เครื่องควบคุมทิศทางของสายอากาศ

DIGITAL ANTENNA DIRECTION CONTROLLER

โดย

นายจิรศักดิ์ สรโยธิน

นายบดี วิสุทธธรรม

นายเมธี สันตสมบัติ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2534

009624

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ โครงการศึกษา 2534

ภาควิชา เทคโนโลยีการวัดคุมทางอุตสาหกรรม

คณะ วิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง เครื่องควบคุมทิศทางของสายอากาศ

ผู้จัดทำ

1. นาย จิรศักดิ์ สรรเยธิน 33.162107
2. นาย บดี วิสุทธธรรม 33.162115
3. นาย เมธี สันตสมบัติ 33.162140

(.....)

อาจารย์ ภาคร ทุตะสังกาศ

อาจารย์ที่ปรึกษา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชื่อเรื่อง เครื่องควบคุมทิศทางของสายอากาศ

ผู้ร่วมงาน

นาย จิรศักดิ์ สรรพกิน 33.162107

นาย บดี วิสุทธธรรม 33.162115

นายเมธี สันตสมบัติ 33.162140

อาจารย์ที่ปรึกษา

อาจารย์ ภาคร พุดตะสังกาศ

ปีการศึกษา 2534

บทคัดย่อ

ในปัจจุบันได้มีอุปกรณ์ที่อำนวยความสะดวก ในการที่จะให้บริการและการควบคุมต่างๆเกิดขึ้น โดยเฉพาะอย่างยิ่ง การควบคุมและการบริการ ในปัจจุบันนั้นส่วนใหญ่จะใช้ Microprocesser ควบคุมระบบต่างๆ เพราะว่าปัจจุบันนี้เทคโนโลยีทางด้าน Microprocesser ได้พัฒนาสูงขึ้น เรื่อยๆทำให้มีความสามารถในการทำงานสูง การใช้งานไม่ยุ่งยากอีกทั้งราคาก็ยังไม่แพง โครงงานนี้ก็เป็นอีกโครงงานหนึ่งที่ใช้ Microprocesser ในการควบคุมโดย Microprocesser เบอร์ 8032 วัตถุประสงค์ของโครงงานนี้ก็เพื่อที่จะใช้ Micro Controller เพื่อควบคุมการหมุนของการหมุนของสายอากาศ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DIGITAL ANTENNA DIRECTION CONTROLLER

Mr. Jirasakdi Sarayotin

Mr. Bordee Wisuthatham

Mr. Mathee Santasombat

Advisor

Mr. Pakorn Hutasangkard

1991

ABSTRACT

Todays, we have many equipments that provide convenience. In order to gine service and control everything. Espceially for both control and service. Todays most of the computer system use Microprocessor to control everything. Because the technology for Microprocessor has developed until the certain stage nowday with high capacity. The operation is not too complicate and the price is cheap. This project use Microprocessor # 8032 in controlling system.

The purpose of this project is to use Microcontroller for control the rotation of antenna.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

			หน้าที่
บทที่	1	บทนำ	1
บทที่	2	ทฤษฎีและหลักการของ MCS 51	2
บทที่	3	การจัดการหน่วยความจำ รูปแบบการกำหนดเลขที่อยู่ และตัวประมวลผลทางบูลีนของ MCS 51	67
บทที่	4	องค์ประกอบของระบบ Servo	76
บทที่	5	Solid-State Relay และ Analog To Digital	90
บทที่	6	การอ่านข้อมูลจากสวิทช์	96

ภาคผนวก

กิตติกรรมประกาศ

บรรณานุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

ในปัจจุบันชีวิตประจำวันของมนุษย์ทุกคนนั้นย่อมเกี่ยวข้องกับเทคโนโลยีอยู่ตลอดเวลา ทั้งทางตรงและทางอ้อม ไม่ว่าจะเป็นอุปกรณ์ที่ช่วยอำนวยความสะดวก อุปกรณ์เพื่อความบันเทิง และอุปกรณ์อื่นๆอีกมากมาย

Micro Processer ก็เป็นส่วนหนึ่งของอุปกรณ์เหล่านั้น และเป็นหัวใจสำคัญที่คอยควบคุมการทำงานในส่วนต่างๆของอุปกรณ์เครื่องใช้เหล่านั้น ให้เป็นไปตามคำสั่งการใช้งานของผู้ใช้ ซึ่งจะทำให้ผู้ใช้สะดวกต่อการใช้งานเป็นอย่างยิ่ง

โครงงานนี้เป็นโครงงานที่เกี่ยวกับการควบคุมทิศทางของสายอากาศ ซึ่งเป็นการควบคุมระหว่างเสาอากาศที่อยู่บน Tower และถูกควบคุมทิศทางโดยผู้ใช้ที่อยู่ทางภาคพื้นดิน โดยการใช้ Micro Controller เบอร์ 8032 เป็นตัวคอยควบคุมในการทำงาน โดยพัฒนาระบบการควบคุมการหมุนของทิศทางของเสาอากาศจากสวิทช์ที่ต้องกดอยู่ตลอดเวลา (Manual) และสามารถเลือกให้หมุนไปทางซ้ายหรือทางขวาได้ จนกระทั่งถึงมุมที่ต้องการจึงทำการหยุดการหมุนโดยการกดปุ่ม Break (ในการ Scan) หรือกดค้างไว้พอถึงมุมที่ต้องการก็ปล่อยสวิทช์ซึ่งมีความยุ่งยาก ได้พัฒนามาเป็นระบบ Keyboard Matrix ซึ่งผู้ใช้สามารถกำหนดตำแหน่งขององศาที่ต้องการให้หมุน โดยการกดคีย์ตัวเลขและสามารถเลือกให้หมุนไปทางซ้ายหรือทางขวาก็ได้ โดยการเลือกปุ่มเช่นกันนอกจากนี้ยังได้พัฒนาระบบภาคแสดงผลจาก 7-Segment มาเป็น LCD 4 Line เพื่อสะดวกต่อการใช้งานอีกด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2

ทฤษฎี และหลักการของ MCS 51

2.1 โครงสร้างของ 8032

1. สร้างโดยใช้ HMOS และ CHMOS เทคโนโลยีและทำงานด้วยแหล่งจ่ายไฟขนาด 5V. เพียงแหล่งเดียว
2. CPU มีขนาด 8 บิต
3. มีวงจรออสซิลเลเตอร์และวงจรนาฬิกาบนชิพ
4. ชุด Bank Register มี 4 ชุด แต่ละชุดมี Register 8 ตัว
5. มีตัวจับเวลา/ตัวนับ ขนาด 16 บิต 3 ชุด
6. มี Port I/O แบบขนานสองทิศทางจำนวน 4 Port Port ละ 8 บิตรวมทั้งหมด 32 เส้น แต่จะเหลือเพียง 16 เส้น สำหรับเบอร์ 8031 อีก 16 เส้นใช้ในการเข้าถึงทาง Address และข้อมูล
7. Port แบบอนุกรมสามารถที่จะโปรแกรมการรับส่งแบบ Full Duplex ที่มีความเร็วสูง
8. หนึ่งวัฏจักรคำสั่งจะใช้เวลา 1 µS. ด้วยการใช้คริสตัล 12 MHz
9. Address ข้อมูลภายนอกได้ 64 Kb
10. Address โปรแกรมภายนอกได้ 64 Kb
11. สามารถกำหนดเลขที่อยู่ข้อมูลขนาดไบต์หรือบิตได้โดยตรง
12. มีซอฟต์แวร์แฟลกสำหรับผู้ใช้ที่จะกำหนดเองได้ถึง 128 ตำแหน่งบิต
13. โครงสร้างอินเทอร์รัพต์ทำได้ 5 แหล่งและ 6 แหล่ง สำหรับ 8032/8052 พร้อมด้วยการจัด Priority ได้ 2 ระดับ
14. ตัว Processor สามารถใช้งานแบบ Boolean ได้ สำหรับการใช้งานควบคุม
15. มีคำสั่งคูณและหารทาง Hardware ทำได้ภายใน 4 µS.
16. ตัวเลขทางคณิตศาสตร์ ใช้ได้ทั้งแบบไบนารี่และเดซิมอล
17. การใช้พื้นที่สแตกสำหรับโปรแกรมย่อยต่างๆทำได้กว้างขึ้น
18. ชุดคำสั่งของ MCS-51 จะมีมากกว่าชุดคำสั่งของ MCS-48

ตระกูล MCS-51 จะมีทั้งแบบมี ROM ในตัวหรือไม่มี ROM หรือมี EPROM บนชิปเดียวกันและจะมีตำแหน่งที่ขาเหมือนกันรูปที่ 2.1 แสดงถึงตารางรายละเอียดของเบอร์ต่างๆในตระกูลMCS-51 ที่มีจำหน่ายในท้องตลาด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เบอร์	หน่วยความจำภายใน		ตัวตั้งเวลา/ ตัวนับจำนวน	อินเทอร์รัพต์
	โปรแกรม	ข้อมูล		
8052 AH	8K * 8 ROM	256 * 8 RAM	3 * 16 Bit	6
8051 AH	4K * 8 ROM	128 * 8 RAM	2 * 16 Bit	5
8051	4K * 8 ROM	128 * 8 RAM	2 * 16 Bit	5
8032 AH	ไม่มี ROM	256 * 8 RAM	3 * 16 Bit	6
8031 AH	ไม่มี ROM	128 * 8 RAM	2 * 16 Bit	5
8031	ไม่มี ROM	128 * 8 RAM	2 * 16 Bit	5
8751 H	4K * 8 EPROM	128 * 8 RAM	2 * 16 Bit	5
8751H-12	4K * 8 EPROM	128 * 8 RAM	2 * 16 Bit	5

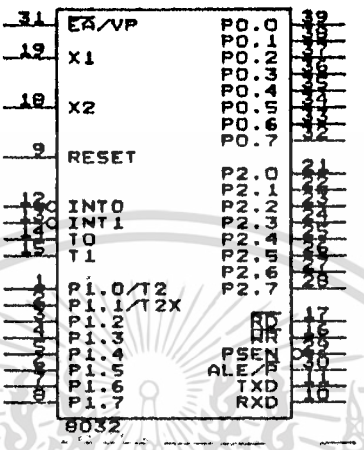
รูปที่ 2.1 ตารางรายละเอียดของตระกูล MCS-51

8751H อยู่ในกลุ่มรุ่นเดียวกับ 8051AH ที่เราสามารถโปรแกรมได้ด้วยระบบไฟและสามารถลบโปรแกรมออกได้ด้วยแสงอุลตร้าไวโอเล็ต นอกเหนือจากไอซีที่แสดงในตารางข้างบน ที่ใช้เทคโนโลยี HMOS แล้วยังมีตระกูลอื่นที่ใช้เทคโนโลยี CHMOS ที่ประหยัดพลังงานได้มากกว่า 4 เท่าของ HMOS ที่มีจำหน่ายขณะนี้คือ เบอร์ 80C51, 80C31 และ 87C51

2.2 การจัดขาลักษณะภายนอกของ MCS-51

รูปที่ 2.2 แสดงการจัดขาตามลักษณะภายนอกของชิป MCS-51 ซึ่งมีรายละเอียดดังนี้คือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.2 ลักษณะขาภายนอกของ MCS-51

1. VSS (ขา 20) เป็นขาสำหรับต่อลงดิน

2. VCC (ขา 40) เป็นขาที่ต่อแรงดันไฟกระแสตรงขนาด 5 V. และใช้สำหรับการโปรแกรม

3. PORT 0 (P0.0-P0.7/AD0-AD7) (ขา 32-39) เป็น PORT I/O 8 บิต แบบ Open Drain Bidirectional สามารถที่จะรับโหลด TTL ได้ 8 ตัว การเขียน "1" ไปที่ PORT นี้จะเป็นการปล่อยลอย (Float) ขาของ PORT นี้ทำให้มันทำงานเป็น Input มีสถานะ Impedance สูง ในการให้ PORT บริการแบบ I/O PORT 0 จะทำงานเป็น Multiplex ด้วยสัญญาณ Address ไบต์ต่ำกับบัสข้อมูล สำหรับการใช้งานด้านหน่วยความจำภายนอก ในการใช้งานแบบนี้จะใช้ลักษณะภายในเป็นตัวพูลอัพ PORT 0 ยังใช้งานเป็นตัวส่งข้อมูลออก PORT นี้ เมื่อใช้บริการทางด้านการตรวจสอบโปรแกรม ROM ภายในและการโปรแกรมตัว EPROM ภายใน ถ้าใช้งานในลักษณะนี้การพูลอัพจากภายนอกจะต้องต่อด้วยค่า 10 Kohm

4. PORT 1 (P1.0-P1.7) (ขา 1-8) เป็น PORT I/O 8 บิต แบบ Open Drain Bidirectional พร้อมด้วยการพูลอัพภายใน ถ้า PORT Output บัฟเฟอร์สามารถขับโหลด TTL

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตระกูล LSI ได้ 4 ตัว PORT 1 เมื่อถูกเขียนค่า "1" ด้วยโปรแกรมมันจะมีสถานะสูงด้วยการพูลอัพภายใน การให้สถานะเช่นนี้จะเป็นการ Initail ใช้งาน PORT นี้ให้เป็น Input ขณะที่ PORT 1 เป็น Input การให้สัญญาณลงต่ำจะเป็นการจ่ายกระแสออก เนื่องจากการพูลอัพภายใน

ในเบอร์ 8052 ขา P1.0 และ P1.7 จะใช้งานเป็น T2 และ T2 EX โดยขา T2 จะทำหน้าที่รับสัญญาณจากภายนอกให้ตัวตั้งเวลา 2 ทำงาน และขา T2 EX จะเป็น Input ผ่านเข้าตัวตั้งเวลา 2 ถูกกระตุ้นให้ทำงานแบบปกติตามโปรแกรมที่ตั้งไว้หรือ Capture

5. PORT 2 (P2.0-P2.7) (ขา 21-28) เป็น PORT I/O 8 บิต แบบ Open Drain Bidirectional ด้วยการพูลอัพภายใน PORT 2 ที่ทำหน้าที่เป็นบัฟเฟอร์ Output สามารถจ่ายโหลด TTL ตระกูล LS ได้ 4 ตัว PORT จะถูกใช้งานเป็นตัวส่ง Address ไบต์สูงด้วย เมื่อใช้งานร่วมกับหน่วยความจำภายนอกเพื่อให้ Address ได้ถึง 16 บิต ด้วยการใช้งานแบบนี้มันจะมีพูลอัพภายในที่ช่วยให้การส่งค่า '1' ได้ระดับที่แน่นอน นอกจากการใช้งานสำหรับ Address อันดับสูงยังเป็นขาควบคุมในการใช้งานตรวจสอบ

6. PORT 3 (P3.0-P3.7) (ขา 10-17) เป็น PORT I/O 8 บิต แบบพูลอัพภายใน นอกจากทำเป็น PORT I/O ที่สามารถรับโหลด TTL พวกตระกูล LS ได้ 4 ตัวแล้วยังใช้งานเป็นพิเศษสำหรับตระกูล MCS-51 ตามรายการข้างล่างนี้ด้วย

ขา PORT	ขา	การทำงานตาม FUNCTION พิเศษ
P3.0	10	Rxd PORT อนุกรม INPUT
P3.1	11	Txd PORT อนุกรม OUTPUT
P3.2	12	INT0 อินเตอร์รัพภายนอกตัวที่ 1
P3.3	13	INT1 อินเตอร์รัพภายนอกตัวที่ 2
P3.4	14	T0 สัญญาณกระตุ้นเข้าที่ตัวตั้งเวลาและตัวนับ
P3.5	15	T1 สัญญาณกระตุ้นเข้าที่ตัวตั้งเวลาและตัวนับ
P3.6	16	WR สัญญาณควบคุมการเขียน
P3.7	17	RD สัญญาณควบคุมการอ่าน

การที่จะให้ทำงานตาม FUNCTION ข้างบน จะต้องเริ่มโปรแกรมด้วยการส่งค่า '1' ไปแลทช์ไว้ก่อนที่ให้ทำงานตาม FUNCTION ข้างบน

7. ขา RST (ขา 9) ต้องคงสถานะค่าสูงเป็นเวลาประมาณอย่างน้อย 2 วัฏจักรระหว่างที่ออสซิเลเตอร์ทำงานขณะที่ต้องการ Reset ทั้งระบบงาน โดยจะต่อ Register พูลดาวน์ (8.2 Kohm) จากขา RST ไปลงดิน และเพื่อตัวชิป Reset ได้โดยอัตโนมัติ ขณะเปิดไฟจะใช้ Capacitor (10 uF) ต่อคร่อมระหว่างขา RST กับขา Vcc

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8. ขา ALE/$\overline{PROG}$ (ขา 30) เป็นขา Address แลทช์อีนาเบิ้ลด้วยการส่งพัลส์ออกไปให้สำหรับแลทช์ค่า Address ไบต์ต่ำจาก PORT 0 ในระหว่างการเข้าถึงข้อมูลจากหน่วยความจำภายใน ALE จะถูกส่งสัญญาณนาฬิกาออกมา ในอัตราความเร็วคงที่ ที่ 1/6 ของความถี่ออสซิเลเตอร์ตลอดเวลา แม้ว่าจะไม่การเข้าถึงข้อมูลจากภายใน ดังนั้นจึงสามารถที่จะใช้สัญญาณจากขานี้เป็นตัวตั้งเวลาภายนอกหรือเป็นความถี่ของสัญญาณนาฬิกา แต่อย่างไรก็ตามความถี่สัญญาณนี้จะลดความถี่ช้าลงไปเท่าหนึ่ง ระหว่างการทำงานแบบการเข้าถึงหน่วยความจำภายนอก ขานี้ยังใช้เป็นสัญญาณพัลส์เข้า สำหรับการควบคุมการโปรแกรม EPROM ภายในชิป

9. ขา PSEN (ขา 29) Program Storage Enable เป็นสโตรบอ่านข้อมูลจากหน่วยความจำภายนอก เมื่อชิปทำงานด้วยโปรแกรมภายนอก ขา PSEN จะสร้างสโตรบต่ำ 2 ครั้งภายในแต่ละวัฏจักรแมคชีน สัญญาณจะมีสถานะสูง หรือพัลส์ต่ำทั้ง 2 ลูกจะหายไป เมื่อทำงานในช่วงการอ่านหรือเขียนข้อมูลจากหน่วยความจำข้อมูลภายนอก และ PSEN จะไม่มีพัลส์ส่งออกถ้าชิปทำงานด้วยโปรแกรมหน่วยความจำภายใน

10. ขา $\overline{EA}$/Vpp (ขา 31) มีสถานะสูง ตัว CPU ในชิปจะทำงานตามโปรแกรมที่อยู่ในหน่วยความจำภายใน (โดยที่โปรแกรมจะต้องไม่ยาวกว่า 4 Kbytes) การทำให้ EA มีสถานะต่ำจะเป็นการควบคุมให้ CPU ทำงานตามโปรแกรมหน่วยความจำภายนอก ซึ่งขยายโปรแกรมได้ยาวถึง 64 Kbytes ในตัว 8031 AH และ 8032 AH ขา EA จะต้องต่อลงดินเช่นกันแม้ว่าจะไม่มี ROM อยู่ภายในก็ตาม

11. ขา XTAL1 (ขา 19) ใช้เป็นตัว INPUT เข้าสู่ตัวออสซิเลเตอร์ขยายแบบ Invert

12. ขา XTAL2 (ขา 18) ใช้เป็นตัว OUTPUT จากตัวออสซิเลเตอร์ขยายแบบ Invert

ตามตาราง 2.1 MCS-51 ทั้งสามกลุ่มคือ กลุ่มที่มี ROM,กลุ่มที่ไม่มี ROM และกลุ่ม EPROM จะมีขาใช้งานเหมือนกันหมด ยกเว้นขา 1 จะใช้งานเป็น T2 และขา 2 เป็น T2 EX ในเบอร์ 8032/8052 ตลอดถึงจังหวะเวลา (TimingDiagram) และคุณสมบัติทางไฟฟ้าทั้งสามจะแตกต่างกันเฉพาะการโปรแกรมบนชิป MCS-51 เท่านั้น ซึ่งแต่ละแบบจัดไปตามความต้องการของผู้ใช้ เช่น 8751 จะมี 4 Kbyte ของ Ultraviolet-Erasable Programmable Read Only Memory (EPROM) เหมาะสำหรับการพัฒนาเครื่องแบบและการผลิตอุปกรณ์ที่มีจำนวนจำกัด เมื่อต้องการจะเขียนโปรแกรมเข้า EPROM จะต้องมีตัวเขียนโปรแกรมพิเศษสำหรับเขียนโปรแกรมที่ผู้ออกแบบเขียนขึ้นมาได้ ถ้าโปรแกรมมีโค้หรือส่วนผิดพลาดที่ต้องการจะแก้ไข ก็สามารถแก้ไขได้โดยการนำตัว 8751 นี้ไปล้างโปรแกรมเดิมออกด้วยแสงอุลตร้าไวโอเลตและอัดข้อมูลโปรแกรมที่ได้แก้ไขแล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เข้าไปใหม่ ทำเช่นนี้จนกระทั่งได้โปรแกรมสมบูรณ์และเมื่อต้องการผลิตจำนวนมากก็สามารถที่จะใช้ MCS-51 เบอร์ 8051 ที่มี 4 kbyte ของ ROM ซึ่งจะถูกอัดข้อมูลโปรแกรมตามความต้องการของผู้ออกแบบโดยโรงงานผู้ผลิตชิปเบอร์นี้ การผลิตลักษณะนี้จะถูกกว่าการใช้เบอร์ 8751 แต่โปรแกรมภายในไม่สามารถลบและโปรแกรมใหม่ได้หลังการผลิตไปแล้ว

ส่วนเบอร์ 8031 จะไม่มีหน่วยความจำของโปรแกรมบนชิปแต่อาจต่อหน่วยความจำโปรแกรมจากภายนอกด้วย ROM EPROM หรือ PROM ได้ถึง 64 Kbyte ดังนั้น 8031 จึงเหมาะสำหรับการใช้งานที่โปรแกรมมีขนาดใหญ่กว่า 4 Kbyte และสำหรับผู้ออกแบบที่ต้องการแยกส่วนของโปรแกรมออกจากชิป

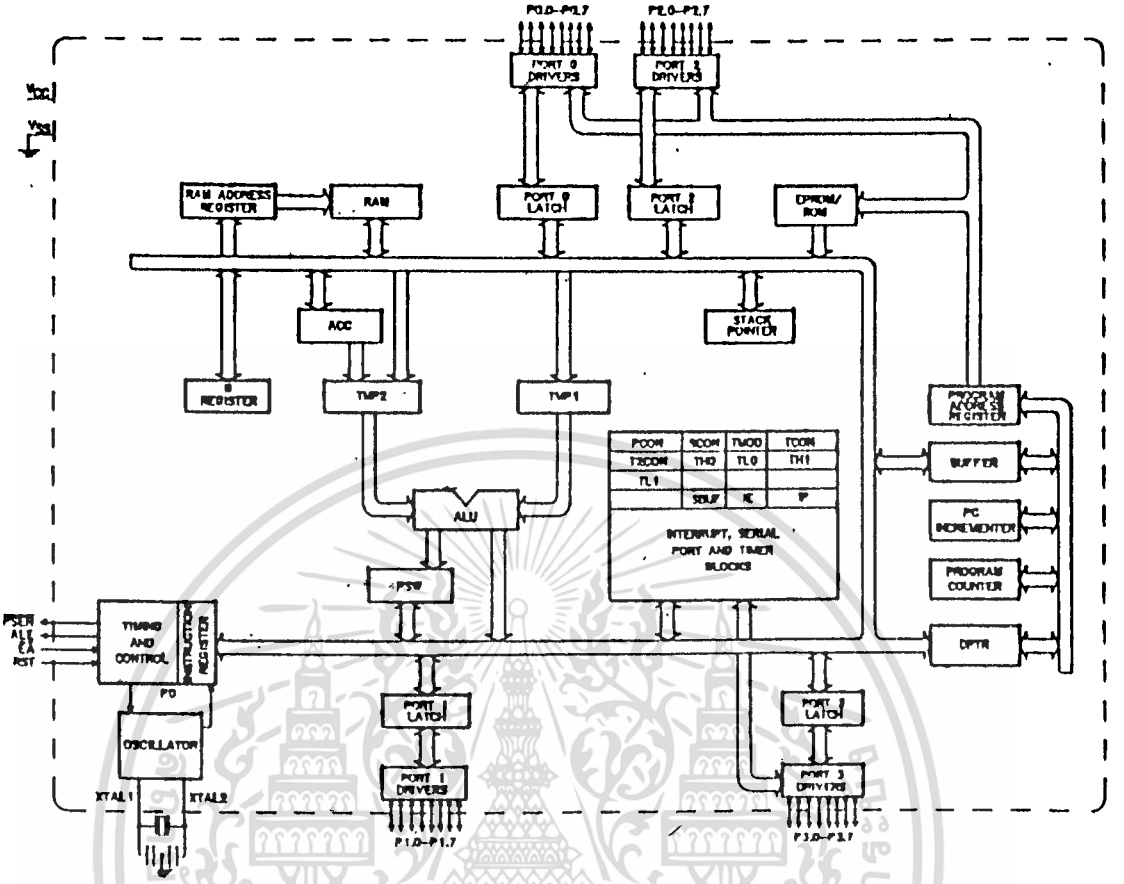
2.3 การจัดการทางสถาปัตยกรรม

รูปที่ 2.3 เป็น Block Diagram ที่แบ่งตามลักษณะงานทางสถาปัตยกรรมภายในของ MCS-51 จากรูปจะเห็นได้ว่าแต่ละตัวของตระกูลนี้จะประกอบด้วยหน่วยศูนย์กลางการประมวลผล หน่วยความจำสองชนิด คือแบบ RAM กับ ROM หรือ EPROM PORT OUTPUT INPUT โหมด Register สถานะและข้อมูล ส่วนวงจรตรรกในการ RANDOM ที่จำเป็นสำหรับตัวแปรของฟังก์ชั่นการต่อพ่วง ส่วนต่างๆที่กล่าวนี้จะติดต่อกันด้วยบัสข้อมูลขนาด 8 บิต และจะมีบัฟเฟอร์สำหรับการติดต่อข้อมูลกับภายนอกผ่าน PORT I/O เมื่อต้องการขยายหน่วยความจำหรือ PORT I/O

2.4 หน่วยศูนย์กลางประมวลผลหรือ CPU

CPU เป็นมันสมองระบบ Microcomputer การอ่านโปรแกรมและการทำงานตามคำสั่งโปรแกรมจะกระทำที่ส่วนนี้โดยการใช้ส่วนคณิตศาสตร์และตรรกศาสตร์ทำงานร่วมกับ Register A,B, PSW (Program Status Word), SP (Stack Poiter) ตัวนับโปรแกรม (PC:Program Counter) ขนาด 16 บิต และตัวชี้ตำแหน่งข้อมูล (DPTR:Data Pointer) ส่วนคณิตศาสตร์และตรรกศาสตร์ (ALU:Arithmetic Logic Unit) ALU นี้ทำงานในฟังก์ชั่นทางคณิตศาสตร์และตรรกศาสตร์ด้วยตัวแปรต่างๆขนาด 8 บิต ที่มีลักษณะการทำงานทางคณิตศาสตร์เป็น บวก ลบ คูณ หาร รวมทั้งทางตรรกศาตร์ เช่น AND OR XOR รวมทั้งการเลื่อนและการวนรอบบิต การเคลียร์ค่าและการกลับค่า (Complement) เป็นต้น ALU ยังสามารถที่จะตัดสินใจในการให้กระโดดไปทำคำสั่งของโปรแกรมในส่วนอื่นๆตามเงื่อนไขที่ตั้งขึ้นและยังแบ่ง Register ทั่วคราวใช้สำหรับเป็นทางผ่านชั่วคราว ของข้อมูลในการถ่ายเทภายในระบบคำสั่งอื่นที่มีการใช้ ALU ยังมีความสามารถที่จะเพิ่มค่าใน Register ในลักษณะการบวกด้วยหนึ่ง (Increment) หรือคำนวณเลขที่อยู่ของข้อมูลที่จะนำไปเก็บหรือการลดค่าลงครั้งละหนึ่งในลักษณะการลบด้วยค่าหนึ่ง (Decrement) โดยอัตโนมัติหรือใช้ในการเปรียบเทียบค่าตัวแปรทั้งสอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.3 สถาปัตยกรรม MCS-51

สิ่งสำคัญในการทำงานทางสถาปัตยกรรมของ MCS-51คือความสามารถในการทำงานสำหรับข้อมูลขนาด 8 บิตและ 1 บิตการใช้งานในระดับบิตในการเซต เคลียร์ หรือกลับค่า การเคลื่อนย้าย การทดสอบและการใช้ในการคำนวณทางตรรกขนาด 1 บิต ความสามารถเช่นนี้เหมาะสำหรับใช้ในการควบคุมสัญญาณเข้าและออกที่มีการคิด และการออกแบบทางตรรกด้วยพีชคณิต Boolean ซึ่งโดยปกติทำได้ลำบากสำหรับ Microprocessor ทั่วๆไป งานในลักษณะเช่นนี้จึงได้ชื่ออีกอย่างหนึ่งว่า ตัวประมวลผลบูลลีน (Boolean Processor)

2.4.1 ACCUMULATOR (ACC)

MCS-51 ก็เช่นเดียวกับ MCS-48 ที่ใช้ ACC ที่มีขนาด 8 บิต เป็น Accumulator หลักคำสั่งส่วนใหญ่จะอ้างถึง Register ตัวนี้ โดยถือค่าภายในเป็นค่าตัวตั้งและรับค่าผลลัพธ์ที่ได้จากคำสั่งทางคณิตศาสตร์ เช่น บวก ลบ คูณ หาร เข้ามาเก็บในตัว ACC ยังสามารถใช้เป็นตัวแหล่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กระทำหรือถูกกระทำในการทำงานหลักของระบบ และเป็นตัวกลางในการถ่ายเทข้อมูลในการติดต่อกับอุปกรณ์ภายนอก I/O และหน่วยความจำภายนอกรวมถึงการตรวจสอบตารางข้อมูล

2.4.2 REGISTER B

เป็น Register พิเศษที่ใช้งานสำหรับคำสั่งของการคูณและการหาร โดยใช้เป็นที่เก็บคูณหรือตัวหารและเป็นที่เก็บผลลัพธ์ตัวที่สองหลังการคูณและเศษหลังการหาร

(MSB) (LSB)

CY	AC	F0	RS1	RS0	OV	-	P

สัญญาลักษณ์	ตำแหน่ง	ข้อกำหนดการทำงาน
CY	PSW7	แฟลกตัวทด จะเซต/เคลียร์ด้วย Hardware หรือ Software ระหว่างผลลัพธ์หลังการใช้คำสั่งทางคณิตศาสตร์หรือตรรกศาสตร์ที่เน่นอน
AC	PSW6	แฟลกตัวทดของ AUXILIARY จะเซต/เคลียร์ด้วย Hardware ระหว่างการบวก และการลบที่แสดงผลจากการทดหรือยืมจากบิตที่ 3 ของ ACC
F0	PSW5	แฟลก 0 จะเซต/เคลียร์ด้วย Software ที่ผู้ใช้กำหนดสถานะแฟลกนี้เอง
RS1	PSW4	Registerตัวควบคุมการเลือกแบงค์ ด้วยค่า RS1 และ RS0
RS0	PSW3	จะเซต/เคลียร์ด้วย Software เพื่อเลือกกลุ่มRegisterทำงานในแต่ละแบงค์โดยปรับค่าใน RS1 และ RS0 ให้ตีนาเบิ้ลคลุมการเลือกลักษณะการเลือกแบงค์ต่อไปนี้

RS1	RS0	เลือกแบงค์	ค่า ADDRESS
0	0	แบงค์ 0	00H - 07H
0	1	แบงค์ 1	08H - 0FH
1	0	แบงค์ 2	10H - 17H
1	1	แบงค์ 3	18H - 1FH

OV	PSW2	แฟลกOVERFLOW จะเซต/เคลียร์ด้วย Hardware ระหว่างการใช้คำสั่งที่แสดงผลถึงการเกิดลักษณะ Overflow ทางคณิตศาสตร์
-	PSW1	บิตสำรอง จะไม่สามารถเซต/เคลียร์ด้วยผู้ใช้เพราะสำรองไว้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

009621

สำหรับโรงงานผู้สร้าง

P	PSW0	แฟลกพาริตี้ จะเซต/เคลียร์ด้วย Hardware ในแต่ละวัฏจักรคำสั่งแสดงถึงตัวเลขค่า '1' ในแต่ละบิตของ Accumulator เช่น '1' มี 6 ตัวจะเป็นพาริตี้คู่ P บิตจะเท่ากับ 0

หมายเหตุ ความหมายของ Hardware และ Software ในตารางต่างๆในแต่ละบิตใน Register การที่จะเซตหรือเคลียร์นั้นถ้าเกิดขึ้นจาก Hardware จะหมายถึงว่าค่าบิตใน Register จะเกิดเซตตัวเอง เนื่องจากผลของความหมายของการทำงานตามคำสั่งของบิตนั้น เช่น TI จะเซตตัวเองด้วย Hardware เมื่อการส่งข้อมูลได้สิ้นสุดถึง Stop บิตแล้วช่วยให้เราสามารถตรวจสอบได้ว่าการส่งข้อมูลครั้งละ Byte สิ้นสุดหรือยัง ถ้ายังจะได้รอต่อไปก่อนหรือมีการคำนวณแล้วผลลัพธ์ที่เกิด Overflow ใน PSW ก็จะเซตตัวเองที่บิต OV ส่วนทาง Software หมายถึงว่าเราสามารถที่จะเซตหรือเคลียร์ได้ด้วยการใช้คำสั่งต่างๆ ในการเซตหรือเคลียร์ในบิตแต่ละบิตของ Register เป็นลักษณะทาง Software

2.4.3 REGISTER คำแสดงสถานะโปรแกรม (Program Status Word : PSW)

Register PSW เป็น Register ที่แสดงผลที่ได้หลังจากการใช้คำสั่งต่างๆและใช้เป็นตัวเลือกกลุ่มการทำงานของ Register กลุ่มต่างๆ

2.4.4 ตัวชี้สแต็ก (Stack Pointer : SP)

MCS-51 จะรวมเอา Stack ทาง Hardware ที่ใช้ RAM ภายในสำหรับการเชื่อมต่อระหว่างโปรแกรมหลัก Stack การผ่านพารามิเตอร์ระหว่างงานในแต่ละโปรแกรมและ Stack เก็บตัวแปรข้อมูลชั่วคราวหรือ Stack การเก็บสถานะระหว่างการบริการงานอินเตอร์รัพท์ไว้ภายในชิป โดยที่ SP จะมีขนาด 8 บิต จะเพิ่มค่าขึ้นโดยอัตโนมัติก่อนที่ข้อมูลจะนำมาเก็บในหน่วยความจำระหว่างการใช้คำสั่ง PUSH และ CALL และจะลดค่า SP ลงหลังจากที่ได้ถ่ายเทข้อมูลออกไปแล้วในคำสั่ง POP และ RETURN โดยทฤษฎีทางสถาปัตยกรรม MCS-51 สามารถใช้Stack ให้มีเนื้อที่ถึง 128 Byte แต่ในทางปฏิบัติสำหรับโปรแกรมทั่วไปจะใช้น้อยกว่านี้ SP จะเริ่มที่ตำแหน่ง 07H ดังนั้น Stack จะเริ่มบรรจุข้อมูลที่ตำแหน่ง 08H MCS-51 สามารถเปลี่ยนแปลงค่าใน SP ได้ ซึ่งจะเป็นการเปลี่ยนตำแหน่ง Stack ไปยังที่ใดๆของ RAM ภายในชิป

2.4.5 ตัวชี้ข้อมูล (Data Pointer : DPTR)

DPTR Register ขนาด 16 บิต ที่ประกอบด้วยไบต์สูง (DPH) และไบต์ต่ำ (DPL) ที่เราสามารถเลือกแบ่งออกเป็น Register 8 บิต สองตัวที่ใช้ได้อย่างอิสระหรือจะรวมกันทั้ง 16 บิต ก็ได้ ในการ Increment หรือ Decrement เพื่อประโยชน์ในการใช้เป็นฐานของเลขที่อยู่ใน Register ในการกระโดดโดยทางอ้อมในการใช้คำสั่งเกี่ยวกับตารางข้อมูลและชี้ตำแหน่งของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน่วยความจำภายนอก

2.4.6 Port 0 ถึง 3

Register P0,P1,P2 และ P3 ของกลุ่ม Specail Function Reigster (SFR) จะเป็นตัว Register ที่เลทช์ค่าของ Port 0,1,2 และ 3 ตามลำดับในขณะใช้งาน

MCS-51 จะจัดแบ่งตำแหน่งสำหรับ SFR ให้ทำงานเป็น Register ต่างๆดังนี้

		ตำแหน่ง
* ACC	Accumulator	0E0H
* B	Register B	0F0H
* PSW	Program Status Word	0D0H
SP	Stack Pointer	081H
DPTR	ตัวชี้ข้อมูล ประกอบด้วย DPH	083H
	และ DPL	082H
* P0	Port 0	080H
* P1	Port 1	090H
* P2	Port 2	0A0H
* P3	Port 3	0B0H
* IP	ตัวควบคุมการอินเตอร์รัพต์ตามลำดับ	0B8H
* IE	ตัวควบคุมการอินเตอร์รัพต์อีนาเบิ้ล	0A8H
TMOD	ตัวควบคุมการการเลือกรูปแบบตัวตั้งเวลา/ตัวนับ	089H
* T2CON	ตัวควบคุมตัวตั้งเวลา/ตัวนับ 2	088H
TCON	ตัวควบคุมตัวตั้งเวลา/ตัวนับ	0C8H
TH0	Register ตัวตั้งเวลา/ตัวนับ 0 (ไบต์สูง)	08CH
TL0	Register ตัวตั้งเวลา/ตัวนับ 0 (ไบต์ต่ำ)	08AH
TH1	Register ตัวตั้งเวลา/ตัวนับ 1 (ไบต์สูง)	08DH
TL1	Register ตัวตั้งเวลา/ตัวนับ 1 (ไบต์ต่ำ)	08BH
+ TH2	Register ตัวตั้งเวลา/ตัวนับ 2 (ไบต์สูง)	0CDH
+ TH2	Register ตัวตั้งเวลา/ตัวนับ 2 (ไบต์ต่ำ)	0CCH
+ RLDH	Register ตัวตั้งเวลา/ตัวนับ 2 ประจุใหม่อัตโนมัติ (ไบต์สูง)	0CBH
+ RLDL	Register ตัวตั้งเวลา/ตัวนับ 2 ประจุใหม่อัตโนมัติ (ไบต์ต่ำ)	0CAH
* SCON	ควบคุมการส่งข้อมูลอนุกรม	098H

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SBUF บัฟเฟอร์ข้อมูลการส่งอนุกรม 099H

PCON ควบคุมการใช้พลังงาน (Power) 097H

หมายเหตุ เครื่องหมาย * หน้าตัว Register แสดงว่า Register นั้นสามารถที่จะ Address ข้อมูลได้ทั้งข้อมูลขนาดไบต์และบิต และเครื่องหมาย + นั้นแสดงว่าจะมีเฉพาะในเบอร์ 8032/8052 เท่านั้น

2.4.7 บัฟเฟอร์ข้อมูลอนุกรม (Serial Data Buffer : SBUF)

บัฟเฟอร์ข้อมูลอนุกรมแบ่งออกเป็น Register สองตัว ตัวหนึ่งเป็นบัฟเฟอร์การส่งและอีกตัวหนึ่งเป็นบัฟเฟอร์การรับ เมื่อข้อมูลถ่ายเทเข้า SBUF มันจะถ่ายเทเข้าบัฟเฟอร์ส่งซึ่งเป็นตัวจัดการส่งข้อมูลอนุกรม วิธีการเคลื่อนย้ายเข้า SBUF ขึ้นอยู่กับการเริ่มแรก (Initial) การส่ง เมื่อข้อมูลย้ายออกจาก SBUF จะเป็นการรับข้อมูลจากบัฟเฟอร์ตัวรับ

2.4.8 Register Capture

ไอซีเบอร์ 8032/8052 จะมีคู่ Register (RCAP2H,RCAP2L) เพิ่มเติมเป็น Register Capture สำหรับตัวตั้งเวลาหมายเลข 2 ในโหมดการใช้งานของ Register ตัวนี้จะรับการเปลี่ยนแปลงที่เข้ามาที่ขา T2 EX ตัว TH2 และ TL2 จะลอกข้อมูลเข้าไปใน Register คู่ RCAP2H และ RCAP2L ด้วยการใช้ตัวตั้งเวลาจะมีโหมดการบรรจุอัตโนมัติขนาด 16 บิต สำหรับการใช้ตัวตั้งเวลา/ตัวนับ 2

2.4.9 Control Register

กลุ่ม SFR ที่เป็น IP,IE,TMOD,TCON,T2CON,SCON และPCON จะประกอบด้วยบิตที่ใช้ในการควบคุมและแสดงสถานะการใช้งานในระบบอินเตอร์รัพต์ ตัวตั้งเวลา/ตัวนับและ Port อนุกรม

2.5 การจัดการหน่วยความจำ

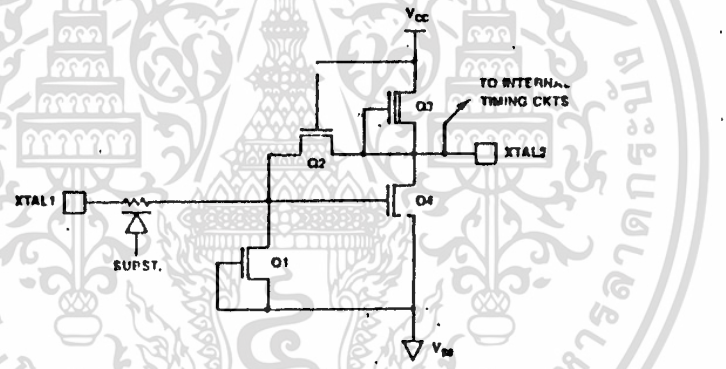
ตัว MCS-51 จะแยก Address สำหรับหน่วยความจำของโปรแกรมและหน่วยความจำของข้อมูลออกจากกัน หน่วยความจำของโปรแกรมขยายได้ถึง 64 Kbyte และจำนวนไบต์ต่ำ 4 Kbyte จะอยู่ใน 8051 หน่วยความจำของข้อมูลมี 128 Byte (256 Byte สำหรับ8032/8052) บนชิปและอีก 128 Byte ใช้สำหรับ SFR และหน่วยความจำของข้อมูลภายนอกอีก 64 Kbyte

2.6 Oscillator และวงจรนาฬิกา

วงจร Oscillator ที่อยู่ในชิปแบบ HMOS ของตระกูล MCS-51จะเป็น Single Linear Inverter เพื่อใช้ Crystal ควบคุมเป็น Oscillator แบบรีแอ็กตีฟบวก ดังรูปที่ 2.5 ในการใช้งาน Crystal นี้จะกำหนดการทำงานที่โหมด Fundamental เสมือนเป็น Inductance โดยต่อขนานอยู่กับตัวคาปาซิเตอร์ภายนอกที่ต่อกับ Crystal การกำหนดตัว Crystal และค่าคาปาซิเตอร์ C1 และ C2 ไม่ค่อยวิกฤตินักอาจจะมีค่าประมาณ 30 PF สำหรับทุกความถี่ของตัว

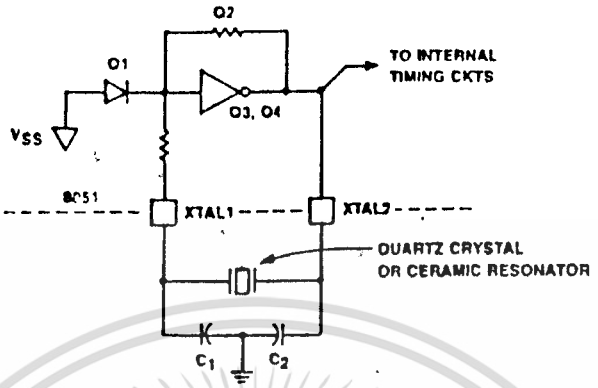
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Crystal ชั้นดี ส่วนการใช้ Ceramic Resonator ค่าคาปาซิเตอร์ที่มาต่อจะมีค่าสูงกว่าโดยมีค่าประมาณ 47 PF การใช้ค่าคาปาซิเตอร์อาจเปลี่ยนแปลงได้ขึ้นอยู่กับตัว Ceramic Resonator นั้นๆ การขับตัว HMOS ด้วยสัญญาณนาฬิกาจากภายนอกก็กระทำได้เช่นกันโดยการต่อเข้าที่ขา XTAL2 และต่อลงดินที่ขา XTAL1 ดังรูปที่ 2.6 ควรจะใช้ตัวต้านทานพลูอัพเพราะที่ XTAL2 ต้องการสัญญาณในระดับ TTL

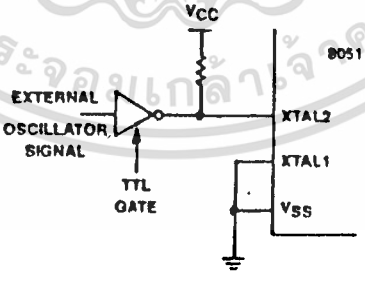


รูปที่ 2.4 วงจร Oscillatator ภายใน MCS-51 แบบ HMOS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.5 การต่อวงจร Oscillatator บน HMOS ชิป



รูปที่ 2.6 การจ่ายสัญญาณนาฬิกาภายนอกในการขับ HMOS MCS-51

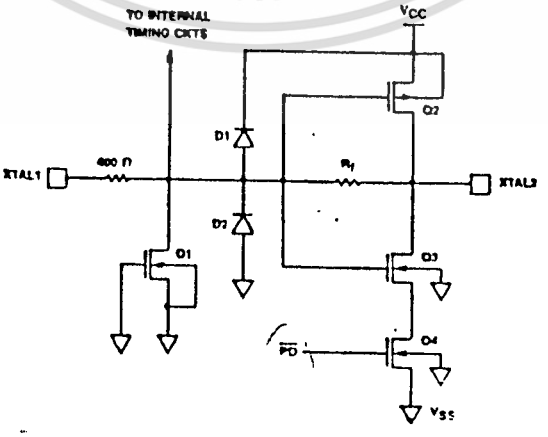
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนแบบ CHMOS วงจร Oscillator ที่อยู่ในชิปของ 80C51 มีดังรูปที่ 2.7 ซึ่งประกอบด้วย Single Stage Linear Inverter สำหรับการใช้ Crystalควบคุมให้เป็นOscillator Reactance บวกเช่นเดียวกับ HMOS แต่ก็มีข้อแตกต่างบางประการที่สำคัญก็คือ 80C51 สามารถที่จะหยุดการทำงานของตัว Oscillator ภายใต้การควบคุมด้วย Software ด้วยการเขียน '1' เข้า PD บิต ของ Register PCON ที่แตกต่างอีกข้อก็คือวงจรสัญญาณนาฬิกาภายนอกของCHMOS จะถูกขับด้วยสัญญาณที่ขา XTAL1 ตรงกันข้ามกับ HMOS

ตัว Register Feedback (Rf) ในรูปที่ 2.8 ประกอบด้วย N และ P Channel FETS ที่ถูกควบคุมด้วยบิต PD ดังนั้น Rf จะเปิดวงจรต่อเมื่อ PD=1 ตัวไดโอด D1,D2 ทำหน้าที่ CLAMP กับ Vcc และ Vss ที่จะเข้า Rf FETS

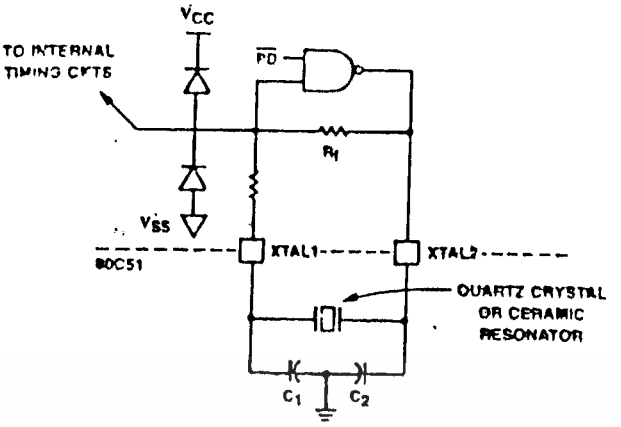
Oscillator สามารถใช้ขนิดเดียวกันกับที่ใช้แบบ HMOS ตามรูปที่ 2.8 ใช้ C1=C2=30PF เมื่อ Feedback เป็นตัว Quartz Crystal และ C1=C2= 47 PF เมื่อใช้ตัว Ceramic Resonator การขับ CHMOS ด้วยสัญญาณนาฬิกาภายนอกจะต่อเข้าที่ขา XTAL1 และปล่อย XTAL2 ลอยไว้แสดงดังรูปที่ 2.9 ไม่มีข้อกำหนดที่แน่นอนสำหรับขนาดของ Duty Cycleของ Oscillator ภายนอก เพียงแต่สัญญาณเข้าจะถูกหารสองให้เป็นสัญญาณนาฬิกาเมื่อต้องการใช้ช่วงเวลาของสัญญาณนาฬิกามาควบคุมงานในระบบ

สัญญาณนาฬิกาที่จะได้จากการนำเอา 2 ไปหารสัญญาณของ Oscillator และแบ่งจ่ายเข้าชิปเป็นสองเฟส เฟสที่หนึ่ง สัญญาณจะ Active ระหว่างครึ่งแรกของแต่ละคาบสัญญาณนาฬิกา และเฟสที่สองสัญญาณจะ Active ระหว่างครึ่งที่สองของแต่ละคาบสัญญาณนาฬิกา โดยให้ XTAL2 เป็นขาอินพุทเข้าของตัวกำเนิดสัญญาณนาฬิกานี้

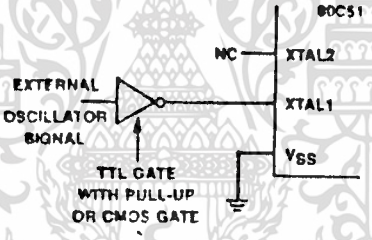


รูปที่ 2.7 วงจร Oscillatator ภายใน MCS-51 แบบ CHMOS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.8 การให้วงจร Oscillatator บน CHMOS ทีป



รูปที่ 2.9 การจ่ายสัญญาณนาฬิกาภายนอกในการขับ CHMOS MCS-51

2.7 ช่วงจังหวะเวลาของ CPU

วัฏจักรแมคชีนประกอบด้วย 6 สถานะหรือเท่ากับ 12 คาบของ Oscilltor แต่ละสถานะแบ่งเป็นเฟส 1(P1) ครึ่งหนึ่งเป็นช่วงเฟส 1 Active และเฟส 2(P2)เป็นช่วงเฟส 2 Active ดังนั้นในแต่ละวัฏจักรแมคชีนจะประกอบด้วย 12 คาบของ Oscilltor เป็นจำนวน S1P1 คือสถานะที่ 1 เฟสที่ 1 ถึง S6P2 คือสถานะที่ 6 เฟสที่ 2 โดยปกติการทำงานแบบคณิตศาสตร์และแบบตรรกศาสตร์จะทำในช่วงเฟส 1 และการถ่ายเทข้อมูลภายในระหว่าง Register จะทำในช่วงเฟส 2

ตามแผนภูมิในรูปที่ 2.10 แสดงถึงช่วงเวลาการเฟ็ทช์และการทำงานที่อ้างถึงลักษณะภายใน

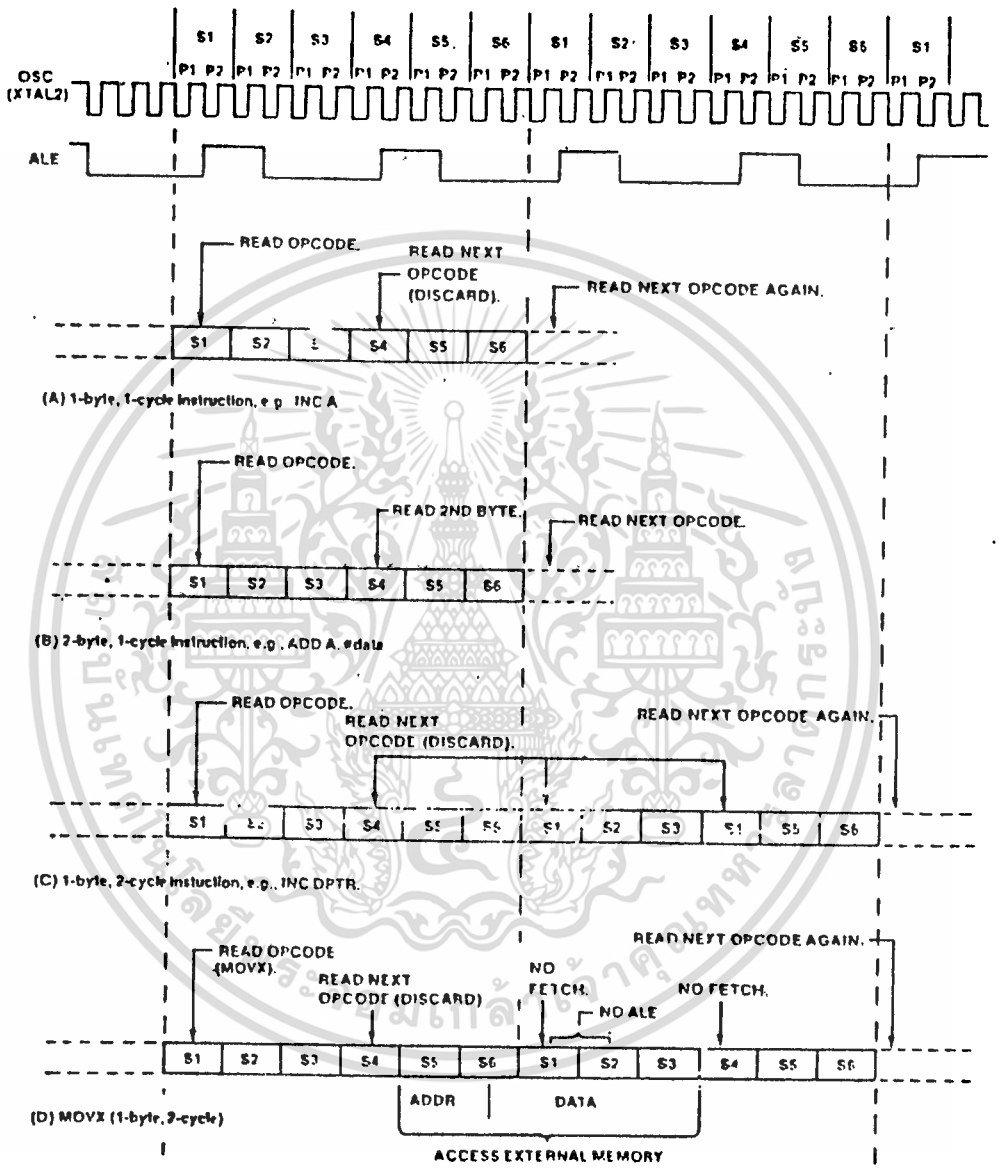
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และเฟส เนื่องจากสัญญาณนาฬิกาภายในผู้ใช้ไม่สามารถที่จะควบคุมการเข้าถึงภายในได้ โดยปกติ ALE จะ Active 2 ครั้งในแต่ละวัฏจักรแมคชีนและจะเกิดขึ้นระหว่าง S1P2 ถึง S2P1 ครั้งหนึ่ง และระหว่าง S4P2 ถึง S5P1 อีกครั้งหนึ่ง

การทำงานของแต่ละวัฏจักรคำสั่งจะเริ่มที่ S1P1 เมื่ออ็อปโค้ดเก็บเข้าในตัว Register คำสั่งหรือการอ่านอ็อปโค้ดเข้ามาถ้าคำสั่งมีสองไบต์ ไบต์ที่สองจะถูกอ่านในช่วง S4 ภายในวัฏจักรแมคชีนเดียวกัน แต่ถ้าเป็นหนึ่งไบต์คำสั่ง จะยังคงเฟ็ทช์ที่ S4 แต่ไบต์ที่ถูกอ่าน (ซึ่งควรเป็นไบต์ที่สองของคำสั่งเดียวกัน) จะไม่มีผลและตัวนับโปรแกรม (PC) จะยังไม่เพิ่มค่าไม่ว่ากรณีใด การทำงานจะสมบูรณ์ที่ปลายของ S6P2 ตามรูปที่ 2.10(A) และ 2.10(B) เป็นการแสดงแผนภูมิเวลาสำหรับ 1 ไบต์ใน 1 รอบคำสั่งกับ 2 ไบต์ใน 1 รอบคำสั่ง

คำสั่ง MCS-51 ส่วนใหญ่จะทำงานในช่วงหนึ่งวัฏจักรยกเว้นคำสั่ง MUL(คูณ) DIV(หาร) ที่ใช้มากกว่าสองวัฏจักรในการที่จะทำงานให้สมบูรณ์ได้จะใช้ถึงสี่วัฏจักร ปกติรหัสสองไบต์จะถูกเฟ็ทช์จากหน่วยความจำโปรแกรมช่วงทุกวัฏจักรแมคชีน ยกเว้นคำสั่งพิเศษ MOVX ซึ่งมี 1 ไบต์คำสั่งแต่จะใช้เวลาสองวัฏจักรในการเข้าถึงหน่วยความจำภายนอก ระหว่างการทำ MOVX การเฟ็ทช์จะถูกสคิปหรือหายไป ขณะที่หน่วยความจำภายนอกจะถูก Address และ Stobe หรือกระตุ้นรับเข้าไปใน CPU รูปที่ 2.10(C) และ 2.10(D) เป็นการแสดงแผนภูมิเวลาปกติของคำสั่งประเภท 1ไบต์แต่ใช้ 2 วัฏจักรแมคชีน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.10 แสดงถึงช่วงจังหวะการ เฟทช์และการทำงานตามลำดับที่อ้างถึงลักษณะภายในและ เฟส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.8 โครงสร้างPortและการทำงาน

ใน MCS-51 มี Port 4 Port และทั้ง 4 Port เป็นแบบสองทิศทางแต่ละ Port จะประกอบด้วยแลทช์เป็น P0 ถึง P3 ของ SFR จะมีตัวขับ Output และ Buffer Input ตัวขับ Output ของ Port 0 และ 2 และ Buffer Input ของ Port 0 จะใช้งานสำหรับการเข้าถึงหน่วยความจำภายนอก ในการใช้งานลักษณะนี้ OutputPort 0 จะทำหน้าที่เป็นตัวกำหนดไบต์ต่ำของ Address หน่วยความจำภายนอก โดยที่ค่า Address และค่าข้อมูลจะถูกมัลติเพล็กซ์ด้วยช่วงจังหวะการเฟทช์และการอ่านหรือเขียนข้อมูล ส่วน OutputPort 2 จะทำหน้าที่เป็นตัวกำหนดส่งไบต์สูงของ Address ในการเข้าถึงหน่วยความจำภายนอก

บางขาของตัวขับเอาท์พุทและบัฟเฟอร์อินพุทของขา 1.0,1.1 และ Port3 ทั้งหมดสามารถนำไปใช้งานเป็นแบบหลายฟังก์ชั่น (Multifunction)

ขา Port	การใช้งานตาม Function
* P1.0	T2 (Timer/Counter 2 สัญญาณอินพุทจากภายนอก)
* P1.1	T2RST(Timer/Counter 2 สัญญาณอินพุทการรีเซตภายนอก)
P3.0	RxD (Port รับข้อมูลอนุกรม)
P3.1	TxD (Port ส่งข้อมูลอนุกรม)
P3.2	INT0 (การใช้อินเทอร์รัพต์ภายนอกตัวที่ 1)
P3.3	INT1 (การใช้อินเทอร์รัพต์ภายนอกตัวที่ 2)
P3.4	T0 (Timer/Counter 0 สัญญาณอินพุทจากภายนอก)
P3.5	T1 (Timer/Counter 1 สัญญาณอินพุทจากภายนอก)
P3.6	WR (สโตรบการเขียนหน่วยความจำภายนอก)
P3.7	RD (สโตรบการอ่านหน่วยความจำภายนอก)

ตัวขับเอาท์พุทแลทช์ในการที่จะให้ทำงานตามตารางข้างบน จะต้องเริ่มโปรแกรมด้วยการเซตค่า '1' เก็บไว้ในแลทช์ก่อน เครื่องหมาย * แสดงถึงการใช้ตัว Timer/Counter 2 ซึ่งมีเฉพาะในเบอร์ 8032/8052 เท่านั้น ทั้งโหมดตัวจับเวลาหรือตัวนับการโหลดใหม่แบบอัตโนมัติของ Timer/Counter 2 ที่ Register RLDH และ RLDL จะเกิดขึ้นถ้าการโหลดใหม่แบบอัตโนมัติถูกเลือกใช้งานด้วยการกำหนดในบิต CP/RL2 = 0

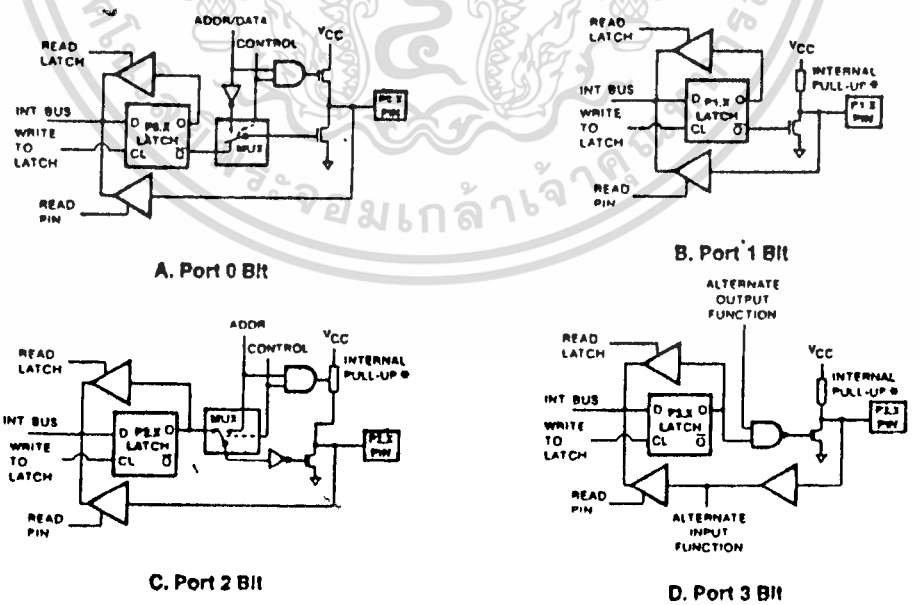
2.8.1 (I/O Configuration)

รูปที่ 2.11 เป็น Port วงจรการแลทช์และบัฟเฟอร์ แสดงรูปแบบของบิตแต่ละ Port ที่ Port 1,2 และ 3 จะมีพูลอัพภายใน Port 0 Output เป็น Open Drain แต่ละเส้นของ I/O จะเป็นอิสระในการกำหนดเป็น Input หรือ Output Port 0 และ2 อาจใช้เป็น I/O หัวบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไม่ได้ ถ้าถูกกำหนดให้เป็น Address Bus และข้อมูลแล้ว กรณีการใช้ Port เป็น Input นั้นจะสามารถรับการเปลี่ยนแปลงระดับลอจิกจากระดับสูงสู่ระดับต่ำเท่านั้น ดังนั้นการใช้ขาใดขาหนึ่งของ Port เป็น Input ต้องทำการเซตขานั้นให้เป็นระดับสูงเสียก่อน ซึ่งจะทำให้ตัวขับ FETS ทรานซิสเตอร์หยุดทำงาน ดังนั้นขาของ Port 1,2 และ 3 ที่มีตัวต้านทานพูลอัพต่ออยู่จะยกระดับแรงดันและทนกระแสเพียงพอกับระดับลอจิก TTL แต่จะสามารถลดระดับขาให้ต่ำด้วยแหล่งวงจรลอจิกภายนอก ด้วยคุณสมบัติของขา 1,2 และ 3 นี้ จะเรียกว่า Quasi-bidirectional สำหรับ Port 0 ค่า 1 ใน Port ที่เลทช์ไว้จะเป็นเหตุให้ขาภายนอกถูกปลดออกหรือลอยตัว ปกติ Port ทั้งหมดของ MCS-51 จะมีค่าเป็น '1' หลังการ Reset แล้ว ถ้าค่า '0' ถูกเขียนเข้าไปเลทช์ไว้ที่ Port มันสามารถจะถูกกำหนดใหม่ให้เป็น Input ด้วยการเขียนค่า '1' เข้าไปใหม่

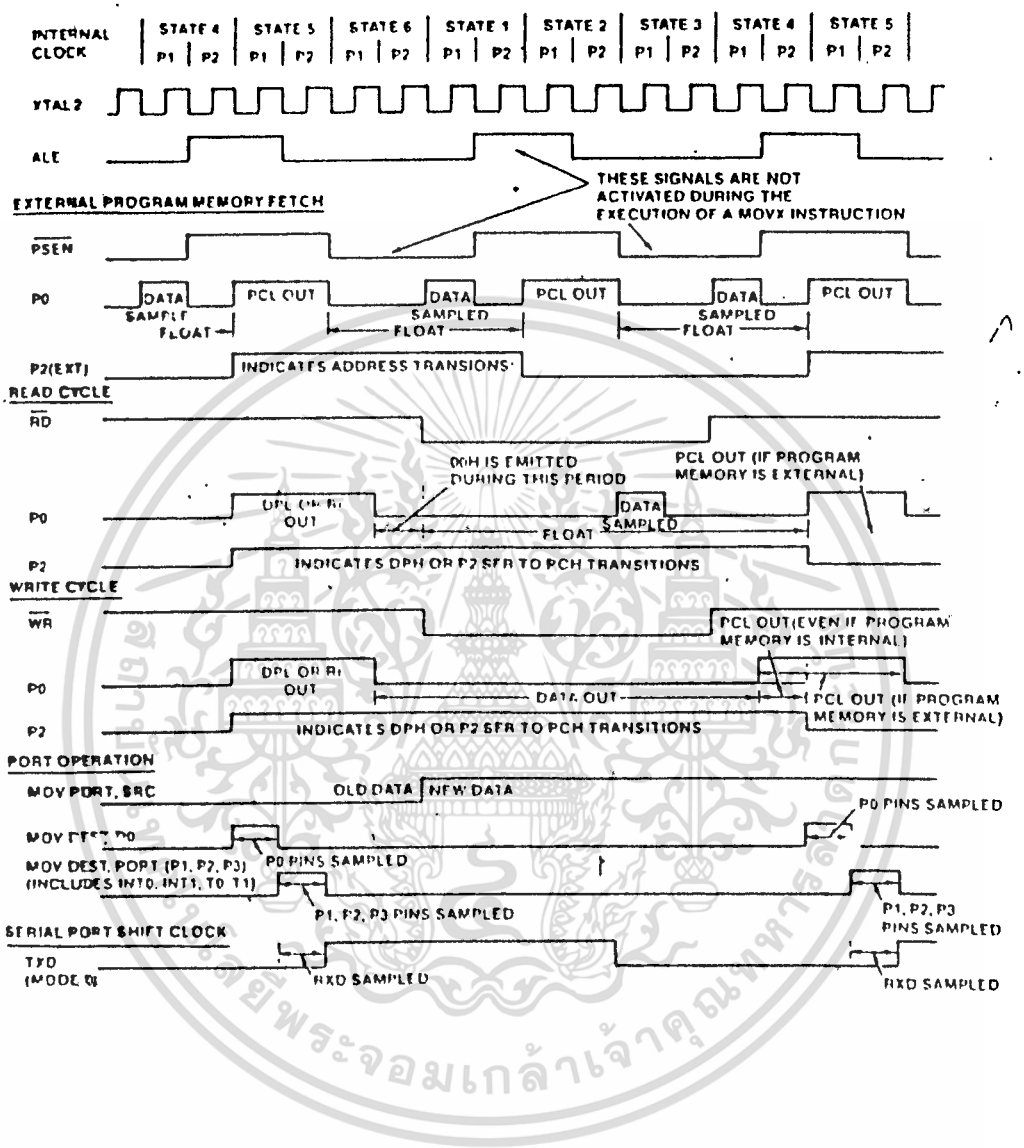
Port 0 มีความแตกต่างจาก Port อื่นที่ไม่มีพูลอัพภายในแต่มี FET ตัวบนมาต่อแทนเป็นตัว Driver Output ดังรูปที่ 2.11(A) โดยถ้ามันทำให้วงจรทำงานก็จะเป็นการทำงานแบบ Address Bus และข้อมูล ด้วยการเข้าถึงข้อมูลภายนอกด้วยสัญญาณควบคุมภายใน แต่งานลักษณะนี้ปกติ FET จะไม่ทำงาน ในการใช้ Register P0 เลทช์ค่า '1' ไว้ตามวงจรเกทของ FET ตัวล่างจะต่อที่ Q ซึ่งเป็นลอจิกระดับต่ำ ทำให้ FET ทั้งไม่ทำงานทั้งคู่ จะเป็นการปล่อยขาของ Port 0 ให้ลอยตัวเป็นการให้ Input มีค่า Impedance สูงนั่นเองขณะที่ Port P2 ใน SFR จะมีค่า Address ที่ไม่เปลี่ยนแปลง



รูปที่ 2.11 เป็น Port การกำหนดเลทช์และบัฟเฟอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.8.3 การเขียนไปยัง Port

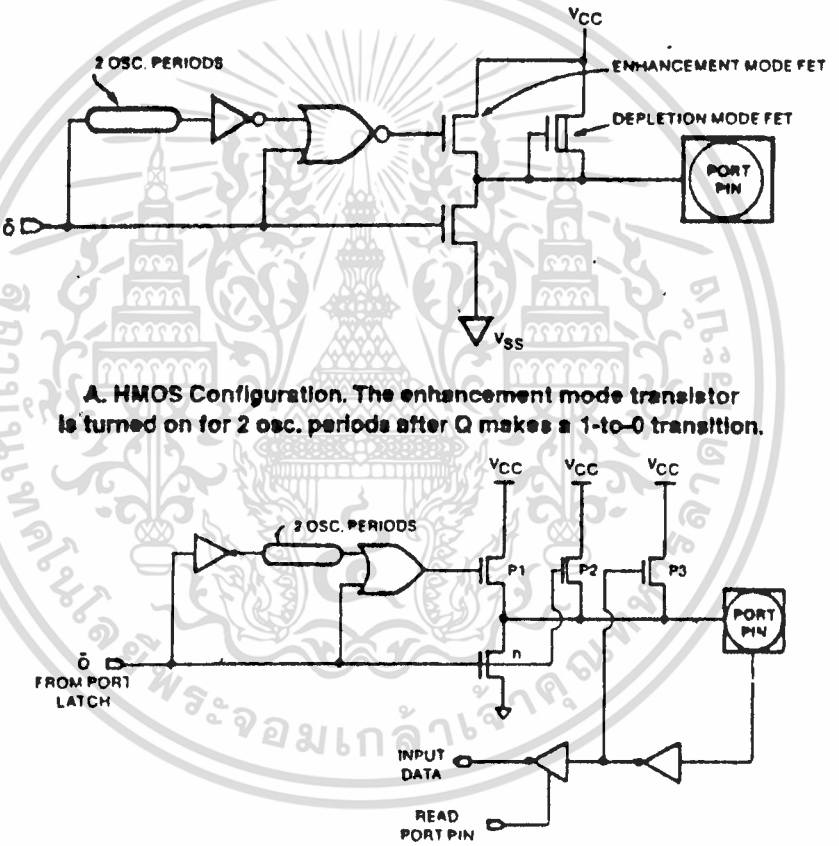


รูปที่ 2.12 วัฏจักรการอ่านและเขียนไปยัง Port

การทำงานตามคำสั่งที่เปลี่ยนค่าในแลทช์ของแต่ละPortค่าใหม่จะเข้ามาเก็บในช่วงระหว่าง S6P2 ของวัฏจักรสุดท้ายของคำสั่ง อย่างไรก็ตาม Port จะเก็บค่าในแลทช์ เมื่อมีการใช้ส่งข้อมูลออกที่ Output Buffer ระหว่าง เฟส 1 ของคาบเวลาใดๆของสัญญาณนาฬิกา (ส่วนระหว่าง เฟส 2 Output Buffer จะยังคง เก็บค่า เริ่มแรกที่ปรากฏใน เฟส 1 ก่อนหน้านั้น) โดยลำดับค่าใหม่ที่แลทช์ไว้จะยังไม่ปรากฏที่ขาของ Port จนกว่าจะถึง เฟส 1 ตัวใหม่ ซึ่งอยู่ในช่วง S1P1 ของวัฏจักรแมคชีนตัวต่อมาดังรูปที่ 2.12

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถ้าการเปลี่ยนจาก '0' เป็น '1' ของ Port 1,2 หรือ 3 จะเป็นการให้พูลอัพทำงานในช่วงระหว่าง S1P1 และ S1P2 ของวัฏจักรเกิดการเปลี่ยนแปลง ลักษณะงานเช่นนี้จะช่วยเพิ่มความเร็วของการเปลี่ยนแปลง การมีพูลอัพเพิ่มขึ้นสามารถที่จะจ่ายกระแสได้เพิ่มขึ้น 100 เท่าตัวของการพูลอัพปกติ จะสังเกตุได้ว่าการพูลอัพภายในเป็นเฟตซึ่งไม่ใช่ตัวต้านทานเส้นตรง การจัดการพูลอัพมีแสดงในรูปที่ 2.13



รูปที่ 2.13 การจัดการพูลอัพภายในเริ่มแรกของ Port 1,2 และ 3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในพวก HMOS8051 ส่วนที่คงที่ในการพูลอัพคือการใช้ทรานซิสเตอร์แบบ Depletion Mode ด้วยการต่อเกตเข้าที่แหล่งกำเนิดแรงดัน ตัวทรานซิสเตอร์แบบนี้ที่ขาของ Port จะจ่ายกระกระแสได้ประมาณ 0.25 mA. เมื่อเกิดการสัดวงจรลงดิน ในการต่อขนานกับพูลอัพคงที่ของทรานซิสเตอร์แบบ Enhancement Mode ดังวงจรรูปที่ 2.13(A) ซึ่งจะ Active ระหว่าง S1 เมื่อไรก็ตามที่โค้ด เกิดการเปลี่ยนแปลงจาก '0' เป็น '1' ระหว่างช่วงเวลานี้ถ้า Port เกิดสัดวงจรลงดินจะทนกระแสได้ถึง 30 mA.

ส่วนในพวก CHMOS พูลอัพจะประกอบด้วย pFET3 ตัวโดยที่ pFET จะเป็นพวกP-Channel ซึ่งจะทำงานเมื่อเกตของ FET มีค่าเป็น '0' และ pFET จะหยุดทำงานเมื่อเกตมีค่า '1' ส่วน nFET จะทำงานตรงกันข้ามกับ pFET

pFET ตัวที่ 1 ในรูปที่ 2.13(B) ทรานซิสเตอร์จะทำงานคลุมระยะ 2 คาบ Oscillator หลังจากมีการเปลี่ยนแปลงค่าใน Port แลทช์จาก '0' เป็น '1' ขณะที่ทำงานมันจะทำให้pFET3 ทำงานผ่าน Inverster Inverster ตัวนี้กับ pFET ที่ต่อจาก Port แลทช์จะเก็บค่า '1' ไว้ จะสังเกตุได้ว่าขาที่มีค่าเป็น 1 ถ้ามี Glitch ลงจากภายนอกเข้ามาก็สามารถที่จะทำให้ pFET3 หยุดทำงานได้ทำให้ขา Port นี้อยู่ในสภาวะลบ pFET2 จะทำงาน ในขณะเดียวกันก็ทำให้ nFET หยุดทำงาน ดังนั้นการเก็บค่า 1 ใน pFET3 ก็จะอ่อนลงและและจะสูญข้อมูลที่เก็บไว้ได้เมื่อมี Glitch เข้าที่ขานี้

2.8.3 การบรรจุข้อมูลและการต่อพ่วงกับ Port

Output ของ Port 1,2 และ 3 สามารถขับ TTL Input แบบ LS ได้ 4 ตัว Port เหล่านี้ของ HMOS สามารถขับได้ทั้งวงจรแบบ TTL หรือ NMOS MCS-51 ทั้งแบบ HMOS และ CHMOS สามารถทำงานเป็นตัวขับลักษณะ Open-Collector และ Open-Drain โดยไม่ต้องมีพูลอัพจากภายนอก

Port 0 ตัว Output Buffer ที่ขาสามารถขับ TTL Input ได้ 8 ตัว อย่างไรก็ตามขาเหล่านี้ถ้าใช้ในการขับ Input แบบ NMOS ต้องต่อพูลอัพจากภายนอกยกเว้นการใช้งานเป็น Address Bus/ข้อมูล

รูปแบบการอ่าน - เปลี่ยนค่า - เขียน (Read-Modify-Write) ผ่านPort

ด้วยลักษณะการใช้ Port เป็น Input ตามโครงสร้างรูป 2.11 บิตแลทช์แต่ละบิต Port ของ SFR จะแทนด้วยวงจร D Flip-Flop ซึ่งจะถูกกระตุ้นให้เก็บค่าจาก Bus ภายในด้วยสัญญาณ 'Write to latch' ที่ถูกสร้างขึ้นด้วย CPU ค่า Q ที่เป็น Output ของ Flip-Flop ก็จะจ่ายไปยัง Bus ภายในที่ถูกควบคุมสัญญาณ Read Latch จาก CPU ระดับสัญญาณที่ขา Port จะไปปรากฏที่ Bus ภายในได้นั้นจะถูกควบคุมด้วยสัญญาณ Read Pin จาก CPU ดังนั้นบางคำสั่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะสร้างสัญญาณ Read Pin ซึ่งควรจะแยกชนิดของคำสั่งทั้งสองนี้ได้ออกด้วยความเข้าใจว่าถ้าเป็นชนิดแบบ Read Latch การทำงานของคำสั่งชนิดนี้จะมีลักษณะด้านการอ่านค่าผ่านเข้า Port แล้วทำการเปลี่ยนแปลงค่าที่อ่านเข้ามาแล้วทำการแลทช์ไว้ ซึ่งลักษณะงานเช่นนี้จะเป็นการอ่านค่าเปลี่ยนค่า และเขียนกลับไปที่แลทช์ใหม่เป็นกลุ่มคำสั่ง ' Read-Modify-Write ' ซึ่งเป็นตัวอเปอร์แรนด์ที่อ่านเข้ามาเป็น Port หนึ่งไบต์หรือบิตพอร์ต คำสั่ง Read-Modify-Write เหล่านี้จะแลทช์ข้อมูลมากกว่าที่จะส่งไปที่ขาโดยตรง ดังคำสั่งต่อไปนี้

ANL	(AND เช่น ANL P1,A)
ORL	(OR เช่น ORL P2,A)
XRL	(EXOR เช่น XRL P3,A)
JBC	(JUMP ถ้าบิต =1 และเคลียร์บิต เช่น JBC P1.1,LABLE)
CPL	(Complement Bit เช่น CPL P3.0)
INC	(Increment Data เช่น INC P2)
DEC	(Decrement Data และกระโดด ถ้าข้อมูลไม่เป็นศูนย์ เช่น DJNZ P3,LABEL)
MOV PX.Y,C	(ย้ายบิตทดไปบิต Y ของ Port X)
CLR PX.Y	(เคลียร์บิต Y ของ Port X)
SET PX.Y	(เซตบิต Y ของ Port X)

สามคำสั่งสุดท้ายไม่อาจสรุปได้ว่าเป็นคำสั่งแบบ Read-Modify-Write เพราะมันจะทำงานด้วยการอ่านทั้ง 8 บิตหรือไบต์จาก Port แทนที่จะเป็นบิตเดียวซึ่งทั้ง 8 บิตจะเซตค่า Address ใหม่ ดังนั้นจึงเป็นการเขียนค่าตัวใหม่ทั้ง 8 บิตกลับเข้าที่แลทช์

เหตุผลหนึ่งที่คำสั่ง Read-Modify-Write เป็นการแลทช์ค่า Port โดยตรงมากกว่าการส่งออกที่ขา เพื่อเป็นการหลีกเลี่ยงความเป็นไปได้ที่อาจจะทำให้การแสดงระดับสัญญาณที่ขา Port เกิดการผิดพลาดขึ้น ตัวอย่างเช่น Port หนึ่งบิตอาจใช้ขับตัวทรานซิสเตอร์ เมื่อค่า '1' ถูกส่งไปที่ขาบิตนั้น ทรานซิสเตอร์จะทำงาน ถ้า CPU อ่านกลับมาใน Port เดียวกันที่ขามากกว่าที่แลทช์ มันจะอ่านเอาแรงดันเบสของตัวทรานซิสเตอร์นั้นกลับเข้าไป ซึ่งก็จะกลายเป็น '0' แทน ดังนั้นการใช้วิธีอ่านค่าจากตัวแลทช์กลับไปก็จะได้ค่าที่ถูกต้องคือ '1'

2.9 การเข้าถึงหน่วยความจำภายนอก

ลักษณะการเข้าถึงหน่วยความจำภายนอกมี 2 แบบคือ การเข้าถึงหน่วยความจำโปรแกรมภายนอกและการเข้าถึงหน่วยความจำข้อมูลภายนอก การเข้าถึงหน่วยความจำโปรแกรมภายนอกจะใช้ขาสัญญาณ PSEN (Program Store Enable) Active ต่ำเป็นสโตรบควบคุมการอ่านและ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเข้าถึงหน่วยความจำข้อมูลภายนอกจะใช้ขา RD และ WR Ative ต่ำเป็นสัญญาณสำหรับควบคุมหน่วยความจำ

การเฟทช์โปรแกรมภายนอกจะใช้ขา Address 16 บิต เสมอ ส่วนการเข้าถึงหน่วยความจำข้อมูลสามารถที่จะใช้กำหนดเลขที่อยู่ได้ทั้ง 16 บิต Address เช่นMOVX @DPTR หรือ 8 บิต Address เช่น MOVX @Ri

เมื่อใดที่ใช้ 16 บิต Address ไบต์สูงของ Address จะถูกส่งออกที่ Port 2 และจะคงสถานะค่านั้นตลอดในช่วงวัฏจักรการอ่านและเขียน ระหว่างช่วงเวลานี้ตัวแลทช์ของ Port 2 ใน SFR จะไม่ต้องประกอบด้วยค่า '1' และค่าข้อมูลใน SFR จะไม่มีการเซต ถ้าช่วงวัฏจักรการใช้หน่วยความจำภายนอกไม่มีการเข้าถึงข้อมูลในวัฏจักรต่อมาค่าใน SFR ของ Port 2 จะปรากฏค่าเดิมกลับมาใหม่ในวัฏจักรตัวต่อมานี้

ในกรณีใช้เป็น 8 บิต Address ค่าใน SFR ของ Port จะยังคงค่าเดิมที่ขา Port 2 ตลอดช่วงวัฏจักรการใช้ความจำภายนอก ซึ่งลักษณะนี้จะเป็นการใช้งานด้านเพจของหน่วยความจำ

ในกรณีใช้ Address ไบต์ต่ำเป็นช่วงเวลามัลติเพล็กซ์กับข้อมูลของ Port 0 ขาสัญญาณ Address/ข้อมูล จะขับ FET ทั้ง 2 ตัวใน Port 0 เป็นบัฟเฟอร์ส่งข้อมูลออก ดังนั้นในการใช้ Port 0 จะไม่มีการรับกระแสเข้าจึงไม่จำเป็นต้องพูลอัพจากภายนอก สัญญาณ ALE : Address Latch Enable ก็จะใช้เป็นขาควบคุมรับไบต์ Address เก็บไว้ภายนอก ซึ่งค่า Address จะคงที่ที่ช่วงขอบขาลงของ ALE ดังนั้นในวัฏจักรการเขียนข้อมูลจะถูกเขียนออกไปที่ Port 0 ก่อนที่ WR จะ Active ต่ำ ส่วนวัฏจักรการอ่านข้อมูลจะรับเข้าที่ Port 0 ก่อนสัญญาณการอ่านจะปรากฏเล็กน้อยและระหว่างการเข้าถึงของหน่วยความจำภายนอก ตัว CPU จะส่งค่า OFFH มาเก็บไว้ที่ Port 0 ของ SFR

การใช้หน่วยความจำโปรแกรมภายนอก จะขึ้นอยู่กับสองกรณีคือ

1. เมื่อใดก็ตามที่ EA Active หรือ
2. เมื่อใดก็ตามที่ตัวนับโปรแกรม PC ประกอบด้วยตัวเลขที่มีค่ามากกว่า OFFFH และ 1FFFH สำหรับ 8052

ในรุ่นที่ไม่มี ROM ในตัวให้ใช้ค่า Active ต่ำป้อนที่ขา EA เพื่อกำหนดเฟทช์โปรแกรม 4 Kbyte ได้

เมื่อโปรแกรมหน่วยความจำภายนอกถูกใช้งานทั้ง 8 บิต ของ Port 2 จะส่งค่า Address ออกมาด้วยทำให้ไม่สามารถใช้งานเป็น Port I/O ได้ในระหว่างการเฟทช์โปรแกรมภายนอก เพราะจะส่งค่าไบต์สูงจาก PC ออกมาที่ Port 2 นี้ และระหว่างการเข้าถึงของข้อมูลภายนอกจะใช้ Port 2 เป็นตัวส่ง Address ไบต์สูงจาก DPH ใน SFR ขึ้นอยู่กับการใช้คำสั่งว่าใช้แบบให้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

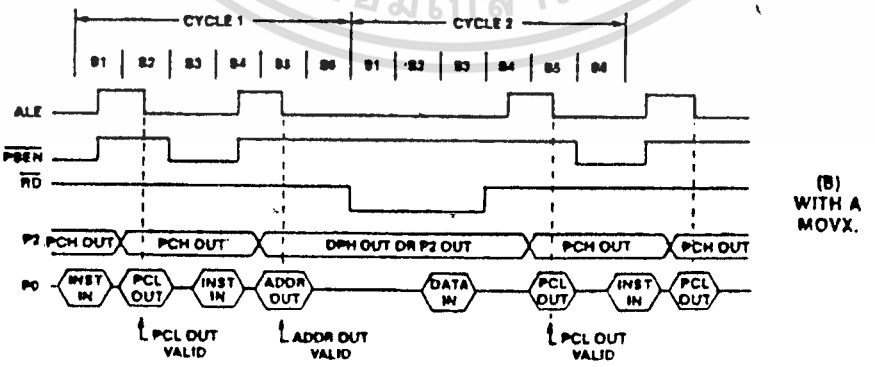
คำสั่งส่ง Output ออกจาก DPH ในการกำหนด Address ข้อมูลภายนอกก็จะใช้คำสั่ง MOVX @DPTR หรือถ้าใช้แบบให้ข้อมูลส่งข้อมูลออกที่ Port ของ SFR ก็จะใช้คำสั่ง MOVX @Ri

2.9.1 สัญญาณ PSEN

ใช้เป็นการควบคุมเฟทช์การอ่านโปรแกรมจากภายนอก PSEN จะไม่ Active ถ้ามีการเฟทช์โปรแกรมภายใน เมื่อ CPU เข้าถึงการโปรแกรมภายนอก PSEN จะ Active 2 ครั้งในแต่ละช่วงวัฏจักรการเฟทช์ ยกเว้นคำสั่ง MOVX ช่วงเวลาของการที่ PSEN เกิด Active จะไม่เหมือนกับช่วง RD Active ช่วงวัฏจักรการอ่านที่สมบูรณ์จะรวมเอาช่วงที่ ALE Active และ Active ซ้ำลูกที่สองกับสัญญาณการควบคุม RD ที่เกิดพัลส์ต่ำประกอบเข้าด้วยกันซึ่งจะใช้เวลา 12 สถานะคาบเวลา ส่วนช่วงเวลาของ PSEN ที่สมบูรณ์จะรวมเอาช่วงที่ ALE Active และActive ซ้ำลูกที่สองกับสัญญาณควบคุม PSEN ประกอบเข้าด้วยกันซึ่งจะใช้เวลา 6 สถานะคาบเวลา ลักษณะการทำงานตามลำดับของวัฏจักรการอ่านทั้ง 2 แบบ แสดงในรูปที่ 2.14

2.9.2 สัญญาณ ALE

ฟังก์ชั่นหลักของ ALE คือการใช้งานในการให้จังหวะที่แน่นอนในการแลทช์เอาไบต์ต่ำของ Address จาก P0 ไปเก็บไว้ภายนอกเพื่อใช้ในการถอดรหัส Address โปรแกรมภายนอกโดยจะให้ ALE ทำงาน Active 2 ครั้งในทุกๆวัฏจักรแมคชีนสัญญาณนี้จะเกิดขึ้นตลอดแม้ว่าจะไม่ได้เฟทช์จากภายนอก มีเพียงช่วงเวลาเดียวเท่านั้นที่ ALE ไม่เกิดพัลส์คือระหว่างการเข้าถึงหน่วยความจำภายนอก ตามรูปที่ 2.12(B) จะเห็นว่าพัลส์แรกของ ALE ในวัฏจักรที่สองของคำสั่ง MOVX หายไปหรือมีเพียงพัลส์เดียวในหนึ่งคำสั่งลักษณะพัลส์ที่เกิดขึ้นจะคงที่ในอัตรา 1/6 ของสัญญาณความถี่ Oscillator และสามารถนำมาใช้เป็นสัญญาณนาฬิกาภายนอกหรือกำหนดเวลาได้



รูปที่ 2.14 จังหวะการเข้าถึงหน่วยความจำภายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.9.3 การใช้เนื้อที่ของโปรแกรมภายนอกซ้อนกับหน่วยความจำข้อมูล

ในการใช้งานบางครั้งอาจทำงานตามโปรแกรมจากหน่วยความจำ ที่คุณลักษณะตำแหน่งจะซ้อนกันกับการใช้ข้อมูล ใน MCS-51 ตัวโปรแกรมภายนอกและหน่วยความจำข้อมูล สามารถที่จะทำงานร่วมกันได้ด้วยการให้สัญญาณ PSEN และ RD เข้า AND เกตกันก่อนจะเห็นว่าที่ Active ของAND ที่เกิดจากสัญญาณทั้งสองตามรูปที่ 2.14(B) จะทำให้สัญญาณสามารถการอ่าน Active ต่ำในการอ่านและ Active สูงในช่วงการเฟทช์ ซึ่งทำให้ใช้ตำแหน่งหน่วยความจำเลขที่เดียวกันได้ เนื่องจากวัฏจักร PSEN จะเกิดขึ้นเร็วกว่าวัฏจักรการอ่าน (RD) ดังนั้นความเร็วของหน่วยความจำโปรแกรมต้องเร็วพอที่จะให้วัฏจักร PSEN ทำงานได้สมบูรณ์

2.10 ตัวจับเวลา/ตัวนับ (Timer/Counter)

MCS-51 มี 16 บิต จะมีตัวจับเวลา/ตัวนับ 2 ตัวคือ Timer/Counter 0 และ Timer/Counter 1 ส่วน 8032/8052 จะมีเพิ่มอีก 1 ชุดคือ Timer/Counter 2 ขณะที่แต่ละตัวจับเวลา/ตัวนับ (Timer/Counter) สามารถกำหนดให้ทำงานได้เป็น ตัวจับเวลา/ตัวนับ

2.10.1 ตัวจับเวลา/ตัวนับ 0 และ ตัวจับเวลา/ตัวนับ 1

แต่ละตัวจะถูกกำหนดให้ทำงานเป็นตัวจับเวลาหรือตัวนับ ได้ด้วยการเซตหรือเคลียร์บิตที่ตัวควบคุมใน Register TMOD ในกลุ่ม SFR

ในฟังก์ชั่นตัวจับเวลาตัว Register จะเพิ่มค่าทุกวัฏจักรแมคชีนดังนั้นตัวเลขในRegister จะเป็นจำนวนของวัฏจักรแมคชีน เนื่องจากแต่ละวัฏจักรแมคชีนประกอบด้วย 12 คาบOscillator อัตราการนับแต่ละครั้งจะกินเวลาประมาณ 1/12 ของความถี่ Oscillator

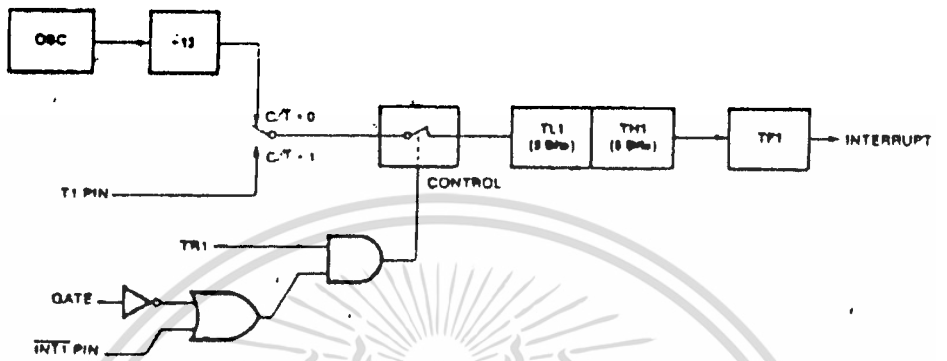
ในฟังก์ชั่นตัวนับตัว Register จะเพิ่มค่าทุกครั้งที่มีการเปลี่ยนแปลงสถานะจาก '1' เป็น '0' ที่เข้ามาที่ขา T0 หรือ T1 ในฟังก์ชั่นนี้สัญญาณภายนอกที่เข้ามาจะถูก Sampling ระหว่างช่วง S5P2 ของทุกวัฏจักรแมคชีน โดยถ้า Sampling สัญญาณเข้าเป็นระดับสูงในวัฏจักรหนึ่งดังนั้นถ้าในวัฏจักรตัวต่อมาของสัญญาณเข้าเป็นระดับต่ำ Register จะนับเพิ่มหนึ่งค่า โดยที่ค่าใหม่ของตัวนับจะปรากฏที่ Register ช่วง S3P1 ของวัฏจักร ซึ่งค่าหนึ่งที่รับเข้าไปจะใช้ช่วง 2 วัฏจักรแมคชีน (เท่ากับ 24 คาบ) ในการรับค่าช่วงการเปลี่ยน '1' เป็น '0' ดังนั้นค่าสูงสุดในการนับจะมีอัตรา 1/24 ของความถี่ Oscillator และสัญญาณ Input ที่นับนั้นจะไม่มีช่วงระยะห่าง ที่เน่นอนของ Duty Cycle แต่จะถูกนับ เมื่อระดับแรงดันที่ถูก Sampling ในแต่ละครั้งจะต้องมีช่วงคงที่อย่างน้อย 1 วัฏจักรแมคชีนก่อนที่จะเปลี่ยนค่าระดับแรงดันใหม่

ในการเลือกทำงานระหว่างตัวนับกับตัวจับเวลา จะเลือกได้ 4 โหมดคือ โหมด 0,โหมด 1 และโหมด 2 เลือกได้ทั้งสองตัวของ Timer/Counter ส่วนโหมด 3 จะทำงานแตกต่างออกไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โหมด 0

การใช้งานตัวจับเวลา/ตัวนับ 0 หรือ 1 ให้อยู่ในโหมด 0 จะทำงานคล้ายกับของ MCS-48 โดยตัวจับเวลาของ MCS-48 มี 8 บิต มีตัว Prescaler เป็นตัวหาร 12 รูปที่ 2.15 แสดงการทำงานในโหมด 0 ของตัวจับเวลา/ตัวนับ 1



รูปที่ 2.15 แสดงการทำงานในโหมด 0 ของตัวจับเวลา/ตัวนับ 1 ขนาด 13 Bit

ในโหมดนี้ Register ตัวจับเวลาจะถูกกำหนดให้มี 13 บิต ด้วยการนับขึ้น เมื่อเป็น '1' หมดทุกบิตจะกลับมาที่ '0' ทุกบิตใหม่จะเป็นการเกิด Overflow ไปทำให้เฟลกอินเตอร์รัพต์ TF1 ปรับเป็น '1' การควบคุมให้เริ่มนับตัว Input จะควบคุมด้วยการอีนาเบิ้ล TR1=1,GATE=0 และขา INT1=1 การปรับ GATE=1 เป็นการตั้งตัวนับให้ถูกควบคุมด้วยสัญญาณจากภายนอกเข้าขา INT1 TR1 จะเป็นบิตควบคุมใน Register TMOD ของ SFR

Register ตัวนับจะมี 13 บิต ประกอบด้วย TH1 8 บิต และ TL1 อีก 5 บิตอันดับต่ำส่วนอีก 3 บิต ที่เหลืออยู่ในอันดับสูงของ TL1 จะไม่ใช้ การปรับเฟลก TR1 ให้ทำงานจะไม่เคลียร์ค่าใน Register

การทำงานในโหมด 0 ในตัวจับเวลา/ตัวนับ 0 จะทำงานเหมือนกับตัวจับเวลา/ตัวนับ 1 โดยใช้ TR0 และ INT0 รวมการควบคุมแทนสัญญาณต่างๆ ในรูปที่ 2.15 มีความแตกต่างในการควบคุมคือ บิตของ GATE ทั้งสองตัวหนึ่งจะแทนตัวจับเวลา/ตัวนับ 1 (TMOD.7) และอีกตัวหนึ่งจะแทนตัวจับเวลา/ตัวนับ 0 (TMOD.3)

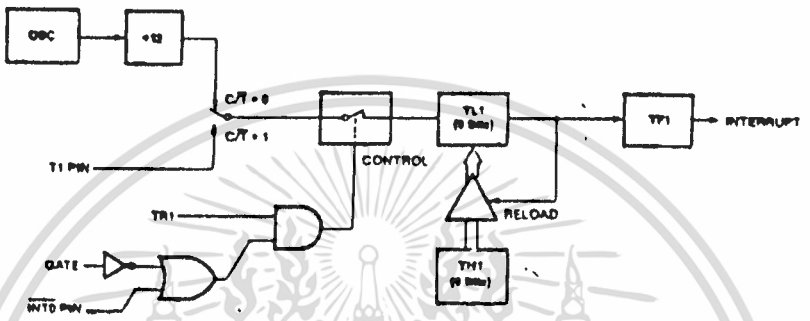
โหมด 1

โหมด 1 ทำงานเหมือนกันโหมด 0 ต่างกันแต่เฉพาะการใช้ Register ตัวจับเวลา/ตัวนับจะทำงานนับด้วยขนาด 16 บิตโดยไม่มี Prescaler คือความถี่ที่ Oscillator เป็นความถี่ที่เข้ามาถูกหารด้วยค่า 16 บิต ใน Register ตัวนับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โหมด 2

โหมด 2 มีการทำงานโดยการกำหนดให้ตัวนับ 8 บิต ของ TL1 และจะโหลดใหม่โดยอัตโนมัติทุกครั้ง เมื่อมีการ Overflow จาก TL1 ดังรูปที่ 2.16 ไม่เพียงแต่ TF1 จะปรับเป็น '1' แต่ TL1 จะถูกโหลดโดยอัตโนมัติจากค่าที่ตั้งไว้ใน TH1 ซึ่งค่าใน TH1 สามารถจะตั้งค่าได้ด้วย Software คือการใช้คำสั่ง MOV และบรรจุเข้าไปใหม่ที่ TL1 ทุกครั้งที่เกิด Overflow THO และ TFO จะเป็นตัวร่วมการทำงานในโหมดนี้



รูปที่ 2.16 ตัวจับเวลา/ตัวนับ 1 ทำงานในโหมด 2 แบบโหลดใหม่ 8 Bit

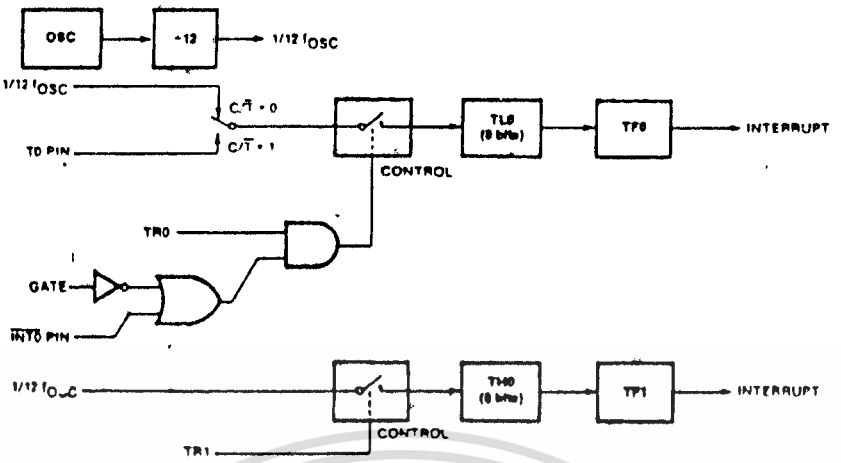
โหมด 3

ใช้ตัวจับเวลา/ตัวนับ 1 ในโหมด 3 มีการทำงานเป็นตัวนับมีผลเช่นเดียวกับการตั้ง TR1=0 และใช้ตัวจับเวลา/ตัวนับ 0 ในโหมด 3 จัดการให้ TL0 และ THO เป็นตัวนับสองตัวแรกที่แยกออกจากกัน วงจรตรรกควบคุมสำหรับโหมด 3 ที่ใช้ตัวจับเวลา/ตัวนับ 0 แสดงในรูปที่ 2.17 TL0 ใช้ตัวจับเวลา/ตัวนับ 0 เป็นบิตควบคุมของ C/T,GATE,TR0,INT0 และ TF0 ตัว THO ถูกล็อคให้ทำงานในฟังก์ชั่น รูปที่ 2.17 ใช้ตัวจับเวลา/ตัวนับ 1 ในโหมด 3 เป็นตัวนับขนาด 8 บิต 2 ตัว

ตัวจับเวลา (เป็นตัวนับวัฏจักรแมคชีนได้) และจะใช้บิตที่ TR1 และ TF1 ของตัวจับเวลา 1 เป็นตัวควบคุม ดังนั้นจึงใช้ THO เป็นตัวจับเวลาเป็นการควบคุมอินเตอร์รัพต์ด้วยตัวจับเวลา 1

โหมด 3 จะใช้งานในความต้องการตัวจับเวลา/ตัวนับขนาด 8 บิต เพิ่มขึ้นด้วยการใช้ตัวจับเวลา 0 ในโหมด 3 ดังนั้น 8051 สามารถที่จะทำงานใช้ตัวจับเวลา/ตัวนับได้เป็น 3 ชุด ขณะที่ 8052 ก็จะใช้งานได้ 4 ชุด เมื่อใช้ตัวจับเวลา 0 อยู่ในโหมด 3 ตัวจับเวลา 1 สามารถที่จะปิด-เปิดให้เข้าสู่หรือออกจากโหมด 3 หรือสามารถที่จะยังคงใช้ตัว Generator ที่สร้างอัตราบ๊อดของการส่งข้อมูลอนุกรมหรือการใช้งานใดๆที่ไม่ต้องการการอินเตอร์รัพต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.17 ให้ตัวจับเวลา/ตัวนับ 1 ในโหมด 3 เป็นกลุ่มตัวนับขนาด 8 Bit สองตัว

TMOD : Timer/Counter Mode Control Register

GATE	C/T	M1	MO	GATE	C/T	M1	MO
<--------Timer R1---------->				<----------Timer R0-------->			

GATE : ควบคุมเกท เมื่อเซตเป็น '1' จะเป็น Enable ตัวจับเวลา/ตัวนับเท่านั้น ขณะที่ขา INTx มีสถานะสูงและขาควบคุม TRx ใน TCON จะถูกเซตเป็น '1' เมื่อตัวนับภายในถูกเคลียร์ให้ Enable เมื่อไรก็ตามที่บิตควบคุม TRx ถูกเซตเป็น '1'

C/T : เลือกการทำงานแบบตัวจับเวลา/ตัวนับ ถ้าเป็น '0' จะเลือกทำงานเป็นตัวจับเวลา (โดยสัญญาณนาฬิกาภายในเป็นสัญญาณเข้าอ้างอิงถึง) ถ้าเป็น '1' จะทำงานแบบตัวนับและรับสัญญาณเข้าที่ขา Tx

M1	MO	การทำงาน
0	0	ทำงานแบบตัวจับเวลาของ MCS-48 ใช้ TLx เป็นตัวป้อนบิตอีก 5 บิต
0	1	การใช้ตัวจับเวลา/ตัวนับขนาด 16 บิตจะใช้ THx และ TLx เป็นตัวนับ ไม่มี Prescaler
1	0	การโหลดขนาด 8 บิตโดยอัตโนมัติที่ตัวจับเวลาและตัวนับ โดยใช้ THx เก็บ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ค่าที่ตั้งไว้และจะถ่ายไปที่ TLx ใหม่ทุกครั้งที่เกิด Overflow คือ TLx ถูกนับเป็น '0'

1 1 ตัวจับเวลา 0 ทำงานโดยให้ TL0 และ TH0 เป็นตัวนับแยกกัน

TCON : Timer/Counter Control Register

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1 TCON.7 ตัวจับเวลา 1 แฟลกเป็น '1' เมื่อเกิด Overflow ถูกเซตเป็นหนึ่งด้วย Hardwaer ทางสัญญาณ เมื่อตัวจับเวลา/ตัวนับ Overflow และจะเคลียร์ตัวเองเมื่ออินเตอร์รัพต์ไปแล้ว

TR1 TCON.6 ตัวจับเวลา 1 เป็นตัวควบคุมบิตให้เริ่มทำงาน จะเซตหรือเคลียร์ด้วย Software ที่จะมาทำให้ ตัวจับเวลา/ตัวนับ 1 เริ่มหรือหยุดการทำงาน

TF0 TCON.5 ตัวจับเวลา 0 แฟลกเป็น '1' เมื่อเกิด Overflow ถูกเซตเป็นหนึ่งด้วย Hardwaer ทางสัญญาณ เมื่อตัวจับเวลา/ตัวนับ Overflow และจะเคลียร์ตัวเองเมื่ออินเตอร์รัพต์ไปแล้ว

TR0 TCON.4 ตัวจับเวลา 0 เป็นตัวควบคุมบิตให้เริ่มทำงาน จะเซตหรือเคลียร์ด้วย Software ที่จะมาทำให้ ตัวจับเวลา/ตัวนับ 0 เริ่มหรือหยุดการทำงาน

IE1 TCON.3 อินเตอร์รัพต์ 1 เป็นแฟลกขอบสัญญาณเซตด้วย Hardware เมื่อสัญญาณขอบการอินเตอร์รัพต์ปรากฏเข้าที่ขา INT1 และเคลียร์เมื่อการทำงานอินเตอร์รัพต์สิ้นสุด

IT1 TCON.2 อินเตอร์รัพต์ 1 รูปแบบการควบคุมบิต จะเซตหรือเคลียร์ด้วย Software ที่จะเป็นตัวกำหนดให้การกระตุ้นอินเตอร์รัพต์จากภายนอกที่ขอบขาลง หรือระดับแรงดันต่ำ

IE0 TCON.1 อินเตอร์รัพต์ 0 เป็นแฟลกขอบสัญญาณเซตด้วย Hardware เมื่อสัญญาณขอบการอินเตอร์รัพต์ปรากฏเข้าที่ขา INT0 และเคลียร์เมื่อการทำงานอินเตอร์รัพต์สิ้นสุด

IT0 TCON.0 อินเตอร์รัพต์ 0 รูปแบบการควบคุมบิต จะเซตหรือเคลียร์ด้วย Software ที่จะเป็นตัวกำหนดให้การกระตุ้นอินเตอร์รัพต์จากภายนอกที่ขอบขาลง หรือระดับแรงดันต่ำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.10.2 ตัวจับเวลา/ตัวนับ 2

ตัวจับเวลา/ตัวนับ 2 เป็นตัวจับเวลาและตัวนับขนาด 16 บิต แบบบรรจุเข้าอัตโนมัติและใช้ Register ควบคุมของ SFR เป็น T2CON

เมื่อตัวจับเวลา/ตัวนับ 2 ทำงานเป็นตัวจับเวลา Register ตัวจับเวลา/ตัวนับ 2 จะเพิ่มค่าทุกครั้งในแต่ละวัฏจักรแมคชีน ขณะที่ใช้เป็นตัวนับ ตัวจับเวลา/ตัวนับ 2 จะเพิ่มค่าขึ้นอยู่กับการเปลี่ยนสถานะจาก '1' เป็น '0' ที่เข้ามาที่ T2 (P1.0) สัญญาณอินพุทถูก Sampling ที่ S3P2 ของทุกวัฏจักรแมคชีนในการใช้ฟังก์ชั่นให้ Active เพราะฉะนั้นเมื่อ Sample แสดงสถานะสูงในช่วงวัฏจักรแมคชีนหนึ่งและระดับต่ำในอีกวัฏจักรหนึ่ง การนับจะเพิ่มขึ้นหนึ่ง ค่าตัวใหม่จะปรากฏที่ Register ตัวจับเวลา/ตัวนับ 2 ช่วง S3P1 ของแต่ละวัฏจักร เป็นการนับหนึ่งที่สัญญาณการเปลี่ยนแปลงถูกกระตุ้น ดังนั้นอัตราการนับสูงสุดจะเป็น 1/24 ของความถี่ Oscillator ไม่มีการกำหนดความกว้างของพัลส์ของสัญญาณที่เข้ามาจากภายนอก แต่ต้องแน่ใจว่าระดับหนึ่งที่ถูก Sample อย่างน้อยจะต้องคลุมหนึ่งวัฏจักรแมคชีนก่อนที่มันจะเปลี่ยนระดับใหม่

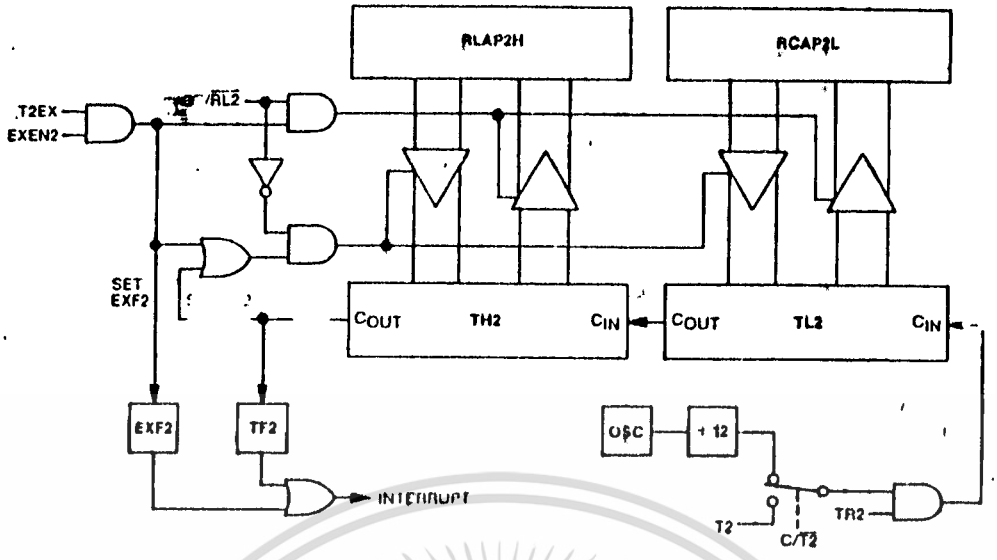
โหมดการทำงานของตัวจับเวลา/ตัวนับ 2 มีสามโหมดด้วยกันคือ แบบธรรมดา Capture แบบบรรจุใหม่อัตโนมัติและเป็นตัวสร้างอัตราบ๊อด ซึ่งสามารถเลือกการทำงานโดยเซตค่าต่างๆใน T2CON

วิธีตั้งให้ทำงานในโหมด Capture จะมีโอกาสให้เลือกการทำงานสองอย่าง ด้วยการเลือกตั้งค่าที่บิต EXEN ของ T2CON โดยตั้ง EXEN2=0 ก็จะเป็นการเลือกตัวจับเวลา/ตัวนับ 2 เป็นตัวจับเวลาหรือตัวนับขนาด 16 บิต ซึ่งเมื่อเกิด Overflow ก็จะไปเซตค่าที่ TF2 ซึ่งเป็น Overflow บิตของตัวจับเวลา 2 ที่สามารถสร้างให้มีการใช้อินเตอร์รัพต์ที่เกิดจากตัวจับเวลาตัวนี้ได้

แต่ถ้า EXEN2=1 ตัวจับเวลา/ตัวนับ ยังคงทำงานเช่นเดิม เพียงแต่เพิ่มการทำงานตรวจสอบสัญญาณจากภายนอกที่เข้ามาที่ขา T2EX ในสภาวะที่เกิดการเปลี่ยนแปลงจาก '1' เป็น '0' จะเป็นเหตุให้จะมีการย้ายค่าข้อมูลใน TL2 และ TH2 ในขณะนั้นเข้าสู่ Register RCAP2L และ RCAP2H ตามลำดับ โดยที่ Register ทั้งสองจัดอยู่ในกลุ่ม DFR ที่มีแต่เฉพาะในเบอร์ 8032/8052 เท่านั้น ในการเปลี่ยนสภาวะที่ n T2EX เช่นนี้ จะทำให้บิต T2EX ใน Register T2CON เซตค่าเป็นหนึ่งและ EXF2 ก็จะเป็นเช่นเดียวกับบิต TF2 คือสามารถนำไปใช้เป็นตัวกระตุ้นให้เกิดการอินเตอร์รัพต์ จากเหตุการณ์ที่เกิดขึ้นเช่นนี้เค้าซึ่งได้ลักษณะการทำงานในโหมด Capture ดังแสดงในรูปที่ 2.18

วิธีตั้งให้ทำงานในโหมดบรรจุใหม่อัตโนมัติก็มีให้เลือกทำงานได้ 2 อย่างเช่นกันด้วยการปรับค่าในบิต EXEN ของ T2CON โดย

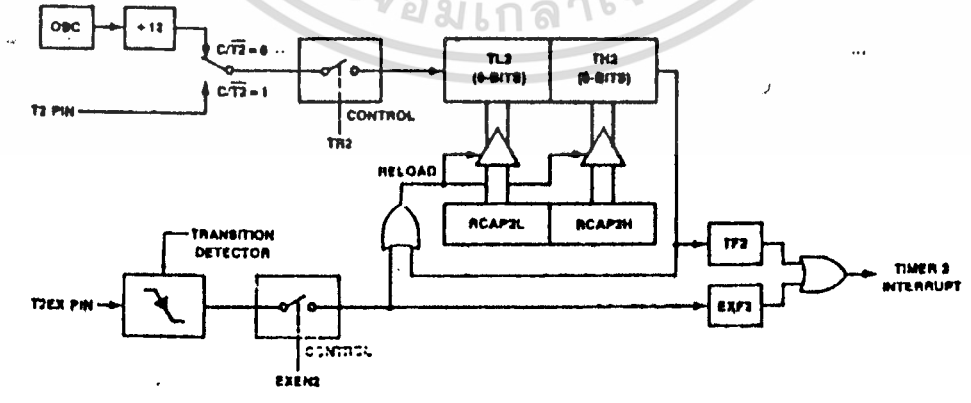
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.18 เป็นการทำงานในโหมดตัวจับเวลา/ตัวนับ 2

1. ถ้าบิต EXEN=0 เมื่อไรก็ตามที่ตัวจับเวลา 2 เพิ่มค่าถึง 0 ไม่เพียงแต่จะเซตบิต TF2 เป็น 1 เท่านั้นแต่ตัวจับเวลา 2 จะถูกบรรจุค่าที่ตั้งไว้ใน RCAP2L และ RAP2H ด้วย Software กลับเข้าไปใหม่

2. ถ้าบิต EXEN=1 ก็ทำงานเช่นเดียวกับวิธีที่ 1 และถ้าเกิดการเปลี่ยนแปลงสภาวะจาก '1' เป็น '0' ที่ T2EX จากภายนอก ก็จะเป็นเหตุให้มีการบรรจุค่าจาก RCAP2L และ RCAP2H กลับเข้าไปใหม่โดยอัตโนมัติ พร้อมทั้งเซตค่าบิตใน EXF2 ของ Register T2CON ด้วย แผนภูมิการทำงานในโหมดบรรจุอัตโนมัติแสดงในรูปที่ 2.19



รูปที่ 2.19 แผนภูมิการทำงานในโหมดบรรจุอัตโนมัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

T2CON : Register ควบคุมตัวจับเวลา/ตัวนับ 2

TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2	CP/RL2

TF2 T2CON.7 ตัวนับเวลา 2 เมื่อเกิด Overflow จะเซตบิตนี้ซึ่งเป็น Overflow แฟลกและจะถูกเคลียร์ด้วย Software TF2 จะไม่ถูกเซต เมื่อค่าใดใน RCLK=1 หรือ TCLK=1

EXF2 T2CON.6 ตัวนับเวลา 2 เป็นแฟลกภายนอกที่ถูกเซตด้วยสัญญาณภายนอก เมื่อมีการทางานหรือบรรจุใหม่ เนื่องจากขอบขาลงของสัญญาณ T2EX และ EXEN2=1 เมื่อการอินเตอร์รัพต์ตัวนับเวลา 2 ถูกยีนาเบิ้ล EXF2=1 เป็นเหตุให้ CPU ชี้ไปยังโปรแกรมการบริการอินเตอร์รัพต์ของตัวนับเวลา 2 EXF2 จะต้องเคลียร์ด้วย Software เสมอ

RCLK T2CON.5 แฟลกสัญญาณนาฬิกาการรับข้อมูล เมื่อบิตนี้เซตจะเป็นเหตุให้ Port อนุกรมใช้พัลส์ Overflow ของตัวนับเวลา 2 สาหรับการให้สัญญาณนาฬิกาการรับในโหมด 1 และ 3 RCLK=0 จะเป็นเหตุให้ Overflow ตัวนับเวลา 1 ถูกใช้ให้เป็นตัวให้สัญญาณนาฬิกาการรับ

TCLK T2CON.4 แฟลกสัญญาณนาฬิกาการส่งข้อมูล เมื่อบิตนี้เซตจะเป็นเหตุให้ Port อนุกรมใช้พัลส์ Overflow ของตัวนับเวลา 2 สาหรับการให้สัญญาณนาฬิกาการส่งในโหมด 1 และ 3 RCLK=0 จะเป็นเหตุให้ Overflow ตัวนับเวลา 1 ถูกใช้ให้เป็นตัวให้สัญญาณนาฬิกาการส่ง

EXEN2 T2CON.3 แฟลกอีนาเบิ้ลภายนอก เมื่อบิตนี้เซตจะทางานแบบปกติหรือบรรจุใหม่ด้วยการเกิดขึ้นจากผลของขอบขาลงบน T2EX ถ้าตัวนับเวลา 2 ไม่ได้ถูกใช้เป็นสัญญาณนาฬิกาสาหรับ Port อนุกรม ถ้า EXEN2=0 ตัวนับเวลา 2 จะไม่มีผลจากสัญญาณที่ T2EX

TR2 T2CON.2 ตัวควบคุมเริ่มและหยุดการทางานของตัวนับเวลา 2 โดยค่า '1' ในบิตนี้จะเป็นค่าเริ่มทางานของตัวนับเวลา 2

C/T2 T2CON.1 ตัวเลือกตัวจับเวลาหรือตัวนับเวลา ถ้าเป็น '0' = ตัวจับเวลาภายใน(osc/12) ถ้าเป็น '1' = ตัวนับภายนอก (ใช้ขอบขาลงเป็นตัวกระตุ้น)

CP/RL2 T2CON.0 แฟลก Capture/Reload เมื่อบิตนี้เซตจะเป็นการทางานปกติรับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สัญญาณขอบขาลงที่ T2EX ถ้า EXEN=1 เมื่อบิตนี้เคลียร์จะเป็นการทำงาน แบบบรรจุอัตโนมัติ เกิดขึ้นได้จากตัวกระตุ้น ในการเกิด Overflow ตัวนับเวลา 2 หรือขอบสัญญาณขาลงที่ T2EX เมื่อ EXEN2=1 เมื่อทั้ง RCLK=1 หรือ TCLK=1 บิตนี้จะไม่ใช้และตัวจับจะถูกบังคับให้ทำงานเป็นโหมดบรรจุใหม่อัตโนมัติของ Overflow ตัวนับเวลา 2 เพื่อใช้ในการสร้างอัตราบ๊อด

ตัวจับเวลา/ตัวนับ 2 สามารถที่จะใช้เป็นตัวกำเนิดสัญญาณนาฬิกาให้อัตราบ๊อดได้ในการใช้โหมดรับส่งอนุกรม 1 และ 3 ด้วยการเซต RCLK หรือ TCLK โดยถ้า RCLK=TCLK=1 เป็นการเลือกตัวจับเวลา/ตัวนับ 1 เป็นตัวกำเนิดอัตราการบ๊อด ในโหมดนี้ตัวจับเวลา/ตัวนับ 2 จะให้พัลส์ของ Overflow มากกว่าที่จะใช้ตัวจับเวลา/ตัวนับเป็นสัญญาณนาฬิกาจ่ายความถี่ Oscillator แก่อนุกรมพอร์ต ถ้าตัวจับเวลา/ตัวนับ 2 = 0 ตัวนับจะเพิ่มหนึ่งที่ช่วง 1/2 ของความถี่ Oscillator ตัวจับเวลา/ตัวนับ 2 = 1 ตัวนับจะเพิ่มหนึ่งตอบสนองตามสัญญาณเข้าที่เปลี่ยนแปลงจาก '1' ไป '0' ที่ขา T2EX (P1.0) ความถี่สูงสุดมีค่าเท่ากับ $1/24*F_{osc}$

ถ้า EXEN2 เซตเป็น 1 ขณะที่ตัวจับเวลา/ตัวนับ 2 ถูกใช้เป็นสัญญาณนาฬิกาพอร์ตอนุกรม ขอบขาลงที่ T2EX (P1.1) จะเซต EXF2 ถ้าไม่มีการทำงานแบบบรรจุใหม่อัตโนมัติหรือแบบธรรมดาเกิดขึ้น โดยปกติขณะที่ตัวจับเวลา/ตัวนับ 2 ใช้เป็นโหมดสร้างอัตราบ๊อด T2EX อาจใช้เป็นตัวรับอินเตอร์รัพต์จากภายนอก TF2 จะเป็นผลตามตัวนับเวลาที่เกิด Overflow ขณะเดียวกันการแบบบรรจุใหม่อัตโนมัติจาก RCAP2H และ RCAP2L จะเกิดขึ้นโดยไม่คำนึงถึงสถานะของ CP/RL2

TH2 และ TL2 ไม่สามารถเขียนหรืออ่านขณะที่เกิดข้อผิดพลาดและขณะที่ ตัวจับเวลา/ตัวนับ 2 ใช้เป็นโหมดตัวสร้างอัตราบ๊อดก็จะไม่สามารถเขียนบนตัว RCAP2H และ RCAP2L ได้ในโหมดนี้เช่นกัน

2.10.3 Timer/Counter Control And Register Status

การกำหนดโหมดการทำงานและควบคุมฟังก์ชั่นต่างๆของตัวจับเวลา/ตัวนับ จะควบคุมได้ที่ SFR : Spectial Function Register TMOD, TCON T2CON (สำหรับ 8032/8052) ด้วย Software โดยที่เมื่อมีคำสั่งเปลี่ยนค่าบิตต่างๆใน TMOD,TCON หรือ T2CON ค่าที่ถูกเปลี่ยนก็ที่จะถูกแลทช์เข้าไปที่ SFR และเกิดมีผลตามคำสั่งควบคุมที่ช่วง S1P1 ของวัฏจักรตัวแรกของชุดคำสั่งต่อมา

2.11 การเชื่อมต่อแบบอนุกรม

พอร์ตอนุกรมเป็นแบบ Full Duplex สามารถที่จะส่งและรับพร้อมกันได้ โดยทำหน้าที่เป็น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บัฟเฟอร์การรับ หมายถึงพอร์ตสามารถที่จะรับไบต์ที่สองก่อนที่ตัวแรกจะรับไปจาก Register ตัวรับ อย่างไรก็ตามไบต์ตัวแรกจะต้องถูกอ่านไปก่อนที่ช่วงเวลาการรับไบต์ตัวที่สองจะสิ้นสุด มิฉะนั้นไบต์ตัวแรกจะถูกซ้อนและสูญหายไปได้ ในพอร์ตอนุกรม Register ตัวรับและตัวส่งจะเข้าถึงติดต่อกันด้วย Register SBUF ใน SFR แม้ว่าทางโครงสร้าง Register ทั้งสองจะแยกกันอยู่ก็ตาม พอร์ตอนุกรมสามารถเลือกการทำงานในโหมดต่างๆได้ 4 โหมด

โหมด 0 : ข้อมูลจะเข้าและออกผ่าน RXD TXD ด้วยการเลื่อนสัญญาณพัลส์ Output ข้อมูลจะเป็นลักษณะ 8 บิต ในการรับและการส่งแต่ละครั้ง โดยที่ส่งค่า LSB ก่อนอัตราบ๊อดจะคงที่ที่ 1/12 Oscillator

โหมด 1 : จะเป็นการส่งข้อมูลขนาด 10 บิตผ่านออก TXD หรือรับเข้าผ่าน RXD โดยรูปแบบบิตจะประกอบด้วย หนึ่งบิต Start เป็น '0' 8 บิตข้อมูลโดย LSB เป็นตัวแรกที่รับและส่งข้อมูลนี้และอีกหนึ่งบิต Stop มีค่า '1' การรับบิต Stop จะนำไปเก็บที่บิต RB8 ของ SFR Register SCON อัตราบ๊อดแปรผันได้ตามการตั้งตัวจับเวลา

โหมด 2 : เป็นการส่งข้อมูลขนาด 11 บิตผ่านออกขา TXD หรือรับเข้ามาผ่านขา RXD ประกอบด้วยหนึ่งบิต Start มีค่า '0' 8 บิตข้อมูลโดย LSB เป็นตัวแรกที่รับและส่งข้อมูล บิตที่ 9 ของข้อมูลสามารถที่จะโปรแกรมเลือกได้และบิต Stop ค่า '1' อีกหนึ่งบิต ในการส่งบิตที่ 9 ที่อยู่ในบิต TB8 ของ Register SCON สามารถเลือกที่กำหนดเป็น '1' หรือ '0' ได้ ตัวอย่างเช่น การใช้งานบิตพาริตี้ โดยการเลื่อนเอาบิต P ของ PSW มาใช้ใน TB8 เพื่อเป็นการส่งข้อมูลแบบมีการตรวจสอบพาริตี้ของข้อมูลที่ส่ง ในการรับข้อมูลบิตที่ 9 จะเข้าไปเก็บที่ RB8 ใน SFR Register SCON ขณะที่บิต Stop จะไปรับเข้ามาเก็บ อัตราบ๊อดสามารถเลือกได้เป็น 1/32 หรือ 1/64 ของความถี่ Oscillator SCON เป็น SFR ที่ใช้ในการกำหนดโหมดการทำงานของพอร์ตอนุกรม เช่น การกำหนดค่า RB8 จะเป็นการใช้ตัวรับรับบิตที่ 9 ด้วยหรือไม่

โหมด 3 : เป็นการส่งข้อมูลขนาด 11 บิตผ่านออกขา TXD หรือรับเข้ามาผ่านขา RXD ประกอบด้วยหนึ่งบิต Start มีค่า '0' 8 บิตข้อมูลโดย LSB เป็นตัวแรกที่รับและส่งข้อมูล บิตที่ 9 ของข้อมูลสามารถที่จะโปรแกรมเลือกได้และบิต Stop ค่า '1' อีกหนึ่งบิต ในความเป็นจริงโหมด 3 จะคล้ายกับโหมด 2 ทุกประการยกเว้นอัตราบ๊อด โดยอัตราบ๊อดในโหมด 3 จะแปรผันได้ไปตามการโปรแกรมการเลือกตัวจับเวลา

ทั้ง 4 โหมดนี้การส่งข้อมูลจะถูก Initiated ด้วยคำสั่งใดๆที่ใช้ตัว Register SBUF เป็น Register ตัวรับข้อมูลจาก CPU และในโหมด 0 การรับข้อมูลจะ Initiated ด้วยการใช้สถานะ RI = 0 และ REN = 1 ส่วนในโหมดอื่นการรับข้อมูลจะถูก Initiated ด้วยการรับบิต Start เข้ามาตรวจสอบ ถ้า REN=1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SCON :- Register ควบคุม Serial Port

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

โดย SM0,SM1 เป็นตัวกำหนดใช้โหมดต่างๆของพอร์ตอนุกรม ดังนี้

SM0	SM1	โหมด	ลักษณะการใช้งาน	อัตราบ๊อด
0	0	0	เลื่อน Register	$f_{osc}/12$
0	1	1	8 Bit UART	แปรผันได้ตามการเลือกตัวจับเวลา
1	0	2	9 Bit UART	$f_{osc}/64$ หรือ $f_{osc}/32$
1	1	3	9 Bit UART	แปรผัน

*UART : Universal Asynchonous Receiver / Tranceiver

SM2 ควบคุมอีนาเบิ้ล การใช้ Processer หลายตัวในการสื่อสารซึ่งกันและกัน

ในโหมด 2 และ 3 ถ้า SM2 เซตเป็น 1 ดังนั้น RI จะต้องไม่ Active ถ้ามีการรับบิตที่ 9 ทำให้บิต RB8 เป็น 0

ในโหมด 1 ถ้า SM2 เซตเป็น 1 ดังนั้น RI จะไม่ Active ถ้า Stop ไม่ถูกรับ

ในโหมด 0 SM2 ควรมีค่าเท่ากับ 0

REN ตัวอีนาเบิ้ลอนุกรมการรับ เซตเป็น 1 ด้วยโปรแกรมในการเลือกอีนาเบิ้ลการรับ และเป็น 0 ด้วยโปรแกรม เมื่อให้ดิสเอเบิ้ล

TB8 เป็นข้อมูลบิตที่ 9 ซึ่งจะถูกส่งในโหมด 2 และ 3 ซึ่งจะให้เป็น 1 หรือ 0 ด้วยการโปรแกรม

RB8 ในโหมด 2 และ 3 ข้อมูลบิตที่ 9 จะถูกรับมา

ในโหมด 1 ถ้า SM2=0 RB8 จะกลายเป็น Stop บิตที่ถูกรับมา

ในโหมด 0 RB8 ไม่ใช้

TI เป็นแฟลกอินเตอร์รัพต์การส่ง เซตด้วย Hardware คือสัญญาณที่ปลายช่วงเวลาของบิตที่ 8 ในโหมด 0 หรือจุดเริ่มต้นของบิต Stop ในโหมดอื่น ในการส่งแบบอนุกรมของทุกโหมดจะต้องเคลียร์โหมดนี้ด้วยโปรแกรมการส่ง

RI เป็นแฟลกอินเตอร์รัพต์การรับ เซตด้วย Hardware คือสัญญาณที่ปลายช่วงเวลาของบิตที่ 8 ในโหมด 0 หรือจุดเริ่มต้นของบิต Stop ในโหมดอื่น ในการรับแบบอนุกรมยกเว้นกรณีการใช้ SM2 จะต้องเคลียร์บิตด้วยโปรแกรมหลังการรับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.11.1 โหมด 0

ข้อมูลอนุกรมจะรับและส่งผ่านตัว Output TXD,RXD แบบเลื่อนข้อมูลเป็นอนุกรม 8 บิตที่จะส่งหรือรับข้อมูลขนาด 8 บิต โดยเลื่อนบิต LSB เป็นตัวแรก อัตราบ๊อดคงที่ที่ 1/12 ของความถี่ Oscillator รูปที่ 2.20 แสดงถึงการใช้งานโหมด 0 พร้อมกับแผนภูมิเวลาสำหรับการรับและส่ง

การส่งจะถูกเริ่ม (Initiated) ด้วยคำสั่งต่างๆในการใช้ SBUF เป็น Register รับข้อมูล (Destination) สัญญาณคำสั่ง 'Write to SBUF' เกิดที่ S6P2 และจะบรรจุค่า 1 เข้าที่ตำแหน่งบิตที่ 9 ของตัว Register การเลื่อนส่งและบอกให้ส่วนควบคุมการส่ง (Tx Control Block) ให้ทำงานการส่ง ภายในช่วงเวลาหนึ่งวัฏจักรแมคชีนจะครอบคลุมให้เกิดพัลส์ 'Write to SBUF' ที่ S6P2 และสัญญาณ SEND ต้อง Active สูงในช่วงการส่ง

การอีนาเบิ้ล SEND ที่มี Active สูงจะเป็นการควบคุมให้ข้อมูลส่งออกจากตัว Register การเลื่อนบิตออกที่ขา P3.0 และตัวเลื่อนสัญญาณนาฬิกา (Shift Clock) ก็จะถูกให้ส่งออกที่ขา P3.1 โดยที่ลักษณะของสัญญาณ Shift Clock จะมีระดับต่ำที่ S3,S4 และ S5 ของทุกวัฏจักรแมคชีนและมีระดับสูงช่วง S6 S1 และ S2 ในช่วง S6P2 ของทุกวัฏจักรแมคชีน ขณะที่ SEND Active ค่าข้อมูลใน Register การเลื่อนส่งจะถูกเลื่อนไปทางขวาหนึ่งตำแหน่งหรือบิต คือเป็นการส่งที่ขา P3.0 ไปหนึ่งบิตขณะนั้น

ในขณะที่เลื่อนบิตไปทางขวา ค่า "0" จะเข้าแทนที่ทางซ้ายเมื่อเลื่อนจน MSB เลื่อนออกจากตัว Register การเลื่อน ดังนั้นค่า '1' จะเริ่มถูกบรรจุเข้าไปที่ตำแหน่งที่ 9 ที่ต่อจาก MSB ที่ถูกเลื่อนออกไปและทุกตำแหน่งทางซ้ายมือจะเป็น '0' หมดทุกตัว สถานะของแฟลกในส่วนควบคุมการส่ง จะทำการเลื่อนครั้งสุดท้าย และให้สัญญาน SEND กลับคืนจากสภาพ Active เป็นระดับต่ำ และปรับ TI เป็นระดับสูงด้วย ส่งออกไปที่พอร์ตอนุกรมการอินเตอร์รัพต์ ทั้งสองสัญญาณที่ปรับระดับนี้จะเกิดช่วง S1P1 ของวัฏจักรแมคชีนที่ 10 หลังจากที่สัญญาน 'Write to SBUF' เริ่มสิ้นสุด

การรับข้อมูลจะถูกควบคุมด้วยการโปรแกรมเริ่มแรก การขอแม้ให้ REN=1 และ RI = 0 ที่ S6P2 ของวัฏจักรแมคชีนตัวใหม่หน่วยควบคุม Rx จะเขียนบิตค่า 1111 1110ไปยัง Register การเลื่อนรับ และที่เฟส Clock ตัวใหม่ จะ Active สัญญาน Receive ให้สูง

เมื่อ Receive ถูกอีนาเบิ้ลให้สูง ก็จะทำให้สัญญาน Shift Clock ส่งพัลส์ขึ้นต่างๆออกที่ขา P3.1 Shift Clock จะเปลี่ยนสถานะที่ S3P1 และ S6P1 ของทุกวัฏจักรแมคชีนที่ S6P2 ของวัฏจักรแมคชีนตัวแรก สัญญาน Receive เริ่ม Active สูงและค่าใน Register การเลื่อนรับจะรับข้อมูลเข้ามา และเลื่อนมาทางซ้ายหนึ่งตำแหน่ง และทุกค่าที่เข้ามาจากทางขวาจะเป็นค่าที่ถูก Sample เข้าที่ขา P3.0 ที่ช่วง S5P2 ของทุกวัฏจักรแมคชีน

ขณะที่ข้อมูลบิตต่ำเข้ามาทางขวา ค่า '1' จะเลื่อนออกไปทางซ้ายเมื่อค่า '0'ถูกบรรจุเริ่ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แรกที่เข้ามาทางตำแหน่งขวาสุด ถูกเลื่อนมาอยู่ตำแหน่งซ้ายสุดใน Register การเลื่อนมันจะมีผลให้ค่าแฟลกในหน่วยควบคุมการรับ (Rx Control Block) ให้ทำการเลื่อนเป็นครั้งสุดท้าย และเริ่มบรรจุข้อมูลทั้งหมดเข้าใน SBUF ที่ช่วง S1P1 ของวัฏจักรแมคชีนที่ 10 หลังจากเริ่มส่งสัญญานสโตรบ 'Write to SCON' ของ RI เคลียร์ ต่อจากนั้นสัญญาน Receive จะเคลียร์และ RI จะปรับเป็น '1'

2.11.2 โหมด 1

จำนวน 10 บิตจะถูกส่งผ่าน TXD หรือรับผ่าน RXD ที่ประกอบด้วยบิต Start บิตข้อมูล 8 บิตและบิต Stop การรับบิต Stop จะส่งเข้า RB8 ใน Register SCON การตั้งอัตราความเร็วของบ๊อดจะแปรผันได้ ทั้งตัวจับเวลา 1 หรือ 2 อาจใช้เป็นสัญญานนาฬิกาสำหรับพอร์ตอนุกรมโดยสร้างอัตราความเร็วแปรผันด้วยการตั้งหรือเคลียร์ค่าบิตใน T2CON เป็น TCLK และ RCLK

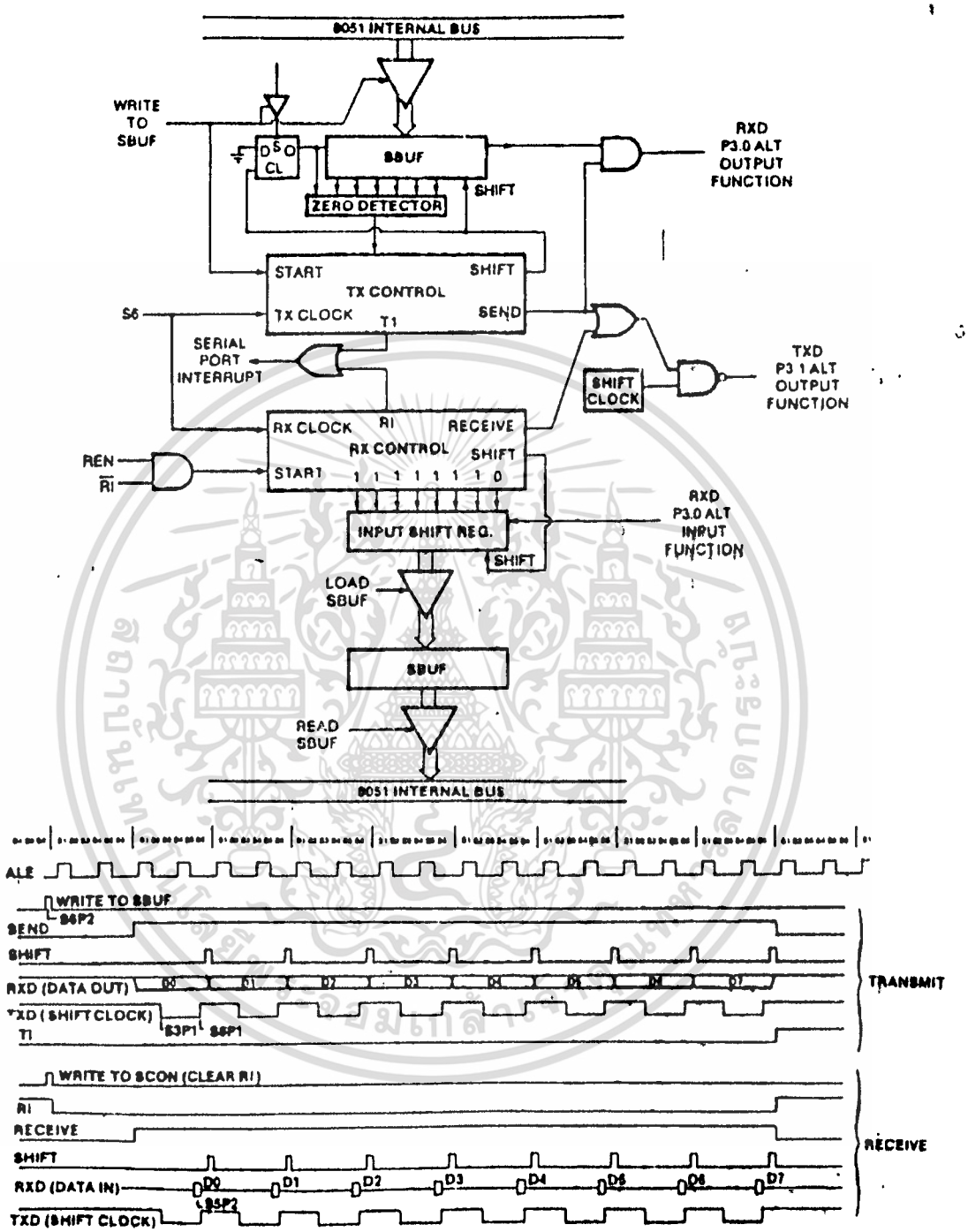
การส่งจะเริ่มต้นงานด้วยคำสั่งที่ใช้ SBUF เป็น Register รับข้อมูล สัญญาณ ' Write to SBUF' ก็จะบรรจุค่า '1' เข้าไปเป็นตำแหน่งที่ 9 ใน Register การเลื่อนส่งและแฟลกในหน่วยควบคุมการส่ง (Tx Control Block) ก็แสดงการถูกร้องขอให้ส่งข้อมูลการส่งข้อมูลจะส่งที่ช่วง S1P1 ของวัฏจักรแมคชีนและจะตามด้วยบิตตัวต่อมา ในช่วงเวลาของสัญญาณนาฬิกาที่หารด้วย 16 ที่ถูกตั้งที่ตัวนับ ดังนั้นแต่ละบิตจะถูกซิงค์ด้วยการหาร 16 ของตัวนับ ไม่ใช่ด้วยสัญญาณ 'Write to SBUF'

การส่งจะเริ่มด้วยการส่ง Active สัญญาณ SEND และใส่บิต Start เข้าที่ TXD ช่วงเวลาหลังจากนั้นหนึ่งบิต สัญญาณข้อมูลก็จะ Active ซึ่งก็จะอีนาเบิ้ลการส่งบิตออกจาก Register การเลื่อนส่งออกไปยังขา TXD พัลส์เลื่อนตัวบิตแรกจะเกิดขึ้นหลังเวลาทำงานแล้วหนึ่งบิต

ขณะที่ข้อมูลเลื่อนออกทางขวา ค่า '0' จะถูกใส่เข้าทางซ้าย เมื่อ MSB ของข้อมูลหนึ่งไบต์อยู่ที่ตำแหน่ง Output ของ Register ตัวเลื่อน ขณะนั้นค่า '1' จะเริ่มถูกบรรจุเข้าเป็นตำแหน่งที่ 9 ที่ Output หลังจากที่ MSB ถูกส่งออกไปและทุกตำแหน่งเมื่อถูกส่งไปแล้ว ที่เหลือใน Register การเลื่อนจะเป็น '0' หมด สถานะแฟลกในหน่วยควบคุมการส่ง (Tx Control Unit) ก็จะเลื่อนเป็นตัวสุดท้าย และส่งสัญญาณ SEND ดิสเอเบิ้ลและการเซต TI จะเกิดขึ้นในช่วงลูกที่ 10 ของการหาร 16 หลังการส่งสัญญาณสโตรบ 'Write to SBUF'

การรับจะถูกเริ่มงานด้วยการกระตุ้นจากการเปลี่ยนแปลง '1' เป็น '0' ที่ RXD สำหรับจุดนี้ RXD ก็จะถูก Sample ด้วยอัตรา 16 เท่าของอัตราบ๊อดที่ถูกกำหนดเริ่มแรก เมื่อการส่งข้อมูลเกิดรับได้ ตัวนับหาร 16 ก็จะถูก Reset และค่า 01FFH ก็จะถูกเขียนเข้าไปใน Register ตัวเลื่อน การ Reset ตัวนับหาร 16 ก็จะเป็นการตั้งวนรอบด้วยการให้ขอบเขตของช่วงเวลาแต่ละบิตที่เข้ามา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูป 2.20 เป็นการแสดงถึงฟังก์ชันแผนภูมิเวลาธรรมดาของพอร์ตอนุกรมในโหมด 0 และช่วงเวลาที่เกิดขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

16 คาบเวลาของตัวนับในแต่ละบิต จะเป็นเวลาที่เข้ามาในคาบที่ 16 ที่ตัวนับนับคาบที่ 7,8 และ 9 จะเป็นช่วงเวลาของบิต เป็นการรับข้อมูลแต่ละบิตที่ Sample ค่าที่เข้ามาทาง RXD และค่าที่รับเข้ามาถูก Sample อย่างน้อย 2-3 ครั้ง การทำเช่นนี้จะเป็นการขจัด Noise ออกไป ถ้าค่าข้อมูลถูกรับในระหว่างช่วงเวลาบิตแรกที่ไม่ใช่ค่า '0' วงจรการรับจะถูก Reset และหน่วยรับก็จะกลับไปตรวจการเปลี่ยนแปลงจาก 0 ไป 1 ใหม่ ลักษณะงานเช่นนี้จะเป็นการป้องกันรับบิต Start ที่ผิดพลาดเข้ามาได้ ถ้าบิต Start ถูกรับเข้ามาถูกต้องมันก็จะถูกเลื่อนเข้า Register ตัวเลื่อน และการรับข้อมูลก็จะเริ่มขึ้น

ขณะที่ข้อมูลเข้ามาจากทางขวา ค่า '1' จะถูกเลื่อนออกไปทางซ้ายเมื่อค่าบิต Start ถูกเลื่อนมาถึงทางซ้ายสุดใน Register ตัวเลื่อนมันก็จะแฟลกในหน่วยควบคุมการรับ (Rx Control Block) ให้เลื่อนอีกหนึ่งครั้งเป็นครั้งสุดท้าย และก็จะบรรจุข้อมูลเข้า SBUF และ RB8 (เพราะมี 9 บิต) และเซต RI สัญญาณการบรรจุเข้า SBUF, RB8 และการเซต RI เป็น '1' จะปรากฏถ้าเพียงแต่กรณีใดต่อไปนี้ปรากฏในช่วงเวลาพัลส์การเลื่อนสุดท้ายเกิดขึ้น คือ

1. RI = 0
2. SM2 = 0 หรือ การรับ Stop บิต = 1

ถ้าไม่เกิดทั้งสองกรณี การรับข้อมูลไบต์ก็จะล้มเหลว ถ้าเกิดทั้งสองกรณี ตัว Stop บิตก็จะไปเก็บที่ RB8 และข้อมูลไบต์ก็จะเข้า SBUF และ RI จะ Active สูง ช่วงเวลานี้ไม่ว่าจะเกิดขึ้นทั้งสองกรณีหรือไม่ หน่วยควบคุมการรับก็จะกลับไปตรวจการเปลี่ยนแปลง 1 - 0 ของการส่งข้อมูลผ่าน RXD ใหม่

2.11.3 โหมด 2 และ 3

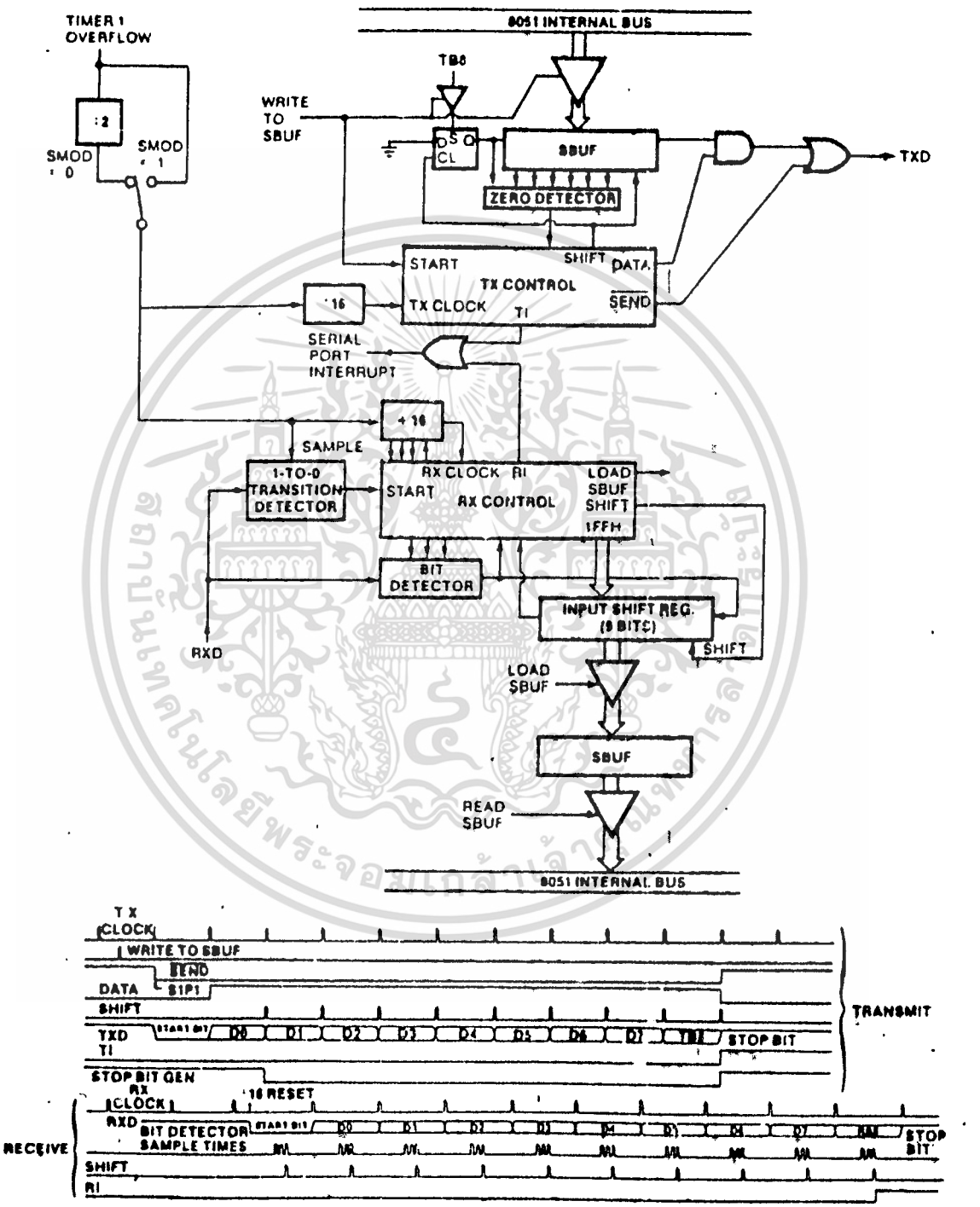
จำนวน 11 บิตจะส่งออกที่ TXD และรับเข้าที่ RXD โดยมีบิต Start มีค่า '0' ข้อมูล 8 บิตมี LSB เป็นบิตแรกและโปรแกรมบิตที่ 9 ได้ และบิต STOP มีค่า '1' การส่งข้อมูลบิตที่ 9 ใช้ TB8 เป็นตัวกำหนดค่า '0' หรือ '1' การรับข้อมูลบิตที่ 9 ใช้ RB8 ใน SCON เป็นตัวรับอัตราบ๊อดสามารถโปรแกรมเลือกได้ทั้งแบบ 1/32 หรือ 1/64 ของความถี่ Oscillator ในโหมด 2 แต่โหมด 3 จะใช้ตัวแปรหลายค่าของอัตราบ๊อด เกิดจากการใช้ตัวจับ 1 หรือ 2 ขึ้นอยู่กับสถานะ TCLK และ RCLK

รูปที่ 2.22 และรูปที่ 2.23 แสดงแผนภูมิฟังก์ชั่นของพอร์ตอนุกรมในโหมด 2 และ 3 ส่วนของตัวรับจะทำงานเหมือนกับโหมด 1 ส่วนของตัวส่งต่างจากโหมด 1 เพียงบิตที่ 9 ของ Register ตัวเลื่อนการส่ง

การส่งจะเริ่มด้วยคำสั่งใด ๆ ที่ใช้ SBUF เป็น Register ตัวรับข้อมูลสัญญาน 'Write to SBUF' ก็จะบรรจุ TB8 เข้าไปเป็นบิตที่ 9 ของตำแหน่งใน Register ตัวเลื่อนการส่งและแฟลก

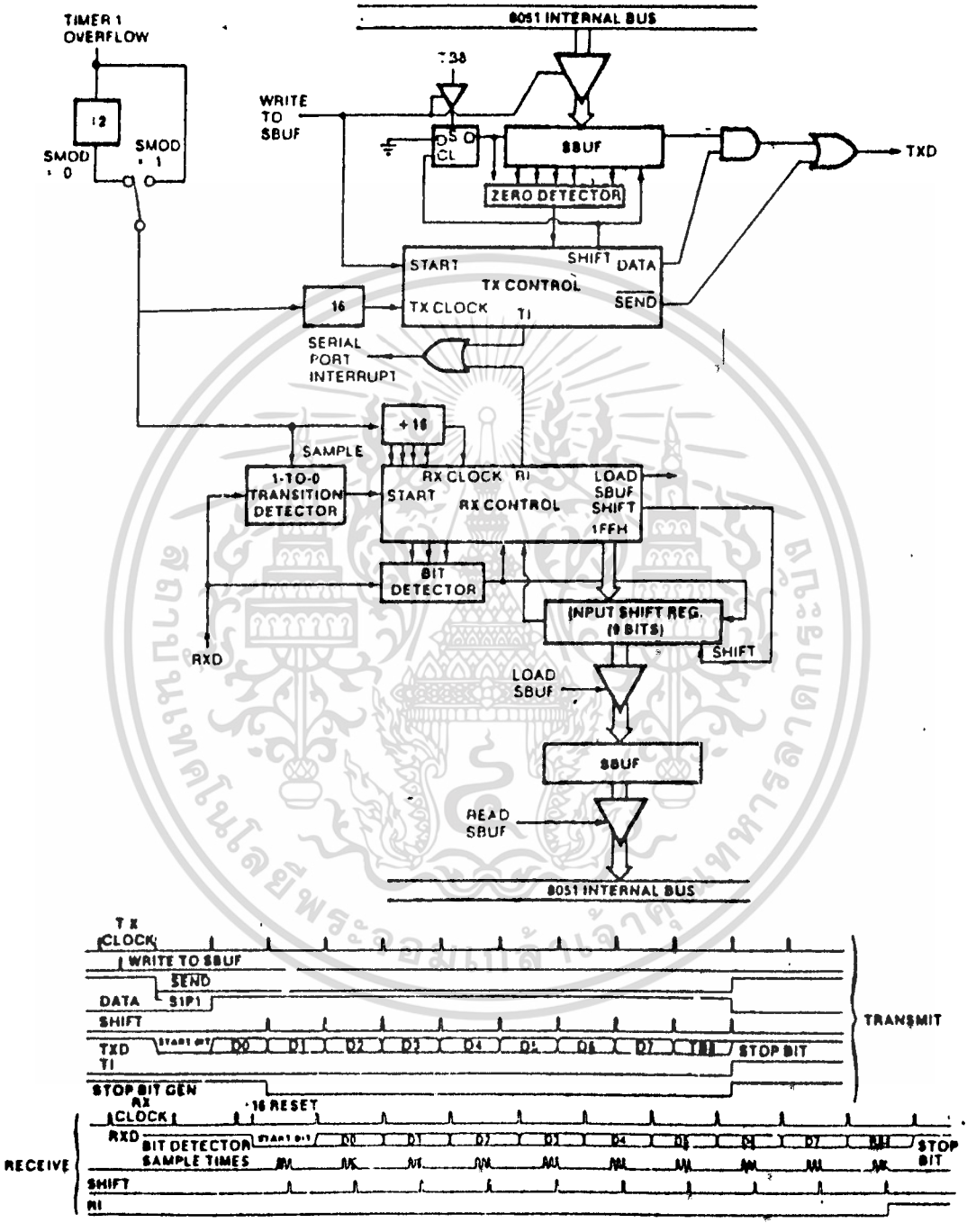
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในหน่วยควบคุมการส่งก็จะให้การส่งถูกร้องขอ ช่วงการส่งจะเริ่มขึ้นที่ S1P1 ของวัฏจักรแมชีนตัวต่อมา ในการใช้ตัวนับหาร 16 ดังนั้นช่วงแต่ละบิตจะซิงค์โครไนส์กับตัวนับหาร 16 โดยไม่ใช้สัญญาณของ 'Write to SBUF'



รูปที่ 2.21 แสดงถึงแผนภูมิการใช้งานโหมด 1 พร้อมกับแผนภูมิเวลาสำหรับสัญญาณการส่งและรับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.23 แสดงแผนภูมิฟังก์ชันของพอร์ตอนุกรมในโหมด 3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การส่งเริ่มด้วยสัญญาน SEND Active ต่ำใส่บิต Start ออกที่ TXD หลังจากนั้นบิตแต่ละตัวของข้อมูลในสัญญาน DATA ก็จะ Active ซึ่งจะอีนาเบิ้ลให้บิตของ Register ตัวเลื่อนส่งออกที่ TXD ตามมา โดยตัวเลื่อนพัลส์ตัวแรกจะเกิดหลังจากนั้นเล็กน้อย ที่สัญญานนาฬิกาเลื่อนตัวแรก ถ้า '1' คือเป็นบิต Stop จะใส่เข้าไปใน Register ตัวเลื่อนทางซ้ายสุดเป็นตำแหน่งบิตที่ 9 ดังนั้นหลังจากนั้นสัญญานนาฬิกาเข้าตัวต่อมาจะใส่ค่า '0' เข้าไปเท่านั้นและทุกสัญญาณนาฬิกาจะเลื่อนเอาบิตออกทางขวา และใส่ค่า '0' เข้าทางซ้ายมือ TB8 อยู่ที่ตำแหน่ง Outputของ Register ตัวเลื่อน ดังนั้นบิต Stop จะส่งออกต่อ TB8 และทุกตำแหน่งใน Register ตัวเลื่อนที่เหลือจะเป็นศูนย์หมด ด้วยสถานะเช่นนี้จะทำให้เฟลกในหน่วยควบคุมการส่งเลื่อนเป็นครั้งสุดท้าย และให้สัญญาน SENT ดิสเอเบิ้ลสูง และเซต TI ด้วยซึ่งจะเกิดขึ้นที่พัลส์ลูกที่ 11 ของตัวหาร 16 สัญญานนาฬิกาหลังจากส่งสโตรบสัญญาณ 'Write to SBUF'

การรับจะเริ่มทำงานด้วยการจับสัญญาณที่ขา RXD ช่วงการเปลี่ยนจาก '1' เป็น '0' สำหรับโหมดนี้ตัว RXD จะถูก Sampled ที่อัตรา 16 เท่าของอัตราโบดที่ถูกกำหนดมา เมื่อสภาวะการเปลี่ยนแปลงถูกรับได้ ตัวนับหาร 16 ก็จะเริ่ม Reset ทันทีและค่า 1FFH ก็จะเขียนเข้าไปใน Register ตัวเลื่อน

ที่ช่วงการนับลูกที่ 7,8 และ 9 ของแต่ละช่วงบิต ตัวดีเทคเตอร์จะ Sample ค่าบิตของสัญญาณที่เข้า RXD ค่าที่รับเข้ามาจะเป็นค่าที่คล้ายกับว่ามีการรับเข้า Sampling อย่างน้อย 2 ถึง 3 Sample ถ้าค่าถูกรับเข้าช่วงระหว่างบิตแรกไม่เป็น '0' วงจรตัวรับก็จะ Reset และหน่วยควบคุมก็จะกลับตรวจการเปลี่ยนแปลงจาก 1 เป็น 0 ใหม่ ถ้าบิต Start ถูกพิสูจน์ว่าถูกต้องมันก็จะเลื่อนเข้าสู่ Register ตัวเลื่อนและการรับก็จะรับจนครบ Frame ของข้อมูลในโหมดนี้

ขณะที่ข้อมูลเข้ามาจากทางขวา ค่า '1' ก็จะถูกเลื่อนไปทางซ้ายออกไป เมื่อบิต Start ถูกเลื่อนมาถึงตำแหน่งทางซ้ายสุดของ Register ตัวเลื่อนโดยในโหมด 2 และ 3 จะมี Register 9 บิต มันจะแฟลกให้หน่วยควบคุมการรับทำการเลื่อนครั้งสุดท้าย แล้วบรรจุค่าใน SBUF และ RB8 และเซต RI สัญญาณการบรรจุ SBUF และ RB8 และการเซต RI เป็น '1' จะถูกสร้างขึ้น ถ้าเพียงแต่เกิดกรณีหนึ่งต่อไปนี้บรากฏในช่วงเวลาพัลส์การเลื่อนลูกสุดท้ายคือ

1. RI = 0

2. SM2 = 0 หรือและการรับบิตที่ 9 มีค่า = 1

ถ้าทั้งสองกรณีไม่เกิดขึ้น การรับ Frame ของข้อมูลที่จะสูญหายและ RI จะไม่เซตถ้าทั้งสองกรณีเกิดขึ้นการรับบิตที่ 9 จะรับเข้า RB8 และ 8 บิตแรกจะบรรจุเข้าใน SBUF ช่วงเวลาหนึ่งบิตหลังจากนั้นไม่ว่าจะได้รับข้อมูลสูญหาย หน่วยควบคุมก็จะกลับตรวจสอบการเปลี่ยนแปลงค่า 1 เป็น 0 ที่ Input ของ RXD ใหม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.12 การสื่อสาร Multiprocessor

โหมด 2 และ 3 มีการใช้ทำงานพิเศษสำหรับการสื่อสารทาง Multiprocessor ทั้งสองโหมดนี้ การรับบิตที่ 9 ของข้อมูลจะรับเข้า RB8 แล้วจึงจะตามมาด้วยบิต Stop Port สามารถถูกโปรแกรมเมื่อบิต Stop ถูกรับเข้ามาการอินเตอร์รัพต์ทางพอร์ตอนุกรมจะ Active เท่านั้น ถ้า RB8 = 1 การทำงานลักษณะนี้จะอีนาเบิ้ลได้ด้วยการเซ็ตบิต SM2 ใน Register SCON เพื่อให้ใช้งาน Multiprocessor ตามลำดับต่อไปนี้

เมื่อตัว Processor หลักต้องการส่งข้อมูลจำนวนหนึ่งไปยัง Processor ตัวลูกทั่วๆไปมันจะต้องส่งไบต์ Address กำหนดเป้าหมายตัวลูกทั่ว ๆไปก่อน ไบต์ Address ต่างจากไบต์ข้อมูลที่บิตที่ 9 มีค่าเป็น '1' ขณะที่บิตที่ 9 ของข้อมูลมีค่าเป็น '0' SM2 = 1 เป็นการอินาเบิ้ลกลุ่ม Processor ลูกไม่ให้ถูกการอินเตอร์รัพต์ด้วยไบต์ข้อมูล อย่างไรก็ตามไบต์ Address จะอินเตอร์รัพต์ Processor ลูกทั้งหมด การทำเช่นนี้จะทำการช่วยให้ Processor ลูกแต่ละตัวตรวจสอบไบต์ที่รับเข้า ถ้ามันเป็นตัว Processor ลูกเป้าหมายที่ถูกกำหนดด้วยไบต์ Address มันจะเคลียร์ค่าบิต SM2 และเตรียมรับไบต์ข้อมูลที่จะเข้ามาต่อไป ตัว Processor ลูกตัวอื่นที่ไม่ได้ถูกไบต์ Address กำหนด ก็ยังคงเซตค่า SM2 และยังคงทำงานในส่วนเฉพาะของตัวเองในขณะเดียวกันก็จะไม่มีการรับข้อมูลไบต์ที่ส่งมา ค่า SM2 ควรจะเคลียร์สำหรับการทำงานในโหมด 0

2.13 อัตราบ็อด

อัตราบ็อดในโหมด 0 ของการใช้พอร์ตอนุกรมจะคงที่ที่ความถี่ของออสซิลเลเตอร์คือ อัตราบ็อดในโหมด 0 = ความถี่ Oscillator /12

อัตราบ็อดในโหมด 2 จะขึ้นอยู่กับการปรับค่าบิตใน SMOD ของ SFR ใน Register PCON ถ้า SMOD = 0 ซึ่งจะเป็นค่าที่ถูก Reset แต่แรก หลังการ Reset อัตราบ็อดจะเป็น 1/64 ของความถี่ Oscillator ถ้า SMOD = 1 อัตราบ็อดจะเป็น 1/32 ความถี่ Oscillator มีสูตรเป็น

$$\text{อัตราบ็อดโหมด 2} = [2^{\text{smod บิต}} * \text{ความถี่ Oscillator}]/64$$

ใน MCS-51 อัตราบ็อดในโหมด 1 และ 3 ถูกกำหนดได้ด้วยอัตรา Overflow ที่เกิดขึ้นจากการกำหนดค่าใน Register TH1 ของตัวจับเวลา 1 ส่วนใน 8052 อัตราบ็อดเหล่านี้สามารถคำนวณได้จากตัวจับเวลา 1 หรือตัวจับเวลา 2 หรือใช้ทั้งสองตัวโดยตัวหนึ่งสำหรับส่งและอีกตัวสำหรับรับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.13.1 การใช้ตัวจับเวลา 1 เป็นตัวสร้างอัตราบ๊อด

เมื่อใช้ตัวจับเวลา 1 เป็นตัวสร้างอัตราบ๊อด อัตราบ๊อดในโหมด 1 และ 3 จะถูกคำนวณด้วยอัตรา Overflow ที่เกิดขึ้นในตัวจับเวลา 1 และค่าบิตใน SMOD ซึ่งสูตรการคำนวณเป็นดังนี้

	TIMER1				
BAUD RATE	f_{osc}	SMOD	C/$\overline{T}$	MODE	RELOAD VALUE
MODE 0 MAX. 1MHz	12 MHz	X	X	X	X
MODE 2 MAX. 375K	12 MHz	1	X	X	X
MODE 1,3 62.5K	12 MHz	1	0	2	FFH
19.2K	11.059MHz	1	0	2	FDH
9.6K	11.059MHz	0	0	2	FDH
4.8K	11.059MHz	0	0	2	FAH
2.4K	11.059MHZ	0	0	2	F4H
1.2K	11.059MHz	0	0	2	E8H
137.5	11.059MHz	0	0	2	1DH
110	6 MHz	0	0	2	72H
110	12 MHz	0	0	1	FEEBH

ตารางรายการอัตราบ๊อดที่ใช้ตัวจับเวลา 1

smod

อัตราบ๊อดในโหมด 1,3 = [2 * อัตรา Overflow ของตัวจับเวลา 1]/32

การอินเตอร์รัพต์ตัวจับเวลา 1 ควรที่จะดิสเอเบิ้ลในการใช้งานแบบนี้ ตัวจับเวลาในตัวมันเองสามารถที่จะถูกกำหนดให้ใช้เป็นตัวจับเวลาหรือตัวนับการทำงานในโหมด 3 ในการใช้งานในลักษณะนี้ มันจะถูกกำหนดให้ทำงานเป็นตัวจับเวลาในโหมดแบบบรรจุอัตโนมัติ (โดยตั้งให้ HIGH NIBBLE ของ TMOD = 0010B) ในกรณีนี้ อัตราบ๊อดคำนวณได้ดังสูตร

smod

อัตราบ๊อดในโหมด 1,3 = [2 /32 * [ความถี่ Oscillatoe /(12*(256-TH1))]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการให้อัตราบ๊อดมีค่าต่ำมาก ก็สามารถที่จะทำได้โดยการตั้งตัวจับเวลา 1 ให้สามารถรับการอินเตอร์รัพท์ได้ และกำหนดให้ตัวจับเวลาทำงานเป็น 16 บิต (โดยตั้งค่า HIGH NIBBLE ของ TMOD = 0001B) และใช้ตัวจับเวลา 1 ให้ทำการอินเตอร์รัพท์เมื่อเกิด Overflow และบรรจุค่า 16 บิตไปใหม่ ในกรณีต้องการอินเตอร์รัพท์ที่ตัวจับเวลา 1 จึงให้ IE.3 = 1

ในกรณีตัวจับเวลา 1 กำลังทำงานที่บิต $C/\overline{T}$ = 0 อัตราการนับเป็น 1/12 ของความถี่ Oscillator ถ้าตัวจับเวลาทำงานที่บิต $C/\overline{T}$ = 1 อัตราการนับจะใช้ความถี่ภายนอกที่ส่งเข้ามา ซึ่งจะมีความถี่สูงสุดที่จะใช้ได้คือ 1/24 ของความถี่ Oscillator

2.13.2 แหล่งกำเนิดสัญญาณนาฬิกาของตัวจับเวลา/ตัวนับ2(ใช้สำหรับ 8032/8052 เท่านั้น)

ตัวจับเวลา/ตัวนับ 2 มีขนาด 16 บิตที่มีเฉพาะในเบอร์ 8052 เท่านั้น เช่นเดียวกับตัวจับเวลา 0 และ 1 สามารถใช้งานในโหมดตัวจับเวลาหรือเป็นตัวนับก็ได้ซึ่งจะถูกโปรแกรมใช้งานได้ที่บิต $C/\overline{T2}$ ใน SFR Register T2CON โหมดการทำงานของตัวจับเวลา 2 นี้มี 3 โหมดด้วยกัน คือโหมด CAPTURE บรรจุใหม่อัตโนมัติ (AUTO- LOAD) และเป็นตัวสร้างอัตราบ๊อด (Baud Rste Generator) ซึ่งสามารถเลือกโปรแกรมได้ด้วยบิตใน T2CON ตามตาราง

RCLK + TCLK	$CP/\overline{RL2}$	TR2	โหมด
0	0	1	16 บิต บรรจุใหม่อัตโนมัติ
0	1	1	16 บิต CAPTURE
1	X	1	ตัวสร้างอัตราบ๊อด
x	x	0	

ตารางโหมดการทำงานตัวจับเวลา 2

ส่วนโหมดตัวสร้างอัตราบ๊อดจะโปรแกรมทำงานได้ด้วยการเซตค่า RCLK และหรือ TCLK=1

อัตราบ๊อดจะทำได้ด้วยอัตรา Overflow ที่เกิดขึ้นโดยไม่คำนึงถึงสถานะของ SMOD เพราะพอร์ตอนุกรมต้องการสัญญาณนาฬิกา 10 ลูกต่อหนึ่งบิต การคำนวณอัตราบ๊อด จะหาได้ด้วยสมการต่อไปนี้ ขณะที่ $C/\overline{T2}$ = 0

อัตราบ๊อด = ความถี่ Oscillator /[2*16 * ตัวหาร]

ตัวหารคือค่า 65536 ลบด้วยค่าที่โปรแกรมไว้ใน Register RCAP2H และRCAP2L ที่ใช้เป็นค่าบรรจุใหม่โดยอัตโนมัติ ซึ่งจากสูตรนี้ได้ตั้งแต่ 5.72 บ๊อดถึง 375 กิโลบ๊อด ที่ความถี่ 12

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MHz Oscillator เมื่อ $C/\overline{T2}$ = 1 อัตรา Overflow จะเท่ากับอัตราการนับของความถี่จากภายนอกที่เข้ามาหารด้วยตัวหารความถี่สูงสุดจากภายนอก จะมีค่าเป็น 1/24 ของความถี่ Oscillator

2.14 การอินเตอร์รัพท์

โดยทั่วไปความสามารถในการควบคุมการอินเตอร์รัพท์ เป็นการทำงานชนิดหนึ่งของ CPU ที่จะศึกษาถึงความสามารถและเทคนิคการทำงาน การอินเตอร์รัพท์ของ Single Ship มีความสัมพันธ์กันอย่างใกล้ชิดระหว่างอุปกรณ์ต่อพ่วงกับระบบ การทำงานของอุปกรณ์ต่อพ่วงเหล่านี้ มีระบบ Hardware ที่ช่วยให้การส่งสัญญาณ Real Time กับมอนิเตอร์ได้อย่างต่อเนื่อง โดยปราศจากการรบกวนต่อสัญญาณการทำงานของ CPU ตัวอย่างเช่น ขณะที่มีการรับสัญญาณอนุกรมจาก CRT ที่ตัวหนึ่งก็จะมีการส่งสัญญาณไปยังอุปกรณ์ตัวอื่นและตัวจับเวลา/ตัวนับ ก็จะนับพัลส์การเปลี่ยนแปลงที่เข้ามาอย่างรวดเร็วไปพร้อมกันได้ด้วย ในขณะตัวจับเวลา/ตัวนับอีกตัวก็กำลังวัดความกว้างของพัลส์ที่เข้ามา

CPU ตัวนี้จะรู้ได้อย่างไรว่า เมื่อไรถึงจะมีการรับและส่งสัญญาณอนุกรม CRT หรือให้ตัวจับเวลา/ตัวนับ มีการนับจำนวนและวัดความกว้างของพัลส์ว่าจะสิ้นสุดลงเมื่อไร

ตัวโปรแกรม MCS-51 สามารถที่จะเลือกการโปรแกรมได้ 3 วิธีด้วยกัน คือพิจารณาโปรแกรมตัว Register TCON และ SCON ที่ประกอบด้วยสถานะบิตที่ถูกเซตทาง Hardware เมื่อตัวจับเวลาตัวหนึ่งเกิด Overflow หรือเมื่อการรับส่งข้อมูลที่พอร์ตอนุกรมสิ้นสุดลง

เทคนิคการโปรแกรมวิธีแรกก็โดยการอ่านสถานะของ Register ควบคุมเข้าไปยัง Accumulator แล้วทดสอบสถานะบิตตามสถานะตามลักษณะการทำงานนั้นๆ แล้วทำการกระโดดไปยังโปรแกรมย่อยตามผลที่เกิดขึ้นชนิดนั้นๆ ลักษณะการทดสอบร่วมกันครั้งละหลายลักษณะงานเช่นนี้เปรียบเสมือนตัวโปรแกรมใช้ระบบ Microprocessor หลายตัวควบคุมชิปอุปกรณ์ต่อพ่วงต่าง ๆซึ่งผู้โปรแกรมจะต้องทำความเข้าใจอย่างลึกซึ้งถึงระบบและจังหวะที่เกิดในแต่ละงาน และการทดสอบแต่ละครั้งจะใช้คำสั่งไม่น้อยกว่า 3 คำสั่ง

วิธีที่สอง MCS-51 สามารถที่จะทำงานด้วยการกระโดดไปตามสถานะของการควบคุม หรือสถานะบิตของ Register ควบคุมงานหรือการรับสัญญาณที่เข้ามาตามขา Input แต่ละบิตด้วยการใช้คำสั่งเพียงคำสั่งเดียว ดังนั้นลักษณะงานสี่อย่างก็สามารถที่ใช้คำสั่งตรวจสอบได้ภายในสี่คำสั่ง ซึ่งจะใช้เวลาประมาณภายใน 8 ไมโครวินาที

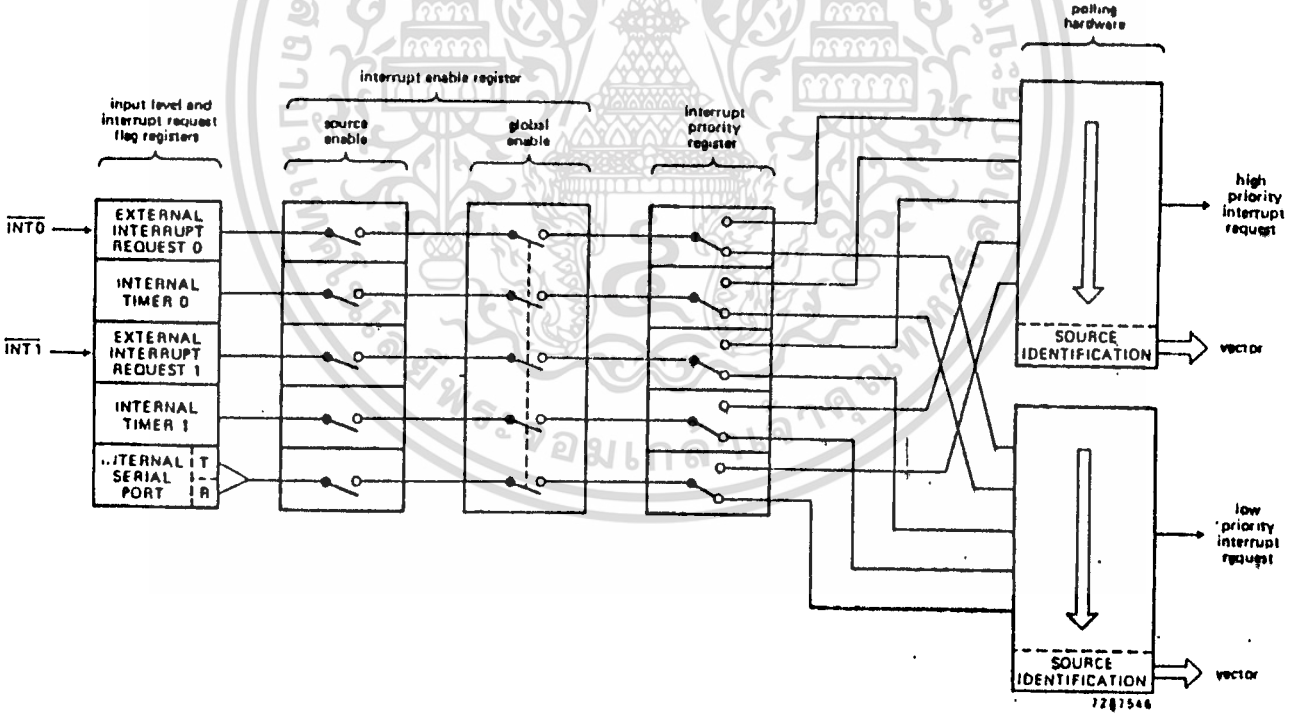
แต่วิธีทั้งสองที่กล่าวมาแล้ว จะต้องใช้ CPU มาทำการตรวจสอบบิตสถานะต่าง ๆ อยู่ตลอดเวลาลองเปรียบตัว CPU เหมือนกับผู้จัดการของบริษัทซึ่งจะบริหารงานในบริษัทให้ก้าวหน้าได้อย่างดีนั้นจะต้องมีเวลาทำงานให้กับหน้าที่หลักของตัวเอง เช่นในการวางแผนต่าง ๆ ให้กับบริษัทได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อย่างต่อเนื่อง และใช้เวลาเพียงบางส่วนสำหรับพนักงานที่จะเข้ามาขัดจังหวะเพื่อขอบริการแก้ไขปัญหาเพียงบางเวลาที่จำเป็นเท่านั้น เช่นเดียวกันแทนที่จะใช้ CPU ทำงานในลักษณะที่ออกไปตรวจสอบสถานะการทำงานของอุปกรณ์ต่อพ่วงรอบข้างต่างๆ ที่ต้องการจะใช้บริการก็จะใช้บริการต่อพ่วงเป็นฝ่ายร้องขอการบริการเข้ามาที่ CPU แทน ซึ่งเมื่อ CPU ถูกร้องขอเข้ามาก็จะปล่อยงานเดิมและเข้าสู่การบริการที่อุปกรณ์ต่อพ่วงชนิดนั้นๆ ได้ร้องขอเข้ามาชั่วระยะเวลาหนึ่งแล้วจึงกลับเข้าทำงานหลักต่อไป เมื่อสิ้นสุดงานบริการนั้นแล้วทำให้รู้สึกว่าตัว Microprocessor ทำงานพร้อมกันได้หลายงานในเวลาเดียวกัน

การใช้วิธีที่สาม จะเป็นวิธีที่ดีที่สุดในการใช้งานในลักษณะนี้ ด้วยการอินเตอร์รัพต์ทาง Hardware

แผนภูมิทาง Hardware ของการอินเตอร์รัพต์ชนิดต่างๆ ตามรูปที่ 2.24 จะมีรายละเอียดดังต่อไปนี้



รูปที่ 2.24 แผนภูมิทาง Hardware ของการอินเตอร์รัพต์ชนิดต่างๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แบบการอินเตอรร์รัพต์จากภายนอก $\overline{INT0}$ และ $\overline{INT1}$ สามารถจะใช้สัญญาณทั้งแบบขอบขาลงหรือระดับแรงดันต่ำขึ้นอยู่กับการกำหนดบิต IT0 และ IT1 ของ Register TCON แฟลกที่เกี่ยวข้องกับการอินเตอร์รัพต์ภายนอกคือ IE0 และ IE1 ใน Register TCON เมื่อเกิดสัญญาณการอินเตอร์รัพต์จากภายนอก. แฟลกดังกล่าวก็จะถูกทำให้เป็น '0' ด้วย Hardware เป็นการดิสเอเบิ้ลการอินเตอร์รัพต์ซ้อนกัน และเป็นการบอกว่าเป็นการอินเตอรร์รัพต์ที่เกิดจากสัญญาณอินเตอร์รัพต์แบบขอบขาลง ถ้าเป็นแบบการอินเตอร์รัพต์ด้วยสัญญาณระดับต่ำ ก็จะเป็นการร้องขอแฟลกให้เป็น '0' ตลอดเวลาที่สัญญาณเข้าที่ขาอินเตอร์รัพต์ยังคงเป็นศูนย์

การอินเตอร์รัพต์ตัวจับเวลา 0 และตัวจับเวลา 1 จะทำงานด้วยค่าของแฟลก TF0 และ TF1 ที่เซ็ตเป็น '1' เมื่อค่าใน Register ของตัวจับเวลา/ตัวนับถูกเพิ่มจาก '1' หมดทุกบิตกลายเป็น '0' หมดทุกบิต(การตั้งการอินเตอร์รัพต์ของตัวจับเวลา 0 จะตั้งให้ทำงานอินเตอร์รัพต์ไม่ได้) ถ้าให้ทำงานในโหมด 3 หลังจากการเกิดการอินเตอร์รัพต์ตัวจับเวลาแฟลกดังกล่าวจะถูกเคลียร์ให้เป็น '0' เมื่อ Hardware บนชิปเข้าทำงานในโปรแกรมบริการของการอินเตอร์รัพต์ตัวจับเวลาแล้ว

Register การอินเตอร์รัพต์อีนาเบิ้ล : IE

EA	X	ET2	ES	ET1	EX1	ET0	EX0

สัญญลักษณ์	ตำแหน่งบิต	ฟังก์ชั่น
EA	IE.7	จะดิสเอเบิ้ลการอินเตอร์รัพต์ทั้งหมด ถ้า EA = 0 จะไม่มีการอินเตอร์รัพต์ในการตอบรับ ถ้า EA = 1 สามารถที่จะอินเตอร์รัพต์ได้ โดยแต่ละแหล่งอินเตอร์รัพต์จะมีอิสระในการเซตหรือเคลียร์ให้อีนาเบิ้ลแต่ละบิตก่อนได้
-	IE.6	สำรอง
ET2	IE.5	จะอีนาเบิ้ลหรือดิสเอเบิ้ลอินเตอร์รัพต์ Overflow ของตัวจับเวลา 2 ถ้า ET = 0 การอินเตอร์รัพต์ตัวจับเวลา 2 จะดิสเอเบิ้ล
ES	IE.4	จะอีนาเบิ้ลหรือดิสเอเบิ้ลอินเตอร์รัพต์พอร์ตอนุกรม ถ้า ES = 0 การอินเตอร์รัพต์พอร์ตอนุกรมจะดิสเอเบิ้ล
ET1	IE.3	จะอีนาเบิ้ลหรือดิสเอเบิ้ลการอินเตอร์รัพต์ Overflow ของตัวจับเวลา 1 ถ้า ET1 = 0 การอินเตอร์รัพต์ตัวจับเวลา 1 จะดิสเอเบิ้ล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

EX1	IE.2	จะอีนาเบิ้ลหรือดิสเอเบิ้ลอินเตอร์รัพต์จากภายนอก 1 ถ้า EX1 =0 การอินเตอร์รัพต์จากภายนอก 1 จะดิสเอเบิ้ล
ET0	IE.1	จะอีนาเบิ้ลหรือดิสเอเบิ้ลการอินเตอร์รัพต์ Overflow ของตัวจับเวลา 0 ถ้า ET0 = 0 การอินเตอร์รัพต์ตัวจับเวลา 0 จะดิสเอเบิ้ล
EX0	IE.0	จะอีนาเบิ้ลหรือดิสเอเบิ้ลอินเตอร์รัพต์จากภายนอก 0 ถ้า EX0 =0 การอินเตอร์รัพต์จากภายนอก 0 (INT0) จะดิสเอเบิ้ล

การอินเตอร์รัพต์ของการส่งข้อมูลอนุกรม จะเกิดขึ้นเมื่อ OR กันทางตรรกด้วยสัญญาณจากแฟลกของ RI กับ TI แฟลกทั้งสองจะไม่ถูกเคลียร์ด้วย Hardware เมื่อ Hardware บนชิบเข้าทำงานตามโปรแกรมบริการของการอินเตอร์รัพต์การส่งข้อมูลอนุกรม บิตในแฟลกของ RI กับ TI จะถูกเคลียร์ได้ด้วย Software เท่านั้น

เฉพาะตัว 8052 จะมีการอินเตอร์รัพต์ด้วยตัวจับเวลา 2 ด้วยการ OR กันทางตรรกด้วยสัญญาณจากแฟลกของ TF2 กับ EXF2 เช่นแฟลกทั้งสองจะไม่ถูกเคลียร์ด้วย HArdware เมื่อ Hardware บนชิบเข้าทำงานตามโปรแกรมบริการของตัวจับเวลา 2 โดยที่โปรแกรมบริการนี้จะต้องแยกให้ถูกว่าเป็นการบริการของ TF2 หรือ EXF2 ส่วนบิตในแฟลกของ TF2 และ EXF2 จะถูกเคลียร์ได้ด้วย Software

บิตทุกบิตที่จะทำให้การอินเตอร์รัพต์ทำงานได้สามารถที่จะเซตหรือเคลียร์ได้ด้วย Software ซึ่งจะให้ผลเช่นเดียวกับการเซตหรือเคลียร์ด้วย Hardware การอินเตอร์รัพต์สามารถที่จะถูกกำหนดให้ทำงานหรือยกเลิกได้ ในขณะใดขณะหนึ่งของการทำงาน ด้วยการควบคุมกำหนดด้วย Software

Register สำคัญความสำคัญการอินเตอร์รัพต์ : IP

-	-	PT2	PS	PT1	PX1	PT0	PX0

สัญญาลักษณ์	ตำแหน่งบิต	ฟังก์ชัน
-	IP.7	สำรอง
-	IP.6	สำรอง
PT2	IP.5	กำหนดความสำคัญการอินเตอร์รัพต์ตัวจับเวลา 2 ถ้า PT2 =1 เป็นการโปรแกรมให้มีระดับการอินเตอร์รัพต์ความสำคัญสูงกว่า
PS	IP.4	การโปรแกรมให้มีระดับการอินเตอร์รัพต์พอร์ตอนุกรม ถ้า PS=1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นการโปรแกรมให้มีระดับการอินเตอร์รัพต์ความสำคัญสูงกว่า

PT1	IP.3	กำหนดระดับความสำคัญอินเตอร์รัพต์ตัวจับเวลา 1
PX1	IP.2	กำหนดระดับความสำคัญอินเตอร์รัพต์ภายนอก 1 ($\overline{INT1}$)
PT0	IP.1	กำหนดระดับความสำคัญอินเตอร์รัพต์ตัวจับเวลา 0
PX0	IP.0	กำหนดระดับความสำคัญอินเตอร์รัพต์ภายนอก 0 ($\overline{INT0}$)

แหล่งกำเนิดการอินเตอร์รัพต์แต่ละชนิดสามารถที่จะอีนาเบิ้ลหรือดีสเอเบิ้ล ด้วยการเซตหรือเคลียร์ค่าบิตภายในตัว Register IE บิต EA ภายใน Register IE จะเป็นตัวควบคุมการอินเตอร์รัพต์ทุกชนิดให้เริ่มทำงาน ส่วนบิต IE.6 เป็นบิตสำรองผู้ใช้ไม่สามารถที่จะเซตหรือเคลียร์บิตนี้ได้เช่นเดียวกันในตัว 8051 บิต IE.5 ก็จะเป็นบิตสำรองที่จะเซตไม่ได้

2.14.1 โครงสร้างลำดับความสำคัญการอินเตอร์รัพต์

แต่ละแหล่งอินเตอร์รัพต์สามารถที่จะโปรแกรมให้มีระดับ Priority สูงหรือต่ำได้ด้วยการเซตหรือเคลียร์ค่าบิตต่างๆใน IP ของ SFR โดยที่ตัวแฟลกอินเตอร์รัพต์ทุกตัวสามารถเซตหรือเคลียร์ได้ด้วย Software ซึ่งจะมีผลเช่นเดียวกับผลที่เกิดขึ้นจาก Hardware

การอินเตอร์รัพต์ความสำคัญต่ำ สามารถที่ถูกอินเตอร์รัพต์ด้วยตัวอินเตอร์รัพต์จากความสำคัญสูงกว่า แต่ไม่สามารถที่จะถูกอินเตอร์รัพต์จากตัวอื่นที่มีความสำคัญต่ำ การอินเตอร์รัพต์ตัวที่มีความสำคัญสูงสุดไม่สามารถที่จะถูกอินเตอร์รัพต์ได้ด้วยการทำงานตามกฎเหล่านี้ ระบบการอินเตอร์รัพต์จะประกอบด้วยตัวที่ไม่สามารถกำหนด Address สองตัวคือ "Priority Level Active" กับ "Flip - Flop" ตัวหนึ่งเป็นตัวแสดงถึงการอินเตอร์รัพต์ความสำคัญสูงกำลังได้รับการบริการและการอินเตอร์รัพต์ตัวอื่นจะถูกกันไว้หมด อีกตัวเป็นการแสดงถึงการอินเตอร์รัพต์ความสำคัญต่ำกำลังได้รับบริการและกันตัวอื่นหมด แต่การอินเตอร์รัพต์ที่มีความสำคัญสูงยังคงเข้าทำงานต่อได้

ในเหตุการที่มีการร้องขอของระดับความสำคัญเดียวกันถูกรับเข้ามาพร้อมกัน การหาลำดับการใช้ร่วมกันก่อนหลังภายในเมื่อการร้องขอได้รับบริการ ดังนั้นระดับความสำคัญภายในแต่ละอันจะมีการกำหนดระดับโครงสร้างความสำคัญการอินเตอร์รัพต์ที่มีลำดับการใช้ร่วมกันก่อนหลังดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แหล่งที่มาการอินเตอร์รัพท์	ลำดับความสำคัญภายใน
การอินเตอร์รัพท์ 0 จากภายนอก (IE0)	(สูงสุด) 1
การเกิด Overflow ของตัวจับเวลา/ตัวนับ 0 (TF0)	2
การอินเตอร์รัพท์ 1 จากภายนอก (IE1)	3
การเกิด Overflow ของตัวจับเวลา/ตัวนับ 1 (TF1)	4
พอร์ตอนุกรม (RI + TI)	5
การเกิด Overflow ของตัวจับเวลา/ตัวนับ 2 การเกิด Overflow ของตัวจับเวลา 2 /ขอบขา ลงบน T2EX(ใช้ใน 8032/8052เท่านั้น)(TF2+EXF2)	(ต่ำสุด) 6

อินเตอร์รัพท์แฟลกจะถูก Sampling ทุกช่วงวัฏจักรแมคชีนช่วง S5P2 ตัวที่ถูก Sampling จะเข้ามาได้ในช่วงวัฏจักรแมคชีนตัวต่อมา ถ้าแฟลกตัวหนึ่งถูกเซตในช่วง S5P2 ของช่วงวัฏจักรตัวต่อมา ในช่วงวัฏจักรการตรวจพบนี้ระบบการอินเตอร์รัพท์จะเริ่มทำงานด้วยการทำคำสั่ง LCALL เพื่อที่จะนำเข้าสู่การบริการการอินเตอร์รัพท์ตามชนิดนั้นๆ โดยสภาวะทาง Hardware คำสั่ง LCALL จะไม่ถูกกันออก เมื่อเกิดเหตุการณ์ต่อไปนี้

1. การอินเตอร์รัพท์ของระดับความสำคัญที่เท่ากันหรือสูงกว่ากำลังทำงานจะสิ้นสุดแล้ว

2. วัฏจักรแมคชีนที่เกิดระหว่างนี้ จะไม่เป็นวัฏจักรสุดท้ายในการทำงานตามคำสั่งที่กำลังทำอยู่ หรือการอินเตอร์รัพท์ที่ร้องขอมาจะยังไม่ได้รับการตอบสนองจนกว่าการทำคำสั่งขณะนั้นจะสิ้นสุดลงสมบูรณ์

3. คำสั่งในขณะนั้นเป็น RETI หรือการเขียนโปรแกรมเข้า Register IE หรือ IP ของ SFR หรือการร้องขอการอินเตอร์รัพท์จะไม่ได้รับการตอบรับหลังคำสั่ง RETI หรือหลังการอ่านและเขียนเข้า Register IE หรือ IP จะได้รับการตอบรับจนกว่าจะต้องทำคำสั่งอย่างน้อยหนึ่งคำสั่งไปแล้ว

ถ้ากรณีใดๆกรณีหนึ่งจากข้างบนนี้เกิดขึ้น ผลของการใช้อินเตอร์รัพท์จะถูกกันออกหลังการใช้ LCALL ถ้าไม่เกิดกรณีใดจากข้างบนนี้ปรากฏผลของการใช้อินเตอร์รัพท์จะทำงานช่วงวัฏจักรแมชชีนลูกต่อมา

2.14.2 การตอบสนองการอินเตอร์รัพท์ Protocol

ตัว Processor จะตอบสนองการร้องขอด้วยการเซต Active ระดับความสำคัญของการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อินเตอร์รัพต์ Flip-Flop แล้วจึงจะทำงานทาง Hardware ในการเรียกโปรแกรมย่อยที่บริการการอินเตอร์รัพต์นั้นๆ มันจะเคลียร์แฟลกการร้องขออินเตอร์รัพต์ (ยกเว้นมันจะไม่เคลียร์ $\overline{\text{INT0}}$ หรือ $\overline{\text{INT1}}$ เพราะมันไม่ได้ควบคุมแหล่งสัญญาณเหล่านี้ และมันจะไม่เคลียร์ TI,RI,TF2 หรือ EXF2) การเรียกโปรแกรมย่อยทาง Hardware จะทำการ PUSH ค่าข้อมูลตัวนับโปรแกรมเข้าที่บริเวณ Stack แต่จะไม่เก็บค่า PSW เข้าไปด้วย (และ PC จะถูกบรรจุใหม่ด้วยค่า Address ที่ขึ้นอยู่กับแหล่งขนิดการร้องขอการอินเตอร์รัพต์) ซึ่งจะมีดังนี้

แหล่งการอินเตอร์รัพต์	ตำแหน่ง Vector Address
External Interrupt 0 (IE0)	0003H
Timer 0 Overflow (TF0)	000BH
External Interrupt 1 (IE0)	0013H
Timer 1 Overflow (TF1)	001BH
Serial Port (RI + TI)	0023H
Timer 2 Overflow/Negative Transition On T2EX (ของเบอร์ 8032/8052 เท่านั้น)	002BH

การทำงานจะเริ่มทำงานจาก Vector Address ที่ให้มาจนกระทั่งถึงคำสั่ง RETI ซึ่งอยู่ในโปรแกรมย่อยการบริการอินเตอร์รัพต์ต่างๆดังกล่าวข้างบน คำสั่ง RETI จะเคลียร์ Active ระดับความสำคัญของการอินเตอร์รัพต์ของ Flip - Flop ที่ถูกเซตเมื่อได้รับการร้องขอการอินเตอร์รัพต์ในตอนแรก ดังนั้นมันจะ POP เอายอดของ Stack สองไบต์มาบรรจุใหม่ใน PC เพื่อจะกลับเข้าทำงานโปรแกรมเดิมก่อนที่จะถูกอินเตอร์รัพต์ต่อไป

2.14.3 การอินเตอร์รัพต์จากภายนอก

แหล่งกำเนิดภายนอกสามารถที่จะถูกโปรแกรมเลือกระดับการ Active หรือช่วงการเปลี่ยนแปลงด้วยการเซตหรือเคลียร์บิตที่ IT1 หรือ IT0 ใน Register TCON ถ้า ITX = 0 การอินเตอร์รัพต์ภายนอก x จะถูกกระตุ้นด้วยการกระตุ้นระดับต่ำที่ขา INTx แต่ถ้า ITx = 1 การอินเตอร์รัพต์ภายนอก x เป็นการกระตุ้นใช้ขอบสัญญาณขาลง ในโหมดนี้ถ้าตัวอย่างสัญญาณของขา $\overline{\text{INTx}}$ แสดงถึงระดับสูงในวัฏจักรสูงหนึ่งและต่ำในวัฏจักรอีกลูกหนึ่ง แฟลกการร้องขออินเตอร์รัพต์ IEx ใน Register TCON จะถูกเซต ดังนั้นแฟลกบิตของ IEx จะเป็นการแสดงถึงการร้องขออินเตอร์รัพต์

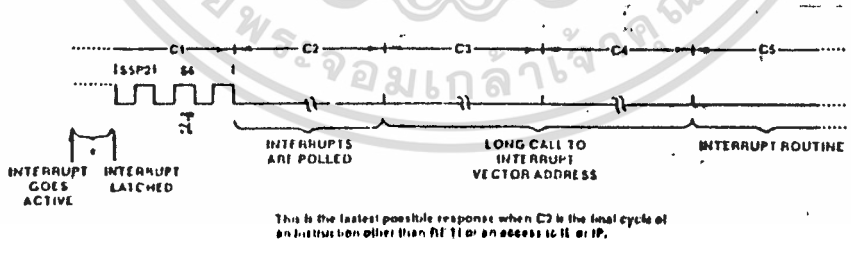
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพราะสัญญาณที่ขาการอินเตอร์รัพท์จะถูกสุ่มตัวอย่างหนึ่งครั้งในแต่ละวัฏจักรแมคชีน สัญญาณที่เข้ามาจะต้องรักษาระดับสูงหรือต่ำอย่างน้อยภายในช่วง 12 คาบของความถี่ Oscillator เพื่อให้มั่นใจในการสุ่มตัวอย่างที่รับรู้ได้ค่าแน่นอน ถ้าการอินเตอร์รัพท์ภายนอกถูก Active เปลี่ยนแปลง แหล่งสัญญาณภายนอกจะต้องรักษาค่าการร้องขอสถานะสูงเป็นเวลาอย่างน้อยหนึ่งลูก และรักษาค่าสถานะต่ำอีกอย่างน้อยเป็นเวลาหนึ่งวัฏจักร เพื่อให้แน่ใจว่าการเปลี่ยนแปลงค่าจะสามารถทำให้เพลกการร้องขออินเตอร์รัพท์ของ IEx จะถูกเซต ค่าใน IEx จะถูกเคลียร์อัตโนมัติด้วย CPU เมื่อโปรแกรมการอินเตอร์รัพท์ถูกเรียกมาใช้ใหม่

ถ้าการอินเตอร์รัพท์ภายนอกอยู่ในระดับการ Active แหล่งภายนอกจะเก็บการ Active การร้องขอไว้ จนกว่าสัญญาณการร้องขออินเตอร์รัพท์จะถูกสร้างขึ้นมาเรียบร้อยแล้ว แล้วมันจะต้องกลับมารับการ Active การร้องขอใหม่ ก่อนที่การทำงานการบริการอินเตอร์รัพท์เดิมจะสิ้นสุดลง หรือการอินเตอร์รัพท์อีกลูกหนึ่งจะถูกสร้างขึ้นมาใหม่

2.14.4 ช่วงเวลาการตอบสนอง

ระดับของ $\overline{\text{INT0}}$ และ $\overline{\text{INT1}}$ จะถูก Invert และแลทช์เก็บไว้ในเพลกบิต IE0 และ IE1 ภายในช่วง S5P2 ของทุกๆวัฏจักรแมคชีน ค่าที่เก็บจะยังไม่นำมาใช้ด้วยวงจรจนกว่าจะถึงวัฏจักรแมคชีนลูกใหม่ ถ้าการร้องขอครั้งหนึ่ง Active และข้อแม้ต่างๆถูกต้องสำหรับการให้มีการตอบรับ Hardware ก็จะเรียกโปรแกรมย่อยเพื่อการตอบรับการบริการการร้องขอของอินเตอร์รัพท์ ในคำสั่งต่อมาของโปรแกรมการบริการจะถูกทำงาน การเรียกโปรแกรมด้วยตัวเองจะใช้เวลาสองลูกคลื่น ดังนั้นจะต้องใช้อย่างน้อยสามวัฏจักรแมคชีนช่วงระหว่างการเริ่มการร้องขออินเตอร์รัพท์ ดังรูปที่ 2.25 จะแสดงช่วงเวลาการตอบสนองการอินเตอร์รัพท์



This is the fastest possible response when C2 is the final cycle of an instruction other than RETI or an access to IE or IP.

รูปที่ 2.25 แสดงช่วงเวลาการตอบสนองการอินเตอร์รัพท์

ช่วงเวลาการตอบสนองที่ยาวนานกว่าอาจเกิดขึ้นได้ ถ้าการร้องขอถูกบล็อกด้วยข้อแม้ต่างๆของการจัดลำดับความสำคัญของการอินเตอร์รัพท์สามกรณีที่กล่าวแล้ว ถ้าการอินเตอร์รัพท์ที่มีความสำคัญเท่ากันหรือสูงกว่าได้ทำงานสมบูรณ์ไปแล้ว ช่วงเวลาที่รอทั้งหมดจะขึ้นอยู่กับการใช้เวลาของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การอินเตอร์รัพต์นั้นๆ ถ้ากำลังทำคำสั่งอยู่ที่ยังไม่ถึงวงรอบสุดท้าย ช่วงเวลาที่รอทั้งหมดจะไม่สามารถมีค่าเกิน 3 วัฏจักรแมคชีนเพราะคำสั่งที่ใช้เวลานานที่สุด เช่น MUL และ DIV จะใช้ 4 วัฏจักรแมคชีน

ถ้าอยู่ในช่วงกำลังทำคำสั่ง RETI หรือกำลังเข้าถึง Register IE หรือ IP เวลาที่รอทั้งหมดก็จะไม่มีค่าเกินกว่า 5 วัฏจักร โดยคำสั่งสูงสุดจะคิดที่ 1 วัฏจักรในช่วงที่กำลังทำงานคำสั่งที่เกิดการอินเตอร์รัพต์อยู่และบวกอีก 4 วัฏจักรเพื่อให้สิ้นสุดคำสั่งต่อมา ถ้าคำสั่งต่อมาคือคำสั่ง MUL หรือ DIV

ดังนั้นในระบบการอินเตอร์รัพต์ครั้งหนึ่ง ช่วงเวลาตอบสนองจะอยู่ระหว่าง 3 ถึง 8 วัฏจักรเสมอ

2.15 การทำงานครั้งละหนึ่งขั้นตอน (Single Step)

โครงสร้างการอินเตอร์รัพต์ของ 8051 ช่วยให้สามารถทดสอบโปรแกรมสั้นๆได้ ด้วยการทำงานที่ละคำสั่งที่กล่าวมาแล้ว การร้องขออินเตอร์รัพต์จะไม่ตอบสนองในการขณะที่กำลังทำงานอินเตอร์รัพต์ในระดับความสำคัญเดียวกัน มันจะไม่ตอบสนองหลังคำสั่ง RETI จนกว่าอย่างน้อยที่สุดคำสั่งได้ทำงานไปแล้วอีกคำสั่งหนึ่ง ดังนั้นในการอินเตอร์รัพต์ครั้งหนึ่งๆที่เข้ามา มันจะไม่สามารถรับเข้ามาใหม่ได้ จนกว่าอย่างน้อยที่สุดได้ทำงานไปแล้วหนึ่งคำสั่งของโปรแกรมอินเตอร์รัพต์ ทางหนึ่งที่ใช้ประโยชน์ของลักษณะเช่นนี้คือ การทำงานแบบครั้งละหนึ่งขั้นตอนด้วยการโปรแกรมการอินเตอร์รัพต์ภายนอกในแต่ละครั้งให้ Active ด้วยระดับและโปรแกรมบริการอินเตอร์รัพต์จะให้สิ้นสุดด้วยรหัสต่อไปนี้

```
JNB   P3.2,B    ;ให้รอที่นี่จนกว่า INT0 จะเข้าสู่ระดับสูง
JB    P3.2,B    ;ให้รอที่นี่จนกว่า INT0 จะเข้าสู่ระดับต่ำ
RETI            ;กลับไปที่เก่าและทำงานหนึ่งคำสั่ง
```

ดังนั้นถ้าขา $\overline{\text{INT0}}$ ที่เป็นขา P3.2 มีค่าเป็นระดับต่ำ CPU จะไปยังโปรแกรมอินเตอร์รัพต์ภายนอกและยังคงอยู่ที่โปรแกรมการบริการอินเตอร์รัพต์จนกว่า $\overline{\text{INT0}}$ จะมีพัลส์ (คือระดับต่ำไปสูงสูงไปต่ำ) ดังนั้นมันจะทำงานคำสั่ง RETI และกลับไปยังโปรแกรมที่ทดสอบอยู่และทำงานหนึ่งคำสั่งและกลับเข้าสู่โปรแกรมการบริการอินเตอร์รัพต์ใหม่และรอจนกว่ามีพัลส์ลูกใหม่ที่ P3.2 การทำงานหนึ่งคำสั่งของโปรแกรมทดสอบจะควบคุมด้วยพัลส์ที่ขา P3.2 ในแต่ละลูก

2.16 Reset

การ Reset นับเป็นส่วนสำคัญของการที่จะทำให้ Microprocessor ทำงานได้อย่างน่าเชื่อถือ เพราะถ้าการให้จังหวะแก่ขา Reset ไม่ถูกต้องจะทำให้ตัว Microcontroller มีปัญหา ดังนั้นจึงควรที่จะทำความเข้าใจกับความต้องการจังหวะเวลาของ Microcontroller แต่ละชนิด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

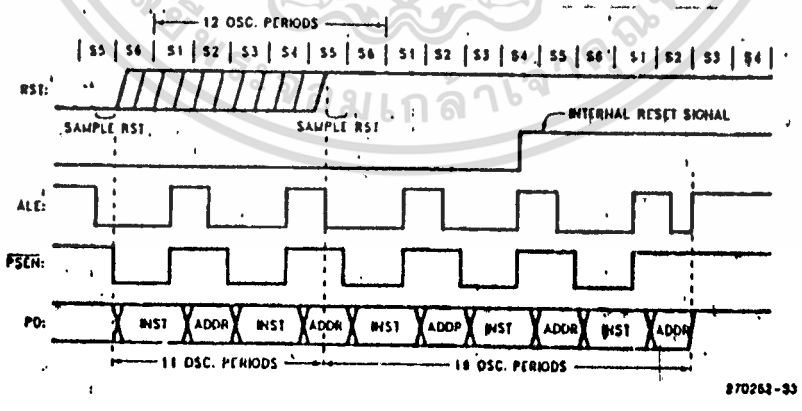
อย่างถูกต้อง เสียก่อนที่จะออกแบบวงจร Reset

วงจร Reset สำหรับรุ่น 8051 ที่เป็น HMOS จะต่อที่ขา Reset คือขา RST/VPD ดังรูปที่ 2.26 วงจร Schmitt Trigger จะถูกใช้เป็นตัว Input สำหรับทำจัดสัญญาณรบกวน (Noise) และที่ Output ของ Schmitt Trigger จะถูกสุ่มเก็บเข้าไบด้วยวงจร Reset เกิดที่ S5P2 ของทุกๆวัฏจักรแมคชีน

ข้อกำหนดวงจรสำหรับ CHMOS ดังแสดงในรูปที่ 2.28 RAM ภายในจะถูกจ่ายพลังไฟด้วย Battary สำรองจากขา Vcc

ทั้ง HMOS และ CHMOS จะทำงานได้ด้วยการ Reset สถานะสูงที่ขา RST/VPD เป็นเวลาอย่างน้อย 2 วัฏจักรแมคชีน นับจากที่ Oscillator เริ่มทำงาน ตัว CPU จะตอบสนองด้วยการสร้างสัญญาณ Reset ภายในที่แสดงแผนภูมิเวลาในรูปที่ 2.26 สัญญาณ Reset ภายนอกจะเป็นแบบ Asyncronous ต่อสัญญาณนาฬิกาภายใน โดยที่สัญญาณ Reset จะถูก Sampling ช่วงระหว่าง S5P2 สำหรับทุกๆวัฏจักรแมคชีน ส่วนขา Port จะยังคงรักษาสภาวะตามสภาพขณะนั้นเป็นเวลา 19 คาบเวลาของ Oscillator หลังจากสภาวะสูงทาง Sampling จะใช้เวลา 19-31 คาบเวลาของ Oscillator หลังจากสัญญาณ Reset จากภายนอกเข้าขา RST

ขณะที่ RST มีสภาวะสูง ALE และ $\overline{\text{PSEN}}$ จะถูกดึงให้สถานะสูงแบบอ่อนหลังจาก RST ถูกดึงลงสถานะต่ำมันจะใช้เวลาประมาณ 1 ถึง 2 วัฏจักรแมคชีน สำหรับการเริ่มสัญญาณของ ALE และ $\overline{\text{PSEN}}$ ด้วยเหตุผลนี้อุปกรณ์อื่นๆไม่สามารถที่ Syncronous กับช่วงจังหวะภายในของ 8051



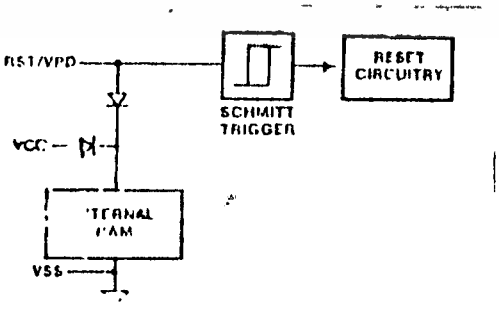
รูปที่ 2.26 แผนภูมิช่วงจังหวะ Reset

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลำดับการทำงานอัลกอริทึมของการ Reset ภายในก็โดยการเขียนค่า '0' ใส่ลงในบิตทุกตัวของ SFR ยกเว้น Port แลทช์ ตัวชี้สแต็กและ SBUF Port แลทช์จะถูกใส่ค่า FFH Stack มีค่า 07H และ SBUF ไม่ทราบค่า ดังแสดงค่าต่างๆใน Register ในตารางต่อไปนี้ หลังการ Reset

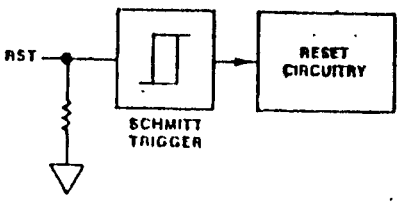
Register	มีค่าข้อมูลเป็น	Register	มีค่าข้อมูลเป็น
PC	000H	T2CON	00H
ACC	00H	TH0	00H
B	00H	TL0	00H
PSW	00H	TH1	00H
SP	07H	TL1	00H
DPTR	0000H	TH2	00H
P0-P3	0FFH	TL2	00H
IP	(XX000000)B	RLDH	00H
IE	(0X000000)B	RLDL	00H
TMOD	00H	SCON	00H
TCON	00H	SBUF	Intermediate
PCON(CMOS)	0XXX0000B	PCON(HMOS)	0XXXXXXXB

* RAM ภายในจะไม่มีผลจากการ Reset เมื่อมีแรงดันไฟจ่ายที่ Vcc ค่าต่างๆใน RAM จะเป็นค่าที่ไม่แน่นอน ถ้าส่วนนั้นไม่ได้กลับมาจากการใช้งานของโหมดลดพลังงาน (Reduced Power Mode)



รูปที่ 2.27 วงจรการจัดการ Reset ของ HMOS

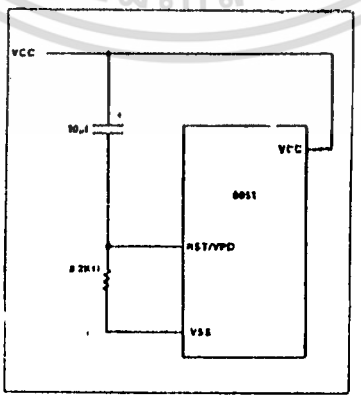
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.28 วงจรการจัดการ Reset ของ CHMOD

Power-on Reset

การ Reset อัตโนมัติสามารถที่จะทำได้เมื่อมีแรงดันไฟที่ V_{cc} ด้วยการต่อกับขา RST ผ่านคาปาซิเตอร์ขนาด 10 μF และต่อตัวต้านทานขนาด 8.2 Kohm ต่อคร่อมระหว่าง V_{ss} กับ RST เพื่อที่จะแบ่งให้ช่วงเวลาขอบขาขึ้นมีไม่เกินมิลลิวินาที และช่วงเวลาที่เกิด Oscillated เริ่มต้นจะไม่เกิน 10 มิลลิวินาที วัดยวงจรการ Reset แบบ Power-on มีแสดงดังรูปที่ 2.29 เมื่อเปิดไฟกระแสจะไหลผ่านเข้า RST ความต่างศักย์ระหว่าง V_{cc} กับแรงดันที่ตกคร่อมตัวคาปาซิเตอร์และจะลดลงจากค่า V_{cc} เป็นลักษณะชาร์ตเข้า ยิ่งถ้าค่าคาปาซิเตอร์มีขนาดใหญ่ขึ้น แรงดัน V_{rst} ที่คร่อม RST ก็จะลดช้าลง ค่า V_{rst} จะต้องรักษาระดับให้นานมากกว่าระดับเทสโฮล (Threshold) ของตัว Schmitt Trigger ที่พอจะมีเวลาให้การ Reset เกิดขึ้นสมบูรณ์ เวลาที่ต้องการคือช่วงการเกิด Oscillated เริ่มต้นบวกกับอีก 2 วัฏจักรแมคชีน



รูปที่ 2.29 วงจร Power-on Reset

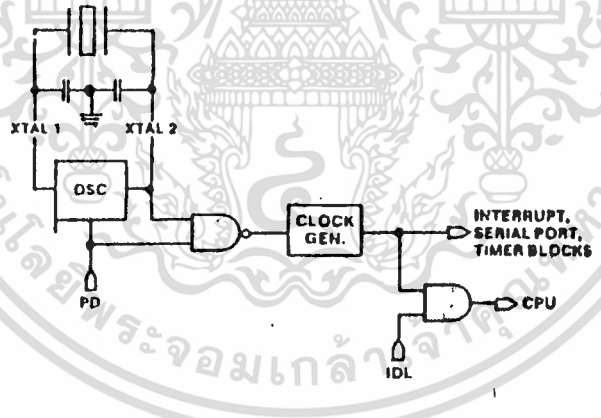
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.17 แหล่งพลังงานและข้อกำหนดการใช้พลังงาน

8051 มีการแบ่งอุปกรณ์บางส่วนสำหรับการใช้แหล่งจ่ายไฟสำรอง โดยใน HMOD จะใช้ขา RST/VPD เป็นตัวรับแรงดันไฟสำรอง และ CHMOSใช้ตัว Register ควบคุมพลังงาน PCON

2.17.1 การทำงานของ HMOS ในโหมดการประหยัดพลังงาน

ระหว่างการทำงานปกติ RAM ภายในจะดึงพลังงานจาก V_{cc} อย่างไรก็ตามตามรูปที่ 2.30 แรงดันที่ขา RST/VPD มีค่ามากกว่า V_{cc} มันจะใช้ RST/VPD เป็นตัวจ่ายไฟให้กับ RAM แทน ในลักษณะเช่นนี้ช่วยให้สามารถใช้แบตเตอร์รี่สำรอง เพื่อรักษาค่าข้อมูลใน RAM ไว้ได้เมื่อเกิดแรงดันตกหรือไฟดับ การใช้งานลักษณะนี้ระบบผู้ใช้จะต้องสามารถรับสัญญาณของแรงดันตกได้ ด้วยการใช้อินเตอร์รัพต์ของตัว Processor ทั้งขา $\overline{INT0}$ หรือ $\overline{INT1}$ เพื่อถ่ายเทข้อมูลต่างๆที่จำเป็นไว้ใน RAM และสามารถที่มีแบตเตอร์รี่สำรองจ่ายให้กับ RAM ได้ที่ขา RST/VPD ก่อนที่ V_{cc} จะมีแรงดันที่ตกลงถึงขีดจำกัดการทำงานของ CPU ได้ และเมื่อมีพลังงานกลับมาขา VPD ก็จำเป็นจะต้องคืนค่าสถานะสูงไว้ได้ จนสามารถที่มีผลให้เกิด Reset โดยใช้เวลาของการ Oscillated เริ่มต้นบวกกับ 2 วัฏจักรแมคชีนและเข้าสู่การทำงานปกติ



รูปที่ 2.30 วงจรที่ทำงานร่วมในโหมดพลังงานตก

2.17.2 การทำงานของ CHMOD ในโหมดการประหยัดพลังงาน

ในรุ่นของ CHMOD มีโหมดการประหยัดพลังงานให้เลือกทำงาน 2 แบบ คือแบบอุดมคติและแบบพลังงานจากสัญญาณเข้าที่ขาที่เรียกใช้พลังงานสำรอง คือ V_{cc} ไม่ใช่เข้าที่ขา Reset แบบรุ่น HMOD แผนภูมิวงจรทาง Hardware ที่ใช้งานลักษณะนี้เสดงดังรูปที่ 2.30 ในการใช้งานโหมดอุดมคติ (IDL=1) ตัว Oscillator ยังคงทำงานต่อไปและการอินเตอร์รัพต์ของพอรต์อนุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และวงจรของตัวจับเวลาจะยังคงทำงานด้วยสัญญาณนาฬิกาต่อไป แต่สัญญาณจะถูกบัฟไม่ให้ป้อนเข้า CPU ส่วนในการใช้โหมดพลังงานตก ตัว Oscillator จะไม่เข้าระบบคือ PD=1 ในการเลือกใช้โหมดทั้งสองจะสั่งให้การเซตบิตใน PCON ของ SFR ค่า Address ของตัวนี้คือ 87H

Register ควบคุมพลังงาน : PCON

SMOD	-	-	-	GF1	GF0	PD	IDL

SMOD PCON.7 เป็นการตั้งอัตราบ๊อดเร็วเป็น 2 เท่า เมื่อเซตเป็น '1' อัตราบ๊อดจะเป็น 2 เท่า เมื่อพอร์ตอนุกรมถูกใช้ให้ทำงานที่โหมด 1,2 หรือ 3

\- PCON.6 สำรอง

\- PCON.5 สำรอง

\- PCON.4 สำรอง

GF1 PCON.3 ใช้เป็นแฟลกบิตตามจุดประสงค์ทั่วไป

GF0 PCON.2 ใช้เป็นแฟลกบิตตามจุดประสงค์ทั่วไป

PD PCON.1 บิตการเลือกพลังงานตก บิตนี้ถูกเซตเมื่อเลือกการทำงานในโหมดพลังงานตก

IDL PCON.0 บิตการเลือกแบบอุดมคติ บิตนี้ถูกเซตเมื่อเลือกการทำงานในโหมดพลังงานตก

2.17.2.1 โหมดอุดมคติ

คำสั่งที่เซตค่า PCON.0 จะเป็นเหตุให้คำสั่งสุดท้ายที่จะกระทำก่อนเข้าโหมดอุดมคติ ในโหมดอุดมคติ สัญญาณนาฬิกาภายในจะถูกเกทไม่ให้เข้าสู่ CPU ส่วนฟังก์ชั่นการอินเตอร์รัพต์ตัวจับเวลาและพอร์ตอนุกรม จะยังคงได้รับสัญญาณที่จะทำงานได้ สถานะต่างๆใน CPU จะยังคงรักษาค่าข้อมูลภายใน Stack Pointer,Program Counter,Program Status Word,Accumulator และข้อมูลใน Register ตัวอื่นๆ ระหว่างการใช้โหมดอุดมคติ ขา Port ต่างๆจะยังคงรักษาค่าระดับสถานะที่มีอยู่ที่เดิม ขณะที่ขา ALE และ $\overline{PSEN}$ จะมีสถานะสูง

มี 2 วิธีการในการที่จะให้ออกจากการทำงานในโหมด ด้วยการเซต Active ให้มีการอินเตอร์รัพต์อินาเบิ้ลได้ ตัวบิต PCON.0 จะถูกเคลียร์ด้วยสัญญาณ Hardware เมื่อมีการอินเตอร์รัพต์เกิดขึ้นเป็นการสิ้นสุดโหมดอุดมคติ การอินเตอร์รัพต์จะได้รับการบริการและจะสิ้นสุดด้วยคำสั่ง RETI เป็นคำสั่งต่อมาก่อนที่ทำงานเข้าสู่โหมดอุดมคติอีก

เราสามารถที่จะใช้บิต GF0 และ GF1 เป็นตัวแสดงว่าเรากำลังใช้อินเตอร์รัพต์ในการทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

งานปกติหรือทำงานในโหมดอุดมคติ ตัวอย่าง เช่นคำสั่งที่จะให้ Active ในโหมดอุดมคติด้วยการปรับแฟลกตัวใดตัวหนึ่งทั้งสองตัว เมื่อโหมดอุดมคติถูกทำให้สิ้นสุดด้วยการอินเตอร์รัพต์ ในการโปรแกรมอินเตอร์รัพต์ก็สามารถที่จะตรวจสอบบิตของแฟลก เหล่านี้ได้

อีกวิธีหนึ่งในการทำให้โหมดอุดมคติสิ้นสุดก็ด้วยการกดปุ่ม Reset เพราะว่าสัญญาณนาฬิกาของ Oscillator ยังคงทำงานอยู่ การรีเซตทาง Hardware จะต้องรักษาระดับ Reset คงอยู่อย่างน้อย 2 วัฏจักรหรือ 24 คาบวงจร Oscillator ที่จะให้การ Reset สมบูรณ์

สัญญาณที่ขา RST จะเคลียร์บิต IDL ของ Register PCON โดยตรงที่จังหวะนี้ CPU จะกลับไปทำงานในส่วนของโปรแกรมที่ต่อจากคำสั่งการใช้โหมดอุดมคติ การทำงานตามคำสั่งช่วง 2 หรือ 3 วัฏจักร โปรแกรมอาจจะถูกใช้งานก่อนที่สัญญาณการ Reset ภายในจะเข้าควบคุม ดังนั้นจึงห้ามไม่ให้ใช้คำสั่งที่เกี่ยวกับการใช้งานถ่ายเทข้อมูล RAM ภายในที่ต่อจากคำสั่งการใช้ โหมดอุดมคติแต่การถ่ายเทข้อมูลในเลทช์ตาม Port ต่างๆจะไม่ถูกห้ามเพื่อไม่ให้เกิดความผิดพลาดคำสั่งหนึ่งคำสั่งหลังการใช้คำสั่งโหมดอุดมคติไม่ควรจะเป็นคำสั่งเกี่ยวกับการถ่ายเทข้อมูลไปยังขา Port โดยตรงหรือการเขียนข้อมูลไปยัง RAM ภายนอก

2.17.2.2 โหมดการประหยัดพลังงาน

คำสั่งที่เซตค่าของ PCON.1 จะเป็นเหตุให้คำสั่งสุดท้ายที่กระทำก่อนเข้าโหมดการประหยัดพลังงาน ในโหมดการประหยัดพลังงานตัว Oscillator บนชิปจะหยุดทำงานด้วยการหยุดสัญญาณนาฬิกาเช่นนี้ ทุกฟังก์ชั่นของชิปจะหยุดการกระทำแต่ RAM บนชิปและตัว Register SFR ส่วนขา ALE และ $\overline{PSEN}$ จะมีสถานะ Output ต่ำและสถานะขา Port ต่างๆจะยังคงมีค่าเท่ากับค่าที่มีอยู่ใน SFR ของ Port Register นั้นๆ

การที่จะให้สิ้นสุดในโหมดการประหยัดพลังงานจะกระทำได้ด้วยการกดปุ่ม Reset โดยที่การ Reset จะกำหนดค่าใหม่ทั้งหมดใน SFR แต่จะไม่เปลี่ยนค่าใน RAM ชิป

ในโหมดการทำงานประหยัดพลังงาน ค่า V_{cc} สามารถจะลดค่าลงเพื่อใช้พลังงานไฟน้อยที่สุด ที่ต้องระวังก็คือจะต้องแน่ใจว่าค่า V_{cc} จะไม่ถูกลดลงก่อนที่จะเข้าสู่การทำงานในโหมดการประหยัดพลังงานและ V_{cc} จะต้องมีค่าแรงดันคืนค่าเดิมก่อนที่จะให้โหมดการประหยัดพลังงานสิ้นสุดการทำงาน การ Reset จะเป็นการสิ้นสุดการใช้การประหยัดพลังงานและยังให้ตัว Oscillator มีอิสระในการทำงานด้วย การ Reset จะต้องไม่กระทำก่อนที่จะให้ V_{cc} มีแรงดันคงเดิมและจะต้องมีค่าแรงดันที่ยาวนานพอที่จะให้ Oscillator เริ่มต้นการทำงานใหม่และมีเสถียรภาพคงที่ (ปกติจะใช้เวลาไม่น้อยกว่า 10 มิลลิวินาที)

2.18 MCS-51 แบบมี EPROM ในตัว

ตัว MCS-51 แบบมี EPROM มีจำหน่ายหลายแบบดังตาราง เช่นเบอร์ 8751H จะมี EPROM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

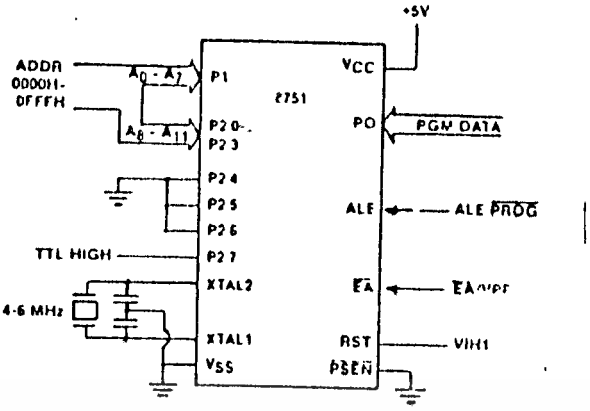
เพิ่มขึ้นมาขนาด 4K * 8 จัดในตระกูล MCS-51 นั่นคือการโปรแกรมสามารถที่จะกระทำด้วยการโปรแกรมทางแรงดันขนาดหนึ่งและลบด้วยแสงอุลตร้าไวโอเลต นอกเหนือจากนั้นตัว 8751 ไม่ให้มีการอ่านข้อมูลโปรแกรมบนชิปได้ด้วย ซึ่งจะเป็นการป้องกันการลอกเลียนแบบโปรแกรมที่ได้ออกแบบสร้างไว้

Device Name	EPROM Version	EPROM Byte	Ckt Type	V_{pp}	Time Required To Program Entry Arrays
8051	(8751)	4 K	HMOD	21.00 V	4 minultes
8051AH	8751H	4 K	HMOD	21.00 V	4 minultes
80C51BH	87C51	4 K	CMOD	12.75 V	13 seconds
8052AH	8752BH	8 K	HMOD	12.75 V	26 seconds

2.18.1 การโปรแกรมตัว EPROM

การโปรแกรมเบอร์ 8751H จะต้องทำงานด้วย Oscillator ขนาด 4-6 MHz เหตุผลที่ต้องมี Oscillator ขณะโปรแกรม ก็เพื่อที่จะใช้ในการเข้าถึงข้อมูล Address และข้อมูลโปรแกรมที่ผ่านตัว Register ภายในก่อนที่จะไปเก็บที่บริเวณ EPROM การให้ Address ในตำแหน่ง EPROM ที่ถูกโปรแกรมจะส่งสัญญาณ Address ที่ Port 1 และขาของ Port 2 ที่ P2.0 ถึง P2.3 ขณะที่ข้อมูลถูกส่งเข้าที่ Port 0 ขาของ Port 2 ที่ P2.4 ถึง P2.6 และ $\overline{PSEN}$ จะต้องรักษาระดับต่ำและขา P2.7 และขา RST จะต้องมีระดับสูง ขา $\overline{EA}/V_{pp}$ ปกติจะมีค่าระดับสูง TTL โดยมีระดับพัลส์เข้า +21 V. ขณะที่ EA/V_{pp} มีค่า 21 V. ขา ALE/$\overline{PROG}$ ซึ่งปกติจะต้องรักษาระดับสูงจะมีการป้อนพัลส์ต่ำเป็นเวลา 50 มิลลิวินาที หลังจากนั้น $\overline{EA}/V_{pp}$ จะลดระดับพัลส์ 21 V. ลงสู่ค่าระดับสูง TTL (มีค่าสูงสุด 2.5 V. ถือเป็นระดับสูง TTL) การป้อนสัญญาณต่างๆในการโปรแกรม 8751 มีแสดงในรูปที่ 2.31 สิ่งที่ต้องระวังก็คือ ขา $\overline{EA}/V_{pp}$ ขณะโปรแกรมป้อนพัลส์เข้าระดับแรงดันจะต้องไม่เกิน 21.5 V. ในช่วงเวลาใดๆและแม้กระทั่งการเกิดพัลส์ลักษณะกลิทช์ (Glitch) ที่มีค่าระดับแรงดันนี้เพราะอาจทำลายตัว 8751 ให้เสียหายได้ ดังนั้น V_{pp} ที่ป้อนเข้ามาจะต้องรักษาระดับให้ดี และจะต้องมีการป้องกันการเกิดกลิทช์ (Glitch) ที่เข้ามาด้วย

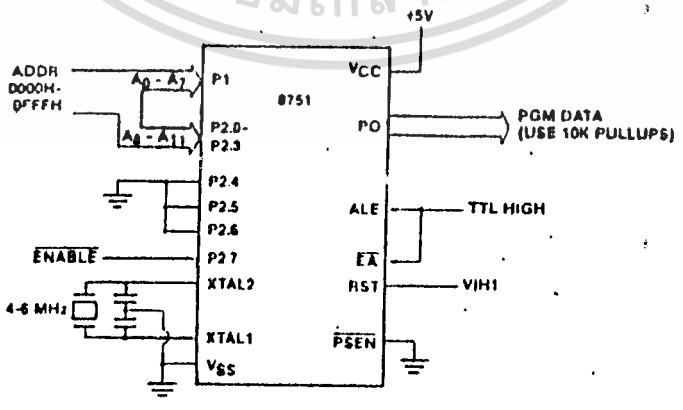
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.31 การต่อวงจรแบบอัตโปรแกรม

2.18.2 การตรวจสอบโปรแกรม

ถ้าไม่มีการใช้โปรแกรมเป็นแบบรักษาความปลอดภัย โปรแกรมหน่วยความจำบนชิปสามารถที่จะอ่านมาเพื่อใช้ในการตรวจสอบโปรแกรม ทั้งในระหว่างหรือหลังการทำงานตามโปรแกรมที่ออกแบบไว้ การต่อขาต่างๆ เพื่อใช้งานลักษณะนี้ มีแสดงในรูปที่ 2.32 ซึ่งแสดงลักษณะการต่อขาเช่นเดียวกับการโปรแกรมเข้า EPROM ยกเว้นขา P2.7 จะรักษาระดับ TTL ต่ำ หรือใช้เป็นตัวถูกอ่านในลักษณะสถิตต่ำ Address ของตำแหน่งหน่วยความจำโปรแกรมจะอ่านเข้าที่ Port 1 และขา P2.0 ถึง P2.3 ส่วนขาอื่นของ Port 2 และขา $\overline{PSEN}$ จะรักษาที่ระดับต่ำหมด ส่วนขา ALE,EA และ RST จะรักษาที่ระดับสูง ดังนั้นถ้าข้อมูลตามตำแหน่ง Address ใน EPROM จะออกมาที่ Port 0 การพูลอัพโดยใช้ตัวต้านทาน 10 Kohm จากภายนอกของ Port 0 จะต้องใช้งานในลักษณะนี้

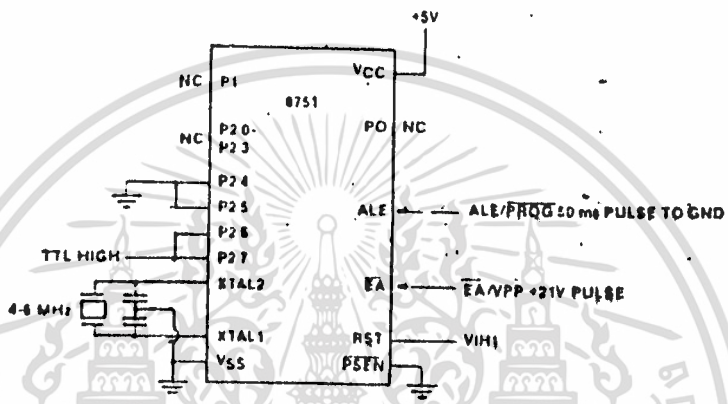


รูปที่ 2.32 การต่อวงจรแบบการตรวจสอบโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.18.3 การโปรแกรมหน่วยความจำแบบรักษาความปลอดภัย

ใน 8751H จะมีบิตสำหรับรักษาความปลอดภัย ซึ่งถูกโปรแกรมครั้งหนึ่งจะป้องกันไม่ให้โปรแกรมหน่วยความจำอ่านออกไปที่ภายนอกได้ การต่อขาและวิธีทำสำหรับการโปรแกรมบิตรักษาความปลอดภัย จะมีวิธีเช่นเดียวกับการโปรแกรมธรรมดา ยกเว้นขา P2.6 จะให้มีระดับ TTL สูง และการต่อขาต่างๆ ดังแสดงในรูปที่ 2.33 Port 0,Port 1 และขา P2.0 ถึง P2.3 ของ Port 2 อาจจะให้อยู่ที่สถานะใดๆ ส่วนขาควบคุมและ Port ให้ต่อตามสถานะที่ต้องการตามรูปที่ 2.33



รูปที่ 2.33 การต่อวงจรแบบการตรวจสอบโปรแกรมแบบรักษาความปลอดภัย

การโปรแกรมใส่บิตรักษาความปลอดภัย สามารถที่จะทำต่อเมื่อได้รับการลบในส่วนโปรแกรมหน่วยความจำโปรแกรมหมดแล้ว ขณะที่ทำการโปรแกรม โปรแกรมหน่วยความจำภายในไม่สามารถจะอ่านออกมาได้ และไม่สามารถที่จะโปรแกรมซ้ำลงไปได้ และมันไม่สามารถจัดการทำงานกับโปรแกรมหน่วยความจำภายนอกได้ด้วย ดังนั้นการลบ EPROM จะเป็นการทำการลบโปรแกรมบิตรักษาความปลอดภัยให้ทำงานตามฟังก์ชั่นของโปรแกรมใหม่ได้อย่างสมบูรณ์

การโปรแกรมแบบรักษาความปลอดภัยจะเป็นระบบการล็อกโปรแกรมแบบระดับที่ประกอบด้วย 2 บิตล็อคและ 32-ไบต์ Encryption Array ที่ป้องกันการ Copy โปรแกรมที่ผู้ใช้สามารถจะสร้างรหัสเองได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

การจัดการหน่วยความจำ รูปแบบการกำหนดเลขที่อยู่ และตัวประมวลผลทางบูลีนของ MCS-51

3.1 บทนำ

สถาปัตยกรรมของของ MCS-51 ได้แบ่งหน่วยความจำมาให้บนชิปพร้อมทั้งสามารถที่จะขยายหน่วยความจำภายนอกได้ โดยทั่วไป การกำหนดเลขที่อยู่ของ เบอร์นี้ จะมีคำสั่ง เพื่อช่วยให้ทำงานได้รวดเร็ว และสามารถใช้คำสั่งได้อย่าง เต็มประสิทธิภาพก่อนที่จะศึกษาถึงชุดคำสั่งอย่างละเอียด ก็ควรจะทำความเข้าใจถึงการแบ่งหน่วยความจำและรูปแบบการกำหนดเลขที่อยู่และการประมวลผลทางบูลีน

3.2 การจัดการหน่วยความจำ

MCS-51 แบ่งตามพื้นฐานหน่วยความจำของการกำหนดเลขที่อยู่ Address ได้เป็น 3 ส่วนที่ประกอบด้วยเนื้อที่

64 กิโลไบต์หน่วยความจำโปรแกรม

64 กิโลไบต์หน่วยความจำข้อมูลภายนอก

256 ไบต์เป็นหน่วยความจำข้อมูลภายใน ส่วนเบอร์ 8032/8052 มีขนาด 384 ไบต์

3.2.1 เนื้อที่หน่วยความจำโปรแกรม

หน่วยความจำโปรแกรมจะประกอบด้วย ส่วนภายในและภายนอกชิป ถ้าขา EA มีสถานะสูง MCS-51 จะบริการโปรแกรมภายใน ถ้าโปรแกรมมีความยาวไม่เกิน OFFFH (หรือ 2000H-OFFFH สำหรับ 8052) จะเป็นการเฟทช์ข้อมูลโปรแกรมภายนอกทั้งหมด ในทุกกรณีตัวนับโปรแกรมขนาด 16 บิต จะเป็นตัวกำหนดเลขที่อยู่โปรแกรม

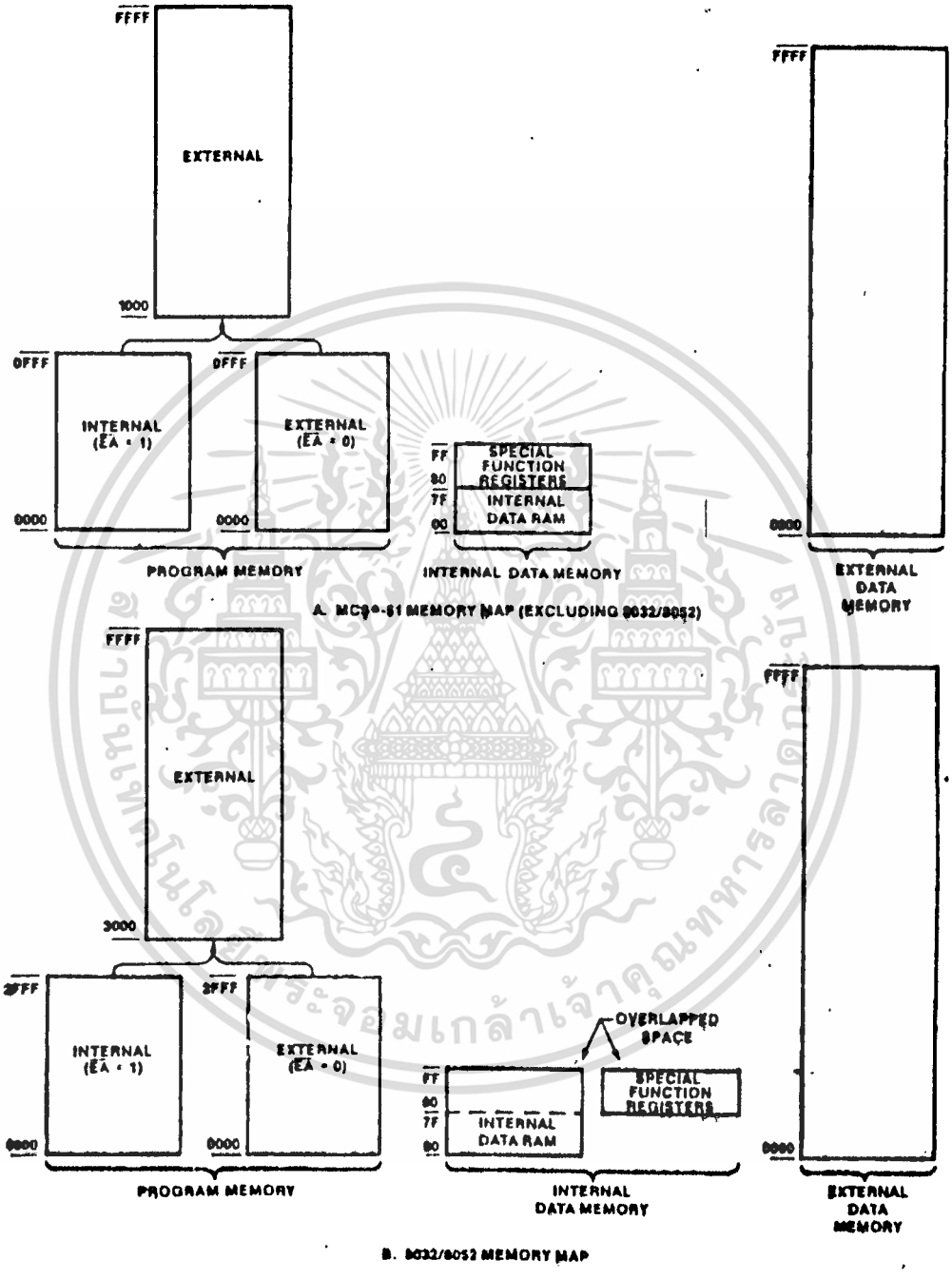
ตำแหน่ง 00 ถึง 32H (หรือ 00 ถึง 2BH สำหรับเบอร์ 8032/8052)ในหน่วยความจำโปรแกรมจะสำรองไว้ สำหรับใช้บริการการอินเทอร์รัพท์ ตามรูปที่ 3.1

3.2.2 เนื้อที่หน่วยความจำข้อมูล

หน่วยความจำข้อมูลจะประกอบด้วยความจำข้อมูลภายในและภายนอก หน่วยความจำข้อมูลภายนอก จะเข้าถึงได้ด้วยการใช้คำสั่ง MOVX

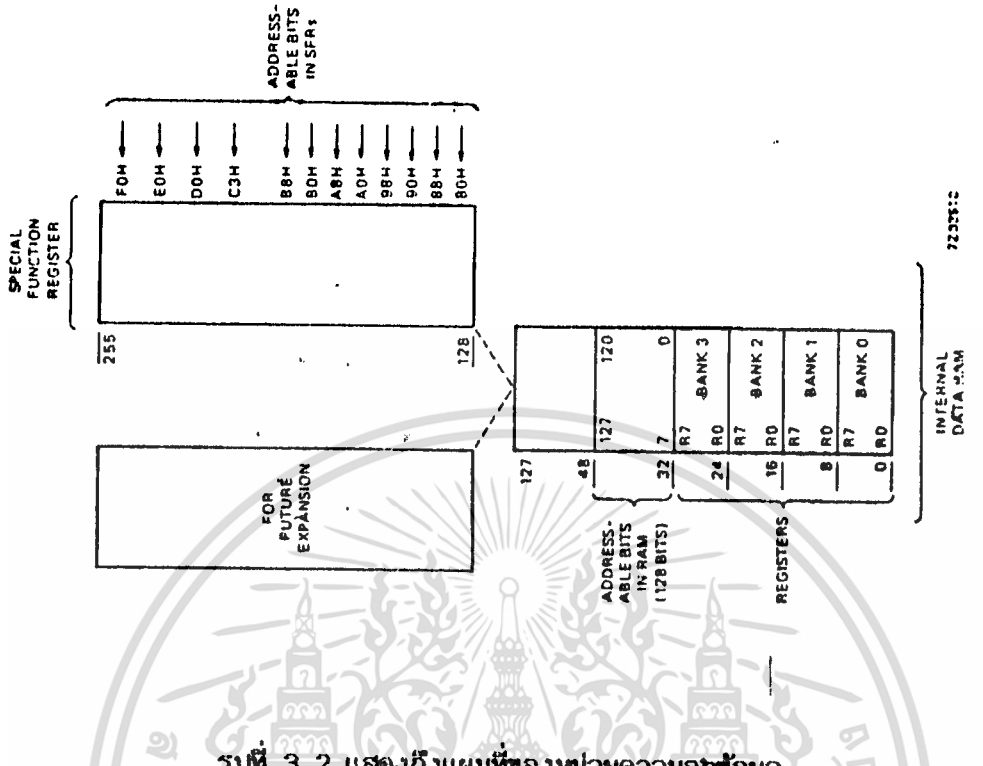
หน่วยความจำข้อมูลภายในจะแบ่ง เป็นลักษณะงาน ดังนี้คือจำนวน 128 ไบต์เป็นของบริเวณตำแหน่งบนของ RAM ภายใน ส่วนบนนี้จะมีเฉพาะในเบอร์ 8032/8052 เท่านั้นและส่วนของ 128 ไบตอีกบริเวณหนึ่ง ใช้เป็น Register ฟังก์ชันพิเศษ ขณะที่ใช้ส่วนบนของ RAM ภายในและบริเวณของ SFR ทั้งสองส่วนนี้จะถูกบันส่วนให้ใช้ค่า Address ภายในแต่จะเข้าถึงในแต่ละบริเวณของทั้งสองบริเวณนี้ได้ ด้วยการใช้รูปแบบการกำหนดเลขที่อยู่ที่ต่างกัน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

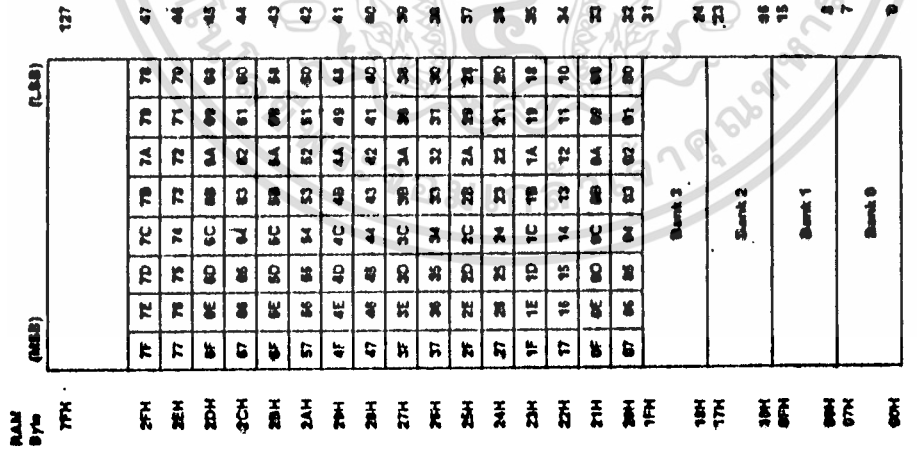


รูปที่ 3.1 แสดงแผนภูมิพื้นที่ของหน่วยความจำ MCS-51

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



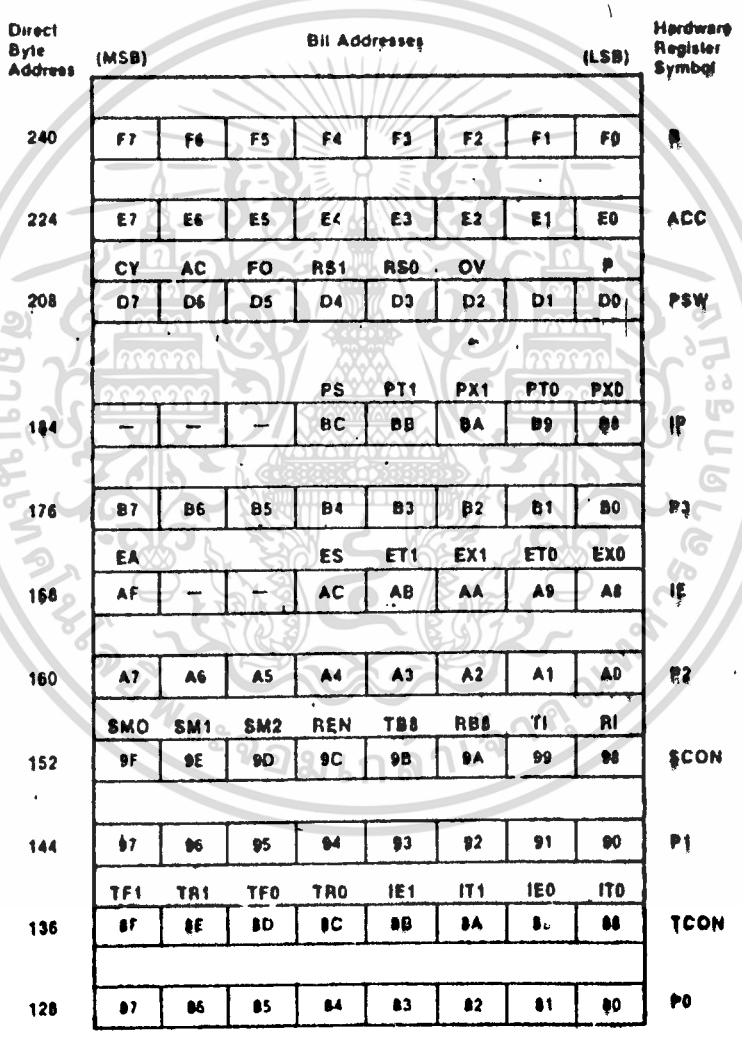
รูปที่ 3.2 แสดงถึงแผนที่ของหน่วยความจำข้อมูล



รูปที่ 3.3 แสดงถึงพื้นที่การกำหนดค่าแหน่งบิต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 3.2 แสดงถึงแผนที่ของหน่วยความจำข้อมูล โดยแบ่งเรียงเป็น 4 แบงค์ในแต่ละแบงค์มี Register 8 ตัว มีตำแหน่งตั้งแต่ 0 ถึง 31 ในบริเวณล่างของ RAM แบงค์เหล่านี้จะถูกเรียกใช้ให้อีนาเบิ้ลได้คราวละหนึ่งแบงค์เท่านั้น ด้วยการกำหนดเริ่มแรกภายในสองบิตของ Register PSW ว่าจะเลือกใช้ในแบงค์ใดภายใน 4 แบงค์และบริเวณตำแหน่งตั้งแต่ 20H ถึง 2FH จำนวน 16 ตำแหน่งวาละ 1 ไบต์ สามารถที่จะกำหนดเลขที่อยู่ของแต่ละบิตได้ ดังแสดงในรูปที่ 2.3 เป็นบิต RAM Address เนื้อที่ Register SFR สามารถที่จะกำหนดตำแหน่งได้เช่นกัน ดังรูปที่ 2.4



รูปที่ 3.4 ตำแหน่งของ Register SFR และบิต Address ของ SFR

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.3 โหมดการกำหนดเลขที่อยู่ข้อมูล

คำสั่งภาษาแอสเซมบลี้ MCS-51 จะประกอบด้วยนิวมอนิกรหัสการทำงานหนึ่งไบต์และตัวโอเปอร์แรนด์อีก ตั้งแต่ 0-3 ไบต์ โดยการใช้ค่าสัญญลักษณ์ต่างๆ ที่ถูกแบ่งด้วย (,) คำสั่งที่ใช้โอเปอร์แรนด์สองสัญญลักษณ์จะให้ตัวแรกเป็นตัวรับการถ่ายเทข้อมูล (Destination) ส่วนตัวหลังจะเป็นตัวส่งข้อมูล (Source) ส่วนที่ใช้คำสั่งต่างๆ ส่วนใหญ่นั้น จะทำงานโดยใช้ Accumulator เป็นตัวหลักในการเป็นตัวส่งข้อมูลในโอเปอร์แรนด์ และจะเป็นตัวรับผลลัพธ์ที่เกิดขึ้นหลังการทำงานคำสั่งนั้นๆ ด้วยการให้ตัวอักษร 'A' เป็นตัวกำหนดตัวส่งข้อมูลหรือรับการถ่ายเทข้อมูลในโอเปอร์แรนด์ฟิลด์ ตัวอย่างเช่น

ADD A,(ตัวส่งข้อมูล)

คำสั่งนี้จะเป็นการบวกเอาข้อมูลที่อยู่ในตัวส่งข้อมูล กับข้อมูลใน Accumulator และรจะเก็บกลับมาที่ Accumulator

การใช้ลักษณะ (ตัวส่งข้อมูล) ในคำสั่งตัวอย่างนี้ สามารถที่จะเลือกใช้ได้ตามลักษณะการกำหนดเลขที่อยู่ได้ 4 โหมดด้วยกันคือ

1. การกำหนดเลขที่อยู่ Register ด้วยการใช้ข้อมูลใน Register ทำงานในแบงค์ที่ถูกเลือกในขณะนั้น

2. การกำหนดเลขที่อยู่โดยตรง ด้วยการใช้เลขที่อยู่ตามตำแหน่งของ RAM ภายใน Port I/O หรือกลุ่ม Register

3. การกำหนดเลขที่อยู่ Register โดยอ้อม ด้วยการใช้ค่าใน Register ทำงานเป็นตัวชี้ตำแหน่ง ค่าข้อมูลของ RAM ภายในชิป

4. การกำหนดเลขที่อยู่ข้อมูลโดยทันที ด้วยการใช้ข้อมูลขนาด 1 ไบต์คงที่ภายในคำสั่ง

โหมดสามโหมดแรกเป็นการเข้าถึง RAM ภายใน และวงจร Register ต่างๆภายในตามโครงสร้างตามสถาปัตยกรรมของ MCS-51 นอกจากจะใช้เป็นตัวส่งข้อมูลในโอเปอร์แรนด์ฟิลด์แล้วยังใช้เป็นตัวรับการถ่ายเทข้อมูลในโอเปอร์แรนด์ฟิลด์ได้อีกด้วยส่วนโหมดที่ 4 เนื่องจากโอเปอร์แรนด์ฟิลด์ข้อมูลคงที่ จึงเป็นตัวส่งข้อมูลคงที่นี้อย่างเดียวเท่านั้น

การกำหนดเลขที่อยู่แบบเรจิสเตอร์

การกำหนดเลขที่อยู่ Register จะให้ข้อมูลเข้าถึงโดยการใช้ Register กลุ่มที่ถูกเลือกจากการกำหนดใช้งานในกลุ่ม Register แบงค์ โดยแต่ละแบงค์จะมี 8 Register และใช้บิตต่ำสุดสามบิตแรกของรหัสคำสั่ง เป็นตัวกำหนดเลือกการใช้ Register ใช้งานตัวใดตัวหนึ่งในแบตตัวทั้งสามบิตนี้จะรวมอยู่ในรหัสออปโคดด้วยกับโอเปอร์แรนด์ ตำแหน่งรวมอยู่ในรูปแบบตำแหน่งสั้นๆ เพียงหนึ่งไบต์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในภาษาแอสเซมบลี้ของ MCS-51 จะชี้การทำงานโหมดกำหนดเลขที่อยู่โดย Registerด้วยสัญญลักษณ์ Rn โดย n จะแทนค่า 0-7 หรือสามารถที่แทนสัญญลักษณ์ Register เหล่านี้ได้ด้วยชื่อที่ถูกกำหนดในคำสั่งเทียม อันได้แก่การใช้ EQU หรือ SET directives

<u>การกำหนดเลขที่อยู่โดยตรง</u>

การกำหนดเลขที่อยู่โดยตรงเป็นวิธีเดียวที่จะเข้าถึงข้อมูลทาง Hardware Register เช่น กลุ่มของ SFR และสามารถกำหนดเลขที่อยู่โดยตรงบริเวณตำแหน่งต่างๆของ RAM ภายในจำนวน 128 ไบต์ ด้วยการใช้ไบต์โอเปอร์แรนด์ตัวต่อตัวจากออปโค้ดของคำสั่ง เป็นตัวกำหนดตำแหน่งที่ถูกใช้

กลุ่มใดกลุ่มหนึ่งในสองกลุ่มของหน่วยความจำภายในขนาด 128 ไบต์จะถูกเลือกใช้ ขึ้นอยู่กับบิตอันดับสูงสุดของไบต์การกำหนดเลขที่อยู่โดยตรง เมื่อเลขที่อยู่โดยตรงอยู่ที่ 0 ถึง 127 (00H - 7FH) ก็จะเป็นการเลือกใช้กลุ่ม 128 ไบต์ ในตำแหน่งอันดับต่ำของ RAM ข้อมูลภายใน ตัว 8032/8052 ได้สร้างให้มีกลุ่มของ RAM เพิ่มเป็น 256 ไบต์ในการใช้คำสั่ง ยังคงใช้การกำหนดเลขที่อยู่โดยตรงได้ภายใน 128 ไบต์ (อันดับต่ำ) เท่านั้น ส่วนที่สูงกว่านั้นจะต้องใช้คำสั่งในโหมดอื่น เช่นด้วย วิธีโดยอ้อมหรือใช้ตัวชี้สแต็กในการเข้าถึงข้อมูลอีก 128 ไบต์ของอันดับสูงใน RAM ภายในชิป

กลุ่ม Register ของ SFR และพอร์ตไอโอทั้งหมด Register ที่ควบคุมและแสดงถานะจะถูกกำหนดไว้ที่ตำแหน่งระหว่าง 128 ถึง 255 (80H - FFH) เมื่อไบต์ที่กำหนดเลขที่อยู่โดยตรงถูกใช้ด้วยค่าระหว่าง 128 ถึง 255 จะเป็นการเข้าถึงข้อมูลในกลุ่ม Register ต่าง ๆ ทาง Hardware รูปที่ 3.4 เป็นการแสดงถึงตำแหน่งของกลุ่ม Register ต่าง ๆ ทาง Hardware ตัวอย่างเช่น การเลือกพอร์ต 0 Register จะให้เลขที่อยู่โดยตรงเป็น 80H และใช้พอร์ต 1 จะให้เลขที่อยู่โดยตรงเป็น 90H เป็นต้น แทนที่จะต้องจำตัวเลข เหล่านี้โดยมากจะใช้สัญญลักษณ์เขียนแทนในภาษาแอสเซมบลี้

การใช้เลขที่อยู่โดยตรง จะเป็นวิธีเดียวเท่านั้นที่ช่วยให้เข้าถึง Register ในกลุ่ม SFR ทั้งหมดด้วยการกำหนดคำสั่งในไบต์โอเปอร์แรนด์และกลุ่มไอโอ ซึ่งต่างกับ MCS-48 ที่ไม่จำเป็นต้องมีกลุ่มคำสั่งแบบการกำหนดเลขที่อยู่โดยตรง เพราะมีเพียง 5 Register ที่เป็นพิเศษ คือ BUS,P1,P2,PSW และ T ไม่เหมือนกับของ MCS-51 ที่มีถึง 16 ชุดคำสั่งของออปโค้ดที่ใช้ควบคุมในการอ่านหรือเขียนใน Register แต่ละตัวกับตัว Accumulator

<u>การกำหนดเลขที่อยู่เรจิสเตอร์โดยอ้อม</u>

เราจะสามารถเก็บค่าตัวแปรต่าง ๆ ที่กำหนดตำแหน่งของ RAM ได้อย่างไร ในขณะที่มีการหา คำนวณข้อมูลเปลี่ยนแปลงอยู่ตลอดเวลาในขณะที่โปรแกรมยังทำงานอยู่ ในการทำงานลักษณะนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เช่น การตรวจสอบการลำดับตำแหน่งหน่วยความจำ การใช้คัชนีเป็นตัวชี้ตารางข้อมูลใน RAM หรือการคูณกันของตัวเลขที่มีจุดทศนิยม หรือการเรียงลำดับตัวอักษร งานเหล่านี้จะไม่สามารถที่จะใช้การกำหนดเลขที่อยู่โดยตรงได้ เนื่องจากการใช้ค่าของเลขที่อยู่ในตัวรีจิสเตอร์แรนด์ของโหมดนี้จะคงที่ในการทำแอสเซมบลี้

ดังนั้น การกำหนดเลขที่อยู่โดยอ้อมใน MCS-51 ด้วยข้อมูลใน R0 หรือ R1 โดยแบงค์ที่ถูกเลือกทำงานเป็นคัชนี หรือ ตัวชี้ตำแหน่งข้อมูลภายใน 256 ไบต์ แบ่งเป็นล็อกต่ำ จำนวน 128 ไบต์ของ RAM ภายในและบล็อกสูงอีก 128 ไบต์ของRAMภายใน ซึ่งมีเฉพาะในเบอร์ 8032/8052 เท่านั้น หรือใช้เป็นตัวชี้จำนวน 256 ไบต์ของหน่วยความจำข้อมูลภายนอก การกำหนดใช้ RAM ภายในจะสามารถกำหนดภายในด้วย R0 และ R1 ซึ่งจะเป็นตัวเลือกด้วยรหัสคำสั่งออปโค้ดที่บิตหลักต่ำสุด (LSB)

ใน MCS-51 ภาษาแอสเซมบลี้ที่กำหนดการใช้สัญลักษณ์ @ นำหน้า R0,R1 โดยที่ R0 และ R1 นี้สามารถที่จะถูกแทนด้วยสัญลักษณ์หรือชื่อใด ๆ ได้

การเข้าถึงให้ได้ถึง 64 กิโลไบต์ จากหน่วยความจำข้อมูลภายนอกที่ว่าง จะ Address ได้ด้วย Register คู่ของ Data Pointer (DPTR) ขนาด 16 บิต

การสั่งงานด้วยคำสั่ง PUSH และ POP สามารถใช้การกำหนดที่อยู่โดยอ้อมได้ โดยที่ค่าในตัวชี้สแตกจะชี้ที่ค่า Address ภายใน RAM บนพื้นที่ของ Stack

ส่วนการกำหนดใช้ตัว Special Function Register ไม่สามารถที่จะถูก Address ได้ด้วยวิธีนี้ ต้องใช้โหมดการกำหนดเลขที่อยู่โดยตรง

<u>การกำหนดเลขที่อยู่โดยทันที</u>

เมื่อแหล่งกำเนิดตัวรีจิสเตอร์แรนด์เป็นค่าคงที่มากกว่าที่จะเป็นตัวแปร ดังนั้นตัวคงที่ที่ใช้ในขณะนั้นจะเป็นส่วนหนึ่งของรหัสคำสั่งรวมอยู่ในรูปแบบคำสั่ง

ค่าที่ถูกใช้เป็นค่าคงที่ที่โปรแกรมอยู่ใน ROM หรือ EPROM และจะไม่สามารถเปลี่ยนแปลงค่าเหล่านี้ได้ในระหว่างการทำงานตามโปรแกรมในภาษาแอสเซมบลี้ การเรียกใช้ในโหมดการกำหนดเลขที่อยู่นี้จะมีรูปแบบเขียนด้วยสัญลักษณ์ '#' ภายในรีจิสเตอร์แรนด์ฟิลด์ด้วยค่าที่ท้ายสัญลักษณ์นี้สามารถเขียนเป็นตัวเลขหรือใช้สัญลักษณ์เชื่อมตัวแปร หรือใช้สูตรการคำนวณทางคณิตศาสตร์ที่ให้ค่าออกมาคงที่จำนวนหนึ่ง

จากที่ได้กล่าวมาแล้วทั้ง 4 โหมด จะเป็นประเภทคำสั่งแบบใช้รีจิสเตอร์แรนด์ 2 ตัว ที่มี Accumulator เป็นหลัก 1 ตัว ซึ่งการกำหนดใช้ Accumulator เช่น MOV,ADD จะใช้รหัสในไบต์แรกเป็นตัวบ่งบอก การทำงานของ ADDC,SUBB,ANL,ORL และ XRL สามารถที่จะแทนคำสั่ง ADD ในตัวอย่างที่กล่าวมาแล้วได้ ส่วนคำสั่ง XCH จะมีลักษณะงานได้ 3 โหมด ส่วนการทำงาน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ข้อมูลแบบ XCH ที่มีค่าคงที่ของรีจิสเตอร์แรนด์ ก็จะใช้วิธีเลี่ยงโดยใช้โหมดแบบการกำหนดเลขที่อยู่ข้อมูลโดยทันที ตัวรีจิสเตอร์แรนด์ตัวหนึ่งในคำสั่ง INC และ DEC DJNZ และ CJNZ ทั้งหมดอาจจะทำงานอยู่บนตัว Accumulator หรือจะกำหนดตัว Register ทำงานตัวใดตัวหนึ่งโดยตรงก็ได้ และกำหนดจากการใช้โหมดการกำหนดเลขที่อยู่ Register โดยอ้อม ยกเว้นตัว MCS-48 คำสั่ง DJNZ จะไม่สามารถใช้ตัว Accumulator หรือใช้โหมดกำหนดเลขที่อยู่โดยอ้อมในคำสั่งนี้ อย่างไรก็ตาม การกำหนดใช้ Accmulator สามารถกำหนดเลขที่อยู่ของแอคคิวมิวเลเตอร์นี้ได้เช่นเดียวกับการเรียกใช้ Register อื่น เช่น กลุ่มของ SFR ด้วยการกำหนดเลขที่อยู่ของ Accmulator เช่น PUSH,POP ฯลฯ ด้วยการกำหนดสัญญลักษณ์เป็น ACC แทน A ในกรณีเรียกใช้ Accmulatro เป็นการกำหนดเลขที่อยู่โดยตรง

<u>ข้อดีของการใช้สัญญลักษณ์ในการกำหนดเลขที่อยู่</u>

ASM51 ก็เหมือนกับโปรแกรมภาษาแอสเซมบลี้ หรือภาษาสูงทั่วไป ที่สามารถที่จะกำหนดตัวแปรต่าง ๆ และผู้ใช้สามารถที่จะกำหนดชื่อสัญญลักษณ์ต่างๆ แทนตัว Register ต่าง ๆได้โดยการใช้รูปแบบทั่วไปของภาษาทั่วไปของภาษาแอสเซมบลี้ คือ

LABEL : Mnemonic (OP-CODE) Operand ; Comment

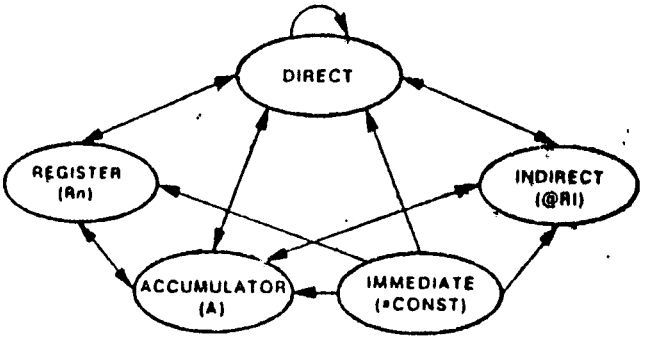
โดยที่ในฟิลด์เลเบลจะสิ้นสุดด้วย : การใช้อักษรเป็นสัญญลักษณ์ซึ่งอาจจะรวมเอาตัวเลขหรือเครื่องหมายคำถาม (?) หรือขีดเส้นใต้ (_) สำหรับการใช้ชื่อที่มีอักษรยาวมาก ๆ ในฟิลด์เลเบลนี้ แอสเซมบลี้จะกำหนดใช้เพียง 10 ตัวอักษรแรกเท่านั้น

โปรแกรมภาษาแอสเซมบลี้อาจเขียนได้ด้วยตัวเล็กและตัวใหญ่ แต่ตัวโปรแกรมจะกำหนดให้เป็นรหัสตัวใหญ่หมด เช่นกำหนด A สำหรับ a LOOP]1 สำหรับ loop]1 การใช้ ']' ช่วยให้ผู้อ่านอ่านเข้าใจถึงความหมายของสัญญลักษณ์นั้น ได้ง่าย เพราะเป็นการแยกคำของสัญญลักษณ์ออกจากกัน

ASM51 ยังช่วยให้ใช้ตัวแปรต่าง ๆ ที่จะแทนค่าใน Register หรือตำแหน่ง RAM ทั้งภายนอกภายในได้ง่าย ด้วยการใช้คำสั่งเทียม คือ EQU หรือ SET ตัวอย่างเช่น การใช้สัญญลักษณ์เลขที่อยู่เป็นตัวแปรกำหนดตำแหน่งใน RAM

จากตารางชุดคำสั่ง MCS-51 ที่สรุปไว้จะมีคำสั่งนิวมอนิก nemonics เพียงเล็กน้อยที่โปรแกรมเมอร์ โปรแกรมจะจดจำได้ง่ายรูปแบบข้อมูลและการกำหนดเลขที่อยู่ที่แตกต่างกันสามารถที่จะถูกกำหนดได้ด้วยการกำหนดรูปแบบในฟิลด์รีจิสเตอร์แรนด์มากกว่าที่เปลี่ยนแปลงบนฟิลด์นิวมอนิก ตัวอย่างเช่นคำสั่ง 'MOV' จะทำงานที่แตกต่างกันได้ถึง 18 แบบขึ้นอยู่กับการเรียกใช้รูปแบบข้อมูล 3 แบบ คือแบบบิต แบบไบต์และการกำหนดเลขที่อยู่ 15 รูปแบบในการย้ายตัวแปรขนาดไบต์ระหว่าง Address ที่ว่างเขียนแบบย่อ ๆ ได้ดังแผนภูมิรูปที่ 3.5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.5 รูปแบบการย้ายตัวแปรระหว่าง Address ที่ต่าง

โดยที่แต่ละลูกศรที่ชี้ เป็นการแสดงถึงทิศทางข้อมูล จะเคลื่อนจากตัวส่งข้อมูลไปยังตัวรับข้อมูล ด้วยการกำหนดเลขที่อยู่ Address ชนิดต่าง ๆ ได้ถึง 15 รูปแบบ จะสังเกตุเห็นว่าคำสั่งส่วนใหญ่ที่ใช้ Rgister เป็นตัวกำหนดเลขที่อยู่สามารถจะใช้คำสั่งของการกำหนดเลขที่อยู่ได้โดยตรงเช่นกันหรือจะใช้กลับกันก็ได้ด้วยลักษณะงานเช่นนี้ทำให้โปรแกรมเมอร์สามารถเลือกใช้วิธีการต่างๆในการเขียนโปรแกรมได้ง่ายขึ้น เช่นโปรแกรมเมอร์อาจจะเลือกการใช้โหมดการกำหนดเลขที่โดยตรงอย่างเดียวในการกำหนดให้ Address ภายในเป็นตัว Register ชั่วคราวจำนวนมากๆแทนที่จะกำหนดด้วยวิธีการกำหนดเลขที่อยู่ Register เป็นการใช้ Register แต่ละแบงค์ซึ่งจะทำให้โปรแกรมเมอร์จำเป็นจะต้องจดจำในส่วนของโปรแกรม ขณะที่ทำงานอยู่ในแบงค์ใดอยู่ เพื่อไม่ให้สับสน ซึ่งการกำหนดเลขที่อยู่โดยตรงก็สามารถกำหนดค่าเป็นตัวสัญลักษณ์แทนตำแหน่งของ Address แต่ละตัวได้โดยตรง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

องค์ประกอบของระบบ Servo

4.1 หลักการทำงานของดีซีมอเตอร์

ดีซีมอเตอร์เป็นทรานสดิวเซอร์แรงบิดซึ่งมีการออกแบบ ให้มีคุณลักษณะพิเศษคือแรงบิดของเพลาของดีซีมอเตอร์จะเป็นสัดส่วนโดยตรงกับกระแสอาร์มาเจอร์ แรงบิดของเพลาของดีซีมอเตอร์จะได้จากผลระหว่างสนามแม่เหล็กและขดลวดตัวนำ หลักการนี้แสดงได้ในรูปที่ 4.1 ในที่นี้กระแสที่ไหลในขดลวดตัวนำจะสร้างฟิลด์ที่ประกอบด้วยเส้นแรงแม่เหล็ก (ϕ) และขดลวดตัวนำเหล่านั้นอยู่ห่างจากจุดศูนย์กลางการหมุนเท่ากับ (r) ความสัมพันธ์ระหว่างแรงบิดของเพลาและกระแสเท่ากับ

$$T = K\phi I$$

เมื่อ T คือแรงบิดของเพลามีหน่วยเป็น นิวตัน-เมตร

ϕ คือเส้นแรงแม่เหล็กมีหน่วยเป็น เวเบอร์

I คือกระแสมีหน่วยเป็น แอมแปร์

K คือค่าตัวคงที่

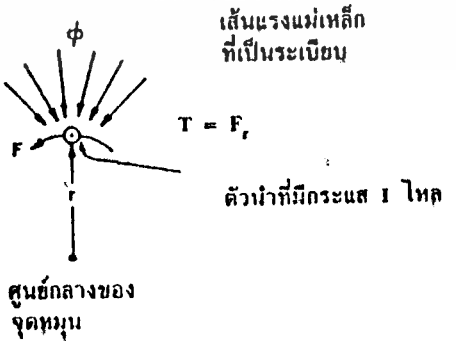
ดังนั้นแรงบิดของเพลาจะเป็นสัดส่วนโดยตรงกับผลคูณของเส้นแรงแม่เหล็กและกระแส เมื่อขดลวดตัวนำเคลื่อนที่ในสนามแม่เหล็กก็จะทำให้เกิดโวลท์เตจตกคร่อมตัวมันเอง โวลท์เตจนี้จะเป็นสัดส่วนกับความเร็วของเพลาของมอเตอร์ และด้านการไหลของกระแส ความสัมพันธ์ระหว่างโวลท์เตจย้อนกลับนี้และความเร็วของมอเตอร์คือ

$$E = K\phi w$$

เมื่อ E คือโวลท์เตจย้อนกลับ emf มีหน่วยเป็น โวลท์

ϕ คือเส้นแรงแม่เหล็กมีหน่วยเป็น เวเบอร์

w คือความเร็วของมอเตอร์มีหน่วยเป็น เรเดียน/วินาที



รูปที่ 4.1 แสดงถึงการเกิดแรงบิดในตัวดีซีมอเตอร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 การแยกประเภทของดีซีมอเตอร์

ดีซีมอเตอร์สามารถแบ่งออกได้เป็นหลายประเภท ขึ้นอยู่กับลักษณะวิธีการสร้างสนามแม่เหล็กของตัวมอเตอร์และขึ้นอยู่กับพื้นฐานการออกแบบโครงสร้างของอาร์มาเจอร์ การแบ่งประเภทตามลักษณะการจ่ายสนามแม่เหล็กแยกออกได้เป็น 2 แบบคือ

1. ดีซีมอเตอร์แบบปรับเส้นแรงแม่เหล็กได้
2. ดีซีมอเตอร์แบบเส้นแรงแม่เหล็กมีค่าคงที่

ถ้าเราจะพิจารณาแยกประเภทตามลักษณะการออกแบบโครงสร้างอาร์มาเจอร์ สามารถแยกออกได้เป็น 3 แบบคือ

1. ดีซีมอเตอร์แบบอาร์มาเจอร์เป็นแกนเหล็ก
2. ดีซีมอเตอร์แบบอาร์มาเจอร์ที่มีขดลวดพันอยู่บนพื้นผิว
3. ดีซีมอเตอร์แบบอาร์มาเจอร์เป็นขดลวดหมุน

นอกจากนี้ยังมีดีซีมอเตอร์ชนิดพิเศษอีกแบบหนึ่งคือ แบบไม่มีแปรงถ่าน (Brushless DC Motor) ซึ่งมีหลักการทางเทคโนโลยีเหมือนกับดีซีมอเตอร์ที่มีแปรงถ่าน ยกเว้นการคอมมิวเทชั่นกระทำโดยเทคนิคทางอิเลคทรอนิคส์แทนที่จะกระทำโดยวิธีการทางเชิงกล

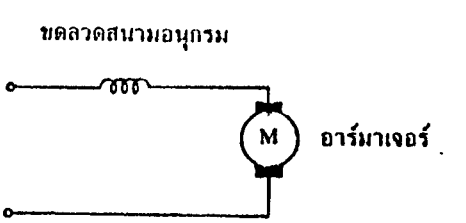
ดีซีมอเตอร์แบบปรับเส้นแรงแม่เหล็กได้

ดีซีมอเตอร์แบบปรับเส้นแรงแม่เหล็กได้ยังแบ่งแยกได้เป็น 2 แบบคือ

1. แบบขดลวดสนามแม่เหล็กต่ออนุกรมกับขดลวดอาร์มาเจอร์
2. แบบขดลวดสนามแม่เหล็กแยกกระตุ้น

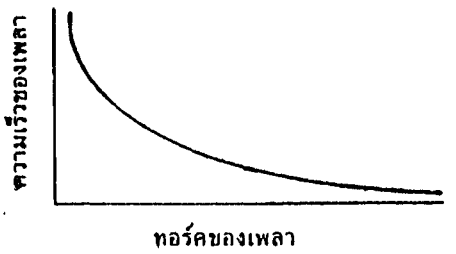
ตัวอย่างของมอเตอร์แบบขดลวดสนามแม่เหล็กต่ออนุกรมแสดงไว้ในรูปที่ 4.2 มอเตอร์แบบนี้จะมีเส้นแรงแม่เหล็กเป็นสัดส่วนกับกระแส ดังนั้นเส้นแรงของแม่เหล็กจึงสามารถปรับค่าได้และเราจะได้ความสัมพันธ์ระหว่างความเร็วและแรงบิดเป็นนอนลิเนียร์ดังแสดงในรูปที่ 4.3

มอเตอร์ต่อดังกล่าวจะใช้งานในสภาวะเฉพาะเมื่อต้องการแรงบิดสูงความเร็วต่ำ และแรงบิดต่ำที่ความเร็วสูง เช่นระบบการทับเคลื่อนของรถลาก



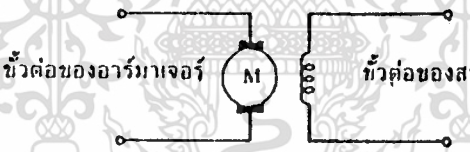
รูปที่ 4.2 ดีซีมอเตอร์แบบอาร์มาเจอร์ต่ออนุกรมกับขดลวดสนามแม่เหล็ก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

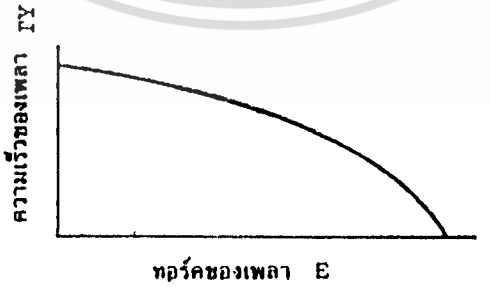


รูปที่ 4.3 คุณสมบัติระหว่างความเร็วและแรงบิดของดีซีมอเตอร์อนุกรมภายใต้ภาวะโรเตอร์เจคงที่

ตัวอย่างของมอเตอร์แบบขดลวดสนามแม่เหล็กแยกกระตุ้นแสดงในรูปที่ 4.4 ดีซีมอเตอร์แบบนี้มักนิยมเรียกกันว่ามอเตอร์ขนาน (Shunt Motor) มอเตอร์แบบนี้สามารถปรับเส้นแรงแม่เหล็กได้อย่างอิสระต่อกระแสของอาร์มาเจอร์ยังผลให้สามารถควบคุมพารามิเตอร์ของมอเตอร์ให้มีค่าคงที่ได้ตลอดช่วงพิสัยที่กว้าง มอเตอร์นี้จะทำงานในกรณีระบบบังคับการเคลื่อนที่ที่ต้องการแรงบิดสูง ในรูปที่ 4.5 แสดงถึงคุณสมบัติระหว่างแรงบิดกับความเร็วของ Shunt Motor ภายใต้ภาวะการกระตุ้นสนามแม่เหล็กคงที่และอาร์มาเจอร์โรเตอร์เจคงที่



รูปที่ 4.4 ดีซีมอเตอร์แกแยกสำหรับสนามแม่เหล็กได้



รูปที่ 4.5 แสดงคุณสมบัติระหว่างความเร็วและแรงบิดของ Shunt Motor ภายใต้ภาวะอาร์มาเจอร์โรเตอร์เจคงที่และการกระตุ้นสนามแม่เหล็กคงที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

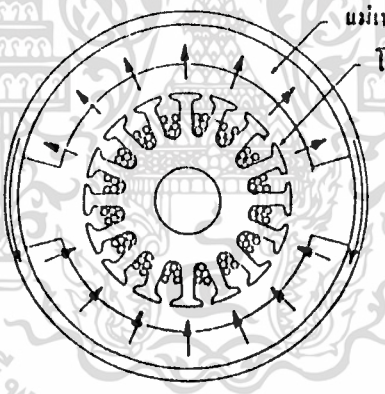
ดีซีมอเตอร์แบบเส้นแรงแม่เหล็กคงที่

ระบบการกระตุ้นฟิลด์ของมอเตอร์โดยทั่วไปในปัจจุบันมักใช้เป็นแบบแม่เหล็กถาวร ในระบบนี้เส้นแรงของฟิลด์มีค่าคงที่ ดังนั้นอัตราส่วนระหว่างกระแสอาร์มาเจอร์และและแรงบิดจะมีค่าคงที่

อย่างไรก็ตาม ข้อดีของมอเตอร์แบบฟิลด์แม่เหล็กถาวรซึ่งเหนือกว่ามอเตอร์แบบมีโครงสร้างของฟิลด์ด้วยการพันของขดลวดก็คือ ไม่มีกำลังสูญเสียในฟิลด์ มีประสิทธิภาพสูงกว่าและมีขนาดเล็กกว่าเมื่อเทียบกับมอเตอร์ที่มีขนาดกำลังม้าเท่ากัน นอกจากนั้นความสัมพันธ์เชิงเส้นยังให้ค่าของกระแสอาร์มาเจอร์ที่สูงกว่าดีซีมอเตอร์แบบฟิลด์เป็นขดลวด การประยุกต์ใช้งานเหมาะกับระบบที่ต้องการแรงบิดของโหลดสูง

ดีซีมอเตอร์แบบอาร์มาเจอร์เป็นแกนเหล็ก

โครงสร้างของโรเตอร์และสเตเตอร์ของมอเตอร์แบบแกนเหล็กโลหะดังในรูปที่ 4.6 โครงสร้างของมอเตอร์แบบนี้มีโมเมนท์ของแรงเฉื่อยสูงที่สุด และมีค่าอินดัคแต้นซ์ของโรเตอร์สูงที่สุดด้วย ดังนั้นมอเตอร์นี้จึงมีปริมาณการจุดความร้อนได้สูง และสามารถจะทนโอเวอร์โหลดได้ในระยะเวลาที่ยาวนานโดยที่ไม่ทำให้มอเตอร์เสียหาย

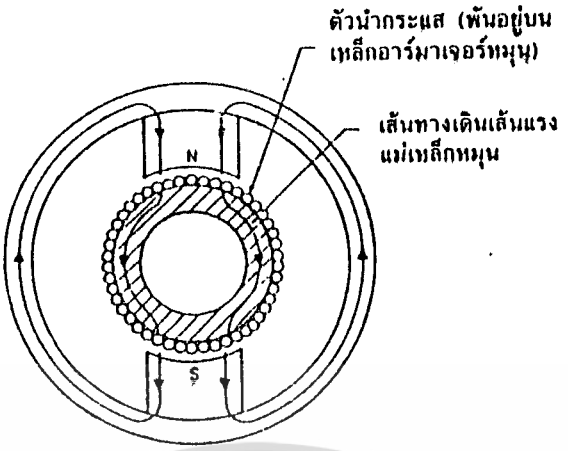


รูปที่ 4.6 แสดงรูปหน้าตัดของดีซีมอเตอร์แบบอาร์มาเจอร์เป็นแกนเหล็กส่วนฟิลด์เป็นแม่เหล็ก

ดีซีมอเตอร์แบบอาร์มาเจอร์มีขดลวดพันอยู่บนพื้นผิว

ในรูปที่ 4.7 แสดงถึงการออกแบบของโรเตอร์ที่มีขดลวดพันอยู่บนพื้นผิว โดยไม่มีสล็อททำให้ได้อินดัคแต้นซ์ของโรเตอร์ต่ำกว่าแบบแกนเหล็ก ข้อเสียคือทำให้ขนาดของมอเตอร์แบบนี้ใหญ่ขึ้นและมีราคาแพงกว่าแบบแกนเหล็กด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



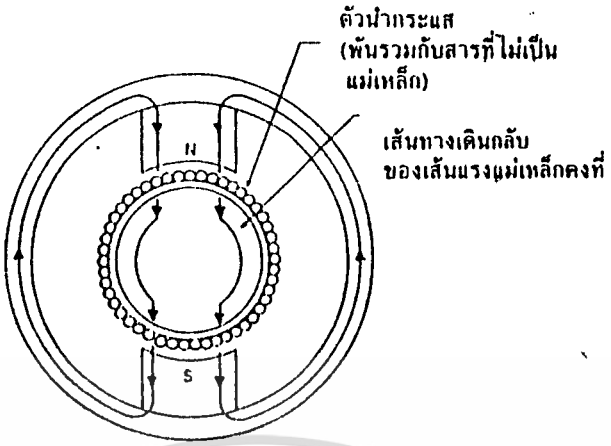
รูปที่ 4.7 แสดงรูปหน้าตัดของดีซีมอเตอร์แบบมีขดลวดบนพื้นผิวและฟิลด์เป็นแม่เหล็กถาวร

ดีซีมอเตอร์แบบอาร์มาเจอร์เป็นขดลวดหมุน

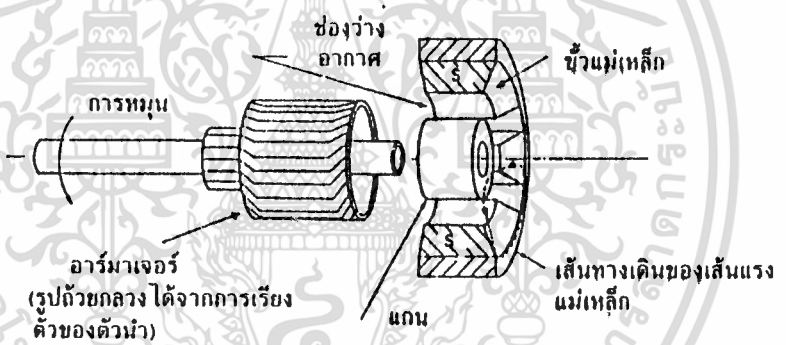
มอเตอร์แบบขดลวดหมุนนี้ได้รับการออกแบบเพื่อให้มีโมเม้นท์ของแรงเฉื่อยน้อยมาก ดังแสดงในรูปที่ 4.8 และรูปหน้าตัดซึ่งมองด้านข้างของมอเตอร์ดังกล่าวแสดงในรูปที่ 4.9 อาร์มาเจอร์แบบเป็นขดลวดเคลื่อนที่และลักษณะโครงสร้างภายนอก และโครงสร้างของแม่เหล็กมอเตอร์แบบนี้มีช่องว่างอากาศ (Air Gap) ระหว่างแม่เหล็กมากกว่ามอเตอร์ทั้งสองแบบที่กล่าวมาแล้ว ดังนั้นจำเป็นต้องออกแบบให้โครงสร้างของแม่เหล็กให้ใหญ่ขึ้น เพื่อให้ได้ช่องว่างของอากาศระหว่างเส้นแรงแม่เหล็กที่เท่ากันกับของมอเตอร์ทั้งสองแบบดังกล่าว ดังนั้นราคาของมอเตอร์ชนิดนี้จึงมีราคาแพง นอกจากนั้นโครงสร้างของโรเตอร์มีความจุความร้อนต่ำมาก ถ้าหากเกิดโอเวอร์โหลดก็จะทำให้มอเตอร์เสียได้ง่ายและโรเตอร์ลักษณะนี้จะมีค่าอินดัคแต้นซ์ต่ำมากคือน้อยกว่า 10 ไมโครเฮนรี่

ในรูปที่ 4.10 แสดงถึงมอเตอร์แบบขดลวดเคลื่อนที่อีกลักษณะหนึ่ง ซึ่งมีโครงสร้างของอาร์มาเจอร์เป็นรูปร่างจาน ซึ่งทำขึ้นจากขดลวดตัวนำซ้อนกันหลายๆชั้น ซึ่งเรามักเรียกว่า "Printed Motor" ในปัจจุบันดีซีมอเตอร์แบบโรเตอร์เป็นขดลวดหมุนนี้ให้คุณสมบัติการทำงานที่ดีเยี่ยมเหมาะสำหรับเป็นตัวขับเคลื่อนในระบบการบังคับตำแหน่ง และยังให้อัตราส่วนระหว่างแรงบิดและแรงเฉื่อยได้สูงและมีค่าอินดัคแต้นซ์ต่ำที่สุดเมื่อเทียบกับมอเตอร์แบบอื่นๆ นอกจากนั้นความสามารถในการเพิ่มอัตราเร่งยังกระทำได้สูง 10^6 เรเดียน/วินาที ^{2}

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.8 หน้าตัดของดีซีมอเตอร์แบบมีร์เคอร์เป็นขดลวดเคลื่อนที่และฟิลด์เป็นแม่เหล็กถาวร

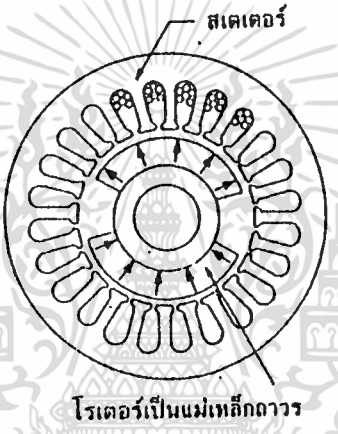


รูปที่ 4.9 แสดงถึงรูปหน้าตัดซึ่งมองด้านข้างของดีซีมอเตอร์แบบมีร์เคอร์เป็นขดลวดหมุน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ดีซีมอเตอร์แบบไม่มีแปรงถ่าน

ดีซีมอเตอร์แบบไม่มีแปรงถ่านต่างจากดีซีมอเตอร์ที่กล่าวมาแล้วคือ การคอมมิวเทชั่น กระแสอาร์มาเจอร์จะใช้วิธีทางอิเลคทรอนิคส์ โครงสร้างของมอเตอร์แบบนี้จะมีโรเตอร์เป็นแม่เหล็กหยัด "Back Iron" และขดลวดคอมมิวเตจจะอยู่ภายนอกส่วนของตัวโรเตอร์ดังแสดงในรูปที่ 4.10 (เปรียบเทียบกับดีซีมอเตอร์ทั่วไปแล้วจะมีโครงสร้างที่มีส่วนประกอบกลับกัน) การประยุกต์ใช้งานของดีซีมอเตอร์แบบไม่มีแปรงถ่านนี้มักจะใช้ในระบบที่ต้องการโมเมนท์ของแรงเฉื่อยต่ำ เนื่องด้วยโครงสร้างของโรเตอร์แบบนี้สร้างขึ้นด้วยสารแม่เหล็กชนิดพิเศษเพื่อให้เป็นมอเตอร์ชนิดที่มีโมเมนท์ของแรงเฉื่อยต่ำ



รูปที่ 4.10 รูปหน้าตัดของดีซีมอเตอร์แบบไม่มีแปรงถ่านและโรเตอร์เป็นแม่เหล็กถาวร

การจัดคอมมิวเทชั่นในมอเตอร์แบบไม่มีแปรงถ่านจะกระทำได้ภายนอกตัวมอเตอร์ โดยใช้วงจรสวิทช์ซึ่งซึ่งรวมเข้ากับมอเตอร์เพื่อให้เข้าใจการบังคับมอเตอร์แบบไม่มีแปรงถ่านดียิ่งขึ้น เอนโคดเดอร์ที่เพลาของมอเตอร์แสดงให้เห็นว่าจำเป็นต้องซิงโครไนซ์คอมมิวเทชั่นสวิทช์ซึ่งกับตำแหน่งมุมของโรเตอร์

4.3 เอซีมอเตอร์แบบสองเฟส (เอซีเซอร์โวมอเตอร์)

เอซีมอเตอร์แบบสองเฟสมักจะใช้กันทั่วไปในระบบคอนโทรลที่ต้องการกำลังต่ำๆและสามารถปรับความเร็วได้ ข้อได้เปรียบเบื้องต้นของเอซีมอเตอร์ที่เหนือกว่าดีซีมอเตอร์คือสามารถใช้ร่วมกับซิงโครซ์ (Synchros) ได้ดีโดยไม่ต้องมีการดีโมดูเลชั่นสัญญาณ Error

มอเตอร์ประกอบด้วยอินดัคชั่นโรเตอร์และขดลวดฟิลด์ 2 ขด วางอยู่ในตำแหน่งทำมุมกัน 90 องศา ฟิลด์ขดหนึ่งทำหน้าที่เป็นฟิลด์อ้างอิงที่มีค่าคงที่ ส่วนอีกฟิลด์หนึ่งเป็นฟิลด์คอนโทรล สัญญาณ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Error AC ที่ได้รับการขยายแล้วจะป้อนให้กับฟิลด์คอนโทรล สัญญาณนี้จะมีขนาดของแมกนิจูดที่แปรค่าได้และเฟสก็จะแปรค่าอยู่ที่ 0 อาศา หรือ 180 องศา ฟิลด์อ้างอิงได้รับการป้อนกลับด้วยโวลท์เตจที่มีค่าคงที่ผ่านวงจรเลื่อนเฟสไป 90 องศา สัญญาณนี้จะมีขนาดของแมกนิจูดคงที่และมุมของเฟสเป็น -90 องศา เอซีโวลท์เตจทั้งสองกำหนดได้เป็น

$$e_c = V_c \cos wt$$

$$e_R = A \cos(wt - 90^o) = A \sin wt$$

เมื่อ e_c = โวลท์เตจของฟิลด์คอนโทรล

e_R = โวลท์เตจของฟิลด์อ้างอิง

w = ความถี่โวลท์เตจที่ใช้งาน

V_c = แอมปลิจูดของคอนโทรลโวลท์เตจที่สามารถแปรค่าได้

A = แอมปลิจูดของโวลท์เตจอ้างอิงที่มีค่าคงที่

แมกนิจูดและเครื่องหมายของ V_c จะแปรไปตามแมกนิจูดและเครื่องหมายของสัญญาณError สัญญาณ Error เป็นลบก็จะยังผลให้ V_c มีค่าเป็นลบหมายถึง e_c จะมีเฟสต่างไปจาก e_R 180 องศา ทิศทางการหมุนของมอเตอร์ขึ้นอยู่กับความสัมพันธ์ระหว่างเฟสของโวลท์เตจ e_c และ e_R

โครงสร้างโรเตอร์ของเอซีมอเตอร์แบบนี้มีอยู่ด้วยกัน 3 ลักษณะคือ

- แบบทรงกระรอก (Squirrel Cage)
- แบบทรงตัน (Solid)
- แบบรูปถ้วย (Drag-cup)

โรเตอร์แบบทรงกระรอกเป็นแบบที่นิยมใช้กันมากที่สุดเพราะมีคุณสมบัติการทำงานได้ดีและให้ผลตอบสนองได้เร็ว นอกจากนั้นขดลวดของโรเตอร์ของมอเตอร์แบบนี้จะได้รับการออกแบบให้มีความต้านทานสูงที่สุดเพื่อให้คุณลักษณะของแรงบิด-ความเร็วเป็นลิเนียร์

ค่าไนโคลนสแตนท์ของมอเตอร์ มีความสำคัญต่อคุณลักษณะการทำงานไดนามิคของระบบการบังคับมอเตอร์เป็นอย่างมาก แมกนิจูดของไนโคลนสแตนท์นี้อยู่ในเทอมของแรงเฉื่อยซึ่งขึ้นอยู่กับขนาดโรเตอร์ของเซอร์โวมอเตอร์นี้ และขึ้นอยู่กับแรงเฉื่อยของโหลดที่สะท้อนกลับมายังเพลาของมอเตอร์ ส่วนประกอบดังกล่าว (ขนาดของโรเตอร์และแรงเฉื่อยของโหลด) ต้องพยายามให้มีค่าลดลงเท่าที่จะทำได้ ตัวอย่างคือโรเตอร์ควรใช้เป็นแบบรูปถ้วยเพื่อให้มีแรงเฉื่อยต่ำหรือใช้หมุนโหลดที่มีขนาดเล็ก

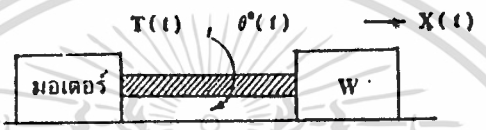
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4 ความสัมพันธ์ระหว่างการเคลื่อนที่แบบเป็นเชิงเส้นและแบบเป็นเชิงมุม

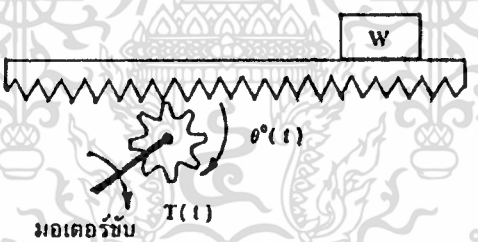
ปัญหาเกี่ยวกับการบังคับการเคลื่อนที่ซึ่งประสบกันบ่อยๆ คือการแปลงการเคลื่อนที่แบบเชิงมุมให้เป็นการเคลื่อนที่แบบเชิงเส้น ตัวอย่างการแก้ปัญหาดังกล่าวแสดงได้ในรูปที่ 4.11,4.12 และ 4.13 ซึ่งเป็นระบบง่ายๆที่มีแรงเฉื่อยสมมูลย์ต่ออยู่โดยตรงกับตัวมอเตอร์

สำหรับตัวอย่างในรูปที่ 4.13 มวล (W) เป็นเสมือนกับมวลที่จุดหนึ่งซึ่งเคลื่อนที่ได้ด้วยลูกรอกซึ่งมีรัศมีเท่ากับ r ถ้าเราไม่คิดแรงเฉื่อยของลูกรอกมอเตอร์ที่แรงเฉื่อยสมมูลย์เท่ากับ

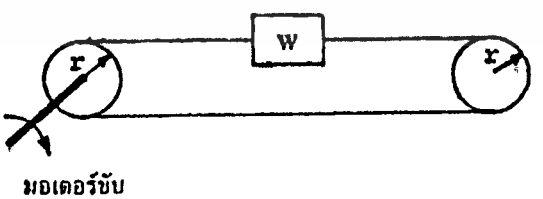
$$J = Mr^2 = (W/g)r^2$$



รูปที่ 4.11 ระบบมอเตอร์และสปริง



รูปที่ 4.12 ระบบที่เฟืองและเฟืองจักร



รูปที่ 4.13 ระบบสายพานและลูกรอก

ถ้าเฟืองตัวเล็กในรูปที่ 4.12 มีรัศมีเท่ากับ r มอเตอร์จะมีแรงเฉื่อยสมมูลย์เหมือนกับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สมการข้างบน พิจารณาถึงระบบในรูปที่ 4.11 ลิดสกรูมีความยาวเส้นตรงเท่ากับและมีมวลเคลื่อนที่ต่อรอบของสกรู ดังนั้นหลักการของระบบทั้งสองในรูปที่ 4.11 และ 4.12 เหมือนกันในรูปที่ 4.12 มวล (W) จะถูกเคลื่อนที่ต่อรอบของเฟืองเล็กเท่ากับ $2\pi r$ ดังนั้นเราใช้สมการข้างบนสำหรับหาแรงเฉื่อยของระบบในรูปที่ 4.13 ได้

$$J = (W/g)(L/2\pi)^2$$

J = แรงเฉื่อย (ออนซ์-นิ้ว-วินาที^{2})

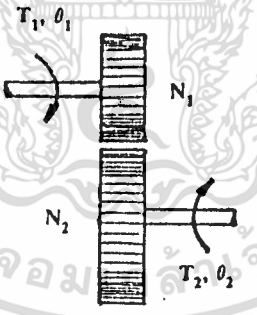
W = น้ำหนัก (ออนซ์)

L = ลีดสกรู (นิ้ว)

G = แรงดึงดูดของโลก (386.4 นิ้ว/วินาที)

4.5 ระบบเกียร์

ระบบเกียร์ คานงัดหรือสายพานในลูกรอก เป็นอุปกรณ์เครื่องกลซึ่งสามารถส่งพลังงานจากส่วนหนึ่งของระบบไปยังส่วนอื่นๆได้ในรูปของแรงงาน แรงบิด ความเร็วและการเคลื่อนที่ นอกจากนั้นอุปกรณ์เหล่านี้ยังเป็นเสมืออุปกรณ์สำหรับประสาน (Matching) ที่สามารถใช้ส่งผ่านกำลังให้ได้ค่าสูงสุด รูปที่ 4.14 แสดงถึงการคัพปลิ้งเกียร์ 2 ตัวเข้าด้วยกัน แรงเฉื่อยและแรงเสียดทานของเกียร์จะไม่นำมาคิดในเมื่อพิจารณาถึงเกียร์ในอุดมคติ



รูปที่ 4.14 ระบบการคัพปลิ้งของเกียร์

ความสัมพันธ์ระหว่างแรงบิด T_1 และ T_2 การเคลื่อนที่เชิงมุม θ_1 และ θ_2 และจำนวนซี่ฟัน N_1 และ N_2 ของระบบเกียร์สามารถหาได้จากหลักเกณฑ์ต่อไปนี้

(1) จำนวนซี่ฟันของเกียร์จะเป็นสัดส่วนกับรัศมี r_1 และ r_2 ของเกียร์ นั่นคือ

$$r_1N_2 = r_2N_1 \qquad \text{----------(1)}$$

(2) ระยะทางการเคลื่อนที่ไปของเกียร์แต่ละตัวจะมีค่าเท่ากัน ดังนั้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$\theta_1 r_1 = \theta_2 r_2 \qquad \text{----------(2)}$$

(3) แรงงานที่ได้จาก เกียร์ตัวหนึ่งจะ เท่ากับแรงงานที่ได้จาก เกียร์อีกตัวหนึ่ง เนื่องจากสมมติให้ว่าไม่มีการสูญเสียแรงงาน ดังนั้น

$$T_1\theta_1 = T_2\theta_2 \qquad \text{----------(3)}$$

ถ้าความเร็วเชิงมุมของ เกียร์ทั้งสองคือ w_1 และ w_2 ในรูปที่ 4.18 สมการที่(1) (2)และ (3) เขียนใหม่ได้เป็น

$$(T_1/T_2)=(\theta_2/\theta_1)=(N_1/N_2)=(w_2/w_1)=(r_1/r_2) \qquad \text{----------(4)}$$

ในทางปฏิบัติตัว เกียร์จริงๆจะต้องมีแรง เฉื่อยและแรง เสียดทาน เกิดขึ้น เนื่องจากการคัพปลิ้งระหว่างซี่ฟันของ เกียร์ซึ่งไม่สามารถตัดทิ้งได้ ระบบสมมูลย์ของ เกียร์ที่มีวิสคอสฟริคชั่นคูว์ลอมบ์ฟริคชั่นและแรง เฉื่อยพิจารณา เป็นส่วนประกอบของระบบดังแสดงดังรูปที่ 4.19 ตัวแปรและพารามิเตอร์ต่อไปนี้อธิบายถึงระบบเกียร์

T = แรงบิดที่ป้อนให้ระบบเกียร์

θ_1 และ θ_2 = ระยะทางการเคลื่อนที่เป็นแบบเชิงมุม

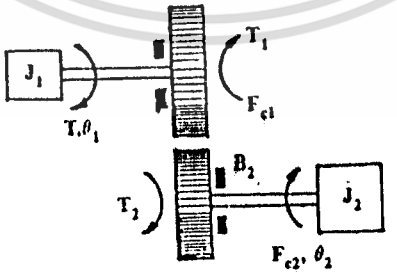
T_1 และ T_2 = แรงบิดที่ส่งผ่านมายัง เกียร์

J_1 และ J_2 = แรง เฉื่อยของ เกียร์

N_1 และ N_2 = จำนวนซี่ฟัน

F_{c1} และ F_{c2} = ตัวสัมประสิทธิ์ของคูว์ลอมบ์ฟริคชั่น

B_1 และ B_2 = ตัวสัมประสิทธิ์ของวิสคอสฟริคชั่น



รูปที่ 4.15 ระบบคัพปลิ้ง เกียร์ที่มีแรง เฉื่อย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สมการแรงบิดของ เกียร์ทั้งสอง เขียนได้เป็น

$$T_2(t) = J_2\{d^2\theta_2(t)/dt^2\}+B_2\{d\theta_2(t)/dt\}+F_{c2}\{\theta_2/\theta_2\} \quad \text{----------(5)}$$

สมการแรงบิดทางข้างของ เกียร์ตังแรงได้เป็น

$$T(t) = J_1\{d^2\theta_1(t)/dt^2\}+B_1\{d\theta_1(t)/dt\}+F_{c1}\{\theta_1/\theta_1\}+T_1(t) \quad \text{--------(6)}$$

จัดยาใช้สมการที่ (4) สมการที่ (5) จะแปลงใหม่ได้เป็น

$$T_1(t) = (N_1/N_2)T_2(t) = (N_1/N_2)^2\ J_2\{d^2\theta_1(t)/dt^2\}+(N_1/N_2)^2\ B_2\{d\theta_1(t)/dt\}+(N_1/N_2)\ F_{c2}\{\theta_2/\theta_2\} \quad \text{----------(7)}$$

สมการที่ (7) แสดงให้เห็นว่าเป็นไปได้ที่จะสะท้อนแรง เฉื่อย แรง เสียดทาน แรงบิด ความเร็วและการ เคลื่อนที่จากข้างหนึ่งของระบบเกียร์ไปยังอีกข้างหนึ่งของระบบเกียร์

ดังนั้น เราจะได้ปริมาณค่าต่างๆต่อไปนี้ เมื่อพารามิเตอร์ต่างๆของ เกียร์ตัวที่สองสะท้อนไปยังเกียร์ตัวที่หนึ่ง

แรง เฉื่อย $:(N_1/N_2)^2J_2$

ตัวสัมประสิทธิ์ของวิสคอฟฟริคชั่น $:(N_1/N_2)^2B_2$

แรงบิด $:(N_1/N_2)T_2$

การ เคลื่อนที่เชิงมุม $:(N_1/N_2)\theta_2$

ความเร็วเชิงมุม $:(N_1/N_2)w_2$

แรงบิดคูว์ลอมโฟริคชั่น $:(N_1/N_2)F_{c2}(w_2/w_2)$

ถ้าผลการบิดของสปริง เกิดขึ้น ค่าคงที่ของสปริงจะได้รับการคูณด้วย $(N_1/N_2)^2$ ในการสะท้อนผลการบิดของแทนค่าสมการ (7) ลงในสมการ (6) จะได้

$$T(t) = J_{1e}\{d^2\theta_1(t)/dt^2\} + B_{1e}\{d\theta_1(t)/dt\} + T_F \quad \text{----------(8)}$$

$$J_{1e} = J_1+(N_1/N_2)^2\ J_2 \quad \text{.----------(9)}$$

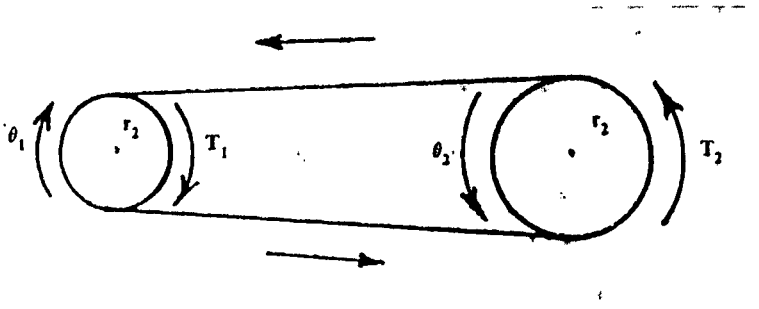
$$B_{1e} = B_1+(N_1/N_2)^2\ B_2 \quad \text{----------(10)}$$

$$T_F = F_{c1}(\tau_1/\tau_1)+(N_1/N_2)F_{c2}(\theta_2/\theta_2) \quad \text{----------(11)}$$

<u>4.6 ระบบสายพานและคานงัด</u>

ระบบการขับด้วยสายพานหรือโซ่ก็มีจุดมุ่งหมาย เดียวกับระบบ เกียร์ นอกจากว่าระบบสายพานหรือโซ่สามารถส่งผ่านพลังงานไปได้ระยะทางไกลกว่าโดยที่ไม่ต้องใช้ เกียร์จำนวนมาก ในรูปที่ 4.16 แสดงถึงไดอะแกรมของระบบการขับด้วยสายพานหรือโซ่ระหว่างลูกรอกทั้งสาม สมมติว่าไม่มีการ เลื่อนไปได้ระหว่างสายพานและลูกรอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.16 ระบบสายพาน

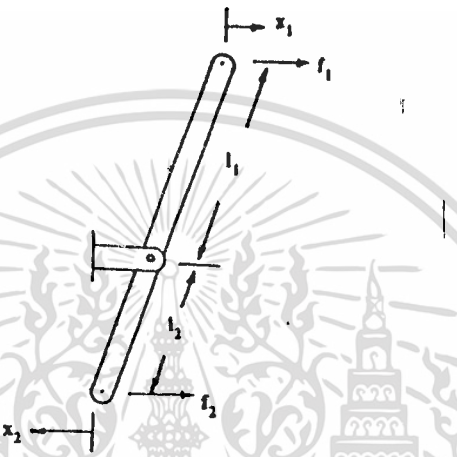
เราก็ยังสามารถใช้สมการที่ (4) กับกรณีนี้ได้ซึ่ง เป็นความจริงที่ว่าการสะท้อนของแรงบิดแรง เฉื่อยและแรง เสียดทาน เป็นต้นก็คล้ายคลึงกับการสะท้อนกลับของระบบเกียร์

ระบบสายพานดังแสดงในรูปที่ 4.17 สามารถใช้ส่งผ่านการ เคลื่อนที่แบบเชิง เส้นและแรงบิดบังคับได้ เช่น เดียวกับการส่งผ่านการ เคลื่อนที่แบบเชิงมุมของระบบเกียร์

ความสัมพันธ์ระหว่างแรงบังคับและระยะทางคือ

$$(f_1/f_2) = r_2/r_1 = x_2/x_1 \qquad \text{----------(12)}$$

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.17 ระบบคานงัด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

SOLID-STATE RELAY AND ANALOG TO DIGITAL

Solid-state Relay

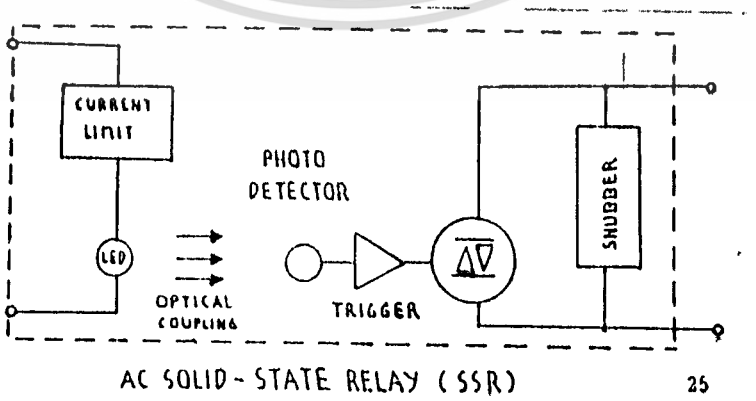
Solid-state Relay หรือเรียกสั้นๆได้ว่า SSR ได้เข้ามาใช้งานในการเป็นอุปกรณ์ติดต่อทางไฟฟ้า โดยเฉพาะแล้วระบบทางไฟฟ้าที่ใช้การทำงานร่วมกับ Microcomputer แล้งจะเข้ามาใช้แทน Relay ธรรมดาๆ โดยใช้หลักการทำงานของอุปกรณ์ที่เป็น Semi-conductor แทนระบบการเคลื่อนไหวทางกลในการตัดต่อวงจร เราอาจจะพิจารณาข้อดีของ SSR ได้ดังนี้

ข้อดีของ SSR

1. การทำงานจะต่อวงจรที่ Voltage ขณะเป็นศูนย์หรือใกล้จุดศูนย์ Volt
2. มีอายุการทำงานที่ยาวนานมาก
3. ไม่มีเสียงรบกวนเกิดขึ้นในเวลาทำงาน
4. สามารถต่อเข้าระบบ Microcomputer ได้โดยง่าย
5. สามารถตัดต่อวงจรได้อย่างรวดเร็ว
6. ไม่มีส่วนเคลื่อนไหวทางกลในการทำงาน
7. ไม่เกิดอาการ Bounce ที่หน้าสัมผัส
8. ไม่ก่อให้เกิดสัญญาณรบกวนต่อระบบไฟฟ้า

ข้อเสียของ SSR

1. ไม่สามารถใช้กับวงจรที่มี Voltage สูงมากๆได้
2. ราคาแพง
3. ไม่สามารถใช้ SSR สำหรับ AC หรือ DC ได้ในตัวเดียวกัน
4. เวลาใช้งานจะเกิดความร้อนขึ้นที่ตัว SSR นั้นๆจำเป็นจะต้องใช้ Heatsink



รูปที่ 5.1 AC Solid-state Relay (SSR)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เราสามารถแบ่งการทำงานของ SSR ได้เป็น 2 ภาคใหญ่ๆ คือ

1. ภาค Input

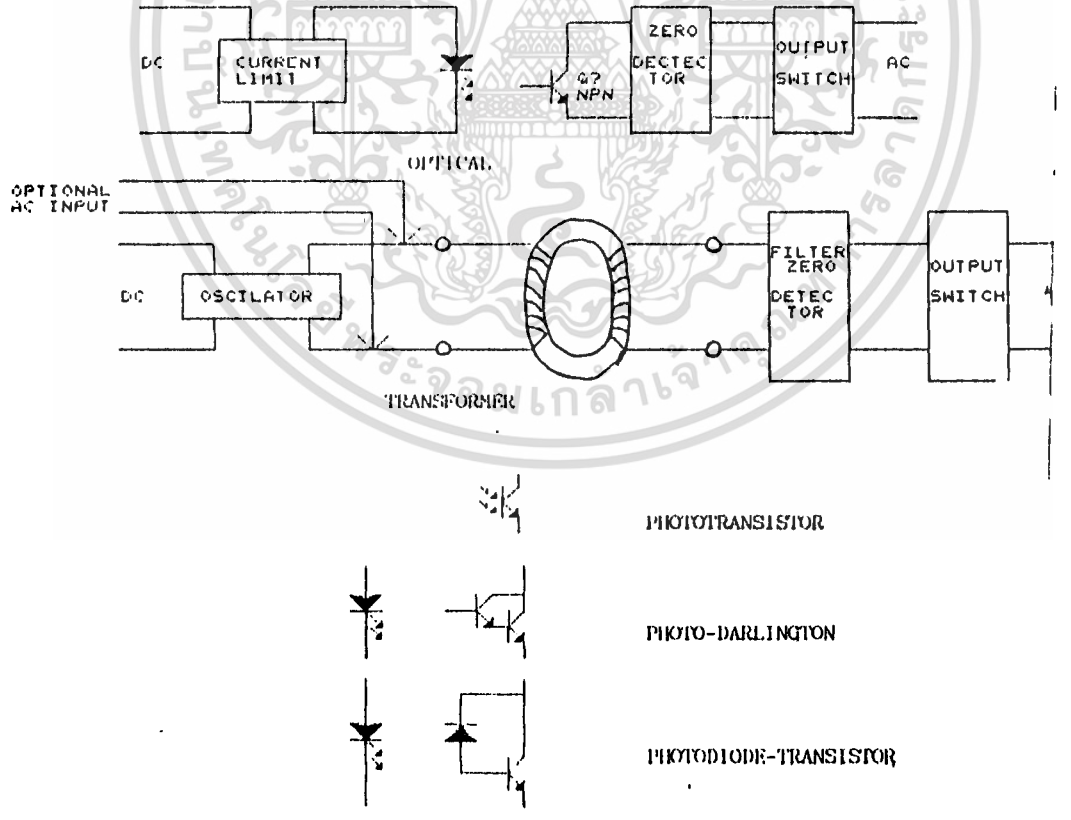
2. ภาค Output

1. ภาค Input

ภาค Input จะเป็นภาครับสัญญาณจากการควบคุมมาทำงาน โดยเป็นการส่งผ่านสัญญาณไปควบคุมอุปกรณ์ติดต่อ ซึ่งจะแยกกันในทางไฟฟ้าโดยที่มีใช้กันอยู่พอแบ่งออกได้เป็น 2 ลักษณะคือ

1.1 ใช้อุปกรณ์ทาง Optical (Photo) เช่น Photo-Transitor, Photo-Darlinton, Photo-Diode, Photo-SCR หรือ Photo-Resistor เป็นต้น ตัวอุปกรณ์จะเปลี่ยนสัญญาณไฟฟ้าที่ส่งมาให้อยู่ในรูปแสงไปควบคุมส่วน Output อีกทีหนึ่งซึ่งจะทำให้เป็นการตัดขาดระหว่าง Input และ Output ทางไฟฟ้าจริงๆ

1.2 ใช้อุปกรณ์ Transformer โดยจะเป็น Ferrite-Core Toroidal Transformer โดยจะทำงานร่วมกันระหว่าง Input และ Oscillator โดยมีการ Oscillator ความถี่สูง 50 KHz ถึง 10 MHz ในการส่งสัญญาณจาก Input ไปยัง Output



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 5.2 แสดงภาค Input แบบต่างๆของ SSR

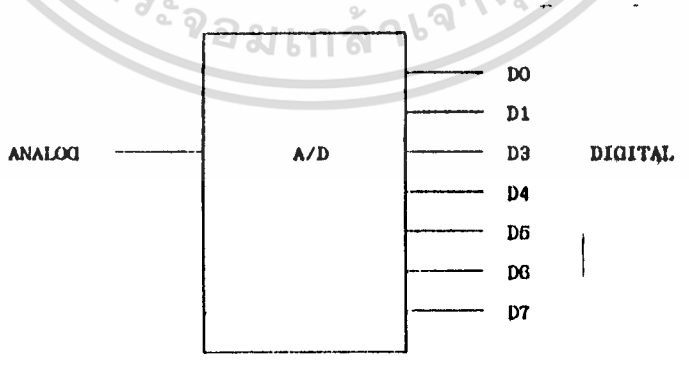
2. ภาค Output

ภาค Output เป็นรับสัญญาณจากภาค Input ซึ่งอาจจะอยู่ในรูปของแสงมาทำงานโดยต่อวงจรหรือตัดอุปกรณ์ โดยอุปกรณ์นี้จะเป็น Transistor,SCR หรือ Triac ก็ได้

Zero Switching

เป็นอีกส่วนหนึ่งที่มีความสำคัญสำหรับการทำงานของ SSR ที่ใช้กับ Output ที่เป็น AC โดยปกติในการทำงานของ SW ธรรมดาต่างๆอาจมีการทำงานที่ไม่สมควรเกิดขึ้นคือ เปิดวงจรในขณะที่สัญญาณของไฟ AC อาจจะอยู่ในค่าสูงสุด ซึ่งจะทำให้เกิดกระแสและสัญญาณรบกวนที่สูงมาก โดยเฉพาะถ้าต่อร่วมกับอุปกรณ์ทาง Computer แล้วก็อาจทำให้เครื่องเกิดอาการเสียหรือทำงานผิดปกติก็ได้ ฉะนั้น Zero Switching ก็คือเป็นการทำงานให้เปิดวงจรในขณะที่สัญญาณ AC อยู่ในระยะ 0 V.แต่ในการใช้งานจริงจะมีค่ามากกว่า 0 V.เล็กน้อยเพื่อให้อุปกรณ์ Semi-Conductor ที่ต่ออยู่สามารถทำงานเปิดวงจรได้

Analog To Digital

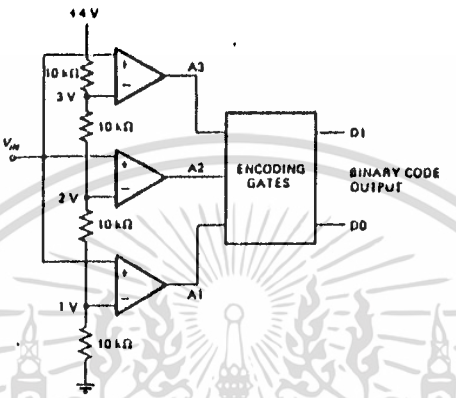


รูปที่ 5.3 Analog To Digital

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คุณสมบัติของ A/D จะมีลักษณะเหมือนกับ D/A และจะมีคุณสมบัติพิเศษอีก 1 อย่างคือ Conversion Time ซึ่งหมายถึงช่วงเวลาที่ A/D ใช้ในการแปลงสัญญาณ Analog เป็นDigital ได้ 1 ค่า เมื่อเราพูดถึง A/D ที่ดีสิ่งนี้หมายถึงการมี Conversion Time แก้ไขน้อยที่สุดเท่าที่จะน้อยได้ วงจร A/D มีหลักการได้หลายแบบโดยจะกล่าวเป็นข้อๆดังนี้

Parallel Comparator A/D



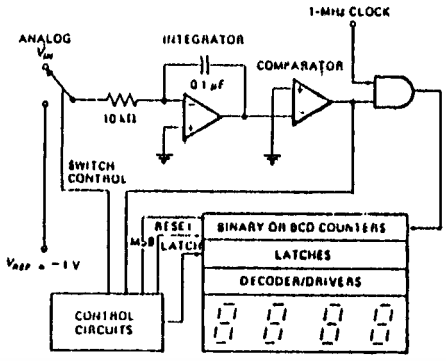
รูปที่ 5.4 Parallel Comparator A/D แบบ 2 บิต

A/D แบบนี้ใช้หลักการของการเปรียบเทียบแบบขนาน โดยสัญญาณที่จะนำมาแปลงนั้นจะต่อขนานกับตัวเปรียบเทียบทุกตัวสัญญาณจะเข้าที่ขาบวกส่วนขาลบของตัวเปรียบเทียบจะต่อกับVoltage อ้างอิงแต่ละระดับโดยได้มาจากการต่อ Divider ของ R จากวงจรนี้ถ้าสัญญาณที่เข้ามามีค่าเท่ากับจุดเปรียบเทียบใดๆ ก็จะทำให้ Output ของตัวเปรียบเทียบนั้นมีค่าเป็น 1 จุดเด่นของ A/D แบบนี้ก็คือสามารถทำการ Convert ได้เร็วมากซึ่งมีค่าทั่วไปอยู่ที่ 20 nS. แต่ข้อเสียก็คือจะสิ้นเปลือง Hardware จำนวนมาก รูปที่ 5.4 คือ A/D ขนาด 2 บิตเท่านั้นซึ่งต้องใช้ตัวเปรียบเทียบถึง 3 ตัว เขียนเป็นสูตรได้ดังนี้ 2^N-1 โดย N คือจำนวนบิตที่ต้องการนั้น หมายความว่าถ้าเราต้องการ A/D ขนาด 8 บิตก็จะใช้ตัวเปรียบเทียบถึง 255 ตัว จึงทำให้ A/D แบบนี้ไม่ค่อยเป็นที่นิยม

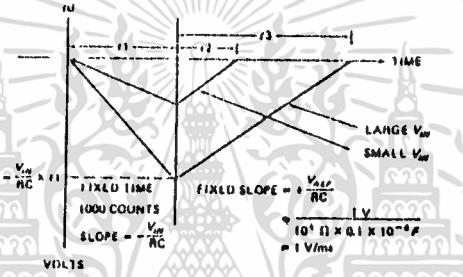
Dual-Slope A/D

A/D แบบนี้มักจะใช้กับ Digital Voltmeter เนื่องจากมีราคาถูกในขณะที่สามารถให้รายละเอียดได้ดี การทำงานเริ่มต้นที่การ Reset ตัว Counters ให้มีค่าเป็น 0 และต่อสัญญาณ Analog(Vin)เท่ากับ Integratorเราสมมติว่า Vin มีค่าเป็นบวก Output จากIntegretor จะแสดงได้ดังรูปที่ 5.6 เมื่อค่า Output จาก Integrator มีค่าเป็นลบจะทำให้ Comparator ทำงานส่งผลให้ AND Gate เปิดและจะทำให้สัญญาณ Clock สามารถผ่านไปยังระบบ Counter

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5.5 Dual-Slope A/D



รูปที่ 5.6 แสดง Output จากการ Integrator

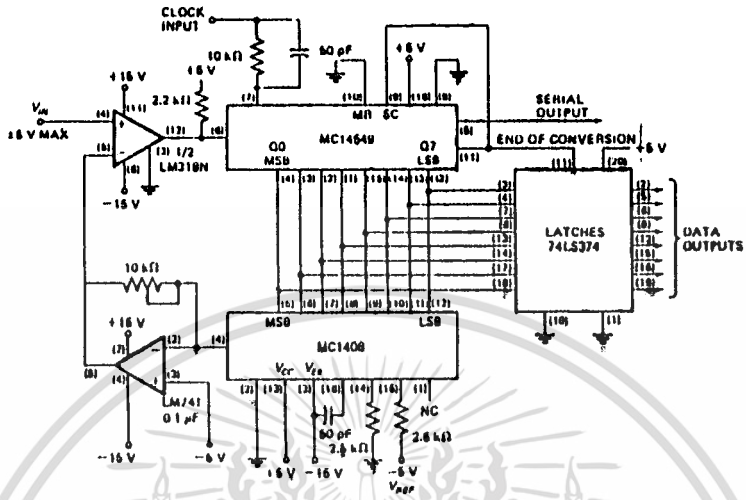
ได้ ค่า Output จาก Integretor นี้จะถูกกำหนด Fixed Time จาก Control Circuit เมื่อถึงจุดนี้จะทำการ Reset Counter อีกครั้งและต่อสัญญาณ Input ของ Integrator เท่ากับ Vref ที่เป็นค่าลบซึ่งจะยังผลให้ Output ของ Integrator มี Slope ขึ้นเป็นบวก Slope ตั้งแต่ขึ้นกระทั่งถึงค่า 0 นี้เอง ที่จะให้ Counter นับความถี่ได้และค่าที่ได้จะสัมพันธ์กับ Vin เริ่มแรกที่วัดได้ A/D แบบนี้มักจะอยู่ในรูปของ IC สำเร็จรูป และมักจะใช้งานในด้าน Digital Voltmeter ซึ่งโดยทั่วไปจะมีค่า Conversion Time ประมาณ 300 mS.

Successive Approximation A/D

จากวงจรเมื่อเริ่มมี Clock เข้ามาที่ Clock Input จะทำให้ Output ของ MC14545 เป็น 1 ที่บิตสูงและจะผ่านไปให้วงจร D/A คือ MC1408 ร่วมกับส่วนขยายสัญญาณ โดยสัญญาณที่ได้จะนำไปเปรียบเทียบกับสัญญาณ Vin ที่ LM319N โดยถ้า Vin มากกว่าก็จะทำให้ Output ของวงจรเปรียบเทียบเป็น 1 ซึ่งก็จะทำให้วงจร SAR ทำงานต่อไป แต่ถ้า Vin น้อยกว่าก็จะทำให้ Output เป็น 0 ซึ่งจะเป็นการหยุดทำงานและให้สัญญาณ End Of Converstion ออกมาเพื่อนำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Data ที่ได้มา Latch ไว้ต่อไป



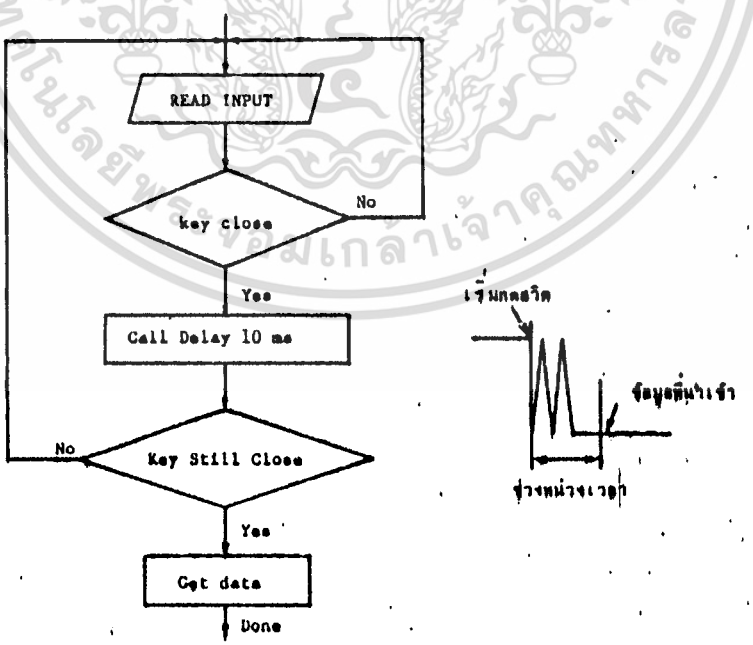
รูปที่ 5.7 Successive Approximation A/D

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการกดสวิทช์โดยทั่วไปจะปรากฏพัลส์ที่ไม่ต้องการขึ้น ซึ่งเกิดผลเนื่องจากการกระเด้งของหน้าสัมผัส ปรากฏการณ์นี้เรียกว่า บาวนซ์ (Bounce) ดังรูปที่ 6.2 ในส่วนวิธีการแก้บาวนซ์ (Debounce) สามารถทำด้วยวิธีการทาง ,Hadrware หรือ Software ก็ได้ วิธีการทาง Hardware โดยทั่วไปจะใช้วงจรแนนด์แลทช์ หรือวงจรชมิทโนสเตเบิ้ล เพื่อแก้สัญญาณนี้

สัญญาณในลักษณะการ Bounce นี้จะมีผลต่อวงจรลอจิกอย่างมาก เช่นถ้าเอาสัญญาณที่จุดนี้ไปเป็น Input ของวงจรนับ จะทำให้การนับผิดพลาดคือเมื่อกดสวิทช์ 1 ครั้งควรจะได้พัลส์ 1 พัลส์ แต่ถ้ามี Bounce เกิดขึ้นจะทำให้ได้พัลส์มากกว่า 1 พัลส์ ค่าที่ได้จึงได้มากกว่าความต้องการ การที่ใช้ CPU อ่านข้อมูลจากสวิทช์ในลักษณะเช่นนี้ข้อมูลที่ได้อาจจะไม่แน่นอน คืออาจจะได้เป็น 1 หรือ 0 ก็ได้ ดังนั้นจึงต้องมีการกำจัดสัญญาณส่วนที่ไม่ต้องการนี้ออกไป ซึ่งอาจทำได้โดยวิธีการ Hadrware หรือ Software แต่วิธีที่นิยมใช้ในระบบ Computer คือใช้ Software เพราะไม่ต้องการอุปกรณ์เพิ่มเติมอื่นๆอีกเลย

การแก้สัญญาณ Bounce โดยใช้ Software โปรแกรมจะทำงานโดยการตรวจสอบการกดสวิทช์ครั้งแรก แล้วรอสักระยะเวลาอีกช่วงหนึ่ง เพื่อให้ผ่านช่วงของสัญญาณ Bounce จากนั้นจึงอ่านสถานะของสวิทช์อีกครั้งหนึ่ง แล้วทดสอบว่าสวิทช์ยังถูกกดหรือไม่ ถ้าสวิทช์ยังถูกกดอยู่ จึงนำข้อมูลนี้เข้าไป วิธีการนี้เป็นวิธีการที่นิยมใช้ทั่วไป และจะใช้เวลาที่หน่วยประมาณ 10 ms พิจารณาฟังงานเพื่อทำการ Debounce ดังรูปที่ 6.3 โดยกำหนดให้สวิทช์ต่ออยู่ที่บิต D_7 ของ Port Input



รูปที่ 6.3 ตัวอย่างฟังงานการแก้ Bounce

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากผังงานจะเห็นว่า เมื่อมีการพบว่าสวิทช์ถูกกดโปรแกรมจะหน่วงเวลาสักระยะหนึ่งก่อนที่จะอ่านข้อมูลใหม่ และตรวจสอบว่าข้อมูลที่อ่านเข้ามาใหม่มีการเปลี่ยนแปลงไปหรือไม่ ถ้ามีการเปลี่ยนแปลงหรือเลิกการกดสวิทช์แล้ว โปรแกรมก็จะตรวจสอบไม่พบการการกดสวิทช์นี้ แต่ถ้าข้อมูลไม่มีการเปลี่ยนแปลงแสดงว่าสวิทช์ยังถูกกดอยู่ ซึ่งข้อมูลนี้จะเป็นข้อมูลที่ถูกต้องและไม่มีการเปลี่ยนแปลงเนื่องจากผลของ Bounce

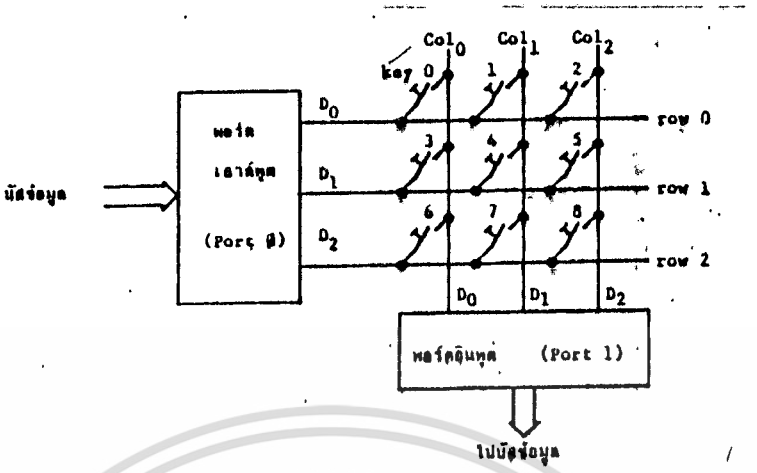
<u>การอ่านข้อมูลจากสวิทช์จำนวนมาก</u>

การต่อสวิทช์แบบปกติจะใช้ 1 บิตต่อสวิทช์ 1 ตัว แต่ในการใช้งานจริงของระบบ Micro computer ต้องใช้สวิทช์หรือคีย์จำนวนมาก ดังนั้นถ้าใช้ 1 สวิทช์ต่อ 1 บิตจะทำให้สิ้นเปลืองPort เป็นจำนวนมาก เช่นถ้าต้องการใช้ 64 สวิทช์ จะต้องใช้ Port แบบขนาน 8 บิตถึง 8 Port วิธีการต่อสวิทช์แบบที่ประหยัด Port และเป็นที่นิยมใช้แบบหนึ่ง คือการต่อแบบเมตริกซ์ ซึ่งจะทำให้ลดจำนวน Port ลง ซึ่งตัวอย่างของวิธีการต่อแบบนี้แสดงดังรูปที่ 6.4 จากรูปเห็นได้ว่าการต่อวงจรสวิทช์แบบนี้ทำให้ลดสายข้อมูลลงไปมาก เช่นถ้าพิจารณาจากจำนวนของสวิทช์และจำนวนของสายเมื่อต่อแบบธรรมดาและจำนวนของสายเมื่อต่อแบบเมตริกซ์ จะเป็นดังนี้

จำนวนสวิทช์	จำนวนสาย	
	ต่อแบบธรรมดา	ต่อแบบเมตริกซ์
1 * 1	1	1
2 * 2	4	4
3 * 3	9	6
4 * 4	16	8
5 * 5	25	10
8 * 8	64	16

จากตารางเห็นได้ชัดว่า เมื่อจำนวนของสวิทช์มากขึ้นก็จะทำให้ประหยัดสายข้อมูลมากขึ้นไปด้วย ข้อยุ่งยากทาง Hardware จะลดลง และการตรวจสอบสวิทช์ที่กดจะใช้วิธีการ Software ช่วยจัดการ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

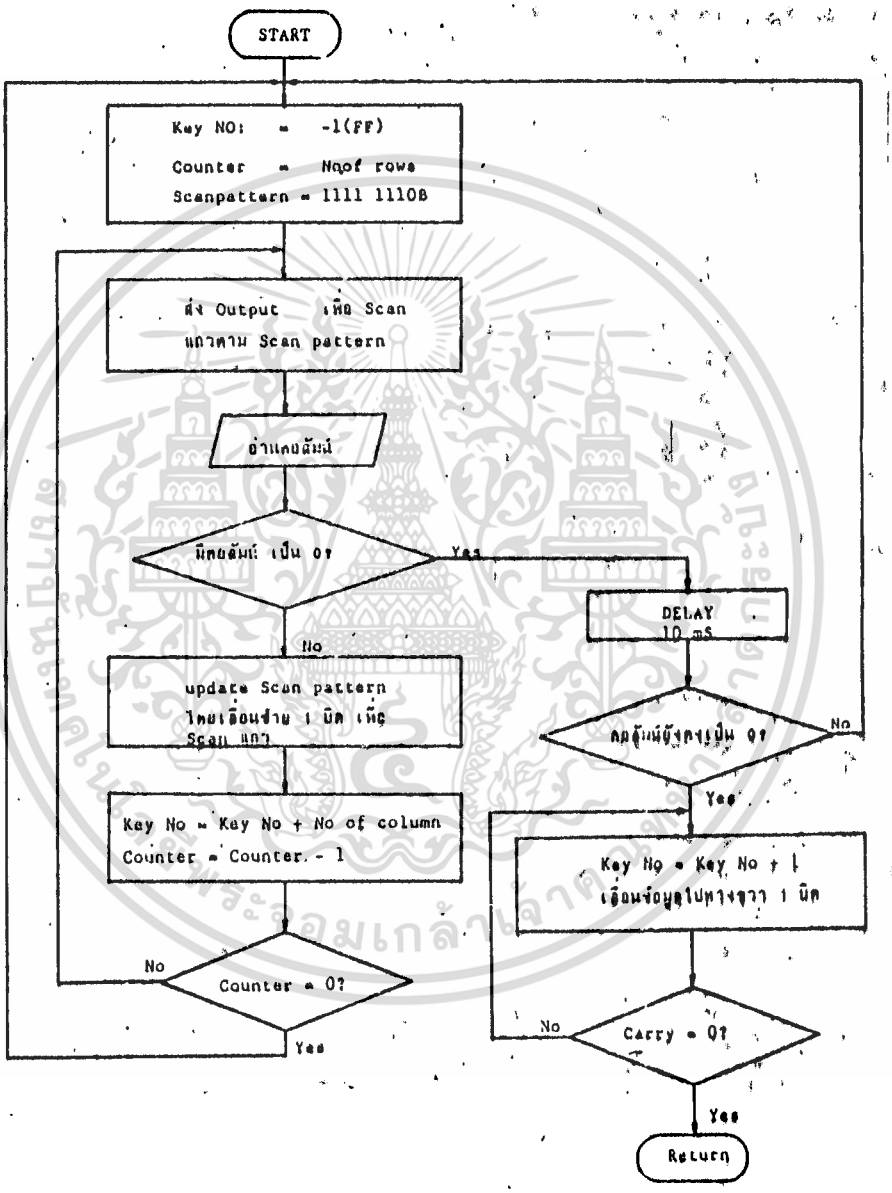


รูปที่ 6.4 การต่อสวิทช์แบบเมตริกซ์ 3 * 3

หลักการทำงานของโปรแกรมจะใช้วิธีการกวาด (Scanning) เพื่อหาว่าสวิทช์ใดที่ถูกกด โดยอาจเริ่มต้นที่เกาที่ 0 โดยกำหนดให้เกาที่ 0 มีระดับเป็น 0 ส่วนเกาอื่นๆ มีระดับเป็น 1 แล้วทำการตรวจสอบทีละ Column โดยเริ่มจาก Column ที่ 0 ตรวจสอบว่าระดับเป็น 0 หรือไม่ ถ้ามีระดับเป็น 0 แสดงว่าสวิทช์ตำแหน่งนี้ถูกกด ถ้าไม่มีก็เลื่อนไป Column ถัดไป และจะทำการเลื่อนและตรวจสอบทั้งทางด้าน Column และ Row ต่อๆไปจนกว่าจะหมดหรือจนกระทั่งพบสวิทช์ที่ถูกกด และเมื่อพบว่าสวิทช์ใดถูกกดแล้วต้องทราบว่าสวิทช์นั้นคือสวิทช์ตำแหน่งใด เพื่อที่จะได้ตีความหมายหรือสร้างรหัสขึ้นมาตามสวิทช์ตำแหน่งนั้นๆ ตัวอย่างการตรวจสอบเพื่อหาตำแหน่งของสวิทช์ที่กดสามารถทำได้ดังแผนงานดังรูปที่ 6.5

จากผังงานแสดงการอ่านสวิทช์ที่ต่อแบบเมตริกซ์ดังรูปที่ 6.5 โดยจะทำการตรวจสอบว่า เมื่อมีการกดสวิทช์ สวิทช์นั้นคือสวิทช์อะไร เพื่อที่จะได้สร้างรหัสของสวิทช์ที่กดนั้น การตรวจสอบสามารถอธิบายเป็นลำดับได้ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6.5 ผังงานแสดงการอ่านคีย์ที่ต่อแบบเมตริกซ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. ตั้งหมายเลขสวิทช์ไว้ที่ -1 ตัวนับ = จำนวนของแถว ตั้งแพทเทอร์นของการกวาดโดยให้ทุกบิตเป็น 1 ยกเว้นบิตของแถว 0 ให้เป็นระดับลอจิก 0
2. ส่งแพทเทอร์นที่ตั้งไว้ไปยัง Port Output ของ Keyboard
3. อ่านค่าข้อมูลจาก Port Input ตรวจสอบว่ามี Column เป็น 0 หรือไม่ ถ้ามีให้ไปลำดับ 7
4. เปลี่ยนค่าแพทเทอร์นที่ใช้ในการกวาดโดยการเลื่อนไปทางซ้าย 1 บิต
5. เพิ่มค่าหมายเลขสวิทช์โดยการบวกด้วยจำนวน Column
6. ลดค่าของตัวนับแถวลงไป 1 ถ้าตัวนับไม่เท่ากับ 0 ให้กลับไปลำดับที่ 2 และถ้าเท่ากับ 0 แล้วแสดงว่าไม่มีการกดสวิทช์ ให้กลับไปเริ่มต้นใหม่
7. เมื่อพบการกดสวิทช์ ให้หน่วงเวลาประมาณ 10 mS เพื่อการ Debounce หลังจากนั้นก็จะอ่านข้อมูลที่ Port Input ใหม่ เพื่อการตรวจสอบว่าสวิทช์ยังคงถูกกดหรือไม่ ถ้ามีการเปลี่ยนแปลงให้กลับไปลำดับที่ 1 ใหม่
8. เพิ่มค่าหมายเลขสวิทช์ขึ้นอีก 1 เลื่อนข้อมูลที่อ่านได้ทาง Column ไปทางขวา 1 บิต
9. ทำการทดสอบเฟลกตัวทด ถ้าเท่ากับ 1 กลับไปลำดับที่ 8
10. เก็บค่าหมายเลขสวิทช์ที่ได้ แล้วกลับสู่โปรแกรมหลัก เพื่อนำค่าหมายเลขสวิทช์ที่ไปตีความเพื่อทำงานต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DOT MATRIX LCD MODULE

เราพอจะแบ่ง Dot Matrix LCD Module นี้ออกได้เป็นพวกๆดังนี้

1. Character LCD Module
2. Graphic LCD Module
3. Segment Display Type LCD Module

โดยในแต่ละแบบนี้ก็จะมีส่วนประกอบใหญ่ๆแบ่งๆได้เป็น

1. **Dot Matrix LCD**

เป็นตัวแสดงผลให้เรามองเห็นในลักษณะการปิดและการเปิดตัวเองกับแสงก็คือ ส่วนของที่เป็นตัวกระจกบรรจุผลึก

2. **Driver**

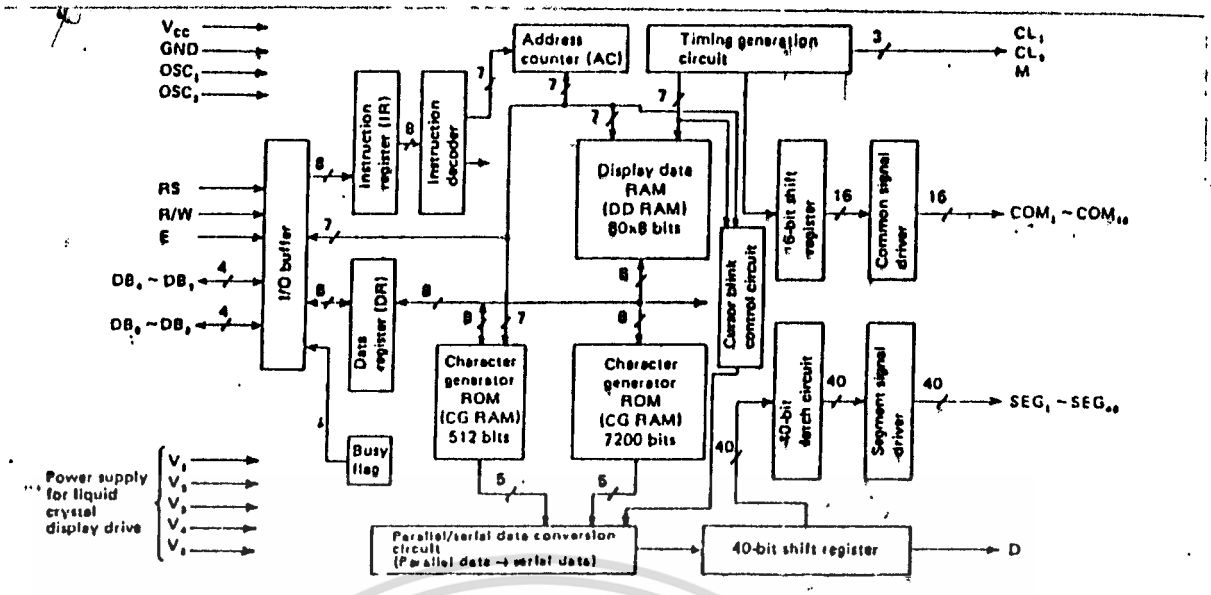
เป็นตัวรับสัญญาณจากตัวควบคุมมาขับผลึก LCD อีกทีหนึ่งโดยที่มีเบอร์ที่นิยมใช้กันใน LCD Module เช่น HD44100H,MSM5259

3. **Controller**

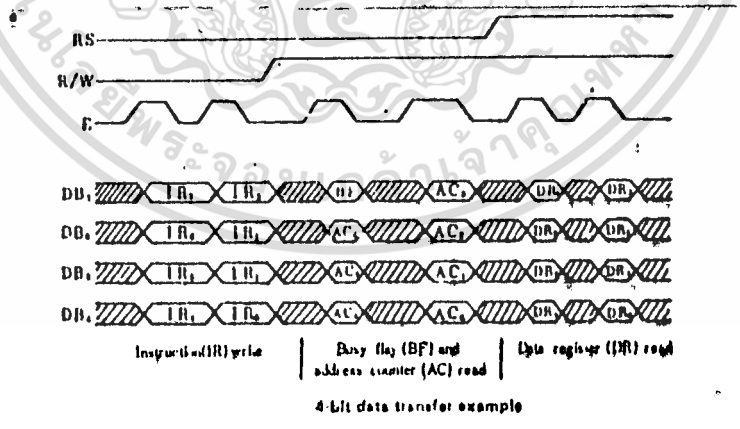
เป็นตัวรับข้อมูลจากภายนอกมาและจัดการควบคุม LCD Module ให้ทำงานแสดงผลในแบบต่างๆ เช่น การลบจอภาพ การเกิดตัวอักษร เป็นต้น ที่มีเบอร์ IC ที่เป็นที่นิยมใช้กันมากก็คือ HD4470 ซึ่งจะใช้ในแบบ Character LCD Module เป็นส่วนใหญ่ ส่วนเบอร์IC HD61830 ก็จะนิยมใช้กันในแบบ Graphic LCD Module

HD4470 เป็น IC แบบ LSI ตัวหนึ่งที่ใช้ในการควบคุม LCD โดยจะมีรูปแบบการแสดงผลในรูปตัวอักษรหรือสัญลักษณ์ต่างๆ ตัวมันเองสามารถที่จะนำมาต่อใช้งานแบบ 4 Bit หรือ 8 Bit ก็ได้ โดยถ้าเราต้องการต่อเพื่อจะนำไปใช้งานในแบบ 4 Bit ก็จะต่อใช้งานที่ DB7-DB4 เท่านั้น โดยข้อมูลครั้งแรกที่ส่งนั้น HD44780 จะถือเป็นข้อมูล 4 Bit บน และข้อมูลที่ส่งต่อมานั้นจะเป็นข้อมูล 4 Bit ล่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

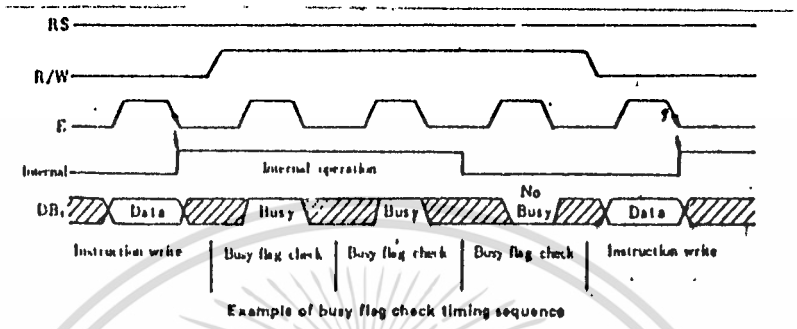


รูปที่ 2 แสดง Block Diagram ของ HD44780



รูปที่ 3 เมื่อข้อมูลเป็นแบบ 4 Bit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4 Busy Flag Check Timing Sequence

หลักการทำงาน

เมื่อเราเริ่มเปิดไฟป้อนให้ HD44780 Reset 10 mS. Vdd 4.5 V. Set

1. Display Clear

จะทำการลบข้อมูลจากจอภาพ LCD

2. Function Set

โดยจะทำการ Set ค่าภายในดังนี้

DL = 1 : เป็นการ Set ให้การติดต่อเป็นแบบ 8 Bit

N = 0 : เป็นการ Set ให้เป็น 1 บรรทัดการแสดงผล

F = 0 : เป็นการ Set ให้เป็น 5*7 Dot ต่อหนึ่งตัวอักษร

3. Display ON/OFF

D = 0 : Display OFF

C = 0 : Cursor OFF

B = 0 : Blink OFF

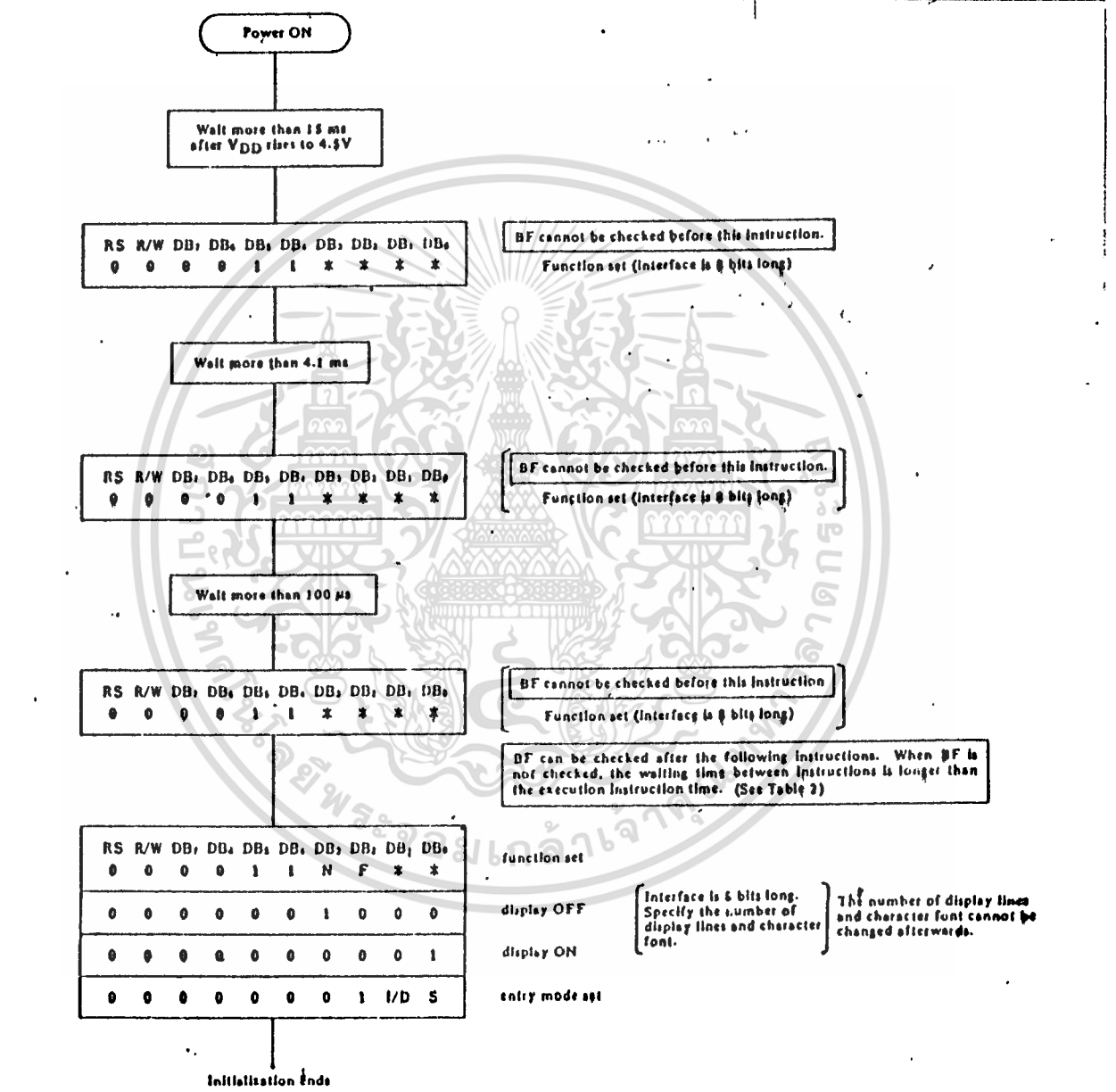
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. Entry Mode Set

I/D = 1 : +1 (เพิ่มค่า Counter ขึ้น 1)

S = 0 : No Shift

เมื่อเราเริ่มเปิดเครื่องให้ทำงานแล้วก็จะต้องส่งคำสั่งควบคุมให้มันเริ่มทำงานดังตารางที่ 5



รูปที่ 5 แสดงคำสั่งควบคุมที่จะต้องป้อนให้เริ่มทำงานหลังเปิดเครื่องแล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Instruction	Code										Description	Execution time (when fosc is 250 kHz) (Note 1)	Execution time (when fosc is 190 kHz) (Note 2)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0			
Clear display	0	0	0	0	0	0	0	0	0	1	Clears all display and returns the cursor to the home position (Address 0).	82 μs ~ 1.64 ms	120 μs ~ 4.9 ms
Return home	0	0	0	0	0	0	0	0	1	*	Returns the cursor to the home position (Address 0). Also returns the display being shifted to the original position. DD RAM contents remain unchanged.	40 μs ~ 1.6 ms	120 μs ~ 4.8 ms
Entry mode set	0	0	0	0	0	0	0	1	I/D	S	Sets the cursor move direction and specifies or not to shift the display. These operations are performed during data write and read.	40 μs	120 μs
Display ON/ OFF control	0	0	0	0	0	0	1	D	C	B	Sets ON/OFF of all display (D), cursor ON/OFF (C), and blink of cursor position character (B).	40 μs	120 μs
Cursor and display shift	0	0	0	0	0	1	S/C	R/L	*	*	Moves the cursor and shifts the display without changing DD RAM contents	40 μs	120 μs
Function set	0	0	0	0	1	DL	N	F	*	*	Sets interface data length (DL) number of display lines (L) and character font (F).	40 μs	120 μs
Set CG RAM address.	0	0	0	1	A_{CG}						Sets the CG RAM address. CG RAM data is sent and received after this setting.	40 μs	120 μs
Set DD RAM address	0	0	1	A_{DD}							Sets the DD RAM address. DD RAM data is sent and received after this setting.	40 μs	120 μs
Read busy flag & address	0	1	BF	AC							Reads Busy flag (BF) indicating internal operation is being performed and reads address counter contents.	1 μs	1 μs
Write data to CG or DD RAM	1	0	Write Data								Writes data into DD RAM or CG RAM.	40 μs	120 μs
Read data to CG or DD RAM	1	1	Read Data								Reads data from DD RAM or CG RAM.	40 μs	120 μs

I/D = 1: Increment (+1) I/D = 0: Decrement (−1)
S = 1: Accompanies display shift.
S/C = 1: Display shift S/C = 0: Cursor move
R/L = 1: Shift to the right.
R/L = 0: Shift to the left.
DL = 1: 8 bits DL = 0: 4 bits
N = 1: 2 lines N = 0: 1 line
F = 1: 5 × 10 dots F = 0: 5 × 7 dots
BF = 1: Internally operating
BF = 0: Can accept instruction

DD RAM: Display data RAM
CG RAM: Character generator RAM
A_{CG}: CG RAM address
A_{DD}: DD RAM address
Corresponds to cursor address.
AC: Address counter used for both of DD and CG RAM address.

Execution time changes when frequency changes.
(Example)
When fosc is 270 kHz:

$$40\ \mu s \times \frac{250}{270} = 37\ \mu s$$

*No effect
Notes 1. Applied to models driven by 1/8 duty or 1/11 duty.
2. Applied to models driven by 1/16 duty.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รายละเอียดของคำสั่ง HD44780

1. **Clear Display**

	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	0	0	0	0	0	0	0	0	0	1

คำสั่งนี้จะเป็นการเขียนช่องว่าง Space (ASCII 20H) เข้าไปใน DD Ram ทั้งหมด และทำการ Set DD Ram Address เป็นศูนย์ ตัว Cursor จะกลับไปอยู่ตำแหน่งบนสุดทางซ้ายมือของจอภาพ Set I/D = 1, S ไม่มีการเปลี่ยน

2. **Return Home**

	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	0	0	0	0	0	0	0	0	1	*

* No Effect

คำสั่งนี้จะทำการ Set DD Ram Address เป็นศูนย์ ตัว Cursor จะกลับไปอยู่ตำแหน่งบนสุดทางซ้ายมือของจอภาพ ข้อมูลในจอภาพไม่เปลี่ยน

3. **Entry Mode Set**

	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	0	0	0	0	0	0	0	1	I/D	S

Bit I/D : โดยจะเป็นตัวกำหนดให้ว่าเมื่อเขียนหรืออ่านข้อมูลแล้วจะทำให้ DD Ram Address เพิ่มขึ้นหนึ่งหรือลดลงหนึ่ง โดย 1 = เพิ่มขึ้นหนึ่ง, 0 = ลดลงหนึ่ง

Bit S : เป็นตัวกำหนดแสดงผลโดย

S = 1 จะเป็นการใส่ข้อมูลแล้วตัว Cursor อยู่กับที่ ข้อมูลจะถูกดันไปทางซ้าย

S = 0 ข้อมูลจะอยู่กับที่ Cursor จะถูกดันไปทางขวามือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. Display ON/OFF Control

	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	0	0	0	0	0	0	1	D	C	B

Bit D : เป็น Bit ให้เปิดปิดหน้าจอภาพรวมถ้า D = 1 จะ ON และ D = 0 จะ OFF

Bit C : จะให้เสดง Cursor ให้Bit C = 1 และถ้าไม่ต้องการแสดงCursor Bit C = 0 โดยที่ตัว Cursor จะอยู่ที่ Line ที่ 8 ในแบบ 5*7 Dot และจะอยู่ Line 11 ในแบบ 5*10 Dot

Bit B : เป็น Bit Set การกระพริบของ Cursorโดย B = 1 มีการกระพริบ B = 0 ไม่มีการกระพริบ โดยระยะเวลาการกระพริบประมาณ 379.2 mS.

5. Cursor Or Display Shift.

	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	0	0	0	0	0	1	S/C	R/L	*	*

* No Effect

เป็นคำสั่งกำหนดให้ตำแหน่ง Cursor หรือข้อมูลไปเกิดทางซ้ายหรือขวามือโดยไม่ต้องใช้คำสั่งเขียนหรืออ่าน โดย

S/C	R/C	
0	0	ทำการย้าย Cursor ไปจากตำแหน่งเดิมไปทางซ้ายมือ 1 ตำแหน่ง
0	1	ทำการย้าย Cursor ไปจากตำแหน่งเดิมไปทางขวามือ 1 ตำแหน่ง
1	0	เป็นการดันตัวอักษรที่เกิดไปทางซ้ายมือ
1	1	เป็นการดันตัวอักษรที่เกิดไปทางขวามือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

6. Function Set

	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	0	0	0	0	1	DL	N	F	*	*

* No Effect

Bit DL : เป็นการ Set การติดต่อว่าจะให้เป็นแบบ 8 Bit หรือ 4 Bit โดย ถ้าต้องการติดต่อแบบ 4 Bit DL = 0 และ 8 Bit DL = 1

Bit N : เป็นการ Set บรรทัดการแสดงผล N = 0 แสดง 1 บรรทัด N = 1 แสดง 2 บรรทัด ในกรณีมากกว่า 2 บรรทัด ก็ให้ Set N = 1

Bit F : เป็นการ Set ขนาด Dot การแสดงผล 5*7 หรือ5*10 โดย F = 0 เป็นแบบ 5*7 และ F = 1 เป็นแบบ 5*10

7. Set CG Ram Address

	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	0	0	0	1	A	A	A	A	A	A

ใน HD44780 นั้นมีหน่วยความจำอยู่ 2 ชุด คือ Display Data Ram (DD Ram) จำนวน 80*8 Bit และ Character Generator Rom CG Ram จำนวน 512 Bit และ 7200 Bit คำสั่งนี้เป็นการ Set Address ใน CG Ram โดยต้องทำการ Set Address ก่อนเขียนหรืออ่านข้อมูลจาก CG Ram ด้วย

8. Set DD Ram Address

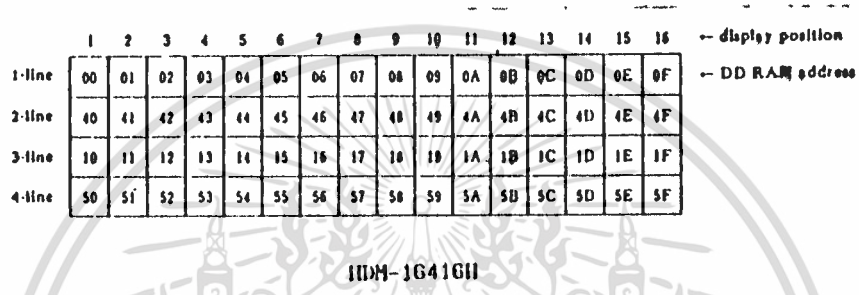
	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	0	0	1	A	A	A	A	A	A	A
				Higher Order Bits					Lower Order Bits	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นการ Set ค่า Address ใน DD Ram ในการเขียนหรืออ่านค่าจาก DD Ram (DD Ram คือส่วนที่จะแสดงผลหน้าจอ LCD) โดยจำนวน Address ที่จะเกิดขึ้นบนจอ LCD จะอยู่กับการ Set ค่า N ด้วย

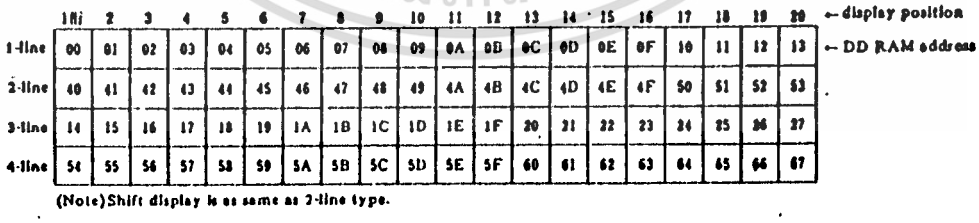
ถ้า N = 0 (1 บรรทัด) Address จะอยู่ที่ 00H-4FH

ถ้า N = 1 (2 บรรทัด) Address จะอยู่ที่ 00H-27H สำหรับในบรรทัดที่ 1 และ 40H-67H สำหรับในบรรทัดที่ 2



HD44780

รูปที่ 6 การจัด Address ของ DD Ram หน้าจอ LCD แบบ 16 ตัวอักษร 4 บรรทัด



รูปที่ 7 การจัด Address ของ DD Ram หน้าจอ LCD แบบ 20 ตัวอักษร 2 บรรทัด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

9. **Read Busy Flag And Address**

	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	0	1	BF	A	A	A	A	A	A	A
			Higher Order Bits						Lower Order Bits	

เป็นการอ่านค่า Busy Flag ซึ่งจะเป็นตัวบอกว่าตัว HD44780 นี้อยู่ในขบวนการทำงานภายในอยู่หรืออยู่ในสภาพพร้อมรับข้อมูล โดย

Bit BF = 1 อยู่ในขบวนการทำงานภายในไม่พร้อมจะรับข้อมูลหรือคำสั่ง

Bit BF = 0 พร้อมที่จะรับข้อมูลหรือคำสั่ง

และนอกจากนี้ยังเป็นคำสั่งอ่านค่าข้อมูล Address ของ CG Ram หรือ DD Ram ด้วย

10. **Write Data To CG Or DD Ram**

	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	1	0	D	D	D	D	D	D	D	D
			Higher Order Bits						Lower Order Bits	

เป็นคำสั่งเขียนข้อมูลเข้าไปใน CG หรือ DD Ram โดยเมื่อเขียนข้อมูลและ Address จะลดและเพิ่มโดยอัตโนมัติตามคำสั่งที่ Set ไว้ที่ Entry Mode ข้อกำหนดที่จะรู้ว่าเป็นการเขียนข้อมูลของ CG Ram หรือ DD Ram ทำได้โดยการ Set Address ของ CG Ram หรือ DD Ram ขึ้นมาก่อนที่จะเขียนข้อมูล

11. **Read Data From CG Or DD Ram**

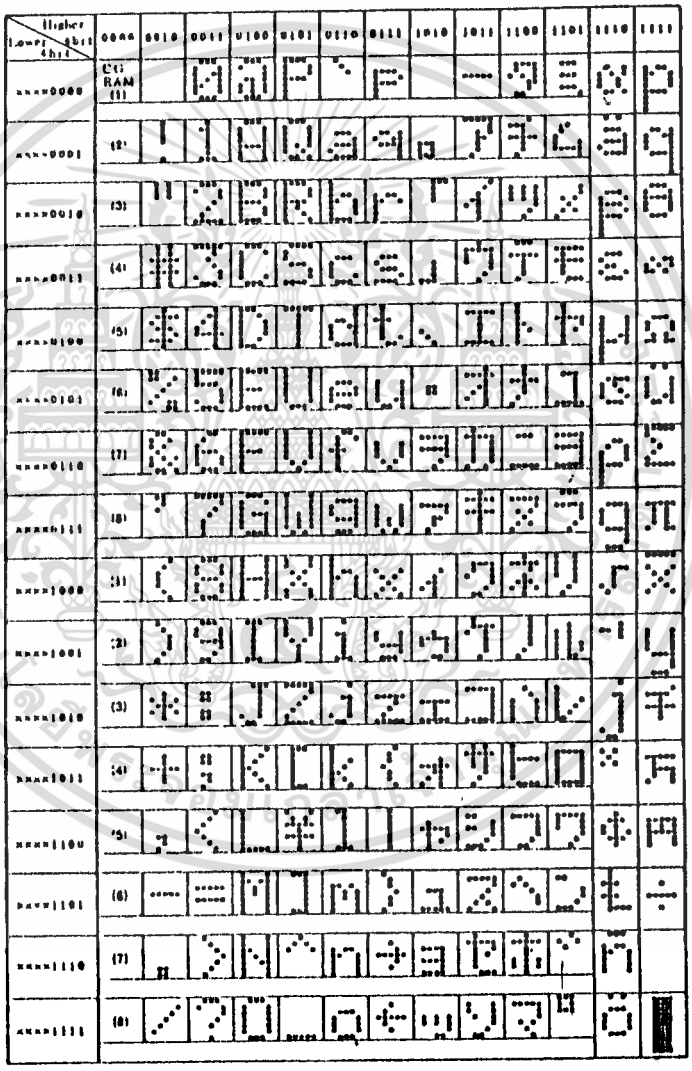
	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0
Code	1	1	D	D	D	D	D	D	D	D
			Higher Order Bits						Lower Order Bits	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นคำสั่งอ่านค่าข้อมูลจาก CG Ram หรือ DD Ram โดยก่อนอ่านค่าจาก DD Ram หรือ CG Ram นี้ควรจะใช้คำสั่ง Set Address ก่อนเพื่อให้รู้ว่าข้อมูลที่อ่านได้นั้นเป็น DD หรือ CG Ram

จากตารางการทำงานจะเห็นว่าการใช้งาน LCD Module นั้นง่ายเพียงแค่เราส่งคำสั่งเริ่มแรกและ Set ความต้องการในขนาดตัวอักษรและตัว Cursor หลังจากนั้นเราก็สามารถเขียนตัวอักษรเข้าไปใน DD Ram ตามตารางตัวอักษรก็จะเกิดอักษรในจอภาพ LCD เรายังสามารถกำหนดตำแหน่งตัวอักษรที่จะทำให้เกิดบนจอได้โดยการ Set DD Ram Address

CHARACTER FONT TABLE



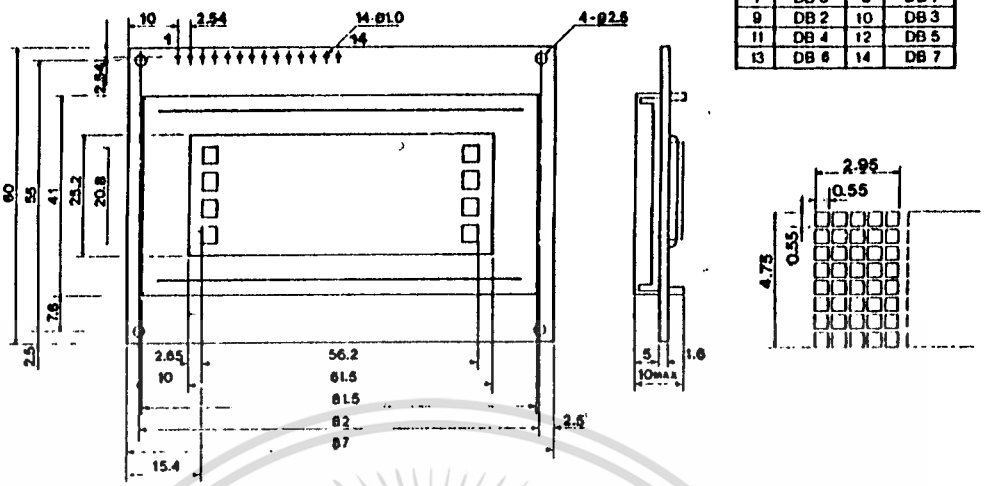
NOTE: CGRAM is a CHARACTER GENERATOR RAM having a storage function of character pattern which enable to change freely by user's program.

รูปที่ 8 Character Font Table

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DV-1604

No.	Signal	No.	Signal
1	VSS	2	VDD
3	V0	4	RS
5	R/W	6	E
7	DB 0	8	DB 1
9	DB 2	10	DB 3
11	DB 4	12	DB 5
13	DB 6	14	DB 7



รูปที่ 9 LCD # DV-1604

ขาต่างๆในการต่อใช้งาน HD44780

1. RS (Registor Selection)

จะเป็นขาเลือก Registor ภายในซึ่งมีอยู่ 2 ตัวคือ Instruction Registor (IR) และ Data Registor (DR) โดยถ้าเลือกเป็น 1 จะเป็นการเลือก Data แต่ถ้าเลือกเป็น 0 จะเป็นการเลือก Instruction

2. R/W (Read/Write)

เป็นตัวเลือกว่าจะเขียนหรืออ่านข้อมูลจากตัว IC โดยอ่านข้อมูล = 1, เขียนข้อมูล = 0

3. E (Enable Signal)

เป็นขากำหนดสภาพการรับเขียนอ่านข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

The relation between the operation and the combination of RS, R/W

RS	RW	E	OPERATION
0	0		Write instruction code
0	1		Read busy flag and address counter
1	0		Write data
1	1		Read data

When performing data and instruction code by 4 bit, transfer RS, R/W every time.

รูปที่ 10 การเลือกว่าจะอ่านหรือเขียนข้อมูล

4. DB0 – DB7

เป็นขารับสัญญาณข้อมูลจาก IC

5. VDD

ไฟเลี้ยงวงจร

6. VSS

เป็นขา Gnd

7. VO

เป็นขารับ Voltage ในการขับ LCD ให้สว่างหรือมืด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ANT - 32

ANT - 32 คือบอร์ดไมโครคอนโทรลเลอร์ขนาดเล็ก ที่สามารถใช้ได้ทั้งภาษา Basic และภาษา Assembly สำหรับการพัฒนาระบบควบคุมอัตโนมัติและใช้งานในลักษณะเป็น Embedded Controller โดยใช้ชิพไมโครคอนโทรลเลอร์ เบอร์ 8032 ซึ่งอยู่ในตระกูล MCS - 51 ทำหน้าที่เป็น CPU

ในการพัฒนาโปรแกรม ANT - 32 สามารถใช้เครื่องคอมพิวเตอร์ PC ทั่วไปเป็นเครื่องมือช่วยการในการพัฒนาโปรแกรม ซึ่งจะทำให้มีความรวดเร็วขึ้น และง่ายต่อการแก้ไข ทั้งนี้ในการพัฒนาโปรแกรมจะกระทำโดยผ่าน Port RS232C โดยที่เครื่องคอมพิวเตอร์ PC จะใช้ Commuication Software ทั่วไป เช่น Procom,Crosstalk จัดการให้เครื่องคอมพิวเตอร์ PC ทำหน้าที่เสมือนเป็น จอภาพ และคีย์บอร์ดของ ANT - 32

ANT - 32 กับภาษาแอสเซมบลี้

การพัฒนาโปรแกรมด้วยภาษาแอสเซมบลี้ มีหลักการดังนี้

1. การเขียนโปรแกรมแอสเซมบลี้ให้ใช้โปรแกรม Editor ทั่วไป และจุดเริ่มต้นของโปรแกรม (Origin) ต้องอยู่ที่ Address 0000H

2. โปรแกรมแอสเซมบลี้ที่เขียนเสร็จแล้ว นำมาแปลงเป็น Object Code โดยใช้โปรแกรมตัวแปลภาษาแอสเซมบลี้ 8051 ทั่วไปหรือใช้โปรแกรม SXA51.EXE ก็ได้ โดย File ที่แปลได้จะอยู่ในลักษณะของ Intel Hex File

3. File ที่แปลเรียบร้อยแล้ว สามารถส่งไปยัง ANT-32 ได้โดยใช้ EE-232 (EPORM Emulator) โดยซ็อคเก็ตของ EE-232 เสียบอยู่ที่ ANT-32 แทนที่ Rom ANT-32MON แล้วทำการส่ง File จากเครื่องคอมพิวเตอร์ PC ไปที่ EE-232 โดยผ่าน Port RS232

4. กรณีที่ไม่มี EE-232 ให้นำโปรแกรมที่แปลได้ ซึ่งอยู่ในลักษณะของ Intel Hex นำไปแปลเป็น Object File ก่อน แล้วนำไปโปรแกรมลง EPORM โดยเครื่องโปรแกรม EPORM ทั่วไป จากนั้นนำ EPORM ที่โปรแกรมเรียบร้อยแล้ว ใส่ลง ANT-32 แทนที่ Rom ANT-32MON (Socket U2)

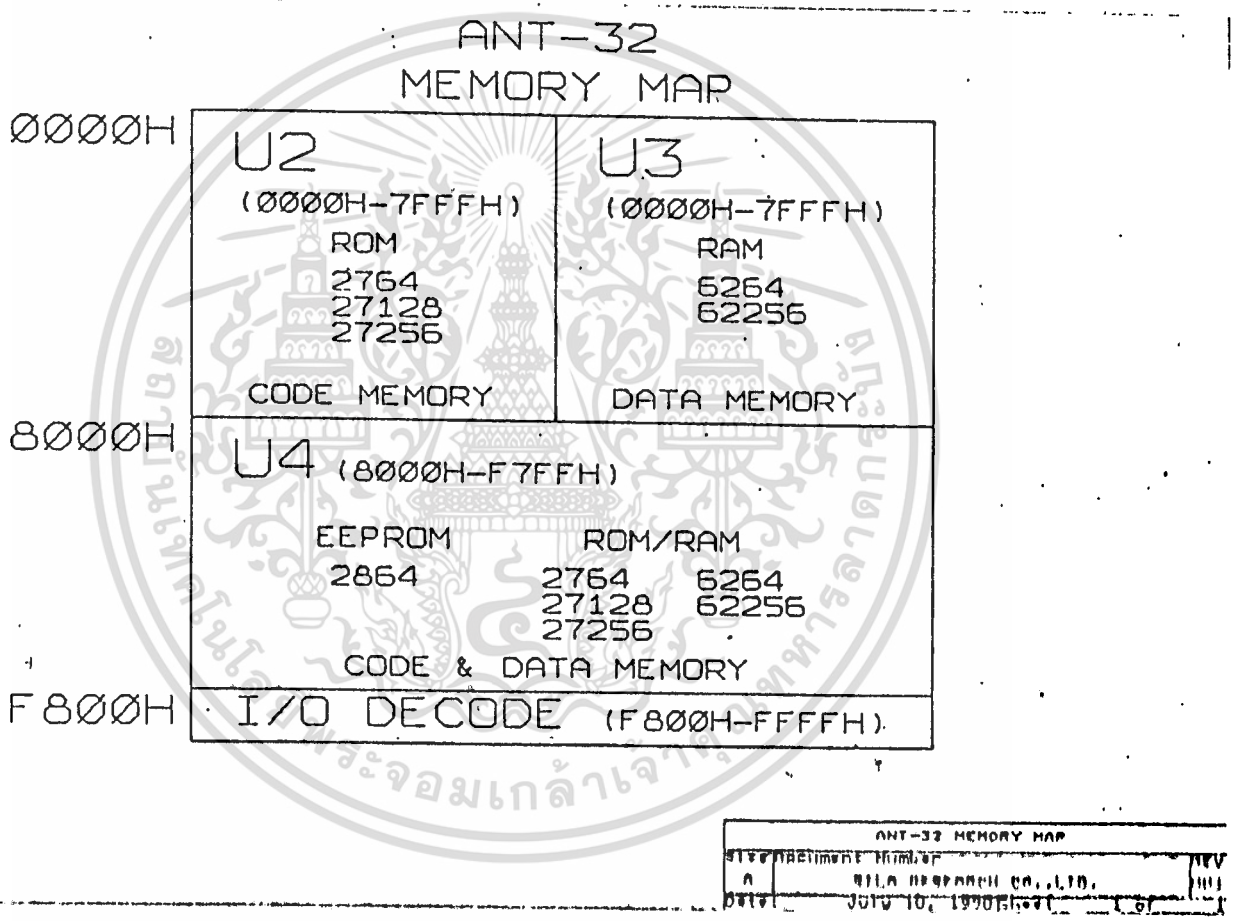
5. กดสวิทช์ Rset บน ANT-32 ถ้าโปรแกรมถูกต้อง ANT-32 จะทำงานตามโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ANT-32 SPECIFICATION

CPU :	8032
CPU CLOCK :	11.0592 MHz
INTERNAL MEMORY :	256 BYTE (RAM)
EXT.CODE MEMORY :	(U2) 8-32K SELECT 2764,27128,27256 (ROM)
EXT.DATA :	(U3) 8-32K SELECT 6264,62256 (RAM)
EXT.CODE & DATA MEMORY :	(U4) 8-30K SELECT 2764,27128,27256 (ROM)
	2864 (EEPROM)
	6264,62256 (RAM)
INTERNAL PORT :	12 BIT I/O
EXTERNAL PORT :	USER1 8255 PORT I/O 24 BIT
	USER2 8255 PORT I/O 24 BIT
BACKUP :	DATA MEMORY (U3) 52 HOUR
CHARGE TIME :	48 HOUR
LANGUAGE :	MCS BASIC-52
	ASSEMBLY (BY DOWNLOAD HEX FILE)
CONNECTOR :	16P INTERNAL PORT
	26P USER1 PORT
	26P USER2 PORT
	2P POWER SUPPLY
	3P SERIAL INTERFACE (RS232)
SERIAL INTERFACE :	RS232C
POWER :	5 V. DC 290 mA. (U4 NOT INCLUDE)
SIZE :	5.25" * 3.9"
OPTION :	BATTARY NI-CAD 3.6 V. 60mA.

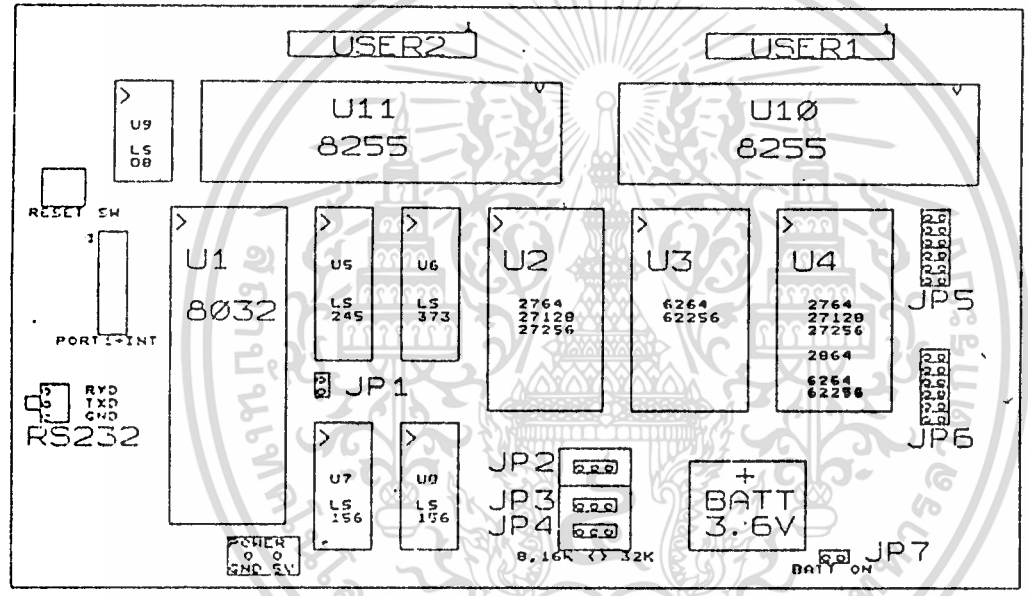
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 11 ANT-32 MEMORY MAP

รูปที่ 12 ANT-32 LAYOUT

รูปที่ 13 ANT-32 LAYOUT



SELECT U2 BY JP2

2764/128 (8,16K)

27256 (32K)

SELECT U3 BY JP3,JP4

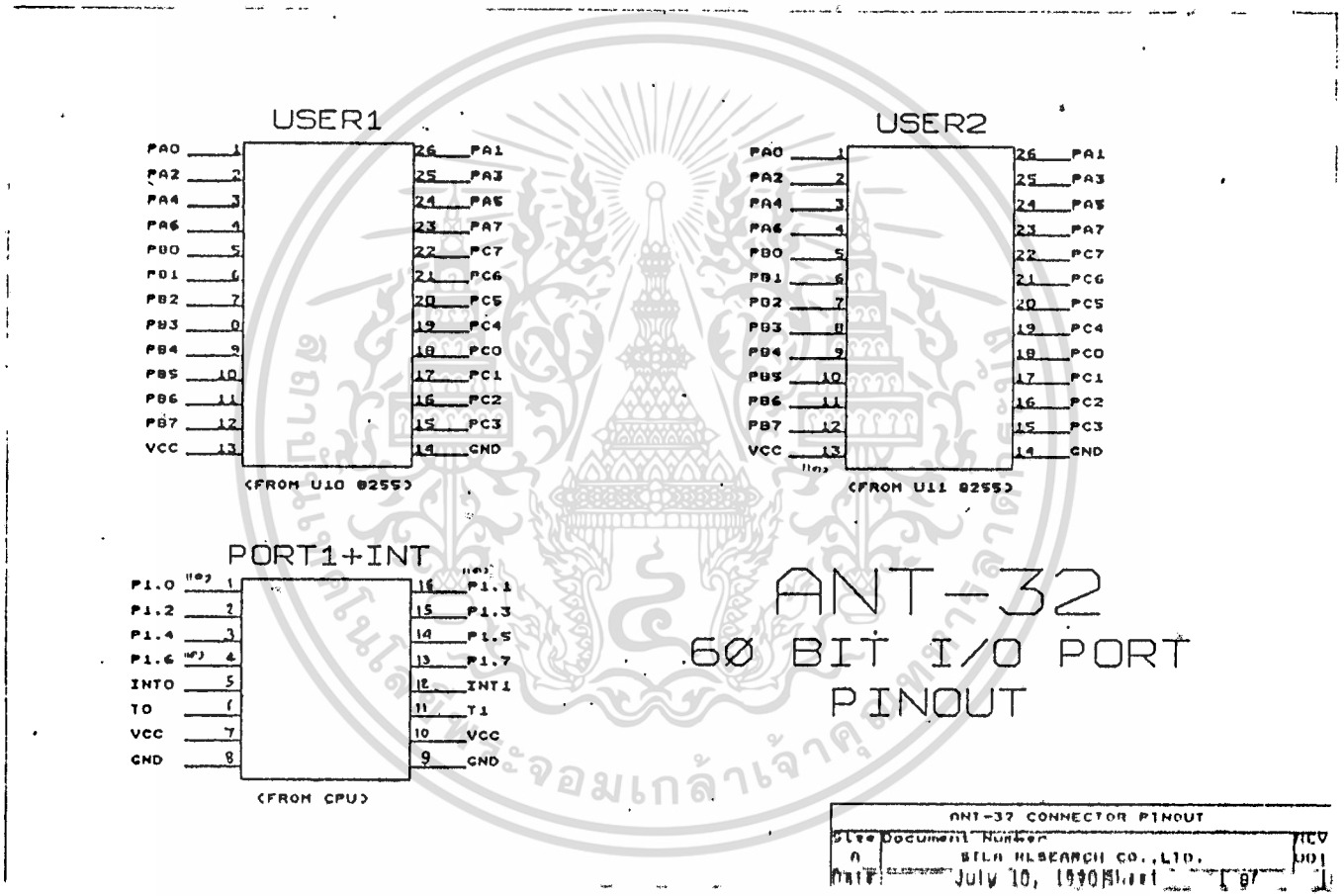
6264 (8K)

62256 (32K)

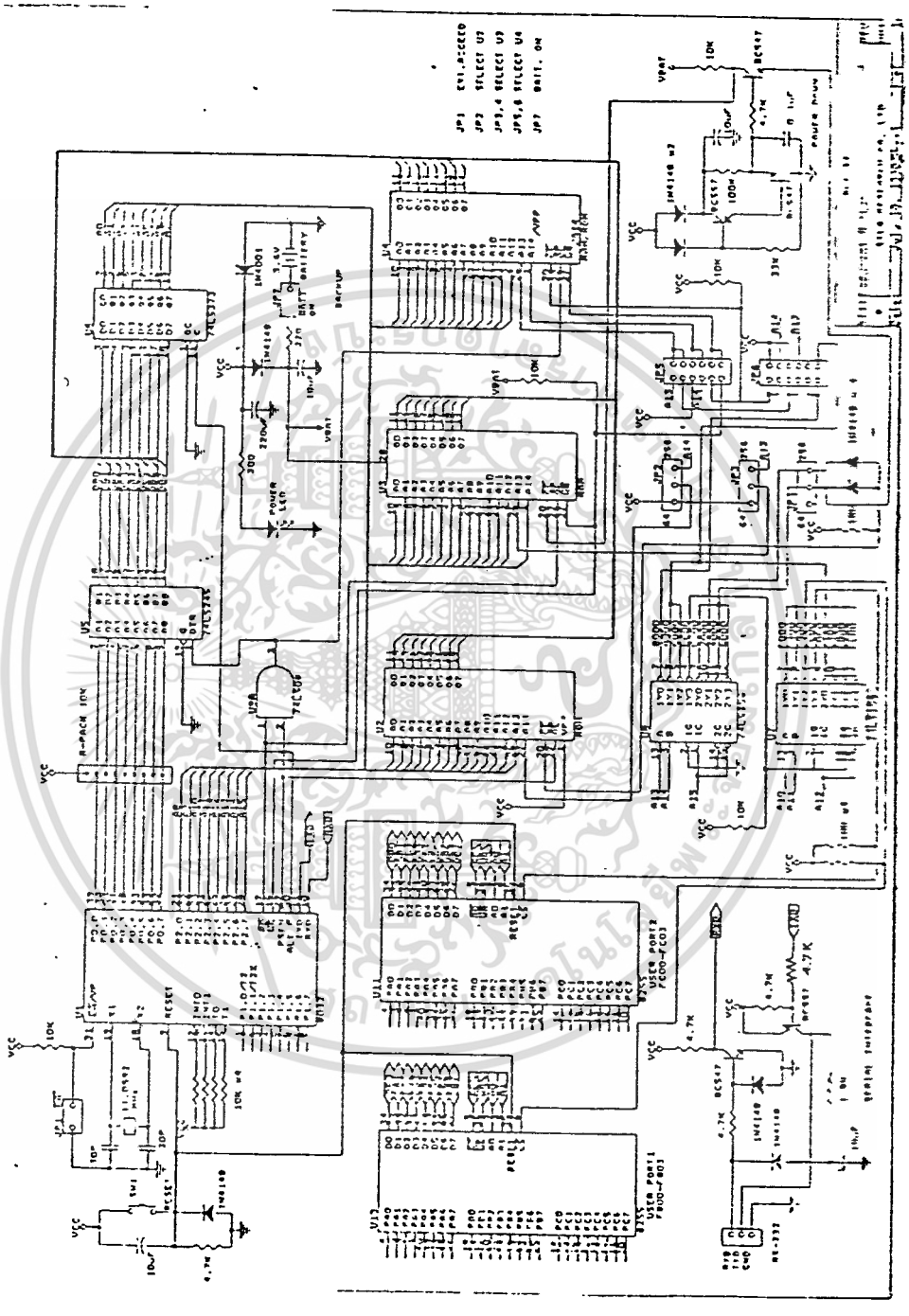
SELECT U4 BY JP5,JP6

SEE NEXT PAGE

ANT-32 LAYOUT		
Size: A	Document Number: SILA RESEARCH CO.,LTD.	REV: 001
Date: July 10, 1990	Sheet 1 of	

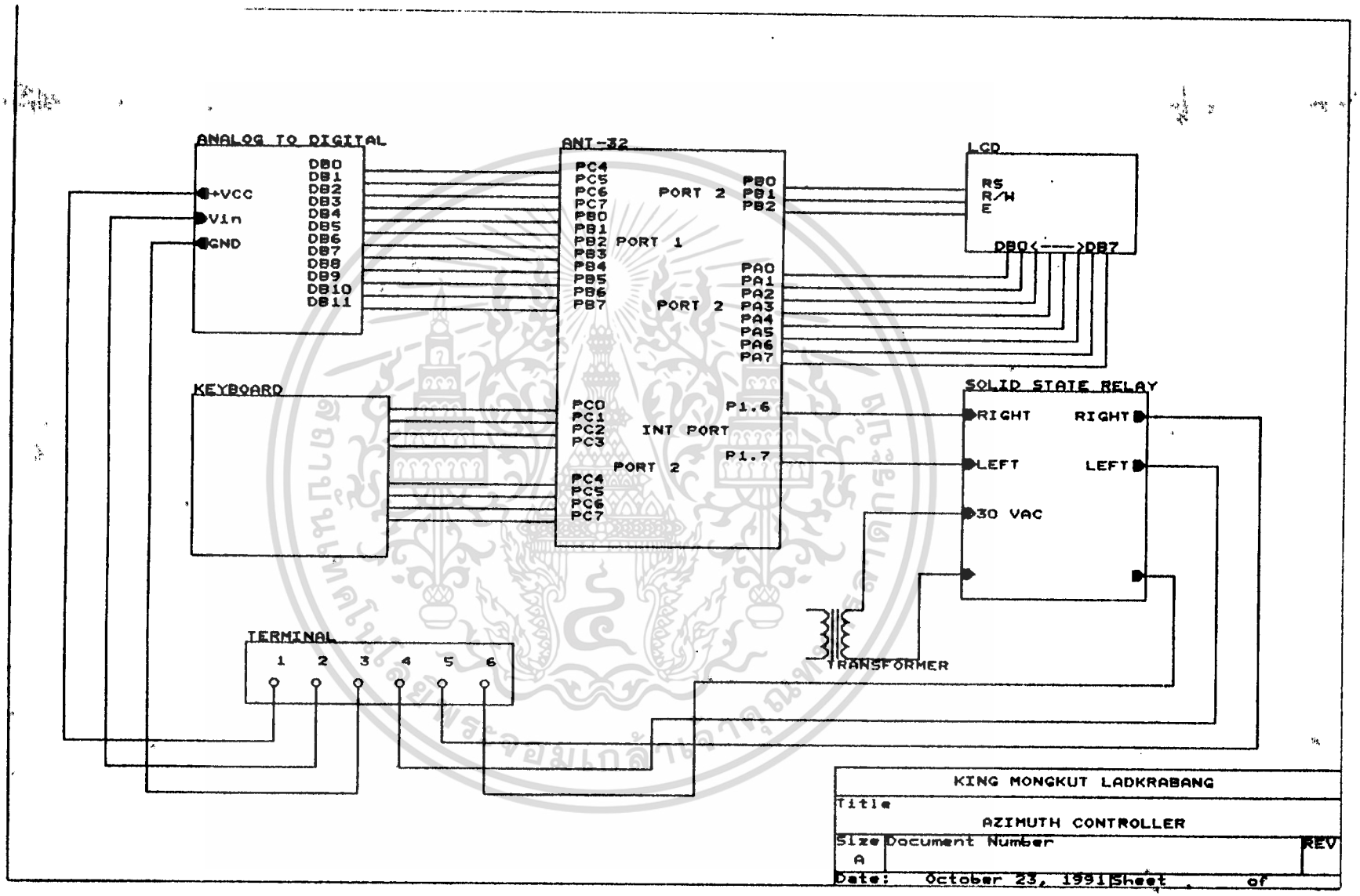


รูปที่ 14 ANT-32 CONNECTOR PINOUT



รูปที่ 15 ANT-32

รูปที่ 17 Keyboard Interface



รูปที่ 18 แสดง Block Diagram ของวงจรควบคุมทิศทาง

กิตติกรรมประกาศ

ในการจัดทำโครงงานนี้ สามารถประสบผลสำเร็จได้ก็ด้วยความร่วมมือจากหลายฝ่าย รวมทั้งอาจารย์ที่ได้ให้คำแนะนำทั้งในด้านทฤษฎีและทางปฏิบัติ

ในโอกาสนี้คณะผู้จัดทำขอขอบคุณ ผู้มีส่วนร่วมในการทำให้โครงงานนี้สำเร็จตามวัตถุประสงค์ทุกท่าน

ท้ายสุดนี้คณะผู้จัดทำขอขอบคุณผู้ที่มีส่วนสำคัญที่สุดของโครงงานนี้คืออาจารย์ภาคร ทุตะสังกาศ ที่ได้ช่วยเป็นที่ปรึกษาในการจัดทำโครงงานนี้ให้สำเร็จลุล่วงไปได้ด้วยดี

ขอขอบพระคุณอย่างสูง

คณะผู้จัดทำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

โยธิน เปรมปราชีวิรัชต์, ระบบเซอร์โวและอิเลคทรอนิกส์คอนโทรลเลอร์ :1990

ETT CO., LTD., ET HARDWARE LAB :1990 .กรุงเทพ

ETT CO., LTD., DOT MATRIX LCD MODULE .กรุงเทพฯ

SILA RESEARCH CO., LTD.,ANT-32 EMBEDDED CONTROLLOR BOARD.กรุงเทพฯ

KENNETH J.AYALA,THE 8051 MICROCONTROLLER ARCHITECTURE,PROGRAMMING AND APPLICATION.WEST PUBLISHING COMPANY :1991

PHILIPS,SIGNATICS MICROCONTROLLER USERS'GUIDE. :1989

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้