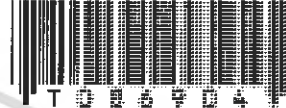


สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

คีย์รหัสสำหรับระบบรักษาความปลอดภัย

KEY CODE FOR SECURITY SYSTEM



ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาอุตสาหกรรมศาสตรบัณฑิต

สาขาเทคโนโลยีอิเล็กทรอนิกส์

ชื่อ

คณะวิศวกรรมศาสตร์

.b.....
.i.....

๒/๖๖

๖๖๖๖/๖๖ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง

๖๖๖๖

เลขหมู่.....

เลขทะเบียน..... 86904

ปีการศึกษา 2538

วัน,เดือน,ปี..... 16 ส.ค. 2552

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หัวข้อปริญญานิพนธ์ : คีย์รหัสสำหรับระบบรักษาความปลอดภัย

KEY CODE FOR SECURITY SYSTEM

โดย : นาย พรชัย ทองเจิม เลขประจำตัว 33-131211

อาจารย์ที่ปรึกษา : ดร.กนก เจนจิระหงส์เวช

ภาควิชา : เทคโนโลยีอุตสาหกรรม

ปีการศึกษา : 2538

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง อนุมัติ
ให้ปริญญานิพนธ์นี้ เป็นส่วนหนึ่งของการศึกษาหลักสูตรปริญญาอุตสาหกรรมศาสตรบัณฑิต

คณะกรรมการตรวจสอบ ปริญญานิพนธ์



..... ประธานกรรมการ

()

..... กรรมการ

()

..... กรรมการ

()

..... กรรมการ

()

..... กรรมการ

()

วันที่.....เดือน.....พ.ศ.....

ลิขสิทธิ์ของ คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหารลาดกระบัง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คีย์รหัสสำหรับระบบรักษาความปลอดภัย

KEY CODE FOR SECURITY SYSTEM

โดย นายพรชัย ทองเจิม

อาจารย์ที่ปรึกษา ดร.กนก เจนจิระพงศ์เวช

บทคัดย่อ

ปริญญานิพนธ์นี้ เป็นการศึกษาถึงการทำงานของไมโครโปรเซสเซอร์ เบอร์ Z-80 และ การเชื่อมต่อเข้ากับอุปกรณ์ภายนอกเช่น หน่วยความจำ พอร์ต อินพุต เอาท์พุต เบอร์ 8255 เพื่อมาควบคุม KEY BORD, LCD DISPLAY นอกจากนี้ยังศึกษาถึง การเขียนโปรแกรมภาษา ASSAMBLY เพื่องานควบคุมโดยผู้ทดลอง ได้ทำ SIGLE BORD ขึ้นมาควบคุม KEY BORD, มอนิเตอร์ แบบ LCD DISPLAY และ KEY LOCK ได้ผลดีพอ ที่จะสามารถนำไปใช้งานจริงได้ คิดว่าโครงการนี้คงเป็นประโยชน์ต่อผู้ที่ต้องการออกแบบ งานควบคุมด้วยไมโครโปรเซสเซอร์ ในโอกาสต่อไป

ABSTRACT

This project introduce the microprocessor Z-80 and its operation by interfacing with memory input/output port of the IC chip 8255 for controlling the keyboard and LCD display. Further more programming used here is the assembly language. Herein, the single board is utilized to control the keyboard, LCD display and keylock. This project will be an asset to designer in microprocessor controlled by various aspected.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

	หน้า
บทที่ 1	
- โครงสร้างไมโครโปรเซสเซอร์ Z-80	4
- ซีพียู ใช้งานทั่วไป	6
- แอคคิวมูเลเตอร์ และซีพียูสถานะ	7
- ซีพียูใช้งานเฉพาะอย่าง	13
- ขาและสัญญาณการเชื่อมต่อของ ซีพียู Z-80	18
- โคอะกแกรมเวลาของไมโครโปรเซสเซอร์	22
บทที่ 2	
- การเชื่อมต่อ ไมโครโปรเซสเซอร์	33
- หน่วยความจำ	33
- ชนิดของหน่วยความจำ	34
- การต่อหน่วยความจำกับ ซีพียู Z-80	35
- ลักษณะภายในหน่วยความจำแบบรวม	36
- ลักษณะภายในหน่วยความจำแบบแรม	38
- การต่อหน่วยความจำกับ ซีพียู Z-80	40
บทที่ 3	
- อินพุท/เอาต์พุท พอร์ต	42
- ขาต่าง ๆ ของ 8255	43
- การเชื่อมต่อ Z-80 กับ 8255	44
- การทำงานของ 8255 ในโหมด 0	45
- การต่ออุปกรณ์ภายนอกกับพอร์ต I/O 8255	47
- วงจรสมบูรณและรายละเอียดการประยุกต์ ใช้ไมโครโปรเซสเซอร์ Z-80	50
บทที่ 4	
- วิธีการใช้งาน	54
- PROGRAM ควบคุมการทำงาน	55

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1

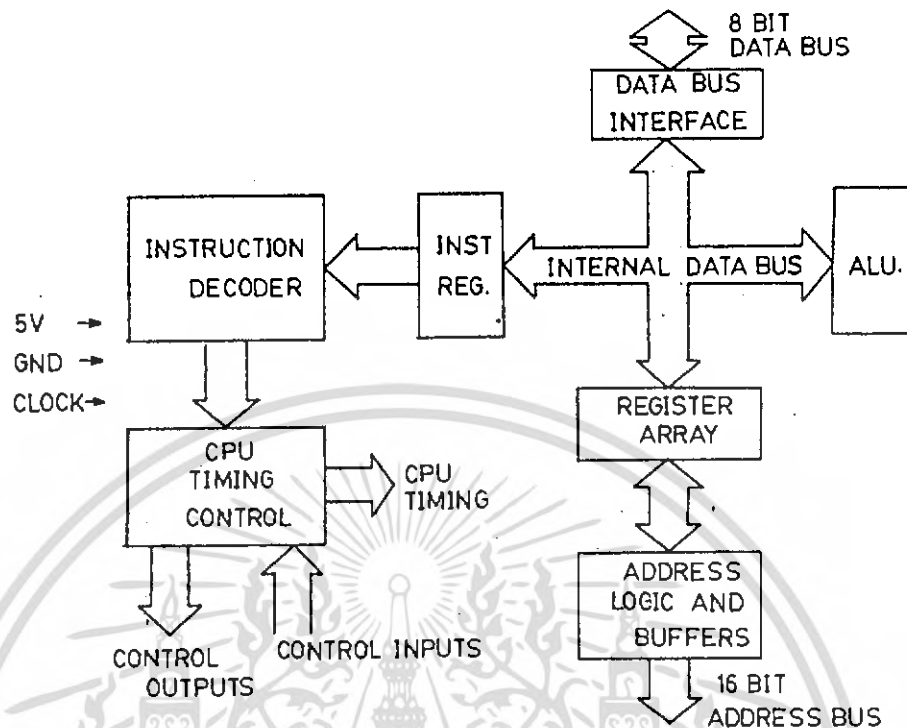
ไมโครโปรเซสเซอร์ (Z-80)

ไมโครโปรเซสเซอร์เบอร์ Z-80 เป็นไมโครโปรเซสเซอร์ขนาด 8 บิต ที่ได้พัฒนา มาจากไมโครโปรเซสเซอร์ เบอร์ 8080 ของบริษัทอินเทล โดยทำ การแก้ไขข้อบกพร่อง บางอย่างของไมโครโปรเซสเซอร์เบอร์ 8080 เช่นทำให้มีคำสั่งมากขึ้น มีวิธีอ้างอิงถึงตำแหน่งข้อมูล (Addressing Mode) ใหม่และมีระบบฮาร์ดแวร์ที่มีความสามารถ และมีความสะดวกในการใช้ งานมากขึ้น นอกจากนี้ ซีพียู Z-80 ยังสามารถใช้ซอฟต์แวร์ที่ใช้กับ ซีพียู 8080 ได้อีกด้วยเหตุผลใน การศึกษาไมโครโปรเซสเซอร์โดยใช้ซีพียู Z-80 เนื่องจากโครงสร้างของซีพียู Z-80 เป็นที่เข้าใจ ง่ายทั้ง ด้านฮาร์ดแวร์ และการนำไปเชื่อมกับอุปกรณ์ภายนอกรวมทั้งชุดคำสั่งต่างๆ และ ระบบซอฟต์แวร์ ยังเข้าใจได้ง่าย และขีดความสามารถทำงานสูง

โครงสร้างซีพียู Z-80

โครงสร้างซีพียู Z-80 แสดงดังรูปที่ 1 จากรูปเห็นได้ว่ามีซีพียูบัสอยู่ 3 ชนิด คือ บัส ตำแหน่ง บัสข้อมูล และบัสควบคุม ซึ่งบัสเหล่านี้ใช้สำหรับทำการเชื่อมต่ออุปกรณ์ภายนอกและภายใน ไมโครโปรเซสเซอร์จะประกอบด้วยพื้นฐานต่างๆ คือ ALU แอคคิวมูเลเตอร์ รีจิสเตอร์ตำแหน่ง และแฟล็ก นอกจากนี้ยังมีรีจิสเตอร์ใช้งานต่าง ๆ อีกคือ B, C, D, E, H, L, B', C', D', E', H', และ L' ส่วนรีจิสเตอร์ตำแหน่งคือ PC, SP, IX, IV ซึ่งรีจิสเตอร์ต่าง ๆ เหล่านี้จะได้กล่าวถึงรายละเอียดต่อไป ส่วนทางด้านซ้ายสุดคือส่วนของหน่วยควบคุม ซึ่งส่วนนี้มีหน้าที่ถอดรหัสคำสั่งแล้วส่งสัญญาณควบคุมไปยังส่วนต่าง ๆ ทั้งภายในและภายนอก ซีพียู รวมทั้งรับสัญญาณควบคุมภายนอกเข้ามาด้วยสัญญาณควบคุมซีพียู Z-80 ภายนอกมี 13 สัญญาณ ซึ่งอาจแบ่งได้ 2 อย่าง คือ สัญญาณควบคุม ซีพียู (CPU and system control) ซึ่งสัญญาณต่าง ๆ จะทำให้มีผลต่อซีพียูและมีผลต่อระบบไมโครคอมพิวเตอร์ด้วย บัสข้อมูลเป็นบัสที่มีขนาด 8 บิต ที่ใช้เป็นทางเดินของข้อมูลระหว่างไมโครโปรเซสเซอร์กับหน่วยความจำ หรืออุปกรณ์อินพุท/เอาต์พุตต่าง ๆ บัสตำแหน่งเป็นบัสที่มีขนาด 16 บิตเพื่อใช้ในการอ้างอิงหน่วยความจำ ดังนั้นจะทำให้การอ้างอิงหน่วยความจำสามารถทำได้ถึง 2^{16} ตำแหน่ง (65536 ตำแหน่ง หรือ 64 K) คือตำแหน่งที่ 0-65535 บล็อกที่มีเครื่องหมาย +/- ที่อยู่ด้านล่างซ้ายของรีจิสเตอร์ หมายถึง การเพิ่มหรือลดข้อมูล (Increment Decrement) ที่มีอยู่ในรีจิสเตอร์ตำแหน่ง หรือของรีจิสเตอร์ๆ คือ SP, PC, BC, DE, HL ซึ่งรีจิสเตอร์ต่างๆ เป็นรีจิสเตอร์ในการกำหนดตำแหน่งโดยตรง (pure address register) ของการอ้างอิงตำแหน่งหน่วยความจำแบบตรง (direct address mode) ซีพียู Z-80 มีค่าที่เกี่ยวกับอินพุท/เอาต์พุต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 1 โครงสร้างภายในของซีพียู Z-80

โดยเฉพาะไม่ใช้ลักษณะ อินพุท/เอาต์พุท Memory mapped (Memorymap I/O ใช้เป็นส่วนหนึ่งของหน่วยความจำเพื่อเป็นตำแหน่งของอุปกรณ์อินพุท/เอาต์พุท) ในการให้ไมโครโปรเซสเซอร์ทำงานตามที่ต้องการ ทำได้โดยการเขียนโปรแกรมไว้ในหน่วยความจำ จากนั้นให้ไมโครโปรเซสเซอร์ ออกคำสั่ง จากหน่วยความจำเพื่อ มาปฏิบัติคำสั่งต่าง ๆ นั้นเราไม่จำเป็นต้องเข้าถึงส่วนรายละเอียดต่างๆ ของไมโครโปรเซสเซอร์ทั้งหมด โดยเราสนใจเฉพาะรีจิสเตอร์ต่าง ๆ ที่เกี่ยวข้องกับการเขียนโปรแกรมเท่านั้น รีจิสเตอร์ภายในที่สามารถอ่านเขียนได้ถึง 208 บิตโดยแยกเป็น รีจิสเตอร์ขนาด 8 บิต 18 รีจิสเตอร์ และ รีจิสเตอร์ขนาด 16 บิต อีก 4 รีจิสเตอร์ รีจิสเตอร์ต่างๆ ภายใน Z-80 เป็นลักษณะของสแตติกแรม และรีจิสเตอร์เหล่านี้แบ่งออกเป็น 3 ประเภท คือ

1. รีจิสเตอร์ใช้งานทั่วไป (General purpose register)
2. แอคคิวมูเลเตอร์ และ รีจิสเตอร์สถานะ (Accumulator and Flag register)
3. รีจิสเตอร์ใช้งานเฉพาะอย่าง (Special purpose register)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

A	F	A'	F'
B	C	B'	C'
D	E	D'	E'
H	L	H'	L'

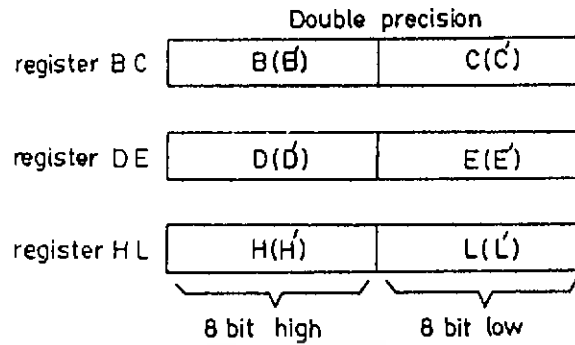
I	R
INDEX REG. IX	
INDEX REG. IY	
STACPOINTER, SP	
PROGRAM COUNTER, PC	

รูปที่ 2 รีจิสเตอร์ของซีพียู Z-80

รีจิสเตอร์ใช้งานทั่วไป (General purpose register) รีจิสเตอร์ ใช้งานทั่วไป ภายในซีพียู Z-80 มี 2 กลุ่มที่เหมือนกัน แต่ละกลุ่มประกอบด้วยรีจิสเตอร์ขนาด 8 บิตจำนวน 6 รีจิสเตอร์ กลุ่มหนึ่งคือกลุ่มรีจิสเตอร์หลัก (Main register set) ซึ่งประกอบด้วย B, C, D, E, H, L ประโยชน์ของรีจิสเตอร์ 2 ชุดนี้คือชุดของรีจิสเตอร์สำรอง (Alternate register set) ซึ่งประกอบด้วย B', C', D', E', H', L' ประโยชน์ของการมีรีจิสเตอร์ 2 ชุดนี้คือ เพื่อให้ผู้เขียนโปรแกรมสามารถสลับระหว่างข้อมูล ในชุดของรีจิสเตอร์ทั้งสองได้อย่างรวดเร็ว ทั้งนี้เพื่อความเร็ว ในการทำงานของโปรแกรมเนื่องจากเข้าถึงข้อมูลของโปรแกรมภายในรีจิสเตอร์ของซีพียู มีความรวดเร็วกว่า การเข้าถึงข้อมูลในหน่วยความจำภายนอก นอกจากนี้รีจิสเตอร์สำรองอาจใช้เพื่อเก็บข้อมูลในรีจิสเตอร์หลักเมื่อ ซีพียูอินเตอร์รัพต์ภายนอก รีจิสเตอร์ ใช้งานทั่วไปนี้อาจเป็นที่เก็บข้อมูลขนาด 8 บิต แบบธรรมดา หรือเก็บข้อมูลขนาด 16 บิต โดยการจับคู่ (Pair) ระหว่างรีจิสเตอร์ก็ได้ ซึ่งคู่รีจิสเตอร์ของรีจิสเตอร์หลักคือ BC, DE, HL ส่วนรีจิสเตอร์อีกชุดหนึ่งคือ BC', DE' และ HL' ประโยชน์ของคู่รีจิสเตอร์ เช่น ใช้เป็นตัวกำหนดตำแหน่ง (Pointer) ของข้อมูลในหน่วยความจำ หรือใช้เพื่อกระทำทางคณิตศาสตร์แบบ 16 บิต ที่เรียกว่า Double-precision arithmetic

การกระทำทางคณิตศาสตร์แบบ Double-precision หมายถึงการบวก การลบ เพิ่มค่าขึ้น 1 (Increment) การลดค่าลง 1 (Decrement) ของข้อมูลขนาด 16 บิตที่อยู่ในคู่รีจิสเตอร์ หรืออยู่ในรีจิสเตอร์แบบ 16 บิต ซึ่งการกระทำทางคณิตศาสตร์หรือลอจิก ส่วนมากของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

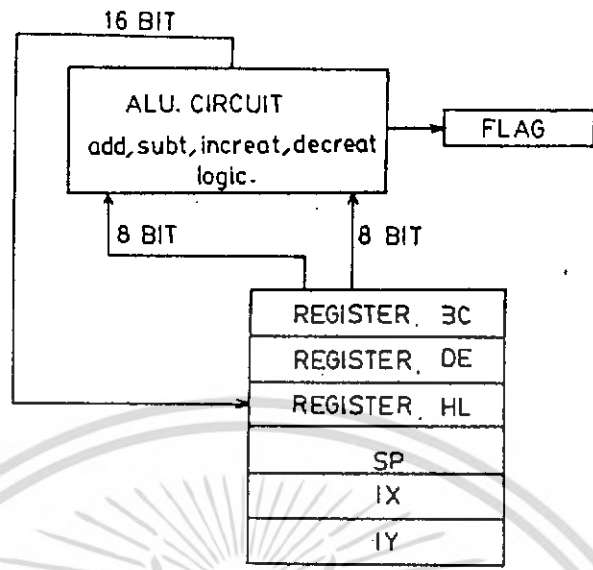


รูปที่ 3 การจับคู่รีจิสเตอร์

ซีพียู Z-80 เป็นกระทำของข้อมูลขนาด 8 บิต ซึ่งเรียกว่า Single-precision แต่การกระทำทางคณิตศาสตร์ขนาด 16 บิต ของซีพียู Z-80 นี้สามารถเปลี่ยนแปลงตัวชี้ (Pointer) ได้สะดวก โดยการเปลี่ยนแปลงข้อมูล ภายในคู่รีจิสเตอร์ นอกจากนี้ยังเห็นได้ว่าถ้าไม่มีการบวกเลขแบบ 16 บิต จะต้องทำการบวกแบบ 8 บิตถึง 2 ครั้งซึ่งทำให้การเขียนโปรแกรมยุ่งยากขึ้น รูปที่ 3 แสดงถึงการจับคู่รีจิสเตอร์ เพื่อการกระทำแบบ Double-precision

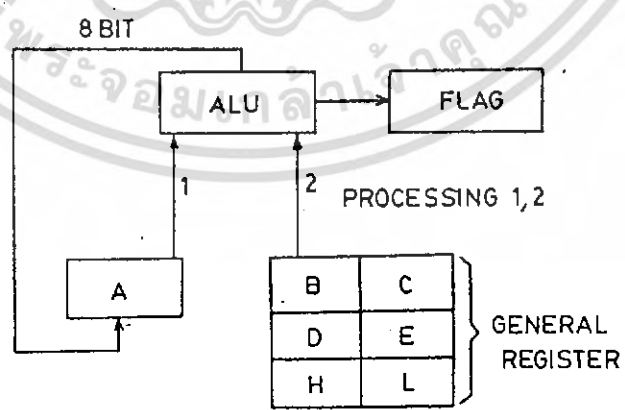
แอกคิวมูลเตอร์และรีจิสเตอร์สถานะ (Accumulator and Flag reg.) จากโครงสร้างของซีพียูจะเห็นได้ว่า โปรเซสเซอร์ทั่วไปจะต้องมี แอกคิวมูลเตอร์ เพื่อเก็บข้อมูลตัวกระทำหลัก (Main operand) ตัวหนึ่ง เพื่อที่จะกระทำทางคณิตศาสตร์ หรือลอจิกกับข้อมูลที่มาจากรีจิสเตอร์ต่าง ๆ หรือข้อมูลที่มาจากหน่วยความจำภายนอก และผลจากการกระทำนั้นก็จะถูกนำมาเก็บไว้ใน แอกคิวมูลเตอร์ ส่วนรีจิสเตอร์สถานะหรือแฟล็กใช้เพื่อแสดงสถานะของแอกคิวมูลเตอร์ หรือรีจิสเตอร์ที่เกี่ยวข้องกับการกระทำแสดงได้ดังรูปที่ 5 ไมโครโปรเซสเซอร์ Z-80 มีรีจิสเตอร์ที่ทำหน้าที่ แอกคิวมูลเตอร์อยู่ 2 ตัวและแฟล็ก 2 ตัว แต่มีอย่างละตัวเท่านั้นที่สามารถทำงานกับหน่วยคณิตศาสตร์ และลอจิกได้คือ แอกคิวมูลเตอร์และแฟล็กที่อยู่ในส่วนของรีจิสเตอร์หลักซึ่งเรียกว่า A และ F ส่วนรีจิสเตอร์ A' และ F' ที่อยู่ในรีจิสเตอร์สำรอง ใช้เป็นที่เก็บข้อมูลชั่วคราวของแอกคิวมูลเตอร์ A และ F เมื่อถึงคราวจำเป็นเท่านั้น ซึ่งคู่ของแอกคิวมูลเตอร์และแฟล็กทั้งสองชุดนี้ คือ AF และ AF' สามารถสลับข้อมูลในรีจิสเตอร์ทั้งคู่ ได้โดยคำสั่งในกลุ่มแลกเปลี่ยนข้อมูล (Exchange group) เหมือนการแลกเปลี่ยนข้อมูล ระหว่างรีจิสเตอร์หลักและรีจิสเตอร์สำรองของกลุ่มรีจิสเตอร์ทั่วไป รีจิสเตอร์สถานะหรือแฟล็กรีจิสเตอร์ขนาด 8 บิต ซึ่งแต่ละบิตใช้แสดงสถานะซีพียู ที่จะเกิดหลังกระทำคำสั่ง ทางคณิตศาสตร์ หากคำสั่งทางลอจิกหรือกระทำคำสั่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



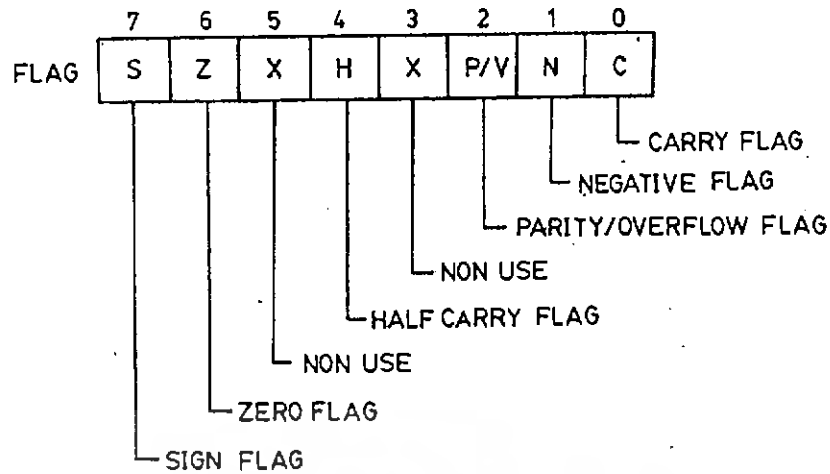
รูปที่ 4 การกระทำทางคณิตศาสตร์แบบ Double precision

อื่น ๆ ของซีพียูต่าง ๆ ของแฟลกใช้แสดงสถานะที่แตกต่างกัน ซึ่งรูปแบบของรีจิสเตอร์นี้แสดงดังรูปที่ 6 ของซีพียู กลุ่มคำสั่งที่มีผลต่อแฟลก เช่น คำสั่งในกลุ่มการกระทำทางคณิตศาสตร์ และ ลอจิกกลุ่มของการเลื่อนหมุนข้อมูล กลุ่มของการทดสอบบิต เป็นต้น และในการทำงานข้าม (Jump or Branch) แบบมีเงื่อนไขที่ต้องหรือต้องการหรือไม่ รายละเอียดของแฟลกต่าง ๆ จะอธิบายได้ดังต่อไปนี้



รูปที่ 5 การกระทำทางคณิตศาสตร์และลอจิก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 6 รูปแบบของแฟล็ก

แฟล็กตัวทศ (Carry flag)

แฟล็กตัวทศ หรือ แฟล็ก C แฟล็กนี้จะเซตหรือรีเซต ขึ้นอยู่กับการทำงานของคำสั่งกลุ่มต่าง ๆ เช่นกลุ่มคณิตศาสตร์ กลุ่มของการเลื่อนและหมุนข้อมูลและตัวอย่าง เช่นการบวกเลขฐานสอง 8 บิต สองจำนวน หรือการลบเลข 8 บิต สองจำนวนดังรูปที่ 7



รูปที่ 7 (A),(B) - ผลของแฟล็ก C หลังจากคำสั่งบวก

(C),(D) - ผลของแฟล็ก C หลังจากคำสั่งลบ

จากรูปที่ 7 (A) เห็นได้ว่าผลของการได้ผลรับเกิน 8 บิต ดังนั้นบิตที่เกินมาเป็น 9 บิต ที่จะทำให้ C เซตเป็น 1 ส่วนรูปที่ 7(B) ผลของการบวกไม่เกิน 8 บิต แฟล็ก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

C ซีเซตเป็น 0 ในทำนองเดียวกันเมื่อทำคำสั่งลบแฟลก C เป็นแฟลกที่แสดงการขอยืม (borrow) ดังรูปที่ 7 (D) เมื่อมีการลบมีการขอยืมแฟลก C นี้ก็จะซีเซตเป็น 0 แต่ถ้า ซีพียูทำคำสั่งกลุ่มลอจิก ไม่ว่าจะคำสั่งใดก็ตาม แฟลก C นี้จะซีเซตเป็น 0 เสมอ

แฟลกการลบ (Negative flag)

แฟลกการลบ จะเป็นเซตหรือซีเซตขึ้นอยู่กับคำสั่งที่ใช้เกี่ยวข้องกับคณิตศาสตร์ นั่นคือเมื่อทำคำสั่งประเภทการลบ แฟลกนี้จะเซต เช่นเมื่อ ซีพียู ทำคำสั่ง SUP CP DEC เป็นต้น แต่ถ้า ซีพียูทำคำสั่งประเภทการบวก เช่น ADD INC แฟลกนี้จะซีเซต แฟลกการลบนี้มีประโยชน์มากเมื่อทำคำสั่ง DAA จะทำให้ ซีพียูรู้ว่าหน้านี้เป็นคำสั่งประเภทการบวก หรือการลบ เพื่อจะได้ทำการบวกหรือลบด้วย 6 เพื่อให้ได้เลข BCD ที่ต้องการ

แฟลกพาริตี/ค่าเกิน (Parity/Overflow flag)

เป็นแฟลกที่อยู่บิต 2 ของแฟลกรีจิสเตอร์ แฟลกนี้มีประโยชน์สองอย่างคือ เมื่อ ซีพียูทำคำสั่งลอจิก แฟลกนี้เป็นแฟลกพาริตี แต่ถ้าซีพียูทำคำสั่งคณิตศาสตร์แฟลกนี้จะเป็ค่าเกิน ในกรณีของแฟลกพาริตี บิตนี้จะเซตเป็น 1 ทางผลของการกระทำทางลอจิก แล้วทำให้จำนวน 1 ของแอกคิวมูลเตอร์ เป็นจำนวนคู่ (Even parity) และซีเซตเป็น 0 ถ้าจำนวน 1 ในแอกคิวมูลเตอร์เป็นจำนวนคี่ (Odd parity) ดังตัวอย่างรูป 8

ในกรณีทำงานเป็นแฟลกค่าเกิน (Overflow หรือ v) คือเมื่อ ซีพียูทำคำสั่งทางคณิตศาสตร์ แฟลก V นี้จะเซตเมื่อผลลัพธ์จากการกระทำที่มีค่าเกิน และจะซีเซตผลการกระทำที่มีค่าไม่เกิน ค่าเกินนี้พิจารณาได้จากผลลัพธ์ที่จะได้ ที่แอกคิวมูลเตอร์ เนื่องจากตัวเลขที่ใช้ไมโครโปรเซสเซอร์เป็น 2's Complement Sign Binary นั่นคือ บิตสูงสุดเป็นบิตเครื่องหมาย และที่เหลือเป็นขนาดของตัวเลข ดังนั้นถ้าแอกคิวมูลเตอร์มีขนาด 8 บิต มักแสดงเลขฐานสองได้ระหว่าง 0111 111 (+127) ถึง 1000 0000 (-128) ดังนั้น ถ้าผลการกระทำทางคณิตศาสตร์สูงกว่า +127 หรือต่ำกว่า -128 จะทำให้แฟลก V เซต แต่ถ้าผลการกระทำอยู่ระหว่าง +127 ถึง-128 จะซีเซต ดังตัวอย่างดัง รูปที่ 9

```

10110110
AND 1000101
ผลลัพธ์ 1000100 --> "1" มี 2 บิต เป็นจำนวนคู่ : แฟลก P=1

01101001
    
```

OR 00110001
 ผลลัพธ์ 01111001 --> "1" มี 5 บิต เป็นจำนวนคือ : แผลก P=0

รูปที่ 8 เมื่อแผลก P/V ทำหน้าที่เป็น พาริตีแผลก

+127 01111111
+64 +01000000 (A)
 +91 V[1] 10111111

-125 10000011
-126 + 10000010 (B)

-251 V[1] 00000101
 +32 00100000
+32 + 00100000 (C)
 +64 V[0] 01000000

รูปที่ 9 แสดงการทำงานของแผลก V

รูปที่ 9 (A)และ(B) แสดงให้เห็นว่าแผลก V เซตสิ่งเกิดได้ว่าเป็นกรณีที่ แผลก V เซตเป็นไปได้อันนี้คือ ถ้าบิตเครื่องหมายทั้งตัวตั้งและตัวบวกเป็น "0" (+) แล้ว ผลลัพธ์มีบิตเครื่องหมายเป็น "1"(-) แสดงว่าเกิดค่าเกิน (รูป 9 (A))ในทำนองเดียวกันถ้าบิต เครื่องหมายของตัวตั้งและตัวบวกเป็น "1"(-) ทั้งคู่แล้วผลลัพธ์มีบิตเครื่องหมายเป็น "0"(+) แสดงว่าเกิดค่าเกิน นอกเหนือจากนี้ แผลก V จะซีเซต

แผลกตัวทศช่วย (Half carry flag)

แผลกตัวทศช่วย มีประโยชน์ในการกระทำทางคณิตศาสตร์ของตัวเลข BCD โดยที่ แผลกนี้จะมีการเซต เมื่อมีการทศหรืออิม ของการกระทำตัวเลข 4 บิต ทางด้านต่ำ (4 Least Sinificant Bit) ประโยชน์ของแผลกนี้ ใช้เพื่อการปรับผลลัพธ์ของตัวเลขที่เกิดจากการกระทำ ทางคณิตศาสตร์ของเลข BCD ตัวอย่างของการเซตแผลกตัวทศนี้ แสดงได้ดังรูปที่ 10

$$\begin{array}{r}
 0001\ 0001 \\
 +\ 0010\ 0010 \quad (A) \quad H=0 \\
 \hline
 \text{ผลลัพธ์} \quad 001100011
 \end{array}$$

$$\begin{array}{r}
 0100\ 1001 \\
 +\ 0010\ 1000 \quad (B) \quad H=1 \\
 \hline
 \text{ผลลัพธ์} \quad 0111\ 0001
 \end{array}$$

รูปที่ 10

แฟลกช่วยทด และแฟลกการลบ เป็นแฟลกเพื่อใช้ประโยชน์ส่วนใหญ่ในการกระทำทางคณิตศาสตร์ของเลข BCD

แฟลกศูนย์ (Zero flag)

แฟลกศูนย์ ถ้าผลของการปฏิบัติคำสั่งแล้วทำให้ผลในรีจิสเตอร์ที่อ้างถึงมีค่าเป็น 0 และรีเซตถ้าผลลัพธ์ที่อ้างถึงไม่เป็น 0 ดังตัวอย่างรูปที่ 11

$$\begin{array}{r}
 01101001 \\
 \text{AND } 00000000 \\
 \hline
 \text{ผลลัพธ์} \quad 00000000 \quad \text{แฟลก } Z=1
 \end{array}$$

รูปที่ 11 แสดงการทำงานของแฟลก Z

แฟลกศูนย์ นี้มีประโยชน์มากในการทำงานเคลื่อนแบบมีเงื่อนไขโดยใช้คำสั่งต่าง ๆ เช่น JP Z,nn JR Z,e JR NZ,nn เป็นต้น แฟลกศูนย์โดยทั่วไปใช้ในประโยชน์ดังนี้ เช่น

- ใช้สำหรับเปรียบเทียบเลขสองจำนวน
- ใช้ทดสอบบิต
- ใช้ตรวจสอบการลตค่าตัวนับ

แฟลกเครื่องหมาย (Sign flag)

แฟลกนี้ใช้ในการตรวจสอบบิตเครื่องหมาย ซึ่งเครื่องหมายบวกหรือลบนี้จะแสดงโดย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตัวเลขบิตที่ 7 ของรีจิสเตอร์แฟล็ก ถ้าบิต 7 เท่ากับศูนย์แสดงว่าเป็นค่าบวก ถ้าบิต 7 เท่ากับ 1 แสดงว่าเป็นลบ แฟล็กเครื่องหมายจะเซตถ้าหลังจากการปฏิบัติคำสั่งให้ผลลัพธ์ ที่มีเครื่องหมายเป็นลบ และรีเซตเมื่อผลลัพธ์ได้เครื่องหมายเป็นบวก

รีจิสเตอร์ใช้งานเฉพาะอย่าง (Special purpose register)

ในไมโครโปรเซสเซอร์ Z-80 มีรีจิสเตอร์ใช้งานเฉพาะอย่างซึ่งประกอบด้วย PC, SP, IX, IV, I และ R ซึ่งรีจิสเตอร์ต่าง ๆ เหล่านี้จะได้กล่าวถึงต่อไป

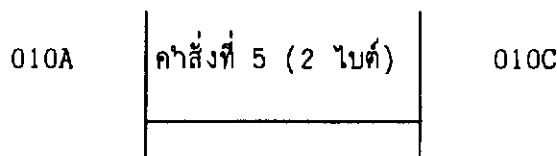
โปรแกรมเคาเคอร์ (PC)

เป็นรีจิสเตอร์ขนาด 16 บิต มีหน้าที่เก็บตำแหน่งต่อไปที่จะถูกเฟรชออกมาจากหน่วยความจำ คำสั่งของซีพียู Z-80 เป็นคำสั่งที่มีความยาว 1, 2, 3 หรือ 4 ไบต์ ดังนั้นข้อมูลใน PC อาจถูกเพิ่มขึ้นครั้งละ 1, 2, 3 หรือ 4 ไบต์โดยอัตโนมัติ ขึ้นอยู่กับความยาวของคำสั่งที่ถูกปฏิบัติก่อนหน้านั้น ถ้าคำสั่งที่มีความยาวต่าง ๆ กัน ถูกวางไว้เป็นลำดับตั้งรูปที่ 12 คำใน PC หลังจากทีคำสั่งนั้นถูกปฏิบัติไปแล้วคือค่าที่อยู่ทางด้านขวา

ตำแหน่งหน่วย ความจำภายนอก	ข้อมูล หน่วยความจำ	ข้อมูลใน PC หลัง จากจบคำสั่ง
------------------------------	-----------------------	---------------------------------

0100	คำสั่งที่ 1 (1 ไบต์)	0101
0101	คำสั่งที่ 2 (2 ไบต์)	0103
0103	คำสั่งที่ 3 (3 ไบต์)	0106
0106	คำสั่งที่ 4 (4 ไบต์)	010A

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



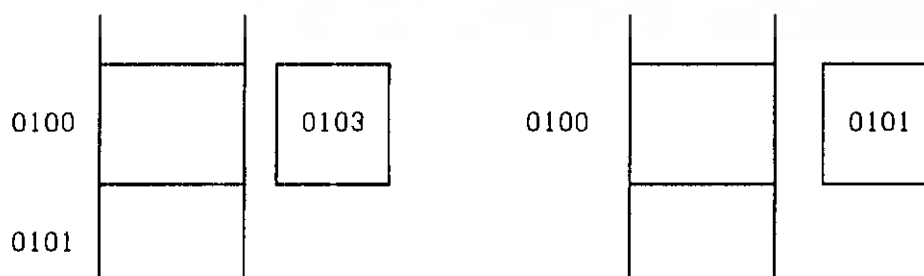
รูปที่ 12 แสดงการเพิ่มของข้อมูลในโปรแกรมเคาเตอร์

จะเห็นได้ว่าค่าของ PC จะเพิ่มขึ้นตามลำดับเรื่อย ๆ นอกจากว่าเมื่อ ซีพียู ทำคำสั่งในกลุ่มทำงานข้าม (Jump) มันจะเปลี่ยนไปตามตำแหน่งที่ต้องกระโดดไป โดยคำสั่งข้ามนั้น ๆ และอีกกรณีหนึ่งคือ เมื่อ ซีพียูมีการ Interrupt ก็จะทำให้ค่าของ PC ไม่เรียงลำดับเช่นกัน และข้อมูลใน PC ไม่สามารถนำมากระทำทางคณิตศาสตร์และลอจิกได้

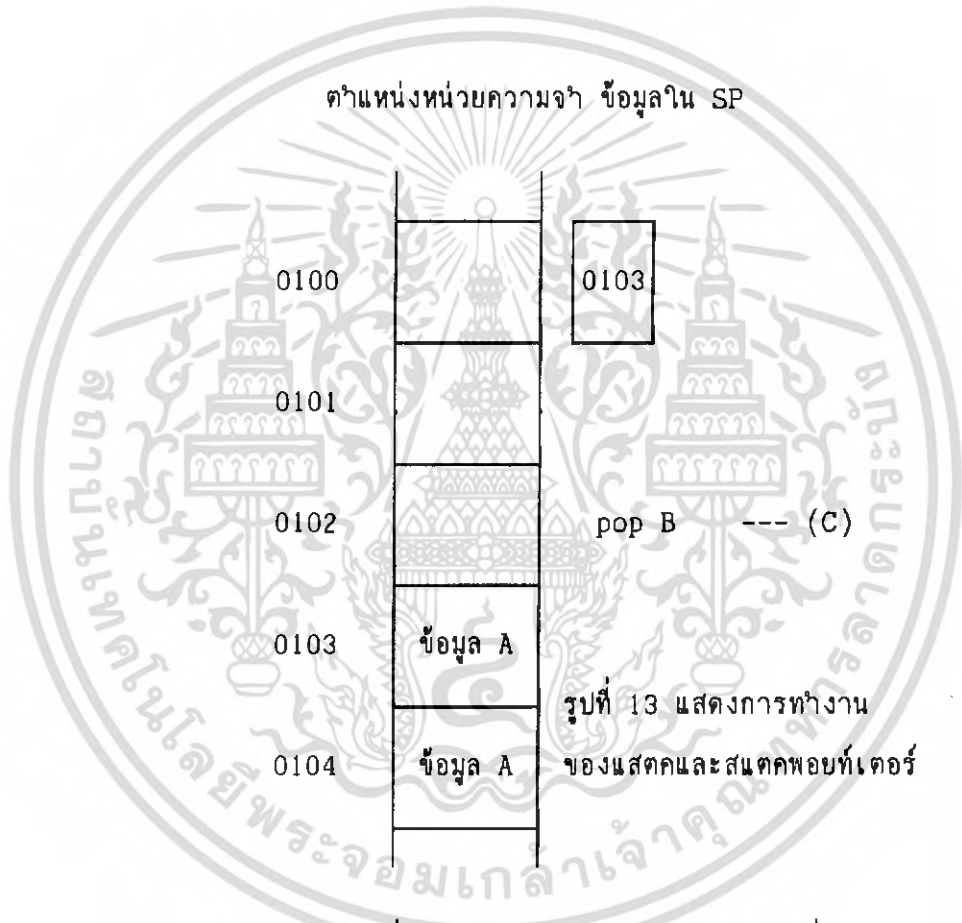
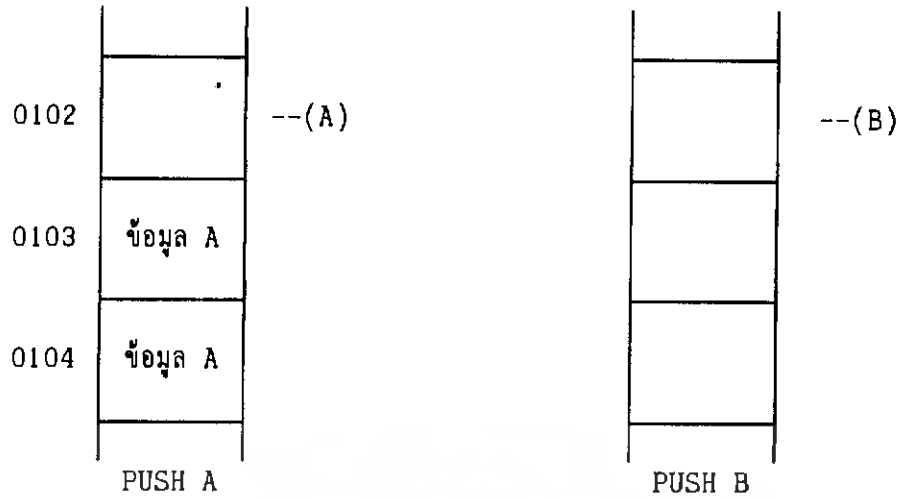
สแตคพอยท์เตอร์ (SP)

สแตคพอยท์เตอร์ของซีพียู Z-80 เป็นรีจิสเตอร์ที่มีขนาด 16 บิต มีหน้าที่ทำหน้าที่ใช้ชี้ตำแหน่งบนสุดของสแตค ซึ่งสแตคของไมโครโปรเซสเซอร์ Z-80 เป็นสแตคแบบใช้หน่วยความจำภายนอก (External memory stack) โดยใช้ส่วนหนึ่งของแรม (RAM) ซึ่งหน้าที่ของสแตคนี้คือใช้เก็บข้อมูลชั่วคราว ซึ่งข้อมูลอาจมาจากรีจิสเตอร์ต่าง ๆ ข้อมูลที่เก็บจากสแตคนี้ อาจเก็บเข้าหรือนำออกโดยอัตโนมัติ เช่นเมื่อซีพียูทำคำสั่ง CALL หรือเมื่อซีพียูถูกขัดจังหวะมักจะเก็บค่าของ PC เข้าในสแตคโดยอัตโนมัติ และเมื่อซีพียูทำคำสั่ง RET RETI หรือ RETN ข้อมูลของ PC ที่เก็บไว้ในสแตคก็จะนำออกมาโดยอัตโนมัติเช่นกัน และนอกจากนี้ข้อมูลจากรีจิสเตอร์ต่าง ๆ ยังสามารถนำเข้ามาหรือนำออกจากสแตคโดยใช้คำสั่งก็ได้ เมื่อใช้คำสั่ง PUSH qq จะให้นำข้อมูลในรีจิสเตอร์เข้ามาเก็บในสแตค ส่วนคำสั่ง POP qq เป็นการนำข้อมูลในสแตคออกมาใส่ไว้ในรีจิสเตอร์ที่อ้างถึง ซึ่งในทุกกรณีของการนำข้อมูลเข้าหรือออกในสแตคนี้ ตำแหน่งบนสุดของสแตคพอยท์เตอร์เสมอ ตัวอย่างดังรูปที่ 13

ตำแหน่งหน่วยความจำ ข้อมูลใน SP ตำแหน่ง หน่วยความจำ ข้อมูลใน SP



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



สังเกตได้ว่าข้อมูลที่อยู่ใน SP คือตำแหน่งสูงของสแตคที่ข้อมูลอยู่เสมอ

อินเด็กซ์รีจิสเตอร์ (Index register)

ไมโครโปรเซสเซอร์ Z-80 มีอินเด็กซ์รีจิสเตอร์ 16 บิต 2 ตัวคือ IX, IY ซึ่งทั้งสองตัวนี้มีลักษณะการทำงานที่เหมือนกันทุกประการ คือ ใช้ในคำสั่งที่อ้างถึงข้อมูลแบบอินเด็กซ์โดยที่ตำแหน่งของข้อมูลใช้ (Effective address) ที่อยู่ในหน่วยความจำ กำหนดโดยนำข้อมูลที่อยู่ในอินเด็กซ์รีจิสเตอร์ ที่ใช้เป็นตำแหน่งฐาน (Base address) บวกกับระยะห่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(Displacement) ขนาด 8 บิต ที่บรรจุในส่วนของคำสั่ง เขียนเป็นสมการได้คือ

$$\text{Effective address} = \text{Base address} + \text{Displacement}$$

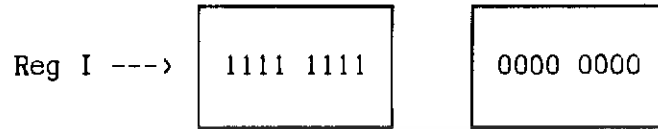
อินเตอร์รัพต์เวกเตอร์รีจิสเตอร์ (I)

รีจิสเตอร์ขนาด 8 บิต ที่สามารถโหลดข้อมูลขนาด 8 บิต ลงไปได้ข้อมูลในรีจิสเตอร์ I นี้เป็นส่วนหนึ่งของตำแหน่งของหน่วยความจำที่อ้างถึง โดยใช้เป็นข้อมูล 8 บิตด้านสูง (High byte) เพื่อใช้ชี้ตำแหน่ง ของตารางเวกเตอร์ (Vector) และเมื่อนำข้อมูลอีก 8 บิตที่ใช้ชี้ตำแหน่งหน่วยความจำในตารางเวกเตอร์ ข้อมูล 8 บิตทางด้านต่ำที่จะนำมาต่อเป็นข้อมูลส่งมาจากอุปกรณ์อินพุท/เอาต์พุท เมื่ออุปกรณ์อินพุท/เอาต์พุท ส่งสัญญาณเพื่อทำการอินเตอร์รัพต์ การทำงานตามปกติของซีพียู และถ้าซีพียูตอบรับการอินเตอร์รัพต์นั้น อุปกรณ์ตัวที่ส่งสัญญาณการอินเตอร์รัพต์ จะส่งข้อมูลขนาด 8 บิตมาให้ ซึ่งข้อมูลขนาด 8 บิต ทางด้านต่ำมารวมกับรีจิสเตอร์ I เพื่อชี้ตำแหน่งตารางเวกเตอร์ และข้อมูลที่อยู่ในหน่วยความจำตามตำแหน่งในตารางที่อ้างถึง 2 ไบต์ จะเป็นตำแหน่งที่ เริ่มต้นของโปรแกรมบริการอินเตอร์รัพต์ของตัวอุปกรณ์นั้น ๆ ลักษณะการทำงานดังกล่าวนี้ อธิบายได้ดังรูปที่ 14 จากรูปกำหนดข้อมูลในรีจิสเตอร์ I คือ 02H เมื่อข้อมูลทั้งสองรวมกันได้ FF02H ซึ่งเป็นตำแหน่งของหน่วยความจำ ภายนอกที่อ้างถึงหน่วยความจำในส่วนนี้ เรียกว่า ตารางเวกเตอร์ โดยตารางนี้จะบรรจุตำแหน่งของโปรแกรมบริการ การอินเตอร์รัพต์ไว้ ดังนั้นค่าของรีจิสเตอร์ I หนึ่งค่าสามารถกำหนดตำแหน่งการบริการของ การอินเตอร์รัพต์ได้ถึง 128 ตำแหน่ง จากรูปที่ 14 ตำแหน่งที่ FF02H และ FF03H จะบรรจุตำแหน่ง เริ่มต้นของโปรแกรมบริการอุปกรณ์ที่ส่งสัญญาณ การอินเตอร์รัพต์เข้ามา ดังนั้น โปรแกรมบริการของอุปกรณ์จึงอยู่ที่ตำแหน่ง B000H รีจิสเตอร์ I นี้ ใช้เพื่อทำการอินเตอร์รัพต์ในโหมด 2 ซึ่งการอินเตอร์รัพต์ของซีพียู Z-80 ซึ่งมี 3 โหมด

รีเฟรชรีจิสเตอร์ (R)

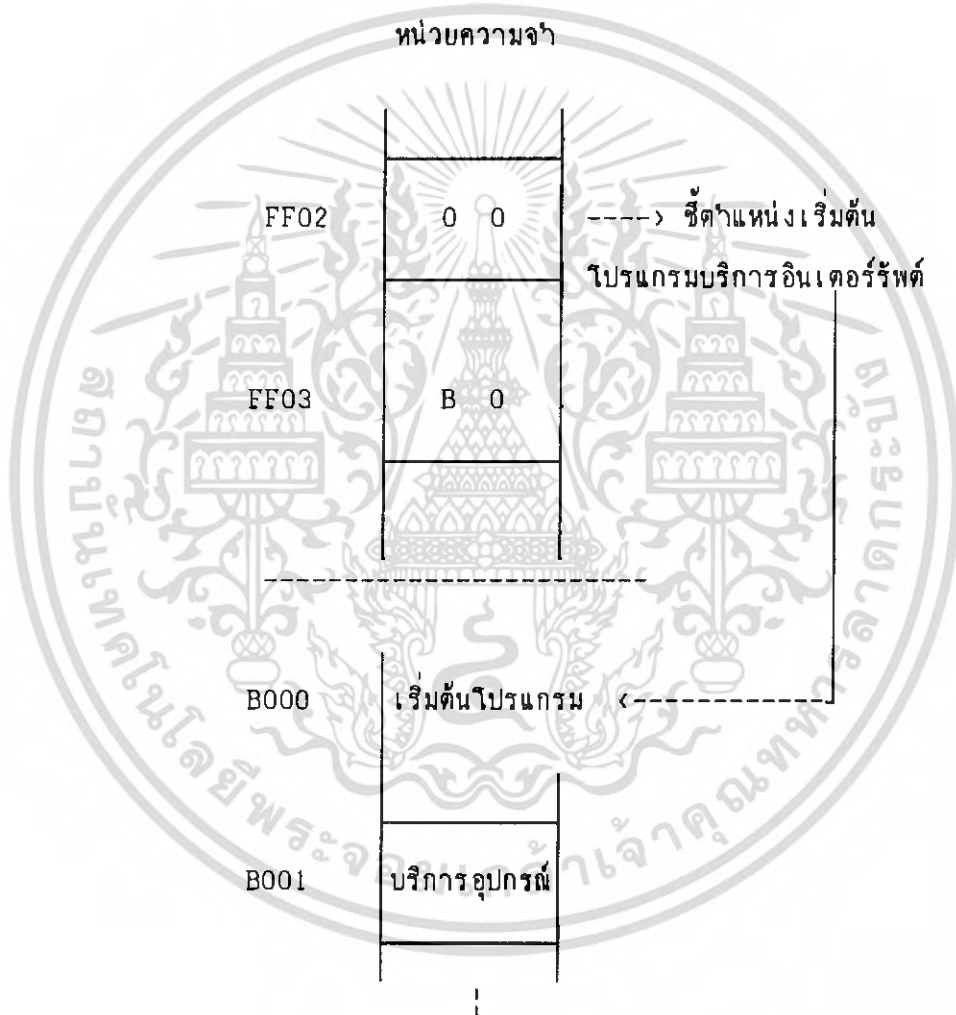
เมื่อซีพียู Z-80 ต่อหน่วยความจำแบบไดนามิก ซึ่งหน่วยความจำแบบนี้จะต้องมีการอัดซ้ำ (Refresh) ตลอดเวลา (ปกติทุก ๆ 2 ms) เพื่อให้ข้อมูลในหน่วยความจำนี้คงอยู่ ซีพียู Z-80 นี้ใช้รีจิสเตอร์ R เพื่อบรรจุตำแหน่งของหน่วยความจำ เพื่อทำการอัดซ้ำ รีจิสเตอร์ R มี ขนาด 7 บิต และข้อมูลในรีจิสเตอร์จะ เพิ่มขึ้นทีละ 1 โดยอัตโนมัติหลังจากที่เพทต์คำสั่งหนึ่ง ๆ แล้วข้อมูลใน

ซีพรีซีเอสเตอร์ ถูกส่งออกไปทางด้านล่างของบัสตำแหน่ง (A6-A0) ในขณะที่สัญญาณรีเฟรชแอดดรีฟ และซีพียูไม่ได้ Access หน่วยความจำรีซีเอสเตอร์ R โดยปกติผู้เขียนโปรแกรมไม่จำเป็นต้องใช้ นอกจากการตรวจสอบเท่านั้น



ข้อมูล 16 บิต ระบุตำแหน่งหน่วยความจำ : FF02H

หน่วยความจำ



รูปที่ 14 การทำงานของรีซีเอสเตอร์ I

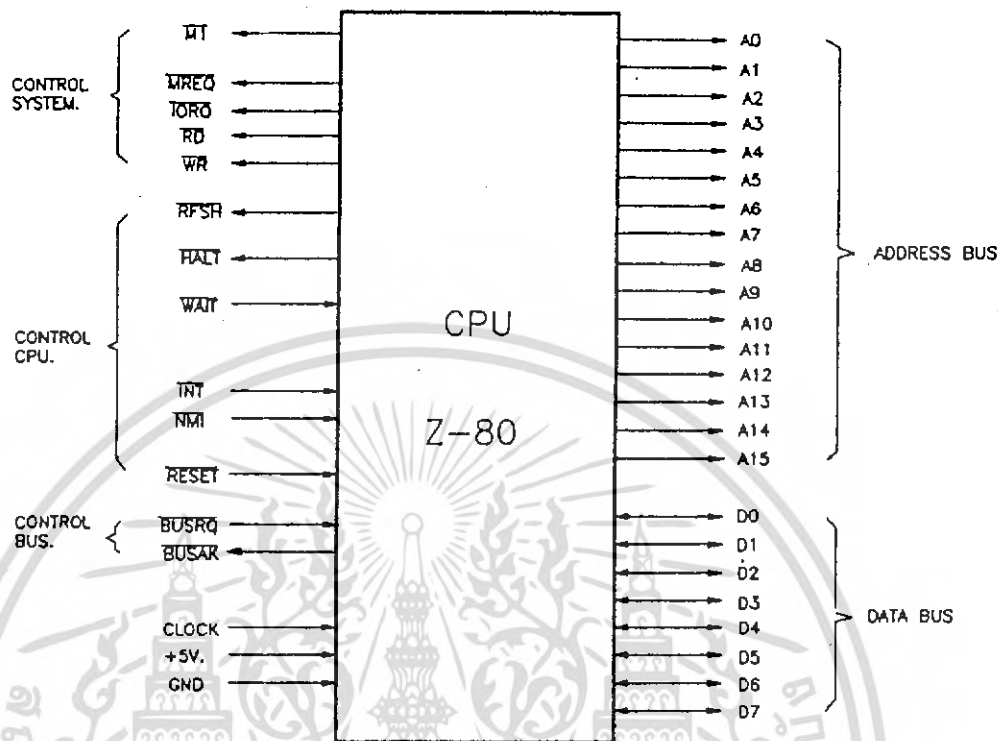
ขาและสัญญาณการเชื่อมต่อ Z-80

ไมโครโปรเซสเซอร์ Z-80 อยู่ในไอโอซีขนาดมาตรฐานอุตสาหกรรม (Industry Standard) แบบ Dual In-Line Package (DIP) หรือที่เรียกว่าแบบตีนตะขาบ 40 ขา ๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้ภายในห้องเรียนเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

86904

ต่าง ๆ แสดงไว้รูปที่ 15 กลุ่มสัญญาณต่าง ๆ ของไมโครโปรเซสเซอร์แบ่งออกได้เป็น 3 กลุ่ม คือ



รูปที่ 15 แสดงขาต่าง ๆ ของซีพียู Z-80

กลุ่มของสายสัญญาณเพื่อกำหนดตำแหน่ง (Address Bus) คือ A-15-A0 กลุ่มของสายสัญญาณข้อมูล (Data Bus) คือสัญญาณที่เหลื่อมยกเว้นขาแหล่งจ่ายไฟและสัญญาณนาฬิกา หน้าที่ของขาต่าง ๆ จะได้อธิบายในรายละเอียดต่อไปนี้

A15-A0 เป็นสายสัญญาณกำหนดตำแหน่ง (Address-Bus) โดยที่ A0 เป็นบิตทางต่ำ (LSB) ขาเหล่านี้เป็นเอาต์พุตแบบสามสถานะ (Tri-State) แอคคิฟที่ลอจิก 1 บัสนี้มีคิวบิตทั้งหมด 16 สาย ดังนั้น จึงสามารถติดต่อกับหน่วยความจำได้ถึง $2^{16} = 65536$ ตำแหน่ง (64 Kbyte) นอกจากนั้นยังสามารถใช้การกำหนดตำแหน่งของอินพุต/เอาต์พุตเมื่อใช้คำสั่งอินพุต/เอาต์พุตได้โดยใช้ 8 บิต ด้านต่ำ (A7-A0) เพื่อแสดงตำแหน่งของพอร์ต ดังนั้นสามารถกำหนดพอร์ตอินพุตได้ 256 พอร์ตเช่นกันและช่วงระยะเวลาซีเพรช

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(RFSH) บัสทางต่ำ 7 บิต (A6-A0) จะใช้แสดงตำแหน่งของหน่วย ความจำแบบไดนามิกที่ได้รับจากรีเฟรช

- D7-D0 เป็นสายสัญญาณข้อมูล (Data Bus) D0-D7 เป็นบิตทางต่ำลักษณะเป็นบัส ข้อมูลแบบสามสถานะ ขนาด 8 บิตและแอสที่ฟที่ลอจิก 1 ใช้เพื่อเป็นเส้นทาง ผ่านของข้อมูลระหว่างไมโครโปรเซสเซอร์กับหน่วยความจำหรืออุปกรณ์อิน พุท/เอาต์พุตต่าง ๆ
- $\overline{M1}$ (Machine Cycle One) เป็นขาเอาต์พุตและแอสที่ฟที่ลอจิก 0 เมื่อ ขานี้แอสที่ฟชี้ให้เห็นว่าขณะนี้กำลังอยู่ในสภาวะของการเฟรชคำสั่ง และ ถ้าเป็นคำสั่งที่มีรหัส 2 ไบต์ ส่วนของ $\overline{M1}$ จะถูกสร้างขึ้นขณะในเฟรช แต่ละไบต์ ลักษณะของคำสั่งที่มีขนาด 2 ไบต์ เช่นคำสั่งที่มีรหัสเริ่มต้นด้วย CBR, DDH, EDH, หรือ FDH นอกจากนั้นสัญญาณ $\overline{M1}$ นี้จะใช้ร่วมกับ เพื่อ สร้าง \overline{IORQ} สัญญาณตอบรับการอินเตอร์รัพท์ (Interrupt Acknowledge)
- \overline{MREQ} (Memory Request) เป็นเอาต์พุตแบบสามสถานะและแอสที่ฟที่ลอจิก 0 เมื่อ สายสัญญาณนี้แอสที่ฟบอกให้ทราบว่า ขณะนี้ไมโครโปรเซสเซอร์ต้อง การติดต่อกับหน่วยความจำเพื่ออ่านหรือเขียนข้อมูลโดยที่ตำแหน่งของหน่วย ความจำจะปรากฏอยู่บัสตำแหน่งแล้ว
- \overline{IORQ} (Input/Request) เป็นเอาต์พุตแบบสามสถานะ และแอสที่ฟที่ลอจิก 0 เมื่อสัญญาณแอสที่ฟบอกให้ทราบว่า ขณะนี้ทางด้านไบต์ต่ำกว่า (A7-A0) ของ บัสตำแหน่งบรรจุตำแหน่งของพอร์ต ที่จะถ่ายข้อมูลระหว่างไมโครโปรเซส เซอร์กับอุปกรณ์ อินพุท/เอาต์พุต นอกจากนี้จะช่วยร่วมสัญญาณ $\overline{M1}$ เพื่อ ตอบรับการอินเตอร์รัพท์ และ ขณะนี้ เวกเตอร์ของอินเตอร์รัพท์จะส่งผ่าน เข้ามาในบัสข้อมูลเพื่อกำหนดตำแหน่งของโปรแกรมบริการการอินเตอร์รัพท์
- \overline{RD} (Memory Read) เป็นขาเอาต์พุตแบบสามสถานะ และแอสที่ฟที่ลอจิก 0 สัญญาณนี้เพื่อชี้ว่าขณะนี้ ไมโครโปรเซสเซอร์ต้องการอ่านข้อมูลจากหน่วย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความจำหรือจากอุปกรณ์อินพุท/เอาต์พุท

$\overline{\text{RFSH}}$

(Memory Write) เป็นขาเอาต์พุทแบบสามสถานะ และแอกติฟที่ลอจิก 0 เมื่อสัญญาณแอกติฟที่ว่างขณะนี้ บัสตำแหน่งด้านต่ำ 7 บิต (A6-A0) บรรจุตำแหน่งหน่วยความจำแบบไดนามิกแรมที่จำรีเฟรชและสัญญาณ $\overline{\text{MERQ}}$ ในช่วงนี้จะนำไปใช้เป็นสัญญาณสำหรับเพื่อรีเฟรช (Refresh Read) ไดนามิกแรมทั้งหมดที่ใช้ในระบบ

$\overline{\text{HALT}}$

(Halt State) เป็นขาเอาต์พุทแอกติฟที่ลอจิก 0 เป็นสัญญาณเพื่อชี้ว่าขณะนี้ไมโครโปรเซสเซอร์ปฏิบัติคำสั่ง $\overline{\text{HALT}}$ จากโปรแกรมและกำลังรอสัญญาณการอินเตอร์รัพต์ชนิดนอนมาสเคเบิล (เมื่อสั่งให้ยอมรับ) จากอุปกรณ์ภายนอก ถ้าได้รับสัญญาณการอินเตอร์รัพต์แล้วจึงจะทำงานต่อไปได้ในขณะที่หยุด (Halted) นี้ซีพียูจะกระทำคำสั่ง NOP (No-Operation) เพื่อให้มีการเพชคำสั่งซึ่งจะไม่ทำให้การรีเฟรชหยุดชะงัก

$\overline{\text{WAIT}}$

(Wait) เป็นขาอินพุท แอกติฟที่ลอจิก 0 เป็นสัญญาณเพื่อชี้ว่าการส่งถ่ายข้อมูลระหว่างไมโครโปรเซสเซอร์และหน่วยความจำ หรืออุปกรณ์อินพุท/เอาต์พุทยังไม่เรียบร้อย และให้ไมโครโปรเซสเซอร์ หยุดรอ ตรวจจับที่ขาอินพุทแอกติฟอยู่ ดังนั้นสัญญาณนี้จะใช้เพื่อให้หน่วยความจำหรืออุปกรณ์อินพุท/เอาต์พุท ที่มีความเร็วใด ๆ สามารถให้เข้าจังหวะกันพอดี (Synchronized) กับไมโครโปรเซสเซอร์

$\overline{\text{INT}}$

(Interrupt Request) เป็นขาอินพุทแอกติฟที่ลอจิก 0 สัญญาณ $\overline{\text{INT}}$ นี้เป็นสัญญาณที่สร้างมาจากอุปกรณ์อินพุท/เอาต์พุทเพื่อต้องการอินเตอร์รัพต์การทำงานตามปกติของไมโครโปรเซสเซอร์จะจดจำไว้ ถ้าหากว่าโปรแกรมกำหนดให้มีการยอมรับสัญญาณ การอินเตอร์รัพต์ได้ (Enable Interrupt) โดย IFF ถูกเซตเป็น 1 และไม่มีการขอใช้บัสเสียก่อน คือขา $\overline{\text{BUSRQ}}$ ต้องไม่แอกติฟ เมื่อไมโครโปรเซสเซอร์รับสัญญาณอินเตอร์รัพต์ มันจะตอบสนองโดยการตอบสนองสัญญาณ $\overline{\text{IORQ}}$ ออกมาในช่วงเวลา $\overline{\text{MI}}$ เพื่อเป็นการตอบรับการอินเตอร์รัพต์ (Interrupt Acknowledge) ใน

ช่วงไซเกิลของคำสั่งต่อมา

NMI (Non Maskable Interrupt) เป็นขาอินพุตและแอกติฟที่ขอบพัลส์ขาลง (Negativedge Trigger) สัญญาณที่ขา NMI นี้ ลำดับความสำคัญสูงกว่า สัญญาณที่ขา INT ไมโครโปรเซสเซอร์ จะทำ การตรวจสอบที่ขา NMI นี้ที่ สเททสุดท้ายของคำสั่ง เช่นเดียวกับขา INT แต่ไม่ขึ้นอยู่กับ IFF เมื่อไมโครโปรเซสเซอร์ได้รับสัญญาณที่ขา NMI จะทำให้เริ่มต้นการทำงานใหม่ที่ ค่าต่ำกว่าตำแหน่ง 0066H ส่วนค่าในโปรแกรมเคาน์เตอร์ที่ชี้ตำแหน่ง ของคำสั่งต่อไปก่อนที่ ซีพียู จะถูกอินเตอร์รัพท์ จะเก็บไว้ในสแตค (ที่ RAM) เพื่อที่ซีพียูสามารถกลับมาทำงานต่อไปหลังจากที่ทำการบริการการอินเตอร์รัพท์เสร็จสิ้นแล้ว ในขณะที่ซีพียูอยู่ในจังหวะ Wait มันจะไม่รับสัญญาณ NMI ลำดับความสำคัญต่ำกว่าสัญญาณ BUSRQ ดังนั้นในขณะที่ซีพียูกำลังทำโปรแกรมการบริการการอินเตอร์รัพท์อยู่ มันสามารถรับสัญญาณ BUSRQ ได้

RESET เป็นอินพุต แอกติฟที่ลอจิก 0 เมื่อไมโครโปรเซสเซอร์ ได้รับสัญญาณ RESET จะทำให้ค่าในโปรแกรมเคาน์เตอร์เริ่มต้นที่ศูนย์ และตั้งต้นการทำงานของไมโครโปรเซสเซอร์ใหม่ และ ในส่วนต่าง ๆ จะเป็นดังนี้

1. จัดอินเตอร์รัพท์ฟิลลอป (IFF) ให้อยู่ในสภาพที่ไม่ยอมรับการอินเตอร์รัพท์แบบมาสเคเบิล (IFF1=IFF2=0)
2. เซตรีจิสเตอร์ I=OOH
3. เซตรีจิสเตอร์ R=OOH
4. เซตเป็นการอินเตอร์รัพท์ โหมด 0

ในช่วงเวลาของการรีเซต บัสข้อมูล บัสตำแหน่ง จะอยู่ในสภาวะอิมพีแดนซ์สูง ส่วนบัสควบคุมจะอยู่ในสภาพที่ไม่แอกติฟ (Inactive)

BUSRQ (Bus Requsst) เป็นขาอินพุตแอกติฟที่ระดับ 0 สัญญาณ BUSRQ นี้มีผลทำให้บัสตำแหน่งข้อมูล และสัญญาณควบคุมที่เป็นขาเอาต์พุตแบบสามสถานะอยู่ในสภาวะอิมพีแดนซ์สูงจากนั้นต่าง ๆ จะถูกควบคุมโดยอุปกรณ์ภายนอกไมโครโปรเซสเซอร์ จะตรวจสอบการขอสัญญาณใช้บัสนี้ทุก ๆ สเททสุดท้ายของแมชชีนไซเกิลของคำสั่ง เมื่อพบการขอใช้บัสซีพียูจะตอบสนองในไซเกิลถัดไป

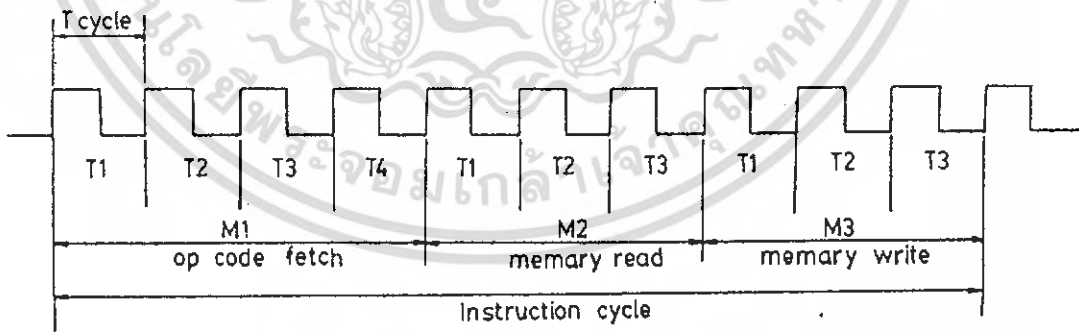
BUSAK (Bus Acknowledge) เป็นเอาต์พุตแอกติฟที่ระดับ 0 สัญญาณนี้ ใช้สำหรับตอบรับการขอใช้บัสและแสดงว่าขณะนี้บัสตำแหน่งข้อมูล และ สัญญาณควบคุมที่เป็น เอาต์พุตแบบสามสถานะอยู่ในสภาวะอิมพีแดนซ์สูงแล้ว อุปกรณ์ควบคุมภายนอก สามารถเข้าควบคุมบัสได้เป็นขาที่รับสัญญาณ นาฬิกาซึ่งเป็นเพียงเฟสเดียวกัน ใช้ระดับสัญญาณแบบ TTL และต้องการตัวต้านทานเพื่อ Pull up ค่า 330 โอห์ม หนึ่งตัวเพื่อต่อกับแหล่งจ่ายไฟ 5 โวลต์ Z-80 ทำงานได้ที่ สัญญาณนาฬิกาไม่เกิน 2.5 MHz Z-80A ทำงานได้ไม่เกิน 4 MHz และ Z-80B ทำงานไม่เกิน 6 MHz

ไคอะแกรมเวลาของไมโครโพรเซสเซอร์

ไมโครโพรเซสเซอร์ Z-80 จำปฏิบัติงานสิ่งต่าง ๆ อย่างมีขั้นตอนที่แน่นอนขั้นตอนเหล่านี้ประกอบด้วยการทำงานพื้นฐานต่าง ๆ คือ

- การอ่าน-เขียนหน่วยความจำ (Memory read or write)
- การอ่าน-เขียนอุปกรณ์อินพุต/เอาต์พุต (I/O device read or write)
- การบอกรับอินเตอร์รัพท์ (Interrupt Acknowledge)

การทำงานของคำสั่งต่าง ๆ เกิดจากลำดับการทำงานพื้นฐานเหล่านี้ซึ่งวนการทำงานพื้นฐานแต่ละส่วนอาจใช้เวลา 3-6 คาบ กว่าที่จะเสร็จสมบูรณ์ หรืออาจขยายออกไปได้มากกว่านี้ เพื่อให้การทำงานไมโครโพรเซสเซอร์พอดีกับความเร็วของอุปกรณ์ภายนอก



รูปที่ 16 ตัวอย่างไคอะแกรมเวลาของไมโครโพรเซสเซอร์

คาบเวลาของสัญญาณหนึ่ง ๆ นี้คือ T ไชเกิล และขบวนการพื้นฐานหนึ่ง ๆ จะเรียกว่า

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แมชชีนไซเคิล และขบวนการพื้นฐานต่าง ๆ จะเรียกว่า แมชชีนไซเคิล (Machine Cycle) ตัวอย่างไคอะแกรมเวลาของคำสั่งหนึ่งซึ่งแสดงดังที่รูป 16 จากรูป 16 เป็นคำสั่งประกอบด้วย 3 แมชชีนไซเคิล (M1, M2 และ M3) แมชชีนไซเคิลที่ 1 (M1) ของคำสั่งใดๆเป็นไซเคิลของการเพชท์ ซึ่งอาจมี 4, 5 หรือ 6T ไซเคิล ยกเว้นในการที่เพิ่มเวลาออกไปเนื่องจากการรอคอย (Wait state) ไซเคิลของการเพชท์ (M2) เพื่อใช้ การอ่านรหัสคำสั่งต่อไปที่จะกระทำลำดับต่อมาทำการเคลื่อนย้ายข้อมูล ระหว่างไมโครโปรเซสเซอร์ กับหน่วยความจำหรืออุปกรณ์ อินพุท/เอาต์พุท ซึ่งใช้เวลาตั้งแต่ 3 ถึง 5 คาบเวลา เว้นแต่การรอคอยเช่นกัน แมชชีนไซเคิลที่ตามหลัง M1 นี้จะมีหรือไม่มีก็แล้วแตชนิดของคำสั่ง เช่น LD r, r จะใช้เพียงหนึ่งแมชชีนไซเคิลก็สามารถทำงานได้ ต่อไปจะได้กล่าวถึงไคอะแกรมเวลาซึ่งปรากฏอยู่ในการทำงานพื้นฐานต่าง ๆ ทั้งในช่วงการรอคอยไม่มีการรอคอย ซึ่งแมชชีนไซเคิลที่มีใช้ สามารถสรุปได้ดังนี้

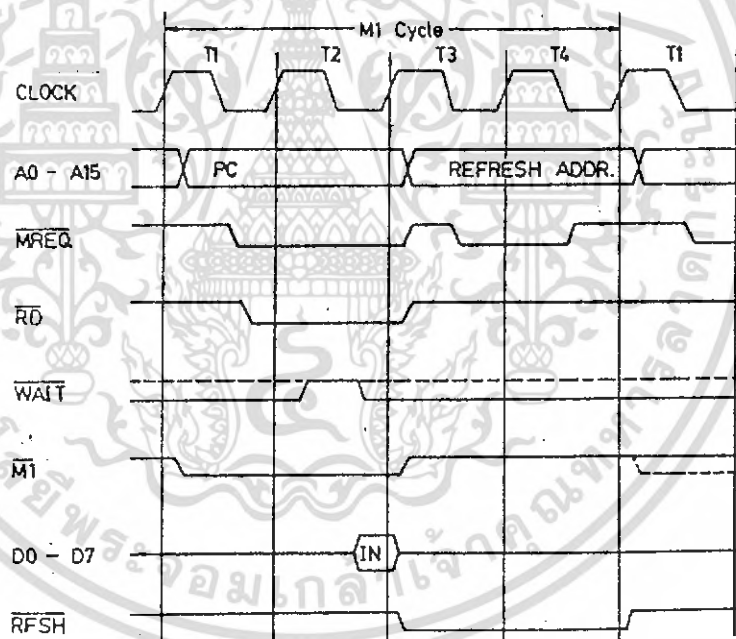
1. ไซเคิลการเพชท์รหัส (Instruction Opcode fetch cycle(M1))
2. ไซเคิลการอ่าน หรือ เขียนหน่วยความจำ (Memory data read or write cycle)
3. ไซเคิลการอ่านหรือเขียนข้อมูลกับอุปกรณ์ อินพุท/เอาต์พุท (I/O read or write cycle)
4. ไซเคิลการขอใช้บัสหรือการตอบสนองการขอใช้บัส (Bus request/acknowledged cycle)
5. ไซเคิลการขออินเตอร์รัพท์ และการตอบรับ การอินเตอร์รัพท์ (Interrupt request/acknowledge cycle)
6. ไซเคิลการขอและตอบรับการอินเตอร์รัพท์แบบนอนมาสเคเบิล (Non-maskable interrupt request/acknowledge cycle)
7. ไซเคิลการออกคำสั่ง HALT (Exit from a HALT instruction) ต่อไปจะได้อธิบายการทำงานในไซเคิลต่าง ๆ

ไซเคิลเพชท์คำสั่ง

ไคอะแกรมของช่วงไซเคิลการเพชท์คำสั่ง (M1 cycle) แสดงดังรูปที่ 17 M1 นี้เป็นสภาวะเริ่มแรกของทุกคำสั่ง สังเกตได้ว่าเมื่อเริ่มไซเคิล M1 ในโปรแกรมเคาเตอร์ จะแทนที่ลงในบัสตำแหน่ง และหลังจากไซเคิลของคาบสัญญาณนาฬิกาต่อมา ซีพียูจะส่งสัญญาณ \overline{MREQ} ออกมาที่เวลานี้ตำแหน่งของหน่วยความจำบนบัสตำแหน่งจะคงที่ ดังนั้นช่วงขอบพัลส์ขาลงของ \overline{MREQ} จะใช้เป็นสัญญาณ chip enable ของหน่วยความจำในขณะนี้สัญญาณ \overline{RD} แอคติฟ เพื่อแสดงว่า ซีพียู

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

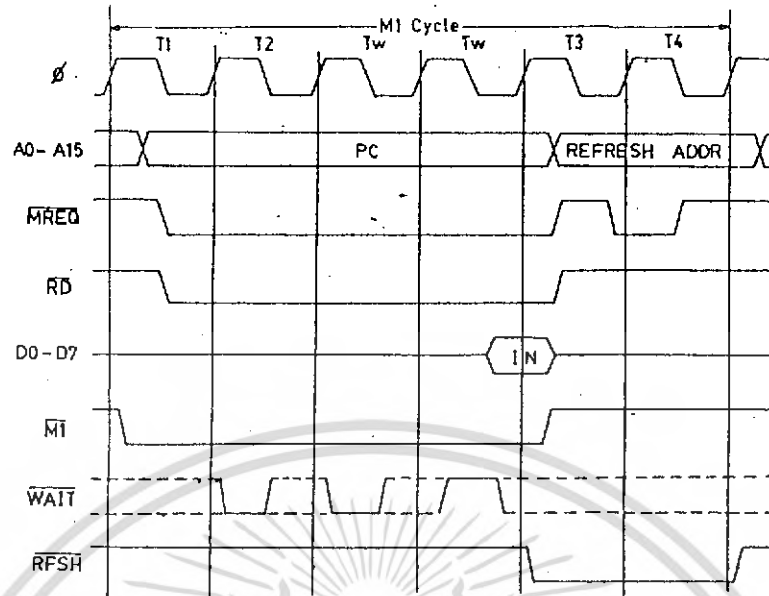
ต้องการอ่านข้อมูลจากหน่วยความจำ ตามตำแหน่งที่ปรากฏบนบัส ตำแหน่ง ดังนั้นหน่วยความจำภายนอกจะส่งข้อมูลจากตำแหน่งที่ระบุมาบนบัสข้อมูล ซีพียูจะสุ่มอ่านข้อมูลจากหน่วยความจำบนบัสข้อมูลเมื่อถึงจังหวะขอบพัลส์ขาขึ้นของ T3 ขอบพัลส์นี้จะใช้ในการยกเลิกสัญญาณ \overline{MREQ} และ \overline{RD} ด้วย ดังนั้นจะเห็นว่าข้อมูลที่ถูกรับเข้าใน IR ก่อนที่สัญญาณ \overline{MREQ} และ \overline{RD} จะหายไปตามเวลา T2 และ T4 ที่เหลือในไซเคิล M2 จะใช้ในการรีเฟรชหน่วยความจำแบบไดนามิก (ในขณะที่เดียวกัน ซีพียูก็ใช้เวลาช่วงนี้ถอดรหัส และปฏิบัติคำสั่ง ในกรณีที่กำลังนั้น ๆ ไม่ต้องการแมชชีนไซเคิลอื่น ๆ อีก เช่นคำสั่ง INC r เป็นต้น) ขณะเวลา T3 และ T4 นี้บัสตำแหน่งทางด้านต่ำ 7 บิต (A6-A0) จะบรรจุด้วยตำแหน่งของหน่วยความจำที่จะทำการรีเฟรช และสัญญาณรีเฟรชจะแอกตีฟ เพื่อ แสดงว่าการอ่านเพื่อรีเฟรช (Refresh read) สำหรับไดนามิกแรมทุกตัวควรจะทำในขั้นตอนนี้ให้เสร็จและสังเกตว่าขณะที่ทำการรีเฟรชจะไม่มีการกำเนิดสัญญาณ \overline{RD} เพื่อป้องกันไม่ให้ หน่วยความจำตำแหน่งอ่านเข้ามาในบัสข้อมูลสัญญาณ \overline{MREQ} ในช่วงเวลาการรีเฟรชใช้เพื่อการอ่านเพื่อการรีเฟรชทุก ๆ ตำแหน่งของหน่วยความจำ



รูปที่ 17 ไซเคิล M1 (Opcode cycle)

ไซเคิลการ เพช้คำสั่งที่มีการรอคอย

สถานะการรอคอย (Wait state) การเกิดเนื่องจากความเร็วในการทำงานของหน่วยความจำ ช้ากว่าการทำงานของซีพียู ดังนั้นการเพช้คำสั่งก็จะช้าลง พิจารณาจากรูปที่ 18 ในช่วงขอบพัลส์ขาของ T2 และทุก ๆ TW ที่ตามมา ซีพียูจะทำการตรวจสอบซึ่งขา \overline{WAIT} แอกตีฟ



รูปที่ 18 ไชเกิล M1 ที่มีช่วงการรอคอย

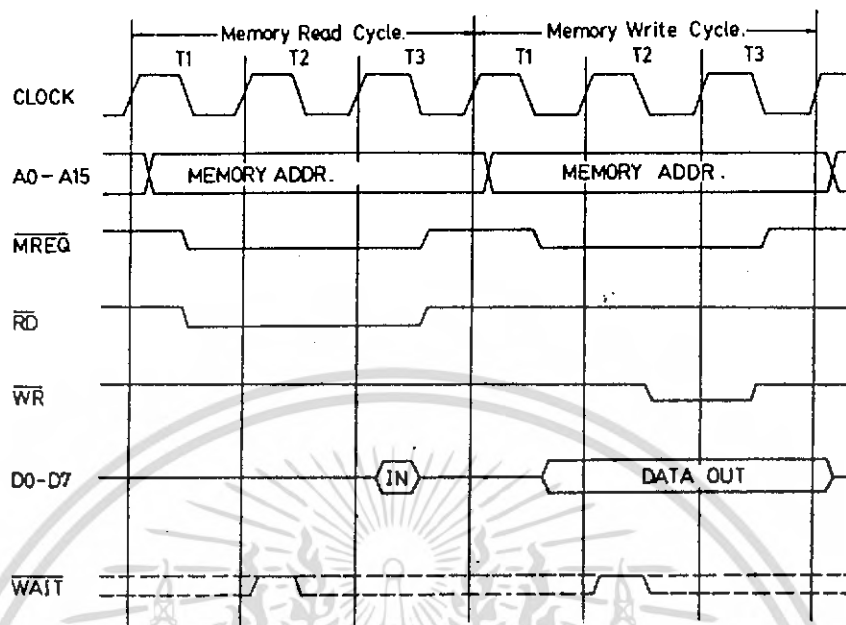
ในช่วงนี้ซีพียูจะสร้าง T_w เข้ามา 1 คาบ ของสัญญาณนาฬิกาถัดไป วิธีการแบบนี้ ทำให้สามารถ
 หน่วงเวลาการอ่านข้อมูลให้ช้าลงเพื่อให้สอดคล้องกับ Access time ของหน่วยความจำแบบต่าง ๆ

ไชเกิล การอ่านหรือเขียน หน่วยความจำ

ไดอะแกรมของการอ่านหรือเขียนหน่วยความจำ แสดงดังรูปที่ 19 ไชเกิลการทำงาน
 ทั้งสองนี้ ใช้สัญญาณนาฬิกาอย่างละ 3 คาบ ไม่รวมคาบเวลาการรอคอยที่เกิดขึ้นที่ขา \overline{WAIT}
 พิจารณาไชเกิลการอ่าน เห็นได้ว่าขั้นแรก ซีพียูจะส่งตำแหน่งที่ต้องการอ่านลงในบัสตำแหน่ง และ
 หลังจากขอบพัลส์ขาของ T1 ก็ส่งสัญญาณ \overline{MREQ} และ \overline{RD} ออกมา (เหมือนช่วงการเพทซ์) จาก
 นั้นหน่วยความจำจะส่งข้อมูลในตำแหน่งนั้นลงมาในบัสข้อมูล เมื่อถึงขอบพัลส์ขาของ T2 ซีพียูจะ
 ตรวจสอบที่ขา \overline{WAIT} ว่ามีสัญญาณการรอคอยหรือไม่ ถ้ามี ซีพียูจะสร้าง T_w ขึ้น ในคาบเวลาต่อไป
 ถ้าไม่มีซีพียูจะทำงานต่อไปคือเมื่อถึงช่วงขอบพัลส์ขาของ T3 มันจะทำการอ่านข้อมูลจากบัสข้อมูล
 ในระยะเวลาสั้น ๆ เข้าไปเพื่อไม่ให้เกิดความผิดพลาด ส่วนไชเกิลของการเขียนข้อมูลลงหน่วย
 ความจำพิจารณาได้จากรูปที่ 19 ซึ่งลักษณะการทำงานจะคล้ายกับช่วงการอ่าน คือหลังจากส่งตำ
 แหน่งที่ต้องการเขียนลงไปในบัสตำแหน่งแล้วก็จะส่งสัญญาณ \overline{MREQ} ออกมาจากนั้นข้อมูลจากรีจิสเตอร์
 ที่ต้องการเขียนลงไปในหน่วยความจำจะถูกถ่ายทอดลงในบัสข้อมูลที่ขอบพัลส์ขาของ T1 และ ขอบ
 พัลส์ขาของ T2 สัญญาณ \overline{WR} จะแอกตีฟ ซึ่งในหน่วยความจำภายนอกเขียนข้อมูลเข้าไปในหน่วย

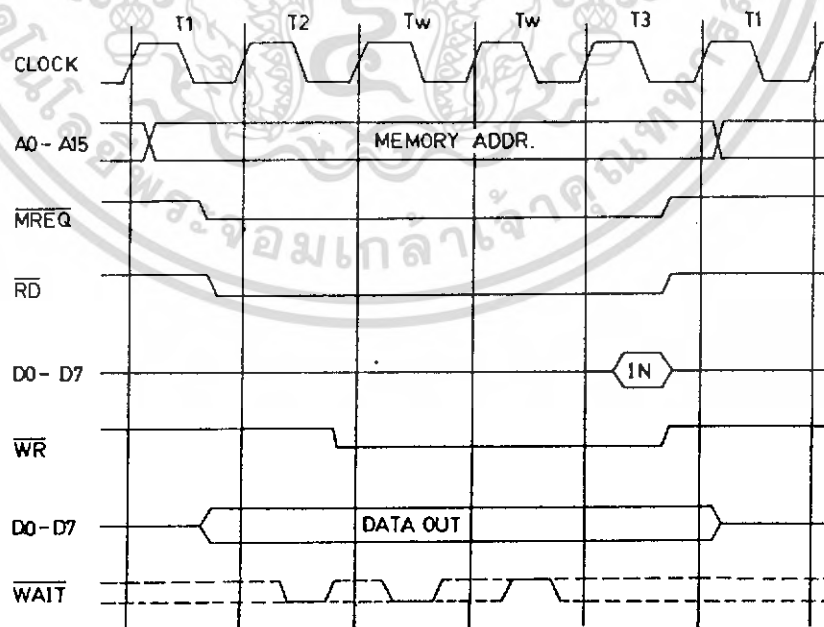
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
 ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความจำตามตำแหน่งที่กำหนดที่บัสตำแหน่งของพัลส์ขาลงของ T3 สัญญาณ \overline{MREQ} และ \overline{WR} จะหาย



รูปที่ 19 ไช้เก็ลการอ่านและเขียนหน่วยความจำ

ไป หลังจากนั้นข้อมูลบนบัสข้อมูลก็สามารถเปลี่ยนแปลงต่อไปได้ แต่ถ้ามีสัญญาณการรอคอยเกิดขึ้นที่ขอบพัลส์ขาลงของ T2 ก็จะมี T_w ขึ้นต่อไปซึ่งเหมือนกับไช้เก็ล การเพ็ช้ค่าสิ้ง และโคะแกรมเวลาเมื่อมีสัญญาณ \overline{WAIT} จะแสดงดังรูปที่ 20

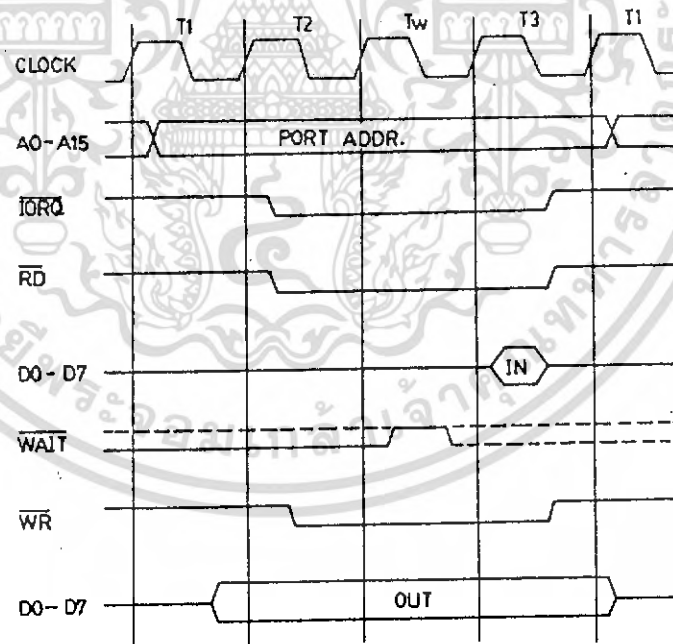


รูปที่ 20 ไช้เก็ลการอ่านหรือเขียนหน่วยความจำเมื่อมีสัญญาณการรอคอย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไซเกิลการอ่านหรือเขียนข้อมูลกับอุปกรณ์อินพุท/เอาต์พุท

ไซเกิลการอ่านหรือเขียนข้อมูลกับอุปกรณ์ อินพุท/เอาต์พุทนี้ เกิดขึ้นในขณะที่ ซีพียู ปฏิบัติคำสั่งเกี่ยวกับการอินพุทหรือเอาต์พุท เช่น IN A, (n) หรือ OUT (n), A เป็นต้น ปกติคำสั่งกลุ่มนี้ใช้ 3 หรือ 4 แมชชีนไซเกิล แต่คำสั่งที่มีความสามารถสูงในบางคำสั่ง เช่น INTR INDR PTIR หรือ OTDR ที่สามารถส่งถ่ายข้อมูลได้ถึง 256 ไบต์ จะใช้แมชชีนไซเกิลบางอันซ้ำ ๆ กันจนกระทั่งส่งถ่ายข้อมูลเสร็จผลก็คือ เวลาในการทำงานของคำสั่งจะขึ้นอยู่กับจำนวนไบต์ที่จะส่งถ่ายและความเร็วของอุปกรณ์ อินพุท/เอาต์พุท ด้วยโคแอดแกรมเวลาของการอ่านข้อมูลจากอุปกรณ์ อินพุทและการเขียนข้อมูลลงในอุปกรณ์ เอาต์พุท แสดงได้ดังรูปที่ 21 จากรูปเห็นได้ว่า เมื่อเริ่มต้นแมชชีนไซเกิลตำแหน่งของ อุปกรณ์อินพุท/เอาต์พุท จะถูกแทนลงในบัสตำแหน่งทางด้านไบต์ต่ำ (A7-A0) หลังจากที่ยอมรับคำสั่งขาขึ้น ของ T2 สัญญาณ \overline{IORQ} ตัวควบคุมอุปกรณ์ภายนอกจะรับรู้สัญญาณการขออ่านนี้และนำข้อมูลส่งไปบนบัสข้อมูลจากนั้นซีพียูจะสร้าง T_w ขึ้นมา 1 คาบเวลาทั้งๆ ที่ไม่มีสัญญาณ \overline{WAIT} และการตรวจสอบสัญญาณ \overline{WAIT} จะทำให้ขอบพัลส์ขาสูงของ T แทนการอ่านข้อมูลจากบัสข้อมูลเข้า ซีพียูจะทำที่ขอบขาพัลส์ลงของ T3



รูปที่ 21 ไซเกิลของการเขียนอุปกรณ์อินพุท/เอาต์พุท

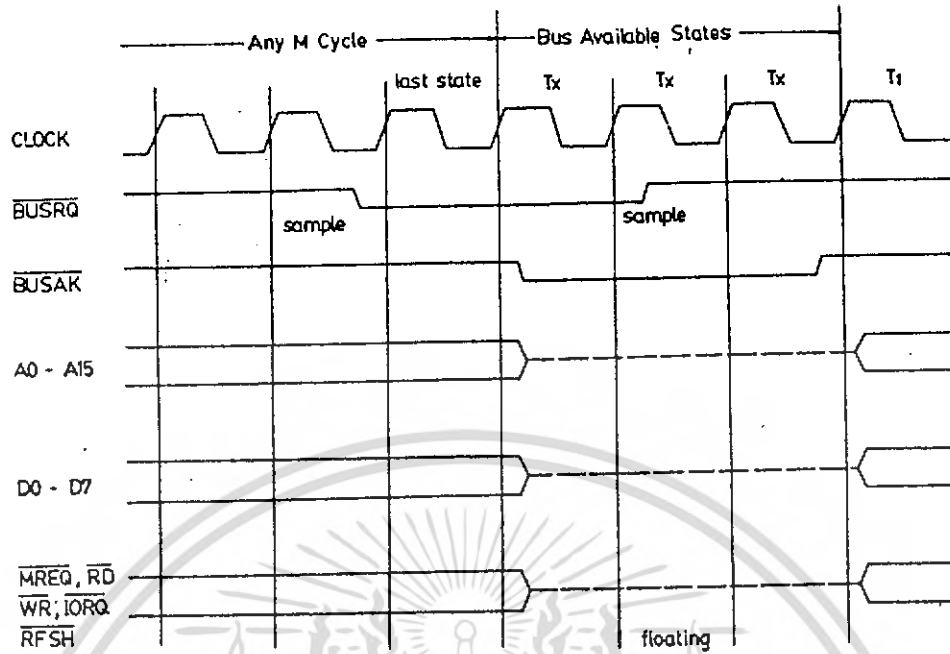
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไซเกิลของการเขียนข้อมูล สัญญาณ \overline{WR} จะแอกตีฟในเวลาเดียวกับสัญญาณ \overline{IORQ} ดังรูปที่ 21 ข้อมูลจากรีจิสเตอร์ที่จะส่งไปยังอุปกรณ์ อินพุท/เอาต์พุท จะถูกถ่ายลงบัสข้อมูลที่ขอบพัลส์ขาลงของ T1 คือก่อนที่มีสัญญาณ \overline{IORQ} และ \overline{WR} หลังจาก T2 ซีพียู จะสร้าง T_w ขึ้นมาอัตโนมัติ 1 คาบ เช่นเดียวกับช่วงอ่าน และจากนั้นในช่วงเวลาที่เหลือ คือ T_w ตัวควบคุมอุปกรณ์ อินพุท/เอาต์พุท จะนำข้อมูลจากบัสข้อมูลเข้าไป

จากการทำงานของการอ่าน หรือ เขียนอุปกรณ์ อินพุท/เอาต์พุท มีข้อสังเกตอย่างหนึ่งคือ หลัง T2 ซีพียูจะใส่คาบการรอคอย (T_w) เข้าอัตโนมัติ 1 คาบ ที่เป็นเช่นนี้เพราะว่าในช่วงการทำงาน อินพุท/เอาต์พุท นี้ ระยะเวลาที่สัญญาณ \overline{IORQ} เริ่มเกิดขึ้นจนถึงเวลาที่ซีพียูจะต้องตรวจสอบสัญญาณ การรอกอายนั้นมันสั้นมากหากไม่ใช้ T_w เข้าไปเป็นกรณีพิเศษแล้ว ช่วงเวลานี้อาจจะสั้นเกินกว่าที่ อินพุท/เอาต์พุท พอร์ตใด ๆ จะถอดรหัสของตัวเอง แล้งส่งสัญญาณ \overline{WAIT} ได้ทัน นอกจากนี้หากไม่มีการใส่ T_w นี้เข้าไป ยังเป็นการยากมากที่จะออกแบบอุปกรณ์ อินพุท/เอาต์พุท ที่เป็นพวก MOS ให้ทำงานพอดีกับการทำงานของซีพียู

ไซเกิลการขอใช้บัสและตอบสนองการขอใช้บัส

ที่เวลาใดๆ ของการทำงานของซีพียู อุปกรณ์ภายนอกสามารถส่งสัญญาณมาควบคุมบัส ตำแหน่ง A-15 บัสข้อมูล D7-DO และสัญญาณควบคุม \overline{MREQ} \overline{RD} \overline{WR} \overline{IORQ} และ \overline{RFSH} ได้โดยการส่งสัญญาณ \overline{BUSRQ} เข้ามาอยู่ที่ซีพียู เหตุผลที่ทำเช่นนี้ เนื่องจากเพื่อให้อุปกรณ์ภายนอกสามารถควบคุมการติดต่อโดยตรงระหว่าง หน่วยความจำภายนอกและอุปกรณ์ อินพุท/เอาต์พุท ซึ่งทำงานด้วยความเร็วสูง โดยไม่ต้องมีซีพียู วิธีการแบบนี้เรียกว่า Direct Memory Access (DMA) สัญญาณ \overline{BUSRQ} นี้สามารถถูกตรวจสอบโดยซีพียูที่ขอบพัลส์ขาขึ้นที่คาบเวลาสุดท้ายของแมชชีนไซเกิล ใด ๆ เมื่อพบว่าที่ขา \overline{BUSRQ} นี้มีระดับ "0" แสดงว่ามีการขอใช้บัสภายนอก ดังรูปที่ 22 เมื่อซีพียูได้รับสัญญาณ \overline{BUSRQ} มันจะส่งสัญญาณตอบรับ \overline{BUSAK} เมื่อถึงขอบพัลส์ขาขึ้นของคาบเวลาต่อมา และในขณะที่เดียวกัน บัสตำแหน่งบัสข้อมูลรวมทั้งสัญญาณควบคุมที่ \overline{MREQ} \overline{RD} \overline{WR} \overline{IORQ} และ \overline{RFSH} มีสถานะเป็นอิมพีแดนซ์สูง ดังนั้นการเปลี่ยนแปลงใด ๆ บนบัสขณะนี้จะไม่ส่งผลต่อซีพียู และบัสต่าง ๆ ก็จะถูกควบคุมโดยตัวควบคุม DMA เพื่อส่งถ่ายข้อมูล ระหว่างหน่วยความจำ และอุปกรณ์ อินพุท/เอาต์พุท บัสต่าง ๆ จะอยู่ในสถานะอิมพีแดนซ์สูงตราบเท่าที่ขา \overline{BUSRQ} นี้แอกตีฟอยู่การ ยกเลิกการขอใช้บัสเมื่อเกินสัญญาณ \overline{BUSRQ} นี้ไม่แอกตีฟซึ่ง ซีพียู ตรวจ

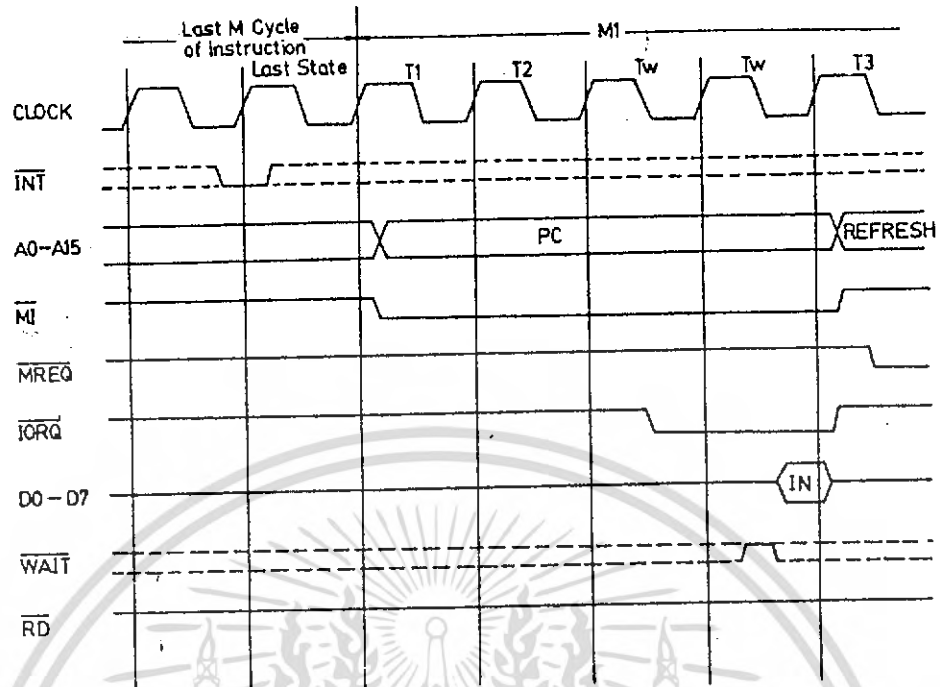


รูปที่ 22 ไซเกิลการใช้ขอบัสน

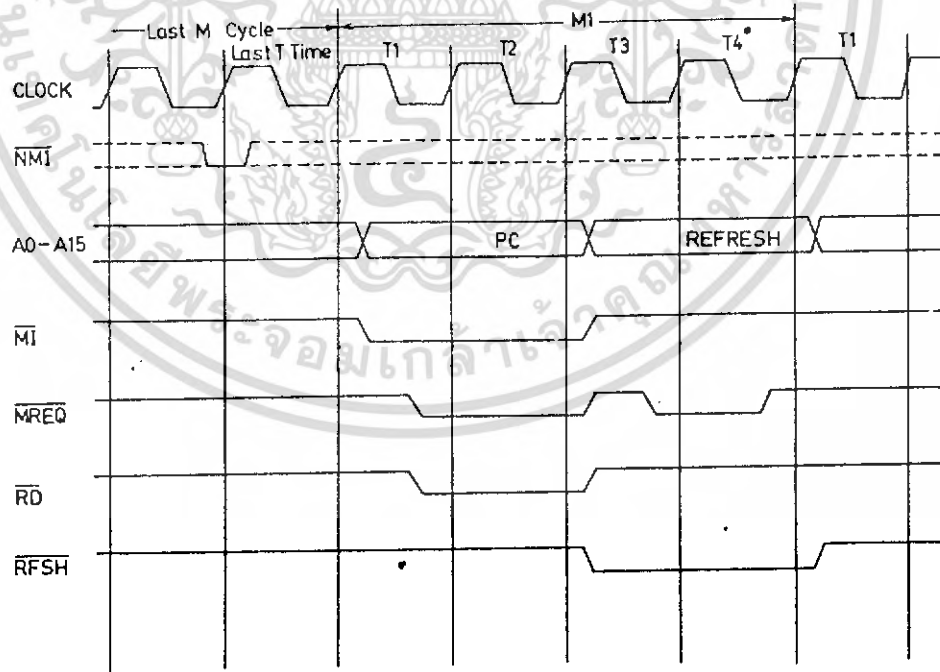
ตรวจสอบที่ ขอบพัลส์ขา ขึ้นของคาบเวลาต่อไปหลังจากที่มีการขอใช้บัสนแล้ว จากคาบเวลาต่อไปสัญญาณ $\overline{\text{BUSAK}}$ จะหายไปและ ขอบพัลส์ขาขึ้น ของคาบเวลาต่อไป บัสต่าง ๆ ก็กลับสู่สภาพเดิม ข้อสังเกตในการทำ DMA อย่างนี้คือ ถ้าระบบใช้หน่วยความจำแบบไดนามิกแรม และขณะที่อยู่ในและช่วงเวลาการ DMA จะต้องสร้างสัญญาณ รีเฟรช หน่วยความจำแบบไดนามิกแรมนี้ และในขณะที่อยู่ในไซเกิลของ $\overline{\text{BUSRQ}}$ ซีพียูจะไม่รับการอินเตอร์รัพต์ทั้งแบบ $\overline{\text{NMI}}$ และ $\overline{\text{INT}}$

ไซเกิลการขออินเตอร์รัพต์และการตอบรับการอินเตอร์รัพต์

ไซเกิลนี้เป็นอินเตอร์รัพต์แบบ มาสเคเบิล ที่อาจเกิดขึ้นที่ขา $\overline{\text{INT}}$ ถ้าอินเตอร์รัพต์ ฟลิปฟลอป (IFF) ถูกเซตให้ยอมรับการอินเตอร์รัพต์ และในขณะที่นั้นไม่มีการขอใช้บัสนโดยสัญญาณ $\overline{\text{BUSRQ}}$ แล้ว แสดงว่าซีพียูสามารถที่จะรับสัญญาณ $\overline{\text{INT}}$ จากอุปกรณ์ภายนอกได้สัญญาณการขอ อินเตอร์รัพต์ จากอุปกรณ์ภายนอกที่ขอบพัลส์ขานี้ ซีพียูจะทำการตรวจสอบที่ขอบพัลส์ขาขึ้นของคาบเวลาสุดท้าย ของแมชชีนไซเกิลการขอ และการตอบรับการอินเตอร์รัพต์แบบ มาสเคเบิล การกระทำของ ซีพียูในไซเกิลของการขอ และตอบรับการ อินเตอร์รัพต์แบบนอนมาสเคเบิล นี้ แสดงดังที่รูปที่ 23



รูปที่ 23 ไช้เก็ลการรับและการตอบการอินเตอรืรืพด์



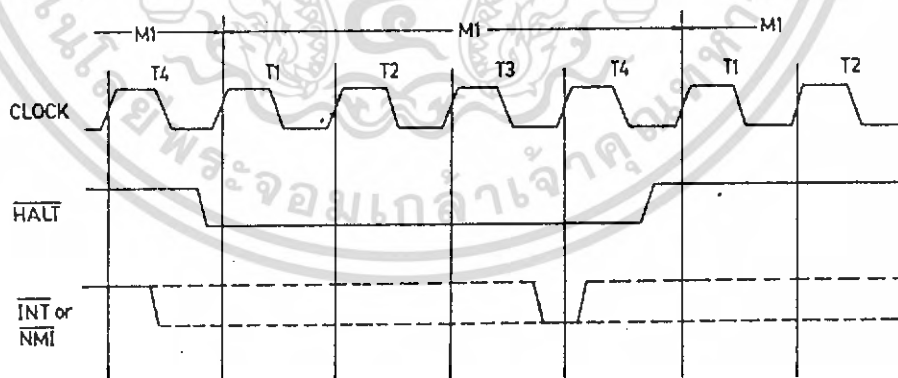
รูปที่ 24 ไช้เก็ลการรับและตอบรับอินเตอรืรืพด์แบบนอนมาสเคเบ็ล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สัญญาณการขออินเทอร์รัพต์ แบบนี้จะถูกส่งมาจากอุปกรณ์ภายนอกเข้ามาทางขา \overline{NMI} ของซีพียูและซีพียูไม่สามารถที่จะกันไม่ให้ได้รับ การอินเทอร์รัพต์แบบนี้ได้ เนื่องจากสัญญาณที่ขา \overline{NMI} นี้จัดว่ามีลำดับความสำคัญสูงกว่าสัญญาณที่ขา \overline{INT} สัญญาณ \overline{NMI} นี้จะแอกตีฟที่ขอบพัลส์ขาลงแต่ซีพียูจะทำการตรวจสอบ สัญญาณที่ขา \overline{NMI} นี้ในเวลาเดียวกับการตรวจสอบที่ขา \overline{INT} ถ้าขา \overline{NMI} มีระดับ 0 แสดงว่ามีการขออินเทอร์รัพต์แบบ นอนมาสเคเบิล และซีพียูจะทำการตอบสนองในลักษณะเดียวกับการอ่านข้อมูล ข้อแตกต่างมีเพียงแต่ ซีพียู จะไม่สนในข้อมูลจากบัสข้อมูล แต่จะส่งค่าโปรแกรมเคาเตอร์ไปเก็บไว้ในสแตคภายนอก และกระโดดไปทำงานที่ตำแหน่ง 0066H โดยอัตโนมัติตั้งนั้นโปรแกรมบริการอินเทอร์รัพต์แบบนี้จะต้องเริ่มต้นที่ตำแหน่ง 0066H นี้เท่านั้น

ไซเกิลการออกคำสั่ง HALT

การกระทำโปรแกรมเมื่อใดก็ตามที่ซีพียูปฏิบัติคำสั่ง HALT สัญญาณ \overline{HALT} จะแอกตีฟ และจากนั้นซีพียูจะสร้างไซเกิล M1 และปฏิบัติคำสั่งในลักษณะ NOP แต่ไม่มีการเพิ่มค่าโปรแกรมเคาเตอร์การปฏิบัติคำสั่ง NOP นี้จะเกิดเรื่อยไปจนกระทั่ง ซีพียูได้รับสัญญาณอินเทอร์รัพต์ ที่ขา \overline{RESET} \overline{NMI} หรือ \overline{INT} (เมื่อ $IFF=1$) สัญญาณอินเทอร์รัพต์ \overline{NMI} และ \overline{INT} จะถูกตรวจสอบที่ขอบพัลส์ขึ้นของ T4 ดังที่รูป 25 ถ้าซีพียูได้รับสัญญาณการอินเทอร์รัพต์ มันจะออกจากสภาวะของการ HALT ที่ขอบพัลส์ขาขึ้นของคาบเวลาต่อไป จากนั้นซีพียูจะตอบรับและกระทำการบริการการขออินเทอร์รัพต์ตามชนิดของการอินเทอร์รัพต์ ที่เกิดขึ้น



รูปที่ 25 ไซเกิลการออกคำสั่ง \overline{HALT}

จุดประสงค์ของการทำคำสั่ง NOP ในช่วง \overline{HALT} ก็เพื่อให้สัญญาณรีเฟรชยังคงทำ

งานต่อไป ในแต่ละไซเกิล HALT ซึ่งก็คือ ไซเกิล M1 ธรรมดา ยกเว้นแต่ว่ามันจะไม่สนใจข้อมูลจากหน่วยความจำและรหัสคำสั่ง NOP จะเกิดขึ้นภายในเอง



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2

การเชื่อมต่อไมโครโพรเซสเซอร์กับหน่วยความจำ

ไมโครโพรเซสเซอร์ไม่สามารถทำงานอิสระได้ แต่จะต้องผสมกับหน่วยต่าง ๆ อีกคือ หน่วยความจำและอินพุท/เอาต์พุตจึงจะสามารถทำงานเพื่อให้เป็นระบบไมโครคอมพิวเตอร์ได้ ซึ่งหน้าที่ที่สำคัญของหน่วยความจำในระบบไมโครคอมพิวเตอร์ก็คือ

1. เป็นหน่วยที่ใช้เก็บลำดับขั้นของคำสั่ง หรือที่เรียกว่าโปรแกรม เพื่อกำหนดให้ไมโครโพรเซสเซอร์ทำงานตามที่ต้องการ
2. เป็นหน่วยที่ใช้เก็บข้อมูลชั่วคราว ซึ่งหมายถึงสามารถเขียนข้อมูลเข้าไปในหน่วยความจำหรือนำข้อมูลเก่ามาจากหน่วยความจำ

หน่วยความจำ

หน่วยความจำในระบบคอมพิวเตอร์ทั้งหลาย คือ อุปกรณ์อิเล็กทรอนิกส์ที่ใช้ในการจดจำข้อมูลที่หน่วยความจำนี้ได้ จะอยู่ในรูปสัญญาณแม่เหล็ก หรือสัญญาณไฟฟ้าที่สามารถแสดงได้ว่าข้อมูลที่มีอยู่นั้นมีสถานะ 0 หรือ 1 ในยุคก่อนหน่วยความจำในระบบคอมพิวเตอร์ส่วนใหญ่จะเป็นสารแม่เหล็ก (Magnetic core Memory) คือใช้แกนเฟอร์ไรท์และใช้ทิศทางของสนามแม่เหล็กเป็นตัวบอกให้ทราบว่าขณะนั้นเป็นสถานะ 0 หรือ 1 แต่ในปัจจุบันหน่วยความจำได้วิวัฒนาการมาใช้อุปกรณ์ สารกึ่งตัวนำ (Semiconductor Memory) ซึ่งมีคุณสมบัติที่เหนือกว่าในหลาย ๆ ด้าน ดังนั้นในที่นี้จะกล่าวถึงแต่หน่วยความจำ แบบสารกึ่งตัวนำเท่านั้น ลักษณะของหน่วยความจำแบบนี้คือ

1. Nondestructive readout คือเมื่อทำการอ่านข้อมูลจากหน่วยความจำแล้วข้อมูลที่ตำแหน่งนั้นจะไม่เปลี่ยนแปลง (หน่วยความจำแบบ Magnetic core ข้อมูลจะมีการเปลี่ยนแปลงหลังการอ่าน) ดังนั้นจึงไม่จำเป็นต้องทำการเขียนข้อมูลกลับเข้าไปอีก

2. Volatile คือความต้องการไฟเลี้ยง หน่วยความจำที่สามารถอ่านหรือเขียนได้ เช่นหน่วยความจำแบบ แรม ข้อมูลจะหายไปเมื่อไม่มีไฟเลี้ยงหน่วยความจำ แต่หน่วยความจำประเภทอ่านได้อย่างเดียว เช่น รอม เป็นหน่วยความจำที่ข้อมูลจะไม่มีการเปลี่ยนแปลงแม้ว่าจะไม่มีไฟเลี้ยงก็ตาม ดังนั้นหน่วยความจำแบบรอม จะเป็นหน่วยความจำ ชนิด Nonvolatile

3. Single chip form หน่วยความจำแบบสารกึ่งตัวนำนี้จะบรรจุอยู่ในชิพไอซี ซึ่งจะมีขนาด ต่าง ๆ กัน เช่นขนาดของคำ (word length) จำนวนตำแหน่งของหน่วยความจำและโครงสร้างภายใน ส่วนการถอดรหัสตำแหน่งและ การเข้าถึงข้อมูลจะเป็นส่วนที่อยู่ภายในไอซีเอง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชนิดของหน่วยความจำ

หน่วยความจำอาจแบ่งตามลักษณะการอ่าน และการอ่านเขียนข้อมูลได้ 3 แบบใหญ่ๆ

คือ

1. Read Only Memory (ROM)
2. Random Access Memory (RAM)
3. Serial Access Memory

ซึ่งลักษณะของหน่วยความจำทั้ง 3 แบบอาจกล่าวโดยสรุปได้ดังนี้ คือ Read only Memory (ROM) เป็นหน่วยความจำที่ใช้เก็บข้อมูลแบบถาวร หรือกึ่งถาวร คือข้อมูลที่จะต้องถูกเขียนลงไปนั้น หน่วยความจำตั้งแต่ต้น หลังจากนั้นก็ใช้เรียกใช้เพียงอย่างเดียวแต่ไม่สามารถแก้ไขเปลี่ยนแปลงได้อีก ดังนั้น เราเรียกหน่วยความจำแบบนี้ว่า Read only หน่วยความจำแบบรวมนี้เป็นหน่วยความจำแบบ Nonvolatile คือข้อมูลในรอมจะไม่สูญหายเมื่อไม่มีไฟเลี้ยง หน่วยความจำรอม จะใช้ระบบไมโครคอมพิวเตอร์ทุกเครื่อง เพื่อเก็บโปรแกรมสำหรับใช้ระบบเริ่มต้นทำงาน ที่เรียกว่า โปรแกรมบูตโหลดหรือ โปรแกรม Bootstrap รอมสามารถจำแนกเป็นชนิดต่างๆ ได้ตามลักษณะการเขียนข้อมูล เข้าไปในหน่วยความจำดังนี้

Mask Programmed ROM เป็นรอม ชนิดที่ได้ทำการเขียนข้อมูลเข้าไปในหน่วยความจำตั้งแต่ตอนสร้างชิพ และไม่สามารถแก้ไขข้อมูลภายในอีก รอมแบบนี้จะต้องสร้างจำนวนครั้งละมาก ๆ เนื่องจากต้นทุนการผลิตค่อนข้างสูง

Programmable Read Only Memory หรือ PROM หน่วยความจำแบบนี้ผู้ใช้สามารถโปรแกรมเองได้ โดยใช้กรรมวิธีจ่ายพัลส์แรงดันสูงเข้าไปทำลายพิวส์ ภายในตัวไอซีหน่วยความจำ เพื่อให้เป็นลอจิก 0 หรือ 1 ตำแหน่งที่กำหนดและเมื่อโปรแกรมเข้าไปแล้วไม่สามารถทำการแก้ไขได้เช่นกัน

Erasable Programmable Read Only Memory หรือ EPROM หน่วยความจำชนิดนี้ ผู้ใช้สามารถโปรแกรมข้อมูลลงไปให้และสามารถลบได้โดยใช้รังสี Ultra violet ฉายผ่านช่องกระจกตัว ไอซี และหลังจากที่ทำการลบข้อมูลออกแล้ว ก็สามารถโปรแกรมข้อมูลใหม่ลงไปได้อีก

Electrically Aoterable Read Only Memory หรือ EAROM เป็นหน่วยความจำที่สามารถโปรแกรมข้อมูลเข้าไปได้ และสามารถลบออกได้โดยใช้สัญญาณไฟฟ้าทำการโปรแกรมและการลบทำได้โดยสะดวก และอาจทำได้โดยไม่ต้องถอดออกจากวงจร

RANDOM ACCESS MEMORY หน่วยความจำแบบแรมนี้ เป็นหน่วยความจำที่มี

การทำงานที่ต่างจากรอม คือมันสามารถที่จะทำการเขียนหรือ อ่านข้อมูลแต่ละคำหรือแต่ละบิต ณ ที่ตำแหน่งใด ๆ ในพื้นที่ของหน่วยความจำนี้จะใช้เวลาเท่า ๆ กันในกรณีของการเขียนข้อมูลเข้า หน่วยความจำ จะใช้เวลามากกว่าการอ่านข้อมูลมาก หน่วยความจำแบบ แรมจะเป็นชนิด Volatile คือต้องการไฟเลี้ยงตลอดเวลา เมื่อขาดไฟเลี้ยงข้อมูลในแรมจะหายไปแรมจะแบ่งออกได้ 2 แบบ คือ

Static RAM หรือเรียกย่อ ๆ ว่า SRAM แรมชนิดนี้ หน่วยความจำแต่ละเซลล์ จะใช้ลักษณะของวงจร ฟลิปฟลอป เป็นพื้นฐาน ดังนั้นเมื่อไม่มีการเขียนข้อมูลเข้าไปใหม่ ข้อมูลนั้น ๆ จะคงที่ตลอดไปเท่าที่มีไฟเลี้ยงอยู่

Dynamic RAM หรือ DRAM ลักษณะของแรมชนิดนี้ ใช้การเก็บประจุที่ขาเกตของ MOSFET เพื่อเป็นการเก็บข้อมูล และเมื่อมีการอ่านข้อมูลออกมา ประจุที่ถูกเก็บไว้จะคายออกมาหมดไป ซึ่งถ้าเป็นเช่นนี้ หน่วยความจำก็จะเป็นแบบ Destructive ดังนั้น DRAM คือ มีความหนาแน่นของเซลล์หน่วยความจำสูงกว่า มีความสิ้นเปลืองกำลังต่ำกว่าและใช้เวลาในการเข้าข้อมูลเร็วกว่าแต่มีข้อเสียในเรื่องการที่จะต้องทำการ รีเฟรช ซึ่งจะยุ่งยากในการใช้มากกว่า SRAM

SERIAL ACCESS MEMORY หน่วยความจำแบบนี้ใช้วิธีการ เก็บข้อมูลตามลำดับก่อนหลังตามกันไป ดังนั้นเวลาอ่านจะต้องเรียงลำดับด้วย เช่นเดียวกับเทป ซึ่งลักษณะของหน่วยความจำแบบนี้คล้ายกับวงจรซีพรีจีสเตอร์ หน่วยความจำแบบนี้ Volatile และ Nonvolatile เช่น

Magmetic Bubble Memory หน่วยความจำแบบนี้มีความจุข้อมูลสูงใกล้เคียงกับเทป สามารถเขียนและอ่านข้อมูลได้เช่นเดียวกับ เทป และหน่วยความจำชนิด Nonvolatile ไม่จำเป็นต้องมีการรีเฟรช แต่ความเร็วในการอ่านหรือ เขียนข้อมูลค่อนข้างช้ากว่าแรมแบบสารกึ่งตัวนำแต่มีความเร็วสูงกว่าเทปมาก หน่วยความจำแบบนี้มีแนวโน้มจะเข้ามาแทนที่ Disk ในอนาคต

Charge Coupled Device (CCD) หน่วยความจำแบบนี้สามารถเก็บข้อมูลได้สูง สามารถอ่านหรือเขียนได้ แต่เป็นแบบ Volatile ความเร็วการอ่านและเขียนข้อมูลค่อนข้างสูงเกือบเท่ากัน แรม แบบสารกึ่งตัวนำ CCD มีราคาถูก แต่การใช้งานยุ่งยากจึงไม่เป็นที่นิยม

การต่อหน่วยความจำกับไมโครโพรเซสเซอร์

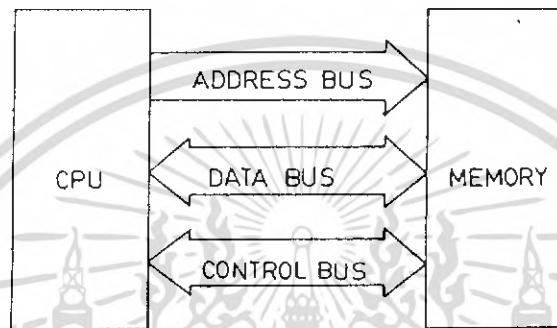
ในการเชื่อมกันระหว่างไมโครโพรเซสเซอร์กับหน่วยความจำโดยทั่วไปนั้น จะต้องมีความสัมพันธ์การเชื่อมต่อไปนี้

1. บัสตำแหน่ง ซีพียูจะใช้ข้อมูลแบบบัสนี้สำหรับการเลือกตำแหน่งของหน่วยความจำที่จะต้องใช้เป็นต้นทางข้อมูล (เมื่อทำการอ่าน) หรือตำแหน่งปลายทางข้อมูล (เมื่อทำการเขียน)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. บัสข้อมูล ใช้สำหรับเป็นทางผ่านของข้อมูล ซีพียู เพื่อออกไปยังหน่วยความจำ หรือข้อมูลจากหน่วยความจำเพื่อมายัง ซีพียู

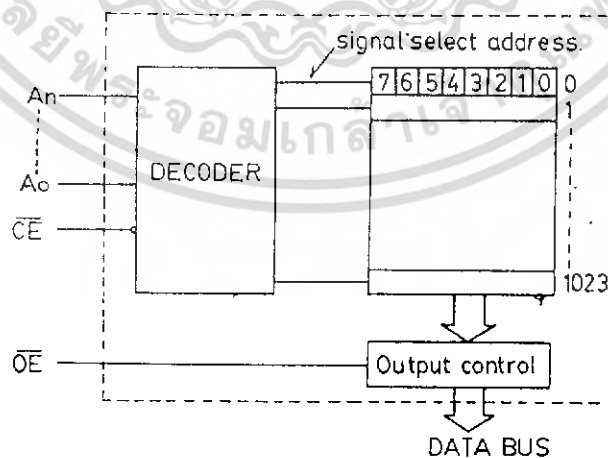
3. บัสควบคุม บัสนี้ใช้สำหรับการส่งสัญญาณควบคุมการทำงานในการอ่านหรือการเขียนข้อมูล เช่นสัญญาณ RD WR และสัญญาณเพื่อให้มีการทำงานพร้อมกันเป็นต้น ซึ่งลักษณะของการเชื่อมต่อซีพียูกับหน่วยความจำ แสดงได้ดังรูปที่ 26



รูปที่ 26 การเชื่อมต่อหน่วยความจำกับซีพียู

ลักษณะภายในหน่วยความจำแบบ ROM

ลักษณะการจัดโครงสร้างภายในของหน่วยความจำรอมแบบที่เป็น EPROM แสดงได้ดัง



รูปที่ 27 การจัดโครงสร้างของหน่วยความจำแบบ EPROM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รูปที่ 27 ซึ่งหน่วยความจำแบบนี้สามารถโปรแกรมได้ และอ่านได้ ในการกำหนดขนาดของหน่วยความจำแบบ EPROM สามารถกำหนดได้ดังนี้ เช่น 1024×8 หมายถึงหน่วยความจำที่มี 1024 คำแห่ง หรือ 1K และแต่ละคำแห่งประกอบด้วย 8 บิต ถ้าหน่วยความจำมี 1024 คำแห่ง ก็จะต้องมีสัญญาณในการกำหนดคำแห่งเท่ากับ 10 สัญญาณ นั่นคือ $2^{10} = 1024$ สัญญาณเหล่านี้ปกติจะมาจากบัสตำแหน่งของไมโครโปรเซสเซอร์ ไอซีหน่วยความจำที่ใช้กันอยู่หลายขนาด เช่น ถ้าเป็น EPROM เบอร์ 2716 จะมีขนาด $2K \times 8$ เบอร์ 2732 มีขนาด $4K \times 8$ เบอร์ 2764 มีขนาด $8K \times 8$ เป็นต้น

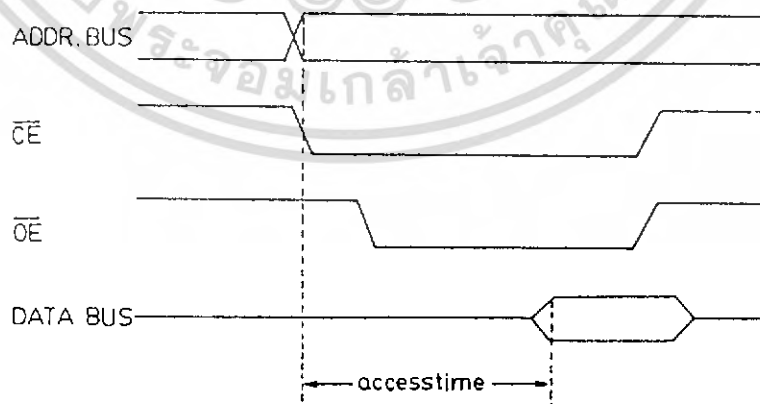
ลำดับการอ่านข้อมูลจากหน่วยความจำ EPROM

ในการใช้ซีพียูจะอ่านข้อมูลจากหน่วยความจำแบบ EPROM จะมีลำดับดังนี้

1. ซีพียูส่งข้อมูลของตำแหน่งที่ต้องการอ่านออกไปบนบัสตำแหน่ง
2. ซีพียูกำเนิดสัญญาณเลือกใช้ชิพ (Chip enable และ Output enable)

และส่งไปยังหน่วยความจำ ซึ่งสัญญาณนี้อาจส่งไปในเวลาเดียวกันกับการส่งสัญญาณในการกำหนดตำแหน่งก็ได้

3. หลังจากนั้น ซีพียูต้องรอสักระยะเวลาหนึ่ง เพื่อให้หน่วยความจำทำการอ่านข้อมูลเพื่อส่งออกไปยังบัสข้อมูลของระบบซึ่งช่วงเวลานี้เรียกว่า Access time ซึ่งมีค่าอยู่ระหว่าง 150–450 nS และหลังจากนั้น ซีพียูจึงทำการอ่านข้อมูลในช่วงนี้เข้าไป
4. สัญญาณเลือกชิพ เลิกแอกติฟ บัสข้อมูลกลับสู่สภาวะอิมพีแดนซ์สูงซึ่งลำดับขั้นการอ่าน EPROM นี้ สามารถเขียนเป็นไคอะแกรมเวลาแสดงได้ดังรูป 28

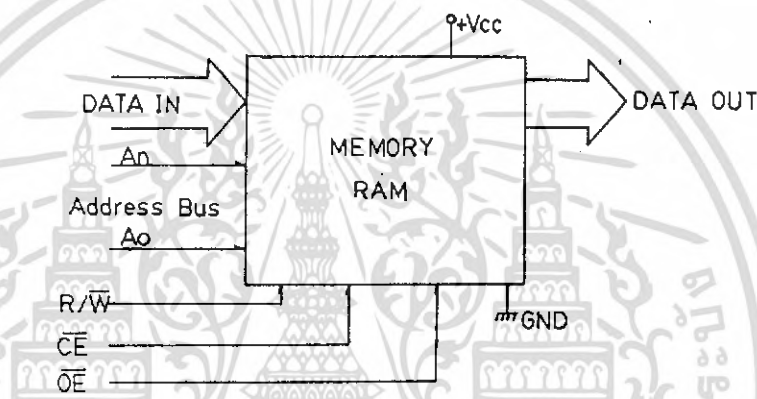


รูปที่ 28 ไคอะแกรมเวลาของการอ่านข้อมูลจาก EPROM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ลักษณะหน่วยความจำแบบ RAM

หน่วยความจำแบบแรมก็จะมีลักษณะภายนอก คล้าย ๆ กับหน่วยความจำแบบรอมแต่แตกต่างกันที่แรม จะมีสายสัญญาณเพิ่มขึ้นมาอีกเส้นหนึ่ง คือสัญญาณควบคุมการอ่านและเขียน บล็อกไดอะแกรมของแรมแสดงได้ดังรูปที่ 29 ซึ่งในที่นี้จะใช้แรมแบบสแตติกบล็อกไดอะแกรม บัสสำหรับข้อมูลเข้าและส่งออกจะเป็นบัสเดียวกันแต่เป็นบัสแบบ 2 ทิศทางเป็นแบบบัสขาเข้าเมื่ออยู่ในช่วงของการเขียน และเป็นแบบบัสขาออกเมื่ออยู่ในช่วงของการอ่านหน่วยความจำแบบ สแตติก หรือ SRAM มีหลายขนาดให้เลือกใช้เช่น เบอร์ 2114 เป็นแรมขนาด 1K x 4 เบอร์ 6116 เป็นแรมขนาด 2K x 8 และเบอร์ 6264 เป็นแรมขนาด 8K x 8 เป็นต้น



รูปที่ 29 บล็อกไดอะแกรมของหน่วยความจำแบบแรม

ลำดับขั้นในการอ่านข้อมูลแบบแรม

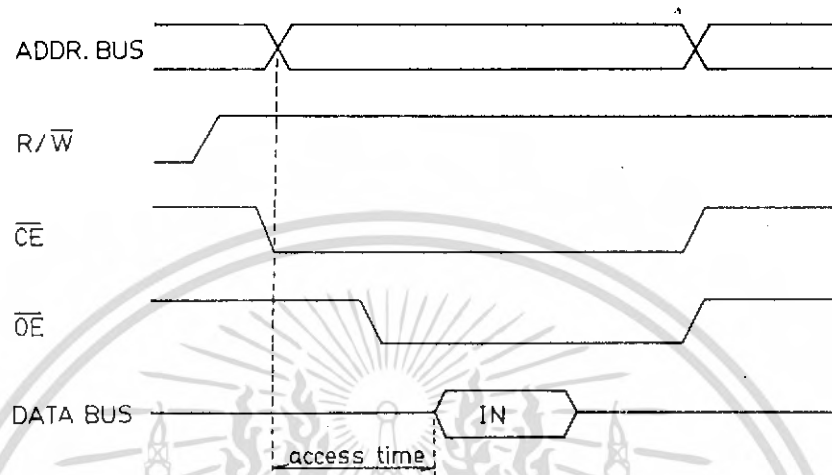
ในการอ่านข้อมูลจากแรมนั้น ต้องใช้สัญญาณควบคุมการอ่าน และเขียนมาช่วยในการควบคุมการทำงานด้วยโดยการกำหนดให้สัญญาณควบคุมการอ่านและเขียนนี้ แอคทีฟตามการลักษณะของการอ่านหรือเขียนข้อมูลที่ต้องการ เช่น ถ้าให้สัญญาณนี้อยู่ในสภาวะแอคทีฟอ่าน (สมมุติ = 1) การทำงานก็เป็นการอ่านข้อมูลจากหน่วยความจำตามตำแหน่งที่ระบุ ซึ่งลำดับขั้นของการเขียนไดอะแกรมเวลาได้ดังรูปที่ 30 และมีลำดับดังนี้

1. ซีพียูส่งข้อมูลของตำแหน่งที่ต้องการอ่านออกไปบนบัสตำแหน่ง (กำหนดให้ในขณะนี้สัญญาณการอ่านแอคทีฟ)
2. ซีพียูส่งสัญญาณเลือกที่ใช้ซีพียูและส่งไปยังหน่วยความจำ
3. ซีพียูรอเวลาสักระยะเวลาหนึ่ง นับตั้งแต่ส่งข้อมูลเพื่อกำหนดตำแหน่งเข้าไปซึ่งเวลานี้ก็คือ Access time ซึ่งอาจมีค่าประมาณ 35-200 nS แล้วแต่ชนิดและขนาดของแรมหลังจากช่วง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Access time ผ่านไปแล้ว ข้อมูลจากหน่วยความจำก็จะปรากฏบนบัสข้อมูล ซึ่งซีพียูจะอ่านข้อมูลในช่วงนี้เข้าไป

4. สัญญาณเลือกชิพเล็กแอกตีฟ บัสข้อมูลกลับสู่สภาวะอิมพีแดนซ์สูง

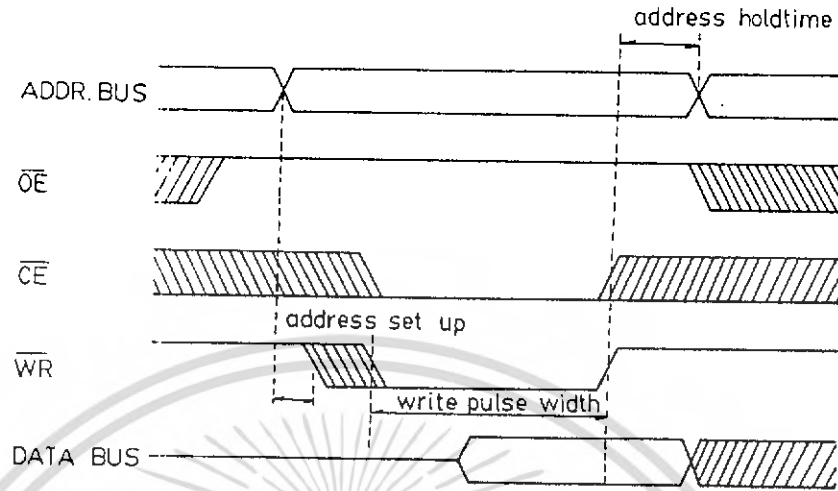


รูปที่ 30 ไคอะแกรมเวลาของการอ่านข้อมูลจากแรม

ลำดับขั้นการเขียนข้อมูลลงหน่วยความจำแบบแรม

ในการเขียนข้อมูลลงหน่วยความจำแบบแรม จะมีลำดับขั้นดังนี้

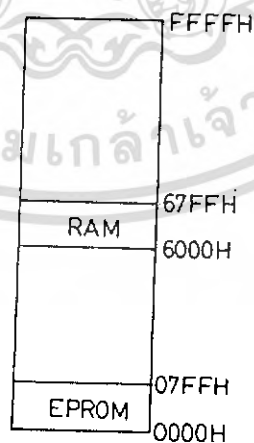
1. ซีพียู ส่งข้อมูลของตำแหน่งที่ต้องการให้เขียนไปบนบัสตำแหน่ง และอาจพร้อมทั้งสัญญาณการเลือกใช้ชิพ (\overline{CE})
 2. ข้อมูลที่จะเป็นข้อมูลอินพุต ถูกส่งเข้ามาบัสอินพุตของแรม
 3. หน่วยความจำต้องการเวลาสักระยะเวลาหนึ่ง เพื่อให้ข้อมูลในการกำหนดตำแหน่งอยู่ในสภาวะคงที่ซึ่งช่วงเวลานี้คือ Address setup time ซึ่งจะมีค่าประมาณ 20 nS จากนั้นระบบจึงส่งสัญญาณควบคุมการเขียนให้ข้อมูลอยู่ในสภาวะแอกตีฟ
 4. หลังจากที่สัญญาณควบคุมการเขียนแอกตีฟสักระยะเวลาหนึ่ง หน่วยความจำจะทำการนำข้อมูลจากอินพุต เข้าเขียนเข้าไปยังหน่วยจำ และเมื่อสัญญาณควบคุมการเขียนเล็กแอกตีฟก็เป็นอันสิ้นสุด ขบวนการเขียนลำดับขั้นของการเขียนข้อมูลลงแรมสามารถแสดงได้โดยไคอะแกรมเวลาดังรูปที่ 31
- จากรูปเห็นได้ว่าการกำหนดให้สัญญาณ \overline{OE} อยู่ในสภาวะไม่แอกตีฟ เพื่อไม่ให้มีสัญญาณเอาต์พุตออกมาที่บัสข้อมูล แต่ถ้าสัญญาณ \overline{OE} อยู่ในสภาวะแอกตีฟ และขณะนั้น สัญญาณ \overline{CE} แอกตีฟด้วยก็จะมีสภาวะ เป็นอิมพีแดนซ์สูงเช่นกัน



รูปที่ 31 ระยะเวลาของการเขียนหน่วยความจำ

การต่อหน่วยความจำกับซีพียู Z-80

ซีพียู Z-80 มีสัญญาณในการเลือกตำแหน่ง 16 สัญญาณ คือ A15-A0 นั้นหมายความว่าซีพียูสามารถติดต่อกับหน่วยความจำได้ถึง 2^{16} หรือ 65536 ตำแหน่งดังนั้นในการออกแบบ

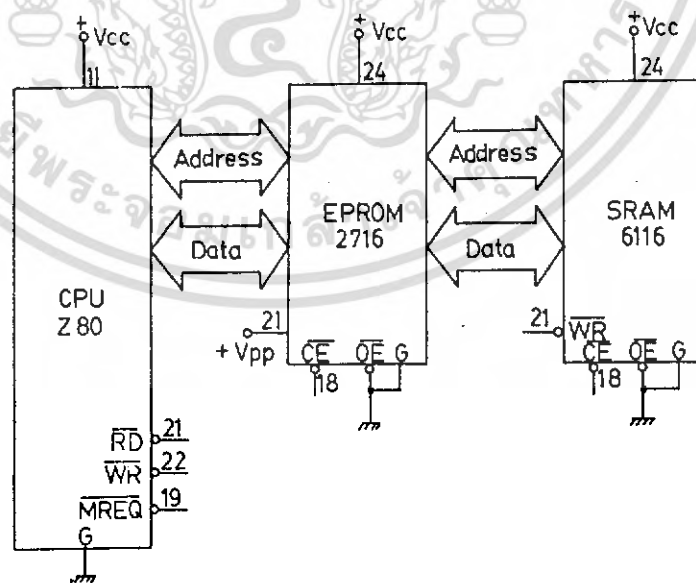


รูปที่ 32 แสดงการจัดแบ่งหน่วยความจำ หรือ Memory map

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ระบบไมโครคอมพิวเตอร์เราต้องทำการจัดสรรเนื้อที่เสียก่อนว่าเราจะใช้หน่วยความจำส่วนไหนเพื่อประโยชน์อะไร และต้องใช้หน่วยความจำแบบใด ดังนั้นผู้ออกแบบระบบ ต้องทำการสร้างตารางเพื่อจัดสรรหน่วยความจำหรือที่ เรียกว่า Memory Map ขึ้นเพื่อกำหนดตำแหน่งของการที่จะใช้หน่วยความจำแบบรอมหรือแบบแรมซึ่งตัวอย่างของ Memory map แสดงได้ดังรูปที่ 32

จากรูปจะเห็นได้ว่าตำแหน่งต่ำสุดของ Memory map คือตำแหน่งที่ 0000H จะต้องใช้หน่วยความจำแบบรอม เนื่องจากเมื่อทำการรีเซ็ตซีพียู ตำแหน่งเริ่มต้นที่ซีพียูจะทำการอ่านในครั้งแรกคือตำแหน่งที่ 0000H นี้เอง และข้อมูลนี้จะต้องเป็นคำสั่งเพื่อให้ซีพียูมีการทำงานได้ต่อไปนั่นคือโปรแกรมในส่วนแรกที่บรรจุในหน่วยความจำแบบรอมนี้ จะเป็นโปรแกรมที่เรียกว่า Bootstrap program หรือมอนิเตอร์โปรแกรมนั่นเอง และส่วนนอกเหนือจากนั้นอาจกำหนดให้ส่วนใดเป็นรอมหรือส่วนใดเป็นแรมก็ได้ จากรูป เราได้ทำการแบ่งหน่วยความจำออกเป็น ส่วน ๆ หรือ เรียกว่า แบงก์ (Bank) โดยมีส่วนขนาดแบงก์ละ 2k แบงก์ 0 คือ ตั้งแต่ตำแหน่ง 0000H-07FFH กำหนดให้พื้นที่ของ ROM A แบงก์ ตั้งแต่ตำแหน่ง 6000H-67FFH ใช้เป็นพื้นที่ของ RAM A จาก Memory map เห็นได้ว่าแต่ละแบงก์มีขนาด 2K ดังนั้นจะให้หน่วยความจำ EPROM เบอร์ 2716 สำหรับพื้นที่ที่เป็นรอมและใช้ SRAM เบอร์ 6116 สำหรับพื้นที่ที่เป็นแรมซึ่งหน่วยความจำทั้งสองนี้มีขนาด 2K เท่ากัน ดังนั้นหน่วยความจำทั้งสองจะต้องมีอินพุตเพื่อ การกำหนดตำแหน่ง 11 เส้นคือ A10-A0 ซึ่งอินพุตทั้ง 11 นี้สามารถต่อได้โดยตรงกับบัสตำแหน่งของซีพียู ที่ A10-A0 ส่วน A15-A11 ของซีพียูจะนำมาใช้เพื่อเลือกว่าจะใช้หน่วยความจำในพื้นที่ส่วนใด วงจรในการต่อหน่วยความจำทั้งสองเข้ากับบัสข้อมูลของซีพียูแสดงได้ดังรูปที่ 33



รูปที่ 33 แสดงการต่อหน่วยความจำเข้ากับบัสตำแหน่งและบัสข้อมูล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

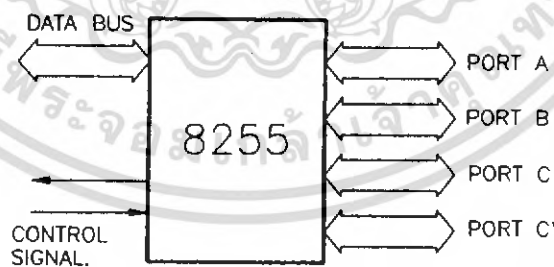
อินพุท/เอาต์พุท พอร์ต

การใช้งานไมโครโปรเซสเซอร์ จะต้องมีการเชื่อมต่อ กับอุปกรณ์ภายนอก เช่น สวิตช์ รีเลย์ หรืออุปกรณ์ตรวจจับ อื่น ๆ การเชื่อมต่อดังกล่าวจะต้องเชื่อมต่อกับพอร์ตอินพุท/เอาต์พุทเพื่อให้ไมโครโปรเซสเซอร์ส่งสัญญาณไปยังอุปกรณ์ต่าง ๆ ความเงื่อนไขที่เกิดขึ้นและสามารถตรวจสอบได้ด้วย ไมโครโปรเซสเซอร์เอง

การเชื่อมต่อที่ง่ายที่สุด คือการเชื่อมต่อโดยการที่ใช้เกตลอจิก 3 สถานะ โดยสัญญาณควบคุมพอร์ตอินพุท จะเป็นตัวเปิดเกตให้ข้อมูลเข้าสู่บัส และไมโครโปรเซสเซอร์อ่านเข้าไป แต่สำหรับพอร์ตเอาต์พุทจะใช้แอสต์ฟลิปฟลอป ทำหน้าที่รับสัญญาณข้อมูลจากไมโครโปรเซสเซอร์ที่ส่งเข้าไปในบัส และได้รับการจับไว้ที่พอร์ตและไมโครโปรเซสเซอร์จะส่งสัญญาณมาควบคุมอีกที่

อินพุท/เอาต์พุท ที่ใช้เกตขนาดเล็กดังกล่าว มีจุดอ่อนในเรื่องของจำนวนไอซีบริษัทผู้ออกแบบไมโครโปรเซสเซอร์ส่วนใหญ่จึงออกแบบชิพ เพื่อทำหน้าที่อินพุท/เอาต์พุท ของระบบ และมีข้อดีในเรื่องการทำงาน สำหรับ ชิพยู 2-80 แล้ว ชิพไอซี ที่รู้จักกันมากที่สุด คือ ไอซี 8255 ของบริษัทอินเทล และสามารถประยุกต์ใช้กับ ชิพยู 2-80 ได้ง่าย

8255 เป็นไอซีที่มี 40 ขา สามารถต่อเป็นพอร์ตให้ไมโครโปรเซสเซอร์ได้ 3 พอร์ต โดยมีโครงสร้างพื้นฐานแสดงได้ดังรูปที่ 34

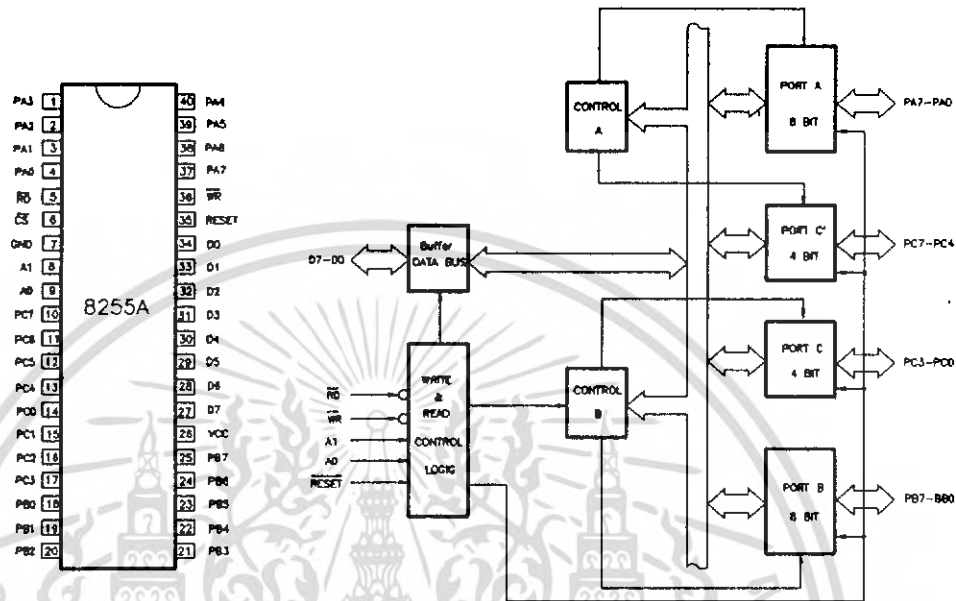


รูปที่ 34 แผนผังแสดงโครงสร้างของไอซี 8255

การเรียกพอร์ตของ 8255 จะเรียกพอร์ตต่าง ๆ ว่า พอร์ต A พอร์ต B พอร์ต C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โดยพอร์ต C แยกออกเป็น 2 ส่วนคือ พอร์ต C ล่างหรือตั้งแต่ PC₀-PC₃ จำนวน 4 บิต และพอร์ต C บน หรือตั้งแต่ PC₄-PC₇ ที่พิเศษคือพอร์ตทุกพอร์ตาใช้ได้ทั้ง อินพุตและเอาต์พุตพอร์ต



รูปที่ 35 แสดงผังภายในและการจัดขาของไอซี 8255

การทำงานของไอซี 8255 จะใช้สัญญาณควบคุมจากไมโครโปรเซสเซอร์ส่งคำสั่งมา จากโปรแกรมการทำงาน หรือกำหนดรูปแบบของพอร์ตาให้เป็นอินพุตหรือเอาต์พุตก็ได้

ขาต่าง ๆ ของ 8255

เพื่อให้เข้าใจวิธีการต่อใช้งานระหว่าง Z-80 กับ 8255 จึงเข้าใจความหมายและ ตำแหน่งของขาต่าง ๆ จำนวน 40 ขา ของไอซีซึ่งประกอบด้วย

D₀-D₇ เป็นขาข้อมูลอินพุตเอาต์พุตจะผ่านเข้าออกทางนี้ D₀-D₇ จึงจะต้อง ต่อเข้ากับระบบบัสของไมโครโปรเซสเซอร์

\overline{CS} ขานี้เป็นขาอินพุตที่จะได้รับสัญญาณจากภายนอกเพื่อเลือกชิพ 8255 โดยขานี้จะ แอคติฟที่ลอจิก "0" จะทำให้ 8255 ต่อเข้ากับ ระบบบัสของไมโครโปรเซสเซอร์เพื่อให้ไมโคร โปรเซสเซอร์ เขียนหรืออ่านข้อมูลได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

\overline{RD} เป็นสัญญาณอินพุตที่ส่งมาจากซีพียู เมื่อสัญญาณที่ขานี้เป็น "0" และสัญญาณ เป็น "0" ด้วย ไอซี 8255 จะทำตัวเป็นทางผ่านให้ไมโครโปรเซสเซอร์อ่านข้อมูลจากพอร์ตอินพุตเข้ามา

\overline{WR} เป็นสัญญาณการเขียน จะแอกตีฟเมื่อสัญญาณ \overline{WR} และสัญญาณ \overline{CS} เป็น "0" สัญญาณนี้จะมาจาก ซีพียู เมื่อต้องการเขียนข้อมูลลงพอร์ตที่กำหนด

A_0-A_1 ลอจิกของสัญญาณทั้งสองจะถอดรหัสออกเป็น 4 รหัส เพื่อกำหนดรีจิสเตอร์ภายในที่เชื่อมต่อกับพอร์ตอินพุตเอาต์พุตของ 8255

RESET เป็นสัญญาณที่ส่งมาจากภายนอกเข้ามาทำการรีเซ็ต 8255 เพื่อทำการเคลียร์สถานะ ต่าง ๆ ของ 8255

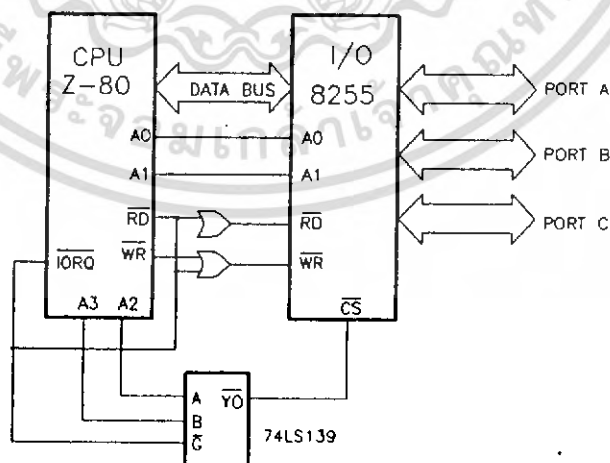
PA_0-PA_7 เป็นสัญญาณพอร์ต A ของ 8255 การเลือกพอร์ตนี้ทำได้โดยสัญญาณ แอดเดรส A_0-A_1

BA_0-PB_7 เป็นสัญญาณพอร์ต B ของ 8255 การเลือกพอร์ตนี้ทำได้โดยสัญญาณ แอดเดรส A_0-A_1

PC_0-PC_7 เป็นสัญญาณพอร์ต C ของ 8255 การเลือกพอร์ตนี้ทำได้โดยสัญญาณ แอดเดรส A_0-A_1 พอร์ต C แบ่งได้เป็น 2 กลุ่มคือ PC_0-PC_3 และกลุ่ม PC_4-PC_7

และการเชื่อมต่อ Z-80 กับ 8255

หากพิจารณาจากขาต่าง ๆ ของ 8255 จะเห็นได้ว่าส่วนควบคุมที่จะเชื่อมต่อกับ บัสได้ง่าย จากรูปที่ 36 ลองต่อ 8255 เป็นพอร์ต Z-80 และให้ พอร์ต A พอร์ต B และพอร์ต C บนล่าง เป็นพอร์ตหมายเลข 00H, 01H, 02H และ 03H ตามลำดับ



รูปที่ 36 การเชื่อมต่อ 8255 กับ Z-80

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รีจิสเตอร์ภายในของ 8255

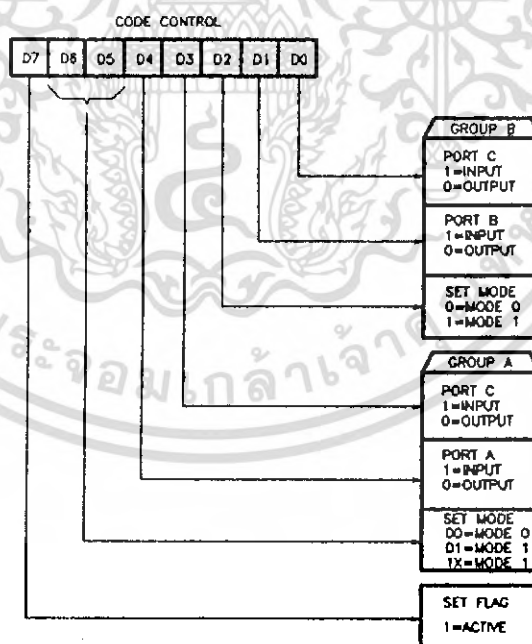
การโปรแกรมให้ 8255 ทำงานตามที่ต้องการจากการที่ 8255 ที่ Z-80 มองเห็น 4 พอร์ต แต่ละพอร์ตจะเสมือนเป็นรีจิสเตอร์ที่สามารถอ่านเขียนได้ รีจิสเตอร์แต่ละตัวนี้จะถูกกำหนดด้วยแอดเรสตามที่ตั้งไว้ เพราะฉะนั้นการที่จะให้พอร์ตของ 8255 เป็นอย่างไรขึ้นอยู่กับ การส่งรหัสควบคุม (control code) ไปควบคุมการทำงานของ 8255 ในที่นี้โดยใช้พอร์ทหมายเลข 03H การควบคุมการทำงานของ 8255 มีทั้งหมด 3 โหมด ในที่นี้จะยกมาให้แค่โหมด 0 เท่านั้น

การทำงานของ 8255 ในโหมด 0

การกำหนดการทำงาน จะส่งข้อมูลที่คำสั่งเข้าโปรแกรมในพอร์ตควบคุมของ 8255 แต่ละบิตที่จะส่งออกไปจะมีความหมายภายในตัวเอง การให้ค่ารหัสบิตต่าง ๆ เข้าไปในรหัสควบคุม แล้วส่งไปยังรีจิสเตอร์ของพอร์ตควบคุม ความหมายของบิตต่าง ๆ มีดังนี้

บิต D_7 เป็นบิตแสดงรหัสควบคุม ถ้าบิตนี้เป็น "1" หมายถึงรหัสควบคุมนี้มีผลต่อการเปลี่ยนแปลงโหมดของ 8255

บิต D_6, D_5 เป็นการเลือกโหมดของ 8255 ซึ่งมี 3 โหมด ดังแสดงในรูปที่ 37



รูปที่ 37 ความหมายของบิตต่าง ๆ ในรหัสควบคุม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บิต D_1 ถ้ามีค่าเป็น "0" หมายถึงการกำหนดพอร์ต A เป็นเอาต์พุตถ้าเป็น "1" จะหมายถึงการกำหนดให้พอร์ต A เป็นอินพุต

บิต D_3 เป็นบิตที่บอกถึงการเซตของพอร์ต C บนเป็น "0" จะทำให้พอร์ต C เป็นเอาต์พุต

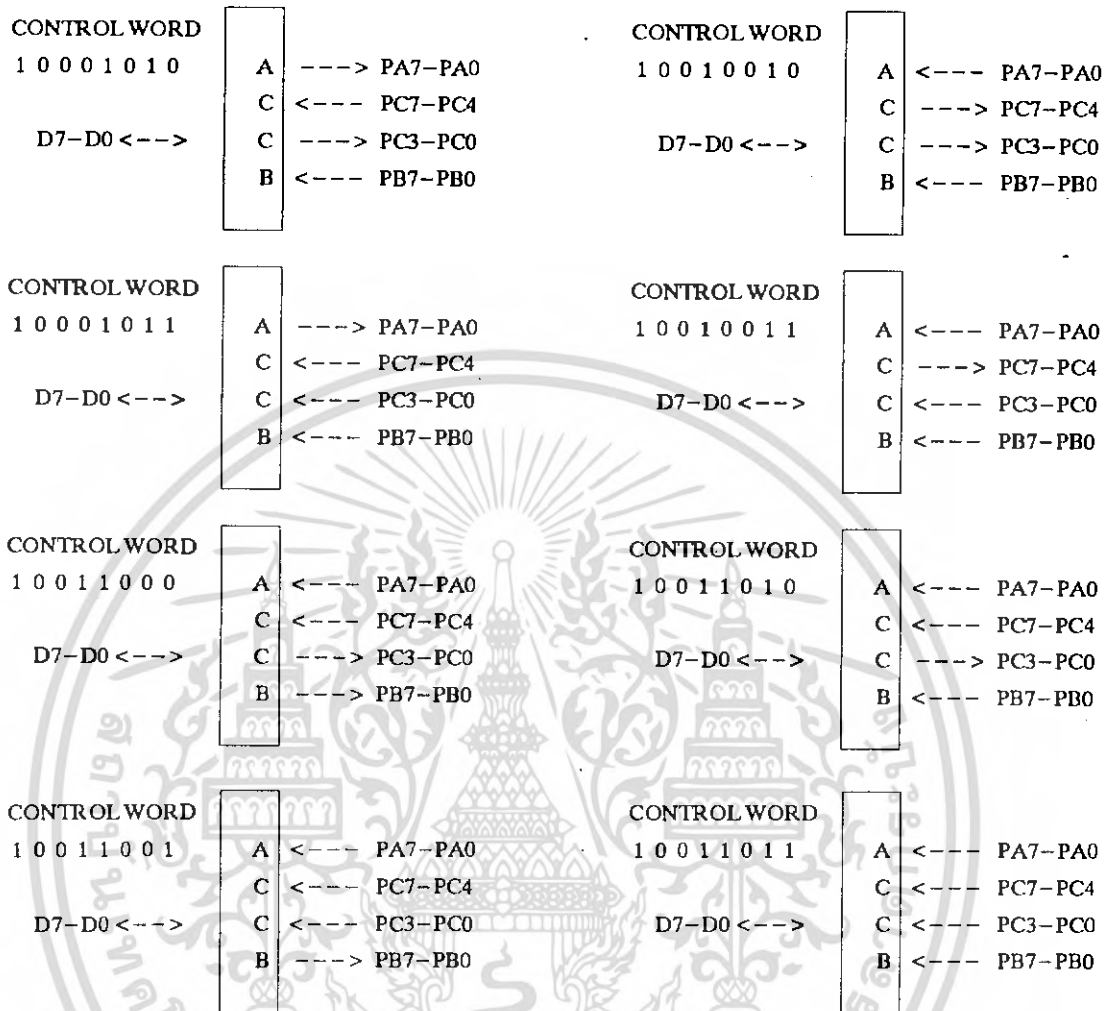
บิต D_2 เป็นบิตที่บอกถึงการเซตใหม่คของพอร์ต B ถ้าเป็น "0" หมายถึงการกำหนดให้พอร์ต B ทำงานในโหมด 0

บิต D_1 เป็นบิตที่บอกถึงการทำงานของพอร์ต B ถ้าเป็น "0" หมายถึงกำหนดให้พอร์ต B ทำงานเป็นพอร์ต เอาต์พุต

บิต D_0 เป็นการกำหนดพอร์ตของพอร์ต C ล่าง ถ้าเป็น "0" หมายถึงเป็นเอาต์พุต ที่กล่าวมาแล้วจะเห็นว่าสามารถที่จะโปรแกรม เลือกรูปแบบพอร์ตได้ทั้งสิ้น 16 แบบ ในโหมด 0

CONTROL WORD 1 0 0 0 0 0 0 0	A ----> PA7-PA0 C ----> PC7-PC4 D7-D0 <----> C ----> PC3-PC0 B ----> PB7-PB0	CONTROL WORD 1 0 0 0 0 0 1 0	A ----> PA7-PA0 C ----> PC7-PC4 D7-D0 <----> C ----> PC3-PC0 B <---- PB7-PB0
CONTROL WORD 1 0 0 0 0 0 0 1	A ----> PA7-PA0 C ----> PC7-PC4 D7-D0 <----> C <---- PC3-PC0 B ----> PB7-PB0	CONTROL WORD 1 0 0 0 0 0 1 1	A ----> PA7-PA0 C ----> PC7-PC4 D7-D0 <----> C <---- PC3-PC0 B <---- PB7-PB0
CONTROL WORD 1 0 0 0 1 0 0 0	A ----> PA7-PA0 C <---- PC7-PC4 D7-D0 <----> C ----> PC3-PC0 B ----> PB7-PB0	CONTROL WORD 1 0 0 1 0 0 0 0	A <---- PA7-PA0 C ----> PC7-PC4 D7-D0 <----> C ----> PC3-PC0 B ----> PB7-PB0
CONTROL WORD 1 0 0 0 1 0 0 1	A ----> PA7-PA0 C <---- PC7-PC4 D7-D0 <----> C <---- PC3-PC0 B ----> PB7-PB0	CONTROL WORD 1 0 0 1 0 0 0 1	A <---- PA7-PA0 C ----> PC7-PC4 D7-D0 <----> C <---- PC3-PC0 B ----> PB7-PB0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



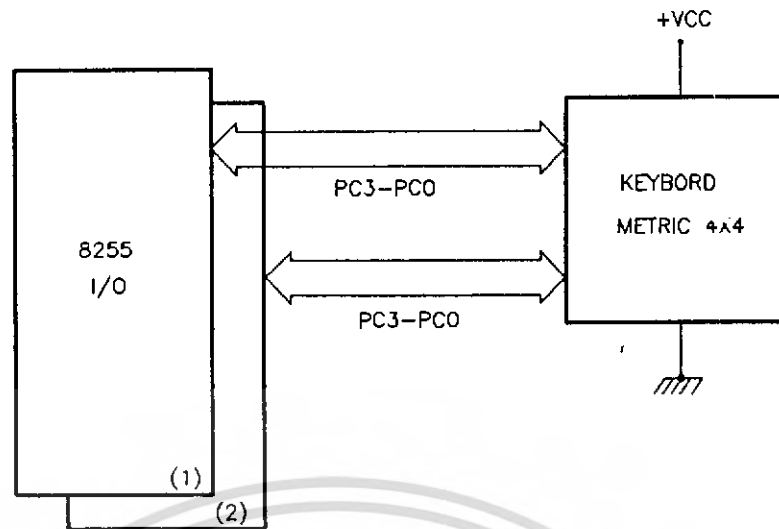
การต่อกับอุปกรณ์ภายนอกกับพอร์ต I/O

การอินเตอร์แฟคอุปกรณ์ภายนอกกับ พอร์ต อินพุท เอาท์พุท สามารถทำได้โดยตรงกับ ชิป 8255 ซึ่งสามารถกำหนด หรือ โปรแกรมพอร์ตอินพุท เอาท์พุทได้อย่างอิสระ สำหรับในที่นี้แล้ว มีอุปกรณ์ 3 ตัวที่จะต้องต่อกับพอร์ต อินพุท/เอาท์พุท คือ Dotmetric display Keyboard และ Control output

KEYBOARD 4 x 4

การต่อ Keyboard เข้ากับชิป 8255 ทำได้โดยการต่อ เข้ากับ PORT C ของชิป 8255(1) และ 8255(2) คือพอร์ต (02H) , (06H) ตามลำดับดังที่รูป 38

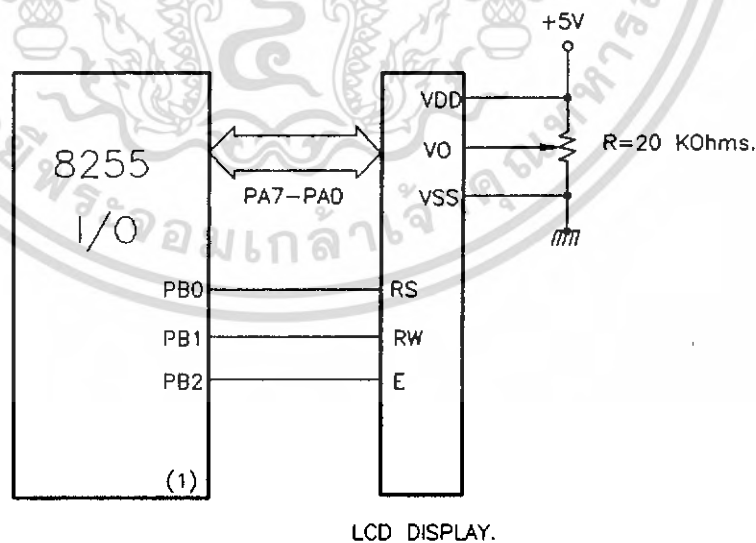
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 38 การต่อ Keyboard กับชิป 8255

จากรูป จะเห็นได้ว่าการต่อ Keyboard กับชิป 8255 ซึ่งไมโครโปรเซสเซอร์ จะมองพอร์ตของชิป 8255 เป็นพอร์ต 02H และ 06H จึงทำให้สามารถ ส่งข้อมูลออกไปทางพอร์ต 06H และอ่านเข้ามาทางพอร์ต 02H ทำให้สามารถรู้ว่า คีย์ ที่โดนกดอยู่นั้นเป็นคีย์อะไร

DOTMETRIC DISPLAY



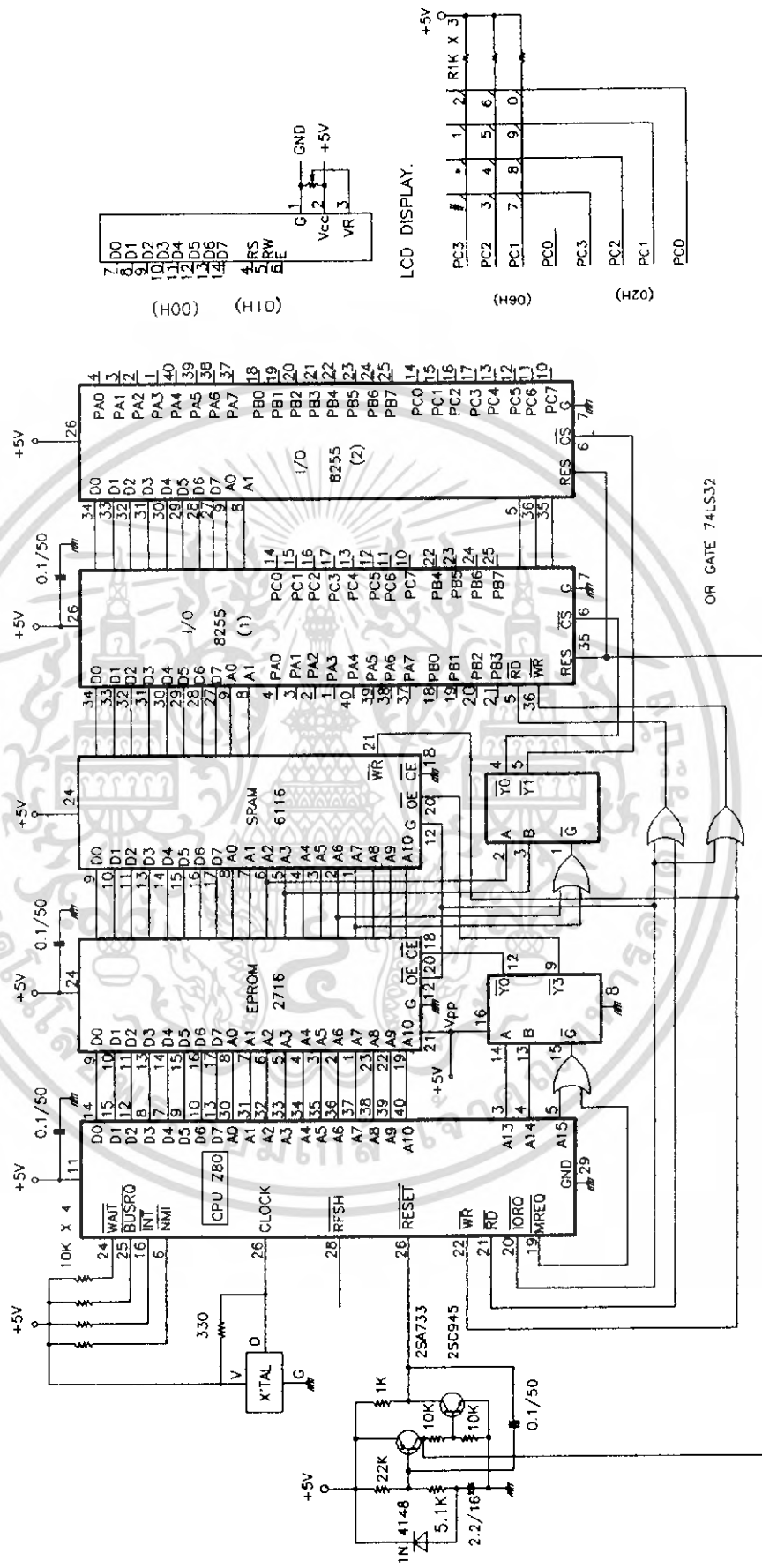
รูปที่ 39 การต่อ LCD display กับชิป 8255

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเชื่อมต่อ Dotmetric LCD display กับชิป 8255 กระทำได้โดยการต่อพอร์ตข้อมูล (D0-D7) ของ LCD Display เข้ากับพอร์ต A (PA₀-PA₇) และต่อ Control ของ LCD Display เข้ากับพอร์ต B ซึ่งใช้แค่ PB₀-PB₂ เพราะ Control ของ LCD Display มีเพียง 3 เส้น คือ RS,RW,E การควบคุม LCD Display มีความสำคัญตรงที่ต้องเข้าใจตัวชิป CONTROLLER ของ LCD Display ในที่นี้ใช้ชิป เบอร์ HD 44780 ซึ่งรายละเอียดคำสั่งควบคุม อยู่ภายในผนวกการต่อ LCD Display กับชิป 8255 ดังรูปที่ 39

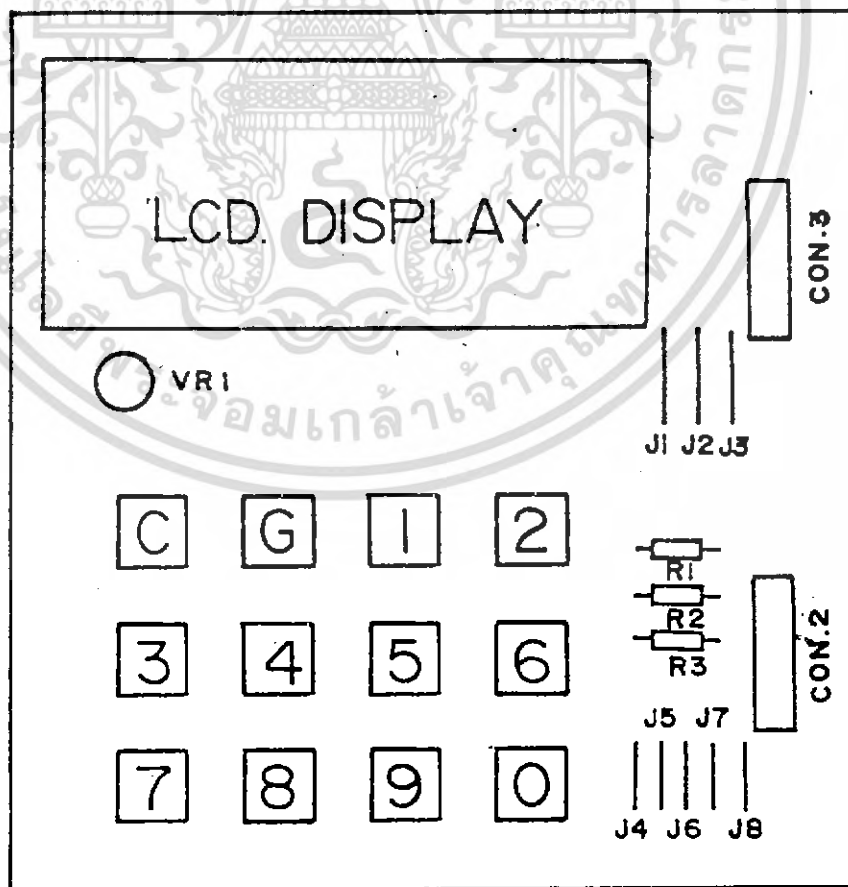
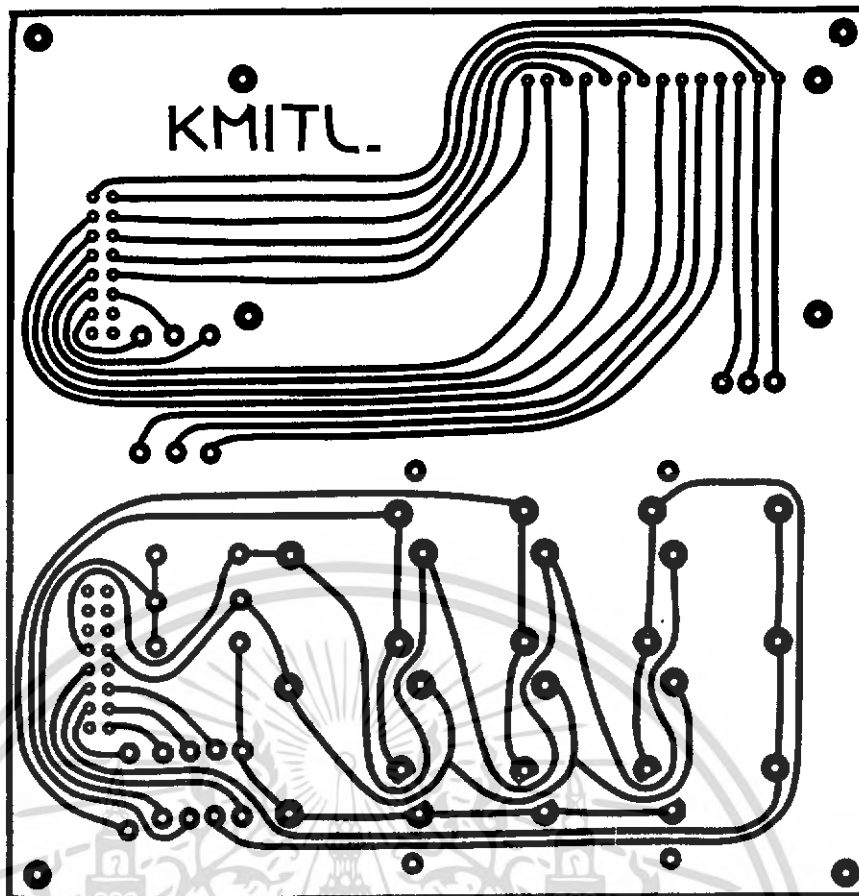
การต่อ LCD Display โดยการต่อเข้าทางพอร์ต A และ B ของชิป 8255 และใช้คำสั่ง OUT ข้อมูลไปยัง LCD Display โดยที่ข้อมูลได้จาก CHARECTER FONT TABLE ในภาคผนวก จำได้อีกขระตามที่ต้องการ



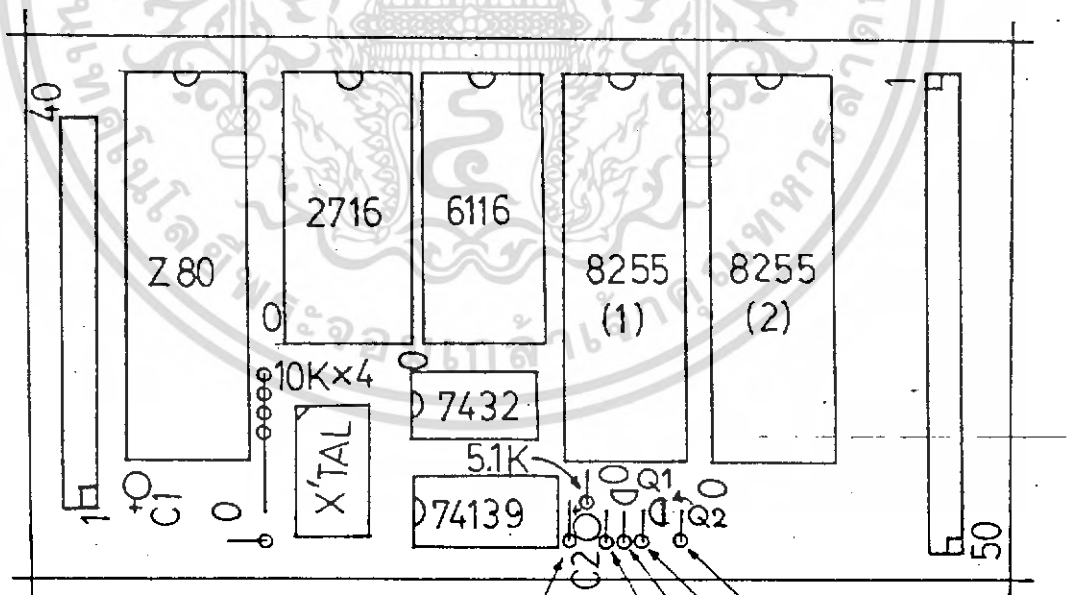
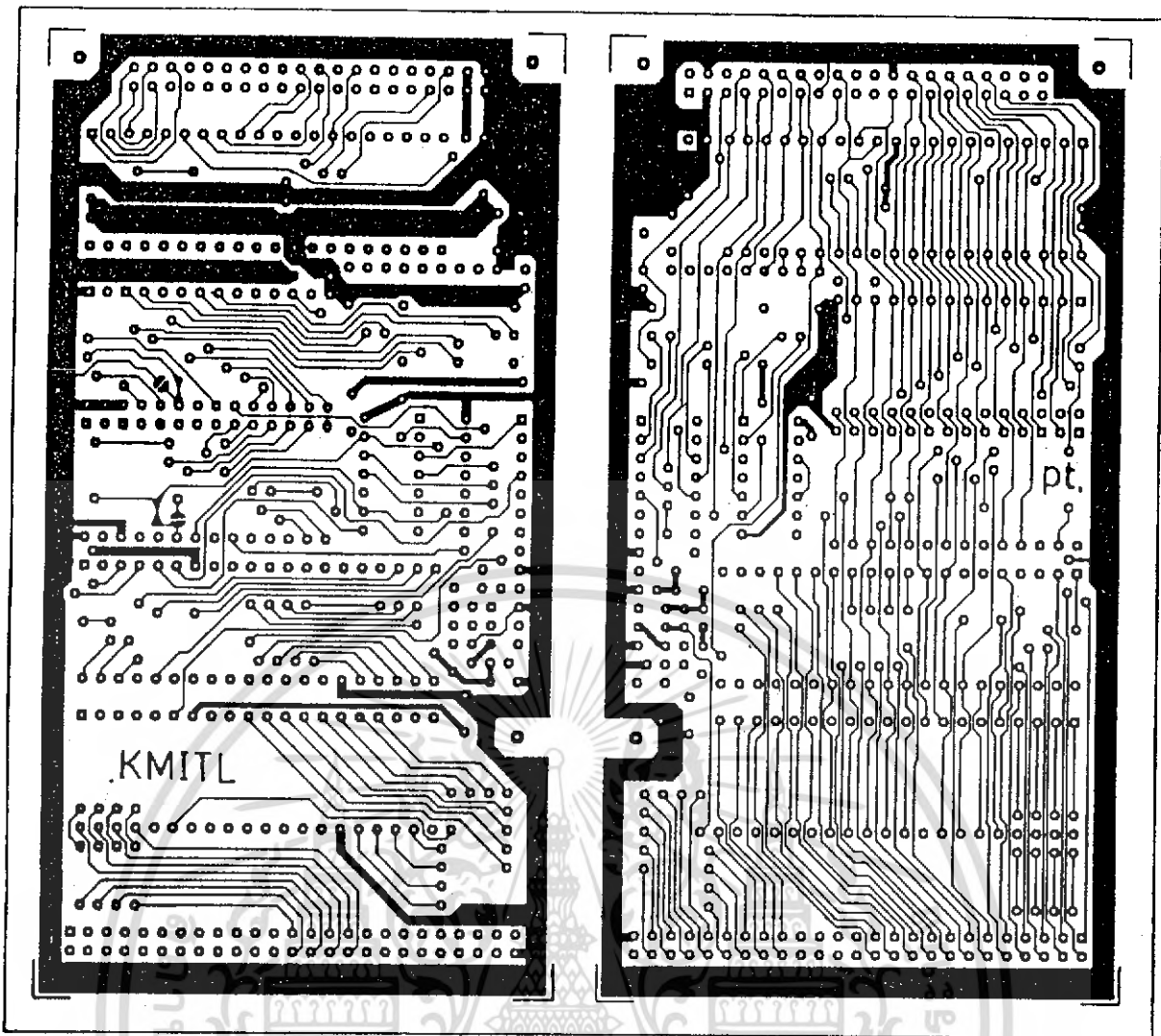


รูปที่ 40 วงจรสมบูรณต์ยรทต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



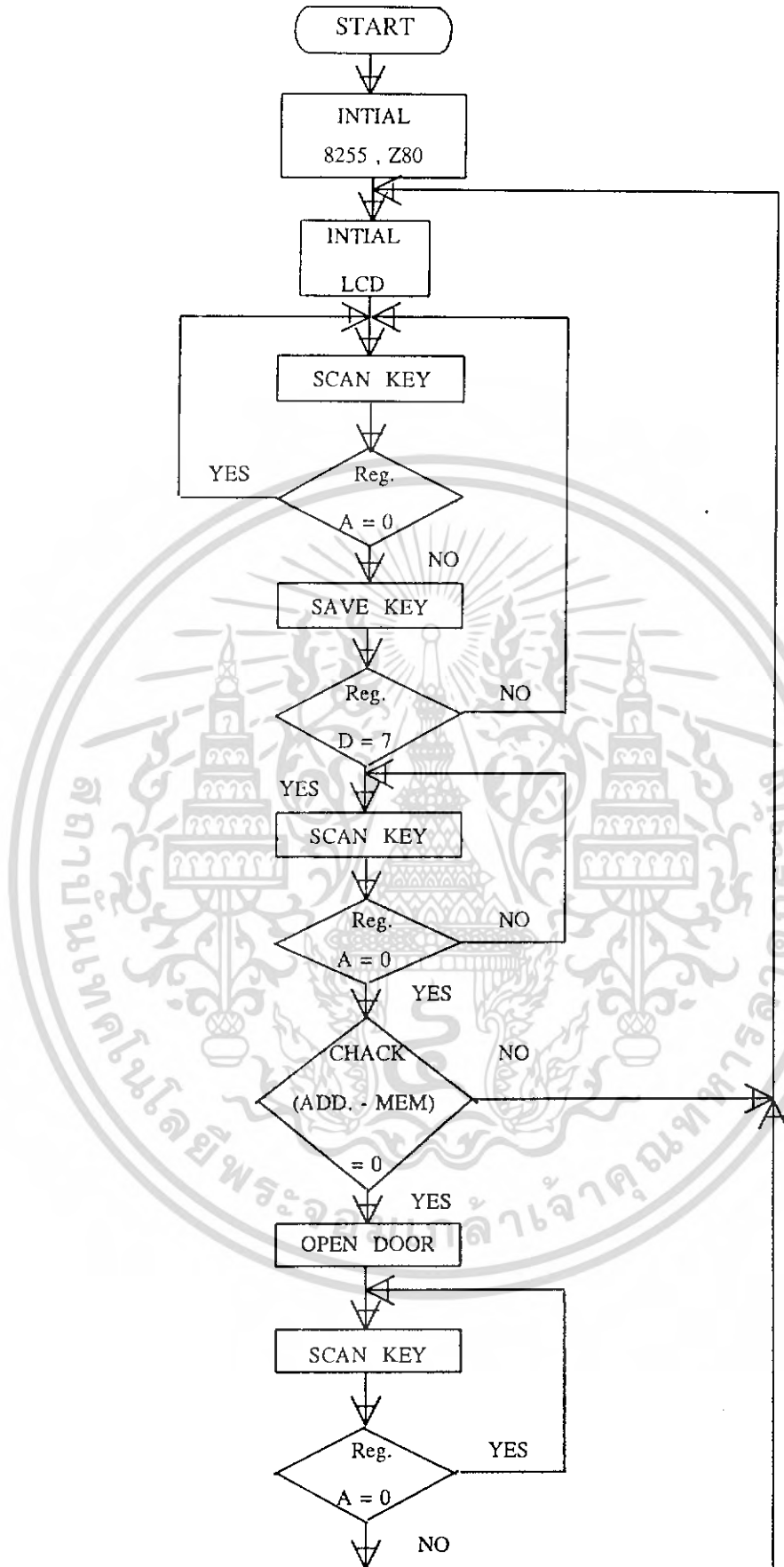
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อวัตถุประสงค์ที่สนับสนุนและส่งเสริมการลงอุปกรณ
 รูปที่ 41 ปลายปรีนวงจรคีย์รหัสและการลงอุปกรณ
 ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



- Q1 2SA733
- Q2 2SA945
- C1 10/16
- C2 2.2/16

- 1K
- 10K
- 10K
- 22K
- 1N4148

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
 รูปที่ 42 ลายปรินวงจรถ่ายและ การลงอุปกรณ์
 ไม่ว่าจะกรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 43 PLOW CHART แสดงการทำงานของเครื่อง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วิธีการใช้งาน

1. เริ่มต้น LCD DISPLAY แสดง

PASSWORD: _

2. ผู้ใช้กด คีย์ รหัส 7 หลัก

PASSWORD: 3313121

3. ผู้ใช้กด คีย์ ใด ๆ เป็นคีย์ ENTER เพื่อให้เครื่องประมวลผล
4. ถ้าผู้ใช้กด รหัส ผิด เครื่องจะให้ผู้ใช้ ไปทำในข้อ 2. ใหม่
5. ถ้าผู้ใช้กด รหัส ถูก ข้อความที่ปรากฏบน LCD DISPLAY

PASSWORD: <pass> _

พร้อม กับ โซลินอย์ ทำงาน เปิดประตู

6. เมื่อผู้ใช้ต้องการปิดประตู เพียงกดคีย์ใด ๆ โซลินอย์ก็จะหยุดทำงานปิดประตู

KEY 180

```

1          ;*****
2          ; SOFTWARE FOR KEY CODE  *
3          ;   by TH.Phornchai  *
4          ;*****
5          6000  K_PRESS:  EQU  6000H
6          ;*****
7          ; MAIN  PROGRAME  *
8          ;*****
9          0000  START:   ORG  0000H
10 0000  AF      XOR  A
11 0001  31 67FF  LD   SP,67FFH
12 0004  3E 80   LD   A,80H
13 0006  D3 03   OUT  (03H),A
14 0008  3E 89   LD   A,89H
15 000A  D3 07   OUT  (07H),A
16 000C  3E 00   LD   A,00H
17 000E  D3 04   OUT  (04H),A
18 0010  CD 0000# MAIN:  CALL  INTIAL
19 0013  16 00   LD   D,0
20 0015  1E 00   LD   E,0
21 0017  CD 0000# M1:   CALL  SCANK
22 001A  78     LD   A,B
23 001B  14     INC  D
24 001C  CD 0000#     CALL  K_MEM
25 001F  CD 0000#     CALL  LCD_DISP

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

26 0022 7A          LD   A,D
27 0023 FE 07      CP   07
2928 0025 20 F0     JR   NZ,M1
2929 0027 CD 0000#  CALL  CHACK
2930 002A 18 E4     JR   MAIN
2931                ;*****
2932                ;   SCAN KEY   *
2933                ;*****
2934 002C CD 0000#  SCANK:  CALL  SCAN
2935 002F FE FF      CP   OFFH
2936 0031 28 F9     JR   Z,SCANK
2937 0033 CD 0000#  CALL  RELEASE
2938 0036 38 F4     JR   C,SCANK
2939 0038 CD 0000#  CALL  K_CODE
2940 003B C9          RET
2941                ;
2942                ;/* scan key */
2943 003C 06 04      SCAN:  LD   B,4
2944 003E 0E FE     LD   C,OFH
2945 0040 79        SCAN_1: LD   A,C
7646 0041 D3 02    OUT  (02H),A
7647 0043 DB 06    IN   A,(06H)
7648 0045 E6 0F    AND  0FH
7649 0047 FE 0F    CP   0FH
7650 0049 20 ??    JR   NZ,SCAN_2
7651 004B CB 01    RLC  C
7652 004D 10 F1    DJNZ SCAN_1
7653 004F 21 6000  LD   HL,K_PRESS

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

7654 0052 CB 86 RES 0,(HL)
7655 0054 3E FF LD A,OFFH
43356 0056 C9 SCAN_2: RET
43357 ;
43358 ;/* chack key release */
43359 0057 21 6000 RELEASE: LD HL,K_PRESS
43360 005A CB 46 BIT 0,(HL)
43361 005C 37 SCF
43362 005D C0 RET NZ
43363 005E CB C6 SET 0,(HL)
43364 0060 3F CCF
43365 0061 C9 RET
43366 0062 CB 21 K_CODE: SLA C
43367 0064 CB 21 SLA C
70268 0066 CB 21 SLA C
70269 0068 CB 21 SLA C
70270 006A B1 OR C
70271 006B 21 0000# LD HL,KEYTAB
70272 006E 06 00 LD B,0
70273 0070 04 K_CODE1: INC B
70274 0071 BE CP (HL)
70275 0072 23 INC HL
70276 0073 20 FB JR NZ,K_CODE1
70277 0075 78 LD A,B
70278 0076 C9 RET
70279 ;
70280 ; /* intial lcd modul */
70281 0077 CD 0000# INTIAL: CALL DELAY

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

70282 007A CD 0000#      CALL  DELAY
70283 007D CD 0000#      CALL  DELAY
70284 0080 3E 00         LD    A,00H
70285 0082 D3 00         OUT   (00H),A
70286 0084 D3 01         OUT   (01H),A ; RS=0,RW=0
70287 0086 3E 38         LD    A,38H
70288 0088 D3 00         OUT   (00H),A
70289 008A CD 0000#      CALL  CLOCK1
70290 008D CD 0000#      CALL  DELAY
70291 0090 3E 0F         LD    A,0FH ; D=1,C=1,B=1
70292 0092 D3 00         OUT   (00H),A
70293 0094 CD 0000#      CALL  CLOCK1
70294 0097 CD 0000#      CALL  DELAY
70295 009A 3E 06         LD    A,06H
70296 009C D3 00         OUT   (00H),A ; I/D=1,S=0
70297 009E CD 0000#      CALL  CLOCK1
70298 00A1 CD 0000#      CALL  DELAY
70299 00A4 3E 01         LD    A,01H
70300 00A6 D3 00         OUT   (00H),A ; cursor home
70301 00A8 CD 0000#      CALL  CLOCK1
70302 00AB CD 0000#      CALL  DELAY
70303 00AE 3E 01         LD    A,01H
70304 00B0 D3 01         OUT   (01H),A ; RS=1,RW=0
70305 00B2 3E 50         LD    A,50H
70306 00B4 D3 00         OUT   (00H),A ; "P"
70307 00B6 CD 0000#      CALL  CLOCK2
70308 00B9 3E 41         LD    A,41H
70309 00BB D3 00         OUT   (00H),A ; "A"

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

70310 00BD CD 0000# CALL CLOCK2
70311 00C0 3E 53 LD A,53H
70312 00C2 D3 00 OUT (00H),A ; "S"
70313 00C4 CD 0000# CALL CLOCK2
70314 00C7 3E 53 LD A,53H
70315 00C9 D3 00 OUT (00H),A ; "S"
70316 00CB CD 0000# CALL CLOCK2
70317 00CE 3E 57 LD A,57H
70318 00D0 D3 00 OUT (00H),A ; "W"
70319 00D2 CD 0000# CALL CLOCK2
70320 00D5 3E 4F LD A,4FH
70321 00D7 D3 00 OUT (00H),A ; "O"
70322 00D9 CD 0000# CALL CLOCK2
70323 00DC 3E 52 LD A,52H
70324 00DE D3 00 OUT (00H),A ; "R"
70325 00E0 CD 0000# CALL CLOCK2
70326 00E3 3E 44 LD A,44H
70327 00E5 D3 00 OUT (00H),A ; "D"
70328 00E7 CD 0000# CALL CLOCK2
70329 00EA 3E C0 LD A,0C0H
70330 00EC D3 00 OUT (00H),A
70331 00EE CD 0000# CALL CLOCK1
70332 00F1 3E 3A LD A,3AH
70333 00F3 D3 00 OUT (00H),A ; ":"
70334 00F5 CD 0000# CALL CLOCK2
70335 00F8 C9 RET
70336 ;
70337 ;/* pulse for enable */

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

70338 00F9 3E 00      CLOCK1:  LD    A,00H
70339 00FB D3 01              OUT    (01H),A
70340 00FD CD 0000#          CALL   DELAY
70341 0100 3E 04              LD     A,04H    ; clock
70342 0102 D3 01              OUT    (01H),A
70343 0104 CD 0000#          CALL   DELAY
70344 0107 3E 00              LD     A,00H
70345 0109 D3 01              OUT    (01H),A
70346 010B C9                      RET
70347 010C 3E 01      CLOCK2:  LD     A,01H
70348 010E D3 01              OUT    (01H),A
70349 0110 CD 0000#          CALL   DELAY
70350 0113 3E 05              LD     A,05H    ; clock
70351 0115 D3 01              OUT    (01H),A
70352 0117 CD 0000#          CALL   DELAY
70353 011A 3E 01              LD     A,01H
70354 011C D3 01              OUT    (01H),A
70355 011E C9                      RET
70356                               ;
70357                               ;/* delay time */
70358 011F 26 6F      DELAY:  LD     H,06FH    ; delay
70359 0121 2E FF      DE2:    LD     L,0FFH
70360 0123 2D          DE1:    DEC    L
70361 0124 20 FD              JR     NZ,DE1
70362 0126 25              DEC    H
70363 0127 20 F8              JR     NZ,DE2
70364 0129 C9                      RET
70365                               ;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

70366                ;/* save key */
70367 012A 1C        K_MEM:    INC    E
70368 012B 7B                LD    A,E
70369 012C FE 01                CP    1
70370 012E CC 0000#           CALL  Z,K1
70371 0131 7B                LD    A,E
70372 0132 FE 02                CP    2
70373 0134 CC 0000#           CALL  Z,K2
70374 0137 7B                LD    A,E
70375 0138 FE 03                CP    3
70376 013A CC 0000#           CALL  Z,K3
70377 013D 7B                LD    A,E
70378 013E FE 04                CP    4
70379 0140 CC 0000#           CALL  Z,K4
70380 0143 7B                LD    A,E
70381 0144 FE 05                CP    5
78682 0146 CC 0000#           CALL  Z,K5
78683 0149 7B                LD    A,E
78684 014A FE 06                CP    6
78685 014C CC 0000#           CALL  Z,K6
78686 014F 7B                LD    A,E
78687 0150 FE 07                CP    7
3=5>8 0152 CC 0000#           CALL  Z,K7
98549 0155 C9                RET
98550 0156 78                K1:    LD    A,B
98551 0157 32 6600           LD    (6600H),A
98552 015A C9                RET
98553 015B 78                K2:    LD    A,B

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

4054 015C 32 6601          LD      (6601H),A
4055 015F C9              RET
4056 0160 78              K3:    LD      A,B
4057 0161 32 6602          LD      (6602H),A
4058 0164 C9              RET
4059 0165 78              K4:    LD      A,B
4060 0166 32 6603          LD      (6603H),A
4061 0169 C9              RET
4062 016A 78              K5:    LD      A,B
4063 016B 32 6604          LD      (6604H),A
4064 016E C9              RET
4065 016F 78              K6:    LD      A,B
4066 0170 32 6605          LD      (6605H),A
4067 0173 C9              RET
4068 0174 78              K7:    LD      A,B
4069 0175 32 6606          LD      (6606H),A
4070 0178 C9              RET
4071                      ;
4072                      ;/* lcd display */
4073 0179 78              LCD_DISP: LD      A,B
4074 017A FE 01            CP      1
4075 017C CC 0000#         CALL    Z,D1
4076 017F 78              LD      A,B
4077 0180 FE 02            CP      2
4078 0182 CC 0000#         CALL    Z,D2
4079 0185 78              LD      A,B
4080 0186 FE 03            CP      3
4081 0188 CC 0000#         CALL    Z,D3

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4082 018B 78	LD	A,B
4083 018C FE 04	CP	4
4084 018E CC 0000#	CALL	Z,D4
4085 0191 78	LD	A,B
4086 0192 FE 05	CP	5
4087 0194 CC 0000#	CALL	Z,D5
4088 0197 78	LD	A,B
4089 0198 FE 06	CP	6
4090 019A CC 0000#	CALL	Z,D6
4091 019D 78	LD	A,B
493?2 019E FE 07	CP	7
24753 01A0 CC 0000#	CALL	Z,D7
24754 01A3 78	LD	A,B
24755 01A4 FE 08	CP	8
24756 01A6 CC 0000#	CALL	Z,D8
24757 01A9 78	LD	A,B
24758 01AA FE 09	CP	9
24759 01AC CC 0000#	CALL	Z,D9
24760 01AF 78	LD	A,B
24761 01B0 FE 0A	CP	10
24762 01B2 CC 0000#	CALL	Z,D10
24763 01B5 78	LD	A,B
24764 01B6 FE 0B	CP	11
24765 01B8 CC 0000#	CALL	Z,D11
24766 01BB 78	LD	A,B
24767 01BC FE 0C	CP	12
24768 01BE CC 0000#	CALL	Z,D12
24769 01C1 C9	RET	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

24770 01C2 3E 23      D1:      LD      A,23H      ; #
24771 01C4 D3 00              OUT     (00H),A
24772 01C6 CD 010C      CALL    CLOCK2
24773 01C9 C9              RET
24774 01CA 3E 2A      D2:      LD      A,2AH      ; *
24775 01CC D3 00              OUT     (00H),A
24776 01CE CD 010C      CALL    CLOCK2
24777 01D1 C9              RET
24778 01D2 3E 31      D3:      LD      A,31H      ; 1
24779 01D4 D3 00              OUT     (00H),A
24780 01D6 CD 010C      CALL    CLOCK2
289>1 01D9 C9              RET
53242 01DA 3E 32      D4:      LD      A,32H      ; 2
53243 01DC D3 00              OUT     (00H),A
53244 01DE CD 010C      CALL    CLOCK2
53245 01E1 C9              RET
53246 01E2 3E 33      D5:      LD      A,33H      ; 3
53247 01E4 D3 00              OUT     (00H),A
53248 01E6 CD 010C      CALL    CLOCK2
53249 01E9 C9              RET
53250 01EA 3E 34      D6:      LD      A,34H      ; 4
53251 01EC D3 00              OUT     (00H),A
53252 01EE CD 010C      CALL    CLOCK2
53253 01F1 C9              RET
53254 01F2 3E 35      D7:      LD      A,35H      ; 5
53255 01F4 D3 00              OUT     (00H),A
53256 01F6 CD 010C      CALL    CLOCK2
53257 01F9 C9              RET

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

53258	01FA	3E 36	D8:	LD	A,36H	; 6
53259	01FC	D3 00		OUT	(00H),A	
53260	01FE	CD 010C		CALL	CLOCK2	
53261	0201	C9		RET		
53262	0202	3E 37	D9:	LD	A,37H	; 7
53263	0204	D3 00		OUT	(00H),A	
53264	0206	CD 010C		CALL	CLOCK2	
53265	0209	C9		RET		
53266	020A	3E 38	D10:	LD	A,38H	; 8
53267	020C	D3 00		OUT	(00H),A	
53268	020E	CD 010C		CALL	CLOCK2	
53269	0211	C9		RET		
53270	0212	3E 39	D11:	LD	A,39H	; 9
53271	0214	D3 00		OUT	(00H),A	
53272	0216	CD 010C		CALL	CLOCK2	
53273	0219	C9		RET		
53274	021A	3E 30	D12:	LD	A,30H	; 0
53275	021C	D3 00		OUT	(00H),A	
53276	021E	CD 010C		CALL	CLOCK2	
53277	0221	C9		RET		
53278	0222	CD 003C	CHACK:	CALL	SCAN	
53279	0225	FE FF		CP	OFFH	
53280	0227	28 F9		JR	Z,CHACK	
53281	0229	CD 0057		CALL	RELEASE	
53282	022C	38 F4		JR	C,CHACK	
53283	022E	3A 6600		LD	A,(6600H)	
53284	0231	FE 05		CP	5	
53285	0233	20 ??		JR	NZ,JOP	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

53286	0235	3A	6601	LD	A,(6601H)	
53287	0238	FE	05	CP	5	
53288	023A	20	??	JR	NZ,JOP	
53289	023C	3A	6602	LD	A,(6602H)	
53290	023F	FE	03	CP	3	
53291	0241	20	??	JR	NZ,JOP	
53292	0243	3A	6603	LD	A,(6603H)	
53293	0246	FE	05	CP	5	
53294	0248	20	??	JR	NZ,JOP	
53295	024A	3A	6604	LD	A,(6604H)	
53296	024D	FE	03	CP	3	
53297	024F	20	??	JR	NZ,JOP	
53298	0251	3A	6605	LD	A,(6605H)	
89279	0254	FE	04	CP	4	
25360	0256	20	??	JR	NZ,JOP	
25361	0258	3A	6606	LD	A,(6606H)	
25362	025B	FE	03	CP	3	
25363	025D	20	??	JR	NZ,JOP	
25364	025F	CD	0077	CALL	INTIAL	
25365	0262	3E	3C	LD	A,3CH	; <
25366	0264	D3	00	OUT	(00H),A	
25367	0266	CD	010C	CALL	CLOCK2	
25368	0269	3E	70	LD	A,70H	; p
25369	026B	D3	00	OUT	(00H),A	
25370	026D	CD	010C	CALL	CLOCK2	
25371	0270	3E	61	LD	A,61H	; a
25372	0272	D3	00	OUT	(00H),A	
25373	0274	CD	010C	CALL	CLOCK2	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

25374 0277 3E 73          LD    A,73H          ; s
25375 0279 D3 00          OUT   (00H),A
25376 027B CD 010C        CALL  CLOCK2
25377 027E 3E 73          LD    A,73H          ; s
25378 0280 D3 00          OUT   (00H),A
25379 0282 CD 010C        CALL  CLOCK2
25380 0285 3E 3E          LD    A,3EH          ; >
25381 0287 D3 00          OUT   (00H),A
25382 0289 CD 010C        CALL  CLOCK2
25383 028C 3E FF          LD    A,OFFH         ;key code correct
25384 028E D3 04          OUT   (04H),A
25385 0290 CD 003C        LOOP: CALL  SCAN
25386 0293 FE FF          CP    OFFH
25387 0295 28 F9          JR    Z,LOOP
25388 0297 3E 00          LD    A,00H
25389 0299 D3 04          OUT   (04H),A
25390 029B CD 0057        CALL  RELEASE
25391 029E 38 F0          JR    C,LOOP
25392 02A0 C9          JOP:  RET
25393
25394                      ;/* table key bord */
25395 02A1 77 B7 D7 E7 KEYTAB: DB 077H,0B7H,0D7H,0E7H ;key #,* ,1,2
25396 02A5 7B BB DB EB          DB 07BH,0BBH,0DBH,0EBH ; 3,4,5,6
25397 02A9 7D BD DD ED          DB 07DH,0BDH,0DDH,0EDH ; 7,8,9,0

```

0 Error(s) Detected.

685 Absolute Bytes. 42 Symbols Detected.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

ตารางคำสั่ง HD44780

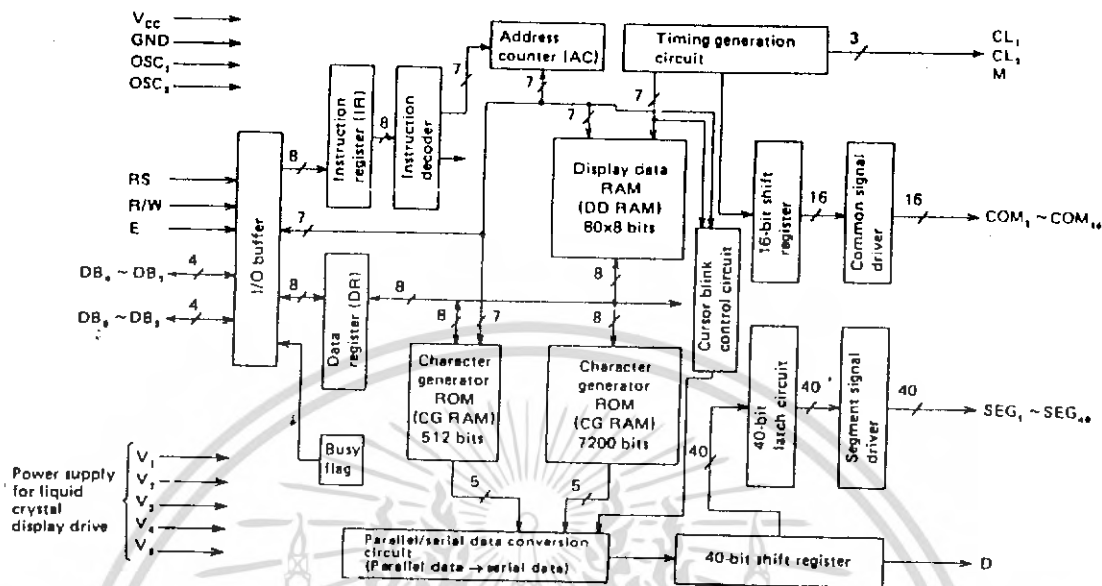
Instruction	Code										Description	Execution time (when fosc is 250 kHz) Note 1	Execution time (when fosc is 160 kHz) Note 2
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0			
Clear display	0	0	0	0	0	0	0	0	0	1	Clears all display and returns the cursor to the home position (Address 0).	82 μ s ~ 1.64 ms	120 μ s ~ 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 μ s ~ 1.6 ms	120 μ s ~ 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 μ s	120 μ s
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 μ s	120 μ s
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 μ s	120 μ s
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 μ s	120 μ s
Set CG RAM address	0	0	0	1	ACG						Sets the CG RAM address. CG RAM data is sent and received after this setting.	40 μ s	120 μ s
Set DD RAM address	0	0	1	ADD						Sets the DD RAM address. DD RAM data is sent and received after this setting.	40 μ s	120 μ s	
Read busy flag & address	0	1	BF	AC						Reads Busy flag (BF) indicating internal operation is being performed and reads address counter contents.	1 μ s	1 μ s	
Write data to CG or DD RAM	1	0	Write Data						Writes data into DD RAM or CG RAM.	40 μ s	120 μ s		
Read data to CG or DD RAM	1	1	Read Data						Reads data from DD RAM or CG RAM.	40 μ s	120 μ s		
	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 ACG: CG RAM address ADD: 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: $40 \mu\text{s} \times \frac{250}{270} = 37 \mu\text{s}$	

*No effect

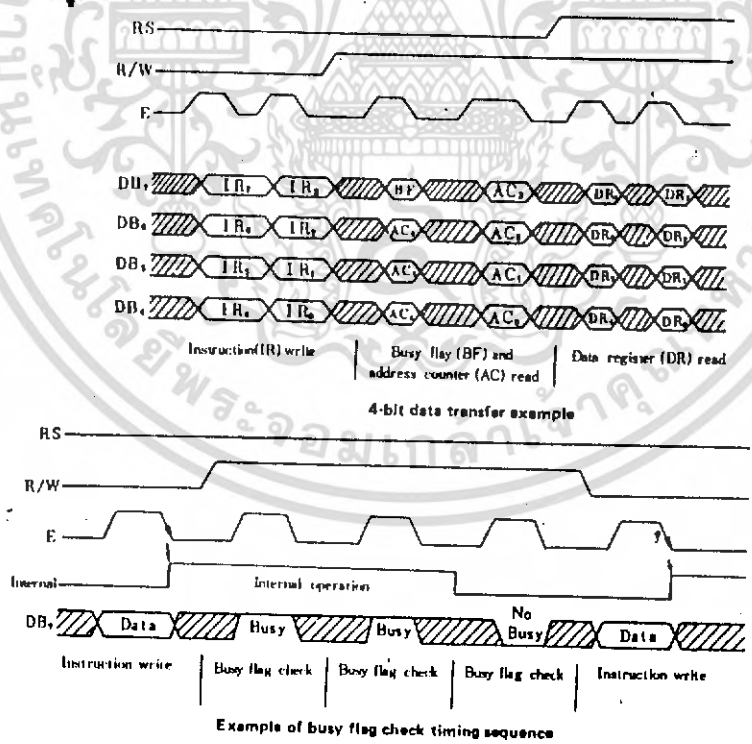
Notes 1. Applied to models driven by 1/8 duty or 1/11 duty.
 2. Applied to models driven by 1/16 duty.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
 ไม่วาระณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Block diagram of HD44780 interior



เท่านั้น โดยข้อมูลครึ่งแรกส่งนั้น HD44780 จะถือเป็นข้อมูล 4 BIT แรก และข้อมูลครึ่งต่อมาถือเป็นข้อมูล 4 BIT ล่าง



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้





ขาค่างๆในการต่อใช้งาน HD44780

1. RS (REGISTER SELECTION) จะเป็นเขาเลือก REGISTER ภายในซึ่งมีอยู่ 2 ตัวคือ INSTRUCTION REGISTER (IR) และ DATA REGISTER (DR) โดยถ้าเป็น 1 จะเป็นการเลือก DATA และถ้าเป็น 0 จะเป็นการเลือก INSTRUCTION

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

3. E (ENABLE SIGNAL) เป็นเขากำหนดสภาพการรับเขียนอ่านข้อมูล

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. DBO-DB7

เป็นขารับส่งข้อมูลจากตัว IC

5. VDD

ไฟเลี้ยงตัววงจร

6. VSS

เป็นขา GND

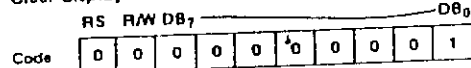
7. VO

เป็นขารับ VOLTAGE ในการขับ LCD ให้สว่างหรือมืด

รายละเอียดของคำสั่ง HD44780

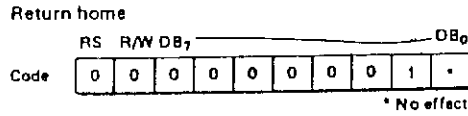
1. CLEAR DISPLAY

Clear display



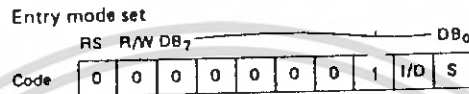
คำสั่งนี้จะเป็นการเขียนช่องว่างหรือ SPACE (ASCII 20H) เข้าไปใน DD RAM ทั้งหมดและทำการ SET DD RAM ADDRESSER เป็นศูนย์ ตัว CURSOR จะกลับไปอยู่ตำแหน่งบนสุดซ้ายมือของจอภาพ SET I/D = 1, S ไม่มีการเปลี่ยน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



คำสั่งนี้จะทำการ SET DD RAM ADDRESSER เป็นศูนย์ ตัว CURSOR จะ
 กลับไปอยู่ตำแหน่งบนสุดซ้ายมือของจอภาพข้อมูลในจอภาพไม่เปลี่ยน

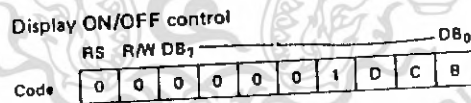
3. ENTRY MODE SET



BIT I/O : โดยจะเป็นตัวกำหนดให้ว่าเมื่อเขียนหรืออ่านข้อมูลแล้วจะทำให้
 DD RAM ADDRESS เพิ่มขึ้นหนึ่งหรือลดลงหนึ่ง โดย 1 = เพิ่ม
 0 = ลดลงหนึ่ง

BIT S : เป็นตัวกำหนดแสดงผลโดยถ้า S = 1 จะเป็นการใส่ข้อมูลแล้วตัว
 CURSOR อยู่ที่ข้อมูลจะถูกดันไปทางซ้าย ถ้า S = 0 ข้อมูลจะ
 อยู่ที่ตัว CURSOR จะถูกดันไปทางขวามือ

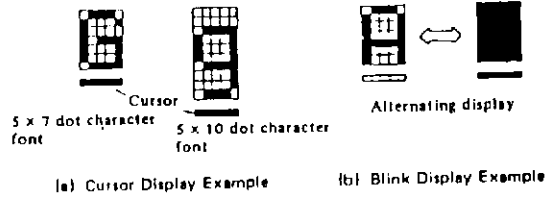
4. DISPLAY ON/OFF CONTROL



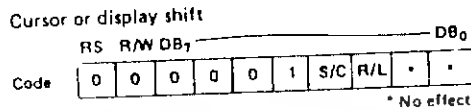
BIT D : เป็น BIT ให้เปิดปิดหน้าจอภาพโดยถ้า D = 1 จะ ON และ
 D = 0 จะ OFF

BIT C : จะให้แสดง CURSOR ให้ BIT C = 1 และถ้าไม่ต้องการ
 แสดง CURSOR BIT C = 0 โดยตัว CURSOR จะอยู่ที่ LINE
 ที่ 8 ในแบบ 5X7 DOT และจะอยู่ที่ LINE ที่ 11 ในแบบ
 5X10 DOT

BIT B : เป็น BIT SET การกระพริบของ CURSOR โดย B = 1
 มีการกระพริบ B = 0 ไม่มีการกระพริบ โดยมีระยะเวลาการ
 กระพริบประมาณ 379.2 ms



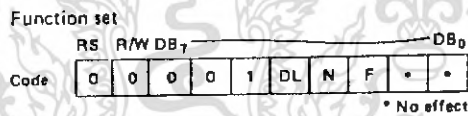
5. CURSOR OR DISPLAY SHIFT



เป็นคำสั่งกำหนดให้ตำแหน่ง CURSOR หรือท่อนุ้ลไปเกิดทางซ้ายหรือขวาโดยไม่ต้องใช้คำสั่งเขียนหรืออ่าน โดย

S/C	R/L	
0	0	ทำการย้าย CURSOR ไปจากตำแหน่งเดิมไปซ้ายมือ 1 ตำแหน่ง
0	1	ทำการย้าย CURSOR ไปจากตำแหน่งเดิมไปขวามือ 1 ตำแหน่ง
1	0	เป็นการค้นตัวอักษรที่เกิดไปทางซ้าย
1	1	เป็นการค้นตัวอักษรที่เกิดไปทางขวามือ

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

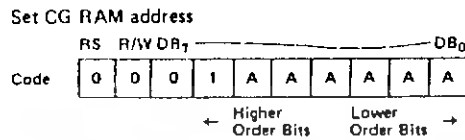
F : เป็นการ SET ขนาด DOT การแสดงผล 5X7 หรือ 5X10 โดย F = 0 เป็นแบบ 5X7 และ F = 1 เป็นแบบ 5X10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

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.

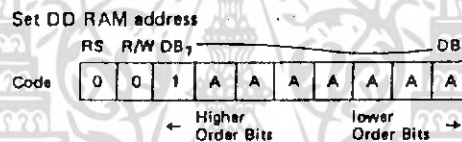
* No effect

7. SET CG RAM ADDRESS



ใน HD44780 นี้จะมีหน่วยความจำอยู่ 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

8. SET DD RAM ADDRESS



เป็นคำสั่ง SET คำ ADDRESS ใน DD RAM ในการเขียนหรืออ่านค่าจาก DD RAM (DD RAM คือ ส่วนที่จะแสดงผลหน้าจอ LCD) โดยจำนวน ADDRESS ที่จะเกิดขึ้นบนจอ LCD จะอยู่กับ SET คำ N ดังนี้

- 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

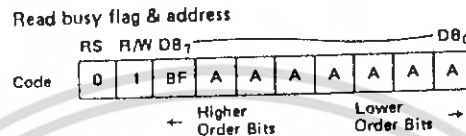
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

	1fr	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.

HDM-20216H

9. READ BUSY FLAG AND ADDRESS



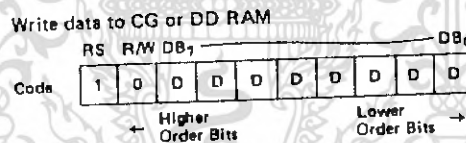
เป็นคำสั่งอ่านค่า BUSY FLAG ซึ่งจะเห็นได้ว่ามากกว่าตัว HD44780 อยู่ในหมวดการทำงานภายในอยู่หรืออยู่ในสถานะพร้อมจะรับข้อมูล โดย

BF = 1 อยู่ในหมวดการทำงานภายในไม่ว่าจะรับข้อมูลหรือคำสั่ง

BF = 0 พร้อมจะรับข้อมูลหรือคำสั่งได้

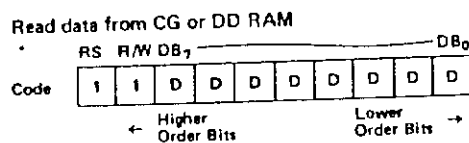
นอกจากนี้ยังเห็นคำสั่งอ่านค่าที่ข้อมูล ADDRESS ของ CG RAM หรือ DD RAM ด้วย

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



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Character Codes (DD RAM Data)								CG RAM Address				Character Patterns (CG RAM Data)							
7 6 5 4 3 2 1 0 ← Higher Lower →				5 4 3 2 1 0 ← Higher Lower →				7 6 5 4 3 2 1 0 ← Higher Lower →											
0 0 0 0 x 0 0 0								0 0 0				x x x				0			
								0 0 1				↑				0 0 - 0			
								0 1 0				0 0 0				0 0 0			
								0 1 1				0 0 0				0 0 0			
								1 0 0				0 0 0				0 0 0			
								1 0 1				0 0 0				0 0 0			
0 u 0 0 x 0 0 1								1 1 0				x x x				0 0 0 0 0			
								1 1 1				↑				0 0 0			
								0 0 0				0 0 0				0 0 0			
								0 0 1				0 0 0				0 0 0			
								1 0 0				0 0 0				0 0 0			
								1 0 1				0 0 0				0 0 0			
0 0 0 0 x 1 1 1								1 1 0				x x x				0 0 0 0 0			
								1 1 1				↑				0 0 0			
								0 0 0				0 0 0				0 0 0			
								0 0 1				0 0 0				0 0 0			
								1 0 0				0 0 0				0 0 0			
								1 0 1				0 0 0				0 0 0			

For 5 x 7 dot character pattern

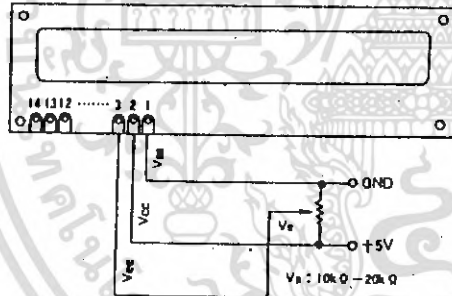
Character Pattern Example (1)

Character Pattern Example (2)

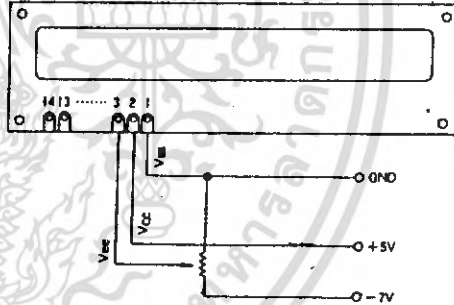
* No effect

สรุป การใช้งาน LCD MODULE นั้นที่สำคัญคือ ต้องเข้าใจในตัว CONTROLLER ของ LCD MODULE นั้น โดย CONTROLLER ทุกบริษัทจะการทำงานที่เหมือนกันเป็นส่วนใหญ่

DMC Module



In case of extended temperature version



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

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

CHARACTER FONT TABLE

Higher Lower 4bit 4bit	0000	0010	0011	0100	0101	0110	0111	1010	1011	1100	1101	1110	1111
XXXX0000	CG RAM (1)		๒	๓	๔	๕	๖	๗	๘	๙	๐	๑	๒
XXXX0001	(2)	๓	๔	๕	๖	๗	๘	๙	๐	๑	๒	๓	๔
XXXX0010	(3)	๔	๕	๖	๗	๘	๙	๐	๑	๒	๓	๔	๕
XXXX0011	(4)	๕	๖	๗	๘	๙	๐	๑	๒	๓	๔	๕	๖
XXXX0100	(5)	๖	๗	๘	๙	๐	๑	๒	๓	๔	๕	๖	๗
XXXX0101	(6)	๗	๘	๙	๐	๑	๒	๓	๔	๕	๖	๗	๘
XXXX0110	(7)	๘	๙	๐	๑	๒	๓	๔	๕	๖	๗	๘	๙
XXXX0111	(8)	๙	๐	๑	๒	๓	๔	๕	๖	๗	๘	๙	๐
XXXX1000	(9)	๐	๑	๒	๓	๔	๕	๖	๗	๘	๙	๐	๑
XXXX1001	(10)	๑	๒	๓	๔	๕	๖	๗	๘	๙	๐	๑	๒
XXXX1010	(11)	๒	๓	๔	๕	๖	๗	๘	๙	๐	๑	๒	๓
XXXX1011	(12)	๓	๔	๕	๖	๗	๘	๙	๐	๑	๒	๓	๔
XXXX1100	(13)	๔	๕	๖	๗	๘	๙	๐	๑	๒	๓	๔	๕
XXXX1101	(14)	๕	๖	๗	๘	๙	๐	๑	๒	๓	๔	๕	๖
XXXX1110	(15)	๖	๗	๘	๙	๐	๑	๒	๓	๔	๕	๖	๗
XXXX1111	(16)	๗	๘	๙	๐	๑	๒	๓	๔	๕	๖	๗	๘

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

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดลอกเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

