สเตปปิ้งมอเตอร์ไดร์ฟ

STEPPING MOTOR DRIVE

โดย

นายพงเทพ ชุติวิทูรชัย

นายมงคล สุชาภิรมย์

นายวิชัย วัชรินทร์พร

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

ภาควิชาวิศวกรรมไฟฟ้า

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2534

ปีการศึกษา 2534

สเตปปิ้งมอเตอร์ไดร์ฟ
(STEPPING MOTOR DRIVE)

โดย

น่าย พงเทพ	ชุติวิทูรชัย	321466
นาย มงคล	สุขาภิรมย์	321471
นาย วิชัย	วัชรินทร์พร	321474

อาจารย์ที่ปรึกษา

รศ. ดร. วิริยะ พิเชฐจำเริญ

ปริญญานิพนธ์ ปีการศึกษา 2534

ภาควิชา ไฟฟ้ากำลัง

คณะ วิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

เรื่อง สเต็ปปิ้งมอเตอร์ไดร์ฟ (STEPPING MOTRO DRIVE)

ผู้จัดทำ

นาย พงเทพ ชุติวิทูรชัย 321466

นาย มงคล สุขาภิรมย์ 321471

นาย วิชัย วัชรินทร์พร 321474

อาจารย์ที่ปรึกษา

(รศ. ดร. วิริยะ พิเชฐจำเริญ)

สเตปปิ้งมอเตอร์ไดร์ฟ

พงเทพ ชุติวิทูรชัย 32.1466
มงคล สุขาภิรมย์ 32.1471
วิชัย วัชรินทร์พร 32.1474
รศ. ดร. วิริยะ พิเชฐจำเริญ
อ. ที่ปรึกษา ปีการศึกษา 2534

บทคัดย่อ

การนำสเตปปิ้งมอเตอร์ มาใช้งานในด้าน X-Y Plotter สำหรับควบคุมตำแหน่งของ ตัวเจาะ และ นำไปประยุกต์ ใช้งาน เป็น เครื่องเจาะรู ของ แผ่น วงจรพิมพ์ (Printed Circuit Board) นั้น ในโปรเจ็คนี้ ได้แยกเป็น 2 ส่วนคือ ส่วน ฮารดแวร์ (Hardware) และ ซอฟแวร์ (Software) ทางด้านฮารดแวร์ ได้แก่ส่วนที่เป็นชุดแมคคานิค ซึ่งได้แก่ ระบบเลื่อนตำแหน่ง โดยใช้ สเตปปิ้งมอเตอร์, วงจรขับ (Driver), บอร์ดเดียวไมโครคอมพิวเตอร์ (Single Board Microcomputer) และ ส่วนที่เป็นซอฟแวร์ (Software) ก็คือ ส่วนที่ เป็น โปรแกรม ที่ใช ้สำหรับ ควบคุม ให้สเตปปิ้งมอเตอร์ ควบคุมให้แกน ทั้งสองเคลื่อนที่ไปยังตำแหน่งที่เราต้องการ ให้เจาะรูบนแผ่นวงจรพิมพ์ ซึ่งจะเขียนด้วยโปรแกรม แอสแซมบลี (Assembly Language)

STEPPING MOTOR DRIVE

Pongtep Chutivitoolchai 32.1466
Mongkol Sukhapirom 32.1471
Wichai Watcharinporn 32.1474
Associate Professor
Wiriya Pichetchumroen Advs.1991

Abstract

Regarding to stepping motor application for X-Y Plotter to control a position of drilling unit and applied as PCB drilling machine, this project divides into 2 sections of hardware and software. Hardware section is position moving controlled by Stepping Motor, Mechanical Unit, Driver and Single Board Microcomputer. Software section is controller program which is performed to control the Stepping Motor to move both of a shaft to a position that we need to drill hole on Print Circuit Board. The application program is created by an Assembly Language.

สารบัญ

บทที่ 1 บทนำ หน้า

- โครงสร้างและการทำงานของสเตปปิ้งมอเตอร์ 1
- สเตปปิ้งมอเตอร์แบบแม่เหล็กถาวร 4
- สเตปปิ้งมอเตอร์แบบรีลัคแต้นซ์ 5
- สเตปปิ้งมอเตอร์แบบไฮบริด 18
- การเพิ่มรายละเอียดในแต่ละสเตปของมอเตอร์ 25

บทที่ 2 ฮาร์ดแวร์ของระบบ

- โครงสร้างของระบบแมคคานิค 31
- โครงสร้างของบอร์ดควบคุม 32
- ความรู้ทั่วไปเกี่ยวกับไมโครโปรเซสเซอร์ 35
- โครงสร้างและการทำงานของ IC PORT 8255 40
- วงจรไดรเวอร์และเซนเซอร์ 45
- บล็อคไดอะแกรม 48

บทที่ 3 ซอฟแวร์ของระบบ

- โฟลชาร์ต 56
- โปรแกรมภาษาแอสแซมบลี้ที่ใช้ควบคุม 59

สเตปปิ้งมอเตอร์ไดร์พ

(STEPPING MOTOR DRIVE)

บทที่ 1

บทนำ

ในปัจจุบันนี้จะเห็นว่า สเตปปิ้งมอเตอร์ มีใช้อยู่ในหลายๆ งาน เช่นในเครื่องพิมพ์, ใน X-Y Plotter ในแขนกล หรือในเครื่องถ่ายเอกสาร เพราะสเตปปิ้งมอเตอร์ มีข้อได้เปรียบมอเตอร์ไฟตรงแบบธรรมดา คือ สเตปปิ้งมอเตอร์มีการ ควบคุม แบบ Open Loop ทำให้ง่ายต่อการควบคุม และเรายังสามารถที่จะรู้ตำแหน่งของมอเตอร์ได้ตลอดเวลาอย่างแน่นอนแม่นยำ ดังนั้น งานที่ต้องการควบคุมตำแหน่งที่แน่ นอน จึงมักเลือก ใช้ สเตปปิ้งมอเตอร์

การควบคุม สเตปปิ้งมอเตอร์ ส่วนใหญ่ใช้ 2 วิธี คือ วิธีแรกใช้วงจรดิจิตอลควบคุม ส่วนวิธีที่สองใช้คอมพิวเตอร์ หรือ ซิงเกิ้ลบอร์ดควบคุม ซึ่งจะต้องทำชุด อินเตอร์เฟสระหว่างคอมพิวเตอร์กับ สเตปปิ้งมอเตอร์ และจะต้องมีซอฟต์แวร์เพื่อควบคุม สเตปปิ้งมอเตอร์ ให้ทำงาน ซึ่งในโครงงานนี้จะเลือกใช้วิธีที่สอง เพราะถึงแม้ว่า จะมีข้อยุ่งยากในการใช้งานมากกว่า แต่ก็มีข้อดีคือสามารถที่จะควบคุมตำแหน่งของ สเตปปิ้งมอเตอร์ ได้แน่นอนถูกต้องและมีความละเอียดดีมาก

โครงสร้างและการทำงานของ สเตปปิ้งมอเตอร์

สเตปปิ้งมอเตอร์ เป็นอุปกรณ์จำพวกเชิงกลทางไฟฟ้า ที่มีอินพุทเป็นกลุ่มของไบนารีโวลท์เตจ และเอาท์พุทเป็นลักษณะของการเคลื่อนที่แบบเชิงมุม หรือหมุนไปเป็นสเตป (แต่ละสเตปอยู่ในอยู่ในช่วง 0.1 ถึง 30 องศา ขึ้นอยู่กับโครงสร้างของสเตปปิ้งมอเตอร์) ตามสัญญาณพัลซ์ที่ป้อนให้กับขดสเตเตอร์ซึ่งจะเกิดแรงผลักต่อโรเตอร์หมุนไป แต่ลักษณะของ สเตปปิ้งมอเตอร์ จะมีขดของสเตเตอร์อยู่หลายขด ซึ่งเรียกว่า "เฟส" ฉะนั้นเมื่อป้อนสัญญาณที่เป็นพัลซ์ในลักษณะซีเควันของเลขไบนารีโดยผ่าน วงจรไดรเวอร์ (Driver) จะทำให้โรเตอร์หมุนได้อย่างต่อเนื่องดังบล๊อคไดอะแกรมรูปที่ 1

1. คุณประโยชน์ของสเตปปิ้งมอเตอร์

ในระบบควบคุมตำแหน่งที่ใช้สเต๊ปปิ้งมอเตอร์นั้นมีข้อดีอยู่หลายประการคือ

1. เป็นลักษณะการควบคุมแบบไม่ต้องการการป้อนกลับ ไม่ว่าจะเป็นการควบคุมตำแหน่งหรือความเร็ว

2. ความผิดพลาดเกี่ยวกับตำแหน่งแทบไม่มีเลย เนื่องจากการ เคลื่อนที่ของสเต็ปปิ้งมอเตอร์นั้นเคลื่อนที่เป็นสเตป ด้วยจำนวนองศาที่มีค่าแน่นอน

3. สเต็ปปิ้งมอเตอร์จะถูกนำมาใช้กับเครื่องมือ ที่ต้องการความละเอียดแม่นยำและใช้อยู่ใน เครื่องมือประเภทดิจิตอล เช่น เครื่องวาดรูป เครื่องคอมพิวเตอร์ นิวเมอริคอลคอนโทรล (Computer Numerical Control) หรือ CNC

4. ไม่จำเป็นต้องใช้วงจรแปลงดิจิตอลเป็นอนาลอกเมื่ออินเตอร์เฟสกับไมโครคอมพิวเตอร์

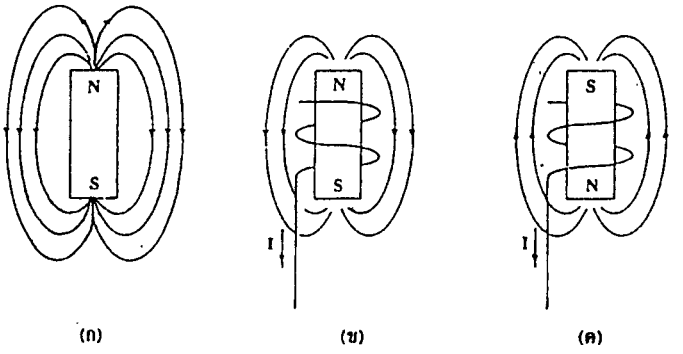
รูปที่ 1 บล๊อกไดอะแกรมแสดงการควบคุมสเต็ปปิ้งมอเตอร์

จากบล๊อกไดอะแกรม สเต็ปปิ้งมอเตอร์ จะทำงานเมื่อเราป้อน

- สัญญาณพัลซ์นาฬิกา (Clock Pulses)
- อินพุทสำหรับควบคุมทิศทางการหมุน

<u>หลักการทำงานของสเต็ปปิ้งมอเตอร์ทั่วๆไป</u>

ในรูปที่ 1.1 แสดงหลักการพื้นฐานของเส้นแรงแม่เหล็ก



รูปที่ 1.1 แสดงถึงสนามแม่เหล็กที่เกิดขึ้นในลักษณะต่างๆ

ในรูป 1.1 (ก) สนามแม่เหล็กที่เกิดขึ้นจากแม่เหล็กถาวร
1.1 (ข) สนามแม่เหล็กของแม่เหล็กไฟฟ้าที่เกิดจากกระแส I
1.1 (ค) ขั้วแม่เหล็กกลับทิศทางเมื่อขดลวดถูกพันกลับทิศทาง และทิศทางการไหลของกระแสไม่เปลี่ยนแปลง

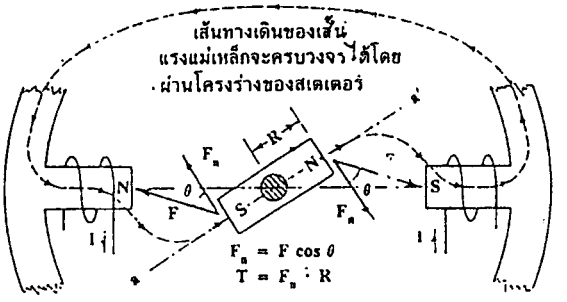
ในรูปที่ 1.2 แท่งแม่เหล็กถาวรติดอยู่กับเพลาและหมุนได้อิสระเหมือนอาร์มาเจอร์ มีขั้วแม่เหล็กไฟฟ้า 2 ขั้วซึ่งเป็นส่วนหนึ่งของโครงโลหะที่เป็นสเตเตอร์ (stator)

ในรูปที่ 1.2 ตำแหน่งแกนของอาร์มาเจอร์แม่เหล็กคือ a-a' ซึ่งต่างไปจากตำแหน่งแกนขั้วของแม่เหล็กไฟฟ้าเล็กน้อยเป็นมุม θ

แรงแม่เหล็กที่เกิดจากการดึงดูดของขั้วแม่เหล็กที่ต่างกัน ทำให้เกิดส่วนของแรงปกติ

$Fn = F \cos \theta$ (แรงนี้ตั้งฉากกับแกน a-a')

ท่อร์คผลรวม $T = Fn\ R$ (ทำให้อาร์มาเจอร์หมุนไปทิศทาง CW จนกว่าแกนของมาเจอร์ a-a' จะอยู่ในแนวเดียวกับแกนขั้วของสเตเตอร์)



รูปที่ 1.2 แสดงแรงดึงดูดทำให้เกิดทอร์คที่หมุนอาร์มาเจอร์ให้ไปอยู่ในตำแหน่งสมดุลย์

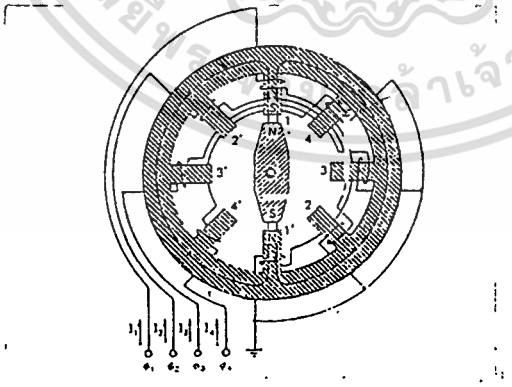
ถ้าหากมีคู่ขั้วแม่เหล็กไฟฟ้าหลายๆคู่ขั้วรอบๆ สเตเตอร์และถ้าหากขั้วเหล่านั้นถูกกระตุ้น ด้วยกระแสพัลซ์ในรูปแบบที่เรียงลำดับกัน ไปอาร์มาเจอร์ก็จะหมุนในรูปลักษณะของสเตปที่เป็นไปตามการหมุนของสนามแม่เหล็ก ที่เกิดจากการสวิทซ์ที่เรียงลำดับของขดลวดขั้วแม่เหล็กไฟฟ้าของสเตเตอร์

<u>สเตปมอเตอร์แบบแม่เหล็กถาวร</u>

โครงสร้างของสเตปมอเตอร์แบบแม่เหล็กถาวรแสดงได้ในรูปที่ 1.3

ในรูปที่ 1.3 เป็นสเตปมอเตอร์แบบ 4 เฟส แต่ละเฟสเป็นขดลวดอยู่บน 2 ขั้วของสเตเตอร์จะต้องมี 8 ขั้ว

โรเตอร์ทำจากแม่เหล็กถาวรและอยู่ในแนวของขั้วสเตเตอร์ 1 และ 1' มันหยุดอยู่ที่ ตำแหน่งนี้ได้ด้วยกระแส I1 ที่ไหลอยู่ในเฟส 1



รูปที่ 1.3 โครงสร้างของสเตปมอเตอร์แบบแม่เหล็กถาวรมี 4 เฟสและแต่ละเฟสพันด้วยขดลวดบน 2 ขั้วของสเตเตอร์มุมสเตปเท่ากับ 45°

ขดลวดของเฟส $\phi 1$, $\phi 4$, $\phi 3$ และ $\phi 2$ (1-4-3-2 ตามลำดับ) จะได้รับพลังงานด้วยกระแสพัลซ์ที่สอดคล้องกัน I_1, I_4, I_3 และ I_2 (กระแสแต่ละเฟสจะไหลในทิศทางที่แสดงในไดอะแกรม) แต่ละสเตปโรเตอร์จะหมุนไปตาม ทิศทาง ตามเข็มนาฬิกา 45° (360/8)

เมื่อขั้วเหนือของโรเตอร์ (แม่เหล็กถาวร) หมุนไปถึงขั้วของสเตเตอร์หมายเลข 2 ลำดับ การขับขดลวดเฟสของสเตปมอเตอร์คือ 1-4-3-2 จะต้องกระทำเหมือนเดิม (เพื่อให้มอเตอร์หมุนไปตามเข็มนาฬิกาอีก 180°) ยกเว้นเราต้องการให้หมุนกลับทิศทางใน 180° ที่เหลือ ด้วยการป้อนกระแสกลับทิศทางเพื่อให้เกิดการเหนี่ยวนำเป็นขั้วใต้ที่ขั้ว สเตเตอร์ 1', 4', 3' และ 2' ตามลำดับ (ทิศทางของกระแสแสดงในรูปที่ 1.3)

สเตปมอเตอร์แบบค่ารีลัคแต้นซ์แปรค่าได้ที่มีสแต็คเดียว

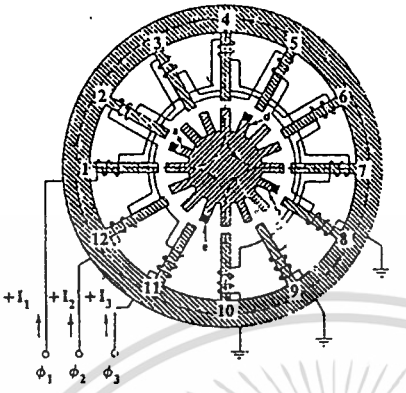
ตัวอย่างโครงสร้างของสเตปมอเตอร์ แบบคารีลัคแต้นซ์แปรค่าได้ที่มีสแต็คเดียวหรือที่เรียกสั้นว่า VR สเตปมอเตอร์ที่มีสแต็คเดียวแสดงได้ในรูปที่ 1.4

VR สเตปมอเตอร์ที่มีสแต็คเดียวจะมีโรเตอร์เดียวเมื่อเทียบกับ VR สเตปมอเตอร์แบบที่หลายสแต็คหมายถึงมีหลายโรเตอร์ โรเตอร์และสเตเตอร์ทำจากสารแม่เหล็ก

สเตปมอเตอร์ในรูปที่ 1.4 มี 3 เฟสแต่ละเฟสใช้ขวดลวดพันบน 4 ขั้ว หรือซี่ฟันของสเตเตอร์

ตัวอย่าง เฟสที่ 1 พันอยู่บนขั้วที่ 1,4,7 และ 10 ของสเตเตอร์ ดังนั้นสเตเตอร์จะมี 12 ซี่ฟัน และในที่นี้กำหนดให้มีโรเตอร์มี 16 ซี่ฟัน

ขั้วของสเตอร์ที่อยู่ตรงกันข้ามจะพันด้วยขดลวดลักษณะที่ต่างกัน เพื่อให้มีความสมดุลย์ ระหว่างเส้นแรงแม่เหล็กเข้าและออกจากโรเตอร์



รูปที่ 1.4 VR สเตปมอเตอร์แบบมีสแต็คเดียวและมีรายละเอียดโครงสร้างดังนี้

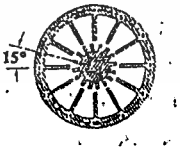
$$Nr = 16,\ Ns = 12,\ X = 4 \text{ โพล/เฟส},\ \theta s = 7.5^{\circ},\ Rs = 48 \text{ สเตป/รอบ}$$

สมมุติว่ากระแส I1 ป้อนให้กับเฟสที่ 1 ดังแสดงในรูปที่ 3.50 และโรเตอร์ทั้ง 4 ซี่ฟันจะอยู่ในแนวซี่ฟันที่ 1,4, 7 และ 10 ของสเตเตอร์ เส้นแรงแม่เหล็กจะเข้าสู่โรเตอร์จากสเตเตอร์ซี่ฟันที่ 4 และ 10 และออกจากโรเตอร์ไปยังซี่ฟันของสเตเตอร์ที่ 1 และ 7 ซึ่งเป็นทางเดินของเส้นแรงแม่เหล็กที่ครบวงจรโดยผ่านโครงร่างของสเตเตอร์ เราจะสังเกตได้ว่าปลายของซี่ฟันของสตเตอร์ที่ 4 จะถูกเหนี่ยวนำเป็นขั้วเหนือ (เนื่องจากเส้นแรงออกจากซี่ฟันที่ 4) และปลายของซี่ฟันโรเตอร์ซึ่งอยู่ในแนวเดียวกับซี่ฟันที่ 4 ของสเตเตอร์ จะเป็นเส้นทางผ่านเข้าไปยัง โรเตอร์ของเส้นแรงแม่เหล็ก และเหนี่ยวนำให้ปลายของซี่ฟันของโรเตอร์นั้นเป็นขั้วใต้ การทำให้เกิดลักษณะเป็นแม่เหล็กนี้ จะทำให้มีเส้นแรงแม่เหล็กอย่างต่อเนื่องผ่านช่องว่าง (gap) ระหว่างซี่ฟันทั้งสอง ที่อยู่ในแนวเดียวกันส่วนซี่ฟันของสเตเตอร์และโรเตอร์ที่เหลืออีก 3 คู่ก็เกิดลักษณะของแม่เหล็กในทำนองเดียวกัน

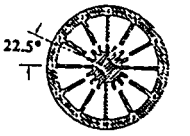
ในสภาวะต่อไปเราจะให้โรเตอร์หมุนไปหนึ่งสเตปในทิศทาง CW เราจะต้องจ่ายพลังงาน ให้กับเฟส 3 ที่มีขดลวดพันอยู่บนซี่ฟันที่ 2, 5, 8 และ 11 ของสเตเตอร์ด้วยกระแส I3 หลังจากหยุดจ่ายกระแส I1 แล้ว ในตอนนี้เส้นแรงแม่เหล็กจะหาทางเดินที่ต่าง ไปจากเดิมเพื่อทำให้วงจรแม่เหล็กครบวงจร (เหมือนกับกระแสในวงจรไฟฟ้าจะหาเส้นทางไหลในส่วนที่มีความต้านทานต่ำที่สุด) ในทำนองเดียวกันเส้นแรงแม่เหล็กในวงจรแม่เหล็กก็จะหาเส้นทางเดินที่มีค่าริลัคแต้นซ์ต่ำที่สุด (ช่องว่างอากาศระหว่างซี่ฟันจะทำให้เกิดค่ารีลัคแต้นซ์ต่อเส้นแรงแม่เหล็กช่องว่างกว้างมากค่ารีลัคแต้นซ์ก็จะมีค่ามาก) ด้วยเหตุผลดังกล่าวเส้นแรงแม่เหล็กจะออกจากขั้วที่ 2 และ 8 ของสเตเตอร์ซึ่งถูกเหนี่ยวนำให้เป็นขั้วเหนือ และเส้นแรงแม่เหล็กนี้ก็จะกระโดดผ่านช่องว่างไปยังซี่ฟัน ของโรเตอร์ที่ใกล้ที่สุด ซี่ฟัน a และ b ของโรเตอร์เป็นโรเตอร์ที่อยู่ใกล้ที่สุดและจะถูกเหนี่ยวนำให้เป็นขั้วใต้ เส้นแรงแม่เหล็กจะออกจากซี่ฟัน d และ e ของโรเตอร์ผ่านช่องว่างอากาศเข้าสู่ซี่ฟันที่ 5 และ 11 ของสเตเตอร์ ดังนั้นส่วนที่เหลือของวงจรแม่เหล็กจะสมบูรณ์โดยผ่านโครงร่างของสเตเตอร์ ในระหว่างเวลานั้นแรงของแม่เหล็กหรือแรงดึงดูดจะเกิดขึ้นระหว่างซี่ฟันที่ 2 ของสเตเตอร์ (ถูกเหนี่ยวนำเป็นขั้วเหนือ) และซี่ฟัน a ของโรเตอร์ (ถูกเหนี่ยวนำเป็นขั้วใต้) แรงดึงดูดจะเกิดขึ้นระหว่างคู่ขั้ว (11, e), (8, 6) และ (5, d) ด้วย ดังที่อธิบายในรูปที่ 1.2 ผลที่เกิดขึ้นนี้ จะทำให้เกิดทอร์คกระทำต่อโรเตอร์หมุนไปจนกระทั่งซี่ฟัน a, d, b และ e ของโรเตอร์อยู่ในแนวเดียวกับซี่ฟัน 2, 5, 8 และ 11 ของสเตเตอร์ตามลำดับ ขณะเวลาดังกล่าวข้างช่องว่างระหว่างซี่ฟันตามลำดับจะมีค่าน้อยที่สุด ผลลัพธ์ของค่ารีลัคแต้นซ์ จะมีค่าต่ำที่สุดและเส้นแรงแม่เหล็กจะมีค่าสูงสุดผ่านวงจรแม่เหล็ก ที่ตำแหน่งนี้เป็นตำแหน่งที่สมดุลย์ของการขับเฟส 3 ในกระบวนการที่กล่าวมาแล้วโรเตอร์จะเคลื่อนที่ในทิศทาง CW หนึ่งสเตปเป็นมุม 7.5 °

ลำดับการทำงานที่สมบูรณ์แสดงได้ในรูปที่ 1.5 เมื่อตำแหน่งเริ่มต้นของซี่ฟันของโรเตอร์ จะเป็นสีดำเพื่อให้เราทำความเข้าใจได้ชัดเจนถึงการหมุนของโรเตอร์ในทิศทาง CW เมื่อเฟสถูกขับในลักษณะเรียงลำดับ 1-3-2-1 ซี่ฟันของโรเตอร์ที่เป็นสีดำจะเคลื่อนที่ไป 3 สเตปคิดเป็นมุมได้เท่ากับ 22.5 ° เราจะขับเฟสในลักษณะเรียงลำดับเดิมซ้ำใหม่อีกเมื่อต้องการ

ให้โรเตอร์หมุนต่อเนื่องในทิศทาง CW แต่ถ้าเราต้องการให้โรเตอร์หมุนในทิศทาง CCW เราต้องกลับการเรียงลำดับเฟสเป็น 1-2-3-1

การเรียงลำดับเฟส	ตำแหน่งของโรเตอร์และเส้นแรงแม่เหล็ก
ตำแหน่งโรเตอร์เริ่มต้น : - เฟส ϕ 1 ได้รับพลังงาน - ซี่ฟันของโรเตอร์จะอยู่ในแนวซี่ฟัน ที่ 1,4,7,10 ของสเตเตอร์	
สเตปที่ 1 : เฟส ϕ 3 ได้รับพลังงาน - ซี่ฟันของโรเตอร์จะอยู่ในแนวซี่ฟัน ที่ 2,5,8,11 ของสเตเตอร์ - โรเตอร์จะเคลื่อนที่ไปในทิศทาง CW เป็นมุม 7.5° (1/3 ช่วงห่างระหว่าง ซี่ฟันของโรเตอร์)	
สเตปที่ 2 : เฟส ϕ 2 ได้รับพลังงาน - ซี่ฟันของโรเตอร์จะอยู่ในแนวซี่ฟัน ที่ 3,6,9,12 ของสเตเตอร์ - โรเตอร์จะเคลื่อนที่ไปในทิศทาง CW รวมเป็นมุม 7.5°	

สเตปที่ 3 : เฟส ϕ 1 ได้รับพลังงาน
- ซี่ฟันของโรเตอร์จะอยู่ในแนวซี่ฟัน
ที่ 1,4,7,10 ของสเตเตอร์
- โรเตอร์จะเคลื่อนที่ไปในทิศทาง CW
รวมเป็นมุม 22.5° (เคลื่อนไปได้
1 ช่วงห่างระหว่างซี่ฟันของโรเตอร์)



รูปที่ 1.5 แสดงลำดับการสวิทช์ 3 สเตปของ VR สเตปมอเตอร์แบบสแต็คเดียวและแสดงถึงตำแหน่งของโรเตอร์และเส้นทางของเส้นแรงแม่เหล็กเมื่อโรเตอร์เคลื่อนที่ไปในแต่ละสเตป

สัญลักษณต่าง ๆ ของ VR สเตปมอเตอร์

Nr	=	จำนวนซี่ฟันของโรเตอร์
Nr	=	จำนวนซี่ฟันของสเตปมอเตอร์
Np	=	จำนวนเฟส
Pr	=	ความห่างระหว่างปลายซี่ฟันของโรเตอร์ (องศา)
Ps	=	ความห่างระหว่างปลายซี่ฟันของสเตเตอร์ (องศา)
Θs	=	มุมสเตป (องศา)
Rs	=	อัตราการสเตปหรือความเร็วในการสเตป (สเตป/รอบ)
X	=	Ns/Np = จำนวน ซี่ฟันของสเตเตอร์ต่อเฟส

พารามิเตอร์ต่าง ๆ ของสเตปมอเตอร์

1. ความห่างระหว่างปลายซี่ฟันของโรเตอร์และสเตเตอร์ (tooth pitch)

$$Pr = \frac{360}{Nr} \quad \text{และ} \quad Ps = \frac{360}{Ns}$$

2. มุมสเตป (step angle)

ในรูปที่ 1.5 โรเตอร์จะเคลื่อนที่ในขนาดมุม P r ได้เท่ากับ N p สเตป ดังนั้นเราจะหามุมสเตปได้

$$\Theta s = \frac{Pr}{Np} = \frac{360}{NrNp} \quad \text{องศา/สเตป}$$

มุมสเตปจะเท่ากับความแตกต่างระหว่าง P r และ P s ดังนั้นเราหามุมสเตปได้เป็น

$$\Theta s = |Pr - Ps| \quad \text{องศา/สเตป}$$

3. อัตราการสเตป (stepping rate)

ความเร็วในการสเตปต่อรอบ (360 องศา) หาได้เป็น

$$Rs = \frac{360}{\Theta s} = NrNp \quad \text{(สเตป/รอบ)}$$

4. ความเร็วของสเตปมอเตอร์ (speed of step motor)

เมื่อเราป้อนอินพุทพัลซ์ที่มีความถี่ (f) สเตปต่อพัลซ์ให้กับสเตปมอเตอร์ มอเตอร์จะสเตปไปด้วยความเร็ว $\frac{(\text{สเตป})}{\text{พัลซ์}} \times f \frac{(\text{พัลซ์})}{\text{วินาที}}$

$$\frac{1\ (\text{รอบ})}{Rs\ \text{สเตป}} \times f \frac{(\text{พัลซ์})}{\text{วินาที} * \text{พัลซ์}} \frac{(\text{สเตป})}{} \times 60 \frac{(\text{วินาที})}{\text{นาที}}$$

$$\text{ความเร็วของมอเตอร์ } (w) = \frac{60f}{Rs} = \frac{60f}{NpNr} = \frac{\Theta sf}{6} \quad (\text{rpm})$$

5. จำนวนโพลของสเตเตอร์ต่อเฟส (number of stator poles per phase)

$$\text{จำนวนโพลของสเตเตอร์ต่อเฟส } (X) = \frac{Ns}{Np}$$

$$\text{หรือ } X = \frac{Rs}{Np(Np \pm 1)} = \frac{Nr}{(Np \pm 1)}$$

จำนวนโพลของสเตเตอร์ต่อเฟส (X) จะสัมพันธ์กับอัตราการสเตปหรือจำนวนซี่ฟันของโรเตอร์

สเตปมอเตอร์ในรูปที่ 1.4 เราสามารถสรุปการเลือกพารามิเตอร์บางตัวของสเตปมอเตอร์ได้ดังตาราง

ตาราง แสดงการเลือกพารามิเตอร์ของสเตปมอเตอร์

Np	Rs	Nr	X	Ns
3	48	16	4 8	12 24
4	48	12	4	16
4	64	16	?	?

ตัวอย่าง การหาพารามิเตอร์ของสเตปมอเตอร์

ขั้นแรกเรากำหนดความต้องการของมุมสเตป = 9°

มุมสเตปจะเป็นตัวจำกัดอัตราการสเตป = $\frac{360}{9}$ = 40 สเตป/รอบ

ในเงื่อนไขเหล่านี้เราอาจจะต้องใช้สเตปมอเตอร์ที่มี 4 หรือ 5 เฟส ที่มีสเตเตอร์ 2 โพล ต่อเฟส

ถ้า Np = 4

Nr = $\frac{Rs}{Np}$ = $\frac{40}{4}$ = 10

Ns = NpS = 4X2 = 8

ถ้า Np = 5

Nr = $\frac{40}{5}$ = 8

Ns = 5X2 = 10

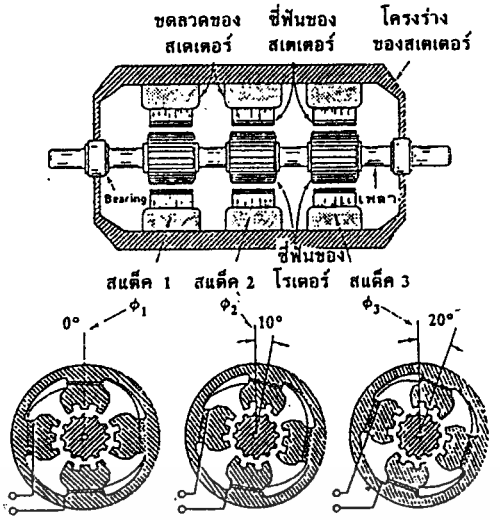
สเตปมอเตอร์แบบรีลัคแตนซ์แปรค่าได้และมีหลายสแต็ค

สเตปมอเตอร์แบบรีลัคแตนซ์แปรค่าได้ (VR) และมีหลายสแต็คหรือมากกว่า 1 สแต็คขึ้นไป

สแต็คในที่นี้หมายถึงเฟสซึ่งประกอบด้วยโรเตอร์ ที่เป็นซี่ฟันและโครงร่างของสเตเตอร์อยู่รอบนอก

สเตปมอเตอร์แบบ VR ที่มี 3 สแต็ค (หมายถึง 3 เฟส) มีโครงสร้างดังแสดงในรูปที่ 1.6

สเตปมอเตอร์ในรูปที่ 1.6 ได้ถูกออกแบบให้สเตเตอร์ของแต่ละสแต็คประกอบด้วย 4 โพล และแต่ละโพลจะมีซี่ฟัน 3 ซี่ซึ่งต่างจาก VR สเตปมอเตอร์แบบสแต็คเดียว (แต่ละโพลจะมีซี่ฟันเดียว) ข้อสังเกตในแต่ละสแต็คจำนวนซี่ฟันของโรเตอร์และสแตเตอร์จะมีจำนวนเท่ากัน ซึ่งต่างกับ VR สเตปมอเตอร์แบบสแต็คเดียวคือจำนวนซี่ฟันของโรเตอร์และสเตเตอร์จะเท่ากันไม่ได้ ถ้าหากมีจำนวนซี่ฟันเท่ากันมันจะไม่ทำงาน



รูปที่ 1.6 แสดงโครงสร้างของสเตปมอเตอร์แบบ VR ที่มี 3 เฟส โรเตอร์และสเตเตอร์ของแต่ละเฟส (สแต็ค) จะมี 12 ซี่ฟันและมุมสเตป (Θs) = 10 ° แต่ละเฟสของสเตเตอร์ที่เรียงลำดับต่อเนื่องกันจะถูกจัดตำแหน่ง ให้ต่างกันเท่ากับ 1/3ของช่องห่างระหว่างซี่ฟันของโรเตอร์ (10 °)

<u>การทำงานของ VR สเตปมอเตอร์ที่มี 3 สแต็ค</u>

ไดอะแกรมส่วนล่างของรูปที่ 1.6 แสดงถึงโครงสร้างของโรเตอร์และสเตเตอร์ของ VR สเตปมอเตอร์ที่มี 3 สแต็ค

แต่ละ สแต็คจะมี $Nr = Ns$

แต่ละสแต็คจะมีตำแหน่งของสเตเตอร์แตกต่างจากตำแหน่ง ของสเตเตอร์ใน สแต็คถัดไปเท่ากัน 10 °

ส่วนซี่ฟันของโรเตอร์ทั้ง 3 อันจะประกอบอยู่บนแกนเดียวกันและได้รับการปรับแต่งให้อยู่แนวเดียวกันอย่างสมบูรณ์

ตามปกติเราจะหาค่ามุมสเตป (หรือ index angle) ได้จากสมการ
ในที่นี้เราจะหา Θi (index angle) ได้จากสมการเดียวกันคือ

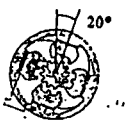
$$\Theta i = \frac{Pr}{Np} = \Theta s$$

ในกรณีนี้ $Nr = Ns = 12$ ดังนั้นเราหา $Pr = 360/12 = 30^\circ$ และค่า $\Theta i = 30/3 = 10^\circ$

สเตปมอเตอร์แบบ 3 สแต็ค ถึงแม้ว่าโรเตอร์ทั้ง 3 อันจะติดอยู่บนเพลาอันเดียวกันสแต็คทั้ง 3 สแต็คจะมีวงจรแม่เหล็กที่แยกกันดังนี้

ถ้าเฟสที่ 1 ถูกขับด้วยกระแสเป็นเฟสเริ่มต้นให้ซี่ฟันของโรเตอร์-สเตเตอร์อยู่ในแนวเดียวกัน ส่วนซี่ฟันของโรเตอร์และสเตเตอร์ในสแต็คที่ 2 ในขณะนั้นจะมีตำแหน่งต่างกัน 10° และซี่ฟันของโรเตอร์และสเตเตอร์ในสแต็คที่ 3 จะมีตำแหน่งต่างกัน 20° ต่อจากนั้นเราหยุดจ่ายกระแส (กระแสขดลวดสเตเตอร์) ในสแต็คที่ 1 และป้อนกระแสให้กับสแต็คที่ 2 อยู่ในแนวเดียวกันในขณะนี้ซี่ฟันของโรเตอร์และสเตเตอร์ในสแต็คที่ 3 จะมีตำแหน่งต่างกัน 10° ต่อจากนั้นเราหยุดจ่ายกระแสในสแต็คที่ 2 และป้อนกระแสให้กับสแต็คที่ 3 โรเตอร์จะหมุนไปอีก 10° ซึ่งจะทำให้ซี่ฟันของโรเตอร์และสเตเตอร์ในสแต็คที่ 3 อยู่ในแนวเดียวกัน ส่วนซี่ฟันของ โรเตอร์และสเตเตอร์ ในสแต็คที่ 1 จะมีตำแหน่งต่างกัน 10°

ลำดับการสวิทช์กระแสให้แต่ละสแต็คแสดงได้ในรูปที่ 1.7 ซึ่งแสดงให้เห็นว่าเพลาของสเตปมอเตอร์จะเคลื่อนที่ไปเท่ากับหนึ่งช่องของระยะห่างระหว่าง ซี่ฟันของโรเตอร์ (30°) ภายใน 3 สเตป

	สแต็คที่ 1	สแต็คที่ 2	สแต็คที่ 3
ตำแหน่งเริ่มต้นของ โรเตอร์ : - เฟส ϕ1 ได้รับพลังงาน			
สเตปที่ 1 : - เฟส ϕ 2 ได้รับพลังงาน - โรเตอร์จะเคลื่อนที่ไป 10 °			
สเตปที่ 2 : - เฟส ϕ 3 ได้รับพลังงาน - โรเตอร์จะเคลื่อนที่ไป 20 °			
สเตปที่ 3 : - เฟส ϕ 3 ได้รับพลังงาน - โรเตอร์จะเคลื่อนที่ไป 30 ° หรือเท่ากับหนึ่งช่องของระยะ ห่างระหว่างซี่ฟันของโรเตอร์			

รูปที่ 1.7 แสดงลำดับการสเตปของ VRSM แบบ 3เฟส N r = N s = 12, P r = 30 ° และ Θ s = 10 ซี่ฟันของโรเตอร์สีดำจะเคลื่อนที่ไปในทิศทาง CW 10° ในแต่ละสเตปรวมทั้งหมด 30 เมื่อสเตปไปครบ 3 สเตป สำหรับการหมุนในทิศทาง CW ตามลำดับการขับเฟส 1-2-3-1 และ

เมื่อต้องการให้หมุนในทิศทาง CCW ลำดับการขับเฟสก็ต้องกลับเป็น 1-3-2-1

ตามปกติเพลาของมอเตอร์จะเคลื่อนที่ไปหนึ่งช่อง ของระยะห่างระหว่างซี่ฟันของโรเตอร์ (rotor tooth pitch) ด้วยการสเตปไป N p สเตป เมื่อ N p คือจำนวนสแต็คที่ใช้ (หรือเท่ากับจำนวนเฟส)

ลำดับการสวิทช์ที่แสดงในรูปที่ 1.7 เราสามารถนำมาเขียนเป็นตารางได้ ดังในรูปที่ 1.8 วงจรสวิทช์ประกอบด้วย VRSM แบบ 3 เฟส (สัญลักษณ์ของสเตปมอเตอร์) การขับ ฟสแสดงได้ด้วยสวิทช์และแหล่งการกำเนิดดีซี

สเตป	S_1	S_2	S_3
1	×		
2		×	
3			×
1	×		

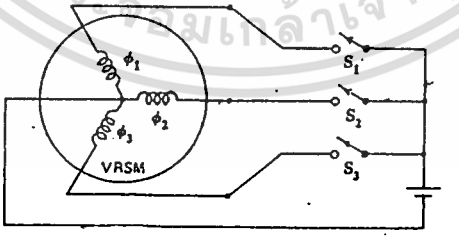
(ก)

สเตป	S_1	S_2	S_3
	×	×	
		×	×
	×		×
	×	×	

(ค)

(ข)

สเตป	S_1	S_2	S_3
1	×	×	
2		×	
3		×	×
4			×
5	×		×
6	×		
1	×	×	



(ง)

รูปที่ 1.8 แสดงถึง VRSM แบบ 3 เฟส (ก) ตารางแสดงลำดับการขับแบบเฟสเดียวในทิศทาง CW (ข) ตารางแสดงลำดับการขับแบบ 2 เฟสในทิศทาง CW

(ค) การขับแบบ ครึ่งสเตปในทิศทาง CCW เราจะต้องกลับลำดับของการขับคือให้อ่านตาราง (ก) (ข) และ (ค) จากข้างล่างขึ้นไปข้างบน

จากตาราง (ก) ถ้าเราขับเฟสที่ 1 และเฟสที่ 2 เรียงตามลำดับมอเตอร์จะหมุนไปหนึ่งสเตป

จากตาราง (ข) ถ้าเราขับเฟสที่ 1 และเฟสที่ 2 พร้อมกันเพลาของมอเตอร์จะหมุนไป 1/2 สเตป ต่อจากนั้นเราขับเฟสที่ 2 และเฟสที่ 3 พร้อมกันอีกก็จะทำให้มอเตอร์หมุนไปครบเต็มหนึ่งสเตป ดังนั้นการขับแบบ 2 เฟสเราเรียงลำดับการขับได้ดังนี้ 1-2, 2-3, 3-1 และ 1-2 กระทำซ้ำเดิมไปเรื่อย ๆ

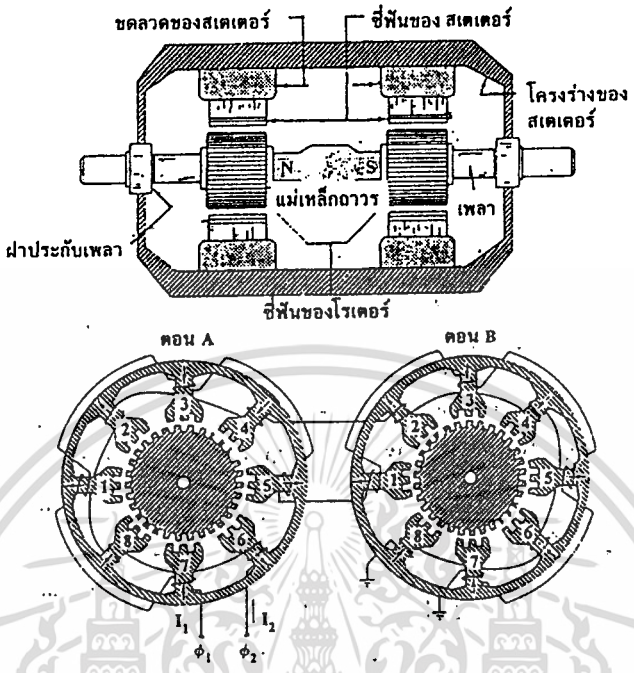
อย่างไรก็ตามการขับแบบ 2 เฟสหรือ 1 เฟสจะให้การหมุนเป็นมุมสเตปเท่ากันที่ต่างกัน ก็คือการขับแบบ 2 เฟสจะให้การหมุนของโรเตอร์นำหน้าการขับแบบเฟสเดียวด้วยขนด 1/2 สเตป นอกจากนั้นการขับแบบ 2 เฟส จะต้องการกระแสเป็น 2 เท่าของการขับแบบเฟสเดียว

ตาราง (ค) แสดงการขับแบบ 2 เฟสสลับกับการขับแบบ 1 เฟส ซึ่งจะทำให้โรเตอร์ หมุนไป 1/2 สเตปเท่านั้น การขับแบบนี้จะทำให้จำนวนสเตปต่อรอบเพิ่มขึ้นเป็น 2 เท่าจากเดิม

สเตปมอเตอร์แบบไฮบริด

ไฮบริดสเตปมอเตอร์ (HSM) มีคุณลักษณะผสมของ PM และ VR สเตปมอเตอร์ในรูปที่ 1.9 แสดงถึงโครงสร้างของ HSM ประกอบด้วย 2 ตอนกับแกนแม่เหล็กอยู่ระหว่าง 2 ตอน แต่ละตอนประกอบด้วยซี่ฟันของโรเตอร์และโพลของสเตเตอร์ที่มีซี่ฟันเช่นกัน

และพันด้วยขดลวด รายละเอียดโครงสร้างของสเตเตอร์และโรเตอร์ของแต่ละตอนแสดงในไดอะแกรมข้างล่างของรูปที่ 1.9



รูปที่ 1.9 โครงสร้างของไฮบริดสเตปมอเตอร์ : $N_r = 30$, $N_s = 24$ ซี่ฟันของสเตเตอร์ทั้ง 2 ตอนจะอยู่ในแนวเดียวกันส่วนซี่ฟันของโรเตอร์ทั้ง 2 ตัว จะมีตำแหน่งต่างกัน 1/2 P_r ($= 6^\circ$), $\Theta_s = 3^\circ$

ลักษณะโครงสร้างของไฮบริดสเตปมอเตอร์

- จำนวนซี่ฟันของโรเตอร์และของสเตเตอร์ไม่เท่ากัน
- ตอน A และ ตอน B มีโครงสร้างเหมือนกัน
- ซี่ฟันของสเตเตอร์ทั้ง 2 ตอนจะอยู่ในแนวเดียวกันอย่างถูกต้อง
- ส่วนซี่ฟันของโรเตอร์ทั้ง 2 ตอนจะมีตำแหน่งที่แตกต่างกัน 1/2 P_r (ในรูปที่ 1.9 กำหนดให้ $P_r = \frac{360}{30} = 12^\circ$ ดังนั้นตำแหน่งซี่ฟันของโรเตอร์ทั้ง 2 ตอนจะแตกต่างกัน 6°)

- สเตเตอร์ของแต่ละตอนมี 8โพลแบ่งของเป็น 2 สเตเตอร์เฟส
- เฟสที่ 1 จะพันขดลวดบนสเตเตอร์โพลหมายเลข 1,3,5 และ 7 ของทั้งในตอน A และ ตอน B
- เฟสที่ 2 จะพันขดลวดบนสเตเตอร์โพลหมายเลข 2,4,6 และ 8 ของทั้งในตอน A และตอน B
- แกนแม่เหล็กถาวรจะเหนี่ยวนำโรเตอร์ในตอน A ให้เป็นแม่เหล็กขั้วเหนือ และโรเตอร์ในตอน B ให้เป็นแม่เหล็กขั้วใต้ ความซับซ้อนจะเพิ่มมากขึ้นเนื่องจากการ แบ่งส่วนของ ขดลวดเฟสใน 2 ตอนทำให้ได้วงจรแม่เหล็กที่ซับซ้อนและได้เส้นทางเดินของเส้นแรงแม่เหล็ก ที่แตกต่างกันเป็นวงกลมทิศทางเดินของสนามแม่เหล็กของสเตเตอร์โพล จะขึ้นอยู่กับทิศทางการไหลของกระแสเฟส ดังแสดงด้วยลูกศรในรูปที่ 1.9

การทำงานของไฮบริดสเตปมอเตอร์

ขณะที่เฟสที่ 1 (ϕ1) ได้รับพลังงานโดยการป้อนกระแส I1 ในทิศทางดังแสดงด้วยลูกศร

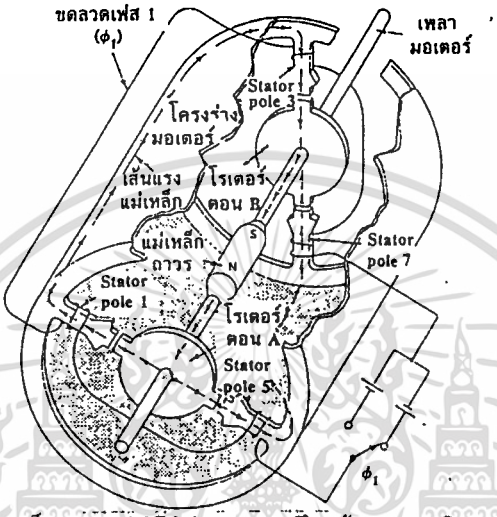
ซี่ฟันของโรเตอร์ในตอน A จะอยู่ในแนวเดียวกับซี่ฟันของสเตเตอร์ของโพลที่ 1 และโพลที่ 3 ส่วนของตอน B จะอยู่ในแนวเดียวกับซี่ฟันของโพลที่ 3 และโพลที่ 7 ดังแสดงในรูปที่ 1.10

เพื่อให้เพลาของมอเตอร์หมุนไปหนึ่งสเตปในทิศทาง CW เราจะต้องหยุดป้อนกระแส I 1 และป้อนกระแส I 2 ให้กับเฟสที่ 2 (ϕ2)

ในรูปที่ 1.11 ซี่ฟันของโรเตอร์ที่เป็นสีดำใช้สำหรับอ้างอิง ซี่ฟันสีดำจะอยู่ใกล้แนวซี่ฟัน ของสเตเตอร์โพลที่ 4 และโพลที่ 8 ในตอน A และโพลที่ 2 และ โพลที่ 6 ในตอน B

มากที่สุด (ซี่ฟันของโรเตอร์ที่เป็นสีดำอยู่ห่างจากแนวซี่ฟันของสเตเตอร์เท่ากับ 1 สเตปพอดี)

เราจะต้องป้อนกระแส I 2 ในทิศทางที่ถูกต้องคือจะต้องทำให้โพลที่ 4 และโพลที่ 8 และโพลที่ 2 และ โพลที่ 6 ถูกเหนี่ยวนำเป็นแม่เหล็กในทิศทางที่ถูกต้อง (เกิดวงจรแม่เหล็กที่สมบูรณ์) ด้วย I2 ในกรณีนี้ I2 จะต้องเป็นลบ



รูปที่ 1.10 วงจรแม่เหล็กของ HSM แสดงถึงเส้นทางเดินของเส้นแรงแม่เหล็ก เมื่อเฟสที่ 1 ได้รับพลังงานและเส้นแรงแม่เหล็กเกิดขึ้นในตอน A จะผ่านโพลที่ 1 และโพลที่ 5 เข้าไปยังโรเตอร์ของตอน B ผ่านโพลที่ 3 และ โพลที่ 7 เข้าสู่ขั้วใต้ (S) ของแม่เหล็กถาวร

ในรูปที่ 1.11 แสดงถึงลำดับการสวิทช์ (ให้กระแสไหล) ให้มอเตอร์หมุนไปในทิศทาง CW 4 สเตป ซึ่งแสดงถึง ตำแหน่งของโรเตอร์และทิศทางการเป็นแม่เหล็กของสเตเตอร์โพลในแต่ละตอนด้วยการกำหนดทิศทางของการไหลของกระแสเฟส สำหรับการหมุนในทิศทาง CW (ดังแสดงในรูป) เราจะต้องกำหนดลำดับของกระแสเฟสดังนี้ 1, + 2,-1,-2 และ 1 + ตามลำดับ ถ้าต้องการหมุนในทิศทาง CCW ลำดับเหล่านั้นก็กลับไปเป็น 1 +, 2+, 1-, 2- และ 1+

เพลาของมอเตอร์หมุนไปได้หนึ่งช่องห่างระหว่างซี่ฟันถายใน 4 สเตป ดังนั้น มุมสเตป จะต้องเท่ากับ 1/4 P r หรือมีค่าเท่ากับ |P s - P s| ดังนั้น

$$\Theta s = \frac{Pr}{4} = \frac{360}{4Nr} = \frac{90}{Nr}$$

$$\Theta s = |Ps - Pr|$$

สเตป	ϕ_1 I_1	ϕ_2 I_2	เส้นแรงออกจากตอน A	เส้นแรงเข้าสู่ตอน B	ตอน A ตอน B
1	+		1,5	3,7	
2		–	4,8	2.6	
3	–		3,7	1,5	
4		+	2,6	4,8	
1	+		1,5	3.7	

รูปที่ 1.11 ลำดับ 4 สเตปของ HSM แบบ 2 เฟส ในแต่ละสเตปแสดงถึงตำแหน่งของโรเตอร์และทิศทางของเส้นแรงแม่เหล็ก N r = 30, = 24, O s = 3 ° ซี่ฟันของโรเตอร์ที่เป็นสีดำจะหมุนในทิศทาง CW ไป 3 °ในแต่ละสเตปได้เป็น 12 °

เมื่อครบตามจำนวนลำดับ (หนึ่งช่องห่างระหว่างซี่ฟันของโรเตอร์) สำหรับการหมุนในทิศทาง CW จะต้องจัดลำดับการขับเป็น 1+, 2-, 1-,2+,1+

ในรูป $Nr = 30$ และ $Ns = 24$

ดังนั้น $\Theta s = \frac{90}{30} = 3^\circ$

$= \frac{360}{24} - \frac{360}{30} = 3^\circ$

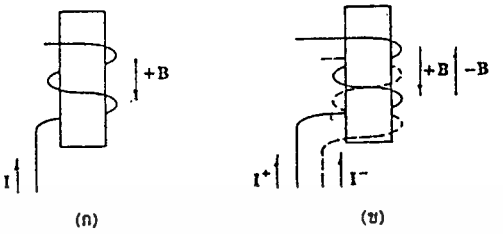
ไฮบริดสเตปมอเตอร์ (HSM) จะทำงานด้วยกระแสเฟสที่มีการไหล ได้สองทิศทางดังนั้นเราจำเป็นต้องใช้เพาเวอร์ซัพพลาย 2 ตัว (bipolar drive)

การแก้ปัญหาเพื่อจะขับไฮบริดสเตปมอเตอร์ให้ทำงานด้วยเพาเวอร์ซัพพลายเพียง ตัวเดียว (unipolar drive) ได้โดยดัดแปลงโครงสร้างการพันขดลวดเฟส ของสเตเตอร์

การพันขดลวดเฟสของสเตเตอร์แบบ bifilar (การพันแบบสองแถวสลับกัน) สามารถขับได้ด้วย ยูนิโพล่าร์ (unipolar dirve)

ขดลวดแบบ unifilar แสดงดังในรูปที่ 1.12 (ก) จะต้องกลับทิศทางของกระแสเพื่อกลับทิศทางของเส้นแรงแม่เหล็ก B

ขดลวดแบบ bifilar แสดงในรูปที่ 1.12 (ข) ถ้าเราต้องการกลับ ทิศทางของเส้นแรงแม่เหล็กเป็น -Bสามารถทำได้โดยป้อนกระแสขนาดเดิมจากเพาเวอร์ซัพพลายตัวเดิมเข้าที่ขดลวดที่เป็นเส้นประในรูปที่ 1.12 (ข) ก็จะทำให้ทิศทางการเหนี่ยวนำแม่เหล็กและทิศทางของเส้นแรงแม่เหล็ก (-B) กลับทิศทางได้

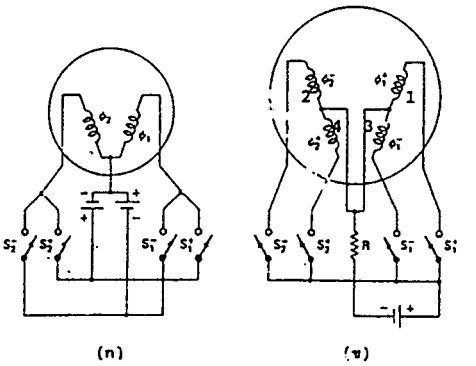


รูปที่ 1.12 การพันขดลวดเฟสของสเตเตอร์
(ก) แบบ unifilar (ข) แบบ bifilar

ถ้าหาก HSM ในรูปที่ 1.9 มีขดลวดเฟสของสเตเตอร์เป็นแบบ bifilar
ขดลวดเฟส ϕ_1 เดิมจะถูกแบ่งตัวออกเป็นสองขดลวดเฟส ϕ_1 + และเฟส ϕ_1-
ขดลวดเฟส ϕ_2 เดิมจะถูกแบ่งตัวออกเป็นสองขดลวดเฟส ϕ_2 + และเฟส ϕ_2-

ในตอนนี้ก็จะทำให้เราได้ขดลวดเฟสถึง 4 เฟสและแต่ละเฟสสามารถขับได้ด้วยกระแสที่ไหลใน ทิศทางเดียว ส่วนเครื่องหมาย + และ - ใช้สำหรับแสดงถึงทิศทางการเกิดสนามแม่เหล็กของสเตเตอร์โพล

ในรูปที่ 1.13(ก) (ข) แสดงวงจรการสวิทช์ 2 วงจร สำหรับ HSM แบบ 2 เฟส และ แบบ 4 เฟส รูป (ค) (ง) และ (จ) แสดงตารางลำดับการขับแบบทีละเฟส และแบบทีละ 2 เฟส และการขับแบบครึ่งสเตปตามลำดับ



(ก)

สเตป	S_1	S_2	S_3	S_4
1	X			
2				X
3		X		
4			X	
1	X			

(ค)

สเตป	S_1	S_2	S_3	S_4	
1	X			X	- 1-2
2		X		X	- 2-3
3		X	X		- 3-4
4	X		X		- 4-1
1	X			X	- 1-2

(ง)

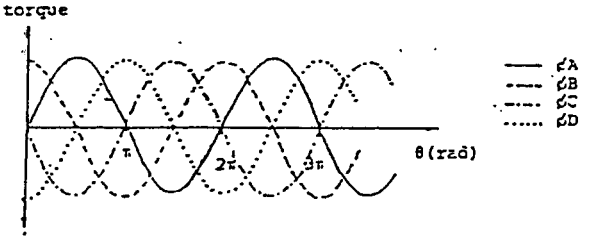
สเตป	S_1	S_2	S_3	S_4	
1	X			X	1-2
2				X	2
3		X		X	2-3
4		X			3
5		X	X		3-4
6			X		4
7	X		X		4-1
8	X				1
1	X			X	1-2

(จ)

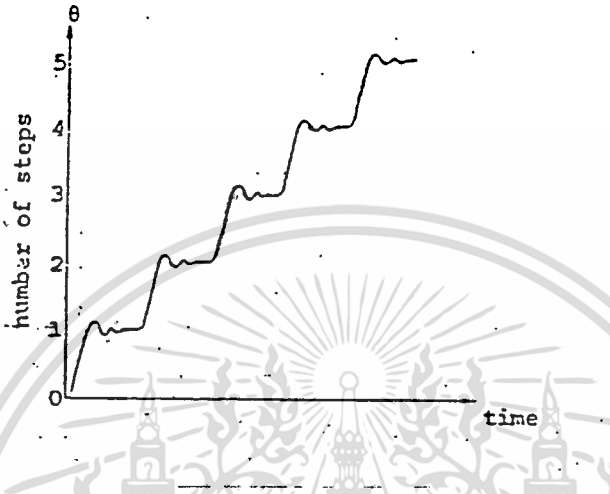
รูปที่ 1.13 ไฮบริดสเตปมอเตอร์ (ก) HSM แบบ 2 เฟส unifilar จะต้องขับแบบไบโพล่าร์ (ข) HSM แบบ 4 เฟส bifilar ใช้การขับแบบยูนิโพล่าร์ (ค) ตาราง แสดงลำดับการขับทีละ 1 เฟส (ง) ตารางแสดงลำดับการขับทีละ 2 เฟส (จ) ตาราง แสดงลำดับการขับ แบบครึ่งสเตปในทิศทาง CW

การเพิ่มความละเอียดในแต่ละ STEP ของมอเตอร์

สำหรับ stpping motor เมื่อให้สัญญาณไฟฟ้าผ่านขดลวดเส้นใดเส้นหนึ่ง จะเกิด torque ขึ้น ซึ่ง torqueที่ได้จะมีการเปลี่ยนแปลงไปตามมุมของแกนมอเตอร์ ดังรูปที่ 1.14



จากรูปจะพบว่าเมื่อจ่ายกระแสให้เฟส A,B,C และ D ตามลำดับจะเกิดจุดสมดุลย์ที่มุม (2n-1) (2n-1/2) 2n และ (2n+1/2) ตามลำดับเช่นกันจากผลดังกล่าวทำให้แกนมอเตอร์มีการหมุนไปทีละ step และมีโอกาสเกิด osoillate ได้ดังแสดงดังรูปที่ 1.15



รูปที่ 1.15 แสดง single - step response

ผลลัพธ์ที่ได้จากรูปที่ 1.15 จะสังเกตได้ง่ายเมื่อ stepping motor มีขนาดแรงเฉื่อยของโหลดสูงขึ้น โดยเฉพาะ stepping motor ที่มีค่าองศาต่อ step มากเช่น 15 /step

จากผลดังกล่าว สามารถสรุปผลเบื้องต้นได้ว่า

1. stepping motor มีการหมุนของแกนที่ไม่ต่อเนื่อง เนื่องจากเป็น step
2. steppign motor เกิดการ damp ได้ง่ายและคาบการ damp จะยาวขึ้นเมื่อมีแรงเฉื่อยมาก

ในการแก้ปัญหานี้ สามารถแก้ได้โดยการลดขนาด องศา/step ของ stepping motor ให้น้อยลง ซึ่งสามารถกระทำได้ 2 วิธีคือ

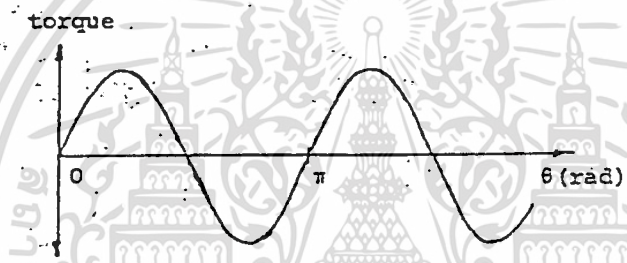
1. พัฒนาทางด้านโครงสร้างของ stepping motor

2. พัฒนาทางด้าน control system

ในกรณีนี้การพัฒนาทางด้าน control system จะสะดวกกว่าอีกทั้งสามารถนำไปประยุกต์ใช้กับ stepping motor ขนาดและชนิดอื่น ๆ ได้

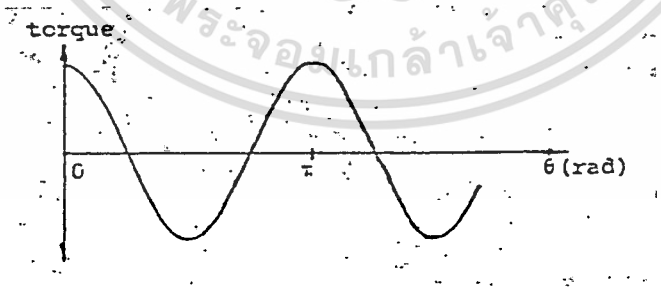
หลักการ

สำหรับ permanent and hybrid stepping motor นี้เมื่อมีการจ่ายกระแสไฟฟ้า Ixผ่านเฟส A จะได้ความสัมพันธ์ดังรูปที่ 1.16



รูปที่ 1.16 แสดงความสัมพันธ์ระหว่าง torque กับ deree เมื่อมีกระแสไหลผ่านเฟส A

เช่นเดียวกันเมื่อมีกระแส I x ผ่านเฟส B จะได้ความสัมพันธ์ใหม่ดังรูปที่ 1.17

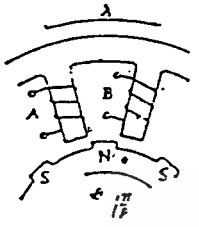


รูปที่ 1.17 แสดงความสัมพันธ์ระหว่าง torque กับ deree เมื่อมีกระแส IM ผ่านเฟส B

จากความสัมพันธ์ดังกล่าวสามารถเขียนรูปแบบสมการเบื้องต้นได้ว่า

สำหรับเฟส A torque A = $-pn\ \phi_M\ i_a \sin p\theta$ —(1)

สำหรับเฟส B torque B = $-pn\ \phi_M\ i_t \sin p\ (\theta - \lambda\ )$ —(2)



pn = number of pairs of magnetic poles

ϕ_M = peak flux linkage

i a, i b = กระแสที่ไหลผ่านเฟส A และ Bตามลำดับ

θ = ตำแหน่งมุมปัจจุบันของแกนโรเตอร์

λ = มุมระหว่างเฟส A และ B

จากข้อมูลที่ได้จากกราฟในรูปที่ 1.16และ 1.17 นั้นจะพบว่าจุดสมดุลย์จะมีการเคลื่อนตำแหน่งไปทีละstep ซึ่งเป็นผลให้เกิดการเคลื่อนที่เป็น step ขึ้น ดังนั้นการที่จะทำให้แกนมอเตอร์ มีการหมุนที่ต่อเนื่องมากขึ้น จึงต้องหาทางปรับปรุง ช่วงกว้างระหว่างจุดสมดุลย์ที่อยู่ใกล้กัน 2 จุด ให้มีขนาดแคบลง ซึ่งจะเป็นการลดความกว้างของ step เช่นกัน ผลที่ได้จะทำให้จำนวน step/รอง ของมอเตอร์มีมากขึ้น ทำให้มอเตอร์สามารถอ้างอิงตำแหน่งของมอเตอร์ได้ละเอียดขึ้น

พิจารณา สมการที่ 1 และ 2 นำมาประยุกษ์เขียนรูปแบบสมการง่าย ๆ ขึ้นใหม่ โดยกำหนดค่า p = 1 และ = 90 องศา จะได้ว่า

torque A = $K i_a \sin\theta$ ——(3)

torque B = $K i_b \sin(\theta - \pi/2)$ ——(4)

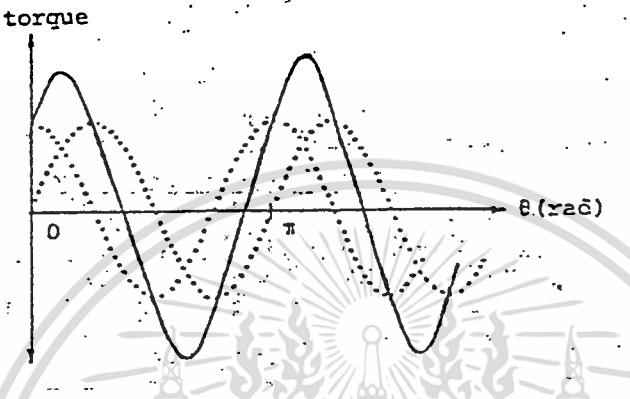
K = $-pn\phi_M$ (ซึ่งเป็นค่าคงที่)

ถ้าให้ i_a และ i_b มีค่าเท่ากับกระแส i max จะได้ผลลัพธ์ของ torque ดังสมการ

$$\text{torque}_{\text{รวม}} = \text{torque A} + \text{torque B}$$

$$= K(i_a \sin\theta + i_b \sin(\theta - \pi/2)) \quad \text{---(5)}$$

นำสมการ 5 มาเขียนกราฟจะได้ดังรูปที่ 1.18

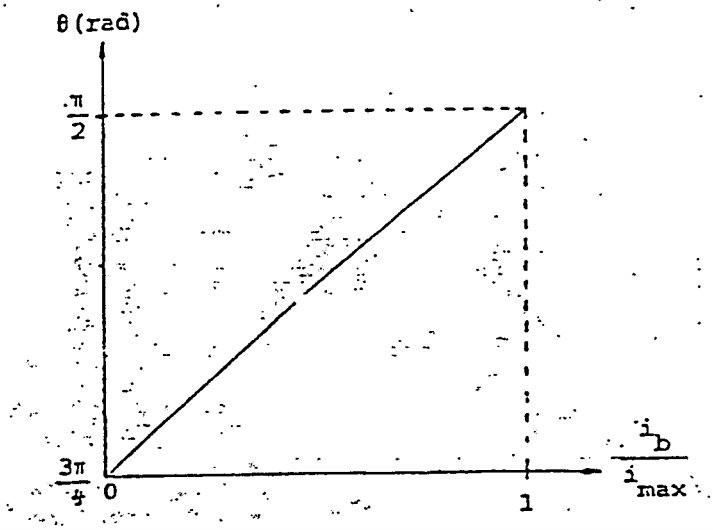


รูปที่ 1.18 แสดงความสัมพันธ์ระหว่าง torque กับ deree เมื่อ $i_a = i_t = I_{max}$

จากรูปที่ 1.18 จะพบว่า torque จะมีค่า peak สูงขึ้นและจุดสมดุลย์จะอยู่ระหว่าง $\pi/2$ กับ π คือ $3\pi/4$ ซึ่งเป็นตำแหน่งกึ่งกลางพอดี ดังนั้นแสดงว่าการขับมอเตอร์แบบ one-two excitiation จะเพิ่มจำนวน step/รอบ ขึ้นเป็น 2 เท่า ซึ่งเรียกกันว่า "การขับแบบ half step"

พิจารณา สมการที่ 5 จะพบว่า ถ้า i_a มีค่าคงที่เท่ากับ i_{max} และ i_b มีค่าระหว่าง O ถึง i_{max} จะพบว่า จุดสมดุลย์ย่อมจะมีการเปลี่ยนแปลงระหว่าง θ กับ $\theta - \frac{\pi}{2}$ ซึ่งสามารถหาจุดสมดุลย์ที่เปลี่ยนแปลงตาม i_b ได้ดังนี้

จากสมการที่ 5 มาเขียนกราฟโดยปรับปรุงค่า θ ที่ได้ให้อยู่ในช่วง $\pi/2$ ถึง จะได้ดังรูปที่ 1.19

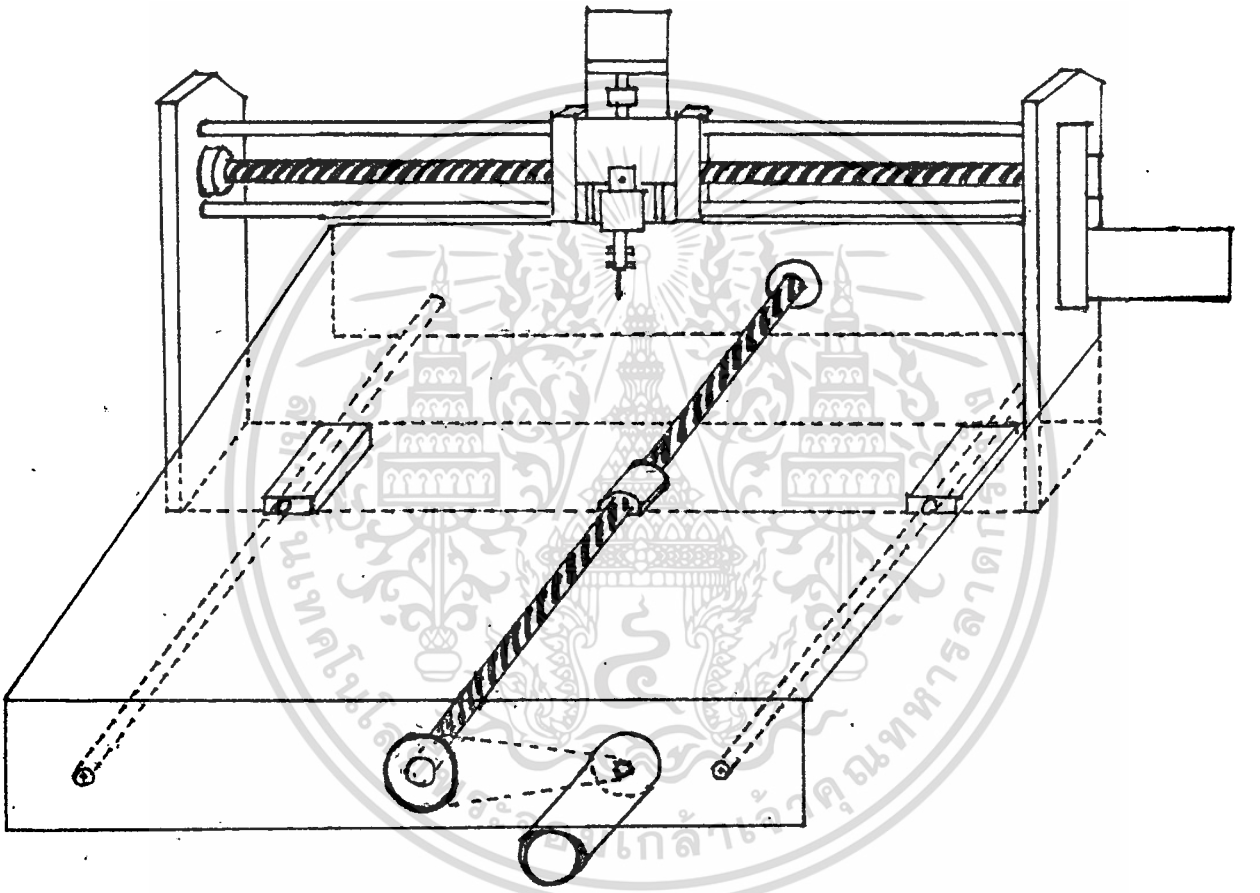


รูปที่ 1.15 แสดงความสัมพันธ์ระหว่าง i b / i max กับ θ

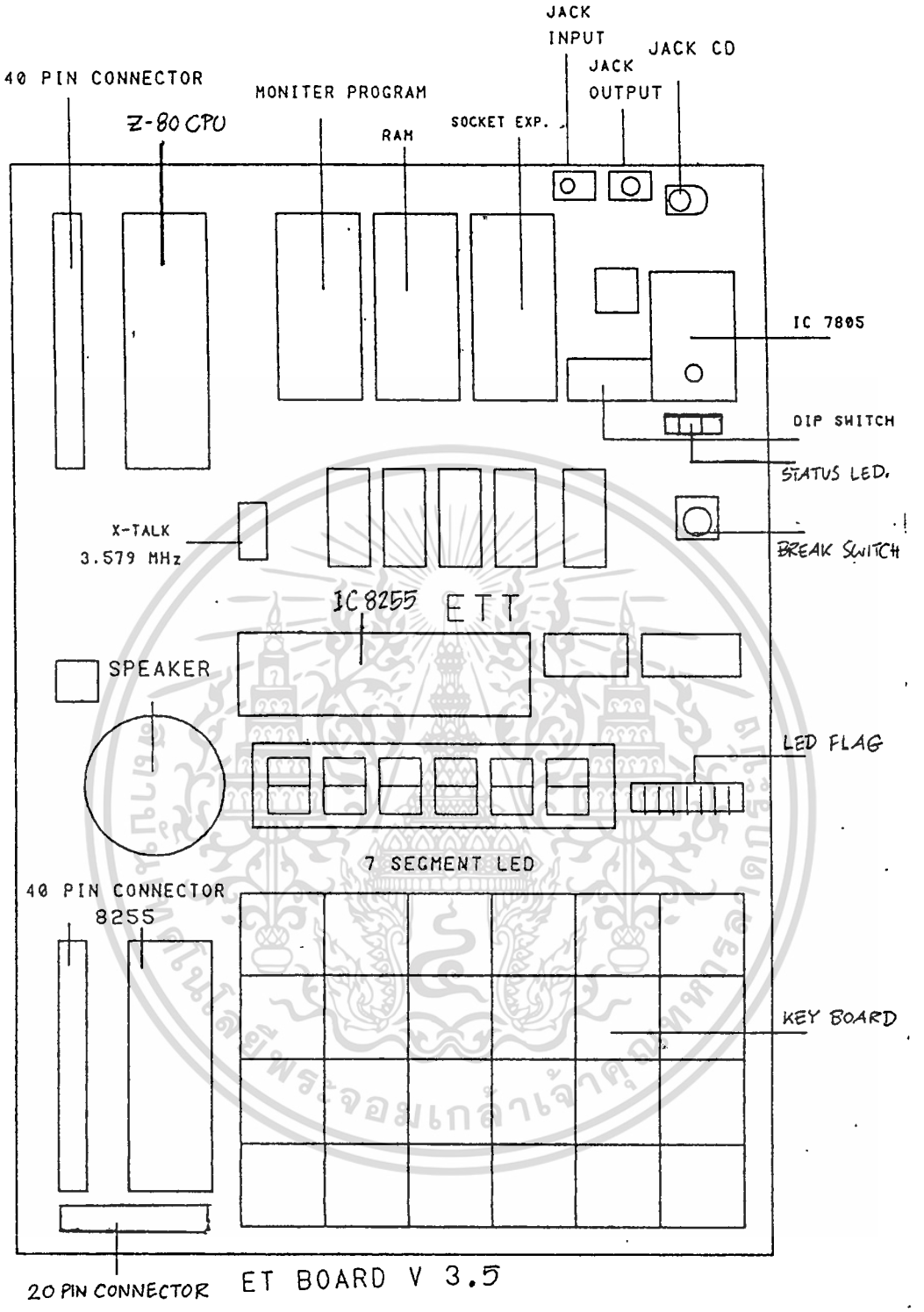
จากกราฟจะพบว่าจุดสมดุลย์มีการเปลี่ยนแปลงตำแหน่งมุมตาม i b / i max แสดงว่าในทางปฏิบัติ หากสามารถควบคุมค่า i b / i max ได้ ย่อมมีทางที่จะควบคุมตำแหน่งจุดสมดุลย์ได้เช่นกัน

บทที่ 2

ฮาร์ดแวร์ของระบบ



รูป 2.1 แสดงโครงสร้างของระบบ MECHANIC



รูป 2.2 แสดงโครงสร้างของบอร์ดควบคุม

รายละเอียดของบอร์ดควบคุม (อี ที บอร์ด)

JACK DC

คือ JACK ที่เสียบ ADAPTER สำหรับจ่ายไฟเลี้ยง โดยมีขนาด 9 V DC 800 mA ขั้วเสียบจะมีลักษณะวงในเป็นขั้วลบ และวงนอกเป็นขั้วบวก

JACK IN, OUT

คือ JACK สำหรับการรับส่งข้อมูล ซึ่งใช้ต่อกับเครื่องเล่นเทปได้ ต่อกับเครื่องคอมพิวเตอร์ APPLE และกับเครื่องคอมพิวเตอร์ IBM / PC หรือแบบเทียบเท่าในกรณีที่ต่อกับ PC จะต้องใช้สาย JACK พิเศษที่ออกแบบโดยเฉพาะ

7 SEGMENT DISPLAY

คือส่วนแสดงข้อมูลและสถานะต่าง ๆ ในการใช้งาน ทั้งนยั้รวมถึง LED อีก 8 ตัวทางด้านขวาด้วย เรียกว่า FLAG LED สำหรับการแสดงข้อมูลในระดับ BIT

KEY BOARD

คือส่วนสำหรับการใส่ข้อมูลและการกำหนดการทำงานต่างๆ คีย์บอร์ดนี้จะแบ่งได้เป็น 2 ส่วนคือ คีย์ตัวเลข O - F จำนวน 16 คีย์ และคีย์คำสั่ง 8 คีย์ คีย์ตัวเลขนี้ ยังรวมถึงคีย์ฟังก์ชั่นด้วย ซึ่งมี 2 ระดับรวมทั้งหมดเป็น 32 ฟังก์ชั่น

BREAK SWITCH

คือ สวิทช์ สำหรับการ BREAK โปรแกรมของผู้ใช้ ในขณที่โปรแกรมของผู้ใช้ กำลังทำงานเมื่อกดสวิทช์นี้จะทำให้โปรแกรมถูก BREAK ทันที ซึ่งจะสามารถตรวจดูการทำงานได้อย่างสะดวก อีกนัยหนึ่งสวิทช์นั้นคือ INT นั่นเอง ทำให้ผู้ใช้ทำการทดสอบเกี่ยวกับการ INTERRUPT ได้อย่างสะดวก

LED แสดงสถานะ

คือ LED 3 ตัวสำหรับแสดงผลทั่ว ๆ ไป สีแดงแสดง POWER สีเหลืองแสดงการกดสวิทช์ BREAK สีเขียวแสดงสถานะ 'HALT ของตัว CPU

SOCKET ว่าง

มีอยู่ 2 ส่วน คือ 28 PIN และ 40 PIN ส่วน 28 PIN สำหรับการขยายหน่วยความจำซึ่งจะใช้ได้กับเบอร์ 6264 2764 2732 2716 และ 6116 โดยการเลือกจาก DIP SWITCH ส่วน 40 PIN สำหรับการขยาย PORT ด้วยเบอร์ 8255

40 PIN CONNECTOR

คือส่วนขยายการใช้งานของบอร์ด Z80 CONNECTOR สำหรับการขยายทั่วไปและรวมถึงการเพิ่ม CARD OPTION ซึ่งเพิ่มความสามารถของ อีทีบอร์ดในงานต่าง ๆ , 8255 CONNECTOR สำหรับการขยาย PORT ซึ่งเหมาะกับงานควบคุมตามจุดประสงค์เฉพาะอย่างของผู้ใช้

DIP SWITCH

สำหรับการเลือกเบอร์ชิพที่ SOCKET ว่างขนาด 28 PINโดยถ้าต้องเลือกเบอร์ไหนก็ให้ ON เบอร์นั้น ตั้งแต่ 6-7 สำหรับเลือกขนาดสัญญาณของ JACK OUT โดยถ้า 6 ON, 7 OFF หมายถึงสัญญาณที่แรงขึ้น เรียกว่า HI GAIN MODE ส่วนเบอร์ 8 สำหรับ ON BATTERY สำหรับการกรองข้อมูล

เมื่อจะเริ่มใช้งาน ให้ผู้ใช้เสียบ ADAPTER เข้ากับบอร์ด จากนั้นที่เสียงร้องและมีอักษรแสดงบน DISPLAY โดยวิ่งจากขวามาซ้าย เรียบร้อยแล้วจะมีเครื่องหมายขีดแสดงบน DISPLAY ที่อักษรซ้ายสุด ซึ่งหมายถึง ความพร้อมที่จะใช้งานต่าง ๆ ได้ต่อไป

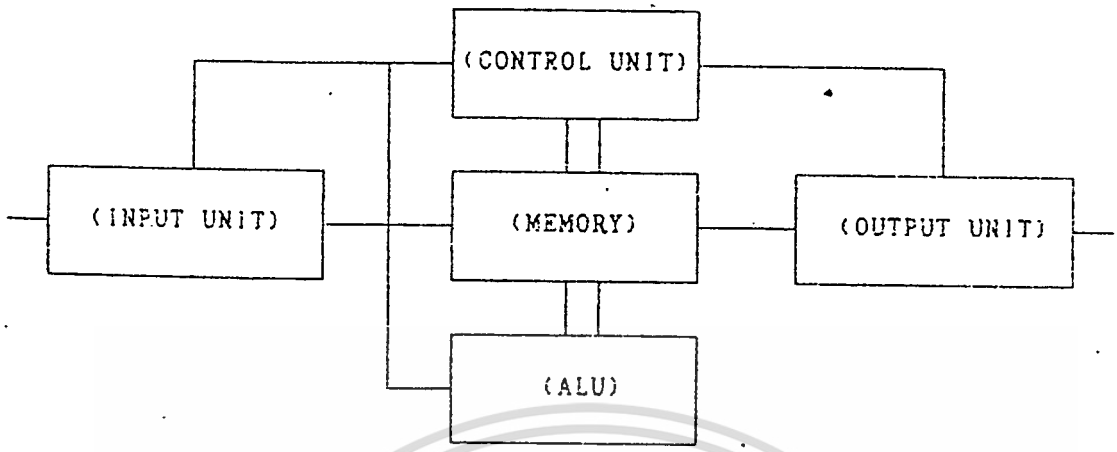
ความรู้ทั่วไปเกี่ยวกับไมโครโปรเซสเซอร์

จากอดีตที่ผ่านมาระบบคอมพิวเตอร์ได้รับการพัฒนาและเปลี่ยนแปลงไปอย่างมากและยังมีแนวโน้ม ที่จะพัฒนาต่อไปอีกในอนาคต ซึ่งในปัจจุบันอุปกรณ์ต่าง ๆ ที่เกี่ยวกับคอมพิวเตอร์ก็ได้มีขนาดเล็กลง ทำงานรวดเร็วและมีความเชื่อถือได้มากขึ้น ดังนั้นจึงทำให้คอมพิวเตอร์ได้เข้ามามีส่วนเกี่ยวข้องกับชีวิตประจำวันในสังคมมนุษย์ไปอย่างหลีกเลี่ยงไม่ได้ จะเห็นได้จาก การนำเอาคอมพิวเตอร์เข้าไปใช้ ในระบบการ จัดการเกี่ยวกับธุรกิจ ระบบการควบคุมต่าง ๆ ภายในโรงงานอุตสาหกรรม เป็นต้น และมีแนวโน้มที่จะนำมาประยุกต์ใช้เพิ่มมากขึ้นเรื่อย ๆ

การใช้ไมโครโปรเซสเซอร์ (MICROPROCESSOR) ควบคุมความเร็วมอเตอร์ก็เป็นอีกวิธีหนึ่งที่นำเอาระบบคอมพิวเตอร์เข้ามาประยุกต์ใช้อย่างมีประสิทธิภาพซึ่งมีผลดีคือให้ความแม่นยำในการควบคุมสูง มีการสนองตอบต่อปฏิกริยาที่เกิดขึ้นค่อนข้างรวดเร็ว และยังให้ความสะดวกสบายในการทำงาน เพราะวิธีการควบคุมโดยซอฟท์แวร์ (SOFTWARE) นั้นเมื่อต้องการแก้ไขคำสั่งก็สามารถกระทำได้ง่ายโดยการเปลี่ยนแปลงทางด้านซอฟท์แวร์เท่านั้นไม่ต้องไปเปลี่ยนโครงสร้างทางด้านฮาร์ดแวร์ (HARDWARE) ซึ่งมีความยุ่งยากในทางปฏิบัติ

พื้นฐานทั่วไปของไมโครคอมพิวเตอร์

ส่วนประกอบต่าง ๆ ของไมโครคอมพิวเตอร์จะเชื่อมต่อกันด้วยบัส (BUS)ส่วนประกอบเหล่านี้จะมีลักษณะเป็น IC แบบ LSI (LARGE SCALE INTREGRATED CIRCUIT ในรูป 2.3 นี้



รูป 2.3 โครงสร้างพื้นฐานของไมโครคอมพิวเตอร์

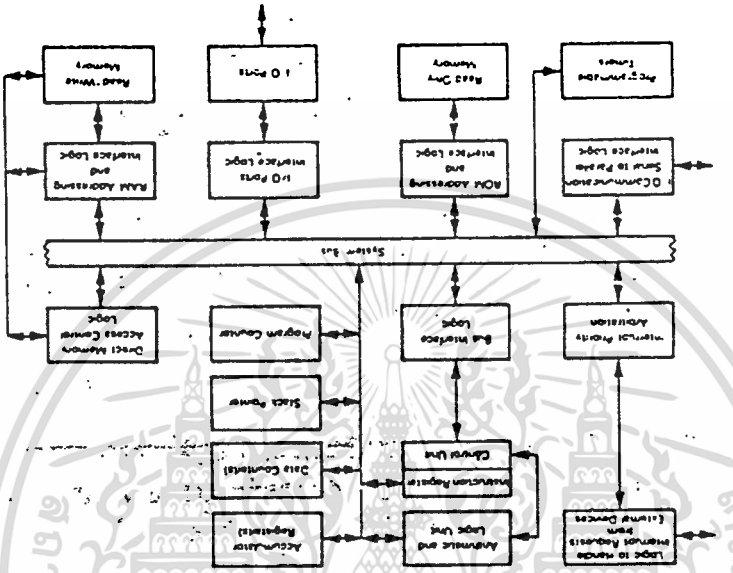
จะแสดงถึงโครงสร้างพื้นฐานของไมโครคอมพิวเตอร์ทั่ว ๆ ไป และเราสามารถอธิบายถึงส่วนประกอบของไมโครคอมพิวเตอร์ที่สำคัญแยกเป็นส่วน ๆ ได้ 3 ส่วน คือ

1.) ไมโครโปรเซสเซอร์

ส่วนไมโครโปรเซสเซอร์นี้ ถือว่าเป็นหัวใจและสมองของระบบคอมพิวเตอร์จะทำหน้าที่ตามคำสั่งหรือโปรแกรมที่ผู้ใช้เขียนขึ้นโดยมีซีพียู (CPU : CENTRAL PROCESSOR UNIT) เป็นตัวควบคุมการทำงานของอุปกรณ์อื่น ๆ ที่ทำหน้าที่ต่างกันให้ทำงานสอดคล้องกัน ซีพียูนี้จะทำหน้าที่จัดลำดับการทำงาน ก่อน--หลัง ตามความสำคัญของคำสั่งของโปรแกรม ซึ่งขณะทำงานโปรแกรมเหล่านี้จะถูกเก็บไว้ในหน่วยความจำ (MEMORY UNIT) ซึ่งมีสายข้อมูล (DATA BUS) และสายตำแหน่ง (ADDRESS BUS) ที่ต่อเข้ากับสายข้อมูลและสายตำแหน่งของไมโครโปรเซสเซอร์ ในรูปที่ 2.4 แสดงโครงสร้างของไมโครโปรเซสเซอร์

ภายในไมโครโปรเซสเซอร์จะมีส่วนประกอบสำคัญอยู่ 3 ส่วนคือ

1.1 <u>หน่วนคำนวน</u> (ALU : A RITHERMATIC AND LOGICAL UNIT) เป็นหน่วยที่ทำหน้าที่คำนวน ผลทางคณิตศาสตร์และตรรกศาสตร์เช่นการบวก, ลบ, AND หรือ ORกันในแต่ละบิทของข้อมูล ซึ่งผลลัพธ์ของการคำนวนทุกครั้งจะถูกเก็บเอาไว้ในหน่วยความจำชั่วคราวหรือรีจิสเตอร์ (REGISTOR) เรียกว่าแอคคิวมูเลเตอร์ (ACCUMULATOR : A REGISTOR) และจะแสดงสภาวะของผลลัพธ์ที่คำนวนได้ที่ แฟลกรีจิสเรอร์ (FLAG : F REGISTOR)



<u>รูปที่ 2.4 โครงสร้างภายในของไมโครโปรเซสเซอร์เบอร์ Z - 80</u>

1.2 <u>รีจิสเตอร์ที่ใช้เก็บข้อมูล</u> (DATA REGISTOR) เป็นรีจิสเตอร์ที่ใช้สำหรับเก็บข้อมูลชั่วคราวหลังจากทำการคำนวนหรือเป็นที่รอพักข้อมูล ระหว่างการย้ายตำแหน่งภายในของ CPU เองหรือหน่วยความจำอื่น ๆ CPU เบอร์ Z - 80 นั้นจะมีรีจิสเตอร์ที่ใช้เก็บข้อมูลหลัก ๆ คือ AF, BC, DE และ HL จะเป็นรีจิสเตอร์ ขนาด 16 บิท (BIT) สามารถใช้แยกกันทีละ 8 บิทได้ รีจิสเตอร์หลักทั้งหมดนี้ จะมีบัสข้อมูล (DATA BUS) 8 สาย ต่อถึงกันหมดและยังต่อกับบัสข้อมูลขนาด 8 บิทจากภายนอกอีกด้วย

1.3 <u>รีจิสเตอร์ที่ใช้เก็บข้อมูล</u> (DATA REGISTOR) เป็นรีจิสเตอร์ที่ใช้สำหรับเก็บข้อมูลชั่วคราวหลังจากทำการคำนวน หรือเป็นที่รอพักข้อมูลระหว่างการย้ายตำแหน่งภายในของ CPU อ้างถึงตำแหน่งของอุปกรณ์เหล่านี้ ได้รีจิสเตอร์ที่ใช้เก็บตำแหน่งข้อมูลใน Z-80 คือโปรแกรมเคาท์เตอร์ (PC : PROGRAM COUNTER) และแสตคพอยท์เตอร์ (SP : STACK POINTER)

1.4 หน่วยควบคุม (CONTROL UNIT)ถือว่าเป็นหัวใจของขบวนการทั้งหมด โดยจะส่งสัญญาณคอยควบคุมให้จังหวะแก่หน่วยอื่น ๆ ให้ทำงานไม่ซ้ำซ้อนกัน หน่วยควบคุมจะรับคำสั่งมาจากหน่วยความจำและแปลคำสั่งนั้น แล้วส่งสัญญาณเท่าที่จำเป็นไปควบคุมหน่วยอื่น ๆ ให้เป็นไปตามคำสั่ง หน่วยควบคุมจะรู้ว่าคำสั่งต่อไปอยู่ที่ไหน ในหน่วยความจำ โดยมีตัวบอกตำแหน่งคือโปรแกรมเคาท์เตอร์

1.5 สัญญาณนาฬิกา (CLOCK) เป็นฐานเวลาที่ป้อนให้กับขาล็อคของ CPU แล้ว CPUจะใช้ฐานเวลานี้เป็นตัวควบคุมจังหวะการทำงาน ความถี่ของสัญญาณนาฬิกานี้จะเป็นตัวกำหนดความเร็วในการทำงานของไมโครโปรเซสเซอร์ ซึ่งในปริญญานิพนธ์ฉบับนี้ใช้แบบซิงเกิลบอร์ด

2 หน่วยความจำ (MEMORY UNIT)

หน่วยความจำเป็นหน่วยที่เก็บคำสั่งและข้อมูลต่าง ๆ ทั้งหมด ขนาดของหน่วยความจำจะขึ้นอยู่กับจำนวนตำแหน่งที่หน่วยความจำจะสามารถเก็บข้อมูลได้ หน่วยความจำแบ่งออกเป็น 2 ชนิด คือ

2.1 รอม (ROM : READ ONLY MEMORY) การใช้งานในคอมพิวเตอร์นั้น ข้อมูลบางอย่างเรานำไปเก็บไว้เพียงครั้งเดียวเท่านั้น หลังจากนั้นก็อ่านออกมาใช้งานแต่เพียงอย่างเดียวหน่วยความจำที่เหมาะกับงานนี้ก็ควรจะเป็นแบบเขียนได้และไม่สูญหายไปแม้จะไม่มีไฟเลี้ยงก็ตามซึ่งก็คือรอมนั่นเอง มีอยู่ 3 แบบคือ

- รอม เป็นหน่วยความจำที่โปรแกรมมาจากโรงงานผู้ผลิตเลย
- พรอม (PROM) เป็นหน่วยความจำที่ผู้ใช้งานมาโปรแกรมเองตอนจะใช้งานตามต้องการ
- อีพรอม (EPROM) เป็นหน่วยความจำที่ผู้ใช้งานมาโปรแกรมเอง แต่สามารถลบออกได้ด้วยวิธีการอันเหมาะสม เช่น การฉายแสงอุลตร้าไวโอเลต

หน่วยความจำแบบ ROM นี้จะใช้เก็บโปรแกรมสำหรับระบบโปรแกรมโมนิเตอร์

2.2 แรม (RAM : READ ACCESS MEMORY) เป็นหน่วยความจำที่สามารถอ่านและเขียนข้อมูลลงไปได้มีข้อดีคือใช้แรมในการพัฒนาโปรแกรมเพราะสามารถเปลี่ยนแปลงแก้ไขได้ง่าย เมื่อพัฒนาโปรแกรมเสร็จแล้วจึงจัดเก็บ ข้อมูลลงไปในรอมอีกทีเพื่อเป็นการเก็บข้อมูลอย่างถาวรต่อไป ข้อเสียคือขณะที่ใช้งานอยู่ถ้าเกิดไฟดับจะทำให้ข้อมูลต่าง ๆ ถูกลบไปด้วย แต่จะมีแรมอีกแบบหนึ่งคือ แรมแพค

(RAM PACK) ซึ่งสามารถเก็บข้อมูลไว้ได้เมื่อไฟดับไป เพราะแรมชนิดนี้จะมีแบตเตอรี่คอยจ่ายไฟเลี้ยงไว้จนกว่าแบตเตอรี่จะหมดไป

3 อุปกรณ์อินพุท - เอาท์พุท (INPUT-OUTPUT PORT)

เป็นอุปกรณ์ภายนอกที่ทำหน้าที่ติดต่อกับไมโครคอมพิวเตอร์ อุปกรณ์อินพุทก็คือแหล่งที่ส่งข้อมูลให้กับไมโครโปรเซสเซอร์อุปกรณ์เอาท์พุทก็คือแหล่งที่รับข้อมูลมาจากไมโครโปรเซสเซอร์โดย CPU นะส่งสัญญาณเลือกอุปกรณ์ตัวใดตัวหนึ่งพร้อมกับข้อมูลที่จะส่งหรือรับ สำหรับอุปกรณ์ที่ใช้ในโปรเจ็คนี้คือ Z - 80 และ 8255 ซึ่งจะเชื่อมต่อกับ CPU และเชื่อมต่อกันเองด้วยบัสต่าง ๆ ดังนี้คือ

3.1 แอดเดรสบัส (ADDRESS BUS) เป็นบัสทางเดียวใช้ส่งผ่านค่าแอดเดรสจาก CPU ออกไปจากหน่วยความจำ เพื่อระบุตำแหน่งที่ต้องการรับหรือส่งข้อมูลหรือใช้ระบุตำแหน่งของพอร์ท อินพุท - เอาท์พุท ที่ CPU ต้องการจะติดต่อด้วย

3.2 บัสควบคุม (CONTROL BUS) เป็นบัสทางเดียวที่ใช้ในการส่งผ่านสัญญาณควบคุมให้กับอุปกรณ์ต่าง ๆ ในระบบ

3.3 บัสข้อมูล (DATA BUS) เป็นบัส 2 ทิศทางที่ใช้ในการส่งผ่านข้อมูลระหว่าง CPU กับ อุปกรณ์อื่น ๆ ใน ระบบ จำนวนเส้นของบัสข้อมูลจะขึ้นอยู่กับชนิดของ CPU ในกรณีของ Z - 80 นั้น CPU จะส่งผ่านข้อมูลที่ละ 8 บิท ดังนั้น จะมีจำนวนเส้นของบัสข้อมูลอยู่ 8 เส้น

โปรแกรมคอมพิวเตอร์หรือคอมพิวเตอร์ซอฟท์แวร์ (COMPUTER SOFTWARE)

การเขียนโปรแกรม ให้คอมพิวเตอร์ทำงานก็คือ การจัดคำสั่งแต่ละคำสั่งให้เป็นไปตามลำดับก่ารทำงานที่ต้องการ ไมโครโปรเซสเซอร์จะรับรู้คำสั่งต่าง ๆ ในรูปรหัสเลขฐานสองเท่านั้นซึ่งถ้าเราจะเขียนโปรแกรมในรูปเลขฐานสองก็จะทำให้เกิดความยุ่งยากและอาจผิดพลาดได้ง่าย จึงได้มีการกำหนดภาษาขึ้นใหม่เพื่อนำมาใช้สะดวกมากขึ้นคือ ภาษาแอสแซมบลี้ (ASSEMBLY LANGUAGE) ซึ่งสะดวกต่อการอ่านและเข้าใจมากขึ้น เพราะจะเปลี่ยนรหัสจากเลขฐานสองเป็นสัญญลักษณ์ภาษาอังกฤษว่า นีโมนิค (MEMONIC) แต่โปรเจคนี้ใช้ชิงเกิลบอร์ด ในการทำงานจะต้องแปลงภาษาแอสแซมบลี้เป็นอ๊อฟโค๊ด

(OPCODE) ก่อนจึงจะป้อนโปรแกรมให้เครื่องรับรู้ได้ เช่น ในคำสั่งภาษาแอสแซมบลี้เขียนว่า

LD A, FFH

ซึ่งจะแทนรหัสตัวเลขฐานสองคือ

0011 1110 1111 1111

และเขียนเป็นอ๊อฟโค้ดได้ว่า

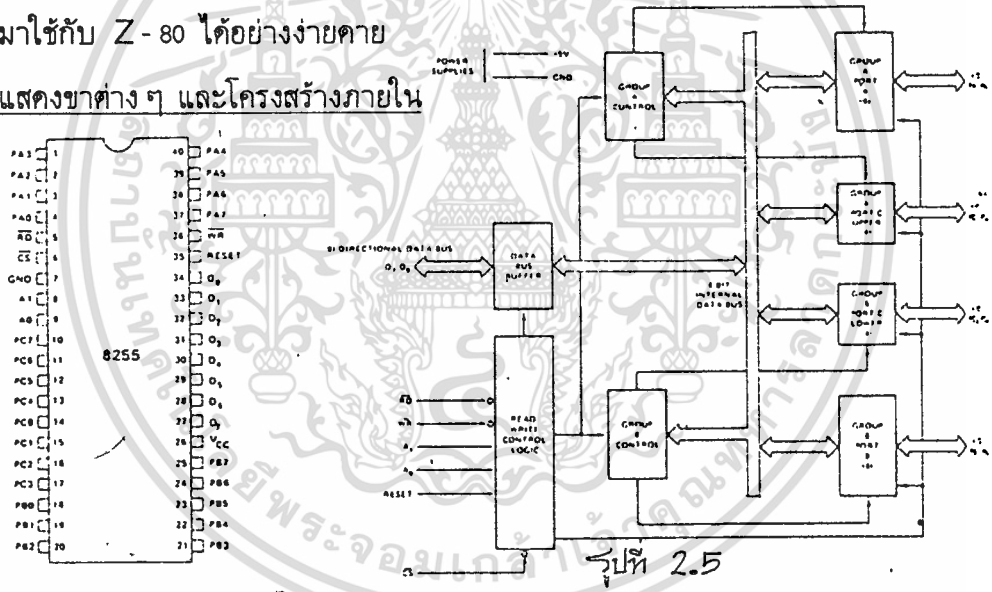
3 EFF

ดังนั้นในการเขียนโปรแกรมให้กับซิงเกิลบอร์ด จึงต้องเขียน 3 EFF ลงไปเท่านั้นเครื่องจึงจะ ปฏิบัติตามคำสั่งไมโครโปรเซสเซอร์ แต่ละตัวจะต้องมีภาษาแอสแซมบลี้เป็นรหัสอ๊อฟโค้ดโดยเรียกตัวแปลภาษานี้ว่า แอสแซมเบลอร์ (ASSEMBLER PROGRAM)

8255

เป็น IC PORT ที่สามารถโปรแกรมการทำงานได้ซึ่งถูกสร้างขึ้นมาใช้กับ 8080 แต่ก็สามารถนำมาใช้กับ Z - 80 ได้อย่างง่ายดาย

รูปแสดงขาต่าง ๆ และโครงสร้างภายใน



รูปที่ 2.5

จะเห็นว่าตัว IC มีอยู่ 3 PORT ที่ใช้งาน คือ PORT A PORT B มีขนาด 8 BIT และ PORT C โดยที่ PORT C นั้นยังสามารถแบ่งกลุ่มออก เป็น PORT และ 4 BIT จึงทำให้มี PORT C บน และ PORT C ล่าง คราวนี้การต่อใช้งานก็จะมีขาที่เป็นอินพุท ให้กับตัว IC ก็มี

DO-D7 ต่อเข้ากับ ดาตาบัส ของซีพียู เพื่อใช้สำหรับรับส่งข้อมูลกันระหว่าง PORT กับ CPU

AO-A1 ขา แอดเดรส ซึ่งเป็นตัวสำคัญในการกำหนด PORT ว่าเรียก PORT อะไรเป็น PORT A B หรือ C จากที่กล่าวมาแล้วสถานะที่เราคิดมีเพียง ON กับ OFF ดังนั้น IC ตัวนี้จึงมีเบอร์ PORTในตัวมัน 4 PORT เพราะที่สาย แอดเดรส 2 เส้น

2^2 และเราได้กล่าวมาแล้ว 3 Port ดังนั้นจึงเหลืออีก Port หนึ่งซึ่ง Port ตัวนี้จะเป็นตัวที่สำคัญที่สุดในการทำงานของ IC ตัวนี้ ซึ่งก่อนที่จะให้ IC ตัวนี้มีหน้าที่ของ IC ให้กับ Port นี้เสียก่อนเรียก Port นี้ว่า Control Port ซึ่งจะมีการเรียงลำดับดังนี้

A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
ไม่สนใจ						0	0	Port A
ไม่สนใจ						0	1	Port B
ไม่สนใจ						1	0	Port C
ไม่สนใจ						1	1	Port Control

ดังนั้นเวลาเราเรียก Port ซึ่งใน 1 คำสั่งนั้นจะต้องเรียกเบอร์ Port เป็น 8 BIT คือเลข HEX 2 หลัก แต่ IC จะให้ Port ไหนทำงานจะมีความสำคัญแค่หลักหลังคือ A_1 กลับ A_0 ว่ามีค่าเป็นอะไร

อย่างเช่น บน ET-Board ที่ Decode ที่ 8255 ว่างไว้คือ Port 20H ถ้าเราเรียก Port 21H บ้างก็เป็น Port B เพราะ A_1 เป็น 0 และ A_0 เป็น 1 หรือถ้าเราเรียก Port เบอร์ 24H บ้าง Port ที่ทำงานก็คือ Port A เพราะ A_1 กับ A_0 เป็น 0 ดังนั้น เวลาเราเรียก Port 20H กับ 24H จึงเป็น Port เดียวกันบน ET-Board เพราะ 1 จุดนั้นอ้างถึง 32 Port คือ 20H-3FH หรือสังเกตอีกอย่างคือ มันจะทับซ้อนตัวเองเพราะตัว IC อ้างได้ 4 Port มันก็จะกลับไปเริ่มต้นใหม่อย่าง 20H อ้างถึง 4 Port ก็จะถึง 23H พอเพิ่มอีก 1 มันก็จะเกิน 4 Port คือ Port 24H ก็เป็น 20H นั้นเองอย่างเช่น Port 20H, 24H, 28H, 30H ทั้งหมดนี้ก็คือ Port เดียวกัน (Port A) ในจุด Decode 1 จุดบน ET-Board นี้

$\overline{CS}$ เป็นขาเลือก IC Port ให้ทำงานนี้จะต่อเข้ากับ IC ที่ Decode เบอร์ Port ไว้โดยการเรียกเบอร์ Port นี้ จะรวมเข้ากับแอดเดรส 2 เส้นที่ต่อเข้ากับ Port ด้วยคือ A_0 กับ A_1 เพราะเวลาที่เรียกเบอร์ Port ต้องใช้คำสั่งซึ่งเป็น 8 Bit ตาม CPU ดังนั้น 1 คำสั่งจึงรวมสายแอดเดรสต่ำอย่างบน ET-Borad 8255 ที่ว่างขา CS จะต่อเข้ากับ IC 74LS138 ขา 14 คือจุดที่ Decode เบอร์ Port ตั้งแต่เบอร์ 20H ไว้นั้นเอง (Port A)

$\overline{RD}$ ใช้ขบวนการ อินพุท เมื่อ CS และ RD Active เป็น 0

$\overline{WR}$ ใช้ขบวนการ เอาท์พุท เมื่อ CS และ WR Active เป็น 0

Reset เป็น 1 ใช้ Clear สถานะต่างๆของ 8255

Port ที่ใช้สำหรับ Control

การใช้ 8255 จะต้องส่งรหัสควบคุม (Control BYTE) เข้าไปยัง Port ข้อมูลควบคุม (Port สุดท้ายใน 4 Port คือที่ A_1 กับ A_0 เป็น 1 เช่น Port 23H บน ET-Board ที่ 8255 ว่าง) เพื่อควบคุมการทำงานของ 8255 ว่าให้ทำงานใน Mode ไหนและให้แต่ละ Port เป็น Input หรือ Output Port

ความหมายของ Bit ต่างๆของรหัสควบคุม (Control Byte) หรือรหัสสั่งงาน 8255 ในตอนเริ่มแรก

ความหมายแต่ละ BIT

D_7 แสดงถึงรหัสควบคุมให้เริ่มทำงาน (1 - ทำงาน) 8255 รับรู้สิ่งต่อไปใน BIT ต่างๆที่กำหนดให้เพราะฉะนั้นเวลาจะสั่งงานหรือหน้าที่ให้กับ 8255 BIT นี้จะเป็น 1 เสมอ

D_6 & D_5 เป็นการเลือก Mode ในการทำงานของ Port A ซึ่งมี 3 Mode ใน 8255 จะได้กล่าวต่อไป

D_4 กำหนดให้ Port A เป็น อินพุท หรือ เอาท์พุท โดย
0 - เอาท์พุท Port
1 - อินพุท Port

D_3 กำหนดให้ Port C บนเป็น อินพุท หรือ เอาท์พุท โดย
0 - เอาท์พุท Port
1 - อินพุท Port

D_2 เป็นการเลือก Mode ให้กับ Port B
0 - Mode 0
1 - Mode 1

D_1 กำหนดให้ Port B เป็น อินพุท หรือ เอาท์พุท โดย
0 - เอาท์พุท
1 - อินพุท

D_0 กำหนดให้ Port C ล่าง (PC_0-PC_3) เป็น อินพุท หรือ เอาท์พุท โดย
0 - เอาท์พุท
1 - อินพุท

แบ่งออกเป็น 3 Mode

Mode 0 กำหนดให้ Port บน ET-Board ทุก Port เป็น Port เอาท์พุท และอินพุท แบบพื้นฐาน

Mode 1 การทำงานของ 8255 ใน Mode นี้เป็น Mode ที่ทำให้ อินพุท-เอาท์พุท มีการตรวจสอบสัญญาณ (Hand Shaking) โดยใช้ อินพุท-เอาท์พุทของ Port A และ B เป็นหลัก และใช Port C บน เป็นสัญญาณ Hand Shake ของ Port A ส่วน Port C ล่างเป็นสัญญาณ Hand Shake ของ Port B

เมื่อโปรแกรม 8255 เป็นโหมด1 Port C จะมีความหมายดังนี้

ขา	กรณีอินพุท	กรณีเอาท์พุท
PC_0	$INTR_B$	$INTR_B$
PC_1	IBF_B	OBF_B
PC_2	STB_B	ACK_B
PC_3	$INTR_A$	$INTR_A$
PC_4	STB_A	I/O
PC_5	IBF_A	I/O
PC_6	I/O	ACK_A
PC_7	I/O	OBF_A

จะเห็นว่าในกรณี I/P PC_6 และ PC_7 สามารถที่จะโปรแกรมให้เป็น อินพุท หรือ เอาท์พุท อิสระได้โดยถ้าให้ BIT มีค่าเป็น 1 ก็จะเป็น อินพุท ส่วนถ้าเป็น 0 ก็จะเป็น เอาท์พุท ส่วน เอาท์พุท O/P ก็เช่นกันเพียงแต่ BIT ที่ใช้ได้ไปอยู่ที่ PC_4 และ 5 แทน และยังสามารถส่งสัญญาณไปอิทเตอร์รัพท์Set Ensable Mode 1

Port	กำหนดให้เป็น	สัญญาณ INT	ทำการที่ Set/Reset
A	Input	INTE A	PC_4
A	Output	INTE A	PC_6
B	Input	INTE B	PC_2
B	Output	INTE B	PC_2

การ Set/Reset นี้ให้ทำการ Set ไปที่ Control Word ในลักษณะเดียวกับการ Set/Reset BIT ที่ Port C ที่กล่าวมาแล้ว

ใน Mode ของการ Set/Reset BIT INTR จะถูก Reset โดยอัตโนมัติเมื่ออยู่ในระหว่างการเลือก Mode หรือ Reset

Mode 2

ใช้ได้เฉพาะ Port A เท่านั้นซึ่งจะทำหน้าที่เป็น Port แบบ 2 ทิศทาง คือ สามารถเป็นได้ทั้ง อินพุท และ เอาท์พุท โดยโครงสร้างของ Port A ทั้ง อินพุท และ เอาท์พุท จะมี Hand Shake ทั้งคู่ ส่วน Port C ทำหน้าที่สำหรับตรวจสอบ

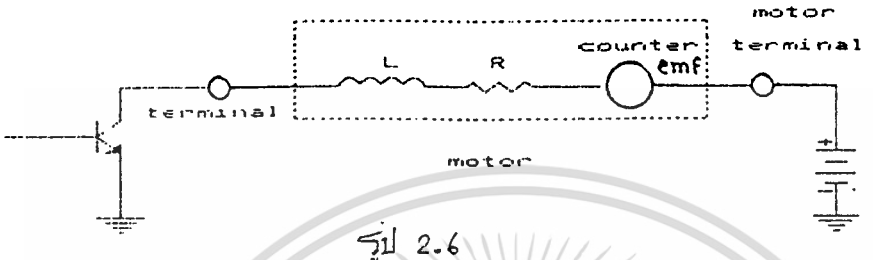
Port C	ความหมาย
PC 0	I/O
PC 1	I/O
PC 2	I/O
PC 3	INTRA
PC 4	STBA
PC 5	IBFA
PC 6	ACKA
PC 7	OBFA

ลักษณะการ Hand Shake ทาง อินพุท และ เอาท์พุท มีหลักการเดียวกับ Mode 1 และการสั่ง Enable INTE 1 เมื่อเป็น เอาท์พุท ให้ Set/Reset ที่ PC 6 และถ้าเป็น อินพุท Enable INTE 2 Set/Reset ที่ PC 4

2.3 วงจร DRIVER และ SENSOR

ปัญหาเกี่ยวกับวงจร DRIVER

ขดลวดของ STEPPING MOTOR เป็น INDUCTIVE และมีค่าเปรียบเสมือนผลรวม ของอินดัคเตนซ์อนุกรมกับความต้านทานดังรูป



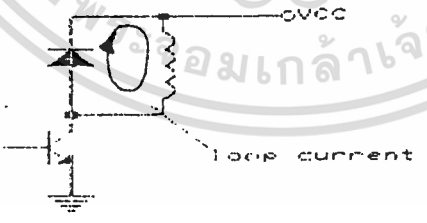
รูป 2.6

รูป A วงจรสมบูรณ์ของขดลวด STEPMOTOR

ซัพเพรสเซอร์

เมื่อ TRANSISTOR รูป A หยุดนำกระแสและทำให้เกิด VOLTAGE ค่าสูงจำนวนหนึ่งเนื่องจากผลของการเปลี่ยนแปลงของกระแสในอินดัคแตนซ์และ VOLTAGE นี้อาจจะเป็นอันตรายแก่ TRANSISTOR ได้ วิธีป้องกัน

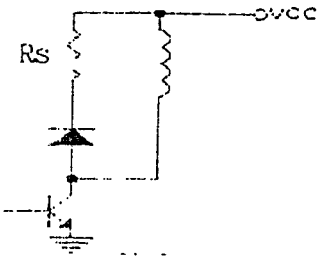
1.) ไดโอดซัพเพรสเซอร์ ดังรูป



รูป 2.7

กระแสหมุนเวียน CIRCULATING CURRENT จะเริ่มไหลหลังจาก TRANSISTOR หยุดนำกระแสและศักดา COLLECTOR จะเท่ากับศักดาของแหล่งจ่าย ข้อเสียคือ กระแสจะหมุนเวียนอยู่นาน และจะทำให้เกิดแรงบิดห้ามล้อขึ้น (BREAKING TORQUE พลังงานส่วนใหญ่สูญเสียในความต้านทานของขดลวด มีปัญหาเรื่องทำความเย็น

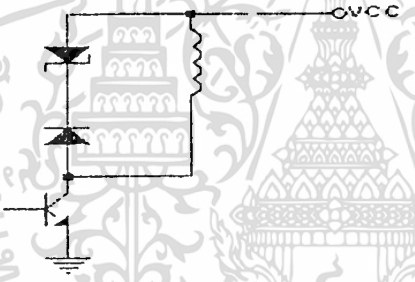
2.) ไดโอดและรีจีสเตอร์ซัพเพรสเซอร์ ดังรูป



รูป 2.8

ถ้า Rs ยิ่งมากกระแสหมุนเวียนก็จะลดลงเร็วขึ้น แต่ศักดาของ COLLECTOR จะมีค่าสูงขึ้น พลังงานส่วนใหญ่สูญเสียใน Rs

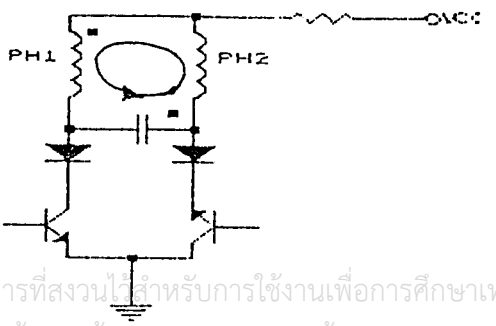
3.) ซีเนอร์ไดโอดซัพเพรสเซอร์ ดังรูป



รูป 2.9

เมื่อ TRANSISTOR CUTOFF กระแสจะลดลง ได้เร็วกว่า 2 แบบแรก และศักดาที่ COLLECTOR จะเท่ากับศักดาของซีเนอร์ บวกกับศักดาของแหล่งจ่าย ซึ่งเป็นอิสระต่อกระแสพลังงานส่วนใหญ่สูญเสียใน ZENER

4.) คอนเดนเซอร์ซัพเพรสเซอร์ ดังรูป



รูป 2.10

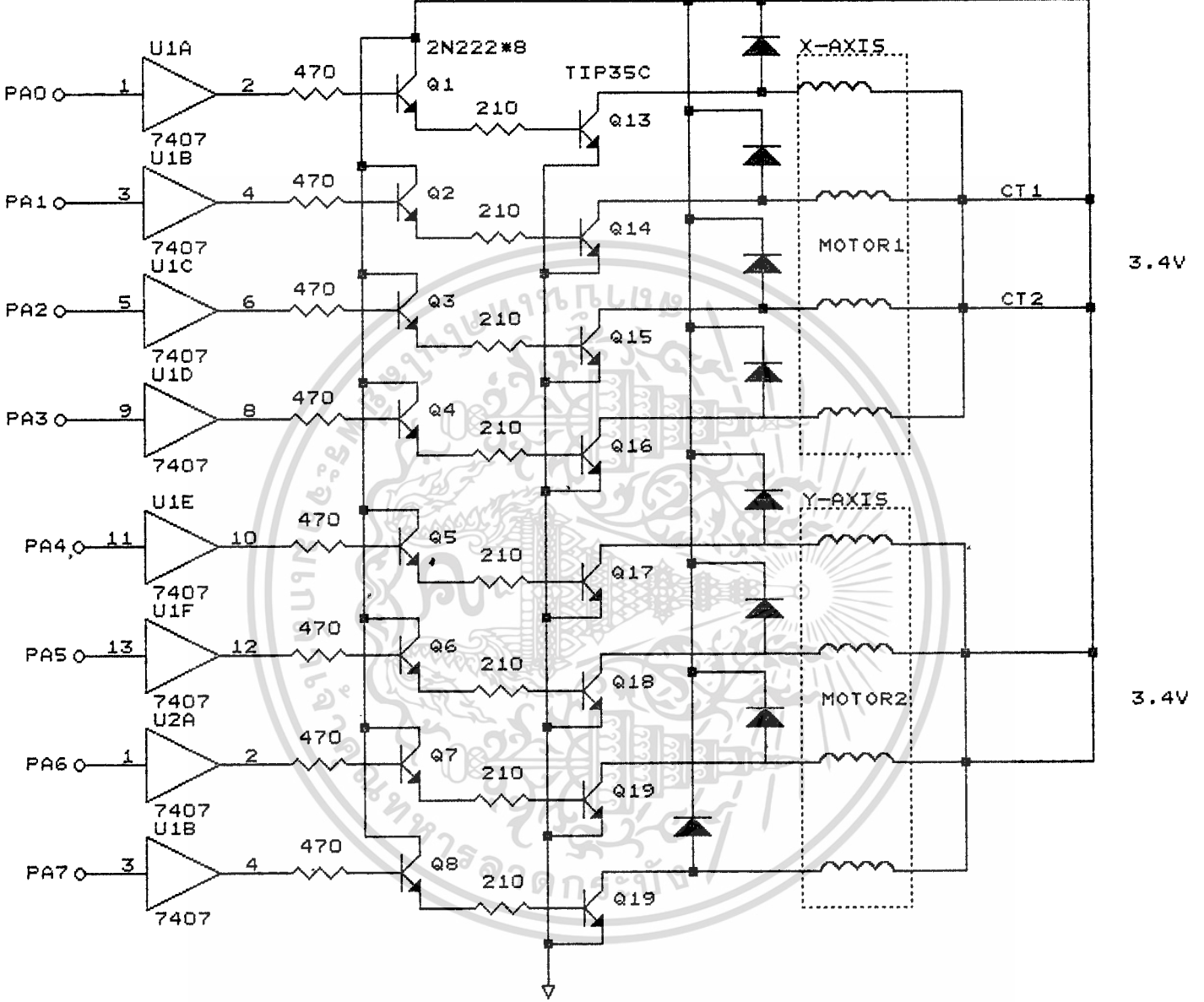
จะใส่ CONDENSER ให้ ϕ_1 กับ ϕ_3 และ ϕ_2 กับ ϕ_4 เมื่อ TR หยุดนำกระแส C จะต่อกับ TR โดยผ่านทาง DIODE และจะดูดกลืนกระแสที่ค่อย ๆ ลดลงจากขดลวดของ MOTOR เพื่อป้องกัน TR เสีย และยังช่วยแดมป์ไฟ คือช่วยลดความร้อนที่เกิดขึ้นในขดลวด STATOR เนื่องจากการ OSCILATE ของ ROTOR จากวงจรเราใช้ไดโอดซัพเพรสเซอร์เพราะต้องการให้ STEPPING MOTOR เกิดแรงบิดห้ามล้อเมื่อถึงตำแหน่งที่ต้องการ

สัญญาณที่ป้อนเข้าชุด DRIVER มาจาก พัลส์ที่สร้างขึ้นจาก SINGLE BOARD โดยกำหนดให้ PORT A และ PORT C เป็นเอาท์พุท และ PORT B เป็นอินพุท สัญญาณเอาท์พุทจะนำไปขับทรานซิสเตอร์เบอร์ 2 N 2222 ซึ่งจะขับทรานซิสเตอร์ เบอร์ TIP 35 C และ เบอร์ 2 N 1061 อีกครั้ง เมื่อทรานซิสเตอร์ทั้งสองตัวทำงานจะทำให้ขดลวดของ STEPPING MOTOR ต่อครบวงจร และเกิดการหมุนขึ้นเมื่อ STEP พัลส์ถูกป้อนเข้าสู่ขดลวด

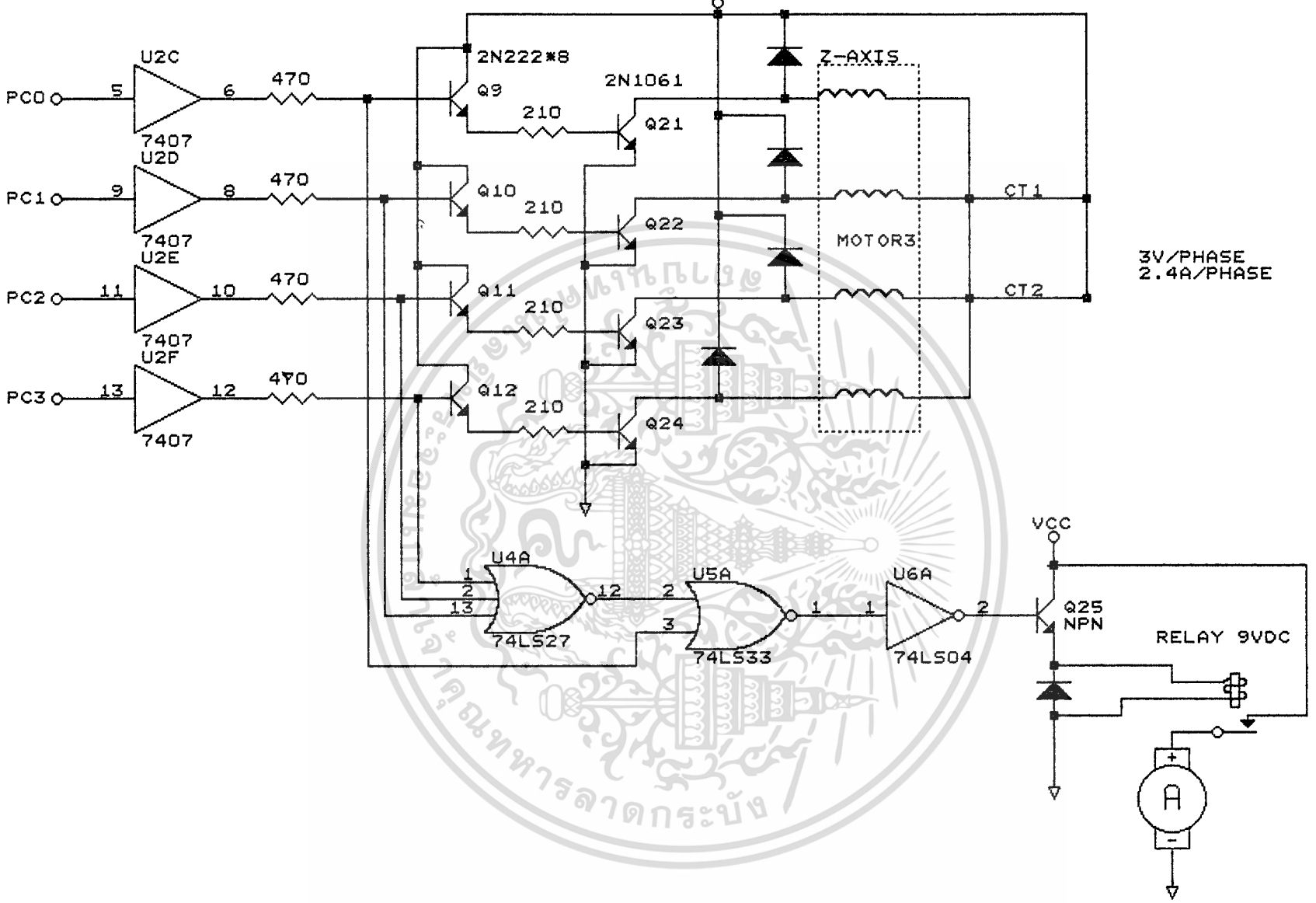
เมื่อ STEPPING MOTOR ตัวที่ 3 ที่ใช้ขับเคลื่อนแกน Z ทำงาน จะมีผลทำให้รีเลย์ทำงานด้วย และจะทำให้ DC MOTOR ทำงาน ซึ่งแกนของ DC MOTOR จะต่ออยู่กับหัวจับดอกสว่าน ทำให้เราสามารถที่จะเจาะแผ่นวงจรพิมพ์ตามที่ต้องการได้

ภาคตรวจจับสัญญาณ (SENSOR CIRCUIT) จะประกอบไปด้วย OPTICAL LIMIT SWITCH เบอร์ H 2181 ที่ใช้ในการกำหนดจุดเริ่มต้นและจุดสิ้นสุดของการเคลื่อนที่ของแกน X และ Y รวมทั้งการเคลื่อนที่กลับของแกน Z ไม่ให้เกินจุดขีดจำกัด สัญญาณที่ส่งออกมาจะถูกป้อนเข้าสู่ PORT B และ ป้อนเข้า NOR GATE เพื่อไปอินเตอร์รัพท์ซีพียูให้ทำการตรวจสอบแหล่งที่มาของสัญญาณ เมื่อได้ CODE ของสัญญาณแล้ว ซีพียูจะทำงานในคำสั่งต่อไปตามที่กำหนดไว้

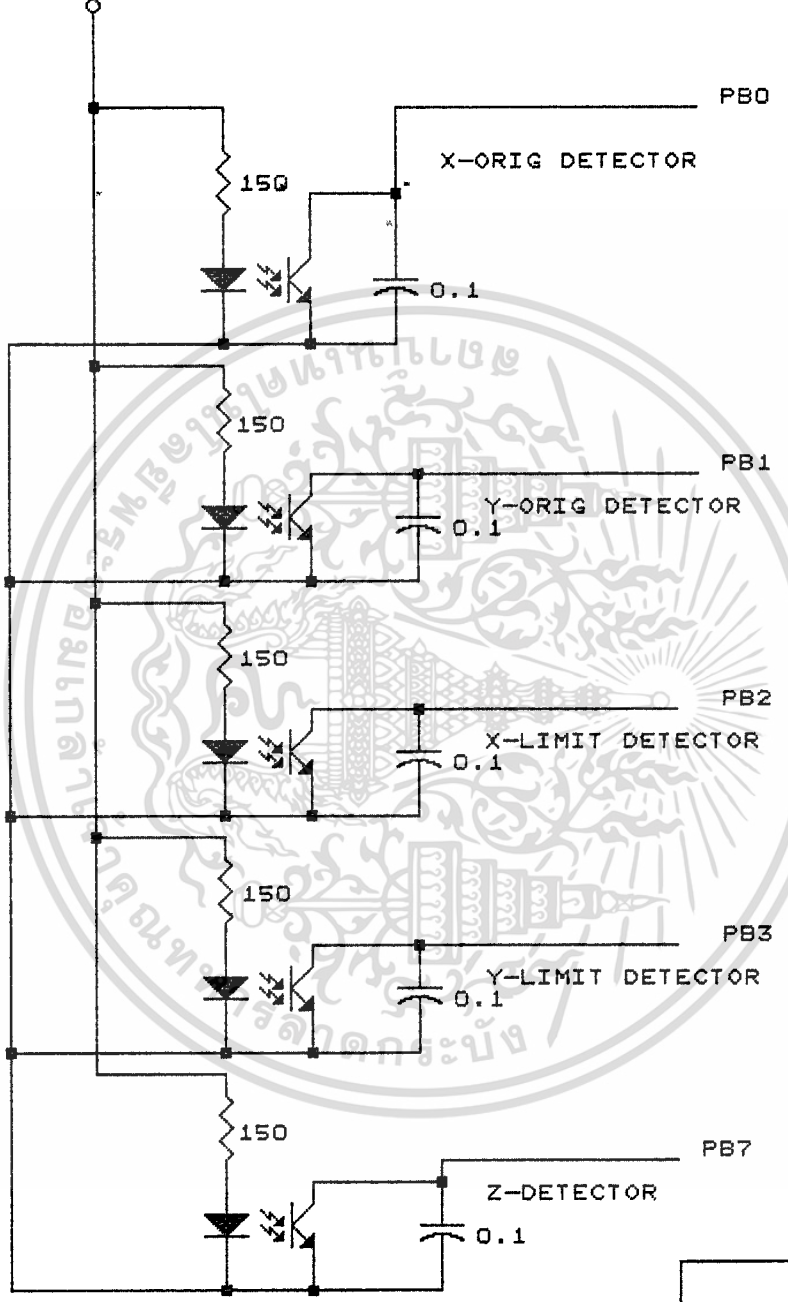
BLOCK DIAGRAM



DRILLING MACHINE		
Size A	Document Number X,Y-AXIS DRIVE	REV 5
Date: January 3, 1980	Sheet	of

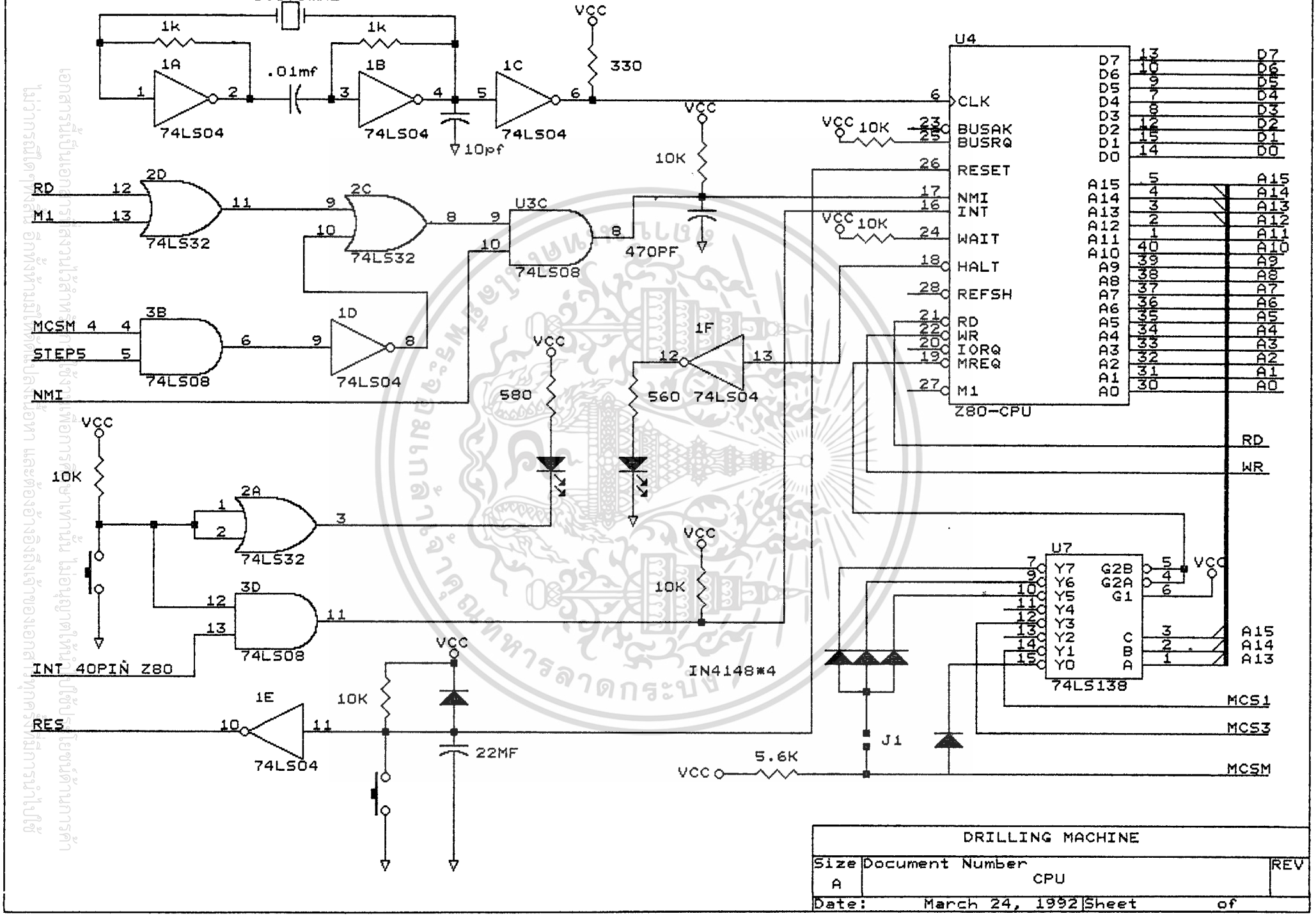


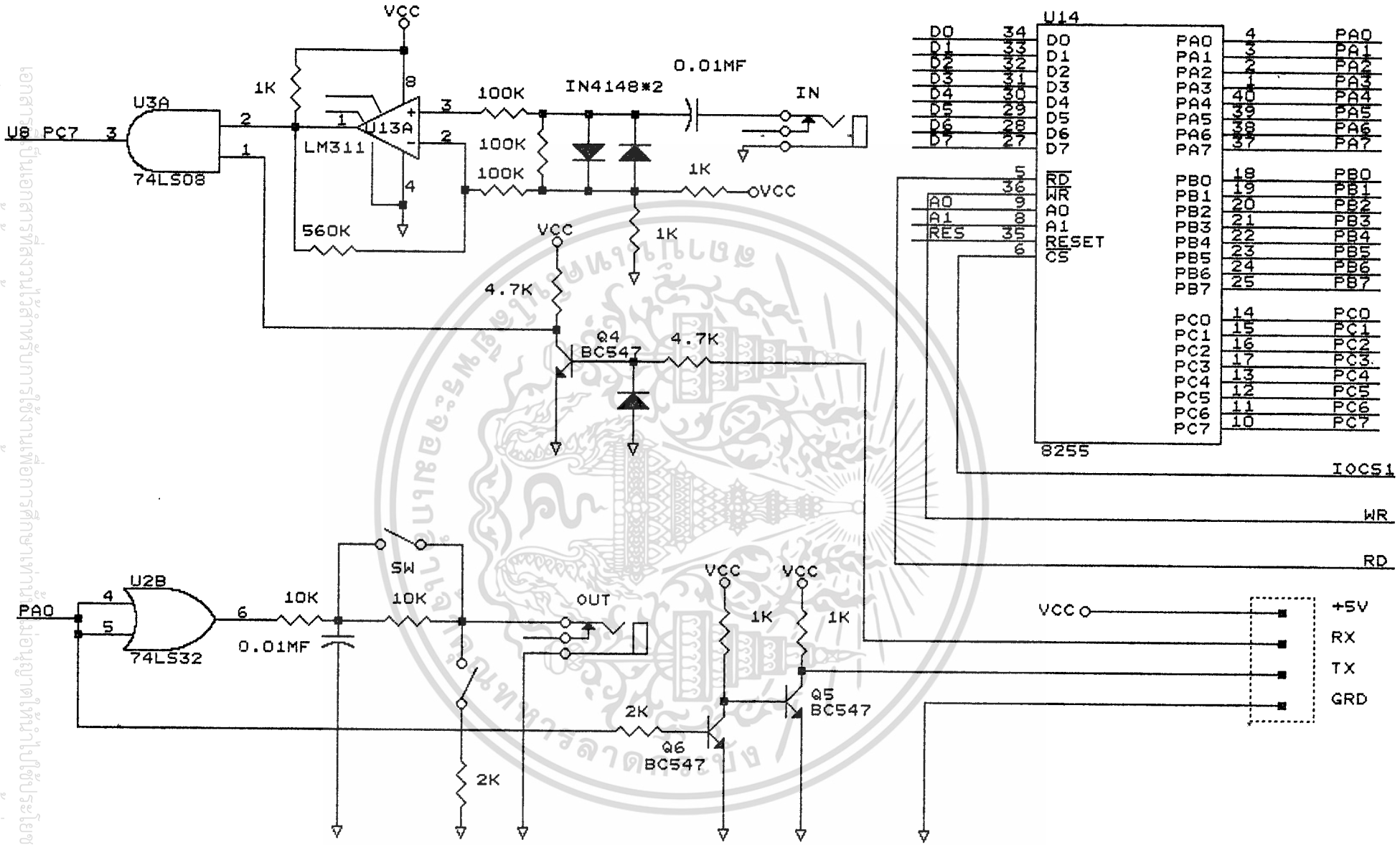
DRILLING MACHINE		
Size A	Document Number Z-AXIS DRIVE	REV
Date: March 24, 1992	Sheet of	



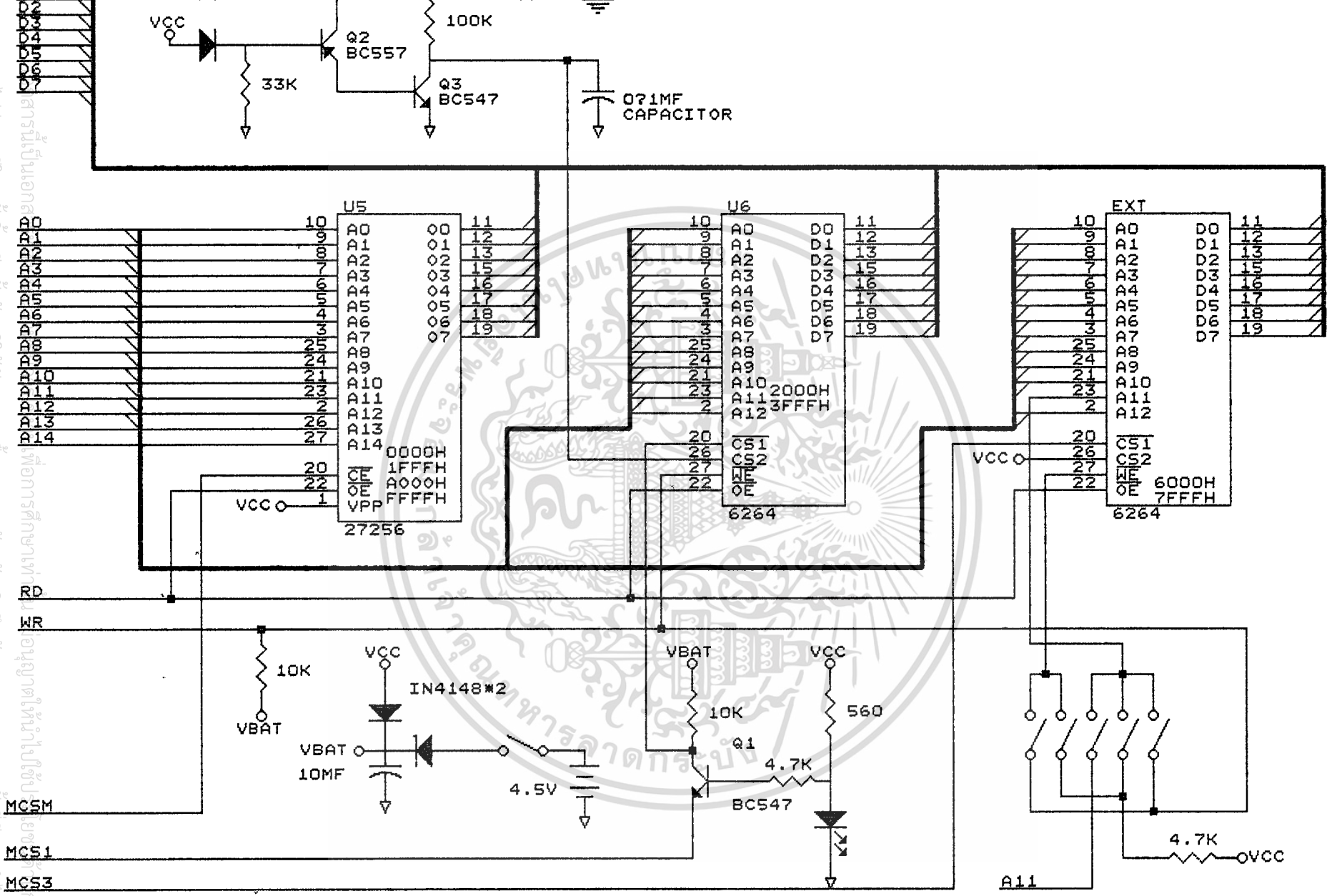
H2181*5

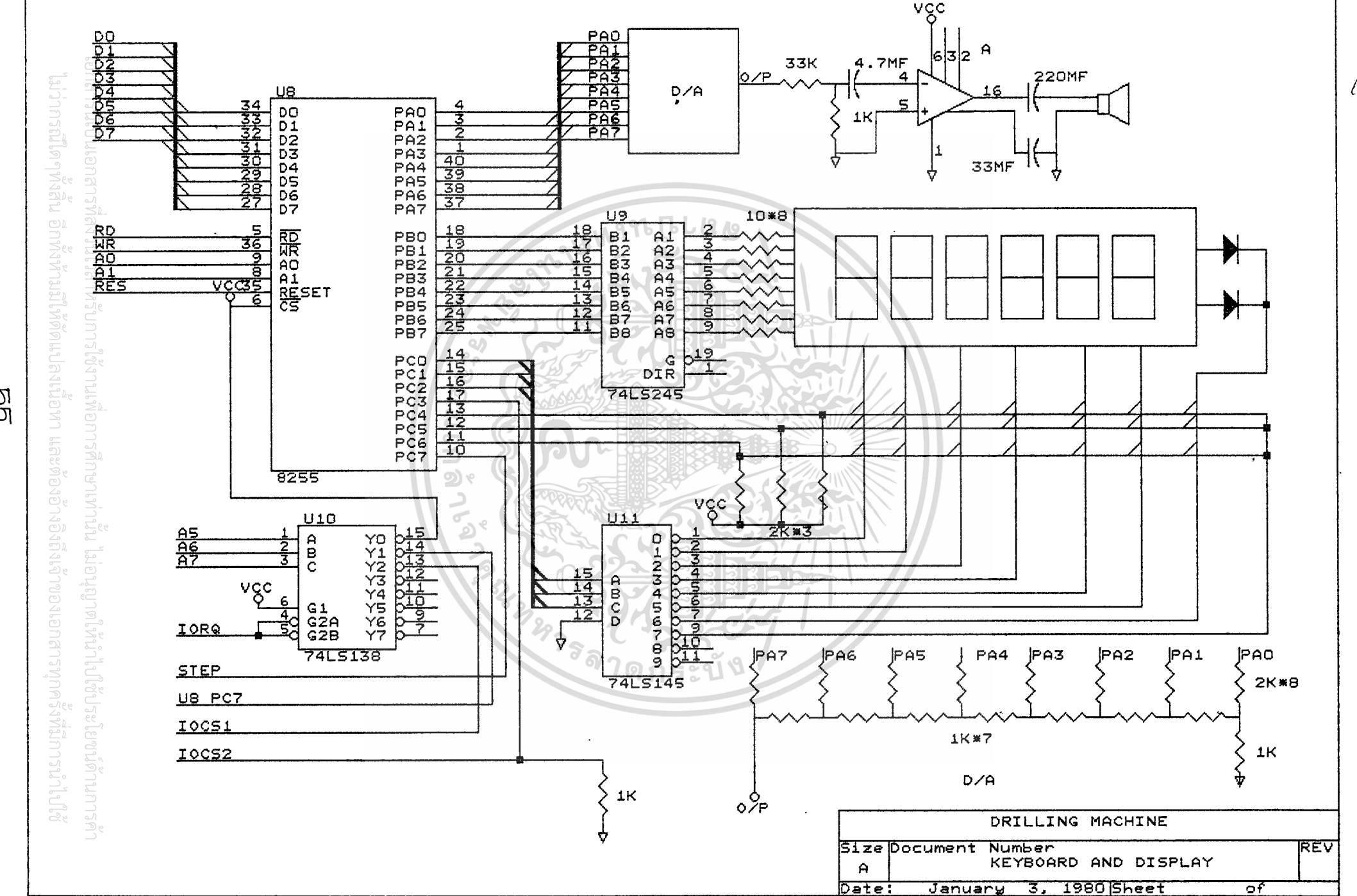
DRILLING MACHINE		
Size A	Document Number X,Y,Z DETECTOR	REV
Date: March 24, 1992	Sheet of	


1k
1k
1A
1B
1C
.01mf
74LS04
74LS04
74LS04
10pf
VCC
330
U4
CLK
BUSAK
BUSRQ
RESET
NMI
INT
WAIT
HALT
REFSH
RD
WR
IORQ
MREQ
M1
Z80-CPU
D7
D6
D5
D4
D3
D2
D1
D0
A15
A14
A13
A12
A11
A10
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
10K
470PF
RD
M1
2D
2C
U3C
74LS32
74LS32
74LS08
MCSM 4
STEP5
3B
1D
74LS08
74LS04
NMI
580
560
1F
74LS04
IN4148*4
10K
2A
74LS32
3D
74LS08
INT 40PIN Z80
RES
1E
74LS04
22MF
U7
Y7
Y6
Y5
Y4
Y3
Y2
Y1
Y0
G2B
G2A
G1
C
B
A
74LS138
MCS1
MCS3
MCSM
J1
5.6K
DRILLING MACHINE
Size A
Document Number CPU
REV
Date: March 24, 1992
Sheet of

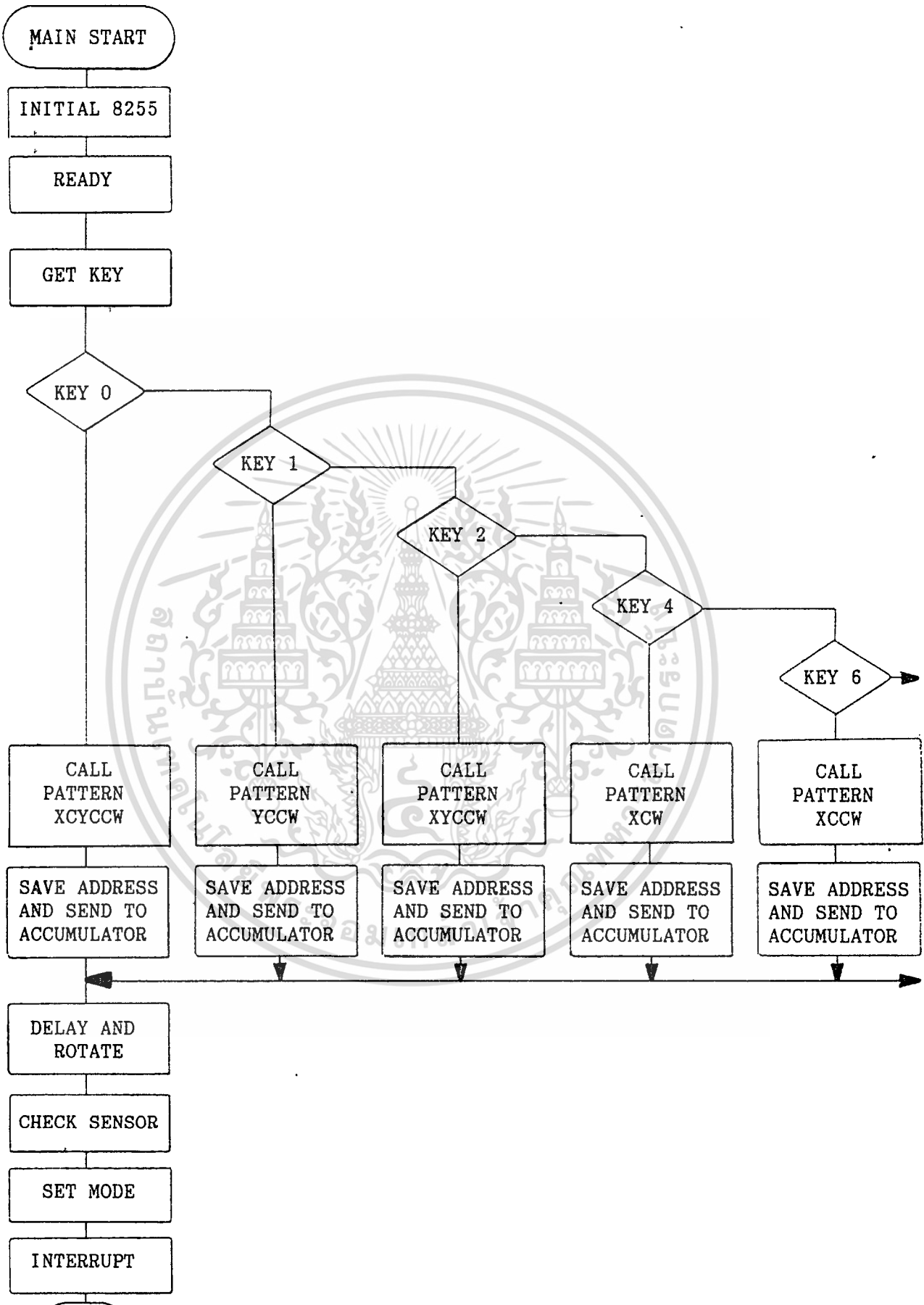


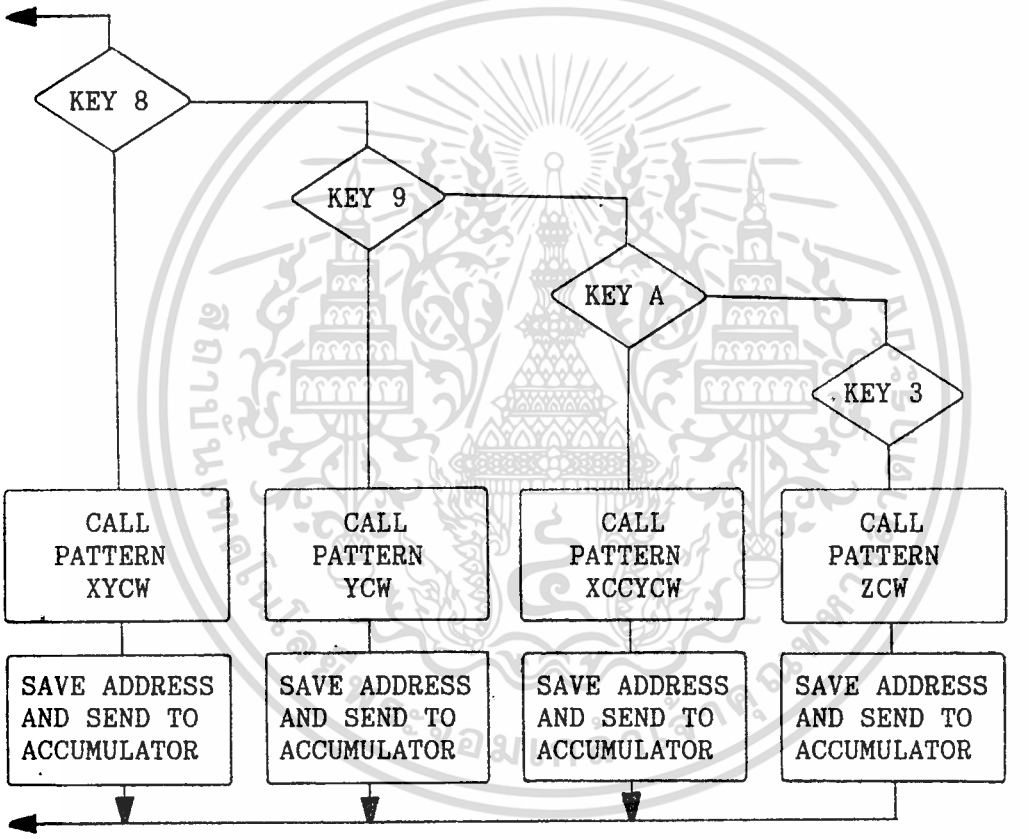
DRILLING MECHINE		
Size A	Document Number SERIAL PORT	REV 2
Date: January 6, 1980	Sheet of	

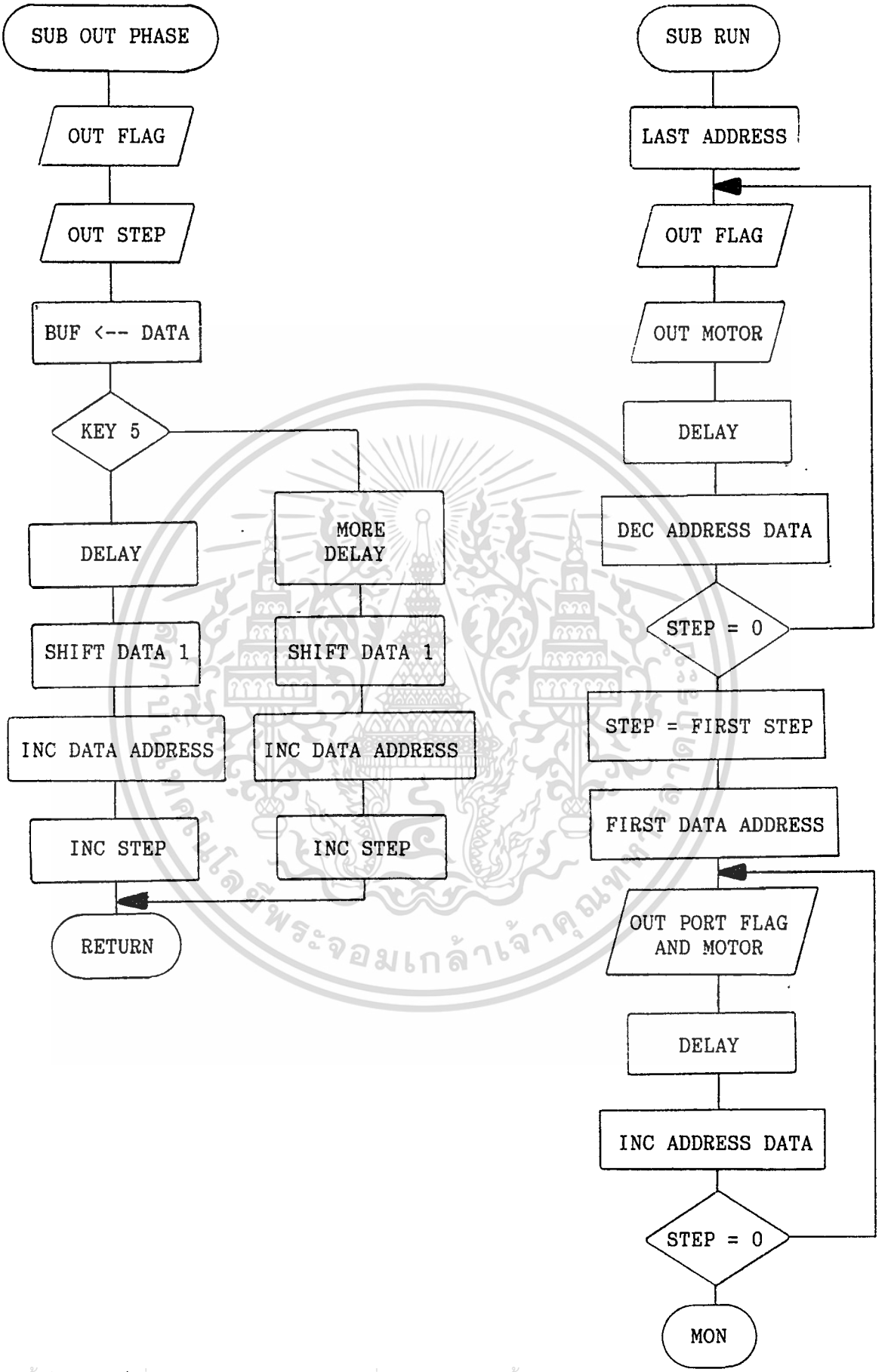

D2
D3
D4
D5
D6
D7
VCC
Q2
BC557
100K
33K
Q3
BC547
0?1MF
CAPACITOR
U5
A0
A1
A2
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
O0
O1
O2
O3
O4
O5
O6
O7
CE
OE
VPP
0000H
1FFFH
A000H
FFFFH
27256
U6
CS1
CS2
WE
2000H
3FFFH
6264
EXT
6000H
7FFFH
RD
WR
10K
VBAT
IN4148*2
10MF
4.5V
560
Q1
4.7K
BC547
MCSM
MCS1
MCS3
A11
DRILLING MACHINE
Size
A
Document Number
REV
Date:
March 24, 1992
Sheet
of


U8
8255
U9
74LS245
U10
74LS138
U11
74LS145
D/A
VCC
33K
4.7MF
220MF
33MF
1K
10*8
2K*3
2K*8
1K*7
STEP
U8 PC7
IOCS1
IOCS2
IORQ
RD
WR
RES
O/P
DRILLING MACHINE
KEYBOARD AND DISPLAY
Size A
Document Number
REV
Date: January 3, 1980
Sheet of

โฟลวชาร์ท







2500 A.D. Z80 Macro Assembler - Version 4.02a

Input Filename : C:\ET\PROCOMM\DRILL.asm
Output Filename : C:\ET\UTIL\DRILL.obj

```
                                ;**********************************
                                ;          PCB  DRILLING         *
                                ;             VER 1.0            *
                                ;       RUN ON Z80 SINGLE BOARD  *
                                ;       DATE  :  MARCH 15, 1992  *
                                ;       BY    :  PONGTEP  C.     *
                                ;                WICHAI   W.     *
                                ;                MONGKOL  S.     *
                                ;                EE 3 W , KMIT'L *
                                ;**********************************
                                ;
2000                            ORG   2000H
                                ;
0000                            MON     EQU   00H
0010                            SYSCAL  EQU   10H
2FFE                            SMBUF   EQU   2FFEH
3000                            SMDAT   EQU   3000H
3FFD                            SYSFAG  EQU   3FFDH
3FE7                            DISPY   EQU   3FE7H
0002                            SCAN    EQU   02H
0002                            DIGIT   EQU   02H
0001                            SEGM    EQU   01H
0003                            CLEAR   EQU   03H
0020                            SMOP    EQU   20H
0021                            SSIN    EQU   21H
0022                            SMZOP   EQU   22H
0023                            SMINT   EQU   23H
                                ;
                                ; NOTE   DELAY INTERVAL CAN BE SET AT ADDRESS - 219A -
                                ;
2000  3E 82                    INI:    LD    A,82H             ;INITIAL PORT
2002  D3 23                            OUT   (SMINT),A
2004  DD 21 00 30                      LD    IX,SMDAT          ;ADDRESS DATA
2008  DD 22 FE 2F                      LD    (SMBUF),IX
                                ;
200C  3E 03                            LD    A,CLEAR           ;CLEAR DISPLAY
200E  D7                               RST   SYSCAL
                                ;
200F  21 FD 3F                 GET:    LD    HL,SYSFAG
2012  CB 86                            RES   0,(HL)            ;AUTO KEY
2014  3E 80                            LD    A,80H             ;DOT
2016  32 E7 3F                         LD    (DISPY),A
2019  3E 02                            LD    A,SCAN            ;GET KEY
201B  D7                               RST   SYSCAL
                                ;
201C  FE 00                    KEY0:   CP    0                 ;XCW YCCW
201E  20 1D                            JR    NZ,KEY1
```

```
 48  2020   CD 23 20                    CALL   RUN0
 49                              ;
 50  2023   16 33              RUN0:     LD     D,33H            ;STEP 1
 51  2025   CD 66 21                    CALL   SMRT
 52  2028   16 69                      LD     D,69H            ;STEP 2
 53  202A   CD 66 21                    CALL   SMRT
 54  202D   16 CC                      LD     D,CCH            ;STEP 3
 55  202F   CD 66 21                    CALL   SMRT
 56  2032   16 96                      LD     D,96H            ;STEP 4
 57  2034   CD 66 21                    CALL   SMRT
 58  2037   CD A3 21                    CALL   SENSOR
 59  203A   C3 1C 20                    JP     KEY0
 60                              ;
 61  203D   FE 01              KEY1:     CP     1                ;YCCW
 62  203F   20 1D                      JR     NZ,KEY2
 63  2041   CD 44 20                    CALL   RUN1
 64                              ;
 65  2044   16 30              RUN1:     LD     D,30H            ;STEP 1
 66  2046   CD 66 21                    CALL   SMRT
 67  2049   16 60                      LD     D,60H            ;STEP 2
 68  204B   CD 66 21                    CALL   SMRT
 69  204E   16 C0                      LD     D,C0H            ;STEP 3
 70  2050   CD 66 21                    CALL   SMRT
 71  2053   16 90                      LD     D,90H            ;STEP 4
 72  2055   CD 66 21                    CALL   SMRT
 73  2058   CD A3 21                    CALL   SENSOR
 74  205B   C3 3D 20                    JP     KEY1
 75                              ;
 76  205E   FE 02              KEY2:     CP     2                ;XCCW YCCW
 77  2060   20 1D                      JR     NZ,KEY4
 78  2062   CD 65 20                    CALL   RUN2
 79                              ;
 80  2065   16 33              RUN2:     LD     D,33H            ;STEP 1
 81  2067   CD 66 21                    CALL   SMRT
 82  206A   16 66                      LD     D,66H            ;STEP 2
 83  206C   CD 66 21                    CALL   SMRT
 84  206F   16 CC                      LD     D,CCH            ;STEP 3
 85  2071   CD 66 21                    CALL   SMRT
 86  2074   16 99                      LD     D,99H            ;STEP 4
 87  2076   CD 66 21                    CALL   SMRT
 88  2079   CD A3 21                    CALL   SENSOR
 89  207C   C3 5E 20                    JP     KEY2
 90                              ;
 91  207F   FE 04              KEY4:     CP     4                ;XCW
 92  2081   20 1D                      JR     NZ,KEY6
 93  2083   CD 86 20                    CALL   RUN4
 94                              ;
 95  2086   16 03              RUN4:     LD     D,03H            ;STEP 1
 96  2088   CD 66 21                    CALL   SMRT
 97  208B   16 09                      LD     D,09H            ;STEP 2
 98  208D   CD 66 21                    CALL   SMRT
 99  2090   16 0C                      LD     D,0CH            ;STEP 3
100  2092   CD 66 21                    CALL   SMRT
101  2095   16 06                      LD     D,06H            ;STEP 4
102  2097   CD 66 21                    CALL   SMRT
103  209A   CD A3 21                    CALL   SENSOR
104  209D   C3 7F 20                    JP     KEY4
```

```
105                                 ;
106  20A0   FE 06                  KEY6:     CP    6                ;XCCW
107  20A2   20 1D                            JR    NZ,KEY8
108  20A4   CD A7 20                         CALL  RUN6
109                                 ;
110  20A7   16 03                  RUN6:     LD    D,03H            ;STEP 1
111  20A9   CD 66 21                         CALL  SMRT
112  20AC   16 06                            LD    D,06H            ;STEP 2
113  20AE   CD 66 21                         CALL  SMRT
114  20B1   16 0C                            LD    D,0CH            ;STEP 3
115  20B3   CD 66 21                         CALL  SMRT
116  20B6   16 09                            LD    D,09H            ;STEP 4
117  20B8   CD 66 21                         CALL  SMRT
118  20BB   CD A3 21                         CALL  SENSOR
119  20BE   C3 A0 20                         JP    KEY6
120                                 ;
121  20C1   FE 08                  KEY8:     CP    8                ;XCW YCW
122  20C3   20 1D                            JR    NZ,KEY9
123  20C5   CD C8 20                         CALL  RUN8
124                                 ;
125  20C8   16 33                  RUN8:     LD    D,33H            ;STEP 1
126  20CA   CD 66 21                         CALL  SMRT
127  20CD   16 99                            LD    D,99H            ;STEP 2
128  20CF   CD 66 21                         CALL  SMRT
129  20D2   16 CC                            LD    D,CCH            ;STEP 3
130  20D4   CD 66 21                         CALL  SMRT
131  20D7   16 66                            LD    D,66H            ;STEP 4
132  20D9   CD 66 21                         CALL  SMRT
133  20DC   CD A3 21                         CALL  SENSOR
134  20DF   C3 C1 20                         JP    KEY8
135                                 ;
136  20E2   FE 09                  KEY9:     CP    9                ;YCW
137  20E4   20 1D                            JR    NZ,KEYA
138  20E6   CD E9 20                         CALL  RUN9
139                                 ;
140  20E9   16 30                  RUN9:     LD    D,30H            ;STEP 1
141  20EB   CD 66 21                         CALL  SMRT
142  20EE   16 90                            LD    D,90H            ;STEP 2
143  20F0   CD 66 21                         CALL  SMRT
144  20F3   16 C0                            LD    D,C0H            ;STEP 3
145  20F5   CD 66 21                         CALL  SMRT
146  20F8   16 60                            LD    D,60H            ;STEP 4
147  20FA   CD 66 21                         CALL  SMRT
148  20FD   CD A3 21                         CALL  SENSOR
149  2100   C3 E2 20                         JP    KEY9
150                                 ;
151  2103   BF                    KEYA:     CP    A                ;XCCW YCW
152  2104   20 1D                            JR    NZ,KEY3
153  2106   CD 09 21                         CALL  RUNA
154                                 ;
155  2109   16 33                  RUNA:     LD    D,33H            ;STEP 1
156  210B   CD 66 21                         CALL  SMRT
157  210E   16 96                            LD    D,96H            ;STEP 2
158  2110   CD 66 21                         CALL  SMRT
159  2113   16 CC                            LD    D,CCH            ;STEP 3
160  2115   CD 66 21                         CALL  SMRT
161  2118   16 69                            LD    D,69H            ;STEP 4
```

```
162  211A  CD 66 21                CALL  SMRT
163  211D  CD A3 21                CALL  SENSOR
164  2120  C3 03 21                JP    KEYA
165                          ;
166  2123  FE 03           KEY3:  CP    3                ;DOWN DRILLING
167  2125  20 1D                   JR    NZ,KEY7
168  2127  CD 2A 21                CALL  RUN3
169                          ;
170  212A  16 03           RUN3:  LD    D,03H            ;STEP 1
171  212C  CD 7F 21                CALL  SMZRT
172  212F  16 06                   LD    D,06H            ;STEP 2
173  2131  CD 7F 21                CALL  SMZRT
174  2134  16 0C                   LD    D,0CH            ;STEP 3
175  2136  CD 7F 21                CALL  SMZRT
176  2139  16 09                   LD    D,09H            ;STEP 4
177  213B  CD 7F 21                CALL  SMZRT
178  213E  CD A3 21                CALL  SENSOR
179  2141  C3 23 21                JP    KEY3
180                          ;
181  2144  FE 07           KEY7:  CP    7                ;UP DRILLING
182  2146  CA 1C 20                JP    Z,KEY0
183  2149  CD 4C 21                CALL  RUN7
184                          ;
185  214C  16 03           RUN7:  LD    D,03H            ;STEP 1
186  214E  CD 7F 21                CALL  SMZRT
187  2151  16 09                   LD    D,09H            ;STEP 2
188  2153  CD 7F 21                CALL  SMZRT
189  2156  16 0C                   LD    D,0CH            ;STEP 3
190  2158  CD 7F 21                CALL  SMZRT
191  215B  16 06                   LD    D,06H            ;STEP 4
192  215D  CD 7F 21                CALL  SMZRT
193  2160  CD A3 21                CALL  SENSOR
194  2163  C3 44 21                JP    KEY7
195                          ;
196  2166  3E 06           SMRT:  LD    A,6              ;STEP ROTATE
197  2168  D3 02                   OUT   (DIGIT),A        ;FLAG DISPLAY
198  216A  7A                      LD    A,D
199  216B  DD 77 00                LD    (IX+0),A
200  216E  D3 01                   OUT   (SEGM),A
201  2170  D3 20                   OUT   (SMOP),A         ;OUT STEPPING
202  2172  CD 98 21                CALL  DELAY
203  2175  C9                      RET
204                          ;
205  2176  57              SMRT1: LD    D,A              ;STEP EXCITE
206  2177  DD 23                   INC   IX
207  2179  DD 22 FE 2F             LD    (SMBUF),IX       ;STEP ADDRESS
208  217D  03                      INC   BC               ;STEP+1
209  217E  C9                      RET
210                          ;
211  217F  3E 06           SMZRT: LD    A,6              ;STEP ROTATE
212  2181  D3 02                   OUT   (DIGIT),A        ;FLAG DISPLAY
213  2183  7A                      LD    A,D
214  2184  DD 77 00                LD    (IX+0),A
215  2187  D3 01                   OUT   (SEGM),A
216  2189  D3 22                   OUT   (SMZOP),A        ;OUT STEPPING
217  218B  CD 98 21                CALL  DELAY
218  218E  C9                      RET
```

```
219                                     ;
220   218F   57                        SMZRT1:   LD     D,A                ;STEP EXCITE
221   2190   DD 23                              INC    IX
222   2192   DD 22 FE 2F                       LD     (SMBUF),IX         ;STEP ADDRESS
223   2196   03                                INC    BC                 ;STEP+1
224   2197   C9                                RET
225                                     ;
226   2198   08                        DELAY:    EX     AF,AF'
227   2199   21 00 70                          LD     HL,7000H           ;DELAY
228                                     ;
229   219C   2B                        DELAY1:   DEC    HL
230   219D   7C                                LD     A,H
231   219E   B5                                OR     L
232   219F   20 FB                             JR     NZ,DELAY1
233   21A1   08                                EX     AF,AF'
234   21A2   C9                                RET
235                                     ;
236   21A3   ED 56                      SENSOR:   IM     1                  ;SET INTERRUPT MODE
237   21A5   F5                                 PUSH   AF                 ;SAVE ALL REGISTOR
238   21A6   E5                                 PUSH   HL
239   21A7   D5                                 PUSH   DE
240   21A8   C5                                 PUSH   BC
241   21A9   CD AC 21                           CALL   INTSUB
242                                     ;
243   21AC   DB 21                      INTSUB:   IN     A,(SSIN)           ;READ PORT 8
244   21AE   CB 47                              BIT    0,A                ;TEST BIT 0 = 1
245   21B0   CA D1 21                           JP     Z,SHUT
246   21B3   CB 4F                              BIT    1,A                ;TEST BIT 1 = 1
247   21B5   CA D1 21                           JP     Z,SHUT
248   21B8   CB 57                              BIT    2,A                ;TEST BIT 2 = 1
249   21BA   CA D1 21                           JP     Z,SHUT
250   21BD   CB 5F                              BIT    3,A                ;TEST BIT 3 = 1
251   21BF   CA D1 21                           JP     Z,SHUT
252   21C2   CB 7F                              BIT    7,A                ;TEST BIT 7 = 1
253   21C4   CA D1 21                           JP     Z,SHUT
254   21C7   C9                                RET
255                                     ;
256   21C8   C1                                 POP    BC                 ;POP ALL REGISTOR
257   21C9   D1                                 POP    DE
258   21CA   E1                                 POP    HL
259   21CB   F1                                 POP    AF
260   21CC   D3 FF                              OUT    (0FFH),A           ;CLEAR INTERRUPT
261   21CE   FB                                 EI                        ;ENABLE INTERRUPTS
262   21CF   ED 4D                              RETI
263                                     ;
264   21D1   C3 0F 20                   SHUT:     JP     GET                ;RESTART
265   21D4   D7                                 RST    SYSCAL
266   21D5                                      END
```

กิตติกรรมประกาศ

ขอแสดงความขอบคุณผู้ที่ให้ความอนุเคราะห์ จนปริญญานิพนธ์นี้สำเร็จลงได้ด้วยดี

รศ. ดร. วิริยะ พิเชฐจำเริญ	อาจารย์ที่ปรึกษา
พี่ ๆ บริษัทเคมเท็ค (Chemtec)	ที่ให้คำแนะนำเกี่ยวกับชุดแมคคานิคส์
คุณ บุญช่วย ประยูรมณีรัตน์	ที่ให้คำแนะนำและช่วยเหลือ

Silicon Gate MOS 8255

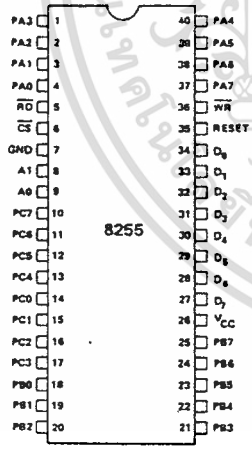
PROGRAMMABLE PERIPHERAL INTERFACE

- 24 Programmable I/O Pins
- Completely TTL Compatible
- Fully Compatible with MCS™-8 and MCS™-80 Microprocessor Families
- Direct Bit Set/Reset Capability Easing Control Application Interface
- 40 Pin Dual In-Line Package
- Reduces System Package Count

The 8255 is a general purpose programmable I/O device designed for use with both the 8008 and 8080 microprocessors. It has 24 I/O pins which may be individually programmed in two groups of twelve and used in three major modes of operation. In the first mode (Mode 0), each group of twelve I/O pins may be programmed in sets of 4 to be input or output. In Mode 1, the second mode, each group may be programmed to have 8 lines of input or output. Of the remaining four pins three are used for handshaking and interrupt control signals. The third mode of operation (Mode 2) is a Bidirectional Bus mode which uses 8 lines for a bidirectional bus, and five lines, borrowing one from the other group, for handshaking.

Other features of the 8255 include bit set and reset capability and the ability to source 1mA of current at 1.5 volts. This allows darlington transistors to be directly driven for applications such as printers and high voltage displays.

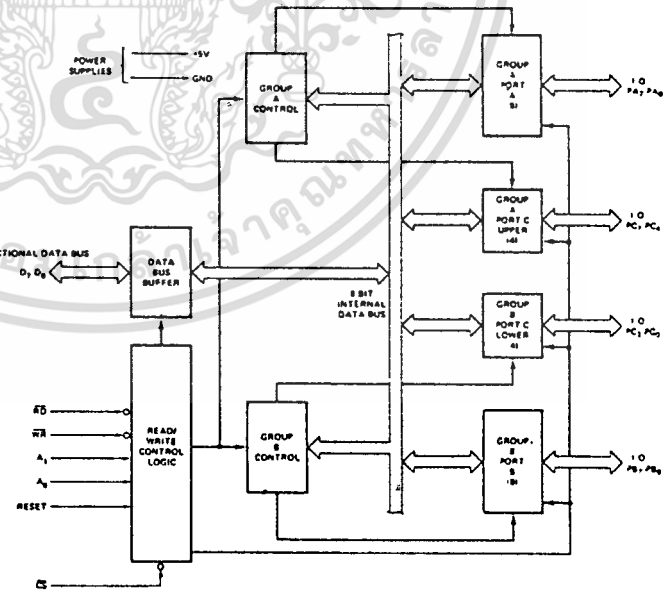
PIN CONFIGURATION



PIN NAMES

D_7–D_0	DATA BUS (BI-DIRECTIONAL)
RESET	RESET INPUT
$\overline{CS}$	CHIP SELECT
$\overline{RD}$	READ INPUT
$\overline{WR}$	WRITE INPUT
A0, A1	PORT ADDRESS
PA7-PA0	PORT A (BIT)
PB7-PB0	PORT B (BIT)
PC7-PC0	PORT C (BIT)
V_{CC}	+5 VOLTS
GND	0 VOLTS

8255 BLOCK DIAGRAM



8255 BASIC FUNCTIONAL DESCRIPTION

General

The 8255 is a Programmable Peripheral Interface (PPI) device designed for use in 8080 Microcomputer Systems. Its function is that of a general purpose I/O component to interface peripheral equipment to the 8080 system bus. The functional configuration of the 8255 is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

Data Bus Buffer

This 3-state, bi-directional, eight bit buffer is used to interface the 8255 to the 8080 system data bus. Data is transmitted or received by the buffer upon execution of INput or OUTput instructions by the 8080 CPU. Control Words and Status information are also transferred through the Data Bus buffer.

Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the 8080 CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

($\overline{CS}$)

Chip Select: A "low" on this input pin enables the communication between the 8255 and the 8080 CPU.

($\overline{RD}$)

Read: A "low" on this input pin enables the 8255 to send the Data or Status information to the 8080 CPU on the Data Bus. In essence, it allows the 8080 CPU to "read from" the 8255.

($\overline{WR}$)

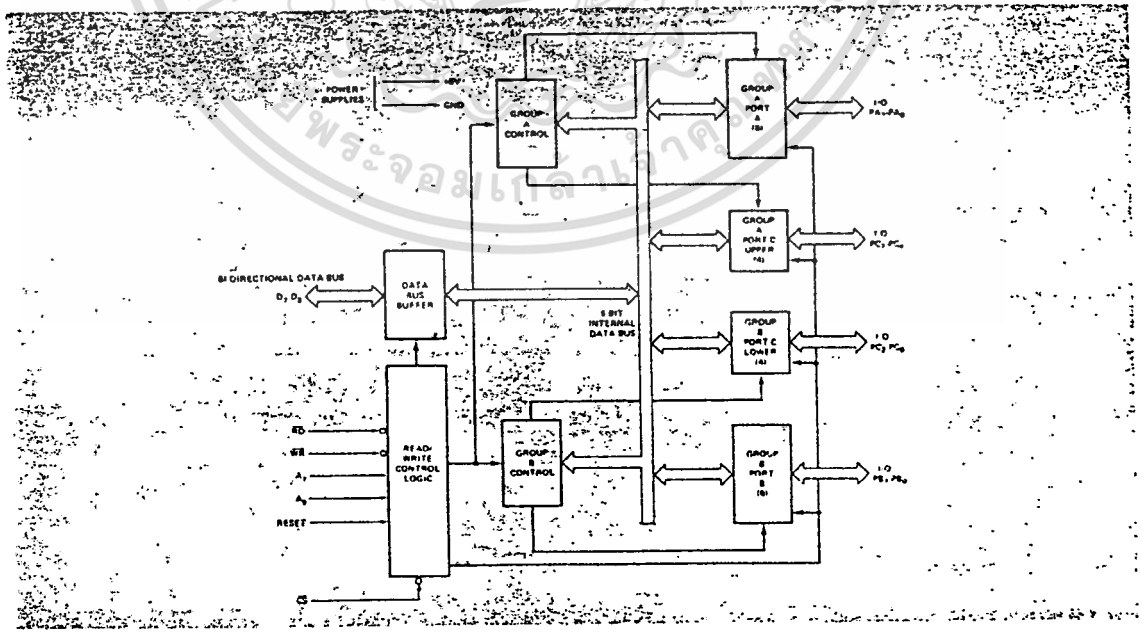
Write: A "low" on this input pin enables the 8080 CPU to write Data or Control words into the 8255.

(A_0 and A_1)

Port Select 0 and Port Select 1: These input signals, in conjunction with the $\overline{RD}$ and $\overline{WR}$ inputs, control the selection of one of the three ports or the Control Word Register. They are normally connected to the least significant bits of the Address Bus (A_0 and A_1).

8255 BASIC OPERATION

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	INPUT OPERATION (READ)
0	0	0	1	0	PORT A → DATA BUS
0	1	0	1	0	PORT B → DATA BUS
1	0	0	1	0	PORT C → DATA BUS
					OUTPUT OPERATION (WRITE)
0	0	1	0	0	DATA BUS → PORT A
0	1	1	0	0	DATA BUS → PORT B
1	0	1	0	0	DATA BUS → PORT C
1	1	1	0	0	DATA BUS → CONTROL
					DISABLE FUNCTION
X	X	X	X	1	DATA BUS → 3-STATE
1	1	0	1	0	ILLEGAL CONDITION



8255 Block Diagram

(RESET)

Reset: A "high" on this input clears all internal registers including the Control Register and all ports (A, B, C) are set to the input mode.

Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the 8080 CPU "outputs" a control word to the 8255. The control word contains information such as "mode", "bit set", "bit reset" etc. that initializes the functional configuration of the 8255.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control Logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Control Group A – Port A and Port C upper (C7-C4)
Control Group B – Port B and Port C lower (C3-C0)

The Control Word Register can Only be written into. No Read operation of the Control Word Register is allowed.

Ports A, B, and C

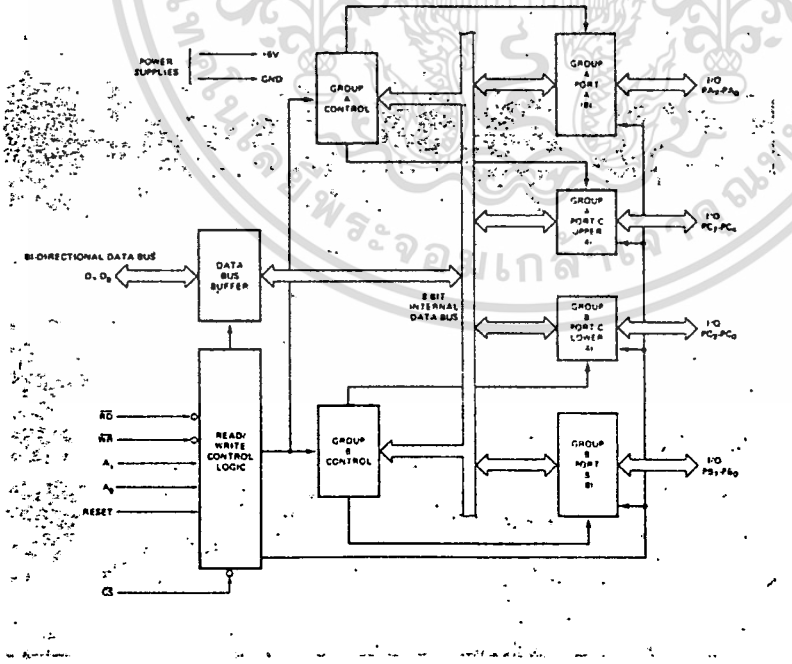
The 8255 contains three 8-bit ports (A, B, and C). All can be configured in a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255.

Port A: One 8-bit data output latch/buffer and one 8-bit data input latch.

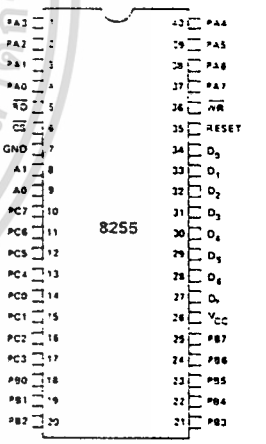
Port B: One 8-bit data input/output latch/buffer and one 8-bit data input buffer.

Port C: One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with Ports A and B.

8255 BLOCK DIAGRAM



PIN CONFIGURATION



PIN NAMES

D_7–D_0	DATA BUS (BI-DIRECTIONAL)
RESET	RESET INPUT
$\overline{CS}$	CHIP SELECT
$\overline{RD}$	READ INPUT
$\overline{WR}$	WRITE INPUT
A0, A1	PORT ADDRESS
PA7-PA0	PORT A (BIT)
PB7-PB0	PORT B (BIT)
PC7-PC0	PORT C (BIT)
V_{CC}	+5 VOLTS
GND	0 VOLTS

8255 DETAILED OPERATIONAL DESCRIPTION

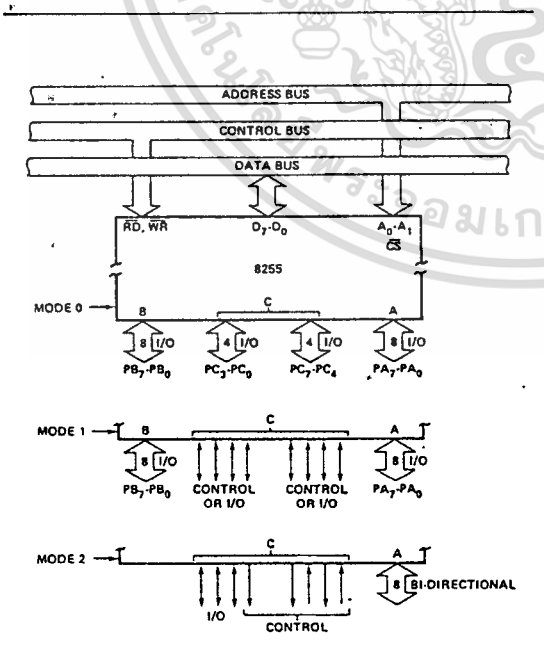
Mode Selection

There are three basic modes of operation that can be selected by the system software:

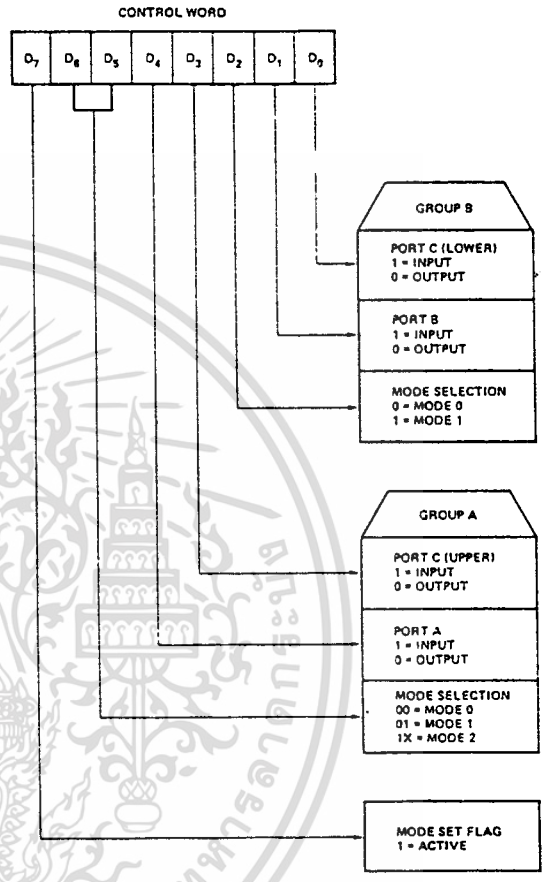
Mode 0 — Basic Input/Output
Mode 1 — Strobed Input/Output
Mode 2 — Bi-Directional Bus

When the RESET input goes "high" all ports will be set to the Input mode (i.e., all 24 lines will be in the high impedance state). After the RESET is removed the 8255 can remain in the Input mode with no additional initialization required. During the execution of the system program any of the other modes may be selected using a single OUTput instruction. This allows a single 8255 to service a variety of peripheral devices with a simple software maintenance routine.

The modes for Port A and Port B can be separately defined, while Port C is divided into two portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed. Modes may be combined so that their functional definition can be "tailored" to almost any I/O structure. For instance; Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results, Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.



Basic Mode Definitions and Bus Interface

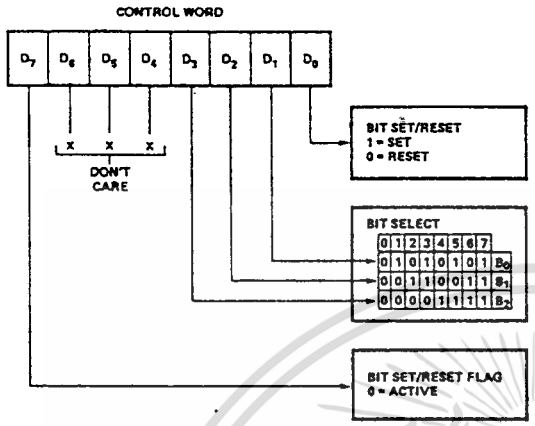


Mode Definition Format

The Mode definitions and possible Mode combinations may seem confusing at first but after a cursory review of the complete device operation a simple, logical I/O approach will surface. The design of the 8255 has taken into account things such as efficient PC board layout, control signal definition vs PC layout and complete functional flexibility to support almost any peripheral device with no external logic. Such design represents the maximum use of the available pins.

Single Bit Set/Reset Feature

Any of the eight bits of Port C can be Set or Reset using a single OUTput instruction. This feature reduces software requirements in Control-based applications.



Bit Set/Reset Format

When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were data output ports.

Interrupt Control Functions

When the 8255 is programmed to operate in Mode 1 or Mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from Port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the Bit set/reset function of Port C.

This function allows the Programmer to disallow or allow a specific I/O device to interrupt the CPU without effecting any other device in the interrupt structure.

INTE flip-flop definition:

(BIT-SET) — INTE is SET — Interrupt enable
(BIT-RESET) — INTE is RESET — Interrupt disable

Note: All Mask flip-flops are automatically reset during mode selection and device Reset.

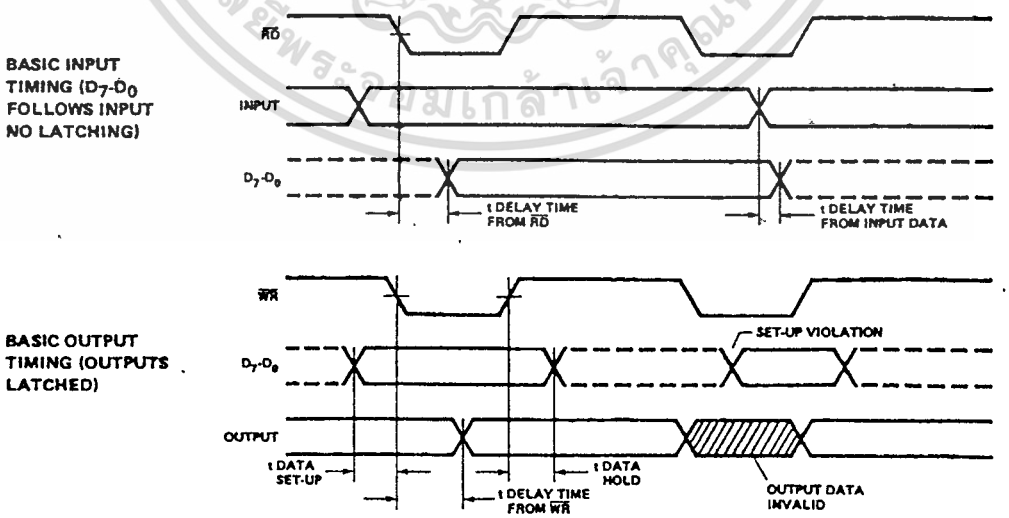
Operating Modes

Mode 0 (Basic Input/Output)

This functional configuration provides simple Input and Output operations for each of the three ports. No "handshaking" is required, data is simply written to or read from a specified port.

Mode 0 Basic Functional Definitions:

- Two 8-bit ports and two 4-bit ports.
- Any port can be input or output.
- Outputs are latched.
- Inputs are not latched.
- 16 different Input/Output configurations are possible in this Mode.

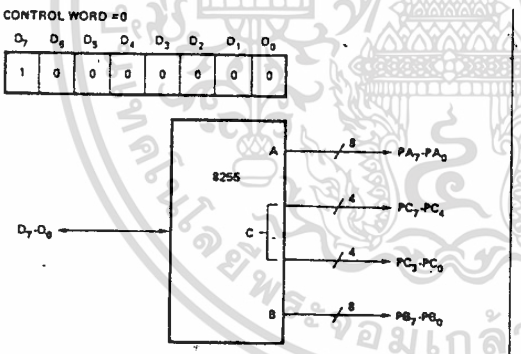


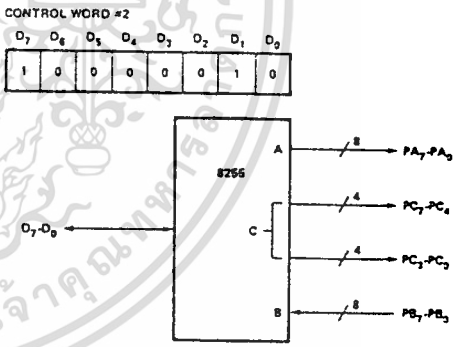
Mode 0 Timing

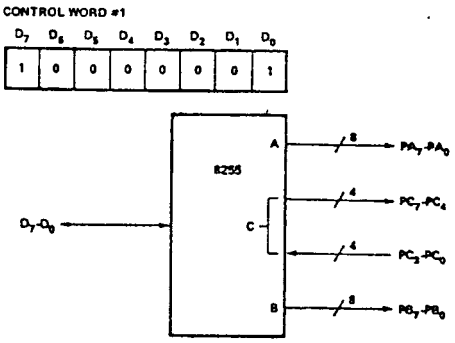
MODE 0 PORT DEFINITION CHART

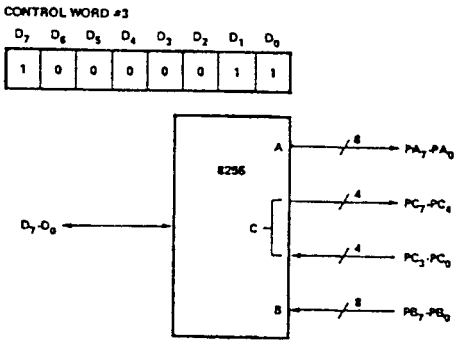
A		B		GROUP A			GROUP B	
D_4	D_3	D_1	D_0	PORT A	PORT C (UPPER)	#	PORT B	PORT C (LOWER)
0	0	0	0	OUTPUT	OUTPUT	0	OUTPUT	OUTPUT
0	0	0	1	OUTPUT	OUTPUT	1	OUTPUT	INPUT
0	0	1	0	OUTPUT	OUTPUT	2	INPUT	OUTPUT
0	0	1	1	OUTPUT	OUTPUT	3	INPUT	INPUT
0	1	0	0	OUTPUT	INPUT	4	OUTPUT	OUTPUT
0	1	0	1	OUTPUT	INPUT	5	OUTPUT	INPUT
0	1	1	0	OUTPUT	INPUT	6	INPUT	OUTPUT
0	1	1	1	OUTPUT	INPUT	7	INPUT	INPUT
1	0	0	0	INPUT	OUTPUT	8	OUTPUT	OUTPUT
1	0	0	1	INPUT	OUTPUT	9	OUTPUT	INPUT
1	0	1	0	INPUT	OUTPUT	10	INPUT	OUTPUT
1	0	1	1	INPUT	OUTPUT	11	INPUT	INPUT
1	1	0	0	INPUT	INPUT	12	OUTPUT	OUTPUT
1	1	0	1	INPUT	INPUT	13	OUTPUT	INPUT
1	1	1	0	INPUT	INPUT	14	INPUT	OUTPUT
1	1	1	1	INPUT	INPUT	15	INPUT	INPUT

MODE 0 CONFIGURATIONS



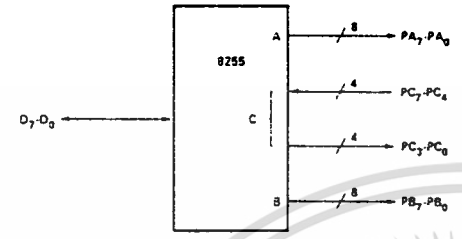






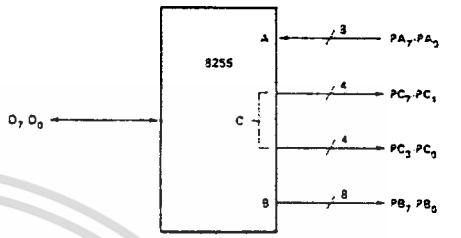
CONTROL WORD =4

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	0	0



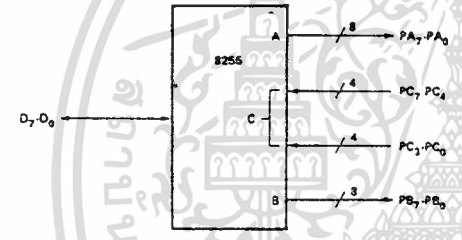
CONTROL WORD =8

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	0	0



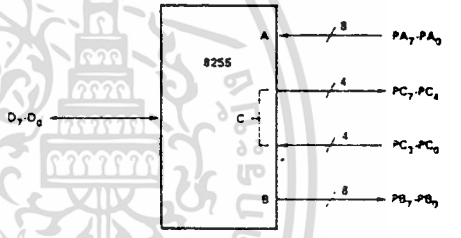
CONTROL WORD =5

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	0	1



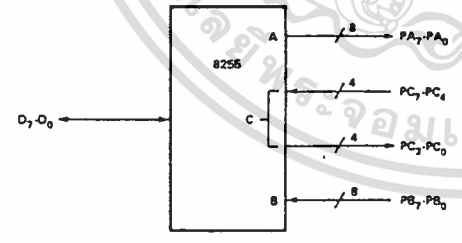
CONTROL WORD =9

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	0	1



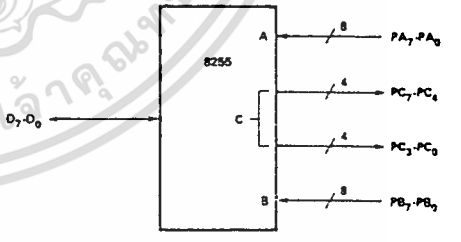
CONTROL WORD =6

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	0



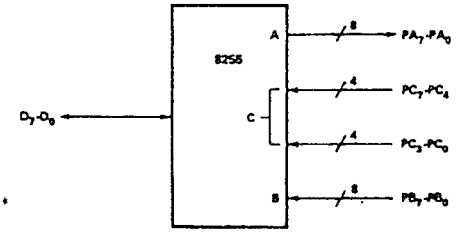
CONTROL WORD =10

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	0



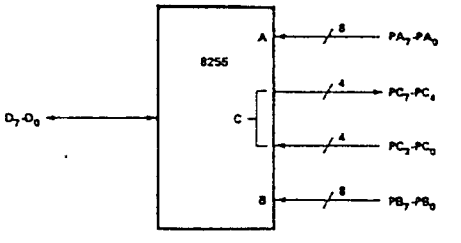
CONTROL WORD =7

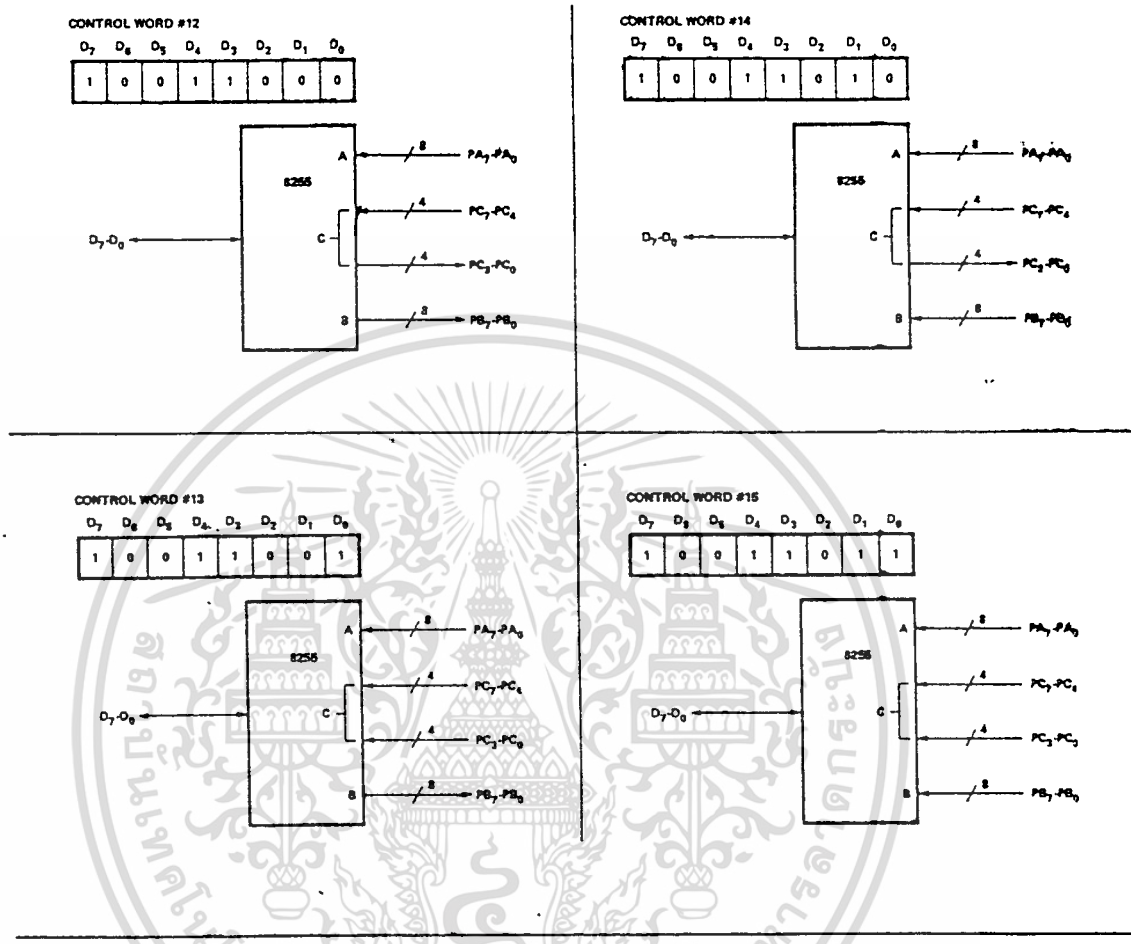
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	1



CONTROL WORD =11

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	1





Operating Modes

Mode 1 (Strobed Input/Output)

This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "handshaking" signals. In Mode 1, Port A and Port B use the lines on Port C to generate or accept these "handshaking" signals.

Mode 1 Basic Functional Definitions:

- Two Groups (Group A and Group B)
- Each group contains one 8-bit data port and one 4-bit control/data port.
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit data port.

Input Control Signal Definition

$\overline{STB}$ (Strobe Input)

A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F)

A "high" on this output indicates that the data has been loaded into the input latch; in essence, an acknowledgement. IBF is set by the falling edge of the STB input and is reset by the rising edge of the RD input.

INTR (Interrupt Request)

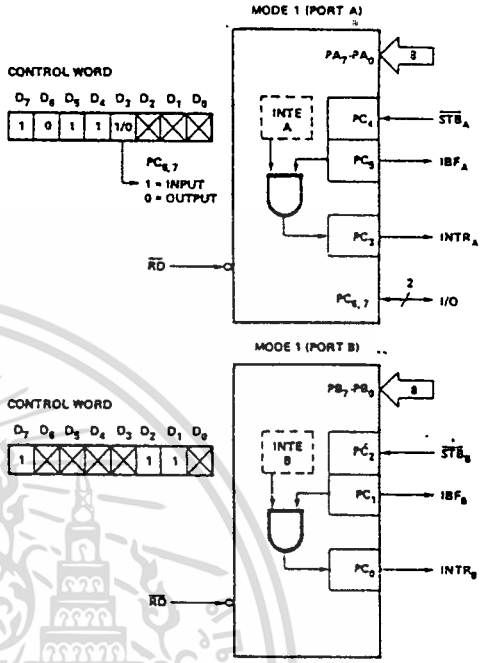
A "high" on this output can be used to interrupt the CPU when an input device is requesting service. INTR is set by the rising edge of $\overline{STB}$ if IBF is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{RD}$. This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

INTE A

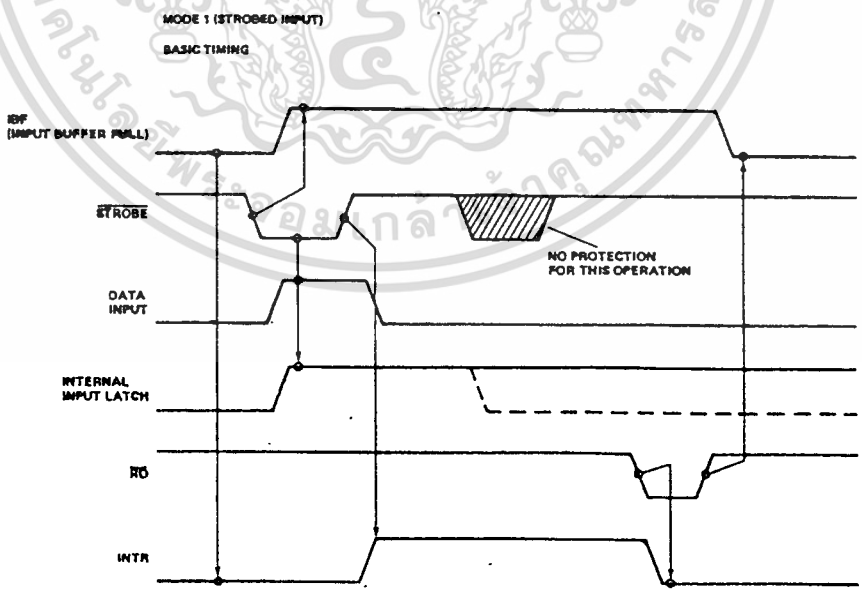
Controlled by bit set/reset of PC_4.

INTE B

Controlled by bit set/reset of PC_2.



Mode 1 Input



Basic Timing Input

Output Control Signal Definition

$\overline{OBF}$ (Output Buffer Full F/F)

The $\overline{OBF}$ output will go "low" to indicate that the CPU has written data out to the specified port. The OBF F/F will be set by the rising edge of the WR input and reset by the falling edge of the $\overline{ACK}$ input signal.

$\overline{ACK}$ (Acknowledge Input)

A "low" on this input informs the 8255 that the data from Port A or Port B has been accepted. In essence, a response from the peripheral device indicating that it has received the data output by the CPU.

INTR (Interrupt Request)

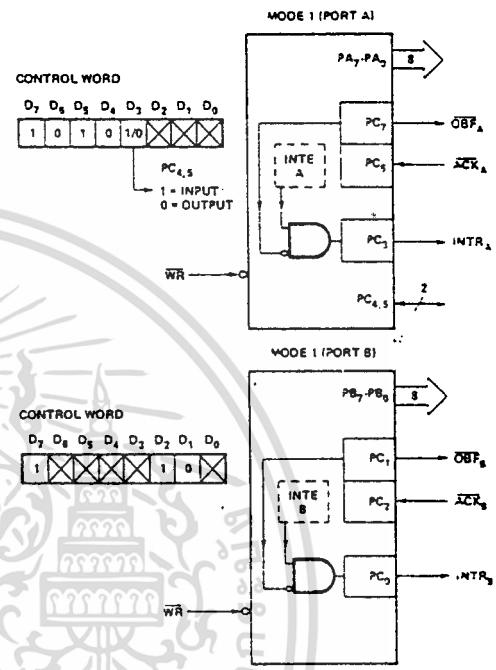
A "high" on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. INTR is set by the rising edge of $\overline{ACK}$ if $\overline{OBF}$ is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{WR}$.

INTE A

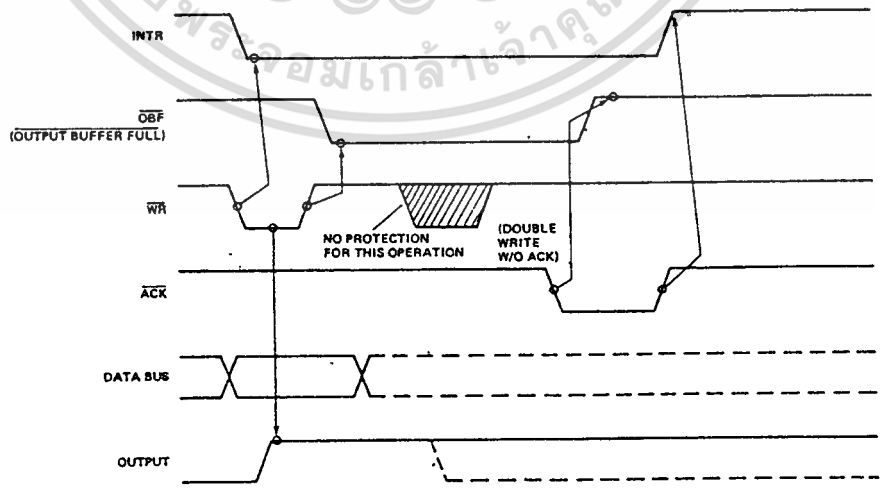
Controlled by bit set/reset of PC_6.

INTE B

Controlled by bit set/reset of PC_2.



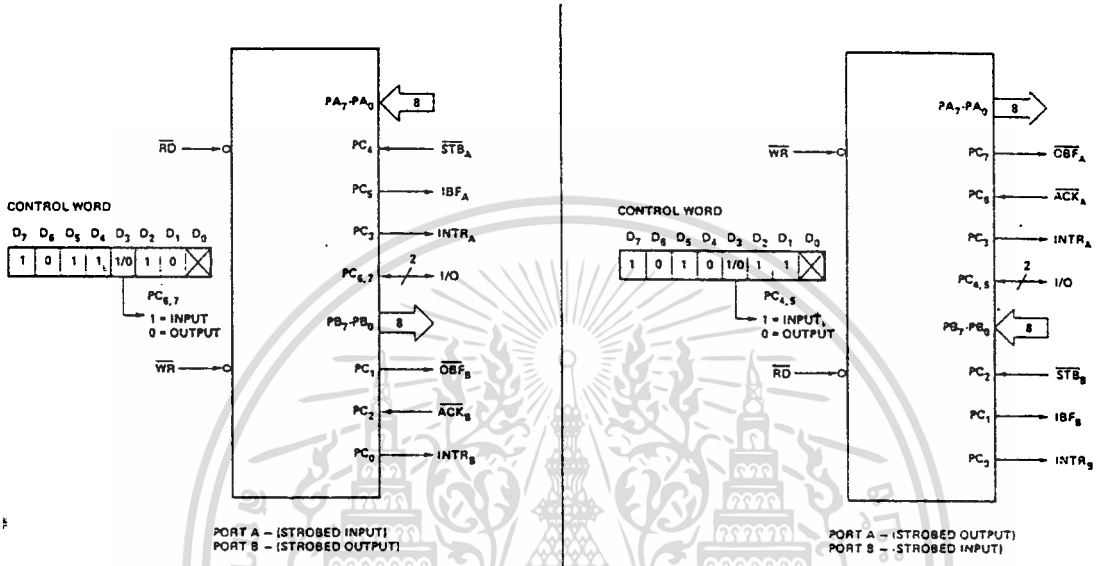
Mode 1 Output



Basic Timing Output

Combinations of Mode 1

Port A and Port B can be individually defined as input or output in Mode 1 to support a wide variety of strobed I/O applications.



Operating Modes

Mode 2 (Strobed Bi-Directional Bus I/O)

This functional configuration provides a means for communicating with a peripheral device or structure on a single 8-bit bus for both transmitting and receiving data (bi-directional bus I/O). "Handshaking" signals are provided to maintain proper bus flow discipline in a similar manner to Mode 1. Interrupt generation and enable/disable functions are also available.

Mode 2 Basic Functional Definitions:

- Used in Group A only.
- One 8-bit, bi-directional bus Port (Port A) and a 5-bit control Port (Port C).
- Both inputs and outputs are latched.
- The 5-bit control port (Port C) is used for control and status for the 8-bit, bi-directional bus port (Port A).

Bi-Directional Bus I/O Control Signal Definition

INTR (Interrupt Request)

A high on this output can be used to interrupt the CPU for both input or output operations.

Output Operations

$\overline{OBF}$ (Output Buffer Full)

The $\overline{OBF}$ output will go "low" to indicate that the CPU has written data out to Port A.

$\overline{ACK}$ (Acknowledge)

A "low" on this input enables the tri-state output buffer of Port A to send out the data. Otherwise, the output buffer will be in the high-impedance state.

INTE 1 (The INTE Flip-Flop associated with $\overline{OBF}$)

Controlled by bit set/reset of PC_6.

Input Operations

$\overline{STB}$ (Strobe Input)

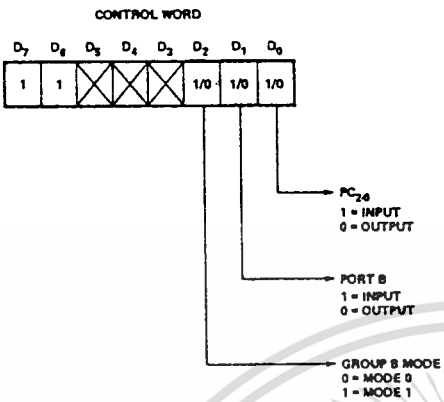
A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F)

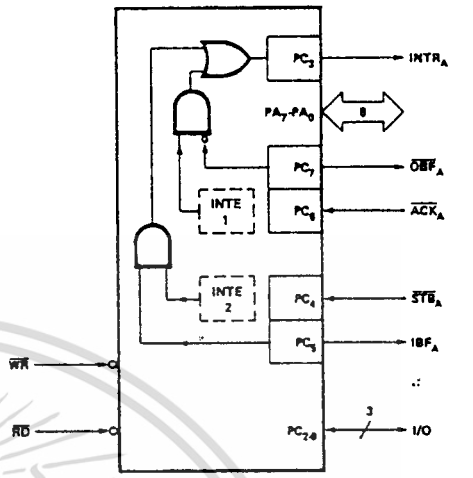
A "high" on this output indicates that data has been loaded into the input latch.

INTE 2 (The INTE Flip-Flop associated with IBF)

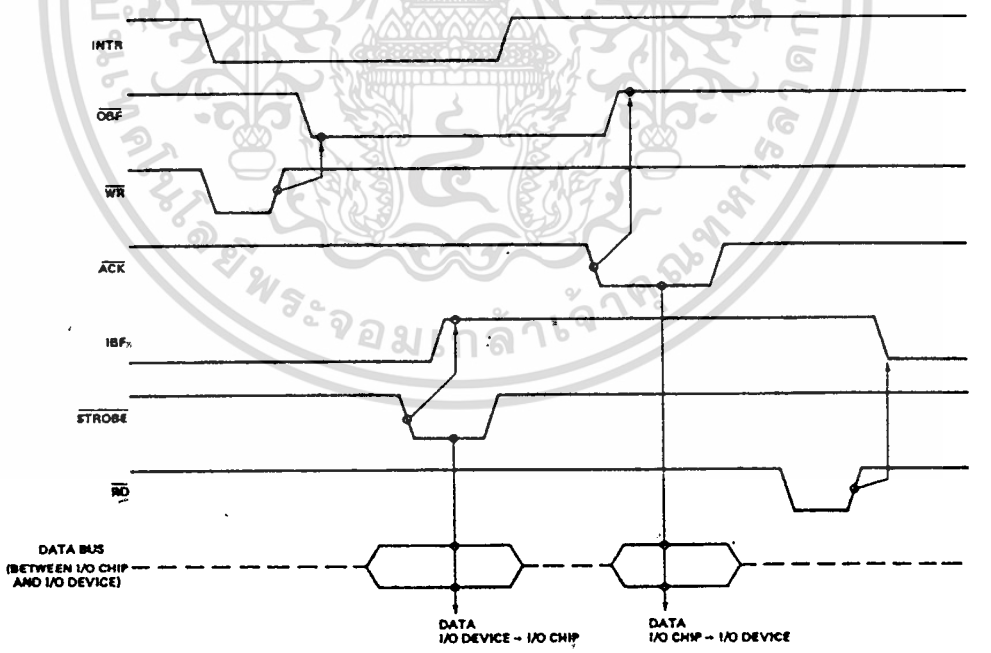
Controlled by bit set/reset of PC_4.



Mode 2 Control Word

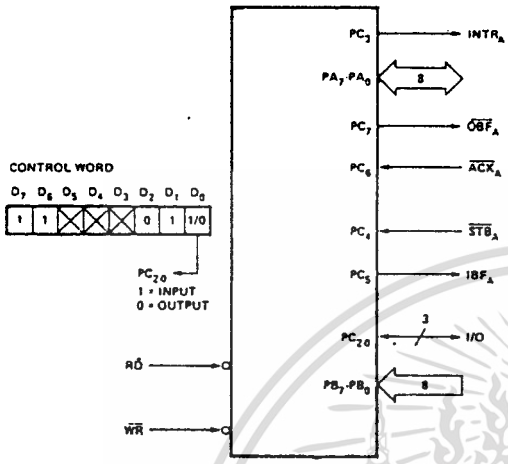


Mode 2

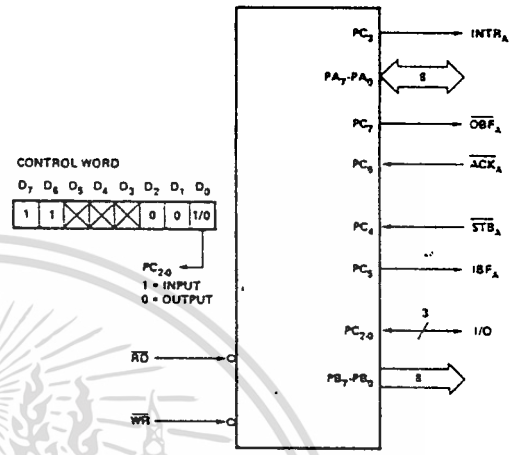


Mode 2 (Bi-directional) Timing

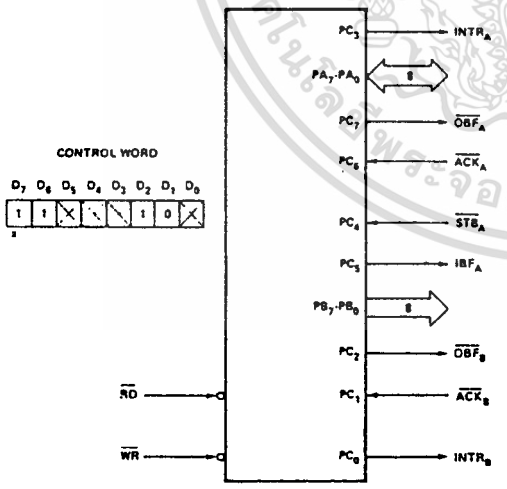
MODE 2 AND MODE 0 (INPUT)



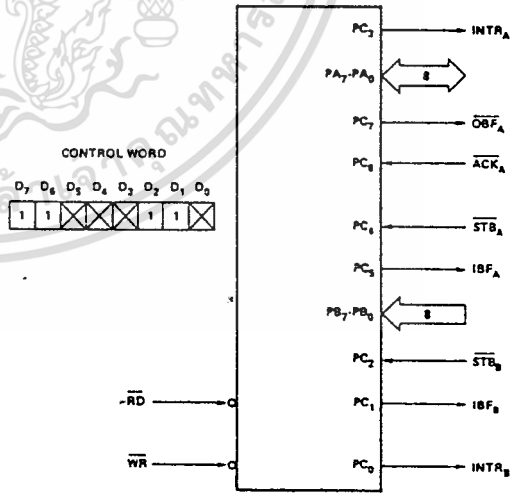
MODE 2 AND MODE 0 (OUTPUT)



MODE 2 AND MODE 1 (OUTPUT)



MODE 2 AND MODE 1 (INPUT)



Mode 2 Combinations

MODE DEFINITION SUMMARY TABLE

	MODE 0		MODE 1		MODE 2	
	IN	OUT	IN	OUT	GROUP A ONLY	
PA_0	IN	OUT	IN	OUT	◄►	
PA_1	IN	OUT	IN	OUT	◄►	
PA_2	IN	OUT	IN	OUT	◄►	
PA_3	IN	OUT	IN	OUT	◄►	
PA_4	IN	OUT	IN	OUT	◄►	
PA_5	IN	OUT	IN	OUT	◄►	
PA_6	IN	OUT	IN	OUT	◄►	
PA_7	IN	OUT	IN	OUT	◄►	
PB_0	IN	OUT	IN	OUT	—	MODE 0 OR MODE 1 ONLY
PB_1	IN	OUT	IN	OUT	—	
PB_2	IN	OUT	IN	OUT	—	
PB_3	IN	OUT	IN	OUT	—	
PB_4	IN	OUT	IN	OUT	—	
PB_5	IN	OUT	IN	OUT	—	
PB_6	IN	OUT	IN	OUT	—	
PB_7	IN	OUT	IN	OUT	—	
PC_0	IN	OUT	$INTR_B$	$INTR_B$	I/O	
PC_1	IN	OUT	IBF_B	$\overline{OBF}_B$	I/O	
PC_2	IN	OUT	$\overline{STB}_B$	$\overline{ACK}_B$	I/O	
PC_3	IN	OUT	$INTR_A$	$INTR_A$	$INTR_A$	
PC_4	IN	OUT	$\overline{STB}_A$	I/O	$\overline{STB}_A$	
PC_5	IN	OUT	IBF_A	I/O	IBF_A	
PC_6	IN	OUT	I/O	$\overline{ACK}_A$	$\overline{ACK}_A$	
PC_7	IN	OUT	I/O	$\overline{OBF}_A$	$\overline{OBF}_A$	

Special Mode Combination Considerations

There are several combinations of modes when not all of the bits in Port C are used for control or status. The remaining bits can be used as follows:

If Programmed as Inputs –
All input lines can be accessed during a normal Port C read.

If Programmed as Outputs –
Bits in C upper (PC_7-PC_4) must be individually accessed using the bit set/reset function.

Bits in C lower (PC_3-PC_0) can be accessed using the bit set/reset function or accessed as a threesome by writing into Port C.

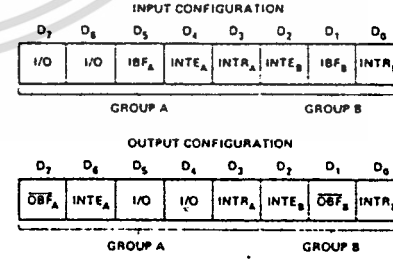
Source Current Capability on Port B and Port C

Any set of eight output buffers, selected randomly from Ports B and C can source 1mA at 1.5 volts. This feature allows the 8255 to directly drive Darlington type drivers and high-voltage displays that require such source current.

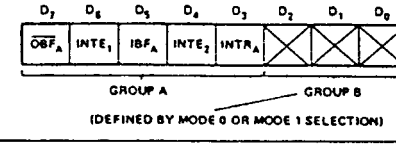
Reading Port C Status

In Mode 0, Port C transfers data to or from the peripheral device. When the 8255 is programmed to function in Modes 1 or 2, Port C generates or accepts "hand-shaking" signals with the peripheral device. Reading the contents of Port C allows the programmer to test or verify the "status" of each peripheral device and change the program flow accordingly.

There is no special instruction to read the status information from Port C. A normal read operation of Port C is executed to perform this function.



Mode 1 Status Word Format



Mode 2 Status Word Format

Z80™-CPU
Z80A™-CPU

Zilog

Product Specification

The Zilog Z80 product line is a complete set of microcomputer components, development systems and support software. The Z80 microcomputer component set includes all of the circuits necessary to build high-performance microcomputer systems with virtually no other logic and a minimum number of low cost standard memory elements.

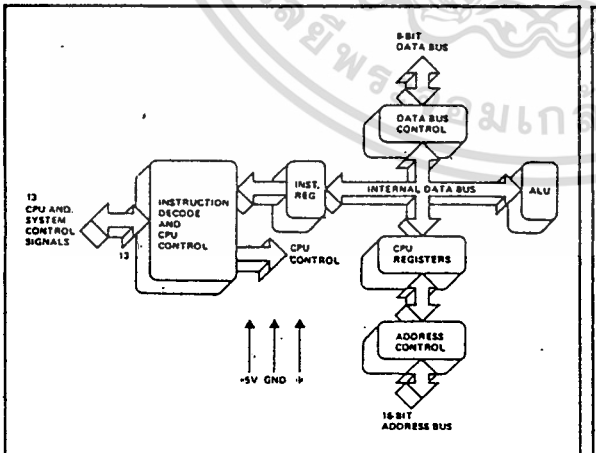
The Z80 and Z80A CPU's are third generation single chip microprocessors with unrivaled computational power. This increased computational power results in higher system through-put and more efficient memory utilization when compared to second generation microprocessors. In addition, the Z80 and Z80A CPU's are very easy to implement into a system because of their single voltage requirement plus all output signals are fully decoded and timed to control standard memory or peripheral circuits. The circuit is implemented using an N-channel; ion implanted, silicon gate MOS process.

Figure 1 is a block diagram of the CPU, Figure 2 details the internal register configuration which contains 208 bits of Read/Write memory that are accessible to the programmer. The registers include two sets of six general purpose registers that may be used individually as 8-bit registers or as 16-bit register pairs. There are also two sets of accumulator and flag registers. The programmer has access to either set of main or alternate registers through a group of exchange instructions. This alternate set allows foreground/background mode of operation or may be reserved for very fast Interrupt response. Each CPU also contains a 16-bit stack pointer which permits simple implementation of multiple level interrupts, unlimited subroutine nesting and simplification of many types of data handling.

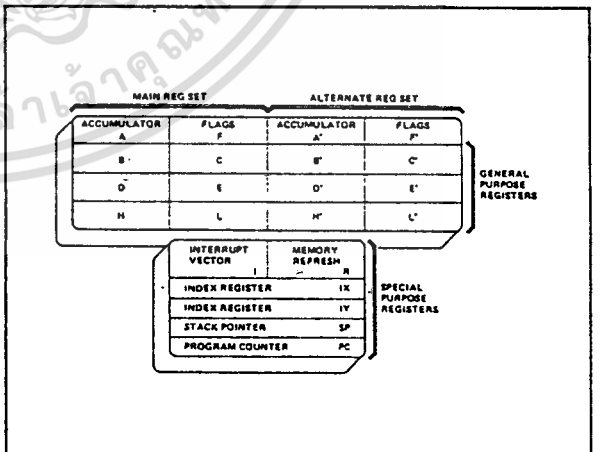
The two 16-bit index registers allow tabular data manipulation and easy implementation of relocatable code. The Refresh register provides for automatic, totally transparent refresh of external dynamic memories. The I register is used in a powerful interrupt response mode to form the upper 8 bits of a pointer to a interrupt service address table, while the interrupting device supplies the lower 8 bits of the pointer. An indirect call is then made to this service address.

FEATURES

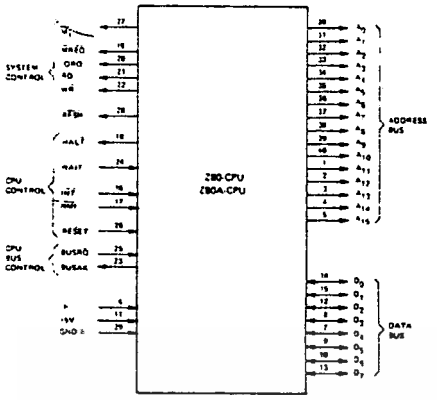
- Single chip, N-channel Silicon Gate CPU.
- 158 instructions—includes all 78 of the 8080A instructions with total software compatibility. New instructions include 4-, 8- and 16-bit operations with more useful addressing modes such as indexed, bit and relative.
- 17 internal registers.
- Three modes of fast interrupt response plus a non-maskable interrupt.
- Directly interfaces standard speed static or dynamic memories with virtually no external logic.
- 1.0 μs instruction execution speed.
- Single 5 VDC supply and single-phase 5 volt Clock.
- Out-performs any other single chip microcomputer in 4-, 8-, or 16-bit applications.
- All pins TTL Compatible
- Built-in dynamic RAM refresh circuitry.



Z80, Z80A CPU BLOCK DIAGRAM



Z80, Z80A CPU REGISTERS



Z80, Z80A CPU PIN CONFIGURATION

A_0-A_{15} (Address Bus) Tri-state output, active high. A_0-A_{15} constitute a 16-bit address bus. The address bus provides the address for memory (up to 64K bytes) data exchanges and for I/O device data exchanges.

D_0-D_7 (Data Bus) Tri-state input/output, active high. D_0-D_7 constitute an 8-bit bidirectional data bus. The data bus is used for data exchanges with memory and I/O devices.

$\overline{M_1}$ (Machine Cycle one) Output, active low. $\overline{M_1}$ indicates that the current machine cycle is the OP code fetch cycle of an instruction execution.

$\overline{MREQ}$ (Memory Request) Tri-state output, active low. The memory request signal indicates that the address bus holds a valid address for a memory read or memory write operation.

$\overline{IORQ}$ (Input/ Output Request) Tri-state output, active low. The $\overline{IORQ}$ signal indicates that the lower half of the address bus holds a valid I/O address for a I/O read or write operation. An $\overline{IORQ}$ signal is also generated when an interrupt is being acknowledged to indicate that an interrupt response vector can be placed on the data bus.

$\overline{RD}$ (Memory Read) Tri-state output, active low. $\overline{RD}$ indicates that the CPU wants to read data from memory or an I/O device. The addressed I/O device or memory should use this signal to gate data onto the CPU data bus.

$\overline{WR}$ (Memory Write) Tri-state output, active low. $\overline{WR}$ indicates that the CPU data bus holds valid data to be stored in the addressed memory or I/O device.

$\overline{RFSH}$ (Refresh) Output, active low. $\overline{RFSH}$ indicates that the lower 7 bits of the address bus contain a refresh address for dynamic memories and the current $\overline{MREQ}$ signal should be used to do a refresh read to all dynamic memories.

$\overline{HALT}$ (Halt state) Output, active low. $\overline{HALT}$ indicates that the CPU has executed a HALT software instruction and is awaiting either a non-maskable or a maskable interrupt (with the mask enabled) before operation can resume. While halted, the CPU executes NOP's to maintain memory refresh activity.

$\overline{WAIT}$ (Wait) Input, active low. $\overline{WAIT}$ indicates to the Z-80 CPU that the addressed memory or I/O devices are not ready for a data transfer. The CPU continues to enter wait states for as long as this signal is active.

$\overline{INT}$ (Interrupt Request) Input, active low. The Interrupt Request signal is generated by I/O devices. A request will be honored at the end of the current instruction if the internal software controlled interrupt enable flip-flop (IFF) is enabled.

$\overline{NMI}$ (Non Maskable Interrupt) Input, active low. The non-maskable interrupt request line has a higher priority than $\overline{INT}$ and is always recognized at the end of the current instruction, independent of the status of the interrupt enable flip-flop. $\overline{NMI}$ automatically forces the Z-80 CPU to restart to location 0066_H.

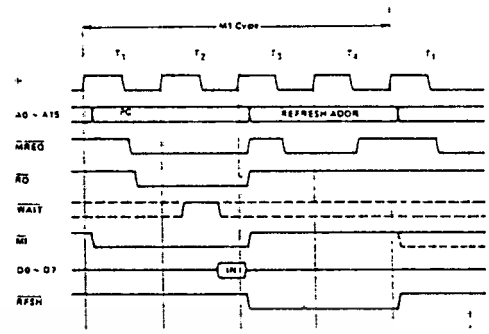
$\overline{RESET}$ Input, active low. $\overline{RESET}$ initializes the CPU as follows: reset interrupt enable flip-flop, clear PC and registers I and R and set interrupt to 8080A mode. During reset time, the address and data bus go to a high impedance state and all control output signals go to the inactive state.

$\overline{BUSRQ}$ (Bus Request) Input, active low. The bus request signal has a higher priority than $\overline{NMI}$ and is always recognized at the end of the current machine cycle and is used to request the CPU address bus, data bus and tri-state output control signals to go to a high impedance state so that other devices can control these busses.

$\overline{BUSAK}$ (Bus Acknowledge) Output, active low. Bus acknowledge is used to indicate to the requesting device that the CPU address bus, data bus and tri-state control bus signals have been set to their high impedance state and the external device can now control these signals.

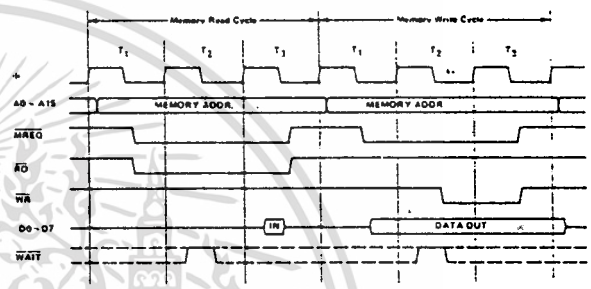
INSTRUCTION OP CODE FETCH

The program counter content (PC) is placed on the address bus immediately at the start of the cycle. One half clock time later $\overline{MREQ}$ goes active. The falling edge of $\overline{MREQ}$ can be used directly as a chip enable to dynamic memories. $\overline{RD}$ when active indicates that the memory data should be enabled onto the CPU data bus. The CPU samples data with the rising edge of the clock state T_3. Clock states T_3 and T_4 of a fetch cycle are used to refresh dynamic memories while the CPU is internally decoding and executing the instruction. The refresh control signal $\overline{RFSH}$ indicates that a refresh read of all dynamic memories should be accomplished.



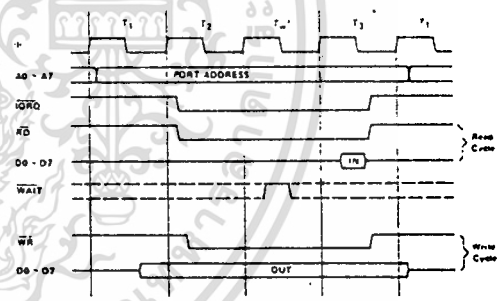
MEMORY READ OR WRITE CYCLES

Illustrated here is the timing of memory read or write cycles other than an OP code fetch (M_1 cycle). The $\overline{MREQ}$ and $\overline{RD}$ signals are used exactly as in the fetch cycle. In the case of a memory write cycle, the $\overline{MREQ}$ also becomes active when the address bus is stable so that it can be used directly as a chip enable for dynamic memories. The $\overline{WR}$ line is active when data on the data bus is stable so that it can be used directly as a R/W pulse to virtually any type of semiconductor memory.



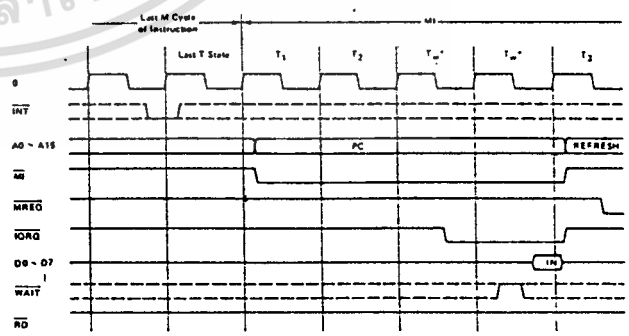
INPUT OR OUTPUT CYCLES

Illustrated here is the timing for an I/O read or I/O write operation. Notice that during I/O operations a single wait state is automatically inserted (Tw*). The reason for this is that during I/O operations this extra state allows sufficient time for an I/O port to decode its address and activate the $\overline{WAIT}$ line if a wait is required.



INTERRUPT REQUEST/ACKNOWLEDGE CYCLE

The interrupt signal is sampled by the CPU with the rising edge of the last clock at the end of any instruction. When an interrupt is accepted, a special M_1 cycle is generated. During this M_1 cycle, the $\overline{IORQ}$ signal becomes active (instead of $\overline{MREQ}$) to indicate that the interrupting device can place an 8-bit vector on the data bus. Two wait states (Tw*) are automatically added to this cycle so that a ripple priority interrupt scheme, such as the one used in the Z80 peripheral controllers, can be easily implemented.



Z80, Z80A Instruction Set

The following is a summary of the Z80, Z80A instruction set showing the assembly language mnemonic and the symbolic operation performed by the instruction. A more detailed listing appears in the Z80-CPU technical manual, and assembly language programming manual. The instructions are divided into the following categories:

8-bit loads	Miscellaneous Group
16-bit loads	Rotates and Shifts
Exchanges	Bit Set, Reset and Test
Memory Block Moves	Input and Output
Memory Block Searches	Jumps
8-bit arithmetic and logic	Calls
16-bit arithmetic	Restarts
General purpose Accumulator & Flag Operations	Returns

In the table the following terminology is used.

b ≡ a bit number in any 8-bit register or memory location

cc ≡ flag condition code
- NZ ≡ non zero
- Z ≡ zero
- NC ≡ non carry
- C ≡ carry
- PO ≡ Parity odd or no over flow
- PE ≡ Parity even or over flow
- P ≡ Positive
- M ≡ Negative (minus)

d ≡ any 8-bit destination register or memory location

dd ≡ any 16-bit destination register or memory location

e ≡ 8-bit signed 2's complement displacement used in relative jumps and indexed addressing

L ≡ 8 special call locations in page zero. In decimal notation these are 0, 8, 16, 24, 32, 40, 48 and 56

n ≡ any 8-bit binary number

nn ≡ any 16-bit binary number

r ≡ any 8-bit general purpose register (A, B, C, D, E, H, or L)

s ≡ any 8-bit source register or memory location

s_b ≡ a bit in a specific 8-bit register or memory location

ss ≡ any 16-bit source register or memory location

subscript "L" ≡ the low order 8 bits of a 16-bit register

subscript "H" ≡ the high order 8 bits of a 16-bit register

( ) ≡ the contents within the ( ) are to be used as a pointer to a memory location or I/O port number

8-bit registers are A, B, C, D, E, H, L, I and R

16-bit register pairs are AF, BC, DE and HL

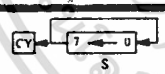
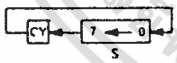
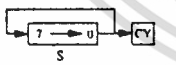
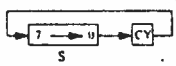
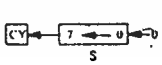
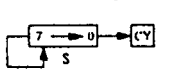
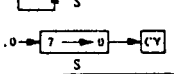
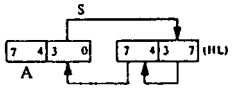
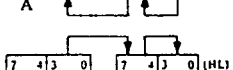
16-bit registers are SP, PC, IX and IY

Addressing Modes implemented include combinations of the following:

Immediate	Indexed
Immediate extended	Register
Modified Page Zero	Implied
Relative	Register Indirect
Extended	Bit

	Mnemonic	Symbolic Operation	Comments
8-BIT LOADS	LD r, s	r ← s	s ≡ r, n, (HL), (IX+e), (IY+e)
	LD d, r	d ← r	d ≡ (HL), r (IX+e), (IY+e)
	LD d, n	d ← n	d ≡ (HL), (IX+e), (IY+e)
	LD A, s	A ← s	s ≡ (BC), (DE), (nn), I, R
	LD d, A	d ← A	d ≡ (BC), (DE), (nn), I, R
16-BIT LOADS	LD dd, nn	dd ← nn	dd ≡ BC, DE, HL, SP, IX, IY
	LD dd, (nn)	dd ← (nn)	dd ≡ BC, DE, HL, SP, IX, IY
	LD (nn), ss	(nn) ← ss	ss ≡ BC, DE, HL, SP, IX, IY
	LD SP, ss	SP ← ss	ss = HL, IX, IY
	PUSH ss	$(SP-1) \leftarrow ss_H$; $(SP-2) \leftarrow ss_L$	ss = BC, DE, HL, AF, IX, IY
	POP dd	$dd_L \leftarrow (SP)$; $dd_H \leftarrow (SP+1)$	dd = BC, DE, HL, AF, IX, IY
EXCHANGES	EX DE, HL	DE ↔ HL	
	EX AF, AF'	AF ↔ AF'	
	EXX	$\begin{pmatrix} BC \\ DE \\ HL \end{pmatrix} \leftrightarrow \begin{pmatrix} BC' \\ DE' \\ HL' \end{pmatrix}$	
	EX (SP), ss	$(SP) \leftrightarrow ss_L$, $(SP+1) \leftrightarrow ss_H$	ss ≡ HL, IX, IY

	Mnemonic	Symbolic Operation	Comments
MEMORY BLOCK MOVES	LDI	(DE) ← (HL), DE ← DE+1 HL ← HL+1, BC ← BC-1	
	LDIR	(DE) ← (HL), DE ← DE+1 HL ← HL+1, BC ← BC-1 Repeat until BC = 0	
	LDD	(DE) ← (HL), DE ← DE-1 HL ← HL-1, BC ← BC-1	
	LDDR	(DE) ← (HL), DE ← DE-1 HL ← HL-1, BC ← BC-1 Repeat until BC = 0	
MEMORY BLOCK SEARCHES	CPI	A-(HL), HL ← HL+1 BC ← BC-1	A-(HL) sets the flags only. A is not affected
	CPIR	A-(HL), HL ← HL+1 BC ← BC-1, Repeat until BC = 0 or A = (HL)	
	CPD	A-(HL), HL ← HL-1 BC ← BC-1	
	CPDR	A-(HL), HL ← HL-1 BC ← BC-1, Repeat until BC= 0 or A = (HL)	
8-BIT ALU	ADD s	A ← A + s	CY is the carry flag s ≡ r, n, (HL) (IX+e), (IY+e)
	ADC s	A ← A + s + CY	
	SUB s	A ← A - s	
	SBC s	A ← A - s - CY	
	AND s	A ← A ∧ s	
	OR s	A ← A ∨ s	
	XOR s	A ← A ⊕ s	

	Mnemonic	Symbolic Operation	Comments
8-BIT ALU	CP s	A - s	s = r, n, (HL), (IX+e), (IY+e)
	INC d	d ← d + 1	d = r, (HL), (IX+e), (IY+e)
	DEC d	d ← d - 1	
16-BIT ARITHMETIC	ADD HL, ss	HL ← HL + ss	ss ≡ BC, DE, HL, SP
	ADC HL, ss	HL ← HL + ss + CY	
	SBC HL, ss	HL ← HL - ss - CY	
	ADD IX, ss	IX ← IX + ss	ss ≡ BC, DE, IX, SP
	ADD IY, ss	IY ← IY + ss	ss ≡ BC, DE, IY, SP
	INC dd	dd ← dd + 1	dd ≡ BC, DE, HL, SP, IX, IY
	DEC dd	dd ← dd - 1	dd ≡ BC, DE, HL, SP, IX, IY
GP ACC. & FLAG	DAA	Converts A contents into packed BCD following add or subtract.	Operands must be in packed BCD format
	CPL	$A \leftarrow \overline{A}$	
	NEG	A ← 00 - A	
	CCF	$CY \leftarrow \overline{CY}$	
	SCF	CY ← 1	
MISCELLANEOUS	NOP	No operation	
	HALT	Halt CPU	
	DI	Disable Interrupts	
	EI	Enable Interrupts	
	IM 0	Set interrupt mode 0	8080A mode
	IM 1	Set interrupt mode 1	Call to 0038_H
	IM 2	Set interrupt mode 2	Indirect Call
ROTATES AND SHIFTS	RLC s		s ≡ r, (HL), (IX+e), (IY+e)
	RL s		
	RRC s		
	RR s		
	SLA s		
	SRA s		
	SRL s		
	RLD		
	RRD		

	Mnemonic	Symbolic Operation	Comments
BIT S, R, & T	BIT b, s	$Z \leftarrow \overline{s_b}$	Z is zero flag
	SET b, s	$s_b \leftarrow 1$	s ≡ r, (HL), (IX+e), (IY+e)
	RES b, s	$s_b \leftarrow 0$	
INPUT AND OUTPUT	IN A, (n)	A ← (n)	
	IN r, (C)	r ← (C)	Set flags
	INI	(HL) ← (C), HL ← HL + 1, B ← B - 1	
	INIR	(HL) ← (C), HL ← HL + 1, B ← B - 1, Repeat until B = 0	
	IND	(HL) ← (C), HL ← HL - 1, B ← B - 1	
	INDR	(HL) ← (C), HL ← HL - 1, B ← B - 1, Repeat until B = 0	
	OUT(n), A	(n) ← A	
	OUT(C), r	(C) ← r	
	OUTI	(C) ← (HL), HL ← HL + 1, B ← B - 1	
	OTIR	(C) ← (HL), HL ← HL + 1, B ← B - 1, Repeat until B = 0	
	OUTD	(C) ← (HL), HL ← HL - 1, B ← B - 1	
	OTDR	(C) ← (HL), HL ← HL - 1, B ← B - 1, Repeat until B = 0	
JUMPS	JP nn	PC ← nn	cc ≡ NZ PO, Z PE, NC P, C M
	JP cc, nn	If condition cc is true PC ← nn, else continue	
	JR e	PC ← PC + e	kk ≡ NZ NC, Z C
	JR kk, e	If condition kk is true PC ← PC + e, else continue	ss = HL, IX, IY
	JP (ss)	PC ← ss	
	DJNZ e	B ← B - 1, if B = 0 continue, else PC ← PC + e	
CALLS	CALL nn	$(SP-1) \leftarrow PC_H$, $(SP-2) \leftarrow PC_L$, PC ← nn	cc ≡ NZ PO, Z PE, NC P, C M
	CALL cc, nn	If condition cc is false continue, else same as CALL nn	
RESTARTS	RST L	$(SP-1) \leftarrow PC_H$, $(SP-2) \leftarrow PC_L$, $PC_H \leftarrow 0$, $PC_L \leftarrow L$	
RETURNS	RET	$PC_L \leftarrow (SP)$, $PC_H \leftarrow (SP+1)$	cc ≡ NZ PO, Z PE, NC P, C M
	RET cc	If condition cc is false continue, else same as RET	
	RETI	Return from interrupt, same as RET	
	RETN	Return from non-maskable interrupt	

T_A = 0°C to 70°C, Vcc = +5V ± 5%, Unless Otherwise Noted.

Signal	Symbol	Parameter	Min	Max	Unit	Test Condition
Φ	t_c	Clock Period	4	[12]	μsec	
	$t_{w(\Phi H)}$	Clock Pulse Width, Clock High	180	[E]	nsec	
	$t_{w(\Phi L)}$	Clock Pulse Width, Clock Low	180	2000	nsec	
	$t_{r,f}$	Clock Rise and Fall Time		30	nsec	
A_{0-15}	$t_{D(AD)}$	Address Output Delay		145	nsec	C_L = 50pF
	$t_{F(AD)}$	Delay to Float		110	nsec	
	t_{acm}	Address Stable Prior to $\overline{MREQ}$ (Memory Cycle)	[1]		nsec	
	t_{aci}	Address Stable Prior to $\overline{IORQ}$, $\overline{RD}$, or $\overline{WR}$ (I/O Cycle)	[2]		nsec	
	t_{ca}	Address Stable from $\overline{RD}$, $\overline{WR}$, $\overline{IORQ}$ or $\overline{MREQ}$	[3]		nsec	
	t_{caf}	Address Stable From $\overline{RD}$ or $\overline{WR}$ During Float	[4]		nsec	
D_{0-7}	$t_{D(D)}$	Data Output Delay		230	nsec	C_L = 50pF
	$t_{F(D)}$	Delay to Float During Write Cycle		90	nsec	
	$t_{S\Phi(D)}$	Data Setup Time to Rising Edge of Clock During M1 Cycle	50		nsec	
	$t_{S\overline{\Phi}(D)}$	Data Setup Time to Falling Edge of Clock During M2 to M5	60		nsec	
	t_{dcm}	Data Stable Prior to $\overline{WR}$ (Memory Cycle)	[5]	—	nsec	
	t_{dci}	Data Stable Prior to $\overline{WR}$ (I/O Cycle)	[6]		nsec	
	t_{cdf}	Data Stable From $\overline{WR}$	[7]			
	t_H	Any Hold Time for Setup Time	0		nsec	
$\overline{MRFQ}$	$t_{DL\Phi(MR)}$	$\overline{MREQ}$ Delay From Falling Edge of Clock, $\overline{MREQ}$ Low		100	nsec	C_L = 50pF
	$t_{DH\Phi(MR)}$	$\overline{MREQ}$ Delay From Rising Edge of Clock, $\overline{MREQ}$ High		100	nsec	
	$t_{DH\overline{\Phi}(MR)}$	$\overline{MREQ}$ Delay From Falling Edge of Clock, $\overline{MREQ}$ High		100	nsec	
	$t_{w(MRL)}$	Pulse Width, $\overline{MREQ}$ Low	[8]		nsec	
	$t_{w(MRH)}$	Pulse Width, $\overline{MREQ}$ High	[9]		nsec	
$\overline{IORQ}$	$t_{DL\Phi(IR)}$	$\overline{IORQ}$ Delay From Rising Edge of Clock, $\overline{IORQ}$ Low		90	nsec	C_L = 50pF
	$t_{DL\overline{\Phi}(IR)}$	$\overline{IORQ}$ Delay From Falling Edge of Clock, $\overline{IORQ}$ Low		110	nsec	
	$t_{DH\Phi(IR)}$	$\overline{IORQ}$ Delay From Rising Edge of Clock, $\overline{IORQ}$ High		100	nsec	
	$t_{DH\overline{\Phi}(IR)}$	$\overline{IORQ}$ Delay From Falling Edge of Clock, $\overline{IORQ}$ High		110	nsec	
$\overline{RD}$	$t_{DL\Phi(RD)}$	$\overline{RD}$ Delay From Rising Edge of Clock, $\overline{RD}$ Low		100	nsec	C_L = 50pF
	$t_{DL\overline{\Phi}(RD)}$	$\overline{RD}$ Delay From Falling Edge of Clock, $\overline{RD}$ Low		130	nsec	
	$t_{DH\Phi(RD)}$	$\overline{RD}$ Delay From Rising Edge of Clock, $\overline{RD}$ High		100	nsec	
	$t_{DH\overline{\Phi}(RD)}$	$\overline{RD}$ Delay From Falling Edge of Clock, $\overline{RD}$ High		110	nsec	
$\overline{WR}$	$t_{DL\Phi(WR)}$	$\overline{WR}$ Delay From Rising Edge of Clock, $\overline{WR}$ Low		80	nsec	C_L = 50pF
	$t_{DL\overline{\Phi}(WR)}$	$\overline{WR}$ Delay From Falling Edge of Clock, $\overline{WR}$ Low		90	nsec	
	$t_{DH\overline{\Phi}(WR)}$	$\overline{WR}$ Delay From Falling Edge of Clock, $\overline{WR}$ High		100	nsec	
	$t_{w(WRL)}$	Pulse Width, $\overline{WR}$ Low	[10]		nsec	
$\overline{M1}$	$t_{DL(M1)}$	$\overline{M1}$ Delay From Rising Edge of Clock, $\overline{M1}$ Low		130	nsec	C_L = 50pF
	$t_{DH(M1)}$	$\overline{M1}$ Delay From Rising Edge of Clock, $\overline{M1}$ High		130	nsec	
$\overline{RFSH}$	$t_{DL(RF)}$	$\overline{RFSH}$ Delay From Rising Edge of Clock, $\overline{RFSH}$ Low		180	nsec	C_L = 50pF
	$t_{DH(RF)}$	$\overline{RFSH}$ Delay From Rising Edge of Clock, $\overline{RFSH}$ High		150	nsec	
$\overline{WAIT}$	$t_{s(WT)}$	$\overline{WAIT}$ Setup Time to Falling Edge of Clock	70		nsec	
$\overline{HALT}$	$t_{D(HT)}$	$\overline{HALT}$ Delay Time From Falling Edge of Clock		300	nsec	C_L = 50pF
$\overline{INT}$	$t_{s(IT)}$	$\overline{INT}$ Setup Time to Rising Edge of Clock	80		nsec	
$\overline{NMI}$	$t_{w(NML)}$	Pulse Width, $\overline{NMI}$ Low	80		nsec	
$\overline{BUSRQ}$	$t_{s(BQ)}$	$\overline{BUSRQ}$ Setup Time to Rising Edge of Clock	80		nsec	
$\overline{BUSAK}$	$t_{DL(BA)}$	$\overline{BUSAK}$ Delay From Rising Edge of Clock, $\overline{BUSAK}$ Low		120	nsec	C_L = 50pF
	$t_{DH(BA)}$	$\overline{BUSAK}$ Delay From Falling Edge of Clock, $\overline{BUSAK}$ High		110	nsec	
$\overline{RESET}$	$t_{s(RS)}$	$\overline{RESET}$ Setup Time to Rising Edge of Clock	90		nsec	
	$t_{F(C)}$	Delay to Float ($\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$ and $\overline{WR}$)		100	nsec	
	t_{mr}	$\overline{M1}$ Stable Prior to $\overline{IORQ}$ (Interrupt Ack.)	[11]		nsec	

[1] $t_{acm} = t_{w(\Phi H)} + t_f - 75$

[2] $t_{aci} = t_c - 80$

[3] $t_{ca} = t_{w(\Phi L)} + t_r - 40$

[4] $t_{caf} = t_{w(\Phi L)} + t_r - 60$

[5] $t_{dcm} = t_c - 210$

[6] $t_{dci} = t_{w(\Phi L)} + t_r - 210$

[7] $t_{cdf} = t_{w(\Phi L)} + t_r - 80$

[8] $t_{w(MRL)} = t_c - 40$

[9] $t_{w(MRH)} = t_{w(\Phi H)} + t_r - 30$

[10] $t_{w(\overline{WRL})} = t_c - 40$

[11] $t_{mr} = 2t_c + t_{w(\Phi H)} + t_f - 80$

[12] $t_c = t_{w(\Phi H)} + t_{w(\Phi L)} + t_r + t_f$

NOTES:

A. Data should be enabled onto the CPU data bus when $\overline{RD}$ is active. During interrupt acknowledge data should be enabled when $\overline{M1}$ and $\overline{IORQ}$ are both active.

B. All control signals are internally synchronized, so they may be totally asynchronous with respect to the clock.

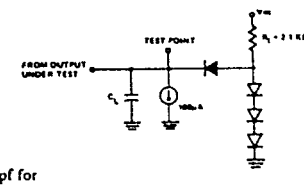
C. The $\overline{RESET}$ signal must be active for a minimum of 3 clock cycles.

D. Output Delay vs. Loaded Capacitance

TA = 70°C Vcc = +5V ±5%

Add 10nsec delay for each 50pf increase in load up to a maximum of 200pf for the data bus & 100pf for address & control lines

E. Although static by design, testing guarantees $t_{w(\Phi H)}$ of 200 μsec maximum

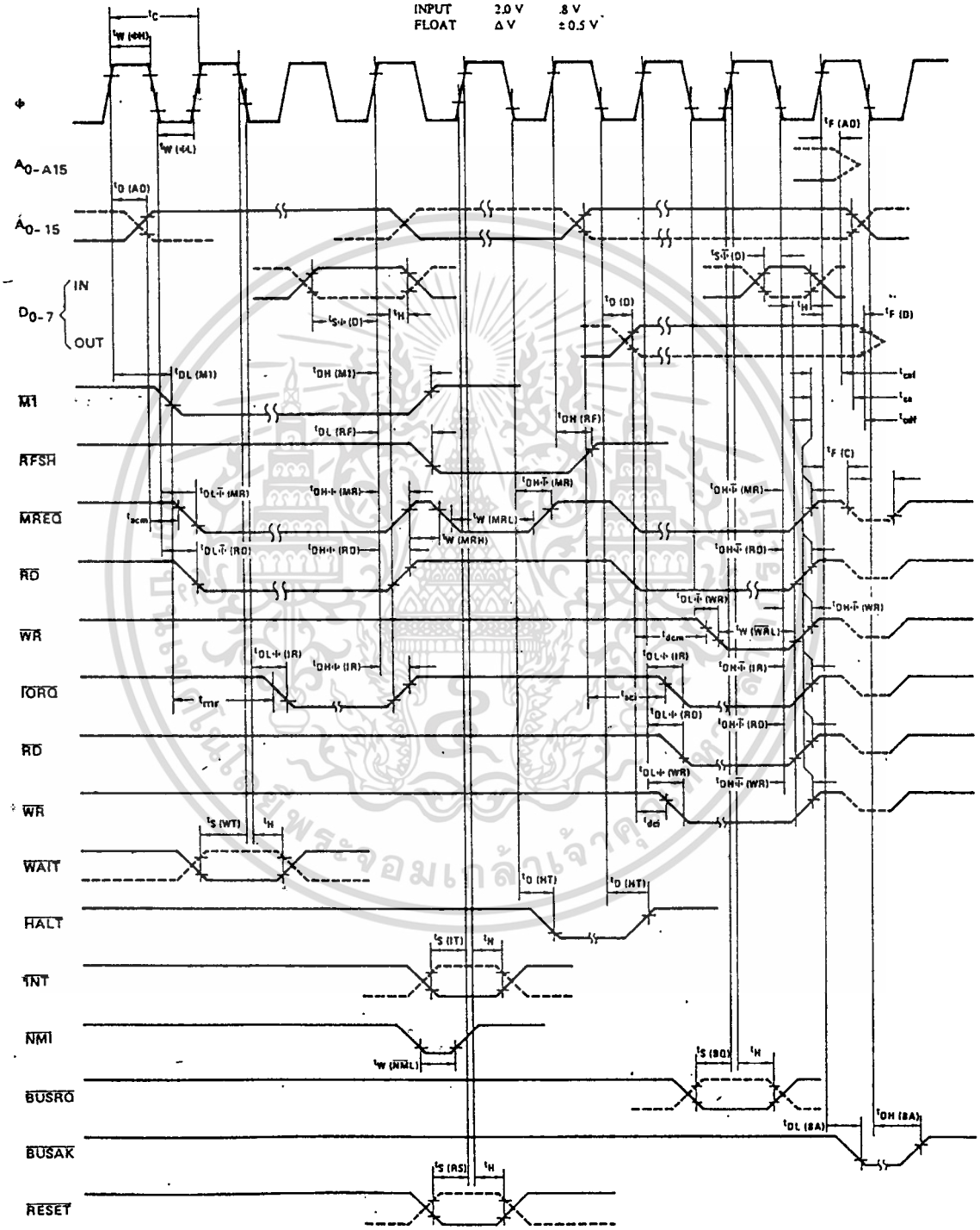


Load circuit for Output

A.C. Timing Diagram

Timing measurements are made at the following voltages, unless otherwise specified:

	"1"	"0"
CLOCK	V_{cc} -.6V	.45V
OUTPUT	2.0 V	.8 V
INPUT	2.0 V	.8 V
FLOAT	Δ V	± 0.5 V



Absolute Maximum Ratings

Temperature Under Bias	Specified operating range.
Storage Temperature	-65°C to +150°C
Voltage On Any Pin with Respect to Ground	-0.3V to +7V
Power Dissipation	1.5 W

*Comment

Stresses above those listed under "Absolute Maximum Rating" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Note. For Z80-CPU all AC and DC characteristics remain the same for the military grade parts except I_{CC}.

I_{CC} = 200 mA

Z80-CPU D.C. Characteristics

T_A = 0°C to 70°C, V_{CC} = 5V ± 5% unless otherwise specified

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	-0.3		0.45	V	
V_{IHC}	Clock Input High Voltage	V_{CC} -.6		V_{CC}+.3	V	
V_{IL}	Input Low Voltage	-0.3		0.8	V	
V_{IH}	Input High Voltage	2.0		V_{CC}	V	
V_{OL}	Output Low Voltage			0.4	V	I_{OL}=1.8mA
V_{OH}	Output High Voltage	2.4			V	I_{OH} = -250μA
I_{CC}	Power Supply Current			150	mA	
I_{LI}	Input Leakage Current			10	μA	V_{IN}=0 to V_{CC}
I_{LOH}	Tri-State Output Leakage Current in Float			10	μA	V_{OUT}=2.4 to V_{CC}
I_{LOL}	Tri-State Output Leakage Current in Float			-10	μA	V_{OUT}=0.4V
I_{LD}	Data Bus Leakage Current in Input Mode			±10	μA	$0 \le V_{IN} \le V_{CC}$

Capacitance

T_A = 25°C, f = 1 MHz,
unmeasured pins returned to ground

Symbol	Parameter	Max.	Unit
C_Φ	Clock Capacitance	35	pF
C_{IN}	Input Capacitance	5	pF
C_{OUT}	Output Capacitance	10	pF

Z80-CPU Ordering Information

C – Ceramic
P – Plastic
S – Standard 5V ±5% 0° to 70°C
E – Extended 5V ±5% -40° to 85°C
M – Military 5V ±10% -55° to 125°C

Z80A-CPU D.C. Characteristics

T_A = 0°C to 70°C, V_{CC} = 5V ± 5% unless otherwise specified

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Condition
V_{ILC}	Clock Input Low Voltage	-0.3		0.45	V	
V_{IHC}	Clock Input High Voltage	V_{CC} -.6		V_{CC}+.3	V	
V_{IL}	Input Low Voltage	-0.3		0.8	V	
V_{IH}	Input High Voltage	2.0		V_{CC}	V	
V_{OL}	Output Low Voltage			0.4	V	I_{OL}=1.8mA
V_{OH}	Output High Voltage	2.4			V	I_{OH} = -250μA
I_{CC}	Power Supply Current		90	200	mA	
I_{LI}	Input Leakage Current			10	μA	V_{IN}=0 to V_{CC}
I_{LOH}	Tri-State Output Leakage Current in Float			10	μA	V_{OUT}=2.4 to V_{CC}
I_{LOL}	Tri-State Output Leakage Current in Float			-10	μA	V_{OUT}=0.4V
I_{LD}	Data Bus Leakage Current in Input Mode			±10	μA	$0 \le V_{IN} \le V_{CC}$

Capacitance

T_A = 25°C, f = 1 MHz,
*unmeasured pins returned to ground

Symbol	Parameter	Max.	Unit
C_Φ	Clock Capacitance	35	pF
C_{IN}	Input Capacitance	5	pF
C_{OUT}	Output Capacitance	10	pF

*Z80A-CPU Ordering Information

C – Ceramic
P – Plastic
S – Standard 5V ±5% 0° to 70°C

T_A = 0°C to 70°C, Vcc = +5V ± 5%, Unless Otherwise Noted.

Signal	Symbol	Parameter	Min	Max	Unit	Test Condition
Φ	t_c	Clock Period	250	[12]	nsec	
	$t_{w(\Phi H)}$	Clock Pulse Width, Clock High	110	[E]	nsec	
	$t_{w(\Phi L)}$	Clock Pulse Width, Clock Low	110	2000	nsec	
	t_r, t_f	Clock Rise and Fall Time		30	nsec	
A_{0-15}	$t_{D(AD)}$	Address Output Delay		110	nsec	C_L = 50pF
	$t_{F(AD)}$	Delay to Float		90	nsec	
	t_{acm}	Address Stable Prior to $\overline{MREQ}$ (Memory Cycle)	[1]		nsec	
	t_{aci}	Address Stable Prior to $\overline{IORQ}$, $\overline{RD}$ or $\overline{WR}$ (I/O Cycle)	[2]		nsec	
	t_{ca}	Address Stable from $\overline{RD}$, $\overline{WR}$, $\overline{IORQ}$ or $\overline{MREQ}$	[3]		nsec	
	t_{caf}	Address Stable From $\overline{RD}$ or $\overline{WR}$ During Float	[4]		nsec	
D_{0-7}	$t_{D(D)}$	Data Output Delay		150	nsec	C_L = 50pF
	$t_{F(D)}$	Delay to Float During Write Cycle		90	nsec	
	$t_{S\Phi(D)}$	Data Setup Time to Rising Edge of Clock During M1 Cycle	35		nsec	
	$t_{S\overline{\Phi}(D)}$	Data Setup Time to Falling Edge of Clock During M2 to M5	50		nsec	
	t_{dcm}	Data Stable Prior to $\overline{WR}$ (Memory Cycle)	[5]		nsec	
	t_{dci}	Data Stable Prior to $\overline{WR}$ (I/O Cycle)	[6]		nsec	
	t_{cdf}	Data Stable From $\overline{WR}$	[7]			
	t_H	Any Hold Time for Setup Time		0	nsec	
$\overline{MREQ}$	$t_{DL\Phi(MR)}$	$\overline{MREQ}$ Delay From Falling Edge of Clock, $\overline{MREQ}$ Low		85	nsec	C_L = 50pF
	$t_{DH\Phi(MR)}$	$\overline{MREQ}$ Delay From Rising Edge of Clock, $\overline{MREQ}$ High		85	nsec	
	$t_{DH\overline{\Phi}(MR)}$	$\overline{MREQ}$ Delay From Falling Edge of Clock, $\overline{MREQ}$ High		85	nsec	
	$t_{w(\overline{MRL})}$	Pulse Width, $\overline{MREQ}$ Low	[8]		nsec	
	$t_{w(\overline{MRH})}$	Pulse Width, $\overline{MREQ}$ High	[9]		nsec	
$\overline{IORQ}$	$t_{DL\Phi(IR)}$	$\overline{IORQ}$ Delay From Rising Edge of Clock, $\overline{IORQ}$ Low		75	nsec	C_L = 50pF
	$t_{DL\overline{\Phi}(IR)}$	$\overline{IORQ}$ Delay From Falling Edge of Clock, $\overline{IORQ}$ Low		85	nsec	
	$t_{DH\Phi(IR)}$	$\overline{IORQ}$ Delay From Rising Edge of Clock, $\overline{IORQ}$ High		85	nsec	
	$t_{DH\overline{\Phi}(IR)}$	$\overline{IORQ}$ Delay From Falling Edge of Clock, $\overline{IORQ}$ High		85	nsec	
$\overline{RD}$	$t_{DL\Phi(RD)}$	$\overline{RD}$ Delay From Rising Edge of Clock, $\overline{RD}$ Low		85	nsec	C_L = 50pF
	$t_{DL\overline{\Phi}(RD)}$	$\overline{RD}$ Delay From Falling Edge of Clock, $\overline{RD}$ Low		95	nsec	
	$t_{DH\Phi(RD)}$	$\overline{RD}$ Delay From Rising Edge of Clock, $\overline{RD}$ High		85	nsec	
	$t_{DH\overline{\Phi}(RD)}$	$\overline{RD}$ Delay From Falling Edge of Clock, $\overline{RD}$ High		85	nsec	
$\overline{WR}$	$t_{DL\Phi(WR)}$	$\overline{WR}$ Delay From Rising Edge of Clock, $\overline{WR}$ Low		65	nsec	C_L = 50pF
	$t_{DL\overline{\Phi}(WR)}$	$\overline{WR}$ Delay From Falling Edge of Clock, $\overline{WR}$ Low		80	nsec	
	$t_{DH\overline{\Phi}(WR)}$	$\overline{WR}$ Delay From Falling Edge of Clock, $\overline{WR}$ High		80	nsec	
	$t_{w(\overline{WRL})}$	Pulse Width, $\overline{WR}$ Low	[10]		nsec	
$\overline{M1}$	$t_{DL(M1)}$	$\overline{M1}$ Delay From Rising Edge of Clock, $\overline{M1}$ Low		100	nsec	C_L = 50pF
	$t_{DH(M1)}$	$\overline{M1}$ Delay From Rising Edge of Clock, $\overline{M1}$ High		100	nsec	
$\overline{RFSH}$	$t_{DL(RF)}$	$\overline{RFSH}$ Delay From Rising Edge of Clock, $\overline{RFSH}$ Low		130	nsec	C_L = 50pF
	$t_{DH(RF)}$	$\overline{RFSH}$ Delay From Rising Edge of Clock, $\overline{RFSH}$ High		120	nsec	
$\overline{WAIT}$	$t_{s(WT)}$	$\overline{WAIT}$ Setup Time to Falling Edge of Clock	70		nsec	
$\overline{HALT}$	$t_{D(HT)}$	$\overline{HALT}$ Delay Time From Falling Edge of Clock		300	nsec	C_L = 50pF
$\overline{INT}$	$t_{s(IT)}$	$\overline{INT}$ Setup Time to Rising Edge of Clock	80		nsec	
$\overline{NMI}$	$t_{w(\overline{NML})}$	Pulse Width, $\overline{NMI}$ Low	80		nsec	
$\overline{BUSRQ}$	$t_{s(BQ)}$	$\overline{BUSRQ}$ Setup Time to Rising Edge of Clock	50		nsec	
$\overline{BUSAK}$	$t_{DL(BA)}$	$\overline{BUSAK}$ Delay From Rising Edge of Clock, $\overline{BUSAK}$ Low		100	nsec	C_L = 50pF
	$t_{DH(BA)}$	$\overline{BUSAK}$ Delay From Falling Edge of Clock, $\overline{BUSAK}$ High		100	nsec	
$\overline{RESET}$	$t_{s(RS)}$	$\overline{RESET}$ Setup Time to Rising Edge of Clock	60		nsec	
	$t_{F(C)}$	Delay to Float ($\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$ and $\overline{WR}$)		80	nsec	
	t_{mr}	M1 Stable Prior to IORQ (Interrupt Ack.)	[11]		nsec	

[12] $t_c = t_{w(\Phi H)} + t_{w(\Phi L)} + t_r + t_f$

[1] $t_{acm} = t_{w(\Phi H)} + t_f - 65$

[2] $t_{aci} = t_c - 70$

[3] $t_{ca} = t_{w(\Phi L)} + t_r - 50$

[4] $t_{caf} = t_{w(\Phi L)} + t_r - 45$

[5] $t_{dcm} = t_c - 170$

[6] $t_{dci} = t_{w(\Phi L)} + t_r - 170$

[7] $t_{cdf} = t_{w(\Phi L)} + t_r - 70$

[8] $t_{w(\overline{MRL})} = t_c - 30$

[9] $t_{w(\overline{MRH})} = t_{w(\Phi H)} + t_f - 20$

[10] $t_{w(\overline{WRL})} = t_c - 30$

[11] $t_{mr} = 2t_c + t_{w(\Phi H)} + t_f - 65$

NOTES:

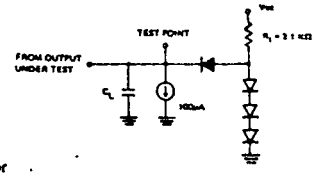
A. Data should be enabled onto the CPU data bus when $\overline{RD}$ is active. During interrupt acknowledge data should be enabled when $\overline{M1}$ and $\overline{IORQ}$ are both active.

B. All control signals are internally synchronized, so they may be totally asynchronous with respect to the clock.

C. The $\overline{RESET}$ signal must be active for a minimum of 3 clock cycles.

D. Output Delays vs. Loaded Capacitance
TA = 70°C Vcc = +5V ±5%
Add 10nsec delay for each 50pf increase in load up to maximum of 200pf for data bus and 100pf for address & control lines.

E. Although static by design, testing guarantees $t_{w(\Phi H)}$ of 200 μsec maximum



Load circuit for Output

หนังสืออ้างอิง

1.) Lance A. Leventhal, "Z 80 Assembly Language Programming", McGraw-Hill, California, International Editions, 1979.

2.) ETT, "Z 80 ET-Board", ETT Co., Ltd., 1st edition, 1989.

3.) Takashi Kenjo, " Stepping motros and their microprocessor controls ", Clarendon Pres, Oxford, Paperback edition, 1985.

4.) กฤษดา วิศวธีรานนท์ ยิน ภู่รรวรรณ,. "ไมโครโปรเซสเซอร์"

5.) ETT, "ET Hardware Lab Experiment", ETT Co., Ltd., 1990.

6.) วารสารแซมิคอนดัคเตอร์ อิเลคทรอนิคส์ ฉบับที่ 85,81.