การพัฒนาโปรแกรมระบบของ 8048 บน MPF-IP

โดย

นายชัยยศ แซ่ลิ้ม

นายวนวัฒน์ ชัยชนะ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาเทคนิคอุตสาหกรรม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2534

008511

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การพัฒนาโปรแกรมระบบของ 8048 บน MPF-IP

นาย ชัยยศ แซ่ลี้
นาย วนวัฒน์ ชัยชนะ

ได้รับการพิจารณาอนุมัติให้นับเป็นส่วนหนึ่งของการศึกษา
ตามหลักสูตรปริญญาอุตสาหกรรมศาสตร์บัณฑิต
สาขาเทคโนโลยีอิเล็กทรอนิกส์

คณะกรรมการตรวจสอบปริญญานิพนธ์

.....ประธานกรรมการ
( )
.....กรรมการ
( )
.....กรรมการ
( )

วันที่....เดือน.....พ.ศ.....

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้า
บทคัดย่อ	ก
บทนำ	ข
ส่วนที่ 1	
-ส่วนของ 8048 INTERFACE	A-1
-โครงสร้างทางฮาร์ดแวร์ภายใน MPF-IP	A-4
-ฮาร์ดแวร์ของชุดพัฒนาระบบ 8048	A-6
-การทดลอง MICRO COMPUTER เบื้องต้น	A-1-1
-การทดลอง SOLID STATE RELAY	A-1-7
-การทดลอง PRINTER	A-1-11
-การทดลอง DISPLAY	A-1-25
-การทดลอง DOT MATRIX LCD MODULE	A-1-29
-การทดลอง ANALOG TO DIGITAL	A-1-46
-SCHMATIC DIAGRAM	A-1-51
ส่วนที่ 2	
-8251 CHIP SUPPORT	B-1
-LINE DRIVER RS-232C	B-10
-การ LINK MPF-IP กับ PC	B-14
-FLOW CHART	B-16
-SOFT WARE PROGRAM	B-18
-SCHMATIC DIAGRAM	B-23
บรรณานุกรม	ค
ภาคผนวก	ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชื่อเรื่องปริญญานิพนธ์	การพัฒนาโปรแกรมระบบของ 8048 บน MPF-IP
ชื่อผู้เขียน	นาย ชัยยศ แซ่ลี้
	นาย วนวัฒน์ ชัยชนะ
อาจารย์ที่ปรึกษา	อาจารย์ อรรถสิทธิ์ หล่าสกุล
ปริญญานิพนธ์	อุตสาหกรรมศาสตร์บัณฑิต สาขาวิชาเทคโนโลยีอิเล็ก-ทรอนิกส์ สถาบันเทคโนโลยีพระจอมเกล้า ลาดกระบัง
	ปีการศึกษา 2534

บทคัดย่อ

เพื่อให้การใช้งานของ MPF-IP ที่มีอยู่ มีประสิทธิภาพมากขึ้น กับผู้ที่ต้องการเรียนรู้ ดังนั้น โครงงานนี้จึงได้เกิดขึ้นเพื่อการดังกล่าว

เนื้อหาภายในเล่มจะแบ่งเป็น 3 ส่วนใหญ่ ๆ ส่วนแรก จะกล่าวถึงการใช้งาน 8048 SINGLE CHIP โดย RUN บน MPF-IP รวมไปถึงตัวอย่างในงานการทดลองซึ่งเนื้อหาจะอยู่ในส่วนที่ 1 ส่วนส่วนที่2 จะกล่าวถึงEMULATOR กับการสื่อสารข้อมูลอนุกรม โดยโปรแกรมที่จะ RUN บน 8048 สามารถที่จะเขียนบนเครื่องคอมพิวเตอร์ แล้วส่งโปรแกรมนั้นผ่านทาง 8251 ไปสู่พื้นที่หน่วยความจำของ MPF-IP เพื่อใช้ RUN ต่อไป ส่วนที่ 3 จะเป็นเอกสารอ้างอิงและรายละเอียดของ 8048 SINGLE CHIP และ 8251 CHIP SUPPORT

ในการจัดทำปริญญานิพนธ์ฉบับนี้ อาจมีสิ่งขาดตกบกพร่องอยู่บ้าง ผู้เขียนขออภัยมา ณ ที่นี้ด้วย

ผู้เขียน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทนำ

ในปัจจุบันการประยุกต์ใช้งานไมโครโปรเซสเซอร์ ในลักษณะงานควบคุม มีการใช้กันอย่างแพร่หลายมาก ซึ่งในด้านการตลาดของผู้ผลิตไมโครโปรเซสเซอร์ ก็พยายามที่จะพัฒนาคุณลักษณะ และความสามารถของไมโครโปรเซสเซอร์ของตนเองให้เหนือกว่าคู่แข่งขันรายอื่น ทำให้ขีดความสามารถของไมโครโปรเซสเซอร์นั้นสูงขึ้นเรื่อย ๆ การศึกษาการทำงานเฉพาะ CPU เบอร์ใดเบอร์หนึ่งในท้องตลาด อาจไม่เพียงพอสำหรับการใช้งาน เมื่อบอกถึงความคล่องตัว, ความเหมาะสม ฯลฯ อย่างเช่น การควบคุมในระบบเล็ก ๆ การใช้ไมโครโปรเซสเซอร์ดูจะเหมาะสมกว่า CPU ธรรมดา เนื่องจากมีการลดอุปกรณ์ภายนอกที่จำเป็น โดยการนำมารวมไว้ในตัวชิพเดียวกัน เช่น ส่วนที่เป็น Port, หน่วยความจำ, วงจรนับ ฯลฯ

สำหรับการศึกษาระบบในด้าน Soft ware ของ CPU แต่ละตัวนั้น ทางที่ดีที่สุด ก็คือ การศึกษาจาก Single board ซึ่งในการสร้าง Single board มาใช้งานแต่ละครั้งค่อนข้างจะยุ่งยาก ทั้งในด้านการออกแบบ Soft ware และ Hard ware และทั้งนี้การสร้าง Single board แต่ละครั้งก็สามารถใช้ศึกษาการทำงานของ CPU ได้เพียงตัวเดียวเท่านั้น ด้วยเหตุนี้ เพื่อเป็นการใช้งาน Single board ที่มีอยู่ให้ได้ประสิทธิภาพสูงสุด ซึ่งไม่เจาะจงว่าจะต้องใช้ศึกษาเฉพาะ CPU ที่มีอยู่บน board เท่านั้น แต่ยังสามารถใช้เขียนโปรแกรม เพื่อใช้ในการพัฒนาระบบให้ CPU ตัวอื่น ๆ ได้อีกด้วย โครงงานปริญญานิพนธ์นี้จึงถูกจัดทำขึ้น โดยใช้หลักการอินเตอร์เฟส ระหว่าง CPU และใช้สัญญาณควบคุมบางเส้น ช่วยในการส่งถ่ายข้อมูล อีกทั้งระบบการพัฒนาได้มีส่วนติดต่อกับ COMPUTER เพื่อนำโปรแกรมผ่านการ ASSEMBLES โดยวิธีทาง Soft ware บน PC ผ่าน I/O 232 SERIAL PORT ลงมา RUN บน MPF-IP ได้ ส่วนในรายละเอียดจะกล่าวในบทต่อ ๆ ไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. ส่วนของ 8048 INTERFACE แบ่งการพัฒนาได้ 2 ขั้นตอนคือ

1.1 การออกแบบโครงสร้างสร้างทาง HARD WARE

- โครงสร้างและการจัดหน้าที่การทำงานของขาต่าง ๆ ของ 8048 SINGLE CHIP

หน้าที่ของขาต่าง ๆ

Pin	Designation	Function
1	TO	Testable input using JUO and JNTO clock output once ENTO CLK instruction is executed. Restored to an input line by reset.
2	XTAL 1	Crystal input for internal oscillator. Also input for external clock source.
3	XTAL 2	Crystal input for internal oscillator.
4	RESET	Reset input for CPU. Active low.
5	SS	Single step input. Used in conjunction with ALE to single step through program execution. Active low.
6	INT	Interrupt input. Generates an interrupt if interrupt is enabled. Disabled after a reset. Active low.
7	EA	External Accexx. when taken high, forces all instruction fetches to come from external ROM.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Pin	Designation	Function
8	RD	Read strobe. Active during bus reads. Can enable data onto bus from external devices. Active low.
9	PSEN	Program store enable. Active during instruction fetches from external ROM. Active low.
10	WR	White strobe. Active during bus writes. Can strobe data into external devices from bus. Active low.
11	ALE	Address latch enable occurs once during each machine cycle. Also useful as a clock output. The falling edge of ALE strobes address into external RAM and ROM.
12-19	DB_0-DB_7	Bus port. Bidirectional port which can be written or read synchronously, using WR or RD Output can also be statically latched. Outputs eight low-order address bits during external instruction fetches. Receives addressed instruction during PSEN. Also passes eight address and data bits from external RAM [under control ALE, RD, and WR]
20	Vss	Circuit ground.
21-24	$P2_0$-$P2_3$	Lower four bits of quasi-bidirectional Port 2. Outputs upper four address bits during external ROM access and data for INS8243 port expander.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Pin	Designation	Function
25	PROG	Output strobe for INS8243 I/O expander.
26	V_{DD}	Provides an input for a standby power source for the internal RAM. When a standby power source is connected. V_{cc} provides a battery chargingpath. The amount of internal RAM connected to V_{DD} is a programmable option.
27-34	$P1_0$-$P1_7$	Port 1, quasi-bidirectional.
35-38	$P2_4$-$P2_7$	Upper four bits of quasi-bidirectional Port 2.
39	T1	Testable input using JT1 and JNT1 instructions. Event counter input once STRT CNT instruction is executed.
40	V_{cc}	Main 48-Series power source [+5V]

1.2 การนำไปใช้งาน เช่นเดียวกันการใช้งานไมโครโปรเซสเซอร์แบบ SINGLE BOARD ทั่วไป คือ สามารถต่อใช้งานจาก JUMPER J1 [Z80 COMPATIBLE] หรือ J2 [8048 SINGLE CHIP] หรือ J3 [8255] ก็ได้

การต่อใช้งานเป็นไปได้อย่างอิสระ ตัวอย่างการใช้งานจะแบ่งเป็นการทดลองในลักษณะต่าง ๆ ซึ่งจะกล่าวถึงในบทต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การออกแบบ 8048 INTERFACE

ด้วยจุดประสงค์เพื่อต้องการใช้อุปกรณ์ภายนอกเพิ่มเติมให้น้อยชิ้นที่สุด ดังนั้นในการสร้างชุดพัฒนาระบบนี้ จะใช้ชุดไมโครโปรเฟสเซอร์ รุ่น MPF-1 ซึ่งเป็นชิงเกิ้ลบอร์ดไมโครคอมพิวเตอร์ของมัลติเทค มาเป็นอุปกรณ์หลักในการป้อนโปรแกรมคำสั่งและข้อมูลของ 8048 (ต่อไปนี้จะเรียกเพียงเบอร์เดียวคือ 8048) เก็บเข้าไว้ในหน่วยความจำส่วนที่เป็น RAM ของตัวมันเอง จากนั้นเมื่อจะให้ 8048 ทำงานก็สามารถทำได้โดยการที่ให้ 8048 เข้าไปแทนที่ Z80CPU ใน MPF-1 โดยวิธีการนี้ก็จะทำให้สามารถใช้อุปกรณ์อื่น ๆ ได้ทั้งหมด จากหลักการดังกล่าวก็จะ ทำให้ได้ชุดพัฒนาระบบ 8048 ที่ใช้อุปกรณ์เพิ่มเติมจากเดิมอีกเพียงเล็กน้อยเท่านั้น ซึ่งทำให้สามารถได้ชุดพัฒนาระบบ 8048 ที่มีคุณลักษณะดังนี้ คือ

1) มีส่วนความจำโปรแกรมภายนอก [external program memory] ที่เป็น RAM ขนาด 2K และเพิ่มได้จนเต็ม 4K

2) มี I/O พอร์ทซึ่งต่ออยู่กับคีย์บอร์ด แผงแสดงตัวเลขขนาด 6 หลัก และ 1 อินพุท 1 เอาท์พุทสำหรับเทปคาสเซทท์

3) มี เพอริเฟอรัลอื่นอีก 2 ตัวคือ Z80PIO และ Z80CTC

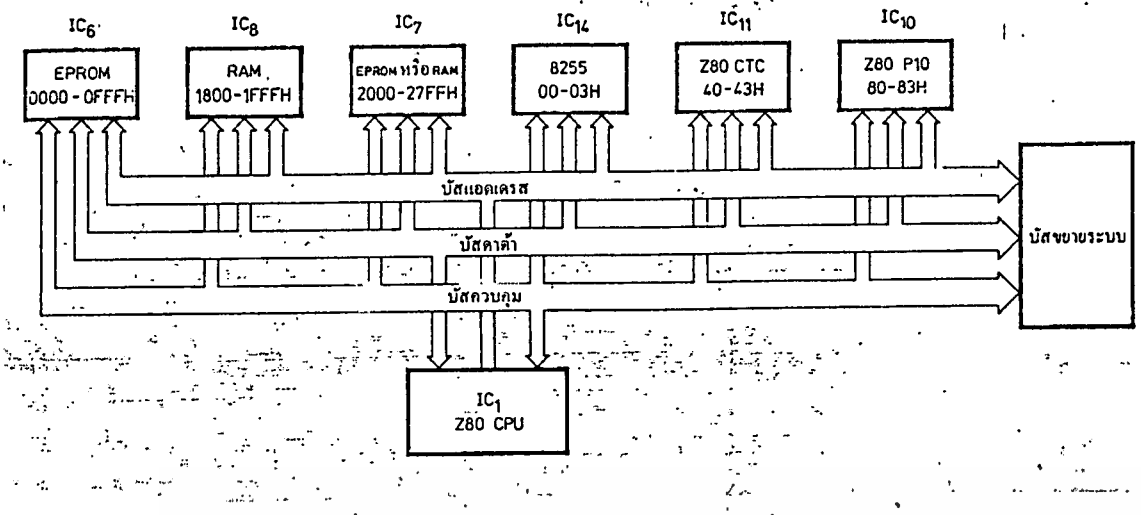
4) ชิงเกิ้ลสเตปโดยใช้ฮาร์ดแวร์

5) สามารถกระทำคำสั่งจากโปรแกรมหน่วยความจำภายในได้

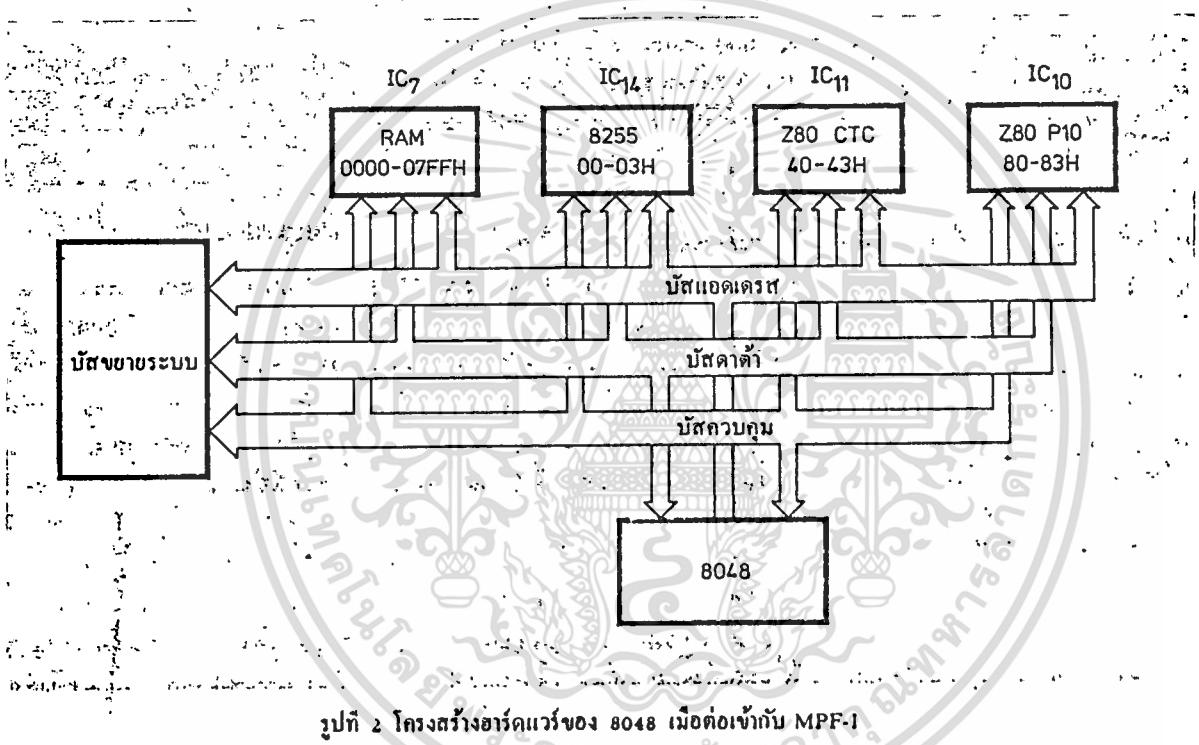
โครงสร้างทางฮาร์ดแวร์ภายในของ MPF-1

ก่อนอื่นมาทำความรู้จักกับโครงสร้างฮาร์ดแวร์ที่สำคัญของไมโครโปรเฟสเซอร์ MPF-1 ก่อน จากรูปที่ 1 ประกอบไปด้วย Z80CPU [IC_1] EPROM 2732 ซึ่งเป็นมอร์นิเตอร์ [IC_9] และ RAM 2 ตัว ตัวละ 2 กิโลไบท์ใช้เบอร์ 6116 [IC_7 และ IC_8 สำหรับ IC_7 นั้นสามารถใส่ EPROM เบอร์ 2732 ได้] อุปกรณ์สนับสนุนประกอบไปด้วย 8255 [IC_3] ซึ่งทำหน้าที่เป็นตัวสแกนคีย์บอร์ด, ดีสเพลย์, อินพุท, เอาท์พุทสำหรับเทปคาสเซทท์, Z80PIO[IC_{10}] และ Z80CTC [IC_{11}]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 1 โครงสร้างฮาร์ดแวร์ของไมโครโปรเซสเซอร์ MPF-I



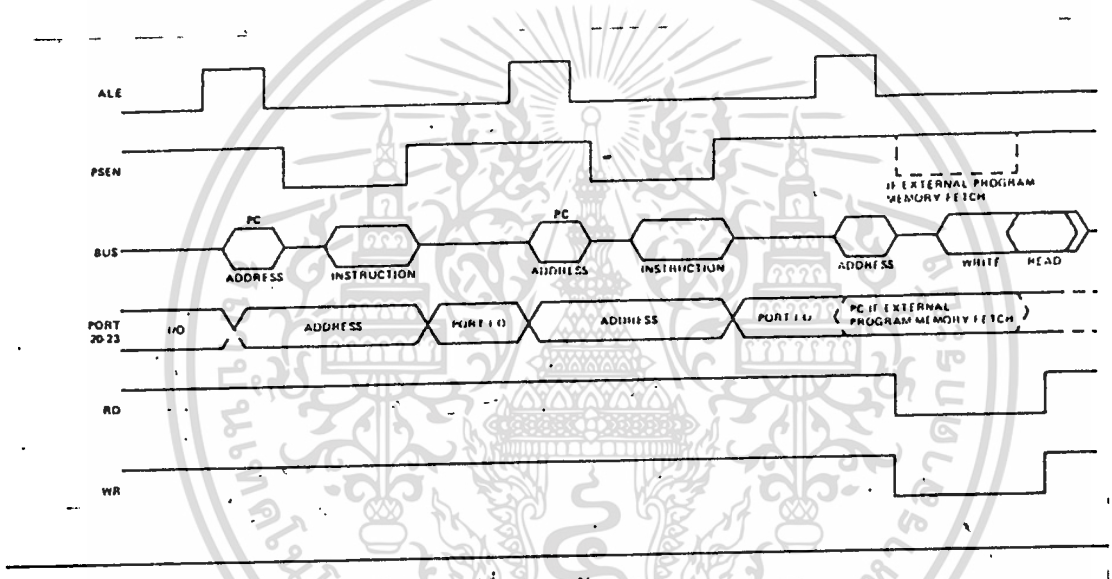
รูปที่ 2 โครงสร้างฮาร์ดแวร์ของ 8048 เมื่อต่อเข้ากับ MPF-I

การจัดตำแหน่งของหน่วยความจำต่าง ๆ มีลักษณะการจัดดังนี้ คือ IC_6 ตำแหน่ง 0000 ถึง 0FFF IC_7 ตำแหน่ง2000-2FFF และ IC_8 ตำแหน่ง 1800-1FFF สำหรับในกรณีของชุดพัฒนาระบบ 8048 นี้จะใช้ IC_7 เป็นหน่วยความจำภายนอกซึ่งถ้าใช้ RAM จะสามารถใช้กับ RAM ขนาด 2 กิโลไบท์เท่านั้น (ใช้กับ RAM เบอร์ 6116) ส่วนอีก 2K ต่อไป ถ้าต้องการเพิ่มก็สามารถต่อขยายได้ภายนอกที่ตำแหน่ง 2800 ถึง 2FFF ได้ สำหรับอุปกรณ์สนับสนุนอื่น ๆ นั้น เช่น IC_{14} [8255] เป็น I/O ที่ตำแหน่ง 00-03 IC_{10} [Z80PIO] อยู่ที่ตำแหน่ง 80-83 และ IC_{11} [Z80CTC] อยู่ที่ตำแหน่ง 40-43

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ฮาร์ดแวร์ของชุดพัฒนาระบบ 8048

ในการที่จะนำเอา MPF-1 มาทำเป็นชุดพัฒนาระบบนี้จะต้องมีการจัดแอดเดรสใหม่ โดยการที่ให้ 8048 มองให้เห็นระบบของตัวเองเป็นดังรูปที่ 2 วิธีการก็คือเมื่อต้องการให้ 8048 ทำงานจะต้องให้สัญญาณขอใช้บัสไปที่ Z80CPU [bus request] จากนั้นเมื่อ Z80CPU ตอบสนองการใช้บัส [bus acknowledge] แล้วคาต้าบัส, แอดเดรสบัสและสายควบคุมบางเส้นของ Z80CPU จะลอยตัวออกจากระบบ สัญญาณตอบสนองการใช้บัสนี้จะไปเปิดให้ 8048 เข้าควบคุมระบบบัสต่าง ๆ ภายใน MPF-1



รูปที่ 3 แผนผังเวลาของ 8048

การแลทช์แอดเดรส

จากการศึกษาถึงเวลาของสัญญาณต่าง ๆ จากรูปที่ 3 ทำให้ทราบว่าดาต้าบัส DB_0 ถึง DB_7 มีการมัลติเพลกซ์ระหว่างแอดเดรสและดาต้า ดังนั้นจะต้องมีการแลทช์แอดเดรสก่อน (ช่วงเวลาของแอดเดรสมาก่อนดาต้า) การแลทช์แอดเดรสนั้น เราจะใช้สัญญาณในช่วงขอบลบของสัญญาณ ALE (ซึ่งเป็นสัญญาณ addresslatch enable) ซึ่งในที่นี้จะใช้ 74LS374 มาทำหน้าที่เป็นแอดเดรสแลทช์ [74LS374]เป็น tri-state octal D-type flip-flop] แต่เนื่องจากทำงานที่ขอบบวก ดังนั้นจึงจำเป็นที่จะต้องใช้อินเวอร์เตอร์อีก 1 ตัว มาเป็นตัวกลับขอบสัญญาณ เอาท์พุทของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แลทช์นี้จะต้องเป็นไตรสเตท ซึ่งจะควบคุมด้วย BUSAK โดยการต่อที่ขาดิสเอเบิ้ลของ 74LS374 (รูปที่ 4) ส่วนแอดเดรส A_8-A_{11} คือที่ขา P_{20}-P_{23} ของ 8048 มีสถานะของแอดเดรสจะคงอยู่ตลอดช่วงเวลาที่มีการเฟทช์คำสั่ง) จึงเพียงแต่ผ่านไตรสเตทเพียง 1 ตัว (74LS244) โดยใช้สัญญาณจาก BUSAK มาควบคุม และเนื่องจากการจัดระบบการแอดเดรสของ 8048 นั้น อยู่ใน RAM ของ MPF-1 ตำแหน่ง 2000 ถึง 2FFF ดังนั้นในส่วน 4 บิทบนของหน่วยความจำ [A_{12}-A_{15}] จะต้องมีการจัดแอดเดรสใหม่เป็น A_{15} = 0, A_{14} = 0, A_{13} = 1 และ A_{12} = 0 (คือเลข 2 ฐานสิบหกนั่นเอง) ซึ่งทั้งหมดก็จะต้องผ่านไตรสเตทโดยใช้สัญญาณจาก BUSAK มาควบคุมเช่นกัน

สัญญาณคาด้า

หลังจากที่ช่วงเวลาของการแลทช์แอดเดรสผ่านไปแล้ว ก็เป็นช่วงเวลาของการอ่านคำสั่งหรือการอ่านและเขียนข้อมูลที่อยู่ใน RAM ภายนอก ซึ่งรวมทั้งตำแหน่งของ I/O ที่เป็นลักษณะของ memory map I/O ด้วย ในช่วงเวลาดังกล่าวนี้จะเห็นได้ว่าสัญญาณ PSEN เป็น low เมื่ออยู่ในช่วงการเฟทช์คำสั่งและสัญญาณ RD หรือ WR เป็น low เมื่ออยู่ในช่วงการอ่านหรือการเขียนข้อมูลภายนอกดังนั้นจึงใช้ 74LS245 ซึ่งเป็น bidirectional bustransceiver มาเป็นตัวส่งผ่านข้อมูลระหว่าง MPF-1 และ 8048 สัญญาณดิสเอเบิ้ลของ 74LS245 ก็ได้จากสัญญาณทั้งสามดังกล่าว คือ PSEN, RD และ WR อีกทั้งยังต้องขึ้นอยู่กับสัญญาณ BUSAK ของ MPF-1 ด้วย ส่วนทิศทางการส่งถ่ายข้อมูลก็ขึ้นอยู่กับว่าช่วงนั้นเป็นช่วงการอ่านหรือเขียน ถ้าหากว่าเป็นช่วงการอ่านข้อมูล ข้อมูลจะต้องถ่ายเทจาก MPF-1 ไปสู่ 8048 และถ้าเป็นช่วงการเขียนข้อมูล ข้อมูลจะต้องถ่ายจาก 8048 สู่อุปกรณ์หรือหน่วยความจำภายนอก ดังนั้นจึงต่อสัญญาณ WR เข้าที่ขาควบคุมทิศทางของ 74LS245 (ขาที่ 1)

สัญญาณควบคุม

สายสัญญาณต่าง ๆ ที่จำเป็นต้องต่อร่วมระหว่าง 8048 กับ MPF-1 ก็คือสัญญาณอ่าน [RD], สัญญาณเขียน [WR], สัญญาณ program store enable [PSEN] ของ 8048 และ IORQ, WR, RD, MREQ ของทางด้าน MPF-1 ก่อนอื่นต้องมาทำความเข้าใจกับระบบสัญญาณของ 8048 สักเล็ก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

น้อย คือ สัญญาณ PSEN จะเกิดขึ้นเมื่อต้องการเฟทช์คำสั่งจากโปรแกรมหน่วยความจำภายนอกเท่านั้น ส่วน RD และ WR เกิดขึ้นเมื่อต้องการอ่านและเขียนข้อมูลกับหน่วยความจำ (หรือ I/O) ที่เพิ่มขึ้นมาจากภายนอก (ซึ่งมีเพียง 2 คำสั่งเท่านั้น ที่กระทำกับหน่วยความจำชุดนี้ได้ คือ MOVX A, @R_r และ MOVX @R_r,A) ส่วนสายสัญญาณของ MPF-1 นั้นมีอยู่ 4 สาย คือ IORQ, WR, RD และ MREQ เมื่อต้องการเฟทช์คำสั่งก็จะต้องให้ MERQ กระทำร่วมกับ RD ดังนั้นในช่วงการเฟทช์คำสั่งของ 8048 จึงสามารถที่จะต่อ MREQ เข้ากับ PSEN ได้โดยตรงโดยผ่านไตรสเตท 1 ตัว ส่วนสัญญาณ RD นั้นมีอยู่แล้วทั้งนี้เนื่องจากว่าโครงสร้างทางฮาร์ดแวร์ของ MPF-1 ที่ IC_7 ซึ่งเป็นส่วนความจำที่นำมาใช้งานนั้นมีสายควบคุมการอ่านและเขียนข้อมูลต่ออยู่กับ WR ซึ่งก็หมายความว่าถ้าหาก WR = 1 ก็คือสภาวะของการอ่านข้อมูลเมื่อย้อนกลับมาดูแผนผังเวลาของ 8048 ในรูปที่ 3 จะเห็นว่าขณะที่ทำการเฟทช์คำสั่งนั้นจะไม่มีสัญญาณ WR = 0 เกิดขึ้นเลย ดังนั้นการต่อ PSEN เข้ากับ MERQ เพียงสายเดียวเพื่อทำการเฟทช์ข้อมูลนั้นเพียงพอแล้ว

สำหรับการอ่านและเขียนข้อมูลในหน่วยความจำภายนอกของ 8048 ในชุดพัฒนาระบบนี้จะใช้เป็น I/O ของ MPF-1 ทั้งหมด โดยเมื่อต้องการอ่านก็จะต้องทำให้ RD และ IORQ ของ MPF-1 แอคตีฟ [active low] และเมื่อต้องการเขียนก็ให้ WR และ IORQ แอคตีฟ [active low] ดังนั้นจึงจะสามารถที่จะต่อ RD และ WR ของ 8048 เข้ากับ RD และ WR ของ MPF-1 ได้โดยตรงโดยการผ่านไตรสเตทเพียงอย่างละตัว ส่วนสายสัญญาณ IORQ นั้นเกิดจากการ AND ระหว่าง WR และ RD จากนั้นก็ผ่านไตรสเตทอีก 1 ตัว

สายสัญญาณอื่น ๆ ที่เหลือ

สายสัญญาณควบคุมของ 8048 ที่ไม่เกี่ยวข้องกับ MPF-1 อีกสายหนึ่งก็คือ EA ซึ่งเป็น external access input ซึ่งถ้าหากว่าเป็น "1" จะกระทำคำสั่งจากโปรแกรมหน่วยความจำภายใน (รวมทั้งโปรแกรมหน่วยความจำที่ต่อขยายจากภายนอกเพื่อให้ครบ 4K)

เมื่อต้องการตรวจสอบการทำงานและขั้นตอนของโปรแกรมที่ได้เขียนไว้ก็สามารถทำซิงเกิ้ลสเตปได้โดยวิธีฮาร์ดแวร์ซึ่งใน '8048 นั้นมีขาซิงเกิ้ลสเตป [SS] ไว้ให้อยู่แล้วจึงเป็นการสะดวกในการตรวจสอบการทำงานของโปรแกรมได้ จาก SHEET 2 OF 3 เมื่อต้องการทำซิ้งเกิ้ลสเตปก็ให้ S_4 อยู่ในตำแหน่งเปิดจากนั้นกด S_3 ซึ่งเป็นการที่จะให้ทำทีละคำสั่งการตรวจสอบซิงเกิ้ลสเตปสามารถตรวจสอบจาก DISP1 - DISP4 ซึ่งจะแสดงตำแหน่งของคำสั่งที่จะกระทำต่อไปในระบบเลขฐานสอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สายสัญญาณอื่น ๆ ที่จำเป็น คือ

1) X'TAL1 และ X'TAL2 เป็นสายที่ต่อเข้ากับ X'TAL

2) RESET เป็นสายสัญญาณเพื่อรีเซท

3) T_0, T_1, P_{10}-P_{17} และ P_{20}-P_{27} ต่อเพื่อใช้งาน

4) EA ปกติเป็น 0V

008511

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การทดลองที่ 1

วัตถุประสงค์

1. เพื่อให้เข้าใจการทำงานของระบบ MICRO COMPUTER
2. เพื่อให้เข้าใจการออกแบบระบบ MICRO COMPUTER
3. เพื่อให้เข้าใจการ DECODE MEMORY และ PORT

ทฤษฎี

ในระบบ COMPUTER นั้นจะประกอบด้วยส่วนใหญ่ ๆ 4 ส่วนด้วยกันคือ

1. หน่วยควบคุม [CONTROL UNIT]
2. หน่วยความจำ [MEMORY UNIT]
3. หน่วยคำนวณ [ARITEMETIC UNIT]

ในระบบ MICRO COMPUTER นั้นเราอาจจะรวม หน่วยควบคุมและหน่วยคำนวณไว้ด้วยกันก็คือ ส่วนของ CPU ซึ่งก็จะคงเหลือส่วนสำคัญ 2 ส่วนคือ หน่วยความจำและ I/O หรือ PORT

หน่วยความจำ

หน่วยความจำหรือ MEMORY UNIT นี้ก็คือ ส่วนเก็บข้อมูลหรือส่วนเก็บตัวโปรแกรม อาจจะอยู่ในรูปของ IC ซึ่งอาจจะแบ่งออกได้

1) หน่วยความจำที่ข้อมูลยังคงอยู่แม้จะไม่มีไฟเลี้ยง [ROM]

ROM [READ ONLY MEMORY] จะเป็น IC ประเภทหนึ่งซึ่งสามารถเก็บข้อมูลได้ถึงแม้จะไม่มีไฟเลี้ยงก็ตาม โดยข้อมูลใน ROM นี้จะเป็นข้อมูลที่เขียนข้อมูลมาแล้วจากโรงงานผู้ผลิตไม่สามารถนำมาแก้ไขเปลี่ยนใหม่ได้อีก โดยมากจะเป็นข้อมูลตัวโปรแกรม MONITOR PROGRAM

ROM [PROGRAMMABLE READ-ONLY] MEMORY เป็น IC ที่ผู้ใช้สามารพนำมาเขียนข้อมูลลงไปได้เองโดยใช้ HIGH VOLTAGE ช่วยในการเขียนข้อมูลลงไปเมื่อเขียนแล้วจะไม่สามารถลบได้อีก

EPROM [ERASABLE PROGRAMMABLE ROM] เป็น IC ซึ่งสามารถเขียนข้อมูลลงไปได้เช่นเดียวกับ PROM แต่สามารถลบเขียนใหม่ได้ใช้แสงอุลตร้าไวโอเลตช่วยลบข้อมูล

RAROM [RLRCTRICALLY ALTERABLE ROM] ข้อมูลจะเขียนเข้าไปได้เหมือน EPROM แต่ลบได้ด้วยไฟฟ้าไม่จำเป็นต้องใช้แสง เช่นเดียวกับ EPROM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2) หน่วยความจำที่ข้อมูลหายถ้าไม่มีไฟเลี้ยง [RAM]

STATIC RAM เป็นหน่วยความจำที่สามารถเขียนและอ่านข้อมูลได้โดย CPU ข้อมูลจะยังคงอยู่ถ้ายังมีไฟเลี้ยงอยู่ไม่จำเป็นต้องมีวงจรพิเศษ ๆ ใช้ร่วมในการใช้งาน

DYNAMIC RAM เป็นหน่วยความจำเช่นเดียวกับ STATIC RAM แต่จำเป็นต้องมีวงจรพิเศษเพิ่มเติมเพื่อให้ข้อมูลยังคงอยู่ คือต้องมีการ REFRESH เพื่อรักษาข้อมูลให้คงอยู่เพราะในหลักการทำงานของการเก็บประจุไฟฟ้า

หน่วยรับและส่งสัญญาณ

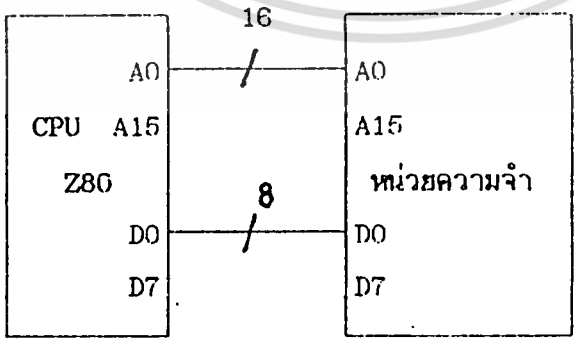
ในที่นี้เราจะพูดถึงเฉพาะหน่วยรับและส่งสัญญาณที่อยู่ในรูปที่อยู่ในรูปของ IC โดยมองเป็น INPUT และ OUTPUT PORT

1. INPUT และ OUTPUT PORT ซึ่งเป็น TTL ธรรมดาเช่น IC เบอร์ 74LS244, 74LS245, 74LS374 เป็นต้น

2. INPUT และ OUTPUT PORT ในรูป IC LSI [LARGE SCALE INTEGRADE CIRCUIT] เช่น IC เบอร์ 8255, 8251, 8253 หรือ Z80 PIO เป็นต้น

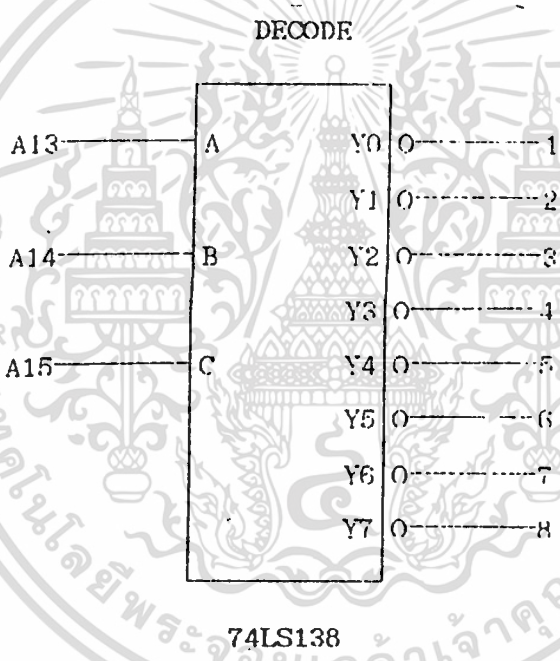
การ DECODE MEMORY และ PORT

ในระบบ Z80 นั้นตัว CPU จะมีขา ADDRESS 16 เส้นคือ A0 ถึง A15 ก็คือมันสามารถมองหน่วยความจำได้ เท่ากับ 2 ยกกำลัง 16 คือ 65535 ตำแหน่งและก็เนื่องจากมีขา DATA ของ CPU 8 ขา ก็จะมองหน่วยความจำได้ 65535 ตำแหน่ง ตำแหน่งละ 8 BIT หรือ 1 BYTE



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จะเห็นว่าเมื่อ CPU มองได้ 65535 ตำแหน่งหรือต่อไปเราเรียกว่า 64KBYTE [1KBYTE = 1024] นั้น เราก็จะสามารถต่อให้ CPU ทำงานร่วมกับหน่วยความจำได้ถึง 64 BYTE แต่ในความเป็นจริงแล้วเราไม่จำเป็นต้องการหน่วยความจำมากมายขนาดนั้นในงานจริง ๆ รวมทั้งก็ยังไม่มีหน่วยความจำตัวเดียวในรูป IC ที่มีความจะมากขนาดนั้นได้ และในการใช้งานจริงเราต้องใช้หน่วยความจำหลายประเภท เช่น ROM กับ RAM ดังนั้นเราจึงจำเป็นที่จะต้องแบ่งการมองหน่วยความจำของ CPU ออกเป็นหลายช่วง คือการ DECODE เช่นเราต้องการใช้ CPU ของหน่วยความจำออกเป็น 8 หน่วยความจำเราก็จำเป็นต้องหาวงจรเพิ่มเข้ามาให้ CPU มองหน่วยความจำเป็นระยะระยะ ดังรูป

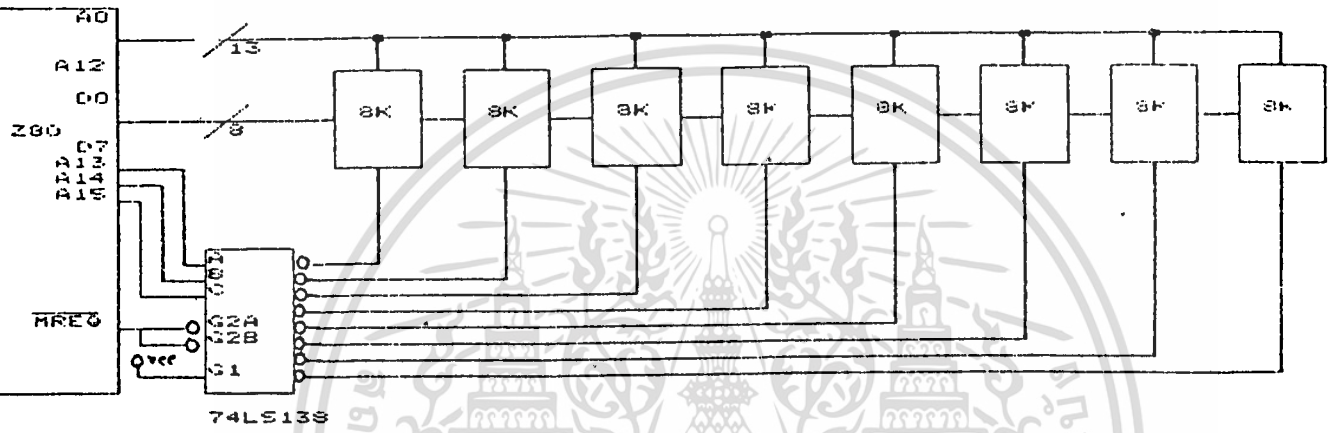


จากรูปจะเป็นวงจร DECODE โดยใช้ IC เบอร์ 74LS138 โดยเราจะใช้ขา A13, A14, A15 เข้าขา A,B,C ซึ่งก็จะทำให้ได้ตำแหน่งเท่ากับ 2 ยกกำลัง 3 คือ 8 ตำแหน่งเรายังคงเหลือขา ADDRESSจาก CPU อยู่คือ A0 ถึง A12 คือเท่ากับ 13 ขาก็จะเท่ากับ 2 ยกกำลัง 13 = 8KBYTE ดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

CPU จะมองเห็นหน่วยความเป็นช่วย ๆ คือ 8KBYTE ดังนั้นเราจึงสามารถนำหน่วยความจำขนาดไม่มากกว่า 8KBYTE เป็นตัว ๆ มาใส่ช่วยในวงจรนี้ได้อาจจะเป็น RAM หรือ EPROM ก็ได้

และก็ในกรณีเดียวกันถ้าเป็น PORT เนื่องจาก Z80 CPU จะมอง PORT ได้โดยใช้ ADDRESS 8 ขา คือ A0 ถึง A7 คือ ยกกำลัง 2 = 256 PORT ถ้าสมมติว่าเราจำเป็นต้องต่อ PORT ใช้งาน 8 ตัว เราก็สามารถต่อได้โดยต่อวงจรนำขา ADDRESS A7, A6, A5 มาใช้งาน ดังรูป



จากรูปวงจรที่ 1 และ 2 จะเห็นว่า CPU ใช้ ADDRESS ขาเดียวกันสำหรับ PORT และ MEMORY เพื่อให้อุปกรณ์ติดต่อรู้ว่า CPU ต้องการติดต่อ PORT หรือ MEMORY CPU จึงมีขาใช้งานในเรื่องนี้อีก 2 เส้น คือ ขา IORQ, MREQ เพื่อบอกอุปกรณ์ที่ติดต่ออยู่ว่าเป็นการใช้ ADDRESS ของ MEMORY หรือ PORT อยู่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SOLID-STATE RELAY

วัตถุประสงค์

1. เพื่อให้เข้าใจการทำงานของ SOLID-STATE RELAY
2. การใช้งานของ SOLID-STATE RELAY
3. ข้อดีข้อเสียของ SOLID-STATE RELAY
4. การต่อวงจร SOLID-STATE RELAY

ทฤษฎี

SOLID-STATE RELAY หรือเรียกสั้น ๆ ได้ว่า SSR ได้เข้ามาใช้งานในการเป็นอุปกรณ์ติดต่อทางไฟฟ้าโดยเฉพาะแล้วในระบบไฟฟ้าที่ใช้การทำงานร่วมกับ MICORCOMPUTER แล้วจะเข้ามาใช้แทน RELAY ธรรมดา ๆโดยใช้หลักการทำงานของอุปกรณ์ที่เป็นเซมิคอนดัคเตอร์แทนระบบการเคลื่อนไหวทางกลไกในการตัดต่อวงจร เราอาจจะพิจารณาข้อดีของ SSR ได้ดังนี้

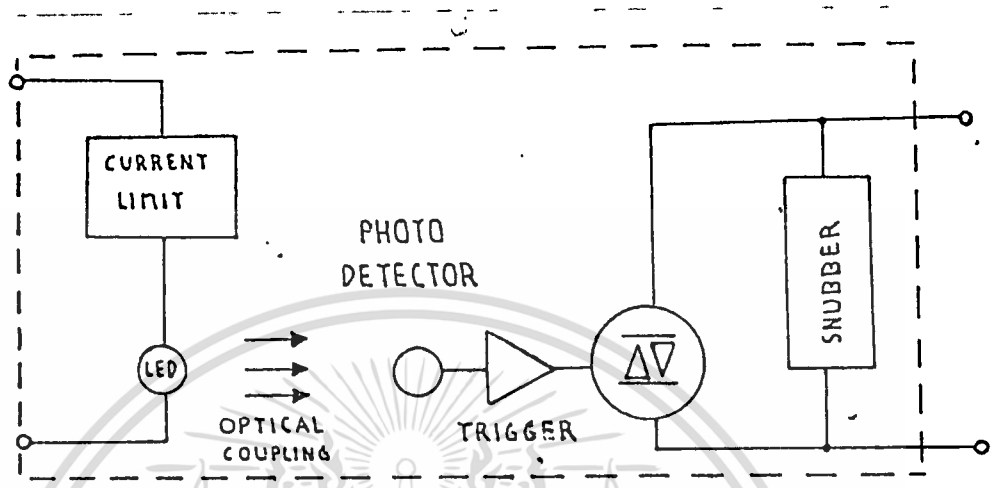
ข้อดีของ SSR

1. การทำงานจะต่อวงจรที่ VOLTAGE ขณะเป็นศูนย์หรือใกล้จุดศูนย์ VOLT
2. มีอายุการทำงานที่ยาวนานมาก
3. ไม่มีเสียงรบกวนเกิดขึ้นในเวลาทำงาน
4. สามารถต่อเข้ากับระบบ MCROPROCESSOR ได้โดยง่าย
5. สามารถตัดต่อวงจรได้อย่างรวดเร็ว
6. ไม่มีส่วนเคลื่อนไหวทางกลในการทำงาน
7. ไม่เกิดอาการ BOUNCE ที่หน้าสัมผัส
8. ไม่ก่อให้เกิดสัญญาณรบกวน ต่อระบบไฟฟ้า

ข้อเสียของ SSR

1. ไม่สามารใช้กับวงจรที่มี VOLTAGE สูงมาก ๆได้
2. ราคาแพง
3. ไม่สามารถใช้ SSR สำหรับ AC หรือ DC ได้ในตัวเดียวกัน
4. เวลาใช้งานจะเกิดความร้อนขึ้นที่ SSR นั้นๆ จำเป็นต้องใช้ HEATSINK

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



AC SOLID-STATE RELAY [SSR]

เราพอแบ่งการทำงานของ SSR ได้เป็น 2 ภาคใหญ่ ๆ

1. ภาค INPUT
2. ภาค OUTPUT

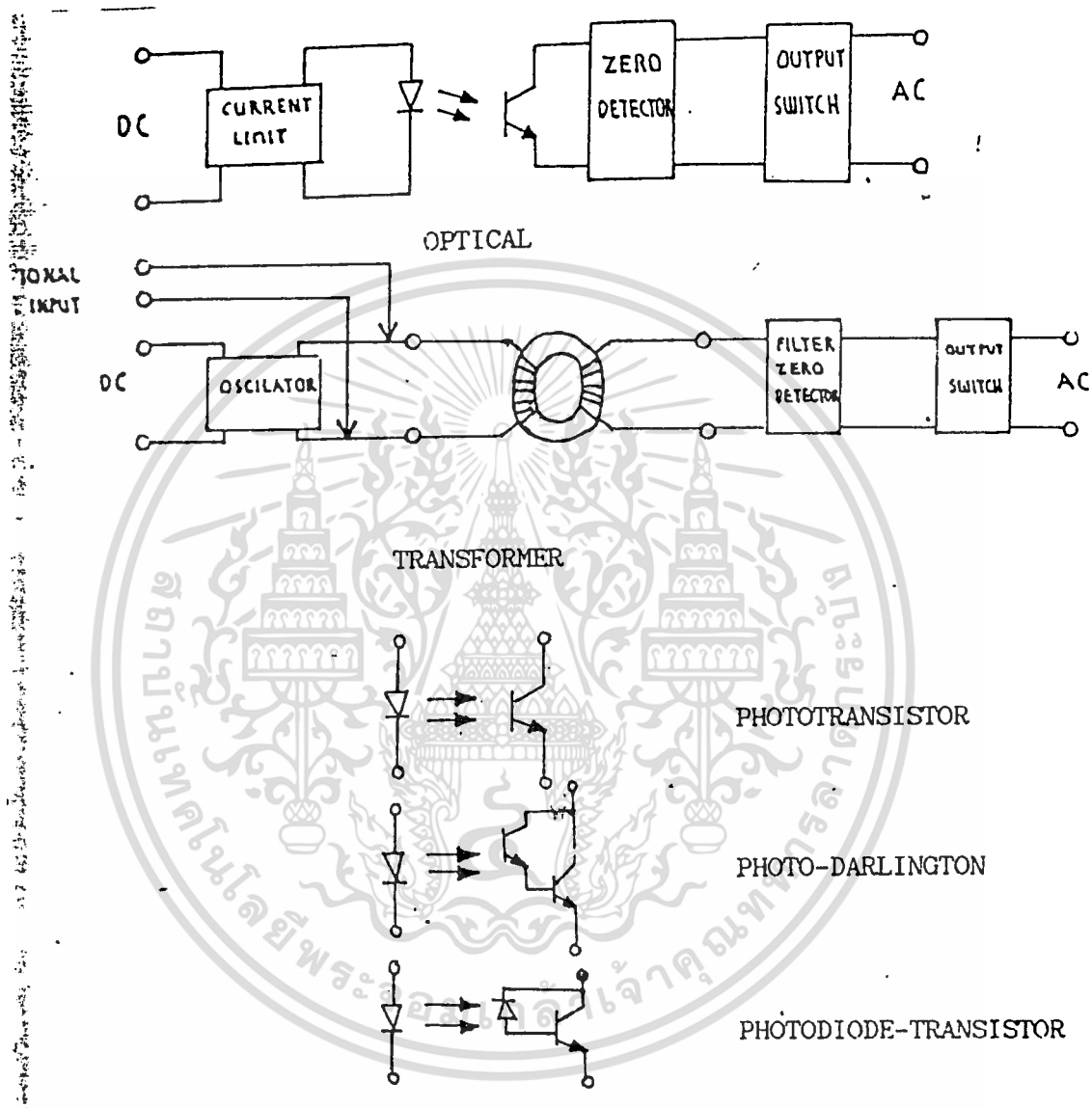
1. ภาค INPUT จะเป็นภาครับสัญญาณจากการควบคุมมาทำงานโดยเป็นการส่งผ่านสัญญาณไปควบคุมอุปกรณ์ติดต่อ ซึ่งจะแยกกันในทางไฟฟ้า โดยที่มีใช้กันอยู่พอแบ่งออกได้เป็น 2 ลักษณะ

1.1 ใช้อุปกรณ์ทาง OPTICAL [PHOTO] เช่นอาจจะใช้PHOTO TRANSITOR, PHOTO-DARLINGTON, PHOTODIODE, PHOTO-SCR หรือ PHOTO RESTORE เป็นต้นตัวอุปกรณ์จะเปลี่ยนสัญญาณไฟฟ้าที่ส่งมาให้อยู่ในรูปของแสงไปควบคุมส่วน OUTPUT อีกทีหนึ่งซึ่งจะทำให้เป็นการตัดขาดระหว่าง INPUT และ OUTPUT ทางไฟฟ้าจริง ๆ

1.2 ใช้อุปกรณ์ TRANSFORMER โดยจะเป็น FERRITE-CORED TOROIDAL โดยจะทำงานร่วมกันระหว่าง INPUT และ OSCILLATOR โดยมีการ OSCILLATOR ความถี่สูง 50 KILOHERTZ ถึง 10 MEGAHERTZ ในการส่งสัญญาณจาก INPUT ไปยัง OUTPUT

2. ภาค OUTPUT เป็นส่วนรับสัญญาณจาก ภาค INPUT ซึ่งอาจจะอยู่ในรูปของแสงมาทำงานโดยต่อวงจรหรือตัดวงจรโดยอุปกรณ์นี้อาจจะเป็น TRANSISTOR, SCR หรือ TRIAC ก็ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SERO SWITCHING

อีกส่วนหนึ่งที่มีความสำคัญสำหรับการทำงานของ SSR ที่ใช้กับ OUTPUT ที่เป็น AC โดยปกติในการทำงานของ SW ธรรมดาต่าง ๆ อาจจะมีการทำงานที่ไม่สมควรเกิดคือ เปิดวงจรในขณะที่สัญญาณของไฟ AC อาจจะอยู่ในค่าสูงสุด ซึ่งทำให้เกิดกระแสและสัญญาณรบกวนที่สูงมาก โดยเฉพาะถ้าต่อร่วมอยู่กับวงจรทาง COMPUTER แล้ว ก็อาจจะทำให้เครื่องเกิดอาการเสียหรือทำงานผิดปกติก็ได้ ฉะนั้น ZERO SWITCHING ก็คือเป็นการทำงานให้เปิดวงจรในขณะที่สัญญาณ AC อยู่ในระยะ 0V แต่ในการใช้งานจริงจะมีค่ามากกว่า 0V เล็กน้อยเพื่อให้อุปกรณ์เซมิคอนดัคเตอร์ ที่ต่ออยู่สามารถทำงานเปิดวงจรได้ เช่น ตัวอย่างเราต้องการให้วงจรทำงานเปิดวงจรที่ 15V จาก 220V ก็จะเป็นมุมทางสัญญาณ AC 2.77 องศา ตามสูตร

$$\theta = \text{SIN} \frac{Z_{sw} \cdot max}{\text{Line } V_{RMS}\sqrt{z}}$$

$$= \text{SIN} \frac{15}{220 \times \sqrt{1.41}}$$

$$= 2.77$$

หรือในทางกลับกันเราอาจจะคำนวณระยะเวลาหลังจาก ที่เกิดขึ้นแล้วนานเท่าไรจึงจะเปิดวงจร

$$T = \frac{\frac{1}{2} \text{ cyc. ms}}{\frac{1}{2} \text{ cyc. deg}} \times \theta$$

$$= \frac{\frac{1}{2}\left(\frac{1}{50}\right)}{\frac{1}{2}\ 360} \times 2.77$$

$$= 0.153 \text{ ms}$$

ตัวอย่างวงจร SSR OUTPUT AC 220V

เราสามารถสั่งทางโปรแกรมให้ SSR ทำงานได้โดยส่ง OUTPUT จาก PORT 8255 โดยให้ LOGIT '0' SSR จะทำงานให้ SSR ไม่ทำงานโดยส่ง LOGIT '1' ออกไปที่ PORT

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PRINTER

วัตถุประสงค์

1. เพื่อให้เข้าใจสัญญาณและส่วนประกอบของ PRINTER
2. ต่อ SINGLE BOARD INTERFACE กับ PRINTER
3. เพื่อให้เข้าใจการ INTERRUPT จากสัญญาณภายนอก
4. เพื่อเป็นแนวทางในการประยุกต์ใช้งานต่อไป

ทฤษฎี

PRINTER นับว่าเป็นสิ่งที่สำคัญที่เดียวที่ใช้เป็นตัวพิมพ์ข่าวสารข้อมูลที่เราต้องการไว้เป็นหลักฐานได้อย่างถาวร แบ่งได้ 3 แบบใหญ่ ๆ ตามลักษณะการใช้งานคือ

1) แบบเดซี่วีล [DAISY WHEEL]
2) แบบดอทแมททริกซ์ [DOT MATRIX]
3) แบบใช้การพลอท [PLOTTER]

DAISY WHEEL ก็คือ เครื่องพิมพ์ดีดแบบอีเลคทรอนิคส์นั่นเองเหมาะสำหรับงานแสดงผลในรูปตัวอักษรเป็นส่วนใหญ่

DOT MATRIX เหมาะสำหรับพิมพ์ข้อมูล ทั้งในรูปตัวอักษรหรือแผนภูมิหรือกราฟ การพิมพ์ใช้เป็นชุดของเข็มยิง ซึ่งเข็มแต่ละตัวจะควบคุมโดย COMPUTER อย่างอิสระ แบ่งได้เป็น

- แบบใช้ความร้อน [DAISY WHEEL]
- แบบใช้การตอก [IMPACT]

PLOTTER ใช้กับงานแผนภูมิและกราฟเป็นส่วนใหญ่ โดยมีแกนอ้างอิงเป็น X-Y

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนประกอบพื้นฐาน

1) หัวพิมพ์ [PRINT HEAD]
2) กลไกการพิมพ์ [PRINTING MECHAMISM]
3) ล้อพิมพ์ [PLATEN]
4) กลไกป้อนกระดาษ [PAPER-FEED MECHAMISM]

หัวพิมพ์ คือส่วนที่ทำให้เกิดตัวอักษรบนกระดาษ โดยมีลักษณะต่าง ๆ อาจเป็นก้านพิมพ์เหมือนเครื่องพิมพ์ดีด หรือเป็นลวดสายไฟเส้นเล็ก ๆ โปรแกรมด้วยระบบอีเลคโทรนิคส์

กลไกการพิมพ์ ทำหน้าที่ควบคุมการทำงานของหัวพิมพ์กลไกบางแบบเป็นแบบ 2 ทิศทาง [BIDIRECTIONAL] คือสามารถพิมพ์จากซ้ายไปขวาและขวามาซ้ายได้พร้อมกัน

ล้อพิมพ์ คือ ล้อหมุน เหมือนกับที่ในเครื่องพิมพ์ดีด เป็นที่หัวพิมพ์เคาะบนกระดาษกับตัวนี้

กลไกป้อนกระดาษ ทำหน้าที่เคลื่อนกระดาษไปข้างหน้าขณะกำลังพิมพ์ แบ่งเป็นการป้อนกระดาษโดยอาศัยความฝืด [FRICTION FEED] เช่น ล้อพิมพ์

การป้อนคล้ายแทรคเตอร์ [TRACTOR FEED] กระดาษเป็นแบบเจาะรู ๆ ข้าง ๆ โดยตลอด แล้วเครื่องพิมพ์ก็ดึงไปด้วยหนามเตยหรือสปรอคเกต [SPROCKET]

การป้อนด้วยเป็ก [PEG LINK PIN] เล็ก ๆ ซึ่งติดบนล้อพิมพ์ แล้วดึงกระดาษไปข้างหน้าวิธีนี้ทำให้กระดาษวางตัวแน่นอนมาก

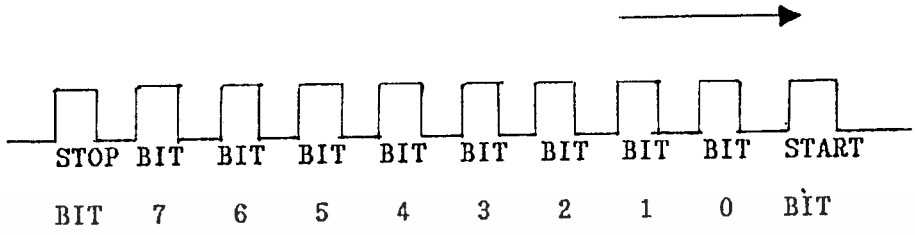
อินเตอร์เฟสกับเครื่องพิมพ์

มี 2 แบบ คือ

1) อนุกรม [SERIAL]
2) ขนาน [PARALLEL]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SERIAL



ข้อมูลถูกส่งไปทีละ BIT โดย BIT ข้อมูลทั้งหมดจะถูกกำหนดด้วย BIT นำหน้าและลงท้าย [SURROUNDED] ซึ่งเรียกว่า "บิทกำหนดเฟรม [FRAMING BITS]" ลักษณะการสื่อสารข้อมูลแบบนี้กระทำผ่านพอร์ทที่เรียกว่า "อนุกรม" ของคอมพิวเตอร์

PARALLEL

ข้อมูลจะถูกส่งออกไปพร้อมกัน โดยมีสัญญาณสุดท้ายเรียกว่า "สโตรบ [STROB]" บอกเครื่องพิมพ์ว่าให้พิมพ์ตัวอักษรนั้นได้แล้วนะ วิธีนี้จะใช้สายเชื่อมต่อไม่ยาวกว่า 3 เมตร

ชุดตัวอักษร

เครื่องพิมพ์ส่วนมากใช้ชุดตัวอักษรตามมาตรฐาน ASCII ซึ่งประกอบด้วยอักษร 96 ตัว คือ ตัวหนังสือ ตัวเลข และสัญญาณต่าง ๆ แต่ ASCII จริง ๆ แล้วจะมี 128 ตัว ความจริงบางเครื่องมีวิธีให้ผู้ใช้สร้างตัวอักษรขึ้นเองได้แทนตัวเดิมหรือเพิ่มเข้าไปใหม่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ทางด้านส่วนควบคุม

PRINTER ก็จะเหมือนกับไมโครเครื่องหนึ่ง ซึ่งในตัว PRINTER ก็จะมี CPU, ROM, และ PORT คือเครื่องจะถูกทำงานโดยโปรแกรมที่อยู่ใน ROM และนอกจาก ROM จะเก็บโปรแกรมควบคุม PRINTER แล้วยังมีหน้าที่เก็บ CHARACTER PATTERN มาตรฐาน

ส่วน RAM จะมีหน้าที่ 2 อย่าง คือ

1. ใช้เป็น PRINTER BUFFER คือเป็นหน่วยความจำเพื่อเก็บข้อมูลที่ส่งมาจากเครื่องไมโครคอมพิวเตอร์ก่อนที่จะนำไปพิมพ์ ประโยชน์ คือจะทำให้ไมโครคอมพิวเตอร์สามารถทำงานอื่นได้ในขณะที่กำลังพิมพ์ข้อมูล

2. ใช้ในการ DOWN LOAD CHARACTER RAM ส่วนนี้จะทำหน้าที่แทน ROM ในการเก็บ CHARACTER PATTERN ชั่วคราว PATTERN นี้อาจได้มาจากการ COPY DATA จาก ROM ในส่วนของ PRINTER เอง หรือจากการใช้ DATA ใหม่โดยผ่านทาง MICRO COMPUTER เช่น การที่จะทำให้ พิมพ์ภาษาไทยก็ทำได้โดยเปลี่ยน PATTERN เดิมที่ ROM เลย ทางด้าน 128-255 ที่เป็นตัวเอียงหรือเขียนโปรแกรม DOWN LOAD เพื่อทำการ RUN โปรแกรมก็จะทำการบรรจุ ที่เราต้องการลงใน RAM

ความหมายของสัญญาณใน PRINTER

STROBE	I/P	เป็นสัญญาณที่ส่งจากอุปกรณ์ภายนอก เช่น ชุดไมโครคอมพิวเตอร์ เพื่อบอกให้เครื่องพิมพ์รับข้อมูลไปได้ดดยทั่วไปจะมีความกว้างของพัลส์ประมาณ 0.5 ถึง 1 us
ACKNLG	O/P	เป็นสัญญาณที่ส่งจากเครื่องพิมพ์ เพื่อบอกให้อุปกรณ์ภายนอกรับรู้ว่าได้รับข้อมูลไว้เรียบร้อยแล้ว โดยทั่วไปจะมีความกว้างของพัลส์ประมาณ 5-12 us
BUSY	O/P	เป็นสัญญาณที่ส่งจากเครื่องพิมพ์ เมื่อเป็น "HIGH" จะหมายถึงยังไม่พร้อมที่จะรับข้อมูล ซึ่งสาเหตุต่าง ๆ อาจเกิดจาก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1) เครื่องพิมพ์อยู่ในระหว่างกระทำข้อมูลที่รับเข้ามา
2) อยู่ในระหว่างพิมพ์ข้อมูล
3) อยู่ในระหว่าง OFF-LINE STATE
4) ข้อมูลที่ส่งเข้ามาไม่สามารถตีความได้ (เช่น ข้อมูลผิดพลาดหรือใช้รหัสไม่ถูกต้อง)

PE	O/P	สัญญาณบอกว่ากระดาษพิมพ์หมด
SELECT	O/P	เป็นสัญญาณบอกว่าเครื่องพิมพ์อยู่ในสถานะใด (ใช้งานหรือไม่ใช้งาน) ถ้า HIGH เครื่องพิมพ์จะใช้งานอยู่ ถ้าเป็น LOW จะอยู่ในสถานะที่ไม่ใช้งานหรือไม่สามารถรับข้อมูลได้โดยสถานะทั้ง 2 นี้ เกิดจาก 1) สถานะที่ใช้งานได้ [SELECT STATE] - มีการกดปุ่ม SELECT หรือ ON LINE ในขณะที่เครื่องพิมพ์อยู่ในสถานะ DESELECT หรือ OFF LINE 2) สถานะที่ไม่ใช้งาน [DESELECT STATE] - มีการกดปุ่ม SELECT หรือ ON LINE ในขณะที่อยู่ในสถานะ SELECT หรือ ON LINE - ได้รับรหัส DC3 [13H] - เมื่อกระดาษพิมพ์หมด - เมื่อเครื่องพิมพ์อยู่ในสถานะ FAULT
AUTO FEEDXT	I/P	เมื่อ PRINTER ได้รับสัญญาณนี้ก็จะเลื่อนบรรทัดใหม่ 1 บรรทัดหลังจากที่พิมพ์เสร็จ
INIT	I/P	เมื่อ PRINTER ได้รับสัญญาณนี้เครื่องจะเลื่อนหัวพิมพ์สู่จุดเริ่มต้นพิมพ์ ข้อมูลต่าง ๆ จะถูกละทิ้งหมด โดยทั่วไปจะมีความกว้างของพัลส์ประมาณ 50 us

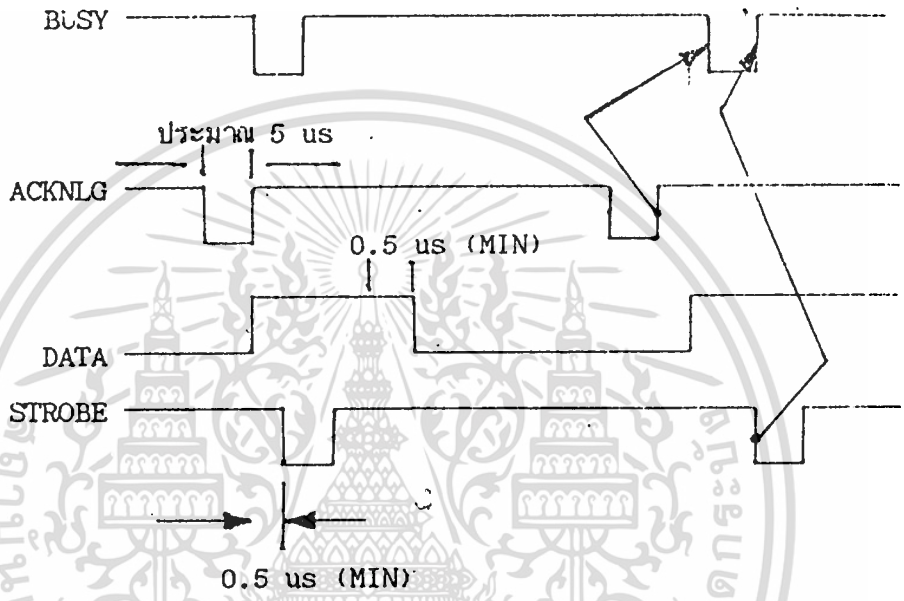
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ERROR O/P PRINTER ส่งสัญญาณไปบอกอุปกรณ์ภายนอกว่ามีข้อผิดพลาดขึ้น เช่น

- กระดาษหมด
- อยู่ระหว่าง OFF LINE
- ข้อมูลที่รับเข้ามาผิดพลาดไม่สามารถตีความได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

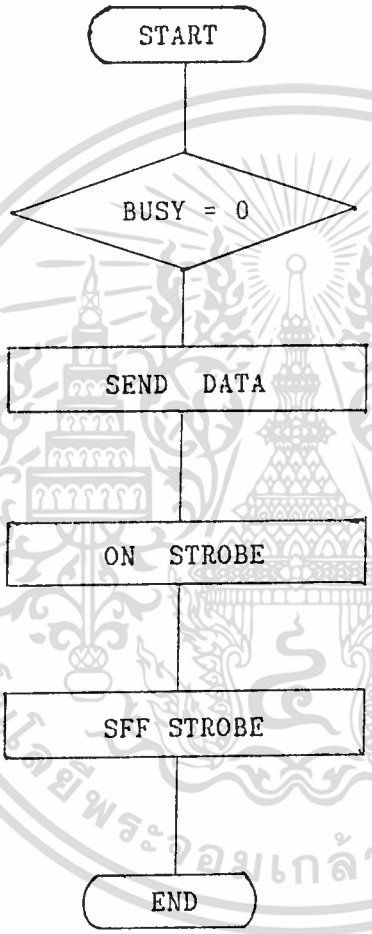
สัญญาณของการส่ง DATA ไปยัง PRINTER



สายสัญญาณ ACKNLG จะเป็นตัวบอกว่า ตอนนี้ PRINTER สามารถที่จะรับข้อมูลได้ซึ่งสัญญาณนี้ PRINTER จะเป็นผู้ส่งออกไป จากรูป พอพัลส์ของ ACKNLG ตกลงจนกระทั่งที่ขอบขาขึ้นของ ACKNLG ซึ่งมีความกว้างของพัลส์ประมาณ 5 us เริ่มเกิด ก็จะทำให้สัญญาณ BUSY จาก PRINTER เกิดขึ้นเป็น LOW ซึ่งมีความกว้างของพัลส์อย่างน้อย 0.5 us เป็นการบอกกับอุปกรณ์ที่จะส่งให้ PRINTER ว่าตอนนี้ให้ส่งข้อมูลมาได้แล้วพอหมดพัลส์ของ BUSY อุปกรณ์ที่ส่งข้อมูลมายัง PRINTER ต้องส่งสัญญาณ STROBE มาให้ PRINTER เพื่อเป็นการบอกว่า PRINTER อ่านข้อมูลที่ส่งไปให้ได้แล้ว ซึ่ง STROBEต้องมีความกว้างของพัลส์อย่างน้อย 0.5 us จากรูป จะเห็นว่าตอนที่ STROBE เกิด BUSY จะกลายเป็น HIGH ซึ่งหมายถึงตอนนี้ข้อมูลตัวต่อไปไม่สามารถส่งมาได้เพราะ PRINTER กำลังกระทำเกี่ยวกับข้อมูลอยู่ พอ STROBE หมดก็ให้ส่ง DATA ตัวเดิมออกไปอีกทีอย่างน้อย 0.5 us เพื่อไม่ให้ข้อมูลเกิดผิดพลาด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

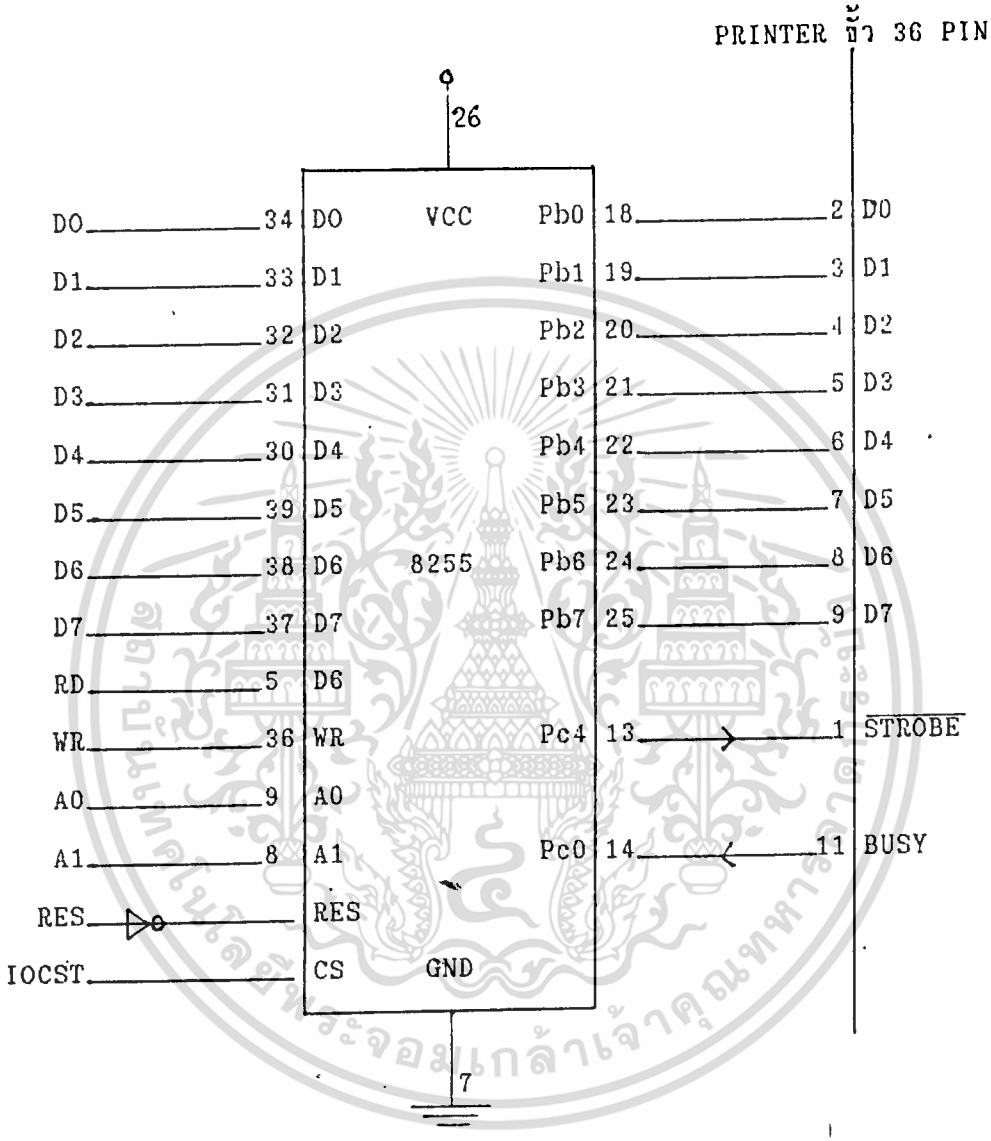
สรุปด้วย FLOW



จาก FLOW จะเห็นว่าไม่มีส่วน DEPLAY 0.5 us เพราะ 1 คำสั่งของ Z80 บน ชุด 8048 INTERFACE ก็เกิน 0.5 us แล้ว

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

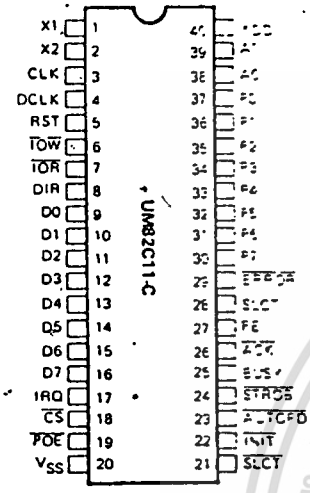
วงจรสำหรับทดลอง 1 (ด้าน 8255 SOCKET ว่าง)

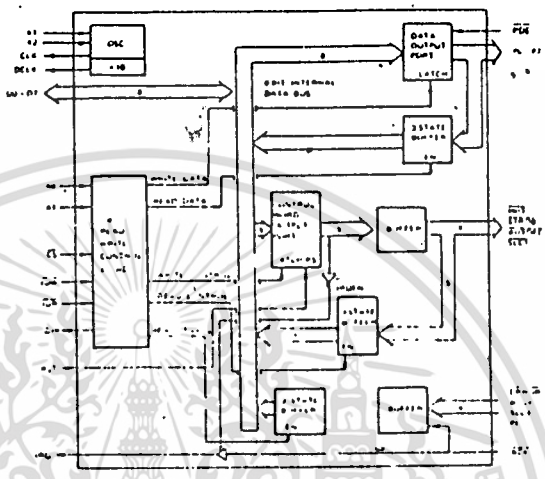


จากวงจรและ ที่กล่าวมานั้นเป็นแบบง่าย ๆ ดังนั้นถ้าอยากทราบสภาวะต่าง ๆ ที่เกิดกับ PRINTER ก็ไม่สามารถรู้ได้เช่น กระดาษหมด เกิด ERROR ขึ้น จึงมี IC ที่ผลิตขึ้นมาสำหรับการ INTERFACE กับ PRINTER คือเบอร์ 82 C 11

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปแสดงขา IC





กล่าวคร่าว ๆ เกี่ยวกับ 82 C 11

เมื่อทำหน้าที่ INTERFACE กับ PRINTER สามารถให้สถานะต่าง ๆ ที่เกิดขึ้นจาก PRINTER ได้ และกระทำการ PRINT ได้โดยวิธี INTERRUPT ทำให้ไมโครสามารถทำงานอื่นได้ด้วย โดยที่ไม่ต้องวน LOOP PRINT จนจบ ทำให้ง่ายต่อการเขียนโปรแกรมและการต่อ INTERFACE

ที่ 82 C 11 นี้มีสาย ADDRESS 2 เส้น จึงทำให้สามารถอ้างตำแหน่งของ PORT ในตัวมันได้ 4 ตำแหน่งคือ เมื่อ 2 BIT หลัง A0, A1 เป็น 00, 01, 10 และ 11 นั่นเอง แต่ตัวมันจะใช้งานเพียง 3 PORT เท่านั้น คือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ที่ A1, A0 เป็น 00 ใช้สำหรับการเขียนหรืออ่านค่าข้อมูลที่อยู่บนBUS PO-P7

01 อ่านค่าสถานะต่าง ๆ ของ PRINTER เช่น อ่าน BUSY หรือ ERROR เป็นต้น

10 ใช้เขียนหรืออ่าน CONTROL WORD เพื่อกำหนดการทำงานของ 82 C 11 เช่น ให้ทำการ INTERRUPT ส่ง STROBE ไปยัง PRINTER เป็นต้น หรืออ่านค่า BIT จาก BIT เหล่านั้นกลับเข้ามา

ในการทดลองนี้เราจะสนใจ 2 สัญญาณด้วยกันคือ

IRQ INTERRUPT REQUEST เป็น OUTPUT ขานี้จะ ACTIVE เมื่อขา ACKNLG ACTIVE LOW โดยเมื่อกำหนด BIT ที่ 4 ใน CONTROL WORD ให้ ENABLE คือ D4 = 1 จะเป็นว่า IRQ = ACKNLG แต่ถ้าให้ D4 = 1 จะเป็นการ DISABLE คือ IRQ จะเป็นสถานะปล่อยลอย [FLOATING]

BUSY จะเป็น INPUT เมื่อ PRINTER ส่ง BUSY มาให้ โดยถ้าเป็น HIGH หมายถึง ไมโครไม่สามารถส่งข้อมูล แต่ถ้าเป็น LOW คืออนุญาตให้ส่งข้อมูลได้สถานะที่จะอ่านได้อยู่ที่ BIT D7 ของ CONTROL WORD

<u>หน้าที่ของแต่ละ PORT</u>

A1, A0 เป็น 00 สำหรับเขียนอ่าน DATA ยัง PRINTER

การเขียน DATA DO-D7 จะถูกส่งไปยัง PO-P7 และจากนั้นก็จะถูกส่งไปยัง PRINTER เมื่อการอ่าน LOW ACTIVE เมื่อข้อมูลจาก PRINTER จะถูกอ่านกลับมายังคือ CPU ข้อมูลบน PO-P7 จะเข้ามายัง DO-D7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

A2, A0 เป็น 01 อ่านค่าสถานะบน PRINTER โดยแต่ละ BIT ถูกกำหนดไว้ดังนี้

DATA	D7	D6	D5	D4	D3	D2	D1	D0
STATE	BUSY	ACK	PE	SLCT	ERROR	-	-	-

ข้อสังเกต สถานะของ BUSY ที่อ่านได้จาก PRINTER จะถูก INVERT เช่น สถานะปกติของ BUSY เป็น 1 แต่เมื่อทำการอ่าน STATUS ที่ BIT D7 นี้จะได้ = 0

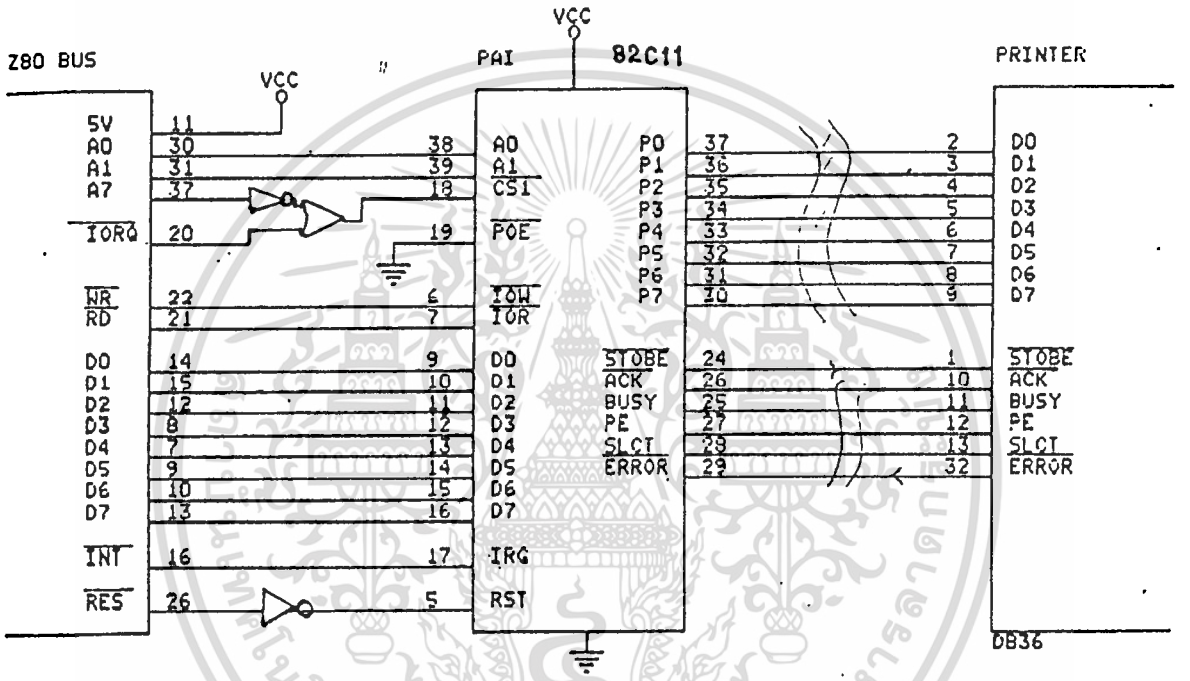
A1, A0 เป็น 10 ใช้เขียนอ่าน BIT ควบคุมไปยัง PRINTER

DATA BUS	D7	D6	D5	D4	D3	D2	D1	D0
CONTROL SINGNAL	-	-	-	IRQEN	SLCT	INIT	AUTOFD	STROB

การเขียนสัญญาณที่ต้องการจะถูก LATCH ที่ CONTROL BUS เมื่อ LOW ACTIVE โดยการที่จะให้สัญญาณใด ACTIVE ก็ดูที่สถานะของ BIT นั้นว่าต้องการ LOGIC อะไรเช่น IRQEN จะACTIVE เมื่อให้ D4 = 1 หรือ SLCT ACTIVE LOW แต่ที่ D3 ถูก INVERT ดังนั้นจึงต้องส่ง 1 ที่ D3 ก็จะเท่ากับ SLCT ACTIVE การอ่านก็สามารถที่จะอ่านค่า BIT ใด ๆ ใน CONTROL WORD กลับเข้ามาได้ เมื่อให้ IOR ACTIVE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วงจรที่ใช้สำหรับทดลอง 2



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

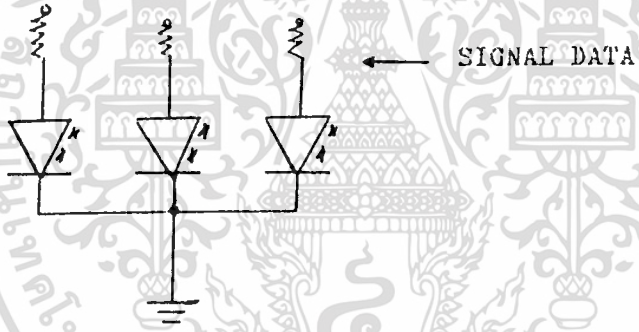
DISPLAY

วัตถุประสงค์

1. เพื่อให้รู้จักการ DISPLAY อย่างง่าย ๆ
2. เพื่อให้รู้จักการ SCAN
3. เพื่อให้เกิดแนวความคิดในการออกแบบ

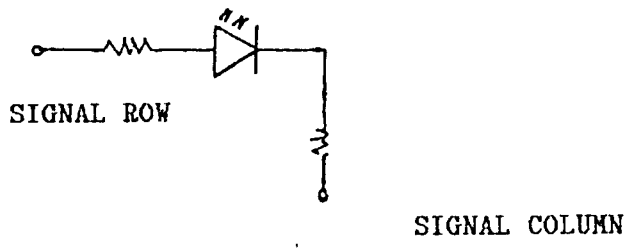
ทฤษฎี

เราจะกล่าวถึงการ DISPLAY อย่างง่าย ๆ ก่อนนั้นก็คือ DISPLAY แบบแนวเดียว



จากรูป เราจะควบคุมสัญญาณเพียงด้านเดียว คือ ส่งแต่ข้อมูล DATA แล้วหน่วงเวลาไว้ระยะหนึ่งแล้วจึงส่งสัญญาณตัวต่อไปก็จะทำให้เราเห็นเป็นไฟวิ่งได้

จากนั้นเราก็จะสามารถให้เห็นเป็นตัวอักษรหรือรูปภาพได้ ถ้าเราเพิ่ม LOGIC การควบคุมให้กับมันซึ่งแทนที่จะควบคุมการติดดับเพียงด้านเดียวก็เป็น 2 ด้าน ดังรูป



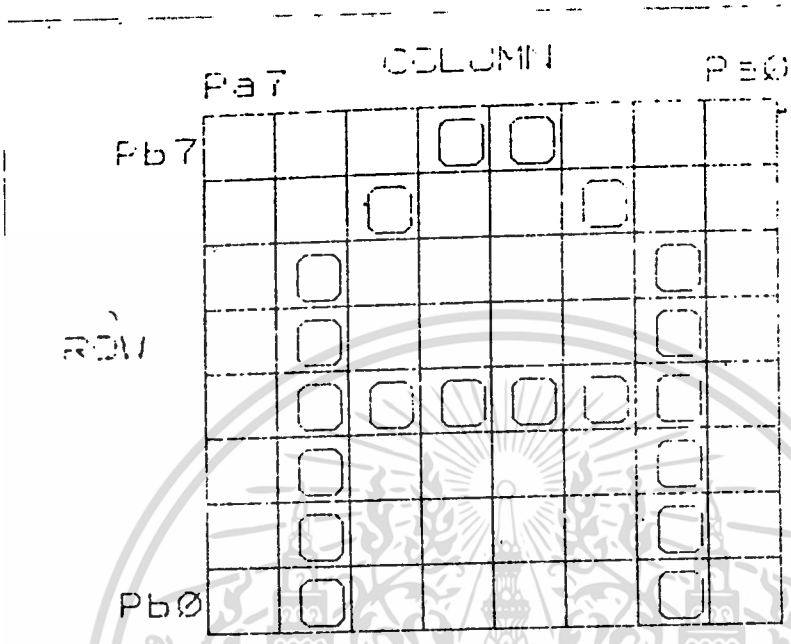
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อเราต่อสัญญาณในลักษณะนี้เราจะควบคุมสัญญาณตามจุดต่าง ๆ ได้เท่ากับ ROW X COLUMN การทำให้เกิดตัวอักษรได้นั้นเราจะต้องส่งสัญญาณ ROW และ COLOMN ให้สอดคล้องกัน ซึ่งเราเรียกว่าวิธีการ MULTIPLEX คือ เราส่งข้อมูลตัวที่ 1 แล้วเราก็ ACTIVE COLUMN ตัวที่ 1 แล้วหน่วงเวลาไว้ระยะหนึ่ง จากลักษณะนี้ก็คล้ายการ OUT แบบด้านเดียว ถ้าเราคิดการ OUT แบบ 2 ด้านเป็นด้านเดียว จากนั้นก็ส่งข้อมูลตัวที่ 2 แล้วก็ ACTIVE COLUMN ตัวที่ 2 แล้วหน่วงเวลาทำอย่างนี้ไปเรื่อย ๆ จนครบ COLUMN สุดท้าย ซึ่งตอนนี้เราจะได้ตัวอักษรหรือรูปภาพแล้ว แต่ภาพที่เกิดจะเกิดในช่วงเวลาสั้น ๆ เพราะ CPU ทำงานเร็วเราจะเป็นต้องให้กลับไปเริ่มส่งข้อมูลแบบเดิมอีกระยะหนึ่งเพื่อให้เห็นภาพนั้นชัดติดตาต่อจากนั้นก็จะเป็นการทำให้ตัวอักษรเลื่อนก็โดยการเลื่อน ADDRESS ของ DATA ตัวแรกไปอีก 1 ADDRESS แล้ว SCAN (คือส่งข้อมูลแบบที่กล่าวไปแล้วจน COLUMN สุดท้าย) แล้วกลับไปวนข้อมูลนั้นระยะหนึ่งแล้วก็เพิ่ม ADDRESS ของ DATA ที่เพิ่มแล้วขึ้นไปอีก 1 ทำอย่างนี้เรื่อยไปจนถึง ADDRESS ของ DATA ตัวอักษรสุดท้ายเป็นการแสดงว่าเลื่อนตัวอักษรตัวสุดท้ายแล้วจากวงจรเรา SET ให้ 8255 PORT A เป็น COLUMN และ PORT B เป็น ROW โดยการสั่ง INITIAL ที่ CONTROL PORT ของ 8255 (ทาง SOFTWARE) โดยจะนำ 8255 ทาง SOCKET ว่างบน 8248 INTERFACE มาใช้เป็นตัวติดต่อกับ MATRIX เนื่องจาก MATRIX นี้กินกระแสพอสมควร เราจึงต้องแยก SUPPLY แล้วก็ต่อ GND กับ 8248 INTERFACE ด้วย

การที่จะให้ MATEIX ใดสว่างก็ส่ง ROW ด้วย 0 และ COLUMN ด้วย 1 เช่นตัวอักษร A จากนั้นเราก็มาดูว่าจะให้อักษรเลื่อนทางไหน ตัวอย่างเช่นให้เลื่อนจากซ้ายไปขวา จากรูปเมื่ออักษร A เลื่อนก็คือ DATA ที่ Pa6 จะต้องเลื่อนมาที่ Pa7 แทน และ Pa7 ก่อนแล้วไปเก็บใน ADDRESS ที่จะนำตัวอักษรออกไปแสดง จากนั้นก็เริ่มต่ำแหน่งที่จะเก็บแล้วนำข้อมูลใน Pa6 ไปเก็บทำเรื่อย ๆ ไปจนถึง Pa0 ก็จะได้ตัวอักษร 1 ตัว นั่นก็คือ 1 ตัวอักษณใช้ข้อมูล 8 BYTE

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การกำหนดตัวอักษร [CHARACTER GEN]



สรุป

1. ข้อมูลของตัวอักษรถูกส่งมาที่ ROW จากวงจร ACTIVE 0 แต่เวลาคิดจะยากให้คิดที่ LOGIC 1
2. ให้ Pa7 ACTIVE ทีแรกแล้วอ่าน DATA จาก ROW ที่ส่งมีค่าอะไรแล้วนำไปเก็บใน MEMORY
3. อ่านข้อมูลในลักษณะเดิมแต่ ACTIVE COLUMN ถัดไป

จากตัวอย่าง เช่นเก็บตัวอักษร A ไว้ที่ ADDRESS 3000H จะได้เป็น

3000	00	F	48	88	88	48	3F	00
3008	XX							

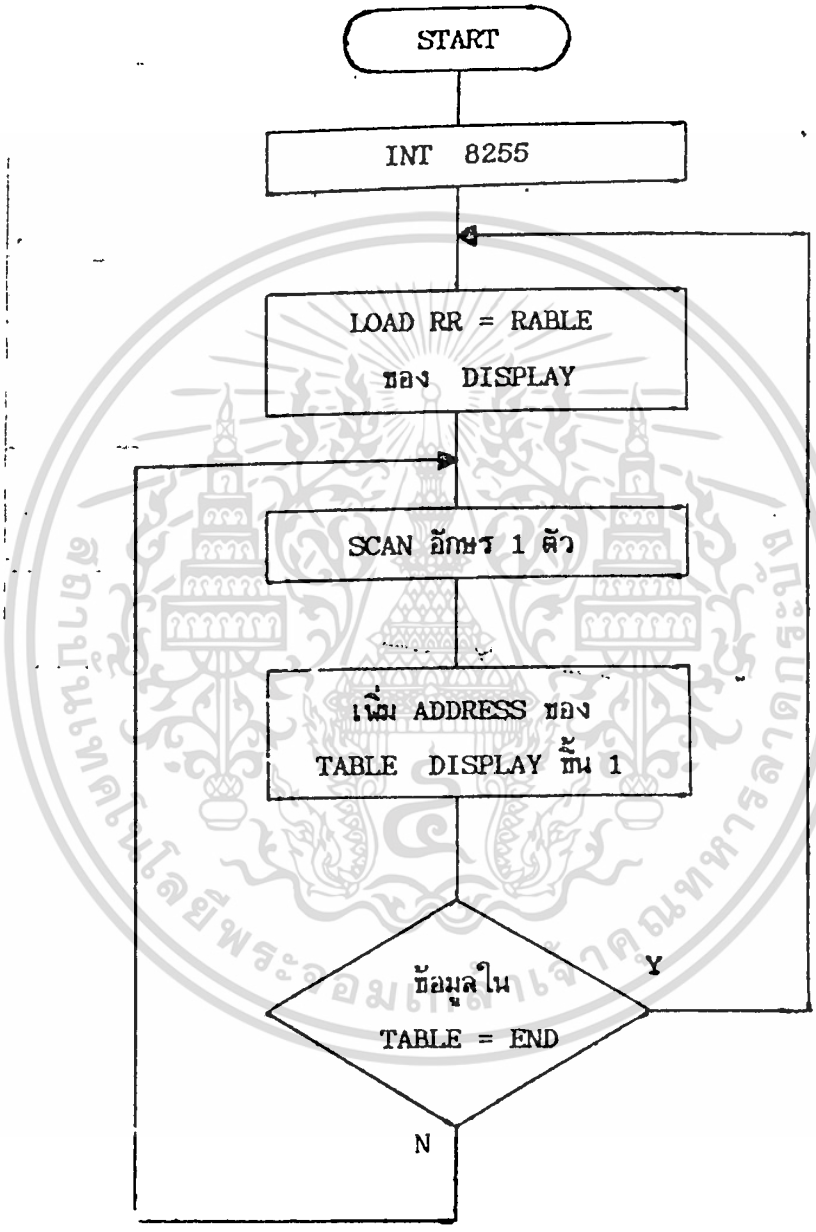
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

<u>อธิบายอักษร A เลื่อนไว้ว่า</u>

1. ส่งสัญญาณไปให้ Pa7 ACTIVE HIGH
2. ส่ง DATA ไปที่ ROW เนื่องจาก ACTIVE ROW ก็ใช้คำสั่ง COMPLEMENT DATA ก่อนที่จะส่ง
3. หน่วงเวลาไว้นิดหนึ่ง
4. ทำการ SCAN ให้ COLUMN ถัดไป ACTIVE
5. เพิ่มตำแหน่งที่เก็บ DATA ตัวอักษร
6. ตรวจดูว่า SCAN จน COLUMN สุดท้าย Pa0 ACTIVE หรือยัง
   - ยัง ไปข้อ 2
   - ครบ ตอนนี้จะเห็นอักษร A สั่วขณะ ดังนั้นต้องวาดภาพนี้ซ้ำกลับไปข้อ 1 อีก
7. เลื่อนตัวอักษร คือ เลื่อนตำแหน่งข้อมูลครั้งแรกในที่นี้คือ 3000 คือ 00 ขึ้นไป 1 ADDRESS คือ 3001 แล้วนำข้อมูลในตำแหน่งนี้กลับไปยังข้อ 1
8. ดังนั้น DATA 3FH ก็จะมาแสดงที่ Pa7 และ DATA ตัวอื่น ๆ ก็จะถูกผ่ามาออกมาที่ Pa ในลักษณะถูกดันให้ไปทางซ้ายซึ่งตอนนี้ที่ Pa0 ก็จะมีข้อมูล XX ขึ้นมาแทนนั่นเอง
9. เมื่อ SHIFT ตำแหน่งตัวอักษรขึ้นเรื่อย ๆ ตัวอักษร A ก็จะหายไปมีอักษรใหม่มาแทนที่นั่นเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

FLOW CHART ของอักษรวิ่ง



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DOT MATRIX LCD MODULE

อุปกรณ์ในปัจจุบันนี้ในส่วนแสดงผลนั้นจะใช้ LCD เสียเป็นส่วนใหญ่ ไม่ว่าจะเป็นเครื่องเล่น VIDEO, เครื่องถ่ายเอกสาร, เครื่องมือวัดคุมต่าง ๆ, เครื่องคอมพิวเตอร์ เราพอจะแบ่ง DOT MATRIX LCD MODULE นี้ออกได้เป็นพวก ๆ ดังนี้ :-

1. CHARACTER LCD MODULE
2. GRAPHIC LCD MODULE
3. SEGMENT DISPLAY TYPE LCD MODULE

โดยในแต่ละแบบนี้ก็จะมีส่วนประกอบใหญ่ ๆ แบ่งได้เป็น

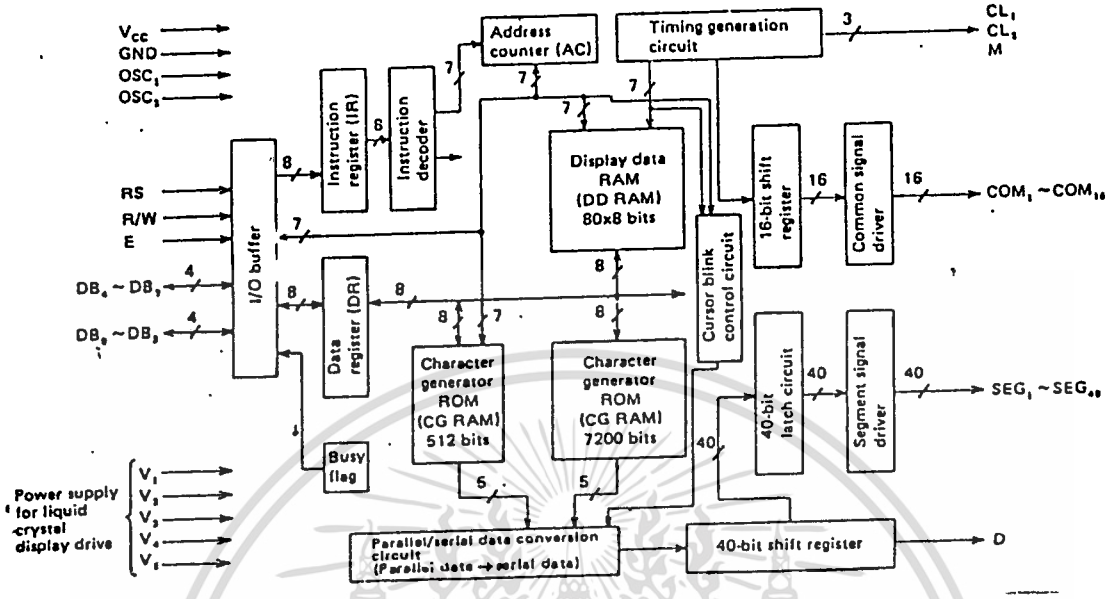
1. DOT MATRIX LCD เป็นตัวแสดงผลให้เรามองเห็นในลักษณะการปิดและเปิดตัวเองกับแสงก็คือ ส่วนของที่เป็นตัวกระจกบรรจุผลึก
2. DRIVER เป็นตัวรับสัญญาณจากตัวควบคุมมาขับผลึก LCD อีกทีหนึ่งโดยมีเบอร์ที่นิยมใช้ใน LCD MODULE เช่น HD44100H, MSM5259
3. CONTROLLER เป็นตัวรับข้อมูลจากอุปกรณ์ภายนอกมา และจัดการควบคุม LCD MODULE ให้ทำงานแสดงผลต่างๆเช่น การลบจอภาพ, การเกิดตัวอักษร เป็นต้น โดยมีเบอร์ IC ที่นิยมใช้กัน คือ HD44780 ซึ่งจะใช้ในแบบ CHARACTER LCD MODULE เป็นส่วนใหญ่ เบอร์ IC HD61830 จะใช้ในแบบ GRAPHIC LCD MODULE

ในการศึกษาทำงานและใช้งาน LCD MODULE นั้น ไม่ใช่เรื่องยากเลย ถ้าเราสามารถทำความเข้าใจในส่วนของ CONTROLLER ได้ก็เพียงพอแล้วและโดยมาก LCD MODULE ในแต่ละบริษัทแล้วจะใช้ตัว CONTROLLER ที่มีหลักการทำงานเหมือนๆ กันเป็นส่วนใหญ่และใน LCD MODULE แต่ละขนาดจำนวนตัวอักษรหรือจำนวนแบรทัดก็มีหลักการทำงานแบบเดียวกันทั้งหมด IC ที่นิยมมากที่สุดตัวหนึ่งก็เป็น CONTROLLER LCD ก็คือ เบอร์ HD44780 โดยรูปแบบการทำงานของมันได้เป็นมาตรฐานให้กับ CONTROLLER LCD ตัวอื่นๆ ด้วย

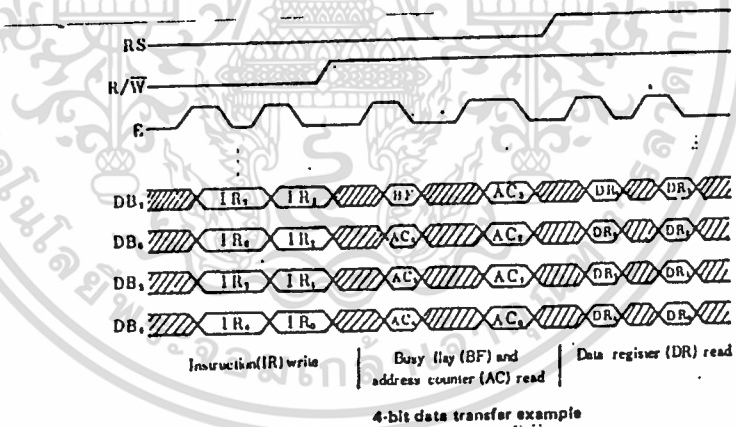
HD44780 เป็นไอซี LSI ตัวหนึ่งใช้ควบคุม LCD โดยแสดงผลในรูปตัวอักษรหรือสัญลักษณ์ต่างๆ ตัวมันเองสามารถต่อใช้งานแบบ 4 BIT หรือ 8 BIT ก็ได้ โดยถ้าเราต่อแบบ 4 BIT จะต่อใช้งานที่ DB7-DB4

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

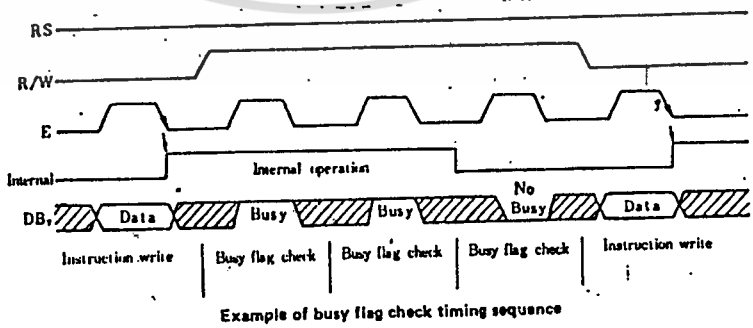
Block diagram of HD44780 interior



เท่านั้นโดยข้อมูลครั้งแรกที่ส่งนั้น HD44780 จะถือเป็นข้อมูล 4 BIT บน และข้อมูลที่ส่งต่อมานั้นเป็นข้อมูล 4 BIT ล่าง



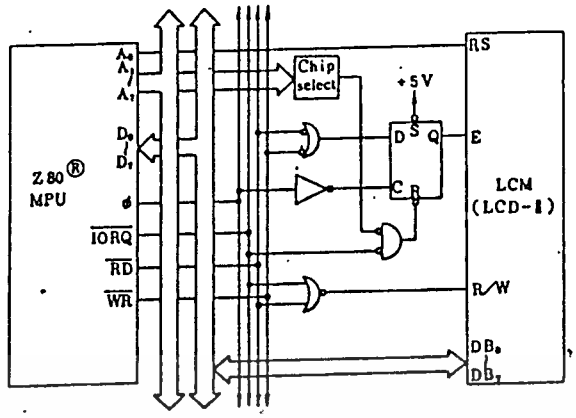
4-bit data transfer example



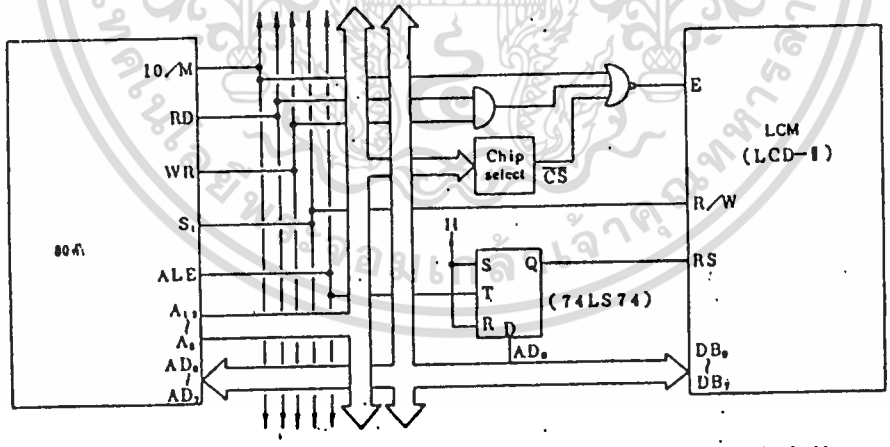
Example of busy flag check timing sequence

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Example of interfacing to Z80 MPU



เราสามารถต่อ LCD MODULE (HD44780 เป็น CONTROLLER) เข้ากับระบบไมโครได้หลายรูปแบบดังรูป



Example of connection with LCM being used as a part of memories on the determined address.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

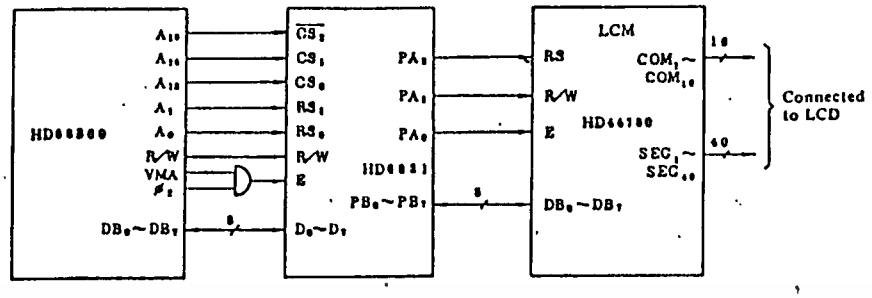
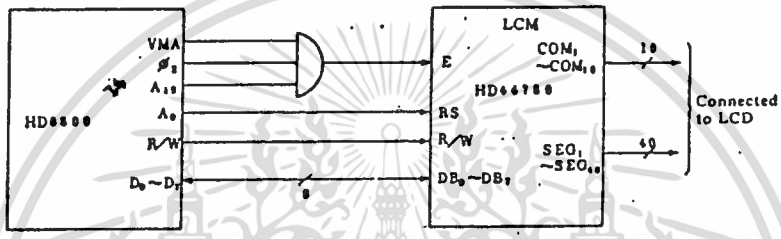
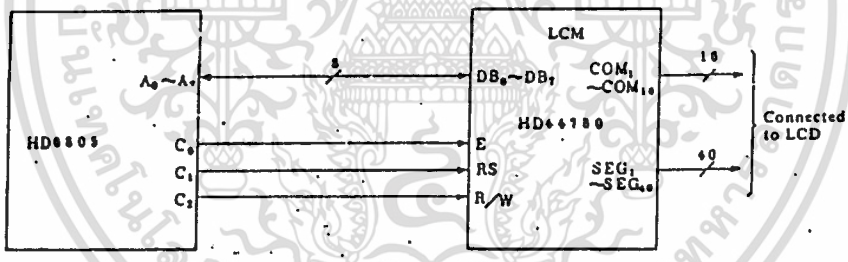


Fig. 4 Example of interface to HD68B00 using PIA (HD68B21)

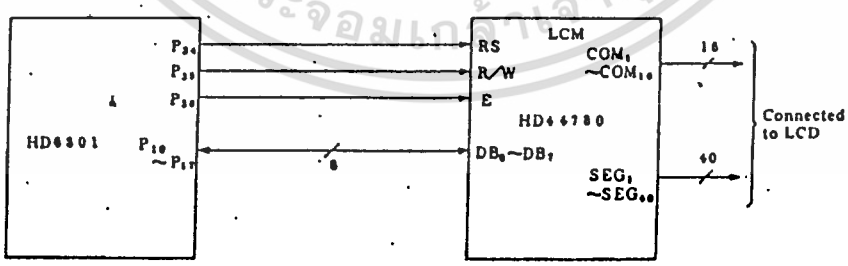
Connecting directly to the 8 bit MPU bus line



Example of interfacing to the HD6805

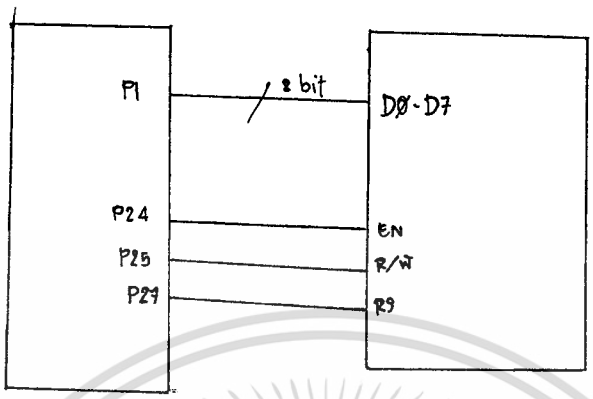


Example of interfacing to the HD6301



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวอย่างการต่อใช้งานจริงกับ 8048 INTERFACE



จากวงจรเป็นการต่อ 8255 ให้เข้าใช้กับ LCD โดยเราจะจำลองสัญญาณต่างๆ ขึ้นมา โดยการใช้ PORT A และ PORT B โดย PORT A นั้นเราให้เป็น DATA PORT และ PORT B นั้นเราให้เป็นสัญญาณควบคุมไปใช้

เมื่อเราเริ่มเปิดไฟป้อนให้ HD44780 นั้น ก็จะทำการ RESET ตัวมันเองโดยจะใช้เวลาประมาณ 10 ms หลักจากไฟ VDD ถึง 4.5 VOLT แล้ว โดยจะ SET ตัวมันเองดังนี้ :-

1. DISPLAY CLEAR จะทำการลบข้อมูล จอภาพ LCD

2. FUNCTION SET โดยจะ SET ค่าภายใน
   DL = 1 : เป็นการ SET ให้การติดต่อแบบ 8 BIT
   N = 0 : SET เป็น 1 บรรทัดการแสดงผล
   F = 0 : 5X7 DOT ต่อหนึ่งต่ออักษร

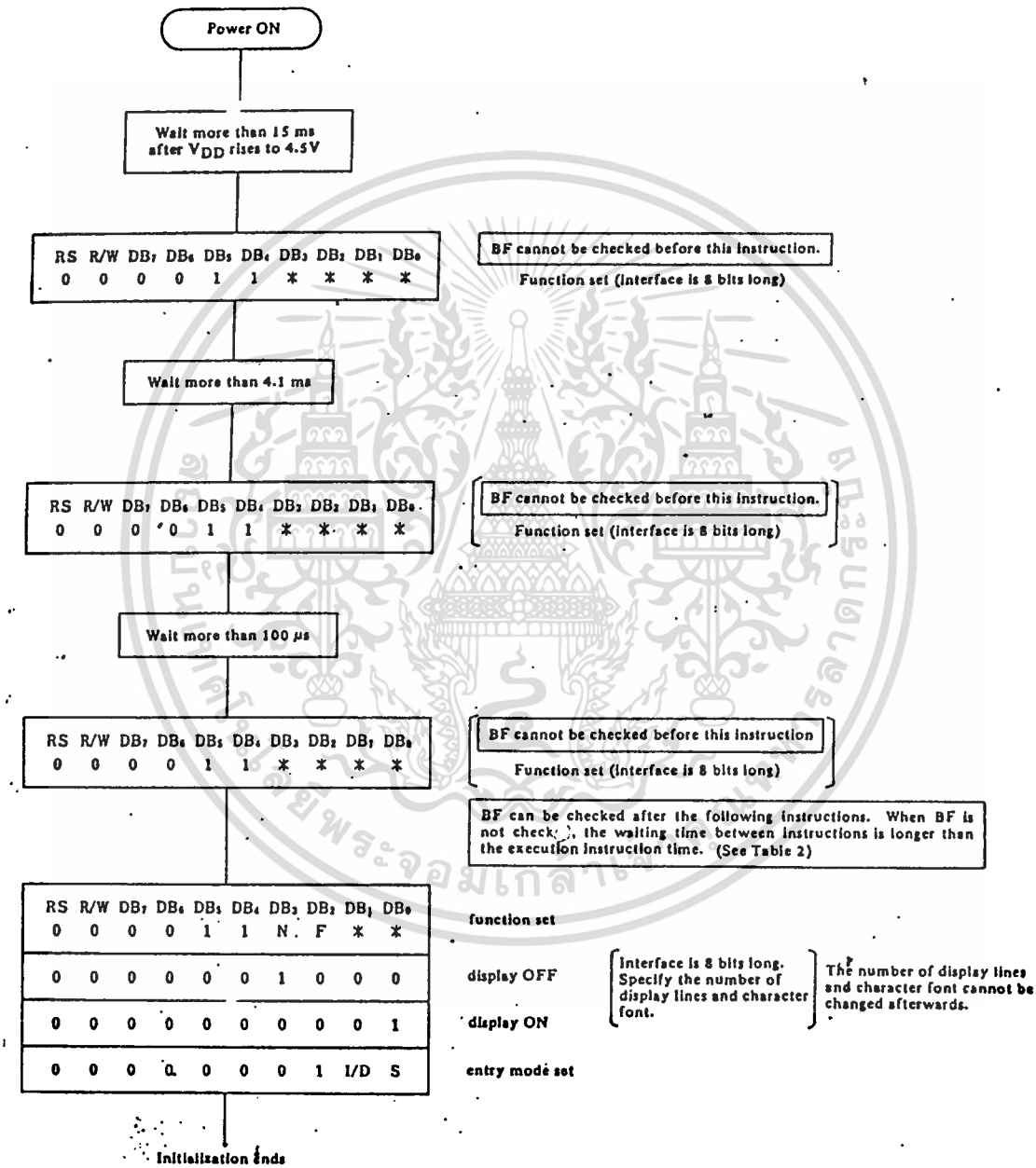
3. DISPLAY ON/OFF D = 0 : DISPLAY OFF
   C = 0 : CURSOR OFF
   B = 0 : BLINK OFF

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4. ENTRY MODE SET I/D = 1 : +1 (เพิ่มค่า COUNTER ขึ้น 1)

S = 0 : NO SHIFT

เมื่อเราเริ่มเปิดเครื่องทำงานแล้ว ก็จะต้องส่งค่าสั่งควบคุมให้มันเริ่มทำงาน ดังตาราง



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางคำสั่ง ID44780

Instruction	Code										Description	Execution time (when fosc is 250 kHz) Note 1	Execution time (when fosc is 160 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.	1μs	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 x 10 dots F = 0: 5 x 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: $40\ \mu s \times \frac{250}{270} = 37\ \mu s$	

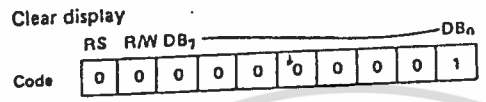
*No effect

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

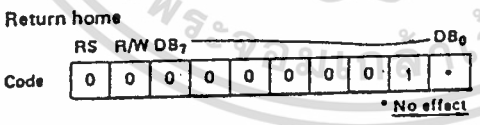
รายละเอียดของคำสั่ง HD44780

1. CLEAR DISPLAY



คำสั่งนี้จะเป็นการเขียนช่องว่างหรือ SPACE (ASCII 20H) เข้าไปใน DD RAM ทั้งหมดและทำการ SET DD RAM ADDRESSER เป็นศูนย์ ตัว CURSOR จะกลับไปอยู่ตำแหน่งบนสุดซ้ายมือของจอภาพ SET I/D = 1, S ไม่มีการเปลี่ยน

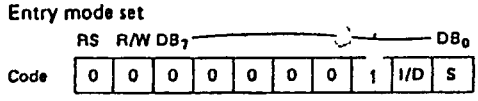
2. RETURN HOME



คำสั่งนี้จะทำการ SET DD RAM ADDRESSER เป็นศูนย์ ตัว CURSOR จะกลับไปอยู่ตำแหน่งบนสุดซ้ายมือของจอภาพข้อมูลในจอภาพไม่เปลี่ยน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

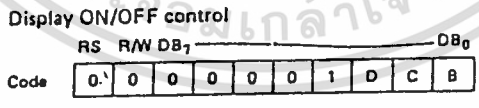
3. ENTRY MODE SET



BIT I/D : โดยจะเป็นตัวกำหนดให้ว่า เมื่อเขียนหรืออ่านข้อมูลแล้ว จะทำให้ DD RAM ADDRESS เพิ่มขึ้นหนึ่งหรือลดลงหนึ่ง โดย 1 = เพิ่ม 0 = ลดลงหนึ่ง

BIT S : เป็นตัวกำหนดแสดงผลโดยถ้า S = 1 จะเป็นการใส่ข้อมูลแล้วตัว CURSOR อยู่กับที่ ข้อมูลจะถูกดันไปทางซ้าย ถ้า S = 0 ข้อมูล จะอยู่กับที่ตัว CURSOR จะถูกดันไปทางขวามือ

4. DISPLAY ON/OFF CONTROL

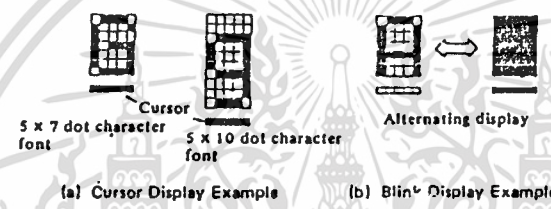


BIT D : เป็น BIT ให้เปิดปิดหน้าจอภาพโดยถ้า D = 1 จะ ON และ D = 0 จะ OFF

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

BIT C : จะให้แสดง CURSOR ให้ BIT C = 1 และถ้าไม่ต้องการแสดง CURSOR BIT C = 0 โดยตัว CURSOR จะอยู่ที่ LINE ที่ 8 ในแบบ 5XY DOT และจะอยู่ LINE ที่ 11 ในแบบ 5X10 DOT

BIT B : เป็น BIT SET เป็นการกระพริบของ CURSOR โดย B = 1 มีการกระพริบ B = 0 ไม่มีการกระพริบ โดยมีระยะเวลาการกระพริบประมาณ 379.2 ms



5. CURSOR OR DISPLAY SHIFT

Cursor or display shift

	RS	R/W	DB_7						DB_0
Code	0	0	0	0	0	1	S/C	R/L	*	*

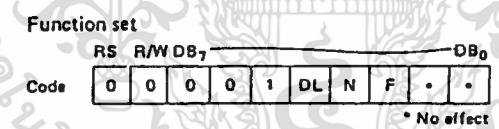
* No effect

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เป็นคำสั่งกำหนดให้ตำแหน่ง CURSOR หรือข้อมูล ไปเกิดทางซ้ายหรือขวาโดย ไม่ต้องใช้คำสั่งเขียนหรืออ่าน โดย

S/C	R/L	
0	0	ทำการย้าย CURSOR ไปจากตำแหน่งเดินไปซ้ายมือ 1 ตำแหน่ง
0	1	ทำการย้าย CURSOR ไปจากตำแหน่งเดินไปขวามือ 1 ตำแหน่ง
1	0	เป็นการดันตัวอักษรที่เกิดไปทางซ้าย
1	1	เป็นการดันตัวอักษรที่เกิดไปทางขวามือ

6. FUNCTION SET



BIT DL : เป็นการ SET การติดต่อว่าจะให้เป็นแบบ 8 BIT หรือ 4 BIT โดยถ้าต้องการติดต่อ 4 BIT DL = 0 และ 8 BIT DL = 1

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

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

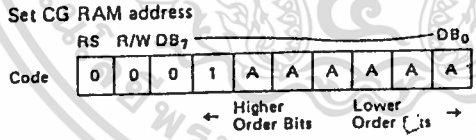
F : เป็นการ SET ขนาด DOT การแสดงผล 5X7 หรือ 5X10

โดย F = 10 เป็นแบบ 5X7 และ F = 1 เป็นแบบ 5X10

N	F	No. of display lines	Character font	Duty factor	Remarks
0	0	1	5 x 7 dots	1/8	
0	1	1	5 x 10 dots	1/11	
1	*	2	5 x 7 dots	1/16	Cannot display 2 lines with 5 x 10 dot character font.

* No effect

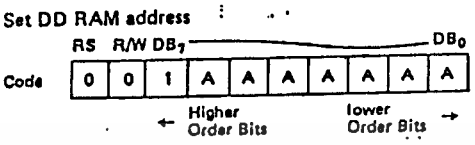
7. SET OG RAM ADDRESS



ใน HD44780 นั้น จะมีหน่วยความจำอยู่ 2 ชุด คือ DISPLAY DATA RAM (DD RAM) จำนวน 80X8 BIT และ CHARACTER GENERATOR ROM CG RAM จำนวน 512 BIT และ 7200 BIT คำสั่งนี้จะเป็นการ SET ADDRESS ใน CG RAM โดยต้องทำการ SET ADDRESS ก่อนเขียนหรืออ่านข้อมูลจาก CG RAM ด้วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8. SET DD RAM ADDRESS



เป็นคำสั่ง SET ค่า ADDRESS ใน DD RAM ในการเขียนหรืออ่านค่า จาก DD RAM (DD RAM คือ ส่วนที่จะแสดงผลหน้าจอ LCD) โดยจำนวน ADDRESS ที่จะเกิดขึ้นบนจอ LCD จะอยู่กับ SET ค่า N ด้วย

ถ้า N = 0 (1 บรรทัด) ADDRESS จะอยู่ 00H-4FH

ถ้า N = 1 (2 บรรทัด) ADDRESS จะอยู่ 00H-27H สำหรับบรรทัดที่ 1 และ 40H-67H สำหรับบรรทัดที่ 2

ตัวอย่างการจัด ADDRESS ของ DD RAM หน้าจอ LCD แบบ 16 ตัวอักษร 4 บรรทัด และ 20 ตัวอักษร 2 บรรทัด HDM-16416H, HDM-20216H

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	← display position
1-line	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	← DD RAM address
2-line	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	
3-line	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	
4-line	50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F	

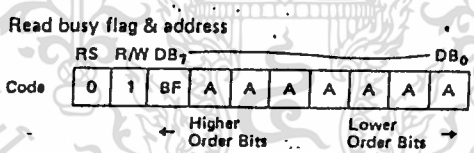
HDM-16416H

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	←display position
1-line	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	← DD RAM address
2-line	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53	
3-line	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26	27	
4-line	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F	60	61	62	63	64	65	66	67	

(Note) Shift display is as same as 2-line type.

9. READ BUSY FLAG AND ADDRESS



เป็นคำสั่งอ่านค่า BUSY FLAG ซึ่งจะเป็นตัวบอกว่าตัว HG44780 นี้อยู่ในขบวนการทำงานภายในอยู่หรือในสภาพพร้อมจะรับข้อมูล โดย

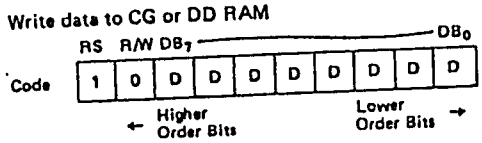
BF = 1 อยู่ในขบวนการทำงานภายในไม่พร้อมจะรับข้อมูลหรือคำสั่ง

BF = 0 พร้อมจะรับข้อมูลหรือคำสั่งได้

และนอกจากนี้ยังเป็นคำสั่งอ่านค่าข้อมูลADDRESS ของ CG RAM หรือ DD RAMด้วย

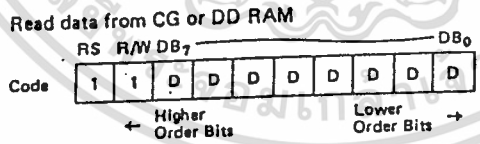
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

10. WRITE DATA TO CG หรือ DD RAM



เป็นคำสั่งเขียนข้อมูลเข้าไปใน CG หรือ DD RAM โดยเมื่อเขียนข้อมูลและ ADDRESS จะเพิ่มหรือลดโดยอัตโนมัติตามคำสั่งที่ SET ใน ENTRY MODE ข้อกำหนดที่จะรู้ว่าเป็นการเขียนข้อมูลของ CG RAM หรือ DD RAM ทำได้โดยการ SET ADDRESS ของ CG RAM หรือ DD RAM ขึ้นมาก่อนจะเขียนข้อมูล

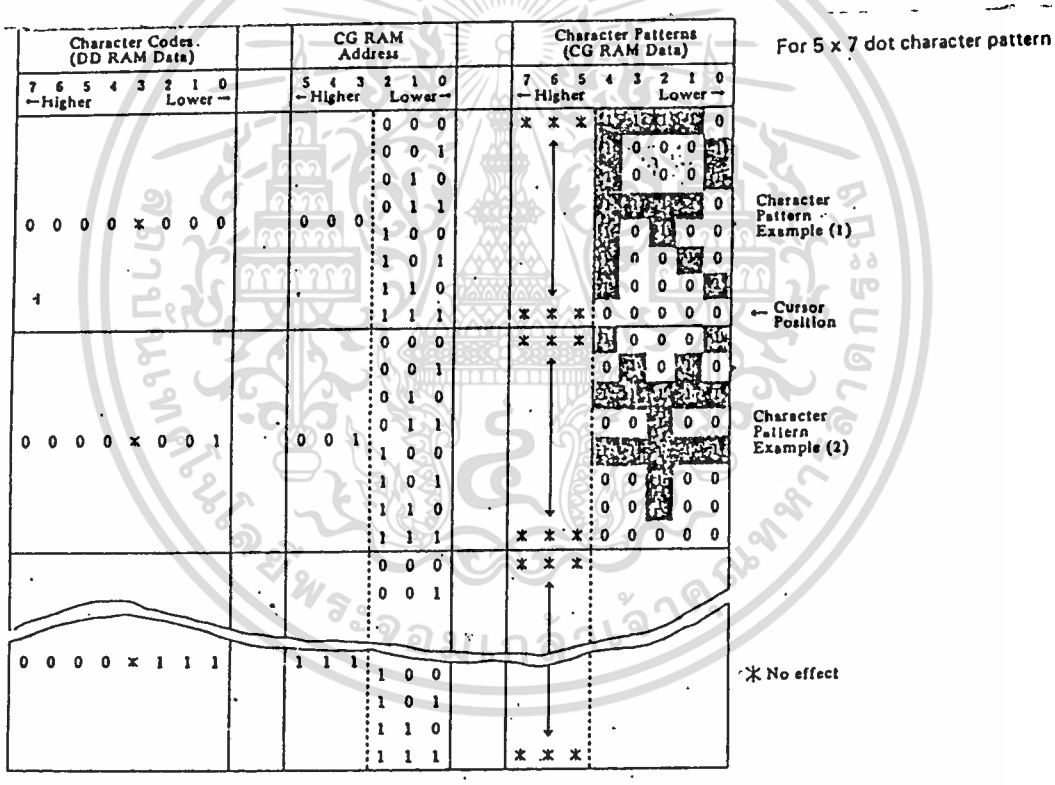
11. READ DATA FROM CG OR DD RAM



เป็นคำสั่งอ่านค่าข้อมูลจาก CG RAM หรือ DD RAM โดยก่อนอ่านค่าจาก DD RAM หรือ CG RAM นี้ควรจะใช้คำสั่ง SET ADDRESS ก่อนเพื่อให้รู้ว่าข้อมูลที่อ่านได้นั้นเป็น DD หรือ CG RAM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากตารางการทำงานจะเห็นว่าการใช้งาน LCD MODULE นั้นง่าย เพียงแต่เราส่งคำสั่งเริ่มแรกและ SET ความต้องการให้ขนาดตัวอักษร, CURSOR หลังจากนั้นเราก็สามารถเขียนตัวอักษรเข้าไปใน DD RAM ตามตารางตัวอักษรที่ให้มานั้นก็จะเกิดอักษรในจอภาพ LCD เรายังสามารถกำหนดตำแหน่งตัวอักษรที่จะให้เกิดบนจอได้โดยการ SET DD RAM ADDRESS ตามตารางที่ให้มาในหัวข้อ SET DD RAM ADDRESS ขอให้ทดสอบทำความเข้าใจกับตัวโปรแกรมที่ใช้กับ 8048 INTERFACE นี้ที่ให้มาจะเห็นว่าจะมีส่วนเริ่มต้นก็คือ ส่วนการ INITIAL LCD เพื่อกำหนดหน้าที่การทำงานต่างๆ



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ส่วนประกอบของโปรแกรม

EPLUSE จะเป็นส่วนกำเนิดสัญญาณ ENABLE SIGNAL โดยการใช้ PORT B BIT ที่ 2 กำเนิด PLUSE สัญญาณ ENABLE ขึ้น

GOTO จะเป็นส่วนกำหนดตำแหน่งของส่วน DD RAM ADDRESS ที่จะเขียนข้อมูลโดยจากโปรแกรม INITIAL ที่เรา SET' ไว้ เพื่อเขียนข้อมูลเข้าไปใน DD RAM แล้ว ADDRESS ของ DD RAM จะเพิ่มขึ้น 1 โดยทันที

WRBYTE เป็นส่วนเขียนข้อมูล 1 BYTE เข้าไปในตำแหน่ง ADDRESS ของ DD RAM ขณะนั้นๆ

WRLINE เป็นส่วนในการเขียนข้อมูลทีละ 1 LINE เพราะตำแหน่ง DD RAM ที่เกิดบนจอภาพ LCD นั้นแต่ละตำแหน่งจะไม่ต่อกันไปในแต่ละบรรทัด

จากตัวอย่างที่ให้จะมี LCD แบบ 20 ตัวอักษร 2 บรรทัด และ 16 ตัวอักษร 4 บรรทัด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
0001   FB00                        .ORG    $0FB0
0002   FB00             ;Initial function of LCD
0003   FB00             ;
0004   FB00 23 00               MOV     A,#0
0005   FB02 3A                  OUTL    P2,A
0006   FB03 23 38               MOV     A,#38H
0007   FB05 39                  OUTL    P1,A
0008   FB06 74 3F               CALL    EPULSE
0009   FB08 74 38               CALL    DELAY
0010   FB0A 23 0F               MOV     A,#0FH
0011   FB0C 39                  OUTL    P1,A
0012   FB0D 74 3F               CALL    EPULSE
0013   FB0F 23 06               MOV     A,#06H
0014   FB11 39                  OUTL    P1,A
0015   FB12 74 3F               CALL    EPULSE
0016   FB14 23 01               MOV     A,#01H
0017   FB16 39                  OUTL    P1,A
0018   FB17 74 3F               CALL    EPULSE
0019   FB19 74 38               CALL    DELAY
0020   FB1B             ;
0021   FB1B             ;Write data
0022   FB1B             ;
0023   FB1B 74 1F               CALL    WRL1
0024   FB1D 74 27               CALL    WRL2
0025   FB1F             ;
0026   FB1F             ;Write line
0027   FB1F             ;
0028   FB1F 23 00       WRL1    MOV     A,#00H
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

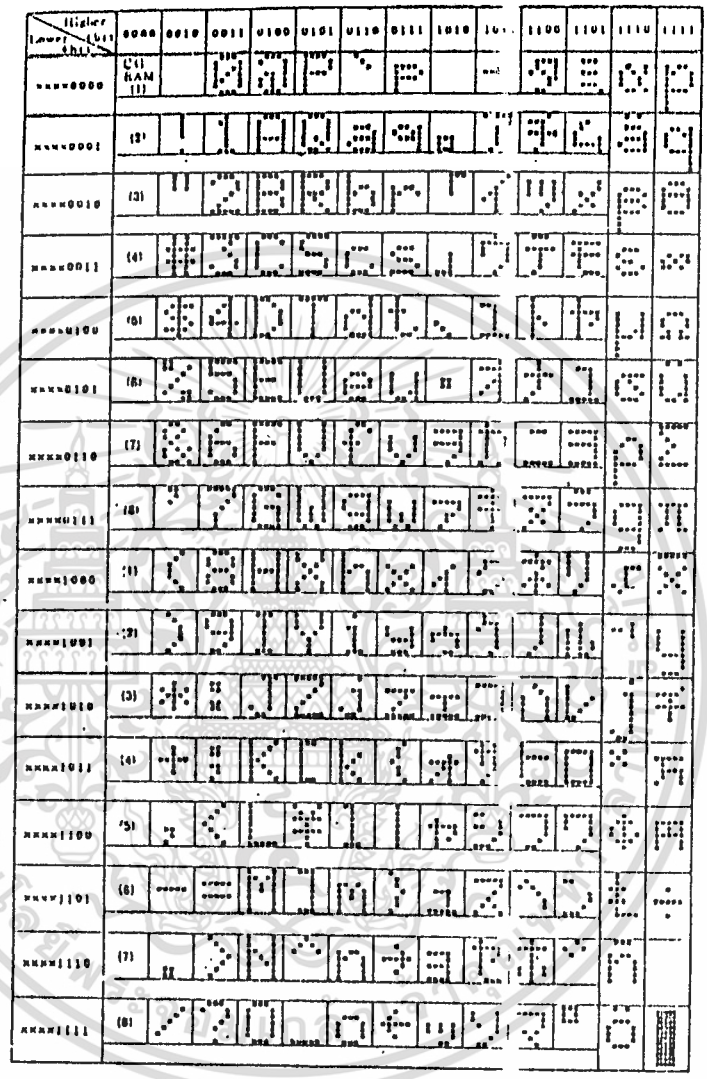
```
0029   FB21 74 4B              CALL    GOTO
0030   FB23 B8 53              MOV     R0,#TAB         ;TABLE POINTER
0031   FB25 64 2B              JMP     WRLM
0032   FB27            ;
0033   FB27 23 40      WRL2    MOV     A,#40H
0034   FB29 74 4B              CALL    GOTO
0035   FB2B BF 08      WRLM    MOV     R7,#08H
0036   FB2D 23 10      WRL     MOV     A,#10H
0037   FB2F 3A                 OUTL    P2,A
0038   FB30 80                 MOVX    A,@R0           ;MOVE DATA FROM TABLE
0039   FB31 39                 OUTL    P1,A
0040   FB32 74 3F              CALL    EPULSE
0041   FB34 18                 INC     R0
0042   FB35 EF 2D              DJNZ    R7,WRL
0043   FB37 83                 RET
0044   FB38            ;
0045   FB38            ; DELAY TIME SUB.
0046   FB38            ;
0047   FB38 BE 00      DELAY   MOV     R6,#0H
0048   FB3A 00         DE1     NOP
0049   FB3B 00                 NOP
0050   FB3C EE 3A              DJNZ    R6,DE1
0051   FB3E 83                 RET
0052   FB3F            ;
0053   FB3F            ; ENABLE SIGNAL
0054   FB3F            ;
0055   FB3F 0A         EPULSE  IN      A,P2
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
0056   FB40 43 40              ORL     A,#40H          ;SET BIT 6 "ENABLE
0057   FB42 3A                 OUTL    P2,A
0058   FB43 BD 00              MOV     R5,#00H
0059   FB45 ED 45       EP1    DJNZ    R5,EP1
0060   FB47 53 BF              ANL     A,#0BFH
0061   FB49 3A                 OUTL    P2,A            ;RESET BIT 6
0062   FB4A 83                 RET
0063   FB4B             ;
0064   FB4B             ; GOTO POSITION
0065   FB4B             ;
0066   FB4B 43 80       GOTO   ORL     A,#80H
0067   FB4D 39                 OUTL    P1,A
0068   FB4E 27                 CLR     A
0069   FB4F 3A                 OUTL    P2,A
0070   FB50 74 3F              CALL    EPULSE
0071   FB52 83                 RET
0072   FB53             ;
0073   FB53             ; TABLE
0074   FB53             ;
0075   FB53 50 52 4F 4A TAB    .TEXT   "PROJ"
0076   FB57 45 43 54 20        .TEXT   "ECT "
0077   FB5B 54 45 53 54        .TEXT   "TEST"
0078   FB5F 2D 4C 43 44        .TEXT   "-LCD"
0079   FB63             ;
0080   FB63                    .END
0081   FB63
tasm: Number of errors = 0
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

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.

HANDOK

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

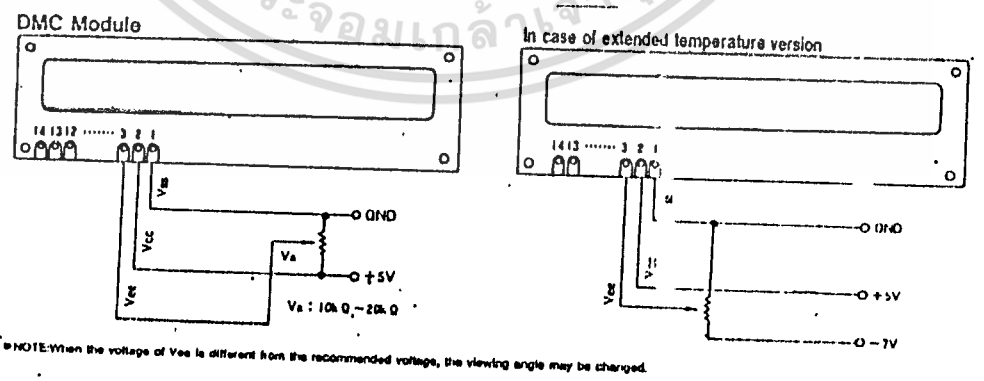
นอกจากนี้ LCD MODULE (HD44780) นี้ จะยังมีส่วนหนึ่งของ CHARACTER GENERATOR ที่เราสามารถเขียนข้อมูลในการเกิดตัวอักษรขึ้นได้เอง จากตารางตัวอักษร 5X7 DOT นั้นจะเห็นว่าคือ ตำแหน่งในตาราง OOH ถึง 07H ส่ ตำแหน่ง 08H-0FH ... เป็นต้อนแห่ง เดียวกับ OOH-07H จะเห็นว่า CHARACTER GENERATOR 8 ตัวที่เราสามารถเปลี่ยนข้อมูลตามที่เรา ต้องการได้และถ้าเป็นแบบ 5X10 DOT จะเขียนได้ 4 ตัวอักษร ซึ่งจากข้อมูลเท่านี้ เราก็สามารถ เขียนตัวอักษรสัญลักษณ์หรือลักษรภาษาไทยได้

การเขียนข้อมูล CHARACTER GENERATOR

เราสามารถเขียนข้อมูลได้โดยกำหนด ADDRESS ของ CG RAM ก่อน ซึ่งมีอยู่ได้ 64 ตำแหน่ง BIT 5-BIT 0 และเมื่อกำหนด ADDRESS แล้ว ก็จะสามารถเขียนข้อมูลลงใน CG RAM โดยเป็นลักษณะ BIT ต่อ BIT บนจอ 1 ตัวอักษร คือ 5X7 DOT นั้น จะใช้ข้อมูล BIT 4 ถึง BIT 0 ต่อ 1 BYTE เท่านั้น 1 ตัวอักษรจะใช้ข้อมูล 8 BYTE ด้วยกัน ได้ดูจากตารางประกอบไปด้วยและเมื่อเขียนข้อมูลลงใน CG RAM แล้วเวลาจะนำไปใช้งานก็ให้เขียนข้อมูล ใน DD RAM คือ ข้อมูลตำแหน่งในตาราง CHARACTER ให้ตำแหน่ง OOH-07H

ตัวอย่างโปรแกรมการเขียนข้อมูลตัวหนังสือภาษาไทยเข้าไปใน CG RAM ตัว ( ) และ ตัว ( ) เข้าไปใน CG RAM ตำแหน่งที่ OOH, 01H และ [illegible] และนำมาแสดงบนตาราง LCD โดยใช้ 2 บรรทัด ในการแสดงผล

สรุป การใช้งาน LCD MODULE นั้นที่สำคัญคือ ต้องเข้าใจในตัว CONTROLLER ของ LCD MODULE นั้น โดย CONTROLLER ทุกๆ บริษัท จะมีคำสั่งมาตรฐานที่เหมือนๆ กัน แต่ต่างกันที่



*NOTE:When the voltage of Vee is different from the recommended voltage, the viewing angle may be changed.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ขาต่างๆ ในการต่อใช้งาน HD44780

1. RS (REGISTOR SELECTION) จะเป็นขาเลือก REGISTOR ภายในซึ่งมีอยู่ 2 ตัว คือ INSTRUCTION REGISTOR (IR) และ DATA REGISTOR (DR) โดยถ้า เป็น 1 จะเป็นการเลือก DATA และถ้าเป็น 0 จะเป็นการเลือก INSTRUCTION

2. R/W (READ/WRITE) เป็นตัวเลือก ว่าจะเขียนหรืออ่านข้อมูลจากตัว IC โดยอ่านข้อมูล = 1, เขียนข้อมูล = 0

3. E (ENABLE SIGNAL) เป็นขาที่ควบคุมการรับส่งข้อมูล

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.

4. DB0-DB7 เป็นขารับส่งข้อมูลจากตัว MPU

5. VDD ไฟเลี้ยงตัววงจร

6. VSS เป็นขา GND

7. VO เป็นขารับ VOLTAGE ในการปรับ LCD ให้สว่างหรือมืด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

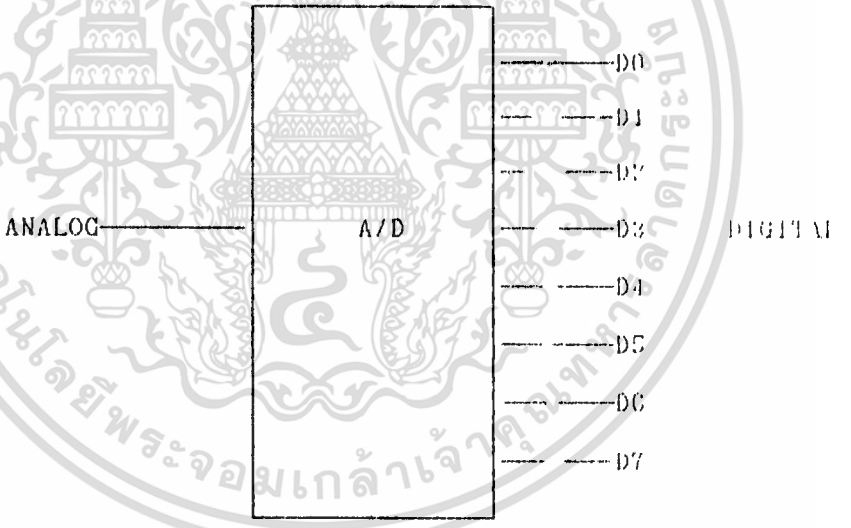
ANALOG TO DIGITAL

วัตถุประสงค์

1. เพื่อให้เข้าใจเกี่ยวกับการเปลี่ยนสัญญาณ ANALOG เป็นสัญญาณ DIGITAL
2. เพื่อเป็นแนวทางในการนำมาประยุกต์ใช้งาน

ทฤษฎี

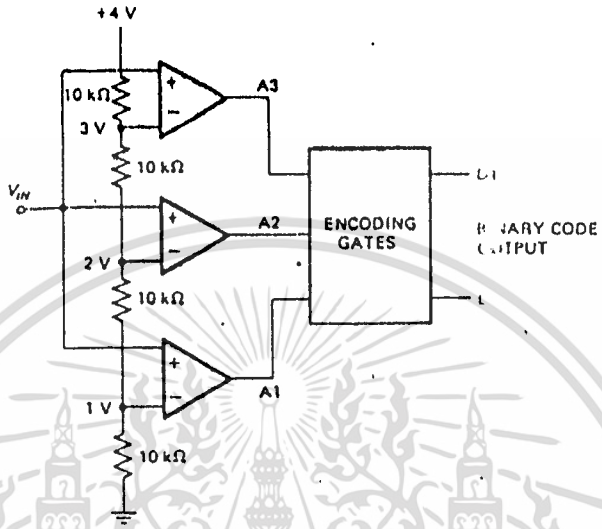
ลักษณะทั่วๆ ไปของ D/A และ A/D จะ ความเกี่ยวข้องกันอย่างมาก เมื่อเรา สามารถทำสัญญาณ DIGITAL เปลี่ยนเป็น ANALOG ได้แล้ว ในทางกลับกันเราก็จะเปลี่ยน สัญญาณ ANALOG เป็น DIGITAL ได้ด้วยซึ่งในจุดนี้จะทำให้เราทำการวัดค่าสัญญาณทาง ANALOG ได้เช่น ค่าความดัน, ค่าอุณหภูมิ โดยจะมีประโยชน์ อย่างมาก ต่องานทางด้านวิศวกรรม



คุณสมบัติของ A/D จะมีลักษณะเหมือนกับ D/A และมีคุณสมบัติพิเศษอีก 1 อย่าง คือ CONVERSION TIME ดังนั้นหมายถึงช่วงเวลาที่ A/D ใช้ในการแปลงสัญญาณ ANALOG เป็น DIGITAL ได้ 1 ค่า เมื่อเราพูดถึง A/D ที่ดี สิ่งนี้หมายถึง. จะมี CONVERSION TIME แก้ไข น้อยที่สุดเท่าที่จะน้อยได้ วงจร A/D มีหลักการได้หลายแบบ ดังจะกล่าวเป็นข้อๆ ดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

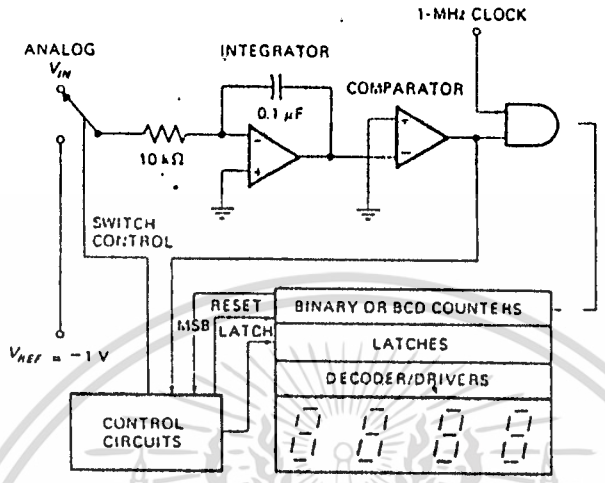
PARALLEL COMPARATOR A/D



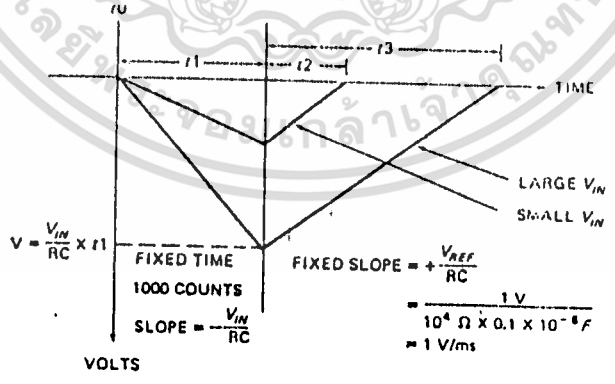
A/D แบบนี้ใช้หลักการของการเปรียบเทียบแรงดัน โดยสัญญาณที่จะนำมาแปลงนั้น จะต่อขนานกับตัวเปรียบทุกตัว สัญญาณจะเข้าที่ขาบวก ส่วนขาลบของตัวเปรียบเทียบจะต่อกับ VOLTAGE อ้างอิงของแต่ละระดับโดยได้มาจากการต่อ DIVIDER ของ R จากวงจรนี้ถ้าสัญญาณที่เข้ามามีค่าเกินจุดเปรียบเทียบใดๆ ก็จะทำให้ OUTPUT ของตัวเปรียบเทียบลันนั้นเป็น 1 เช่นถ้าสัญญาณเข้ามาก 2.6 V จะทำให้ A1 และ A2 มีค่าเป็น 1 OUTPUT A1, A2, A3 นี้ ไม่ได้อยู่ในรูปของ BINARY จึงจำเป็นต้องทำการแปลงโดยใช้วงจรทาง LOGIC ทั่วๆ ไปได้ จุดเด่นของ A/D แบบนี้ก็คือสามารถทำการ CONVERT ได้เร็วมาก ซึ่งมีค่าเวลาอยู่ที่ 20ns เท่านั้น แต่ข้อเสียก็คือจะต้องสิ้นเปลือง HARDWARE จำนวนมาก รูปนี้คือ A/D ขนาด 2 BIT เท่านั้นซึ่งต้องใช้ตัวเปรียบเทียบถึง3ตัวเขียนเป็นสูตรได้ดังนี้2~1โดย N คือจำนวน BIT ที่ต้องการนั้น หมายความว่า ถ้าเราต้องการ A/D ขนาด 8 BIT ก็จะต้องใช้ วเปรียบเทียบถึง 255 ตัว จึงทำให้ A/D ไม่ค่อยเป็นที่นิยมนัก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DUAL-SLOPE A/D



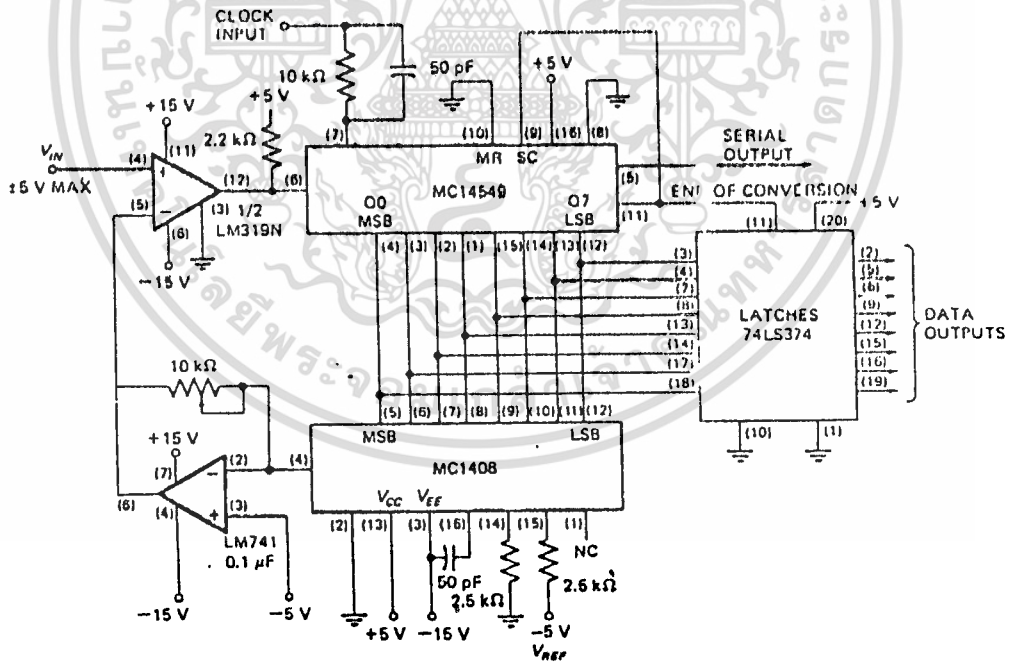
A/C แบบนี้มักจะใช้กับ DIGITAL VOLTMETER เนื่องจากมีราคาถูกและใช้งานได้ง่าย และเมื่อเริ่มต้นทำงานจะเริ่มต้นด้วยการ RESET ตัว COUNTERS ให้มีค่าเป็น 0 และป้อนสัญญาณ ANALOG (VIN) เท่ากับ INTEGRATOR เราสมมุติว่า VIN มีค่าเป็นบวก OUTPUT จาก INTEGRATOR จะแสดงได้ดังภาพนี้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อค่า OUTPUT จาก INTEGRATOR มีค่าเป็นลบ จะทำให้ COMPARATOR ทำงานส่งผลให้ ANDGATE เปิด และจะทำให้สัญญาณ CLOCK สามารถผ่านไปยังระบบ COUNTER ได้ค่า OUTPUT จาก INTEGRATOR นี้จะถูกกำหนด FIXED TIME จาก CONTROL CIRCUIT เมื่อถึงจุดนี้จะทำการ RESET COUNTER อีกครั้งและต่อสัญญาณ INPUT ของ INTEGRATOR เท่ากับ VREF ที่เป็นค่าลบซึ่งจะทำผลให้ OUTPUT ของ INTEGRATOR มี SLOPE ขึ้นเป็นบวก ค่า SLOPE ตั้งแต่ขึ้นกระทั่งถึงค่า 0 นี่เอง ที่จะให้ COUNTER นับความถี่ได้และค่าที่ได้จะสัมพันธ์กับ VIN เริ่มแรกที่วัดได้ A/D แบบนี้มักจะอยู่ในรูปของ IC สำเร็จรูปใช้กับงานด้าน DIGITAL VOLTMETER ซึ่งโดยทั่วไปจะมี CONVERSION TIME ประมาณ 300 มธ

SUCCESSIVE APPROXIMATION A/D



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

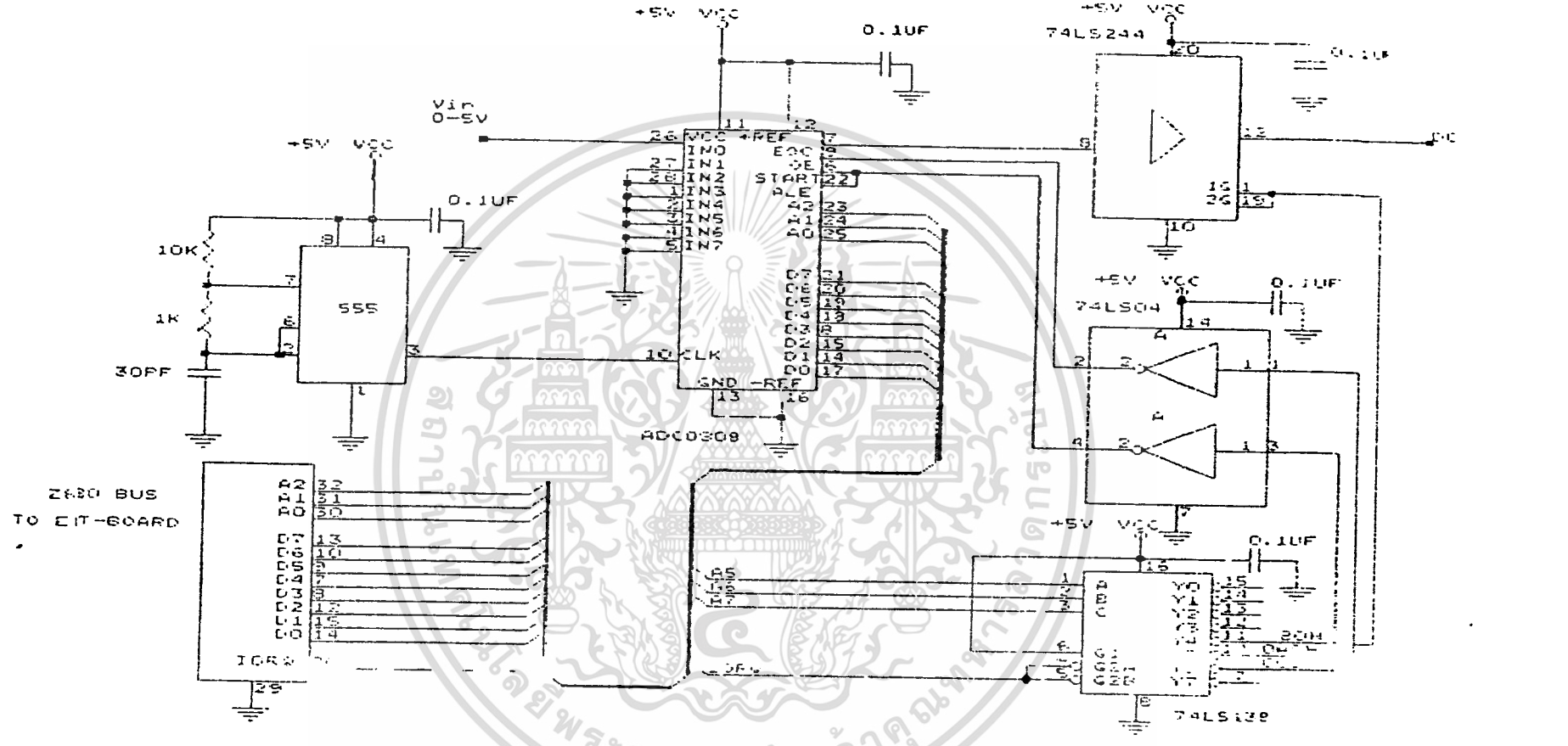
จากวงจรเมื่อเริ่มมี CLOCK เข้ามาที่ CLOCK INPUT จะทำให้ OUTPUT ของ MC14545 เป็น 1 ที่ปลายสูงและจะผ่านไปให้วงจร D/A คือ MC1408 ร่วมกับส่วนของสัญญาณ ซึ่งสัญญาณที่ได้จะผ่านไปเปรียบเทียบกับสัญญาณ VIN ที่ LM319N โดยถ้า VIN มากกว่าก็จะทำให้ OUTPUT ของวงจรเปรียบเทียบเป็น 1 ซึ่งก็จะทำให้วงจร SAR ทำงานต่อไป แต่ถ้า VIN น้อยกว่าก็จะทำ OUTPUT เป็น 0 ซึ่งเป็นการหยุดการทำงาน และให้สัญญาณ END OF CONVERSION ออกมาเพื่อนำ DATA ที่ได้มา LATCH ไว้ต่อไป

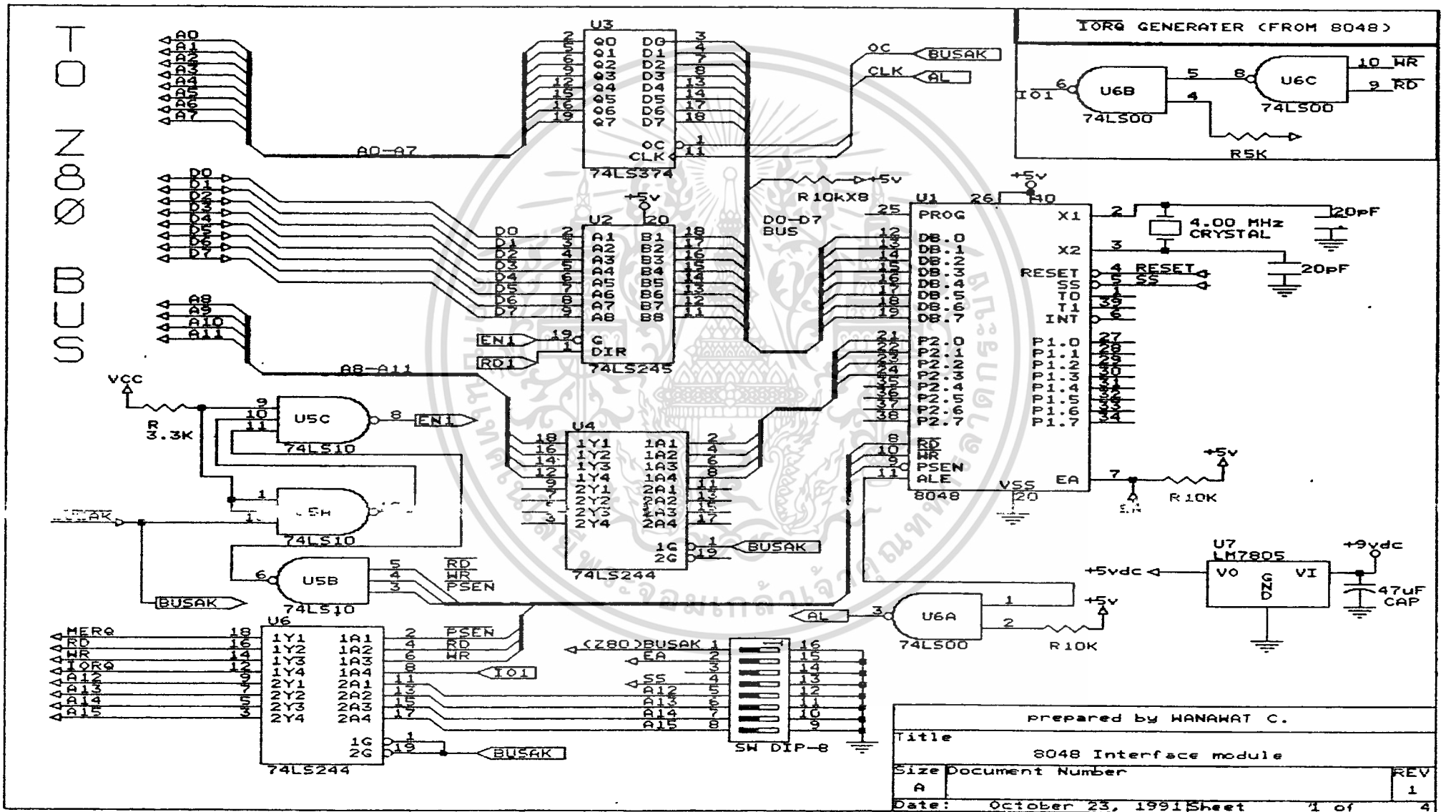
[illegible]

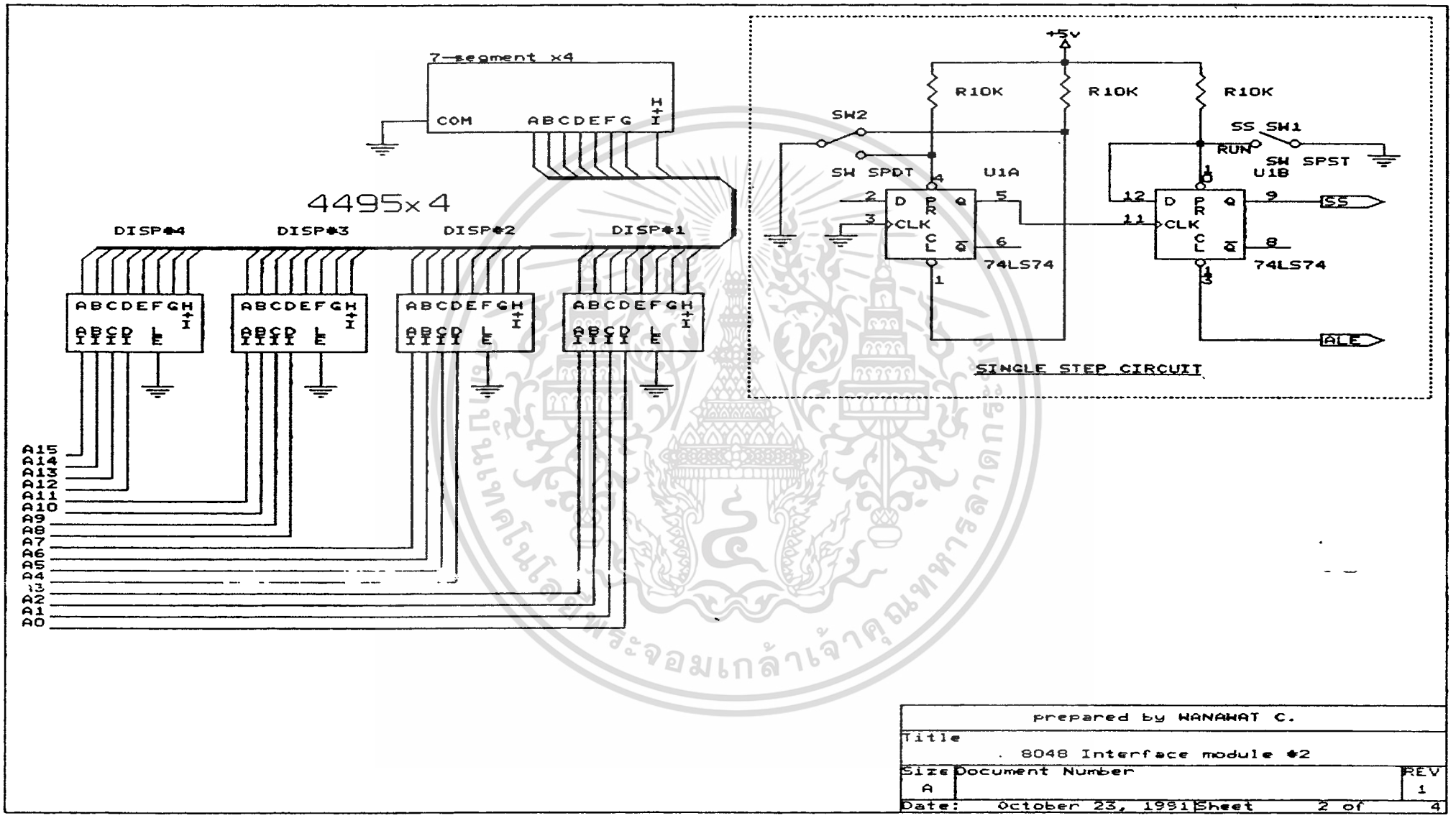
1. การใช้ A/D โดยทั่วไปจะใช้ขาม IC [illegible] ขนาด ในการทดลองนี้เราจะใช้เบอร์ ADC 0808 ประกอบวงจรดังรูป [illegible]

จากนั้นให้ป้อนโปรแกรมต่อไปนี้เข้าเครื่อง 8048 INTERFACE แล้วทดลอง RUN โปรแกรม โดยทำสัญญาณ 05V จาก SUPPLY ที่ปรับค่าได้มาเป็น VIN ให้สังเกตการเปลี่ยนค่า VIN แล้วไปดูที่ DISPLAY

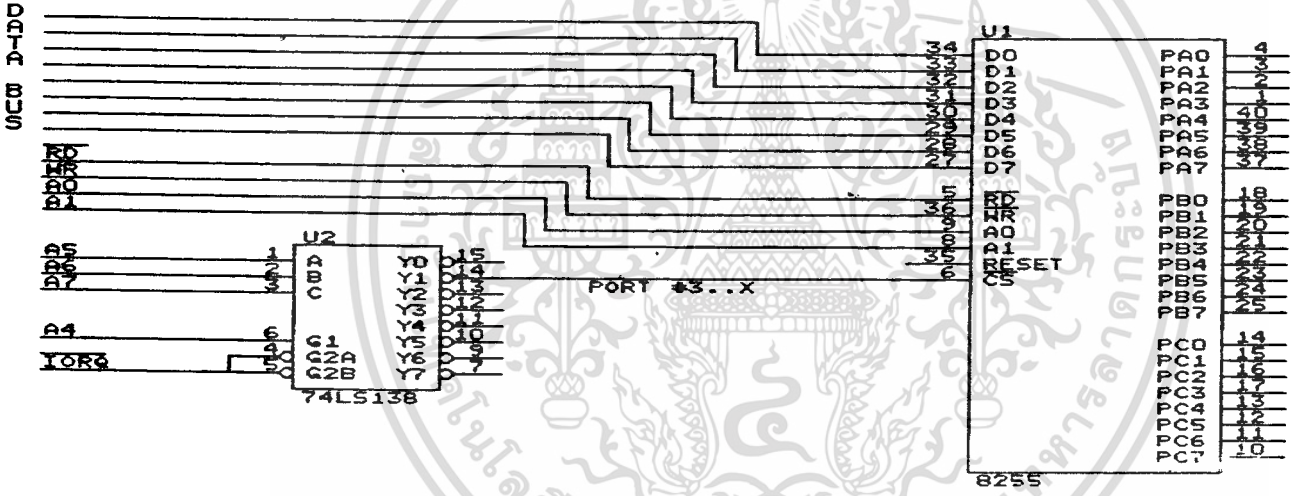
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


+5V VCC
0.1UF
Vin
0-5V
10K
1K
30PF
555
ADC0809
CLK
GND -REF
+REF
74LS244
74LS04
74LS138
Z80 BUS
TO EIT-BOARD


TO 8088 BUS
IORQ GENERATER (FROM 8048)
U6B
74LS00
U6C
74LS00
IO1
WR
RD
R5K
U3
74LS374
OC
CLK
BUSAK
AL
A0-A7
R 10kX8
+5v
D0-D7
BUS
U2
74LS245
EN1
RD1
A8-A11
VCC
R
3.3K
U5C
74LS10
U5B
74LS10
U5A
74LS10
U4
74LS244
U1
8048
PROG
X1
X2
RESET
SS
TO
T1
INT
EA
PSEN
ALE
VSS
4.00 MHz
CRYSTAL
20pF
20pF
R 10K
U6
74LS244
MERQ
RD
WR
IORQ
PSEN
IO1
(Z80)BUSAK
SW DIP-8
U6A
74LS00
R 10K
U7
LM7805
+5vdc
+9vdc
47uF
CAP
prepared by WANAWAT C.
Title
8048 Interface module
Size
A
Document Number
REV
1
Date: October 23, 1991
Sheet 1 of 4

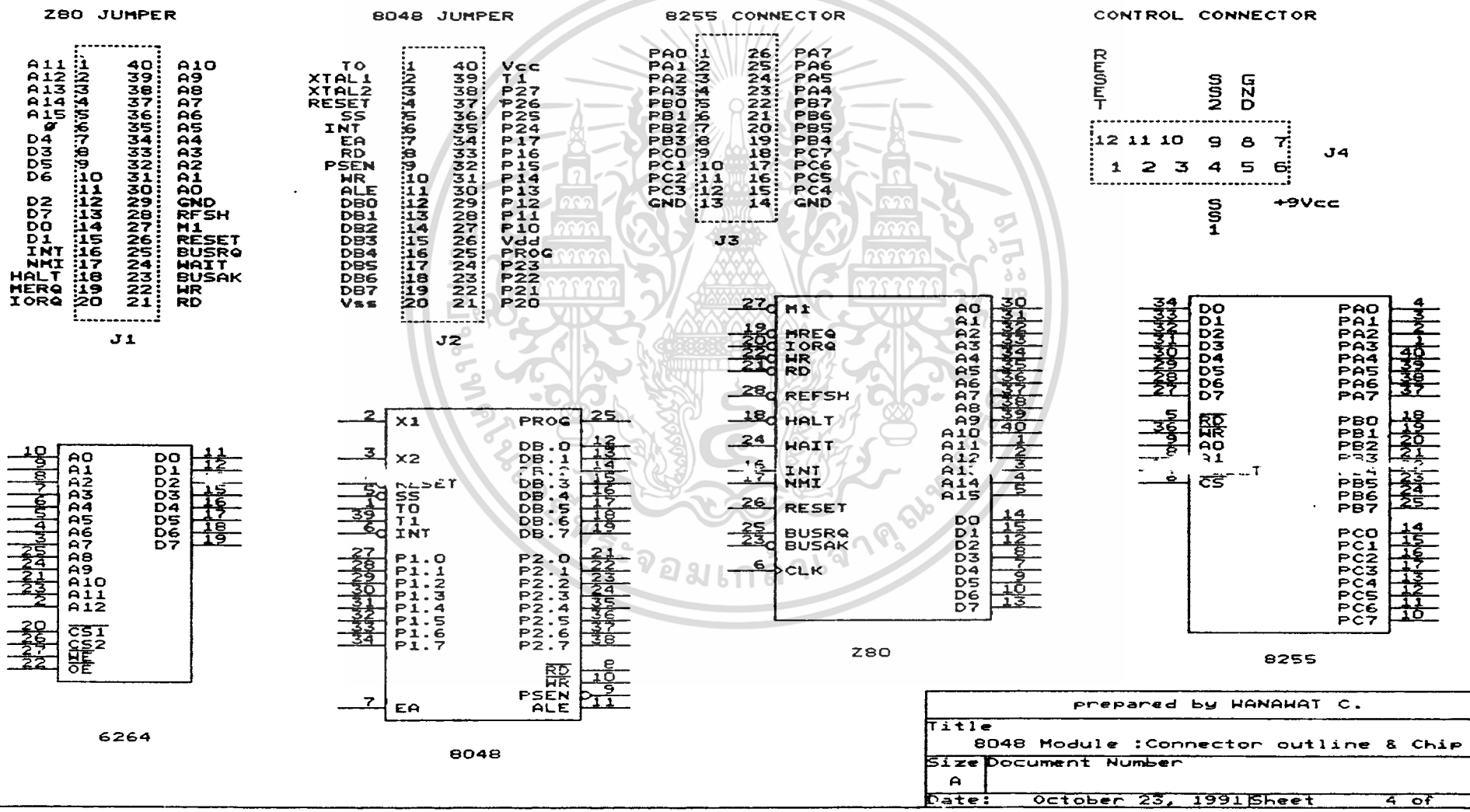


prepared by WANAWAT C.		
Title: 8048 Interface module #2		
Size: A	Document Number	REV: 1
Date: October 23, 1991	Sheet 2 of 4	

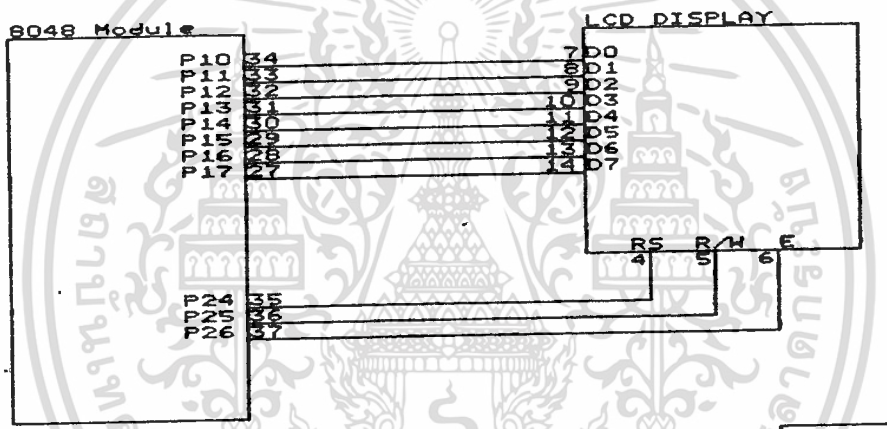


USER PORT

Title	8048 Module : USER PORT	
Size A	Document Number	REV 1
Date: October 23, 1991	Sheet 3 of 4	



prepared by WANAWAT C.		
Title: 8048 Module :Connector outline & Chip		
Size: A	Document Number	REV: 1
Date: October 23, 1991	Sheet 4 of 4	



E :enable pulse

R/W : [illegible] ,write = 0

RS :register instruction

;1 = data select

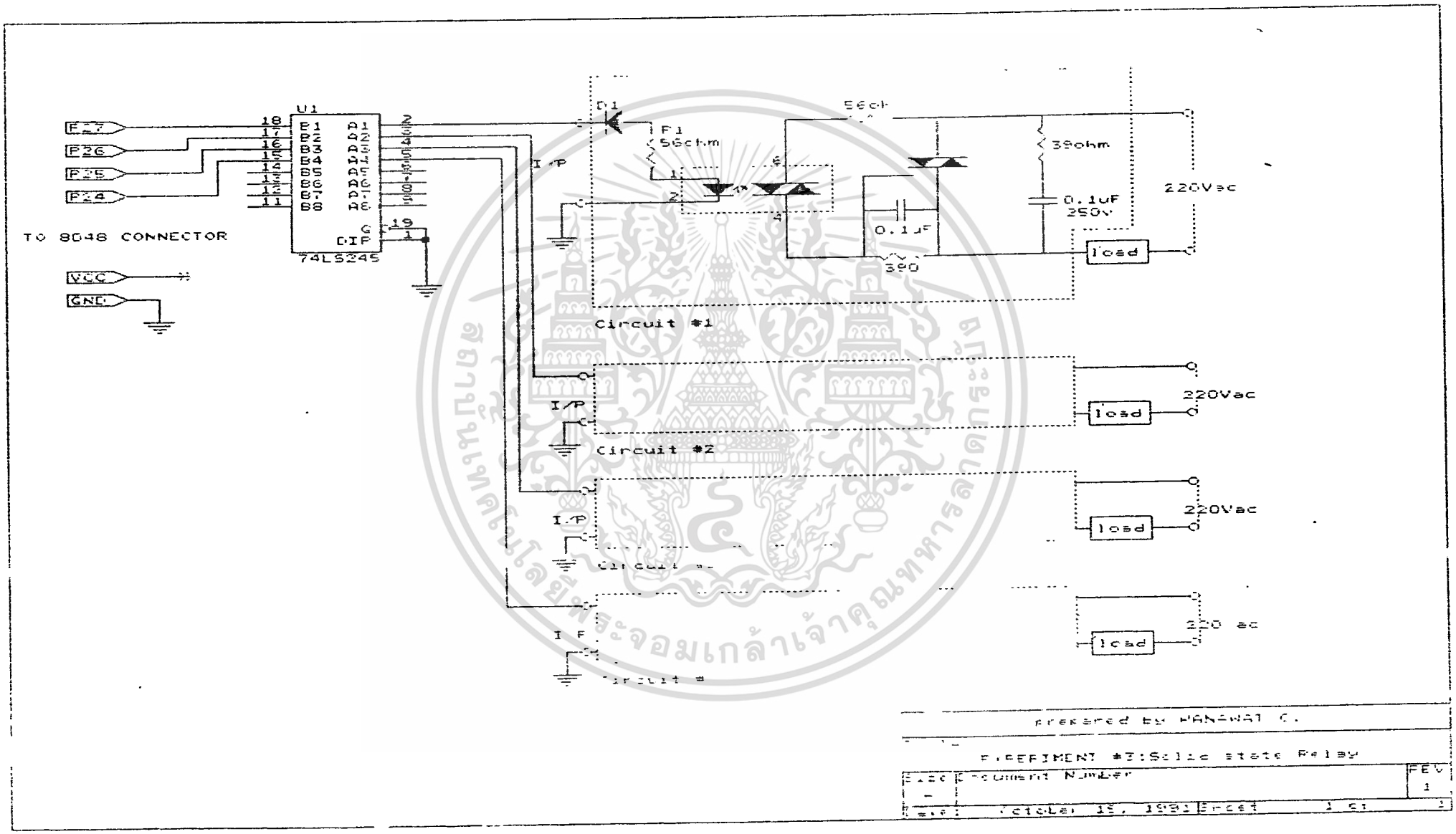
;0 = instruction select

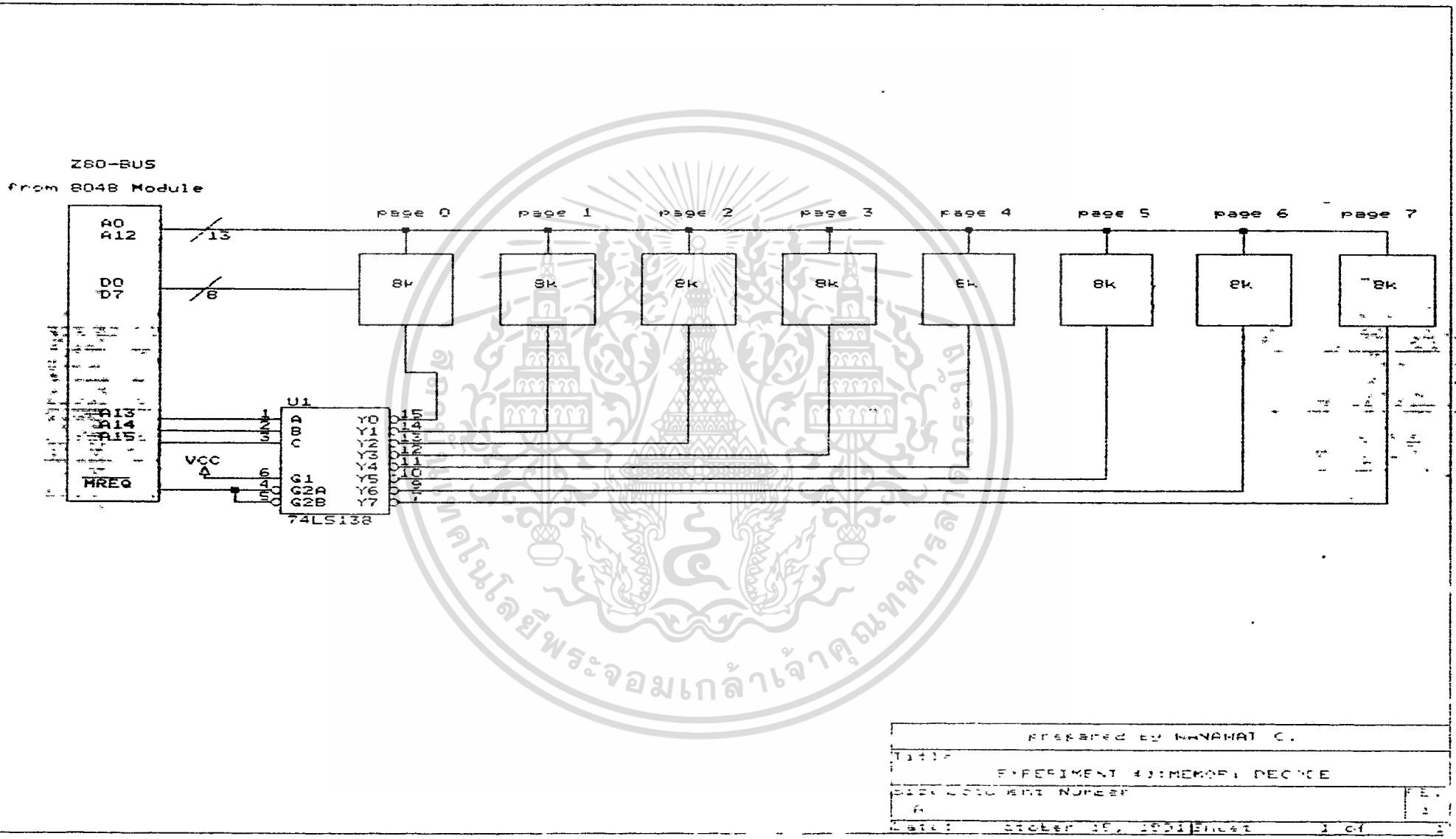
prepared by WANAWAT C.

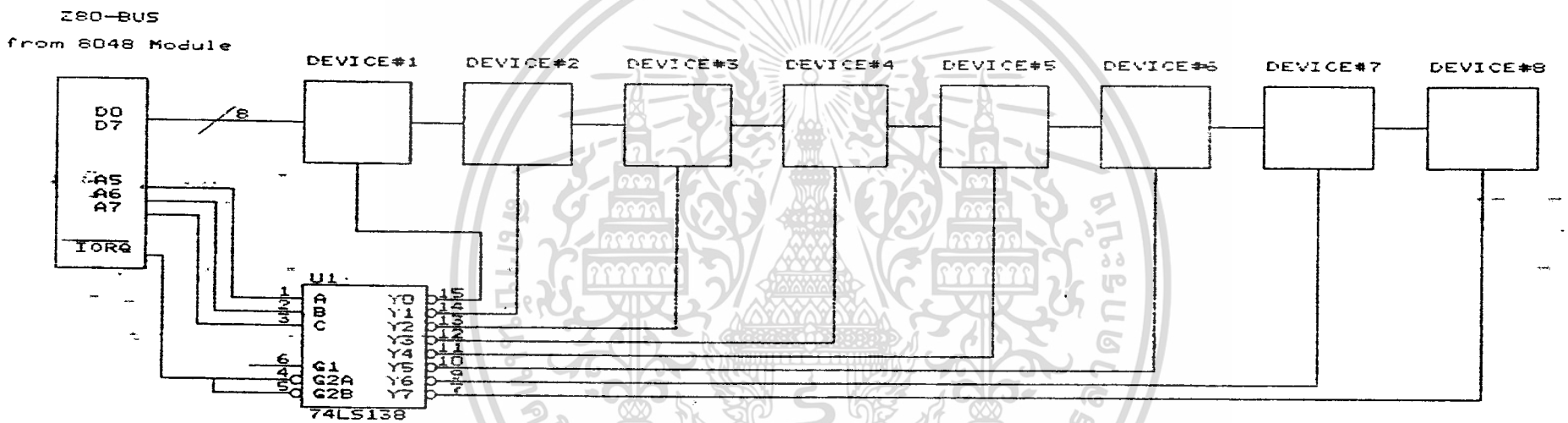
Title: EXPERIMENT #4 :LCD Display

Size: A | Document Number | REV: 1

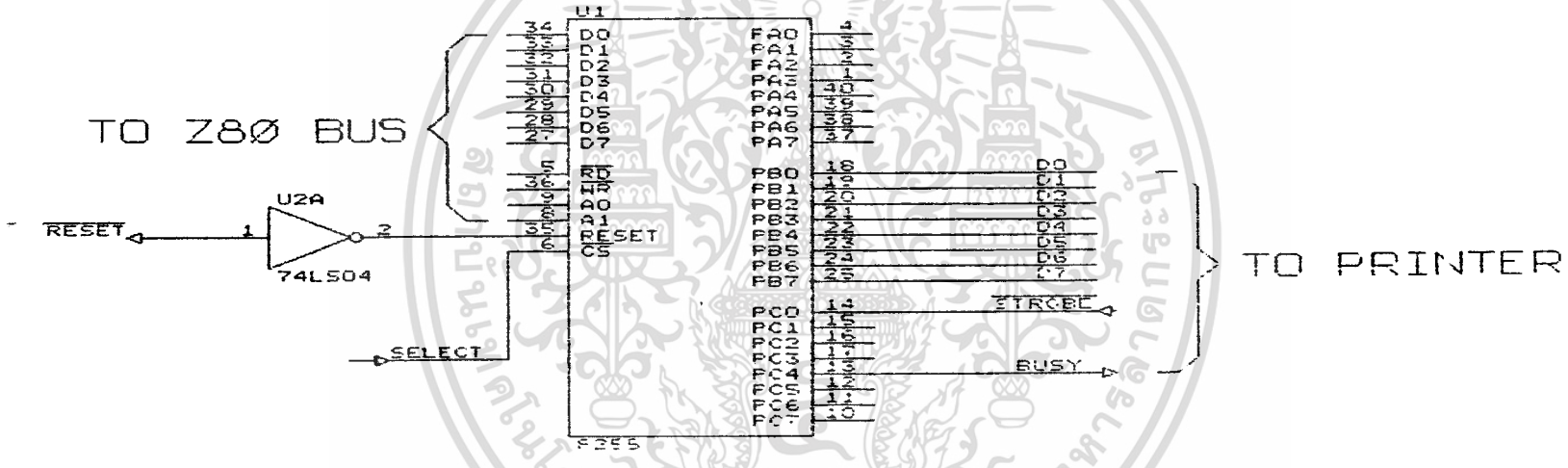
Date: October 17, 1991 | Sheet 1 of 1


U1
74LS245
DIP
TO 8048 CONNECTOR
VCC
GND
D1
56ohm
390ohm
0.1uF
250V
0.1uF
390
220Vac
load
I/P
Circuit #1
Circuit #2
220Vac
load
220Vac
load
220 ac
load
prepared by PANAWAT C.
EXPERIMENT #3:Solid state Relay
Document Number
REV
1
October 15, 1991
Sheet 1 of 1


Z80-BUS
from 8048 Module
A0
A12
/13
D0
D7
/8
MREQ
A13
A14
A15
U1
A
B
C
G1
G2A
G2B
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
VCC
74LS138
page 0
page 1
page 2
page 3
page 4
page 5
page 6
page 7
8k
Prepared by NAVAWAT C.
Title:
EXPERIMENT 4: MEMORY DECODE
Size: A
Number
Date: October 15, 1991
Sheet 1 of 1



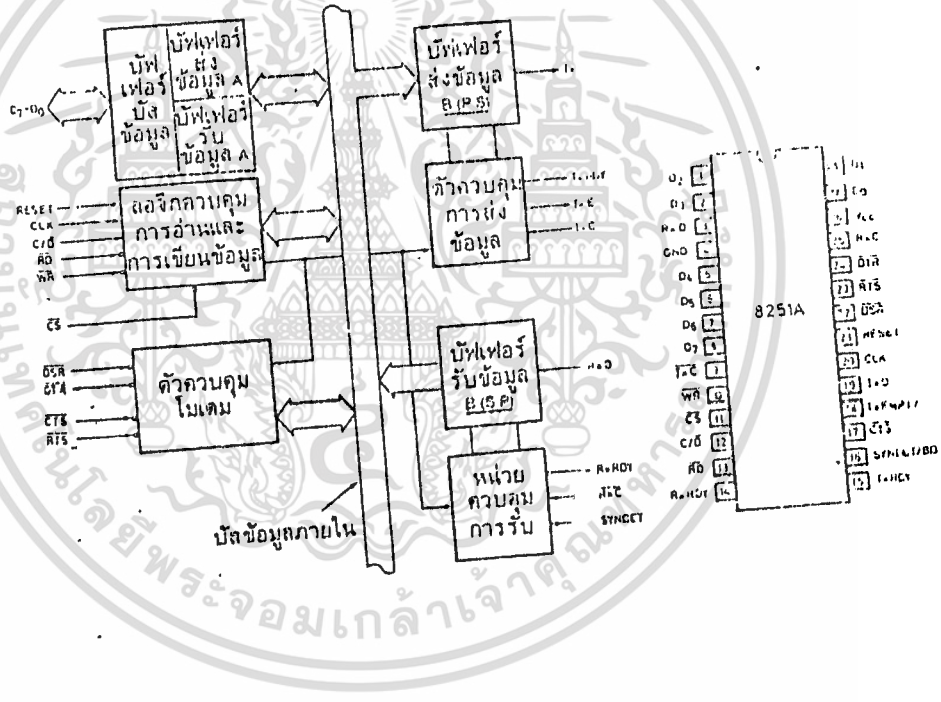
prepared by WANAWAT C.		
Title: EXPERIMENT #2: PORT DECODE		
Size: A	Document Number	REV: 1
Date: October 16, 1991	Sheet 1 of 1	



สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

รายละเอียดเกี่ยวกับ 8251 Chip Support

8251 เป็นชิบที่ทำหน้าที่เป็นตัวรับและส่งข้อมูล ระหว่างไมโครโปรเซสเซอร์ กับอุปกรณ์อินพุท เอาท์พุทแบบอนุกรม ชิบนี้ทำงานได้ทั้งโหมดอะซิงโครนัสและโหมดซิงโครนัส แต่ที่ประยุกต์ใช้กันมาก และใช้ในโครงงานนี้เป็นแบบอะซิงโครนัส ชิบนี้มีขนาด 28 ขา มีการจัดขาและโครงสร้างภายใน ดังรูป 1



รูปที่ 1 โครงสร้างภายในและการจัดขาของไอซี 8251

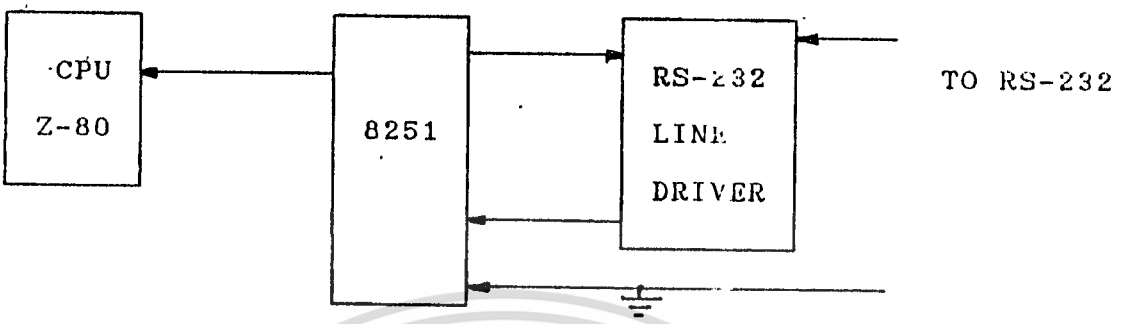
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

จากสัญญาณต่าง ๆ ในรูปที่ 1 สามารถสรุปหน้าที่ได้ดังตารางที่ 1

ชื่อสัญญาณ	หน้าที่โดยย่อ	ชนิดของสัญญาณ
D_0-D_7	บัสข้อมูล	สองทิศทาง
RESET	สัญญาณรีเซ็ต	อินพุท
CLK	สัญญาณนาฬิกา	อินพุท
C/D	สัญญาณเลือก Control/data	อินพุท
RD	สัญญาณแสดงการอ่าน	อินพุท
WR	สัญญาณแสดงการเขียน	อินพุท
CS	สัญญาณเลือกชิป 8251	อินพุท
DSR	Data Set ready	อินพุท
DTR	data terminal ready	เอาท์พุท
CTS	clear to send	เอาท์พุท
RTS	request to send	เอาท์พุท
TXD	ข้อมูลเอาท์พุทแบบอนุกรม	เอาท์พุท
TXRDY	พร้อมจะรับข้อมูลไปส่ง	เอาท์พุท
TXEMPTY	บัฟเฟอร์ส่งข้อมูลว่าง	เอาท์พุท
TXC	สัญญาณนาฬิกากำหนดการส่ ข้อมูล	อินพุท
RXD	ข้อมูลอินพุทแบบอนุกรม	อินพุท
TXRDY	ข้อมูลพร้อมที่จะส่งไปยังบัสข้อมูล	อินพุท
RXC	สัญญาณนาฬิกากำหนดการรับ	อินพุท
SYNDET/BD	Synchronous detect/ break detect	อินพุท
V_{cc} GND	ไฟเลี้ยง	สองทิศทาง
TXRDY	พร้อมจะรับข้อมูลไปส่ง	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การใช้งาน 8251



รูปที่ 2 แสดงการติดต่อระหว่าง PC กับ CPU Z-80 โดยใช้ IC 8251

ลำดับการทำงาน

1. สัญญาณ RESET เป็นการเริ่มต้นก่อนสัญญาณใด ๆ
2. ทำคำสั่งเลือกกระบวนการทำงาน (MODE SELECT)
3. ทำคำสั่งควบคุม (COMMAND)

ขั้นที่	สัญญาณ	รหัสคำสั่งหรือข้อมูลที่ส่งมาให้ 8251
1	C/D = 1	MODE INSTRUCTION : เลือกโหมด
2	C/D = 1	COMMAND INSTRUCTION : รหัสควบคุม
3	C/D = 0	DATA
4	C/D = 1	COMMAND INSTRUCTION : ทำเมื่อต้องการเปลี่ยนรหัส
5	C/D = 0	DATA
6	C/D = 1	COMMAND INSTRUCTION : ควบคุมแต่ไม่เปลี่ยนโหมด

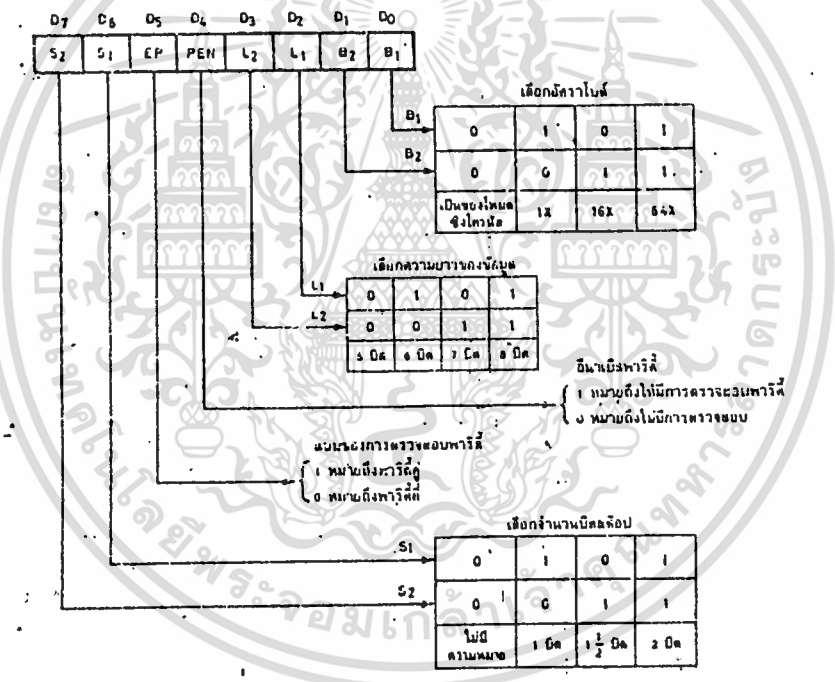
ตาราง 2 ลำดับที่ซีพียูติดต่อกับ 8251 ในโหมดอะซิงโครนัส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หลังจากใช้สัญญาณ RESET แล้ว ซีพียูต้องออกคำสั่งเลือกโหมด จบแล้วจะตามด้วยรหัสคำสั่งควบคุม สำหรับในตารางนั้นตอนที่ 3, 5 หมายถึง การรับหรือการส่งข้อมูลต่อเนื่องกันไป จนกว่าจะเปลี่ยนโหมดหรือรหัสควบคุม ส่วนขั้นตอนที่ 4 และ 6 จะทำถ้าต้องการเปลี่ยนรหัสควบคุม แต่ไม่เปลี่ยนแปลงโหมด

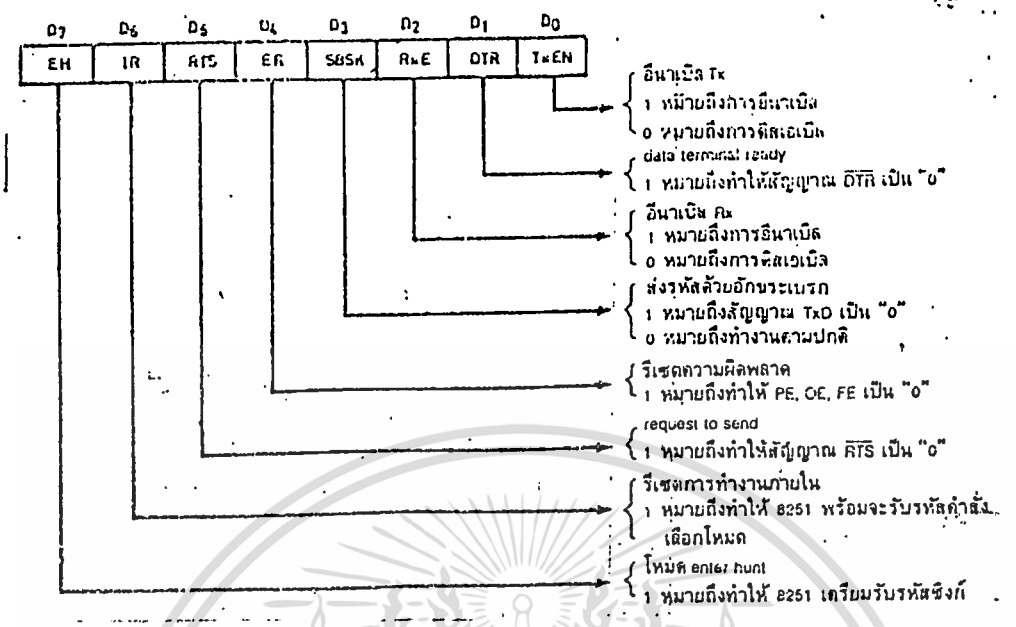
<u>คำสั่งโหมดและรหัสควบคุม</u>

ลักษณะคำสั่งเลือกโหมดทั้ง 8 บิต ของคำสั่งที่ซีพียูส่งมาให้ 8251 เพื่อเริ่มต้นการทำงาน จะมีความหมายดังรูปที่ 2 สำหรับโหมดแบบอะซิงโครนัส ต่อจากนั้นจึงเป็นการส่งรหัสควบคุม ซึ่งมีความหมายดังรูปที่ 3



<u>รูปที่ 2</u> แสดงรหัสเลือกโหมดแบบอะซิงโครนัส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



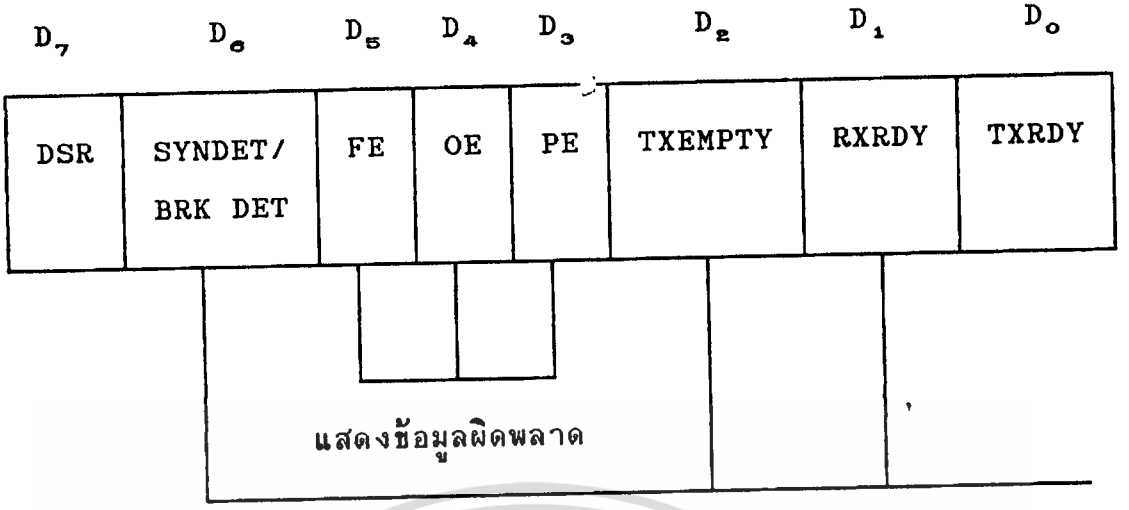
รูปที่ 3 แสดงรหัสคำสั่งของแต่ละบิตที่ควบคุมการทำงานภายในของ 8251

สังเกตที่การเลือกอัตราโบด์ของบิต D_0 และ D_1 ในรูป 2 ความหมายของ 16X และ 64X คือ การนำความถี่ของสัญญาณฐานเวลา TXC หรือ RXC มาหารด้วย 16X และ 64X ตามลำดับ แล้วนำไปกำหนดอัตราโบด์ ซึ่งมีประโยชน์ในกรณีที่ต้องใช้วงจรกำเนิดสัญญาณนาฬิกาความถี่สูง ๆ แล้วต้องนำมาหารให้เหลือความถี่เท่ากับอัตราโบด์ การเลือก 16X และ 64X จะช่วยลดวงจรหารให้น้อยลงได้ และวงจรกำเนิดความถี่ดังกล่าว ต้องมีความเที่ยงตรงสูง จึงนิยมใช้คริสตอลเป็นตัวกำหนดความถี่

ไบต์แสดงสถานะ

ไบต์แสดงสถานะนี้มีขนาด 8 บิตแต่ละบิตจะแสดงสถานะของ 8251 โดยมีความหมายเป็น ON กับ OFF ความหมายของแต่ละบิต เป็นดังรูปที่ 4

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4 ไบต์แสดงสถานะของ 8251

ความหมายของแต่ละบิตในรูปที่ 4 มีดังนี้

TXRDY (D_0) จะเป็น "1" ทันทีที่บัฟเฟอร์ส่งข้อมูล A ว่าง ซึ่งหมายถึง พร้อมที่จะรับข้อมูลไปส่งและยังแสดงถึงสัญญาณ TXRDY ที่ขา 15 อีกด้วย โดยถ้าสัญญาณ CTS เป็น "0" สัญญาณ TXRDY ที่ขา 15 จะมีลอจิกเหมือนขา 15 แต่ถ้าสัญญาณ CTS เป็น "1" สัญญาณ TXRDY ที่ขา 15 เป็น "0" ทันที

RXRDY แสดงถึงได้รับข้อมูลเข้ามาครบไบต์แล้ว ให้ซีพียูมาอ่านไปได้ นอกจากนี้ RXRDY ยังมีลอจิกเหมือนกับสัญญาณ RXRDY ที่ขา 14 อีกด้วย

TXEMPTY แสดงลอจิกของสัญญาณ TXEMPTY ที่ขา 18 โดยจะเป็น "1" เมื่อ 8251 ว่างหรือไม่มีข้อมูลจะส่ง แต่จะกลับเป็น "0" เมื่อได้รับข้อมูลจากซีพียูในขณะที่ภาคส่งได้รับการอีนาเบิล

PE จะเป็น "1" เมื่อเกิดการผิดพลาดจากการตรวจสอบบิตพาริตี้ และจะ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถูกรีเซตเป็น "0" โดย ER ในรหัสควบคุมจากซีพียู ซึ่ง PE นี้มาจาก Parity error

OE จะเป็น "1" ถ้าซีพียูไม่มาอ่านข้อมูลที่ 8251 ซึ่ง 8251 จะรับ OE เข้ามาไว้ในบัฟเฟอร์ รับข้อมูล A แล้วมีข้อมูลใหม่เข้ามาอีก และ OE จะถูกรีเซตเป็น "0" โดย ER ในรหัสควบคุม OE มาจาก OVERRUN ERROR

FE ใช้ในกระบวนการอะซิงโครนัสเท่านั้น ถ้าข้อมูลที่รับมาขาดบิตสต็อปไป FE จะเป็น "1" และ FE จะถูกรีเซตเป็น "0" โดย ER ในรหัสควบคุม ซึ่ง FE นี้มาจาก framing error

ทั้ง PE, OE และ FE จะไม่ขัดขวางการทำงานของ 8251 แต่ประการใด เป็นเพียงการเตือนให้ทราบถึงการเกิดความผิดพลาดแล้วเท่านั้น

4. การรับข้อมูลแบบอะซิงโครนัส

สัญญาณที่เกี่ยวข้องกับการรับข้อมูลแบบอะซิงโครนัส แสดงได้ดังรูปที่ 5

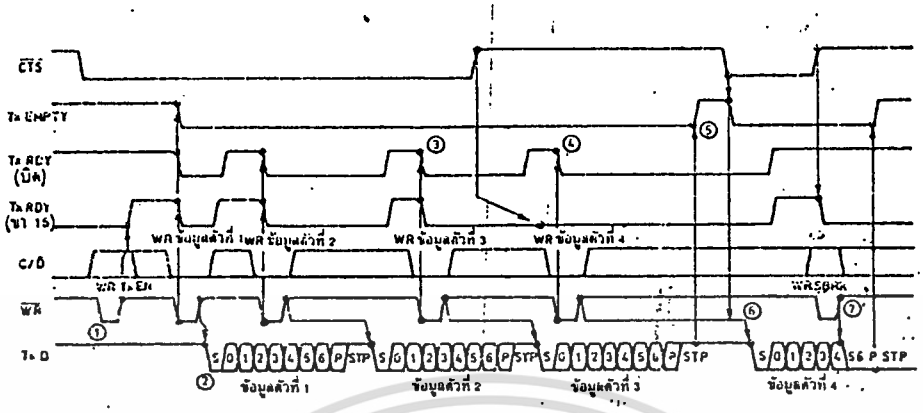
จากรูปที่ 5 เป็นตัวอย่างของข้อมูลขนาด 6 บิต บิตพาริตี้ 1 บิต บิตสต็อป 2 บิต กับบิตสตาร์ตอีก 1 บิต รวมเป็นความยาวทั้งหมด 10 บิต ลำดับขั้นการรับข้อมูลแบบอะซิงโครนัส มีดังนี้

1. ซีพียู ส่งรหัสควบคุม RXEN (D_2) มายัง 8251 ทำให้สัญญาณ C/D เป็น "1" เพื่อให้ 8251 รับข้อมูลได้

2. ข้อมูลเริ่มมีเข้ามายัง 8251 ทางบิต RXD เริ่มด้วยบิตสตาร์ตตามด้วยข้อมูลและบิตพาริตี้ จนลงด้วยบิตสต็อป 2 บิต

3. ที่บิตสต็อปนี้เองเป็นการบอกถึงการจบ 1 ตัว อักขระ RXDY จึงเป็น "1" แสดงว่าได้รับข้อมูลเข้ามาครบ 1 ตัวแล้ว ซีพียูจะทำการอ่านมายัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 5 แสดงรหัสเลือกโหมดแบบอะซิงโครนัส

8251 (C/D เป็น "0") แสดงว่ารับข้อมูลไปได้เลย และเมื่อซีพียูอ่านข้อมูลไปแล้ว RXDY จะกลับเป็น "0"

4. ข้อมูลตัวที่ 2 ถูกส่งเข้ามาอีก บิต RXRDY จึงเป็น "1" อีก

5. สมมติว่า สัญญาณ RXRDY เป็น "1" แล้ว ซีพียูยังไม่ยอมมาอ่านข้อมูลไปแล้วมีข้อมูลตัวที่ 3 เข้ามาอีก ทำให้บิต OE ในไบต์แสดงสถานะเปลี่ยนเป็น "1" แสดงให้เห็นว่า มีข้อมูลบางตัวสูญหายไป เพราะถูกข้อมูลตัวถัดไปส่งมาทับ

6. ซีพียูทำการอ่านข้อมูลจาก 8251 จะได้ข้อมูลตัวที่ 3 (ข้อมูลตัวที่ 2 ถูกทับไปเสียแล้ว)และบิต OE จะไม่ขัดขวางการทำงานของซีพียูแต่ประการใด แต่ซีพียูจะรู้ว่าเกิดข้อผิดพลาดขึ้นแล้ว ได้โดยการอ่านไบต์แสดงสถานะ ซึ่งไม่ได้แสดงไว้ในรูปที่ 5

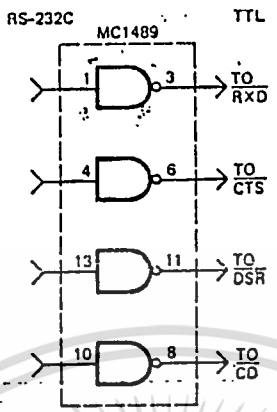
7. ซีพียูสามารถรีเซตบิต OE ในไบต์แสดงสถานะได้ โดยส่งรหัสควบคุม ER (D_4) มารีเซตได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8. ในช่วงเวลานี้ตรวจพบว่า ข้อมูลหรือตัวอักขระที่รับมา เกิดขาดหายไป 2 ตัว คือ รับได้เป็น "0" ตลอดช่วงความยาว 2 ไบต์ติดต่อกัน จึงเกิดสถานะเบรกหรือหยุด ทำให้สัญญาณ BD (ขา 16) และบิต D6 ในไบต์แสดงสถานะเป็น "1" ทันที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

LINE DRIVER RS-232C



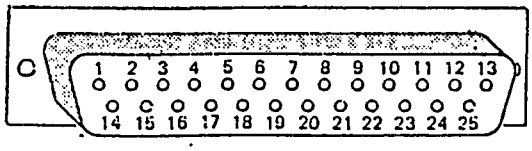
PIN 14 = +5 V.

PIN 7 = GND

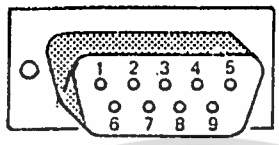
รูปแสดงการเปลี่ยนระดับสัญญาณใน RS-232 ไปเป็น LOGIC TTL.

ในการนำ EMULATOR มา INTERFACE กับ MICRO COMPUTER PORT RS-232C โดยมีตำแหน่งขา ดังนี้ (อาจเป็นได้ทั้งแบบ DB-9P หรือ DB-25)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(a)



(b)

รูปแสดง Connector ที่ใช้สำหรับ RS-232C ทั้งแบบ DB-25P MALE CONNECTOR และแบบ DB-9P DIN CONNECTOR

ตารางแสดงรายละเอียดขาสัญญาณทั้งแบบ 9 ขา และ 25 ขา

ชนิด 9 PINS	ชนิด 25 PINS	ชื่อเรียกทั่วไป	ชื่อเรียกทาง RS-232	คำอธิบาย	ทิศทางสัญญาณ
	1		AA.	PROTECTIVE GND	-
3	2	TXD	BA.	ส่งข้อมูล	อินพุท
2	3	RXD	BB.	รับข้อมูล	เอาท์พุท
7	4	RST	CA.	REQUEST TO SEND	อินพุท
8	5	CTS	CB	CLEAR TO SEND	เอาท์พุท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชนิด 9 PINS	ชนิด 25 PINS	ชื่อเรียกทั่วไป	ชื่อเรียกทาง RS-232	คำอธิบาย	ทิศทางสัญญาณ
6	6	DSR	CC	DATA SET READY	เอาท์พุท
5	7	GND	AB	SIGNAL GROUND	-
1	8	CD	CF	RECEIVED LINE SIGNAL DETECTOR	เอาท์พุท
	9		-	เพื่อใช้สำหรับการทดสอบ	-
	10		-	เพื่อใช้สำหรับการทดสอบ	-
	11			ไม่ใช้	-
	12		SCF	SECONDARY RECEIVED LINE SIGNAL DETECTOR	เอาท์พุท
	13		SCB	SECONDARY CLEAR TO SEND	เอาท์พุท
	14		SBA	เส้นส่งข้อมูลเส้นที่ 2	อินพุท
	15		DB	DCE SOURCE	เอาท์พุท
	16		SBB	เส้นรับข้อมูลเส้นที่ 2	เอาท์พุท
	17		DD	DCE SOURCE	เอาท์พุท
	18			ไม่ใช้	-
	19		SCA	SECONDARY REQUEST TO SEND	อินพุท

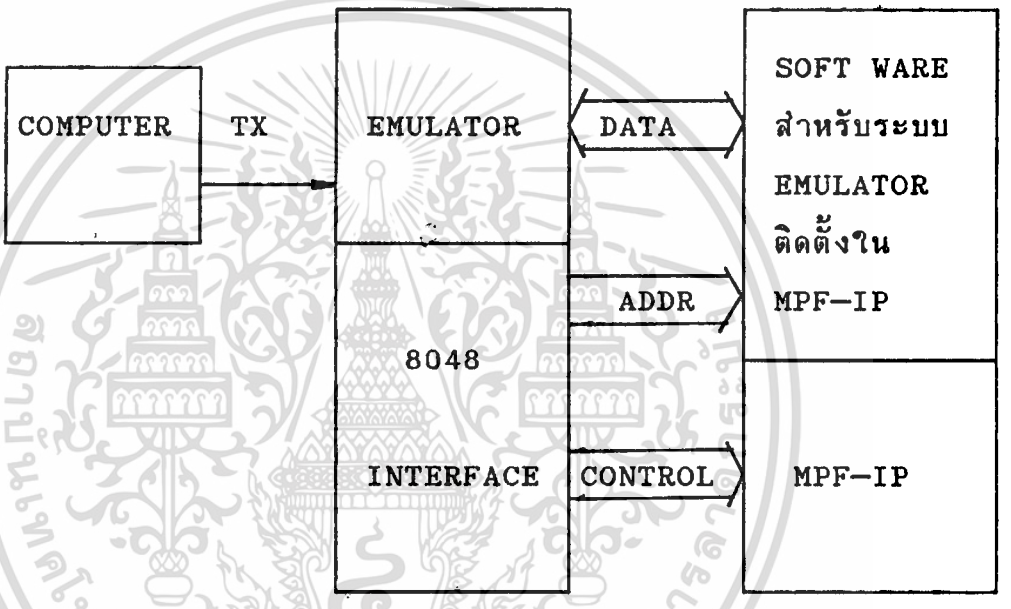
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชนิด 9 PINS	ชนิด 25 PINS	ชื่อเรียกทั่วไป	ชื่อเรียกทาง RS-232	คำอธิบาย	ทิศทางสัญญาณ
4	20	DTR	CD	DATA TERMINAL READY	อินพุท
9	21		CG	SIGNAL QUALITY DETECTOR	เอาท์พุท
	22		CE	RING INDICATOR	เอาท์พุท
	23		CH/CI	DATA SIGNAL RATE SELECTOR	อินพุท/เอาท์พุท
	24		DA	DTE SOURCE	อินพุท
	25			ไม่ใช้	-

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การ LINK MPF-IP กับ PC

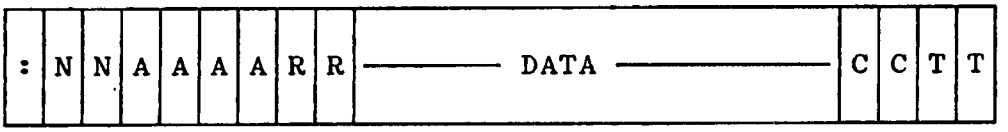
จากโครงสร้างของ MPF-IP สามารถออกแบบให้ LINK กับ COMPUTER โดยผ่าน EMULATOR ได้ โดยเขียนเป็น BLOCK DIAGRAM ได้ดังนี้



EMULATOR หรือส่วนของ #8251 ใช้สื่อสารระหว่าง COMPUTER กับ กับ MPF-IP โดยโปรแกรมที่ใช้ควบคุมส่วนนี้ เก็บไว้ใน EPROM ที่ ADDRESS 2000H ซึ่งโปรแกรมส่วนนี้จะเป็นตัวจัดการ ตั้งแต่การรับข้อมูลที่ส่งมาจาก COMPUTER แล้วจัดเก็บข้อมูลลงใน RAM ในตำแหน่งที่ ORIGIN รวมกับค่า OFFSET ที่ผู้ใช้กำหนด

INTEL HEX FORMAT เป็นรูปแบบของข้อมูลที่ส่งมาจาก COMPUTER ผ่าน SERIAL PORT (COM1,COM2) โดยมีรูปแบบดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เมื่อ : = RECORD START CHARACTER

NN = BYTE COUNT (HEX)

AAAA = ADDRESS OF FIRST BYTE (HEX)

RR = RECORD TYPE (HEX,00 EXCEPT FOR LAST RECORD WHICH IS 01)

CC = CHECK SUM (HEX)

HH = DATA (ASCII)

TT = LINE TERMINATOR(CARRIAGE RETURN,LINE FEED)

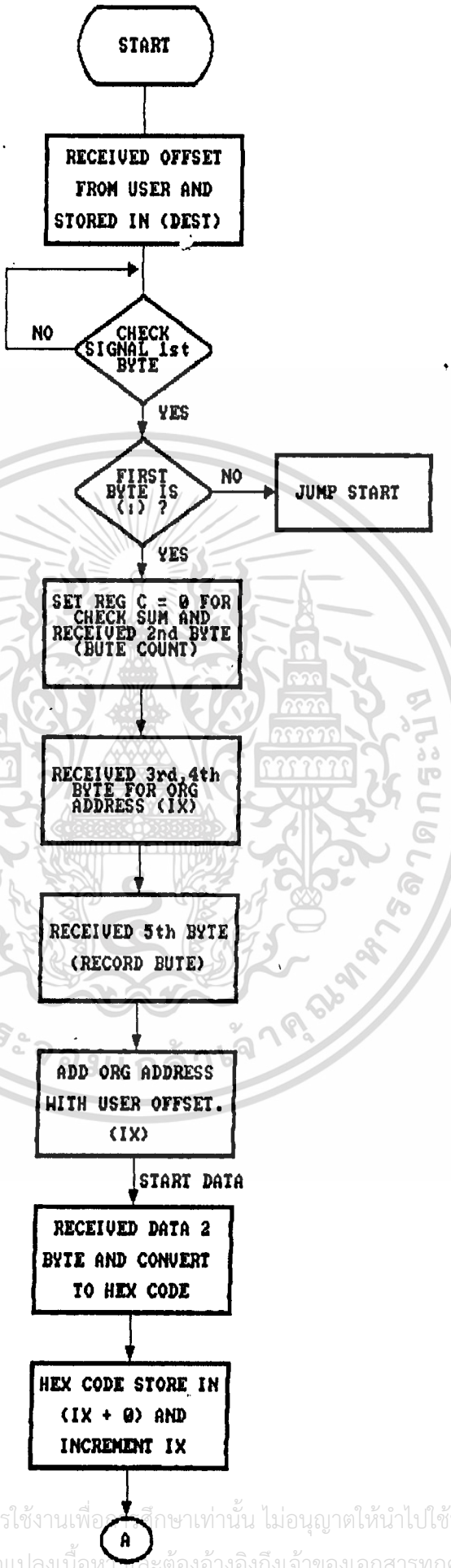
โดย sum = BYTE - COUNT + ADDRESS_HI + ADDRESS_LO + RECORD - BYTE + (SUM OF DATA BYTE)

CHECKSUM = [(-SUM)AND FFH]

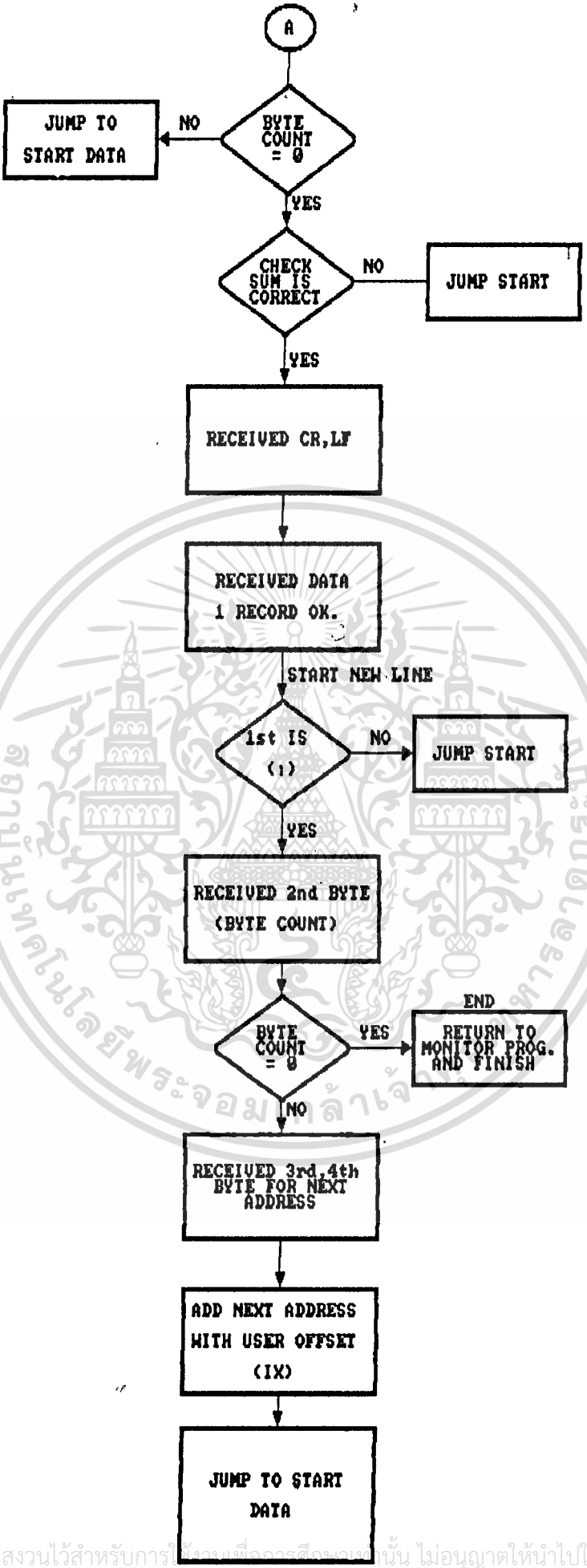
เนื่องจาก DATA (HH) ที่ใช้ส่งนี้เป็น ASCII CODE ดังนั้นจึงต้องทำการ CONVERTเป็น HEX CODEก่อนจึงสามารถใช้งานได้ ค่า BYTE COUNT(NN) จะใช้เป็นตัวกำหนดความยาวของข้อมูล (HH) ใน 1 บรรทัด (1 RECORD) ค่า AAAA จะเป็น ADDRESS ORG ของโปรแกรมที่ถูกส่ง CHECKSUM ใช้สำหรับตรวจสอบความถูกต้องของข้อมูลทั้ง 1 RECORD

จาก INTEL HEX FORMAT เราจะนำมาออกแบบโปรแกรมที่จะใช้รับข้อมูลได้ดัง FLOW CHART โดยตำแหน่งของโปรแกรมที่ใช้รับจะอยู่ ADDRESS 2000H ของ MPF-IP ณ.ตำแหน่ง U3 โดยใช้ EPROM เบอร์ 2732 , 2764

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
2500 A.D. Z80 CROSS ASSEMBLER   -   VERSION 3.00b
-------------------------------------------------

          INPUT  FILENAME :  MPF4.ASM
          OUTPUT FILENAME :  MPF4.OBJ


                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                    ;THIS PROGRAM USED FOR SEND ;
                    ;DATA FROM PC TO MPF1-P.    ;
                    ;BAUD RATE IS 9600.         ;
                    ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

                    ;DESIGN PROGRAM BY CHAIYOT L.

A0 FE                    USESTK:  EQU  0FEA0H
F6 07                    CLRBF:   EQU  07F6H
46 02                    SCAN:    EQU  0246H
21 08                    CONVER:  EQU  0821H
2C FF                    DISPB:   EQU  0FF2CH
14 0B                    ONE:     EQU  0B14H
70 FE                    DEST:    EQU  0FE70H

2000                              ORG 2000H
2000   31 A0 FE          MAIN:  LD SP,USESTK
2003   DD 21 2C FF              LD IX,DISPB
2007   CD F6 07                 CALL CLRBF
200A   DD 36 00 C0              LD (IX+0),0C0H
200E   DD 36 01 FF              LD (IX+1),0FFH     ;  "O"
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
2012   DD 36 02 0E          LD (IX+2),0EH
2016   DD 36 03 FF          LD (IX+3),0FFH      ;  "F"
201A   DD 36 04 0E          LD (IX+4),0EH
201E   DD 36 05 FF          LD (IX+5),0FFH      ;  "F"
2022   DD 36 06 12          LD (IX+6),12H
2026   DD 36 07 FF          LD (IX+7),0FFH      ;  "S"
202A   DD 36 08 06          LD (IX+8),06H
202E   DD 36 09 FF          LD (IX+9),0FFH      ;  "E"
2032   DD 36 0A FE          LD (IX+0AH),0FEH
2036   DD 36 0B FC          LD (IX+0BH),0FCH    ;  "T"

                    ; DISPLAY "OFFSET" AND RECIEVED OFFSET ADDR.

203A   FD 21 70 FE          LD IY,DEST
203E   CD 46 02             CALL SCAN
2041   F5                   PUSH AF
2042   CD F6 07             CALL CLRBF
2045   F1                   POP AF
2046   06 04                LD B,04H
2048   C5            LOOP:  PUSH BC
2049   F5                   PUSH AF
204A   CD 14 0B             CALL ONE
204D   FD 77 00             LD (IY+0),A
2050   F1                   POP AF
2051   CD 21 08             CALL CONVER
2054   CD 46 02             CALL SCAN
2057   C1                   POP BC
2058   FD 23                INC IY
205A   10 EC                DJNZ LOOP
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
205C   FD 21 70 FE          LD IY,DEST
2060   FD CB 00 26          SLA (IY+0)
2064   FD CB 00 26          SLA (IY+0)
2068   FD CB 00 26          SLA (IY+0)
206C   FD CB 00 26          SLA (IY+0)
2070   FD 7E 01             LD A,(IY+1)
2073   FD B6 00             OR (IY+0)
2076   57                   LD D,A
2077   FD CB 02 26          SLA (IY+2)
207B   FD CB 02 26          SLA (IY+2)
207F   FD CB 02 26          SLA (IY+2)
2083   FD CB 02 26          SLA (IY+2)
2087   FD 7E 03             LD A,(IY+3)
208A   FD B6 02             OR (IY+2)
208D   FD 77 00             LD (IY+0),A
2090   7A                   LD A,D
2091   FD 77 01             LD (IY+1),A ;NOW THE DEST HAVE BEEN
                                         ;OFFSET ADDR  ALREADY.

                  ;SET MODE AND COMMAND FOR #8251.
2094   3E 00                LD A,0        ;RESET #8251
2096   D3 1F                OUT (1FH),A
2098   D3 1F                OUT (1FH),A
209A   D3 1F                OUT (1FH),A
209C   3E 40                LD A,40H
209E   D3 1F                OUT (1FH),A
20A0   3E DD                LD A,0DDH     ; 8 BIT DATA,2 BIT STOP
20A2   D3 1F                OUT (1FH),A   ; 1BIT START,ODD PARITY
20A4   3E 14                LD A,14H      ; RXD ENABLE
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
20A6   D3 1F              OUT (1FH),A
20A8   CD 4A 21    MPF2:  CALL MPFBY
20AB   3E 3A              LD A,3AH        ; 3AH = ":"
20AD   BA                 CP D
20AE   C2 00 20           JP NZ,MAIN
20B1   0E 0C              LD C,0          ;CHECKSUM
20B3   CD 23 21           CALL MPFBY2
20B6   42                 LD B,D          ;BYTE COUNT
20B7   CD 23 21           CALL MPFBY2     ;ADDRESS 1st BYTE
20BA   D5                 PUSH DE
20BB   CD 23 21           CALL MPFBY2     ;ADDRESS 2nd BYTE
20BE   E1                 POP HL
20BF   6A                 LD L,D
20C0   E5                 PUSH HL
20C1   DD E1              POP IX
20C3   CD 23 21           CALL MPFBY2     ;RECORD BYTE 00.
20C6   ED 5B 70 FE        LD DE,(DEST)
20CA   DD 19              ADD IX,DE ;ADDRESS FOR STORE DATA.
20CC   CD 23 21    MPF3:  CALL MPFBY2     ;LINE DATA
20CF   7A                 LD A,D
20D0   DD 77 00           LD (IX+0),A
20D3   DD 23              INC IX
20D5   10 F5              DJNZ MPF3
20D7   79                 LD A,C          ;CHECKSUM
20D8   ED 44              NEG
20DA   47                 LD B,A
20DB   CD 23 21           CALL MPFBY2
20DE   7A                 LD A,D
20DF   B8                 CP B
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
20E0    C2 00 20            JP NZ,MAIN
20E3    CD 4A 21            CALL MPFBY      ;CR
20E6    CD 4A 21            CALL MPFBY      ;LF

                    ;START NEW LINE.
20E9    CD 4A 21            CALL MPFBY
20EC    3E 3A               LD A,3AH
20EE    BA                  CP D
20EF    C2 00 20            JP NZ,MAIN ;CHECK FIRST BYTE IS ":"?
20F2    0E 00               LD C,0          ;CHECKSUM
20F4    CD 23 21            CALL MPFBY2
20F7    7A                  LD A,D          ;BYTE COUNT
20F8    FE 00               CP 0
20FA    28 18               JR Z,MPFEND
20FC    47                  LD B,A
20FD    CD 23 21            CALL MPFBY2
2100    D5                  PUSH DE
2101    CD 23 21            CALL MPFBY2
2104    E1                  POP HL
2105    6A                  LD L,D
2106    E5                  PUSH HL
2107    DD E1               POP IX
2109    CD 23 21            CALL MPFBY2
210C    ED 5B 70 FE         LD DE,(DEST)
2110    DD 19               ADD IX,DE       ;ADDRESS START
2112    18 B8               JR MPF3

                    ;END OF DATA.
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

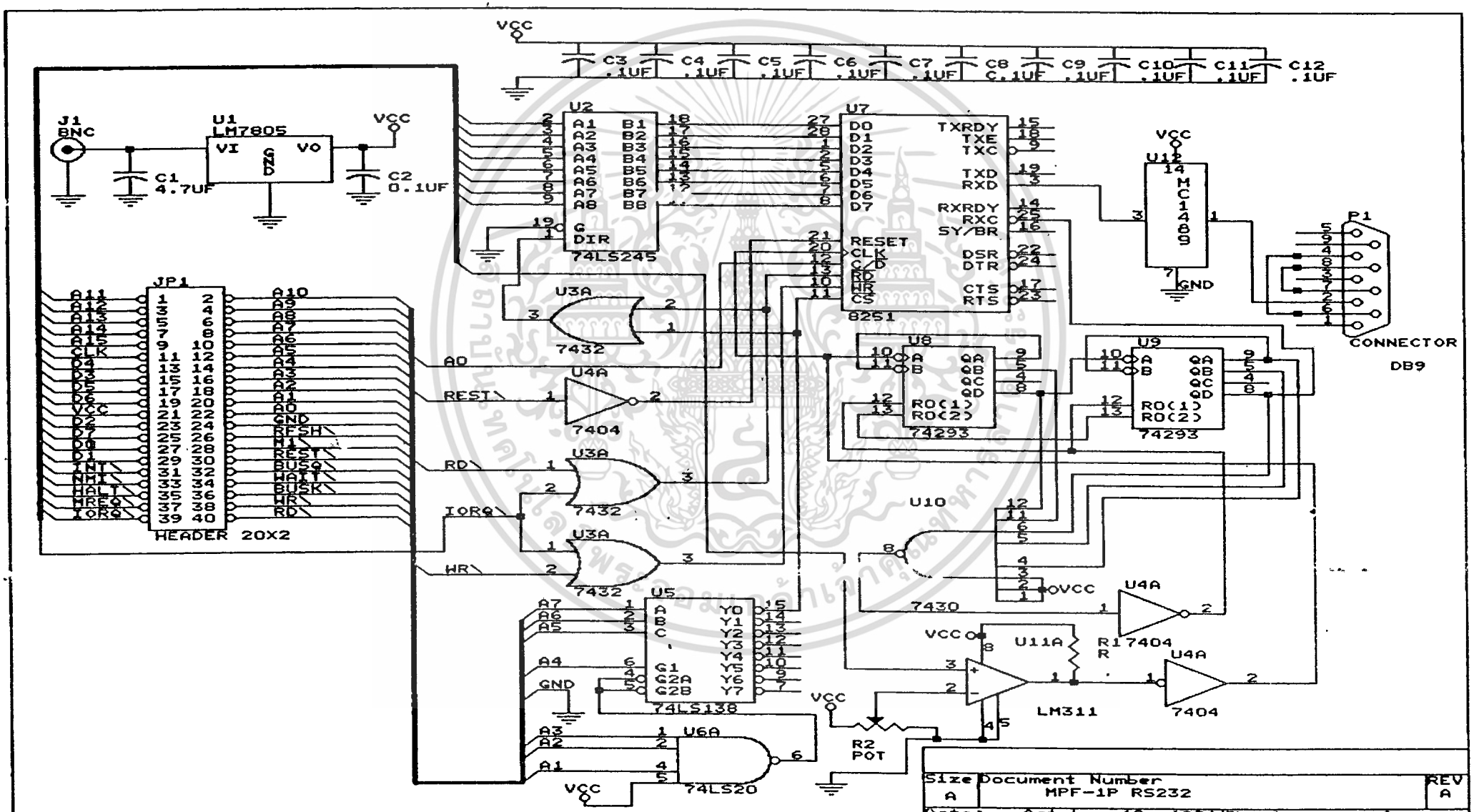
```
2114   06 04          MPFEND: LD B,4
2116   CD 23 21       MPFND1: CALL MPFBY2
2119   10 FB                  DJNZ MPFND1
211B   CD 4A 21               CALL MPFBY     ;CR
211E   CD 4A 21               CALL MPFBY     ;LF
2121   AF                     XOR A
2122   FF                     RST 38H
                    ;SUB READ TWO BYTE
                    ;AND CONVERT TO HEX CODE.
2123   CD 4A 21       MPFBY2: CALL MPFBY
2126   CD 14 0B               CALL ONE
2129   CB 07                  RLC A
212B   CB 07                  RLC A
212D   CB 07                  RLC A
212F   CB 07                  RLC A
2131   F5                     PUSH AF
2132   C5             DELAY:  PUSH BC
2133   06 98                  LD B,152  ;DELAY 1120 us
2135   10 FE          LOOP1:  DJNZ LOOP1
2137   00                     NOP
2138   00                     NOP
2139   00                     NOP
213A   00                     NOP
213B   C1                     POP BC
213C   CD 4A 21               CALL MPFBY
213F   CD 14 0B               CALL ONE
2142   57                     LD D,A
2143   F1                     POP AF
2144   B2                     OR D
2145   57                     LD D,A
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
2146   79                LD A,C
2147   82                ADD A,D
2148   4F                LD C,A
2149   C9                RET

                         ;SUB READ ONE BYTE
214A   DB 1F     MPFBY:  IN A,(1FH)
214C   CB 4F             BIT 1,A
214E   28 FA             JR Z,MPFBY
2150   00                NOP
2151   00                NOP
2152   00                NOP
2153   00                NOP
2154   DB 1E             IN A,(1EH)
2156   DB 1E             IN A,(1EH)
2158   57                LD D,A
2159   C9                RET
215A                     END
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


VCC
C3 .1UF
C4 .1UF
C5 .1UF
C6 .1UF
C7 .1UF
C8 C.1UF
C9 .1UF
C10 .1UF
C11 .1UF
C12 .1UF
J1 BNC
U1 LM7805
VI
GND
VO
C1 4.7UF
C2 0.1UF
U2
74LS245
A1 A2 A3 A4 A5 A6 A7 A8
B1 B2 B3 B4 B5 B6 B7 B8
G
DIR
U7
8251
D0 D1 D2 D3 D4 D5 D6 D7
TXRDY
TXE
TXC
TXD
RXD
RXRDY
RXC
SY/BR
DSR
DTR
CTS
RTS
RESET
CLK
C/D
RD
WR
CS
U12
MC1489
GND
P1
CONNECTOR DB9
JP1
HEADER 20X2
A11 A12 A13 A14 A15 CLK D4 D3 D5 D6 VCC D2 D7 D0 D1 INT NMI HALT MREQ IORQ
A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0 GND RFSH M1 RST BUSQ WAIT BUSK WR RD
U3A
7432
U4A
7404
AD
RESI
RD
IORQ
WR
U8
U9
74293
RO(1)
RO(2)
U10
7430
U5
74LS138
A7 A6 A5 A4 GND
Y0 Y1 Y2 Y3 Y4 Y5 Y6 Y7
G1 G2A G2B
U6A
74LS20
A3 A2 A1
U11A
LM311
R1
R
7404
R2
POT
Size A
Document Number MPF-1P RS232
REV A
Date: October 18, 1991
Sheet of

บทสรุปและวิจารณ์

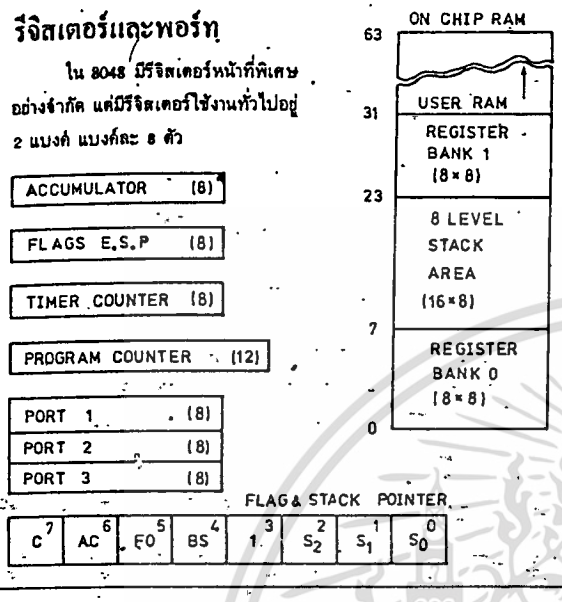
จากการทำโปรเจ็ก "ชุดพัฒนา 8048 บน MPF-IP" ปัญหาที่เกิดขึ้นในระหว่างการทำงาน คือ สัญญาณรบกวน ซึ่งเกิดจากระบบกราว์ดไม่ดีพอนั่นเอง ในการแก้ปัญหา ต้องต่อสายกราว์ด จากบอร์ด MPF-IP (แยกต่างหากจากสายแพ 40เส้น) มายัง ชุดพัฒนานี้ นอกนี้ ยังมีปัญหาในเรื่องของสัญญาณนาฬิกา ที่ดึงมาจากบอร์ด MPF-IP เนื่องจากมีสัญญาณ NOISE รบกวนมากทางแก้ปัญหาคือใช้วงจร คอมพาเรเตอร์ จัดรูปคลื่นสัญญาณให้คมชัดขึ้น

ในส่วนของ 8048 จะใช้คำสั่ง BUS REQUEST เมื่อจะใช้ระบบของ MPF-IP ในการ RUN การป้อนข้อมูลให้กับ 8048 SINGLE CHIP เราสามารถที่จะป้อน OP-CODE เข้าตรงๆ ทาง KEY BOARD ของ MPF-IP ได้ หรืออาจจะ เขียนเป็นภาษานีโมนิคบนเครื่อง PC โดยใช้ EDITER อะไรก็ได้ จากนั้นก็ใช้ คอมไพเลอร์ ของ 8048 ทำการ คอมไพล์ จากนีโมนิคมาเป็น OP-CODE ซึ่งจะได้ชื่อไฟล์ มีนามสกุลเป็น .OBJ แล้วจึงส่งผ่าน 8251 มายัง MPF-IP ซึ่งวิธีหลังนี้จะสะดวกกว่าที่จะป้อน OP-CODE เข้าทาง KEY BOARD โดยตรง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

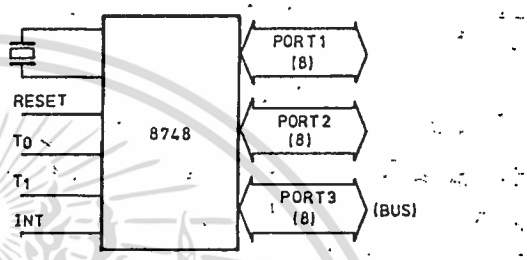
รีจิสเตอร์และพอร์ท

ใน 8048 มีรีจิสเตอร์หน้าที่พิเศษอย่างจำกัด แต่มีรีจิสเตอร์ใช้งานทั่วไปอยู่ 2 แบงค์ แบงค์ละ 8 ตัว



ชุดคำสั่งและซอฟท์แวร์

ชุดคำสั่งของ 8048 ประกอบด้วยคำสั่งต่าง ๆ 71 คำสั่ง แต่ละคำสั่งจะรวมการอ้างแอดเดรสไว้ในตัว คำสั่งหนึ่งจะมีเพียง 1 ถึง 2 ไบท์เท่านั้นและส่วนมากเป็นขนาด 1 ไบท์ และเป็นคำสั่งเฉพาะเรื่องมากกว่าเป็นคำสั่งในการคำเนินกรรมวิธีโดยทั่ว ๆ ไปทำให้ลดเวลาในการปฏิบัติลงได้มาก ที่พิเศษคือมีคำสั่งสำหรับคำเนินกรรมวิธีกับพอร์ทได้โดยตรง

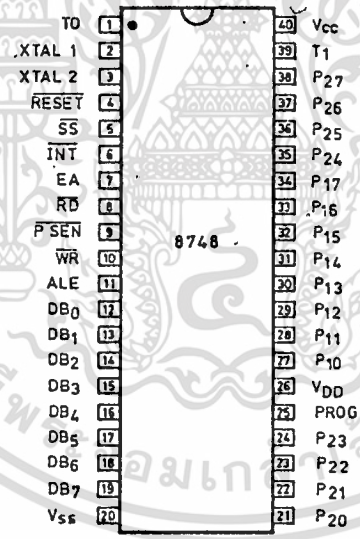


วงจรพื้นฐาน

ฮาร์ดแวร์

อ้างแอดเดรสถึงหน่วยความจำได้	: ROM 4 k ไบท์
	: RAM 320 ไบท์
อ้างแอดเดรสถึงพอร์ท	: 3 พอร์ท (ปกติ)
สัญญาณนาฬิกาของระบบ	: 2 MHz*
แรงดันไฟเลี้ยง	: + 5V
ขาอินเตอร์รัพท์	: $\overline{INT}$

* ใช้คริสตอล 6 MHz เพื่อให้ได้สัญญาณนาฬิกา 2 MHz



เวลาในการทำคำสั่ง

บวกค่าในรีจิสเตอร์กับแอคคิวว	2.5 µS
เอาท์พุทค่าในแอคคิวว ไปยังพอร์ท	5.0 µS
ย้ายข้อมูลระหว่างหน่วยความจำ	5.0 µS

ซีพียูในชุด 8048

8035 เช่นเดียวกับ 8048 แต่ไม่มี ROM ในตัว และมี RAM 64 ไบท์

8039 เหมือนกับ 8035 แต่มี RAM 128 ไบท์

8049 เช่นเดียวกับ 8048 แต่มี ROM ในตัวมากกว่า คือมี 2 k ไบท์

8021 ขนาด 28 ขา ROM ในตัว 1 k ไบท์

8022 ปรับปรุงจาก 8048 มี ROM 2 k ไบท์ และวงจร A/D ในตัว

8748 เช่นเดียวกับ 8048 แต่เป็น EPROM แทน ROM

8749 เหมือนกับ 8748 แต่ EPROM 2 k ไบท์

8051 ประสิทธิภาพสูงที่สุดในตระกูลนี้ มี ROM 4 k ไบท์ RAM 128 ไบท์ และ UART ในตัวด้วย และยังได้เพิ่มคำสั่งในการคูณและหาร

โรงงานผู้ผลิต

ผู้ริเริ่ม : INTEL

ผู้ผลิตรอง : N.E.C., NATIONAL, FUJITSU, TOSHIBA, SIGNATICS, AMD, SIEMENS

ชิพ ซัพพอร์ท

ถึงแม้ 8048 จะเป็นชิพไมโครคอมพิวเตอร์ที่สมบูรณ์และไม่ต้องการอุปกรณ์สนับสนุนใด ๆ แต่เราก็สามารถต่อเพื่อขยายหน่วยความจำหรือขยายระบบได้ อุปกรณ์ที่สนับสนุน 8048 โดยตรงคือ 8243 เพื่อขยายพอร์ทออกไปอีก 4 พอร์ท พอร์ทละ 4 บิท นอกจากนี้เรายังสามารถนำอุปกรณ์สนับสนุนของ 8085 มาใช้ได้โดยตรง ที่สำคัญคือ 8155 และ 8755

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MNEMONIC	FUNCTION	DESCRIPTION	CYCLES	BYTES	C	AC	F0	F1
ACCUMULATOR								
ADD A, #data	(A) ← (A) + data	Add immediate the specified data to the Accumulator	2	2	•	•		
ADD A, Rr	(A) ← (A) + (Rr) for r = 0 - 7	Add contents of designated register to the Accumulator	1	1	•	•		
ADD A, @ Rr	(A) ← (A) + ((Rr)) for r = 0 - 1	Add Indirect the contents the data memory location to the Accumulator	1	1	•	•		
ADDC A, #data	(A) ← (A) + (C) + data	Add Immediate with carry the specified data to the Accumulator	2	2	•	•		
ADDC A, Rr	(A) ← (A) + (C) + (Rr) for r = 0 - 7	Add with carry the contents of the designated register to the Accumulator	1	1	•	•		
ADDC A, @ Rr	(A) ← (A) + (C) + ((Rr)) for r = 0 - 1	Add Indirect with carry the contents of data memory location to the Accumulator	1	1	•	•		
ANL A, #data	(A) ← (A) AND data	Logical AND specified Immediate Data with Accumulator.	2	2				
ANL A, Rr	(A) ← (A) AND (Rr) for r = 0 - 7	Logical AND contents of ted register with Accumulator.	1	1				
ANL A, @ Rr	(A) ← (A) AND ((Rr)) for r = 0 - 1	Logical AND Indirect the contents of data memory with Accumulator.	1	1				
CPL A	(A) ← NOT (A)	Complement the contents of the Accumulator.	1	1				
CLR A	(A) ← 0	CLEAR the contents of the Accumulator.	1	1				
DA A		DECIMAL ADJUST the contents of the Accumulator.	1	1	•			
DEC A	(A) ← (A) - 1	DECREMENT by 1 the accumulator's contents	1	1				
INC A	(A) ← (A) + 1	Increment by 1 the accumulator's contents	1	1				
ORL A, #data	(A) ← (A) OR data	Logical OR specified Immediate data with Accumulator.	2	2				
ORL A, Rr	(A) ← (A) OR (Rr) for r = 0 - 7	Logical OR contents of designated register with Accumulator.	1	1				
ORL A, @ Rr	(A) ← (A) OR ((Rr)) for r = 0 - 1	Logical OR Indirect the contents of data memory location with Accumulator.	1	1				
RL A	(An + 1) ← (An) for n = 0 - 6 (A0) ← (A7)	Rotate Accumulator left by 1-bit without carry.	1	1				
RLC A	(An + 1) ← (An); n = 0-6 (A0) ← (C) (C) ← (A7)	Rotate Accumulator left by 1-bit through carry.	1	1	•			
RR A	(An) ← (An + 1); n = 0-6 (A7) ← (A0)	Rotate Accumulator right by 1-bit without carry.	1	1				
RRC A	(An) ← (An + 1); n = 0-6 (A7) ← (C) (C) ← (A0)	Rotate Accumulator right by 1-bit through carry.	1	1	•			
SWAP A	(A4-A7) ← (A0 - A3)	Swap the 2, 4-bit nibbles in the Accumulator.	1	1				
XRL A, #data	(A) ← (A) XOR data	Logical XOR immediate specified data with Accumulator.	2	2				
XRL A, Rr	(A) ← (A) XOR (Rr) for r = 0 - 7	Logical XOR contents of designated register with Accumulator.	1	1				
XRL A, @ Rr	(A) ← (A) XOR ((Rr)) for r = 0 - 1	Logical XOR Indirect the contents of data memory location with Accumulator.	1	1				
BRANCH								
DJNZ Rr, addr	(Rr) ← (Rr) - 1 if (Rr) ≠ 0, (PC 0-7) ← addr	Decrement the specified register and test contents	2	2				
JBb addr	(PC 0-7) ← addr if Bb = 1 (PC) ← (PC) + 2 if Bb = 0	Jump to specified address if Accumulator bit is set	2	2				
JC addr	(PC 0-7) ← addr if C = 1 (PC) ← (PC) +2 if C = 0	Jump to specified address if carry flag is set	2	2				
JF0 addr	(PC 0-7) ← addr if F0 = 1 (PC) ← (PC) + 2 if F0 = 0	Jump to specified address if Flag F0 is set	2	2				
JF1 addr	(PC 0-7) ← addr if F1 = 1 (PC) ← (PC) + 2 if F1 = 0	Jump to specified address if Flag F1 is set	2	2				
JMP addr	(PC 8-10) ← addr 8-10 (PC 0-7) ← addr 0-7 (PC 11) ← DBF	Direct Jump to specified address within the 2K address block	2	2				
JMPP @ A	(PC 0-7) ← ((A))	Jump indirect to specified address pointed to by the accumulator in current page.	2	1				
JNC addr	(PC 0-7) ← addr if C = 0 (PC) ← (PC) + 2 if C = 1	Jump to specified address if carry flag is low.	2	2				
JNI addr	(PC 0-7) ← addr if I = 0 (PC) ← (PC) + 2 if I = 1	Jump to specified address if interrupt is low.	2	2				
JNT0 addr	(PC 0-7) ← addr if T0 = 0 (PC) ← (PC) + 2 if T0 = 1	Jump to specified address if Test 0 is low.	2	2				
JNT1 addr	(PC 0-7) ← addr if T1 = 0 (PC) ← (PC) + 2 if T1 = 1	Jump to specified address if Test 1 is low	2	2				
JNZ addr	(PC 0-7) ← addr if A ≠ 0 (PC) ← (PC) + 2 if A = 0	Jump to specified address if accumulator is non-zero.	2	2				
JTF addr	(PC 0-7) ← addr if TF = 1 (PC) ← (PC) + 2 if TF = 0	Jump to specified address if Timer Flag is set to 1.	2	2				
JT0 addr	(PC 0-7) ← addr if T0 = 1 (PC) ← (PC) + 2 if T0 = 0	Jump to specified address if Test 0 is a 1.	2	2				
JT1 addr	(PC 0-7) ← addr if T1 = 1 (PC) ← (PC) + 2 if T1 = 0	Jump to specified address if Test 1 is a 1.	2	2				
JZ addr	(PC 0-7) ← addr if A = 0 (PC) ← (PC) + 2 if A = 1	Jump to specified address if Accumulator is 0.	2	2				
CONTROL								
EN I		Enable the External Interrupt input.	1	1				
DIS I		Disable the External Interrupt input.	1	1				
ENT0 CLK		Enable T0 as the Clock Output.	1	1				
SEL MB0	(DBF) ← 0	Select Bank 0 (locations 0 - 2047) of Program Memory.	1	1				
SEL MB1	(DBF) ← 1	Select Bank 1 (locations 2048 - 4095) of Program Memory.	1	1				
SEL RB0	(BS) ← 0	Select Bank 0 (locations 0 - 7) of Data Memory.	1	1				
SEL RB1	(BS) ← 1	Select Bank 1 (locations 24 - 31) of Data Memory.	1	1				
DATA MOVES								
MOV A, #data	(A) ← data	Move Immediate the specified data into the Accumulator.	2	2				
MOV A, Rr	(A) ← (Rr); r = 0 - 7	Move the contents of the designated registers into the Accumulator.	1	1				
MOV A, @ Rr	(A) ← ((Rr)); r = 0 - 1	Move Indirect the contents of data memory location into the Accumulator	1	1				

MNEMONIC	FUNCTION	DESCRIPTION	CYCLES	BYTES	C	AC	F0	F1
		DATA MOVES (Cont'd)						
MOV A, PSW	(A) ← (PSW)	Move contents of the Program Status Word into the Accumulator	1	1				
MOV Rr, #data	(Rr) ← data; r = 0 - 7	Move immediate the specified data into the designated register	2	2				
MOV Rr, A	(Rr) ← (A); r = 0 - 7	Move Accumulator contents into the designated register	1	1				
MOV @ Rr, A	((Rr)) ← (A); r = 0 - 1	Move Indirect Accumulator contents into data memory location	1	1				
MOV @ Rr, #data	((Rr)) ← data; r = 0 - 1	Move Immediate the specified data into data memory.	2	2				
MOV PSW, A	(PSW) ← (A)	Move contents of Accumulator into the Program Status Word.	1	1	•	•	•	
MOVP A, @ A	(PC 0 - 7) ← (A) (A) ← ((PC))	Move the content of program memory location in the current page addressed by the content of accumulator into the accumulator.	2	1				
MOVP3 A, @ A	(PC 0 - 7) ← (A) (PC 8 - 10) ← 011 (A) ← ((PC))	Move the content of program memory location in page 3 address by the content of accumulator into the accumulator.	2	1				
MOVX A, @ R	(A) ← ((Rr)); r = 0 - 1	Move Indirect the contents of external data memory into the Accumulator.	2	1				
MOVX @ R, A	((Rr)) ← (A); r = 0 - 1	Move Indirect the contents of the Accumulator into external data memory.	2	1				
XCH A, Rr	(A) ↔ (Rr); r = 0 - 7	Exchange the Accumulator and designated register's contents.	1	1				
XCH A, @ Rr	(A) ↔ ((Rr)); r = 0 - 1	Exchange Indirect contents of Accumulator and location in data memory.	1	1				
XCHD A, @ Rr	(A0 - A3) ↔ ((Rr)) 0 - 3; r = 0 - 1	Exchange Indirect 4-bit contents of Accumulator and data memory.	1	1				
		TIMER COUNTER						
EN TCNTI		Enable Internal Interrupt Flag for Timer/ Counter output.	1	1				
DIS TCNTI		Disable Internal Interrupt Flag for Timer/ Counter output.	1	1				
MOV A, T	(A) ← (T)	Move contents of Timer/Counter into Accumulator.	1	1				
MOV T, A	(T) ← (A)	Move contents of Accumulator into Timer/Counter.	1	1				
STOP TCNT		Stop Count for Event Counter.	1	1				
STRT CNT		Start Count for Event Counter.	1	1				
STRT T		Start Count for Timer.	1	1				
		INPUT/OUTPUT						
ANL BUS, #data	(BUS) ← (BUS) AND data	Logical AND immediate specified data with contents of BUS	2	2				
ANL Pp, #data	(Pp) ← (Pp) AND data; p = 1 - 2	Logical AND immediate specified data with designated port (1 or 2)	2	2				
ANLD Pp, A	(Pp) ← (Pp) AND (A0 - A3), p = 4-7	Logical AND contents of Accumulator with designated port (4 - 7)	2	1				
IN A, Pp	(A) ← (Pp); p = 1-2	Input data from designated port (1 - 2) into Accumulator	2	1				
INS A, BUS	(A) ← (BUS)	Input strobed BUS data into Accumulator	2	1				

MNEMONIC	FUNCTION	DESCRIPTION	CYCLES	BYTES	C	AC	F0	F1
		INPUT/OUTPUT (Cont'd)						
MOVD A, Pp	(A0 A3) ← (Pp) p = 4 7 (A4-A7) ← 0	Move contents of designated port (4 - 7) into Accumulator	2	1				
MOVD Pp, A	(Pp) ← (A0 - A3), p = 4 - 7	Move contents of Accumulator to designated port (4 - 7)	2	1				
ORL BUS, #data	(BUS) ← (BUS) OR data	Logical OR immediate specified data with contents of BUS	2	2				
ORLD Pp, A	(Pp) ← (Pp) OR (A0 - A3), p = 4-7	Logical OR contents of Accumulator with designated port (4 - 7)	2	1				
ORL Pp, #data	(Pp) ← (Pp) OR data; p = 1 - 2.	Logical OR immediate specified data with designated port (1 - 2)	2	2				
OUTL BUS, A	(BUS) ← (A)	Output contents of Accumulator onto BUS	2	1				
OUTL Pp, A	(Pp) ← (A); p = 1 - 2	Output contents of Accumulator to designated port (1 - 2).	1	1				
		REGISTERS						
DEC Rr	(Rr) ← (Rr) -1; r = 0-7	Decrement by 1 contents of designated register.	1	1				
INC Rr	(Rr) ← (Rr) +1; r = 0-7	Increment by 1 contents of designated register.	1	1				
INC @ Rr	((Rr)) ← ((Rr)) + 1; r = 0-1.	Increment Indirect by 1 the contents of data memory location.	1	1				
		SUBROUTINE						
CALL addr	((SP)) ← (PC) ((SP)) ← (PSW 4-7) (SP) ← (SP) + 1 (PC8-10) ← addr 8-10 (PC 0-7) ← addr 0-7 (PC 11) ← DBF	Call designated Subroutine.	2	2				
RET	(SP) ← (SP) - 1 (PC) ← ((SP))	Return from Subroutine without restoring Program Status Word.	2	1				
RETR	(SP) ← (SP) - 1 (PC) ← ((SP)) (PSW 4-7) ← ((SP))	Return from Subroutine restoring Program Status Word.	2	1	•	•		
		FLAGS						
CPL C	(C) ← NOT (C)	Complement Content of carry bit.	1	1	•			
CPL F0	(F0) ← NOT (F0)	Complement Content of Flag F0	1	1			•	
CPL F1	(F1) ← NOT (F1)	Complement Content of Flag F1.	1	1				•
CLR C	(C) ← 0	Clear content of carry bit to 0	1	1	•			
CLR F0	(F0) ← 0	Clear content of Flag 0 to 0.	1	1			•	
CLR F1	(F1) ← 0	Clear content of Flag 1 to 0.	1	1				•
		MISCELLANEOUS						
NOP		No operation.	1	1				

intel®

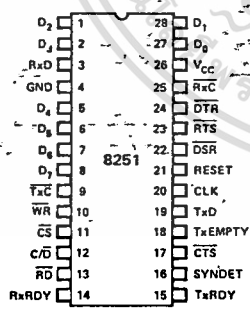
Silicon Gate MOS 8251

PROGRAMMABLE COMMUNICATION INTERFACE

- **Synchronous and Asynchronous Operation**
  - **Synchronous:**
    - 5-8 Bit Characters
    - Internal or External Character Synchronization
    - Automatic Sync Insertion
  - **Asynchronous:**
    - 5-8 Bit Characters
    - Clock Rate — 1, 16 or 64 Times Baud Rate
    - Break Character Generation
    - 1, 1½, or 2 Stop Bits
    - False Start Bit Detection
- **Baud Rate —DC to 56k Baud (Sync Mode)**
  DC to 9.6k Baud (Async Mode)
- **Full Duplex, Double Buffered, Transmitter and Receiver**
- **Error Detection — Parity, Overrun, and Framing**
- **Fully Compatible with 8080 CPU**
- **28-Pin DIP Package**
- **All Inputs and Outputs Are TTL Compatible**
- **Single 5 Volt Supply**
- **Single TTL Clock**

The 8251 is a Universal Synchronous/Asynchronous Receiver/Transmitter (USART) Chip designed for data communications in microcomputer systems. The USART is used as a peripheral device and is programmed by the CPU to operate using virtually any serial data transmission technique presently in use (including IBM Bi-Sync). The USART accepts data characters from the CPU in parallel format and then converts them into a continuous serial data stream for transmission. Simultaneously it can receive serial data streams and convert them into parallel data characters for the CPU. The USART will signal the CPU whenever it can accept a new character for transmission or whenever it has received a character for the CPU. The CPU can read the complete status of the USART at any time. These include data transmission errors and control signals such as SYNDET, TxEMPT. The chip is constructed using N-channel silicon gate technology.

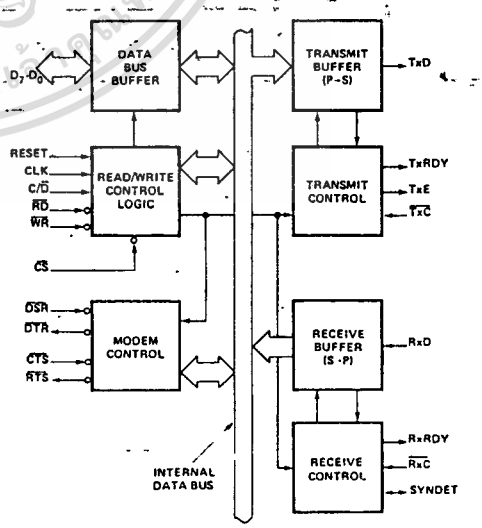
PIN CONFIGURATION



Pin Name	Pin Function
D_7-D_0	Data Bus (8 bits)
C/D	Control or Data is to be Written or Read
$\overline{RD}$	Read Data Command
$\overline{WR}$	Write Data or Control Command
$\overline{CS}$	Chip Enable
CLK	Clock Pulse (TTL)
RESET	Reset
$\overline{TxC}$	Transmitter Clock
TxD	Transmitter Data
$\overline{RxC}$	Receiver Clock
RxD	Receiver Data
RxRDY	Receiver Ready (has character for 8080)
TxRDY	Transmitter Ready (ready for char. from 8080)

Pin Name	Pin Function
$\overline{DSR}$	Data Set Ready
$\overline{DTR}$	Data Terminal Ready
SYNDET	Sync Detect
$\overline{RTS}$	Request to Send Data
$\overline{CTS}$	Clear to Send Data
TxE	Transmitter Empty
V_{CC}	+5 Volt Supply
GND	Ground

BLOCK DIAGRAM



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ

SILICON GATE MOS 8251

8251 BASIC FUNCTIONAL DESCRIPTION

General

The 8251 is a Universal Synchronous/Asynchronous Receiver/Transmitter designed specifically for the 8080 Microcomputer System. Like other I/O devices in the 8080 Microcomputer System its functional configuration is programmed by the systems software for maximum flexibility. The 8251 can support virtually any serial data technique currently in use (including IBM "bi-sync").

In a communication environment an interface device must convert parallel format system data into serial format for transmission and convert incoming serial format data into parallel system data for reception. The interface device must also delete or insert bits or characters that are functionally unique to the communication technique. In essence, the interface should appear "transparent" to the CPU, a simple input or output of byte-oriented system data.

Data Bus Buffer

This 3-state, bi-directional, 8-bit buffer is used to interface the 8251 to the 8080 system Data Bus. Data is transmitted or received by the buffer upon execution of INput or OUTput instructions of the 8080 CPU. Control words, Command words and Status information are also transferred through the Data Bus Buffer.

Read/Write Control Logic

This functional block accepts inputs from the 8080 Control bus and generates control signals for overall device operation. It contains the Control Word Register and Command Word Register that store the various control formats for device functional definition.

RESET (Reset)

A "high" on this input forces the 8251 into an "Idle" mode. The device will remain at "Idle" until a new set of control words is written into the 8251 to program its functional definition.

CLK (Clock)

The CLK input is used to generate internal device timing and is normally connected to the Phase 2 (TTL) output of the 8224 Clock Generator. No external inputs or outputs are referenced to CLK but the frequency of CLK must be greater than 30 times the Receiver or Transmitter clock inputs for synchronous mode (4 times for asynchronous mode).

$\overline{WR}$ (Write)

A "low" on this input informs the 8251 that the CPU is outputting data or control words, in essence, the CPU is writing out to the 8251.

$\overline{RD}$ (Read)

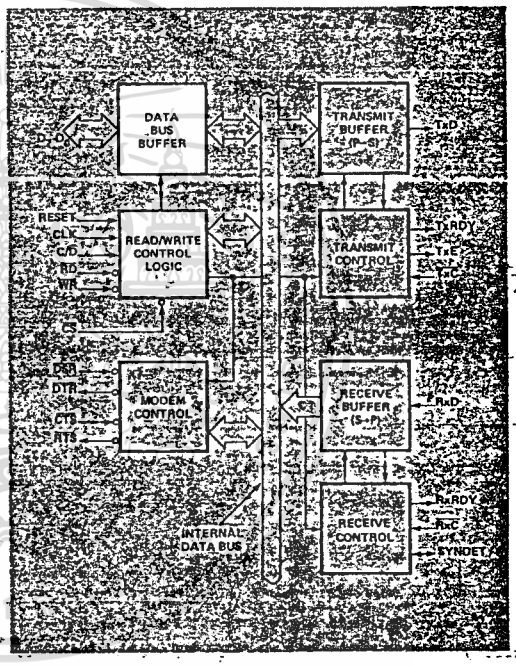
A "low" on this input informs the 8251 that the CPU is inputting data or status information, in essence, the CPU is reading from the 8251.

C/$\overline{D}$ (Control/Data)

This input, in conjunction with the $\overline{WR}$ and $\overline{RD}$ inputs informs the 8251 that the word on the Data Bus is either a data character, control word or status information.
1 = CONTROL 0 = DATA

$\overline{CS}$ (Chip Select)

A "low" on this input enables the 8251. No reading or writing will occur unless the device is selected.



C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	
0	0	1	0	8251 ➡ DATA BUS
0	1	0	0	DATA BUS ➡ 8251
1	0	1	0	STATUS ➡ DATA BUS
1	1	0	0	DATA BUS ➡ CONTROL
X	X	X	1	DATA BUS ➡ 3-STATE

SILICON GATE MOS 8251

Modem Control

The 8251 has a set of control inputs and outputs that can be used to simplify the interface to almost any Modem. The modem control signals are general purpose in nature and can be used for functions other than Modem control, if necessary.

$\overline{DSR}$ (Data Set Ready)

The $\overline{DSR}$ input signal is general purpose in nature. Its condition can be tested by the CPU using a Status Read operation. The $\overline{DSR}$ input is normally used to test Modem conditions such as Data Set Ready.

$\overline{DTR}$ (Data Terminal Ready)

The $\overline{DTR}$ output signal is general purpose in nature. It can be set "low" by programming the appropriate bit in the Command Instruction word. The $\overline{DTR}$ output signal is normally used for Modem control such as Data Terminal Ready or Rate Select.

$\overline{RTS}$ (Request to Send)

The $\overline{RTS}$ output signal is general purpose in nature. It can be set "low" by programming the appropriate bit in the Command Instruction word. The $\overline{RTS}$ output signal is normally used for Modem control such as Request to Send.

$\overline{CTS}$ (Clear to Send)

A "low" on this input enables the 8251 to transmit data (serial) if the Tx EN bit in the Command byte is set to a "one."

Transmitter Buffer

The Transmitter Buffer accepts parallel data from the Data Bus Buffer, converts it to a serial bit stream, inserts the appropriate characters or bits (based on the communication technique) and outputs a composite serial stream of data on the TxD output pin.

Transmitter Control

The Transmitter Control manages all activities associated with the transmission of serial data. It accepts and issues signals both externally and internally to accomplish this function.

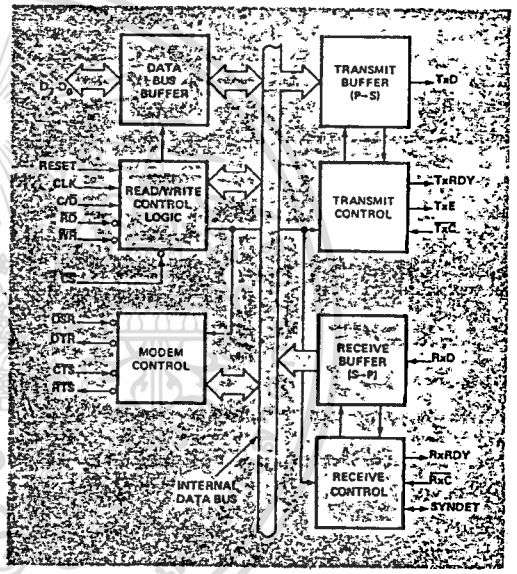
TxRDY (Transmitter Ready)

This output signals the CPU that the transmitter is ready to accept a data character. It can be used as an interrupt to the system or for the Polled operation the CPU can check TxRDY using a status read operation. TxRDY is automatically reset when a character is loaded from the CPU.

TxE (Transmitter Empty)

When the 8251 has no characters to transmit, the TxE output will go "high". It resets automatically upon receiving a character from the CPU. TxE can be used to indicate the end of a transmission mode, so that the CPU "knows" when to "turn the line around" in the half-duplexed operational mode.

In SYNChronous mode, a "high" on this output indicates that a character has not been loaded and the SYNC character or characters are about to be transmitted automatically as "fillers".



$\overline{TxC}$ (Transmitter Clock)

The Transmitter Clock controls the rate at which the character is to be transmitted. In the Synchronous transmission mode, the frequency of $\overline{TxC}$ is equal to the actual Baud Rate (1X). In Asynchronous transmission mode, the frequency of $\overline{TxC}$ is a multiple of the actual Baud Rate. A portion of the mode instruction selects the value of the multiplier; it can be 1x, 16x or 64x the Baud Rate.

For Example:

If Baud Rate equals 110 Baud,
$\overline{TxC}$ equals 110 Hz (1x)
$\overline{TxC}$ equals 1.76 kHz (16x)
$\overline{TxC}$ equals 7.04 kHz (64x).
If Baud Rate equals 9600 Baud,
$\overline{TxC}$ equals 614.4 kHz (64x).

The falling edge of $\overline{TxC}$ shifts the serial data out of the 8251.

SILICON GATE MOS 8251

Receiver Buffer

The Receiver accepts serial data, converts this serial input to parallel format, checks for bits or characters that are unique to the communication technique and sends an "assembled" character to the CPU. Serial data is input to the RxD pin.

Receiver Control

This functional block manages all receiver-related activities.

RxRDY (Receiver Ready)

This output indicates that the 8251 contains a character that is ready to be input to the CPU. RxRDY can be connected to the interrupt structure of the CPU or for Polled operation the CPU can check the condition of RxRDY using a status read operation. RxRDY is automatically reset when the character is read by the CPU.

$\overline{\text{RxC}}$ (Receiver Clock)

The Receiver Clock controls the rate at which the character is to be received. In Synchronous Mode, the frequency of $\overline{\text{RxC}}$ is equal to the actual Baud Rate (1x). In Asynchronous Mode, the frequency of $\overline{\text{RxC}}$ is a multiple of the actual Baud Rate. A portion of the mode instruction selects the value of the multiplier; it can be 1x, 16x or 64x the Baud Rate.

For Example:
If Baud Rate equals 300 Baud,
$\overline{\text{RxC}}$ equals 300 Hz (1x)
$\overline{\text{RxC}}$ equals 4800 Hz (16x)
$\overline{\text{RxC}}$ equals 19.2 kHz (64x).
If Baud Rate equals 2400 Baud,
$\overline{\text{RxC}}$ equals 2400 Hz (1x)
$\overline{\text{RxC}}$ equals 38.4 kHz (16x)
$\overline{\text{RxC}}$ equals 153.6 kHz (64x).

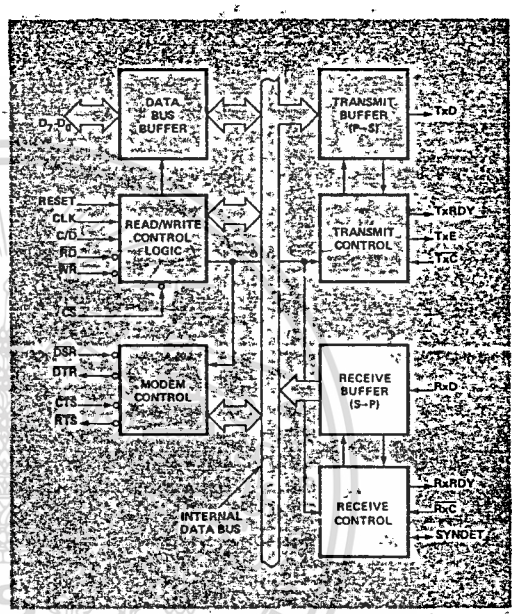
Data is sampled into the 8251 on the rising edge of $\overline{\text{RxC}}$.

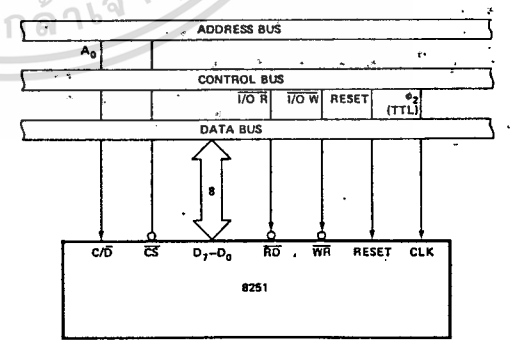
NOTE: In most communications systems, the 8251 will be handling both the transmission and reception operations of a single link. Consequently, the Receive and Transmit Baud Rates will be the same. Both $\overline{\text{TxC}}$ and $\overline{\text{RxC}}$ will require identical frequencies for this operation and can be tied together and connected to a single frequency source (Baud Rate Generator) to simplify the interface.

SYNDET (SYNC Detect)

This pin is used in SYNChronous Mode only. It is used as either input or output, programmable through the Control Word. It is reset to "low" upon RESET. When used as an output (internal Sync mode), the SYNDET pin will go "high" to indicate that the 8251 has located the SYNC character in the Receive mode. If the 8251 is programmed to use double Sync characters (bi-sync), then SYNDET will go "high" in the middle of the last bit of the second Sync character. SYNDET is automatically reset upon a Status Read operation.

When used as an input, (external SYNC detect mode), a positive going signal will cause the 8251 to start assembling data characters on the falling edge of the next $\overline{\text{RxC}}$. Once in SYNC, the "high" input signal can be removed. The duration of the high signal should be at least equal to the period of $\overline{\text{RxC}}$.





8251 Interface to 8080 Standard System Bus

SILICON GATE MOS 8251

DETAILED OPERATION DESCRIPTION

General

The complete functional definition of the 8251 is programmed by the systems software. A set of control words must be sent out by the CPU to initialize the 8251 to support the desired communications format. These control words will program the: BAUD RATE, CHARACTER LENGTH, NUMBER OF STOP BITS, SYNCHRONOUS or ASYNCHRONOUS OPERATION, EVEN/ODD PARITY etc. In the Synchronous Mode, options are also provided to select either internal or external character synchronization.

Once programmed, the 8251 is ready to perform its communication functions. The TxRDY output is raised "high" to signal the CPU that the 8251 is ready to receive a character. This output (TxRDY) is reset automatically when the CPU writes a character into the 8251. On the other hand, the 8251 receives serial data from the MODEM or I/O device, upon receiving an entire character the RxRDY output is raised "high" to signal the CPU that the 8251 has a complete character ready for the CPU to fetch. RxRDY is reset automatically upon the CPU read operation.

The 8251 cannot begin transmission until the TxEN(Transmitter Enable) bit is set in the Command Instruction and it has received a Clear To Send (CTS) input. The TxD output will be held in the marking state upon Reset.

Programming the 8251

Prior to starting data transmission or reception, the 8251 must be loaded with a set of control words generated by the CPU. These control signals define the complete functional definition of the 8251 and must immediately follow a Reset operation (internal or external).

The control words are split into two formats:

1. Mode Instruction
2. Command Instruction

Mode Instruction

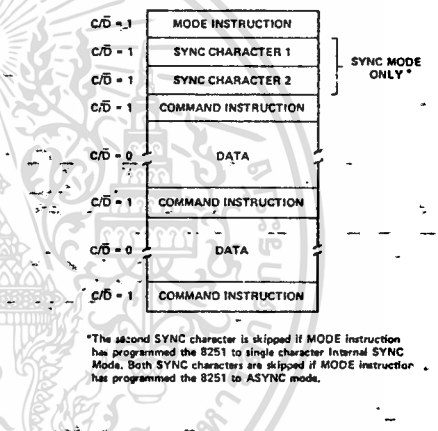
This format defines the general operational characteristics of the 8251. It must follow a Reset operation (internal or external). Once the Mode instruction has been written into the 8251 by the CPU, SYNC characters or Command instructions may be inserted.

Command Instruction

This format defines a status word that is used to control the actual operation of the 8251.

Both the Mode and Command instructions must conform to a specified sequence for proper device operation. The Mode Instruction must be inserted immediately following a Reset operation, prior to using the 8251 for data communication.

All control words written into the 8251 after the Mode Instruction will load the Command Instruction. Command Instructions can be written into the 8251 at any time in the data block during the operation of the 8251. To return to the Mode Instruction format a bit in the Command Instruction word can be set to initiate an internal Reset operation which automatically places the 8251 back into the Mode Instruction format. Command Instructions must follow the Mode Instructions or Sync characters.



Typical Data Block

SILICON GATE MOS 8251

Mode Instruction Definition

The 8251 can be used for either Asynchronous or Synchronous data communication. To understand how the Mode Instruction defines the functional operation of the 8251 the designer can best view the device as two separate components sharing the same package. One Asynchronous the other Synchronous. The format definition can be changed "on the fly" but for explanation purposes the two formats will be isolated.

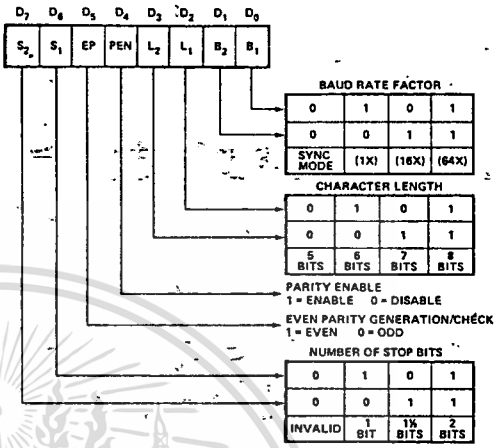
Asynchronous Mode (Transmission)

Whenever a data character is sent by the CPU the 8251 automatically adds a Start bit (low level) and the programmed number of Stop bits to each character. Also, an even or odd Parity bit is inserted prior to the Stop bit(s), as defined by the Mode Instruction. The character is then transmitted as a serial data stream on the TxD output. The serial data is shifted out on the falling edge of $\overline{TxC}$ at a rate equal to 1, 1/16, or 1/64 that of the $\overline{TxC}$, as defined by the Mode Instruction. BREAK characters can be continuously sent to the TxD if commanded to do so.

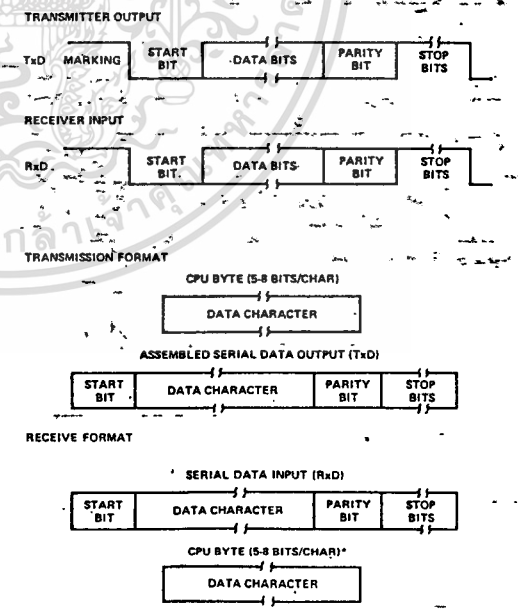
When no data characters have loaded into the 8251 the TxD output remains "high" (marking) unless a Break (continuously low) has been programmed.

Asynchronous Mode (Receive)

The RxD line is normally high. A falling edge on this line triggers the beginning of a START bit. The validity of this START bit is checked by again strobing this bit at its nominal center. If a low is detected again, it is a valid START bit, and the bit counter will start counting. The bit counter locates the center of the data bits, the parity bit (if it exists) and the stop bits. If parity error occurs, the parity error flag is set. Data and parity bits are sampled on the RxD pin with the rising edge of $\overline{RxC}$. If a low level is detected as the STOP bit, the Framing Error flag will be set. The STOP bit signals the end of a character. This character is then loaded into the parallel I/O buffer of the 8251. The RxRDY pin is raised to signal the CPU that a character is ready to be fetched. If a previous character has not been fetched by the CPU, the present character replaces it in the I/O buffer, and the OVERRUN flag is raised (thus the previous character is lost). All of the error flags can be reset by a command instruction. The occurrence of any of these errors will not stop the operation of the 8251.



Mode Instruction Format, Asynchronous Mode



*NOTE: IF CHARACTER LENGTH IS DEFINED AS 5, 6 OR 7 BITS THE UNUSED BITS ARE SET TO "ZERO"

Asynchronous Mode

SILICON GATE MOS 8251

Synchronous Mode (Transmission)

The TxD output is continuously high until the CPU sends its first character to the 8251 which usually is a SYNC character. When the $\overline{CTS}$ line goes low, the first character is serially transmitted out. All characters are shifted out on the falling edge of $\overline{TxC}$. Data is shifted out at the same rate as the $\overline{TxC}$.

Once transmission has started, the data stream at TxD output must continue at the $\overline{TxC}$ rate. If the CPU does not provide the 8251 with a character before the 8251 becomes empty, the SYNC characters (or character if in single SYNC word mode) will be automatically inserted in the TxD data stream. In this case, the TxEMPTY pin is raised high to signal that the 8251 is empty and SYNC characters are being sent out. The TxEMPTY pin is internally reset by the next character being written into the 8251.

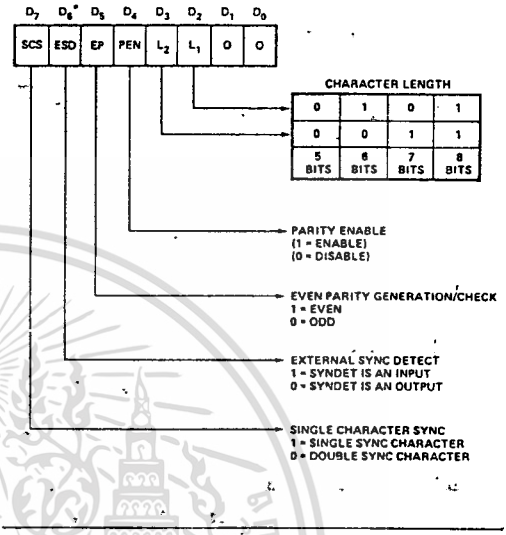
Synchronous Mode (Receive)

In this mode, character synchronization can be internally or externally achieved. If the internal SYNC mode has been programmed, the receiver starts in a HUNT mode. Data on the RxD pin is then sampled in on the rising edge of $\overline{RxC}$. The content of the Rx buffer is continuously compared with the first SYNC character until a match occurs. If the 8251 has been programmed for two SYNC characters, the subsequent received character is also compared; when both SYNC characters have been detected, the USART ends the HUNT mode and is in character synchronization. The SYNDET pin is then set high, and is reset automatically by a STATUS READ.

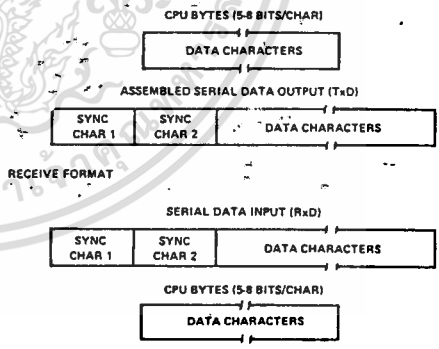
In the external SYNC mode, synchronization is achieved by applying a high level on the SYNDET pin. The high level can be removed after one $\overline{RxC}$ cycle.

Parity error and overrun error are both checked in the same way as in the Asynchronous Rx mode.

The CPU can command the receiver to enter the HUNT mode if synchronization is lost.



Mode Instruction Format, Synchronous Mode



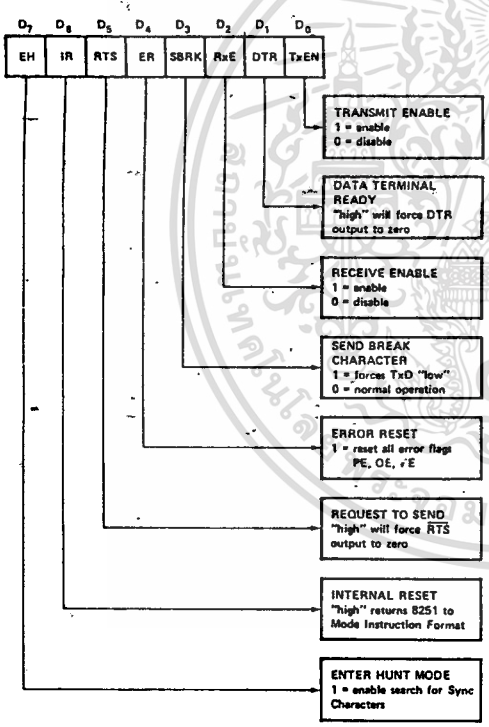
Synchronous Mode, Transmission Format

SILICON GATE MOS 8251

COMMAND INSTRUCTION DEFINITION

Once the functional definition of the 8251 has been programmed by the Mode Instruction and the Sync Characters are loaded (if in Sync Mode) then the device is ready to be used for data communication. The Command Instruction controls the actual operation of the selected format. Functions such as: Enable Transmit/Receive, Error Reset and Modem Controls are provided by the Command Instruction.

Once the Mode Instruction has been written into the 8251 and Sync characters inserted, if necessary, then all further "control writes" (C/$\overline{D}$ = 1) will load the Command Instruction. A Reset operation (internal or external) will return the 8251 to the Mode Instruction Format.



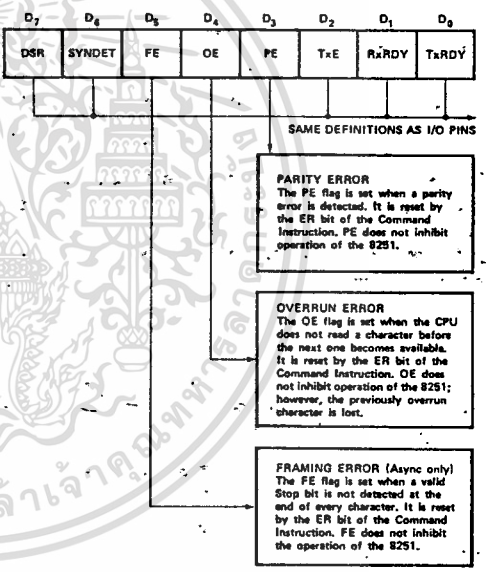
Command Instruction Format

STATUS READ DEFINITION

In data communication systems it is often necessary to examine the "status" of the active device to ascertain if errors have occurred or other conditions that require the processor's attention. The 8251 has facilities that allow the programmer to "read" the status of the device at any time during the functional operation.

A normal "read" command is issued by the CPU with the C/D input at one to accomplish this function.

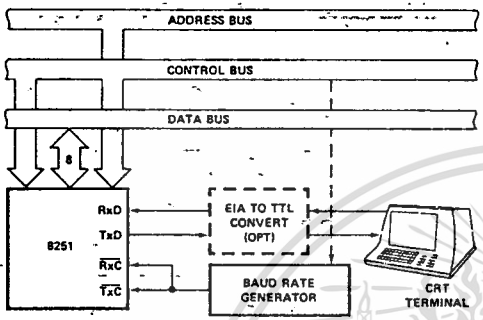
Some of the bits in the Status Read Format have identical meanings to external output pins so that the 8251 can be used in a completely Polled environment or in an interrupt driven environment.



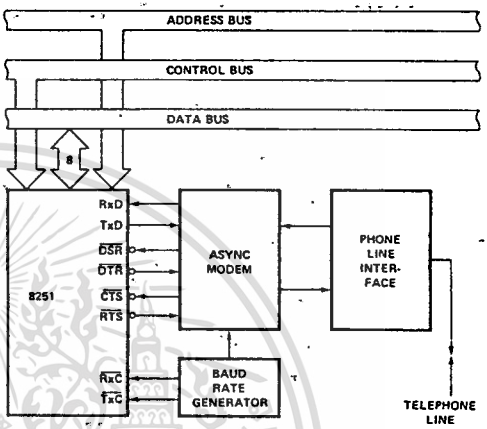
Status Read Format

SILICON GATE MOS 8251

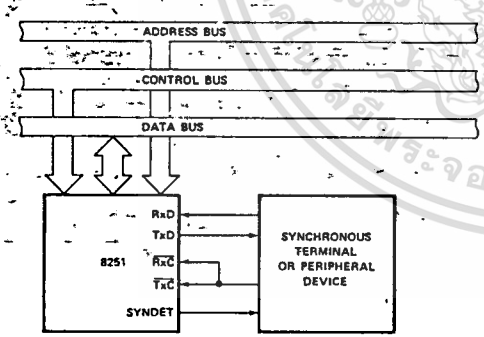
APPLICATIONS OF THE 8251



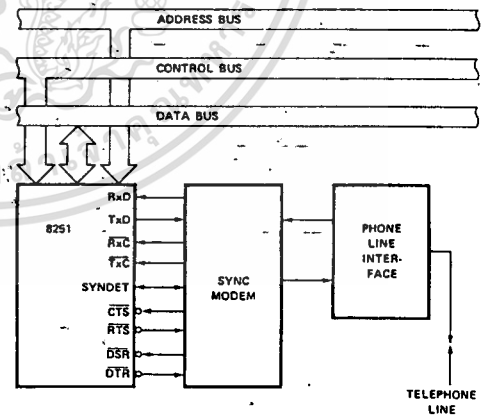
Asynchronous Serial Interface to CRT Terminal, DC-9600 Baud



Asynchronous Interface to Telephone Lines



Synchronous Interface to Terminal or Peripheral Device



Synchronous Interface to Telephone Lines

SILICON GATE MOS 8251

D.C. Characteristics: $T_A = 0°C$ to $70°C$; $V_{CC} = +5V \pm 5\%$; $V_{SS} = 0V$

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
V_{IL}	Input Low Voltage			.8	V	
V_{IH}	Input High Voltage	2.0			V	
V_{OL}	Output Low Voltage			.4	V	I_{OL}=2.0mA (DB_{0-7}), 1.6mA (Others)
V_{OH}	Output High Voltage	2.4			V	I_{OH}=150μA (DB_{0-7}), 100μA (Others)
I_{CC}	Power Supply Current		80		mA	
I_{LI}	Input Load Current		10		μA	V_{IN}=0V to 5.25V
I_{LOL}	Output Leakage Current (DB Low)		-100		μA	V_{OUT}=0.4V
I_{LOH}	Output Leakage Current (DB High)		+10		μA	V_{OUT}=V_{CC}

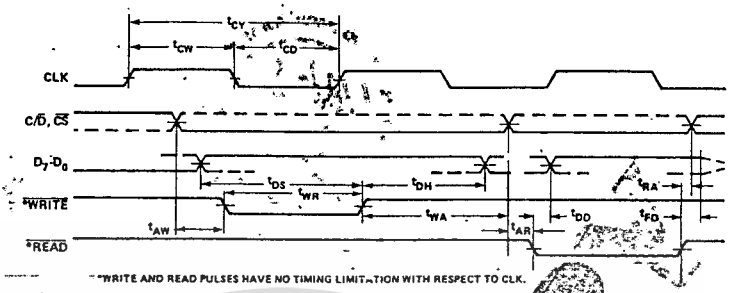
A.C. Characteristics: $T_A = 0°C$ to $70°C$; $V_{CC} = +5V \pm 5\%$; $V_{SS} = 0V$

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
t_{CY}	Clock Period		480		ns	
t_{CW}	Clock Pulse Width		220		ns	
t_{CD}	Clock Pulse Delay		260		ns	
t_R, t_F	Clock Rise and Fall Time		50		ns	
t_{WR}	Write Pulse Width		400		ns	
t_{DS}	Data Set-Up Time for $\overline{WRITE}$		300		ns	
t_{DH}	Data Hold Time for $\overline{WRITE}$		20		ns	
t_{DD}	Data Delay from $\overline{READ}$		350		ns	C_L = 150pF
t_{FD}	$\overline{READ}$ to Data Floating		160		ns	C_L = 150pF
t_{AW}	Address Stable before $\overline{WRITE}$	0			ns	
t_{WA}	Address Hold Time for $\overline{WRITE}$		40		ns	
t_{AR}	Address Stable before $\overline{READ}$		0		ns	
t_{RA}	Address Hold Time for $\overline{READ}$		0		ns	
t_{DTx}	TxD Delay from Rising Edge of TxC		300		ns	
t_{SRx}	Rx Data Set-Up Time to Sampling Pulse		500		ns	
t_{HRx}	Rx Data Hold Time to Sampling Pulse		6		CLK Period	
f_{Tx}	Transmitter Input Clock Frequency 1X Baud Rate 16X and 64X Baud Rate		 56 615		 KHz KHz	
f_{Rx}	Receiver Input Clock Frequency 1X Baud Rate 16X and 64X Baud Rate		 56 615		 KHz KHz	
t_{Tx}	TxRDY Delay from Center of Data Bit		6		CLK Period	C_L=50pF
t_{Rx}	RxRDY Delay from Center of Data Bit		6		CLK Period	
t_{IS}	Internal SYNDET Delay from Center of Data Bit		6		CLK Period	
t_{ES}	External SYNDET Set-Up Time before Rising Edge of $\overline{RxC}$		6		CLK Period	

THESE SPECIFICATIONS ARE SUBJECT TO CHANGE WITHOUT NOTICE

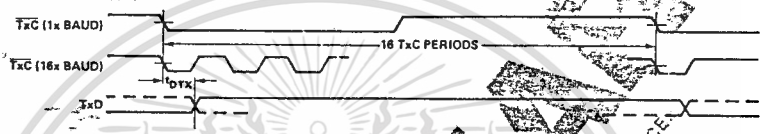
SILICON GATE MOS 8251

READ AND WRITE TIMING

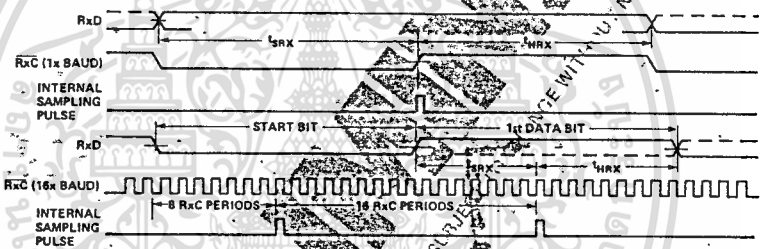


*WRITE AND READ PULSES HAVE NO TIMING LIMITATION WITH RESPECT TO CLK.

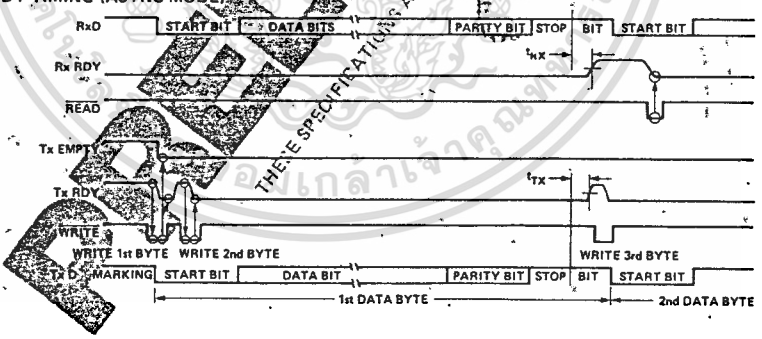
TRANSMITTER CLOCK AND DATA



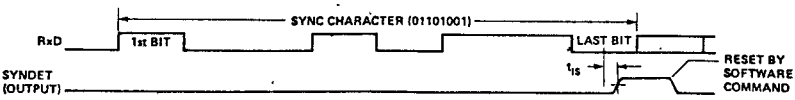
RECEIVER CLOCK AND DATA



Tx RDY AND Rx RDY TIMING (ASYNC MODE)



INTERNAL SYNC DETECT



EXTERNAL SYNC DETECT