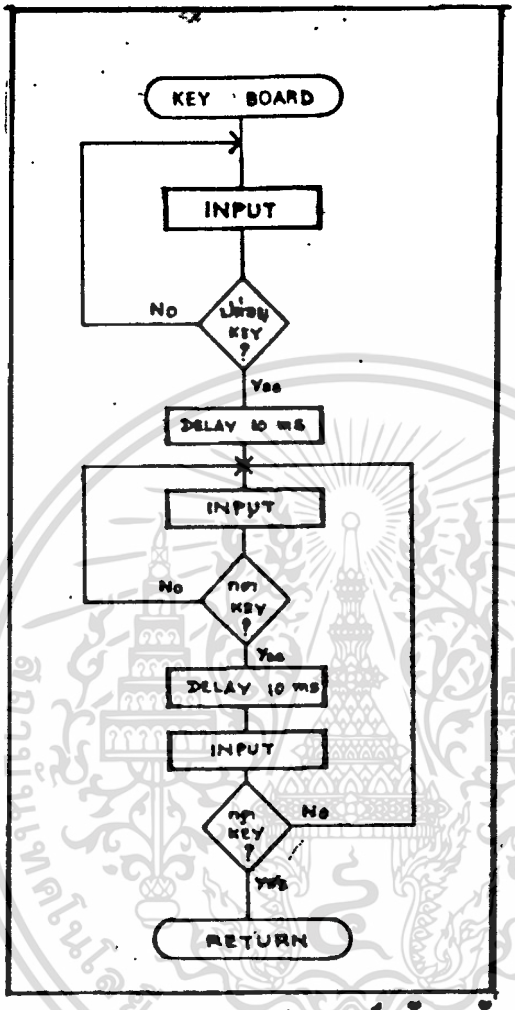
รูปแสดง เบาน์ส์ที่เกิดจากการกดปุ่ม



รูปแสดง วงจรฟลิบฟล็อบ ทำหน้าที่ดีเบาน์ส์

ข) ใช้หลักการหน่วงเวลา (delay) การตรวจสอบสภาพการกดด้วยซอฟท์แวร์ วิธีนี้จะหน่วงเวลาประมาณ 10 - 20 mS และคอยตรวจสอบสเตตัสหลังการกดปุ่มด้วยเทคนิคทางซอฟแวร์ วิธีนี้เหมาะสำหรับแป้นกดข้อมูลที่มีปุ่มกดจำนวนมากปุ่ม หลักการของวิธีนี้จะอธิบายให้เห็นเป็นผังงานดังรูปที่แสดง จากผังงานจะสังเกตเห็นว่าเวลาหน่วง 10 mS ช่วงแรก

เป็นการดีเบานส์ช่วงตอนปล่อยนิ้วมือจากปุ่มกดเวลาหน่วง 10 mS ช่วงหลังเป็นการดีเบานส์ช่วงตอนเริ่มกดปุ่ม



รูปแสดง ผังงานโปรแกรมดีเบานส์แป้นกดข้อมูล

2.1.2 ROLLOVER

Rollover เป็นปัญหาที่เกิดขึ้นกรณีที่บังเอิญผู้ใช้กดปุ่มของแป้นกดข้อมูลสองปุ่มหรือมากกว่าในเวลาเดียวกัน ทำให้รหัสที่จะส่งเข้าเครื่องคอมพิวเตอร์ผิดพลาดได้ ดังนั้นจึงจำเป็นต้องหาวิธี ป้องกันไม่ให้เกิดปัญหาเช่นนี้ขึ้นเพื่อให้เครื่องคอมพิวเตอร์รับรหัสของปุ่มที่ผู้ใช้ต้องการจะกดเพียงค่าเดียวเท่านั้น วิธีการในการป้องกันมีอยู่ 3 วิธีด้วยกันคือ

ก) Two - key rollover เป็นการป้องกันการกดปุ่ม 2 ปุ่มในขณะเดียวกัน มีวิธีที่นิยมใช้กันอยู่ 2 วิธีคือ อย่างที่ง่ายที่สุดจะไม่มีการอ่านค่าจากแป้นข้อมูล ในขณะที่มีการกดปุ่ม 2 ปุ่มพร้อมกัน จนกว่าจะปล่อยตัวหนึ่งตัวใดก่อน จะอ่านค่าของปุ่มสุดท้ายที่เหลืออยู่เท่านั้น วิธีนี้มักนิยมใช้กรณีใช้ซอฟแวร์ ในการสแกน (scan) และถอดรหัสจากแป้นกดข้อมูลเท่านั้น อีกวิธีก็คือการใช้วงจรทางด้านฮาร์ดแวร์ในการป้องกันไม่ให้ปุ่ม ที่อยู่ข้างเคียงปุ่มที่ถูกกด สามารถผลิตสัญญาณส่งเข้าสู่เครื่องคอมพิวเตอร์ได้ หลังจากที่กดตัวแรกไปแล้วถ้ากด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

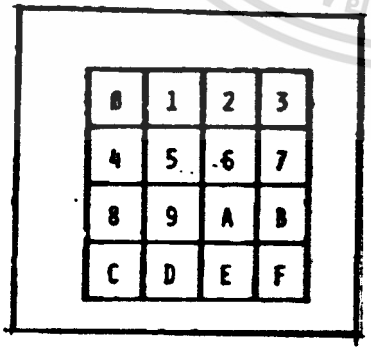
ปุ่มที่สองตามมาในขณะที่ยังไม่ปล่อยมือจากตัวแรก การกดปุ่มที่สองนั้นจะไม่มีผล สัญญาณจากปุ่มที่สองยังไม่สามารถจะส่งเข้าเครื่องได้จนกว่าจะปล่อยมือจากปุ่มกดตัวแรกเสียก่อน วงจรที่ทำหน้าที่นี้จะประกอบด้วยตัวหน่วงเวลาและแลทช์

ข) N - key rollover วิธีนี้จะไม่ยอมรับข้อมูลที่เกิดจากการกดปุ่มหลายปุ่มพร้อมกันจนกว่าจะปล่อยมือเหลือปุ่มที่กดเพียงปุ่มเดียว วิธีนี้คล้ายคลึงกับกรณี Two - key rollover เพียงแต่จำนวนการป้องกันปุ่มที่กดพร้อมกันมีมากกว่าสองเท่านั้น วิธีนี้มักจะใช้กับแป้นกดข้อมูลของระบบที่มีราคาแพง

ค) N - key lockout วิธีนี้จะยอมรับการกดหรือปล่อยมือของปุ่มเพียงปุ่มเดียวเท่านั้น ปุ่มอื่นที่กดในเวลาใกล้เคียงกันจะไม่มีผล วิธีนี้ง่ายเป็นที่นิยมใช้กันมาก เพียงแต่มีข้อเสียตรงที่ไม่สามารถทำการกดปุ่มได้รวดเร็วเท่านั้น เพราะสัญญาณจากปุ่มที่กดตามมาจะมีผลต่อเมื่อปล่อยมือจากปุ่มแรกแล้วเท่านั้น

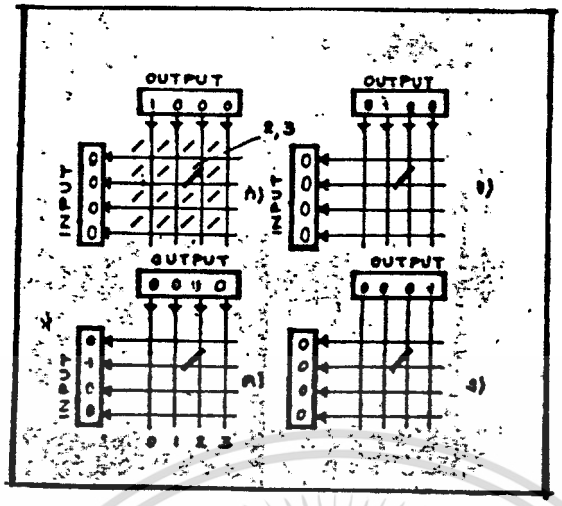
2.1.3 <u>แป้นกดข้อมูลชนิดไม่เข้ารหัส</u> (Non - Encoded Keyboard)

แป้นกดข้อมูลแบบนี้จะใช้ซอฟแวร์ในการเข้ารหัสตัวเลขและตัวอักษรต่าง ๆ บนแป้นกดแทนการใช้ตัวเข้ารหัสทางฮาร์ดแวร์ แต่จะใช้ LSI พิเศษบางตัวช่วย เช่น PIA(8255) เป็นต้น โดยปกติแป้นกดข้อมูลแบบนี้ จะจัดการเรียงปุ่มกดบนแป้นกดให้เป็นแถวและคอลัม ซึ่งจะกำหนดให้มีขนาด n x m และจะใช้หลักการสแกน (Scan) ทีละเส้น ทั้งทางด้านแถวและคอลัมคล้ายกับว่าเดินสำรวจทีละตัว จากรูปที่แสดง ซึ่งเป็นแป้นกดข้อมูลขนาด 16 ตัว สามารถแทนด้วย 4 แถว และ 4 คอลัม ได้ดังรูปที่แสดง



รูปแสดง HEX KEYBOARD

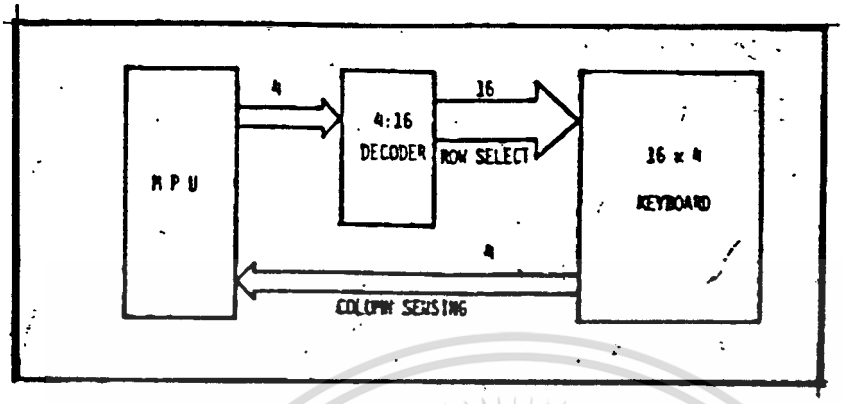
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง WALKING ONES KEYBOARD DECODE

จากรูปที่แสดงจะเห็นว่า HEX KEYBOARD ซึ่งมีอยู่ 1ค ปุ่มกดจะแทนด้วย 4 แถว ซึ่งเป็นอินพุท และ 4 คอลัม ซึ่งเป็นเอ้าท์พุท เราจะใช้เทคนิคการสแกนคอลัม (COLOUM SCANING) เพื่อหาปุ่มที่ถูกกด จากรูปที่แสดง จะเห็นว่าจะเริ่มการสแกนคอลัมที่ 1 ก่อนโดยให้ D_0 เป็น "1" และD_1, D_2, D_3 เป็น "0" โดยใช้คำสั่ง OUT ตามด้วยข้อมูล(1000) หลังจากนั้นจะทำการอ่านข้อมูลซึ่งเป็นการสแกนทางด้านแถวด้วยคำสั่ง IN จากพอร์ท 2 จะเห็นว่าตอนนี้ข้อมูลที่รับเข้ามาทางพอร์ท 2 ยังเป็น (0000) เนื่องจากปุ่มที่กดอยู่ที่แถวที่ 2 คอลัมที่ 3 เมื่อไม่เจอก็ทำการเอ้าท์พุทใหม่ คือส่งข้อมูล (0100) ออกพอร์ท 1 เพื่อสแกนคอลัม 2 ให้ D_1 เป็น "1" แล้วทำการอินพุทเข้ามาใหม่ทางพอร์ท 2 ขณะนี้ค่าที่อินพุทเข้าไปก็ยังคงเป็น (0000) อยู่ เนื่องจากปุ่มกดที่กดอยู่ที่คอลัม 3 แถว 2 โปรแกรมจะกลับมาทำการสแกนทางด้านคอลัม 3 โดยการ OUT ค่า (0010) ออกพอร์ท 1 อีก หลังจากนั้นจะทำการอินพุทเข้าไปอีกเพื่อสแกนทางด้านแถว คราวนี้ค่าที่อ่านเข้ามาจะมีค่า (0100) เนื่องจากปุ่มที่กดอยู่ที่ตำแหน่งแถว 2 คอลัม 3 ทำให้สายเดิดต่อถึงกัน เมื่อพบว่ามีการกดปุ่มรหัสที่อ่านเข้ามาจะถูกนำไปถอดรหัสอีกที่ว่าเป็นตัวอักษรหรือตัวเลขอะไร ถ้ามองจากรูปที่แสดงซึ่งเป็น HEX KEYBOARD เมื่อทำการถอดรหัสจากคอลัม 3 แถว 2 แล้วก็จะรู้ว่าเป็นเลข 6 จะเห็นว่าจากหลักอันนี้เราสามารถจะถอดรหัสจากการกดปุ่มได้ทั้งหมด และสามารถแก้เบานส์ได้ด้วย การเขียนโปรแกรมและยังสามารถจะเพิ่มปุ่มกดขึ้นได้อีก เมื่อต้องการเพียงแต่เขียนโปรแกรมเพิ่มขึ้นเท่านั้น จากรูปที่แสดงข้างล่างจะแสดงให้เห็น โครงสร้างวงจรของแป้นกด ข้อมูลชนิด 64 ปุ่มกด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง แป้นกดข้อมูลชนิด 64 ปุ่ม

2.1.4 แป้นกดข้อมูลชนิดเข้ารหัส (ENCODED KEYBOARD)

น้อยคนนักที่จะยอมลำบากในการเขียนโปรแกรม เพื่อช่วยในการเข้ารหัสแป้นกดข้อมูลโดยวิธี NON ENCODED KEYBOARD ยังมีวิธีที่สะดวกกว่าก็คือให้แป้นกดข้อมูลแบบเข้ารหัส ซึ่งแบบนี้จะใช้ฮาร์ดแวร์เป็นตัวช่วยเข้ารหัส ตัวเลขและตัวอักษรต่าง ๆ บนแป้นกด ซึ่งเครื่องคอมพิวเตอร์จะสามารถรับรู้ได้ว่าปุ่มที่กดเป็นตัวเลขหรือตัวอักษรอะไร และรหัสนั้นจะถูก HOLD ไว้ชั่วขณะหนึ่งจนกว่าจะมีการกดตัวใหม่เข้ามา แป้นกดข้อมูลแบบนี้จะถูกทำสำเร็จในรูปของ L S I โดยจะใช้การสแกนแบบแมททริกส์ซึ่งจะมีวงจรดีเบานส์และการป้องกัน ROLLOVER รวมทั้งแลทช์สำหรับเก็บข้อมูลขณะกดไว้ใช้งานรวมอยู่ในชิพตัวเดียวกัน ชิพบางตัวจะมี ROM รวมอยู่ด้วยไว้สำหรับเป็นตาราง (LOOK UP TABLE) เพื่อที่จะจ่ายรหัสที่ตรงกับปุ่มกดที่กดเข้าไปซึ่งอาจจะเป็น ASCII หรือ EBEDIC ก็ได้ขึ้นอยู่กับการโปรแกรมใน ROM นั้น

2.1.5 ตัวเข้ารหัสแป้นกดข้อมูล (KEY BOARD ENCODER)

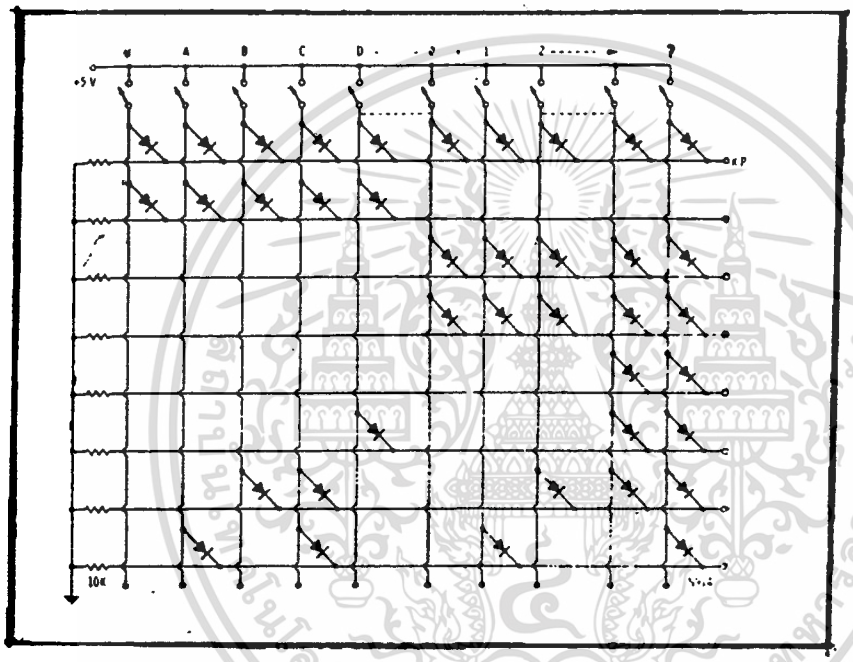
บทบาทขั้นพื้นฐานของตัวเข้ารหัสแป้นกดข้อมูลก็คือจะเป็นตัวให้กำเนิดข้อมูล 8 บิท ออกมาหลังจากที่มีการกดปุ่มกด และขณะเดียวกันชิพตัวนี้จะทำหน้าที่ดีเบานส์และป้องกันการเกิด ROLLOVER อีกด้วย ตัวเข้ารหัสที่สำคัญมีอยู่ 3 ชนิดคือ

1. STATIC ENCODER
2. SCANNING ENCODER
3. CONVERTING ENCODER

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้ง

1. STATIC ENCODER เป็นแป้นกดข้อมูลที่จะให้กำเนิดรหัสที่สอดคล้องกับปุ่มกด

ที่ลด ซึ่งทำให้ง่ายต่อการแก้ปัญหา ตัวอย่างก็คือแป้นกดข้อมูลลักษณะสายตรง (LINEAR KEYBOARD) ถ้ามี 64 ปุ่มกดก็จะต้องมีสายสำหรับทุก ๆ ปุ่มกดซึ่งจะทำให้รับรู้ได้ง่าย เพราะจะทำให้เกิดพัลส์ขึ้นในแต่ละสายที่ถูกกดโดยปุ่มกด พัลส์ที่เกิดขึ้นนี้จะถูกให้เปลี่ยนรูปให้อยู่ในรูปที่เหมาะสมเช่น รหัส 8 บิท เพื่อที่จะลดจำนวนสายลง ตัวเข้ารหัส (ENCODER) จึงจำเป็นต้องจัดให้อยู่ในรูปแมททริกส์ซึ่งสามารถจัดให้อยู่ในขนาด 8 x 8 ได้ จึงสามารถลดจำนวนสายให้เหลือแค่ 16 เส้นเท่านั้น ราคาของแป้นกดข้อมูลก็ถูกลงกว่าเดิม ตัวอย่างของ STATIC ENCODER)ดังแสดงในรูป



รูปแสดง STATIC ENCODER

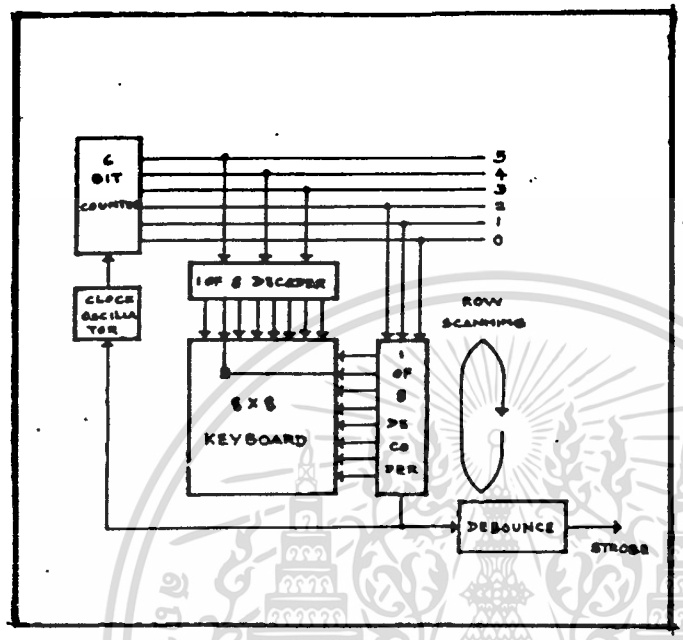
2. SCANNING ENCODER

SCANNING CHIP จะเป็นตัวช่วยแก้ปัญหาในการที่จะระบุว่าปุ่มกดตัวใดที่ถูกกดเข้ามา เมื่อใช้แป้นกดข้อมูลที่จัดในรูปแมททริกส์ แต่ละแถวของปุ่มกดจะถูกสแกนโดยตัวนับ (COUNTER) หลังจากทำการสแกนแล้วถ้ายังไม่มีการกดปุ่มการสแกนจะทำไปเรื่อย ๆ จนครบรอบเป็นวงกลมกลับมายังตำแหน่งเดิมทำแบบนี้หมุนเวียนไปเรื่อย ๆ ถ้ามีการกดปุ่มจังหวะนั้นวงจรจะสร้างสัญญาณ STROBE ขึ้นและการสแกนจะหยุดลงตัวนับจะหยุดนับ ดังนั้นค่าที่นับได้จะเป็นค่าที่บอกตำแหน่งของแถวและคอลัมของตำแหน่งของปุ่มที่ถูกกดนั้น

การสแกนแบบเดินหน้านี้ไม่อาจจะป้องกันการกดปุ่มทีเดียวสองปุ่มพร้อมกันได้ เนื่องจากการสแกนจะหยุดถ้ามีการกดปุ่มตัวแรกลง ดังนั้นถ้าจะกดปุ่มสองปุ่มควรเว้นระยะห่างกันเพราะจะทำให้ข้อมูลที่เข้าไปถูกต้อง เมื่อไรก็ตามถ้ามีการกดปุ่มพร้อม ๆ กันมากกว่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หนึ่งปุ่มในเวลาเดียวกัน ค่านั้นจะถือว่าเป็นค่าที่ไม่ถูกต้อง (ERROR) มันจะไม่หยุดการสแกนจนกว่าจะมีการกดปุ่มเพียงปุ่มเดียวเท่านั้นจึงจะหยุดการสแกน สำหรับข้อดีของวิธีนี้ยังช่วยในการดีเบานส์โดยอัตโนมัติอีกด้วย ตัวอย่างแสดงในรูป

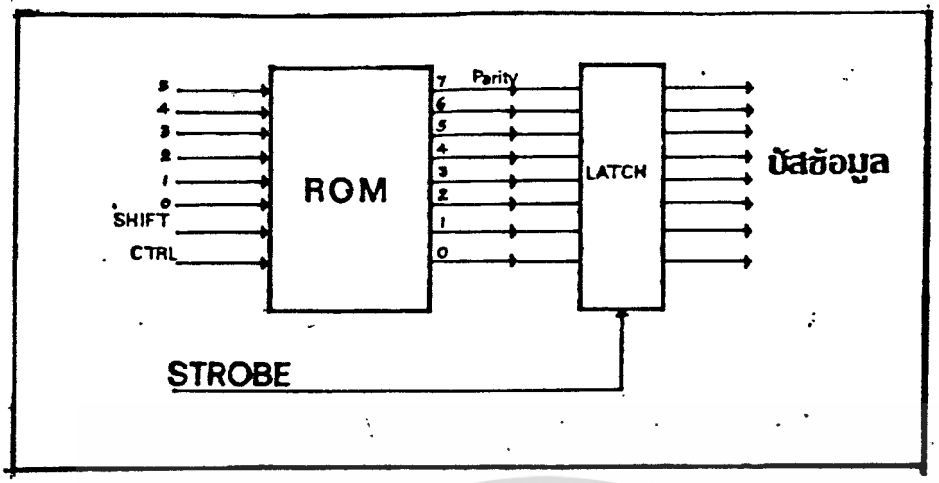


รูปแสดง SCANNING ENCODER

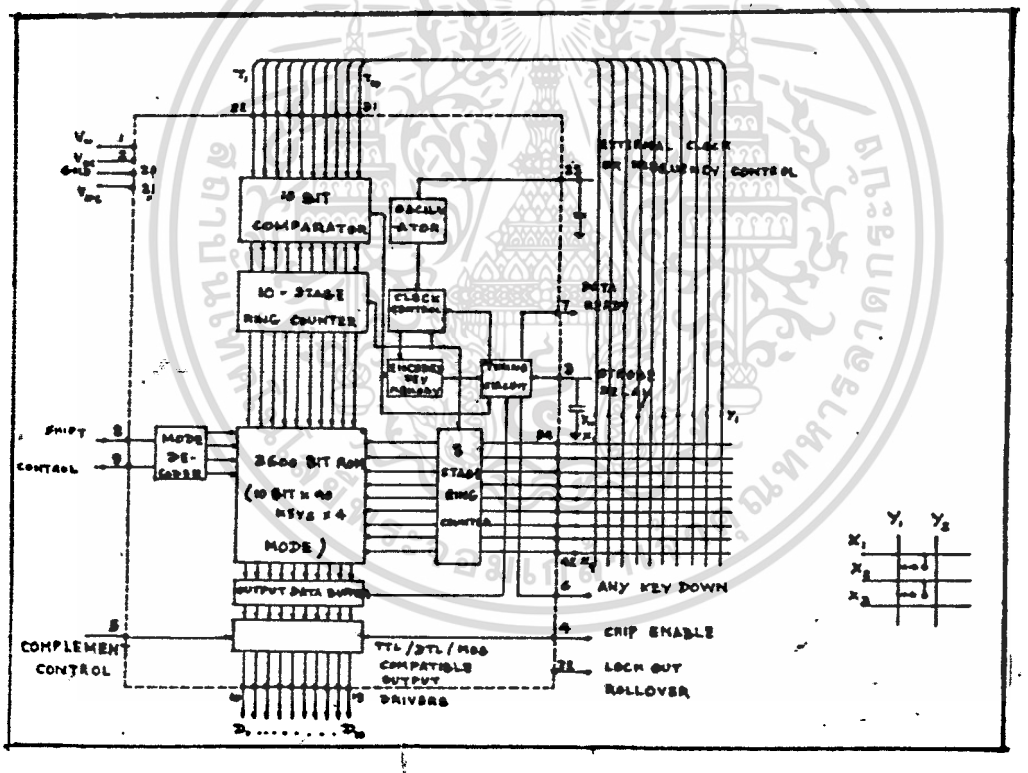
จากรูปที่แสดง จะเห็นว่าตัวนับจะมี 6 บิทโดยจะแบ่งบิท 0, 1, 2 มาผ่านตัวถอดรหัสเป็น 8 เส้นเพื่อทำการสแกนแถว R 3 บิทของตัวนับนี้เปลี่ยนแปลงเร็ว ดังนั้นสแกนแถวจะทำเร็วได้ สำหรับด้านคอลัมน์นั้นจะใช้บิท 3, 4, 5 ของตัวนับมาถอดรหัสออกมาเป็น 8 คอลัมเท่ากัน แต่จะเปลี่ยนแปลงช้ากว่า 3 บิทแรก (ทางด้านแถว) ดังนั้นถ้ามีการกดที่คอลัม 2 แถว 2 ทางด้านคอลัมจะถูกเลือกต่อ เมื่อบิท 5, 4, 3 เป็น 010 เมื่อมีการกดปุ่มและตัวนับได้นับมาถึงตำแหน่งของปุ่มนั้น วงจรภายในแป้นกดข้อมูลจะผลิตพัลส์สัญญาณ STROBE ขึ้นเพื่อไปใช้ในการหยุดในการนับของตัวนับ ขณะเดียวกันรหัสเอาท์พุทของตัวนับนี้ก็จะส่งไปที่ ROM ซึ่งเป็นตัวถอดรหัสให้เป็นรหัสปุ่มกดที่ต้องการ สัญญาณ STROBE จะส่งผ่านไปแลทช์ค่ารหัสของปุ่มเพื่อส่งเข้าเครื่องคอมพิวเตอร์อีกทีหนึ่ง ตามแสดงในรูป แสดงรูปของ ROM และแลทช์นี้

และจากรูปที่แสดงถัดมา แสดงให้เห็นโครงสร้างวงจรภายในของ LSI สำเร็จรูป NEC KEYBOARD ENCODER ซึ่งประกอบด้วย n - key rollover, n - key lockout และดีเบานส์ frequency control oscillator และสามารถเลือกได้ 4 MODE คือ SHIFT, CONTROL, SHIFT PLUS CONTROL และไม่กดทั้ง SHIFT และ CONTROL

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ROM และแลทช์



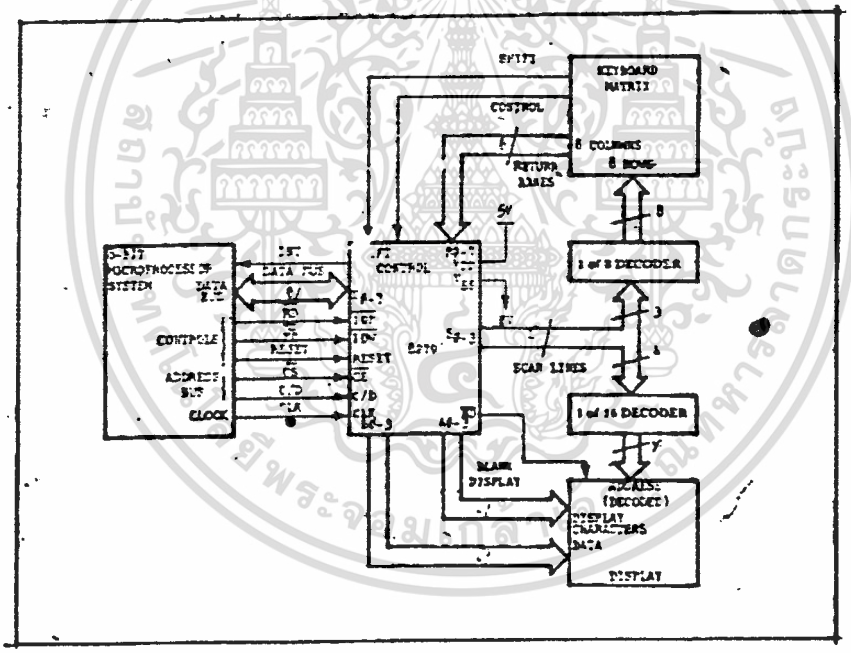
รูปแสดง NEC KEYBOARD ENCODER

ภายใน LSI นี้จะมี ROM ขนาด 3600 บิทซึ่งจะแบ่งเป็น 10 บิทเอาท์พุท สำหรับ 90 ปุ่ม 9๐ ปุ่มกดนี้จะจัดในรูปแมททริกส์ขนาด 9 x 10 ซึ่งจะมี 10 STAGE RING COUNTER ทำหน้าที่ควบคุมทางด้านคอลัมและ 9 STAGE RING COUNTER สำหรับควบคุมทางด้านแถว ส่วนทางด้านการนำข้อมูลออกจาก ROM จะผ่านทาง OUTPUT DATA BUFFER ซึ่งสามารถรับประกันได้ว่าค่ารหัสที่ออกมาจาก ROM จะไม่มีการ

เอกสารนี้มีลิขสิทธิ์ ไม่อนุญาตให้ผู้ใดนำไปใช้ประโยชน์ทางการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

RANDOM แน่ ๆ ในขณะที่ยังสแกนอยู่และยังไม่มีการกดปุ่มกด สำหรับค่ารหัสใน ROM นั้น ทางฝ่ายผู้ผลิตจะเป็นผู้กำหนดซึ่งอาจจะเป็น ASCII หรือ EBCDIC แป้นกดข้อมูลจะทำหน้าที่เป็น INPUT DEVICES ต่อร่วมกับระบบคอมพิวเตอร์โดยส่งข้อมูลผ่านทาง DATA BUS ซึ่ง DATA READY LINE จะเป็นตัวบอกให้ไมโครโปรเซสเซอร์ ทราบว่าได้มีการกดปุ่มแล้ว และพร้อมที่จะอ่านค่าจากแป้นกดข้อมูลได้

จากรูปที่แสดงตัวอย่าง 8279 ซึ่งเป็น KEYBOARD DISPLAY CONTROLLER จากรูปจะเห็นว่า LSI นี้จะใช้กับปุ่มกดขนาด 8 x 8 พร้อมกับ SHIFT และ CONTROL KEY ซึ่งสามารถจะมีได้ถึง 256 รหัส ตัวอย่างเช่นการกดปุ่ม SHIFT และ CONTROL และเครื่องหมาย "P" ก็จะถือว่าเป็นรหัสตัวหนึ่ง นอกจากจะทำหน้าที่เป็นตัวเข้ารหัสแล้ว อุปกรณ์ตัวนี้ยังทำหน้าที่แสดงผลข้อมูลที่อยู่ใน RAM ของ 8279 อีกด้วย

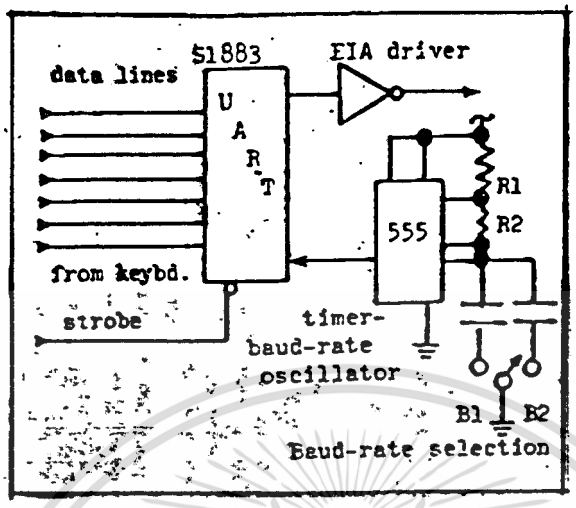


รูปแสดง 8279 KEYBOARD DISPLAY CONTROLLER.

2.1.6 แป้นกดข้อมูลรหัส ASCII

แป้นกดข้อมูลแบบนี้จะให้กำเนิดสัญญาณรหัส 7 บิท ASCII แป้นกดข้อมูลในลักษณะนี้อาจใช้ LSI Key board Controller chip ทำหน้าที่ถ่ายรหัส 7 บิท รวมทั้งสัญญาณ STROBE เพื่อใช้อินเตอร์เฟสร่วมกัน UART ก็ได้ ซึ่งวงจรการอินเตอร์เฟสร่วมกันก็ได้ ซึ่งวงจรการอินเตอร์เฟส ดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ASCII Serial Keyboard Interface

จากรูปจะเห็นว่าถ้ามีการกดปุ่มกดจะมีสัญญาณ STROBE มาควบคุม UART เพื่อรับข้อมูล ASCII 7 บิทแบบขนานเข้ามา UART จะแปลงสัญญาณเป็นแบบอนุกรม (Serial) ซึ่งมี FORMAT เป็น 10-11 บิท ขณะที่ UART ทำการส่งข้อมูลอยู่นั้น แม้ที่แป้นกดข้อมูลจะไม่มีการรับข้อมูลใหม่เข้ามา คือถ้ามีการกดก็จะไม่รับรู้ จากวงจรเราสามารถกำหนดอัตราเร็วในการส่งข้อมูลต่อบิทใดในหนึ่งวินาที ในการส่งแต่ละบิทจะใช้ 16 CLOCK CYCLE ฉะนั้นถ้าส่ง 110 BAUD จะใช้ต้องปรับให้ OSCILLATOR (555) มีความถี่ 1760 H_z ถ้าส่ง 300 BAUD ก็จะต้องใช้ 4800 H_z สำหรับรหัส 7 บิท ASCII ดูได้จากตาราง

ตารางรหัส ASCII

b_7	b_6	b_5	b_4	b_3	b_2	b_1	BIT NUMBERS / HEX	0	1	2	3	4	5	6	7
			0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
			0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
			0	0	1	0	2	STX	DC2	"	2	B	R	b	r
			0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
			0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
			0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
			0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
			0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
			1	0	0	0	8	BS	CAN	(	8	H	X	h	x
			1	0	0	1	9	HT	EM	)	9	I	Y	i	y
			1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
			1	0	1	1	11	VT	ESC	+	;	K	[	k	{
			1	1	0	0	12	FF	FS	,	<	L	\	l	|
			1	1	0	1	13	CR	GS	-	=	M	]	m	}
			1	1	1	0	14	SO	RS	.	>	N	^	n	~
			1	1	1	1	15	SI	US	/	?	O	_	o	DEL

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2 การอินเตอร์เฟสเพื่อแสดงผลด้วย LED (LED display interface)

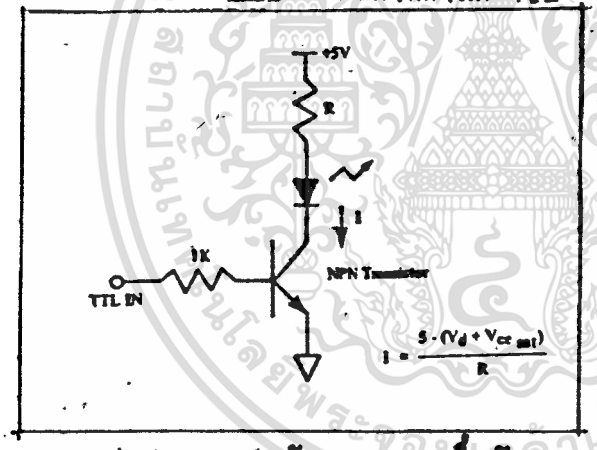
การอินเตอร์เฟสเพื่อแสดงผลด้วย LED เป็นอีกวิธีที่นิยมใช้อย่างแพร่หลายโดยเฉพาะอย่างยิ่งเครื่องคอมพิวเตอร์ขนาดเล็กที่แสดงผลเป็นตัวเลขจำนวนไม่กี่หลัก เนื่องจาก LED มีราคาถูก สิ่งที่แสดงได้แก่แอดเดรส ข้อมูลและสเตตัส การแสดงผลมีหลายรูปแบบดังนี้

1) LED แบบที่เปล่งแสงจุดเดียว (Single LED)

2) LED แบบที่เปล่งแสงเจ็ดส่วน (Seven segment LED)

3) LED แบบที่เปล่งแสงหลาย ๆ จุด (Dot-matrix LED)

2.2.1 LED แบบที่เปล่งแสงจุดเดียว แสงที่เปล่งมีลักษณะเป็นจุดเล็ก ๆ เพียงจุดเดียว เหมาะที่จะใช้แสดงค่าข้อมูลที่เป็นตัวเลขฐานสองและสเตตัสต่าง ๆ โดยทั่ว ๆ ไป LED แบบนี้จะมีแสงสีแดง สีเขียว หรือสีส้ม การออกแบบง่ายกว่าแบบอื่น ๆ ดังแสดงในรูป แรงดันตกคร่อม LED มีขนาดตั้งแต่ 1.2 – 2.4 โวลท์ ขึ้นอยู่กับชนิดของ LED



รูปแสดง วงจรขับ LED หนึ่งตัว

ความสว่างของ LED ขึ้นอยู่กับค่ากระแสที่ไหลผ่าน (I) โดยที่

$$I = \frac{V_{cc} - (V_d + V_{ce\ sat})}{R}$$

จากรูปที่แสดง $V_{cc} = 5$

$$I = \frac{5 - (V_d + V_{ce\ sat})}{R} = \frac{3.5}{R}$$

โดยทั่ว ๆ ไปกระแส I มีค่า 2 – 20 mA

ปกติ $V_{BE\ (Active)}$ ของทรานซิสเตอร์ชนิดซิลิกอนมีค่า 0.6 โวลท์ ถ้าแรงดันขาเข้า (V_{BE}) มีค่ามากกว่า 0.6 โวลท์ ทรานซิสเตอร์จะ ON ทำให้ LED เปล่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แสง แต่ถ้าแรงดันขาเข้ามีค่าต่ำกว่า 0.6 โวลท์ ทรานซิสเตอร์จะ OFF ทำให้ LED ไม่เปล่งแสง $V_{ce\ sat}$ ของทรานซิสเตอร์ชนิดซิลิกอนมีค่าประมาณ 0.3 โวลท์

สมมุติ V_d = 1.2 โวลท์ LED เปล่งแสงเมื่อ I = 15 mA และให้ V_{cc} = 5 โวลท์ เราต้องการกำหนดค่าความต้านทาน R

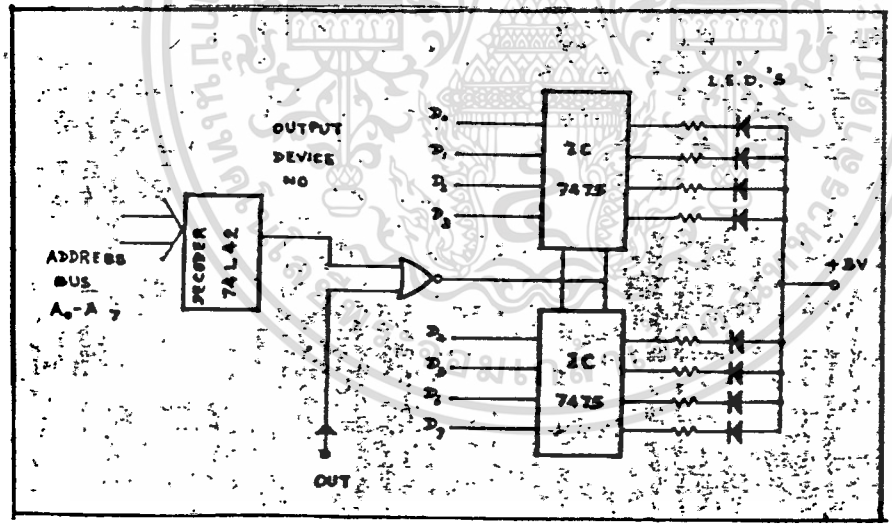
จาก $$I = \frac{V_{cc}-(V_d + V_{ce\ sat})}{R}$$

$$R = \frac{5-(1.2 + 0.3)}{15 \times 10^{-3}} \text{ Ohms}$$

$$= 233 \text{ Ohms}$$

ในทางปฏิบัติอาจใช้ 220 หรือ 270 โอห์มก็ได้

ในบางกรณีอาจไม่ต้องใช้ตัวทรานซิสเตอร์เป็นตัวช่วยขับ (driver) ถ้าหาก IC ตัวที่ใช้สามารถจ่ายกระแสให้แก่ LED ได้เพียงพอดังแสดงในรูป ซึ่งเป็นตัวอย่างการต่อวงจรระหว่าง IC 7457 กับ LED



รูปแสดง LED interface

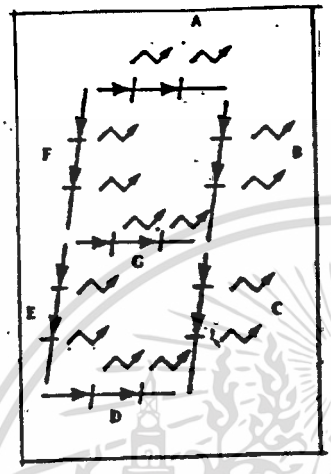
จากรูปจะเห็นว่าการแสดงผลด้วย LED เริ่มจากสัญญาณ OUT (เกิดขึ้นทุกครั้งที่มีคำสั่ง OUT) ถูกนำมา NOR กับสัญญาณ OUT PUT DEVICE NO (ซึ่งถูกถอดรหัสจากบัสแอดเดรส) ให้เป็นสัญญาณควบคุม IC 7475 ให้รับข้อมูลเข้าข้อมูลจากบัสข้อมูล (D_0-D_7) จะถูกนำออกไปแสดงที่ LED พร้อมกับแลทช์ไว้จนกว่าจะมีข้อมูลใหม่เข้ามา

2.2.2 <u>LED แบบเปล่งแสงเจ็ดส่วน</u> (SEVEN SEGMENT LED)

LED แบบนี้ประกอบด้วย LED แบบตัวเดียวหลาย ๆ ตัว จัดรูปให้มีลักษณะคล้าย

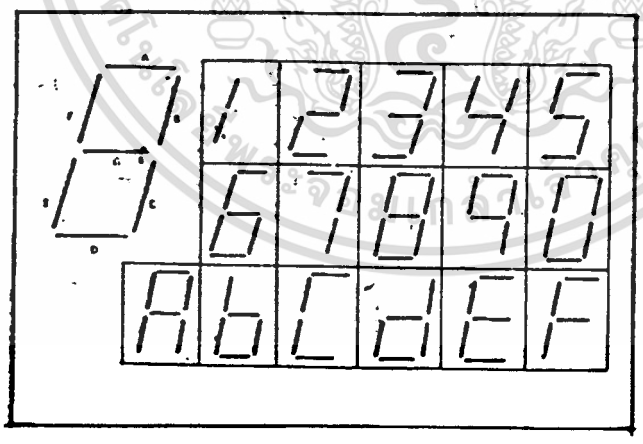
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กับเลข 8 ซึ่งจะมีอยู่ 7 ส่วนดังแสดงในรูปแต่ละส่วนจะมีชื่อกำกับเช่นส่วน A, B, C, D, E, F และ G เราสามารถกำหนดให้ส่วนใดส่วนหนึ่งเปล่งแสงได้ โดยมากมักจะให้เปล่งแสงเป็นตัวเลข 0 – 9 และตัวอักษร A – F นิยมใช้กันมากในเครื่องคอมพิวเตอร์ขนาดเล็กเพื่อใช้แสดงค่าแอดเดรสและค่าข้อมูลซึ่งเป็นตัวเลขฐานสิบหก



รูปแสดง Seven segment LED

โดยปกติการอินเตอร์เฟสกับ LED แบบนี้จะใช้ร่วมกับ BCD to seven segment decoder/driver ซึ่งทำหน้าที่เปลี่ยน 4 บิท BCD ให้แสดงผลเป็นตัวเลขและตัวอักษรดังแสดงในรูป

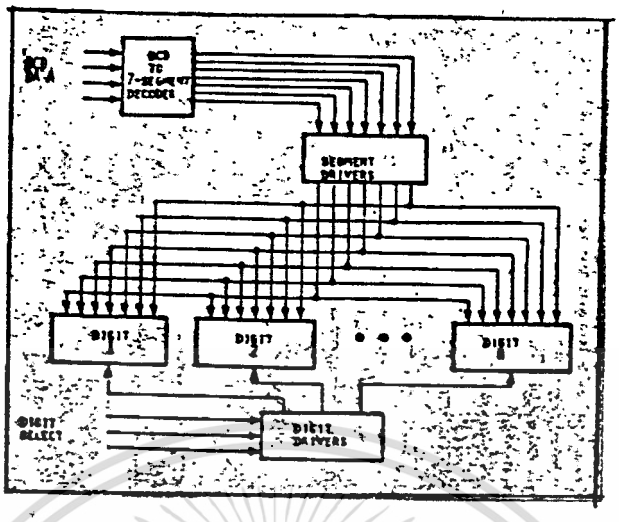


รูปแสดง Seven Segment charactors

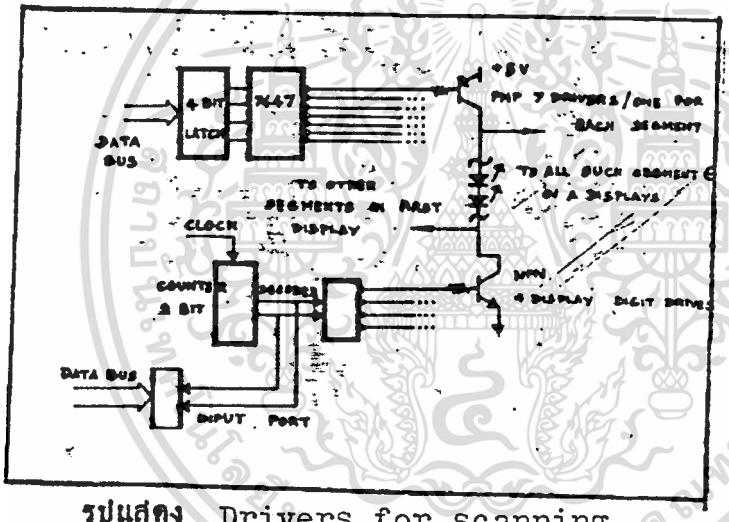
ไอซีเหล่านี้มักมีทรานซิสเตอร์ทำหน้าที่เป็นตัวขับอยู่ภายในแล้ว ดังนั้นจึงไม่จำเป็นต้องต่อทรานซิสเตอร์เติมเติมอีก

กรณีที่ต้องการที่จะแสดงผลเป็นตัวเลขหลายตัว เพื่อการประหยัด ไอซีที่ทำหน้าที่ถอดรหัสอาจใช้วิธีมัลติเพลกซ์ (multiplex) คือทำให้ LED แต่ละหลักสว่างในช่วงสั้น ๆ กลับกันไป การมัลติเพลกซ์อาจทำได้หลายวิธี ในที่นี้จะขอยกตัวอย่างได้เพียง 2 วิธี ดังแสดงในรูปทั้งสองรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง Multiplexing LED's (SCAN DIGIT AND DATA)



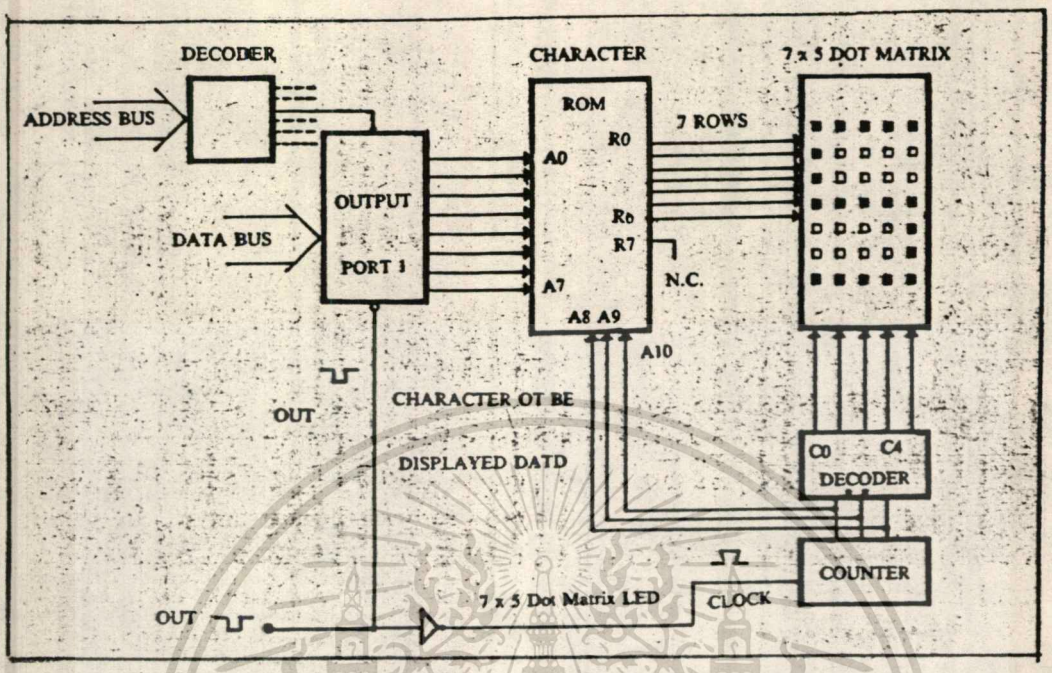
รูปแสดง Drivers for scanning

เนื่องจาก LED ไม่ได้เปล่งแสงตลอดเวลา ถ้าจำนวนเลขทั้งหมด N หลัก (LED N ตัว) ดังรูปที่แสดงข้างบน LED จะเปล่งแสงเป็นเวลาเพียง $\frac{1}{N}$ เท่าเมื่อเทียบกับตอนที่ไม่ได้มัลติเพลกซ์ LED สว่างตลอดเวลา) ตาเราจะมองเห็น LED สว่างน้อยลง จึงจำเป็นต้องเติมกระแสเป็น N เท่า โดยปกติ IC ทั่ว ๆ ไปจะทนกระแสที่เติมขึ้นไม่ได้ จำเป็นต้องใช้ทรานซิสเตอร์เป็นตัวขับดังแสดงในรูปข้างบน

2.2.3 <u>LED แบบเปล่งแสงหลายจุด</u> (Dot - MATRIX LED)

LED แบบสุดท้ายนี้ประกอบด้วย LED แบบจุดเดียวจัดให้อยู่ในลักษณะแบบแมทริกส์ (MATRIX) เพื่อจะแสดงผลเป็นตัวอักษรหรือเครื่องหมายที่ต้องการ ในท้องตลาดทั่ว ๆ ไป เป็นแบบ 7x5 หรือ 7x9 (ตัวเลขตัวแรกแสดงจำนวนแถว และตัวที่สองแสดงจำนวนคอลัม) เราสามารถควบคุมให้แสดงผลเป็นตัวเลขหรือตัวอักษรได้ ดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง 7x5 Dot matrix LED

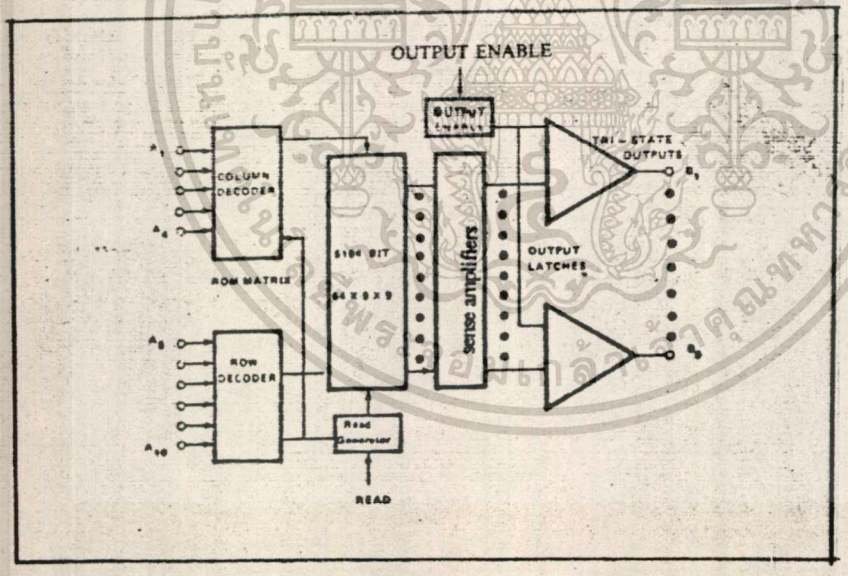
สำหรับรูปที่แสดงข้างบนนั้นจะใช้ตัวนับ (COUNTER) นับตั้งแต่ 0-4 คือเมื่อนำไปถอดรหัสโดยผ่านวงจรผ่านถอดรหัสแล้ว จะเลือกได้ 5 คอลัม ขณะเดียวกันจะต่อไปเข้าแอดเดรส A_{10}, A_9, A_8 ของ ROM ด้วย สำหรับสัญญาณที่ออกจาก ROM ซึ่งเป็นข้อมูลในแต่ละแอดเดรสจะเป็นตัวเลือกทางด้านแถวที่จะเลือกได้ 7 แถว

เมื่อต้องการจะแสดงตัวเลข 5 จะมีคำสั่ง OUT ที่พอร์ท 1 โดยเริ่มนำเอาข้อมูลเลข 5 จากบัสข้อมูลที่จะไปเป็นอินพุทให้กับแอดเดรส A_0 - A_7 ของ ROM และที่แอดเดรสที่ ROM จะถูกโปรแกรมด้านข้อมูล 11111001 ไว้ จะสังเกตเห็นว่าขาเอาท์พุท R_7 เราไม่ใช้ (N.C.) เนื่องจากว่าใช้ข้อมูลเพียง 7 บิทเท่านั้น (ในการเลือกทางด้านแถวซึ่งมีแค่ 7 แถว) ข้อมูล 1111001 (ซึ่งมี 7 บิท) จะถูกนำไปควบคุม LED ซึ่งอยู่ทางด้านแถว ขณะเดียวกันเราจะนำพัลส์ OUT ไปทำหน้าที่เป็นสัญญาณนาฬิกาให้แก่ตัวนับด้วย ขณะนี้ตัวนับจะนับ 000 ซึ่งจะถูกถอดรหัสให้ไปเลือกคอลัมที่ 0 ของLED ทำให้ LED ที่อยู่ในคอลัมที่ 0 เท่านั้นที่จะสว่างหรือมืด (ขึ้นอยู่กับค่าข้อมูลที่เข้ามาทางด้านแถว) นอกจากนั้นเอาท์พุทจากตัวนับยังเป็นตัวกำหนดแอดเดรสให้แก่ ROM ด้วยคือต่อเข้าที่ขา A_{10}, A_9, A_8 (มีค่าเป็น 000) ข้อมูล 1111001 ที่ออกมาควบคุมทางด้านแถวจะอยู่ที่แอดเดรส (000,0000, 0101) ของ CHARACTER ROM

เวลาต่อมาจะมีคำสั่ง OUT ข้อมูลเลข 5 อีกที่พอร์ท 1 ค่า A_0 - A_7 ของ ROM

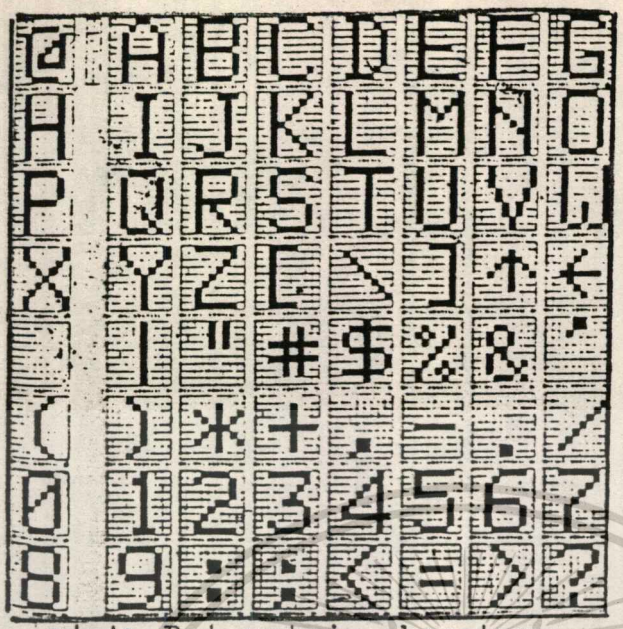
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้นำไปดัดแปลง เผยแพร่ หรือนำไปฝากไว้ในสื่ออื่นๆ

ก็ยังมีค่าคงเดิมคือ (0000,0101) แต่ขณะเดียวกันพัลส์ OUT ที่เกิดครั้งที่ 2 นี้จะทำให้ตัวนับนับเพิ่มขึ้นอีก 1 คือเป็น (001) เมื่อนำไปถอดรหัสจะเปลี่ยนไปเลือกคอลัมที่ 1 ขณะเดียวกัน A_{10}, A_9, A_8 เปลี่ยนเป็น (001) ด้วยข้อมูลใน ROM ที่แอดเดรส (001,0000, 0101) จะเป็น 1001001 (ตัด R_7 ทิ้ง) ซึ่งจะออกจาก ROM ไปควบคุมทางด้านแถว การทำงานจะเป็นลักษณะนี้จนกระทั่งถึงคำสั่ง OUT ครั้งที่ 5 คือ ครั้งสุดท้ายของการแสดงผล (5 คอลัม) ขณะนั้นแอดเดรสจะเป็น (100,0000,0101) ข้อมูลที่ออกจาก ROM เป็น 1001111 วงจรนี้จะ Loop แสดงผลเลข 5 จนกว่าสัญญาณจากบัสข้อมูลเปลี่ยนเป็นเลขอื่น วิธีนี้ใช้เทคนิคของซอฟแวร์ช่วยในการแสดงผล สำหรับรูปที่แสดง ก็คล้าย ๆ กันเพียงแต่เราใช้เอาท์พุทพอร์ทเพิ่มขึ้นอีก เพื่อทำหน้าที่เป็นตัวนับ หลักการที่กล่าวมานี้สามารถนำไปใช้กับการแสดงผลบนจอภาพทีวีได้ดังแสดงในรูปถัดไป และเรายังจะสามารถแสดงผลเป็นรหัสแบบ ASCII หรือ EBCDIC อย่างใดก็ได้ขึ้นอยู่กับโปรแกรมข้อมูลใน ROM ลักษณะของ DOT matrix



รูปแสดง Dot matrix ROM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

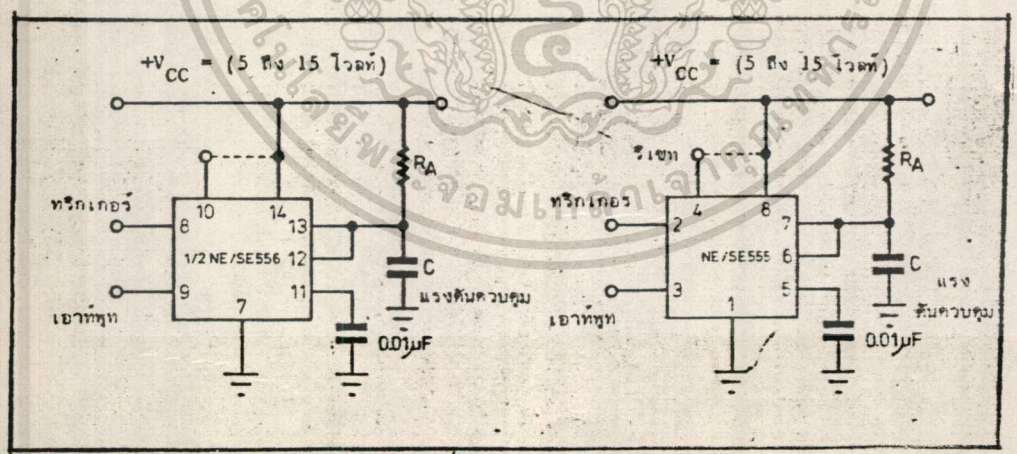


รูปแสดง Dot matrix chracters

2.3 การใช้งานไอซี 555

2.3.1 การใช้เป็นวงจรโมโนสเตเบิ้ล

วงจรแบบง่ายของวงจรโมโนสเตเบิ้ลมัลติไวเบรเตอร์เป็นดังรูปที่แสดง ซึ่งสามารถนำไปประยุกต์ ใช้งานในวงจรเครื่องจับการเต้นของหัวใจ เครื่องหารความถี่ การมอดูเลทด้วยพัลส์ หรือในเครื่องจ่ายไฟตรงชนิดสวิทชิ่งเรกูเลเตอร์



รูปแสดง วงจรโมโนสเตเบิ้ล

การทำงานของวงจรกล่าวโดยย่อได้ดังนี้ เมื่อให้วงจรทำงานในหน้าที่เป็นวงจรโมโนสเตเบิ้ล เราจะต้องต่อตัวต้านทานหนึ่งตัวและตัวเก็บประจุ ในการกำหนดเวลาของพัลส์ที่เอาท์พุท

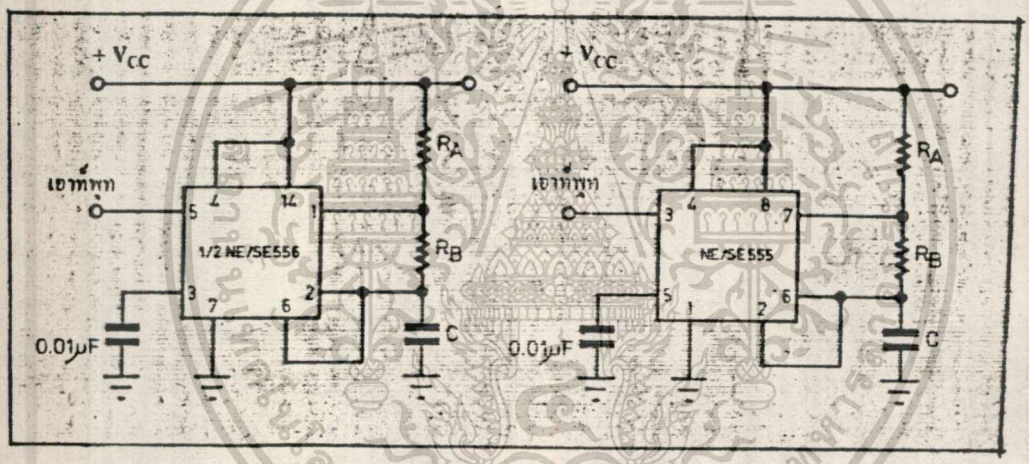
เมื่อพิจารณาจากรูปที่แสดง ตัวเก็บประจุ C ที่ต่อภายนอกจะคายประจุผ่านทรานซิสเตอร์ ในขณะที่ตัวทรานซิสเตอร์อยู่ในสภาวะอิ่มตัว เมื่อป้อนสัญญาณทริกเกอร์เข้าที่ขาทริกเกอร์

ของไอซีจะทริกฟลิบฟลอบ พัลส์ของสัญญาณทริกเกอร์จะต้องเป็นพัลส์ลบ โดยจะป้อนผ่านตัวเปรียบเทียบ หลังจากทริกพัลส์ผ่านไปแล้วฟลิบฟลอบจะทำให้ทรานซิสเตอร์ลัดวงจรตัวเก็บประจุ C เอาท์พุทจะออกที่ขาเอาท์พุทของไอซี ในขณะที่ C คายประจุ แรงดันจะลดลง เมื่อแรงดันลดลงถึงค่าหนึ่งคือลดต่ำลงต่ำกว่า 1/3 ของ V_{CC} ก็จะไปทริกฟลิบฟลอบทำให้เอาท์พุทเปลี่ยนเป็นพัลส์ได้ ดังนั้นเอาท์พุทพัลส์จะไม่ขึ้นอยู่กับรูปร่างของอินพุท แต่จะถูกควบคุมโดยค่าคงตัวเวลา RC ซึ่งช่วงเวลาของพัลส์เอาท์พุท T จะมีค่าความกว้างพัลส์ประมาณ

$T = 1.1\ RC$

2.3.1.1 วงจรอะสเตเบิ้ล

ในวงจรอะสเตเบิ้ลมัลติไวเบรเตอร์ แต่เมื่อใช้ไอซี 555 การสร้างวงจรย่อมสะดวกและง่ายต่อการใช้งานมากกว่า ดังนั้นจึงมีผู้นิยมใช้กันมาก



รูปแสดง วงจรอะสเตเบิ้ลมัลติไวเบรเตอร์

การทำงานของวงจรอธิบายได้จากวงจรดังแสดงในรูป ตัวเก็บประจุภายนอก C จะรับประจุจนมีค่าแรงดันคร่อมตัวมัน 2/3 V_{CC} โดยการประจุผ่านตัวต้านทาน R_A และ R_B มันจะคายประจุทันที จนกระทั่งมีแรงดันตกคร่อม C เหลือเพียง 1/3 V_{CC} โดยการคายประจุผ่านตัวต้านทาน R_B นั่นคือแรงดันคร่อมตัวประจุจะออสซิลเลทอยู่ระหว่าง 1/3 V_{CC} และ 2/3 V_{CC} สัญญาณเอาท์พุทจะอยู่ในสภาวะ "0" เมื่อตัวเก็บประจุคายประจุ การคายและรับประจุจะไม่ขึ้นกับแรงดันจ่ายไฟเลี้ยงแต่จะขึ้นอยู่กับค่า R และ C เราอาจจะทำให้วงจรออสซิลเลทหรือหยุดออสซิลเลท ด้วยสวิทช์ที่ต่อที่ขา 6 กับขา 14

ช่วงเวลาของการที่ตัวเก็บประจุรับประจุหรือเอาท์พุทอยู่ในสภาวะ "1" หาได้โดย

$$t_1 = 0.695(R_A + R_B)C$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ช่วงเวลาของการคายประจุของตัวเก็บหรือเอาท์พุทอยู่ในสภาวะ "0" หาได้โดย

$$T = 0.695\ R_B C$$

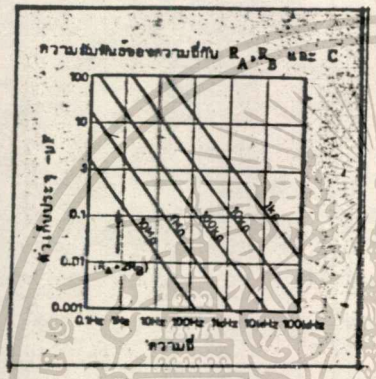
คาบเวลาหาได้โดย

$$T = t_1 + t_2 = 0.695(R_A + 2R_B)C$$

ความถี่ของการออสซิลเลทหาได้จาก

$$f = \frac{1 \times 1.44}{T(R_A + 2R_B)}$$

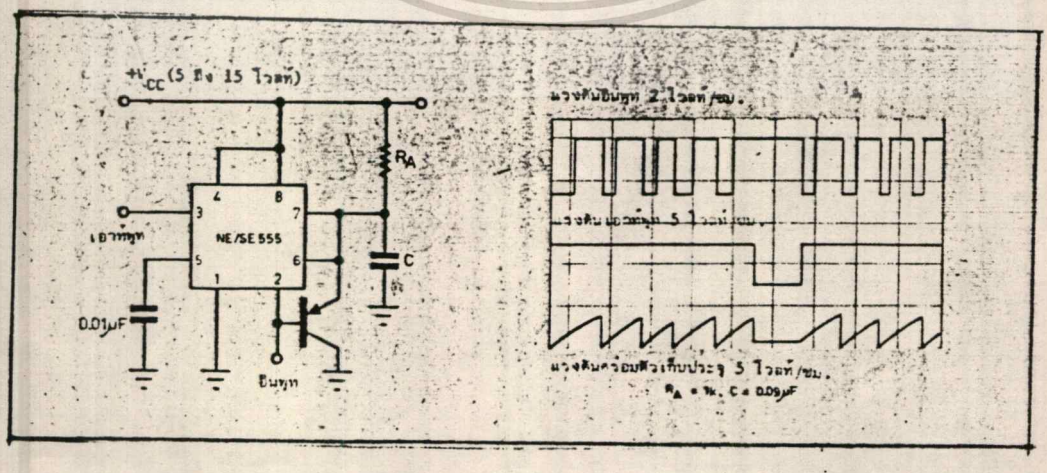
ในการหาความถี่นี้สามารถหาได้โดยตรงจากกราฟรูปที่แสดง



รูปแสดง การหาความถี่จากกราฟ

2.3.1.2 <u>วงจรหาพัลส์ที่ขาดหายไป</u>

เมื่อต่ไอซี 555 ดังแสดงในรูป จะได้วงจรที่สามารถตรวจจับพัลส์ที่ขาดหายไปหรือพัลส์ที่ผิดปกติ ซึ่งมีคาบยาวผิดปกติหรือสั้นผิดปกติระหว่างสองพัลส์ที่ติดกัน ดังนั้นวงจรนี้จึงสามารถนำมาใช้ในการตรวจการจัดระเบียบของหัวเทียนในเครื่องยนต์ หรือเครื่องเตือนการเต้นของหัวใจของผู้ป่วยเป็นต้น



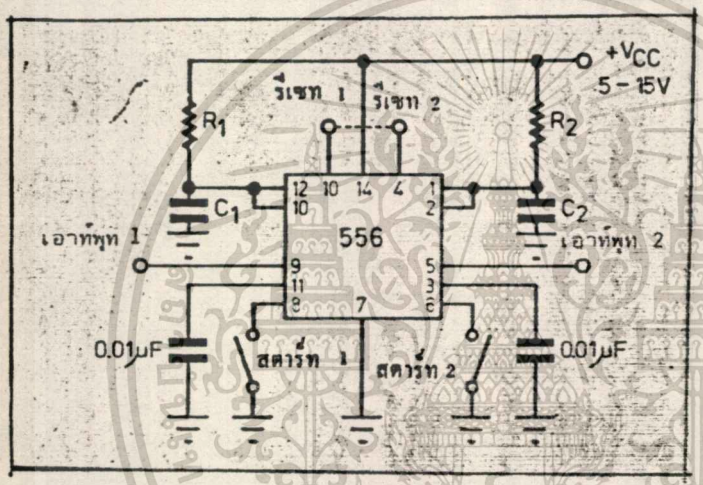
เมื่อป้อนพัลส์ที่ต่อเนื่องและตั้งมาตรฐานเวลาในการดีเทคไว้ เมื่อพัลส์อินพุทมีช่วงห่างระหว่างพัลส์มีค่าน้อยกว่ามาตรฐานที่ตั้งไว้จากพัลส์โดยเฉลี่ยของอินพุทเป็นผลทำให้ช่วงห่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระหว่างพัลส์ของอินพุทมีค่ามากขึ้นจะเป็นผลทำให้ระดับเอาท์พุทสูงขึ้นซึ่งจะใช้ไปทริกอุปกรณ์อื่น ๆ เพื่อเป็นเครื่องเตือนหรือให้วงจรอื่นทำงานตามต้องการ

2.3.2 <u>การใช้เป็นวงจรตั้งเวลาสองตัวที่แยกจากกัน</u>

ในวงจรไอซี 556 เป็นวงจรที่ประกอบด้วยวงจรตั้งเวลาสองตัวที่แยกจากกัน ดังนั้นเราสามารถสร้างวงจรตั้งเวลาสองวงจรที่แยกจากกัน ซึ่งอาจจะมีสัญญาณทริกเกอร์ร่วมกันหรือทริกเกอร์แยกจากกันก็ได้ ดังรูปที่แสดง เป็นวงจรตั้งเวลาสองวงจรที่แยกจากกัน ช่วงเวลาของการตั้งเวลาแต่ละตัวคือค่า T_1 และ T_2 สามารถคำนวณหาได้เช่นเดียวกับวงจรโมโนสเตเบิ้ลดังในรูปที่แสดงนั่นเอง หรือ $T_1 = 1.1\ R_1C_1$ และ $T_2 = R_2C_2$



รูปแสดง วงจรตั้งเวลาสองตัวที่แยกจากกัน

2.3.2.1 <u>การตั้งเวลาเป็นลำดับ</u>

ในการประยุกต์ใช้งานของวงจรไอซี 556 ส่วนของไอซีครึ่งหนึ่ง (วงจรตั้งเวลาหนึ่งตัว) เมื่อป้อนสัญญาณทริกเกอร์ให้กับไอซีส่วนแรกที่ขา 6 จะเป็นเหตุให้เอาท์พุทที่ขา 1 ของไอซีอยู่ในสภาวะ "1" เมื่อครบเวลา $T_1 = 1.1R_1C_1$ ในช่วงสุดท้ายของเวลา T_1 นี้ขา 1 จะเริ่มมีแรงดันลดลงเข้าสู่สภาวะ "0" ซึ่งจะเป็นผลทำให้เกิดสัญญาณทริกเกอร์ป้อนผ่านตัวเก็บประจุ C_1 ระหว่างขา 1 และ ขา 2 ซึ่งจะเป็นผลทำให้เอาท์พุทของวงจรอีกส่วนหนึ่งกลับสภาวะ มาอยู่สภาวะ "1" ซึ่งช่วงเวลาของการอยู่สภาวะ "1" ของขา 13 จะยาวเท่ากับ T_2 ซึ่งมีค่าเท่ากับ $1.1R_2C_2$

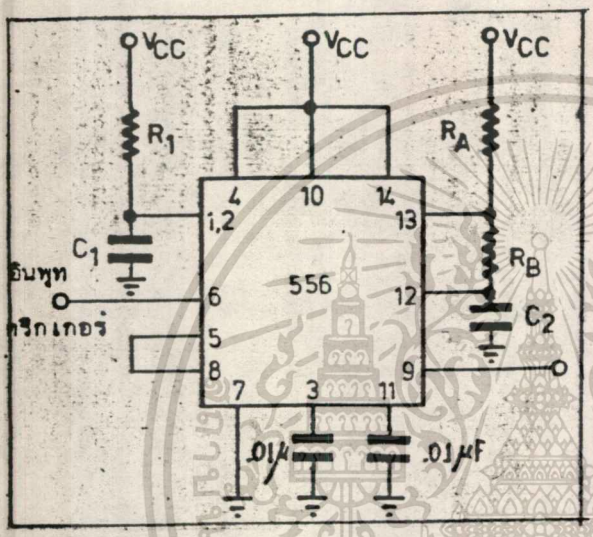
ในกรณีที่เราต้องการตั้งเวลาเป็นลำดับหลาย ๆ ขั้น เราก็อาจทำได้โดยการใช้ไอซีหลาย ๆ ตัวมาต่อร่วมกัน

2.3.2.2 <u>คีย์ออสซิลเลเตอร์</u>

ไอซี 556 ยังสามารถทำเป็นออสซิลเลเตอร์ในลักษณะพิเศษโดยให้สัญญาณออสซิลเลท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นห้วง ๆ หรือออสซิลเลทในช่วงเวลาจำกัดช่วงใดช่วงหนึ่ง โดยใช้วงจรตั้งเวลาส่วนหนึ่งในไอซี เป็นตัวกำหนดการออน (ON) และออฟ (OFF) วงจรที่ใช้งานแสดงให้เห็นดังรูปที่แสดง ส่วนของวงจรตั้งเวลาส่วนที่ 2 จะเป็นส่วนที่ใช้ในการสร้างสัญญาณ ค่าความถี่ของสัญญาณที่เกิดขึ้นจากส่วนนี้ สามารถปรับค่าได้ด้วยตัวต้านทาน $R_A R_B$ และตัวเก็บประจุ C_2 ตัวตั้งเวลาตัวแรกจะทำงานในหน้าที่เป็นวงจรโมโนสเตเบิ้ล และจะป้อนสัญญาณเอาท์พุทเข้าไปยังอินพุทของวงจรส่วนหลังที่ขา 8



รูปแสดง คีย์ออสซิลเลเตอร์

ในภาวะปกติเอาท์พุทของไอซีส่วนแรกจะอยู่ในสภาวะ "0" ดังนั้นตัวไอซีส่วนหลังจึงไม่สามารถที่จะทำการออสซิลเลทสัญญาณได้ และเมื่อใดก็ตามที่ป้อนสัญญาณทริกเกอร์ที่ขา 2 ของไอซี จะทำให้เอาท์พุทที่ขา 1 จะอยู่ในสภาวะ "1" และเริ่มทำให้วงจรตั้งเวลาส่วนหลังเกิดการออสซิลเลท ช่วงเวลาของการออสซิลเลทสามารถปรับได้ด้วยตัวต้านทาน R_1 และตัวเก็บประจุ C_1

2.3.2.3 <u>การหารความถี่</u>

ถ้าเราทราบความถี่อินพุทของสัญญาณ เราสามารถสร้างวงจรให้หารความถี่ด้วยค่าเท่าใดก็ได้ โดยการใช้วงจรเป็นวงจรโมโนสเตเบิ้ลแล้วปรับค่าช่วงเวลา $T_1 (1.1 R_1 C_1)$ โดยให้ค่าของ T_1 มีค่ามากกว่าค่าคาบเวลาของสัญญาณอินพุทที่ป้อนเข้ามาทางขาทริกเกอร์ นั่นคือพัลส์อินพุทที่อยู่ในช่วงคาบเวลา $1.1 R_1 C_1$ ไม่สามารถทริกวงจรให้ทำงานได้ พัลส์ที่จะทริกให้วงจรทำงานจึงเป็นพัลส์ตัวแรกเมื่อพ้นช่วงเวลา $1.1 R_1 C_1$

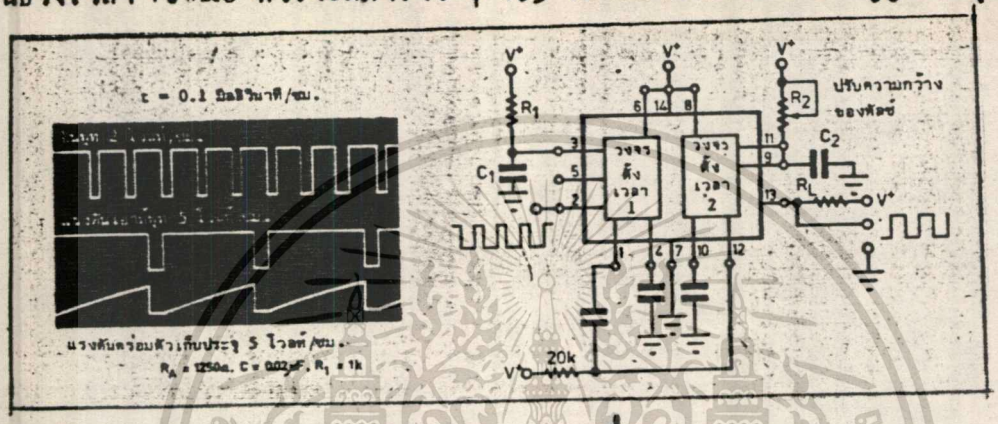
ความถี่ของสัญญาณเอาท์พุทจะมีค่าเท่ากับ $\frac{1}{N}$ เท่าของความถี่สัญญาณอินพุทค่า N ที่ได้จะต้องมีค่าอยู่ในช่วงดังนี้

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\left(\frac{T_1}{T_P} - 1\right) < N < \frac{T_1}{T_P}$$

เมื่อ T_P เป็นค่าคาบเวลาของสัญญาณอินพุท

จากรูปที่แสดง รูปลักษณะสัญญาณเมื่อใช้วงจรเป็นวงจรหารความถี่ด้วย 5 ในกรณีนี้ช่วงเวลา 1.1RC ควรจะมีค่าราว ๆ 4.5 เท่าของคาบเวลาของสัญญาณอินพุทนั่นเอง



รูปแสดง รูปสัญญาณของการหารความถี่

แต่เนื่องจากไอซี 556 ประกอบด้วยวงจรตั้งเวลา 2 ส่วนที่แยกจากกัน ดังนั้นในแต่ละส่วนสามารถใช้เป็นวงจรหารความถี่ได้ หรือในบางครั้งหลังจากที่เราหารความถี่ได้สัญญาณใหม่แล้ว แต่ช่วงคาบความกว้างของพัลส์อาจจะไม่สบอารมณ์เราได้ ดังนั้นเราอาจใช้ส่วนของวงจรตั้งเวลาอีกส่วนหนึ่งเป็นตัวจัดความกว้างของพัลส์ใหม่ได้ดังแสดงดังรูปข้างบน

2.3.2.4 <u>วงจรกำเนิดสัญญาณความถี่คงที่แต่ปรับความกว้างของพัลส์</u>

ในกรณีที่ให้วงจรตั้งเวลาชุดที่ 1 ทำงานเป็นวงจรอะสเตเบิ้ล และให้วงจรตั้งเวลาชุดที่ 2 ทำงานเป็นวงจรโมโนสเตเบิ้ล ดังแสดงในรูป ซึ่งจะเป็นผลทำให้วงจรสร้างสัญญาณที่มีความถี่คงที่ และสามารถปรับความกว้างของพัลส์ได้ตามใจชอบ

วงจรตั้งเวลาชุดที่ 1 จะสร้างสัญญาณรายคาบ ซึ่งเป็นสัญญาณทริกเกอร์ของวงจรตั้งเวลาชุดที่ 2 ถ้าให้ช่วงเวลา T_2 ของวงจรตั้งเวลาชุดที่ 2 ตัวปรับความกว้างของสัญญาณพัลส์ ซึ่งจะเป็นผลให้เอาท์พุทที่ขา 13 ของวงจรตั้งเวลาชุดที่ 2 มีความถี่เดียวกับความถี่ของเอาท์พุทของวงจรตั้งเวลาชุดที่ 1 ความกว้างของพัลส์สามารถปรับได้ในช่วงความกว้างราว 1% ถึง 99% โดยการปรับค่าความต้านทาน R_2

ความถี่ของสัญญาณที่เกิดขึ้นหาได้จาก

$$f = \frac{1.44}{(R_A + 2R_B)C_1}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

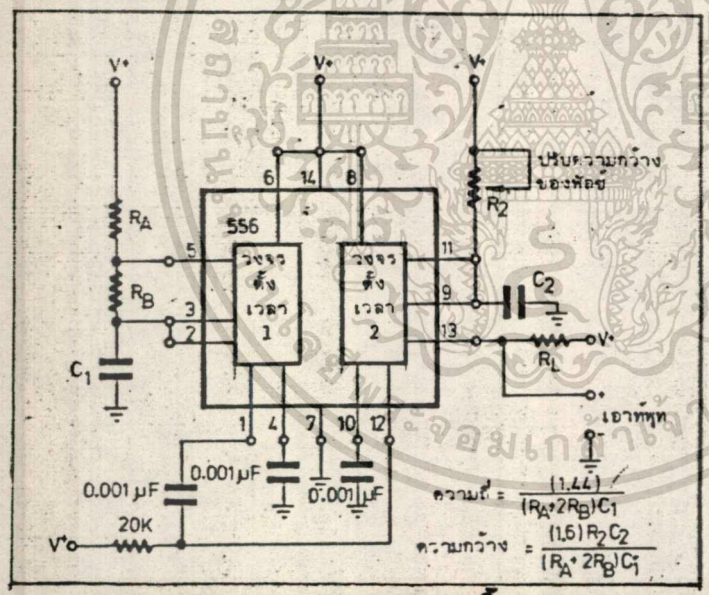
ความกว้างของพัลส์หาได้จาก

$$\frac{1.6\ R_2\ C_2}{(R_A + 2R_B)\ C_1}$$

ในกรณีที่ปรับค่า R_2C_2 จนกระทั่งความกว้างของพัลส์มีค่ามากกว่า 1 คาบเวลาของสัญญาณเอาท์พุทของวงจรตั้งเวลา ส่วนของเอาท์พุทของวงจรตั้งเวลาชุดที่ 2 จะเสมือนกับสร้างความถี่ในลักษณะที่มีความถี่น้อยกว่าความถี่อินพุท หรือวงจรตั้งเวลาชุดที่ 2 จะทำหน้าที่เป็นตัวหารความถี่นั่นเอง ดังนั้น

$$f_2 = \frac{f_1}{N}$$

เมื่อ N เป็นอัตราส่วนระหว่างความถี่ f_2/f_1 ซึ่งขึ้นอยู่กับค่าตัวต้านทาน R_2 และตัวเก็บประจุ C_2 นั่นเอง



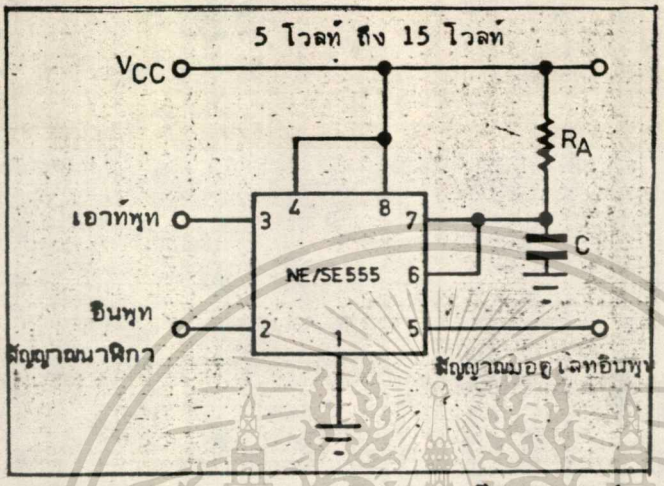
รูปแสดง วงจรโมโนสเตเบิ้ล

2.3.2.5 <u>การมอดูเลททางความกว้างของพัลส์</u>

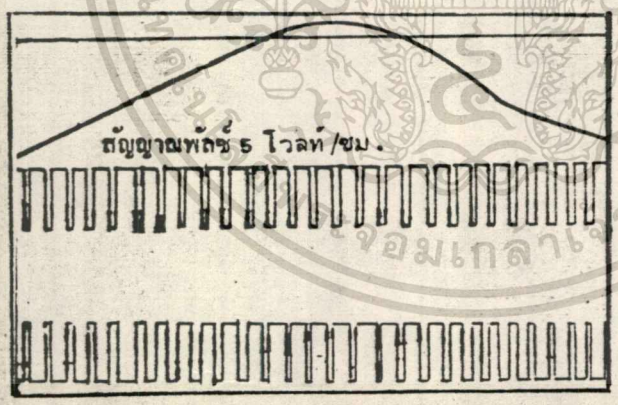
เมื่อต่อวงจรตั้งเวลาให้ทำหน้าที่เป็นวงจรโมโนสเตเบิ้ลดังรูปที่แสดง และถ้าเราป้อนสัญญาณพัลส์ที่ต้องการเข้าที่ขาทริกเกอร์ของวงจรตั้งเวลา ความกว้างของพัลส์ที่เอาท์พุทจะขึ้นอยู่กับค่า R และ C ที่ประกอบอยู่ภายนอกหรือกล่าวได้ว่า ความกว้างของพัลส์นี้จะเป็นช่วงเวลาของการประจุ ให้ตัวเก็บประจุจนมีค่าเพิ่มแรงดันคร่อมตัวมันจาก 1/3 VCC ถึง 2/3 VCC นั่นเอง ดังนั้นถ้าเราสามารถเปลี่ยนค่าเวลาของการประจุนี้ได้ ก็จะทำให้ความกว้างของพัลส์ที่เอาท์พุทเกิดการเปลี่ยนแปลงได้เช่นกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความถี่ของสัญญาณที่เอาท์พุทจะยังคงมีค่าเท่ากับความถี่ของสัญญาณที่มาหริกเกอร์อินพุทแต่ความกว้างของพัลส์จะเกิดการเปลี่ยนแปลงไป เนื่องจากการป้อนสัญญาณที่ต้องการนำมาม็อดูเลทไปยังขั้วแรงดันควบคุม (ขา 4 หรือขา 10) ซึ่งจะทำให้ช่วงเวลาหรือระดับการเติมแรงดัน (ประจุตัวเก็บประจุ) จนถึงการลดแรงดัน (คายตัวเก็บประจุ) เกิดการเปลี่ยนแปลง



รูปแสดง วงจรม็อดูเลทางความกว้างของพัลส์



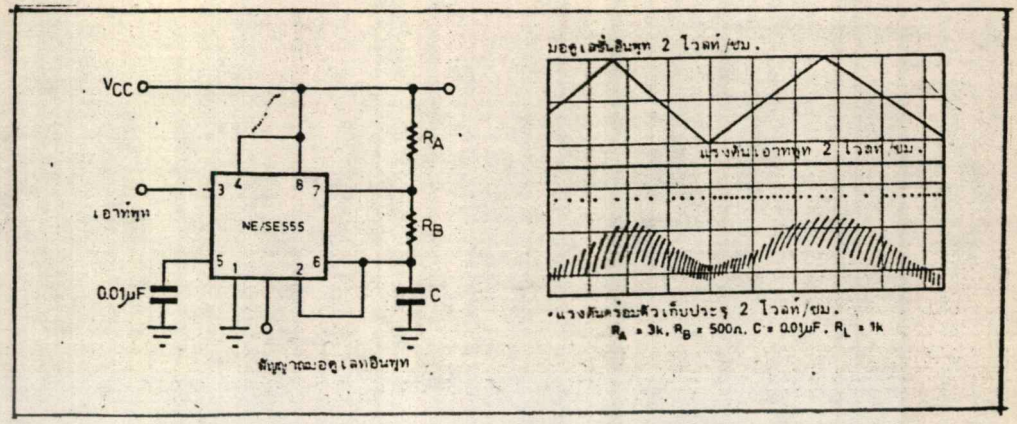
รูปแสดง รูปลักษณะของสัญญาณที่สำคัญ

โดยปกติสัญญาณอินพุทที่ใช้เป็นตัวหริกเกอร์นั้น โดยปกติก็มาจากสัญญาณเอาท์พุทของวงจรตั้งเวลาอีกตัวในไอซีนั่นเอง

2.3.2.6 <u>การม็อดูเลทางตำแหน่งของพัลส์</u>

เมื่อไอซีทำงานเป็นวงจรอะสเตเบิ้ล ช่วงคาบเวลาที่ขณะใดขณะหนึ่งของสัญญาณเอาท์พุทสามารถเปลี่ยนแปลงไปได้ โดยการป้อนแรงดันที่ต้องการนำมาม็อดูเลทเข้าที่ขั้วแรงดันควบคุม ดังแสดงในรูป ส่วนรูปถัดมาเป็นการแสดงความสัมพันธ์ของรูปคลื่นสัญญาณ

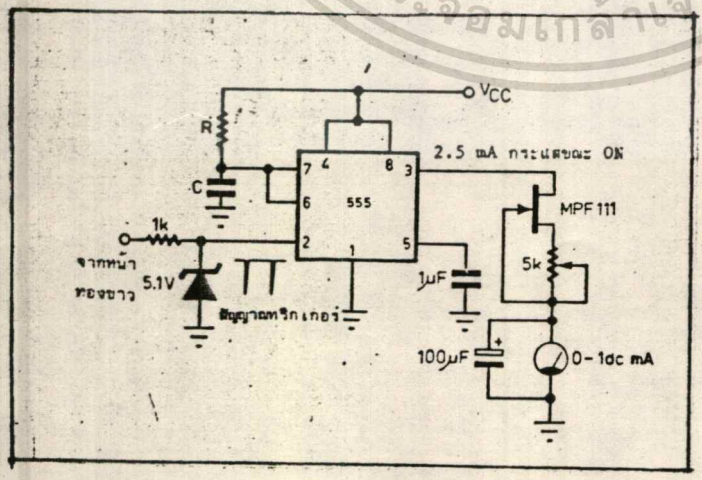
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า การแสวงหากำไร หรือการใช้งานในเชิงพาณิชย์ ไม่ว่ากรณีใดๆ ทั้งสิ้น



รูปแสดง การนำไอซีมามอดูเลทเข้าที่ขั้วแรงดันควบคุม และแสดงความสัมพันธ์ของรูปคลื่นสัญญาณ

2.3.3 ใช้ไอซี 556 ในวงจรวัดความเร็วของรถยนต์

จากรูปที่แสดง เป็นวงจรวัดความเร็วรอบหรือที่เรียกว่า แทคโคมิเตอร์ (tachometer) อย่างง่าย ๆ วงจรจะทำงานได้เนื่องจากพัลส์ที่เกิดขึ้น ขณะที่หน้าทอง - ขาวเปิด-ปิด ในขณะเครื่องยนต์ทำงานพัลส์เหล่านี้จะถูกจัดรูปร่างเสียใหม่ด้วยตัวต้านทาน 1 KΩ และซีเนอร์ไดโอด 5 โวลท์ จากนั้นจึงป้อนพัลส์เข้าที่ขาอินพุทของทริกเกอร์ของไอซี เป็นผลทำให้เอาท์พุทเปลี่ยนสภาวะมาเป็น "1" (high) โดยมีคาบเวลา 1.1RC กระแสที่เกิดจากพัลส์จะไหลผ่านมิเตอร์ ดังนั้นจำนวนกระแสที่ไหลผ่านมิเตอร์จึงแปรโดยตรงกับความถี่ของพัลส์นั่นเอง



รูปแสดง วงจรวัดความเร็วรอบ

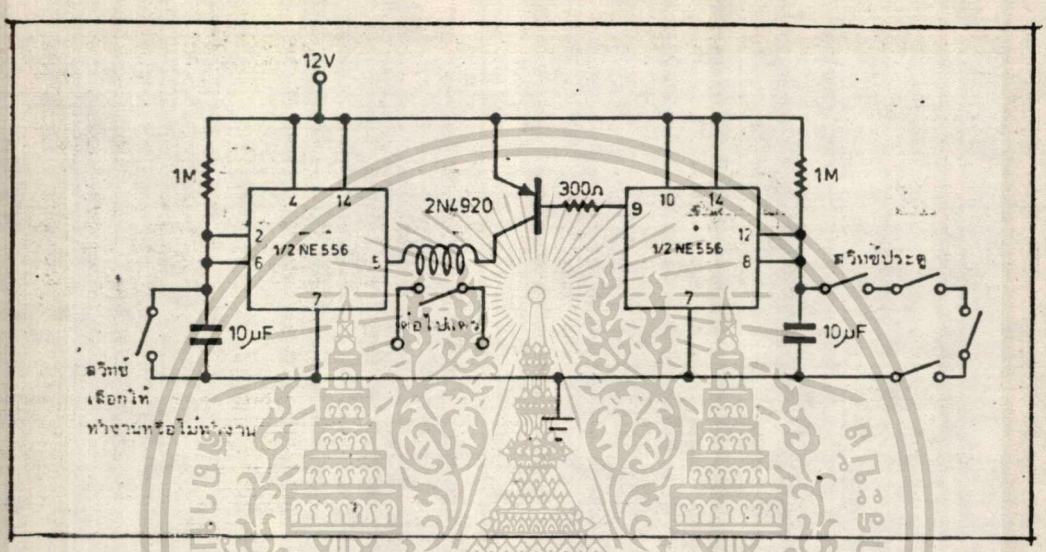
2.3.3.1 การทำงานของไอซีที่ไม่ใช่เป็นตัวตั้งเวลา

การทำงานของไอซี 556 ข้างต้นจะถูกใช้เป็นตัวโมโนสะเตเบิ้ลหรืออะสเตเบิ้ล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เท่านั้น แต่เรายังสามารถนำมาใช้ไอซี 556 , 555 ไปใช้ในวงจรอื่น ๆ ได้อีก คือ เป็นตัวควบคุมการทำงานของฟลิบฟลอบ ซึ่งวิธีการอยู่ที่ว่าเราจะต้องจัดให้แรงดันคร่อม C มีค่าเปลี่ยนจาก 1/3 V_{CC} กับ 2/3 V_{CC} นั่นเอง

ในการเปลี่ยนค่าแรงดันนี้ เราสามารถเปลี่ยนได้โดยการใช้ผลที่ได้มาจากแสง ความดัน ความชื้น หรืออุณหภูมิ โดยจะต่อเอาท์พุทกับขดลวดของรีเลย์ทำงานได้



รูปแสดง วงจรป้องกันขโมยที่ตั้งเวลาได้

2.4 ไอซีซีมอส

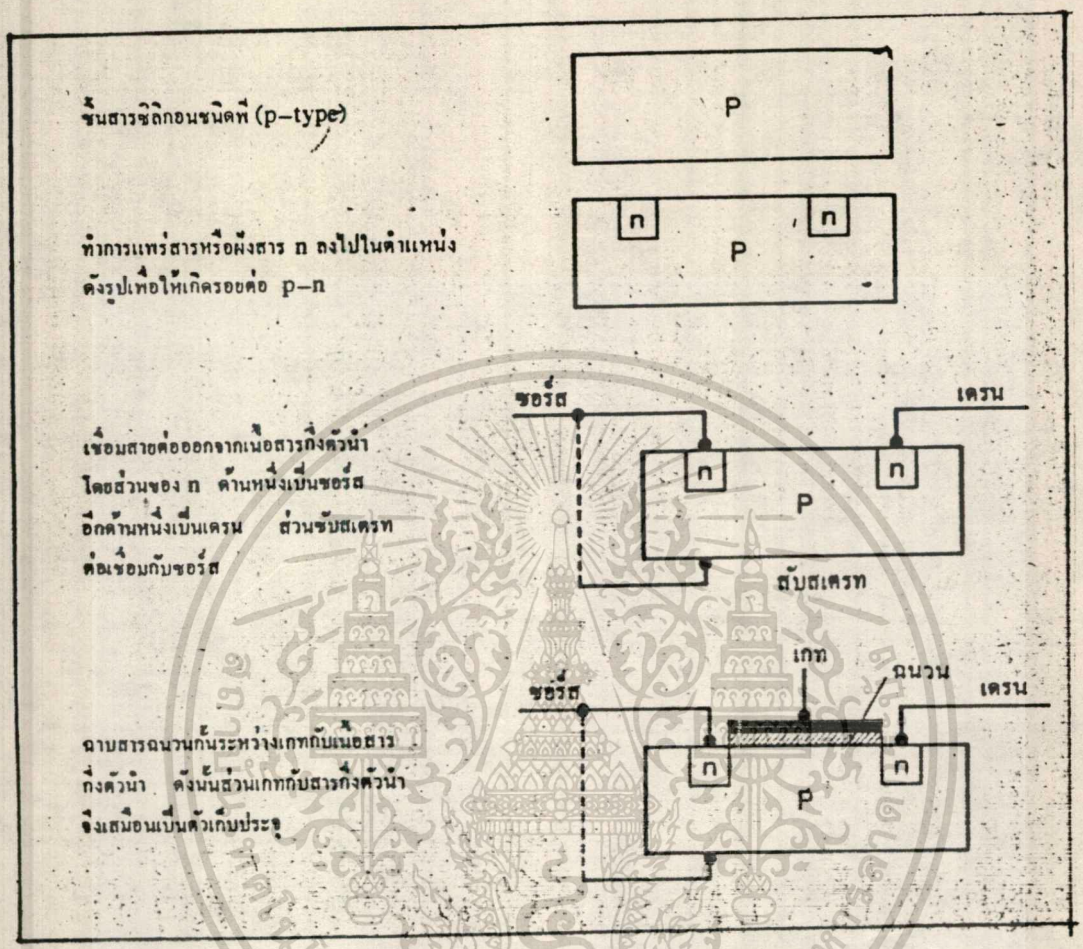
ตระกูลซีมอสเป็นตระกูลไอซีลอจิกตระกูลหนึ่งที่มีผู้ใช้กันมาก ซีมอส (CMOS) เป็นตัวย่อของคำว่า Complementary Metal-Oxide-Semiconductor ข้อดีของวงจรซีมอสเห็นจะได้แก่ราคาตัวไอซีถูก ใช้กำลังงานไฟน้อยใช้ไฟเลี้ยงได้สูง การสวิงของเอาท์พุทระหว่างลอจิก "0" และลอจิก "1" เป็นไปได้กว้าง จึงทำให้สัญญาณรบกวนจะสอดแทรกการทำงานได้ยาก

ไอซีที่เป็นซีมอสจะมีความลักษณะหนาแน่นของตัวอุปกรณ์ต่อพื้นที่สูง จึงทำให้มีฟังก์ชั่นการทำงานได้ยุ่งยากซับซ้อน เช่นไอซีจำพวก นาฬิกาดิจิตอล เครื่องคิดเลข ไมโครโปรเซสเซอร์ ฯลฯ

2.4.1 ลักษณะของมอสฟิลส์เอฟเฟคทรานซิสเตอร์

มอสฟิลด์เอฟเฟคทรานซิสเตอร์มีด้วยกันสองชนิดคือ พีมอส (PMOS) และเอ็นมอส (NMOS) และถ้าหากว่าวงจรที่ใช้งานเป็นวงจรที่มีทั้งพีมอส และเอ็นมอสอยู่ร่วมกันและเป็นคู่กันเราก็เรียกวงจรชนิดนั้นใหม่ว่าเป็นชนิด ซีมอส เรามาเข้าใจเกี่ยวกับตัวมอสกันให้ดีก่อน

ขอให้พิจารณาดังรูปที่แสดง



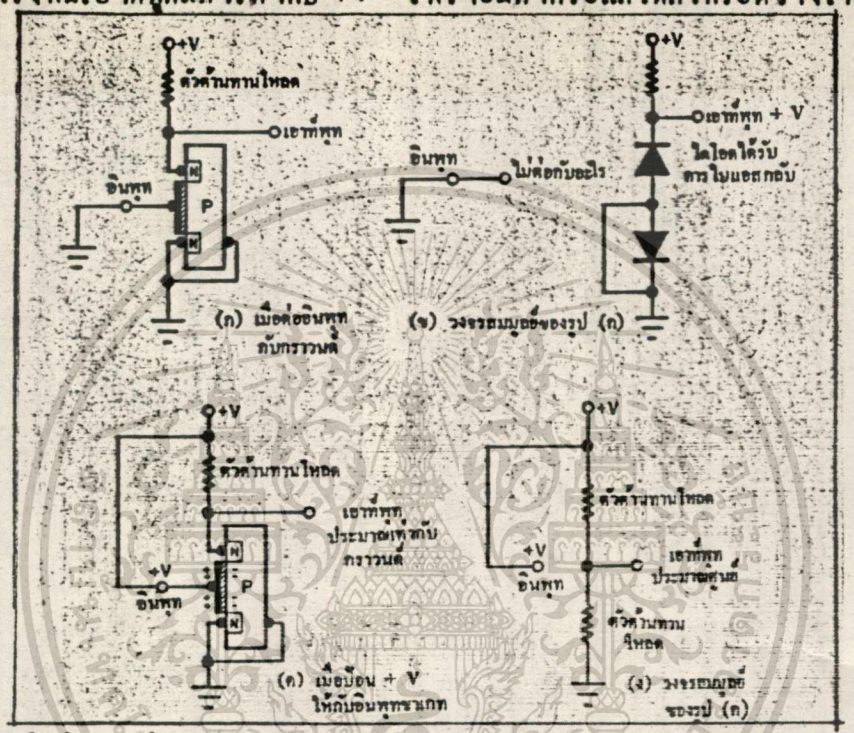
รูปแสดงโครงสร้างของเอ็นมอส

จากรูปที่แสดงนี้เป็นการแสดงโครงสร้างของเอ็นมอสชนิดเอ็นชานท์เมนท์(n-chann enchancement mode MOS) การสร้างตัวเอ็นมอสเริ่มจากนำชิ้นสารกึ่งตัวนำชนิด P ซึ่งโดยปกติจะมีโฮลอยู่มากกว่าอิเลคตรอนอิสระ ลำดับต่อไปเราจะทำการแพร่อะตอมของสาร n เข้าในตำแหน่งที่จะทำเป็นส่วนขั้วเดรนและซอส การทำให้เป็นส่วน n นี้ก็คือการทำให้บริเวณนั้นมีจำนวนอิเลคตรอนอิสระมากกว่าจำนวนโฮลนั้นเอง บริเวณส่วนต่อระหว่าง p กับ n จะเกิดเป็นรอยต่อที่เสมือนเป็นไดโอด ส่วนของสารกึ่งตัวนำชนิด n ทั้งสองแห่งนี้เราจะต่อขั้วออกภายนอก ส่วนหนึ่งจะเป็นขั้วเดรน อีกขั้วหนึ่งจะเป็นขั้วซอร์ด ส่วนฐานที่เรียกว่าซับสเตรทนั้นเราจะต่อเข้ายังขั้วซอร์ท

ลำดับสุดท้ายคือการใช้สาร SiO_2 ซึ่งเป็นฉนวนเคลือบผิวหน้าแล้วจึงประกอบด้วยตัวนำเป็นขั้วเกท ส่วนของฉนวนนี้ทำตัวเปรียบเสมือนสารไดอิเลคทริคที่ประกอบอยู่ในตัวเก็บประจุ ส่วนของเกทจึงมีฉนวนกั้นมิให้ต่อถึงเนื้อสารกึ่งตัวนำใด ๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อเข้าใจถึงโครงสร้างของ MOS ทรานซิสเตอร์ดีแล้วเราลองมาดูลักษณะการทำงานของวงจรดู ดังแสดงในรูป เป็นการนำเอาเอ็นมอสทรานซิสเตอร์เป็นสวิทช์ซึ่งลักษณะการทำงานจะเหมือนทรานซิสเตอร์ธรรมดามาก เมื่อเราให้แรงดันอินพุทมีค่าเป็นกราวนด์ ส่วนของเดรนดับซอร์สจะเสมือนต่อดับไดโอด ดังรูปที่แสดง (ข) นั่นคือไดโอดจะได้รับไบอัสกลับทำให้แรงดันเอาท์พุทมีค่าเท่ากับ +V เพราะมีค่ากระแสไหลได้ระหว่างเดรนกับซอร์ส



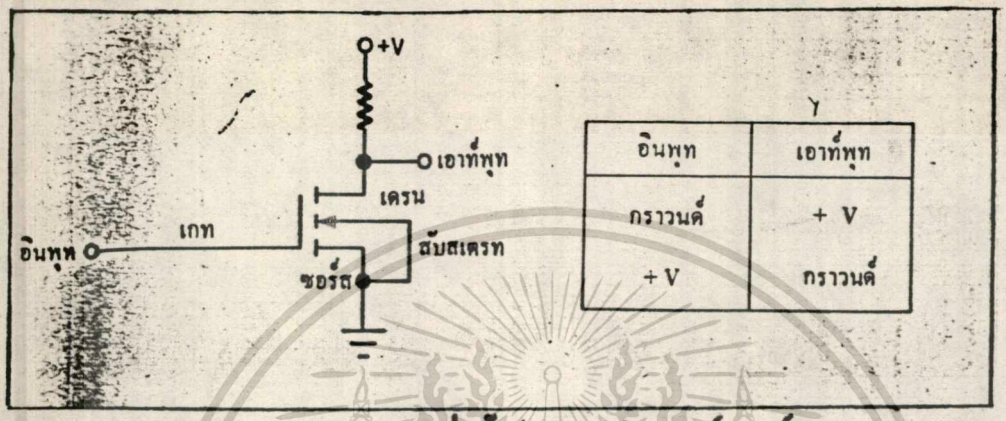
แต่เมื่อเราต่อแรงดันไฟ +V ให้กับเกทของมอสเฟทดังรูป (ค) ประจุบวกจะเหนี่ยวนำให้เกิดประจุลบในเนื้อสารซับสเตรทหรือสาร p ทำให้เกิดช่องต่อระหว่างเดรนกับซอร์ส การไหลของกระแสระหว่างเดรนกับซอร์สจึงเกิดขึ้นกระแสส่วนนี้จึงเสมือนเป็นกระแสอิ่มตัวของมอสเฟท วงจรสมมูลย์ของมอสเฟทในขณะนี้จะแทนได้ดังรูป (ค) ถ้าให้ค่าความต้านทานโหลดมีค่าสูงมาก ๆ แรงดัน +V ส่วนใหญ่จะไปตกคร่อมตัวต้านทานโหลดหมด แรงดันเอาท์พุทที่ได้จึงมีค่าประมาณเท่ากับกราวนด์

การทำงานของมอสเฟทชนิดเอ็นแชนแนล จึงคล้ายกับทรานซิสเตอร์ npn มาก คือเมื่อไบแอสที่เบสด้วยแรงดันไฟบวก ทรานซิสเตอร์จะอิ่มตัว แต่ข้อแตกต่างที่เห็นชัดก็คือในกรณีมอสเฟทไม่มีกระแสเกทเกิด ขึ้นเลยค่าความต้านทานอินพุทจึงมีค่าสูงมาก และในทำนองเดียวกันกระแสเดรนก็มีค่าได้น้อยมากจนแทบไม่มีกระแสไหลได้อีกเช่นเดียวกัน (ให้ R_L มีค่าสูงมาก)

สัญญลักษณ์ของมอสเฟทชนิดเอ็นแชนแนล (n-channel MOS-FET) ดังแสดงในรูป และถ้าเรายอนกลับไปดูโครงสร้างของมอสเฟทอีกครั้งหนึ่งจะเห็นว่าโครงสร้างมอสเฟท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

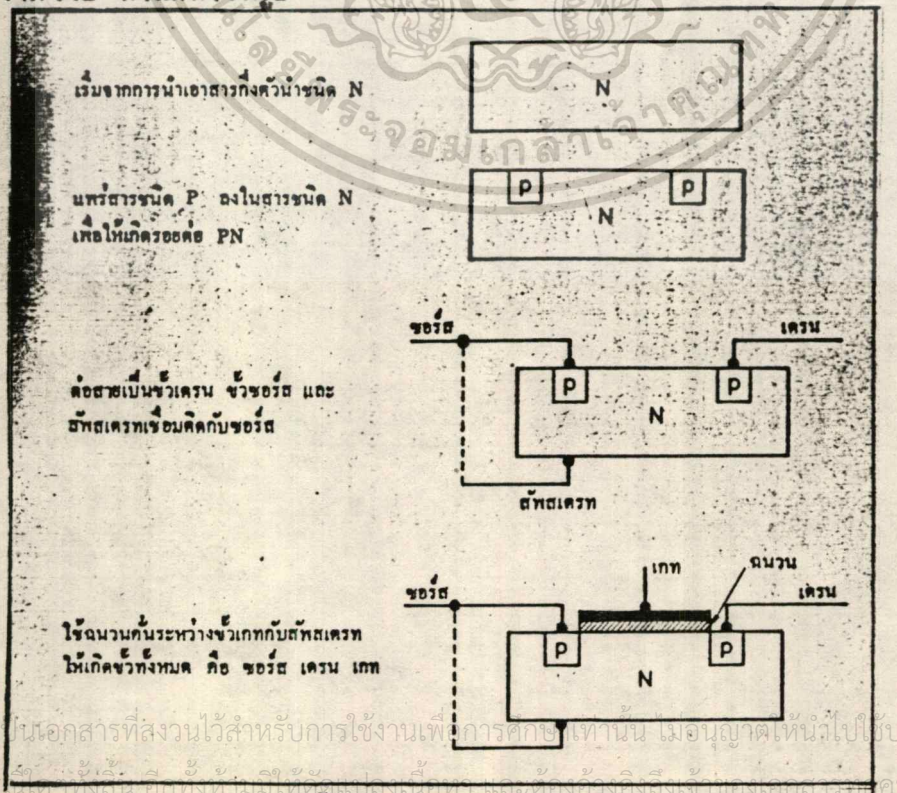
จะกระทัดรัดเล็กกว่าทรานซิสเตอร์ได้ การแพร่สารก็จะกระทำเพียงครั้งเดียว (ทรานซิสเตอร์กระทำสองครั้งคือเบสและอิมิสเตอร์) วงจรมอสเฟทจึงใช้ประกอบในวงจรไอซีที่มีฟังก์ชั่นการทำงานที่ซับซ้อนได้ดีมาก เช่นไอซีหน่วยความจำ (RAM) ไมโครโปรเซสเซอร์ เครื่องคิดเลข นาฬิกาดิจิตอล ฯลฯ



อินพุท	เอาท์พุท
กราวนด์	+ V
+ V	กราวนด์

รูปแสดง วงจรมอสเฟทที่ใช้เป็นวงจรอินเวอร์เตอร์

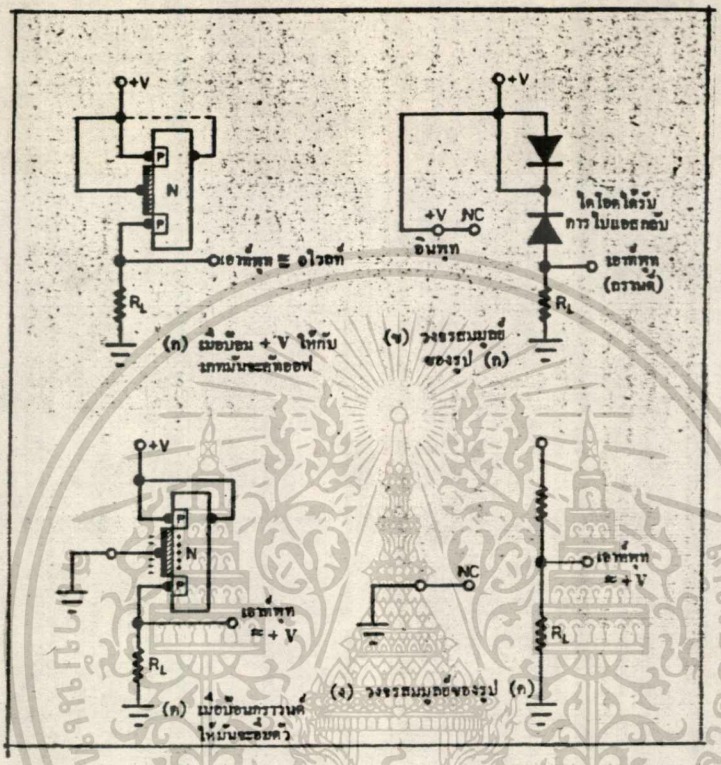
ในทำนองเดียวกันเราสามารถสร้างมอสเฟทชนิด พีแชนแนล (p channel MOS FET) ได้ด้วยหลักการเดียวกัน ลักษณะของเนื้อสารกึ่งตัวนำที่ใช้จะมีลักษณะตรงกันข้ามกัน คือ ซับสเตรทจะใช้สารกึ่งตัวนำชนิด n แล้วทำการแพร่ให้เกิดแอ่ง(บ่อ) ของสารกึ่งตัวนำชนิด p เพื่อเป็นขั้วเดรนและซอร์ส จากนั้นก็ต่อเกทโดยมีฉนวนชั้นเกท ลักษณะโครงสร้างทั่วไป ดังแสดงในรูป



รูปแสดง โครงสร้างของมอสเฟทชนิด p แชนเนล

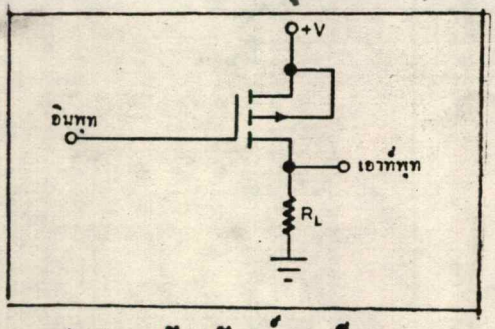
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มอสเฟทชนิด p แชนเนลนี้เราเรียกสั้น ๆ ว่า พีมอส พีมอสมีลักษณะการทำงานเหมือนกับเอ็นมอส แต่มีทิศทางของกระแสและแรงดันตรงข้ามกัน ขอให้พิจารณาการทำงานของพีมอสพร้อมวงจรสมมูลย์ดังรูปที่แสดง



เมื่อป้อนแรงดัน +V ให้กับเกทของพีมอส มันจะอยู่ในภาวะคัทออฟทำให้เอาท์พุทที่ได้มีค่าเท่ากับกราวนด์ และถ้าป้อนแรงดันกราวนด์ให้กับเกทพีมอสที่อิ่มตัว เป็นผลทำให้เอาท์พุทมีค่าประมาณเท่ากับ +V

สัญญลักษณ์ของพีมอส ดังแสดงในรูป สังเกตให้ดีจำเราจะต่อซาซอร์สเข้ากับ V+ และเดรนเข้ากับ RL เอาท์พุทเอาออกที่ขาเดรน

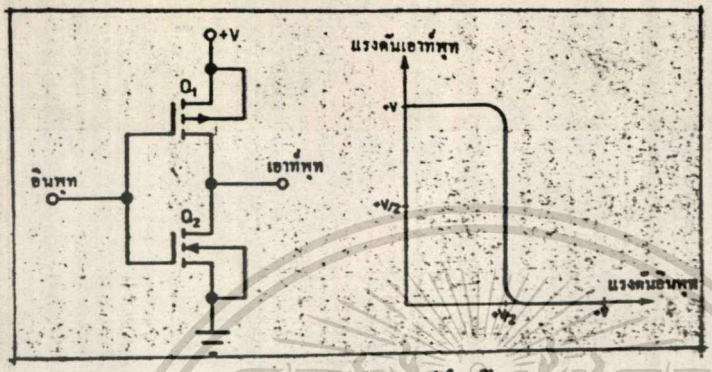


รูปแสดง สัญญลักษณ์ของพีมอส

โดยปกติในการสร้างไอซีนั้นเราพยายามหลีกเลี่ยงการสร้างตัวต้านทานในวงจร ทั้งนี้เพราะการสร้างตัวต้านทานเราไม่สามารถสร้างตัวต้านทานที่มีค่าสูง ๆ ได้ และการสร้างตัวต้านทานต้องใช้พื้นที่เนื้อสารค่อนตัวนำกว้างกว่าปกติทำให้กินเนื้อที่ ดังนั้นเราจึงหาทางที่จะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แทนตัวต้านทานเหล่านี้ด้วยมอสเฟทโดยใช้พีมอสเป็นโหลดของเอ็นมอส และตัวเอ็นมอสเป็นโหลดของพีมอส ดังนั้นจึงนำเอาเอ็นมอสและพีมอสมาประกอบรวมกันและเรียกว่า ซีมอส เป็นคำย่อมาจาก (complementary metal oxide semiconductor) ลักษณะของวงจรอินเวอร์เตอร์ซีมอส ดังแสดงในรูป

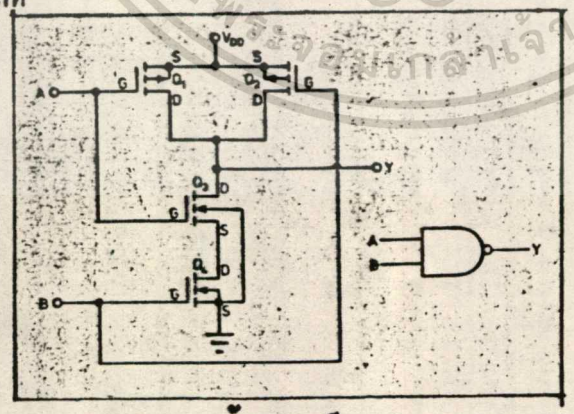


รูปแสดง วงจรอินเวอร์เตอร์ที่ใช้ซีมอส

จากรูปที่แสดงข้างบน เป็นวงจรอินเวอร์เตอร์ สังเกตว่าเมื่อป้อนแรงดัน +V Q_2 จะอิ่มตัวแต่ Q_1 จะคัทออฟเป็นโหลดให้กับ Q_2 โหลด Q_1 จะมีค่าความต้านทานสูงมาก ในทำนองเดียวกันเมื่อป้อนแรงดันกราวนด์ Q_1 จะอิ่มตัวโดยมี Q_2 คัทออฟเป็นโหลดให้กับ Q_1

2.4.2 <u>ลักษณะของวงจรซีมอสที่ใช้เป็นเกท</u>

วงจรซีมอส NAND เกท เนื่องจากมอสเฟทแต่ละตัวทำหน้าที่เหมือนเป็นสวิทช์ ดังนั้นเราจึงสามารถที่จะจัดวงจรซีมอสให้เป็นวงจรเกทต่าง ๆ ได้ ดังรูปที่แสดง เป็นวงจร NAND เกท



รูปแสดง โครงสร้างของซีมอส NAND เกท

พิจารณาการทำงานของวงจร CMOS NAND เกท ตามแสดงในรูปได้ดังนี้

ก. เมื่ออินพุท A และ B มีค่า $+V_{DD}$ ทั้งคู่จะเป็นผลทำให้ Q_3 และ Q_4 อิ่มตัว ส่วน Q_1 และ Q_2 จะอยู่ในภาวะคัทออฟ ดังนั้นเอาท์พุท Y ที่ได้จึงเสมือนต่อกับกราวนด์

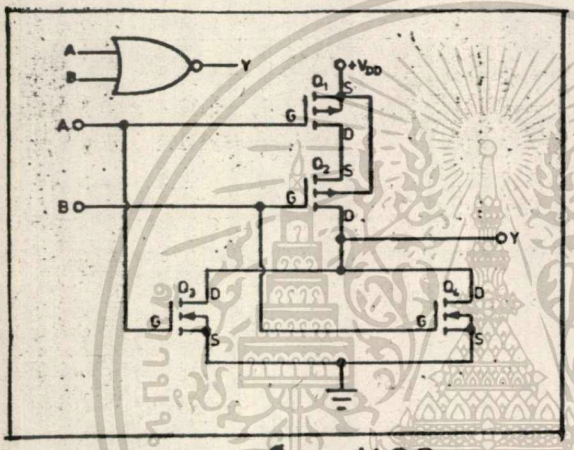
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข. เมื่ออินพุท A ต่อกับ $+V_{DD}$ ส่วน B จะต่อกับ $+V_{DD}$ หรือกราวนด์ก็ตาม จะทำให้ Q_1 อิ่มตัว Q_3 คัทออฟ ดังนั้นเอาท์พุท Y จึงเสมือนต่อกับ V_{DD} ให้เอาท์พุทลอจิก "1" หรือ V_{DD}

ค. เมื่ออินพุท B ต่อกับ $+V_{DD}$ ส่วน A จะต่อกับ $+V_{DD}$ หรือกราวนด์ก็ตาม จะทำให้ Q_2 อิ่มตัว Q_4 คัทออฟ ทำให้เอาท์พุทได้เท่ากับ $+V_{DD}$

ง. เมื่อทั้งสองอินพุทต่อกับกราวนด์ทั้งคู่ Q_1 และ Q_2 จะอิ่มตัวส่วน Q_3 และ Q_4 จะคัทออฟ ทำให้เอาท์พุทที่ได้มีค่าเท่ากับ $+V_{DD}$

วงจร CMOS NOR เกท วงจร CMOS NOR เกทดังแสดงในรูป



รูปแสดง วงจรซีมอส NOR เกท

ในทำนองเดียวกันเราสามารถอธิบายการทำงานของ CMOS NOR เกท ได้ดังนี้

ก. เมื่ออินพุท A และ B ต่อเข้ากับ $+V_{DD}$ ทั้งคู่จะเป็นผลทำให้ Q_1 และ Q_2 คัทออฟ ส่วน Q_3 และ Q_4 จะอิ่มตัวให้ผลลัพธ์ Y เท่ากับกราวนด์

ข. เมื่อทั้งอินพุท A และ B มีต่อกับกราวนด์ทั้งคู่จะเป็นผลทำให้ Q_1 และ Q_2 อิ่มตัวทั้งคู่ ส่วน Q_3 และ Q_4 จะอยู่ในสภาวะคัทออฟ เอาท์พุท Y ที่ได้จึงมีค่าเท่ากับ $+V_{DD}$

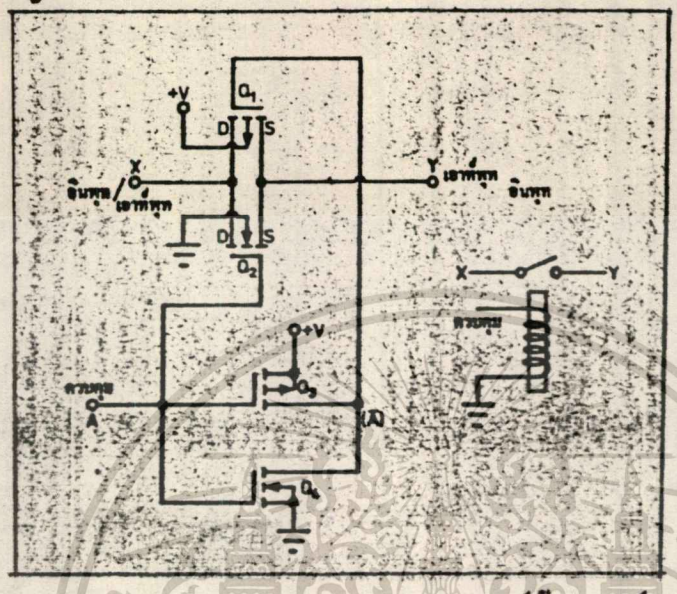
ค. เมื่ออินพุท A ต่อกับ $+V_{DD}$ ส่วน B จะต่อกับอะไรก็ได้ Q_3 จะอิ่มตัวทำให้เอาท์พุท Y มีค่าเป็นลอจิก "0" คือกราวนด์

ง. ในทำนองเดียวกันถ้าอินพุท B ต่อกับ $+V_{DD}$ เป็นลอจิก "1" จะทำให้ Q_4 อิ่มตัวเป็นผลทำให้เอาท์พุทอยู่ในลอจิก "0" หรือกราวนด์

เกทถ่ายทอดหรือ transmitsion gate เกทชนิดนี้เป็นเกทชนิดพิเศษที่มีอยู่ในไอซีพวก CMOS เท่านั้น ตัวเกทถ่ายทอดแตกต่างกับเกทลักษณะเกทอื่น ๆ คือตัววงจรทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

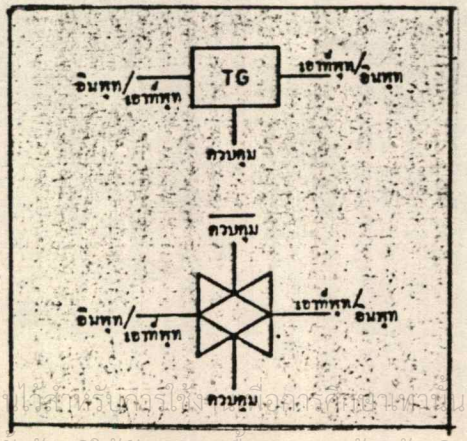
หน้าที่เหมือนสวิทช์ SPST หรือ single pole single throw ที่สามารถควบคุมการทำงานของสวิทช์ให้ ON และ OFF ได้ ข้อดีของเกทแบบนี้คือสัญญาณสามารถผ่านเกทได้ทั้งสองทิศทาง โดยปกติเราใช้ตัวย่อ TG แทนเกทชนิดนี้ ลักษณะของวงจรเกทถ่ายทอดดังแสดงในรูป



รูปแสดง วงจร TG และลักษณะสมมูลย์ด้วยรีเลย์

การทำงานของวงจรพอสรุปได้ดังนี้คือ เมื่อสายควบคุมอยู่ในภาวะลอจิก "1" $\overline{A}$ จะเป็น "0" Q_2 และ Q_4 ซึ่งเป็นชนิด n แชนเนลจะนำกระแส ส่วน Q_1 จะคัทออฟ ส่วนของ X จะต่อกับ Y ในลักษณะจากเดรนมายังซอร์สผ่านทาง Q_2 นั่นคือ สัญญาณผ่านจาก X มา Y ได้

ในทำนองเดียวกันถ้าให้ A เป็นลอจิก "0" จะได้ $\overline{A}$ เป็นลอจิก "1" Q_1 และ Q_3 จะอิ่มตัว ส่วน Q_2 และ Q_4 จะคัทออฟ ทำให้ X ต่อกับ Y ในลักษณะการไหลของกระแสจากซอร์สมายังเดรนผ่าน Q_1 หรือทิศทางจาก Y มายัง X คือ Y เป็นอินพุท X เป็นเอาท์พุท เพื่อให้การเขียนสัญญลักษณ์ของวงจรได้ง่ายขึ้น เราจะใช้สัญญลักษณ์ ดังแสดงในรูป



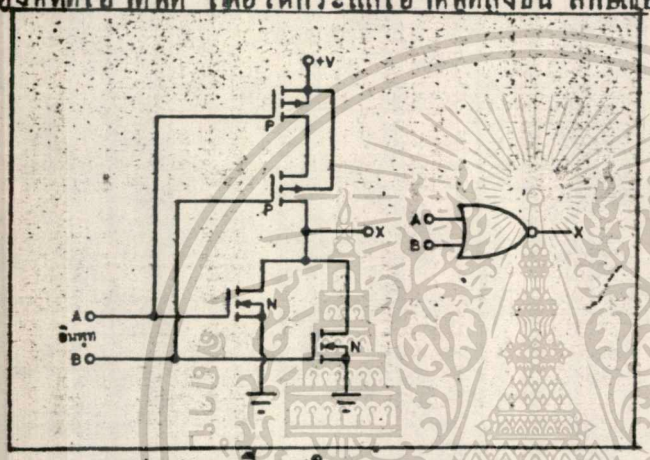
รูปแสดง สัญญลักษณ์ของ transmission gate

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

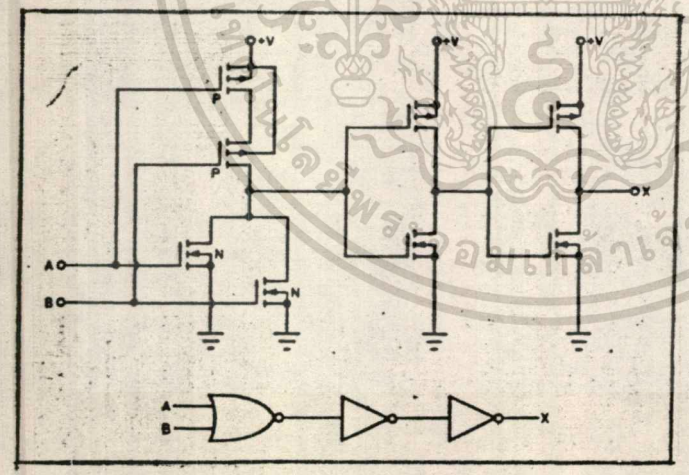
2.4.3 ชนิดของไอซีซีมอส

ซีมอสแบ่งออกเป็นชนิดใหญ่ ๆ ได้สามกลุ่มด้วยกันคือ ซีมอสกลุ่มเอ (A-series) ซีมอสที่ใช้เป็นบัฟเฟอร์ หรือกลุ่มบี (B-series) และกลุ่มซิลิกอนออน เซปฟายร์ (silicon-on sapphire) และเราใช้ตัวย่อว่า SOS

ซีมอสในกลุ่มเอเป็นซีมอสรุ่นแรกสุด มีราคาถูกและมีชนิดของฟังก์ชั่นการทำงานต่าง ๆ มากมาย ส่วนในกลุ่มบีเป็นกลุ่มที่ปรับปรุงมาจากกลุ่มแรกที่เพิ่มเติม วงจรอินเวอร์เตอร์สองชุดที่เอาท์พุท เพื่อให้กระแสเอาท์พุทสูงขึ้น ลักษณะซีมอสทั้งสองกลุ่มนี้ ดังแสดงในรูป



รูปแสดง ซีมอสในกลุ่มเอ



รูปแสดง ซีมอสในกลุ่มบี

การเพิ่มอินเวอร์เตอร์เข้าไปอีก จะทำให้การสวิทช์ระหว่าง ลอจิก "0" และ "1" มีช่วงเวลาขาขึ้นและลงสั้นลงกว่าเดิม ซึ่งเป็นผลให้ความเร็วในการสวิทช์ดีขึ้นกว่ากลุ่มเอ

ส่วน SOS นั้นแตกต่างกับซีมอสทั้งสองกลุ่ม ในส่วนของฉนวนที่กั้นเกทจะใช้สาร Silicon on Sapphire ซึ่งทำให้ค่าตัวเก็บประจุที่เกทมีค่าลดลง เป็นผลทำให้ความถี่ของการใช้งานสูงขึ้นอีกมาก ซึ่งสามารถทำการสวิทช์ที่ความถี่ได้สูงถึง 50 MHz

2.4.4 การใช้งานไอซีซีมอส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เนื่องจากซีมอสเป็นอุปกรณ์ที่มีอิมพีแดนซ์สูงมาก ดังนั้นประจุไฟฟ้าสถิตย์ที่มีค่ามาก ๆ อาจทำให้ส่วนของฉนวนที่เกทเกิดการเสียหายได้ ในการใช้งานซีมอสจึงต้องระวังในการจับต้อง ปกติเราต้องเก็บไอซีพวกนี้ไว้บนแผ่นสารตัวนำเพื่อกันไม่ให้มีการสะสมประจุที่ขาของไอซี และในการใช้หัวแร้งบัดกรีต้องระวังในเรื่องไฟฟ้าสถิตย์เช่นกัน ทางที่ดีแล้วเราควรจะกราวนด์ตัวหัวแร้ง และต้องบัดกรีขา V_{DD} ก่อน แล้วจึงบัดกรีขา V_{SS}

นอกจากนี้เรายังต้องคำนึงถึงหลักการสำคัญในการใช้ซีมอสคือ

1. อินพุทที่ไม่ได้ใช้ควรต่อกับ V_{SS} หรือ V_{DD} โดยมีตัวต้านทานขนาด 200K ถึง 1 M Ω ต่อคั่น

2. ขนาดของแรงดันอินพุทจะต้องไม่เกินช่วงแรงดันที่ใช้งาน V_{DD} และ V_{SS} เพราะจะทำให้ไดโอดที่มีไว้ป้องกันซีมอส ในไอซีเกิดการเสียหายได้

3. ไม่ควรให้สัญญาณอินพุทกับไอซีก่อนป้อนไฟเลี้ยง

4. ในการโหลดเอาท์พุทของซีมอสควรจะต้องพิจารณาขนาดของกระแสซิงค์ และกระแสซอร์ส

5. ไม่ควรต่อ C ครอมเอาท์พุทของซีมอส และเอาท์พุทก็ไม่ควรต่อกับ V_{DD} หรือ V_{SS} โดยตรง

6. ในการใช้ซีมอสถ้าบังคับให้แรงดันอยู่ในช่วงกำลังเปลี่ยนสถานะจะทำให้ซีมอสร้อนมาก และอาจเกิดการเสียหายได้

7. สัญญาณนาฬิกาใช้กับซีมอสควรมีช่วงเวลาขาขึ้น และขาลงไม่น้อยกว่า 5 nS และ 15nS ตามลำดับ เพราะถ้าช้าและความถี่สูงจะทำให้ซีมอสร้อน

8. แหล่งจ่ายไฟที่ใช้ควรอยู่ในช่วง 3 – 15 โวลท์

9. ไม่ควรมีตัวต้านทานต่ออนุกรมกับแหล่งจ่ายไฟก่อนป้อนเข้า V_{DD}

10. การสับข้วไฟเลี้ยงโดยบังเอิญจะทำให้ซีมอสเกิดการเสียหายได้โดยทันทีทันใด

2.4.5 <u>บทสรุปไอซีซีมอส</u>

ไอซีซีมอสเป็นไอซีที่ได้สร้างความสำเร็จให้กับวงการอิเล็คทรอนิกส์เป็นอย่างมาก ซีมอสทำให้เกิดไอซีประเภท LSI ซึ่งทำงานได้อย่างมหัศจรรย์ เช่น เครื่องคิดเลข นาฬิกาดิจิตอล คอมพิวเตอร์ ฯลฯ ในอนาคตไอซีประเภทนี้ จะได้รับการพัฒนาให้การทำงานได้ดียิ่งขึ้น และสามารถสร้างวงจรที่ยุ่งยากซับซ้อนลงบนไอซี เพื่อให้ทำงานตามจุดมุ่งหมายดียิ่งขึ้น

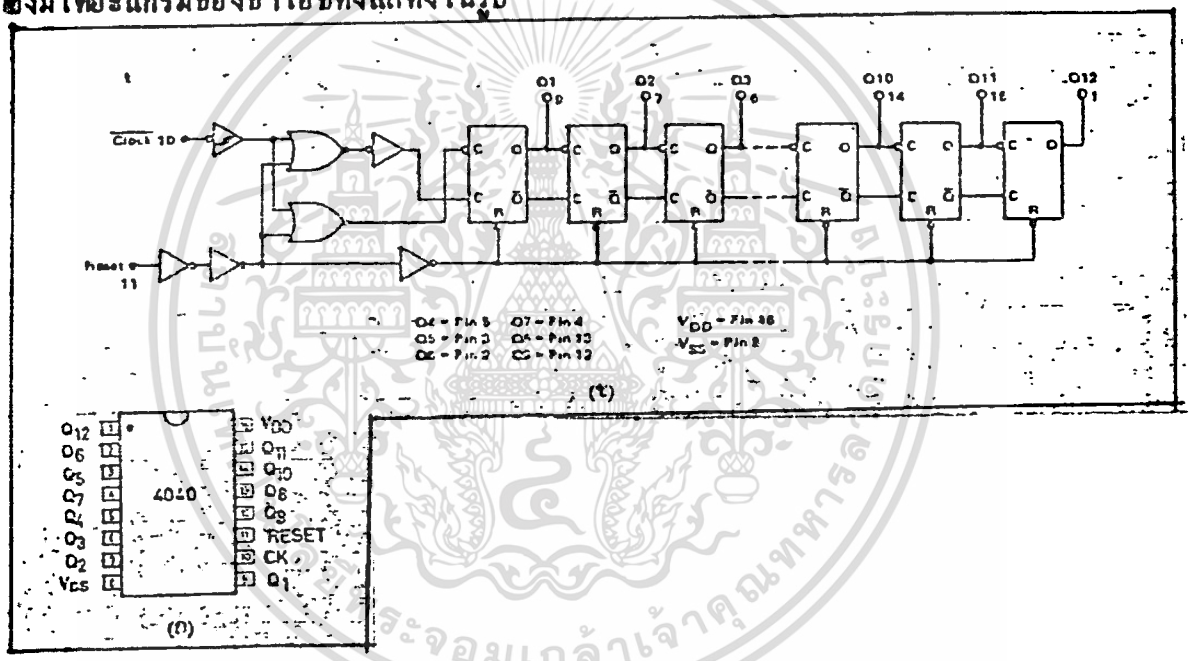
ข้อดีของไอซีซีมอสอีกประการหนึ่งได้แก่ การใช้งานกำลังงานไฟฟ้าที่ต่ำมาก เมื่อเทียบกับไอซีชนิดอื่น และยังสามารถทนแรงดันได้สูงกว่าอีกด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5 โปรแกรมเคาน์เตอร์

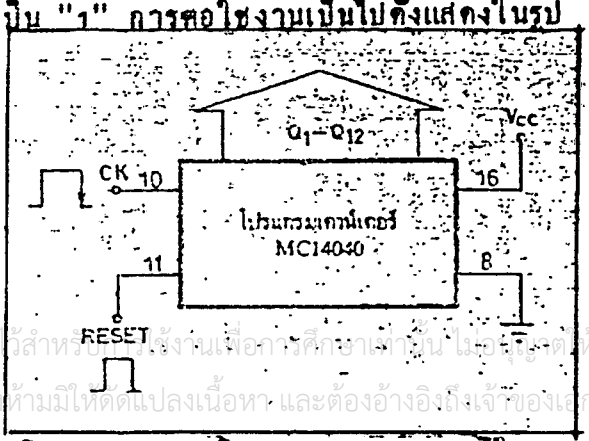
เมื่องจากตัว ICU เบอร์ MC14500 นี้ มีสัญญาณนาฬิกาส่งออกมา เราจะนำสัญญาณนาฬิกานี้ส่งต่อไปกับวงจรนับ เพื่อนับให้ค่าเปลี่ยนแปลงไปทีละ 1 แล้วส่งต่อให้กับหน่วยความจำ เพื่อเฟทช์ข้อมูลคำสั่งไปกระทำตาม ปกติเมื่อขา RST ของ MC14500 เป็น "1" มันจะรีเซทตัวมันเอง และสัญญาณนาฬิกาใน MC14500 ก็จะหยุดด้วย เราจึงใช้สัญญาณ RST สั่งรีเซท MC14500 มาทำการรีเซทตัวโปรแกรมเคาน์เตอร์ด้วย

โปรแกรมเคาน์เตอร์ตัวที่ราคาถูกที่สุด เราจะเลือกจากวงจรนับที่มีขายอยู่ทั่วไปแล้ว ในที่นี้จะใช้ไอซี CMOS เบอร์ 4040 ซึ่งเป็นวงจรนับ 12 สถานะ มีเอาท์พุท 12 เส้น ซึ่งมีไดอะแกรมของขาไอซีดังแสดงในรูป



รูปแสดง วงจรนับ 4040 ที่จะใช้เป็นโปรแกรมเคาน์เตอร์

วงจรนับ 4040 นี้ จะนับเมื่อสัญญาณนาฬิกาที่เข้ามาเปลี่ยนจากลอจิก "1" ไปเป็นลอจิก "0" สำหรับการรีเซทจะทำให้ Q_1-Q_{12} มีค่าเป็น "0" หมด โดยการรีเซทจะต้องให้ขา Reset นี้เป็น "1" การต่อใช้งานเป็นไปดังแสดงในรูป



รูปแสดง โครงสร้างของโปรแกรมเคาน์เตอร์ที่ใช้เอาท์พุทถึง 12 บิท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

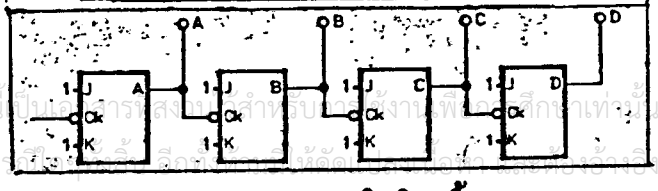
เราสามารถเปลี่ยนแปลงวงจรโปรแกรมเคาน์เตอร์ โดยใช้ไอซีเบอร์อื่นอีกก็ได้ เช่นเบอร์ 4516 ซึ่งเป็น binary up/down counter ที่สามารถพรีเซทค่าได้และหลักการของการพรีเซ็ทค่านี้ จะทำให้เกิดการกำหนดค่าให้กับโปรแกรมเคาน์เตอร์ได้ ทำให้สามารถทำคำสั่ง Jump ได้ แต่ในที่นี้เราจะลองใช้แบบง่าย ๆ โดยยังไม่มีการ Jump หรือ RTS (restart) แต่ในตอนต่อไปเราจะเปลี่ยนแปลงโปรแกรมเคาน์เตอร์ให้มีความคล่องตัวในการทำงานยิ่ง ๆ ขึ้นไป

2.5.1 วงจรนับชนิดริปเปิ้ล

วงจรนับหรือเคาว์นเตอร์ (counter) หมายถึง วงจรที่ใช้นับจำนวนพัลซ์ของสัญญาณดิจิตอล วงจรนับมีประโยชน์มากเพราะเป็นส่วนของวงจรที่ใช้ในเครื่องมือต่าง ๆ เช่น เครื่องนับความถี่ เครื่องวัดแรงดัน กระแสแสดงผลด้วยตัวเลข

วงจรนับชนิดริปเปิ้ล (ripple counter) เป็นวงจรนับพัลส์ชนิดที่มีการทำงานโดยไม่ต้องใช้สัญญาณนาฬิกามาบังคับจังหวะการเปลี่ยนสถานะของฟลิบฟลอบ ดังแสดงในรูป เป็นวงจรนับชนิดริปเปิ้ลชนิดขนาดสี่บิท

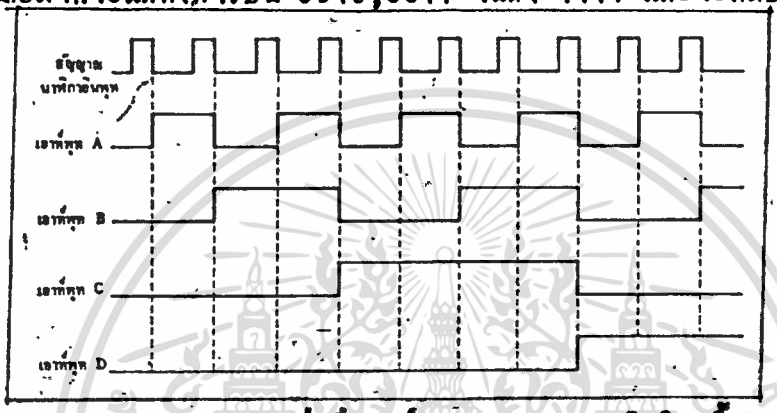
จำนวนพัลซ์ที่เข้าที่อินพุท	D	C	B	A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1



รูปแสดง วงจรนับชนิดริปเปิ้ล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การป้อนอินพุทเราจะป้อนเข้าทางขา CK และทุกครั้งที่สัญญาณนาฬิกาเปลี่ยนจากลอจิก "1" ไปเป็น "0" จะทำให้ฟลิบฟลอบเปลี่ยนสถานะ (ฟลิบฟลอบที่ใช้ให้ขา J และ K เป็นลอจิก "1") ลักษณะของลูกคลื่นและการเปลี่ยนสถานะของฟลิบฟลอบ ดังแสดงในรูป ถ้าพิจารณาจากลูกคลื่นเอาท์พุทของฟลิบฟลอบทั้งสี่ตัวคือ A,B,C และ D จะมีลักษณะของการนับค่าจำนวนพัลซ์ตามเลขไบนารี่ นั่นคือ เมื่อพัลซ์แรกผ่านไปเอาท์พุทก็แสดงค่า 0001 และพัลซ์ต่อมาก็จะแสดงค่าเป็น 0010,0011 จนถึง 1111 และจะกลับมา 0000 อีกครั้ง

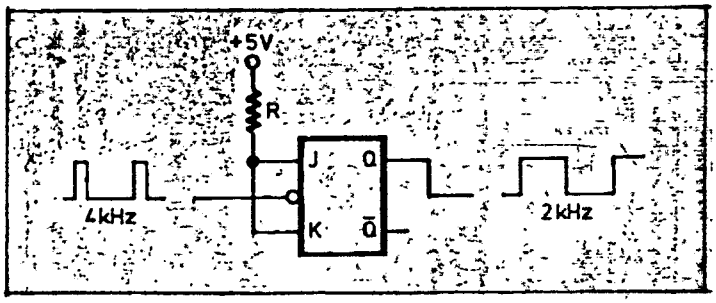


รูปแสดง รูปสัญญาณคลื่นที่เอาท์พุทของวงจรนับชนิดริปเปิ้ล

วงจรนับริปเปิ้ลที่มีฟลิบฟลอบ 4 ตัว เราอาจเรียกได้ว่าเป็นวงจรมอด-4 (MOD-4) และถ้าใช้ฟลิบฟลอบ N ตัว เราก็เรียกว่าวงจร MOD-N ซึ่งคำว่า MOD เป็นคำย่อของคำว่า mudulu N นอกจากนี้ เรายังเรียกลักษณะของวงจร MOD-N ว่า วงจรหารด้วย N ซึ่งกล่าวถึงในหัวข้อถัดไป

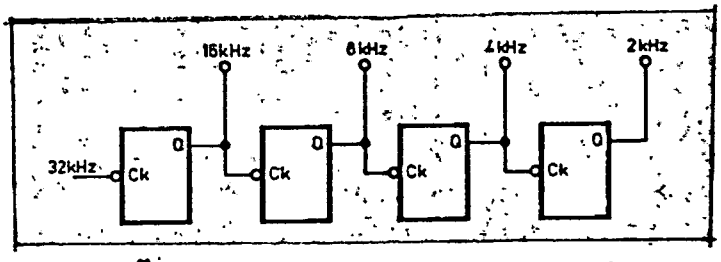
2.5.2 <u>วงจรหาร</u>

ฟลิบฟลอบมีคุณสมบัติพิเศษคือตัวมันสามารถหารความถี่ของสัญญาณนาฬิกาที่อินพุทได้ แต่อย่างไรก็ตามการหารความถี่ของฟลิบฟลอบจะเกิดได้ก็ต่อเมื่อต่อขา J และ K ของฟลิบฟลอบมาอยู่ที่ลอจิก "1"



รูปแสดง การใช้ฟลิบฟลอบหารความถี่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ใช้ฟลิบฟลอบ 4 ตัว สามารถหารความถี่ได้ด้วย 16

2.5.3 วงจรนับ-หารด้วย N

ในการต่อฟลิบฟลอบ N ตัว เป็นวงจรนับชนิดริบเปิ้ล จะทำให้การนับเลขไบนารี่เป็นไปได้ถึง 2^N สถานะ เช่นถ้าใช้ฟลิบฟลอบ 2 ตัวก็นับได้ 4 สถานะ ถ้าใช้ 3 ตัวก็นับได้ 2^3 เท่ากับ 8 สถานะ ปัญหาก็คือทำอย่างไร เราจึงทำให้วงจรนับหารด้วยค่าตัวเลขใด ๆ ก็ได้เช่น 3,5,6

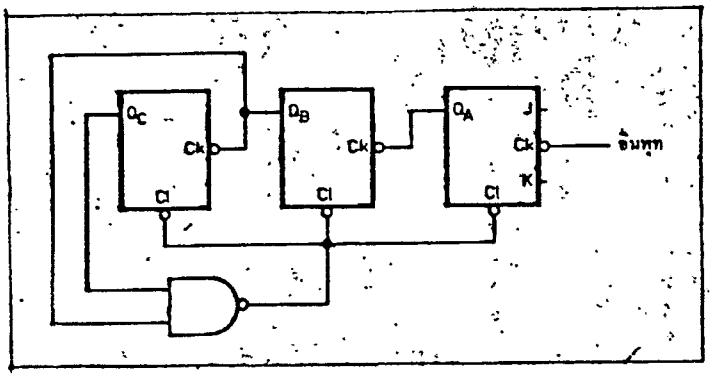
วิธีที่จะทำให้วงจรนับ นับได้ถึงค่าที่ต้องการแล้วเริ่มต้นกลับมานับใหม่ เช่น นับจาก 0 ถึง 9 แล้วกลับมาเป็น 0 ใหม่นั้นทำได้หลายวิธีด้วยกัน วิธีแรกทำให้วงจรนับหยุดนับ เมื่อนับถึงค่าที่เราต้องการ เช่น เมื่อมีฟลิบฟลอบสี่ตัววงจรจะนับจาก 0000 ถึง 1111 แล้วกลับมาเริ่ม 0000 ใหม่ แต่ถ้าเราต้องให้นับจาก 0000 ถึง 1001 แล้วกลับมาเป็น 0 ใหม่

หลักการเบื้องต้นในการสร้างวงจรนับ MOD-N คือใช้ขาเคลียร์ของฟลิบฟลอบให้เป็นประโยชน์ โดยทำให้ฟลิบฟลอบถึงสถานะที่เราต้องการ แล้วกระโดดข้ามสถานะที่เหลือไปอยู่ที่สถานะ 0 ใหม่ นั่นคือเราต้องมีการดีเทคค่าวงจรนับ ว่าถึงค่าที่เราต้องการแล้วหรือยัง เมื่อถึงแล้วจึงไปทำการเคลียร์ฟลิบฟลอบ

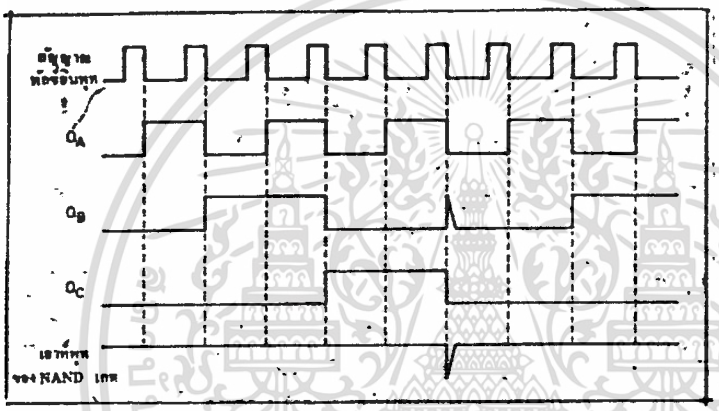
วิธีการที่จะออกแบบวงจรนับ MOD-N ทำได้ไม่ยากนัก เราเริ่มด้วยการหาจำนวนฟลิบฟลอบน้อยที่สุดที่ใช้ ซึ่งโดยปกติถ้าเป็นจำนวนฟลิบฟลอบเราจะได้ $2^n \gg N$ เช่น ต้องการ MOD-10 เราก็ต้องใช้ฟลิบฟลอบ 4 ตัว เพราะ $2^4 \gg 10$ เมื่อได้ฟลิบฟลอบแล้วเราก็จะนำฟลิบฟลอบเหล่านั้นมาต่อเป็นวงจรริบเปิ้ลเคาน์เตอร์ใช้ NAND เกทตัวหนึ่ง โดยให้เอาท์พุทของ NAND เกทต่อไปยังขาเคลียร์ของฟลิบฟลอบทุกตัว ส่วนอินพุทของ NAND เกทนำมาจากเอาท์พุทที่แสดงลอจิก " 1" เมื่อนับถึงสถานะ N หรือแสดงค่าไบนารี่เป็นเลข N-1

ดังตัวอย่างดังรูปที่แสดง เป็นวงจรนับ-หารด้วย 6 หรือ MOD-6 การนับจะนับจาก 000-001-010-011-100-101 และแทนที่จะนับต่อเป็น 110 วงจรจะได้รับการเคลียร์เป็น 000 แล้วเริ่มนับใหม่ ลักษณะของลูกคลื่น ดังแสดงในรูปถัดไป

เอกสารนี้ใช้สำหรับการศึกษาเพื่อการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ในเชิงพาณิชย์ ตลอดจนการนำไปดัดแปลงเนื้อหาและอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



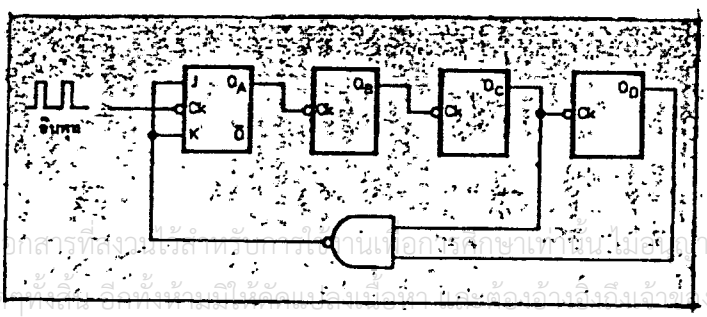
รูปแสดง วงจรนับ-หาร MOD - 6



รูปแสดง สัญญาณการทำงานของวงจร MOD - 6

การเคลียร์ฟลิบฟลอบจะเกิดขึ้นเมื่อขาเคลียร์ของฟลิบฟลอบแสดงสภาวะลอจิก"0" ข้อเสียของวงจรนับเช่นนี้คือมันจะให้เอาท์พุทบางส่วนเป็นพัลซ์แหลม ดังตัวอย่างเช่นพัลซ์จากเอาท์พุท Q_B ในขณะขาเคลียร์จะมีลักษณะเป็นพัลซ์แหลม ๆ เกิดขึ้น

นอกจากนี้เรายังมีวิธีการที่จะทำให้วงจรนับนับถึงค่าที่เราต้องการแล้ววงจรจะหยุดนับคือไม่มีผลต่อสัญญาณพัลซ์ที่เข้ามาที่อินพุทเลย จนกว่าเราจะทำการเคลียร์บางสิ่งบางอย่างเสร็จก่อนจึงค่อยให้เริ่มต้นนับใหม่ หลักการนี้ก็คล้ายคลึงกับหลักการที่กล่าวมาแล้วนั้นเอง เพียงแต่เราเอาเอ้าท์พุทที่ออกจาก NAND เกทต่อเข้ามายังขา J และ K ของฟลิบฟลอบเป็น 0 หมด เอาท์พุทของฟลิบฟลอบก็จะไม่เปลี่ยนแปลงสถานะคือจะคงค่าเช่นนั้นเรื่อยไป



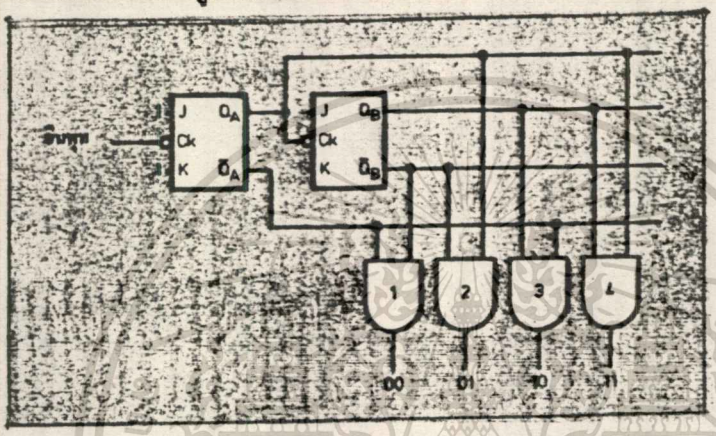
รูปแสดง วงจรนับเมื่อนับถึง 1100 แล้วมันจะหยุดนับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5.4 การถอดรหัสวงจรนับ

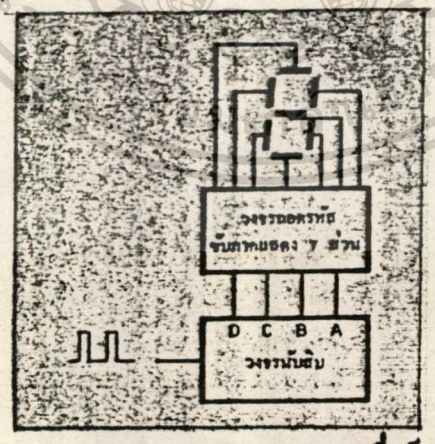
ในบางครั้งการที่เอาท์พุทของฟลิบฟลอบทุกตัวแสดงสถานะทางไบนารี่เป็นสิ่งที่เราไม่สามารถแสดงความหมายได้โดยตรง เราจึงต้องมีวงจรถอดรหัสวงจรนับ การถอดรหัสวงจรนับก็คือวงจรเกทที่ต่อเป็นวงจรถอดรหัสนั่นเอง

เอาท์พุทของการนับจะแสดงลอจิก "1" ที่เอาท์พุทของวงจร AND เกท ตามตัวเลขไบนารี่ เช่น เมื่อแสดง 00 เอาท์พุทของเกท "1" และ 10 ที่เอาท์พุทของเกท 2 ตัวนั้นถ้าให้พัลซ์เข้าที่อินพุทแบบต่อเนื่อง



รูปแสดง วงจรถอดรหัสวงจรนับไบนารี่ 2 บิท

ในวงจรนับที่ใช้งานเช่นวงจรนับสิบ เรามักจะให้เอาท์พุทส่งต่อไปภาคแสดงเอาท์พุทเป็นตัวเลขที่เราเข้าใจได้ทันที ดังแสดงในรูป และวงจรที่ใช้มักจะประกอบอยู่ในไอซีที่เราต่อเพื่อใช้งานได้ทันที

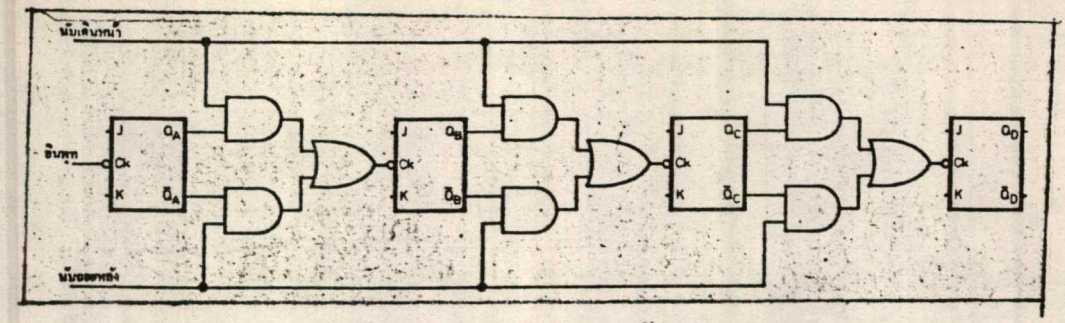


รูปแสดง วงจรถอดรหัสจากตัวเลขไบนารี่เป็นเอาท์พุทภาคแสดง 7 ส่วน

2.5.5 วงจรนับเดินหน้า-ถอยหลัง

วงจรนับที่กล่าวถึงแล้วนั้นมักจะเป็นวงจรนับชนิดเดินหน้า หรือกล่าวทางเลขฐานสิบก็คือนับจาก 0,1,2,.....แต่บางครั้งเราอาจจะต้องการใช้งานในการนับในลักษณะถอยหลังเช่น 9,8,7.....0 เป็นต้น วิธีการที่จะนับเดินหน้าและถอยหลัง เราก็ต่อเกทเพิ่มเข้าไปในวงจร ขอให้พิจารณาวงจรนับเดินหน้าและถอยหลังที่เป็นวงจรของไอซี 74191

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

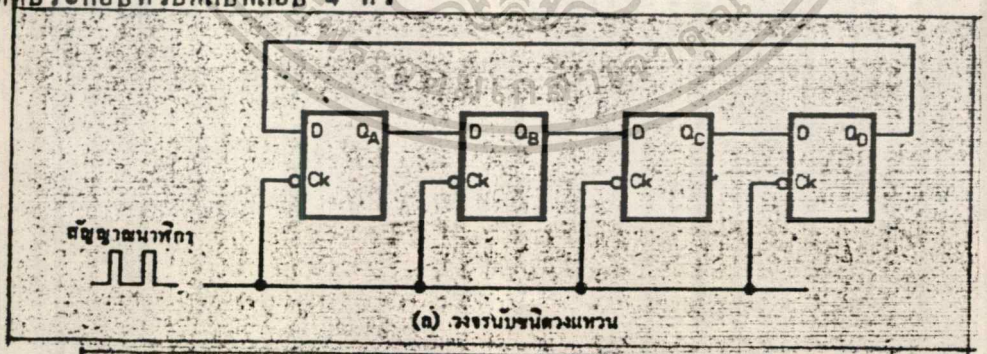


รูปแสดง โครงสร้างของวงจรนับเดินหน้า-ถอยหลังชนิด 4 บิท

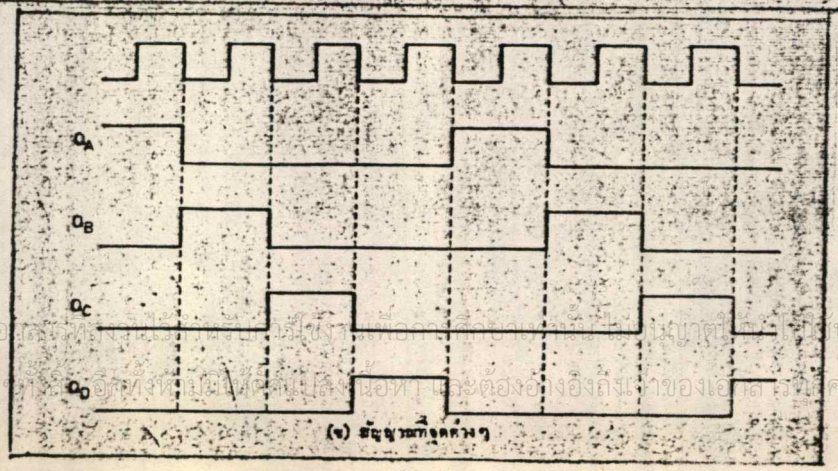
การนับนั้นถ้าเราเอาเอาท์พุทออกจากวงจรริปเปิ้ลเคาวน์เตอร์ทางขา Q ของฟลิบฟลอบหมดทุกตัว วงจรจะเป็นการนับเดินหน้า และถ้าเอาท์พุทออกทางขา $\bar{Q}$ เอาท์พุทก็จะได้เป็นวงจรนับถอยหลัง โครงสร้างของไอซีเบอร์ 74191 เป็นการกำหนดเกทขึ้นมาเพื่อที่จะป้อนเอาท์พุทจาก Q หรือ $\bar{Q}$ ไปยังฟลิบฟลอบอีกตัวหนึ่ง นั่นคือถ้าที่ขานับเดินหน้าเป็นลอจิก "1" และขานับถอยหลังเป็นลอจิก "0" เมื่อผ่านวงจรเกทก็เสมือนกับว่า เป็นการต่อเอาเอาท์พุททางขา Q เข้าหาขา CK และในทำนองเดียวกันถ้าต้องการนับถอยหลังก็เป็นการต่อขา $\bar{Q}$ เข้ากับขา CK นั่นเอง

2.5.6 วงจรนับชนิดวงแหวน

วงจรนับชนิดวงแหวนเป็นแบบหนึ่งของชิฟรีจิสเตอร์ที่มีการป้อนกลับ และเป็นแบบที่มีโครงสร้างง่ายมาก ลักษณะของวงจร ดังแสดงในรูป ซึ่งเป็นวงจรนับชนิดวงแหวนขนาดสี่บิทที่ประกอบด้วยฟลิบฟลอบ 4 ตัว



(ก) วงจรนับชนิดวงแหวน



(ข) สัญญาณที่จุดต่างๆ

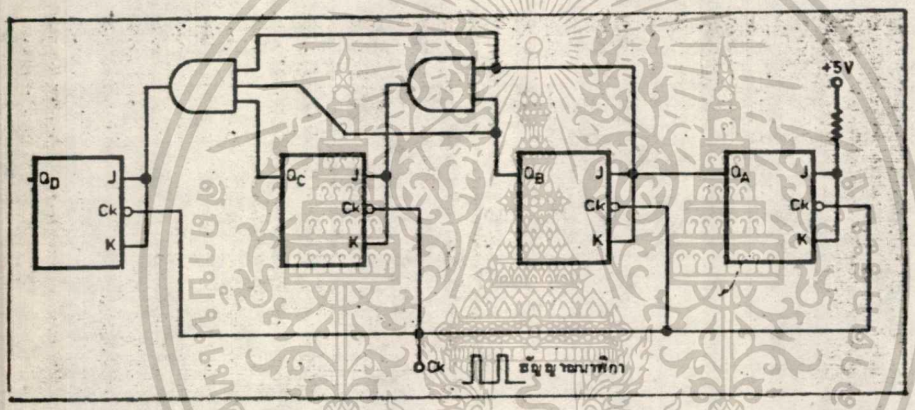
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น พร้อมนี้ขอให้นิสิตนักศึกษาและอาจารย์ทุกท่านที่ใช้เอกสารนี้ โปรดอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การทำงานของวงจรก็เป็นการนำเอาสัญญาณไหลวนเวียนกันไปจากตัวแรกไปจนถึงตัวสุดท้าย แล้วกลับมาเริ่มต้นที่ตัวแรกใหม่

ปัญหาของวงจรมีอยู่ว่า ในขณะเริ่มต้นฟลิบฟลอบตัวแรกควรจะมีสถานะ "1" ส่วนฟลิบฟลอบตัวอื่น ๆ ควรจะมีค่าเป็น "0" ซึ่งโดยปกติเรามีการต่อวงจรเกทเพื่อป้อนให้วงจรได้

2.5.7 <u>วงจรนับชนิดซิงโครนัส</u>

ปัญหาที่เกิดขึ้นกับวงจรนับชนิดริปเปิ้ลคือเมื่อต้องการสร้างเป็นวงจร MOD-N แล้ว จะเกิดพัลซ์เล็ก ๆ ที่ไม่พึงปรารถนาจากการป้อนกลับมาเคลียร์ฟลิบฟลอบ แต่สำหรับวงจรนับชนิดซิงโครนัสสามารถแก้ไขปัญหานี้ได้ เพราะฟลิบฟลอบทุกตัวในวงจรนับจะเปลี่ยนสถานะพร้อมกัน ฟลิบฟลอบที่ใช้นี้ต้องเป็นฟลิบฟลอบชนิดมาสเตอร์เลฟฟลิบฟลอบ



รูปแสดง วงจรนับชนิดซิงโครนัส

ในการต่อเป็นวงจรนับชนิดซิงโครนัสสามารถนับได้ตามปกติอีกทั้งความเร็วในการนับจะนับได้สูงกว่าวงจรนับชนิดริปเปิ้ลมาก เพราะไม่มีการหน่วงเวลาเกิดขึ้นระหว่างฟลิบฟลอบ

ลักษณะการทำงานของวงจรนับชนิดซิงโครนัส ก็คือการควบคุมสภาวะของขา J และ K นั่นเอง ดังจะเห็นได้ว่าในการเริ่มแรก Q_A Q_B Q_C Q_D เป็นลอจิก "0" ดังนั้นพัลซ์แรกที่เข้าทำให้ Q_A เปลี่ยนจาก "0" ไปเป็น "1" ได้เพียงตัวเดียว ครั้นพัลซ์ที่สองต่อมา Q_B จะเปลี่ยนไปเป็น "1" ส่วน Q_A จะเปลี่ยนกลับเป็น "0" เป็นต้น

2.5.8 <u>สรุปวงจรนับ-หาร</u>

วงจรชิฟท์รีจิสเตอร์และวงจรนับ-หารมีประโยชน์มากในวงจรดิจิตอล วงจรการทำงานต่าง ๆ ของเครื่องมือทางดิจิตอล จนถึงเครื่องคอมพิวเตอร์ย่อมต้องมีส่วนของวงจรเหล่านี้ประกอบอยู่เสมอ และการใช้งานนั้นเรายังมีเทคนิคบางประการที่จะต้องนำมาคิดถึงอยู่ด้วย เช่นปัญหาของการเกิดพัลซ์เล็ก ๆ อันไม่พึงปรารถนา ทั้งนี้เพราะพัลซ์นั้นอาจไปหรือการทำงานของฟลิบฟลอบบางตัวให้ทำงานได้

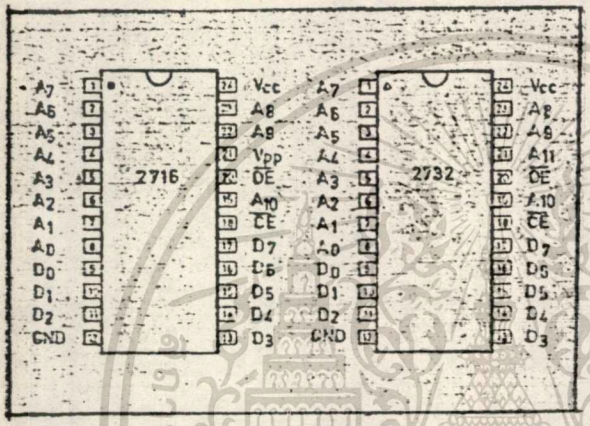
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6 หน่วยความจำ

หน่วยความจำที่จะเก็บโปรแกรมในที่นี้ จะใช้ EPROM ซึ่งเป็นหน่วยความจำ ROM ที่เรารู้จัก และใช้งานกันได้เป็นอย่างดี จึงขอถือโอกาสกล่าวโดยย่อเกี่ยวกับ EPROM ที่ใช้ในงานทดลองนี้ ซึ่งจะขอใช้ ไอซี เบอร์ 2716 หรือ 2732

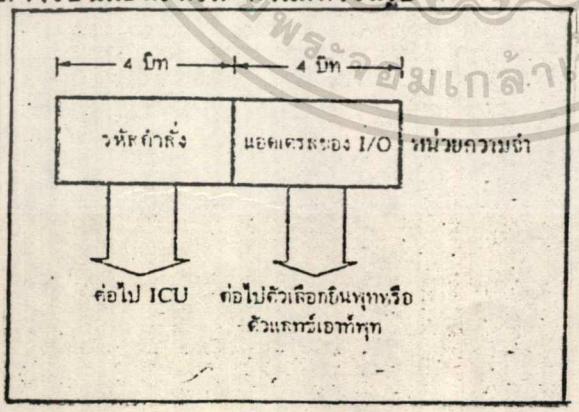
EPROM เบอร์ 2716 และ 2732 นี้เป็นไอซีที่มีขนาดหน่วยความจำ 2K ไบท์ และ 4K ไบท์ ที่หาได้ง่าย และโปรแกรมข้อมูลลงไปได้ง่ายเช่นกัน

การจัดขาของไอซีหน่วยความจำ EPROM เป็นไปดังรูปที่แสดง



รูปแสดง การจัดขาของ EPROM

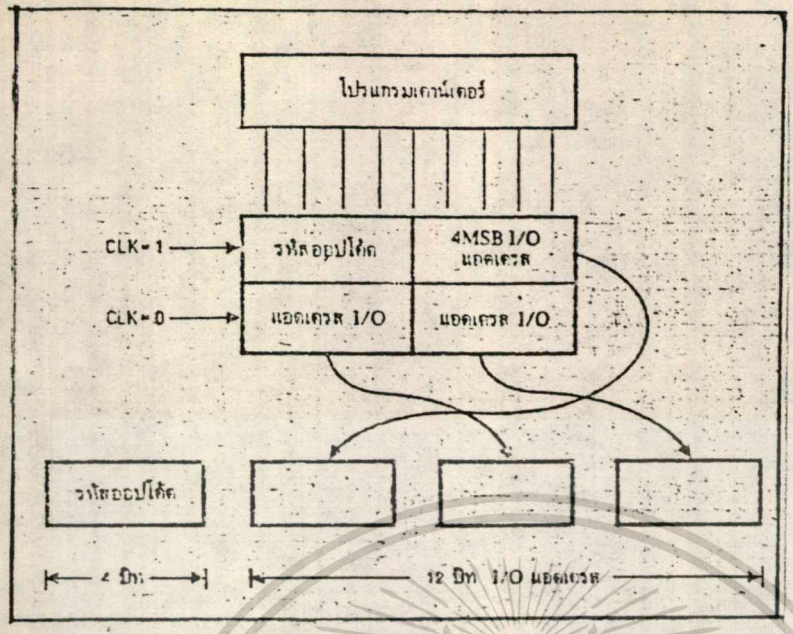
ส่วนของหน่วยความจำนี้ จะเก็บข้อมูลขนาด 8 บิท ซึ่งจะใช้เป็นรหัสคำสั่งเพียงแค่ 4 บิท ในที่นี้จะใช้อีก 4 บิท เป็นแอดเดรสของ I/O โดยใช้ 4 บิทบนเป็นคำสั่ง และ 4 บิทล่างเป็นแอดเดรส ดังแสดงในรูป



รูปแสดง โครงสร้างข้อมูลใน ROM

การใช้หน่วยความจำ EPROM เบอร์ 2716 นี้ จะมีความจุได้ถึง 2048 คำสั่ง ซึ่งมากเกินพอสำหรับการเขียนโปรแกรมลงไป ในการออกแบบถ้าต้องการขยายขนาดของแอดเดรสหน่วยความจำออกไปอีกก็ได้ โดยใช้คำสั่งละ 2 ไบท์ก็ได้ โดยโครงสร้างของคำสั่งอาจจะเป็นแบบการอ่านทีละ 2 ไบท์ ดังแสดงในรูป เพื่อขยายการอ้างแอดเดรส I/O

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้



รูปแสดง การใช้คำสั่งยาว 2 ไบท์

ตามปกติ ไมโครโปรเซสเซอร์เป็นหัวใจของระบบไมโครคอมพิวเตอร์ทั่ว ๆ ไป และหน่วยความจำเปรียบเสมือนสมองของระบบ ซึ่งมีความสำคัญไม่ยิ่งหย่อนไปกว่าตัวไมโครโปรเซสเซอร์เอง ที่จริงแล้วหน่วยความจำยังเป็นส่วนที่มีราคาแพงมากในระบบอีกด้วย ดังนั้นจึงควรได้รับความสนใจอย่างจริงจัง หน้าที่ที่สำคัญของหน่วยความจำในระบบไมโครคอมพิวเตอร์ก็คือ

1. เป็นหน่วยเก็บโปรแกรมคำสั่งที่จะสั่งให้ CPU ทำงานตามความต้องการ

2. เป็นหน่วยเก็บข้อมูลชั่วคราว ซึ่งหมายถึงข้อมูลสามารถเคลื่อนเข้าไปในหน่วยความจำหรือเอาออกจากหน่วยความจำได้ตามความต้องการ ในขณะใช้งานหรือไม่ใช้ก็ได้

จะเห็นจากหน้าที่ทั้งสองของหน่วยความจำว่า ในบางครั้งสิ่งที่ถูกเก็บไว้อาจเป็นในลักษณะถาวร นั่นคือไม่ต้องเปลี่ยนแปลงอีกต่อไป หลังจากเก็บไว้แล้ว เช่นเป็นโปรแกรมสั่งงานของ CPU ในงานประจำ แต่ในบางครั้ง อาจมีการเก็บข้อมูลในลักษณะชั่วคราวเช่นเป็นตัวเลขจากการคำนวณและพร้อมจะนำไปใช้งานต่อไป จากความจำเป็นนี้ ทำให้ต้องมีหน่วยความจำมากกว่า 1 ประเภท เช่น อาจเป็นหน่วยความจำถาวร และหน่วยความจำชั่วคราวที่สามารถเก็บและเรียกออกมาใช้งานได้ทุกเมื่อ จะแบ่งออกเป็น 2 ตระกูล คือตระกูลROM (Read Access Memory)และตระกูล RAM (Random Access Memory) หรือที่บางคนรู้จักในชื่อ RWM (Read/Write Memory) ส่วนตระกูล Sequentially – Access Memoryเช่นพวก ชิฟท์รีจิสเตอร์ เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คุณสมบัติที่น่าสนใจ 2 อย่างของหน่วยความจำ ได้แก่ Destructiveness คือการที่เราอ่านข้อมูลออกมาแล้ว หน่วยความจำยังจำข้อมูลนั้นได้หรือไม่ และVolatility คือการที่เมื่อแหล่งจ่ายไฟตับลง ข้อมูลในหน่าวยความจำจะสูญหายหรือไม่ หน่วยความจำในสมัยแรก ๆ ซึ่งเป็นพวก Magnetic Core Memory มักเป็น Destructive Memory แต่เป็นชนิด Nonvolatile ส่วนพวกหน่วยความจำที่ทำจากสารกึ่งตัวนำในสมัยปัจจุบัน มักเป็นพวก Non destructive Memoryแต่ Volatility ขึ้นอยู่กับชนิดและตระกูลของหน่วยความจำนั้น เช่น ROM มักเป็นพวก Nonvolatileส่วน RAM มักเป็นพวก Volatil จึงต้องได้รับการเอาใจใส่เป็นพิเศษ โดยเฉพาะในแง่ของแหล่งจ่ายไฟเลี้ยงวงจร

2.6.1 ตระกูล ROM

ROM (Read-Onry Memory) เป็นหน่วยความจำที่ใช้เก็บข้อมูลแบบถาวร หรือกึ่งถาวร นั่นคือข้อมูลต้องถูกเขียนไว้ในหน่วยความจำตั้งแต่ต้น หลังจากนั้นก็เป็นการอ่านข้อมูลออกมาเท่านั้น (read onry) การเขียนข้อมูลทำได้ 2 วิธีคือ การเขียนข้อมูลมาจากโรงงานผู้ผลิต ตามความต้องการของลูกค้า เช่นเป็นชนิด Mask-Programmed ส่วนอีกแบบหนึ่ง คือผู้ใช้สามารถโปรแกรมได้เองเรียกว่าเป็นชนิด PROM (Programmable Read-Onry Memory) และก็มีบางประเภทที่ผู้ใช้สามารถเปลี่ยนข้อมูลในหน่วยความจำได้ แต่ในการเขียนข้อมูลแต่ละครั้งต้องใช้เวลามากกว่าเวลาที่ใช้ในการ อ่านข้อมูลจากหน่วยความจำมาก ประเภทนี้เป็นชนิด Reprogrammable ROM หรือบางครั้งอาจเรียกว่า READ -Mostly Memory (RMM)

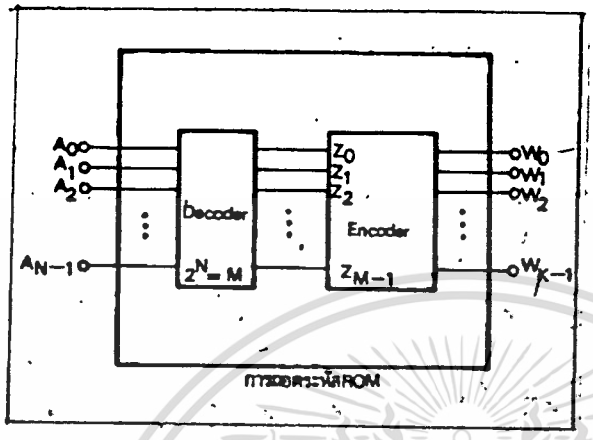
ข้อดีอย่างหนึ่งของหน่วยความจำตระกูลนี้ คือข้อมูลจะไม่สูญหาย ถึงแม้ไฟฟ้าจะตับก็ตาม เราเรียกลักษณะอย่างนี้ว่าnon-volatileซึ่งต่างไปจากหน่วยความจำตระกูล RAM ซึ่งเป็นชนิดvolatile

อย่างที่เราทราบไปแล้วว่า ROM นั้น ทำหน้าที่เป็นหน่วยความจำที่เก็บข้อมูลที่ค่อนข้างถาวร เช่นเก็บโปรแกรมที่ใช้บ่อยหรือสม่ำเสมอ แต่เราอาจมอง ROM ว่าทำหน้าที่แปลงรหัสจากชนิดหนึ่งไปเป็นอีกชนิดหนึ่งก็ได้ ในกรณีนี้ ROM ทำงานเป็นวงจรแปลงรหัส

ดังแสดงในรูป ซึ่งแสดงให้เห็นโครงสร้างภายในของ ROM ทั่วไป ตามรูป A_0 ถึง A_{N-1} เป็นแอดเดรสของข้อมูงที่ต้องการอ่านซึ่งหลังจากผ่านวงจรถอดรหัส จะมีเพียงสายเดียวจากจำนวนทั้งหมด $2^N = M$ สาย ที่ถูกกระตุ้น และหลังจากผ่านวงจรเข้ารหัสแล้ว ข้อมูลจะมาปรากฏที่ขั้วออก W_0 ถึง W_{K-1} จะเห็นว่าสัญญาณเข้าที่มี N บิท และทำหน้าที่เสมือนเป็นแอดเดรสจะถูกแปลงเป็นสัญญาณออก K บิท ซึ่งก็คือ ข้อมูลที่เราต้องการโดยไม่

เอกสารนี้สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สนใจว่า N จะมากกว่าหรือเท่ากับหรือน้อยกว่า K ในที่นี้จะบอกได้ว่า หน่วยความจำตระกูล ROM นี้ มีขนาดเป็น M x K บิท โดยที่ M คือจำนวนเวิร์ด ทั้งหมดในหน่วยความจำ และ K คือจำนวนบิทของแต่ละเวิร์ด



รูปแสดง โครงสร้างภายในของ ROM

2.6.2 ชนิดของ ROM

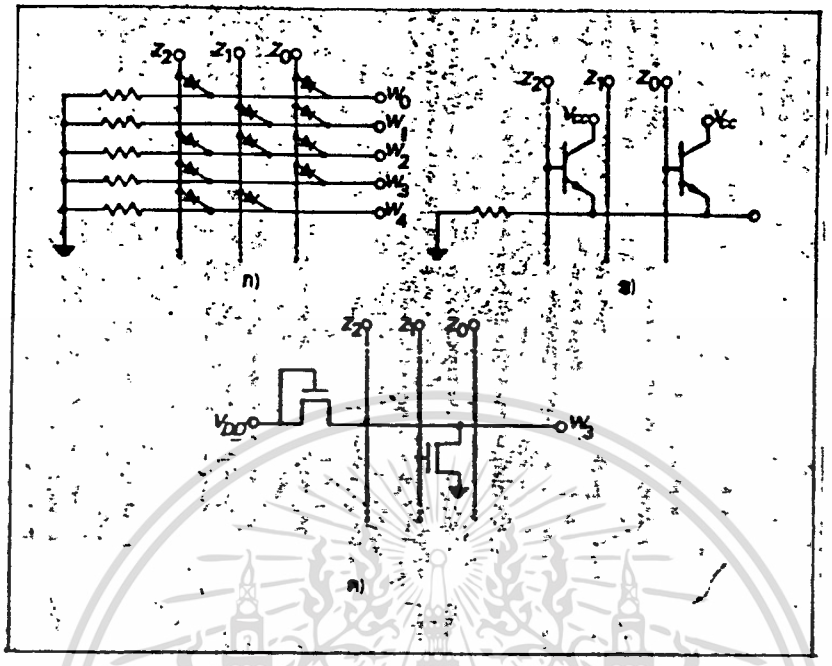
ต่อไปจะพูดถึงชนิดของ ROM ที่มีการผลิตและจำหน่ายในท้องตลาด

2.6.2.1 Mask-Programmed ROM

Mask-Programmed ROM เป็นหน่วยความจำที่โปรแกรมไว้อย่างถาวรในระหว่างขบวนการผลิต โดยการทำ mask บางอันเฉพาะสำหรับงานนั้น ๆ ROM ประเภทนี้สามารถทำได้จากสิ่งประดิษฐ์จำพวกไดโอด ทรานซิสเตอร์ (BJT) หรือ MOSFET ก็ได้ ดังแสดงในรูป ปัญหาใหญ่ของ ROM ที่ทำจากไดโอด (ดังรูป ก) ไดโอดเป็นสิ่งประดิษฐ์ชนิดพาสซีฟ ดังนั้น มันจึงไม่สามารถจ่ายโหลดได้มากนัก เพราะขั้วออกต่อกับขั้วเข้า ผ่านไดโอดโดยไม่มีการขยายกระแสเลย ปัญหานี้สามารถแก้ไขได้โดยใช้พวก สิ่งประดิษฐ์ชนิดแอคทีฟ (ดังรูป ข และรูป ค) ในทางปฏิบัติ BJT มักต้องการเนื้อที่ในชิ้นส่วนสารกึ่งตัวนำมากกว่าMOSFET อีกทั้งมีการสูญเสียกำลังค่อนข้างสูง ROM ที่ทำมาจาก BJT จึงมีจำนวนบิทน้อยกว่า ROM ROM ที่ทำจาก MOSFET บนชิ้นสารกึ่งตัวนำที่มีพื้นที่เท่ากัน แต่ข้อเสียของ MOSFET ก็มีได้แก่ความเร็วในการตอบสนอง (access time) ซึ่งหมายถึงช่วงเวลาที่ต้องรอช่วงรหัสหลังจากให้แอดเดรสแก่หน่วยความจำ จนกระทั่งได้ข้อมูลจากแอดเดรสนั้น ๆ มาปรากฏที่ขั้วออกของหน่วยความจำ จะเห็นได้จากรูปที่แสดง ซึ่งแสดงเฉพาะภาคเข้าวงจรเข้ารหัสของ ROM ว่า สายแอดเดรสและสายสัญญาณออกนั้นออกทั้งหมดจะประกอบกันเป็นตารางกริด ที่จุดตัด จะมีสิ่งประดิษฐ์ ซึ่งอาจเป็นไดโอดหรือ BJT หรือ MOSFET ก็ได้ ต่อเชื่อมโยงระหว่างสายแอดเดรสกับสายสัญญาณออกหรือไม่นั้น ขึ้นอยู่กับว่า ที่บิทนั้นของแอดเดรสนั้น ๆ ต้องการให้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็น 1 หรือ 0



รูปแสดง วงจรภายในของภาควงจรเข้ารหัสของ ROM

ก) ไดโอด

ข) ทรานซิสเตอร์ (BJT)

ค) MOSFET

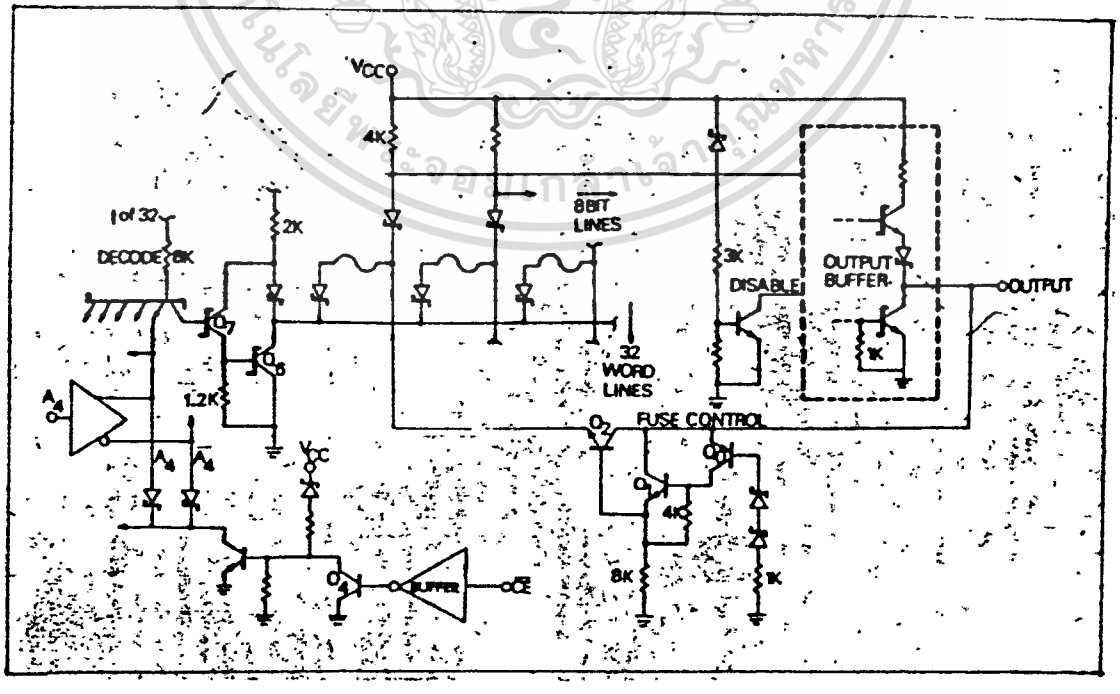
คุณสมบัติของ ROM ชนิดนี้ มีดังต่อไปนี้คือ เก็บข้อมูลได้ตลอดกาล แม้จะมีไฟเลี้ยงวงจรหรือไม่ก็ตาม (non-volatile) ผู้ผลิตเป็นผู้โปรแกรมข้อมูลในหน่วยความจำ และเราไม่สามารถเปลี่ยนแปลงข้อมูลได้ ข้อดีข้อเสีย ก็คล้ายคลึงกับของ ROM ชนิดอื่นแต่ต่างกันบ้าง ตรงที่ขนาดของหน่วยความจำ (จำนวนเวิร์ด) คูณกับจำนวนบิทต่อ 1 เวิร์ด อาจเล็กหรือใหญ่ตามต้องการได้ และจำนวนบิทต่อ 1 หน่วยพื้นที่ค่อนข้างสูง แต่กระนั้นก็ดี ราคาต่อ 1 หน่วยค่อนข้างสูงด้วย อันเนื่องมาจากค่าใช้จ่ายในการกระทำ mask ดังนั้น ROM ชนิดนี้จึงมีประโยชน์ เฉพาะกรณีที่มีความต้องการสูง มิฉะนั้นจะไม่คุ้มกับต้นทุนในการผลิต

2.6.2.2 Programmable ROM (PROM)

ที่จริงแล้ว PROM นั้น มีลักษณะวงจรภายในคล้ายคลึงกับพวก Mask-Programmed ROM และตามปกติมักจะทำเป็นตัวไอซี ที่มีขาอย่างเดียวกับ Mask-Programmed ROM ที่เป็นคู่ของมันด้วย (pin-for-pin compatible)ใน Mask-Programmed ROM ทางผู้ผลิตได้ต่อหรือไม่ต่อเชื่อมโยงสายแอดเดรสกับสายสัญญาณออก ด้วยสิ่งประดิษฐ์สารกึ่งตัวนำที่จุดตัดในตารางกริด ตามที่เราต้องการให้เป็น 1 หรือ 0 ไว้เรียบร้อย ส่วนใน PROM

ที่จุดตัดทุกจุดในตารางกริด จะมีสิ่งประดิษฐ์สารกึ่งตัวนำต่ออยู่ แต่การโปรแกรมให้ข้อมูลของแต่ละบิทเป็น 0 หรือ 1 นั้น ใช้วิธีปล่อยกระแสสูง ๆ ไหลผ่าน ฟิวส์ที่มีต่อร่วมกับสิ่งประดิษฐ์ เพื่อให้ฟิวส์หลอมขาด ตัดการต่อเชื่อมโยงที่บิทนั้น ๆ หลังจากโปรแกรมเสร็จ ก็จะเหลือบิทเฉพาะที่ต้องการให้มีการต่อเชื่อมโยงด้วยสิ่งประดิษฐ์ ลักษณะอย่างนี้ทำให้มีอีกชื่อหนึ่งว่า Field-ProgrammableROM ดังแสดงในรูป แสดงวงจรภายในบางส่วนของ PROM หมายเลข 821 S 23/123 ของบริษัทซิกเนติคส์

คุณสมบัติของ PROM ก็คือเป็นชนิด non-volatile ซึ่งโปรแกรมข้อมูลได้ แต่ไม่สามารถเปลี่ยนแปลงข้อมูลหลังจากที่โปรแกรมไปแล้วได้ เพราะฟิวส์ที่เคยต่ออยู่ได้ขาดไปแล้วและไม่มีทางต่อกลับถึงกันอย่างเดิมได้อีก ขนาดความจุของ PROM ไม่ค่อยสูงนัก อันเนื่องมาจากการมีฟิวส์ ประกอบร่วมอยู่ด้วย และสาเหตุข้อนี้ด้วย ที่ทำให้จำนวนบิทต่อ 1 หน่วยพื้นที่ต่ำกว่าของ Mask-ProgrammedROM ราคาต่อหน่วยไม่สูงเหมือน Mask-Programmed ROM เพราะปริมาณที่ผลิตมีมากกว่า แต่สิ่งหนึ่งที่ผู้ใช้จำเป็นต้องมีก็คือ เครื่องโปรแกรมข้อมูล (Programmed) ซึ่งก็มีราคาไม่แพงนักด้วย จึงเป็นที่นิยมของผู้ที่กำลังพัฒนาโปรแกรม หรือในกรณีที่ต้องการผลิตอุปกรณ์จำนวนน้อย ๆ อย่างไรก็ดี นับว่า PROM สามารถเข้ามาแทนที่ Mask-Programmed ROM ได้อย่างดี เพราะขนาดของหน่วยความจำ ก็สูงไม่แพ้ของ Mask-Programmed ROM มากนัก อีกทั้งราคาไม่แพง และโปรแกรมได้ง่ายด้วย

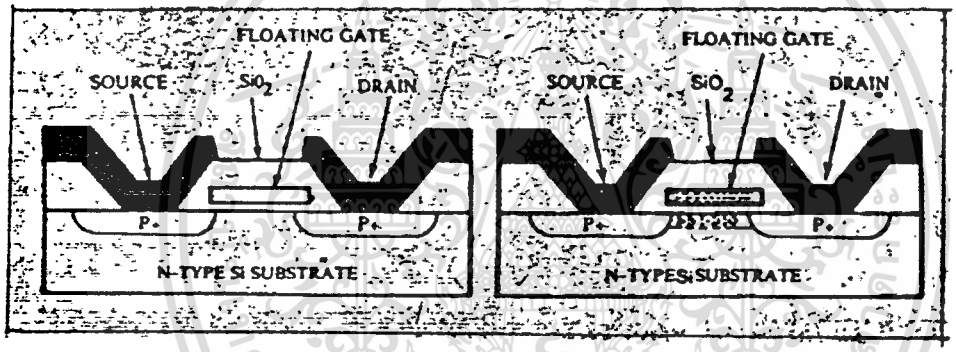


รูปแสดง วงจรภายในบางส่วนของ PROM เบอร์ 82 เอส 23/123 ของบริษัทซิกเนติคส์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6.2.3 Reprogrammable ROM

ลักษณะโครงสร้างภายในและแนวความคิดของ ROM ชนิดนี้ แตกต่างจากของ PROM มาก ตามปกติ Reprogrammable ROM ทำจาก MOSFET โดยที่สภาวะ 0 หรือ 1 แทนการนำกระแสหรือไม่นำกระแสของ MOSFET ซึ่งการนำกระแสหรือไม่ของ MOSFET นั้น ถูกควบคุมด้วยแรงดันที่ขาเกท เมื่อพิจารณาโครงสร้างของ MOSFET แล้ว เราจะเห็นว่า ขาเกทถูกแยกจากส่วนที่นำกระแสอย่างเด็ดขาด ดังแสดงในรูป ดังนั้น แรงดันที่ขาเกทจึงขึ้นอยู่กับประจุที่ขาเกท (ลักษณะโครงสร้างของขาเกทกับส่วนนำกระแสของ MOSFET นั้น เปรียบเสมือนตัวเก็บประจุ ที่มีชั้นออกไซด์ เป็นตัวกลางไดอิเลคตริค) นี่เอง เป็นแนวความคิดในการพัฒนาReprogrammable ROM ขึ้นมา โดยวิธีการกักประจุไว้ที่ขาเกท และประจุนั้นสามารถค้างอยู่ที่ขาเกทได้เป็นเวลานาน



รูปแสดง โครงสร้างของ MOSFET

แนวความคิดดังกล่าวยังแยกออกได้เป็น 2 วิธี ทั้งนี้ขึ้นอยู่กับผลของการกักประจุในขาเกท วิธีแรกคือการกักประจุไว้ที่ขาเกท ซึ่งถูกแยกจากส่วนนำกระแสของMOSFET ด้วยชั้นของ ซิลิกอน ไดออกไซด์ ทำให้ MOSFET นำกระแสได้ วิธีการกักประจุสามารถทำได้โดยการป้อนแรงดันไฟสูงเข้าที่ MOSFET ทำให้พาหะประจุทะลุผ่านชั้นออกไซด์เข้าไปค้างอยู่ที่ขาเกทได้ ส่วนวิธีการเปลี่ยนแปลง ข้อมูลที่ถูกโปรแกรมไปแล้ว ทำได้วิธีเดียวคือการลบ โดยฉายด้วยรังสีอุลตราไวโอเล็ตเพื่อให้พาหะประจุที่ถูกกักอยู่ มีพลังงานพอเพียงที่จะหลุดรอดออกมาได้ แต่วิธีการนี้ลบข้อมูลทั้งหมดของ ROM เราไม่สามารถเลือกลบข้อมูลเฉพาะบางบิทได้ หน้าต่างที่ปล่อยให้รังสีเข้าไป ต้องเป็นพวกควอทซ์ หรือวัสดุที่ยอมให้รังสีผ่านได้ ทำให้มีราคาสูงขึ้น เราเรียก ROM ชนิดนี้ว่า EPROM(Erasable Programmable ROM)

วิธีที่สอง ใช้สิ่งประดิษฐ์สารกึ่งตัวนำจำพวก MNOS (Metral Nitride Oxide Semiconductor FET การมีชั้นไนไตรด์อีกชั้นหนึ่ง เป็นการเพิ่มฉนวนอีกชั้นหนึ่งต่างหากจากชั้นออกไซด์ การกักประจุไว้ที่ผิวสัมผัสระหว่างฉนวนทั้งสอง จะเปลี่ยนแปลง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนำไปใช้

threshold voltage ของ FET (threshold voltage)คือแรงดันวิกฤติที่ขาเกท เทียบกับขาซอสของ FET ซึ่งพอดีทำให้ FET เริ่มนำกระแสได้) การอ่านข้อมูลออกมาสามารถทำได้โดยป้อนแรงดันไฟสูงพอที่จะทำให้ FET ที่มีการดักประจุอยู่นำกระแสได้ แต่ต่ำพอที่จะไม่ทำให้ FET ที่ไม่มีการดักประจุนำกระแส การดักประจุ สามารถทำได้โดยการป้อนแรงดันไฟสูงเข้าที่ MOSFET ทำให้ประจุทะลุผ่านออกไซด์ไปค้างอยู่ที่ผิวสัมผัสระหว่างชั้นออกไซด์กับชั้นไนไตรด์ ซึ่งเป็นขบวนการคล้าย ๆ กับของ EPROM แต่วิธีการลบข้อมูลแตกต่างกันน้อยออกไป การลบข้อมูลของ ROM ชนิดหลังนี้ทำได้โดยป้อนแรงดันไฟสูงกลับทางกับขณะโปรแกรมก็จะทำให้พาหะประจุที่ถูกดัก สามารถทะลุผ่านชั้นออกไซด์กลับมาได้แทนที่จะต้องใช้รังสีอุลตราไวโอเลตแบบเดียวกับของ EPROM ทำให้เราสามารถลบข้อมูลที่บิทใด ๆ ที่เราต้องการลบได้ เราเรียก ROM ชนิดนี้ว่า EAROM(Electrically Alterable ROM ปัญหาที่ EAROM ยังไม่แพร่หลาย อยู่ที่ขบวนการผลิต ซึ่งยังได้ผลไม่เป็นที่น่าพอใจ และพื้นที่ต่อจำนวนบิทเท่า ๆ กันของ EAROM มากกว่าของ EPROM

ข้อดีของ Reprogrammable ROMอยู่ที่ความสามารถในการเปลี่ยนแปลงข้อมูลในหน่วยความจำได้ ซึ่งสะดวกมากสำหรับการพัฒนาทาง Software แต่ในการให้ได้ประโยชน์เช่นนี้มา ก็ต้องลงทุนในด้านเครื่องโปรแกรม ROM ซึ่งมีราคาค่อนข้างสูง

2.6.3 ตระกูล RAM

RAM(Random-Access Memory) เป็นหน่วยความจำที่ต่างออกไปจาก ROM ทั้งในลักษณะโครงสร้างและการทำงาน เราสามารถเขียนและอ่านข้อมูลจาก RAM โดยใช้เวลาพอ ๆ กัน แต่กรณีของ EPROM การเขียนข้อมูลเข้าในเซลหน่วยความจำ ต้องใช้เวลานานกว่าการอ่านข้อมูลมากทีเดียว RAM สามารถทำจากสิ่งประดิษฐ์จำพวก BJT หรือ MOSFET ก็ได้ โดยมาก RAM ที่ทำจาก BJT มักมีความเร็วสูงกว่าพวกที่ทำจาก MOSFET แต่ในขณะเดียวกัน BJT RAM สิ้นเปลืองกำลังไฟสูง ดังนั้น ในไอซีตัวหนึ่ง ๆ จึงมักมีจำนวนเซลได้ไม่มาก

การเก็บข้อมูลในเซลหน่วยความจำแต่ละเซลของ RAM มี 2 ลักษณะคือ

1. การใช้ลักษณะวงจรฟลิบฟลอบเป็นเซลพื้นฐานในการจำข้อมูล เราใช้ลักษณะนี้มากในวงจร BJT RAM และวงจร MOS RAM บางชนิด

2. การใช้การประจุตัวเก็บประจุที่ขาเกทของ MOSFET เป็นการเก็บข้อมูล นิยมใช้มากในวงจร MOSRAM

จากลักษณะการเก็บข้อมูลในเซลหน่วยความจำทั้งสองวิธีนี้ จึงทำให้ RAM แบ่งได้

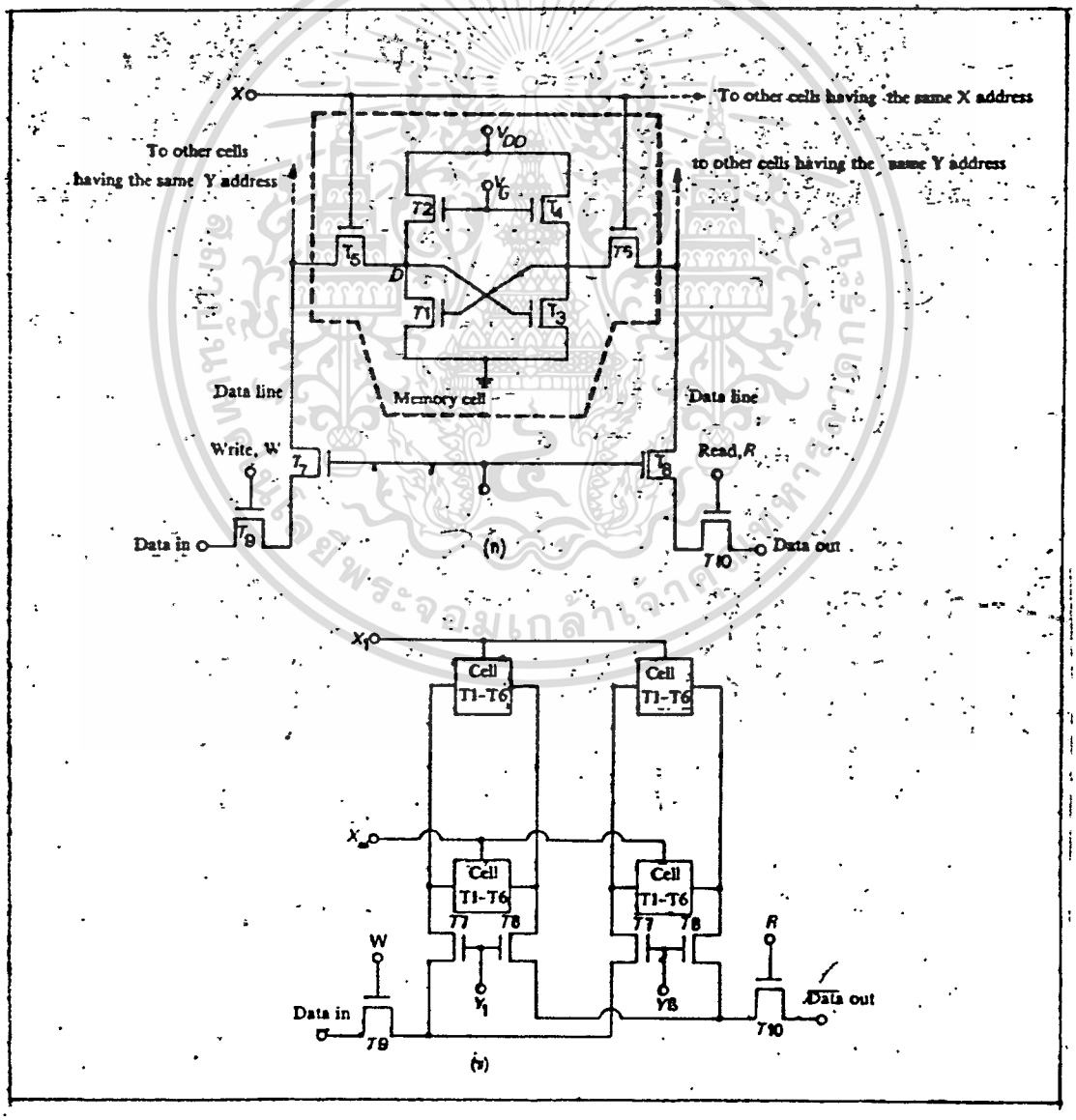
เอกสารนี้ใช้เพื่อการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็น 2 ชนิดคือ RAM ชนิดสตาติค และ RAM ชนิดไดนามิค

2.6.5 ชนิดของ RAM

2.6.5.1 RAM ชนิดสตาติค

RAM ชนิดนี้ใช้ลักษณะวงจรฟลิบฟลอบเป็นวงจรพื้นฐาน สำหรับเซลหน่วยความจำแต่ละเซล ตามปกติ วงจรพื้นฐานก็มักไม่แตกต่างกันมากนัก และโดยมากจะเป็นไปดังแสดงในรูป (ก) $T_1 - T_6$ คือทรานซิสเตอร์ที่ประกอบเป็นวงจรฟลิบฟลอบในเซลพื้นฐาน ส่วน $T_7 - T_{10}$ เป็นเกทที่เปิดให้ข้อมูลไหลผ่านไปมาได้ เป็นวงจรร่วมของเซลหน่วยความจำหลาย ๆ เซล ดังแสดงในรูป (ข) ดังนั้น เราจึงเรียกเซลหน่วยความจำแบบนี้ว่า เซลหน่วยความจำชนิดสตาติคแบบทรานซิสเตอร์ 6 ตัว



รูปแสดง (ก) เซลพื้นฐานของ RAM ชนิดสตาติค

(ข) การต่อเซลต่าง ๆ เข้าด้วยกันภายในตัวไอซี

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลักษณะการทำงานเป็นดังนี้ T_1,T_2 และ T_3, T_4 ประกอบกันเป็นวงจรคลับเฟสที่ต่อโยงไขว้กันอยู่ ทำหน้าที่เป็นฟลิบฟลอบ โดย T_5, T_6 เป็นเกทเปิดหรือปิดตามแต่จะเลือกด้วยระดับแรงดันในสาย X (มาจากวงจรถอดรหัสเลือกแถว) T_7, T_8 เป็นเกทเปิดหรือปิด ตามแต่จะเลือกด้วยระดับแรงดันในสาย Y (มาจากวงจรถอดรหัสเลือกคอลัมน์) T_9,T_{10} เลือกว่าจะให้เขียนข้อมูลเข้าไปหรืออ่านข้อมูลออกมา

เมื่อต้องการเลือกเซลใด ก็ให้สาย X, Y ของเซลนั้นเป็น 1 ทำให้สาย Data และสาย Data ต่อกับเซลได้ และยังต่อออกมาที่สาย Data in หรือสาย Data out ก็ได้ ด้วยการให้ W หรือ R เป็น 1 ตามลำดับ ถ้าต้องการเขียนข้อมูลเข้าไปในเซล ก็ให้ข้อมูลนั้นปรากฏที่สาย Data inและ W = 1 ตามปกติเรามักให้ R = $\overline{W}$ = 0 เมื่อต้องการอ่านข้อมูลออกมา ก็ให้ R = 1, W = 0 ข้อมูลที่ถูกเก็บเป็นสภาวะของ T_1 จะถูกคลับเฟส โดย T_3 ผ่าน T_6, T_8, T_{10} ออกมาเป็น Data

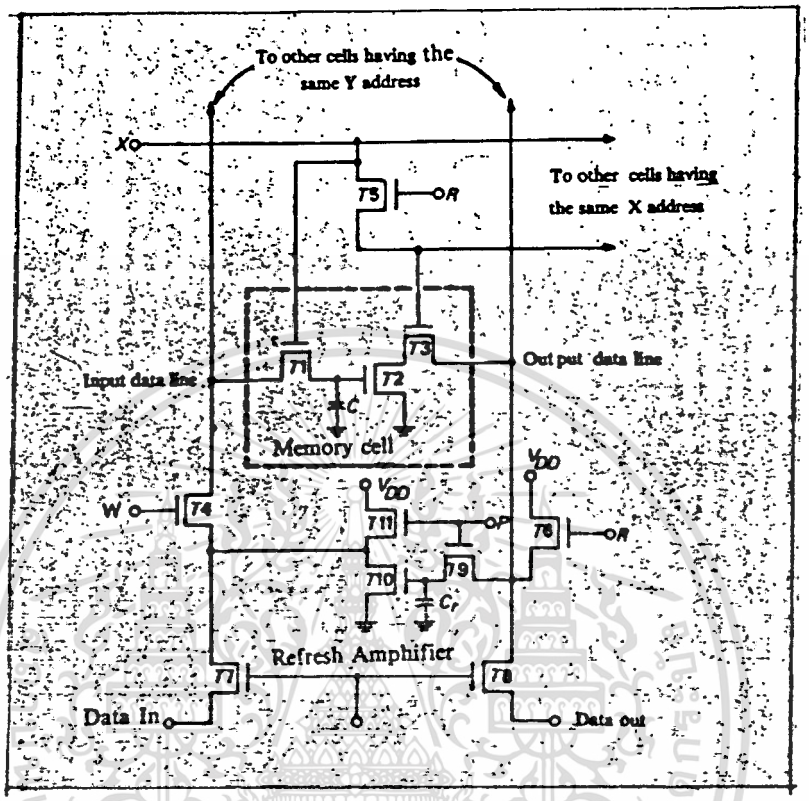
จากรูปที่แสดง แสดงการต่อเซลหลาย ๆ เซลเข้าด้วยกันเป็นหน่วยความจำ ขนาดใหญ่ขึ้น โดยมีเซลพื้นฐาน T_1 - T_6 เป็นองค์ประกอบของแต่ละเซล ส่วนการทำงานยังคงเหมือนที่กล่าวไปแล้ว เพียงแต่มีสาย X มากขึ้น เป็น 2 สาย และสาย Y มากขึ้นเป็นสามสาย โดยมากทั้งสาย X, Y นี้ มาจากวงจรถอดรหัสเลือกแถวและคอลัมน์ ที่อยู่ในตัวไอซี และตามปกติแอดเดรสที่จะเข้ามักอยู่ในรูปรหัสเลขฐาน 2 n บิท (จึงมีได้ทั้งหมด 2^n แอดเดรส การบอกขนาดของ RAM ก็เช่นเดียวกับใน ROM คือบอกเป็น M xK บิท เมื่อ M คือจำนวนเวิร์ด และ K คือจำนวนบิทของแต่ละเวิร์ด และหรือจำนวนบิททั้งหมดที่สามารถเข้าถึงได้ด้วยแอดเดรสเดียวกัน

2.6.5.2 <u>RAM ชนิดไดนามิค</u>

RAMชนิดนี้ใช้การเก็บประจุในตัวเก็บประจุที่ขาเกทของ MOSFET แทนการเก็บข้อมูล วงจรพื้นฐานของ RAM ชนิดไดนามิคมีได้หลายรูปแบบ และใช้จำนวนทรานซิสเตอร์แตกต่างกันไป RAMชนิดไดนามิคได้มีข้อดีเหนือกว่า RAM ชนิดสตาติคมากนัก นอกจากการมีเวลาเข้าถึงหน่วยความจำน้อยกว่าของ RAM ชนิดสตาติคเล็กน้อย แต่ต้องเสียเวลาในการรีเฟรชช่วยความจำทั้งหมด การมีการสิ้นเปลืองกำลังไฟฟ้าต่ำกว่าของ RAM ชนิดสตาติคอันเนื่องจากกระแสไหลผ่านวงจร ขณะเขียนหรืออ่านรีเฟรชเท่านั้น แต่สิ่งที่น่าสนใจอย่างยิ่งก็คือการใช้จำนวนทรานซิสเตอร์ต่อ 1 เซลน้อยกว่าของ RAM ชนิดสตาติคค่อนข้างมาก ทำให้เนื้อที่ 1 ตารางหน่วย สามารถบรรจุ RAM ชนิดไดนามิค ได้จำนวนเซลมากกว่าของ RAM ชนิดสตาติคค่อนข้างมาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ที่จริงแล้ว แต่ละเซลของ RAM ชนิดไดนามิคอาจประกอบด้วยทรานซิสเตอร์ 1 ตัว และตัวเก็บประจุ 1 ตัวเท่านั้น (คือ T_1 และ C ดังแสดงในรูป)



รูปแสดง เซลหน่วยความจำของ RAM ชนิดไดนามิคแบบทรานซิสเตอร์ 3 ตัว

แต่ปัญหาใหญ่ของ RAM แบบทรานซิสเตอร์ 1 ตัวก็คือ ในขณะที่อ่านข้อมูลออกมา ประจุที่เก็บไว้จะถูกคายข้อมูลออกมาในสาย Data ซึ่งตามปกติมักมีค่าความจุค่อนข้างสูงจนเกือบหมด ทำให้การทำงานของ RAM แบบนี้เป็นลักษณะ Destructive ทำให้มีการลบข้อมูลทุกครั้งที่มีการอ่านข้อมูลของเซลนั้น ๆ แต่ถ้าให้ค่า C ใหญ่กว่าค่าความจุในสายมาก ๆ ก็คือต้องเสียเวลาในการเขียนข้อมูลนานเกินไป เพราะต้องเสียเวลาในการประจุ C นานขึ้น

เพื่อแก้ปัญหาของ RAM ชนิดไดนามิคส์แบบทรานซิสเตอร์ 1 ตัว จึงได้เติมทรานซิสเตอร์ขึ้นอีก 2 ตัว (T_2, T_3 ดังแสดงในรูปบน) เพื่อแยกทางสำหรับเขียนข้อมูลเก็บไว้ใน C และอ่านข้อมูลออกมา (T_2,T_3 ทำหน้าที่เบียบัฟเฟอร์กลับเฟส ระหว่าง C กับสาย $\overline{\text{Data out}}$) T_1 - T_3 เป็นทรานซิสเตอร์ที่ประกอบกันเป็นวงจรพื้นฐานของเซลหน่วยความจำ T_4 - T_6 เป็นทรานซิสเตอร์สำหรับเลือกว่าเป็นการเขียนหรืออ่านข้อมูล T_7-T_8 ถูกควบคุมโดยสาย Y (มาจากวงจรถอดรหัสเลือกคอลัมน์) T_9 - T_{11} เป็นวงจรรีเฟช ขอให้สังเกตว่า T_9 - T_{11} ต่อกันเป็นวงจรลักษณะเดียวกันกับ T_1 - T_3 การที่เรา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ต้องมีวงจรรีเฟรชก็เพราะตามปกติ C มีการรั่วไหลอยู่บ้าง ดังนั้น หลังจากเวลาผ่านไปเล็กน้อย ประจุที่เคยมีอยู่ใน C จะรั่วไหล จนเปลี่ยนสภาวะไปได้ และทำให้ข้อมูลที่จำไว้ผิดพลาดไป

เพื่อจะเข้าถึงเซลใดเซลหนึ่ง เราต้องให้สาย X และ Y ของเซลนั้นเป็น 1 และแยกวงจรรีเฟรชออกจากเซลหน่วยความจำ โดยให้ P = 0 ถ้าต้องการเขียนข้อมูลเข้าในเซลหน่วยความจำ เราให้ W = 1, R = $\overline{W}$ = 0 ข้อมูลจากสาย Data in ผ่าน T_7, T_4, T_1 และจะประจุให้ C หรือไม่ ขึ้นกับสภาวะของข้อมูลเข้า ในทำนองคล้ายกัน ถ้าต้องการอ่านข้อมูลจากเซลหน่วยความจำ ต้องให้ R = 1, W = 0, T_5, T_6, T_3 จะนำกระแส โดย T_2 เป็นวงจรกลับเฟรสของข้อมูลที่เก็บใน C ออกมาปรากฏในสาย Data out

พอเวลาผ่านไปเล็กน้อย เราต้องรีเฟรชหน่วยความจำทั้งหมด โดยให้ Y = 0, X = 1, P = 1 และ R = 1, T_7 - T_8 จะหยุดนำกระแสข้อมูลใน C ถูกกลับเฟส โดย T_2 และผ่าน T_9 มาประจุให้ C_r เราเรียกช่วงขณะนี้เองว่าช่วงเวลาประจุล่วงหน้า(Precharge period) หลังจากให้ประจุแก่ C_r เต็มที่แล้ว ก็ให้ R = 0, W =1, T_{10} จะกลับเฟสของข้อมูลที่เก็บไว้ใน C_r ผ่าน T_4, T_1 กลับไปประจุให้ C ใหม่อีกครั้งหนึ่ง จะเห็นว่าข้อมูลของ C ถูกกลับเฟส 2 ครั้ง ด้วยวงจรกลับเฟส T_2, T_3, T_6 และ T_{10}, T_{11} ดังนั้น เมื่อรีเฟรชกลับข้อมูลมายัง C จึงไม่มีการกลับเฟสแต่อย่างไร ในขณะประจุล่วงหน้าต้องระวังมิให้ T_4 นำกระแส มิฉะนั้น T_{11} จะประจุให้ C โดยผ่านให้ T_4, T_1 ในการรีเฟรชหน่วยความจำ เราจะรีเฟรชทีละแถว (แถว X_0 , X_1, X_2,.....ตามลำดับ) ดังนั้น ถ้าหากมีทั้งหมด β คอลัมน์ ต้องมีวงจรรีเฟรช β วงจรด้วย เพื่อให้ β เซลในแถว X ใด ๆ ถูกรีเฟรชพร้อมกัน เวลาในการรีเฟรชหน่วยความจำทั้งหมด α แถว จะเป็นα t_r เมื่อ t_r คือเวลาในการรีเฟรชแต่ละแถว

2.6.5.3 <u>การรีเฟรช RAM ชนิดไดนามิค</u>

อย่างที่ทราบไปแล้วว่า RAM ชนิดไดนามิคอาศัยการเก็บประจุของการตัวเก็บประจุ ซึ่งมีการรั่วไหลเป็นธรรมดา ทำให้ข้อมูลสูญหายไปภายในเวลาไม่กี่มิลลิวินาที ข้อเสียนี้สามารถแก้ไขได้ด้วยการรีเฟรชข้อมูลในเซลหน่วยความจำทั้งหมดในทุก ๆ 2 มิลลิวินาที การรีเฟรชนี้จะอธิบายวิธีการรีเฟรชที่มีใช้กันอยู่ในปัจจุบัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เทคนิคการรีเฟรชพื้นฐานมีอยู่ 2 โหมดดังนี้คือ

1.Burst Mode เป็นโหมดการรีเฟรชแต่ละแถวตามลำดับจนครบทุกแถว โดยไม่มีการขาดตอนเลย ดู ๆ แล้วมันเป็นวิธีที่ง่าย แต่ก็มีข้อเสียเหมือนกัน ได้แก่การที่ไมโครโปรเซสเซอร์ ไม่สามารถเข้าถึงหน่วยความจำในขณะรีเฟรชได้ ซึ่งในบางครั้งก็คิดเป็นช่วงเวลานานพอสมควร เราเรียกช่วงเวลาที่สูญเสียนี้ไปว่า overhead ยกตัวอย่างเช่น RAM ชนิดไดนามิคขนาด 4Kx1 ซึ่งจัดเป็นเมตริกขนาด 64 x 64 จึงต้องใช้การรีเฟรชทั้งหมด 64 ไซเคิลทุก ๆ 2 มิลลิวินาที สมมุติว่าแต่ละไซเคิลต้องเสียเวลา 500 นาโนวินาที นั่นคือ ทุก ๆ 2 มิลลิวินาทีต้องเสียเวลาในการรีเฟรชไป 64 x .5 = 32 ไมโครวินาที ซึ่งคิดเป็นอัตราสูญเสีย $\frac{32}{2000}$ เท่ากับ 1.6 % ในบางครั้งอัตราสูญเสียนี้ก็มีความสำคัญมาก ดังนั้น เราจึงมักการเลี่ยงการรีเฟรชวิธีนี้ และหันไปใช้วิธีอื่นแทน

2.Distributed หรือ Single - Cycle Mode แทนที่จะใช้วิธีเฟรชทุกแถวรวดเดียว วงจรรีเฟรชจะทำการรีเฟรชแถวต่อแถวทุก ๆ n ไมโครวินาที ทำให้ไมโครโปรเซสเซอร์ทำงานมีประสิทธิภาพมากขึ้น

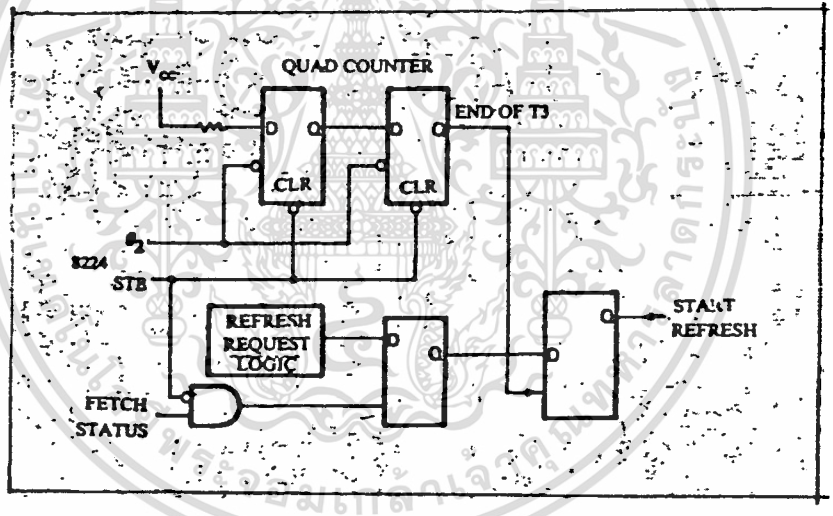
ไม่ว่าจะเป็นวิธีใดก็ตาม การรีเฟรชจะเกิดขึ้นได้ก็ต่อเมื่อ หน่วยความจำนั้นต้องไม่อยู่การใช้งาน ทั้งนี้ การรีเฟรชอาจกระทำ ในลักษณะที่ซิงโครไนซ์กับสัญญาณนาฬิกาของระบบหรือไม่ซิงโครไนซ์ก็ได้

1. Asynchronous Access สัญญาณของการทำรีเฟรชจะถูกส่งออกมาด้วยอัตราคงที่เช่น ทุก ๆ 31 ไมโครวินาที (ในกรณีที่มี 64 แถว) โดยไม่ขึ้นกับสภาวะของไมโครโปรเซสเซอร์ ดังนั้น วิธีการนี้จึงไม่ขึ้นกับชนิดหรือเบอร์ของไมโครโปรเซสเซอร์ ข้อเสียคือต้องใช้วงจรควบคุมการรีเฟรชที่ยุ่งยาก และแน่นอน ผลเสียที่ตามมาอีกอย่างหนึ่งก็คือ ปัญหาความสำคัญก่อนหลัง (priority)ในการเข้าถึงหน่วยความจำ

2. Synchronous Access ซึ่งเรายังรู้จักกันในชื่ออื่นอีก อย่างเช่น Hidden refresh หรือTransparent refresh จากชื่อหลังทั้งสองนี้ ทำให้พอนึกภาพออกได้ว่า มันเป็นวิธีการรีเฟรชหน่วยความจำในขณะที่ MPU ไม่ได้เรียกใช้งาน ในการใช้งานโดยทั่วไป มันมีช่วงเวลาอยู่อย่างน้อยหนึ่งไซเคิล ที่ MPU ไม่ได้ใช้งานหน่วยความจำ และถ้าเราสามารถทราบช่วงเวลานี้ได้อย่างแน่ชัด เราก็สามารถทำการรีเฟรชหน่วยความจำได้โดยที่ MPU ไม่ได้สูญเสียประสิทธิภาพอะไร (เรียกได้ว่าไม่มีโอเวอร์เฮด) แน่นอน เมื่อ MPU ไม่มีการสูญเสียเวลาไป MPU จึงย่อมไม่รู้ว่ามีการรีเฟรช หน่วยความจำขึ้นในจังหวะนี้ วิธีการนี้ข้อดีในเรื่องความเร็ว แต่ข้อเสียก็มีเช่นกัน ได้แก่วงจรควบคุมการรีเฟรชของไมโครโปรเซสเซอร์ชนิดใด ก็เป็นของชนิดนั้น เราไม่สามารถนำไปใช้กับไมโครโปรเซสเซอร์

ชนิดอื่นได้ นอกจากนี้เรายังต้องระวังเกี่ยวกับเหตุการณ์ที่ผิดธรรมดา เช่นกรณีของการใช้ไมโครโปรเซสเซอร์ 8080เอ ในการใช้คำสั่ง HALT, RESET คำสั่ง WAIT นาน ๆ ในกรณีที่ใช้หน่วยความจำที่มีความเร็วต่ำ หรือในกรณีที่ทำงานแบบทีละจังหวะ (single step และภาวะ Hold ในระหว่างของ DMA ทั้งนี้การระมัดระวังจะต้องรวมไปจนถึงการมีวงจรลอจิก ควบคุมให้มีการโอเวอร์ไรด์ เหตุการณ์เหล่านี้ เพื่อให้มีการรีเฟรชหน่วยความจำทุก ๆ 2 มิลลิวินาทีตามเงื่อนไขได้

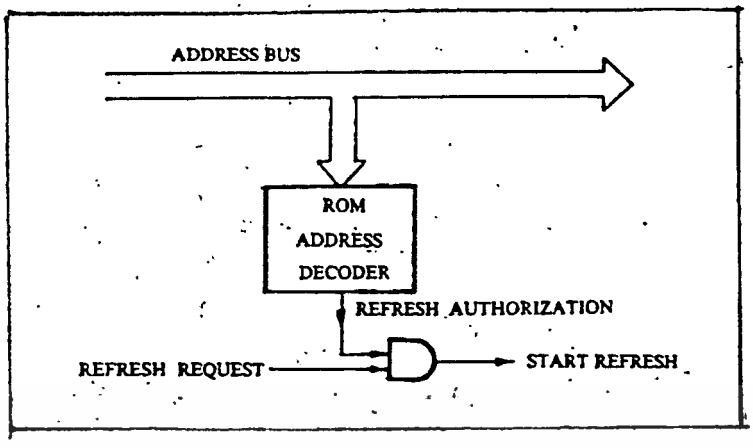
ในกรณีของไมโครโปรเซสเซอร์ 8080 เอ เราสามารถใช้จังหวะ T_4 ของแมชีลไซเคิล M_1 ในการรีเฟรชหน่วยความจำได้ จังหวะนี้เป็นจังหวะที่ MPU กำลังถอดรหัสและทำคำสั่งภายใน จึงไม่จำเป็นต้องใช้งานหน่วยความจำ ตามแสดงในรูป แสดงวงจรควบคุมการรีเฟรชแบบซิงโครนัสของไมโครโปรเซสเซอร์ 8085 เอ ตามรูปจะมีวงจรนับสี่ เป็นวงจรคอยตรวจสอบการสิ้นสุดของจังหวะ T_3 เพื่อนำไปสร้างสัญญาณเริ่มต้นการรีเฟรช



รูปแสดง วงจรควบคุมการรีเฟรชแบบซิงโครนัสที่จังหวะ T_4 ของแมชีลไซเคิล M_1 ของ 8080 เอ

เรายังสามารถรีเฟรชหน่วยความจำได้ขณะที่ใช้งาน ROM ตามแสดงในรูป แสดงวงจรควบคุมการรีเฟรชในขณะใช้งาน ROM โดยนำแอดเดรสมาถอดรหัส และตรวจสอบดูว่าเป็นแอดเดรสของ ROM หรือไม่ ถ้าเป็นก็สามารถนำไปสร้างสัญญาณเริ่มต้นการรีเฟรชได้ นอกจากนี้ยังมีวิธีการรีเฟรชอย่างอื่นอีก ทั้งนี้ขึ้นอยู่กับโครงสร้างไมโครโปรเซสเซอร์แต่ละชนิด ยกตัวอย่างเช่น เราทราบว่าในขณะที่สัญญาณนาฬิกา ϕ_2 เปลี่ยนจากระดับ 0 เป็นระดับ 1 ตัว 8080 เอ ยังไม่มีความจำเป็นต้องใช้หน่วยความจำ เราก็สามารถใช้ช่วงเวลาของการเปลี่ยนแปลงจากระดับ 0 เป็น 1 ของสัญญาณนาฬิกา ϕ_2 มากำหนดหารีเฟรชแบบไม่ซิงโครนัสได้ จึงลดทอนความยุ่งยากของวงจรลงไปได้มาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง วงจรควบคุมการรีเฟรชในขณะใช้งาน ROM

ในการรีเฟรชหน่วยความจำ ไม่ว่าจะเป็นแบบใดก็ตาม ถ้าเราต้องการให้มีการรีเฟรชที่สมบูรณ์และน่าเชื่อถือเพียงพอ วงจรตรรกที่ใช้ในการรีเฟรชก็ต้องประกอบด้วยองค์ประกอบต่าง ๆ หลายส่วนด้วยกัน สำหรับระบบที่ค่อนข้างสมบูรณ์ ก็มักจะประกอบด้วยส่วนต่าง ๆ ดังต่อไปนี้

1. วงจรนับการรีเฟรช ซึ่งมักเป็นวงจรนับชนิด 6 หรือ 7 บิท เพื่อเป็นแอดเดรสของหน่วยความจำที่ต้องการรีเฟรช

2. วงจรมัลติเพลกซ์แอดเดรส เพื่อจ่ายแอดเดรสเลือกแถว ของเซลหน่วยความจำใน RAM มันจะเลือกที่ว่าแอดเดรสที่ว่านี้มา จากวงจรนับการรีเฟรช (ในจังหวะรีเฟรช) หรือจากบัสแอดเดรส (ในจังหวะใช้หน่วยความจำยามปกติ)

3. หน่วยอนุญาตการขอใช้หน่วยความจำ เพื่อยอมให้มีการรีเฟรชหน่วยความจำ หรือยอมให้ MPU เข้าถึงหน่วยความจำได้

4. วงจรแจกแจงความสำคัญก่อนหลัง เป็นส่วนที่จะอนุญาตให้มีการรีเฟรชได้ตามเงื่อนไขที่กำหนดขึ้น

5. วงจรกำหนดสัญญาณรีเฟรช เพื่อป้อนสัญญาณพัลส์ ด้วยอัตราที่ต้องการ เช่น 64 ครั้งทุก ๆ 2 มิลลิวินาที

6. แลทช์ เพื่อใช้จำสภาวะต่าง ๆ ทั้งหมดก่อนที่จะมีการรีเฟรช

2.6.6 <u>ระบบหน่วยความจำ</u>

เนื่องจากระบบหน่วยความจำ มักใช้ควบคู่กับโปรเซสเซอร์ไม่อย่างใดก็อย่างหนึ่ง ดังนั้น ในการออกแบบระบบหน่วยความจำ เราจะต้องคำนึงถึงสเปคของโปรเซสเซอร์ที่มันใช้ประกอบอยู่ด้วย สิ่งที่สำคัญที่สุดในการเลือกใช้หน่วยความจำแบบใด เบอร์ใด จะขึ้นอยู่กับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความยากง่ายและความเข้ากันได้กับโปรเซสเซอร์ในระบบนั้น สเปคที่เกี่ยวข้องในการเลือกใช้หน่วยความจำต่าง ๆ ก็ได้แก่

1. Read Access Time ความเร็วในการให้ข้อมูลออกมา เมื่อได้รับแอดเดรสเข้าไปแล้ว

2.Write Setup Time ช่วงเวลาที่ต้องให้ข้อมูลคงที่ ในขณะที่เขียนข้อมูลเข้าในหน่วยความจำ

3. ความสามารถในการขับ ซึ่งหมายรวมถึงกระแสที่ต้องการ และโหลดชนิดตัวเก็บประจุ ทั้งในบัสแอดเดรสและบัสข้อมูล

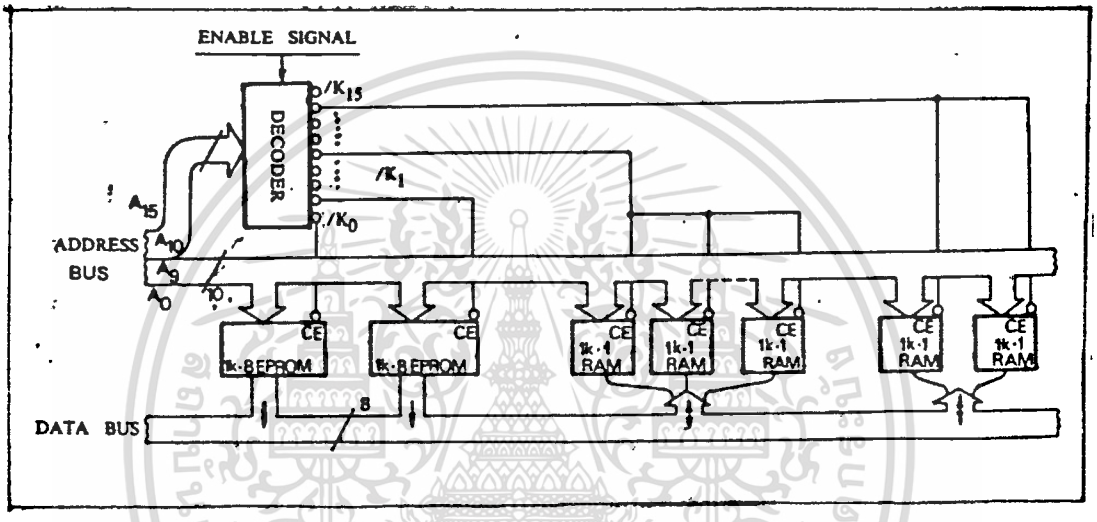
เนื่องจากหน่วยความจำที่นิยมใช้ในระบบไมโครโปรเซสเซอร์ มีอยู่ 2 ชนิดคือ ROM และ RAM เราจึงต้องสามารถกะประมาณ ขนาดของหน่วยความจำทั้ง 2 แบบให้ได้ แล้วเลือกใช้หน่วยความจำ ตามที่มีขายในตลาด สำหรับ ROM นั้น ที่นิยมใช้กันก็คือ PROM และ EPROM ส่วน Mask-Programmed ROM นั้น นิยมใช้ในกรณีที่ต้องการใช้จำนวนมาก มิฉะนั้นจะไม่คุ้มกับที่ลงทุนผลิตออกมา ถ้าหากสิ่งที่ต้องการเก็บไว้ใน ROM อาจเปลี่ยนแปลงได้ EPROM จะเป็นที่นิยมกว่า PROM สำหรับ EPROM ที่นิยมใช้ในปัจจุบัน มีอยู่หลายเบอร์ เช่น 1702 เอ (256 คูณ 8 บิท) 2708 (1024 คูณ 8 บิท) และที่เพิ่งออกมาเมื่อไม่นานนี้เองคือ 2716 (2048 คูณ 8 บิท)

ในกรณีของ RAM มีทางเลือกได้มากกว่า เพราะมีทั้ง RAM ชนิดสตาติคและ RAM ชนิดไดนามิค RAM ชนิดสตาติคมักนิยม ใช้ในกรณีที่ระบบหน่วยความจำมีจำนวนบิทไม่มากกว่าพวก RAM ชนิดไดนามิค จึงขอพิจารณาเฉพาะ RAM ชนิดสตาติคที่นิยมใช้กันมาก เห็นจะได้แก่ RAM ที่มีขนาด 1K bit ต่อไอซี 1 ตัว ซึ่งอาจอยู่ในลักษณะ 128 คูณ 8, 256 คูณ 4 และ 1024 คูณ 1 ทั้งนี้อาจมีวิธีเลือกมาจากจำนวนเวิร์ดหรือไบท์ที่ต้องการ และโดยมากการใช้ RAM ขนาด 1Kx 1 บิท จำนวนหลายตัว ดูจะเป็นที่นิยมกว่า RAM ขนาด 256 คูณ 4 บิท หรือ 128 คูณ 8 บิท ที่ใช้จำนวนไอซีเท่ากัน ทั้งนี้เพราะ มีปัญหาในแง่ของแต่ของโหลดชนิดตัวเก็บประจุในบัสข้อมูลน้อยกว่า

จากรูปที่แสดง แสดงการต่อระบบหน่วยความจำที่ใช้ EPROM ขนาด 1Kx1 บิท จำนวน 2 ตัว (เช่นเบอร์ 2708) และ RAM ขนาด 1K x 1 บิท จำนวนหลาย ๆ ตัว (เช่นเบอร์ 2102) ในบัสแอดเดรส มีทั้งหมด 16 สาย จึงสามารถต่อกับหน่วยความจำได้สูงถึง 2^{16} เท่ากับ 65,536 เวิร์ด ทั้งนี้รวมทั้งของ EPROM และ RAM (ในที่นี้แต่ละเวิร์ด มี 8 บิท) ในบัสข้อมูล มีทั้งหมด 8 สาย ตามปกติไอซี 1 ตัว ไม่สามารถมีจำนวน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

เวิร์ดและจำนวนบิทได้มากตามต้องการ เช่นนี้จึงต้องใช้ไอซีหลายตัว และเนื่องจากข้อมูลจากไอซี แต่ละตัวต้องมาปรากฏในบัสข้อมูล ซึ่งเป็นสายสัญญาณร่วม จึงนิยมใช้วิธีต่อขั้วออกเข้าด้วยกันในลักษณะ OR - Tie นั่นคือ D_0 ของไอซี ทุกตัวต่อถึงกันหมด D_1 ก็เช่นกันจนถึง D_7 ทีนี้ในการใช้งาน เมื่อต้องการเข้าถึงเซลหน่วยความจำในไอซี ตัวไหน เราเลือกให้ไอซีตัวนั้นทำงาน โดยให้ขา $\overline{CE}$ ของไอซีตัวนั้นเป็นสภาวะ 0 มนขณะที่ตัวอื่นไม่ทำงาน (ขา $\overline{CE}$ เป็นสภาวะ 1) ข้อนี้ต้องการการระวังเป็นพิเศษ มิฉะนั้นข้อมูลที่ได้มาอาจผิดพลาด



ระบบหน่วยความจำขนาด 8 บิทที่ประกอบจาก EPROM(2708) ANDRAM(2102A

เรามักแบ่งหน่วยความจำเป็นล็อค (block) หรือหน้า (page) โดยกำหนดขนาดจากหน่วยความจำที่สามารถเข้าถึงได้โดยไม่ต้องมีวงจรอื่นมาประกอบ เช่นในที่นี้ เราสามารถเข้าถึงหน่วยความจำขนาด 1K เวิร์ดได้โดยตรง เพราะทั้ง EPROM และ RAM แต่ละตัวมีขนาด 1K เวิร์ดอยู่แล้วทั้งสิ้น (แต่ RAM ต้องใช้ไอซี 8 ตัว เพื่อให้ครบ 8 บิทตามต้องการ) หากต้องการมีไอซีจำนวนมากก็จำเป็นต้องใช้ไอซีวงจรถอดรหัสมาแปลงเป็นสัญญาณชิพแอนเอเบิ้ล สำหรับกระตุ้นให้ไอซี บล็อคใดบล็อคหนึ่งทำงาน ในรูปสาย A_0-A_9 สามารถเข้าไปยังไอซีทุกตัวได้ (ปัญหาในที่นี้คือความสามารถในการขับ ถ้าจำเป็นก็ควรมีบัฟเฟอร์ประกอบด้วย) ส่วนสายแอดเดรส A_{10} - A_{15} จะผ่านเข้ายังวงจรถอดรหัสมาเป็นเพียง 1 ใน 2^6 สายเท่านั้น ที่มีสภาวะเป็น 0 นอกนั้นล้วนเป็น 1 ทั้งสิ้น เพื่อกระตุ้นบล็อคใดบล็อคหนึ่งทำงาน

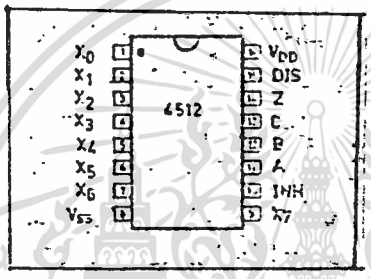
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.7 โครงสร้างของอินพุท-เอาท์พุท

โครงสร้างของอินพุทและเอาท์พุท เป็นโครงสร้างที่ง่ายต่อการเข้าใจ เพราะระบบของ MC 14500 จะมีบัสที่ติดต่อกับ อินพุท-เอาท์พุทได้ทีละบิท ถ้าหากจะอ่านข้อมูลจากอินพุทก็จะต้องใช้หลักการ ของตัวเลือกข้อมูลเพื่อจะกำหนดบิทที่ต้องการเลือก เพื่อทำการอ่าน สำหรับเอาท์พุทก็จะแลทช์ยังบิทที่ต้องการใช้

2.7.1 อินพุทด้วยตัวเลือกข้อมูล MC 14512

ตัวเลือกข้อมูล MC 14512 เป็นไอซี CMOS ที่ใช้งานง่ายโดยมีตัวกำหนดแอดเดรส 3 บิท คือ A, B, C เพื่อเลือกอินพุท $X_0 - X_7$ มายังเอาท์พุท Z โดยมีการจัดขาไอซีดังแสดงในรูป



รูปแสดง การจัดขาของไอซี MC14512

การทำงานของตัวเลือกข้อมูลจะกำหนดจังหวะเวลาด้วยขา INH และ DIS ด้วย โดยมีตารางการทำงาน ดังตารางที่แสดง

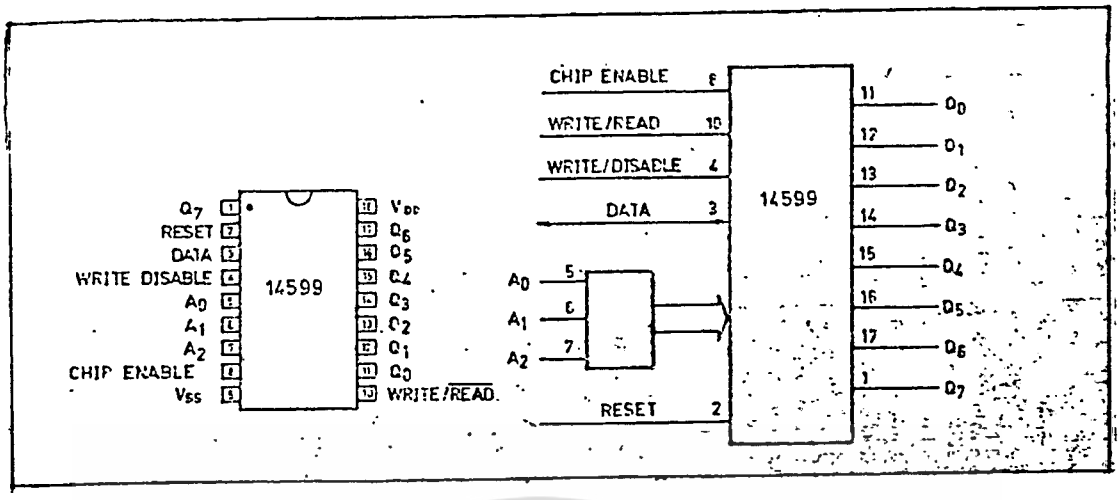
ตาราง แสดงการทำงานของ MC14512

C	B	A	INHIBIT	DISABLE	Z
0	0	0	0	0	X_0
0	0	1	0	0	X_1
0	1	0	0	0	X_2
0	1	1	0	0	X_3
1	0	0	0	0	X_4
1	0	1	0	0	X_5
1	1	0	0	0	X_6
1	1	1	0	0	X_7
X	X	X	1	0	0
X	X	X	X	1	อิมพีแดนซ์สูง

2.7.2 เอาท์พุทด้วยวงจรแลทช์ MC14599

วงจรแลทช์ MC14599 นี้ แตกต่างจากแลทช์ตัวอื่น ๆ ที่เคยพบเห็นมา เพราะ MC14599 มีแอดเดรสในการเลือกแลทช์ได้ตามความต้องการกล่าวคือ มีข้อมูลอินพุทที่จะนำเข้าไปแลทช์เพียงบิทเดียว แต่จะไปแลทช์ที่เอาท์พุทใด ขึ้นอยู่กับตัวกำหนดแอดเดรส ลองดูการจัดขาไอซี MC14599 นี้ดังแสดงในรูป

เอกสารนี้มีไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง การจัดขาและแผนผังภายในของ MC14599

ไอซีตัวนี้ เหมือนหน่วยความจำ กล่าวคือ ขา 3 ซึ่งเป็นขาข้อมูล จะเป็นได้ในสองทิศทาง ขึ้นกับขา Write/$\overline{\text{Read}}$ ถ้าขา 10 นี้ เป็น "0" คือ Read มันจะอ่านข้อมูลจาก Q_0 - Q_7 ออกมาทางขาข้อมูลนี้ ข้อมูลที่ออกมาขึ้นกับ A_0 - A_2 ในการเขียนข้อมูลจะต้องให้ขา Write/$\overline{\text{Read}}$ เป็น "1" และขา Write Disable เป็น "0" ด้วย ทั้ง 2 กรณีขา Chip Enable ต้องเป็น "1" ด้วย

จากข้อพิจารณาการทำงานของไอซี ดังกล่าวนี้ จะเห็นว่าไอซีเบอร์นี้เหมาะที่จะเป็นเอาท์พุทพอร์ทให้กับ ICU นี้มาก เพราะสามารถเลือกเขียนข้อมูลที่บิทใดก็ได้ และที่สำคัญคือ เขียนแล้วอ่านได้ด้วย จึงเสมือนการเขียนเข้าไปเก็บไว้ แบบฝากไว้ในหน่วยความจำ เพื่อเรียกข้อมูลกลับมาดูได้ ซึ่งเป็นจุดเด่นของไอซีตัวนี้ เมื่อใช้กับระบบนี้

2.7.3 I/O พอร์ท

อุปกรณ์ภายนอกที่ต้องการส่งถ่ายข้อมูลกับ CPU จะต้องอาศัยพอร์ทเป็นตัวกลาง โดยพอร์ทจะทำหน้าที่เป็นช่องทางส่งผ่านข้อมูล สำหรับอุปกรณ์ส่งผ่านแต่ละตัว ในระบบคอมพิวเตอร์หนึ่ง ๆ อาจมีหลาย ๆ พอร์ทเพื่อเป็นช่องทางส่งผ่านข้อมูลของอุปกรณ์หลาย ๆ ชนิด

I/O พอร์ทจะประกอบด้วยวงจรที่สามารถเลือกสับเปลี่ยนนำข้อมูลเข้ามาออกCPU ได้ พอร์ทเหล่านี้จะถูกเลือกเมื่อเอ็กซีคิวท์ คำสั่งเกี่ยวกับอินพุทและเอาท์พุท I/O พอร์ทมีทั้งสำหรับเป็นอินพุทพอร์ทและเอาท์พุทพอร์ท อินพุทพอร์ทเป็นช่องทางสำหรับนำข้อมูลเข้า CPU เอาท์พุทพอร์ทเป็นช่องทางสำหรับนำข้อมูลออกจาก CPU ส่งให้อุปกรณ์ภายนอก ปกติ I/O พอร์ทมีแบบส่งถ่ายข้อมูล 2 แบบ คือ การส่งถ่ายข้อมูลแบบขนาน และการส่งถ่ายข้อมูลแบบต่อเนื่อง ด้านหนึ่งของพอร์ทจะต่อกับบัสข้อมูลของ CPU ส่วนอีกด้านสำหรับต่อเข้ากับอุปกรณ์ภายนอก ถ้าเป็นอินพุทพอร์ทด้านที่ต่อกับบัสข้อมูลของ CPU เป็นลอจิกเกทพวกไตรสเตทบัฟ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เฟอร์ (Tristate Buffer) คือ เมื่อพอร์ทไม่ถูกเลือกให้ทำงานจะมีลักษณะเหมือนกับปลดสายออกจากบัสข้อมูลของระบบ มีคุณสมบัติเป็นอิมพีแดนส์สูง (high-impedance) ไม่มีผลของสัญญาณลอจิกให้กับระบบ

2.7.3.1 I/O พอร์ทแบบขนาน

I/Oพอร์ทสำหรับการส่งถ่ายข้อมูลแบบขนาน เป็นไอซีจำพวก LSI ภายในเป็นวงจรลอจิกประกอบกันขึ้นอยู่ในตัวเดียวกัน มีส่วนประกอบที่สำคัญ ๆ ดังนี้ คือ

1. บัสบัฟเฟอร์ (bus buffer) คือ ช่องทางส่งผ่านข้อมูล ประกอบด้วยลอจิกประเภทไตรสเตทบัฟเฟอร์ โดยทั่วไปจะมีสายอินพุทและเอาท์พุท 8 สาย ใช้ในการส่งผ่านข้อมูลแบบขนาน 8 บิท พอร์ทนี้อาจใช้สำหรับทำหน้าที่เป็นอินพุทหรือเอาท์พุทโดยเฉพาะได้ หรืออาจต่อไว้สำหรับเป็นทั้งอินพุทและเอาท์พุท โดยด้านที่ต่อกับบัสข้อมูลของ CPU จะต้องมีคุณสมบัติเป็นบัสสองทิศทาง (bi-directional bus)

2. การกำหนดแอดเดรสของ I/O พอร์ทแต่ละตัวจะมีสาย สำหรับกำหนดหมายเลขของพอร์ท ซึ่งใช้ในการเรียกพอร์ท ปกติจะมีสายกำหนด 1 สายหรือมากกว่า อุปกรณ์ที่ต่อเข้ากับพอร์ทแต่ละพอร์ท เมื่อต้องการให้ทำงานก็ใช้หมายเลขที่กำหนดไว้ที่พอร์ทเป็นรหัสสำหรับเลือก

3. บัสควบคุมและไทมิ่งต์ของสัญญาณ I/O พอร์ทมีสายสำหรับควบคุมการทำงานอีกหลายสาย สิ่งจะควบคุมการทำงานเช่น การส่งข้อมูลผ่านเข้าออก และการเคลียร์ การแลทช์ข้อมูล นอกจากนี้ ยังต้องคำนึงถึงไทมิ่งต์ของสัญญาณต่าง ๆ ในการทำงานด้วย คือ ช่วงสัญญาณควบคุมอะไรมาก่อนมาหลัง ข้อมูลที่ส่งรับทำในช่วงใดของสัญญาณ สิ่งต่าง ๆ เหล่านี้จำเป็นต้องคำนึงถึงในการออกแบบ

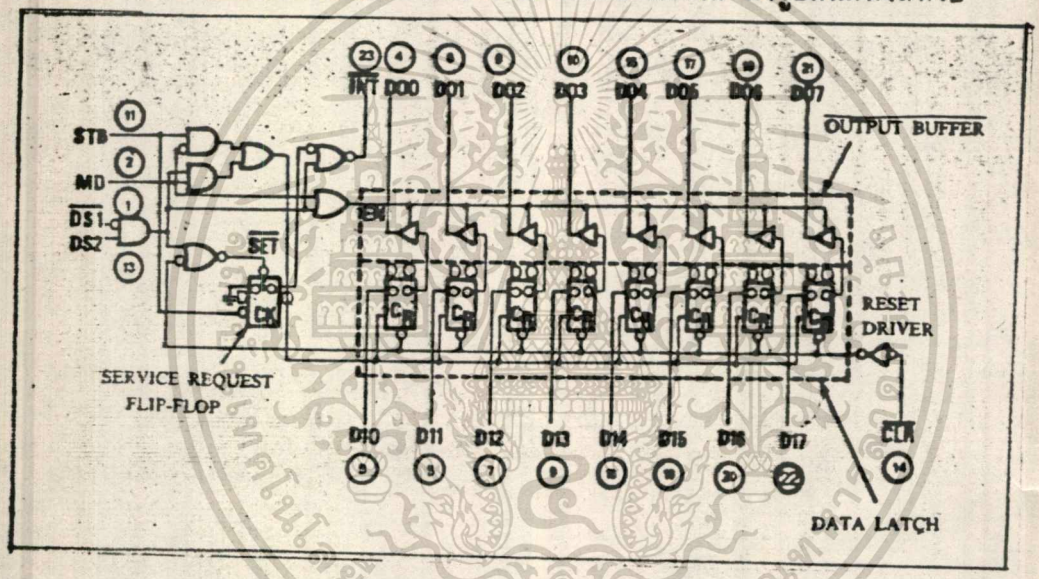
ตัวอย่างในรูปที่แสดง เป็น I/O พอร์ทแบบขนาน (Parallel I/O port) ชนิด 8 บิท อินพุท เอาท์พุทพอร์ท เบอร์ 8212 ซึ่งเป็น I/O พอร์ทสำหรับการรับส่งข้อมูลแบบขนานอย่างธรรมดา ช่องทางส่งผ่านข้อมูลแบ่งเป็น 2 ส่วน คือ ด้านรับข้อมูลเข้า DI_0-DI_7 ด้านรับข้อมูลเข้าประกอบด้วย 8 บิทแลทช์ (latch) เป็นพวก D ฟลิบฟลอบ ด้านส่งข้อมูลออกประกอบด้วย 8 บิทเอาท์พุทบัฟเฟอร์ เป็นพวกไตรสเตทบัฟเฟอร์ นอกจากช่องทางส่งผ่านข้อมูลแล้ว ยังมีสายควบคุมซึ่งใช้ควบคุมการเคลื่อนย้ายข้อมูลต่าง ๆ อีก เช่น mode select (MD) strobe (STB) และ device select logic ($\overline{DS1}$, DS2) เบอร์ 8212 นี้สามารถใช้เป็นอินพุทพอร์ทหรือเอาท์พุทพอร์ทก็ได้ ถ้าใช้เป็นอินพุทพอร์ทก็ต่อด้านเอาท์พุทบัฟเฟอร์ DO_0 - DO_7 เข้ากับบัสข้อมูลของ CPU ส่วนด้าน 8 บิทแลทช์ DI_0 - DI_7 ต่อ

กับบัสข้อมูลของอุปกรณ์ภายนอก ถ้าจะใช้เป็นเอาท์พุทพอร์ทก็จะต้องต่อในทางตรงกันข้าม

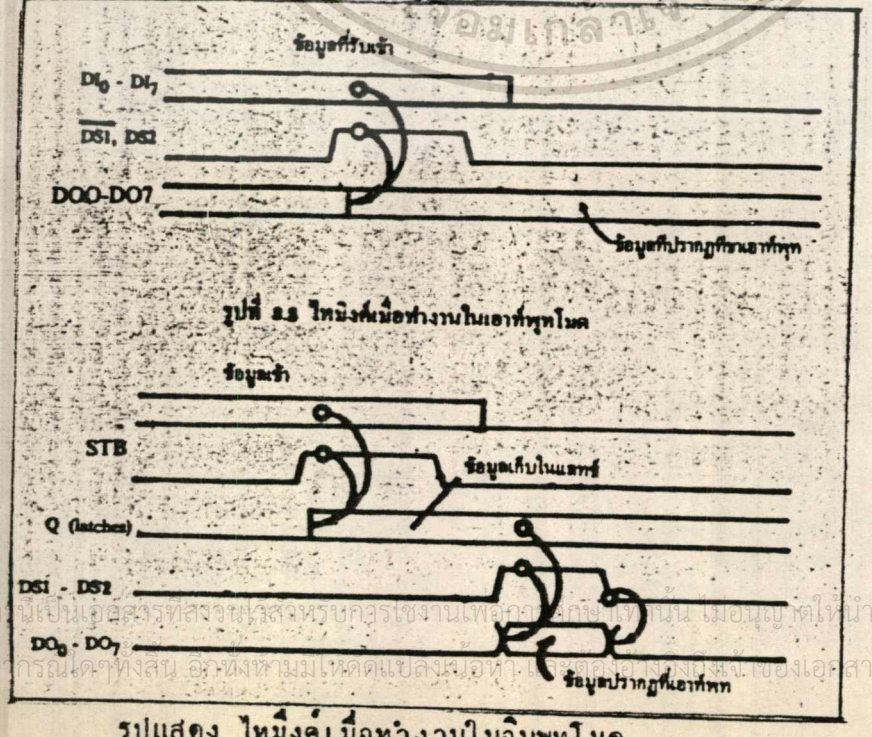
MD เป็นขั้วสัญญาณสำหรับการเลือกโหมด (Mode) การทำงานว่าเป็นอินพุทโมด หรือเอาท์พุทโมด ถ้าขา MD มีสัญญาณอินพุทเข้ามาเป็นลอจิก "0" วงจรจะทำงานในอินพุทโมด ถ้าเป็นลอจิก "1" ก็จะทำงานในเอาท์พุทโมด

ในกรณีของเอาท์พุทโมด ขา STB จะไม่มีผลต่อการส่งถ่ายข้อมูล การทำงานจะเป็นไปตามไทมิ่งค์ ดังแสดงในรูป สัญญาณ $\overline{DS1}$ และ DS2 จะสโตรบ (Strobe) ข้อมูลใน 8 บิทแลทช์ส่งออกทางขาเอาท์พุท DO_0 - DO_7

ในกรณีของอินพุทโมด สัญญาณที่ขา STB จะสโตรบข้อมูลที่ขา DI_0 - DI_7 เข้าเก็บใน 8 บิทแลทช์ การทำงานจะเป็นไปตามไทมิ่งค์ ดังรูปที่แสดงถัดไป



รูปแสดง โครงสร้างภายในของ 8212 I/O พอร์ท

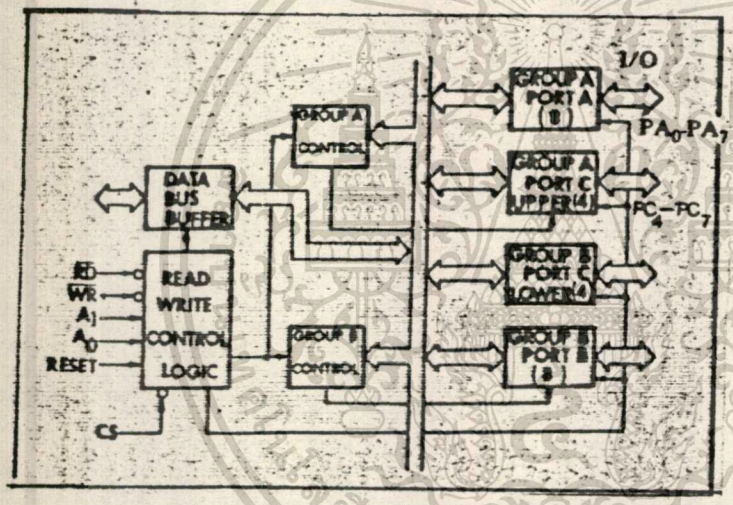


รูปแสดง ไทมิ่งค์เมื่อทำงานในอินพุทโมด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รายละเอียดของสัญญาณต่าง ๆ ในชิบ I/O พอร์ทตัวอื่น อาจแตกต่างกันไปบ้าง ขึ้นอยู่กับการใช้งาน เมื่อต้องการนำชิบตัวใดมาใช้จะต้องศึกษาคู่มือการใช้งานของชิบตัวนั้น เสมอ

พอร์ทรับส่งข้อมูลแบบขนานอีกตัวหนึ่งที่ใช้กันมากคือเบอร์ 8255 (24 Programmable I/O) เป็นพอร์ทรับส่งข้อมูลแบบขนานสามารถเลือกช่องทางเข้าออกจากอุปกรณ์ได้ถึง 3 ช่องทาง คือ พอร์ท A, B และ C บัสข้อมูลทั้งของพอร์ท A, B, C และด้านที่ต่อกับบัสข้อมูลของ CPU เป็นแบบบัสสองทิศทาง ดังนั้น จึงสามารถใช้ส่งถ่ายข้อมูลเข้าหรือออกก็ได้ การกำหนดทิศทางของบัสเหล่านี้สามารถทำได้โดยการเลือกสัญญาณที่ขา $\overline{RD}$, $\overline{WR}$, A_1 และ A_0 จากรูปที่แสดง เป็นโครงสร้างภายในของพอร์ทรับส่งข้อมูลแบบขนานเบอร์ 8255 แสดงให้เห็นถึงสายข้อมูลที่พอร์ทต่าง ๆ พร้อมทั้งสัญญาณควบคุมการทำงาน



รูปแสดง พอร์ทเบอร์ 8255

การโปรแกรม 8255 ให้ทำงานในโมดต่าง ๆ ดังแสดงในตาราง จากการให้สัญญาณที่ขั้ว $\overline{CS}$, A_1, A_0, $\overline{RD}$ และ $\overline{WR}$ ก็สามารถโปรแกรมให้พอร์ทต่าง ๆ ของ 8255 ทำงานเป็นอินพุทหรือเอาท์พุทได้

ตารางแสดง การเลือกวิธีการทำงานของ 8255

CS	A1	A0	RD	WR	OPERATION	
0	0	0	0	1	PORT A TO DATA BUS	MPU READ (A.B.C)
0	0	1	0	1	PORT B TO DATA BUS	
0	1	0	0	1	PORT C TO DATA BUS	
0	0	0	1	0	DATA BUS TO PORT A	MPU WRITE
0	0	1	1	0	DATA BUS TO PORT B	
0	1	0	1	0	DATA BUS TO PORT C	
0	1	1	1	0	DATA BUS TO CONTROL	
0	1	1	0	1	ILLEGAL	
1	X	X	X	X	DATA BUS TO 3-STATE	(DISABLE)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.7.3.2 I/O พอร์ทแบบอนุกรม (Serial I/O port)

I/O พอร์ทที่ใช้ในการส่งถ่ายข้อมูลแบบต่อเนื่องก็เป็นไอซีจำพวก LSI เช่นเดียวกัน มีจงจรต่าง ๆ ประกอบกันไว้ในไอซีตัวเดียวใช้สำหรับส่งถ่ายข้อมูลกับอุปกรณ์ I/O พวกเครื่องโทรพิมพ์ (teleype) คาสเซทและจานแม่เหล็ก (disk) ปัจจุบันได้มีผู้ผลิต LSI ที่ทำหน้าที่ทั้งรับและส่งข้อมูลอยู่ในตัวเดียวกัน มีชื่อเรียกกันว่า USART (Universal Synchrous Asynchronous Receiver/Transmiterและ UART (Universal Asynchrous Receiver/ Transmitter) ไอซีที่ใช้ในการรับส่งข้อมูลแบบต่อเนื่องมีส่วนประกอบที่สำคัญ ๆ ดังนี้

1. วงจรส่งข้อมูลออกแบบอนุกรมและรับข้อมูลเข้าแบบอนุกรม ประกอบด้วยบัฟเฟอร์ด้านรับและส่ง ชิฟท์รีจิสเตอร์ (Shift register) สำหรับและส่งข้อมูลแบบอนุกรม

2. การกำหนดแอดเดรสด้านรับและส่งจะมีชั้วกำหนดการทำงานของพอร์ท โดยรับสัญญาณรหัสจากบัสแอดเดรสหรือตัวถอดรหัสแอดเดรส

3. บัสควบคุมพอร์ทรับส่งข้อมูลแบบอนุกรมจะมีสายสำหรับควบคุมทั้งด้านรับและส่ง มีการควบคุมรูปแบบของสัญญาณที่ส่งออกเช่น การกำหนดจำนวนสตอปบิท (Stop bit) จำนวนบิทข้อมูล จำนวนบิทพาริตี้ (Parity bit) นอกจากนี้ยังควบคุมการตรวจสอบพาริตี้ของสัญญาณที่รับเข้าว่ามีการผิดพลาดหรือไม่ด้วย

สำหรับรายละเอียดอื่น ๆ ของพอร์ทรับส่งข้อมูลแบบอนุกรมจะได้กล่าวต่อไป

2.7.4 สัญญาณควบคุมการทำงานของพอร์ทและการกำหนดแอดเดรส

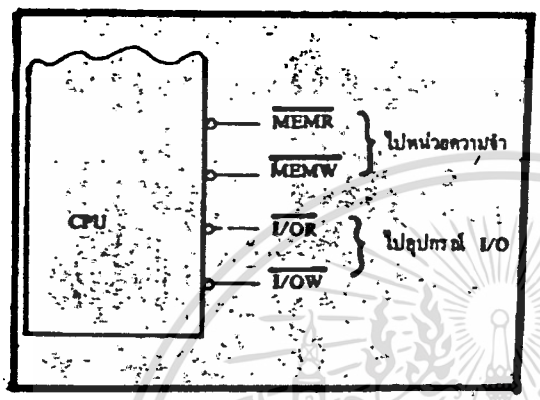
วิธีรับส่งข้อมูลโดยใช้คำสั่งในไมโครโปรเซสเซอร์ ปกติจะมีคำสั่งเฉพาะสำหรับการรับและส่งข้อมูล เมื่อ CPU ทำการเอ็กซ์ซีคิวท์คำสั่งอินพุท เอาท์พุท จะมีสัญญาณควบคุมออกจากระบบควบคุมโดยตรง สัญญาณนี้เรียกว่า สัญญาณ Isolate I/O นอกจากนี้ยังมีวิธีของการใช้คำสั่ง เพื่อติดต่อกับอุปกรณ์ I/O อีกวิธี คือใช้คำสั่งเกี่ยวกับการอ่านเขียนหน่วยความจำ ซึ่ง CPU จะให้สัญญาณการควบคุมการอ่านและเขียนข้อมูลที่หน่วยความจำ สัญญาณนี้สามารถ นำมาดัดแปลงใช้เป็นสัญญาณควบคุมอุปกรณ์ I/O ได้ เราเรียกสัญญาณนี้ว่า Memory mapped I/O

2.7.4.1 Isolated I/O

ไมโครโปรเซสเซอร์ 8080 มีคำสั่งเพื่ออินพุทและเอาท์พุท ข้อมูลโดยเฉพาะ คือคำสั่ง IN และ OUT ซึ่งเป็นคำสั่ง 2 ไบท์ ไบท์แรกเป็น Op Code ไบท์ที่สองเป็นแอดเดรสหรือหมายเลขของ I/O สำหรับบอกรหัสหรืออุปกรณ์ I/O ที่จะติดต่อกับ CPU ค่าใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารนี้ทุกครั้งที่มีการนำไปใช้

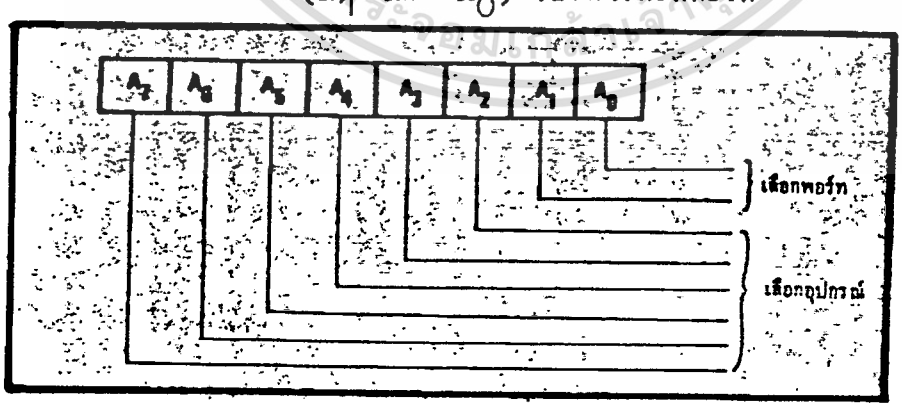
ไบท์ที่สองนี้จะถูกส่งออกที่แอดเดรสบัส ($A_{15} - A_0$) โดยค่าที่ $A_{15}-A_8$ และ A_7-A_0 มีค่าเหมือนกัน ส่วนสัญญาณควบคุมจะออกมาจาก CPU โดยตรง ดังแสดงในรูป เมื่อ CPU ทำคำสั่ง IN จะให้พัลส์ออกมาที่ขา $\overline{\text{I/O R}}$ และถ้าทำคำสั่ง OUT จะให้สัญญาณพัลส์ออกมาที่ขา $\overline{\text{I/O W}}$ เมื่อใช้คำสั่ง IN หรือ OUT การส่งถ่ายข้อมูลจะต้องส่งผ่านแอคคิวมูเลเตอร์ทุกครั้ง



รูปแสดง Isolated I/O

การถอดรหัสแอดเดรสเพื่อเลือกพอร์ทหรืออุปกรณ์ I/O สามารถทำโดยใช้แอดเดรสบิทต่าง ๆ ควบคุมแต่ละพอร์ทโดยตรง หรืออาจใช้ตัวถอดรหัสแอดเดรส เพื่อให้ได้สัญญาณรหัสในการเลือกมาตามต้องการ

จากรูปที่แสดง เป็นการใช้แอดเดรสบิทควบคุมการเลือกอุปกรณ์โดยตรง เมื่อใช้วิธีนี้ I/O พอร์ทเบอร์ 8255 ใช้ 6 บิท แอดเดรส ($A_7 - A_2$) เป็นตัวเลือกอุปกรณ์ และใช้ 2 บิทแอดเดรส (A_1 and A_0) เป็นตัวเลือกพอร์ท



รูปแสดง การถอดรหัสแอดเดรสสำหรับ 8255

2.7.4.2 Memory mapped I/O

การส่งถ่ายข้อมูลระหว่างอุปกรณ์ I/O กับ CPU ด้วยวิธี Memory mapped I/O สามารถส่งถ่ายผ่านรีจิสเตอร์ต่าง ๆ ได้โดยตรง ไม่จำเป็นต้องส่งผ่านแอคคิวมูเลเตอร์ก่อนเหมือนวิธี Isolated I/O ทำให้สามารถลดเวลาในการส่งถ่ายข้อมูลลงได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

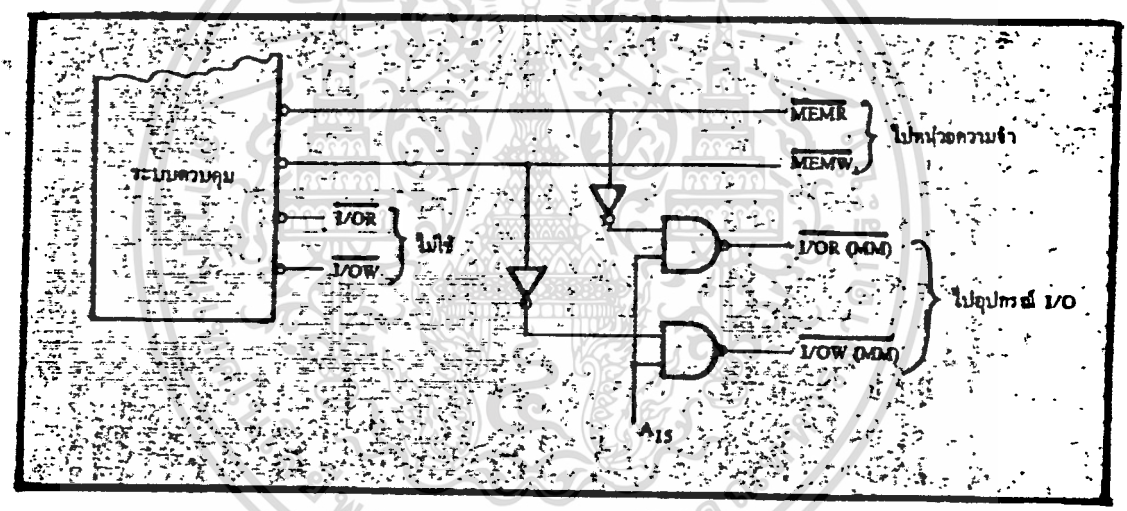
การทำงานจะทำในขณะเอ็กซีคิวท์คำสั่ง เช่นเดียวกับคำสั่ง IN หรือ OUT โดยใช้คำสั่งเกี่ยวกับการอ่านหรือเขียนข้อมูลในหน่วยความจำ

ในการใช้ 16 บิท แอดเดรสอ้างถึงตำแหน่งในหน่วยความจำสามารถทำได้ถึง 64 K แต่ถ้าใช้ในหน่วยความจำเพียง 32 K แอดเดรสบิทสูงสุด (A_{15}) จะไม่ถูกใช้ เราสามารถนำมาเป็น I/O แฟลกได้โดยกำหนดดังนี้

ถ้า A_{15} เป็น "0" สำหรับการทำงานของหน่วยความจำ

ถ้า A_{15} เป็น "1" สำหรับการทำงานของ I/O

ตามรูปที่แสดง สัญญาณควบคุมถูกสร้างขึ้นใหม่โดยใช้สัญญาณ $\overline{MEMR}$ และ $\overline{MEMW}$ ร่วมกับแอดเดรสบิท 15 เป็นสัญญาณควบคุมอุปกรณ์ I/O ใหม่ ซึ่งจะใช้แทนสัญญาณ Isolated I/O



รูปแสดง Memory mapped I/O

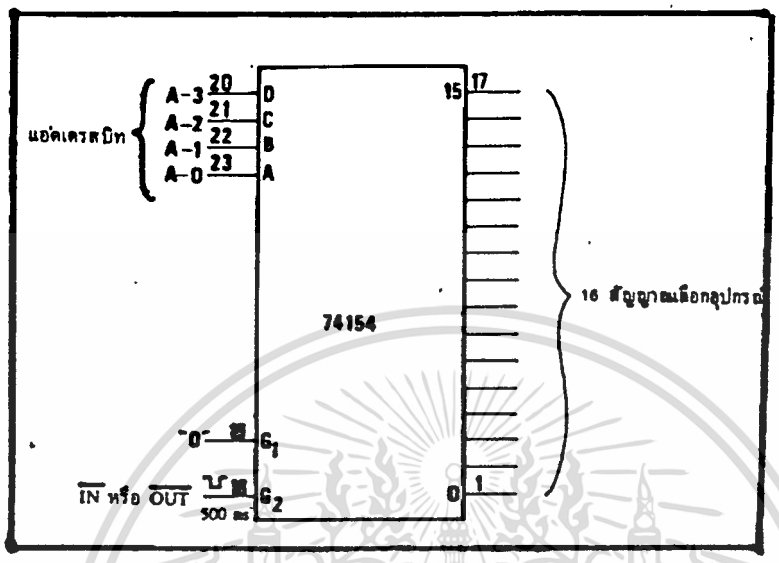
การถอดรหัสแอดเดรสเพื่อเลือกพอร์ทหรืออุปกรณ์ I/O ทำได้เช่นเดียวกับแบบ Isolate I/O ผิดกันแต่เพียงค่าของแอดเดรสที่ใช้เลือก ซึ่งในแต่ละคำสั่งที่ทำงานเกี่ยวกับหน่วยความจำ จะมีถึง 2 ไบท์จะเป็นแอดเดรส ดังนั้น รหัสที่จะถอดจะใช้ค่าตั้งแต่แอดเดรสบิท 0 ถึงบิท 15

2.7.4.3 <u>การสร้างสัญญาณเลือกอุปกรณ์โดยใช้ตัวถอดรหัส</u>

การสร้างสัญญาณเลือกอุปกรณ์โดยใช้ตัวถอดรหัส decoder เพื่อถอดรหัสแอดเดรส ซึ่งเป็นรหัสไบนารี่ให้เป็นสัญญาณออกมาหลาย ๆ ตัว เพื่อเลือกอุปกรณ์ตามต้องการ ในที่นี้จะยกตัวอย่างโดยใช้ไอซีเบอร์ 74154 ซึ่งเป็นตัวถอดรหัส 4 สายเป็น 16 สาย 4 to 16 line decoder เป็นวงจรง่าย ๆ ดังแสดงในรูป ซึ่งใช้ 4 บิท ต่ำสุด(A_3 - A_0)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้ง

ของแอดเดรสร่วมกับสัญญาณ $\overline{IN}$ หรือ $\overline{OUT}$ โดยสัญญาณ $\overline{IN}$ หรือ $\overline{OUT}$ จะเป็นสัญญาณสโตรปให้กับตัว 74154 ที่ขา G1 ส่วน G2 ต่อกับกราวน์หรือลอจิก "0" ไว้



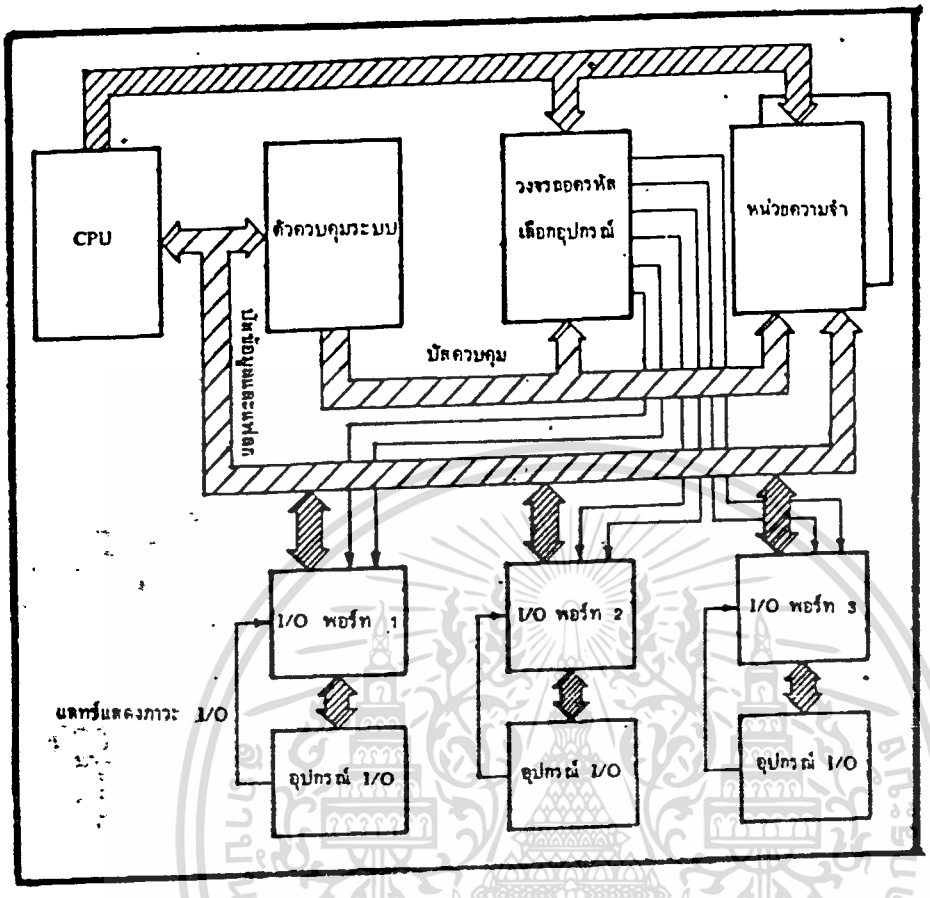
รูปแสดง การสร้างสัญญาณเพื่อเลือกอุปกรณ์ 16 ตัว

2.7.5 <u>วิธีโพลลิ่งหรือโปรแกรม I/O</u>

ในแบบของการโพลลิ่งหรือโปรแกรม I/O การส่งถ่ายข้อมูลของอุปกรณ์ทำโดยใช้โปรแกรม ไมโครโปรเซสเซอร์จะส่งหรือรับข้อมูลต่าง ๆ ทุกครั้ง อยู่ภายใต้การควบคุมของโปรแกรมที่กำลังทำงานอยู่นั้น การส่งถ่ายข้อมูลจำเป็นต้องทำร่วมกับการตรวจสอบสเตตัสซึ่งกันและกัน ระหว่าง CPU กับอุปกรณ์โดยใช้วิธีแฮนเชคกิ้ง (handshaking) วิธีการขั้นพื้นฐาน ได้แก่ การตรวจเช็คความต้องการส่งข้อมูล ความพร้อมที่จะรับข้อมูล โดยการใช้แฟลกต่าง ๆ สำหรับบอก แฟลกเป็นบิทซึ่งกำหนดไว้ เพื่อแสดงภาวะว่ามีอะไรเกิดขึ้น ต้องการทำอะไร ตัวอย่างเช่น แฟลกที่แสดง "device ready" ถ้าเป็นอุปกรณ์เอาท์พุทก็จะบอกว่าบัฟเฟอร์ว่างพร้อมกับข้อมูล เป็นต้น

ในวิธีของการโพลลิ่งเมื่อ CPU ต้องการติดต่อกับอุปกรณ์ภายนอก จะใช้การตรวจเช็คแฟลกที่พอร์ทต่าง ๆ โดยเริ่มด้วยการอ่านภาวะของแฟลกทุกครั้งก่อน แล้วจึงเข้าสู่โปรแกรมที่จะจัดการกับแต่ละอุปกรณ์อีกวิธีหนึ่ง จะเห็นว่าที่อุปกรณ์แต่ละตัวจะมีสัญญาณซึ่งจะแลทช์อยู่ เพื่อแสดงภาวะว่าอุปกรณ์นั้นพร้อม (ready) หรือไม่ที่จะทำการติดต่อกับ CPU การอ่านแฟลกหรือการส่งถ่ายข้อมูลจะทำได้ก็โดยการส่งสัญญาณสโตรปจากวงจรถอดรหัสเลือกอุปกรณ์ไปควบคุมการอ่านอีกทีหนึ่ง ดังรูปที่แสดง แสดงโครงสร้างระบบการโอนย้ายข้อมูลด้วยวิธีโพลลิ่งนี้

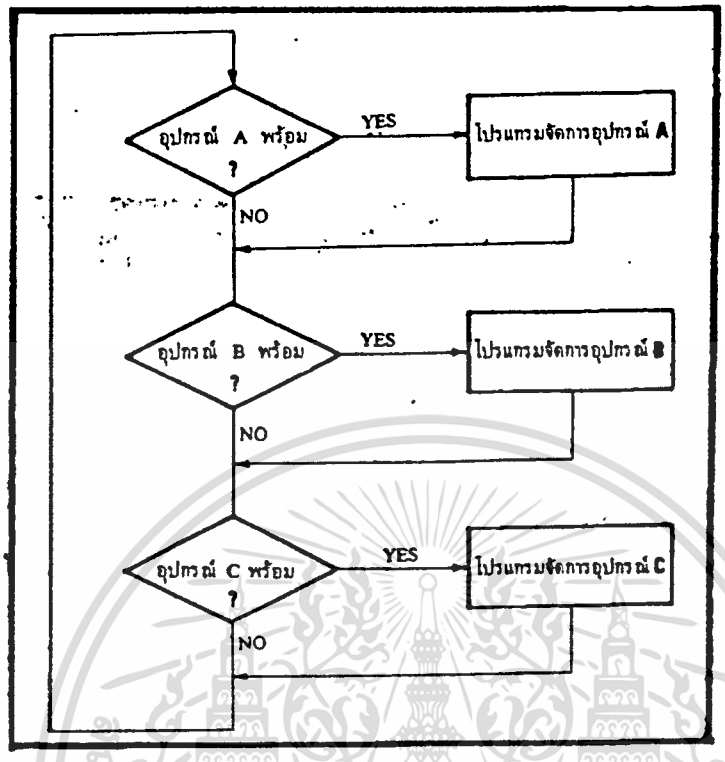
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ระบบวงจรที่ใช้วิธีโพลลิ่ง

ผังงานของการโพลลิ่ง ดังแสดงไว้ในรูป มีโปรแกรมที่จะจัดการกับอุปกรณ์ A, B และ C โดยเฉพาะ การทำงานจะเป็นลักษณะวนตรวจสอบภาวะของแฟลกที่อุปกรณ์ต่าง ๆ อยู่ตลอดเวลา เริ่มด้วยการตรวจสอบแฟลกของอุปกรณ์ A ถ้าพร้อมก็จะเข้าสู่โปรแกรมจัดการอุปกรณ์ A ถ้าไม่พร้อมจะผ่านลงมาตรวจสอบแฟลกของอุปกรณ์ B ซึ่งถ้าพร้อมก็จะเข้าไปทำในโปรแกรมจัดการอุปกรณ์ B และตรวจสอบต่อมาจนถึง C แล้วจึงวนกลับไปที่อุปกรณ์ A ใหม่ โปรแกรมจัดการอุปกรณ์ A, B และ C จะเป็นอะไรก็ขึ้นอยู่กับ อุปกรณ์และความต้องการของผู้ใช้ที่จะเขียนขึ้นโดยอาจเป็นการเรียก เพื่อนำข้อมูลเข้าคำนวณเก็บหรือแสดงผล เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง แผนภูมิแสดงวิธีการโพลลิ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

การคำนวณและการสร้าง

เมื่อกล่าวถึงระบบไมโครโปรเซสเซอร์ 1 บิท จะพบว่าเป็นระบบที่แตกต่างกับระบบไมโครโปรเซสเซอร์แบบหลายบิท ทั้งนี้เพราะการควบคุมการทำงานจะง่าย ระบบที่สร้างขึ้นจึงเหมาะกับการใช้งานโปรแกรมการควบคุมแทนอุปกรณ์บางอย่าง อย่างไรก็ดีวิธีการทดลองหรือพัฒนาระบบ เราก็มักอยากได้ชุดพัฒนาในลักษณะของซิงเกิ้ลบอร์ดไมโครคอมพิวเตอร์ แต่การสร้างไมโครคอมพิวเตอร์ในระบบ 8 บิท เราจะต้องมีโปรแกรมใน ROM เพื่อควบคุมการทำงาน แต่ในชุดพัฒนาระบบไมโครคอมพิวเตอร์ที่จะกล่าวถึงเป็นระบบที่สร้างขึ้นทางฮาร์ดแวร์ และไม่จำเป็นต้องมีโปรแกรมควบคุมการทำงานที่เรียกว่าโมนิเตอร์โปรแกรม

3.1 ระบบไมโครคอมพิวเตอร์สำหรับการพัฒนา

จากบทที่ 2 ได้แนะนำวงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์ MC14500 สำหรับการใช้งานแล้ว และได้อธิบายโครงสร้างทางฮาร์ดแวร์ของระบบซึ่งสามารถนำเอาไปใช้งานในวงจรควบคุมการโปรแกรมลอจิกได้ไม่ยากด้วยการเขียนโปรแกรม สำหรับบทที่ 3 นี้จึงนำเอาฮาร์ดแวร์ของ MC14500 มาสร้างเป็นชุดพัฒนา โดยยึดโครงร่างการออกแบบโดย นายเอดวาร์ด เอ็มนอล ซึ่งได้ใช้ไอซีปีมอสเป็นส่วนใหญ่ แต่จะดัดแปลงวงจรให้เข้ากับการหาไอซีได้ในท้องตลาดบ้านเราเพื่อว่าผู้สนใจจะได้สร้างระบบนี้ได้

จุดมุ่งหมายของระบบไมโครคอมพิวเตอร์สำหรับการพัฒนานี้ ก็เพื่อจะได้ใช้เป็นชุดฝึกโปรแกรมที่สลับซับซ้อนขึ้น หรือจะใช้เป็นชุดทดลองโปรแกรมการทำงานในระบบ PLC ตามหลักการแทนวงจรรีเลย์ที่นิยมเขียนเป็นขั้นบันไดนั้นได้ เมื่อทดลองได้ดีแล้วจึงนำมาสร้างเป็นระบบโดยให้โปรแกรมอยู่ ใน ROM ต่อไป

ชุดพัฒนาระบบใหม่ในที่นี้จึงใช้ฮาร์ดแวร์ควบคุมการทำงานทั้งสิ้น โดยชุดฮาร์ดแวร์แต่ละส่วนจะใช้หลักการอย่างง่าย ๆ ตรงไปตรงมา

3.2 โครงสร้างของชุดพัฒนาระบบ

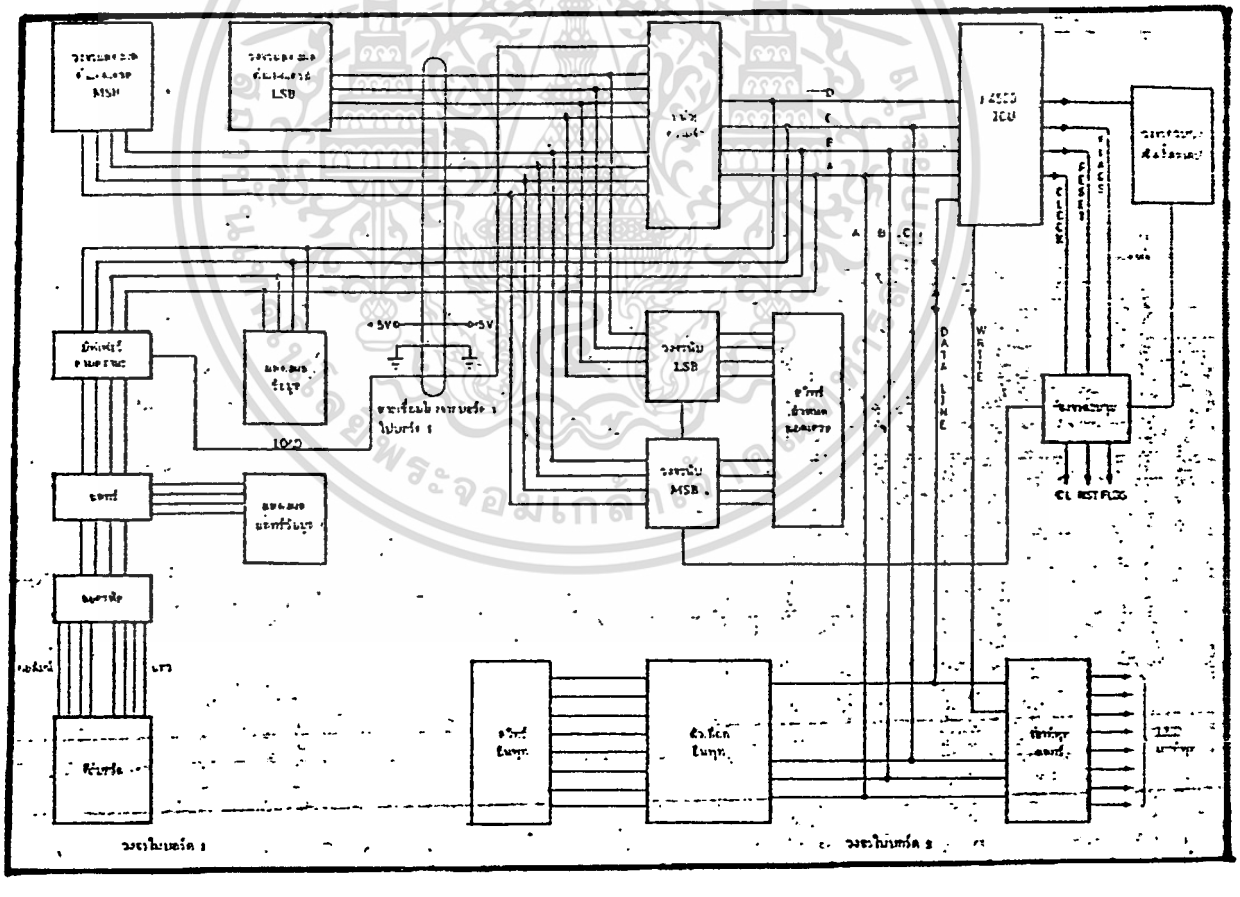
โครงสร้างของชุดพัฒนาระบบ ประกอบด้วยคีย์บอร์ด 16 คีย์ สำหรับป้อนโปรแกรมเข้าไปเก็บไว้ในหน่วยความจำ โดยมี LED แบบ 7 ส่วน แสดงผลตัวเลขฐานสิบหกเป็นตัวแสดงค่าแอดเดรสและข้อมูล สำหรับการกำหนดแอดเดรสจะใช้สวิทช์เล็ก ๆ โยกเป็น "0" และ "1" เพื่อป้อนเป็นค่าแอดเดรสให้ระบบ

ลักษณะของวงจรที่ใช้นี้เหมือนกับซิงเกิ้ลบอร์ดทั่วไป คือใช้ MC14500 เป็นไมโครโปรเซสเซอร์ มีหน่วยความจำที่จะให้ผู้ใช้ป้อนโปรแกรมใส่ได้ โดยมีสวิทช์ที่จะเซ็ทค่าให้

เอกสารนี้เป็นเอกสาร... ไม่ว...

กับโปรแกรมเคาวน์เตอร์แบบที่ละขั้น LED 7 ส่วนที่ใช้ถูกขับโดยไอซีที่สามารถถอดรหัสจากเลขไบนารี่เป็นเลขฐานสิบหกของภาคแสดงผลได้โดยตรง โดยส่วนของข้อมูล

วงจรที่เสนอนี้ แยกเป็น 2 แผง โดยใช้สายเชื่อมโยงที่ผ่านซ็อคเก็ตไอซีเพียงตัวเดียว (สายแผงขนาด 16 สาย ย่อยที่ติดกัน) และเพื่อลดขนาดของวงจรลง เราจะใช้คีย์บอร์ดเพียง 16 คีย์ และป้อนโปรแกรมทีละ 4 บิท ซึ่งในหนึ่งคำสั่งจะป้อน 2 ครั้ง โดยครั้งแรกป้อนออปโค้ด 4 บิท ครั้งที่ 2 ป้อนโอเปอแรนด์หรือแอดเดรสของ I/O อีก 4 บิท ดังนั้นจะใช้หน่วยความจำเพียง 4 บิท ซึ่งความจริงแล้ว หน่วยความจำมีความกว้าง 8 บิท แต่เราไม่ใช้ทั้ง 8 บิท เพราะจะทำให้ต้องสิ้นเปลืองวงจร ในการแยกข้อมูลจากคีย์บอร์ดไปเป็นแอดเดรสของ I/O หรือออปโค้ด ในที่นี้จึงใช้แบบง่ายนี้ และจำนวนโปรแกรมที่ป้อนนี้ก็เพียงพอสำหรับหน่วยความจำ โดยในที่นี้จะใช้แอดเดรสได้ 7 หรือ 8 เส้น ซึ่งก็หมายถึงได้ข้อมูลทั้งสิ้น 128 หรือ 256 ตำแหน่ง



รูปแสดง โครงสร้างของชุดพัฒนาระบบ

เพื่อให้วงจรนี้สมบูรณ์สำหรับการเรียนรู้และทดลองโปรแกรมการทำงาน จึงใช้วงจรควบคุม เพื่อให้ได้การทำงานครบ คือทำงานทีละคำสั่งหรือทำงานต่อเนื่องได้ ไอซีที่สำคัญในที่นี้มีดังนี้คือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. 4495 วงจรถอดรหัสไบนารี่ เป็นตัวเลขแสดงผลแบบตัวเลขฐานสิบหก
2. 4419 วงจรเข้ารหัสคีย์บอร์ด 16 คีย์
3. 4175 วงจรแลทช์
4. 4503 บัฟเฟอร์ 3 สถานะ
5. 4013 ฟลิปฟลอบ
6. 4011 NAND เกท
7. 4099 เอาท์พุทแลทช์
8. 4049 อินเวอร์เตอร์
9. 4500 ไมโครโปรเซสเซอร์บิทเดียว
10. 4512 อินพุทซีเลคเตอร์
11. 4070 ORเกท
12. 2112 หรือ 6116 หน่วยความจำ
13. 4516 วงจรนับ

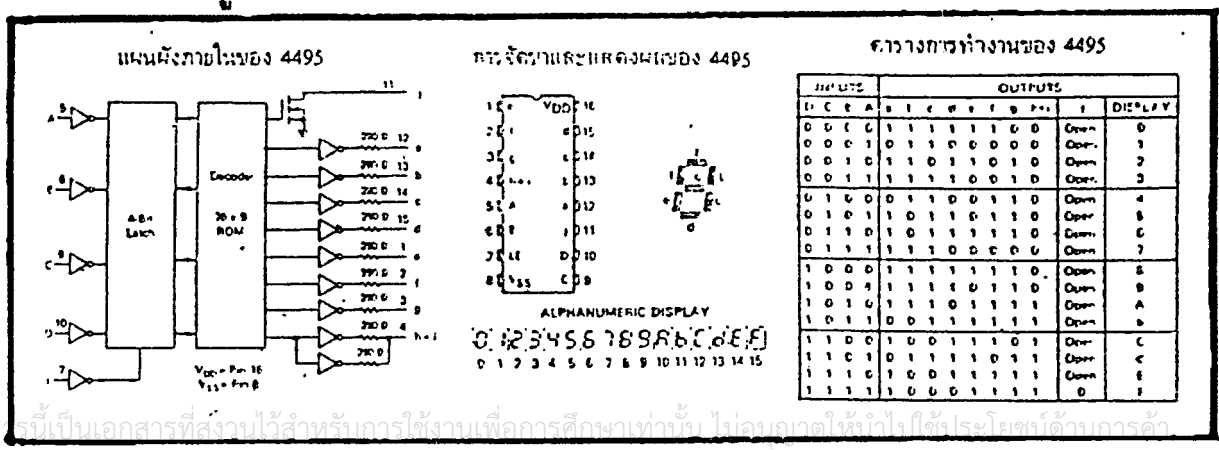
ไอซีตระกูล 4XXX เบอร์ข้างบนนี้ถ้าเป็นของยี่ห้อโมโตโรร่าจะมีตัว MC 1 นำหน้าเช่นเบอร์ MC14495, MC14500 เป็นต้น

การประกอบวงจรในรายละเอียด และการทำงานจะแยกกล่าวเป็นภาค ๆ ดังนี้

3.3 <u>วงจรภาคแสดงผล</u>

วงจรภาคแสดงผลนี้ เป็นวงจรที่ต้องการแสดงผลให้ได้ตัวเลขฐานสิบหก ดังนั้นจึงเลือกใช้ไอซีเบอร์ 4495 ซึ่งเป็นวงจรถอดรหัสจากไบนารี่ไปเป็นตัวเลขเจ็ดส่วนที่แสดงผลเป็นตัวเลข 0 ถึง F ได้

ไดอะแกรมรายละเอียด การจัดขา และตารางการทำงานของไอซีเบอร์ 4495 เป็นไปดังแสดงในรูป

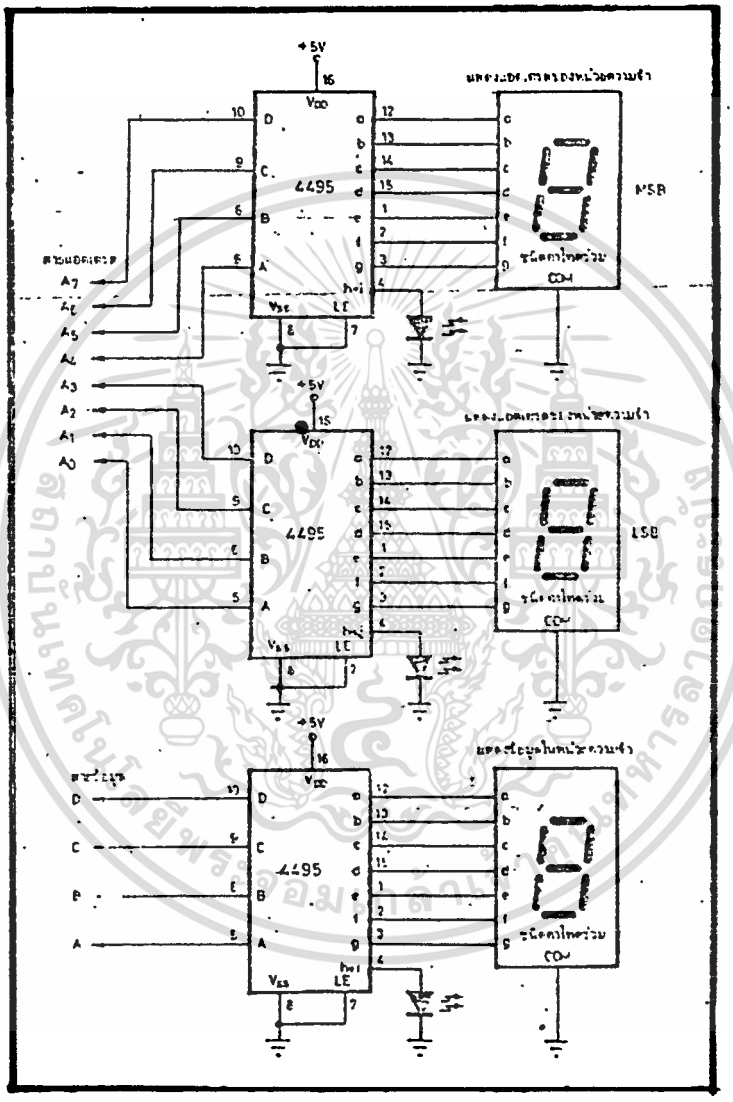


รูปแสดง ไดอะแกรมรายละเอียดของไอซีเบอร์ 4495

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น ทั้งนี้ต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปกติไอซี 4495 นี้ สามารถขับ LED ให้ส่วางได้โดยตรง โดยมีกระแสพอเพียงประมาณ 10 mA 10 มิลลิแอมแปร์ การแสดงผลจะใช้ลอจิก "1" เป็นลอจิกแอคทีฟ ดังนั้น LED ที่ใช้จึงใช้ชนิดคาโทดร่วม (common cathode)

วงจรที่ต่อแสดงการขับ LED ภาคแสดงตัวเลข 7 ส่วน เมื่อต่อเป็นวงจร สำหรับการแสดงข้อมูลแอดเดรส 8 บิท และข้อมูล 4 บิท เป็นดังแสดงในรูป



รูปแสดง วงจรของส่วนแสดงผลแอดเดรสและข้อมูลในหน่วยความจำ

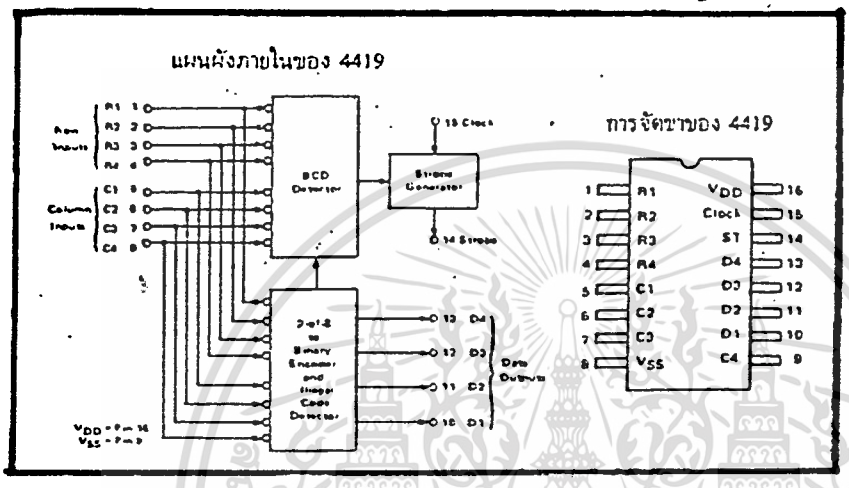
ส่วนการแสดงผลแอดเดรสนี้จะต่อโดยตรงจากเอาท์พุทของวงจรโปรแกรมเคาว์นเตอร์หรือแอดเดรสของหน่วยความจำนั่นเอง ค่าที่จะแสดงผลจะเป็นตัวเลขที่บอกว่าแอดเดรสขณะนั้นเป็นแอดเดรสอะไร และทำนองเดียวกับส่วนแสดงผลข้อมูลก็จะบอกว่าข้อมูลที่แอดเดรสนั้นมีอะไรอยู่

เมื่อท่าลองสร้างวงจรส่วนแสดง ผลเสร็จแล้วลองป้อนไฟเลี้ยงให้กับวงจรแล้วป้อนอินพุทด้วยตัวเลข "0" หรือ "1" ดูเอาท์พุทที่แสดงผลบน LED ว่าถูกต้องหรือไม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารเมื่อมีการนำไปใช้

3.4 วงจรรับคีย์บอร์ดและส่วนแสดงค่าที่กดคีย์

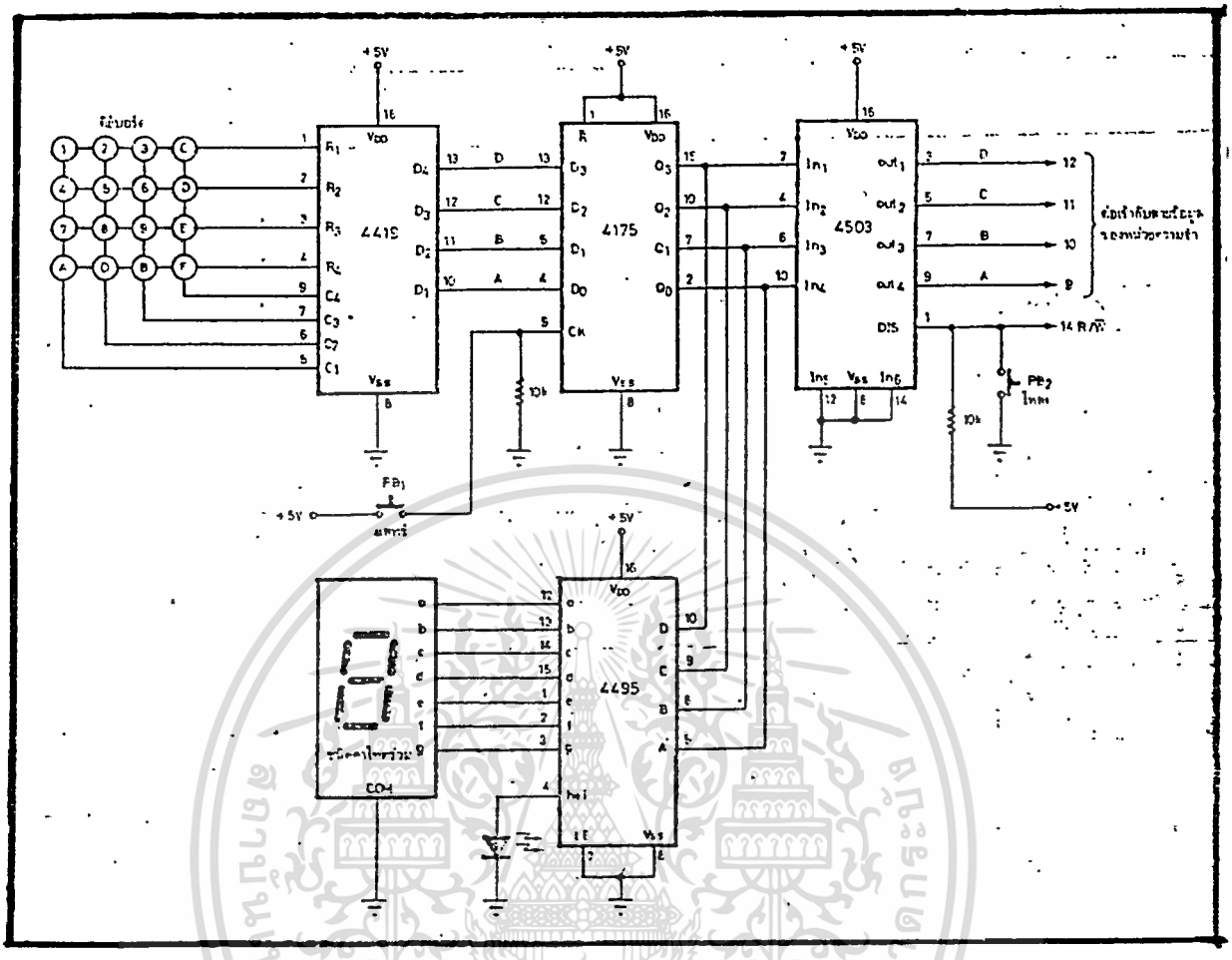
หลังจากที่สร้างวงจรภาคแสดงผลได้แล้ว คราวนี้ก็ลองมาดูวงจรรับคีย์บอร์ด ซึ่งคีย์บอร์ดที่จะใช้นี้เป็นคีย์บอร์ดที่จะรับนี้เป็นคีย์บอร์ดแบบแมทริกซ์ ขนาด 4 x 4 คือ 16 คีย์ ซึ่งจะแทนตัวเลข 0 –F ของตัวเลขไบนารี่ ไอซีที่จะใช้ในการเข้ารหัสคีย์บอร์ดเป็นตัวเลขไบนารี่คือ 4419 ซึ่งมีรายละเอียดของไอซี ดังแสดงในรูป



รูปแสดง แสดงรายละเอียดของไอซีเบอร์ 4419

ไดอะแกรมการต่อจะใช้ 4419 เข้ารหัสค่าคีย์บอร์ดออกมาเป็นตัวเลขไบนารี่จาก 0 ถึง F โดยปกติไอซี 4419 เป็นไอซีที่ไว้ใช้สำหรับโทรศัพท์กดปุ่ม แต่ในที่นี้จะใช้สำหรับแปลงค่าคีย์ให้เป็นตัวเลขไบนารี่ตามที่ต้องการ เอาท์พุทของ 4419 จะมีค่าตามที่คีย์กด และเราใช้ 4175 ซึ่งเป็นฟลิบฟลอบ – แลทช์ ทำหน้าที่แลทช์ข้อมูลให้กับการแสดง ผลว่ามีข้อมูลอะไรก่อนการกดแล้วโหลดให้หมวยความจำ ดังแสดงในรูป

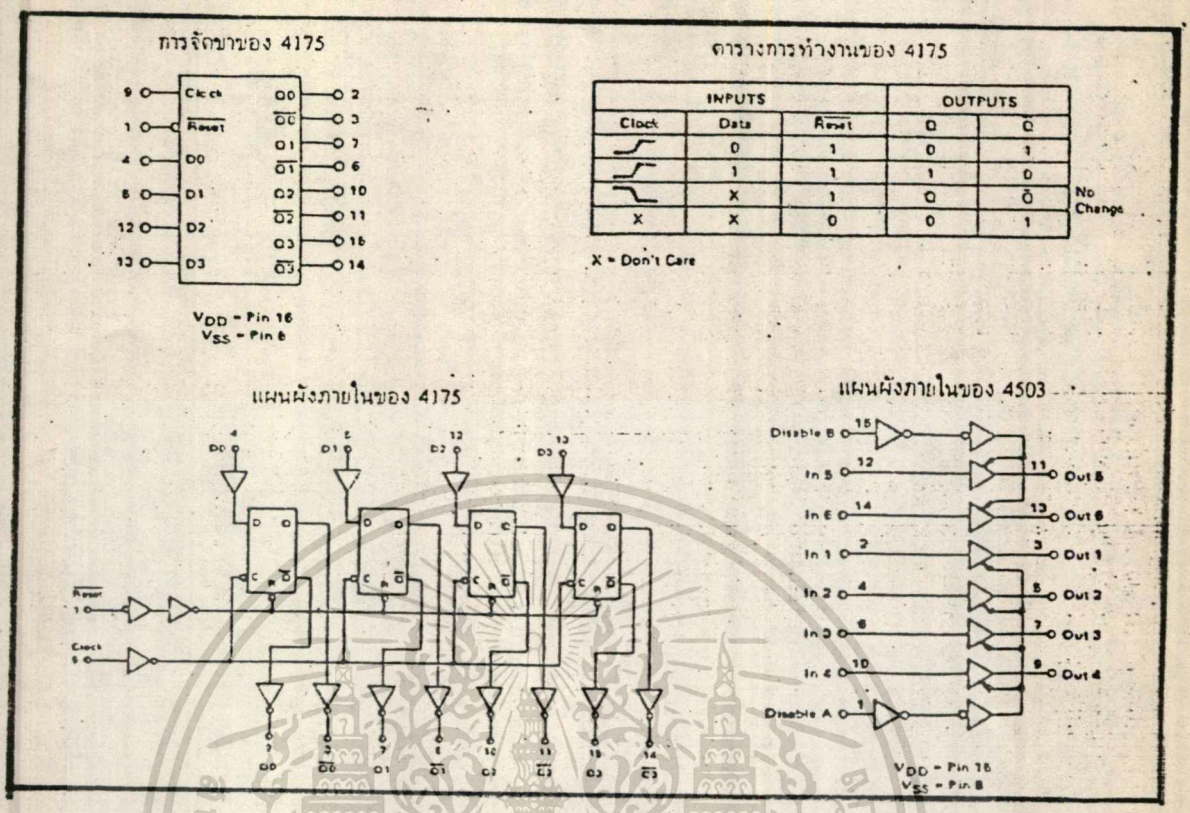
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง วงจรการรับอินพุทจากคีย์บอร์ด

ในที่นี้เราใช้สวิทช์กด PB_1 และ PB_2 ในวงจร 2 ตัว โดย PB_1 ทำหน้าที่แลทช์ข้อมูลที่กดมาแสดงผล ส่วน PB_2 ทำหน้าที่โหลดข้อมูลให้กับหน่วยความจำต่อไป ลองดูการจัดขาของไอซี 4175 และ 4503 ดังแสดงในรูป

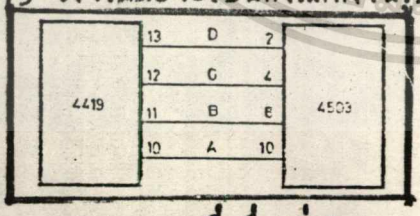
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง แสดงรายละเอียดของไอซีเบอร์ 4175 และ 4503

สังเกตในรูปที่แสดงว่าไอซี 4503 เป็นไอซีลอจิก 3 สถานะ ที่จะเปิดข้อมูลเข้าไปเขียนลงในหน่วยความจำตามแอดเดรสที่กำหนดโดยโปรแกรมเคาวน์เตอร์ ลักษณะการเขียนข้อมูลและเขียนลงทีละ 4 บิท ตามค่าที่แสดงผลจากเอาท์พุทของ 4175 นี้

โดยปกติเอาท์พุทที่ได้จาก 4419 จะต่อตรงเข้า 4503 โดยตรงก็ได้ โดยไม่จำเป็นต้องมี 4175 วงจรที่ต่อจะเป็นดังแสดงในรูป



รูปแสดง วงจรที่เชื่อมต่อโดยตรง

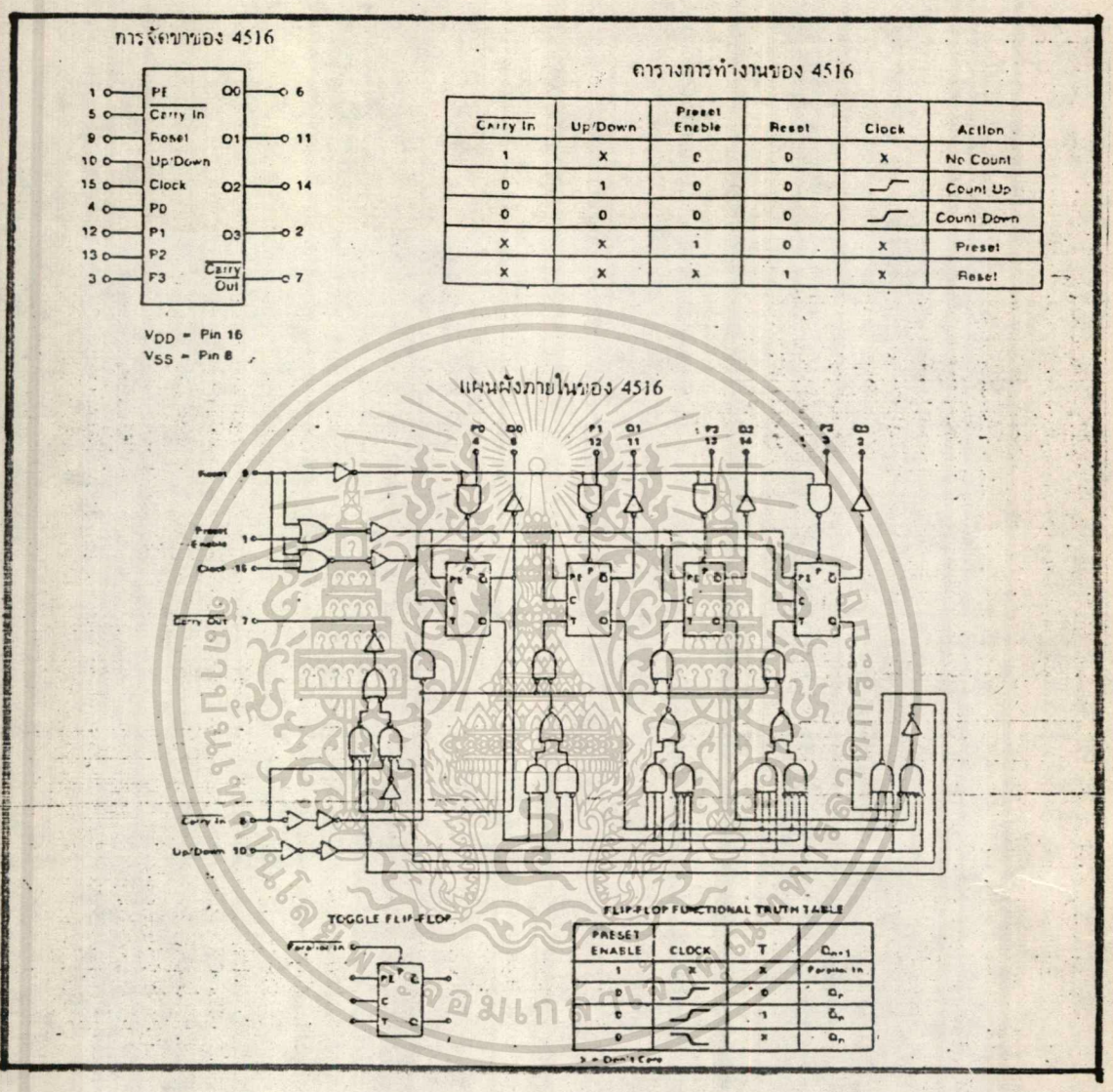
จะเห็นว่าเอาท์พุทที่ได้จากวงจรป้อนข้อมูลนี้จะป้อนเข้าได้ทีละเพียง 4 บิทเท่านั้น ถ้าหากจะทำให้เป็น 8 บิท จะต้องเพิ่มวงจรเลือก 4 บิทบน 4 บิทล่าง ซึ่งจะต้องเพิ่มฮาร์ดแวร์ขึ้นอีก ในที่นี้จะใช้เพียง 4 บิทเท่านั้น ลองทดลองสร้างวงจรส่วนนี้ดูแล้วดูค่าที่ได้จาก LED ตัวเลข เมื่อกดคีย์บอร์ดตามหมายเลขต่าง ๆ

3.5 โปรแกรมเคาวน์เตอร์กับหน่วยความจำ

ในการตั้งค่าแอดเดรสให้กับโปรแกรมเคาวน์เตอร์ หรือหน่วยความจำของชุดพัฒนา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

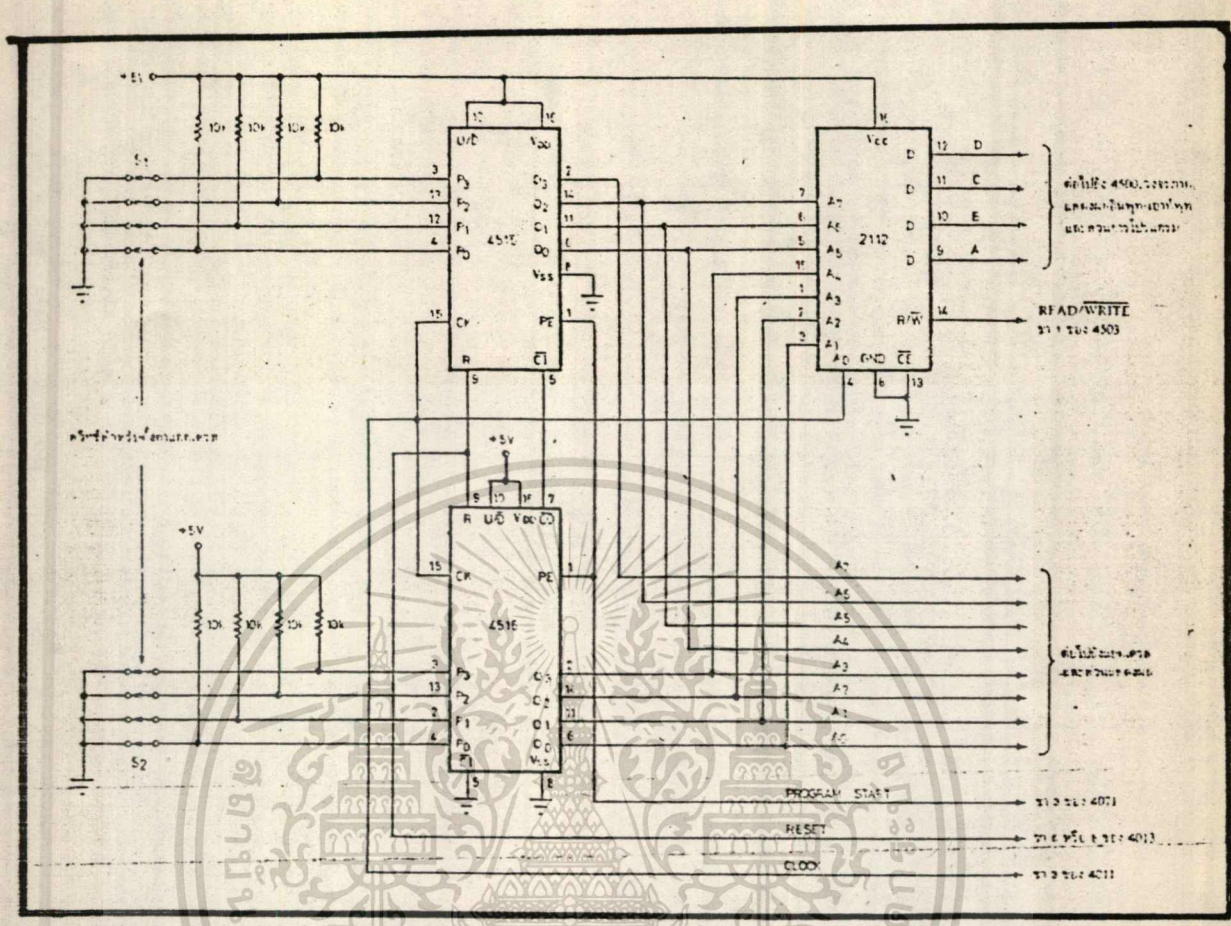
ระบบนี้ จะใช้สวิทช์ 8 ตัว ทำหน้าที่กำหนดแอดเดรสแต่ละบิทจาก A_0-A_7 โดยผ่านวงจรนับที่พรีเซทค่าได้คือไอซีเบอร์ 4516 ไดอะแกรมวงจรของ 4516 ดังแสดงในรูป



รูปแสดง แสดงรายละเอียดของไอซีเบอร์ 4516

ในการโหลดข้อมูลจากสวิทช์ จะผ่านการโหลดโดยควบคุมที่ขา 1 ของ 4516 ซึ่งถ้ามีค่าเป็น "1" มันจะโหลดข้อมูลเข้าไปเก็บไว้ในภายในวงจรส่วนนี้ดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



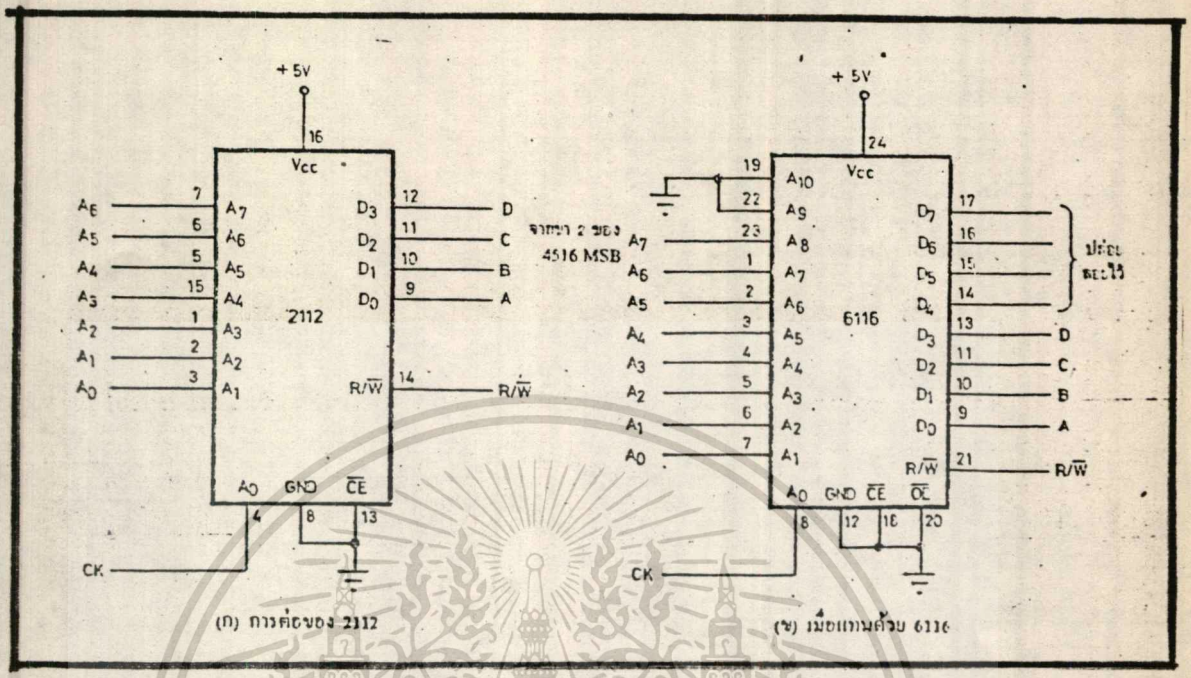
รูปแสดง วงจรเชื่อมโยงหน่วยความจำกับโปรแกรมเคาวน์เตอร์

สังเกตการเชื่อมโยงกับหน่วยความจำซึ่งทำให้สามารถโหลดโปรแกรมเข้าเก็บในหน่วยความจำได้ และทำการเคลื่อนโปรแกรมเคาวน์เตอร์ มาเริ่มที่ 0 ได้ หรือส่งสัญญาณาฬิกามาให้นับได้

สังเกตที่ขา A_0 ของหน่วยความจำ เบอร์ 2112 (ขา 4) จะต่อเข้ากับขาสัญญาณนาฬิกาจาก 4500 ดังนั้นคำสั่ง 1 คำสั่ง ในหน่วยความจำจึงต้องใช้แอดเดรส 2 ตำแหน่ง โดยแอดเดรสที่เป็นเลขคู่ (A_0 เป็น "0") จะเก็บคำสั่งที่เป็นแอดเดรสเลขคี่ (A_0 เป็น "1") จะเก็บตำแหน่งอินพุท-เอาท์พุทพอร์ท เนื่องจากหน่วยความจำนี้มีขนาดเพียง 4 บิท จึงต้องใช้ 2 ตำแหน่ง จึงจะได้คำสั่งที่สมบูรณ์ขนาด 8 บิท

สำหรับหน่วยความจำที่ใช้คือเบอร์ 2112 ซึ่งเป็น RAM ขนาด 128 คูณ 4 บิท แต่ถ้าเปลี่ยนมาเป็น 6116 ซึ่งเป็นขนาด 2K x8 บิท เราจะต่อวงจรใหม่ โดยแทนเบอร์ 2112 ได้ดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง การต่อวงจรของ RAM เบอร์ 6116 เพื่อใช้แทนเบอร์ 2112

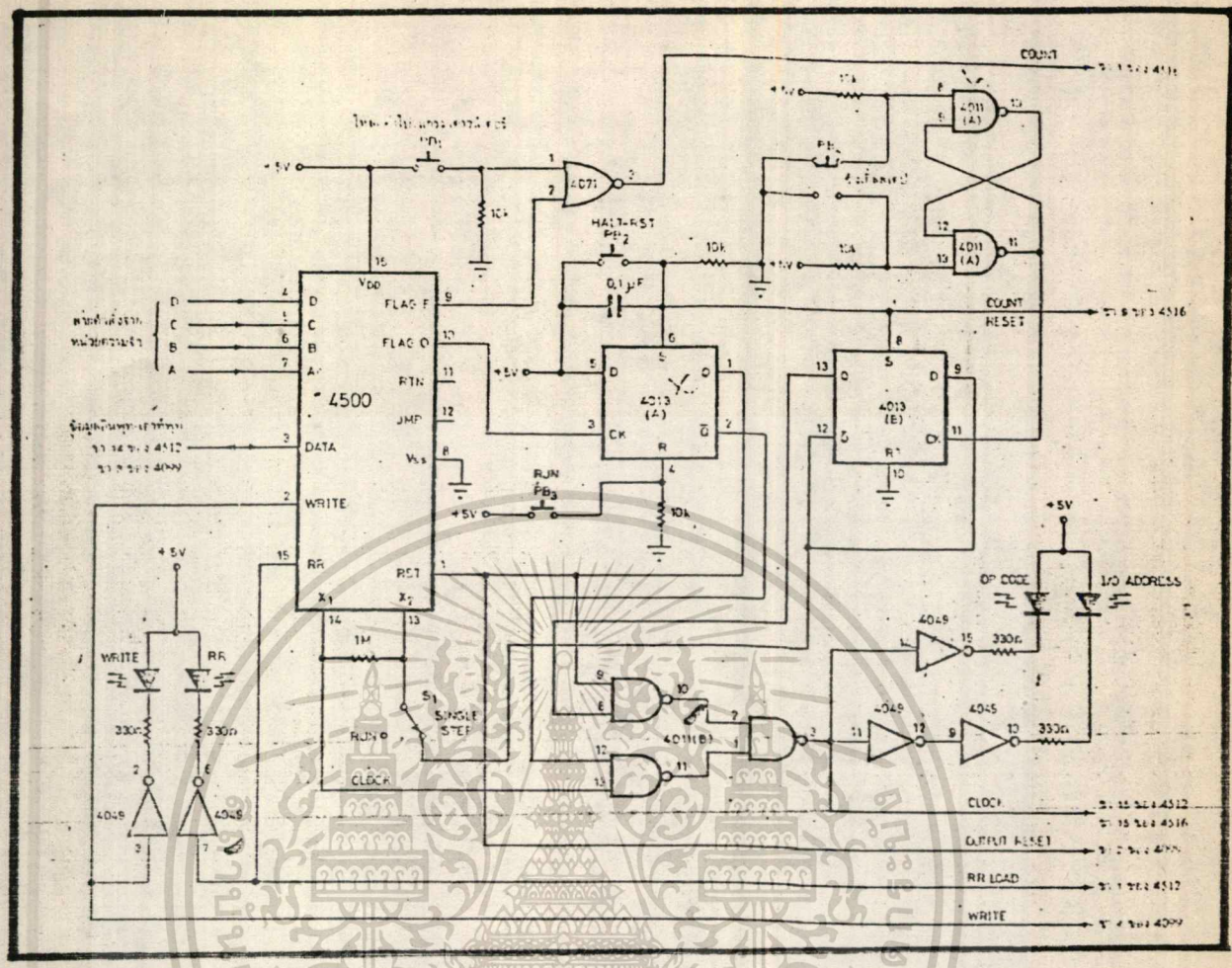
ในการใช้เบอร์ 2112 นี้ จะได้ข้อมูลเพียง 128 ค่า ดังนั้น A_7 จึงไม่สามารถนำมาใช้ได้ แต่หากใช้เบอร์ 6116 แทน จะใช้ A_7 ต่อเข้า เบอร์ 6116 ได้ ทำให้ได้หน่วยความจำเพิ่มขึ้นมาเป็น 256 ตำแหน่ง

3.6 วงจรไมโครโปรเซสเซอร์

วงจรไมโครโปรเซสเซอร์ที่ใช้นี้ เป็นไมโครโปรเซสเซอร์ที่ต้องอาศัยวงจรควบคุมจากภายนอกอยู่มาก เช่น ต้องใช้วงจรควบคุมโปรแกรมที่เรียกว่าโปรแกรมเคาวน์เตอร์

วงจรที่เสนอนี้เป็นวงจรที่จะแสดง สถานะการทำงานของไมโครโปรเซสเซอร์ได้ครบเหมือนชิงเกิ้ลบอร์ดทั่วไป คือทำงานทีละขั้นก็ได้ วงจรไมโครโปรเซสเซอร์ ดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ส่วนของไมโครโปรเซสเซอร์และวงจรประกอบ

ปกติเมื่ออยู่ในโหมดป้อนโปรแกรม เราจะรีเซทตัวซีพียูไว้ก่อน ดังนั้นซีพียูจะไม่สร้างสัญญาณนาฬิกา แล้วเราใช้สวิทช์ PB_1 ซึ่งเป็นสวิทช์แกนเดียวกับโหลดค่าคีย์ ส่งสัญญาณให้กับโปรแกรมเคาวน์เตอร์ เพื่อโหลดค่าแอดเดรสเข้ามาตามที่ตั้งไว้ด้วยสวิทช์ดังแสดงในรูปวงจรเชื่อมโยงหน่วยความจำกับโปรแกรมเคาวน์เตอร์ ในส่วนของวงจรนี้มี LED 4 ดวง แสดงสถานะการทำงานต่าง ๆ ไว้อย่างพร้อมมูล เช่น สถานะที่เป็นข้อมูลที่เป็นออปโค้ด หรือข้อมูลที่เป็นแอดเดรสสถานะของ RR หรือสถานะของการเขียนข้อมูลให้กับเอาท์พุทพอร์ท การทำงานแบบซิงเกิ้ลสเตปหรือทีละคำสั่งนั้น เราจะใช้สวิทช์ PB_4 กดสร้างพัลส์ที่ละพัลส์แทนพัลส์ที่ออกจากซีพียู โดยโยกสวิทช์ S_1 ไปที่ตำแหน่งซิงเกิ้ลสเตป

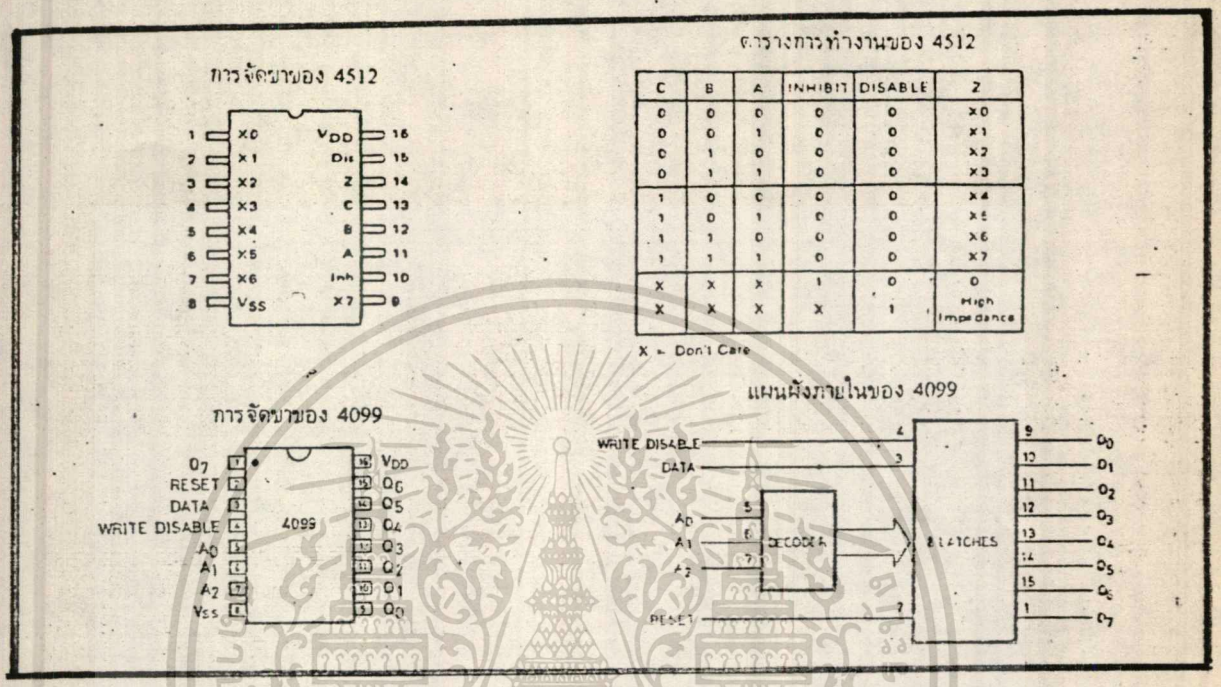
สวิทช์สำคัญอีก 2 ตัว คือ PB_3 เป็นสวิทช์ที่บอกให้ซีพียูเริ่ม RUN โปรแกรมจากแอดเดรสที่ตั้งไว้ และ PB_4 เป็นสวิทช์กดสำหรับให้ทำงานทีละคำสั่ง

3.7 วงจรอินพุท-เอาท์พุทพอร์ท

เนื่องจากข้อมูลที่รับเข้าและส่งออก จากซีพียูจะกระทำได้ครั้งละ 1 บิทเท่านั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในที่นี้อินพุทหรือเอาท์พุทพอร์ทก็จะกระทำได้ทีละบิท สำหรับการทำงานในวงจรนี้จะใช้แอดเดรสของอินพุทเอาท์พุท 3 บิท ซึ่งจะได้พอร์ทอินพุทเอาท์พุท 8 พอร์ท วงจรการใช้ไอซีสำหรับอินพุทจะใช้เบอร์ 4512 และเอาท์พุทเบอร์ 4099 ซึ่งไอซีทั้งสอง มีรายละเอียดดังแสดงในรูป

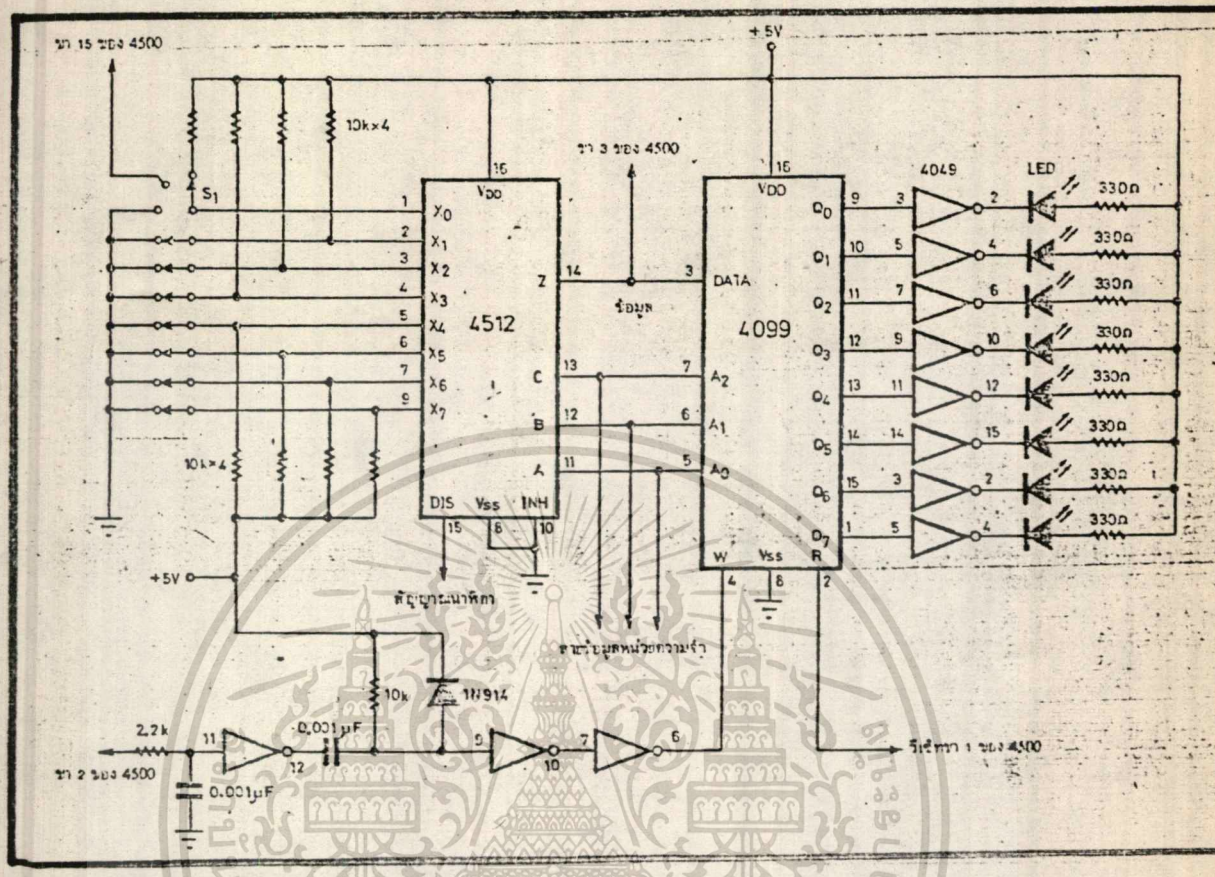


รูปแสดง รายละเอียดของไอซีส่วนอินพุท-เอาท์พุทเบอร์ 4512 และ 4099

วงจรอินพุท-เอาท์พุทนี้ ดังแสดงไว้ในรูป เป็นวงจรที่สามารถนำมาใช้ได้ในการเขียนโปรแกรม ดังนั้นจึงให้อินพุทเป็นสวิทช์และเอาท์พุทเป็น LED เพื่อสร้างสถานะการณ์หรือดูสถานะการณ์ของอินพุทเอาท์พุทเหล่านี้ได้

จากวงจรทั้งหมดที่กล่าวถึงนี้ เมื่อประกอบกันจะเป็นไมโครคอมพิวเตอร์ แผ่นพิมพ์เดียวที่ทำงานครบทุกฟังก์ชั่น โดยไม่ต้องมีโปรแกรมมอนิเตอร์เลยแม้แต่น้อย การใช้งานจะใช้สวิทช์กดปุ่มเป็นตัวเลือก เช่น โยกสวิทช์ S_1 มาที่ซิงเกิ้ลสเตป, กด PB_2 เพื่อรีเซทกำหนดแอดเดรสด้วยสวิทช์ที่โปรแกรมเคาวน์เตอร์, กดปุ่มคีย์บอร์ดตามด้วยปุ่มโหลดแล้วโยกสวิทช์แอดเดรสไปเป็นแอดเดรสถัดไป กดปุ่มคีย์บอร์ดตามด้วยปุ่มโหลด ทำเช่นนี้จนป้อนโปรแกรมสำเร็จ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง วงจรอินพุท-เอาท์พุทที่สมบูรณ์

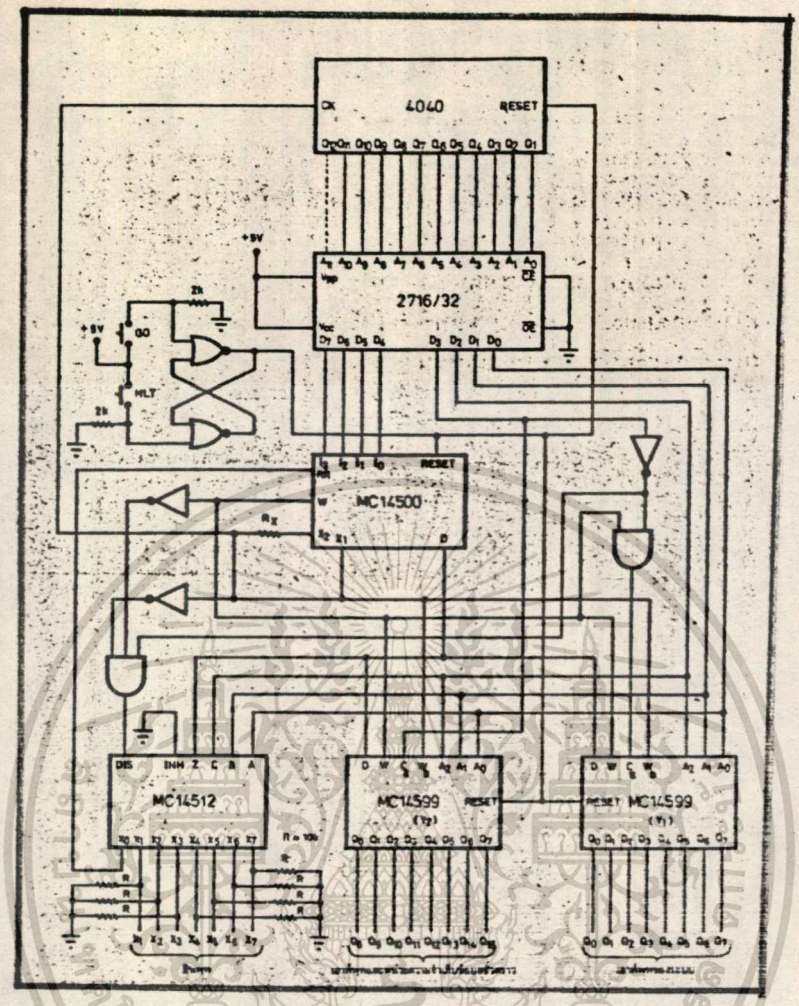
3.8 เมื่อต่อเป็นระบบ

จากการที่เราได้ใช้ไอซี 4040 เป็นวงจรสำหรับโปรแกรมคาวน์เตอร์และใช้ EPROM เบอร์ไอซี 2716/2732 เสมือนเป็น ROM ของระบบ และใช้ MC14512 เป็นอินพุทของระบบ และใช้ไอซีMC 14599 เป็นเอาท์พุทของระบบ ในที่นี้จะนำไอซีดังกล่าวนี้มาต่อเป็นระบบเพื่อใช้งาน

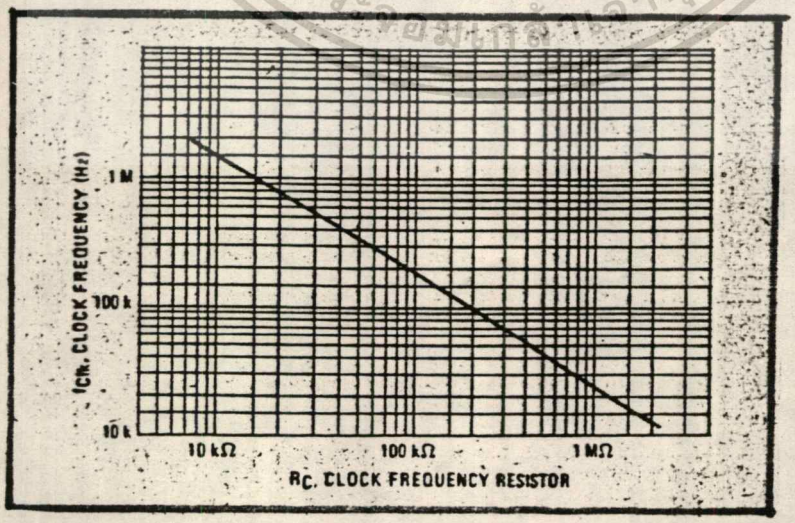
วงจรที่ต่อขึ้นมาดังแสดงในรูป ซึ่งเป็นวงจรที่เราจะนำเอา ไอซี MC 14500ไมโครโปรเซสเซอร์ 1 บิท นี้ไปใช้งานได้ โดยการเขียนโปรแกรมลงใน ROM เพื่อสั่งให้ทำงานตามที่ต้องการ สังเกตว่าการเริ่มต้น RUN โปรแกรมจะเริ่มต้นที่แอดเดรส 0 และวิ่งวนไปเรื่อย ๆ ตามการนับของโปรแกรมเคาวน์เตอร์

การต่อเป็นระบบเพื่อใช้งานนี้ เราต่อ R_X เข้ากับระบบเพื่อให้วงจรออสซิลเลเตอร์ภายในทำงาน โดยค่า R_X ที่ต่อกับระบบนี้ จะเป็นตัวกำหนดความถี่ โดยค่าของ R_X จะใช้ได้ตั้งแต่ 10KΩเป็นต้นไป ดังแสดงในรูปของกราฟ เป็นการฟแสดงค่า R_X กับความถี่ที่ได้ออกมาจากระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น ห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง วงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์ไอซี MC 14500 สำหรับการใช้งานได้จริง ๆ



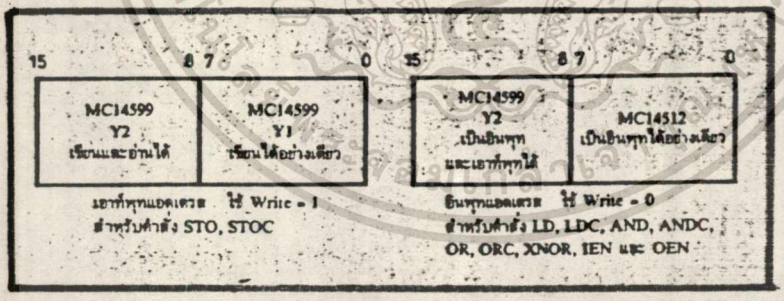
รูปแสดง ค่าความถี่ของสัญญาณนาฬิกาแปรค่ากับ R_x

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากระบบแสดงในรูปวงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์ไอซี สำหรับการใช้งานได้จริง ๆ เราได้สัญญาณนาฬิกาออกทางขา X_1 (ขา 14) ของไอซี MC 14500 ต่อเข้ากับขา CK ของไอซี 4040 พิจารณาจากสวิทช์ GO เมื่อกด GO จะได้เอาท์พุทที่ต่อกับขารีเซ็ทเป็น "0" เป็นการเริ่มทำงานของระบบ และถ้ากดสวิทช์ HLT ก็จะทำให้หยุดการทำงาน การเริ่มรีเซ็ทจะกระทำเมื่อ ขารีเซ็ทเปลี่ยนจากลอจิก "1" เป็น "0" ซึ่งจะเคลียร์ ไอซี4040 และเอาท์พุทของ ไอซีMC14599ให้เอาท์พุทเป็น "0" แล้ววงจรออสซิลเลเตอร์ของไอซีMC 14500 ก็จะเริ่มทำการออสซิลเลท ทำให้ไอซี 4040 เริ่มนับ

ระบบเอาท์พุทที่ใช้ไอซี MC14599 ซึ่งเป็นวงจรแลทช์ 8 บิท นี้ มีการกำหนดชื่อของวงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์ MC14500 สำหรับการใช้งานได้จริง ๆ เป็นตัว Y1 และ Y2 โดยตัว Y1 เป็นเอาท์พุทของระบบไมโครคอมพิวเตอร์นี้ ส่วน Y2 เป็นเอาท์พุทที่ทำหน้าที่เสมือนหน่วยความจำที่จะให้ไอซี MC14500 อ่านข้อมูลเข้ามาได้ Y2 จึงเสมือนเป็นหน่วยความจำเก็บข้อมูลชั่วคราวในระหว่างการทำงาน

การกำหนดพอร์ทอินพุทและเอาท์พุทของระบบเป็นการกำหนดในรูปแบบเสมือนเป็นบิท โดยสมารถเขียนตาราง อินพุท-เอาท์พุทของระบบแสดงในรูปวงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์ไอซี MC14500 ได้ ดังแสดงในรูป



รูปแสดง แสดงค่าตำแหน่งอินพุท-เอาท์พุทของระบบ

การเลือกเอาท์พุท-อินพุทจะใช้แอดเดรส 4 บิท ซึ่งเลือกแอดเดรสไปยังอินพุท-เอาท์พุทได้ 16 บิท เช่น ถ้าคำสั่ง LD0000 จะมีความหมายอ่านข้อมูลบิท X_0 จาก MC14512 ไปไว้ที่ RR หรือถ้าใช้คำสั่งLDC0000 ของวงจรนี้ก็หมายถึงการทำ RR ให้เป็นคอมพลีเมนท์เพราะบิท X_0 ต่อกับรีจิสเตอร์ RR

หากทำคำสั่ง LD1000 ก็จะหมายถึงอ่านข้อมูลบิท D_0 ของไอซี MC14599 ในตัว Y_2 มาไว้ที่ RR และถ้าทำคำสั่ง STO 1000 ก็จะมีความหมายถึงการโหลดข้อมูลจาก RR ไปไว้ที่ Q_0 ของ Y2 สังเกตบิท D_3 ของไอซี 2716/2732 เป็นตัวกำหนด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แอดเดรสของอินพุท-เอาท์พุท ถ้าบิท D_3 เป็น "1" ก็จะเลือก CE ของ Y2 และถ้า D_3 เป็น "0" มันจะผ่านอินเวอร์เตอร์ เพื่อมาทำการเลือก ไอซี MC14512 หรือ Y1 ขึ้นกับคำสั่ง STO หรือ LD ซึ่งก็ขึ้นกับสัญญาณขา W ของ MC14500 นั่นเอง

ดังนั้นถ้าทำคำสั่ง LD 0001 หมายถึง โหลดบิท X_1 ของ MC14512 มาที่ RR จะได้ขา Write ของไอซี MC14500 เป็น "0" บิท D_3 เป็น "0" ผ่านอินเวอร์เตอร์มาเป็น "1" ขา X_1 เป็นสัญญาณนาฬิกาเป็น "0" ก็จะทำให้อินพุทของ AND เกททั้งสามเป็น "1" หมด ซึ่งก็ไปทำให้ขา DIS ของไอซี MC14512 เป็น "1" เพื่อทำให้ข้อมูลบิท X_1 มาที่ Z เพื่ออ่านเข้าในไอซี MC14500 เก็บไว้ที่ใน RR

หากต้องการดูการทำงานให้เข้าใจ ลองดูการถอดรหัสที่ใช้ AND เกทตัวที่จะเลือก Y1 และตัวที่จะเลือก MC14512 จะเห็นว่าเราสามารถทำคำสั่ง LD และ STO ในแอดเดรสต่าง ๆ ได้

3.9 นำระบบมาสร้างเป็น โปรแกรม เมเบิ้ล คอนโทรลเลอร์ (PLC)

ปัจจุบันอุปกรณ์จำพวก PLC กำลังได้รับการกล่าวขวัญในระบบการควบคุมขบวนการผลิตอย่างมาก วงจรที่เราสร้างมาในรูปวงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์ที่ใช้งานได้จริง ๆ สามารถทำเป็น PLC ได้อย่างดี โดยการเขียนโปรแกรมสั่งงานใน ROM แล้วเริ่มทำงานมันจะทำงานตาม แล้ววนโปรแกรมเคาน์เตอร์ไปจนครบทุกแอดเดรส และถ้าโปรแกรมเรายาวเพียงไม่กี่ไบท์ เราจะใส่แอดเดรสที่ไม่ใช้โปรแกรมเป็นคำสั่ง NOPO ไว้

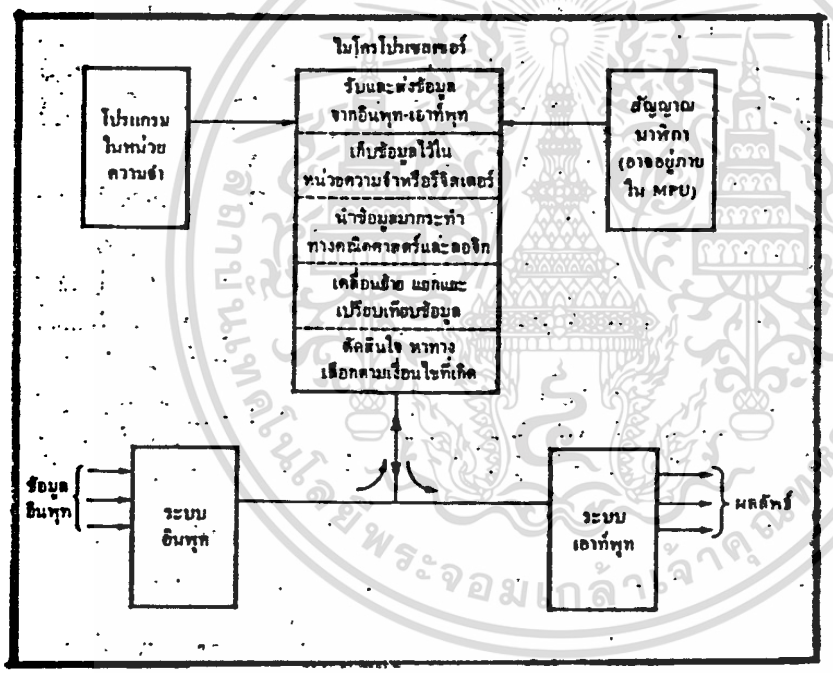
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

การทดลอง และ ผลการทดลอง

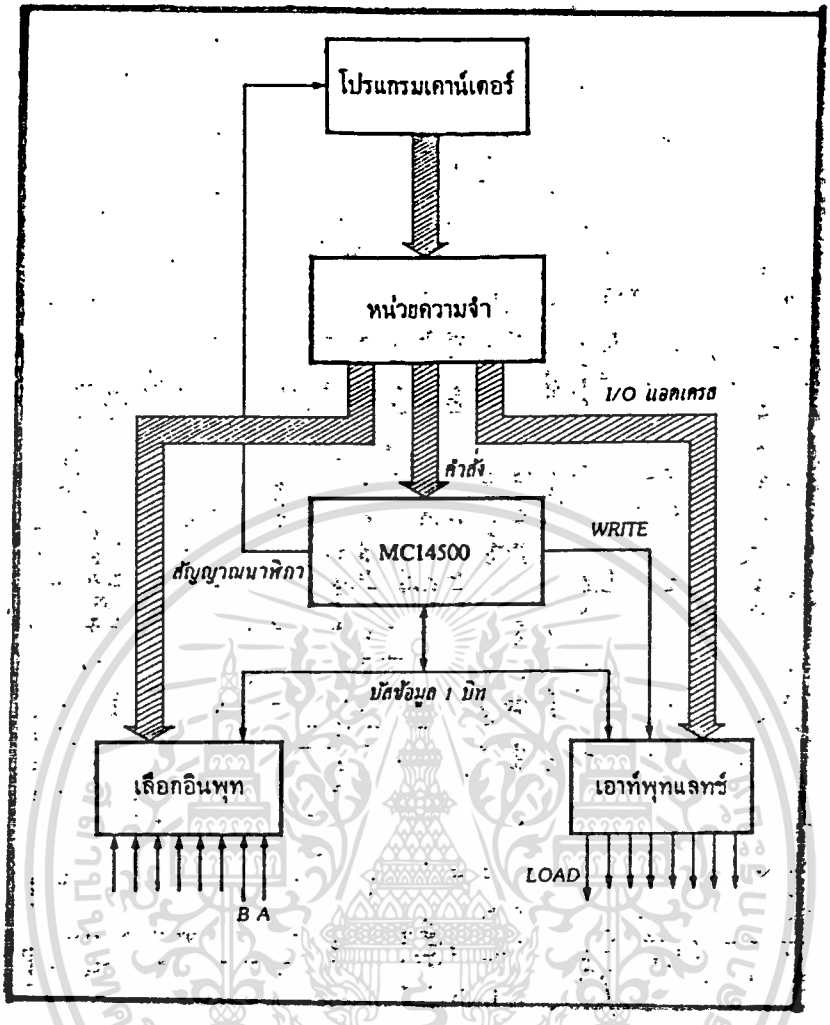
4.1 โครงสร้างระบบพื้นฐาน

ลำพัง MC14500 เพียงตัวเดียวยังทำงานไม่ได้ จำเป็นต้องประกอบร่วมกับหน่วยความจำเพื่อเก็บโปรแกรมคำสั่งและมีส่วนของอินพุทเอาท์พุทเพื่อเชื่อมต่อให้เป็นระบบเหมือนดังแสดงในรูประบบไมโครโปรเซสเซอร์ แต่สำหรับไมโครโปรเซสเซอร์เบอร์นี้จะต้องมีวงจรนับ ซึ่งทำหน้าที่เป็นโปรแกรมเคาวน์เตอร์อยู่ภายนอก ดังนั้นแผนผังของระบบไมโครโปรเซสเซอร์ MC 14500 นี้จึงเป็นไปดังแสดงในรูปถัดไป



รูปแสดง ระบบไมโครโปรเซสเซอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ระบบไมโครคอมพิวเตอร์ที่ใช้ MC 14500 เป็นซีพียู

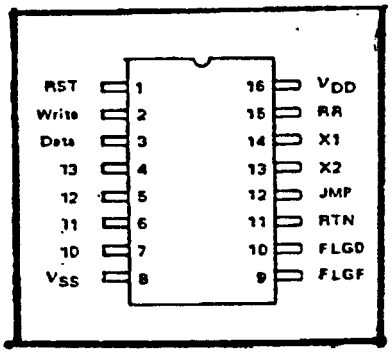
ระบบไมโครคอมพิวเตอร์นี้จึงดูไม่ซับซ้อนอะไร เพราะโครงสร้างการทำงานจะทำตามคำสั่งที่ส่งมาให้สั่งด้วยรหัส 4 บิท ส่งมามันจะทำงานตามจังหวะของสัญญาณนาฬิกาทันที และที่เด่นชัดของไอซีตัวนี้คือ มันทำงานได้แม้สัญญาณนาฬิกาจะช้าเพียงไรก็ได้ การทดลองจึงทำให้เห็นจริงจังได้ง่าย

MC 14500 กับการทดลอง ทำได้ดีที่สุด สือเนื่องจาก MC 14500เป็นไอซีที่ใช้ได้กับสัญญาณนาฬิกาความถี่ต่ำมากจึงทำการทดลองและเหมาะกับนักศึกษาที่จะเรียนรู้หลักการของไมโครโปรเซสเซอร์ได้เป็นอย่างดี ประจวบกับโครงสร้างการต่อทางฮาร์ดแวร์ได้น้อย และทดลองบนแผ่นโปรโตบอร์ดได้ทันที ในที่นี้จึงขอถือโอกาสเรียนรู้ไมโครโปรเซสเซอร์ด้วยการทดลอง

ก่อนที่จะทำการทดลองใด ๆ มาทำความเข้าใจกับขาต่าง ๆ ของไอซีไมโครโปรเซสเซอร์ตัวนี้กันก่อน ไอซีซีมอสไมโครโปรเซสเซอร์ตัวนี้มี 16 ขา และมีการจัดขาดังแสดงให้ดูดังรูป การจัดขา MC 14500ซึ่งเป็น 1 บิท ไมโครโปรเซสเซอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความหมายและหน้าที่ของขาต่าง ๆ ดังแสดงไว้ในตาราง



รูปแสดง การจัดขาของ MC14500 ซึ่งเป็น 1 บิท ไมโครโปรเซสเซอร์

ตาราง แสดงรายละเอียดของขาต่าง ๆของ MC14500

ขา	ความหมาย	หน้าที่และการทำงาน
1	RST	เป็นขาที่จะทำการรีเซ็ทการทำงานทั้งชิฟ มันจะทำการเคลียร์ค่าในรีจิสเตอร์ทุกตัวให้เป็น "0" และจะเซ็ทให้สัญญาณแฟลกเป็น "0" ด้วย การรีเซ็ทจะเกิดขึ้นเมื่อขา RST นี้ มีลอจิกเป็น "1" และชิฟจะเริ่มทำงานเมื่อขา RST นี้ลงเป็น "0" อีก
2	Write	ขาพัลส์สำหรับเขียน เป็นขาที่จะส่งพัลส์ออกมาหนึ่งพัลส์เมื่อซีพียูทำคำสั่ง STO หรือ STOC แต่อย่างไรก็ตาม ถ้าสังเกตการจากแผนผังภายในจะพบว่าพัลส์ Write จะได้รับการอีนาเบิ้ลหรือดิสเอเบิ้ล ด้วย OEN (เอาท์พุท อีนาเบิ้ล รีจิสเตอร์)
3	DATA IN/OUT	เป็นขาที่เป็นได้ทั้งอินพุทและเอาท์พุท จึงเป็นบัสแบบสองทิศทางที่จะรับหรือส่งข้อมูล ซึ่งเป็นเพียงการรับส่งข้อมูลทีละบิทเดียว
4-7	I_3-I_0	ขา 4 เป็นขารับคำสั่งบิท MSB ของรหัสคำสั่ง ขาทั้ง 4 นี้จะรับคำสั่งมาถอดความหมายและตรวจสอบ
8	V_{ss}	เป็นขากราวนด์
9	Flag F.	เป็นแฟลกเมื่อทำคำสั่ง NOPF (จะให้พัลส์ที่ขานี้)
10	Flag 0	เป็นแฟลกเมื่อทำคำสั่ง NOPO (จะให้พัลส์ที่ขานี้)
11	RTN	จะให้พัลส์ เมื่อทำคำสั่ง RTN
12	JMP	จะให้พัลส์ เมื่อทำคำสั่ง JMP
13	X2	เป็นอินพุทที่จะรับสัญญาณนาฬิกาจากภายนอก
14	X1	เป็นเอาท์พุทที่ส่งสัญญาณนาฬิกาออกไป โดยปกติถ้าต่อ R คร่อมขา 13, 14 จะทำให้เป็นการสร้างสัญญาณนาฬิกาในตัวเองได้
15	RR	เป็นขาเอาท์พุทแสดงค่าใน RR (Result Register)
16	V_{DD}	เป็นขาแหล่งจ่ายไฟเลี้ยง + V_{cc} ตั้งแต่ 3 - 18 V.

จากโครงสร้างทางฮาร์ดแวร์นี้เอง เราสามารถนำไอซี MC 14500 เพียงตัวเดียวมาทำการทดลองได้ เพราะขาต่าง ๆ ของไอซีก็เหมือนกับวงจรดิจิตอลทั่วไปที่กำหนด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

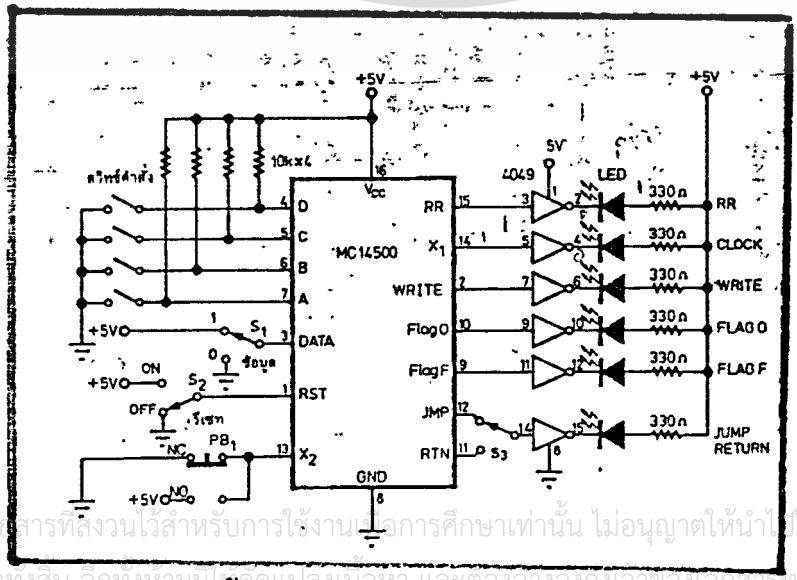
คำสั่งทางขาอินพุทแล้วดูการทำงานโดยตรง

4.2 วงจรไมโครโปรเซสเซอร์สำหรับการทดลอง

หลังจากที่ได้ศึกษาถึงไดอะแกรม โครงสร้างของ MC 14500 และได้ทราบว่าขาต่าง ๆ เป็นอะไรเรียบร้อยแล้ว ผู้เขียนขอให้กลับมาดูคำสั่งของคำสั่งต่าง ๆ ทั้ง 16 คำสั่ง เมื่อเทียบกับการทำงานในแผนผังทางฮาร์ดแวร์ดู จะเห็นว่าแต่ละคำสั่งเป็นเพียงการทำงานพื้นฐานที่ง่าย ๆ โดยผลรับส่วนใหญ่จะตกอยู่ที่รีจิสเตอร์ RR และเนื่องจากโครงสร้างของไมโครโปรเซสเซอร์เบอร์นี้เป็นโครงสร้างเปิดเราจึงดูการทำงานได้อย่างดี

วงจรที่จะใช้ในการทดลองนี้ใช้ไอซีเบอร์ MC 14500กับไอซีซีมอสอินเวอร์เตอร์เบอร์ MC14049(4049) ต่อเพื่อขับ LED แสดงสถานะต่าง ๆ ให้เราทราบ โดยถ้าที่ขาของไมโครโปรเซสเซอร์เป็น "1" LED จะสว่าง ทั้งนี้เพราะอินเวอร์เตอร์จะให้ลอจิก "0" และ LED สว่าง

วงจรที่ใช้ทดลองนี้เป็นไปดังแสดงในรูป วงจรที่ต่อไว้นี้เราใช้สวิทช์ 4 ตัวแทนคำสั่งที่จะป้อนให้กับซีพียู (โดยปกติซีพียูจะอ่านคำสั่งมาจาก RAM หรือROM) ดังนั้นสถานะของสวิทช์ที่ต่อเป็น DCBA นี้จะแทนคำสั่งที่สั่งให้ซีพียูทำงาน ทำนองเดียวกันเราใช้สวิทช์กดต่อที่ขา 13 แทนสัญญาณนาฬิกาจากภายนอกที่ต่อเข้าให้กับระบบ โดยปกติสวิทช์ PB_1 นี้ จะอยู่ที่ NC (Normally Close) คือต่อกับกราวนด์ ครั้นถ้ามีใครกดจะต่อกับ "1" แล้วปล่อยมือจะเป็น"0" ขา 14 ซึ่งคือ X_1 จะให้ลอจิก "1" ดังนั้น ถ้าต่อ LED ไว้ LED ที่แสดง CLOCK จะสว่าง และเมื่อกด PB_1 ตัว LED จะดับ ในการต่อสายนี้ ขา 12 ของไอซี MC14500 คือ Jump และขา 11 คือ RETURN แต่เนื่องจาก ไอซี 4049 มี 6 เกท ดังนั้นจึงทำสวิทช์เลือกไว้ว่าจะให้แสดง JUMP หรือ RETURN



รูปแสดง วงจรพื้นฐานไมโครโปรเซสเซอร์ไอซี MC14500

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

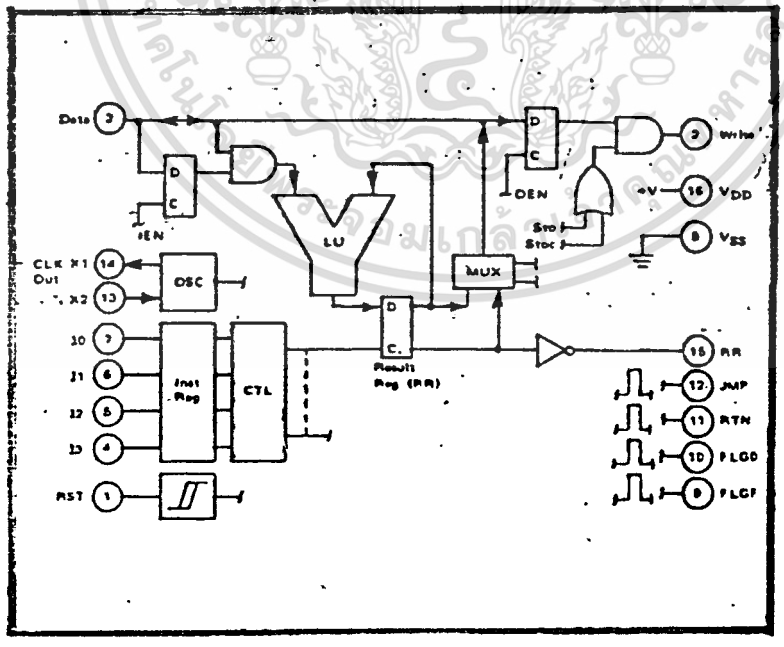
สังเกตการต่อพีขา DATA (ขา 3) ซึ่งเป็นบัสของระบบ ในที่นี้เราจะต่อกับสวิทช์เพื่อเลือก "0" หรือ "1" นั่นคือ ข้อมูลที่จะนำเข้านี้เลือกได้ว่าเป็น "0" หรือ "1"

4.3 การทดลองคำสั่งของไมโครโปรเซสเซอร์

เมื่อเริ่มต้นการทดลอง ให้สวิทช์คำสั่ง DCBA เป็น 0000 ซึ่งแทนคำสั่ง NOPO หรือ No Operation และเมื่อต่อสวิทช์ไฟเลี้ยงเข้าวงจรจะยังไม่มี LED ใดติด ยกเว้น CLOCK ให้กดสวิทช์ PB_1 แทนสัญญาณนาฬิกา จะปรากฏว่า LED ที่แทน FLAGE ติดสว่าง แล้วทำการรีเซ็ทระบบ FLAGE 0 ติดสว่างซึ่งหมายความว่าซีพียู ได้ทำตามคำสั่ง NOPO แล้ว เราจะทำให้ LED แสดง Flage 0 นี้ดับได้ด้วยการโยกสวิทช์รีเซ็ท (S_2) ไปยัง 5 โวลท์ แล้วแยกกลับมาที่เดิม ตรานี้จะเห็นว่า LED ทุกดวงดับหมด (ยกเว้น CLOCK)

ให้โยกสวิทช์คำสั่งมาเป็นคำสั่ง NOPF หรือรหัส 1111 กดสวิทช์สัญญาณนาฬิกา จะเห็นว่า LED แสดง FLAGE ติดสว่าง แล้วทำการรีเซ็ทระบบด้วยสวิทช์ S_2 ใหม่

คำสั่งที่อยากจะให้ทดลองต่อไปคือคำสั่งโหลดข้อมูลจากบัสข้อมูล (ขา 3) เข้าไปเก็บไว้ที่ RR แต่ก่อนที่จะโหลดจะต้องอินาเบิ้ลอินพุทหรือทำให้ D ฟลิบฟลอบที่ IEN เป็น "1" ก่อน ดังแสดงในรูป โครงสร้างของไมโครโปรเซสเซอร์



รูปแสดง โครงสร้างของไมโคโปรเซสเซอร์ MC14500

ในการนี้จะทำคำสั่ง IEN ซึ่งเป็นรหัสไบนารี่ 1010 แล้วกด PB_1 จะพบว่าไม่มีอะไรเกิดขึ้นให้เห็นเพราะรีซิสเตอร์ IEN อยู่ภายใน หลังจากนั้นจะทำคำสั่ง LD ซึ่งรหัส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ในรูปแบบการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำสั่ง 0001 โดยให้ขา 3 หรือบัสข้อมูลต่อกับลอจิก "1" หลังจากนั้นกดสวิทช์ PB_1 เพื่อกระทำคำสั่ง LD จะพบว่ารีจิสเตอร์ RR เป็น "1" และ LED แสดง RR จะสว่างให้เห็นได้

คราวนี้ลองทำใหม่ด้วยการดิสเอเบิ้ล IEN ด้วยการให้ขา 3 ของ MC14500 ต่อกับ "0" โยกสวิทช์คำสั่งเป็น IEN คือ 1010 กดสวิทช์สัญญาณนาฬิกาจากนั้นต่อขา 3 กับ $+V_{cc}$ หรือ 5 โวลท์ด้วย S_1 แล้ว เปลี่ยนคำสั่งมาเป็น LD คือรหัส 0001 ใหม่ กดสวิทช์ PB_1 คราวนี้ LED แสดง RR จะไม่สว่าง

ลองทำคำสั่ง LDC แต่ก่อนทำ ให้ทำการอีนาเบิ้ลด้วยคำสั่ง IEN โดยโหลดข้อมูล "1" ไว้ที่รีจิสเตอร์ IEN ก่อน เมื่อทำคำสั่ง LDC จะเห็นว่า LED แสดง RR ดิสสว่างให้ลอจิกตรงข้ามกับข้อมูลบนบัส จะเห็นว่าคำสั่ง NOPO, LD, LDC, IEN และ NOPF เป็นคำสั่งที่ทดลองดู การทำงานได้ด้วยตาอย่างชัดเจน

ทำการรีเซ็ทสวิทช์ S_2 โดยโยกมาที่ลอจิก "1" แล้วกลับไปลอจิก "0" เพื่อรีเซ็ทระบบทั้งหมด จากนั้นลองต่อสวิทช์ S_2 มาที่ JUMP หรือ RETURN เซ็ทรหัสคำสั่งให้เป็นคำสั่งตรงกับ JUMP หรือ RETURN เช่นถ้า JUMP ก็เป็น 1100 ถ้า RTN ก็เป็น 1101 ลองกระทำดูจะเห็น LED ที่แสดง JUMP หรือ RETURN ดิสสว่างและที่สังเกตคือมันจะข้ามคำสั่งถัดมาโดยไม่ทำอะไร (ลองดูด้วยการป้อนคำสั่งให้)

ให้ลองทำคำสั่งซ้ำด้วยคำสั่ง OEN, STO และ STOC โดยลองทำขั้นตอนดังนี้

1. ให้โหลดตัวเลขจากบัสข้อมูลไปเก็บไว้ยัง RR (ในที่นี้ลองใช้ลอจิก "1")

2. ในการโหลดตัวเลขจากบัสข้อมูลมายัง RR นี้ ต้องใช้ 2 คำสั่งคือ IEN และ และ LD เหมือนที่ได้ทดลองมาแล้ว โดยขณะทำ IEN ต้องให้บัสข้อมูลเป็น "1"

3. ทำคำสั่ง OEN ด้วยการให้บัสข้อมูลเป็น "1" เซ็ทรหัสคำสั่งเป็น 1011 กดสัญญาณนาฬิกา ลอจิก "1" จากบัสข้อมูลจะได้รับการแลทช์ไปดักบที่รีจิสเตอร์ OEN

4. ตอนนี้อย่าเพิ่งทำอะไรในตัวบีพียู ให้ย้ายสายไฟฟ้าจากบัสข้อมูลขา 3 ของ MC14500 ไม่ให้ต่อกับ S_1 แต่มาต่อเข้าขา 14 ของ 4049 เพื่อรับข้อมูลที่จะส่งออกมาจากบัสต้องทำด้วยความระมัดระวัง

5. เมื่อย้ายสายต่อได้เรียบร้อยแล้ว เซ็ทรหัสคำสั่ง STO ด้วยรหัส 1000 กดสวิทช์สัญญาณนาฬิกา จะปรากฏมีลอจิก "1" ออกมาทางขา 3 ของ MC14500 และขณะเดียวกันจะมีสัญญาณออกทางขา Write

6. ลองทำใหม่อีกครั้งโดยใช้คำสั่ง STOC ซึ่งมีรหัสคำสั่ง 1001 จะปรากฏข้อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น และห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

มูลออกมาทางขา 3 เป็นค่าที่คอมพลีเมนท์จาก RR และ LED ที่ขา Write ก็สว่างด้วย จะเห็นได้ว่าการสั่งการของคำสั่ง LD และ STO จะเป็นการนำข้อมูลจากบัสมาที่ RR และส่งจาก RR ออกไปยังบัสข้อมูล

ให้ทำการเซ็ทวงจรใหม่เหมือนเดิมโดยต่อสวิทช์ S_1 กับขา 3 อย่างเดิม และสวิทช์ S_3 กับขา JUMP และ RETURN แบบเดิม

4.4 การทดลองทำ AND-OR-XNOR

คำสั่งของ MC14500 ยังมีอีกหลายคำสั่งโดยเฉพาะคำสั่งที่เกี่ยวกับการทำงานทางลอจิก เช่น AND, OR และ XNOR เป็นต้น

ในขั้นนี้เราจะลองทำคำสั่งทางลอจิกเหล่านี้ดู โดยทำตามขั้นตอนต่อไปนี้

1. ให้เซ็ท IEN ด้วยลอจิก "1"
2. ให้ LD ข้อมูลจากบัสข้อมูลลอจิก "1" ไปเก็บไว้ที่ RR
3. ให้เซ็ทสวิทช์คำสั่งเป็น AND ด้วยรหัสคำสั่ง 0011 เพื่อทำ AND ฟังก์ชั่น (การทำ AND นี้ MC14500 จะนำข้อมูลจากบัสข้อมูลมา AND กับ RR ผลลัพธ์เก็บไว้ที่ RR)
4. ตั้งสวิทช์ S_1 เป็น "1" กดสัญญาณนาฬิกาให้ทำงาน

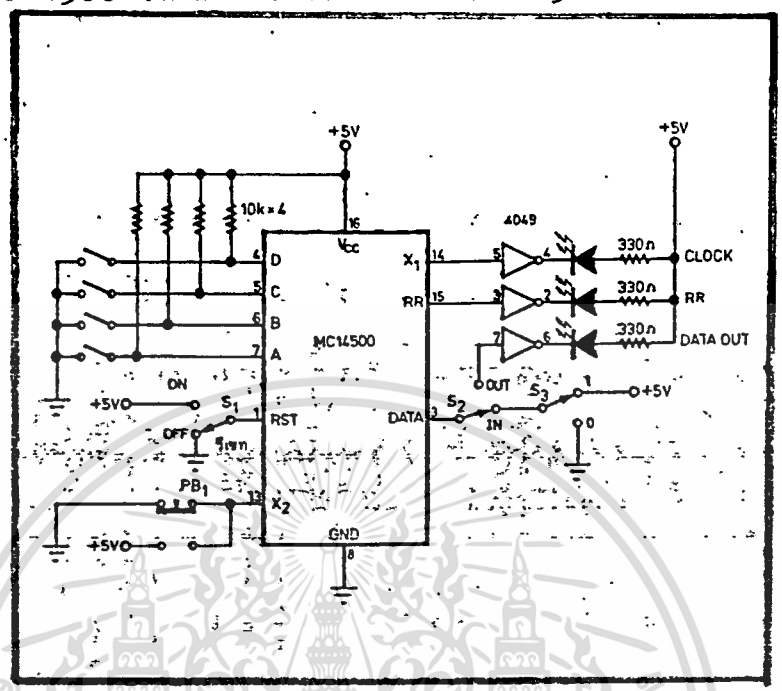
สังเกตค่าใน RR โดยดูจาก LED จะเห็นว่า RR คือผลลัพธ์ของการ AND กัน ลองทำ AND ใหม่โดยเปลี่ยนแปลงค่าในบัสข้อมูลเป็น "0" ในการทำงานอาจส่งข้อมูลออกมาทางบัสข้อมูลด้วยคำสั่ง OEN และ STO ก็ได้ หากต้องการทำคำสั่ง ANDC (รหัสคำสั่ง 0100) ก็ทดลองดูได้ โดยการทำงานของไมโครโปรเซสเซอร์ จะนำค่าจากบัสข้อมูลมาคอมพลีเมนท์แล้ว AND กับ RR ผลลัพธ์เก็บไว้ที่ RR และด้วยวิธีการเดียวกันนี้ เราสามารถทดลองทำคำสั่ง OR และ ORC โดยคำสั่ง 0101 ส่วน ORC มีรหัสคำสั่ง 0110 ผลลัพธ์ที่ได้จะเป็นการนำเอาข้อมูลจากบัสข้อมูลเข้ามาทำการ OR กับ RR โดยหากใช้คำสั่ง OR ก็จะไม่มีการคอมพลีเมนท์บัสข้อมูล แต่ถ้าใช้ ORC ก็จะมีการคอมพลีเมนท์บัสข้อมูล

คำสั่งทางลอจิกอีก 1 คำสั่ง คือ XNOR (รหัสคำสั่ง 0111) นี้ ก็ทำงานไม่ทำนองเดียวกัน

4.5 จะทำไอซี MC14500 ให้เป็นเกทธรรมดาได้อย่างไร

จุดมุ่งหมายของไมโครโปรเซสเซอร์การใช้ซอฟท์แวร์สั่งการให้ไมโครโปรเซสเซอร์ทำงานตามต้องการเช่น สั่งให้ทำงานแทนฮาร์ดแวร์ลอจิกอื่น ๆ ได้ด้วยคำสั่ง ทำให้สามารถแทนไมโครโปรเซสเซอร์เป็นวงจรลอจิกได้มากและคล่องตัวได้ดียิ่งขึ้น ผู้ออกแบบสามารถออก

แบบทางฮาร์ดแวร์เพื่อให้ทำงานตามต้องการได้ ตัวอย่างของวงจรดังแสดงในรูป เป็นตัวอย่างของการต่อไอซี MC14500 เพื่อทดลองสั่งงานให้ไอซี MC14500 เป็นวงจรตามที่ต้องการ



รูปแสดง การต่อไอซี MC14500ให้เป็นวงจรทดลองการทำงานของฮาร์ดแวร์ลอจิก

สิ่งที่น่าสังเกตของการต่อในขณะนี้คือการนำสวิทช์ S_2 และ S_3 มาต่อกับบัสข้อมูลคือขา 3 เพื่อให้บัสข้อมูลเป็นได้ทั้งอินพุทและเอาท์พุท กล่าวคือ ถ้าบัสข้อมูลต่อกับสวิทช์ S_3 ก็จะเป็นอินพุท ถ้าต่อกับไอซี 4049 ทางขา 7 ก็จะเป็นบัสเอาท์พุท ในการทดลองวงจรลอจิกเราสามารถทำการทดลองได้ดังนี้

4.6 <u>การทำเป็นวงจรเกทชนิดต่าง ๆ</u>

ในการแสดงการทำงานเป็นเกทต่าง ๆ ขอแสดงเป็นขั้นตอนของ AND เกทก่อนดังนี้

1. เริ่มจากเซ็ทบัสข้อมูลเป็นอินพุทต่อกับลอจิก "1"
2. ทำคำสั่ง IEN ด้วยรหัส 1010
3. ทำคำสั่ง OEN ด้วยรหัส 1011
4. โหลดข้อมูลจากบัสอินพุทด้วยคำสั่ง LD รหัส 0001 ข้อมูลจากบัสอินพุทจะมาเก็บไว้ที่ RR
5. เซ็ทข้อมูลอินพุทที่ต้องการ AND กับ RR จากนั้นให้ทำคำสั่ง AND ด้วยรหัส 0011

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. ย้ายบัสข้อมูลมาทางเอาท์พุทด้วยสวิทช์ S_2

7. ทำคำสั่ง STO ด้วยรหัส 1000 ผลลัพธ์จะไปปรากฏที่ LED ตามต้องการ

โปรแกรมที่จัดการข้อมูลด้วยการ AND จึงเขียนได้ดังนี้

IEN	1010
OEN	1001
LD	0001
AND	0011
STO	1000

ทำนองเดียวกันถ้าทำ NAND จะเขียนโปรแกรมได้เป็น

IEN	1010
OEN	1011
LD	0001
AND	0011
STOC	1001

ถ้าทำคำสั่ง OR จะเขียนโปรแกรมได้เป็น

IEN	1010
OEN	1011
LD	0001
OR	0101
STO	1000

หากทำคำสั่ง XNOR จะเขียนโปรแกรมได้ดังนี้

IEN	1010
OEN	1011
LD	0001
XNOR	0111
STO	1000

จะเห็นว่าเราไม่มีคำสั่ง XOR แต่ถ้าจะทำ XOR ก็เพียงแต่ส่งข้อมูลคอมพลีเมนท์ออกมาดังนี้

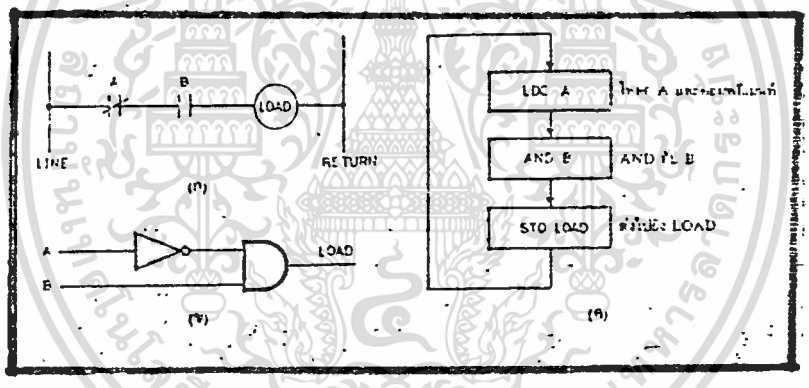
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

IEN	1010
OEN	1011
LD	0001
XNOR	0111
STOC	1001

ในกรณีของการเขียนโปรแกรมนี้จะทำการทดลองทีละขั้นดังที่กล่าวแล้ว จะทำให้เข้าใจระบบไมโครโปรเซสเซอร์ได้ดียิ่งขึ้น

4.7 การใช้คำสั่ง LDC

จากการแสดงในรูป เป็นตัวอย่างของโปรแกรมการใช้คำสั่ง LDC ในการโปรแกรมการทำงานทางลอจิก โดยการอ่านอินพุท A มาลอมพลีเมนท์ แล้วมาทำการ AND กับ B จากนั้นจึงนำมาส่งให้เอาท์พุทซึ่งเป็นโหลด



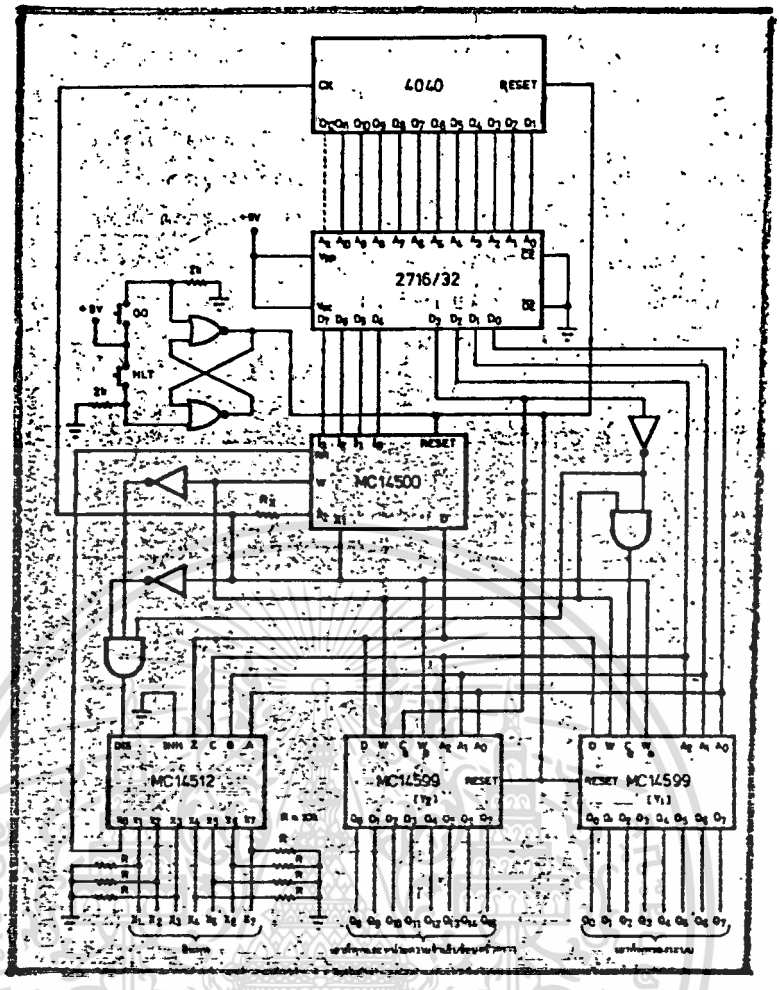
รูปแสดง วงจรและโปรแกรมที่เขียนขึ้นแสดงการทำงานอย่างง่าย ๆ

สังเกตว่าจากวงจรในรูป วงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์สำหรับการใช้งานได้จริง ๆ ให้ A เป็น X_1, B เป็น X_2 และ LOAD คือ Q_0 ของ Y1 เราก็เขียนเป็นโปรแกรมได้

LDC	0001
AND	0010
STO	0000

เมื่อแอดเดรส 0001 ของอินพุทคือ X_1 และ 0010 คือ X_2 ส่วนแอดเดรสเอาท์พุท 0000 คือ Q_0 (Y1)

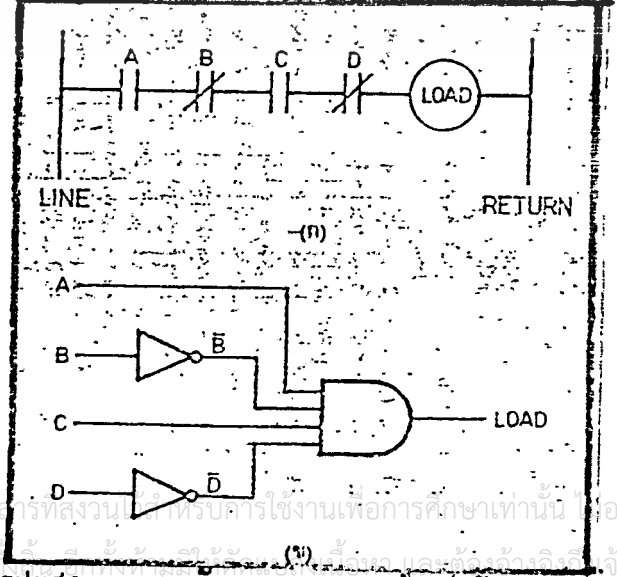
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง วงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์ MC14500 สำหรับการใช้งานได้จริง ๆ

4.8 การใช้คำสั่ง ANDC

คำสั่ง ANDC เป็นการ AND กับข้อมูลโดยก่อน AND ให้นำข้อมูลมาทำคอมพลีเมนท์ก่อน ดังแสดงในรูป เป็นการใช้คำสั่ง ANDC กับการโปรแกรมลอจิกดังรูปที่แสดง



รูปแสดง วงจรตัวอย่างของการใช้คำสั่ง ANDC

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งไม่อนุญาตให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากรูปที่แสดงนี้ ถ้าเขียนเป็นโปรแกรมบรรจุไว้ใน ROM จะได้ดังโปรแกรมที่แสดง

โปรแกรมที่ 1 การใช้คำสั่ง ANDC

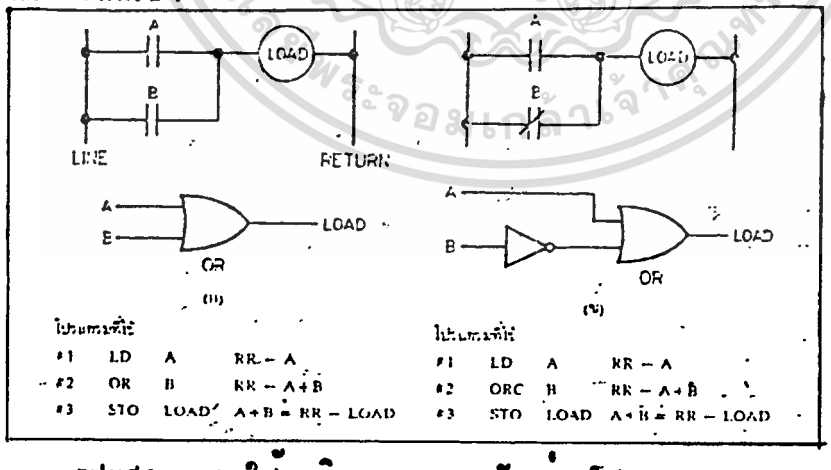
บรรทัดที่	คำสั่ง	โอเปอร์แรนด์	ความหมาย
#1	LD	A	RR – A
#2	ANDC	B	RR – $A.\overline{B}$
#3	AND	C	RR – $A.\overline{B}.C$
#4	ANDC	D	RR – $A.\overline{B}.C.\overline{D}$
#5	STO	LOAD	RR – LOAD

ถ้าหากแทน A, B, C, D เป็นอินพุท X_1, X_2, X_3, X_4 เราก็สามารถนำมาสร้างเป็นรหัสเพื่อบรรจุใน ROM ได้ อย่าลืมว่าแอดเดรสของโปรแกรมเคาน์เตอร์จะวิ่งวนรอบ โดยจะเสมือนทำทุกคำสั่งใน ROM อย่างรวดเร็วโดยคำสั่งที่ไม่ใส่ไว้ ให้ใส่เป็นคำสั่ง NOPO

4.9 การใช้ OR และ ORC

ในการโปรแกรมลอจิกนั้น เราโปรแกรมในรูปแบบของ OR ด้วยและมีความจำเป็นในเรื่องของลอจิกที่จะใช้งานในระบบประมวลผลสำหรับการทำงานในอุตสาหกรรม

จากรูปที่แสดง เป็นการใช้ลอจิก OR และตัวอย่างโปรแกรมด้วย การใช้ลอจิก OR นี้ จึงเท่ากับการต่อขนาน และเราสามารถใช้ได้ทั้ง OR และ ORC ซึ่งทำให้เราเขียนโปรแกรมได้สั้นเข้า

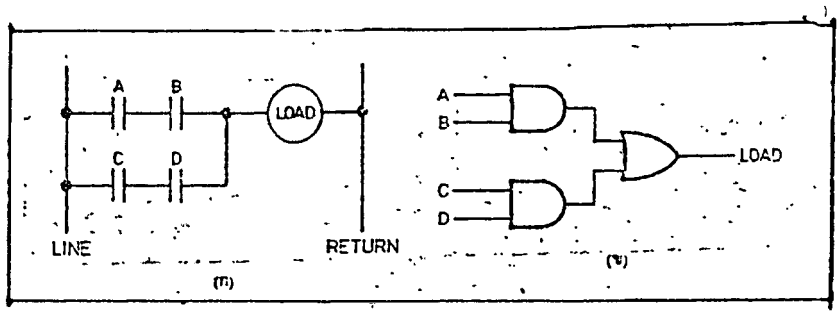


รูปแสดง การใช้ลอจิก OR และตัวอย่างโปรแกรม

4.10 การใช้หน่วยความจำเก็บข้อมูลชั่วคราว

จากวงจรสมบูรณ์ระบบไมโครโปรเซสเซอร์ไอซี MC14500 ที่ใช้งานได้จริง ๆ เราให้เอาท์พุทในแอดเดรส Q_8 – Q_{15} เป็นเสมือนหน่วยความจำที่เขียนและอ่านได้ ซึ่งก็จะทำให้ การใช้งานได้คล่องตัวยิ่งขึ้น ลองดูจากรูปที่แสดง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ตัวอย่างของวงจรซึ่งต้องการใช้หน่วยความจำเก็บข้อมูลชั่วคราว

จากรูปที่แสดง จะเห็นว่า ถ้าเราไม่มีหน่วยความจำเก็บสถานะลอจิกบางตัวไว้ก่อน เราจะไม่สามารถประมวลผล วงจรลอจิกตามรูปที่แสดงได้ ลองดูโปรแกรมที่ 2

โปรแกรมที่ 2 การใช้คำสั่ง OR โดยตรงทำให้โปรแกรมผิดพลาด

#1	LD	A	RR — A
#2	AND	B	RR — A.B
#3	OR	C	RR — A.B+C (ผิดแล้ว)
#4	AND	D	RR — (A.B+C).D
#5	STO	LOAD	RR — LOAD

การเขียนแบบนี้จะผิดพลาดจากความเป็นจริง เพราะเราต้องนำผลลัพธ์ของ A,B ไปเก็บไว้ก่อน ที่ถูกต้องจะต้องเขียนดังโปรแกรมที่ 3

โปรแกรมที่ 3 การแก้ไขโปรแกรมที่ 2 โดยนำผลลัพธ์ของ A,B ไปเก็บไว้ที่ TEMP ก่อน

#1	LD	A	RR — A
#2	AND	B	RR — A.B
#3	STO	TEMP	RR = A.B — TEMP
#4	LD	C	RR — C
#5	AND	D	RR — C.D
#6	OR	TEMP	RR — C.D+(TEMP=A.B)
#7	STO	LOAD	RR = A.B+C.D — LOAD

ถ้าเป็นวงจรสมบูรณ์ระบบไมโครโปรเซสเซอร์ไอซีMC14500 ที่ใช้งานได้จริง ๆ ซึ่งเป็นโปรแกรมแมเบิ้ล ลอจิก คอนโทรลเลอร์ เราจะเขียนตามลอจิคนี้ได้ เช่นให้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

A TO X_1

B TO X_2

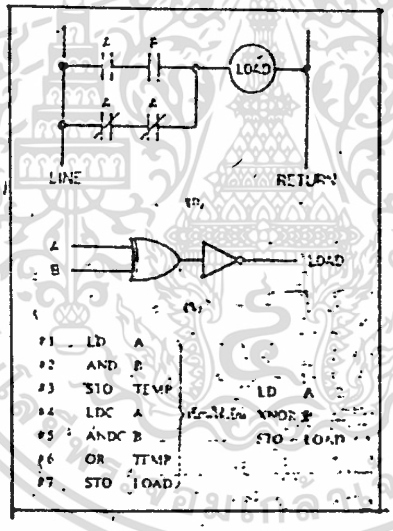
C TO X_3

D TO X_4

และ TEMP เป็น Q_8 ส่วน LOAD เป็น Q_0 สังเกตว่า TEMP เป็น Q_8 เพราะเราสามารถเขียนและอ่านผ่านพอร์ทนี้ได้

4.11 คำสั่ง XNOR

คำสั่ง EX -NOR เป็นคำสั่งที่เมื่ออินพุทเหมือนกันจะได้เอาท์พุทเป็น "1" ถ้าอินพุทที่ต่างกันจะให้เอาท์พุทเป็น "0" ซึ่ง XNOR จะทำให้การเขียนคำสั่งได้สั้นเข้า ลองพิจารณาจากรูปที่แสดง



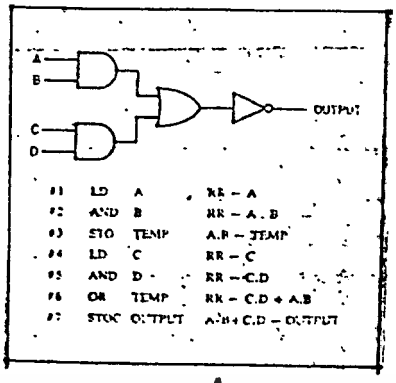
รูปแสดง วงจรและโปรแกรมของการใช้คำสั่ง XNOR

จะเห็นว่าคำสั่ง XNOR ทำให้การเขียนโปรแกรมได้สั้นขึ้น เพราะสามารถแทนได้หลายคำสั่ง

4.12 คำสั่ง STOC

ICU ของโมโตโรล่านี้ ได้เน้นคำสั่งสำหรับงานโปรแกรมลอจิกโดยเฉพาะ ดังนั้นการกำหนดคำสั่งต่าง ๆ ขึ้นมาก็เพื่อให้การทำงานต่าง ๆ ได้ดี และรวดเร็วขึ้น ดังแสดงในรูป

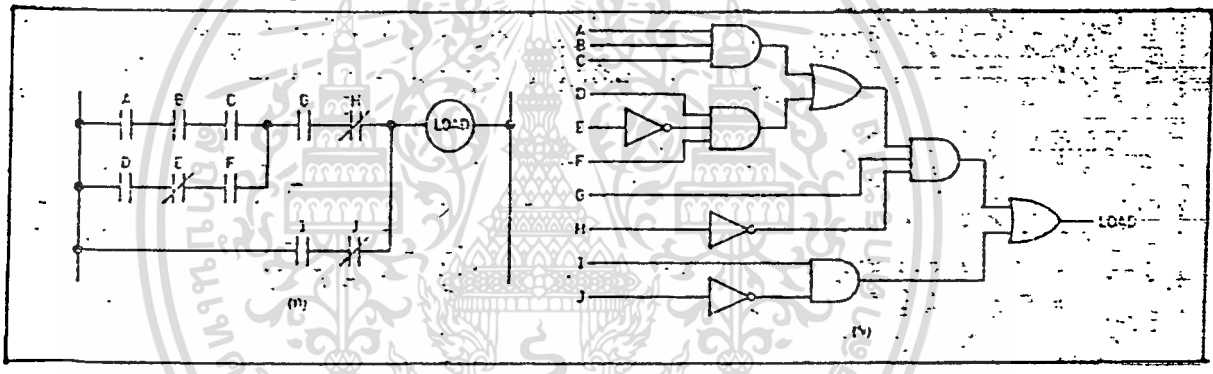
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง วงจรและโปรแกรมของคำสั่ง STOC เมื่อมีอินเวอร์เตอร์หางเอาท์พุท

คำสั่ง STOC ก็เสมือนการคอมพลีเมนท์ก่อน แล้วจึงส่งไปยังเอาท์พุทตามต้องการ คำสั่งเหล่านี้ทำให้เราเขียนโปรแกรมแทนการทำงานของลอจิกต่าง ๆ ได้

พิจารณาจากรูปที่แสดง เป็นตัวอย่างการใช้คำสั่งในการทำงานลอจิกที่ซับซ้อนขึ้น



รูปแสดง ตัวอย่างของวงจรลอจิกที่ซับซ้อน

เมื่อนำมาเขียนเป็นโปรแกรมจะได้ดังโปรแกรมที่ 4

โปรแกรมที่ 4 โปรแกรมจากวงจรลอจิกที่ซับซ้อน

#1	LD	A	RR – A
#2	AND	B	RR – A.B
#3	AND	C	RR – A.B.C.
#4	STO	TEMP	A.B.C. – TEMP
#5	LD	D	RR – D
#6	ANDC	E	RR – $D.\overline{E}$
#7	AND	F	RR – $D.\overline{E}.F$
#8	OR	TEMP	RR – $A.B.C.+D.\overline{E}.F$
#9	AND	G	RR – $(A.B.C.+D.\overline{E}.F).G$
#10	ANDC	H	RR – $(A.B.C.+D.\overline{E}.F).G.\overline{H}$
#11	STO	TEMP	$(A.B.C.+D.\overline{E}.F).G.\overline{H}$ – TEMP
#12	LD	I	RR – I
#13	ANDC	J	RR – $I.\overline{J}$
#14	OR	TEMP	RR – $(A.B.C.+D.\overline{E}.F).G.\overline{H}+I.\overline{J}$
#15	STO	LOAD	$(A.B.C.+D.\overline{E}.F.).G.\overline{H}+I.\overline{J}$ – LOAD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

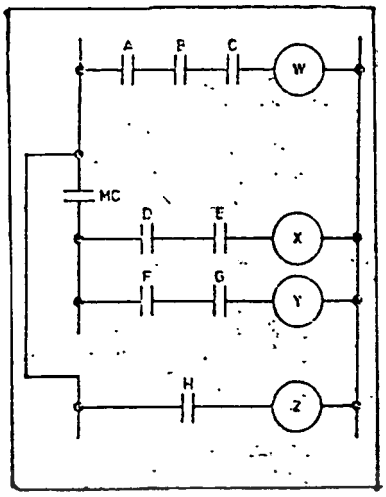
4.13 การใช้คำสั่ง IEN และ OEN

คำสั่งที่เพิ่มเติมเข้ามาและใช้งานได้ดี เพื่อให้สามารถควบคุมการทำงานเป็นลูปได้ โดยเมื่อทำคำสั่ง IEN จะมีความหมายว่า ให้ทำการนำข้อมูลจากบัสข้อมูลมาแลทช์ไว้ใน IEN รีจิสเตอร์ ทำให้ X,Y มีผลลัพธ์ได้ตามต้องการ ดังนั้น MC จึงเสมือนตัวอินาเบิ้ลอินพุทตามที่ต้องการ โปรแกรมที่เขียนตามลอจิก ดังได้แสดงในรูป ดังนี้

```
#1    LD    A
#2    AND   B
#3    AND   C
#4    STO   W
#5    IEN   MC
#6    LD    D
#7    AND   E
#8    STO   X
#9    LD    F
#10   AND   G
#11   STO   Y
#12   ORC   RR
#13   IEN   RR
#14   LD    H
#15   STO   Z
```

โปรแกรมดังกล่าวนี้จะเป็นตัวอย่างที่น่าศึกษามาก โดยเฉพาะบรรทัดคำสั่งที่ 5 คือ IEN MC จะทำให้บรรทัดที่ 6 ถึง 11 ขึ้นกับ MC ทันที บรรทัดที่ 12 เป็นบรรทัดที่น่าสนใจอีกบรรทัดหนึ่งเพราะ RR คือ Result Register ซึ่งถ้าในวงจรสมบูรณ์ระบบไมโครโปรเซสเซอร์ไอซี MC14500 ที่ใช้งานได้จริง ๆ จะต่อ X_0 กับ RR ดังนั้น ORC RR ก็คือ ORC X_0 ซึ่งจะทำให้ RR มีค่าเป็น 1 หรือคำสั่งเซ็ท RR ให้เป็น 1 นั่นเอง เมื่อเซ็ท RR เป็น 1 แล้ว X_0 ก็จะเป็น 1 ดังนั้น บรรทัดที่ 13 IEN RR ก็จะเป็น IEN X_0 ซึ่งจะทำให้อินาเบิ้ล เพื่อให้บรรทัดที่ 14 และ 15 จะทำงานตามปกติ

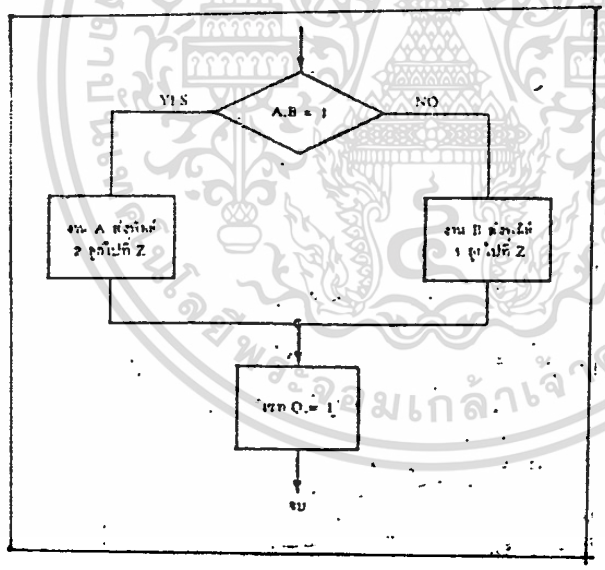
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ลอจิกที่มีทางเลือก

4.14 คำสั่ง OEN

คำสั่ง OEN ก็เป็นคำสั่งที่จะทำให้เกิดการเลือกไปยังเอาท์พุทต่าง ๆ ตามต้องการตามเงื่อนไขได้ ลองดูจากตัวอย่างการส่งไปยังเอาท์พุทแบบมีเงื่อนไข จากผังงาน ดังแสดงในรูป



รูปแสดง ผังงานแสดงเงื่อนไข

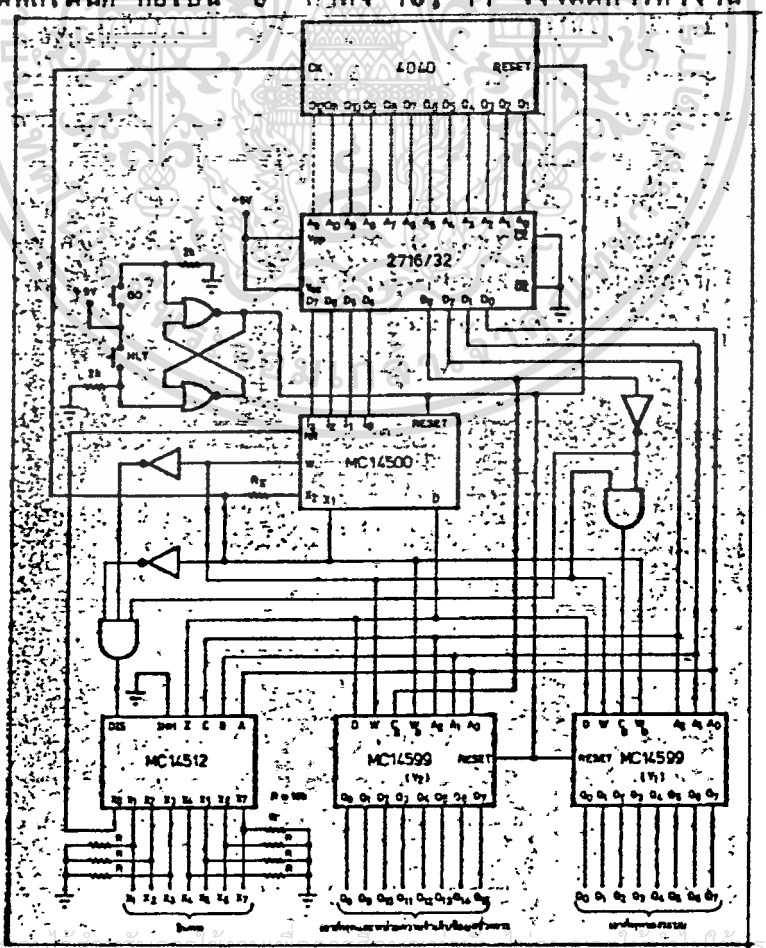
สังเกตจากรูปที่แสดง จะเห็นรีจิสเตอร์ OEN ซึ่งจะเป็นตัวยอมให้มีการส่งพัลซ์ Write ไปยังเอาท์พุทได้หรือไม่ ถ้าหากเรากำหนดให้ OEN รีจิสเตอร์เป็น "0" ก็จะไม่มีพัลส์ Write ออกไปทำให้คำสั่ง STO และ STOC ไม่เกิดผลในการทำงานได้ ด้วยเหตุผลอันนี้เองเราจึงใช้เป็นเงื่อนไขในการทำงานตามผังที่แสดงในรูป ได้ดังโปรแกรมที่ 5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมที่ 5 การใช้คำสั่ง OEN เพื่อสร้างเงื่อนไขของผังงานแสดงเงื่อนไข

#1	LD	A	RR – A	
#2	AND	B	RR – A.B	
#3	OEN	RR	OEN – RR = A.B	
#4	STO	Z	พัลส์	งาน A
#5	STOC	Z	พัลส์	
#6	STO	Z	พัลส์	
#7	STOC	Z	พัลส์	
#8	LDC	RR	RR – $\overline{RR}$ = $\overline{A.B}$	
#9	OEN	RR	OEN – RR	
#10	STO	Z	พัลส์	งาน B
#11	STOC	Z	พัลส์	
#12	ORC	RR	RR – 1	
#13	OEN	RR	OEN – RR = 1	
#14	STO	Q	1 = RR – Q	

สังเกตคำสั่งที่ 3 OEN RR ซึ่งถ้าเป็นวงจรแสดงรูปวงจรสมบูรณ์ระบบไมโครโปรเซสเซอร์ไอซี MC14500 อย่างใช้งานได้จริง ๆ จะใช้เป็น OEN X_0 ซึ่งถ้า X_0 = "1" ก็จะเป็นการอินาเบิ้ลให้ทำการส่งพัลส์ 2 พัลส์ไปที่ Z ด้วยคำสั่งที่ 4–7 หลังจากนั้นจะทำให้ค่า RR เป็นค่าคอมพลีเมนท์ คือเป็น "0" คำสั่ง 10, 11 จึงไม่มีการทำงาน



วงจรสมบูรณ์ของระบบไมโครโปรเซสเซอร์ MC14500 สำหรับการใช้งานได้จริง ๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แต่ถ้า A,B เป็น "0" คำสั่ง 4-7 จะไม่ทำให้เกิดพัลส์ได้ เพราะ OEN ห้ามพัลส์ Write ไว้ บรรทัดที่ 8-9 จะทำให้ OEN เป็น "1" เพื่อทำคำสั่ง 10-11 ได้ บรรทัดที่ 12-13 ทำการเซ็ท RR เป็น 1 เสมอ แล้วอินาเบิ้ลเอาท์พุทไว้เสมอ

4.15 การทดสอบโปรแกรม

โปรแกรมที่ใช้นี้จะใช้การป้อนค่าทีละ 4 บิท ลงในหน่วยความจำแต่ต้องใช้ 2 ตำแหน่ง จึงจะได้ข้อมูล 1 คำสั่ง ลองดูวิธีการป้อนโปรแกรมทดสอบดังโปรแกรมที่ 6

โปรแกรมที่ 6 ตัวอย่างโปรแกรมง่าย ๆ เพื่อแสดงการป้อนโปรแกรม

โปรแกรมแอดเดรส	โหลดค่า HEX	ข้อมูลไบการี	นิวโมนิค
00	A	1010	NOPO
00	0	0000	
01	B	1011	OEN
01	0	0000	
02	1	0001	LD
02	0	0000	
03	8	1000	STO
03	0	0000	
04	F	1111	NOPF
04	0	0000	

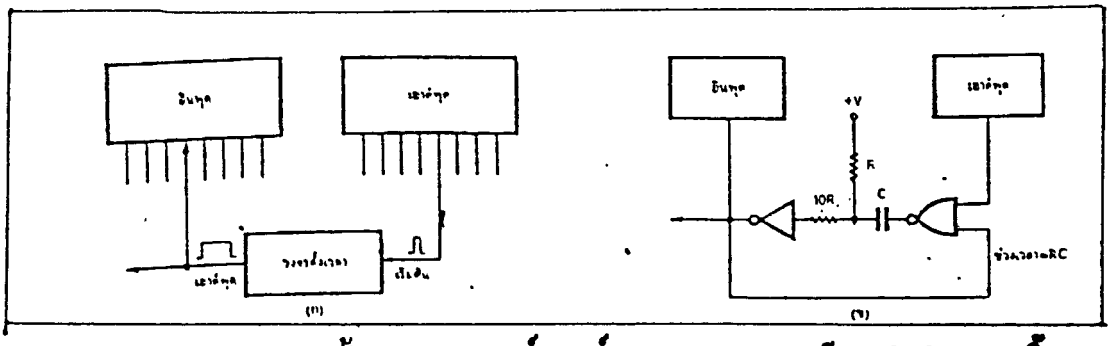
ลองดูโปรแกรมวนการอ่านข้อมูลจากอินพุทบิทที่ 5 แล้วส่งมาให้เอาท์พุทบิทที่ 1 และ 6 ดังนี้

A	IEN	
0		
B	OEN	
0		
1	LD	พอร์ท 5
5		
8	STO	พอร์ท 1
1		
8	STO	พอร์ท 6
6		
F	NOPF	

4.16 การโปรแกรมเกี่ยวกับการตั้งเวลา

ในการควบคุมระบบงานทั่วไป จะต้องเกี่ยวข้องกับระบบเวลาด้วยเสมอ ระบบการตั้งเวลาที่ใช้นี้สามารถทำได้ทั้งทางด้านฮาร์ดแวร์และซอฟต์แวร์ การใช้ซอฟต์แวร์จะต้องใช้โปรแกรมในการเขียนให้ค่าในรีจิสเตอร์เพิ่มขึ้นหรือลดลงจนถึงค่าที่กำหนด แต่สำหรับ MC14500 จะต้องใช้เทคนิคบางประการในการกำหนดรีจิสเตอร์ภายนอก ส่วนทางฮาร์ดแวร์เป็นเรื่องที่ค่อนข้างง่าย โดยใช้บิตทางเอาท์พุทไปหริกวงจรตั้งเวลาและใช้อินพุทตรวจสอบดูสถานะ แผนผังการสร้างวงจรกำหนดเวลา ดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ก. วงจรตั้งเวลาทางฮาร์ดแวร์ และ ข. วงจรซีมอสโมโนสเตเบิ้ล

ในการเขียนโปรแกรมนั้น จะให้โปรแกรมทำงาน โดยการส่งเอาท์พุทไปทริกเท่านั้น เช่น ถ้าต้องการส่งเอาท์พุทไปให้วงจรตั้งเวลาทำงาน โดยมีเงื่อนไข ถ้าอินพุท A,B เป็น "1" ทั้งคู่ เราเขียนโปรแกรมได้ดังโปรแกรมที่ 7

โปรแกรมที่ 7 แสดงการส่งสัญญาณเอาท์พุทไปทริกวงจรตั้งเวลา

```
ONTIMER :  LD      A          ; อ่านค่าสัญญาณ A
           AND     B          ; AND กับสัญญาณ B
           ANDC    INPUT      ; ตรวจสอบดูว่าวงจรตั้งเวลา ON หรือไม่
           STO     OUTPUT     ; ส่งค่าไปเอาต์พุต
```

ในการกำหนดช่วงเวลาของการทำงานด้วยฮาร์ดแวร์นี้ สามารถทำได้ด้วยการทริกดังกล่าวแล้ว ลองพิจารณาดังตัวอย่างต่อไปนี้ ซึ่งเป็นผลที่ได้จากโปรแกรมดังกล่าวมาแล้ว

-การ turn on delay

สมมติว่าต้องการ ON เอาท์พุท U หลังจากที่อินพุท A เป็น "1" เป็นเวลา X วินาที

-การ turn off delay

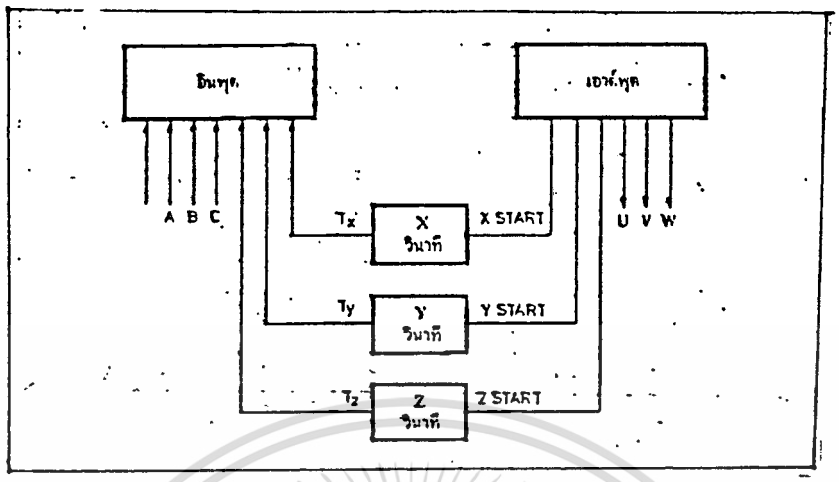
สมมติว่าต้องการ OFF เอาท์พุทหลังจากที่อินพุท B เป็น "0" เป็นเวลา Y วินาที

-การ delay on-delay off

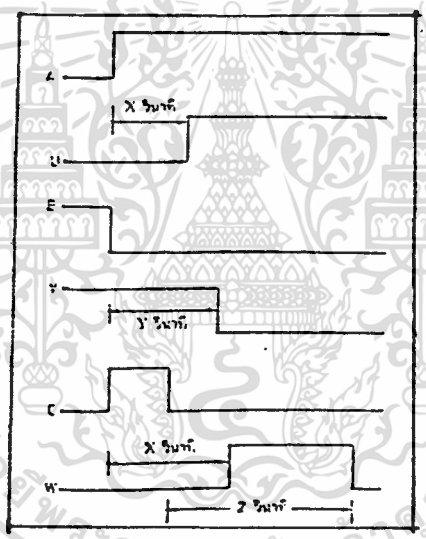
สมมติต้องการ ON เอาท์พุท W หลังจากที่อินพุท C เป็น "1" เป็นเวลา วินาที และ OFF เอาท์พุท W หลังจากที่อินพุท C เป็น "0" Z วินาที

รูปลักษณะของวงจรเป็นไปดังแสดงในรูป และแผนผังเวลาเป็นไปดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง ตัวอย่างของแผนผังการตั้งเวลา turn on-turn off



รูปแสดง แผนผังเวลาที่ต้องการของแผนผังการตั้งเวลา turn on-turn off

เมื่อเขียนเป็นโปรแกรมให้ทำงานรวมกับฮาร์ดแวร์ จะเขียนเป็นโปรแกรมได้ดัง โปรแกรมที่ 8

โปรแกรมที่ 8 แสดงการทำงานของวงจรการตั้งเวลา สำหรับโปรแกรม delay on-delay off เขียนได้ง่าย ๆ โดยนำโปรแกรมทั้ง 2 มารวมเข้าด้วยกัน แล้วเปลี่ยนค่าเวลาเท่านั้น

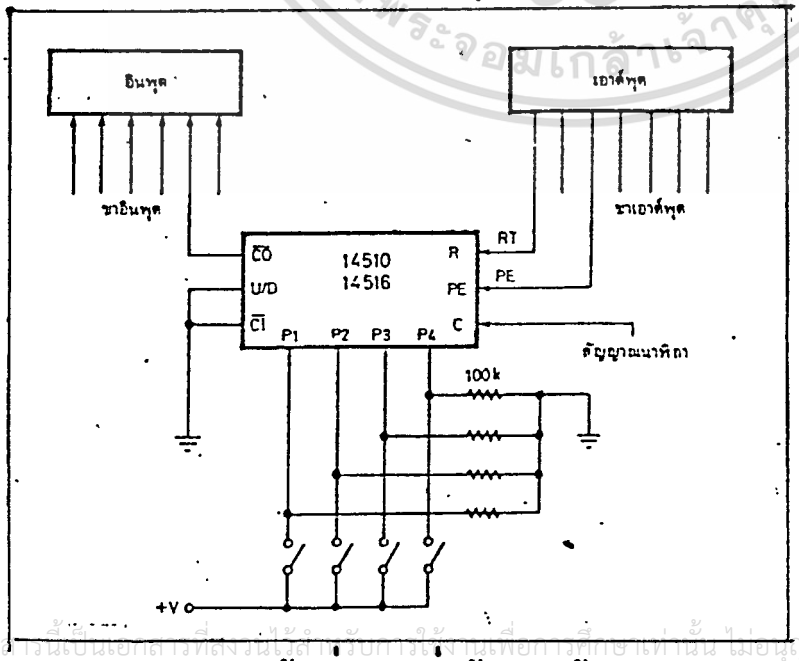
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมที่ 8 แสดงการทำงานของวงจรตั้งเวลา turn on-turn off

การ turn on delay		การ turn off delay	
X Start = $A.\overline{U}.\overline{Tx}$		Y Start = $V.\overline{Ty}$	
Store U = $\overline{Tx}.\overline{U}$		Store V = V.Ty	
สมมติว่า IEN = OEN = 1			
LD	A	LD	V
ANDC	U	AND	B
ANDC	Tx	ANDC	Ty
STO	X Start	STO	Y Start
LDC	Tx	LD	Ty
ANDC	U	AND	V
OEN	R	OEN	R
STO	U	STO	V
XNOR	R	XNOR	R
OEN	R	OEN	R

4.17 การกำหนดช่วงเวลาแบบดิจิตอลที่โปรแกรมได้

การควบคุมช่วงเวลาด้วยไมโครโปรเซสเซอร์ MC14500 นี้ ใช้รูปแบบฮาร์ดแวร์ได้ แต่จะต้องทำร่วมกับซอฟต์แวร์ โดยใช้วงจรเคาว์เตอร์แบบนับเดินหน้าหรือถอยหลังร่วมด้วย ดังแสดงในรูป เป็นตัวอย่างการใช้ซีมอสเคาว์เตอร์ต่อเป็นวงจรตั้งเวลาที่โปรแกรมค่าตามสวิทช์กำหนดค่าเริ่มต้น วงจรเคาว์เตอร์ดังกล่าว ได้รับการจัดวงจรให้เป็นวงจรนับลง (count dow ในสภาวะปกติขารีเซทจะเป็น "1" และเมื่อต้องการให้เริ่มนับก็จะส่งสัญญาณมาที่ขารีเซทให้เป็น "0" คงค้างไว้ พร้อมกับส่งสัญญาณพัลส์มาที่ขา PE เพื่อทำการโหลดค่าเริ่มต้นให้กับวงจรนับ พร้อมเริ่มการนับสัญญาณนาฬิกาเอาท์พุทที่ได้จะได้ออกทางขา $\overline{CO}$ ของวงจรนับ



รูปแสดง ตัวอย่างการต่อกับวงจรนับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การทำงานของโปรแกรมควบคุมจะเริ่มจากสมมติตั้งค่าแฟลกอินพุทที่มีค่าเป็น "1" โดยตั้งชื่อว่า แฟลก T โดยเมื่อนับครบเวลาแล้วจะรีเซท MC14500 เพื่อไม่ให้มีการวนนับต่อไป โปรแกรมที่เขียนขึ้นเป็นไปดังโปรแกรมที่ 9

โปรแกรมที่ 9 แสดงการควบคุมวงจรนับในรูปตัวอย่างการต่อกับวงจรนับ

```
START   LD      T       ; test start flag
        ANDC    END     ; test timer reset
        OEN     RR      ; enable outputs
        STOC    RESET   ; remove timer reset
        STO     LOAD    ; start load pulse
        STOC    LOAD    ; remove load pulse
        STOC    T       ; clear T
        ORC     RR      ; RR gets "1" (RR+RR = 1)
FIN     OEN     RR      ; enable outputs
```

และหากว่าเราจะสมมติให้มีการเริ่มกำหนดการทำงานของวงจรนับนี้ โดยดูจากเงื่อนไข A.B = 1 ผลลัพธ์ของการเขียนโปรแกรมจะเปลี่ยนไป ลองพิจารณาจากโปรแกรมที่ 10

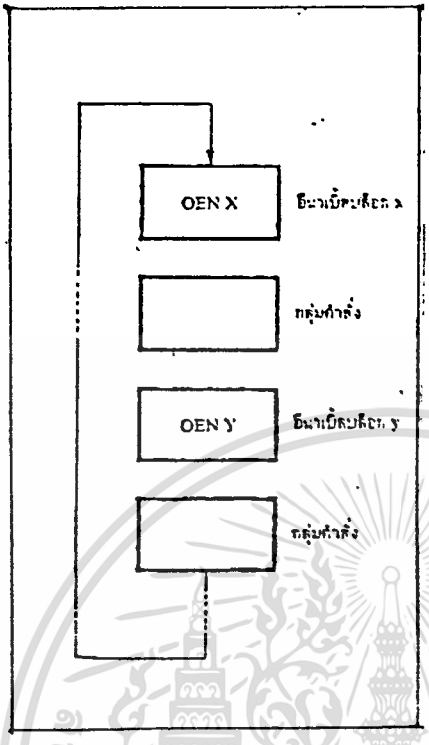
โปรแกรมที่ 10 แสดงการตรวจเงื่อนไข A.B = 1 ก่อนเริ่มการทำงานของวงจรนับ

```
        LD      A       ; load A
        AND     B       ; result is now A.B
        OEN     R       ; output enable if A.B = 1
        STOC    RT      ; remove counter reset
        STO     PE      ; pull PE high
        STOC    PE      ; restore PE low
```

4.18 การเขียนโปรแกรมแบบ IF...THEN

ในการเขียนโปรแกรมแบบทางเลือกจะใช้คำสั่งที่สำคัญช่วยคำสั่งหนึ่งคือ OEN โดยคำสั่งนี้เป็นคำสั่งที่สำคัญมากสำหรับ MC14500 โครงสร้างการทำงานของคำสั่ง OEN ดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง หลักการการใช้คำสั่ง OEN

ความสำคัญของ OEN มี 4 ข้อดังนี้

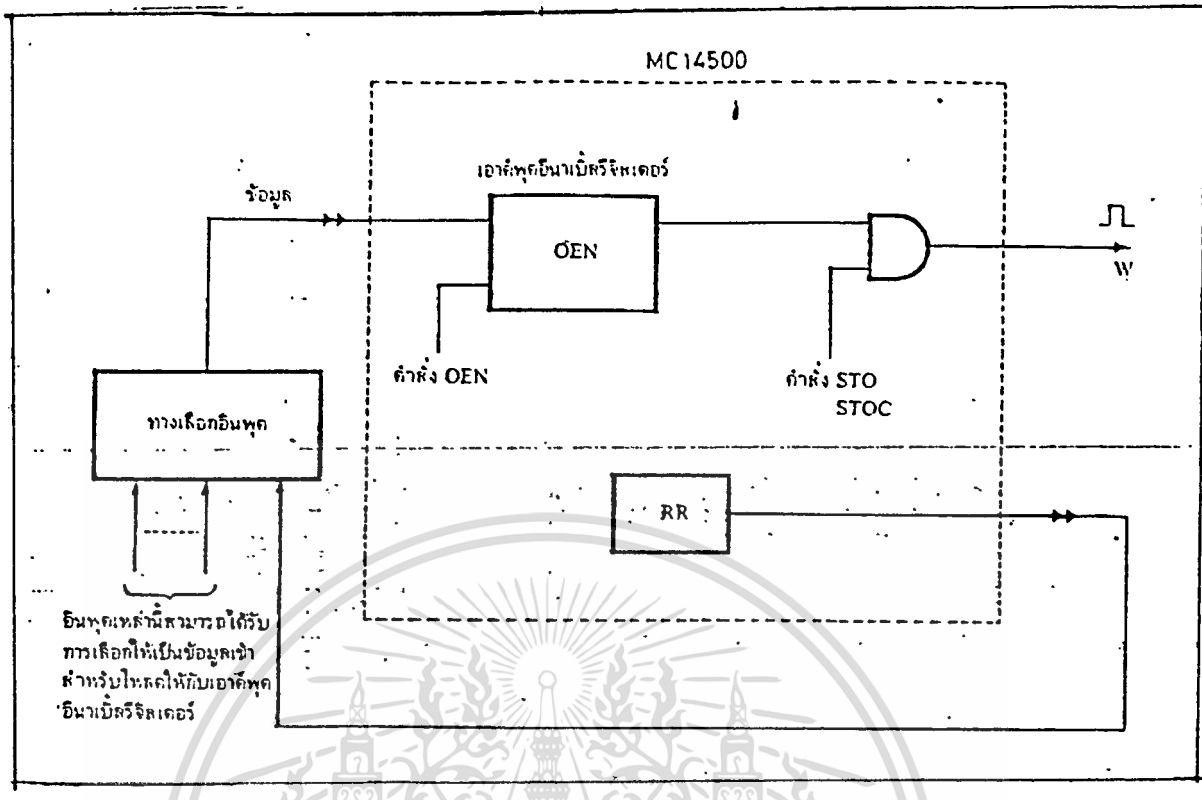
1. ถ้ารีจิสเตอร์ OEN มีค่าเป็น "1" ไมโครโปรเซสเซอร์สามารถเขียนข้อมูลไปที่เอาท์พุท หรือหน่วยความจำได้ แต่ถ้า OEN เป็น "0" สัญญาณพัลส์ WRITE จะไม่มีออกมาจึงไม่สามารถนำไปใช้ได้

2. เราสามารถให้ข้อมูลอินพุท โหลดเข้าโดยตรงไปยังรีจิสเตอร์ OEN และเราก็สามารถใช้รีจิสเตอร์ RR เป็นอินพุทได้

3. การต่อเชื่อมให้ RR ได้รับการกำหนดเสมือนเป็นแอดเดรสหนึ่งของอินพุทจะทำให้เราสามารถเขียนโปรแกรมได้ง่ายและมีประโยชน์

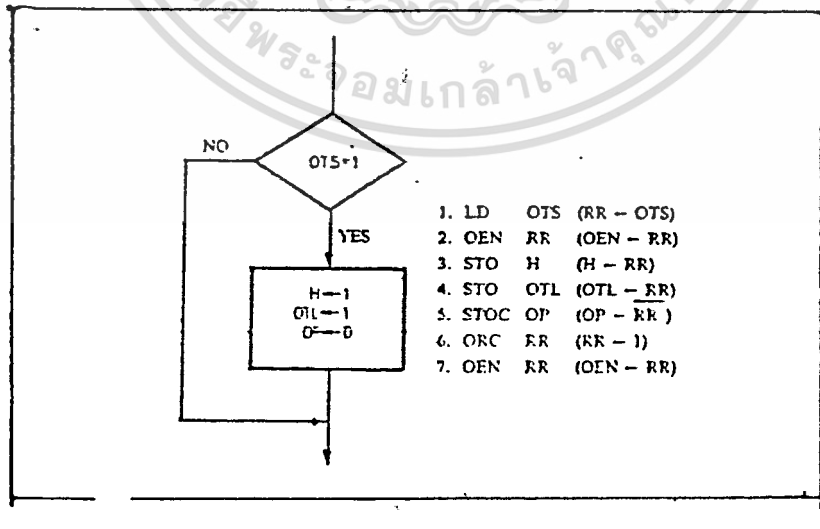
4. เราสามารถเขียนโปรแกรมเป็นบล็อก แบบ if...then โดยใช้ OEN ลองดูจากแผนผังดังแสดงในรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง แผนผังการใช้คำสั่ง OEN

ตัวอย่างที่จะกล่าวต่อไปนี้ เราตั้งข้อสมมติว่า ถ้าอุณหภูมิสูงเกินค่าหนึ่ง จะทำให้สวิทช์ OTS ปิดวงจร (เท่ากับ 1) ให้ทำการส่งสัญญาณไปยังกระดิ่ง (H) ปิดการจ่ายกำลังงานให้เตาอบ (OP) และส่งสัญญาณไปแสดงผลบนหลอดไฟที่เตาอบ (OTL) จากตัวอย่างนี้ มาเขียนผังงานและโปรแกรมได้ดังรูปที่แสดง



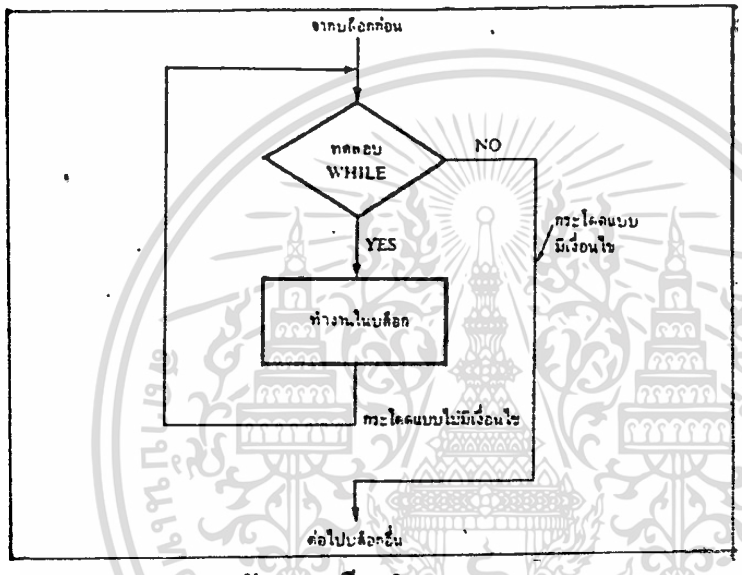
รูปแสดง แผนผังและโปรแกรมของตัวอย่างการใช้ if...then

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.19 การเขียนโปรแกรมแบบ WHILE

โครงสร้างของ WHILE เป็นที่รู้จักกันดีในการเขียนโปรแกรมระดับภาษาสูง สำหรับในการเขียนโปรแกรมของ MC14500 เราก็สามารถใช้โครงสร้าง WHILE ได้เช่นเดียวกัน

โครงสร้าง WHILE ดังแสดงในรูปและนำผังของรูปตามที่แสดงในรูปมาเขียนเป็นโปรแกรมจะได้ดังโปรแกรมที่ 11



รูปแสดง แผนผังการเขียนโปรแกรมแบบ WHILE

โปรแกรมที่ 11 แสดงการทำงานของแผนผังการเขียนโปรแกรมแบบ WHILE

```
        START   LD      FS      ; load float SW signal
                SKZ             ; skip next inst if FS = 0
                JMP     NEXT    ; jump to next
                STO     PUMP    ; turn on pump
                JMP     START   ; jump back to start
        NEXT    STOC    PUMP    ; turn off pump
                        NEXT INSTRUCTION
```

จากพื้นฐานของการเขียนโปรแกรมนี้ จะทำให้เราสามารถเขียนโปรแกรมให้ MC14500 ทำงานให้เราได้เป็นอย่างดี ผู้ใช้งานลองฝึกฝนและทำความเข้าใจกับหลักการโปรแกรมจะพบว่าการเขียนก็ใช้หลักการของภาษาสูงอื่นได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

บทวิจารณ์และสรุป

ก่อนอื่นต้องขอทบทวนระบบของเดิมก่อนซึ่งได้กล่าวถึงระบบเพื่อให้ MC14500 มีอินพุทเอาท์พุทต่อกับภายนอกได้ การเขียนอ่านข้อมูลจากอินพุท-เอาท์พุทจะทำทีละบิท โดยแสดงให้เห็นการกำหนดแอดเดรสของอินพุท-เอาท์พุท ซึ่งใช้ 3 บิท เป็นตัวกำหนด จึงต่อกับอินพุท-เอาท์พุทได้ 8 ช่องทาง

การทำงานใด ๆ จะใช้เอาท์พุทและอินพุทเป็นตัวกำหนด เช่น การควบคุมการเปิดปิดมอเตอร์จะให้สายสัญญาณขาเอาท์พุทเป็นตัวส่งสัญญาณไปทำการ ON -OFF รีเลย์ ในขณะเดียวกันอาจต้องให้มีการตรวจสอบสภาพการทำงานว่าถูกต้องหรือไม่ อย่างไรก็ตามคำสั่งของ MC14500 มีจำกัด แต่เราก็สามารถโปรแกรมให้มีการทำงานต่าง ๆ ได้

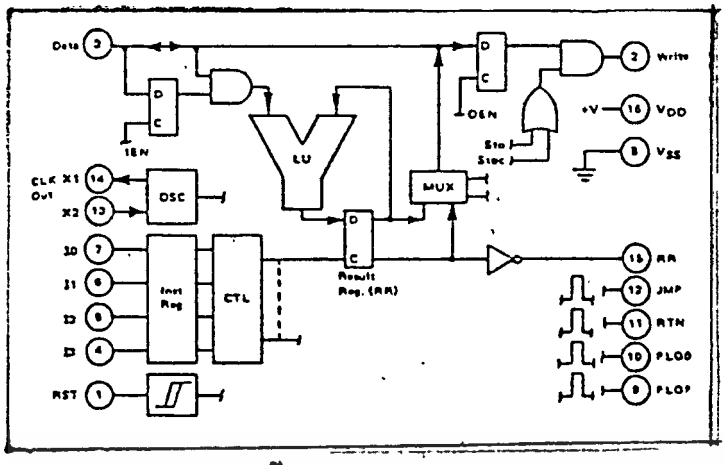
เพื่อจะนำคำสั่งต่าง ๆ มาใช้ ในที่นี้ขอทบทวนชุดคำสั่งทั้ง 16 คำสั่ง ที่จะใช้กับไอซี MC14500 ดังตารางที่ 1

ตารางที่ 1 ชุดคำสั่งทั้ง 16 คำสั่งของไอซี MC14500

รหัสคำสั่ง	นีโมนิค	ความหมาย
0000	NOPO	ค่ารีจิสเตอร์ไม่เปลี่ยน R → R, FLG0 → ⎍
0001	LD	โหลดค่ารีจิสเตอร์ผลลัพธ์ DATA → RR
0010	LDC	โหลดคอมพลีเมนต์ $\overline{DATA}$ → RR
0011	AND	AND ลอจิก RR.D → RR
0100	NNDC	AND ลอจิกคอมพลีเมนต์ RR.$\overline{D}$ → RR
0101	OR	OR ลอจิก RR+D → RR
0110	ORC	OR ลอจิกคอมพลีเมนต์ RR+$\overline{D}$ → RR
0111	XNOR	เอ็กคลูซีฟ NOR ถ้า RR = D, RR → 1
1000	STO	เก็บค่า RR → สายข้อมูล, Write → 1
1001	STOC	เก็บค่าคอมพลีเมนต์ $\overline{RR}$ → สายข้อมูล, Write → 1
1010	IEN	อินพุตอีนาเบิ้ล D → IEN
1011	OEN	เอาต์พุตอีนาเบิ้ล D → OEN
1100	JMP	กระโดด JMP แฟลก → ⎍
1101	RTN	รีเทิร์น RTN แฟลก → ⎍ , ข้ามคำสั่งต่อไป
1110	SKZ	ข้ามคำสั่งต่อไป ถ้า RR = 0
1111	NOPF	ไม่มีการเปลี่ยนค่าในรีจิสเตอร์ RR → RR, FLGF → ⎍

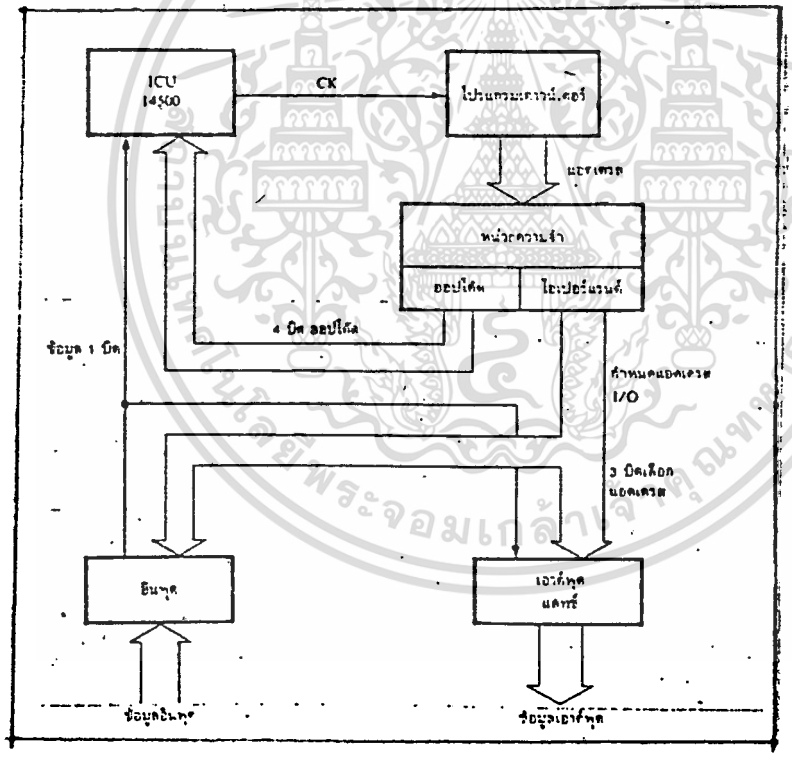
โดยโครงสร้างไอออแกรมของไมโครโปรเซสเซอร์เป็นดังรูปที่แสดง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปแสดง โครงสร้างภายในของไอซี MC14500

ในการทำงานของไอซี MC14500 จะใช้คำสั่ง 4 บิท โดยมีไดอะแกรมของระบบไมโครคอมพิวเตอร์เป็นไปดังแสดงในรูป



รูปแสดง ระบบไมโครคอมพิวเตอร์ไอซี MC14500

จุดมุ่งหมายในการจัดทำชุดฝึก Single Board Microprocessor 1 bit เพื่อต้องการให้เห็นหลักการของไมโครโปรเซสเซอร์แบบพื้นฐานและง่ายที่สุด แต่อย่างไรก็ตาม การใช้ไมโครโปรเซสเซอร์แบบ 1 บิท นี้ ยังมีข้อยุ่งยากกว่าแบบ 8 หรือ 16 บิท จึงทำให้ผู้ใช้ไมโครโปรเซสเซอร์เหล่านั้นในระบบงานมากกว่า เพราะสามารถเขียนโปรแกรมที่สลับซับซ้อนได้ดีกว่า รายงานฉบับนี้จึงเสมือนได้สร้างแนวความคิดให้เห็นหลักการทั่วไป และสร้างความเข้าใจพื้นฐานของฮาร์ดแวร์และซอฟต์แวร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สำหรับโปรเจคนี้ได้ทำการออกแบบและทดลอง สร้างชุดฝึกไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุม ทั้งทางด้านฮาร์ดแวร์และซอฟท์แวร์ ซึ่งประโยชน์โดยตรงก็คือกลุ่มที่ทำโปรเจคนี้ มีความเข้าใจในเนื้อหาเกี่ยวกับ MC14500 ไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุมเป็นอย่างดี ทั้งทางด้านทฤษฎีการออกแบบและการตัดสินใจเลือกใช้อุปกรณ์ที่เหมาะสม หาซื้อได้สะดวกในประเทศเป็นการประหยัดเงินและยังเป็นการพึ่งตัวเองอีกด้วย ตลอดจนการแก้ปัญหาต่าง ๆ ที่เกิดขึ้น นอกจากนี้ยังเป็นแนวทางในการสร้างชุดฝึกไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุม จำนวนมากขึ้นเองภายในวิทยาลัย เพื่อให้นักศึกษาได้ศึกษาและพัฒนาเพิ่มเติมเพื่อออกแบบสำหรับการควบคุมอุปกรณ์บางอย่างโดยเฉพาะ

ชุดฝึกไมโครโปรเซสเซอร์ 1 บิท สำหรับงานควบคุม สามารถสร้างได้เองในวิทยาลัยนับว่าเป็นประโยชน์อย่างมาก และสามารถสร้างขึ้นให้มีจำนวนเพียงพอสำหรับนักศึกษาแผนกอิเล็กทรอนิกส์ ตลอดจนสามารถเผยแพร่วิชาการด้านนี้ไปสู่ระดับอาชีวศึกษาและระดับมัธยมศึกษาต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

MOTOROLA

MC14500B

CMOS LSI

(LOW-POWER COMPLEMENTARY MOS)

INDUSTRIAL CONTROL UNIT

INDUSTRIAL CONTROL UNIT

The MC14500B Industrial Control Unit (ICU) is a single bit CMOS processor. The ICU is designed for use in systems requiring decisions based on successive single bit information. An external ROM stores the control program. With a program counter (and output latches and input multiplexers, if required) the ICU in a system forms a stored program controller that replaces combinatorial logic. Applications include relay logic processing, serial data manipulation and control. The ICU also may control an MPU or be controlled by an MPU.

- 16 Instructions
- DC to 1.0 MHz Operation at V_{DD} = 5 V
- On Chip Clock (Oscillator)
- Executes One Instruction per Clock Cycle
- 3 V to 18 V Operation
- Noise Immunity Typically 45% of V_{DD}
- Quiescent Current 5.0 μAdc Typical at V_{DD} = 5 V
- Capable of Driving One Low-Power Schottky Load or Two Low-Power TTL Loads over Full Temperature Range



L SUFFIX
CERAMIC PACKAGE
CASE 620

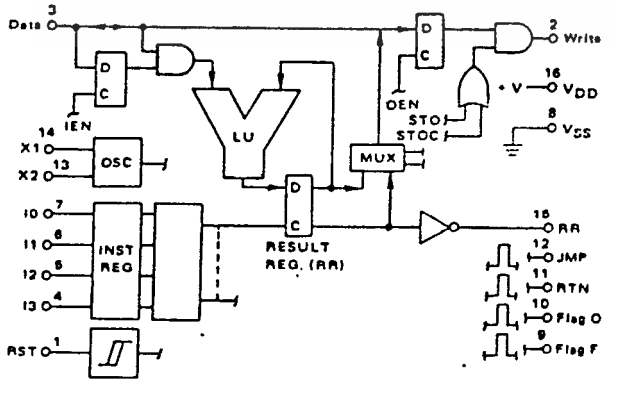
P SUFFIX
PLASTIC PACKAGE
CASE 648

ORDERING INFORMATION

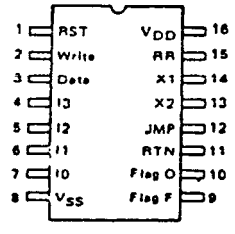
MC14XXXB Suffix Denotes
- L Ceramic Package
- P Plastic Package
- A Extended Operating Temperature Range
- C Limited Operating Temperature Range

Detailed operation and applications are given in the "MC14500B Industrial Control Unit" handbook.

BLOCK DIAGRAM



PIN ASSIGNMENT



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14500B

MAXIMUM RATINGS (Voltages referenced to V_{SS})

Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	-0.5 to +18	Vdc
Input Voltage, All Inputs	V_{in}	-0.5 to V_{DD} + 0.5	Vdc
DC Current Drain per Pin	I	10	mAdc
Operating Temperature Range — AL Device CL/CP Device	T_A	-55 to +125 -40 to +85	°C
Storage Temperature Range	T_{stg}	-65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$.

ELECTRICAL CHARACTERISTICS

Characteristic		Symbol	V_{DD} Vdc	T_{low}* Min	T_{low}* Max	25°C Min	25°C Typ	25°C Max	T_{high}* Min	T_{high}* Max	Unit
Output Voltage $V_{in} = V_{DD}$ or 0	"0" Level	V_{OL}	5.0	–	0.05	–	0	0.05	–	0.05	Vdc
			10	–	0.05	–	0	0.05	–	0.05	
			15	–	0.05	–	0	0.05	–	0.05	
$V_{in} = 0$ or V_{DD}	"1" Level	V_{OH}	5.0	4.95	–	4.95	5.0	–	4.95	–	Vdc
			10	9.95	–	9.95	10	–	9.95	–	
			15	14.95	–	14.95	15	–	14.95	–	
Input Voltage # RST, D, X2	"0 Level"	V_{IL}									Vdc
(V_O = 4.5 or 0.5 Vdc)			5.0	–	1.5	–	2.25	1.5	–	1.5	
(V_O = 9.0 or 1.0 Vdc)			10	–	3.0	–	4.50	3.0	–	3.0	
(V_O = 13.5 or 1.5 Vdc)			15	–	4.0	–	6.75	4.0	–	4.0	
	"1" Level	V_{IH}									Vdc
(V_O = 0.5 or 4.5 Vdc)			5.0	3.5	–	3.5	2.75	–	3.5	–	
(V_O = 1.0 or 9.0 Vdc)			10	7.0	–	7.0	5.50	–	7.0	–	
(V_O = 1.5 or 13.5 Vdc)			15	11.0	–	11.0	8.25	–	11.0	–	
Input Voltage # I0, I1, I2, I3	"0" Level	V_{IL}									Vdc
(V_O = 4.5 or 0.5 Vdc)			5.0	–	0.8	–	1.1	0.8	–	0.8	
(V_O = 9.0 or 1.0 Vdc)			10	–	1.6	–	2.2	1.6	–	1.6	
(V_O = 13.5 or 1.5 Vdc)			15	–	2.4	–	3.4	2.4	–	2.4	
	"1" Level	V_{IH}									Vdc
(V_O = 0.5 or 4.5 Vdc)			5.0	2.0	–	2.0	1.9	–	2.0	–	
(V_O = 1.0 or 9.0 Vdc)			10	6.0	–	6.0	3.1	–	6.0	–	
(V_O = 1.5 or 13.5 Vdc)			15	10	–	10	4.3	–	10	–	
Output Drive Current Data, Write (AL/CL/CP Device)	Source	I_{OH}									mAdc
(V_{OH} = 4.6 Vdc)			5.0	-1.0	–	-1.0	-2.0	–	-1.0	–	
(V_{OH} = 9.5 Vdc)			10	–	–	–	-6.0	–	–	–	
(V_{OH} = 13.5 Vdc)			15	–	–	–	-12	–	–	–	
(V_{OL} = 0.4 Vdc)	Sink	I_{OL}	5.0	1.6	–	1.6	3.2	–	1.6	–	mAdc
(V_{OL} = 0.5 Vdc)			10	–	–	–	6.0	–	–	–	
(V_{OL} = 1.5 Vdc)			15	–	–	–	12	–	–	–	
Output Drive Current Other Outputs (AL Device)	Source	I_{OH}									mAdc
(V_{OH} = 2.5 Vdc)			5.0	-3.0	–	-2.4	-4.2	–	-1.7	–	
(V_{OH} = 4.6 Vdc)			5.0	-0.64	–	-0.51	-0.88	–	-0.36	–	
(V_{OH} = 9.5 Vdc)			10	-1.6	–	-1.3	-2.25	–	-0.9	–	
(V_{OH} = 13.5 Vdc)			15	-4.2	–	-3.4	8.8	–	-2.4	–	
(V_{OL} = 0.4 Vdc)	Sink	I_{OL}	5.0	0.64	–	0.51	0.88	–	0.36	–	mAdc
(V_{OL} = 0.5 Vdc)			10	1.6	–	1.3	2.25	–	0.9	–	
(V_{OL} = 1.5 Vdc)			15	4.2	–	3.4	8.8	–	2.4	–	
Output Drive Current Other Outputs (CL/CP Device)	Source	I_{OH}									mAdc
(V_{OH} = 2.5 Vdc)			5.0	-2.5	–	-2.1	-4.2	–	-1.7	–	
(V_{OH} = 4.6 Vdc)			5.0	-0.52	–	-0.44	-0.88	–	-0.36	–	
(V_{OH} = 9.5 Vdc)			10	-1.3	–	-1.1	-2.25	–	-0.9	–	
(V_{OH} = 13.5 Vdc)			15	-3.6	–	-3.0	-8.8	–	-2.4	–	
(V_{OL} = 0.4 Vdc)	Sink	I_{OL}	5.0	0.52	–	0.44	0.88	–	0.36	–	mAdc
(V_{OL} = 0.5 Vdc)			10	1.3	–	1.1	2.25	–	0.9	–	
(V_{OL} = 1.5 Vdc)			15	3.6	–	3.0	8.8	–	2.4	–	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14500B

ELECTRICAL CHARACTERISTICS (continued)

Characteristic	Symbol	V_{DD} Vdc	T_{low}* Min	T_{low}* Max	25°C Min	25°C Typ	25°C Max	T_{high}* Min	T_{high}* Max	Unit
Input Current, RST (AL/CL/CP Device)	I_{in}	15	25	–	–	150	–	–	250	μAdc
Input Current (AL Device)	I_{in}	15	–	±0.1	–	±0.00001	±0.1	–	±1.0	μAdc
Input Current (CL/CP Device)	I_{in}	15	–	±0.3	–	±0.00001	±0.3	–	±1.0	μAdc
Input Capacitance (Data)	C_{in}	–	–	–	–	15	–	–	–	pF
Input Capacitance (All Other Inputs) (V_{in} = 0)	C_{in}	–	–	–	–	5.0	7.5	–	–	pF
Quiescent Current (AL Device) (Per Package)	I_{DD}	5.0 10 15	– – –	5.0 10 20	– – –	0.005 0.010 0.015	5.0 10 20	– – –	150 300 600	μAdc
Quiescent Current (CL/CP Device) (Per Package)	I_{DD}	5.0 10 15	– – –	20 40 80	– – –	0.005 0.010 0.015	20 40 80	– – –	150 300 600	μAdc
**Total Supply Current at an External Load Capacitance (C_L) on All Outputs	I_T	–	$I_T = (1.5\ \mu A/kHz)\ f + I_{DD}$ $I_T = (3.0\ \mu A/kHz)\ f + I_{DD}$ $I_T = (4.5\ \mu A/kHz)\ f + I_{DD}$							μAdc

* T_{low} = -55°C for AL Device, -40°C for CL/CP Device.
T_{high} = +125°C for AL Device, +85°C for CL/CP Device.
** The formulas given are for the typical characteristics only at 25°C.
# Noise immunity specified for worst-case input combination.

SWITCHING CHARACTERISTICS (T_A = 25°C, $t_r = t_f$ = 20 ns for X and I inputs, C_L = 50 pF for JMP, X1, RR, Flag O, Flag F; C_L = 130 pF + 1TTL load for Data and Write.)

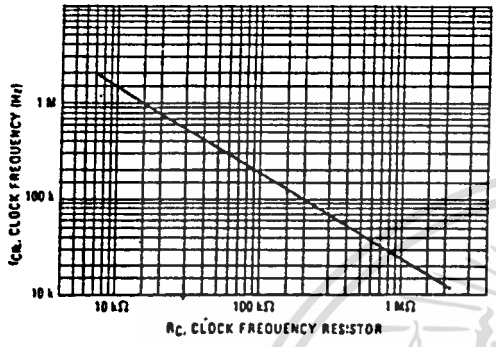
Characteristic	Symbol	V_{DD} Vdc	All Types Min	All Types Typ	All Types Max	Unit
Propagation Delay Time X1 to RR	t_{PLH}, t_{PHL}	5.0	–	250	500	ns
		10	–	125	250	
		15	–	100	200	
X1 to Flag F, Flag O, RTN, JMP		5.0	–	200	400	ns
		10	–	100	200	
		15	–	85	170	
X1 to Write		5.0	–	225	450	ns
		10	–	125	250	
		15	–	100	200	
X1 to Data		5.0	–	250	500	ns
		10	–	120	240	
		15	–	100	200	
RST to RR		5.0	–	250	500	ns
		10	–	125	250	
		15	–	100	200	
RST to X1		5.0	–	450	Note 1	ns
		10	–	200		
		15	–	150		
RST to Flag F, Flag O, RTN, JMP		5.0	–	400	800	ns
		10	–	200	400	
		15	–	150	300	
RST to Write, Data		5.0	–	450	900	ns
		10	–	225	450	
		15	–	175	350	
Clock Pulse Width, X1	$t_{W(cl)}$	5.0	400	200	–	ns
		10	200	100	–	
		15	180	90	–	
Reset Pulse Width, RST	$t_{W(R)}$	5.0	500	250	–	ns
		10	250	125	–	
		15	200	100	–	
Setup Time — Instruction	$t_{su(I)}$	5.0	400	200	–	ns
		10	250	125	–	
		15	180	90	–	
Data	$t_{su(D)}$	5.0	200	100	–	ns
		10	100	50	–	
		15	80	40	–	
Hold Time — Instruction	$t_{h(I)}$	5.0	100	0	–	ns
		10	50	0	–	
		15	50	0	–	
Data	$t_{h(D)}$	5.0	200	100	–	ns
		10	100	50	–	
		15	100	50	–	

NOTE 1. Maximum Reset Delay may extend to one-half clock period.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14500B

FIGURE 1 – TYPICAL CLOCK FREQUENCY versus RESISTOR (R_C)

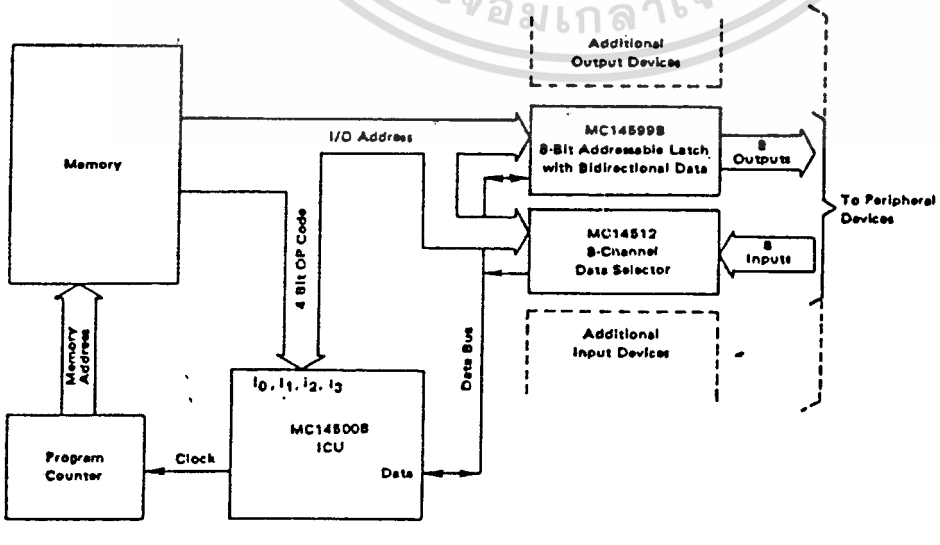


Pin No.	Function	Symbols
1	Chip Reset	RST
2	Write Pulse	Write
3	Data In/Out	Data
4	MSB Instruction Word	I_3
5	Bit 2 Instruction Word	I_2
6	Bit 1 Instruction Word	I_1
7	LSB Instruction Word	I_0
8	Negative Supply (Ground)	V_{SS}
9	Flag on NOP F	Flag F
10	Flag on NOP 0	Flag 0
11	Subroutine Return Flag	RTN
12	Jump Instruction Flag	JMP
13	Oscillator Input	X2
14	Oscillator Output	X1
15	Result Register	RR
16	Positive Supply	V_{DD}

TABLE 1. MC14500B INSTRUCTION SET

Instruction Code	Mnemonic	Action
0 0000	NOP0	No change in registers. RR → RR, Flag 0 → ⎍
1 0001	LD	Load result register. Data → RR
2 0010	LDC	Load complement. $\overline{Data}$ → RR
3 0011	AND	Logical AND. RR·Data → RR
4 0100	ANDC	Logical AND complement. RR·$\overline{Data}$ → RR
5 0101	OR	Logical OR. RR + Data → RR
6 0110	ORC	Logical OR complement. RR + $\overline{Data}$ → RR
7 0111	XNOR	Exclusive NOR. If RR = Data, RR → 1
8 1000	STO	Store. RR → Data Pin, Write → ⎍
9 1001	STOC	Store complement. $\overline{RR}$ → Data Pin, Write → ⎍
A 1010	IEN	Input enable. Data → IEN Register
B 1011	OEN	Output enable. Data → OEN Register
C 1100	JMP	Jump. JMP Flag → ⎍
D 1101	RTN	Return. RTN Flag → ⎍ and skip next instruction
E 1110	SKZ	Skip next instruction if RR = 0
F 1111	NOPF	No change in registers. RR → RR, Flag F → ⎍

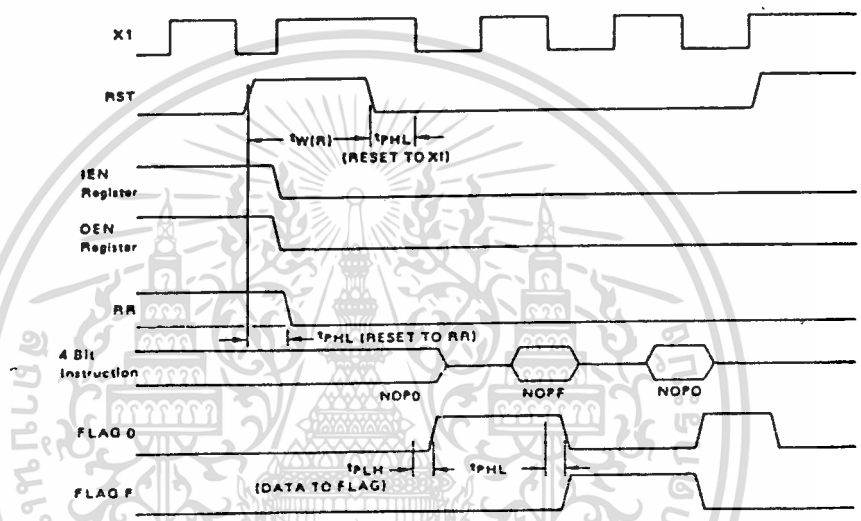
FIGURE 2 – OUTLINE OF A TYPICAL ORGANIZATION FOR A MC14500B-BASED SYSTEM



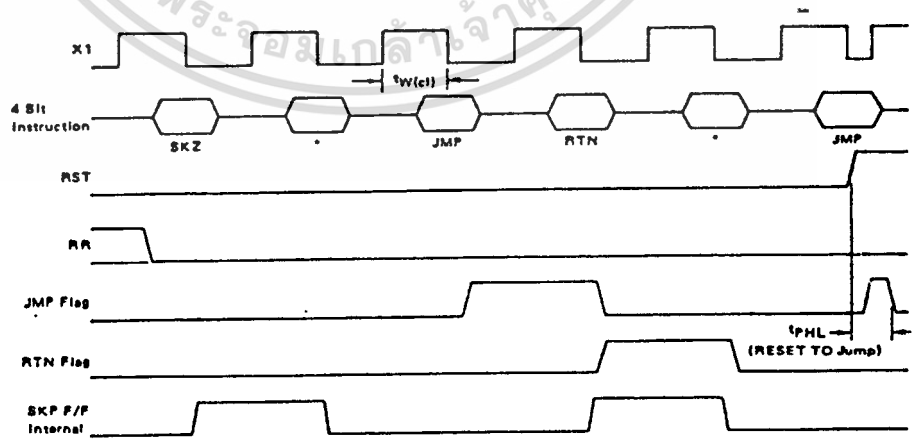
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TIMING WAVEFORMS

Instructions NOPO, NOPF
RR, IEN, OEN remain unaffected



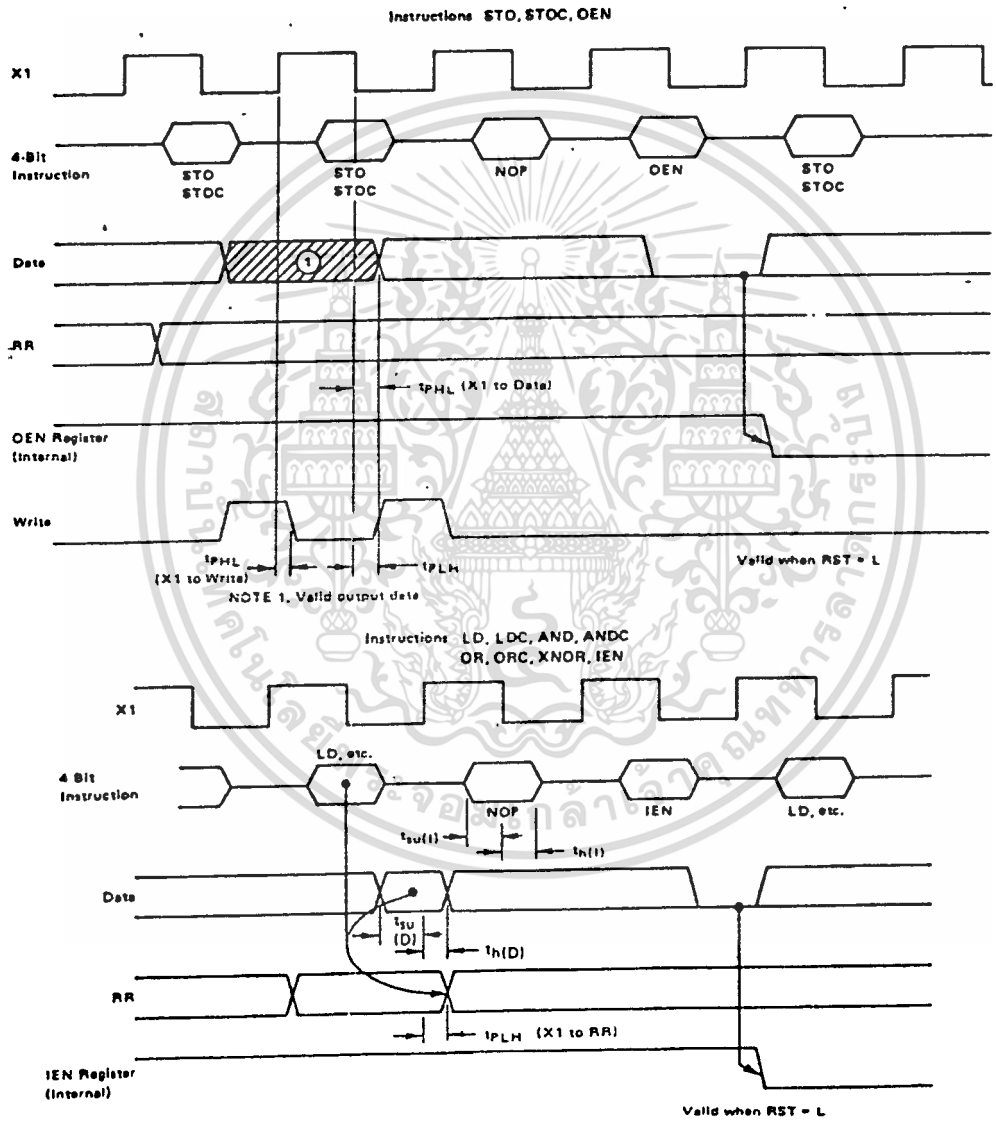
Instructions SKZ, JMP, RTN
RR, IEN, OEN remain unaffected



*Instructions Ignored.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TIMING WAVEFORMS



Circuit diagrams utilizing Motorola products are included as a means of illustrating typical semiconductor applications; consequently, complete information sufficient for construction purposes is not necessarily given. The information has been carefully checked and is believed to be entirely reliable. However, no responsibility is assumed for inaccuracies. Furthermore, such information does not convey to the purchaser of the semiconductor devices described any license under the patent rights of Motorola Inc. or others.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MOTOROLA

MC14495

CMOS MSI

(LOW-POWER COMPLEMENTARY MOS)

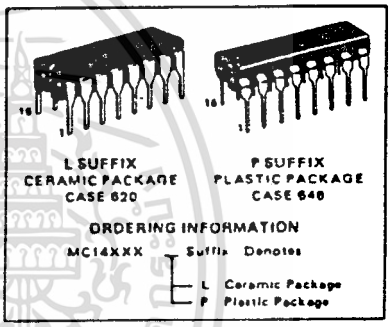
BCD-TO-SEVEN SEGMENT HEXADECIMAL LATCH/DECODER/DRIVER

BCD-TO-SEVEN SEGMENT HEXADECIMAL LATCH/ DECODER/DRIVER

The MC14495 BCD-to-seven segment hexidecimal latch/decoder/driver is constructed with complementary MOS (CMOS) enhancement mode devices and NPN bipolar output drivers in a single monolithic structure. The circuit provides the functions of a 4-bit storage latch. It can be used with LED seven segment displays without resistor interface at 5 volt supply. The resistors of typically 290 ohms are internal to the part.

Applications include MPU systems display driver, instrument display driver, computer/calculator display driver, cockpit display driver, and various clock, watch, and timer uses.

- Low Logic Circuit Power Dissipation
- High-Current Sourcing Outputs With Internal Limiting Resistance
- Latch Storage of Code
- Supply Voltage Range = 4.5 Vdc to 16 Vdc
- Internal Input Level Shift:
  Input +5 CMOS to V_{DD} of +5 to +16 Vdc
  Input +5 V TTL with Pull-up, to V_{DD} of +5 to +16 Vdc



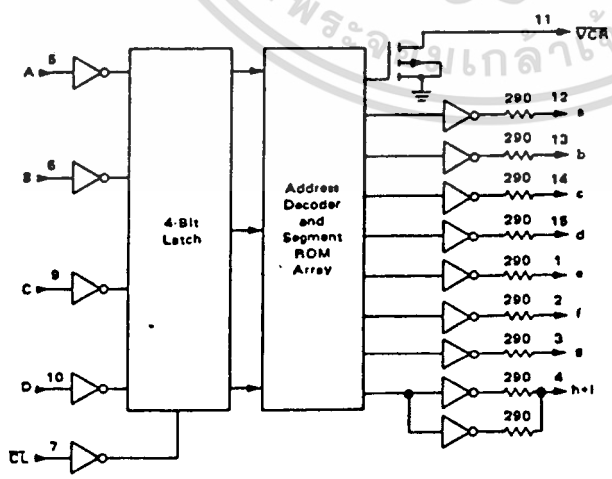
L SUFFIX
CERAMIC PACKAGE
CASE 620

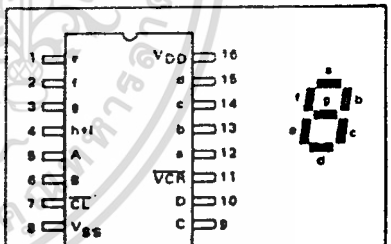
P SUFFIX
PLASTIC PACKAGE
CASE 648

ORDERING INFORMATION

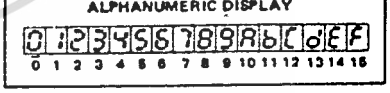
MC14XXX — Suffix Denotes
- L Ceramic Package
- P Plastic Package

BLOCK DIAGRAM





ALPHANUMERIC DISPLAY



TRUTH TABLE

INPUTS				OUTPUTS								
D	C	B	A	a	b	c	d	e	f	g	h•l	VCR	DISPLAY
0	0	0	0	1	1	1	1	1	1	0	0	Open	0
0	0	0	1	0	1	1	0	0	0	0	0	Open	1
0	0	1	0	1	1	0	1	1	0	1	0	Open	2
0	0	1	1	1	1	1	1	0	0	1	0	Open	3
0	1	0	0	0	1	1	0	0	1	1	0	Open	4
0	1	0	1	1	0	1	1	0	1	1	0	Open	5
0	1	1	0	1	0	1	1	1	1	1	0	Open	6
0	1	1	1	1	1	1	0	0	0	0	0	Open	7
1	0	0	0	1	1	1	1	1	1	1	0	Open	8
1	0	0	1	1	1	1	1	0	1	1	0	Open	9
1	0	1	0	1	1	1	0	1	1	1	1	Open	A
1	0	1	1	0	0	1	1	1	1	1	1	Open	b
1	1	0	0	1	0	0	1	1	1	0	1	Open	C
1	1	0	1	0	1	1	1	1	0	1	1	Open	d
1	1	1	0	1	0	0	1	1	1	1	1	Open	E
1	1	1	1	1	0	0	0	1	1	1	1	0	F

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14495

MAXIMUM RATINGS (Voltages referenced to V_{SS}).

Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	-0.5 to +18	Vdc
Input Voltage, All Inputs	V_{in}	-0.5 to V_{DD} + 0.5	Vdc
DC Current Drain per Input Pin	I	10	mAdc
Operating Temperature Range	T_A	-40 to +85	°C
Storage Temperature Range	T_{stg}	-65 to +150	°C
Maximum Continuous Output Power (Source) per Output @ 25 °C Pins 1, 2, 3, 12, 13, 14, 15 Pin 14	P_{OHmax}	 50 100	mW

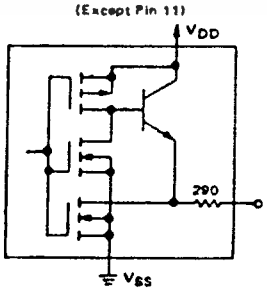
$^{\$}P_{OHmax} = I_{OH}(V_{DD} - V_{OH})$

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$.

ELECTRICAL CHARACTERISTICS (All voltages referenced to V_{SS} = 0, T_A -25°C)

Characteristics	Symbol	Condition	Min	Typ	Max	Unit
DC Supply Voltage	V_{DD}		4.5		16	Vdc
Input Voltage	V_{IL}		–		0.8	Vdc
	V_{IH}	V_{DD} = 15V V_{DD} = 5.0V	4.0 3.5		– –	Vdc
Input Current	I_{in}		–		±10	µAdc
Output VCR, Pin 11 Open Drain Output	I_{OH}	$V_{OH} = V_{DD}$	–		±10	µAdc
	I_{OL}	V_{OL} = 0.5V, V_{DD} = 5.0V V_{OL} = 0.5V, V_{DD} = 15V	0.2 1.0		– –	mA
Outputs a, b, c, d, e, f, g	I_{OH}	V_{OH} = 2.0V, V_{DD} = 5.0V V_{OH} = 1.5V, V_{DD} = 5.0V V_{OH} = 12V, V_{DD} = 15V V_{OH} = 11.5V, V_{DD} = 15V	-7.5 – -7.5 –		 -11.5 -11.5	mA
	I_{OL}	V_{OL} = 1.0V, V_{DD} = 5.0V V_{OL} = 1.0V, V_{DD} = 15V	0.1 0.6		– –	mA
Output h + I	I_{OH}	V_{OH} = 2.0V, V_{DD} = 5.0V V_{OH} = 1.5V, V_{DD} = 5.0V V_{OH} = 12V, V_{DD} = 15V V_{OH} = 11.5V, V_{DD} = 15V	-15 – -15 –		– -23 – -23	mA
	I_{OL}	V_{OL} = 1V, V_{DD} = 5.0 V V_{OL} = 1.0 V, V_{DD} = 15V	0.2 1.0		– –	mA

OUTPUT CIRCUIT
(Except Pin 11)



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14495

INPUT/OUTPUT FUNCTIONS

Segment Driver (a, b, c, d, e, f, g, h, i; Pins 1, 2, 3, 4, 12, 13, 14, 15)

The segment drivers are emitter-follower NPN transistors. To limit the output current, a resistor typically 290 ohms is integrated internally at each output. Therefore, external resistors are not necessary when driving an LED at the supply voltage of V_{DD} = 5.0 volts.

OUTPUT ($\overline{VCR}$; Pin 11)

This output is activated (goes to low) whenever the address corresponding to program 16 is selected. Otherwise the output is open. See the truth table.

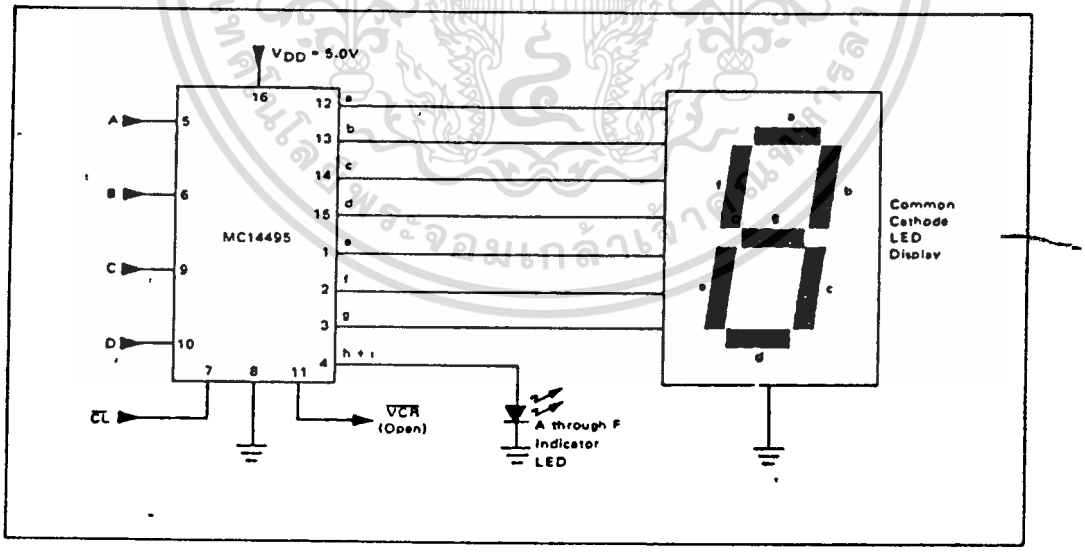
INPUT LATCH (A, B, C, D; Pins 5, 6, 9, 10)

The block diagram is shown on page 1. The inputs A, B, C, and D are fed to a 4-bit latch which is controlled by clock ($\overline{CL}$). Two modes of operation are available.

CLOCK ($\overline{CL}$; Pin 7)

The data on the inputs A, B, C and D will pass through the latch and will be displayed immediately when the clock is low. In this mode of operation the circuit is performing the function of a conventional decoder/driver. The data may be loaded into the latch when $\overline{CL}$ = low and will be latched with the rising edge of $\overline{CL}$. The data will remain stored as long as $\overline{CL}$ is high.

TYPICAL CIRCUIT @ V_{DD} = 5.0V



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MOTOROLA

MC14175B

CMOS SSI

(LOW-POWER COMPLEMENTARY MOS)

QUAD TYPE D FLIP-FLOP

QUAD TYPE D FLIP-FLOP

The MC14175B quad type D flip-flop is constructed with MOS P-channel and N-channel enhancement mode devices in a single monolithic structure. Each of the four flip-flops is positive-edge triggered by a common clock input (C). An active-low reset input ($\overline{R}$) asynchronously resets all flip-flops. Each flip-flop has independent Data (D) inputs and complementary outputs (Q and $\overline{Q}$). These devices may be used as shift register elements or as type T flip-flops for counter and toggle applications.

- Complementary Outputs
- Static Operation
- All Inputs and Outputs Buffered
- Diode Protection on All Inputs
- Supply Voltage Range = 3.0 Vdc to 18 Vdc
- Noise Immunity = 45% of V_{DD} typical
- Output Compatible with Two HTL Loads, Two Low-Power TTL Loads or One Low-Power Schottky TTL Load
- Functional Equivalent to TTL 74175



L SUFFIX
CERAMIC PACKAGE
CASE 620

P SUFFIX
PLASTIC PACKAGE
CASE 648

ORDERING INFORMATION

MC14XXXB — Suffix Denotes
- L Ceramic Package
- P Plastic Package
- A Extended Operating Temperature Range
- C Limited Operating Temperature Range

MAXIMUM RATINGS (Voltages referenced to V_{SS})

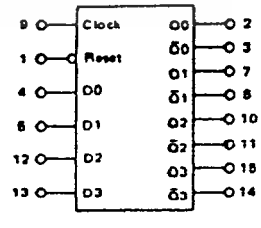
Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	-0.5 to +18	Vdc
Input Voltage, All Inputs	V_{in}	-0.5 to V_{DD} + 0.5	Vdc
DC Current Drain per Pin	I	10	mAdc
Operating Temperature Range — AL Device CL/CP Device	T_A	-55 to +125 -40 to +85	°C
Storage Temperature Range	T_{stg}	-65 to +150	°C

TRUTH TABLE
(Positive Logic)

INPUTS			OUTPUTS		
Clock	Data	$\overline{Reset}$	Q	$\overline{Q}$	
↑	0	1	0	1	
↑	1	1	1	0	
↓	X	1	Q	$\overline{Q}$	No Change
X	X	0	0	1	

X = Don't Care

BLOCK DIAGRAM



V_{DD} = Pin 16
V_{SS} = Pin 8

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$.

Unused inputs must always be tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14175B

ELECTRICAL CHARACTERISTICS

Characteristic	Symbol	V_{DD} Vdc	T_{low}* Min	T_{low}* Max	25°C Min	25°C Typ	25°C Max	T_{high}* Min	T_{high}* Max	Unit
Output Voltage "0" Level V_{in} = V_{DD} or 0	V_{OL}	5.0	–	0.05	–	0	0.05	–	0.05	Vdc
		10	–	0.05	–	0	0.05	–	0.05	
		15	–	0.05	–	0	0.05	–	0.05	
"1" Level V_{in} = 0 or V_{DD}	V_{OH}	5.0	4.95	–	4.95	5.0	–	4.95	–	Vdc
		10	9.95	–	9.95	10	–	9.95	–	
		15	14.95	–	14.95	15	–	14.95	–	
Input Voltage# "0" Level	V_{IL}									Vdc
(V_O = 4.5 or 0.5 Vdc)		5.0	–	1.5	–	2.25	1.5	–	1.5	
(V_O = 9.0 or 1.0 Vdc)		10	–	3.0	–	4.50	3.0	–	3.0	
(V_O = 13.5 or 1.5 Vdc)		15	–	4.0	–	6.75	4.0	–	4.0	
"1" Level	V_{IH}									Vdc
(V_O = 0.5 or 4.5 Vdc)		5.0	3.5	–	3.5	2.75	–	3.5	–	
(V_O = 1.0 or 9.0 Vdc)		10	7.0	–	7.0	5.50	–	7.0	–	
(V_O = 1.5 or 13.5 Vdc)		15	11.0	–	11.0	8.25	–	11.0	–	
Output Drive Current (AL Device)	I_{OH}									mAdc
(V_{OH} = 2.5 Vdc) Source		5.0	-3.0	–	-2.4	-4.2	–	-1.7	–	
(V_{OH} = 4.6 Vdc)		5.0	-0.64	–	-0.51	-0.88	–	-0.36	–	
(V_{OH} = 9.5 Vdc)		10	-1.6	–	-1.3	-2.25	–	-0.9	–	
(V_{OH} = 13.5 Vdc)		15	-4.2	–	-3.4	-8.8	–	-2.4	–	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.64	–	0.51	0.88	–	0.36	–	mAdc
(V_{OL} = 0.5 Vdc)		10	1.6	–	1.3	2.25	–	0.9	–	
(V_{OL} = 1.5 Vdc)		15	4.2	–	3.4	8.8	–	2.4	–	
Output Drive Current (CL/CP Device)	I_{OH}									mAdc
(V_{OH} = 2.5 Vdc) Source		5.0	-2.5	–	-2.1	-4.2	–	-1.7	–	
(V_{OH} = 4.6 Vdc)		5.0	-0.52	–	-0.44	-0.88	–	-0.36	–	
(V_{OH} = 9.5 Vdc)		10	-1.3	–	-1.1	-2.25	–	-0.9	–	
(V_{OH} = 13.5 Vdc)		15	-3.6	–	-3.0	-8.8	–	-2.4	–	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.52	–	0.44	0.88	–	0.36	–	mAdc
(V_{OL} = 0.5 Vdc)		10	1.3	–	1.1	2.25	–	0.9	–	
(V_{OL} = 1.5 Vdc)		15	3.6	–	3.0	8.8	–	2.4	–	
Input Current (AL Device)	I_{in}	15	–	±0.1	–	±0.00001	±0.1	–	±1.0	μAdc
Input Current (CL/CP Device)	I_{in}	15	–	±0.3	–	±0.00001	±0.3	–	±1.0	μAdc
Input Capacitance (V_{in} = 0)	C_{in}	–	–	–	–	5.0	7.5	–	–	pF
Quiescent Current (AL Device) (Per Package)	I_{DD}	5.0	–	5.0	–	0.005	5.0	–	150	μAdc
		10	–	10	–	0.010	10	–	300	
		15	–	20	–	0.015	20	–	600	
Quiescent Current (CL/CP Device) (Per Package)	I_{DD}	5.0	–	20	–	0.005	20	–	150	μAdc
		10	–	40	–	0.010	40	–	300	
		15	–	80	–	0.015	80	–	600	
Total Supply Current**† (Dynamic plus Quiescent, Per Package) (C_L = 50 pF on all outputs, all buffers switching)	I_T	5.0	I_T = (1.7 μA/kHz) f + I_{DD}							μAdc
		10	I_T = (3.4 μA/kHz) f + I_{DD}							
		15	I_T = (5.0 μA/kHz) f + I_{DD}							

*T_{low} = -55°C for AL Device, -40°C for CL/CP Device
T_{high} = +125°C for AL Device, +85°C for CL/CP Device
#Noise immunity specified for worst-case input combination.
Noise Margin for both "1" and "0" level = 1.0 Vdc min @ V_{DD} = 5.0 Vdc
2.0 Vdc min @ V_{DD} = 10 Vdc
2.5 Vdc min @ V_{DD} = 15 Vdc
†To calculate total supply current at loads other than 50 pF:
$I_T(C_L) = I_T(50\ pF) + 4 \times 10^{-3} (C_L - 50) V_{DD} f$
where: I_T is in μA (per package), C_L in pF, V_{DD} in Vdc, and f in kHz is input frequency.
**The formulas given are for the typical characteristics only at 25°C.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14175B

SWITCHING CHARACTERISTICS* (C_L = 50 pF, T_A = 25°C)

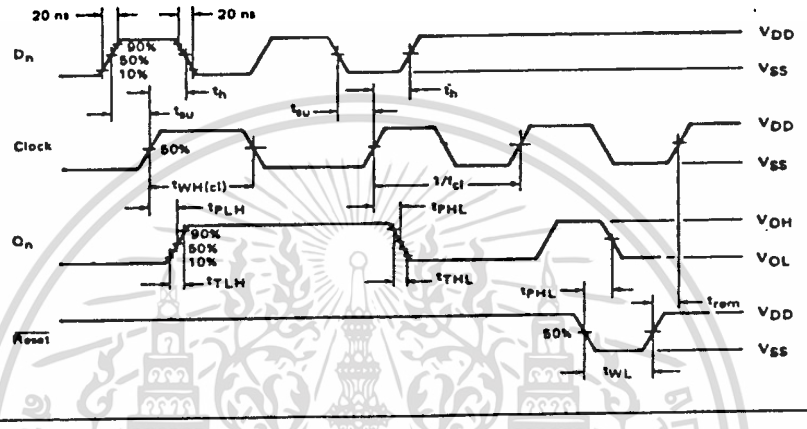
Characteristic	Symbol	V_{DD} Vdc	All Types			Unit
			Min	Typ	Max	
Output Rise and Fall Time t_{TLH}, t_{THL} = (1.35 ns/pF) C_L + 32 ns t_{TLH}, t_{THL} = (0.6 ns/pF) C_L + 20 ns t_{TLH}, t_{THL} = (0.4 ns/pF) C_L + 20 ns	t_{TLH}, t_{THL}	5.0 10 15	– – –	100 50 40	200 100 80	ns
Propagation Delay Time – Clock to Q t_{PLH}, t_{PHL} = (0.9 ns/pF) C_L + 175 ns t_{PLH}, t_{PHL} = (0.36 ns/pF) C_L + 72 ns t_{PLH}, t_{PHL} = (0.26 ns/pF) C_L + 57 ns	t_{PLH}, t_{PHL}	5.0 10 15	– – –	220 90 70	400 180 120	ns
Propagation Delay Time – $\overline{\text{Reset}}$ to Q t_{PHL} = (0.9 ns/pF) C_L + 280 ns t_{PHL} = (0.36 ns/pF) C_L + 112 ns t_{PHL} = (0.26 ns/pF) C_L + 87 ns	t_{PHL}	5.0 10 15	– – –	325 130 100	500 200 150	ns
Clock Pulse Width	t_{WH}	5.0 10 15	250 100 75	110 45 35	– – –	ns
$\overline{\text{Reset}}$ Pulse Width	t_{WL}	5.0 10 15	200 80 60	100 40 30	– – –	ns
Clock Pulse Frequency	f_{cl}	5.0 10 15	– – –	4.5 11 14	2.0 5.0 6.5	MHz
Clock Pulse Rise and Fall Time	t_{TLH}, t_{THL}	5.0 10 15	– – –	– – –	15 5.0 4.0	μs
Data Setup Time	t_{su}	5.0 10 15	120 50 40	60 25 20	– – –	ns
Data Hold Time	t_h	5.0 10 15	80 40 30	40 20 15	– – –	ns
$\overline{\text{Reset}}$ Removal Time**	t_{rem}	5.0 10 15	250 100 80	125 50 40	– – –	ns

*The formulas given are for the typical characteristics only.
**The reset signal must be high prior to a positive-going transition of the clock.

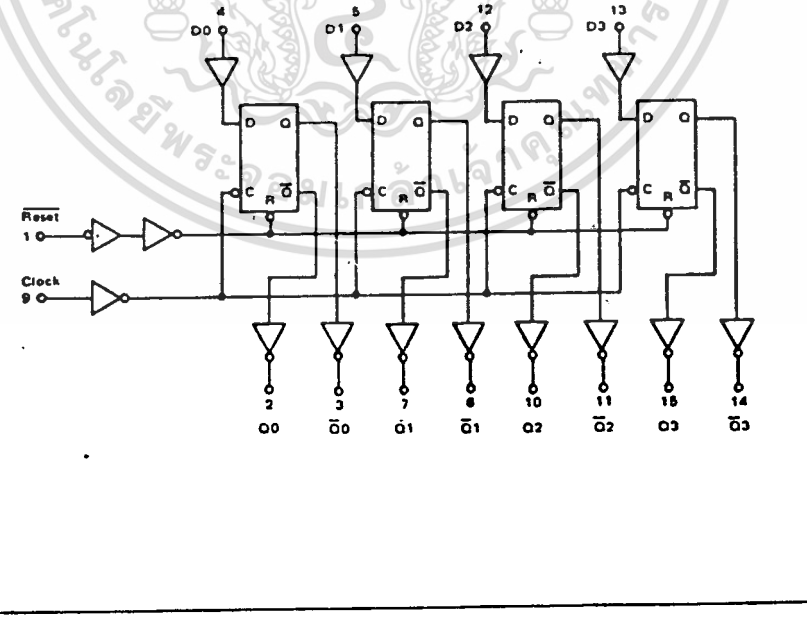
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14175B

TIMING DIAGRAM



FUNCTIONAL BLOCK DIAGRAM



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MOTOROLA

MC14419

2-OF-8 KEYPAD-TO-BINARY ENCODER

The MC14419 is designed for phone dialer system applications, but finds many applications as a keypad-to-binary encoder. The device contains a 2-of-8 to binary encoder, a strobe generator, and an illegal state detector. The encoder has four row inputs and four column inputs, and is designed to accept inputs from 16 keyswitches arranged in a 4 x 4 matrix. For an output on the four data lines, one and only one row along with one and only one column input line must be activated. All other combinations are suppressed by the illegal state detector to eliminate false data output.

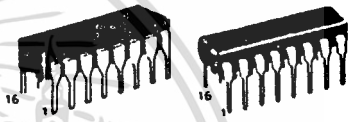
The strobe generator produces a strobe pulse when any of the 10 keys corresponding to numerals 0 through 9 are depressed. The strobe output can be used to eliminate erroneous data entry due to contact bounce. For a strobe output to occur, the key row and column input lines must remain stable for 80 clock pulses after activation. When the contact bounce has settled and 80 clock pulses have occurred, the output will be a single strobe pulse equal in width to that of the clock low state. The strobe generator will output one and only one pulse each time a numerical key is depressed. After the pulse has occurred, noise and bounce due to contact break will not cause another strobe pulse. With a 16 kHz input clock frequency, the pulse occurs 5 ms after the last bounce.

- Suppressed Output for Illegal Input Codes
- On-Chip Pullup Resistors for Row and Column Inputs
- Clock Input Conditioning Circuit
- Low Current Drain in Standby Mode
  5.0 μA Typical @ 5.0 Vdc
- Subsystem Complement to the MC14408/14409 Phone Pulse Converter
- Codes for Numbers 0-9 Produce a Strobe Pulse

CMOS MSI

(LOW-POWER COMPLEMENTARY MOS)

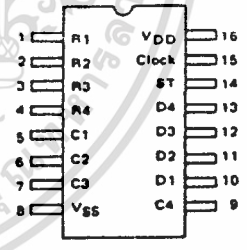
2-OF-8 KEYPAD-TO-BINARY ENCODER



L SUFFIX
CERAMIC PACKAGE
CASE 620

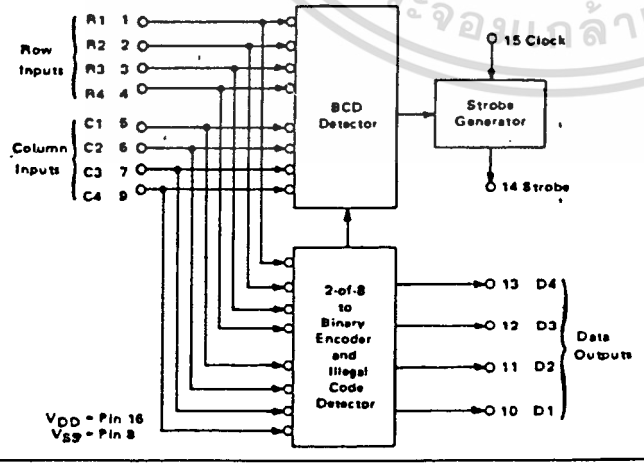
P SUFFIX
PLASTIC PACKAGE
CASE 648

PIN ASSIGNMENT



V_{DD} = Pin 16
V_{SS} = Pin 8

BLOCK DIAGRAM



This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14419

MAXIMUM RATINGS (Voltages referenced to V_{SS}, Pin 8.)

Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	+6.0 to -0.5	Vdc
Input Voltage, All Inputs	V_{in}	V_{DD} + 0.5 to V_{SS} -0.5	Vdc
DC Current Drain per Pin	I	10	mAdc
Operating Temperature Range	T_A	-40 to +85	°C
Storage Temperature Range	T_{stg}	-65 to +150	°C

ELECTRICAL CHARACTERISTICS

Characteristic	Symbol	V_{DD} Vdc	-40°C Min	-40°C Max	25°C Min	25°C Typ	25°C Max	+85°C Min	+85°C Max	Unit
Supply Voltage Operating Range	V_{DD}	–	3.0	6.0	3.0	5.0	6.0	3.0	6.0	Vdc
Output Voltage "0" Level	V_{out}	5.0	–	0.05	–	0	0.05	–	0.05	Vdc
"1" Level		5.0	4.95	–	4.95	5.0	–	4.95	–	Vdc
Input Voltage "0" Level (V_O = 4.5 or 0.5 Vdc)	V_{IL}	5.0	1.5	–	1.5	2.25	–	1.4	–	Vdc
"1" Level (V_O = 0.5 or 4.5 Vdc)	V_{IH}	5.0	3.5	–	3.5	2.25	–	3.6	–	Vdc
Output Drive Current (V_{OH} = 2.5 Vdc) Source	I_{OH}	5.0	-0.23	–	-0.20	-1.7	–	-0.16	–	mAdc
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.23	–	0.20	0.78	–	0.16	–	mAdc
Input Leakage Current (V_{in} = V_{DD})	I_{IH}	5.0	–	±0.1	–	±0.00001	±0.1	–	±1.0	μAdc
Pullup Resistor Source Current (Row and Column Inputs) (V_{in} = V_{SS})	I_{IL}	5.0	265	460	190	250	330	125	215	μAdc
Input Capacitance (V_{in} = V_{SS})	C_{in}	–	–	–	–	5.0	–	–	–	pF
Standby Supply Current (f_{clock} = 16 kHz, No Keys Depressed)	I_{DDS}	3.0	–	3.0	–	1.0	3.0	–	6.0	μAdc
		5.0	–	15	–	5.0	15	–	30	
		6.0	–	60	–	20	60	–	120	
Standby Supply Current as a Function of Clock Frequency* (No Keys Depressed)	I_{DDS}	5.0	I_{DDS} = 0.09 μA/kHz + 3.0 μA							μAdc

*The formula given is for the typical characteristics only.

SWITCHING CHARACTERISTICS (C_L = 50 pF, T_A = 25°C)

Characteristic	Symbol	V_{DD}	Min	Typ	Max	Unit
Output Rise and Fall Times, D1 thru D4 (Figure 1)	t_{TLH}, t_{THL}	5.0	–	300	–	ns
Propagation Delay Time, Row or Column Input to Data Output (Figure 1)	t_{PLH}, t_{PHL}	5.0	–	1000	–	ns
Clock Pulse Frequency Range	PRF	3.0 to 6.0	–	–	80	kHz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MC14419

FIGURE 1 – SWITCHING TIME WAVEFORMS

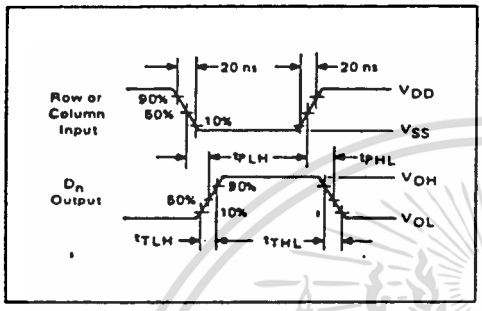
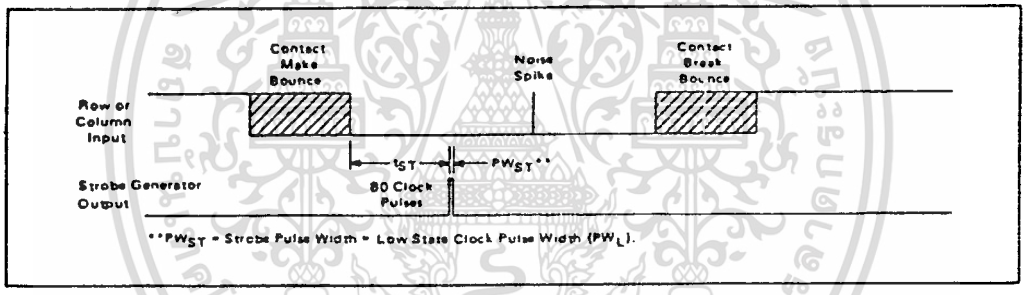


FIGURE 2 – TYPICAL STROBE PULSE DELAY TIMES

PRF Clock Frequency kHz	tST* Strobe Pulse Delay Time ms
4.0	20
8.0	10
16	5.0
32	2.5
80	1.0

*t_{ST} = (1/PRF) • 80, with PRF in kHz, t_{ST} in ms.

FIGURE 3 – STROBE GENERATOR TIMING DIAGRAM



**PW_{ST} = Strobe Pulse Width = Low State Clock Pulse Width (PW_L).

TRUTH TABLE

	Inputs								Outputs				
	Row				Column								
Key**	R4	R3	R2	R1	C4	C3	C2	C1	D4	D3	D2	D1	Strobe
1	1	1	1	0	1	1	1	0	0	0	0	1	⎍
2	1	1	1	0	1	1	0	1	0	0	1	0	⎍
3	1	1	1	0	1	0	1	1	0	0	1	1	⎍
A	1	1	1	0	0	1	1	1	1	1	0	0	0
4	1	1	0	1	1	1	1	0	0	1	0	0	⎍
5	1	1	0	1	1	1	0	1	0	1	0	1	⎍
6	1	1	0	1	1	0	1	1	0	1	1	0	⎍
B	1	1	0	1	0	1	1	1	1	1	0	1	0
7	1	0	1	1	1	1	1	0	0	1	1	1	⎍
8	1	0	1	1	1	1	0	1	1	0	0	0	⎍
9	1	0	1	1	1	0	1	1	1	0	0	1	⎍
C	1	0	1	1	0	1	1	1	1	1	1	0	0
*	0	1	1	1	1	1	1	0	1	0	1	0	0
0	0	1	1	1	1	1	0	1	0	0	0	0	⎍
#	0	1	1	1	1	0	1	1	1	0	1	1	0
D	0	1	1	1	0	1	1	1	1	1	1	1	0
	All Other Combinations								0	0	0	0	0

**See Figure 4 for keypad designation.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

FIGURE 4 – TYPICAL KEYPAD INTERFACE APPLICATION

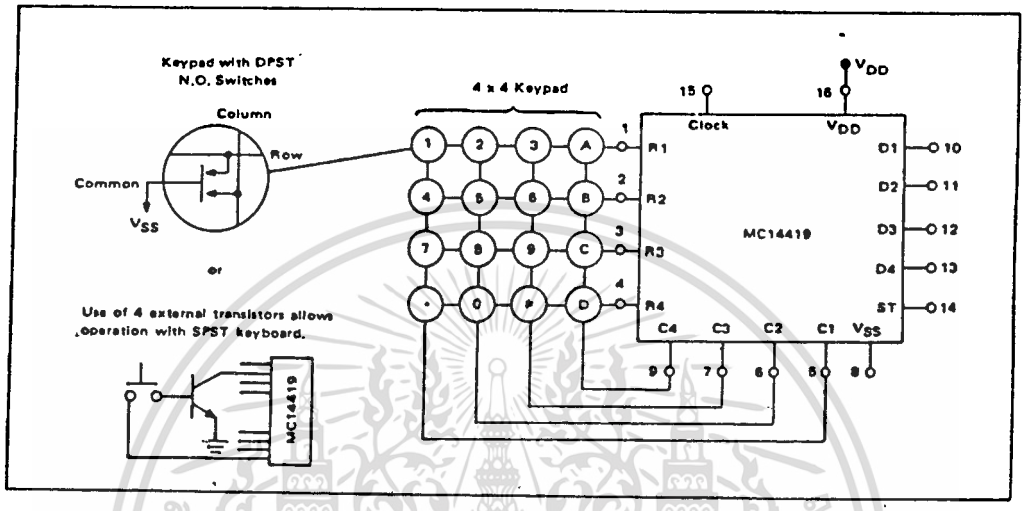
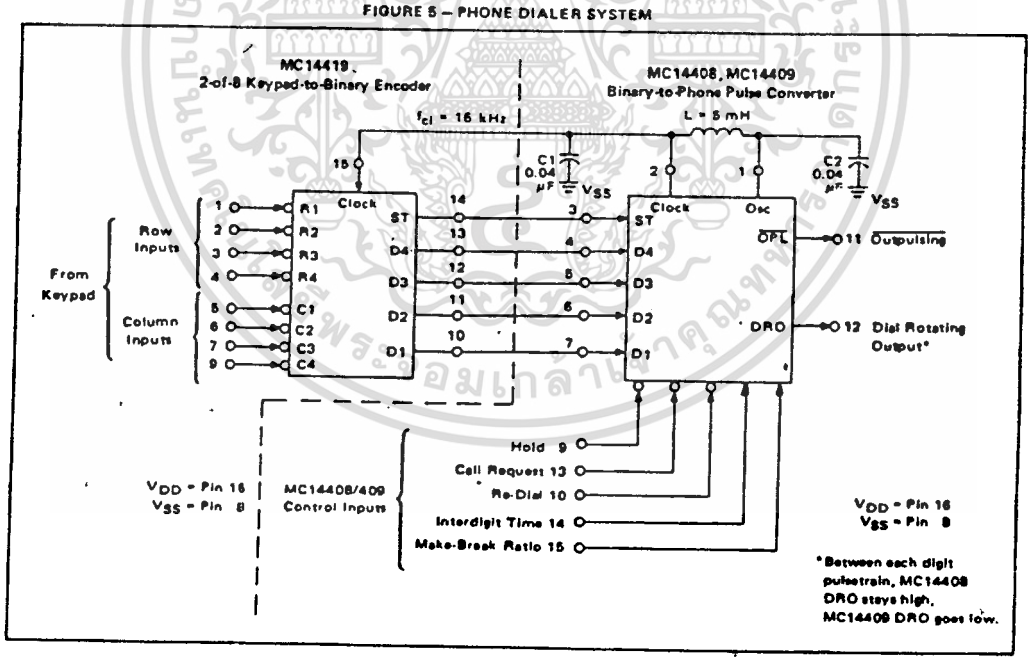


FIGURE 5 – PHONE DIALER SYSTEM



Circuit diagrams utilizing Motorola products are included as a means of illustrating typical semiconductor applications; consequently, complete information sufficient for construction purposes is not necessarily given. The information has been carefully checked and is believed to be entirely reliable. However, no responsibility is assumed for inaccuracies. Furthermore, such information does not convey to the purchaser of the semiconductor devices described any license under the patent rights of Motorola Inc. or others.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SCL4516B

CMOS BINARY UP/DOWN COUNTER

FEATURES

- Internally Synchronous for High Speed
- Asynchronous Preset Enable
- Asynchronous Reset
- Logic Edge-Clocked Design
- 6MHz Counting Rate @ 10Vdc
- Carry Output for Cascading Stages

DESCRIPTION

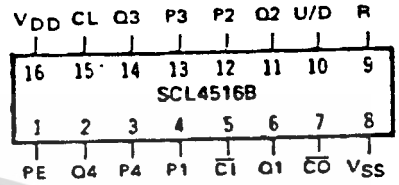
The SCL4516B consists of a four-stage Up/Down Counter with provisions for "look-ahead" carry in both counting modes. The inputs consist of a single Clock, Carry-in (Clock Enable), Reset, Up/Down, Preset Enable, and four individual Jam signals. Four separate buffered Q signals and a Carry-out signal are provided as outputs.

A high Preset Enable signal allows information on the Jam inputs to preset the counter to any state asynchronously with the Clock. A high on the Reset line resets all stages to the "zero" state. The counter is advanced one count at the positive transition of the Clock when the Carry-in and Preset Enable signals are low. Advancement is inhibited when the Carry-in or Preset Enable signals are high. The Carry-out signal is normally high and goes low when the counter reaches its maximum count in the Up mode or the minimum count in the Down mode, provided the Carry-in signal is low. The Carry-in signal in the low state can thus be considered a "Clock Enable." The Carry-in terminal must be connected to V_{SS} when not in use.

The counter counts Up when the Up/Down input is high, and Down when the Up/Down input is low. Multiple packages can be connected in either a parallel-clocking or a ripple-clocking arrangement. Parallel clocking provides synchronous control and hence faster response from all counting outputs. Ripple-clocking allows for longer clock input rise and fall times.

This counter finds primary use in up/down and differential counting and frequency synthesizer applications. It is also useful in A/D and D/A conversion and for magnitude and sign generation.

CONNECTION DIAGRAM
(all packages)



Add suffix for package:

C	16-pin Cerdip
D	16-pin Ceramic
E	16-pin Epoxy
F	16-pin Flat
H	Chip

RECOMMENDED OPERATING CONDITIONS

For maximum reliability:

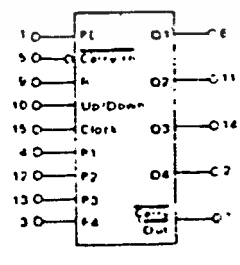
DC Supply Voltage	$V_{DD} - V_{SS}$	3 to 15	Vdc
Operating Temperature	T_A		
C, D, F, H Device		-55 to +125	°C
E Device		-40 to +85	°C

TRUTH TABLE

CARRY IN	UP/DOWN	PRESET ENABLE	RESET	ACTION
1	X	0	0	No Count
0	1	0	0	Count Up
0	0	0	0	Count Down
X	X	1	0	Preset
X	X	X	1	Reset

X = Don't Care

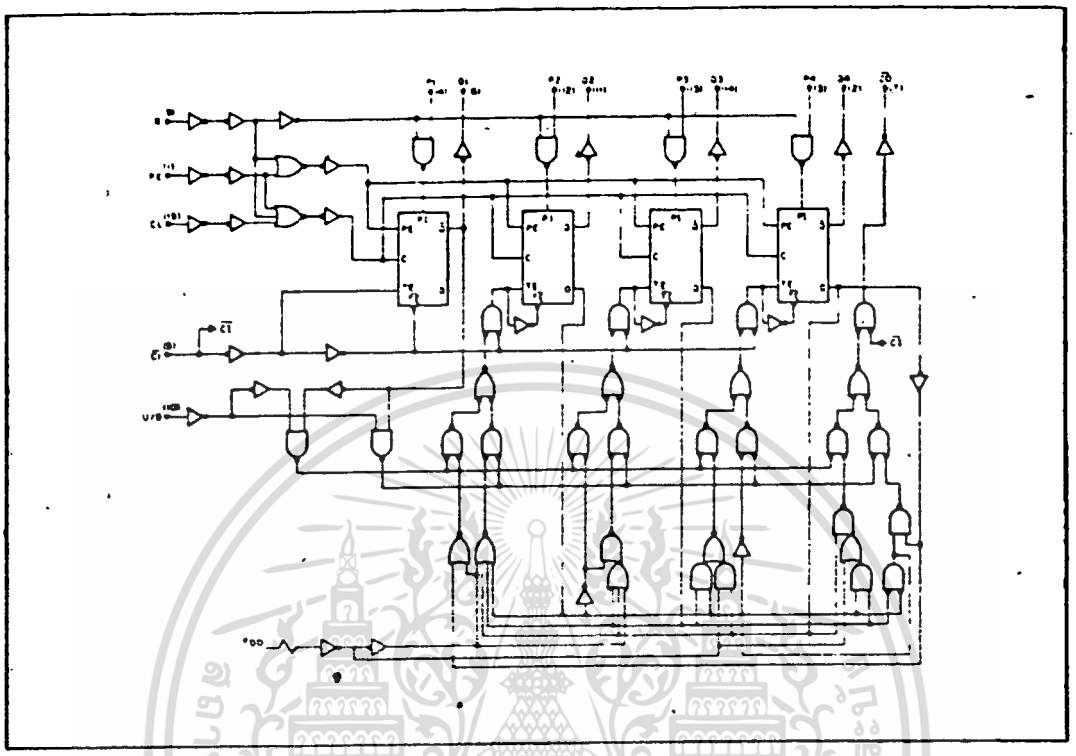
BLOCK DIAGRAM



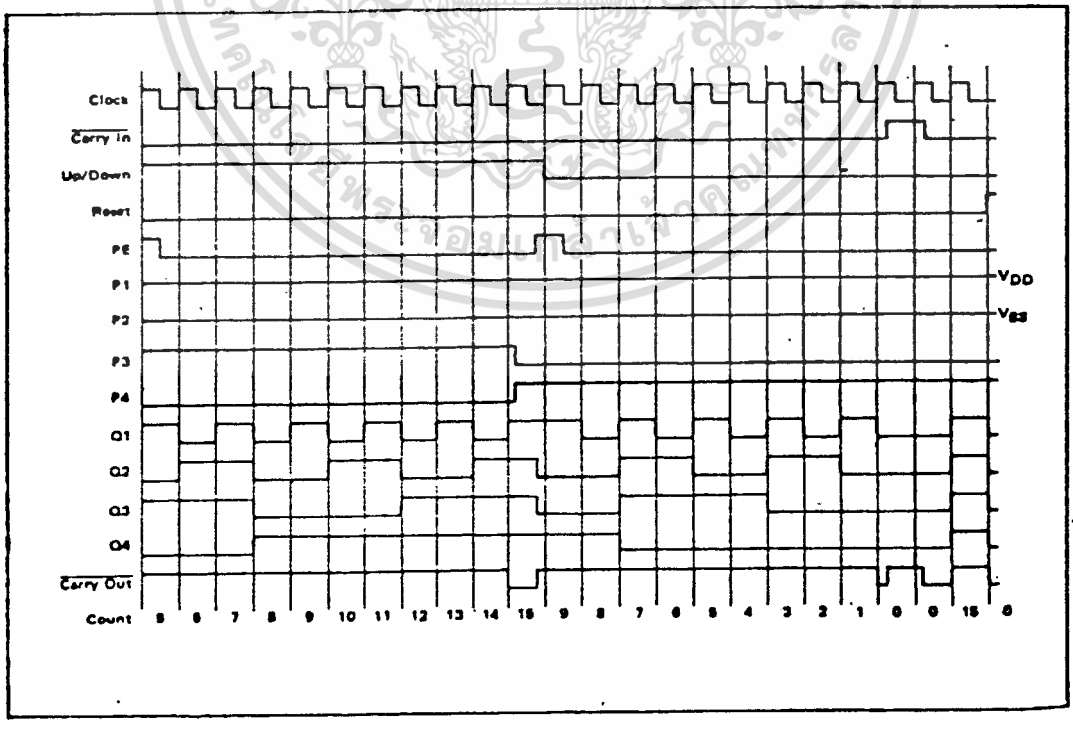
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SCL4516B

LOGIC DIAGRAM



TIMING DIAGRAM



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SCL4516B

ELECTRICAL CHARACTERISTICS

STATIC CHARACTERISTICS[1]

PARAMETER		V_{DD} (Vdc)	CONDITIONS	T_{LOW}[2]		+25°C			T_{HIGH}[2]		Units
				Min.	Max.	Min.	Typ.	Max.	Min.	Max.	
QUIESCENT DEVICE CURRENT	I_{DD}	5	$V_{IN} = V_{SS}$ or V_{DD}	–	5	–	0.05	5	–	150	µAdc
		10	All valid input	–	10	–	0.1	10	–	300	
		15	combinations	–	20	–	0.2	20	–	600	

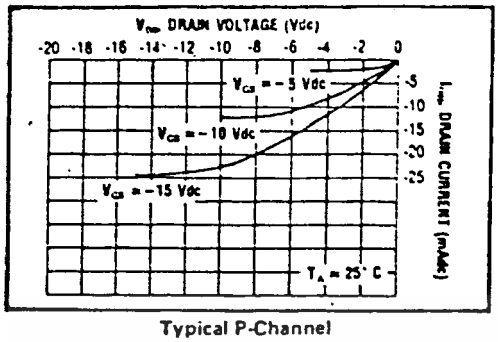
NOTES: [1] Remaining Static Electrical Characteristics are listed under "SCL4000B Series Family Specifications".
[2] T_{LOW} = -55°C for C, D, F, H device.
= -40°C for E device.
T_{HIGH} = +125°C for C, D, F, H device.
= + 85°C for E device.

DYNAMIC CHARACTERISTICS (C_L = 50pF, T_A = 25°C)

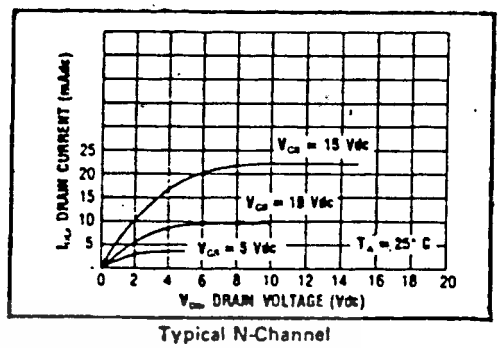
PARAMETER		V_{DD} (Vdc)	Min.	Typ.	Max.	Units
CLOCKED OPERATION						
PROPAGATION DELAY TIME Clock to Q	t_{PLH}, t_{PHL}	5	–	325	650	ns
		10	–	125	250	
		15	–	100	200	
Clock to $\overline{\text{Carry Out}}$		5	–	325	650	ns
		10	–	125	250	
		15	–	100	200	
$\overline{\text{Carry In}}$ to $\overline{\text{Carry Out}}$		5	–	175	350	ns
		10	–	75	150	
		15	–	55	110	
OUTPUT TRANSITION TIME	t_{TLH}, t_{THL}	5	–	130	260	ns
		10	–	65	130	
		15	–	50	100	
MINIMUM CLOCK PULSE WIDTH	PW_{CL}	5	–	170	340	ns
		10	–	85	170	
		15	–	70	140	
MAXIMUM CLOCK FREQUENCY	f_{CL}	5	1.5	3.0	–	MHz
		10	3.0	6.0	–	
		15	4.0	8.0	–	
MAXIMUM CLOCK RISE AND FALL TIME[1]	t_{rCL}, t_{fCL}	5	15	–	–	µs
		10	15	–	–	
		15	15	–	–	
MINIMUM SETUP TIME $\overline{\text{Carry In}}$	t_{setup}	5	–	130	260	ns
		10	–	65	130	
		15	–	60	100	
Up/Down		5	–	250	500	ns
		10	–	100	200	
		15	–	75	150	
PRESET OR RESET OPERATION						
PROPAGATION DELAY TIME Preset Enable or Reset to Q	t_{PLH}, t_{PHL}	5	–	325	650	ns
		10	–	125	250	
		15	–	100	200	
Preset Enable or Reset to $\overline{\text{Carry Out}}$		5	–	410	820	ns
		10	–	165	330	
		15	–	130	260	
MINIMUM PRESET ENABLE OR RESET PULSE WIDTH	PW_{PE}, PW_R	5	–	100	200	ns
		10	–	50	100	
		15	–	40	80	
PRESET ENABLE OR RESET REMOVAL TIME	t_{rem}	5	–	325	650	ns
		10	–	110	220	
		15	–	90	180	

[1] When units are cascaded, the maximum rise and fall times of the clock input should be equal to or less than the transition times of the data outputs driving data inputs, plus the propagation delay of the output driving stage for the output capacitive load.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



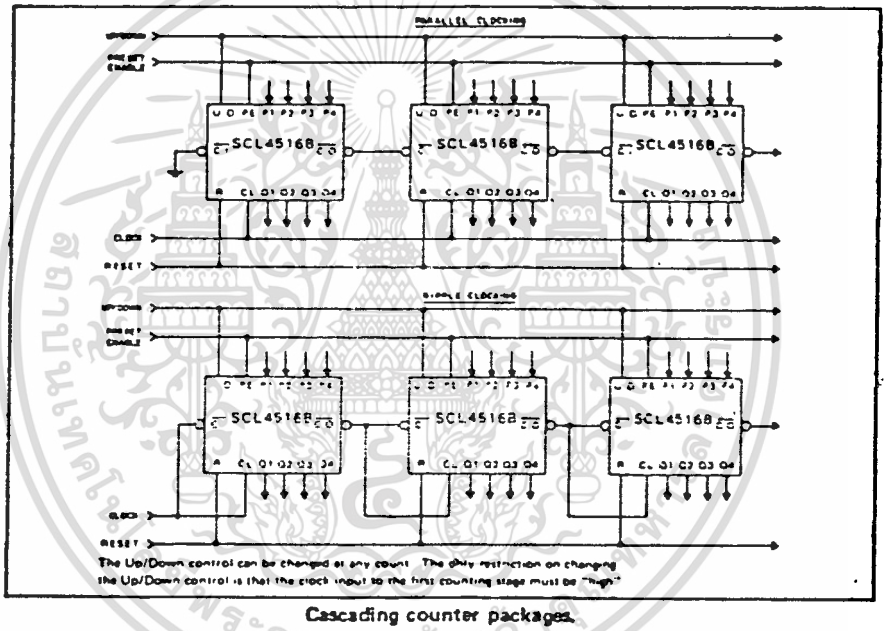
Typical P-Channel
Source Current Characteristics



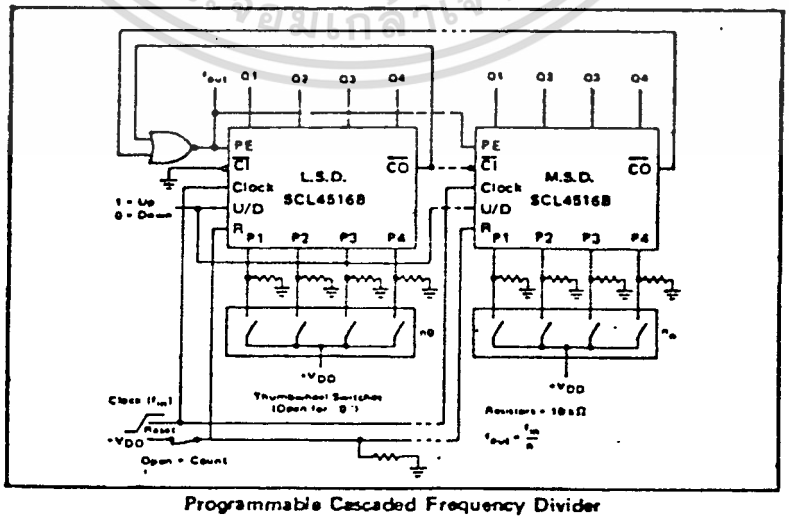
Typical N-Channel
Sink Current Characteristics

APPLICATIONS INFORMATION

CASCADING COUNTERS



Cascading counter packages



Programmable Cascaded Frequency Divider

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้