อุปกรณ์ส่งถ่ายข้อมูลอัตโนมัติ

โดย

นายเฉลียว บวบทอง

นายบัณฑิต ปั้นรัตน์

นายวิบูลย์ ทองอู่

นายวัลลภ นิยม

นายสมศักดิ์ ลี้เจริญ

ปริญญานิพนธ์วิศวกรรมศาสตร์บัณฑิต

ภาควิชาเทคนิคอุตสาหกรรม คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2533

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ได้สำเร็จรุล่วงไปด้วยดี เนื่องจากได้รับอนุเคราะห์ จากท่านอาจารย์ กฤดากร กล่อมการ ซึ่งเป็นอาจารย์ที่ปรึกษา ตลอดการทำปริญญานิพนธ์ฉบับนี้ พร้อมทั้งอาจารย์อีกหลายท่านที่ได้ให้ข้อมูลและคำแนะนำ อันเป็นประโยชน์ แก่คณะผู้จัดทำ จึงขอบพระคุณอย่างสูงมา ณ ที่นี้

นาย เฉลียว บวบทอง

นาย บัณฑิตย์ ปั้นรัตย์

นาย วิบูลย์ ทองอู่

นาย วัลลภ นิยม

นาย สมศักดิ์ ลี้เจริญ

นักศึกษา ภาควิชาเทคนิคอุตสาหกรรม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้า

เจ้าคุณทหารลาดกระบัง

ชื่อเรื่องปริญญานิพนธ์ อุปกรณ์ส่งถ่ายข้อมูลอัตโนมัติ

จัดทำโดย นาย เฉลียว บวบทอง 31-3005
นาย บันทิตย์ ปั้นรัตย์ 31-3015
นาย วิบูลย์ ทองยู่ 31-3022
นาย วัฒนา นิยม 31-3023
นาย สมศักดิ์ สี่เจริญ 31-3024

หลักสูตร เทคนิคอุตสาหกรรม

อาจารย์ที่ปรึกษา ดร. ทนก เจนจิระพงศ์เวช
อาจารย์ กฤตากร กล่อมการ

คณะวิศวกรรมศาสตร์ เทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง อนุมัติให้นำปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาอุตสาหกรรม

.....ประธานกรรมการ

.....กรรมการ

.....กรรมการ

.....กรรมการ

คณบดีของคณะวิศวกรรมศาสตร์ เทคโนโลยีพระจอมเกล้า เจ้าคุณทหาร ลาดกระบัง

พ.ศ. 2552

วัตถุประสงค์ของโครงงาน

โครงงานนี้สร้างขึ้นเพื่อมุ่งหวังจะพัฒนาระบบส่งถ่ายข้อมูลที่นอกเหนือจากการรับส่งข้อมูลซึ่งเป็นโปรแกรมของการ Bond ตามปกติเพียงอย่างเดียว แต่พัฒนาขึ้นเพื่อรองรับการส่งข้อมูลทางด้านสถิติ เช่นข้อมูลทางด้านเวลาทำงานของเครื่อง Bond (Down time) และอื่น ๆซึ่งข้อมูลเหล่านี้จะเกิดขึ้นในเวลาที่ไม่แน่นอน ฉนั้นจึงจำเป็นที่ต้องใช้ระบบการรับส่งข้อมูลที่เป็นไปอย่างอัตโนมัติ ข้อมูลเหล่านี้จะถูกเก็บไว้ในไมโครคอมพิวเตอร์ เพื่อประมวลผลทางด้านสถิติต่อไป

โครงงานนี้ได้แบ่งออกเป็น 2 ขั้นตอนคือ

ขั้นตอนแรก จะทำการออกแบบระบบเพื่อให้ใช้ได้กับการส่งข้อมูลโปรแกรม การ Bond อย่างอัตโนมัติ แต่เพียงอย่างเดียวก่อน เพื่อเป็นต้นแบบหาข้อปกพร่องและแก้ไขในขั้นตอนต่อไป

ขั้นตอนที่สอง จะทำการพัฒนาในด้าน Hard Ware & Soft Ware เพื่อใช้รับส่งข้อมูลทางด้านสถิติทั้งในระบบส่งถ่ายข้อมูล และในไมโครคอมพิวเตอร์

ผลที่จะได้ในการพัฒนาในระบบแรก

1. สามารถเรียกใช้งานได้ทันที
2. สามารถเรียกใช้งานพร้อมกันได้
3. ลดเวลาในการเรียกใช้ข้อมูล มี Buffer อยู่ในตัวทำให้ใช้เวลาในการเรียกใช้ข้อมูลน้อยลงในกรณีที่มีข้อมูลอยู่ใน Buffer

อุปกรณ์ส่งถ่ายข้อมูลอัตโนมัติสำหรับเครื่องฟรีคอมบอนเดอร์

AUTOMATICS DATA TRANSFER FOR PHICOM WRIE BONDER

อุตสาหกรรมศาสตร์บัณฑิต

ภาควิชาเทคนิคอุตสาหกรรม

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้า

เจ้าคุณทหารลาดกระบัง

นาย เฉลียว บวบทอง

นาย บัณฑิตย์ ปั้นรัตย์

นาย วิบูลย์ ทองอู่

นาย วัลลภ นิยม

นาย สมศักดิ์ ลี้เจริญ

บทคัดย่อ

บทความนี้กล่าวถึงการออกแบบและการสร้างอุปกรณ์สับเปลี่ยนพอร์ทข้อมูล RS-232 แบบอัตโนมัติ เพื่อใช้ในการส่งผ่านข้อมูลระหว่างเครื่อง WIRE BONDER จำนวนมากกับไมโครคอมพิวเตอร์ 1 เครื่อง โครงสร้างของบัฟเฟอร์ข้อมูลจะใช้แบบจำลองการเข้าแถวรอคอย

ABSBRACT

THIS PAPER PRESCNTS DESIGNING AND CONSTRUCTION OF THE AUTOMATIC DATA TRANSFER SWITCHES FOR RS-232 BUS.THIS EQUIPMENT IS USED TO INTERFACE THE PHICOM WRIE BONDER MORE UNIT IN.QDER TO SHARE SINGLE MICROCOMPUTER THE BUFFER ARE DREIEVD BY USING QUEUEING MODEL.

สารบัญ

บทนำ

ในขบวนการผลิต IC นั้นมีขั้นตอนการผลิตหลายขั้นตอนด้วยกัน Wire Bond เป็นขั้นตอนที่สำคัญขั้นตอนหนึ่ง ขั้นตอนนี้จะทำการเชื่อมต่อวงจรของแผงวงจรเล็ก ๆ ซึ่งเรียกว่า DIE เข้ากับขาตัว IC ด้วยลวดทองขนาดเล็ก เพื่อเป็นทางผ่านของกระแสไฟฟ้าจากภายนอกตัว IC โดยใช้เครื่องจักรอัตโนมัติของบริษัท PHILIPS เรียกว่า PHICOM AUTOMETIC WIRE BONDER ซึ่งมีประสิทธิภาพในการทำงานสูงมาก

การที่จะให้เครื่องเริ่มต้นทำงานนั้น พนักงานต้องทำการป้อน โปรแกรมให้กับเครื่อง โดยมี รวมทั้งจุดอะไลเมนท์, บอนดิ้งพารามิเตอร์ และตำแหน่งจุดบอนด์ต่าง ๆ ตามชนิดของ DIE เพียงครั้งเดียวเครื่องก็จะทำการบอนด์งานตัวต่อไปโดยอัตโนมัติ โปรแกรมที่ป้อนให้กับเครื่องจะใช้กับ DIE แต่ละชนิดของ DIE เท่านั้นและจะสามารถเก็บข้อมูลโปรแกรมของโปรดักนั้น ไว้ใช้ในคราวต่อไป ใน ฟลอฟฟีดิสุ (Floppy disk)

การเก็บข้อมูล ลงไปในฟอลฟฟีดิสโดยผ่านทางพอร์ทอนุกรม RS-232 นั้นเริ่มแรกใช้เครื่องขับดิสขนาด 3 1/4 นิ้ว 1 เครื่องต่อเครื่อง Bond ประมาณ 5 เครื่อง ซึ่งก็เพียงพอต่อการใช้งานในขณะนั้น ซึ่งยังมีเครื่อง Bond ไม่มากนัก ต่อมาได้เพิ่มปริมาณการผลิต ทำให้ต้องใช้เครื่อง Bond เพิ่มขึ้น (ปริมาณ 100 เครื่อง) การใช้งานเครื่อง Disk drive นั้นมีเริ่มปัญหาเนื่องจากการติดตั้งเครื่อง Disk drive บนรถเข็นเคลื่อนที่ไปมาระหว่างการใช้งาน ทำให้เกิดการชำรุดเสียหาย การซ่อมบำรุงเป็นไปด้วยความยากลำบากเนื่องจากหา Spar part ไม่ได้จึงทำให้อัตราส่วน Disk drive กับเครื่อง Bond ลดลงทำให้ลำบากต่อการใช้งานมาก

ต่อมาได้แก้ปัญหาโดยการเก็บข้อมูลลงใน DISK ของเครื่องไมโครคอมพิวเตอร์ โดยใช้ไมโครคอมพิวเตอร์ 1 เครื่องต่อเครื่อง Bond 20 เครื่อง เดินสายผ่าน Switch box เพื่อเลือกว่าจะใช้ติดต่อกับเครื่อง Bond เครื่องใดซึ่งทำให้การใช้งานได้สะดวกขึ้นคือเก็บข้อมูลได้มากขึ้นไม่มีปัญหาในการซ่อมบำรุง

แต่อย่างไรก็ตามระบบที่ใช้งานยังเป็นระบบ Manual การเก็บข้อมูลหรือเรียกใช้ข้อมูล จากเครื่อง Bond จะต้องกด Switch เพื่อเลือกตำแหน่งเครื่องเสียก่อน และไม่สามารถเรียกใช้งานได้พร้อมกันได้จะทำให้เกิด Error ขึ้นและยังไม่สามารถส่งข้อมูลชนิดอื่น ๆได้ นอกจากการเก็บและเรียกใช้ข้อมูลโปรแกรมการ Bond เท่านั้น

โครงงานนี้เป็นการพัฒนาระบบการส่งถ่ายข้อมูลใหม่ ให้เป็นแบบอัตโนมัติ เพื่อเพิ่มประสิทธิภาพในการใช้งาน และยังสามารถส่งข้อมูลชนิดอื่น ๆได้

พีคอมบอนเดอร์ (PHICOM BONDER)

เครื่องพีคอมบอนเดอร์ เป็นเครื่องบอนด์ลวดทอง ทางด้านพลาสติก ซึ่งสามารถบอนด์งานแต่ละตัว ด้วยระบบอัตโนมัติและเร็วมาก

หน้าที่หลักและลักษณะเด่น ๆ ของเครื่องมีดังนี้ คือ

1. เครื่องนี้สามารถเลือกใช้ได้ทั้งระบบ เทอร์โม - คอมเพรสชั่น หรือระบบ เทอร์โม - โซนิค
2. สามารถใส่โปรแกรมจำนวนเส้นลวด ได้ถึง 125 เส้น
3. เลื่อนงานเข้ามาเอง และบอนด์อัตโนมัติ
4. เครื่องทำการวัดตำแหน่งของไดก่อนทำการบอนด์ด้วยเทคนิคการจำรูปแบบ แพทเทอร์นเร็กคอคิชั่น
5. สอนตัวเองได้
6. ใช้ระบบทีวี และกล้องถ่ายภาพ
7. มีระบบเออเรอร์ เมสเซส (รายงานเครื่องทำงานผิด)
8. ใช้อีเลคโทรนิค เฟรมออฟ (การทำลูกบอนด์ด้วยระบบไฟฟ้า)
9. มีระบบตรวจลวดขาด ในขณะทำการบอนด์
10. สามารถตั้งโปรแกรมส่วนโค้งย้อนกลับได้ (ลูฟคอนสะแต้นท์แก้ลวดนาบหน้าไดและลวดแอ่นตรงกลาง)
11. ควบคุมการทำงานทั้งหมดของเครื่องด้วยระบบคอมพิวเตอร์

พนักงานทำการใส่โปรแกรมให้กับเครื่อง รวมทั้งจุดอะไลเม้นท์, บอนดิ้งพารามิเตอร์, ไพลอทเฟรมตามพิกัดต่างๆ บนได และตำแหน่งจุดบอนด์ต่างๆ ทั้งที่แพดและหลีด หลังจากการป้อนโปรแกรมเพียงครั้ง-เดียว เครื่องก็จะทำการบอนด์งานตัวต่อไปด้วยระบบอัตโนมัติ และยังสามารถเก็บโปรแกรมของโปรดักนั้นไว้ใช้ในคราวต่อไปในฟลอฟพี ดิส (FLOPPY DISC) ในโปรแกรมโหมด TEACH จุดยูนิค (จุดอะไลเม้นท์)ทั้ง 3 ถูกเก็บไว้เป็นจุดเปรียบเทียบใน ออโตเมติคโหมด (AUTOMATIC MODE) งานทุกตัวที่เคลื่อนเข้ามาในแคลมป์เฟรม จะถูกเทแพทเทอร์นที่เหมาะสมที่สุด 2-3 แพทเทอร์น จากนั้นคอมพิวเตอร์เริ่มคำนวนหาพิกัด ทั้งที่แพดและหลีดและเริ่มการบอนด์ตามโปรแกรม

เวลาในการบอนด์ ลวดแต่ละเส้น ขึ้นอยู่กับความยาวของลวด (ระยะห่างของแพดและเหลีด) เวลาที่เข็มขยับขึ้นแพดและเหลีด ความหนา บาง และส่วนสูงของไค และการเลือกรูปแบบของลูป

การวัดตำแหน่งไค

การวัดตำแหน่งของไคในการบอนด์ แบบอัตโนมัติของเครื่อง ต้องการ 3 จุดอะไลเม้นท์ ซึ่งจุดทั้ง 3 นี้ จะต้องประกอบด้วย

1. แต่ละจุดต้องเป็นส่วนหนึ่งของแพทเทอร์น ขาว และ ดำ บนหน้าไคด้วยขนาด 16 X 16 พิกเซล (16X16 = 256, เกิน 220 ขึ้นไปก็ใช้ได้)
2. แพทเทอร์น ขาว และ ดำ ในไพลอทเฟรมจะต้องมี
   - 2.1 การเปลี่ยนแปลงน้อยที่สุด
   - 2.2 สีดำ และ ขาว บนหน้าไคพอๆ กัน
   - 2.3 สีขาว อยู่ระหว่าง 35 - 65 %
   - 2.4 จุดที่ไม่เหมือนใครบนหน้าไค
3. แต่ละจุดต้องอยู่ห่างกันมากที่สุดเท่าที่จะทำได้
4. จุดอะไลเม้นท์เหล่านี้ จะต้องถูกจำไว้ด้วยหน่วยจุดอะไลเม้นท์ของเครื่อง ตามด้วยการวัดตำแหน่งของจุดเป็นตัวกำหนดตำแหน่งของไคทั้งตัว ดังนั้นคอมพิวเตอร์สามารถจำลองบอนดิ่งโปรแกรมในวิถีทางที่ตรงกับตำแหน่งไคจริง

การสอนโปรแกรมด้วยตัวเอง

คีย์บอร์ดทางแผงด้านขวามือของพนักงาน ทำให้พนักงานสามารถบอนดิ่งโปรแกรมขณะที่ทำการบอนด์งานตัวแรกของโปรดักใหม่

โปรแกรมจะบรรจุอยู่ด้วย

- ค่าเฉลี่ยระดับสูงต่ำของไค
- ค่าเฉลี่ยระดับความลึกของเหลีด
- การเลือนลักษณะของลูฟ
- ค่าของ ดะเวลไทม, โปรเซสไทม, และบอนด์ฟอชของแต่ละแพด และเหลีด

- พิกัดของ X,Y,Z ของแต่ละจุดบอนด์
- ลักษณะของจัมเปอร์ไวร์

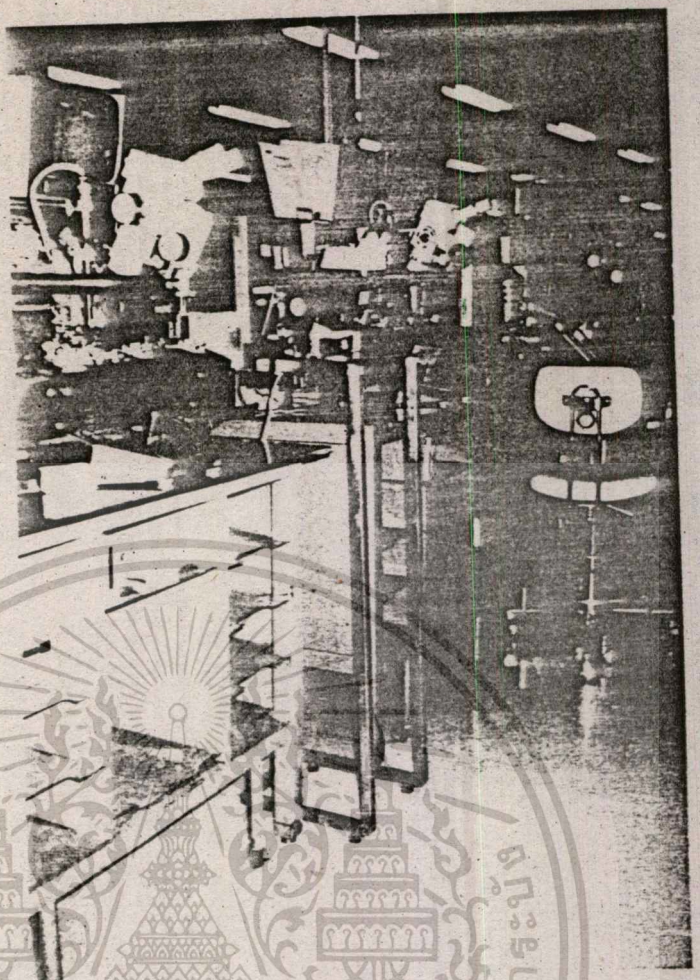
ส่วนประกอบของเครื่อง

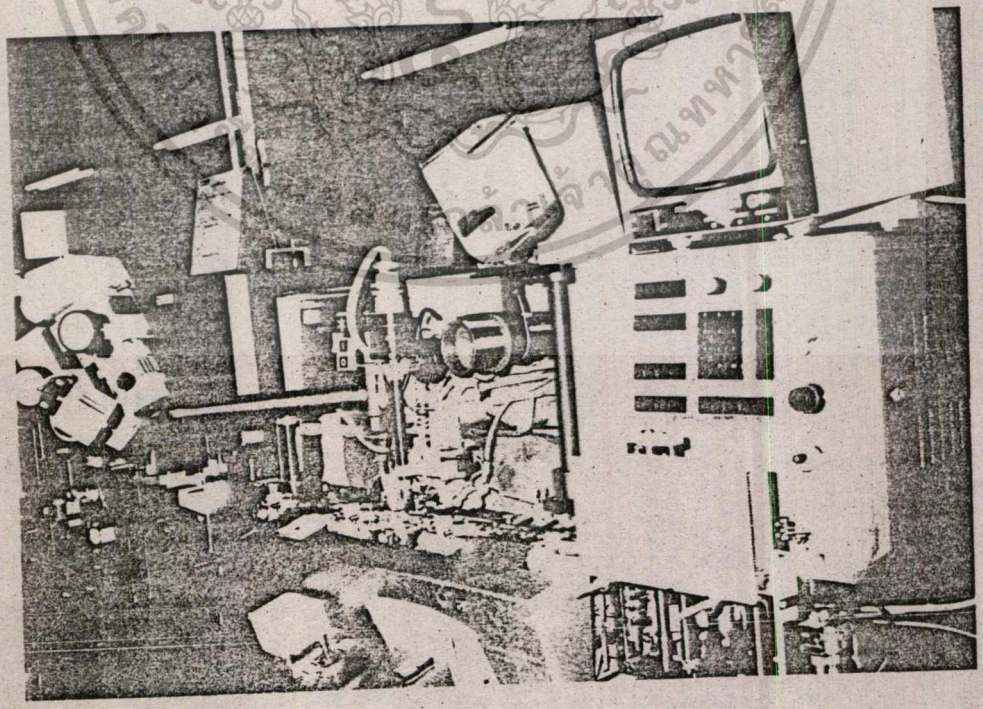
BONDING HEAD	=	หัวเครื่อง
T.V. MONITOR	=	จอ T.V.
RIGHT FRONT PANEL	=	หน้าปัดข้าง ขวา
LEFT FRONT PANEL	=	หน้าปัดข้าง ซ้าย
OPERATOR PANEL	=	ที่ใช้บอนด์
พิกเซล (PIXELS)	=	ภาพขาวดำ ที่ปรากฏบนจอ ทีวี
แพทเทิร์น (PATTERN)	=	ภาพทั้งหมด ที่ปรากฏบนจอ ทีวี
ยูนิคพอยท์	=	จุดที่ไม่เหมือนใครบนหน้าได
ดิสเพล (DISPLAY)	=	เลขที่แสดงบนหน้าปัดด้าน ขวา
ไพลอตเฟรม (PILOT FRAME)	=	ที่อยู่บนจอ ทีวี
จุด SPOT	=	จุดที่อยู่กึ่งกลาง PILOT FRAME

ความหมายต่างๆ ของอักษรที่ดิสเพลแสดง

A	= AUTOMATIC	คือ	การบอนด์แบบอัตโนมัติ
B	= MODIFY	คือ	การแก้ไข
C	= TEACH	คือ	การโปรแกรม
D	= MANUAL	คือ	การบอนด์ทีละเส้น
E	= ERROR	คือ	ระบบการรายงานความผิดพลาดของเครื่อง
F	= SERVICE	คือ	ขั้นตอนต่างๆ ที่จะโปรแกรม

PHICOM WIRE-
BONDER





คุณสมบัติของอุปกรณ์ส่งถ่ายข้อมูลที่ต้องการ

การออกแบบได้ออกแบบให้เครื่องมีคุณสมบัติดังนี้

1. สามารถรับและส่งข้อมูลระหว่างเครื่องไมโครคอมพิวเตอร์ 1 เครื่องกับเครื่อง Bond หลาย ๆ เครื่อง

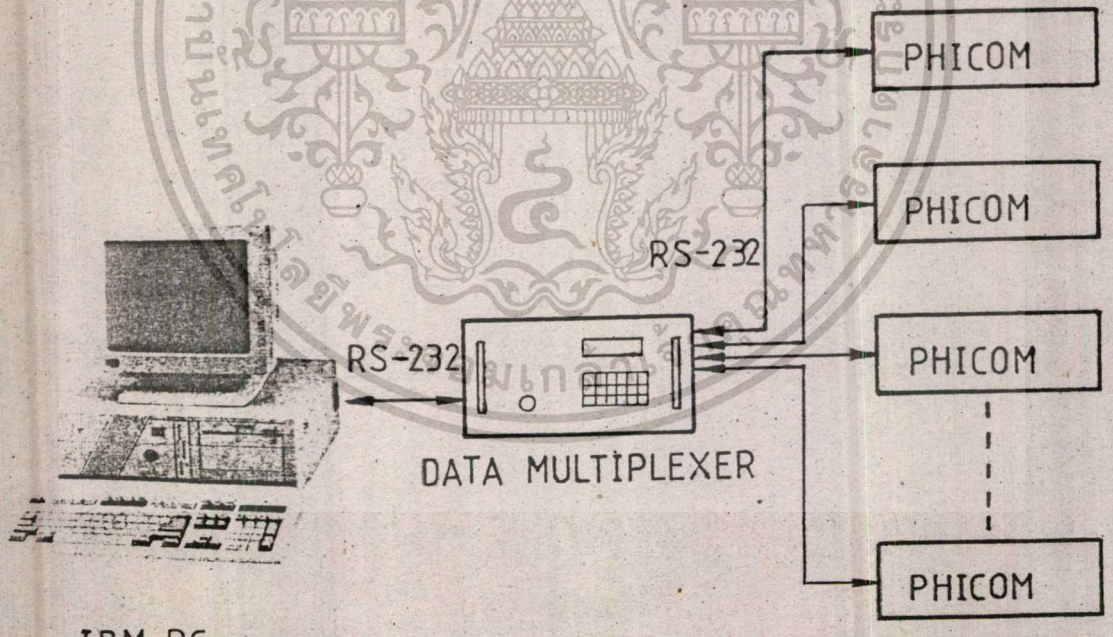
2. จัดคิวการรับส่งเมื่อมีการเรียกใช้พร้อมกันหลาย ๆเครื่อง

3. มี Buffer เพื่อเก็บข้อมูลบางส่วนไว้เมื่อมีการเรียกใช้ข้อมูลตรงกันที่มีอยู่ก็จะส่งข้อมูลออกไปได้เลยซึ่งทำให้ใช้เวลาน้อยกว่าการอ่านจาก Disk

4. สามารถ Down load ข้อมมูลบางส่วน จากไมโครคอมพิวเตอร์มาเก็บไว้ ใน Buffer ได้โดยคำสั่งจาก Key board

5. ลบข้อมูลใน Buffer ได้

6. ตรวจสอบชื่อและข้อมูลใน Buffer ได้



แสดงการเชื่อมโยงสายสัญญาณ

ระหว่าง IBM PC - เครื่องเชื่อมต่อ - PHICOM

การออกแบบวงจร

การออกแบบวงจรให้ระบบสามารถติดต่อระหว่างไมโครคอมพิวเตอร์ 1 เครื่องกับเครื่อง Bond (PHICOM) 24 เครื่องโดยติดต่อผ่าน บัสข้อมูลแบบอนุกรม RS-232 ส่วนควบคุมใช้ไมโครโปรเซสเซอร์ เบอร์ Z-80

วงจรแบ่งออกเป็นส่วนใหญ่ ๆ ได้ 4 ส่วน คือ

1. ส่วนควบคุม (CPU)
2. ส่วนรับส่งข้อมูล แบบ อนุกรม (RS-232)
3. ส่วนแสดงผล รับคำสั่งและตรวจสอบการขอใช้บริการ
4. ส่วนควบคุมการทำงานของ CPU (Wacth dog)

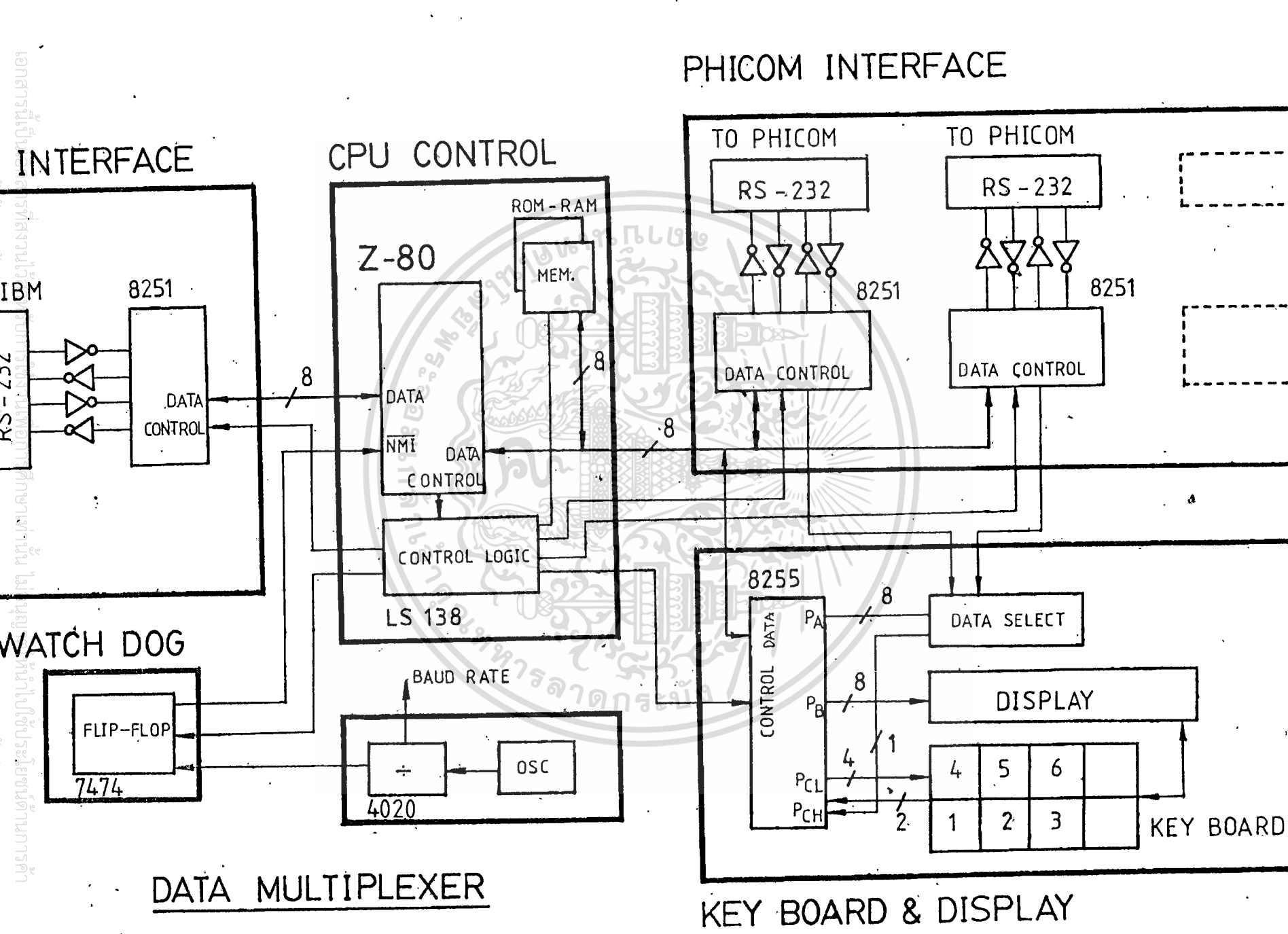
วงจรถอดรหัส (Decoder) สำหรับอุปกรณ์ อินพุท เอาท์พุท

วงจรควบคุมความเร็วในการรับส่งข้อมูล (Boad rate)

ขั้นตอนออกแบบวงจร

การออกแบบวงจร ได้ทำการออกแบบโดยการพิจารณาออกแบบที่ละส่วนตามที่ได้แบ่งเอาไว้ โดยมีขั้นตอนดังนี้

1. กำหนดคุณสมบัติของวงจรที่ต้องการ
2. รวบรวมข้อมูลเกี่ยวกับอุปกรณ์ที่ใช้ในแต่ละส่วน
3. ศึกษาการทำงาน การเชื่อมต่อและวิธีการใช้งานอย่างละเอียด
4. เลือกใช้อุปกรณ์ให้เหมาะสม
5. ออกแบบวงจร ให้มีคุณสมบัติตามที่กำหนดไว้


PHICOM INTERFACE
INTERFACE
CPU CONTROL
TO PHICOM
TO PHICOM
RS-232
RS-232
ROM-RAM
Z-80
MEM.
IBM
8251
8251
8251
DATA
CONTROL
DATA
NMI
DATA
CONTROL
DATA CONTROL
DATA CONTROL
8
8
8
CONTROL LOGIC
LS 138
8255
WATCH DOG
BAUD RATE
DATA SELECT
DISPLAY
FLIP-FLOP
7474
÷
OSC
4020
CONTROL DATA
PA
PB
PCL
PCH
4
1
2
4 5 6
1 2 3
KEY BOARD
DATA MULTIPLEXER
KEY BOARD & DISPLAY

ส่วนประกอบทาง HARD WARE และการทำงานของวงจร

ส่วนควบคุม CPU

ในการออกแบบส่วนควบคุมนั้น ได้กำหนดให้มีคุณสมบัติดังนี้

1. ใช้ CPU ที่หาง่าย และสามารถหาข้อมูลรายละเอียดที่ใช้งานตลอดจนเครื่องมือที่ช่วยในการพัฒนา Soft Ware
2. มีหน่วยความจำหลัก 2-4 K
3. มีหน่วยความมจำข้อมูล 2-4 K
4. สามารถเชื่อมต่ออุปกรณ์ภายนอกได้ง่าย

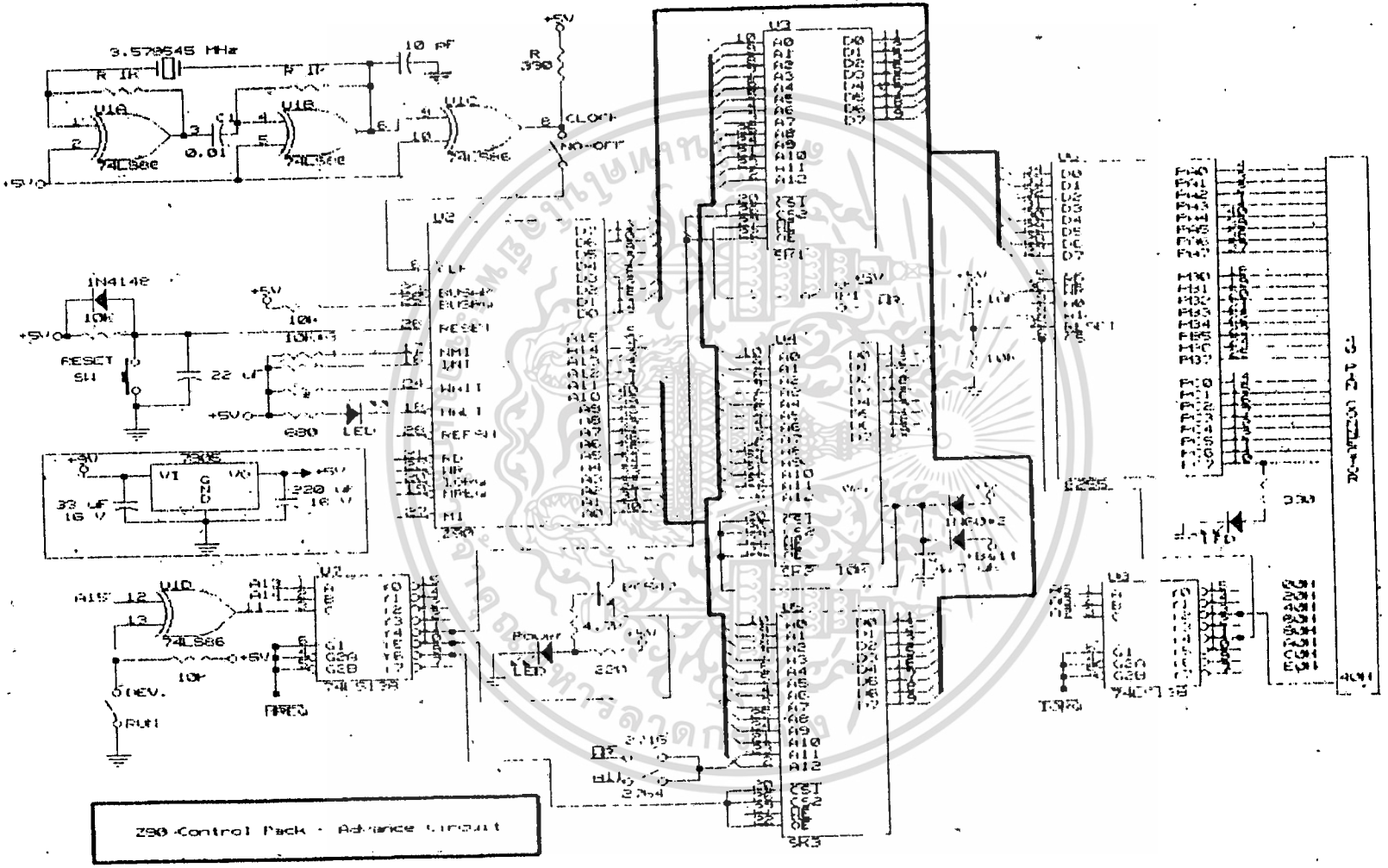
ส่วนควบคุมประกอบด้วยไมโครโปรเซสเซอร์ CPU เบอร์ Z-80 ซึ่งเป็นไมโครโปรเซสเซอร์ ที่นิยมใช้อย่างแพร่หลาย สามารถหาได้ง่าย ราคาถูก และ ยังสามารถพัฒนาซอฟต์แวร์ได้ง่ายเนื่องจากปัจจุบันมีเครื่องมือที่จะพัฒนาซอฟต์แวร์ของ CPU เบอร์นี้มากมายในส่วนของระบบควบคุมนั้น โครงสร้างทางฮาดต์แวร์ เหมือนกับระบบคอมพิวเตอร์ทั่ว ๆ ไป คือประกอบด้วย CPU หน่วยความจำ ROM หน่วยความจำ RAM ส่วนที่กำหนดสัญญาณ นาฬิกา และ Control logic

จากการพิจารณาคุณสมบัติของวงจร Z-80 CR-A Control PACK ของบริษัท ETT มีคุณสมบัติที่เราต้องการครบถ้วนและมี 8255 เป็น Port ขนานรวมอยู่ด้วยกัน และยังสามารถพัฒนา Soft ware โดยผ่าน ET-BOARD

ดังนั้นโครงงานนี้จึงใช้ Z-80 Control PACK เป็นส่วนควบคุมโดยการดัดแปลงแก้ไขวงจรบางส่วนเล็กน้อยเพื่อให้เหมาะสมในการจัดวาง Address ของอุปกรณ์ Input Output

ส่วนประกอบทาง Hard ware ของส่วนควบคุมประกอบด้วย ส่วนสำคัญดังต่อไปนี้

1. ไมโครโปรเซสเซอร์เบอร์ Z-80 ทำหน้าที่ควบคุมการทำงานของระบบ
2. วงจรสร้างความถี่นาฬิกา โดยใช้ X-TAL 3.57954 MHz และวงจร Logic EX-RO 74LS86 ทำหน้าที่สร้างสัญญาณนาฬิกาเพื่อควบคุมการทำงานของ CPU และ ระบบ
3. หน่วยความจำหลัก E-PROM เบอร์ 2732 เริ่มต้นที่ตำแหน่ง 0000-0FFFH
4. หน่วยความจำข้อมูล RAM 8K เริ่มต้นที่ตำแหน่ง 2000H ใช้ RAM เบอร์ 6164
5. ส่วนถอดระหัส DECODER เพื่อใช้ในการ Address หน่วยความจำใช้ IC TTL เบอร์ 74LS138
6. พอร์ทสื่อสารแบบขนาน 8255 ใช้ต่อกับ Key board และ Display



Z80 Control Pack - Advance Circuit

ส่วนรับส่งข้อมูล

การออกแบบวงจรในส่วนรับส่งข้อมูลนั้น การรับส่งข้อมูลของระบบเป็นแบบอนุกรม โดยใช้บัสแบบ RS-232C ได้ใช้ชิพซัพพอร์ทเบอร์ 8251 ของบริษัท อินเทล ซึ่งทำหน้าที่เป็นพอร์ทอนุกรมที่นิยมใช้กันมากเนื่องจาก ใช้ง่ายและหาซื้อได้ในตลาด และสามารถหาข้อมูลในการใช้งานของ IC เบอร์นี้ได้ง่าย 8251 จึงเป็นหัวใจสำคัญในส่วนของการรับส่งข้อมูลนี้

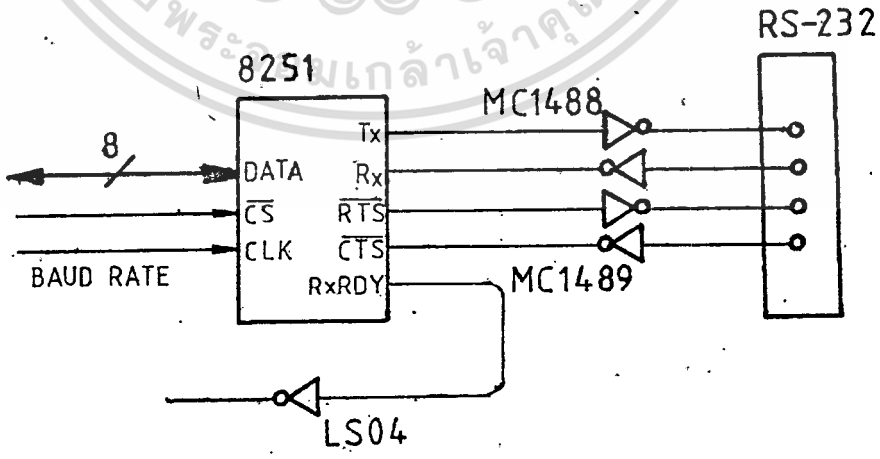
ส่วนควบคุมประกอบด้วยวงจรส่วนต่าง ๆ ดังนี้

1. IC เบอร์ 8251 ทำหน้าที่รับส่งข้อมูล โดยการแปลงข้อมูลแบบขนานไปเป็นแบบอนุกรมแล้วส่งออกไป และรับข้อมูลอนุกรมแล้วแปลงเป็นแบบขนานเพื่อใช้ในระบบ

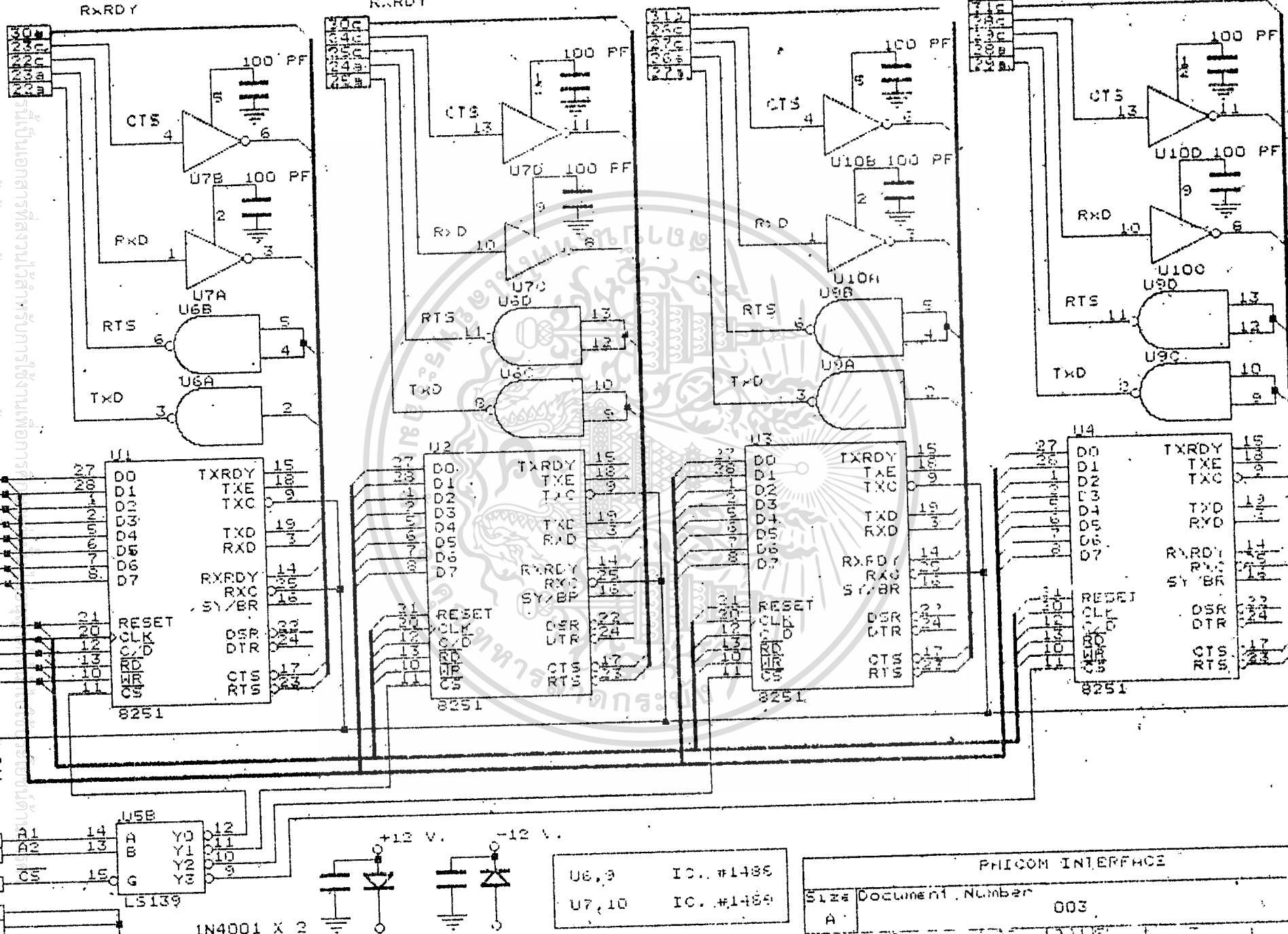
2. IC เบอร์ MC1488 เป็น IC Driver ของบริษัทโมโตโลร่า โดยรับข้อมูลแบบอนุกรมจาก 8251 แล้วแปลงสัญญาณระดับ Digital ให้มีระดับ Voltage สูงขึ้น เพื่อส่งไปในบัสข้อมูลทำให้สามารถส่งไปได้ไกล ๆ

3. IC เบอร์ MC1489 เป็น IC Buffer ของบริษัทโมโตโลร่า เช่นเดียวกัน ทำหน้าที่รับสัญญาณจากบัสข้อมูล RS-232C ที่มีระดับ Voltage สูงและปรับให้เป็นระดับ Digital แล้วส่งไปให้กับ 8251

4. สัญญาณ TXRDY จะถูกส่งผ่าน IC 74LS04 ไปยังส่วนควบคุมเพื่อเป็นการบอกการขอใช้บริการ การรับส่งข้อมูล ของพอร์ทข้อมูลนี้



แสดงส่วนประกอบของส่วนรับส่งข้อมูล


RxRDY
100 PF
CTS
U7B
RxD
U7A
RTS
U6B
TxD
U6A
U1
DO
D1
D2
D3
D4
D5
D6
D7
RESET
CLK
C/D
RD
WR
CS
TXRDY
TXE
TXC
TXD
RXD
RXRDY
RXC
SY/BR
DSR
DTR
CTS
RTS
8251
U7D
U7C
U5D
U6C
U2
U10B
U10A
U9B
U9A
U3
U10D
U10C
U9D
U9C
U4
U5B
A1
A2
CS
A
B
G
Y0
Y1
Y2
Y3
LS139
+12 V.
-12 V.
1N4001 X 2
U6,9 IC. #1486
U7,10 IC. #1489
PHICOM INTERFACE
Size
A
Document Number
003

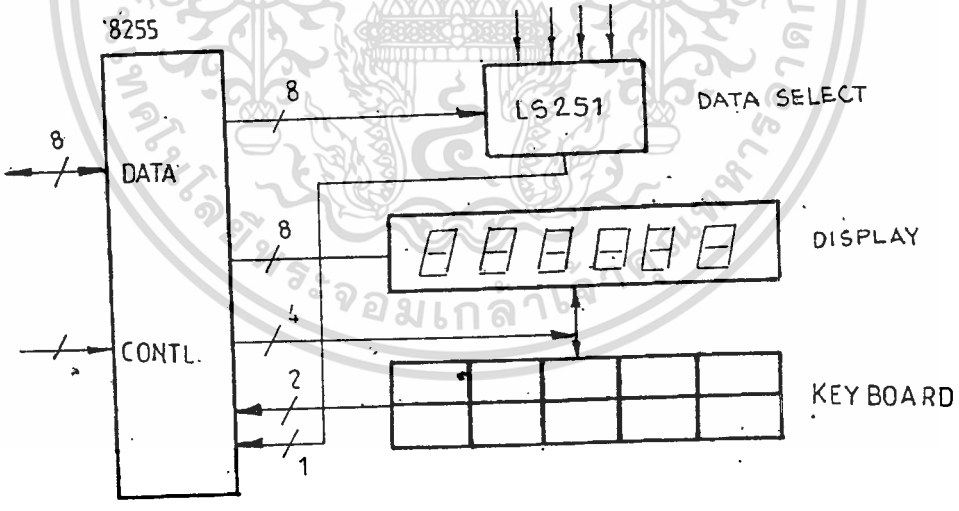
ส่วนแสดงผล รับคำสั่งและตรวจสอบการขอใช้บริการ

ส่วนนี้ควบคุมผ่านพอร์ทขนานโดยใช้ 8255 เป็นตัวควบคุม การแสดงผล รับ Key board และตรวจสอบการขอใช้บริการโดยกำหนดการทำงานดังนี้

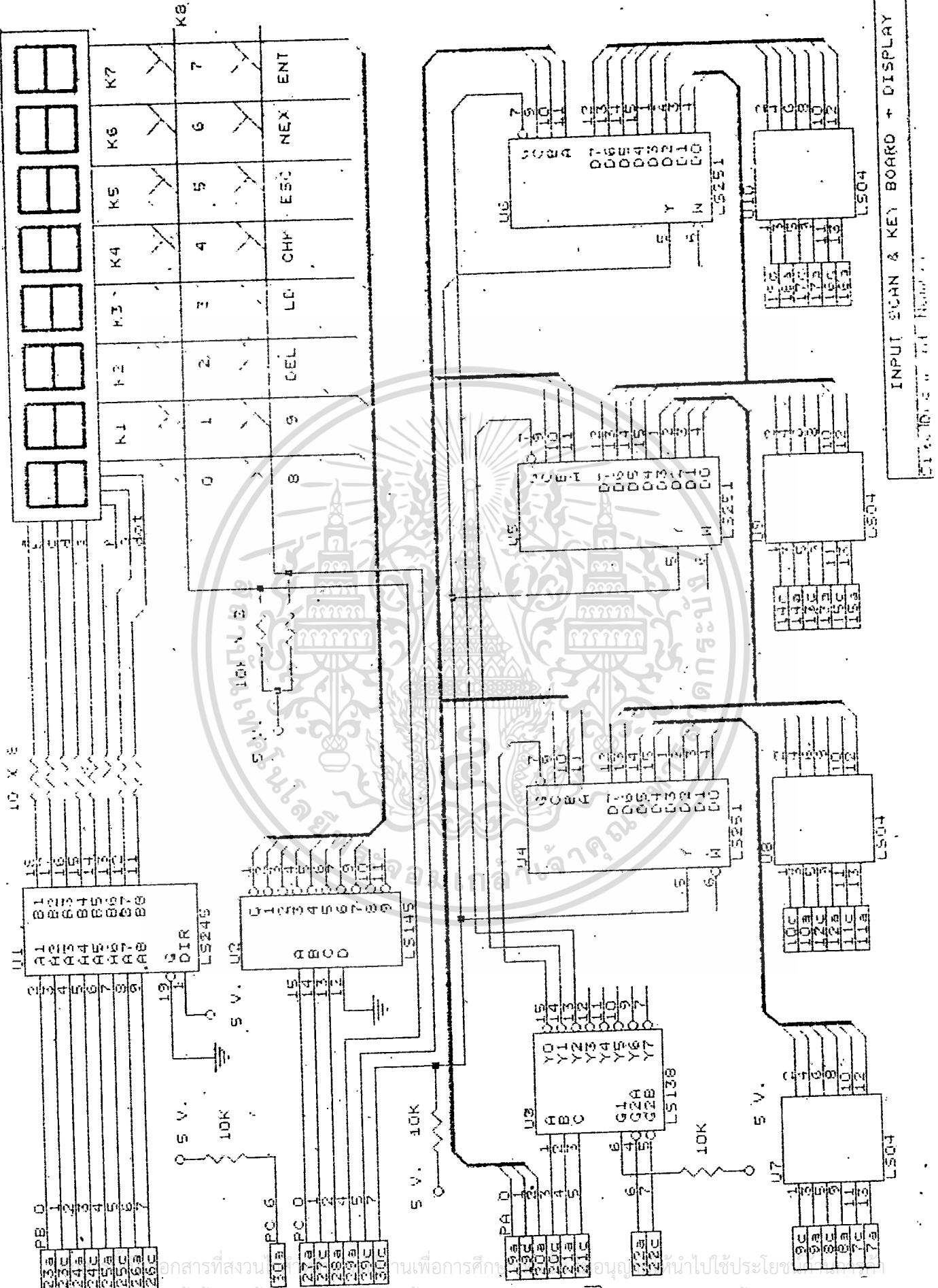
1. พอร์ท A ใช้เป็นพอร์ทเอ้าท์พุท ทำการส่ง Scan code โดยผ่านวงจร Decoder 74LS138 แล้วส่งสัญญาณควบคุมไปยังวงจร Data selector 74LS251 (8 Line to 1 Line) ส่งอินพุทที่ได้ไปพิจารณา ที่ Bit 6 ของ พอร์ท C

2. พอร์ท B ใช้เป็น Data Output เพื่อแสดงผลทาง Display ชนิด 7 Segment

3. พอร์ท C แบ่งออกเป็น 2 ส่วน ส่วน 4 Bit แรกใช้เป็นพอร์ทเอาพุท ใช้ในการ Scan Display และ Key board ส่วน 4 Bit หลัง เป็นพอร์ทอินพุท ใช้ในการอ่าน Key board และตรวจสอบการใช้บริการรับส่งข้อมูลจากภายนอก



แสดงส่วนประกอบและการเชื่อมต่อของวงจร



การทำงานของเครื่องอุปกรณ์ส่งผ่านข้อมูล

เมื่อเปิดเครื่องจะอยู่ในตำแหน่ง READY คือ พร้อมที่จะอ่านโดย CHECK ว่ามีเครื่อง PHICOM มีเรียกใช้บริการหรือไม่ ถ้าเรียกก็จะไปทำที่เครื่อง PHICOM ก่อน ถ้าไม่เรียกก็จะมา CHECK ว่ามี KEY อะไรใช้อยู่หรือไม่ ถ้าไม่มีก็จะไป CHECK อีกว่ามีเครื่อง PHICOM ใช้หรือไม่ ถ้ามีก็จะดูว่ามี KEY FUNCTION ไหนใช้อยู่ ถ้าไม่มี FUNCTION KEY ใช้ STEP ก็จะอยู่ที่เดิมและจะไป CHECK ดูว่ามีเครื่อง PHICOM เรียกใช้บริการหรือไม่ ถ้า KEY FUNCTION LOAD เรียกใช้ DISPLAY ก็จะแสดง "LD" และ STEP จะเลื่อนไป 1 STEP และจะดูว่ามีเครื่อง PHICOM เบอร์ใดเรียกใช้ ถ้า KEY FUNCTION DEC ก็จะ CHECK จาก BUFFER ว่ามี DATA อยู่หรือไม่ ถ้าพบว่าไม่มี DISPLAY ก็จะแสดง ERROR STEP จะอยู่ที่เดิมและจะไปดูที่เครื่อง PHICOM อีกครั้งหนึ่ง ถ้าพบ DISPLAY จะแสดง DEC และ STEP จะเลื่อนไป 1 STEP เพื่อให้ KEY หมายเลขของ PROGRAM ที่ลบ และจะไปดูที่เครื่อง PHICOM อีกครั้งหนึ่ง

FUNCTION CHECK จะดูว่าใน BUFFER มี PROGRAM ที่ต้องการหรือไม่ ถ้ามี DISPLAY ก็จะแสดงหมายเลขของ PROGRAM ออกมาตามลำดับสิ่งใน BUFFER สามารถเก็บไว้ได้ประมาณ 5 หมายเลข PROGRAM และจะไปดูที่เครื่อง PHICOM อีก ถ้าไม่มี DISPLAY ก็จะแสดง ERROR โดยที่ STEP จะอยู่ที่เดิม

FUNCTION ESE คือ FUNCTION ที่จะออกจาก MODE KEY BOARD และจะกลับไปที่เดิมคือด READY

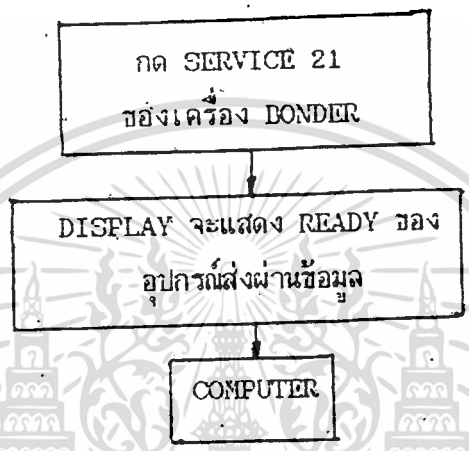
FUNCTION NEXY คือ KEY FUNCTION ที่จะเลื่อนหมายแลข PROGRAM ไปข้างหน้าซึ่งถ้า STEP จะอยู่ที่ 0 ก็จะไปดูว่าเครื่อง PHICOM หมายเลขอะไรต้องการ ถ้า STEP ไม่อยู่ที่เดิมก็จะ CHECK ว่ามี FUNCTIO ใดเรียก ถ้า FUNCTION LOAD เรียก DISPLAY จะก็เลื่อนหมายแลขของ PROGRAM ไปอีก 1 ครั้ง และจะไป CHECK ดูว่าเครื่อง PHICOM หมายแลขอะไรที่ต้องการ

ถ้า FUNCTION DEC เรียกก็จะ CHECK ว่าใน BUFFER มีหรือไม่ ถ้ามี DISPLAY ก็เลื่อนหมายเลขของ PROGRAM ไปข้างหน้า 1 ครั้ง และไป CHECKF ดูว่ามีเครื่อง PHICOM หมายเลขอะไรที่ต้องการ

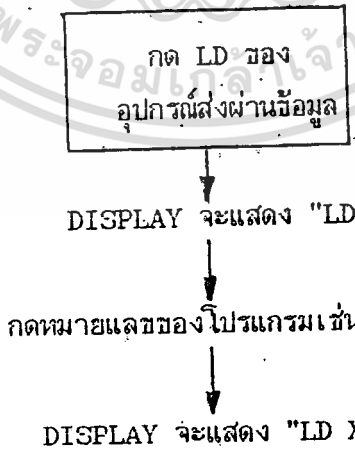
อุปกรณ์ส่งถ่ายข้อมูล

KEY BOARD FUNCTION

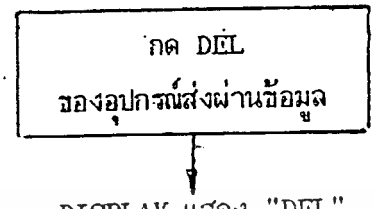
READY คือ เครื่อง BONDER ติดต่อเรียกโปรแกรมจาก P.C. ซึ่งเครื่องส่งถ่ายข้อมูลจะแสดง ทิวอักษร READY เช่น



LOAD คือ การเรียกโปรแกรมที่มีอยู่ใน COMPUTER นำออกมาใช้ เมื่อนำไปใช้แล้ว PROGRAM จะถูกเก็บไว้ใน BUFFER ของอุปกรณ์ส่งผ่านข้อมูลได้ไม่เกิน 5 PROGRAM โดยต้องกด LOAD และหมายเลขของโปรแกรมที่ต้องการนำมาใช้เช่น



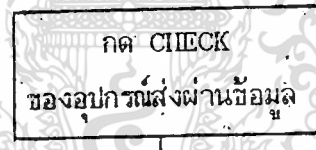
DEL คือ การลบโปรแกรมที่อยู่ใน BUFFER ซึ่งเครื่องจะทำการ CHECK ก่อนว่ามี PROGRAM ที่ต้องการที่จะลบอยู่ใน BUFFER หรือไม่ ถ้ามีก็จะทำการลบให้เลย แต่ถ้าไม่มีเครื่องจะไม่ทำงานในคำสั่งนั้น



DISPLAY แสดง "DEL"

เครื่องจะตรวจ CHECK ว่ามี PROGRAM ที่ต้องการลบอยู่ใน BUFFER หรือไม่ ถ้ามีจะลบให้ทันที่แต่ ถ้าไม่มีเครื่องจะไม่ทำงานในคำสั่งนั้น

CHECK คือ สามารถตรวจสอบดูว่ามี PROGRAM ใดบ้างที่อยู่ใน BUFFER ของอุปกรณ์ผ่านข้อมูล ซึ่งจะใช้ร่วมกับ NEXT



DISPLAY แสดง "CH"

DISPLAY จะแสดงหมายเลขของ PROGRAM

กด NEXT

หมายเลขของ PROGRAM ที่มีอยู่จะแสดงไปเรื่อยๆจนครบ 5 PROGRAM แล้วจะวนกลับไปที่ PROGRAM แรก

ESC คือ คำสั่งที่จะยกเลิก FUNCTION KEY ทั้งหมด เพื่อให้ไปอยู่ใน READY

กด ESC

ของอุปกรณ์ส่งผ่านข้อมูล

DISPLAY จะแสดงไปที่ READY

NEXT คือ คำสั่งที่ให้เลื่อนหมายเลขชื่อ PROGRAM ที่มีอยู่ใน BUFFER ไปเรื่อยๆ จนครบ 5 PROGRAM

ENT คือ คำสั่งที่สามารถให้ FUNCTION นั้น ทำงานได้ทันที.

0-9 คือ ตัวเลขที่สามารถ KEY ลงไปได้ตามชื่อ หมายเลขของ PROGRAM.

NO PIICOM ใดเรียก -----> จัด Q -----> อยู่ใน MOD อะไร.

ถ้า READ ดูว่าเป็น PROGRAM อะไร -----> CHECK BUFFER ถ้ามีก็เอาออกไป ถ้าไม่มีก็จะเอาที่ IBM ก่อน แต่ถ้าไม่มีเครื่องก็จะไม่ SERVICE ให้หรือข้ามไปทำใน Q ตัวต่อไป.

WRITE จะ CHECK ใน BUFFER ก็เอา PROGRAM ไปเก็บใน PC.

ดู Q ต่อไปถ้ามีก็จะทำในลักษณะเดียว ถ้าไม่มีก็จะไปอยู่ใน MODE READY.

****

ESC คือ คำสั่งที่จะยกเลิก FUNCTION KEY ทั้งหมด เพื่อให้ไปอยู่ใน READY

กด ESC

ของอุปกรณ์ส่งผ่านข้อมูล

DISPLAY จะแสดงไปที่ READY

NEXT คือ คำสั่งที่ให้เลื่อนหมายเลขชื่อ PROGRAM ที่มีอยู่ใน BUFFER ไปเรื่อยๆ จนครบ 5 PROGRAM.

ENT คือ คำสั่งที่สามารถให้ FUNCTION นั้น ทำงานได้ทันที.

0-9 คือ ตัวเลขที่สามารถ KEY ลงไปได้ตามชื่อ หมายเลขของ PROGRAM.

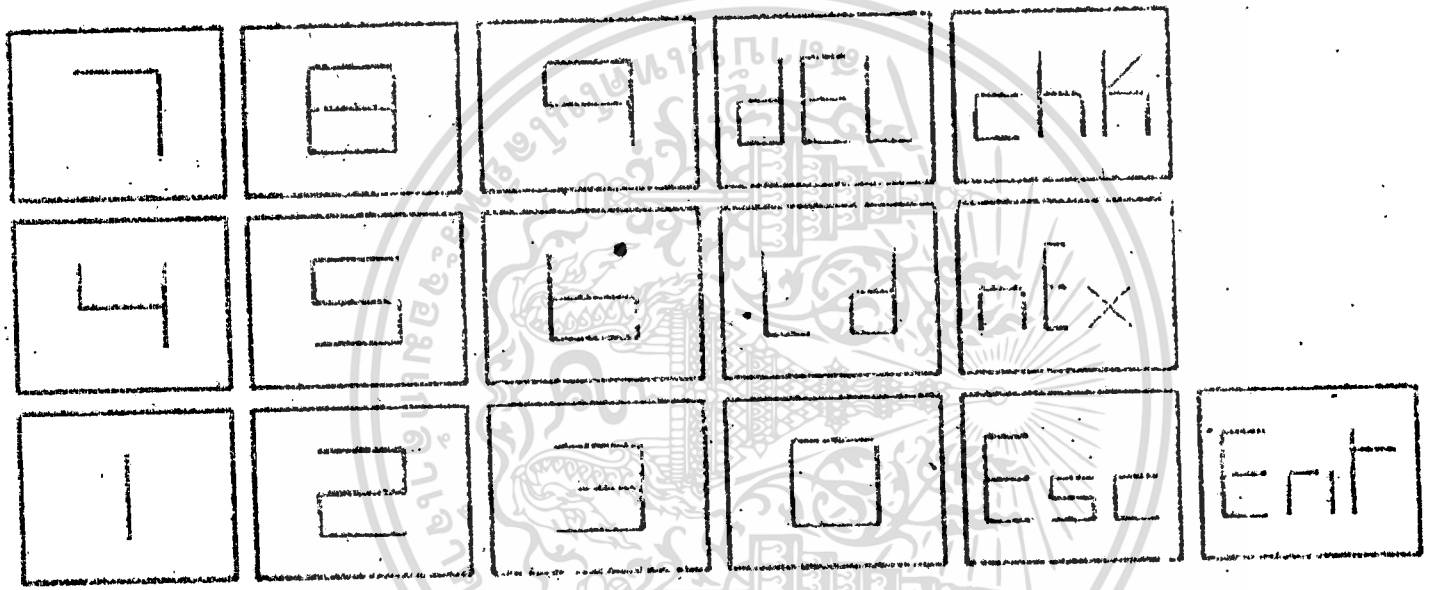
NO PHICOM ใดเรียก -----> จัด Q -----> อยู่ใน MOD อะไร.

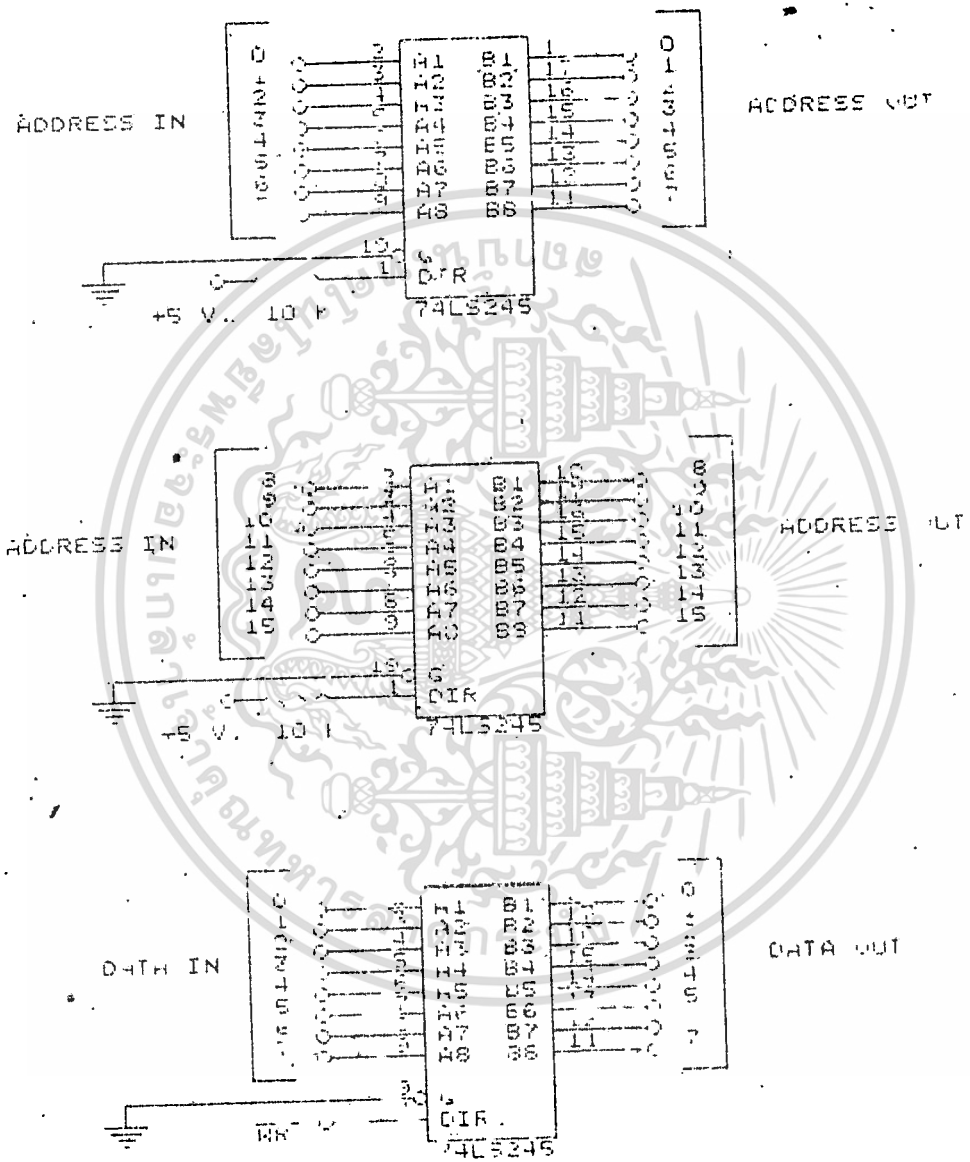
ถ้า READ ดูว่าเป็น PROGRAM อะไร -----> CHECK BUFFER ถ้ามีก็เอาออกไป ถ้าไม่มีก็จะเอาที่ IBM ก่อน แต่ถ้าไม่มีเครื่องก็จะไม่ SERVICE ให้หรือข้ามไปทำใน Q ตัวต่อไป.

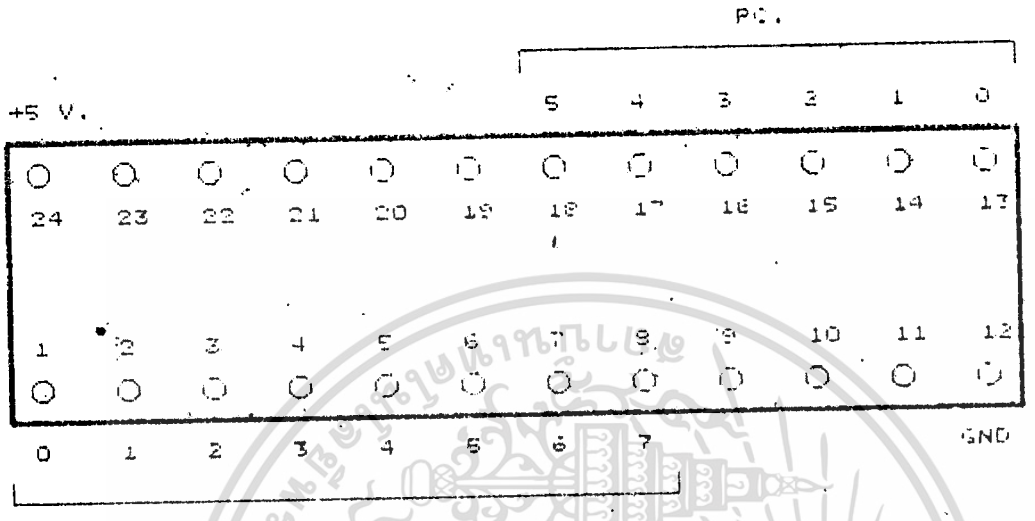
WRITE จะ CHECK ใน BUFFER ก็เอา PROGRAM ไปเก็บใน PC.

ดู Q ต่อไปถ้ามีก็จะทำในลักษณะเดียว ถ้าไม่มีก็จะไปอยู่ใน MODE READY.

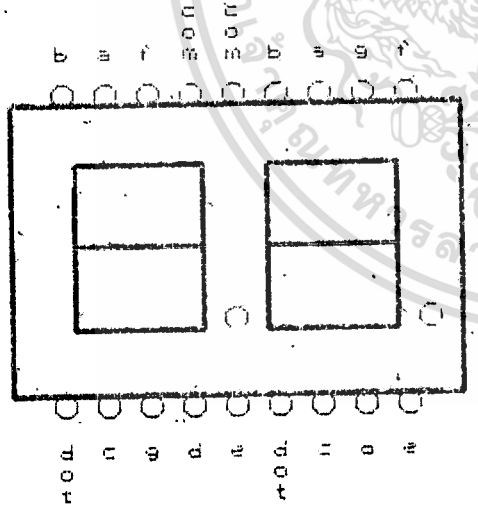
****




ADDRESS IN
ADDRESS OUT
74LS245
DIR
ADDRESS IN
ADDRESS OUT
74LS245
DIR
DATA IN
DATA OUT
74LS245
DIR

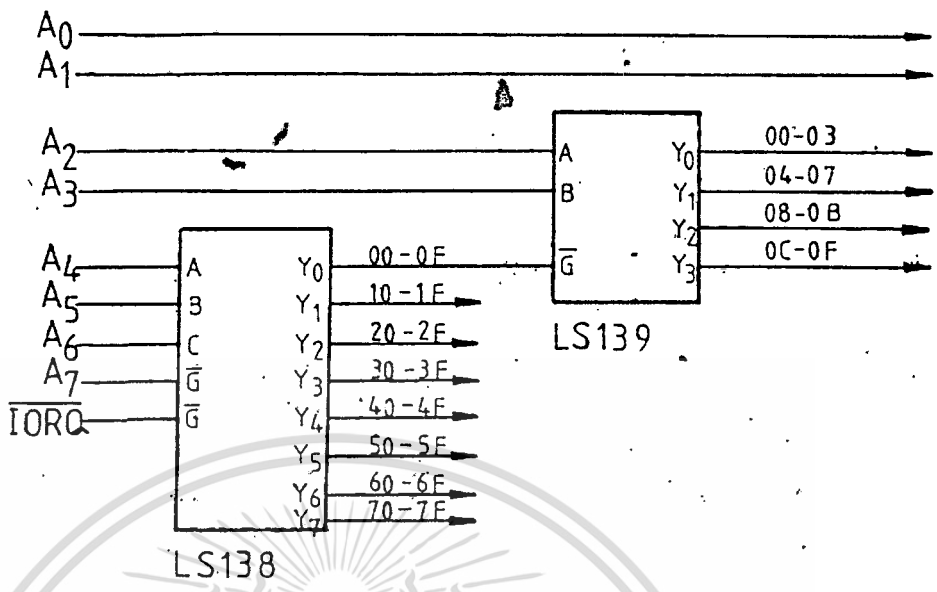


FRONT PANNEL SOCKET



LED. 8 DIGIT DISPLAY

Size	Document Number	
A		

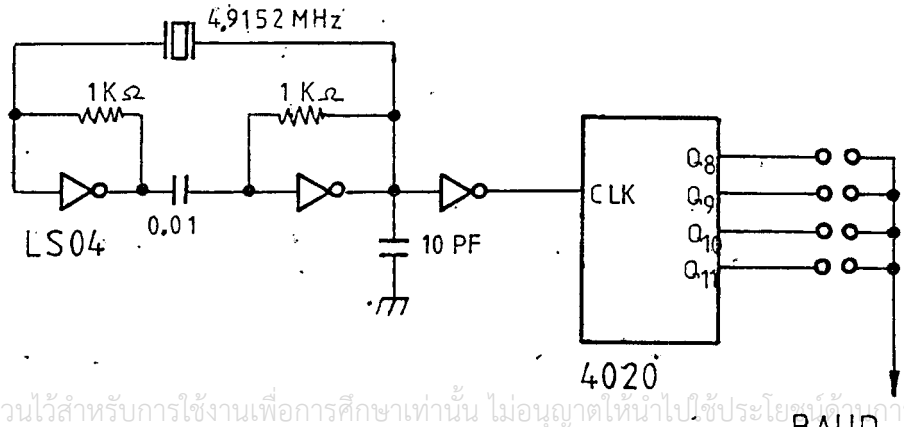


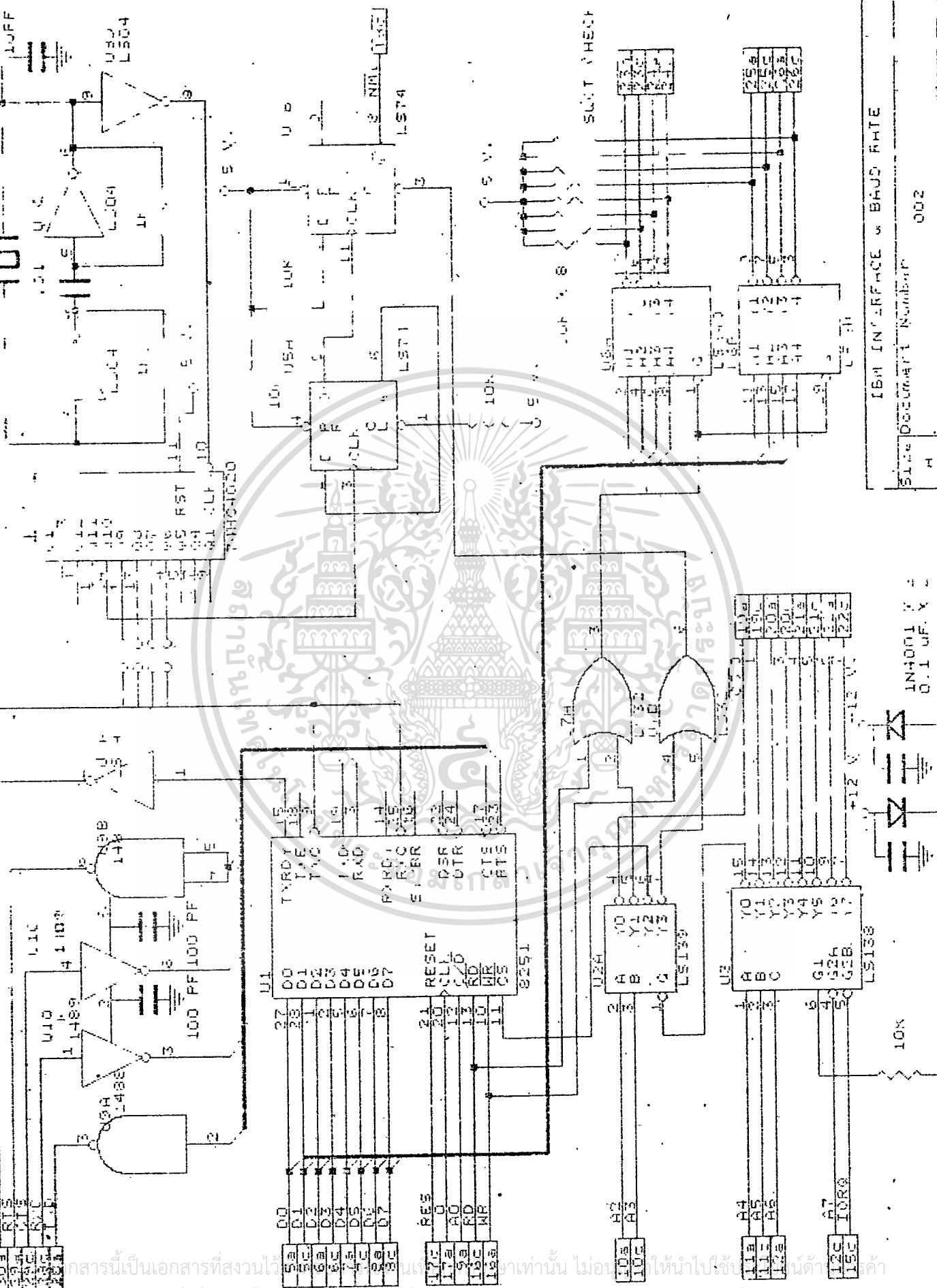
แสดงวงจรถอดรหัสสำหรับอินพุท เอาท์พุท

จากวงจรสามารถถอดรหัสได้ตั้งแต่ 00- 7F

<u>วงจรสร้างความถี่ ควบคุมความเร็วของการรับส่งข้อมูล</u>

วงจรสร้างสัญญาณนาฬิกา เพื่อใช้ ควบคุมความเร็วในการรับส่งข้อมูลแบบอุกรม ออกแบบโดยใช้ X-TAL ความถี่ 4.9152 MHz ทำงานร่วมกับ IC TTL เบอร์ 74LS04 ได้สัญญาณนาฬิกา ส่งให้กับวงจรหารซึ่งใช้ IC CMOS เบอร์ 4020 ได้ความถี่เท่ากับ 9600, 4800, 2400 และ 1200 Hz ตามลำดับ แล้วเลือกส่งไปควบคุมการทำงานของ IC 8251 ต่อไป





การสื่อสารแบบอนุกรม (RS-232C)

การควบคุมการทำงานในส่วนของคอมพิวเตอร์

การควบคุมการทำงานในส่วนของ Personal Computer

Personal Computer เป็นไมโครคอมพิวเตอร์ ที่นิยมใช้กันแพร่หลายในปัจจุบัน ซึ่งระบบของ Personal Computer จะประกอบด้วยอุปกรณ์ต่าง ๆ คือ

1. คีย์บอร์ด
2. จอมอนิเตอร์
3. เครื่องพิมพ์
4. ดิสค์ไดร์ฟ
5. อื่น ๆ

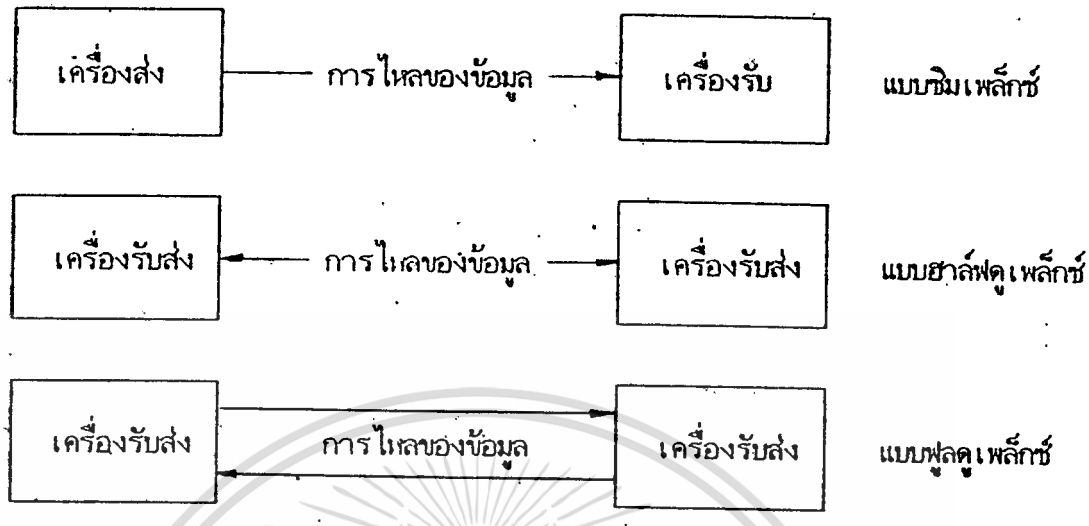
โดยอุปกรณ์ต่าง ๆ เหล่านี้ จะทำการติดต่อสื่อสารกับเมนซีพียู (Main CPU) บางชนิดอาจทำการส่งข้อมูลเข้าสู่เมนซีพียูอย่างเดียว เช่น คีย์บอร์ด หรือรับข้อมูลจากเมนซีพียูอย่างเดียว เช่น เครื่องพิมพ์ และจอมอนิเตอร์ บางชนิดอาจทั้งรับทั้งส่ง เช่น ดิสค์ไดร์ฟ อุปกรณ์เหล่านี้เป็นอุปกรณ์หลักของ Personal Computer การสื่อสารของ Personal Computer ในโครงงานนี้จะใช้ส่งผ่านทาง RS-232C ซึ่งมีรายละเอียดดังนี้

การสื่อสารข้อมูลบนไมโครคอมพิวเตอร์

รูปแบบการติดต่อสื่อสารแบบอนุกรม

เมื่อคอมพิวเตอร์เครื่องหนึ่ง ต้องการติดต่อสื่อสารรับส่งข้อมูลกับอีกเครื่องหนึ่ง สิ่งที่ต้องกระทำคือ การเชื่อมโยงให้เกิดช่องสัญญาณระหว่างเครื่องทั้งสอง ถ้าเป็นการส่งในลักษณะทางเดียวในสายสัญญาณช่องเดียว เรียกการส่งแบบนี้ว่า ซิมเพล็กซ์ (Simplex) แต่ถ้าสัญญาณช่องเดียวแต่ผลัดกันรับ-ส่ง โดยใช้เวลาคนละขณะ เรียกการส่งแบบนี้ว่า ฮาล์ฟดูเพล็กซ์ (Half duplex) แต่ถ้าสามารถส่งสัญญาณไปกลับในเวลาเดียวกันหรือพร้อมกันเรียกแบบนี้ว่า ฟูลดูเพล็กซ์ (full duplex)

อนึ่ง การสื่อสารที่กล่าวถึงนี้ จะเน้นเฉพาะการสื่อสารแบบอนุกรม ซึ่งเป็นแบบที่มีการใช้งานกันอย่างกว้างขวาง ทั้งระหว่าง ไมโครคอมพิวเตอร์ กับ ไมโครคอมพิวเตอร์ หรือ จาก ไมโครคอมพิวเตอร์ กับเมนเฟรม หรือมินิ



รูปแบบของการติดต่อสื่อสารข้อมูลแบบอนุกรม

1. แบบซิมเพล็กซ์ (Simplex) ข้อมูลจะส่งได้ในทางเดียวเท่านั้น บางครั้งก็เรียกว่าการส่งทิศทางเดียว (Unidirectional data bus)

2. แบบฮาล์ฟดูเพล็กซ์ (Half duplex) ข้อมูลสามารถส่งได้ทั้งสองสถานี แต่จะต้องผลัดกันส่งผลัดกันรับ จะส่งและรับพร้อมกันไม่ได้

3. แบบฟูลดูเพล็กซ์ (Full duplex) ทั้งสองสถานีสามารถรับและส่งพร้อมกันได้ในเวลาเดียวกัน

ความเร็วในการโอนถ่ายข้อมูลแบบอนุกรม

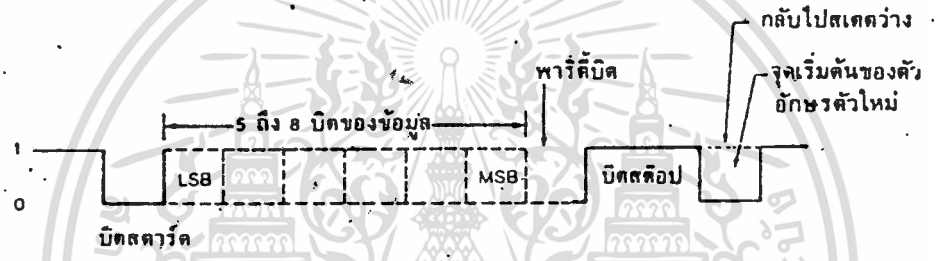
ในการส่งผ่านข้อมูลแบบอนุกรม มีหน่วยวัดความเร็วเป็นจำนวนบิทต่อวินาที (bps) ซึ่งเครื่องที่ใช้ส่งและรับจะต้องมีความเร็วเท่ากันจึงจะรับส่งข้อมูลกันได้ อัตราความเร็วเป็นบิทต่อวินาทีนี้ เป็นผลลัพธ์ที่เกิดขึ้นจากระบบการสื่อสาร แต่ยังมีอีกหน่วยหนึ่งที่วัดการเปลี่ยนแปลงของสัญญาณในหนึ่งวินาที ซึ่งเรียกว่าบอดเรต (baud rate) การเปลี่ยนแปลงสัญญาณ 1 ครั้ง อาจจะหมายถึงการส่งข้อมูลได้หลายบิทก็ได้ ดังนั้น อัตราบอดจึงเป็นตัวเลขที่ไม่เท่ากับอัตราบิทก็ได้ และถ้าเขียนเป็นรูปของสมการจะได้

อัตราบิท = อัตราบอด X (บิทใน 1 บอด)

ความเร็วของการถ่ายข้อมูลแบบอนุกรม หน่วยวัดเป็นบิทต่อวินาที (BPS) หน่วยที่บรรยายถึงการเปลี่ยนแปลงของสัญญาณใน 1 วินาที เรียกว่า บอด (baud rate) หรือ อัตราบอด บางครั้งเข้าใจสับสนระหว่างอัตราบอดและอัตราบิท (bit rate) การเปลี่ยนแปลงสัญญาณ 1 ครั้ง อาจจะแสดงถึงการส่งข้อมูลแบบอนุกรมมากกว่า 1 บิท

การสื่อสารแบบอะซิงโครนัส

การสื่อสารแบบ อะซิงโครนัส นี้พัฒนามาจากการส่งโทรพิมพ์ในสมัยก่อน ลักษณะของสัญญาณ แสดงไว้ในรูป เพื่อเพิ่มกลไกการรับส่งอย่างถูกต้อง สัญญาณอะซิงโครนัส จะประกอบด้วยบิทเริ่มต้นหรือบทสตาร์ท (start bit) และบิทสิ้นสุดหรือบิทสต็อป (stop bit)



แสดงฟอร์แมตการสื่อสารแบบอะซิงโครนัส

ขณะที่สถานะของการส่งเป็นแบบว่าง (Idle) คือ ยังไม่มีสัญญาณส่งออกมา จะมีสัญญาณหรือแรงดัน (หรือกระแส) ตลอดเวลา เพื่อความแน่ใจว่าหน่วยรับยังติดต่ออยู่กับฝ่ายส่ง เมื่อเริ่มจะส่งข้อมูลสัญญาณของ อะซิงโครนัส จะเป็น 0 หนึ่งช่วงสัญญาณนาฬิกา บิทนี้เรียกว่าสตาร์ทบิทก็จะเป็นข้อมูลสำหรับ 1 ตัวอักษร ซึ่งอาจจะมีขนาดตั้งแต่ 5 บิท ถึง 8 บิท โดยบิทที่มีค่าน้อยที่สุด (LSB) จะถูกส่งออกมาก่อนไล่ไปถึงบิทที่มีค่ามากที่สุด (MSB) การเข้ารหัสอักขระนี้ส่วนมากจะนิยมใช้รหัส ASCII แรกเริ่มทีเดียว ในงานของโทรพิมพ์เราใช้รหัส Baudot ซึ่งใช้ 5 บิท ในการแทนอักขระ 1 ตัว ตามหลังข้อมูลก็จะเป็นพาร์ตี้บิท ซึ่งอาจจะใช้หรือไม่ใช้ก็ได้ พาร์ริตี้บิทจะทำหน้าที่เป็นตัวตรวจสอบความถูกต้องของสัญญาณที่ได้รับ พาร์ริตี้บิทอาจจะเป็นแบบคู่ (Even) หรือแบบคี่ (Odd) หมายความว่าถ้าหากเป็นพาริตี้คู่ จำนวนบิทที่เป็น 1 ในช่วงบิทข้อมูลกับบิตพาริตี้ รวมแล้วจะต้องเป็นจำนวนคู่ ผู้ส่งจะต้องทำหน้าที่ตรวจสอบข้อมูลแล้วใส่พาริตี้บิทเอง ฝ่ายรับเมื่อรับแล้วก็ต้องตรวจสอบดูว่า เป็นจริงดังสถานการณ์ที่ตั้งเอาไว้หรือไม่ หากผิดพลาดก็หมายความว่า สัญญาณที่รับนั้นผิดไปจากที่สถานีได้ส่งออกมา ทั้งนี้จะต้องผิดเป็นจำนวนคี่เท่านั้น คือ

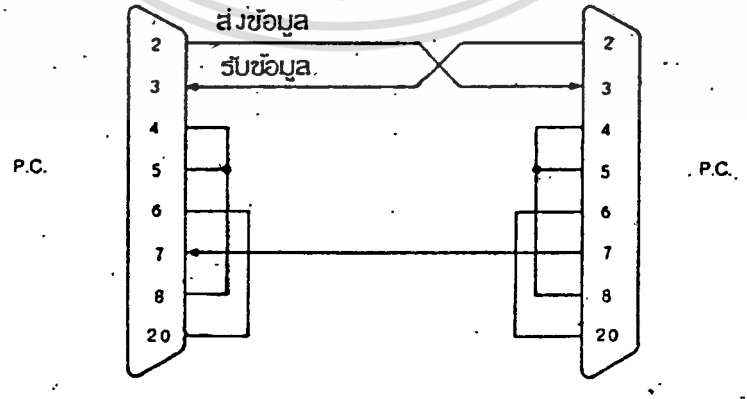
ผิดไป 1 บิท 3 บิท หรือ 5 บิท พร้อมกันนั้นจะต้องตรวจสอบได้ว่าผิดมองกันง่าย ๆ ว่าถ้าผิดเป็นจำนวนคู่ ผลรวมของจำนวนหนึ่งก็ยังเป็นคู่อยู่ดี ทั้งนี้ไม่ได้หมายความว่าพาริตี้คี่ (Odd Parity) จะตรวจสอบการผิดพลาดเป็นจำนวนคี่ ความจริงแล้ว การตรวจสอบความผิดพลาดได้เหมือนกับพาริตี้คู่ (Even Parity) แต่แทนที่จะตรวจสอบดูว่า สัญญาณที่รับเข้ามามีจำนวนคู่ ก็ตรวจสอบดูว่ามีจำนวนคี่หรือเปล่า อย่างไรก็ตามโอกาสที่จะผิดพลาด 2 บิท พร้อมกันมีน้อยมาก

ย้อนกลับมาดูสัญญาณ อะซิงโครนัส ใหม่ หลังจากพาริตี้บิทแล้วก็จะต้องมีสต็อปบิทซึ่งเป็น 1 ความกว้างของสต็อปบิทอาจจะเป็น 1, 1.5 หรือ 2 พัลส์ของนาฬิกา แล้วแต่ผู้รับและผู้ส่งจะตกลงใช้กันเองการเริ่มใช้เออร์ดอนุกรม (RS-232C) จึงจำเป็นจะต้องตั้งค่าต่าง ๆ สำหรับการส่งแบบอนุกรมอันได้แก่

1. ความเร็วในการส่ง
2. ความยาวของรนัส 1 อักขระ
3. บิทตรวจสอบ
4. จำนวนสต็อปบิท

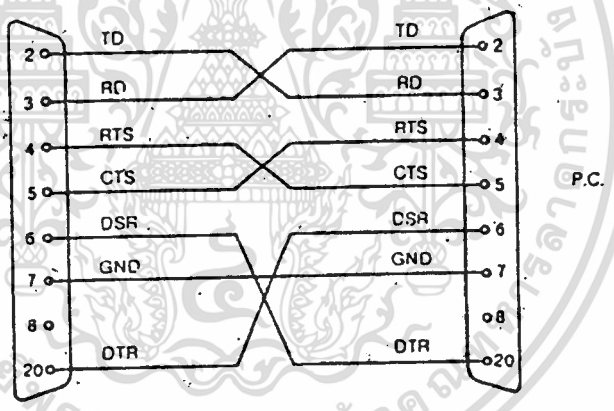
ในการส่งโทรพิมพ์หรือโทรเลขเมื่อก่อนนี้ใช้ความเร็วแค่ 70 บอด และ 110 บอด สำหรับคอมพิวเตอร์ ความเร็วในการส่งมีให้เลือกตั้งแต่ 1200, 2400, 4800, 9600 บอด และสูงไปกว่านั้น

จะเห็นว่ากลไกในการ อะซิงโครนัส ของการสื่อสาร อะซิงโครนัส มีลักษณะไปทีละตัวอักขระ จำนวนพัลส์ของสัญญาณที่ส่งออกบางส่วนใช้ในการควบคุมการส่งอยู่ อันได้แก่ บิทสตาร์ท บิทสต็อป และพาริตี้บิท ทำให้ความเร็วของการส่งอักขระ ต่อวินาทีน้อยลงไป



การเชื่อมต่อ RS-232C ระหว่างไมโครคอมพิวเตอร์แบบ 3 สาย

ปกติ OS ที่ให้บริการเกี่ยวกับ Port RS-232C จะส่งสัญญาณ RTS หรือ Request to Send ออกมาที่ขา 4 ก่อน เมื่อ CTS (เมื่อ Clear to Send ที่ขา 5 เป็นลอจิก "1" (หรือไฟลบ) จึงจะเริ่มทำการส่งข้อมูลที่โอเปอร์เรตอร์บอกให้ส่งออกไปที่ขา 2 ในกรณีที่เป็นการต่อแบบง่าย ๆ ดังในรูป จึงถือว่าเป็นการหลอก คอมพิวเตอร์ โดยเอาขา 4 RTS ต่อเข้ากับขา 5 หรือ CTS เพื่อให้ คอมพิวเตอร์ ส่งข้อมูลได้ทันทีโดยไม่ต้องการความเรียบร้อยของฝายรับ สำหรับขา 6 Data Set Ready ต่อเข้ากับขาที่ 20 Data Terminal Ready ก็ทำนองเดียวกัน โดยปกติ คอมพิวเตอร์ จะถามอุปกรณ์ที่มาต่อพ่วงกับ RS-232C ว่าพร้อมจะส่งหรือไม่ โดยส่งสัญญาณทางที่ขา 20 ก็ได้รับคำตอบกลับที่ขา 6 ทันที ในการต่อแบบนี้ฝายรับจะต้องรอรับอยู่ก่อนแล้ว ก่อนที่ฝายส่งจะเป็นผู้ส่งไม่เช่นนั้นข้อมูลที่ส่งออกมาจะหายแน่นอน เพราะฝายส่งไม่ได้ตรวจสอบความเรียบร้อยของฝายรับก่อน เราอาจจะต่อสายให้มีการตรวจสอบ สัญญาณโต้ตอบ (Hand Shake) ที่ดีกว่านี้



รูปแสดงการต่อไมโครคอมพิวเตอร์ผ่าน RS-232C แบบมี Hand Shake

ในกรณีเช่นนั้นจะมีการโต้ตอบที่ดีขึ้น เมื่อฝายรับยังไม่พร้อมที่จะรับก็จะยังไม่มีสัญญาณ RTS ออกมา (Port อนุกรมยังไม่เปิด หรือ OPEN "COM 1") ในภาษาเบสิคยังไม่ถูกเอกซีคิวท์ ฝายส่งซึ่งถือเอา RTS ของฝายรับเป็น CTS ก็จะยังไม่ส่ง

พอร์ท RS-232C

โดยปกติไมโครคอมพิวเตอร์มักจะติดตั้งพอร์ทแบบอนุกรม RS-232C เพื่อเป็นทางติดต่อสื่อสารข้อมูลกับอุปกรณ์ควบคุมหรือคอมพิวเตอร์อื่น ๆ และเนื่องจากการสื่อสารข้อมูลแบบอนุกรมผ่านทาง RS-232C มีลักษณะการส่งข้อมูลทีละบิทเรียงกันไป ไม่ขึ้นกับสัญญาณคล็อก จึงมีชื่อเรียกอีกอย่างหนึ่งว่า พอร์ทสื่อสารแบบอะซิงโครนัส (Asynchronous Communication Port)

หน้าที่สำคัญของพอร์ทสื่อสารแบบอะซิงโครนัส ก็คือ

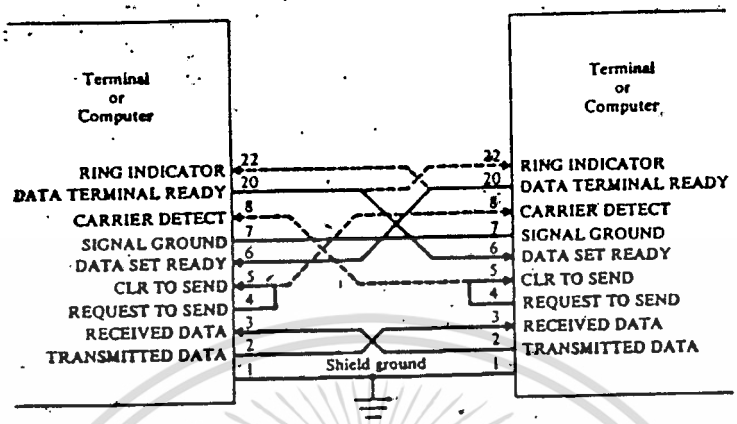
รับสัญญาณ

1. เปลี่ยนสัญญาณที่เข้ามาเป็นชุดแบบอนุกรมให้เป็นแบบขนาน
2. ตรวจสอบความผิดพลาดของสัญญาณที่รับเข้ามา
3. แยกสต๊อปบิท และพาริตี้บิท
4. ส่งสัญญาณให้ ซีพียู รู้ว่ารับข้อมูลไว้แล้ว และส่งข้อมูลให้ ซีพียู

ส่งสัญญาณ

1. เปลี่ยนสัญญาณแบบขนานที่รับจาก ซีพียู และทยอยส่งออกไปเป็นชุดแบบอนุกรม
2. เพิ่มสต๊อปบิท และ พาริตี้บิท ออกจากสัญญาณข้อมูล
3. เพิ่มสัญญาณชุดควบคุมอุปกรณ์อื่น ๆ หรือโมเด็มที่เชื่อมต่อ (ถ้ามี)

สายสัญญาณที่ใช้ในพอร์ทสื่อสารแบบอนุกรม RS-232



แสดงสายสัญญาณของ RS-232C

Shiled Ground (SG ขาที่ 1)

เป็นสาย Ground ของระบบ

Transmitte Data (TD ขาที่ 2)

เป็นสัญญาณที่ส่งออกมาจากตัวไมโครคอมพิวเตอร์ ไปต่อเข้าโดยตรงกับไมโครคอมพิวเตอร์เครื่องอื่น หรืออาจเป็นโมเด็ม, เครื่องพิมพ์ เมื่อไม่มีสัญญาณส่งออกมา สถานภาพลอจิกของขานี้มีค่าเท่ากับ "1" หรือเทียบเท่ากับสถานะปิด

Receive Data (RD ขาที่ 3)

เป็นเส้นทางการส่งสัญญาณเข้าไปยังตัวไมโครคอมพิวเตอร์ ขานี้จะมีสถานะภาพของลอจิกเป็น "1" เมื่อไม่มีสัญญาณเข้ามา

Reques to Send (RTS ขาที่ 4)

สำหรับส่งสัญญาณไปยังคอมพิวเตอร์ เป็นการขอส่งสัญญาณทางขา 2 สัญญาณนี้จะถูกใช้คู่กับ CTS หรือ Clear to Send หากอุปกรณ์ ทางด้านรับได้รับสัญญาณ RTS จะทำการตรวจสอบตัวเองว่าพร้อมจะรับสัญญาณได้หรือยัง หากพร้อมก็ส่งสัญญาณบอกไปที่สาย CTS

Clear to Send (CTS ขาที่ 5)

เมื่อสัญญาณที่ขานี้อยู่ในสภาวะ off (negative voltage หรือลอจิก "1") หมายความว่า อุปกรณ์ทางด้านรับพร้อมที่จะรับข้อมูลแล้ว

Data Set Ready (DSR ขาที่ 6)

เมื่อสัญญาณขาที่นี้อยู่ในสถานะ ON (หรือลอจิก "0") เป็นการบอกไมโครคอมพิวเตอร์ว่าทำการต่อพอร์ทเข้ากับสายโทรศัพท์เรียบร้อย พร้อมที่จะส่งได้แล้ว โทรศัพท์ที่มีการหมุนหมายเลขอัตโนมัติจะส่งสัญญาณนี้ไปบอกให้ไมโครคอมพิวเตอร์รู้ว่าโทรศัพท์ได้รับการต่อสำเร็จเรียบร้อยแล้ว

Signal Ground (SG ขาที่ 7)

ทำหน้าที่เป็นระดับแรงดันอ้างอิงสำหรับทุก ๆ สายของสัญญาณ มีระดับแรงดันเป็นลอจิก "0" เมื่อเทียบกับสัญญาณตัวอื่น

Carrier Detect (CD ขาที่ 8)

ไมโครคอมพิวเตอร์ทางด้านส่งจะส่งสัญญาณที่อยู่ในสถานะ ON (ลอจิก "0"). เมื่อไมโครคอมพิวเตอร์ทางด้านรับได้รับสัญญาณนี้แล้วจะนำไปจุด LED เป็นการบอกว่าได้รับสัญญาณจากอีกฝ่ายหนึ่งเรียบร้อยแล้ว

Data Terminal Ready (DTR ขาที่ 20)

ไมโครคอมพิวเตอร์มีสถานะ ON (ลอจิก "0") เมื่อพร้อมที่จะติดต่อกับไมโครคอมพิวเตอร์อีกเครื่องหนึ่ง ไมโครคอมพิวเตอร์ส่วนมากจะไม่รายงานสถานะภาพของตัวเอง (CD, DSR and CTS) ถ้าหากไม่ทำการเปิดสัญญาณ DTR

Ring Indicator (RI ขาที่ 22)

สัญญาณนี้ใช้ในไมโครคอมพิวเตอร์ที่เป็นระบบอัตโนมัติ (Auto Answer) สัญญาณนี้มีสถานะ ON (ลอจิก "0") เมื่อมีสัญญาณกระดิ่งออกมา และ OFF (ลอจิก "1") เมื่อหมดสัญญาณกระดิ่ง

ในการใช้งานนั้น อาจมีข้อสับสนระหว่างสถานภาพของลอจิกกับสถานภาพของสัญญาณ โดยปกติจะคุ้นเคยกับที่ว่า เมื่อแรงดันเป็นบวกหรือสัญญาณ ON สถานภาพลอจิกจะเป็น "1" สำหรับสัญญาณต่าง ๆ ที่กล่าวมาแล้วจะมีสถานะที่ตรงกันข้าม เนื่องจากเพื่อให้การทำงานของสัญญาณครบวงจรทั้งฝ่ายส่งและฝ่ายรับเมื่อลอจิกเป็น "0" หรือขณะที่ไม่มีอะไรส่ง ควรจะมีสัญญาณทางไฟฟ้าครบวงจรอยู่ตลอดเวลา จะได้ทราบว่าวงจรไม่ขาดช่วงระหว่างทางตรงไหน การทำให้รู้ว่าวงจรทำงานครบอยู่ตลอดเวลาก็โดยการให้ค่าแรงดันที่ฝ่ายส่ง ดังนั้นจึงกำหนดให้สัญญาณไฟบวกมีสถานะลอจิกเป็น "0"

ความจริงอีกประการหนึ่งของ RS-232C ก็คือความเร็วและระยะทางของการเชื่อมต่อ RS-232C สามารถเชื่อมต่อถ่ายโอนข้อมูลได้จาก 0-20,000 บิทต่อวินาที ซึ่งเพียงพอสำหรับไมโครคอมพิวเตอร์ที่มีขนาดอัตรา บอด 110 ถึง 9600 บอด ความยาวของสายเชื่อมต่อโดยสัญญาณตามมาตรฐาน RS-232C จำกัดอยู่แค่ 50 ฟุต ซึ่งเพียงพอสำหรับการสื่อสาร

ลักษณะของสัญญาณ RS-232C

เพื่อเป็นหลักประกันว่าข้อมูลถูกส่งออกไปอย่างถูกต้อง และอุปกรณ์ทุกควบคุมอย่างถูกต้อง จำเป็นจะต้องมีการตกลงในเรื่องของสัญญาณที่ใช้ มาตรฐาน RS-232C กำหนดย่านแรงดันไฟฟ้าเพื่อสนองจุดประสงค์ข้างบน ดังแสดงในตารางและรูปข้างล่าง

แรงดันไฟฟ้า	สถานภาพลอจิก	สถานภาพของสัญญาณ	ฟังก์ชั่นในการควบคุม
บวก	0	สเปซ	ออน
ลบ	1	มาร์ค	ออน

รูปแสดง ย่านของแรงดันไฟฟ้าที่ใช้ในสัญญาณ RS -232 C

สำหรับไมโครคอมพิวเตอร์บางเครื่อง ใช้แต่สัญญาณลอจิกออกมา เป็นสัญญาณของ RS-232C เลย อย่างเช่น อะซิงโครนัส อะแดปเตอร์ ของ Personal Computer ในกรณีเช่นนี้ระยะทางของสายที่เชื่อมต่ออาจจะไปได้สั้นกว่า 50 ฟุต เนื่องจากระดับกราวน์ดเปลี่ยนแปลงไปอันเนื่องจากการสูญเสียไปในความต้านทานของสาย ผู้ที่เคยใช้ Personal Computer อาจจะเคยประสบปัญหานี้มาแล้วว่า ถ้าต่อสัญญาณ RS-232C เกินกว่า 10 ฟุต แล้วใช้งานไม่ได้ แต่อย่างไรก็ตาม RS-232C ของ Personal Computer ยังมีโอกาสให้เลือกใช้ 20 มิลลิแอมแปร์ กระแสวนกลับทวนแรงดันไฟฟ้า (Current loop)

การติดตั้งพอร์ทสื่อสารบนเครื่อง MICRO COMPUTER 16 BIT

บนไมโครคอมพิวเตอร์ 16 บิทที่เหมือนหรือคล้ายของ IBM PC จะมี Hardware Adaptor ที่เรียกว่า Asyncbronous Adaptor shift ที่ ไมโครคอมพิวเตอร์ ใช้ คือ 8250 ซึ่งมีพอร์ทต่อออกมาใช้งานได้ตลอดกำหนดเป็น COM1 หรือ COM2 โปรแกรมไบออสจะเขียนสนับสนุนการใช้พอร์ทสื่อสารนี้แล้ว และหากไม่ต้องการเขียนติดต่อกับพอร์ทสื่อสารในลักษณะเชื่อมต่อกับไบออส ก็สามารถใช้ภาษาชั้นสูงเช่น เบสิก หรือปาสคาลได้

การเขียนโปรแกรมสื่อสารโดยภาษาเบสิก

ภาษาเบสิกเป็นภาษาที่ใช้ได้ง่ายสำหรับ ไมโครคอมพิวเตอร์ สำหรับในภาษาเบสิกที่ใช้กับเครื่อง Personal Computer มีคำสั่งที่จะใช้งานสำหรับการสื่อสารอยู่ด้วย แต่ก่อนการใช้งานพอร์ทสื่อสารจะต้องกำหนดบัฟเฟอร์ที่ใช้ในการสื่อสารก่อน

บัฟเฟอร์สำหรับการสื่อสารก็คือหน่วยความจำในคอมพิวเตอร์ ที่แบ่งแยกออกมาจากหน่วยความจำหลักสำหรับเก็บพักข้อมูลในการติดต่อชั่วคราว บัฟเฟอร์สำหรับการสื่อสารนี้ส่วนมากใช้สำหรับฝ่ายรับเท่านั้น เนื่องจากฝ่ายรับจำเป็นจะต้องตามฝ่ายส่งให้ทัน ถ้าหากฝ่ายรับใช้ภาษาเบสิก ซึ่งจะต้องผ่านตัวแปลภาษาทำให้การทำงานช้าลง ข้อมูลที่จัดส่งมาให้ คอมพิวเตอร์ จะได้รับการนำมาไว้ในแผ่นบัฟเฟอร์ก่อน แล้วจึงอ่านจากบัฟเฟอร์ลงไว้ในดิสค์ บัฟเฟอร์ของฝ่ายรับมีผลกระทบต่อการรับ ฝ่ายรับจะรับข้อมูลเข้ามาเก็บไว้ในบัฟเฟอร์ก่อน จนกว่าโปรแกรมควบคุมการสื่อสารจะนำข้อมูลออกไปจากบัฟเฟอร์รับเพื่อไปแสดงหรือพิมพ์หรือเก็บไว้ในแฟ้ม ในระบบควบคุมการทำงานของ ไมโครคอมพิวเตอร์ มีกลไกในการจัดการบัฟเฟอร์รับส่งนี้อยู่แล้ว โปรแกรมในระดับสูงจึงเพียงแต่ทำหน้าที่ดึงเอาข้อมูลจากบัฟเฟอร์นี้ไปใช้ ซึ่งจะเห็นได้ชัดถึงความจำเป็นในการใช้บัฟเฟอร์เมื่อความเร็วในการส่งเกินกว่า 600 บอต ภาษาเบสิกจะไม่สามารถรับข้อมูลจากพอร์ทอนุกรมทันจึงต้องมีบัฟเฟอร์

หน้าที่ของโปรแกรมควบคุมการรับส่งก็คือ การอ่านข้อมูลจากบัฟเฟอร์รับไปใช้เมื่อข้อมูลถูกอ่านไปแล้วตัวข้อมูลก็จะหายไปจากบัฟเฟอร์ ภาษาเบสิกนำข้อมูลออกไป ถ้าเปรียบเสมือนว่าไบออสรับน้ำใส่ตุ่ม เบสิกน้ำออกไป ถ้าอัตราการใส่น้ำมากกว่าเอาออกไปก็จะเกิดปรากฏการณ์ที่เรียกว่า บัฟเฟอร์รับไบโอเวอร์โฟลว์

ในเครื่อง Personal Computer อาจกำหนดบัฟเฟอร์รับได้โดยจะกระทำเมื่อเริ่มต้นเรียกใช้ เช่น

A:\BASIC/C:4096

เป็นการกำหนดบัฟเฟอร์การสื่อสารด้วยขนาด 4 กิโลไบต์ ถ้าหากไม่ใส่ค่ากำหนดลงไป MS DOS จะตั้งค่าไว้ให้เท่ากับ 256 ไบต์ โดยการใส่ /C: ทำให้สามารถกำหนดขนาดบัฟเฟอร์ได้จาก 0 ถึง 32,767 ไบต์ อย่างไรก็ตาม ภาษาเบสิกที่ใช้นั้นจะเรียกใช้หน่วยความจำได้เพียง 64 KB หากกำหนดค่าบัฟเฟอร์สื่อสารเอาไว้มากย่อมจะมีหน่วยความจำเหลือสำหรับใช้อย่างอื่นน้อยลง

คำสั่งใช้พอร์ทสื่อสาร

เมื่อเข้ามาในภาษาเบสิกแล้ว สามารถเปิดพอร์ทสื่อสารเหมือนเป็นไฟล์ ๆ หนึ่งเพื่อการเขียนอ่านได้ คำสั่งที่ใช้ คือ

OPEN "COM n:[speed] [,parity] [,data] [,stop] [.RS] [,CS[n]] [,CS] [,DS[n]] [,LF]" AS[#] filenum [IEN = number]

เป็นคำสั่งสำหรับเปิดแฟ้มสื่อสารซึ่ง ไมโครซอฟท์ ได้กำหนดให้บอกสื่อสารเป็นแฟ้มหนึ่ง ก่อนจะใช้งานจึงต้องเปิดแฟ้มเสียก่อน การเปิดแฟ้มจะต้องบ่งบอกพารามิเตอร์ต่าง ๆ ของการสื่อสารเอาไว้ ถ้าหากพารามิเตอร์ที่อยู่ใน [ ] ไม่ได้บ่งบอกเอาไว้หรือไม่ได้ใส่ ภาษาเบสิกจะเอาค่าที่ตั้งเอาไว้โดย OS ใส่ให้

ความหมายของพารามิเตอร์ต่าง ๆ มีดังนี้

COM n จะเป็น 1 หรือ 2 สำหรับบอกว่าใช้อะซิงโครนัส อะแดปเตอร์ตัวไหน ใน MS-DOS มีได้ 2 พอร์ท

Speed คือ ความเร็วที่จะใช้ในการรับส่งว่าจะเป็นเท่าไรในหน่วยของบิทต่อวินาที มีให้เลือกดังนี้ 1200, 1800, 2400, 4800, และ 9600 ตั้งค่าเอาไว้เป็น 9600

Parity เป็นอักษรตัวเดียว สำหรับบอกชนิดของการตรวจสอบข้อผิดพลาดมีให้เลือกดังนี้

S : space ในช่วงที่เป็นบิทพาริตี้จะส่ง 0 ไปแทน

O : odd เป็นการตรวจสอบแบบ odd parity

M : mark ในช่วงบิทพาริตี้นี้ จะใส่ค่า "1"

E : even ให้มีการตรวจสอบพาริตี้แบบ even

N : none ไม่มีการตรวจสอบหรือไม่ใส่บิทพาริตี้นี้

Data เป็นตัวเลขที่จะกำหนดว่าบิทข้อมูลมีกี่บิท ค่าที่ให้เลือก คือ 1 และ 2 ค่าที่ตั้งโดย OS คือ 1

Filenum คือ ตัวเลขที่บอกว่าแฟ้มนี้ถือว่าเป็นแฟ้มที่เท่าไร เพื่อเป็นการอ้างถึงคำสั่งต่าง ๆ ที่เกี่ยวกับแฟ้มจะได้ทราบว่าเป็นแฟ้มไหน อนึ่ง ในเบสิกยอมให้เปิดแฟ้มพร้อมกันได้ไม่เกิน 3 แฟ้ม ดังนั้น เฉพาะเลขแฟ้มจึงมีค่า 0, 1 และ 2 เท่านั้น ในกรณีที่ต้องการเปิดแฟ้มมากกว่า 3 จำเป็นจะต้องบอกก่อนในขณะเริ่มใช้เบสิกโดยการใช้

A> BASIC/F:7

/F:7 เฉพาะถึงการขอเปิดแฟ้มพร้อมกันได้ 7 แฟ้ม แต่ถ้าเป็นคอมพายเลอร์จะเปิดแฟ้มพร้อมกันได้ไม่จำกัดจำนวน

number เลขที่ต่อด้วย LEN เป็นการบอกภาษาเบสิกว่าจำนวนไบต์สูงสุด ซึ่งสามารถอ่านมาจากบัฟเฟอร์สื่อสารโดยคำสั่ง GET หรือ PUT โดยปกติถ้าไม่ตั้งไว้ OS จะตั้งค่าเป็น 128 ไว้สำหรับ RS, CS, DS, CD และ LF เป็นการเปิดโอกาสให้ผู้ใช้ควบคุมสัญญาณที่มาตามสายดังนี้

RS เป็นการตัด RTS ออก สัญญาณ RTS จะมีแรงดันเป็นเมื่อภาษาเบสิก เปิดแฟ้มสื่อสารด้วยคำสั่ง OPEN "COM นอกจากจะใส่ RS ในคำสั่งเปิดแฟ้มนั้น

CS[n] เป็นการควบคุม CTS (Clear to Send) n คือ ค่าเป็นมิลลิวินาที ที่จะกำหนดให้ซีพียูรอสัญญาณ CTS ก่อนที่ซีพียูจะถือว่าเป็นข้อผิดพลาดชนิดที่เรียกว่า "Drivce Time Out" n มีค่าได้ตั้งแต่ 0 ถึง 65535 ถ้า n = 0 หมายความว่าไม่เอาสัญญาณ CTS มาตรวจสอบ OS จะตั้งค่า n เป็น 1000 ไว้

DS[n] เช่นเดียวกับ CS[n] แต่เป็นการควบคุม DSR (Data Set Ready) ค่าที่ OS ตั้งไว้คือ 1000

CD[n] เช่นเดียวกัน CS[n] และ DS[n] แต่ใช้ควบคุม CD หรือ Carrier Detect OS ตั้งค่าเอาไว้คือ 0

สำหรับ LF เป็นการใช้สำหรับการถ่ายโอนข้อมูลในแฟ้มไปยังเครื่องพิมพ์ เป็นการส่งสัญญาณ LF(0A) โดยอัตโนมัติ เมื่อพบ CR หรือ Carriage Return

<u>ความสัมพันธ์ระหว่าง DOS กับการใช้ภาษาเบสิกสำหรับการสื่อสาร</u>

เมื่อภาษาเบสิกเปิดแฟ้มสื่อสารดดยคำสั่ง OPEN "COM.." ภาษาเบสิกจะเรียกใช้ DOS ในการดึงข้อมูลจาก RS-232 พอร์ตมาใส่ไว้ในบัฟเฟอร์ ดังที่กล่าวมาแล้ว และถ้าจะส่งคือดึงข้อมูลจากบัฟเฟอร์ส่งออกมาที่ RS-232 รูปแบบการทำงานจะเป็นดังไดอะแกรม

```
10 REM PROGRAM FOR PHICOM BONDER DATA TRANSFER PROJ.I-II
20 CLS:GOTO 770
30 ON ERROR GOTO  730
40 OPEN"COM1:1200,E,7,2,CS,DS" AS #1
50 REM CONTROL CODE
60 C1$=CHR$(19):C2$=CHR$(4):C3$=CHR$(46):C4$=CHR$(42):C5$=CHR$(17)
   :C6$=CHR$(78)
70 C7$=C2$+C6$+"4"+C1$ :C8$=CHR$(13)+CHR$(10) :C9$=CHR$(13)
80 LEAD$=C2$+C4$+C1$
90 GOSUB 950
100 IF EOF(1) THEN 100
110 FOR L0=1 TO 400 :NEXT L0:PRINT LOC(1); :IF LOC(1)>=2 THEN 160
120 PIN$=INPUT$(LOC(1),#1)
130 IF PIN$=CHR$(123) THEN PRINT "mux check ok" :GOTO 100
140 IF PIN$=CHR$(124) THEN PRINT "pc send error" :GOTO 100
150 PRINT "COMMUNICATION ERROR PLEASE CHECK CABLE BEFORE NEXT X-SFER."
   : PRINT PIN$:GOTO 100
160 PIN$=INPUT$(LOC(1),#1) :PM$=MID$(PIN$,1,1)  :PRINT PIN$
170 ONO=INSTR(PIN$,C9$)
180 IF PM$="R" THEN GOTO 210
190 IF PM$="W" THEN GOTO 480
200 GOTO 100
210 ONO$=MID$(PIN$,1,ONO-1) :ONO$="c:\data"+ONO$
220 OPEN "R" ,#2,ONO$
230 FIELD #2,128 AS IP$
240 I=1
250 GET #2,I :A$=IP$
260 FOR K=1 TO 128
270 K$=MID$(A$,K,1)
280 IF K$=C1$ THEN GOSUB 330
290 IF K$=":" THEN GOTO 380
300 PRINT #1,K$;
310 NEXT K
```

```
320 I=I+1 :GOTO 250
330 IF EOF(1) THEN 330
340 FOR LO=1 TO 20 : NEXT LO
350 Y$=INPUT$(LOC(1),#1): PRINT Y$;
360 RETURN
370 REM :EOF
380 PRINT #1,":EOF" +C8$;:PRINT  ":EOF"+C8$;
390 GOSUB 330
400 PRINT #1,C1$+CHR$(0)+C1$+C8$;:PRINT  C1$+CHR$(0)+C1$+C8$;
410 GOSUB 330
420 PRINT #1,C2$+C4$+C1$+C8$; :PRINT C2$+C4$+C1$+C8$;
430 GOSUB 330
440 PRINT " END OF PROGRAM"
450 CLOSE #2: PRINT #1,CHR$(254);
    :PRINT CHR$(254)+"  PROGRAM SEND ALREADY"
460 GOSUB 950
470 GOTO 100
480 IF (ONO-2)<0 THEN GOTO 100
490 ONO$="R"+MID$(PIN$,2,ONO-2) :ONO$="c:\data"+ONO$
500 OPEN "R",#2,ONO$,128
510 FIELD #2,128 AS OP$  :J=1  : BF$=""
520 PRINT #1,C1$+C3$
530 IF EOF(1) THEN 530
540 BF$=BF$+INPUT$(LOC(1),#1):BF=INSTR(BF$,C2$)
550 IF BF<>0 THEN GOTO 600
560 IF LEN(BF$) < 128 THEN GOTO 530
570 LSET OP$=BF$ : BF$=RIGHT$(BF$,LEN(BF$)-128)
580 PUT #2,J: J=J+1
590 GOTO 530
600 BF1$=MID$(BF$,1,BF-1): LSET OP$=BF1$
    :IF LEN(BF1$)>128 THEN GOTO 710
610 FOR Z =1 TO 1000:NEXT Z
```

```
620 PUT #2,J :J=J+1
630 IF EOF(1) THEN 630
640 Y$=INPUT$(LOC(1),#1):PRINT Y$
650 PRINT #1,LEN$
    :PRINT ONU$+" DATA TRANSFER READY  AND WAIT FOR NEXT PROGRAM"
660 FOR Q = 1 TO 400:NEXT Q
670 FOR Y=1 TO 10 : NEXT Y
680 IF EOF(1) THEN 680
690 Y$=INPUT$(LOC(1),#1): PRINT Y$
700 CLOSE #2 :PRINT #1,CHR$(254);:GOSUB 950:GOTO 100
710 PUT #2,J: J=J+1 :BF1$=RIGHT$(BF1$,LEN(BF1$)-128)
    :LSET UP$=Bf1$
720 GOTO 620
730 IF ERR=57 THEN GOTO 750
740 PRINT ERR,ERL: PRINT #1,CHR$(254);
    :PRINT "***COMMUNICATION ERROR PLEASE PRESS PHICOM ***"
    :RESUME  100
750 PRINT "COMMUNICATION ERROR PLEASE CHECK CABLE ";:PRINT ERR
760 PRINT #1,CHR$(254) :CLOSE:RESUME 100
770 A$="rest" :KEY 5,A$  :B$="check":KEY 6,B$
780 ON KEY(1) GOSUB 850:ON KEY(2) GOSUB 860:ON KEY(3) GOSUB 870
790 ON KEY(4) GOSUB 880
800 ON KEY(5) GOSUB 890:ON KEY(6) GOSUB 900:ON KEY(7) GOSUB 910
810 ON KEY(8) GOSUB 920:ON KEY(9)GOSUB 930:ON KEY(10) GOSUB 940
820 KEY(1) ON :KEY(2) ON :KEY(3) ON :KEY(4)ON :KEY(5)  ON
830 KEY(6) ON :KEY(7) ON :KEY(8) ON :KEY(9)ON :KEY(10) ON
840 GOTO 30
850 GOSUB 1160:SHELL"TYPE PROD1.TXT":RETURN
860 GOSUB 1160:SHELL"TYPE PROD2.TXT":RETURN
870 GOSUB 1160:SHELL"TYPE PROD3.TXT":RETURN
880 GOSUB 1160:SHELL"TYPE PROD4.TXT":RETURN
890 GOSUB 1160:SHELL"TYPE PROD5.TXT":RETURN
```

```
900 GOSUB 1160:SHELL"TYPE PROD6.TXT":RETURN
910 GOSUB 1160:SHELL"TYPE PROD7.TXT":RETURN
920 GOSUB 1160:SHELL"TYPE PROD8.TXT":RETURN
930 GOSUB 1160:SHELL"TYPE PROD9.TXT":RETURN
940 LOCATE 1,1:GOTO 90
950 REM MENU
960 CLS
970 LOCATE 3,10 :PRINT "*****************************************"
980 LOCATE 4,10 :PRINT "*                                       *"
990 LOCATE 5,10 :PRINT "*       WELCOME TO NEW PHICOM DISK      *"
1000 LOCATE 6,10 :PRINT "*       ==========================     *"
1010 LOCATE 8,10 :PRINT "*                                       *"
1020 LOCATE 7,10 :PRINT "*  INSTRUCTION :                        *"
1030 LOCATE 8,10 :PRINT "*  -----------                          *"
1040 LOCATE 8,10 :PRINT "*                                       *"
1050 LOCATE 9,10 :PRINT "*    SELECT PROGRAM NAME LIST BY F1- F4  *"
1060 LOCATE 10,10:PRINT "*                                       *"
1070 LOCATE 11,10:PRINT "*   -  PRESS   F1 = LIST PROGRAM PAGE 1  *"
1080 LOCATE 12,10:PRINT "*              F2 = LIST PROGRAM PAGE 2  *"
1090 LOCATE 13,10:PRINT "*     F3 - F9 = LIST PROGRAM PAGE 3 - 9  *"
1100 LOCATE 14,10:PRINT "*                                       *"
1110 LOCATE 15,10:PRINT "*                                       *"
1120 LOCATE 16,10:PRINT "*                            E-SHOP     *"
1130 LOCATE 17,10:PRINT "*****************************************"
1140 LOCATE 19,10:PRINT "[Mux. AND Disk Program OK and Wait for you]"
1150 RETURN
1160 LOCATE 1,1:CLS:LOCATE 15,25:PRINT"WAIT FOR LOADING !!"
1170 RETURE
```

```
0000                    PAGE   56
                ;*****************************************
                ;*      AUTOMETIC DATA TRANSFER          *
                ;*          FOR PHICOM WIRE BONDER       *
                ;*                                       *
                ;*             Version 1.00              *
                ;*   FEB'1990 E-SHOP SIGNETICS THAILAND  *
                ;*****************************************
                ;
0000                    CPU    "Z80.TBL"
0000                    HOF    "INT8"
0000                    ORG    0000H

0000 =          PHICOM:         EQU    00H             ;PHICOM CALL
0001 =          SEGM:           EQU    01H             ;SEGMENT
0002 =          DIGIT:          EQU    02H             ;DIGIT
0003 =          CTRL:           EQU    03H             ;CONTROL
0004 =          SLOTCHK:        EQU    04H             ;SLOT CHECK
0008 =          IBMDAT:         EQU    08H             ;IBM DATA PORT
0009 =          IBMCTRL:        EQU    09H             ;IBM CONTROL PORT
000C =          WATDOG:         EQU    0CH             ;WATCH DOG


                ;*---------------- POWER UP ----------------------*

0000 AF         POWER:          XOR    A               ;POWER UP DELAY
0001 3D         POWER1:         DEC    A
0002 00                         NOP
0003 20FC                       JR     NZ,POWER1
0005 C30001                     JP     INIT

                ;*---------RESTART  10H SYSTEM CALL --------------*
                ;*                                              *
                ;*      INPUT   A = FUNCTION CODE               *
                ;*                                              *
                ;*----------------------------------------------*
0010                    ORG    0010H

0010 FE16               CP     16H      ; 16H = NOUMBER OF FUNC CALL
0012 D0                 RET    NC
0013 E5                 PUSH   HL
0014 C5                 PUSH   BC
0015 C32E01             JP     SYSCAL

                ;*------- NMI FOR DISPLAY AND KEY BOORD -----------*
                ;*
                ;*
0066                    ORG    0066H

0066 E5                 PUSH   HL
0067 D5                 PUSH   DE
0068 C5                 PUSH   BC
0069 F5                 PUSH   AF
006A FDE5               PUSH   IY
006C DDE5               PUSH   IX
006E 3E07               LD     A,07H            ;DISPALY SCAN
```

```
0070 D7                 RST   10H
0071 3E08               LD    A,08H             ;PHICOM SCAN
0073 D7                 RST   10H
0074 0E0C               LD    C,WATDOG          ;RESET WATCH_DOG
0076 3E00               LD    A,00H
0078 ED79               OUT   (C),A
007A DDE1               POP   IX
007C FDE1               POP   IY
007E F1                 POP   AF
007F C1                 POP   BC
0080 D1                 POP   DE
0081 E1                 POP   HL
0082 ED45               RETN

            ;*------------ INITIAL PORT ----------------------*

0100                    ORG   0100H

0100 31FE27   INIT:     LD    SP,STACK      ;SYSTEM STACK
0103 ED56               IM    1
0105 0E0C               LD    C,WATDOG      ;RESET WATCH_DOG
0107 3E00               LD    A,00H
0109 ED79               OUT   (C),A
010B 0E03               LD    C,CTRL        ;SET 8255
010D 3E88               LD    A,88H
010F ED79               OUT   (C),A
0111 0E09               LD    C,I8MCTRL     ;SET I8M
0113 3ED9               LD    A,0D9H        ;1200,E,7,2
0115 ED79               OUT   (C),A
0117 3E16               LD    A,16H         ;RXSET
0119 ED79               OUT   (C),A
011B 110018             LD    DE,1800H      ;CONT = 24
011E 218004             LD    HL,PHIC_TAB
0121 4E       INIT1:    LD    C,(HL)
0122 3ED9               LD    A,0D9H        ;1200,E,7,2
0124 ED79               OUT   (C),A
0126 3E16               LD    A,16H         ;8251
0128 ED79               OUT   (C),A         ;ALL
012A 23                 INC   HL
012B 10F4               DJNZ  INIT1
012D C9                 RET

            ;*-------------------------------------------------*

012E D5       SYSCAL:   PUSH  DE
012F 211805             LD    HL,CAL_TAB
0132 47                 LD    B,A
0133 CD3A01             CALL  TABLE
0136 D1                 POP   DE
0137 E1                 POP   HL
0138 E3                 EX    (SP),HL
0139 C9                 RET

            ;*------------ TABLE SUB -------------------------*
            ;*       LOOK UP TABLE 2 BYTE                    *
            ;*     INPUT   B  = No.                          *
```

```
                ;*          HL = START TABLE                    *
                ;*     OUTPUT  HL = DATA FROM TABLE             *
                ;*     REG.    A,BC,DE,HL                       *
                ;*----------------------------------------------*
013A 78         TABLE:      LD    A,B
013B 87                     ADD   A,A
013C 0600                   LD    B,0
013E 4F                     LD    C,A
013F 09                     ADD   HL,BC
0140 5E                     LD    E,(HL)
0141 23                     INC   HL
0142 56                     LD    D,(HL)
0143 EB                     EX    DE,HL
0144 C9                     RET

                ;*------------DISPLAY NUMBER -------------------*
                ;*                                             *
                ;*    CODE     01H                             *
                ;*    INPUT    E = NUMBER                      *
                ;*    OUTPUT                                   *
                ;*    REG.     A,BC,DE,HL                      *
                ;*----------------------------------------------*

0145 1600       DISP_NUM:   LD    D,00H
0147 21CC04                 LD    HL,NUM_TAB      ;NUMBER TABLE
014A 19                     ADD   HL,DE           ;LOOK UP TABLE
014B 7E                     LD    A,(HL)
014C 110520                 LD    DE,DISP_BUFF+5
014F 210620                 LD    HL,DISP_BUFF+6  ;MOVE DIGIT
0152 010200                 LD    BC,0002H
0155 EDB0                   LDIR
0157 77                     LD    (HL),A          ;ADD NUMBER
0158 C9                     RET

                ;*------------ DISPLAY FUNCTION ----------------*
                ;*                                             *
                ;*    CODE     02H                             *
                ;*    INPUT    HL = FUNC_TAB                   *
                ;*             B  = DIGIT START                *
                ;*    OUTPUT   DISP_BUFF                       *
                ;*    REG      A,BC,DE,HL                      *
                ;*----------------------------------------------*

0159 110020     DISP_FUNC:  LD    DE,DISP_BUFF    ;SET DIGIT
015C 78                     LD    A,E
015D 80                     ADD   A,B
015E 5F                     LD    E,A
015F 3001                   JR    NC,DISP_FUNC1
0161 14                     INC   D
0162 7E         DISP_FUNC1: LD    A,(HL)          ;MOVE DIGIT
0163 4F                     LD    C,A
0164 23                     INC   HL
0165 0600                   LD    B,00H
0167 EDB0                   LDIR
0169 C9                     RET
```

```
                ;*------------ SAVE NUMBER ------------------------*
                ;*                                                 *
                ;*    CODE     03H                                 *
                ;*    INPUT    E  = NUMBER                         *
                ;*    OUTPUT   NUM_BUFF                            *
                ;*    REG.     A,E,HL                              *
                ;*-------------------------------------------------*

016A 210F20     SAVE_NUM:       LD      HL,NUM_BUFF+1       ;MOVE ONE DIGIT
016D 111020                     LD      DE,NUM_BUFF+2
0170 78                         LD      A,E
0171 010200                     LD      BC,02H
0174 EDB8                       LDDR
0176 77                         LD      (HL),A
0177 C9                         RET

                ;*------------ CHECK KEY PRESS --------------------*
                ;*                                                 *
                ;*     CODE      04H                               *
                ;*     INPUT     KEY_IN                            *
                ;*     OUTPUT    KEY_BUFF                          *
                ;*               CY  0 =  KEY NOT PRESS            *
                ;*                   1 =  KEY PRESS                *
                ;*               A     =  KEY                      *
                ;*     REG.      A,DE,HL                           *
                ;*-------------------------------------------------*

0178 210A20     CHK_KEY:        LD      HL,KEY_IN
0178 7E                         LD      A,(HL)
017C FEFF                       CP      0FFH
017E 200A                       JR      NZ,CHK_KEY1         ;KEY NOT PRESS
0180 210B20                     LD      HL,KEY_BUFF
0183 7E                         LD      A,(HL)
0184 FEFF                       CP      0FFH
0186 2807                       JR      Z,CHK_KEY2
0188 37                         SCF                         ; SET CY = 1
0189 C9                         RET
018A 110B20     CHK_KEY1:       LD      DE,KEY_BUFF
018D 7E                         LD      A,(HL)              ; COPY KEY_IN TO KEY_BUFF
018E 12                         LD      (DE),A
018F 3F         CHK_KEY2:       CCF                         ; SET CY.= 0
0190 C9                         RET


                ;*------------ SCANS ------------------------------*
                ;*                                                 *
                ;*     CODE    05H                                 *
                ;*     INPUT   B  =  DELAY                         *
                ;*     OUTPUT                                      *
                ;*     REG     A,B,DE,HL                           *
                ;*-------------------------------------------------*

0191 110008     SCAND:          LD      DE,800H
0194 210020                     LD      HL,DISP_BUFF
0197 CDA101     SCAND1:         CALL    SCANS
019A 1C                         INC     E
```

```
0198 15                 DEC     D
019C 20F9               JR      NZ,SCAND1
019E 10F1               DJNZ    SCAND
01A0 C9                 RET
01A1 78         SCANS:  LD      A,E
01A2 D302               OUT     (DIGIT),A   ;OUT DIGIT
01A4 7E                 LD      A,(HL)
01A5 D301               OUT     (SEGM),A    ;OUT SEGMENT
01A7 AF                 XOR     A
01A8 3D         SCANS1: DEC     A
01A9 20FD               JR      NZ,SCANS1   ;DELAY
01AB D301               OUT     (SEGM),A
01AD 23                 INC     HL
01AE C9                 RET

                ;*------------ CLEAR DISPLAY ----------------------*
                ;*                                                *
                ;*      CODE      06H                             *
                ;*      INPUT                                     *
                ;*      OUTPUT    DISP_BUFF                       *
                ;*      REG.      A,B,HL                          *
                ;*------------------------------------------------*

01AF AF         CLR_DISP:       XOR     A
01B0 210020                     LD      HL,DISP_BUFF
01B3 0608                       LD      B,08H
01B5 23         CLEAR1:         INC     HL
01B6 77                         LD      (HL),A
01B7 10FC                       DJNZ    CLEAR1
01B9 C9                         RET

                ;*------------DISP/SCAN KEY ----------------------*
                ;*                                                *
                ;*      CODE      07H                             *
                ;*      INPUT                                     *
                ;*      OUTPUT    KEY_IN                          *
                ;*      REG.      A,BC,DE,HL                      *
                ;*------------------------------------------------*

01BA 110008     SCANK:          LD      DE, 800H
01BD 210020                     LD      HL,DISP_BUFF
01C0 3EFF                       LD      A,0FFH
01C2 320A20                     LD      (KEY_IN),A
01C5 CDA101     SCANK1:         CALL    SCANS
01C8 D802                       IN      A,(DIGIT)
01CA E638                       AND     38H
01CC FE38                       CP      38H
01CE 2816                       JR      Z,SCANK2
01D0 83                         OR      E
01D1 C5                         PUSH    BC
01D2 D5                         PUSH    DE
01D3 E5                         PUSH    HL
01D4 219304                     LD      HL,KEY_TAB+15
01D7 060F                       LD      B,0FH           ;16 KEY
01D9 BE         SCANK11:        CP      (HL)            ;LOOK UP TABLE
01DA 2803                       JR      Z,SCANK12
```

```
01DC 28                 DEC     HL
01DD 10FA               DJNZ    SCANK11
01DF 78         SCANK12:LD      A,B
01E0 E1                 POP     HL
01E1 D1                 POP     DE
01E2 C1                 POP     BC
01E3 320A20             LD      (KEY_IN),A      ;SAVE KEY
01E6 1C         SCANK2: INC     E
01E7 15                 DEC     D
01E8 2008               JR      NZ,SCANK1
01EA C9                 RET


                ;*---------- SCAN PROT (PHICOM SCAN) ------------*
                ;*                                              *
                ;*      CODE      08H                           *
                ;*      INPUT                                   *
                ;*      OUTPUT    PORT_BUFF                     *
                ;*      REG.      A,BC,HL.                      *
                ;*----------------------------------------------*

01EB 01001C     SCANP:  LD      BC,1C00H
01EE 3EFF               LD      A,0FFH
01F0 320820             LD      (PORT_BUFF),A   ;CLEAR PORT BUFFOR
01F3 79         SCANP1: LD      A,C
01F4 D300               OUT     (PHICOM),A
01F6 3E0F               LD      A,0FH
01F8 3D         SCANP2: DEC     A               ;DELAY
01F9 20FD               JR      NZ,SCANP2
01FB DB02               IN      A,(DIGIT)
01FD CB7F               BIT     7,A
01FF 200F               JR      NZ,SCANP3       ;NOT CALL
0201 210A05             LD      HL,PHI_TAB
0204 C5                 PUSH    BC              ;TABLE
0205 0600               LD      B,00H           ;LOOKUP
0207 09                 ADD     HL,BC
0208 7E                 LD      A,(HL)
0209 C1                 POP     BC
020A 320820             LD      (PORT_BUFF),A   ;SAVE PHICOM CALL
020D CD1402             CALL    COMPQ
0210 0C         SCANP3: INC     C
0211 10E0               DJNZ    SCANP1
0213 C9                 RET


                ;*------------ COMPAIR QUEUE ---------------------*
                ;*                                              *
                ;*      CODE      09H.                          *
                ;*      INPUT     E = PHICOM NUMBER             *
                ;*      OUTPUT    A = 0  NOT FOUND IN QUEUE     *
                ;*                  = 1  FOUND IN QUEUE         *
                ;*      REG.      A,B,HL                        *
                ;*----------------------------------------------*

0214 211B20     COMPQ:  LD      HL,QUE_BUFF
0217 3A1A20             LD      A,(QUE_CONT)
021A 47                 LD      B,A
```

```
0218 85                 ADD     A,L
021C 6F                 LD      L,A
021D 78                 LD      A,E
021E 8E         COMP1:  CP      (HL)
021F C8                 RET     Z
0220 28                 DEC     HL
0221 10F8               DJNZ    COMP1
0223 211820             LD      HL,QUE_BUFF
0226 3A1A20             LD      A,(QUE_CONT)
0229 3C                 INC     A
022A 321A20             LD      (QUE_CONT),A
022D 85                 ADD     A,L
022E 6F                 LD      L,A
022F 78                 LD      A,E
0230 77                 LD      (HL),A
0231 C9                 RET

                ;*---------- CHECK DATA IN BUFFER ----------*
                ;*                                         *
                ;*    CODE    0AH                          *
                ;*    INPUT   HL = DIR NUMBER              *
                ;*    OUTRUT  HL = DIR NUMBER              *
                ;*            CY  0 = NOT FOUND            *
                ;*                1 = FOUND                *
                ;*    REG.    A,BC,DE,HL                   *
                ;*-----------------------------------------*
0232 3F         CHK_BUFF:   CCF                 ; CY = 0
0233 110E20                 LD      DE,NUM_BUFF
0236 0603                   LD      B,03H
0238 220D20                 LD      (CURR_PROG),HL
023B 7E                     LD      A,(HL)
023C FE00                   CP      0
023E 2809                   JR      Z,CHK_BUFF2
0240 23         CHK_BUFF1:  INC     HL
0241 1A                     LD      A,(DE)
0242 BE                     CP      (HL)
0243 2004                   JR      NZ,CHK_BUFF2
0245 13                     INC     DE
0246 10F8                   DJNZ    CHK_BUFF1
0248 37                     SCF                 ; CY = 1
0249 2A0D20     CHK_BUFF2:  LD      HL,(CURR_PROG)
024C C9                     RET

                ;*---------- DISPLAY CURRENT PROG.----------*
                ;*                                         *
                ;*    CODE    0BH                          *
                ;*    INPUT                                *
                ;*    OUTPUT  DISP_BUFF+3                  *
                ;*    REG.    A,DE,HL                      *
                ;*-----------------------------------------*

024D 210D20     DISP_CURR:  LD      HL,CURR_PROG
0250 7E                     LD      A,(HL)
0251 3C                     INC     A
0252 5F                     LD      E,A
0253 1600                   LD      D,00H
```

```
0255 21CC04                 LD      HL,NUM_TAB
0258 19                     ADD     HL,DE
0259 7E                     LD      A,(HL)
025A 210320                 LD      HL,DISP_BUFF+3
025D 77                     LD      (HL),A
025E C9                     RET

                ;*-------- MOVE DIR PROG TO CURRENT PROG.----------*
                ;*                                                *
                ;*  CODE       0CH                                *
                ;*  INPUT                                         *
                ;*  OUTPUT     CURR_PROG.                         *
                ;*  REG.       A,BC,DE,HL                         *
                ;*------------------------------------------------*

025F 110D20     DIRTOCURR:      LD      DE,CURR_PROG
0262 213320                     LD      HL,DIR1
0265 1A                         LD      A,(DE)
0266 47                         LD      B,A
0267 2806                       JR      Z,CHK_DIR
0269 110800                     LD      DE,08H
026C 19         SEARCH_DIR:     ADD     HL,DE           ;SEARCH DIR
026D 10FD                       DJNZ    SEARCH_DIR
026F 110D20     CHK_DIR:        LD      DE,CURR_PROG    ;CHECK AND MOVE
0272 7E                         LD      A,(HL)
0273 23                         INC     HL
0274 13                         INC     DE
0275 010300                     LD      BC,03H
0278 B7                         OR      A
0279 2803                       JR      Z,EMTY
027B EDB0                       LDIR
027D C9                         RET
027E 3E00       EMTY:           LD      A,00H           ;DIR EMTY
0280 77         EMTY1:          LD      (HL),A
0281 23                         INC     HL
0282 10FC                       DJNZ    EMTY1
0284 C9                         RET

                ;*-------- READ HEADER  TO IBM ---------------------*
                ;*                                                *
                ;*    CODE     0DH                                *
                ;*    INPUT                                       *
                ;*    OUTPUT   (C)                                *
                ;*    REG.     A,C,D,HL                           *
                ;*------------------------------------------------*

0285 211420     RD_HEAD:        LD      HL,FC8+1
0288 7E                         LD      A,(HL)
0289 FE08                       CP      IBMDAT
028B C0                         RET     NZ              ;IF NOT IBM
028C 4F                         LD      C,A
028D CDFC02                     CALL    TXSET
0290 1652                       LD      D,52H           ;ASCII "R"
0292 CDC802                     CALL    TXBYTE          ;OUT TO IBM
0295 CD8002                     CALL    OUT_PNUM        ;OUT PROG.No.
0298 CD0203                     CALL    RXSET           ;SET PORT TO RX
```

```
0298 C9                    RET

           ;*-------- WRITE HEADER TO IBM ------------------*
           ;*                                              *
           ;*    CODE     0EH                              *
           ;*    INPUT                                     *
           ;*    OUTPUT   (C)                              *
           ;*    REG.     A,C,D,HL                         *
           ;*----------------------------------------------*

029C 211420  WR_HEAD:      LD      HL,FC8F1
029F 7E                    LD      A,(HL)
02A0 FE08                  CP      IBMDAT
02A2 C0                    RET     NZ              ;IF NOT IBM
02A3 4F                    LD      C,A
02A4 CDFC02                CALL    TXSET
02A7 1657                  LD      D,57H           ;ASCII "W"
02A9 CDC802                CALL    TXBYTE          ;OUT TO IBM
02AC CD8002                CALL    OUT_PNUM        ;OUT PROG.No.
02AF C9                    RET

           ;*-------- OUT PROG. No. TO RS-232 SUB ----------*
           ;*                                              *
           ;*    INPUT    C =  PORT NUMBER                 *
           ;*    OUTPUT   (C)                              *
           ;*    REG.     A,BC,D,HL                        *
           ;*----------------------------------------------*

0280 210E20  OUT_PNUM:     LD      HL,NUM_BUFF
0283 0603                  LD      B,03H           ;DIGIT COUNTER
0285 7E      OUT_PNUM1:    LD      A,(HL)
0286 87                    OR      A
0287 2806                  JR      Z,OUT_PNUM2     ;IF = 0 NOTOUT
0289 C630                  ADD     A,30H           ;CONV TO ASCII
028B 57                    LD      D,A
028C CDC802                CALL    TXBYTE          ;OUT TO PROT
028F 23      OUT_PNUM2:    INC     HL
02C0 10F3                  DJNZ    OUT_PNUM1       ;NEXT BYTE
02C2 160D                  LD      D,0DH           ;CARRIAGE RETURN
02C4 CDC802                CALL    TXBYTE
02C7 C9                    RET

           ;*-------- OUT ONE BYTE TO RS-232 SUB ----------*
           ;*                                              *
           ;*    INPUT    C =  PORT NUMBER                 *
           ;*             D =  DATA                        *
           ;*    OUTPUT   (C),A = 00  OUT OK,A = 01 NOT OK *
           ;*    REG.     A,C,D                            *
           ;*----------------------------------------------*

02C8 0C      TXBYTE:       INC     C               ;CONTROL PORT
02C9 CD7801  TXBYTE1:      CALL    CHK_KEY         ;CHECK KEY PRESS
02CC 3004                  JR      NC,TXBYTE2      ;NOT PRESS
02CE FE0D                  CP      0DH             ;KEY ESC
02D0 280D                  JR      Z,TXBYTE3       ;IF KEY = ESC STOP
02D2 ED78    TXBYTE2:      IN      A,(C)           ;WAIT TxRDY
```

```
02D4 CB47                       BIT     0,A
02D6 28F1                       JR      Z,TXBYTE1
02D8 7A                         LD      A,D
02D9 0D                         DEC     C               ;DATA PORT
02DA ED79                       OUT     (C),A
02DC 3E00                       LD      A,00H
02DE C9                         RET
02DF 3E01       TXBYTE3:        LD      A,01H           ;OUT NOT OK!
02E1 C9                         RET

                ;*-------- IN ONE BYTE FROM RS-232 SUB ----------*
                ;*                                              *
                ;*    INPUT     C = (PORT NUMBER)               *
                ;*    OUTPUT    D = DATA,A = 00 OK,A = 01 NOT OK *
                ;*    REG.      A,C,D                           *
                ;*----------------------------------------------*

02E2 0C         RXBYTE:         INC     C               ;CONTROL PORT
02E3 CD7801     RXBYTE1:        CALL    CHK_KEY         ;CHECK KEY PRESS
02E6 3004                       JR      NC,RXBYTE2      ;NOT PRESS
02E8 FE0D                       CP      0DH             ;KEY ESC
02EA 280D                       JR      Z,RXBYTE3       ;IF KEY = ESC STOP
02EC ED78       RXBYTE2:        IN      A,(C)           ;WAIT RxRDY
02EE CB4F                       BIT     1,A
02F0 28F1                       JR      Z,RXBYTE1
02F2 0D                         DEC     C               ;DATA PORT
02F3 ED78                       IN      A,(C)
02F5 57                         LD      D,A
02F6 3E00                       LD      A,00H           ;INPUT OK
02F8 C9                         RET
02F9 3E01       RXBYTE3:        LD      A,01H           ;INPUT NOT OK!
02FB C9                         RET

                ;*------------ TX SET SUB ----------------------*
                ;*                                              *
                ;*    INPUT     C = (PORT NUMBER)               *
                ;*    OUTPUT                                    *
                ;*    REG.      A,C                             *
                ;*----------------------------------------------*

02FC 0C         TXSET:          INC     C               ;CONTROL PORT
02FD 3E31                       LD      A,31H           ;RTS,ER,TXE
02FF ED79                       OUT     (C),A
0301 C9                         RET

                ;*------------ RX SET SUB ----------------------*
                ;*                                              *
                ;*    INPUT     C = (PORT NUMBER)               *
                ;*    PUTPUT                                    *
                ;*    REG.      A,C                             *
                ;*----------------------------------------------*

0302 0C         RXSET:          INC     C               ;CONTROL PORT
0303 3E16                       LD      A,16H           ;DTR,ER,RXE
0305 ED79                       OUT     (C),A
0307 ED78       RXSET1:         IN      A,(C)
```

```
0309 C87F                   BIT    7,A          ;DSR
0308 23FA                   JR     Z,RXSETI
0300 C9                     RET

            ;*---------- FCB SET SUB ------------------------*
            ;*    INPUT  B = DIR No.                        *
            ;*           C = INPUT  PORT                    *
            ;*           D = OUTPUT PORT                    *
            ;*           E = MODE   00 = READ BY KEY        *
            ;*                      01 = READ BY PHICOM     *
            ;*                      02 = BUFF TO PHICOM     *
            ;*                      03 = WRITE BY PHICOM    *
            ;*    OUTPUT FCB                                *
            ;*    REG.   BC,DE,HL                           *
            ;*------------------------------------------------*

030E 211320 FCB_SET:    LD     HL,FCB
0311 73                 LD     (HL),E        ;SET MODE
0312 23                 INC    HL
0313 71                 LD     (HL),C        ;SET INPUT
0314 23                 INC    HL
0315 72                 LD     (HL),D        ;SET OUTPUT
0316 23                 INC    HL
0317 E5                 PUSH   HL
0318 210E05             LD     HL,BUFF_TAB
0318 CD3A01             CALL   TABLE         ;LOOK UP TABLE
031E D1                 POP    DE
031F 010200             LD     BC,02H        ;SET BUFF ADDR
0322 ED80               LDIR
0324 C9                 RET

            ;---------- DISPLAY PROG. NUMBER SUB ----------

0325 0603   DISP_PROG:  LD     B,03H         ;DISP NUM 3 DIGIT
0327 210E20             LD     HL,NUM_BUFF
032A 5E     DISP_PROG1: LD     E,(HL)
0328 E5                 PUSH   HL
032C C5                 PUSH   BC
032D 3E01               LD     A,01H         ;DISPLAY NUMBER
032F D7                 RST    10H
0330 C1                 POP    BC
0331 E1                 POP    HL
0332 10F6               DJNZ   DISP_PROG1    ;NEXT DIGIT
0334 211220             LD     HL,STEP       ;SET STEP = 1
0337 3E01               LD     A,01H
0339 77                 LD     (HL),A
033A C9                 RET

            ;*---------- MAIN PROGRAM ----------------------*

0338 21E404 MAIN:       LD     HL,RDY_TAB    ;DISPLAY READY
033E 0608               LD     B,08H
0340 3E02               LD     A,02H         ;DISPLAY FUNC.
0342 D7                 RST    10H
0343 210820 MAIN1:      LD     HL,PORT_BUFF  ;CHECK PHICOM CALL
0346 3EFF               LD     A,0FFH
```

```
                ;----------- KEY DEL PRESS ---------------------------

03A5 210C20     KEY_DEL:        LD      HL,FUNC_NEW     ;SET FUNC.
03A8 3E00                       LD      A,00H
03AA 77                         LD      (HL),A
03AB 3E0C                       LD      A,0CH           ;DIR TO CURRENT
03AD D7                         RST     10H
03AE 3E06                       LD      A,06H           ;CLEAR DISPLAY
0380 D7                         RST     10H
0381 21FA04                     LD      HL,DEL_TAB      ;DISPLAY FUNC.
0384 0600                       LD      B,00H
0386 3E02                       LD      A,02H
0388 D7                         RST     10H
0389 3E08                       LD      A,08H           ;DISPLAY CURRENT
038B D7                         RST     10H
038C CD2503                     CALL    DISP_PROG
038F C34303                     JP      MAIN1

                ;----------- KEY LOAD PRESS --------------------------

03C2 210C20     KEY_LD:         LD      HL,FUNC_NEW     ;SET FUNC.
03C5 3E01                       LD      A,01H
03C7 77                         LD      (HL),A
03C8 3E0C                       LD      A,0CH           ;DIR TO CURRENT
03CA D7                         RST     10H
03CB 3E06                       LD      A,06H           ;CLEAR DISPLAY
03CD D7                         RST     10H
03CE 21FE04                     LD      HL,LD_TAB       ;DISP FUNC.LOAD
03D1 0600                       LD      B,00H           ;START DIGIT 0
03D3 3E02                       LD      A,02H
03D5 D7                         RST     10H
03D6 3E08                       LD      A,08H           ;DISPLAY CURRENT
03D8 D7                         RST     10H
03D9 CD2503                     CALL    DISP_PROG
03DC C34303                     JP      MAIN1

                ;----------- KEY CHK PRESS ---------------------------

03DF 210C20     KEY_CHK:        LD      HL,FUNC_NEW     ;SET FUNC.
03E2 3E02                       LD      A,02H
03E4 77                         LD      (HL),A
03E5 3E0C                       LD      A,0CH           ;DIR TO CURRENT
03E7 D7                         RST     10H
03E8 3E06                       LD      A,06H           ;CLEAR DISPLAY
03EA D7                         RST     10H
03EB 210205                     LD      HL,CHK_TAB      ;DISP FUNC.LOAD
03EE 0600                       LD      B,00H           ;START DIGIT 0
03F0 3E02                       LD      A,02H
03F2 D7                         RST     10H
03F3 3E08                       LD      A,08H           ;DISPLAY CURRENT
03F5 D7                         RST     10H
03F6 CD2503                     CALL    DISP_PROG
03F9 C34303                     JP      MAIN1

                ;----------- KEY ESC PRESS ---------------------------
```

```
                ;---------- KEY DEL PRESS ----------------------------

03A5 210C20     KEY_DEL:        LD      HL,FUNC_NEW     ;SET FUNC.
03A8 3E00                       LD      A,00H
03AA 77                         LD      (HL),A
03AB 3E0C                       LD      A,0CH           ;DIR TO CURRENT
03AD D7                         RST     10H
03AE 3E06                       LD      A,06H           ;CLEAR DISPLAY
03B0 D7                         RST     10H
03B1 21FA04                     LD      HL,DEL_TAB      ;DISPLAY FUNC.
03B4 0600                       LD      B,00H
03B6 3E02                       LD      A,02H
03B8 D7                         RST     10H
03B9 3E08                       LD      A,08H           ;DISPLAY CURRENT
03BB D7                         RST     10H
03BC CD2503                     CALL    DISP_PROG
03BF C34303                     JP      MAIN1

                ;---------- KEY LOAD PRESS ---------------------------

03C2 210C20     KEY_LD:         LD      HL,FUNC_NEW     ;SET FUNC.
03C5 3E01                       LD      A,01H
03C7 77                         LD      (HL),A
03C8 3E0C                       LD      A,0CH           ;DIR TO CURRENT
03CA D7                         RST     10H
03CB 3E06                       LD      A,06H           ;CLEAR DISPLAY
03CD D7                         RST     10H
03CE 21FE04                     LD      HL,LD_TAB       ;DISP FUNC.LOAD
03D1 0600                       LD      B,00H           ;START DIGIT 0
03D3 3E02                       LD      A,02H
03D5 D7                         RST     10H
03D6 3E08                       LD      A,08H           ;DISPLAY CURRENT
03D8 D7                         RST     10H
03D9 CD2503                     CALL    DISP_PROG
03DC C34303                     JP      MAIN1

                ;---------- KEY CHK PRESS ----------------------------

03DF 210C20     KEY_CHK:        LD      HL,FUNC_NEW     ;SET FUNC.
03E2 3E02                       LD      A,02H
03E4 77                         LD      (HL),A
03E5 3E0C                       LD      A,0CH           ;DIR TO CURRENT
03E7 D7                         RST     10H
03E8 3E06                       LD      A,06H           ;CLEAR DISPLAY
03EA D7                         RST     10H
03EB 210205                     LD      HL,CHK_TAB      ;DISP FUNC.LOAD
03EE 0600                       LD      B,00H           ;START DIGIT 0
03F0 3E02                       LD      A,02H
03F2 D7                         RST     10H
03F3 3E08                       LD      A,08H           ;DISPLAY CURRENT
03F5 D7                         RST     10H
03F6 CD2503                     CALL    DISP_PROG
03F9 C34303                     JP      MAIN1

                ;---------- KEY ESC PRESS ----------------------------
```

```
03FC 211220    KEY_ESC:     LD    HL,STEP          ;SET STEP = 0
03FF 3E00                   LD    A,00H
0401 77                     LD    (HL),A
0402 C33803                 JP    MAIN

               ;---------- KEY NEXT PRESS ------------------------

0405 211220    KEY_NEXT:    LD    HL,STEP          ;CHECK STEP
0408 7E                     LD    A,(HL)
0409 CB47                   BIT   0,A
040B CA4303                 JP    Z,MAIN1          ;END
040E 210D20                 LD    HL,CURR_PROG
0411 7E                     LD    A,(HL)
0412 3C                     INC   A
0413 FE05                   CP    05H              ;MAX PROG. = 5
0415 2002                   JR    NZ,KEY_NEXT1     ;IF NOT TO SAVE
0417 3E00                   LD    A,00H            ;IF=6 SET TO 0
0419 77        KEY_NEXT1:   LD    (HL),A           ;SAVE NEXT PROG.
041A 3E0C                   LD    A,0CH            ;DIR TO CURRENT
041C D7                     RST   10H
041D 3E08                   LD    A,08H            ;DISPLAY CURRENT
041F D7                     RST   10H
0420 CD2503                 CALL  DISP_PROG
0423 C34303    KEY_NEXT2:   JP    MAIN1

               ;-------------------------------------------------

0426 3A1220    KEY_ENT:     LD    A,(STEP)         ;CHECK STEP
0429 CB47                   BIT   0,A
042B 2809                   JR    Z,KEY_ENT1
042D 3A0C20                 LD    A,(FUNC_NEW)     ;CHECK FUNC.
0430 B7                     OR    A
0431 2806                   JR    Z,DEL_FUNC
0433 3D                     DEC   A
0434 2822                   JR    Z,LOAD_FUNC
0436 C34303    KEY_ENT1:    JP    MAIN1
0439 213320    DEL_FUNC:    LD    HL,DIR1
043C 3A0D20                 LD    A,(CURR_PROG)
043F 47                     LD    B,A
0440 B7                     OR    A
0441 2806                   JR    Z,DEL_MARK
0443 110800                 LD    DE,08H
0446 19        DEL_SEARCH:  ADD   HL,DE
0447 10FD                   DJNZ  DEL_SEARCH
0449 3E00      DEL_MARK:    LD    A,00H
044B 77                     LD    (HL),A           ;MARK EMTY
044C 3E0C                   LD    A,0CH            ;DIR TO CURRENT
044E D7                     RST   10H
044F 3E08                   LD    A,08H            ;DISPLAY CURRENT
0451 D7                     RST   10H
0452 CD2503                 CALL  DISP_PROG
0455 C34303                 JP    MAIN1
0458 213320    LOAD_FUNC:   LD    HL,DIR1
045B 3A0D20                 LD    A,(CURR_PROG)
045E 47                     LD    B,A
045F B7                     OR    A
```

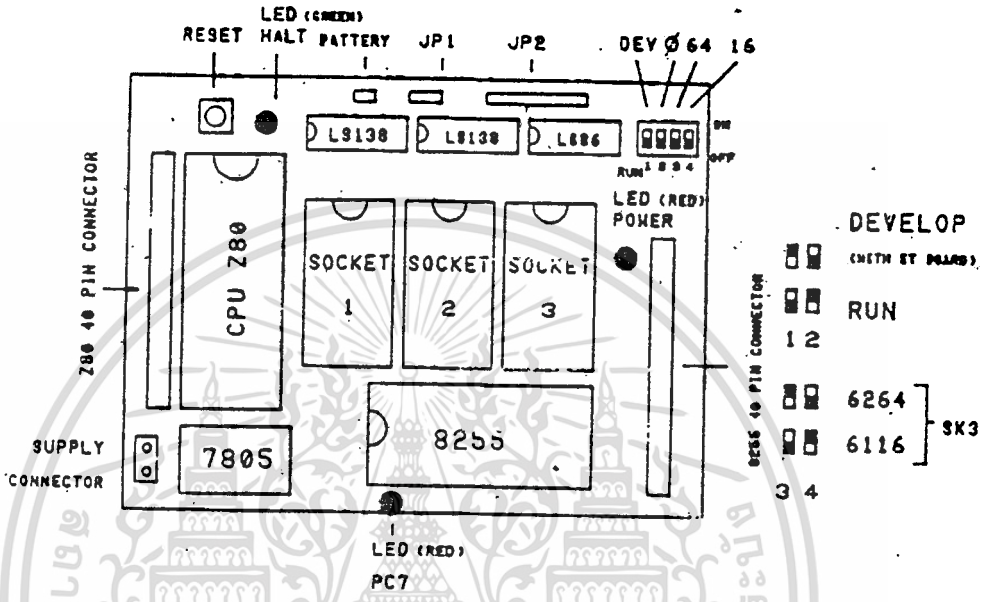
```
0460 2806                JR     Z,LOAD_PROG
0462 110800              LD     DE,08H
0465 19        LOAD_SEARCH:  ADD    HL,DE
0466 10FD                DJNZ   LOAD_SEARCH
0468 110400    LOAD_PROG:    LD     DE,04H          ;ADDR BUFFER
0468 19                  ADD    HL,DE
046C 5E                  LD     E,(HL)
046D 23                  INC    HL
046E 56                  LD     D,(HL)
046F 010000              LD     BC,00H
0472 E5                  PUSH   HL
0473 C5                  PUSH   BC
0474 D5                  PUSH   DE
0475 0E08                LD     C,I8MDAT        ;READ DATA FROM I8M
0477 3E0E                LD     A,0EH
0479 D7                  RST    10H
047A 7A                  LD     A,D             ;SAVE DATA
0478 D1                  POP    DE
047C 12                  LD     (DE),A
047D 13                  INC    DE              ;ADDR BUFFER+1
047E C1                  POP    BC
047F 03                  INC    BC              ;COUNT+1
0480 E1                  POP    HL

                ;-------------------------------------------------
0481 C33803    PHI_IRNF:     JP     MAIN
                ;-------------------------------------------------

                        ; ROM DATA & TABLE AREA

                ;------------ KEY TABLE -----------------------------

0484 28292A2B  KEY_TAB:      DFB    28H,29H,2AH,28H
0488 2C2D2E2F                DFB    2CH,2DH,2EH,2FH
048C 18191A1B                DFB    18H,19H,1AH,18H
0490 1C1D1E1F                DFB    1CH,1DH,1EH,1FH

                ;------------ PHICOM PORT TABLE ---------------------

0494 10121416  PHIP_TAB:     DFB    10H,12H,14H,16H
0498 20222426                DFB    20H,22H,24H,26H
049C 30323436                DFB    30H,32H,34H,36H
04A0 40424346                DFB    40H,42H,43H,46H
04A4 50525456                DFB    50H,52H,54H,56H
04A8 60626466                DFB    60H,62H,64H,66H
04AC 70727476                DFB    70H,72H,74H,76H

                ;-------------PHICOM CONTROL PORT -------------------

04B0 11131517  PHIC_TAB:     DFB    11H,13H,15H,17H
04B4 21232522                DFB    21H,23H,25H,22H
04B8 31333537                DFB    31H,33H,35H,37H
04BC 41434547                DFB    41H,43H,45H,47H
04C0 51535557                DFB    51H,53H,55H,57H
04C4 61636567                DFB    61H,63H,65H,67H
04C8 71737577                DFB    71H,73H,75H,77H
```

```
                ;---------------- DISPLAY TABLE ----------------------

04CC 3F605B4F  NUM_TAB:       DFB    3FH,60H,58H,4FH          ;0 1 2 3
04D0 666D7D07                 DFB    66H,6DH,7DH,07H          ;4 5 6 7
04D4 7F6F0000                 DFB    7FH,6FH,00H,00H          ;8 9 0 0
04D8 00000000                 DFB    00H,00H,00H,00H          ;0 0 0 0
04DC 07007940  ESHOP_TAB:     DFB    07H,00H,79H,40H          ;E-SHOP
04E0 6D763F73                 DFB    6DH,76H,3FH,73H
04E4 06005079  RDY_TAB:       DFB    06H,00H,50H,79H
04E8 775E6E                   DFB    77H,5FH,6EH              ;READY
04EB 06007950  ERR_TAB:       DFB    06H,00H,79H,50H
04EF 505C50                   DFB    50H,5CH,50H              ;ERRER
04F2 0477505C  FRM_TAB:       DFB    04H,77H,50H,5CH
04F6 55                       DFB    55H                      ;FROM
04F7 02785C    TO_TAB:        DFB    02H,78H,5CH              ;TO
04FA 035F7938  DEL_TAB:       DFB    03H,5FH,79H,38H          ;DEL.
04FE 03385CDE  LD_TAB:        DFB    03H,38H,5CH,0DEH         ;LOD.
0502 033976FA  CHK_TAB:       DFB    03H,39H,76H,0FAH         ;CHK.
0506 03607C55  IBM_TAB:       DFB    03H,60H,7CH,55H          ;IBM
050A 03737660  PHI_TAB:       DFB    03H,73H,76H,60H          ;PHI

                ;---------------- BUFFER TABEL ----------------------

050E 00283333  BUFF_TAB:      DWL    BUFFER1,BUFFER2
0512 663E9949                 DWL    BUFFER3,BUFFER4
0516 CC54                     DWL    BUFFER5

                ;---------------- CALL TABEL ------------------------

0518 0000      CAL_TAB:       DWL    00H                      ;00
051A 45015901                 DWL    DISP_NUM,DISP_FUNC       ;01,02
051E 6A017801                 DWL    SAVE_NUM,CHK_KEY         ;03,04
0522 9101AF01                 DWL    SCAND,CLR_DISP           ;05,06
0526 8A01E801                 DWL    SCANK,SCANP              ;07,08
052A 14023202                 DWL    COMPQ,CHK_BUFF           ;09,0A
052E 4D025F02                 DWL    DISP_CURR,DIRTOCURR      ;0B,0C
0532 85029C02                 DWL    RD_HEAD,WR_HEAD          ;0D,0E
0536 FC020203                 DWL    TXSET,RXSET              ;0F,10

                ;---------------- RAM AREA  16 K --------------------

2000                          ORG    2000H

2000            DISP_BUFF:    DFS    8            ;DISPLAY BUFFER (8 BYTE)
2008            PORT_BUFF:    DFS    1            ;PHICOM CALL BUFFER
2009            KEY_MARK:     DFS    1            ;KEY MARK
200A            KEY_IN:       DFS    1            ;KEY PRESS BUFFER
200B            KEY_BUFF:     DFS    1            ;KEY BUFFER
200C            FUNC_NEW:     DFS    1            ;FUNCTION KEY BUFFER
200D            CURR_PROG:    DFS    1            ;CURRENT PROGRAM
200E            NUM_BUFF:     DFS    3            ;NUMBER BUFFER (3 BYTE)
2011            FUNC_OLD:     DFS    1            ;FUNCTION OLD BUFFER
2012            STEP:         DFS    1            ;STEP COUNTER

                ;*---------- FILES CONTROL BLOCK 7 BYTE ----------*
```

```
0007 A                  0000 B                  2800 BUFFER1
3333 BUFFER2            3E66 BUFFER3            4999 BUFFER4
64CC BUFFER5            050E BUFF_TAB           0001 C
0518 CAL_TAB            0232 CHK_BUFF           0240 CHK_BUFF1
0249 CHK_BUFF2          026F CHK_DIR            0178 CHK_KEY
018A CHK_KEY1           018F CHK_KEY2           0502 CHK_TAB
0185 CLEAR1             01AF CLR_DISP           202F COMM_BUFF
021E COMP1              0214 COMP0              0003 CTRL
200D CURR_PROG          0002 D                  0439 DEL_FUNC
0449 DEL_MARK           0446 DEL_SEARCH         04FA DEL_TAB
0002 DIGIT              2033 DIR1               2039 DIR2
203F DIR3               2045 DIR4               204B DIR5
025F DIRTOCURR          2000 DISP_BUFF          024D DISP_CURR
0159 DISP_FUNC          0162 DISP_FUNC1         0145 DISP_NUM
0325 DISP_PROG          032A DISP_PROG1         0003 E
027E EMTY               0280 EMTY1              04E8 ERR_TAB
04DC ESHOP_TAB          2013 FCB                030E FCB_SET
04F2 FRM_TAB            038F FUNC_KEY           200C FUNC_NEW
2011 FUNC_OLD           0004 H                  0009 I8MCTRL
0008 I8MDAT             0506 I8M_TAB            0100 INIT
0121 INIT1              200B KEY_BUFF           03DF KEY_CHK
03A5 KEY_DEL            0426 KEY_ENT            0436 KEY_ENT1
03FC KEY_ESC            200A KEY_IN             03C2 KEY_LD
2009 KEY_MARK           0405 KEY_NEXT           0419 KEY_NEXT1
0423 KEY_NEXT2          0484 KEY_TAB            0005 L
04FE LD_TAB             0458 LOAD_FUNC          0468 LOAD_PROG
0465 LOAD_SEARCH        0338 MAIN               0343 MAIN1
034E MAIN2              0360 MAIN3              200E NUM_BUFF
0372 NUM_KEY            04CC NUM_TAB            0280 OUT_PNUM
0285 OUT_PNUM1          028F OUT_PNUM2          0000 PHICOM
0480 PHIC_TAB           0494 PHIP_TAB           050A PHI_TAB
0481 PHI_TRNF           2008 PORT_BUFF          0000 POWER
0001 POWER1             2018 QUE_BUFF           201A QUE_CONT
04E4 RDY_TAB            0285 RD_HEAD            02E2 RXBYTE
02E3 RXBYTE1            02EC RXBYTE2            02F9 RXBYTE3
0302 RXSET              0307 RXSET1             016A SAVE_NUM
0191 SCAND              0197 SCAND1             01BA SCANK
01C5 SCANK1             01D9 SCANK11            01DF SCANK12
01E6 SCANK2             01EB SCANP              01F3 SCANP1
01F8 SCANP2             0210 SCANP3             01A1 SCANS
01A8 SCANS1             026C SEARCH_DIR         0001 SEGM
0004 SLOTCHK            27FE STACK              27E0 STACK_AREA
2012 STEP               012F SYSCAL             013A TABLE
```

```
02D2  TX8YTE2           02DF  TX8YTE3           02FC  TXSET
000C  WATDOG            029C  WR_HEAD
```

ภาพแสดงลักษณะ Z80 CP-A



Socket 1 ใช้กับเบอร์ 2732,2764,6264

Socket 2 ใช้กับเบอร์ 6264 โดยจะสามารถ Backup ข้อมูลได้ ถ้าต่อ Battery

Socket 3 ใช้กับเบอร์ 6116,6264 โดยเลือกได้จาก Dip Switch ตัวที่ 3,4

JP 1 สำหรับเลือกสถานะของ Socket 1 โดยถ้าต่อกับ WR (62) จะใช้กับ 6264 และ 2732 แต่ถ้าต่อกับ Vcc (27) จะใช้กับ 2764

JP 2 สำหรับเลือกเบอร์ Port ของ 8255

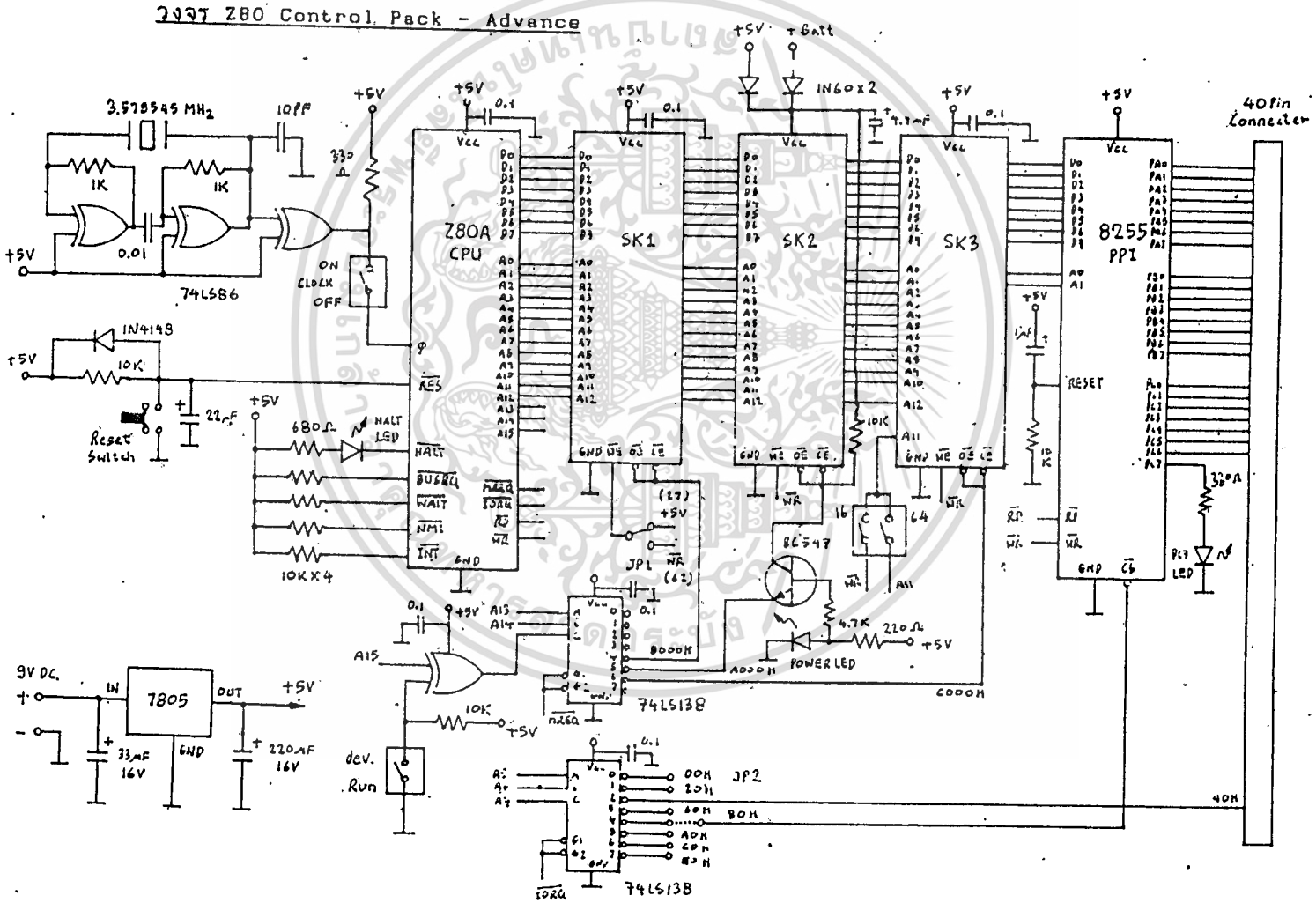
Dip 1 สำหรับเลือกโหมดในการใช้งาน Dev. คือโหมดในขณะทำการพัฒนา Run คือโหมดสำหรับการใช้งาน

Dip 2 สำหรับตัดหรือต่อความถี่ Clock ปกติจะใช้ร่วมกับตัวที่ 1 คือถ้าขณะทำการพัฒนาจะ Off ไว้

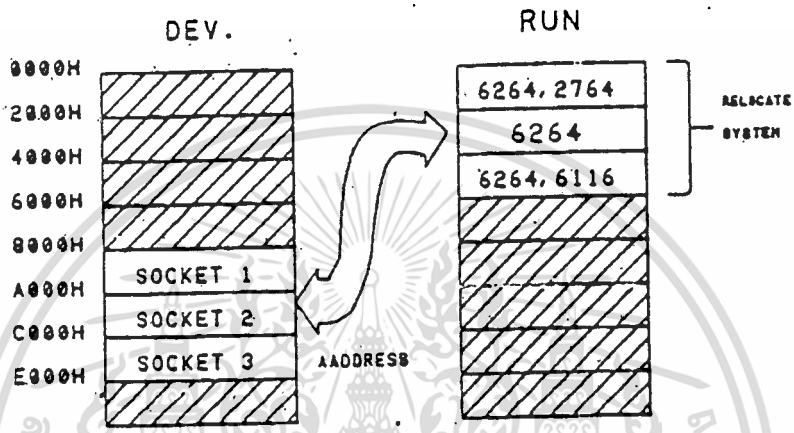
Dip 3 เลือก Socket 3 ให้ใช้กับ 6264

Dip 4 เลือก Socket 3 ให้ใช้กับ 6116

(ตัวที่ 3,4 นี้จะ On ตัวใดตัวหนึ่งเท่านั้น)


วงจร Z80 Control Pack - Advance
3.579545 MH2
10PF
1K
1K
0.01
74LS86
+5V
330 Ω
ON CLOCK OFF
Z80A CPU
SK1
SK2
SK3
8255 PPI
IN60x2
+Batt
4.7mF
0.1
40Pin Connector
IN4148
10K
Reset Switch
22mF
680Ω
HALT LED
10K x4
74LS138
JP1
JP2
8000H
A000H
C000H
E000H
8C547
4.7K
220Ω
POWER LED
9V DC.
7805
33mF 16V
220mF 16V
dev. Run
40H
80H

รายละเอียดการจัดหน่วยความจำ และ Port



การจัดหน่วยความจำของ CP-A นี้จะอ้าง Address เริ่มต้นที่ 8000H เพื่อให้ทำโปรแกรมได้สะดวกจากเครื่องอีทีบอร์ด และเมื่อจะนำโปรแกรมไปใช้งาน CP-A มีระบบ Relocate เพื่อให้หน่วยความจำไปเริ่มต้นที่ 0000H ได้ (ในโหมด Run) เพื่อให้เป็น Address ที่ทำงานเมื่อเริ่มเปิดเครื่องหรือกด Reset

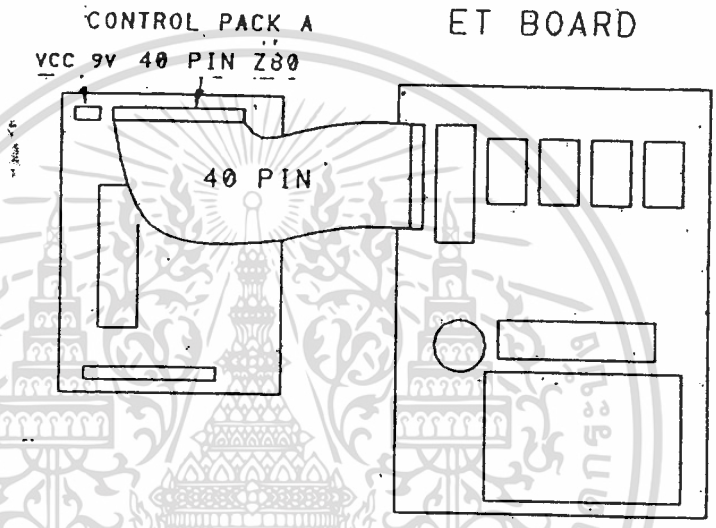
ระบบ Port จะทำการ Decode ไว้ทั้งหมด 8 ช่วง และจะสามารถเลือกเบอร์ได้ 7 ตำแหน่ง โดยการ Jump สายบนแผ่นปริ๊น ซึ่งมีเบอร์ Port ดังนี้

ตำแหน่ง	เบอร์ Port	
1	00 - 1F	(จริงๆใช้เพียง 00-03)
2	20 - 3F	
3	60 - 7F	
4	80 - 9F	
5	A0 - BF	
6	C0 - DF	
7	E0 - FF	

ส่วนเบอร์ Port 40 - 5F นั้น ไว้สำหรับการขยายได้อีกทางด้าน 8255 Connector

การใช้งานร่วมกับอีทีบอร์ด

Z80 CP-A มีโหมดในการทำงาน 2 ลักษณะคือ การพัฒนาและการนำไปใช้งาน ในขั้นของการพัฒนานั้นจะทำได้โดยเสียบเข้ากับอีทีบอร์ดได้เลย หรือจะทำโดยใช้เครื่องอื่นๆก็ได้ แล้วจึงนำโปรแกรมมาทำงานโดยผ่าน Eprom หรือ Ram-pack ซึ่งผู้ใช้จะต้องเขียนโปรแกรมโดยอ้าง Address ตามลักษณะที่กำหนดไว้ ในการเสียบกับอีทีบอร์ดนั้นจะแสดงได้ดังภาพต่อไปนี้



ซึ่งพอจะสรุปขั้นตอนได้ดังนี้

1. ทำการปรับ Dip switch ตัวที่ 1 ให้อยู่ในตำแหน่ง Dev. และตัวที่ 2 ให้อยู่ในตำแหน่ง Off ซึ่งเป็นการตัดความถี่ Clock ให้เหลือเพียง Clock จากอีทีบอร์ด
2. ผู้ใช้จะต้องเอา CPU Z80 ตัวใดตัวหนึ่งออกจากบอร์ด คือบนบอร์ด CP-A หรือบนอีทีบอร์ดก็ได้ เพื่อที่จะได้ไม่ซ้อนกัน
3. แหล่งจ่ายไฟ 5V ของทั้งสองบอร์ดจะต่อถึงกันหมด และเนื่องจากใช้กระแสมากขึ้น ผู้ใช้ควรจะใช้ Adapter ที่มีขนาดใหญ่พอสมควร (ของที่มากับเครื่องอีทีบอร์ดจะไม่พอ) และก็เสียบกับบอร์ดใดบอร์ดหนึ่งได้เลย ทั้งนี้ถ้ามีการต่อ Port ออกไปใช้งาน ก็ต้องดูด้วยว่ากระแสจ่ายได้เพียงพอหรือไม่ ถ้าไม่พอก็ควรแยกแหล่งจ่ายไฟออกเป็นอีกชุดหนึ่ง

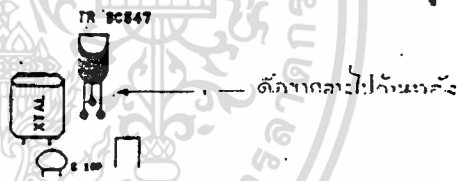
ในโหมด Dev นี้ ผู้ใช้จะเห็น Address ในตำแหน่ง 8000H A000H C000H ตามลำดับ และถ้านำ Ram 6264 มาเสียบที่ Socket 1 ก็จะสามารถทำโปรแกรมได้เลย โดยใช้การทำงานต่างๆทั้งหมดของอีทีบอร์ด การนำ Ram 6264 มาเสียบ จะต้องเปลี่ยน JP1 ให้อยู่ในตำแหน่ง 62 ด้วย ซึ่งจะต้องตัดลายปริ้นที่ Default ไว้ที่ตำแหน่ง 27 ก่อน แล้วจึงต่อลายใหม่

การนำไปใช้งาน

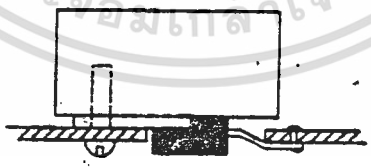
1. ทำการปรับ Dip switch ตัวที่ 1 ให้อยู่ในตำแหน่ง Run และตัวที่ 2 ให้อยู่ในตำแหน่ง On
2. ใส่ตัว CPU Z80 ลงใน Socket ให้เรียบร้อย
3. นำโปรแกรมที่พัฒนาเรียบร้อยแล้วอัดลงในตัว Eprom โดยใช้เครื่อง BT Burner หรือจะใช้เครื่องอื่นๆก็ได้ โดยโปรแกรมนี้จะต้องแปลงให้ทำงานที่ Address 0000H จากนั้นก็นำมาใส่ลงบนบอร์ดที่ Socket 1
4. ถ้ามีการแก้ไขที่ JP1 ในขณะที่พัฒนา ก็ต้องเปลี่ยนกลับเป็นตำแหน่ง 27 เหมือนเดิม

คำแนะนำในการประกอบ

1. ควรใส่อุปกรณ์ตามลำดับคือ ตัวต้านทาน, ไดโอด, ตัวเก็บประจุ, Socket
2. ตัวเก็บประจุและไดโอดต้องใส่ให้ถูกขั้ว รวมทั้งทรานซิสเตอร์ต้องใส่ขาให้ถูก โดยให้ดูจากรูปดังนี้

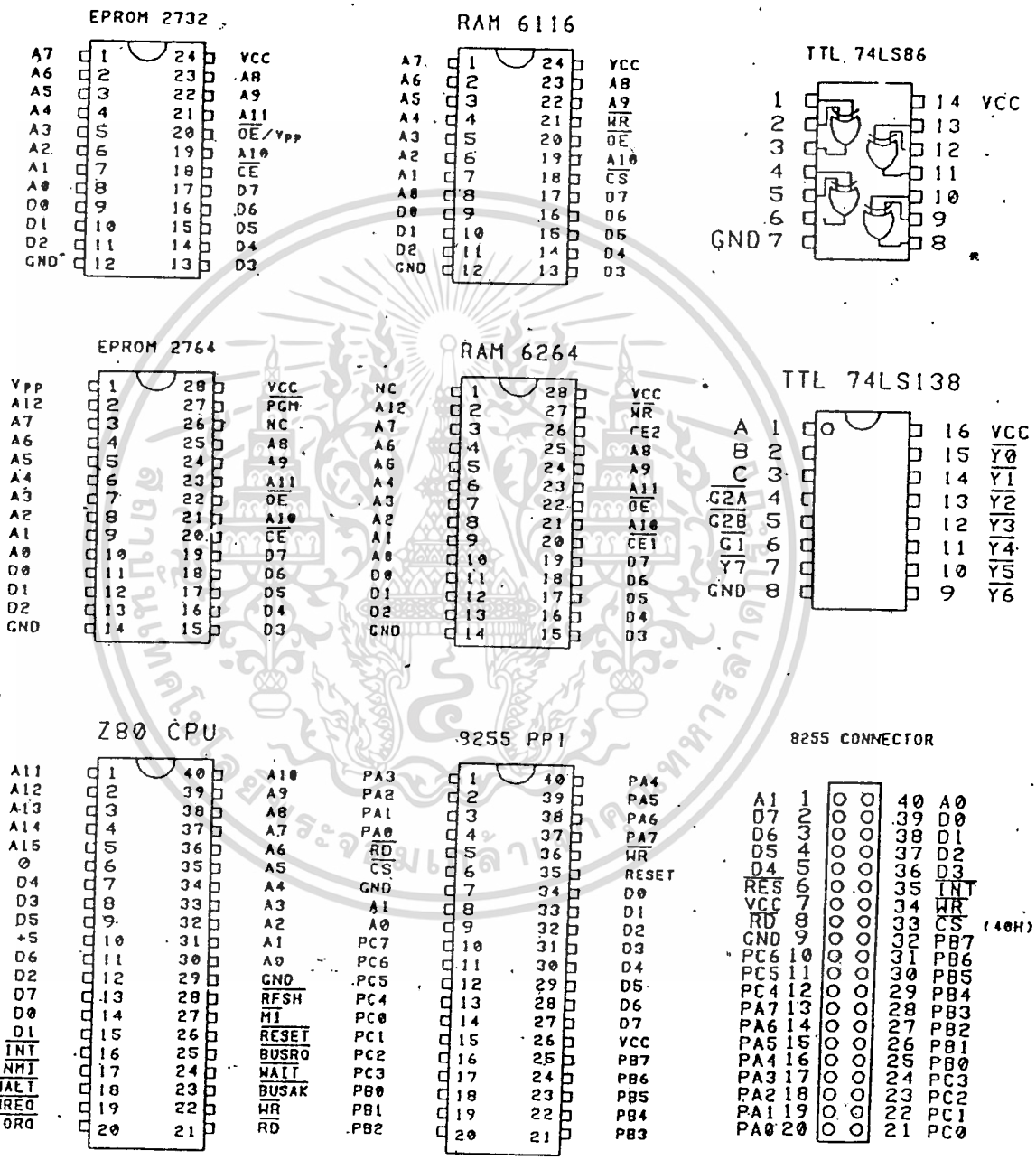


3. LED ขายาวคือ A ขาสั้นคือ K
4. การบัดกรีให้กรุยทำเฉพาะด้านล่างเพียงด้านเดียว ด้านบนไม่ต้องบัดกรีเพราะแผ่นปริ้นเป็นแบบ Plate Through Holes รวมทั้งรูว่างๆก็ไม่ต้องบัดกรี
5. การติดตั้งและบัดกรี Regulator IC ให้ดูดังรูป



6. เมื่อเรียบร้อยแล้วจึงใส่ IC ลงใน Socket โดยต้องระวังไม่ให้ผิดเบอร์ด้วย

รายละเอียดขา IC และ Connector



รายการอุปกรณ์

ชุด IC

1 ตัว	Z-80A CPU	1 ตัว	8255 I/O port
1 ตัว	6264 Ram	1 ตัว	74LS86
2 ตัว	74LS138		

ชุด Socket

2 ตัว	40-Pins Socket	3 ตัว	28-Pins Socket
3 ตัว	16-Pins Socket		

ชุด Resistor

8 ตัว	10 K	2 ตัว	1 K
1 ตัว	4.7 K	1 ตัว	220
2 ตัว	330	1 ตัว	680

ชุด Condensor

1 ตัว	220 uf ELE	1 ตัว	33 uf ELE
1 ตัว	22 uf ELE	1 ตัว	4.7 uf TAN
1 ตัว	1 uf TAN	6 ตัว	0.1 uf TAN
1 ตัว	0.01 uf ไมล่าร์	1 ตัว	10 pf CER

ชุดเบ็ดเตล็ด

3 ตัว	LED (แดง 2 ตัว , เขียว 1 ตัว)		
2 ตัว	1N60	1 ตัว	1N4148
1 ตัว	7805	1 ตัว	Head Sink พร้อมน็อต
2 ตัว	Connector 40 Pin	1 ชุด	ขั้วต่อ Supply
1 ตัว	Dip Switch (4*)	1 ตัว	Reset Switch
1 ตัว	X'TAL 3.579545 MHz	1 ตัว	BC 547

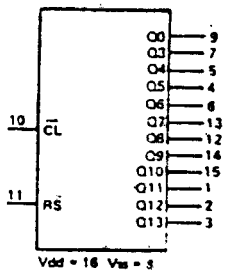
อื่นๆ

1 แผ่น	CP-A PCB
1 เล่ม	ชีทรายละเอียดการใช้งาน

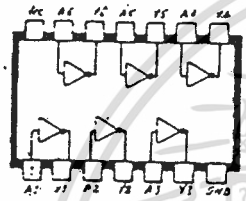
Specification

CPU	Z-80A
Memory	6264/2764/2732 Socket (8 Kbyte at 8000H)
	6264 Ram (8 Kbyte at A000H) & Backup Circuit
	6264/6116 Socket (8 Kbyte at C000H)
Port	8255 I/O Port (8*3 bit)
LED	1 Power Red LED
	1 PC7 Port Red LED
	1 Halt Green LED
Diode	2 IN60
Transistor	1 BC547
TTL IC	3 74LS86 74LS138*2
Regulator IC	1 7805
Connector	1 40-Pin Expansion Header-Strip (Z80 Pin)
	1 40-Pin Peripheral Header-Strip (8255 Port)
	1 2-Pin For DC 9V Supply
Switch	4 Dip-Switch
	(Run/Develope, Clock-On, 6116, 6264 Select)
Clock Rate	3.579545 MHz
Power Supply	Comsumption 5V DC
	Main Input 9-12V DC
PCB Size	8.5 * 12 cm.

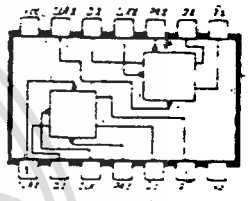
4020 14 Stage Binary Counter



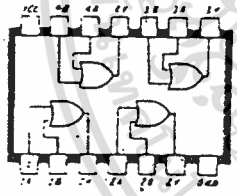
7404 Hex Inverter



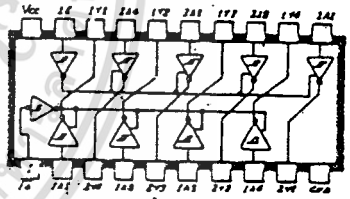
7474 Dual D, Pos Edge Trig F/F , Pre-S & Clear



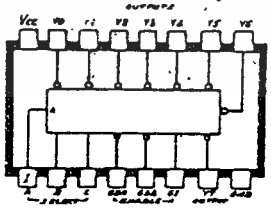
7432 Quad 2-Input OR Gates



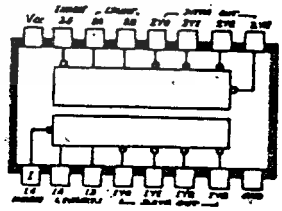
74240 Octal Buffer/Line Driver, 3 State Output



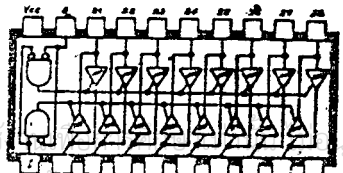
74138 1-of-8 Decoder Multiplexer



74139 Dual 1-of-4 Decoder



74245 Octal Bus Transceiver, 3 State Output



MOTOROLA

MC1488

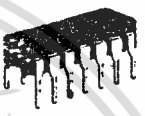
QUAD LINE DRIVER

The MC1488 is a monolithic quad line driver designed to interface data terminal equipment with data communications equipment in conformance with the specifications of EIA Standard No. RS-232C.

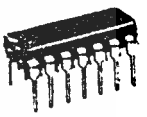
Features:

- Current Limited Output
  ±10 mA typ
- Power-Off Source Impedance
  300 Ohms min
- Simple Slew Rate Control with External Capacitor
- Flexible Operating Supply Range
- Compatible with All Motorola MDTL and MTTL Logic Families

QUAD MDTL LINE DRIVER
RS-232C
SILICON MONOLITHIC
INTEGRATED CIRCUIT

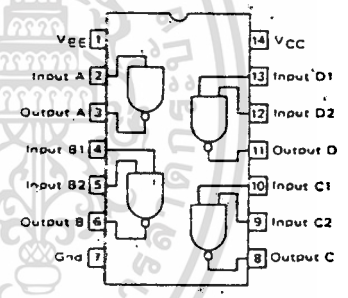


L SUFFIX
CERAMIC PACKAGE
CASE 632-02
MO-001AA

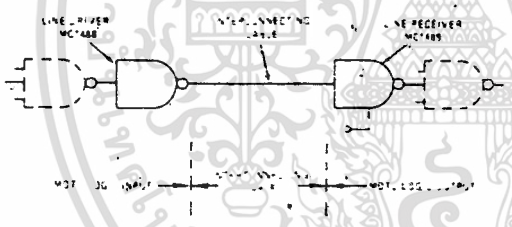


P SUFFIX
PLASTIC PACKAGE
CASE 646-05

PIN CONNECTIONS

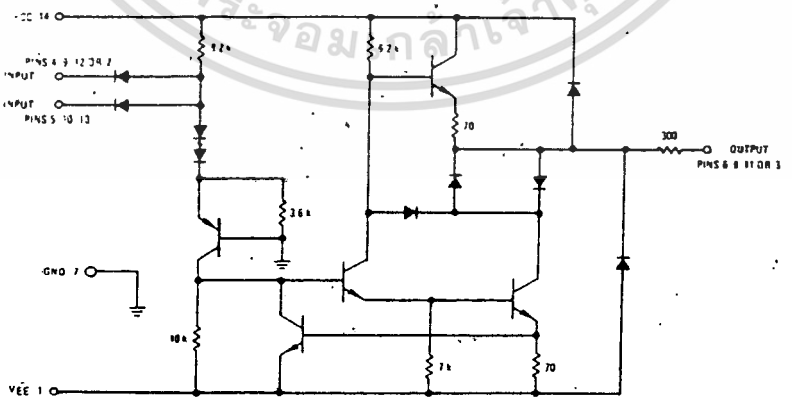


TYPICAL APPLICATION



CIRCUIT SCHEMATIC

(1/4 OF CIRCUIT SHOWN)



MOTOROLA

MC1489
MC1489A

QUAD LINE RECEIVERS

The MC1489 monolithic quad line receivers are designed to interface data terminal equipment with data communications equipment in conformance with the specifications of EIA Standard No. RS-232C.

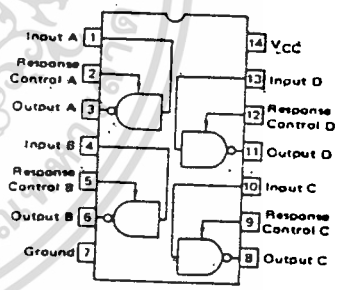
- Input Resistance – 3.0 k to 7.0 kilohms
- Input Signal Range – ± 30 Volts
- Input Threshold Hysteresis Built In
- Response Control
  - a) Logic Threshold Shifting
  - b) Input Noise Filtering

QUAD MDTL
LINE RECEIVERS
RS-232C

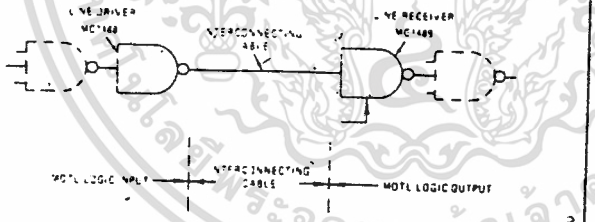
SILICON MONOLITHIC
INTEGRATED CIRCUIT

L SUFFIX
CERAMIC PACKAGE
CASE 632-02
MO-001AA

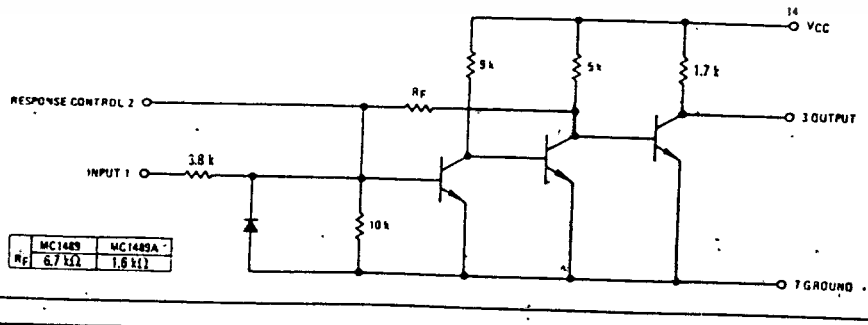
P SUFFIX
PLASTIC PACKAGE
CASE 646-05



TYPICAL APPLICATION



EQUIVALENT CIRCUIT SCHEMATIC (1/4 OF CIRCUIT SHOWN)



	MC1489	MC1489A
R_F	6.7 kΩ	1.6 kΩ

intel

8251A
PROGRAMMABLE COMMUNICATION INTERFACE

- Synchronous and Asynchronous Operation
- Synchronous 5-8 Bit Characters; Internal or External Character Synchronization; Automatic Sync Insertion
- Asynchronous 5-8 Bit Characters; Clock Rate—1, 16 or 64 Times Baud Rate; Break Character Generation; 1, 1½, or 2 Stop Bits; False Start Bit Detection; Automatic Break Detect and Handling
- Synchronous Baud Rate—DC to 64K Baud
- Asynchronous Baud Rate—DC to 19.2K Baud
- Full-Duplex, Double-Buffered Transmitter and Receiver
- Error Detection—Parity, Overrun and Framing
- Compatible with an Extended Range of Intel Microprocessors
- 28-Pin DIP Package
- All Inputs and Outputs are TTL Compatible
- Available in EXPRESS
  —Standard Temperature Range
  —Extended Temperature Range

The Intel® 8251A is the enhanced version of the industry standard, Intel 8251 Universal Synchronous/Asynchronous Receiver/Transmitter (USART), designed for data communications with Intel's microprocessor families such as MCS-48, 80, 85, and iAPX-86, 88. The 8251A is used as a peripheral device and is programmed by the CPU to operate using virtually any serial data transmission technique presently in use (including IBM "bi-sync"). The USART accepts data characters from the CPU in parallel format and then converts them into a continuous serial data stream for transmission. Simultaneously, it can receive serial data streams and convert them into parallel data characters for the CPU. The USART will signal the CPU whenever it can accept a new character for transmission or whenever it has received a character for the CPU. The CPU can read the complete status of the USART at any time. These include data transmission errors and control signals such as SYNDET, TxEMPTY. The chip is fabricated using N-channel silicon gate technology.

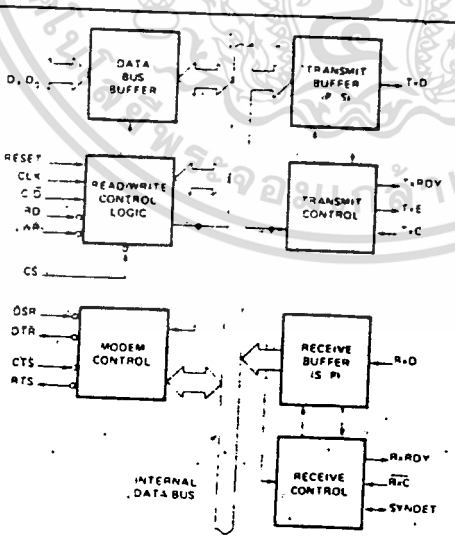


Figure 1. Block Diagram

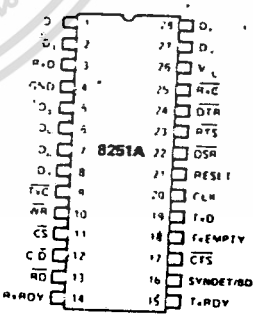


Figure 2. Pin Configuration

September 1984
205222-002

FEATURES AND ENHANCEMENTS

The 8251A is an advanced design of the industry standard USART, the Intel® 8251. The 8251A operates with an extended range of Intel microprocessors and maintains compatibility with the 8251. Familiarization time is minimal because of compatibility and involves only knowing the additional features and enhancements, and reviewing the AC and DC specifications of the 8251A.

The 8251A incorporates all the key features of the 8251 and has the following additional features and enhancements:

- 8251A has double-buffered data paths with separate I/O registers for control, status, Data In, and Data Out, which considerably simplifies control programming and minimizes CPU overhead.
- In asynchronous operations, the Receiver detects and handles "break" automatically, relieving the CPU of this task.
- A refined Rx initialization prevents the Receiver from starting when in "break" state, preventing unwanted interrupts from a disconnected USART.
- At the conclusion of a transmission, TxD line will always return to the marking state unless SBRK is programmed.
- Tx Enable logic enhancement prevents a Tx Disable command from halting transmission until all data previously written has been transmitted. The logic also prevents the transmitter from turning off in the middle of a word.
- When External Sync Detect is programmed, Internal Sync Detect is disabled, and an External Sync Detect status is provided via a flip-flop which clears itself upon a status read.
- Possibility of false sync detect is minimized by ensuring that if double character sync is programmed, the characters be contiguously detected and also by clearing the Rx register to all ones whenever Enter Hunt command is issued in Sync mode.
- As long as the 8251A is not selected, the $\overline{RD}$ and $\overline{WR}$ do not affect the internal operation of the device.
- The 8251A Status can be read at any time but the status update will be inhibited during status read.
- The 8251A is free from extraneous glitches and has enhanced AC and DC characteristics, providing higher speed and better operating margins.
- Synchronous Baud rate from DC to 64K.

FUNCTIONAL DESCRIPTION

General

The 8251A is a Universal Synchronous/Asynchronous Receiver/Transmitter designed for a wide range of Intel microcomputers such as 8048, 8080, 8085, 8086 and 8088. Like other I/O devices in a microcomputer system, its functional configuration is programmed by the system's software for maximum flexibility. The 8251A can support most serial data techniques in use, including IBM "bi-sync."

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

Data Bus Buffer

This 3-state, bidirectional, 8-bit buffer is used to interface the 8251A to the system Data Bus. Data is transmitted or received by the buffer upon execution of INput or OUTput instructions of the CPU. Control words, Command words and Status information are also transferred through the Data Bus Buffer. The Command Status, Data-In and Data-Out registers are separate, 8-bit registers communicating with the system bus through the Data Bus Buffer.

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

RESET (Reset)

A "high" on this input forces the 8251A into an "Idle" mode. The device will remain at "Idle" until a new set of control words is written into the 8251A to program its functional definition. Minimum RESET pulse width is 6 t_{CY} (clock must be running).

A command reset operation also puts the device into the "Idle" state.

CLK (Clock)

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

$\overline{WR}$ (Write)

A "low" on this input informs the 8251A that the CPU is writing data or control words to the 8251A.

$\overline{RD}$ (Read)

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

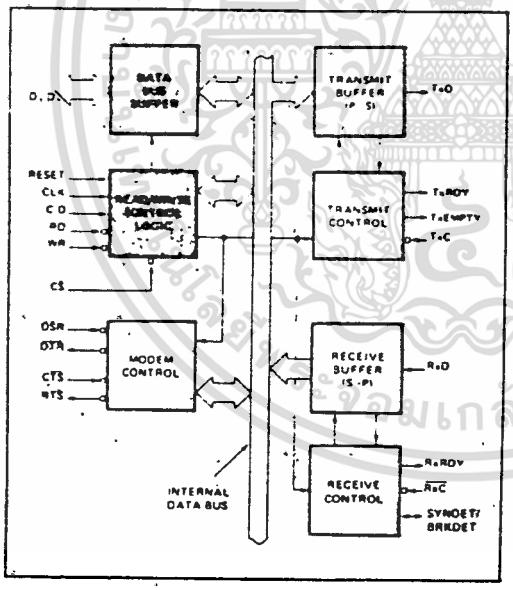


Figure 3. 8251A Block Diagram Showing Data Bus Buffer and Read/Write Logic Functions

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

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

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

1 = CONTROL/STATUS; 0 = DATA.

$\overline{CS}$ (Chip Select)

A "low" on this input selects the 8251A. No reading or writing will occur unless the device is selected. When $\overline{CS}$ is high, the Data Bus is in the float state and $\overline{RD}$ and $\overline{WR}$ have no effect on the chip.

Modem Control

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

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

The $\overline{DSR}$ input signal is a general-purpose, 1-bit inverting input port. Its condition can be tested by the CPU using a Status Read operation. The $\overline{DSR}$ input is normally used to test modem conditions such as Data Set Ready.

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

The $\overline{DTR}$ output signal is a general-purpose, 1-bit inverting output port. It can be set "low" by programming the appropriate bit in the Command Instruction word. The $\overline{DTR}$ output signal is normally used for modem control such as Data Terminal Ready.

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

The $\overline{RTS}$ output signal is a general-purpose, 1-bit inverting output port. It can be set "low" by programming the appropriate bit in the Command Instruction word. The $\overline{RTS}$ output signal is normally used for modem control such as Request to Send.

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

A "low" on this input enables the 8251A to transmit serial data if the Tx Enable bit in the Command byte is set to a "one." If either a Tx Enable off or $\overline{CTS}$ off condition occurs while the Tx is in operation, the Tx will transmit all the data in the USART, written prior to Tx Disable command before shutting down.

Transmitter Buffer

The Transmitter Buffer accepts parallel data from the Data Bus Buffer, converts it to a serial bit stream, inserts the appropriate characters or bits (based on the communication technique) and outputs a composite serial stream of data on the TxD output pin on the falling edge of $\overline{TxC}$. The transmitter will begin transmission upon being enabled if $\overline{CTS}$ = 0. The TxD line will be held in the marking state immediately upon a master Reset or when Tx Enable or $\overline{CTS}$ is off or the transmitter is empty.

Transmitter Control

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

TxRDY (Transmitter Ready)

This output signals the CPU that the transmitter is ready to accept a data character. The TxRDY output pin can be used as an interrupt to the system, since it is masked by TxEnable; or, for Polled operation, the CPU can check TxRDY using a Status Read operation. TxRDY is automatically reset by the leading edge of $\overline{WR}$ when a data character is loaded from the CPU.

Note that when using the Polled operation, the TxRDY status bit is *not* masked by TxEnable, but will only indicate the Empty/Full Status of the Tx Data Input Register.

TxE (Transmitter Empty)

When the 8251A has no characters to send, the TxEMPTY output will go "high." It resets upon receiving a character from CPU if the transmitter is enabled. TxEMPTY remains high when the transmitter is disabled. TxEMPTY can be used to indicate the end of a transmission mode, so that the CPU "knows" when to "turn the line around" in the half-duplex operational mode.

In the Synchronous mode, a "high" on this output indicates that a character has not been loaded and the SYNC character or characters are about to be or are being transmitted automatically as "fillers." TxEMPTY does not go low when the SYNC characters are being shifted out.

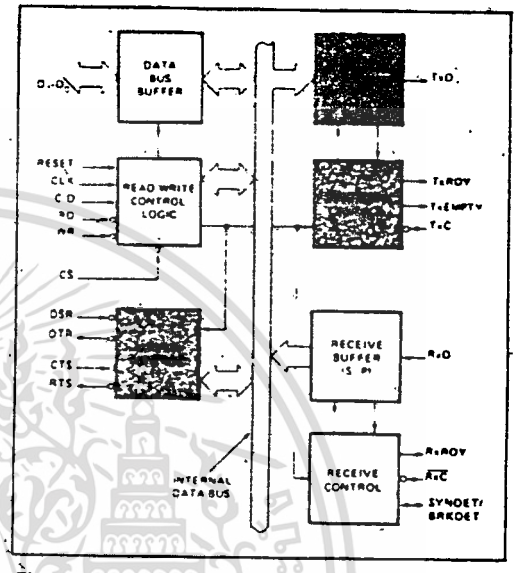


Figure 4. 8251A Block Diagram Showing Modem and Transmitter Buffer and Control Functions

$\overline{TxC}$ (Transmitter Clock)

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

For Example:

If Baud Rate equals 110 Baud,
$\overline{TxC}$ equals 110 Hz in the 1x mode.
$\overline{TxC}$ equals 1.72 kHz in the 16x mode.
$\overline{TxC}$ equals 7.04 kHz in the 64x mode.

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

Receiver Buffer

The Receiver accepts serial data, converts this serial input to parallel format, checks for bits or characters that are unique to the communication technique and sends an "assembled" character to the CPU. Serial data is input to RxD pin, and is clocked in on the rising edge of $\overline{RxC}$.

Receiver Control

This functional block manages all receiver-related activities which consists of the following features.

The RxD initialization circuit prevents the 8251A from mistaking an unused input line for an active low data line in the "break condition." Before starting to receive serial characters on the RxD line, a valid "1" must first be detected after a chip master Reset. Once this has been determined, a search for a valid low (Start bit) is enabled. This feature is only active in the asynchronous mode, and is only done once for each master Reset.

The False Start bit detection circuit prevents false starts due to a transient noise spike by first detecting the falling edge and then strobing the nominal center of the Start bit (RxD = low).

Parity error detection sets the corresponding status bit.

The Framing Error status bit is set if the Stop bit is absent at the end of the data byte (asynchronous mode).

RxRDY (Receiver Ready)

This output indicates that the 8251A contains a character that is ready to be input to the CPU. RxRDY can be connected to the interrupt structure of the CPU or, for polled operation, the CPU can check the condition of RxRDY using a Status Read operation.

RxEnable, when off, holds RxRDY in the Reset Condition. For Asynchronous mode, to set RxRDY, the Receiver must be enabled to sense a Start Bit and a complete character must be assembled and transferred to the Data Output Register. For Synchronous mode, to set RxRDY, the Receiver must be enabled and a character must finish assembly and be transferred to the Data Output Register.

Failure to read the received character from the Rx Data Output Register prior to the assembly of the next Rx Data character will set overrun condition error and the previous character will be written over and lost. If the Rx Data is being read by the CPU when the internal transfer is occurring, overrun error will be set and the old character will be lost.

$\overline{RxC}$ (Receiver Clock)

The Receiver Clock controls the rate at which the character is to be received. In Synchronous Mode, the Baud Rate (1x) is equal to the actual frequency of $\overline{RxC}$. In Asynchronous Mode, the Baud Rate is a fraction of the actual $\overline{RxC}$ frequency. A portion of the mode instruction selects this factor: 1, 1/16 or 1/64 the $\overline{RxC}$.

For example:

Baud Rate equals 300 Baud, if
$\overline{RxC}$ equals 300 Hz in the 1x mode;
$\overline{RxC}$ equals 4800 Hz in the 16x mode;
$\overline{RxC}$ equals 19.2 kHz in the 64x mode.

Baud Rate equals 2400 Baud, if
$\overline{RxC}$ equals 2400 Hz in the 1x mode;
$\overline{RxC}$ equals 38.4 kHz in the 16x mode;
$\overline{RxC}$ equals 153.6 kHz in the 64x mode.

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

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

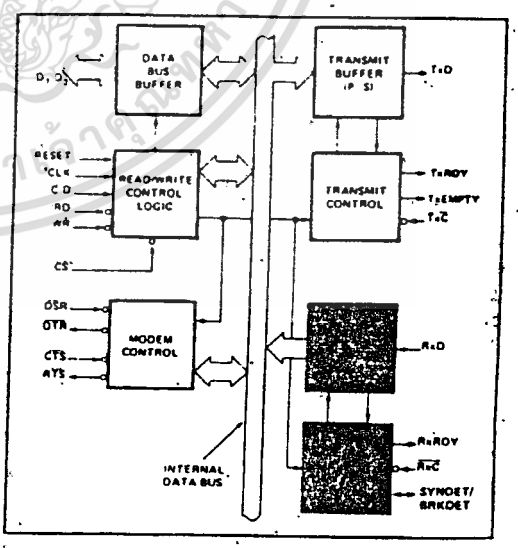


Figure 5. 8251A Block Diagram Showing Receiver Buffer and Control Functions

205222-002

SYNDET (SYNC Detect/ BRKDET Break Detect)

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

When used as an input (external SYNC detect mode), a positive going signal will cause the 8251A to start assembling data characters on the rising edge of the next $\overline{RxC}$. Once in SYNC, the "high" input signal can be removed. When External SYNC Detect is programmed, Internal SYNC Detect is disabled.

BREAK (Async Mode Only)

This output will go high whenever the receiver remains low through two consecutive stop bit sequences (including the start bits, data bits, and parity bits). Break Detect may also be read as a Status bit. It is reset only upon a master chip Reset or Rx Data returning to a "one" state.

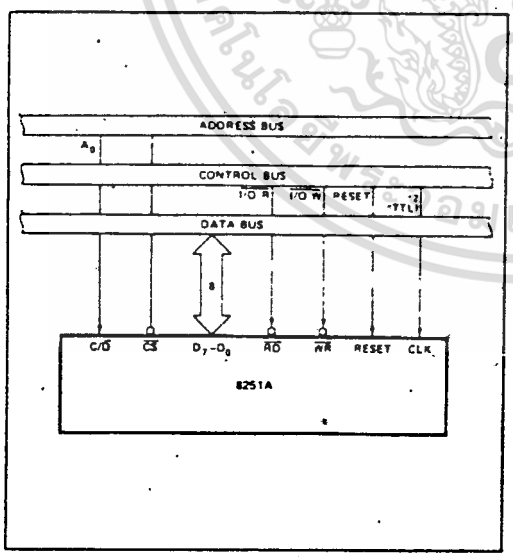


Figure 6. 8251A Interface to 8080 Standard System Bus

DETAILED OPERATION DESCRIPTION

General

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

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

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

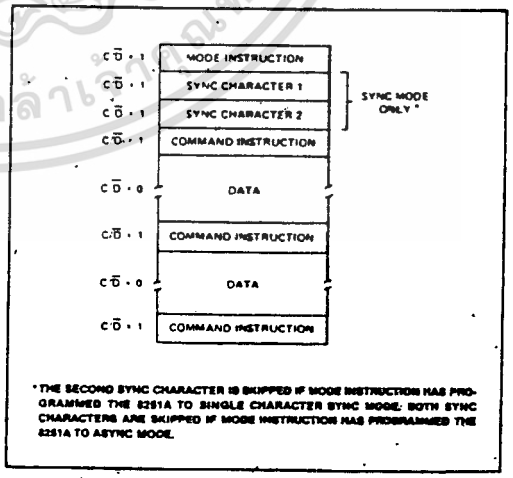


Figure 7. Typical Data Block

Programming the 8251A

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

The control words are split into two formats:

1. Mode Instruction
2. Command Instruction

Mode Instruction

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

Command Instruction

This instruction defines a word that is used to control the actual operation of the 8251A.

Both the Mode and Command Instructions must conform to a specified sequence for proper device operation (see Figure 7). The Mode Instruction must be written immediately following a Reset operation, prior to using the 8251A for data communication.

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

Mode Instruction Definition

The 8251A can be used for either Asynchronous or Synchronous data communication. To understand how the Mode Instruction defines the functional operation of the 8251A, the designer can best view the device as two separate components, one Asynchronous and the other Synchronous, sharing the same package. The format definition can be changed only after a master chip Reset. For explanation purposes the two formats will be isolated.

NOTE: When parity is enabled it is not considered as one of the data bits for the purpose of programming the word length. The actual parity bit received on the Rx Data line cannot be read on the Data Bus. In the case of a programmed character length of less than 8 bits, the least significant Data Bus bits will hold the data; unused bits are "don't care" when writing data to the 8251A, and will be "zeros" when reading the data from the 8251A.

Asynchronous Mode (Transmission)

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

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

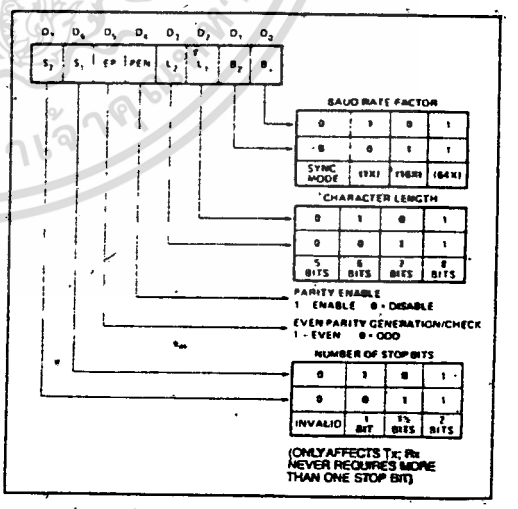


Figure 8. Mode Instruction Format, Asynchronous Mode

Asynchronous Mode (Receive)

The RxD line is normally high. A falling edge on this line triggers the beginning of a START bit. The validity of this START bit is checked by again strobing this bit at its nominal center (16X or 64X mode only). If a low is detected again, it is a valid START bit, and the bit counter will start counting. The bit counter thus locates the center of the data bits, the parity bit (if it exists) and the stop bits. If parity error occurs, the parity error flag is set. Data and parity bits are sampled on the RxD pin with the rising edge of $\overline{RxC}$. If a low level is detected as the STOP bit, the Framing Error flag will be set. The STOP bit signals the end of a character. Note that the *receiver* requires only *one* stop bit, regardless of the number of stop bits programmed. This character is then loaded into the parallel I/O buffer of the 8251A. The RxRDY pin is raised to signal the CPU that a character is ready to be fetched. If a previous character has not been fetched by the CPU, the present character replaces it in the I/O buffer, and the OVERRUN Error flag is raised (thus the previous character is lost). All of the error flags can be reset by an Error Reset Instruction. The occurrence of any of these errors will not affect the operation of the 8251A.

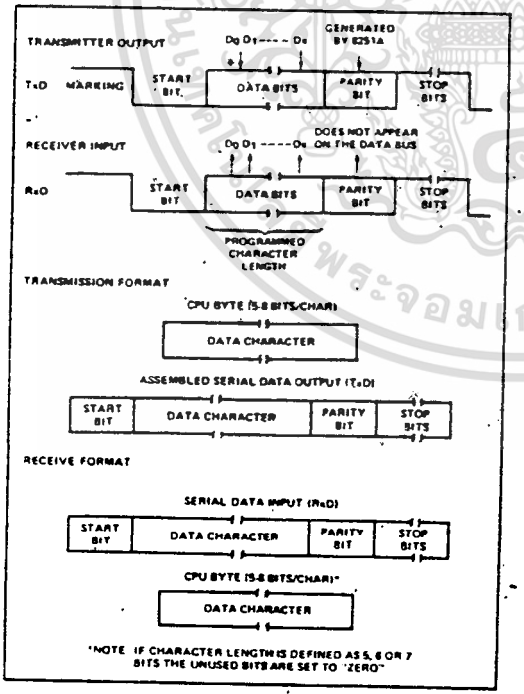
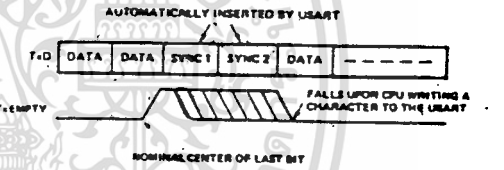


Figure 9. Asynchronous Mode

Synchronous Mode (Transmission)

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

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



Synchronous Mode (Receive)

In this mode, character synchronization can be internally or externally achieved. If the SYNC mode has been programmed, ENTER HUNT command should be included in the first command instruction word written. Data on the RxD pin is then sampled on the rising edge of $\overline{RxC}$. The content of the Rx buffer is compared at every bit boundary with the first SYNC character until a match occurs. If the 8251A has been programmed for two SYNC characters, the subsequent received character is also compared; when both SYNC characters have been detected, the USART ends the HUNT mode and is in character synchronization. The SYNDET pin is then set high, and is reset automatically by a STATUS READ. If parity is programmed, SYNDET will not be set until the middle of the parity bit instead of the middle of the last data bit.

In the external SYNC mode, synchronization is achieved by applying a high level on the SYNDET pin, thus forcing the 8251A out of the HUNT mode. The high level can be removed after one $\overline{RxC}$ cycle. An ENTER HUNT command has no effect in the asynchronous mode of operation.

Parity error and overrun error are both checked in the same way as in the Asynchronous Rx mode. Parity is checked when not in Hunt, regardless of whether the Receiver is enabled or not.

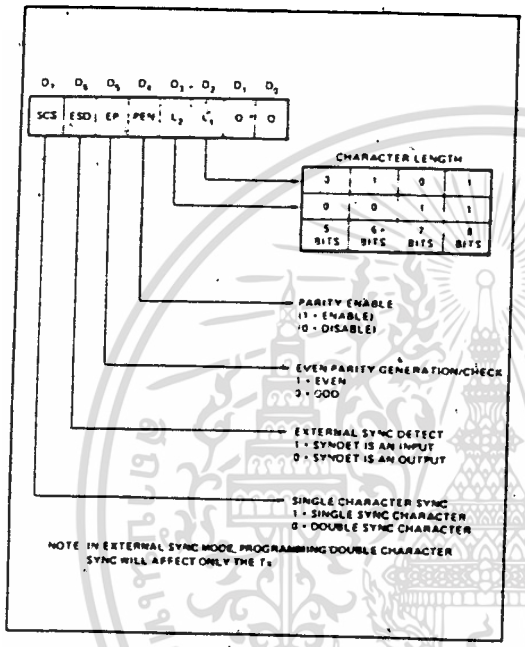


Figure 10. Mode Instruction Format, Synchronous Mode

The CPU can command the receiver to enter the HUNT mode if synchronization is lost. This will also set all the used character bits in the buffer to a "one," thus preventing a possible false SYNDET caused by data that happens to be in the Rx Buffer at ENTER HUNT time. Note that the SYNDET F/F is reset at each Status Read, regardless of whether internal or external SYNC has been programmed. This does not cause the 8251A to return to the HUNT mode. When in SYNC mode, but not in HUNT, Sync Detection is still functional, but only occurs at the "known" word boundaries. Thus, if one Status Read indicates SYNDET and a second Status Read also indicates SYNDET, then the programmed SYNDET characters have been received since the previous Status Read. (If double character sync has been programmed, then both sync characters have been contiguously received to gate a SYNDET indication.) When external SYNDET mode is selected, internal Sync Detect is disabled, and the SYNDET F/F may be set at any bit boundary.

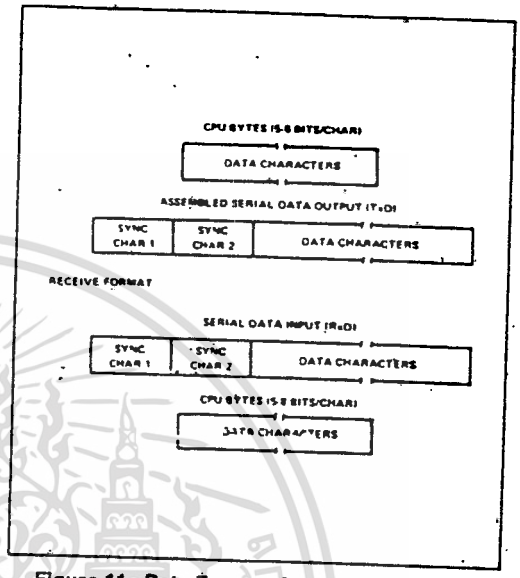


Figure 11. Data Format, Synchronous Mode

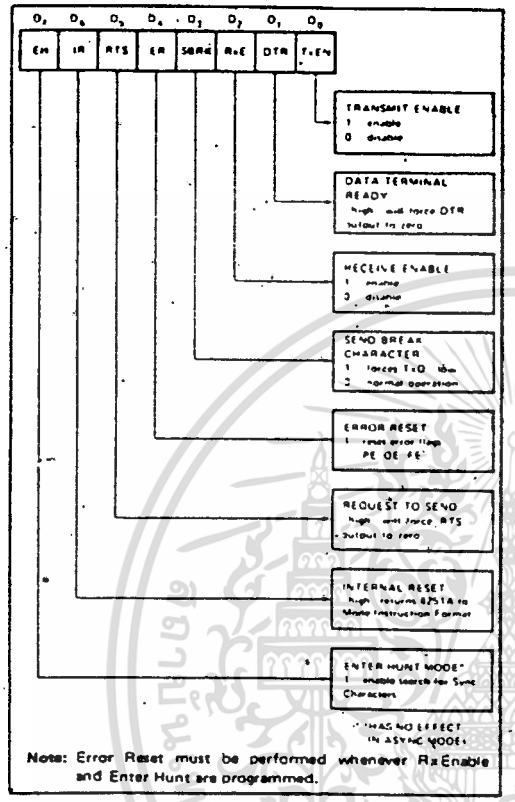
COMMAND INSTRUCTION DEFINITION

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

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

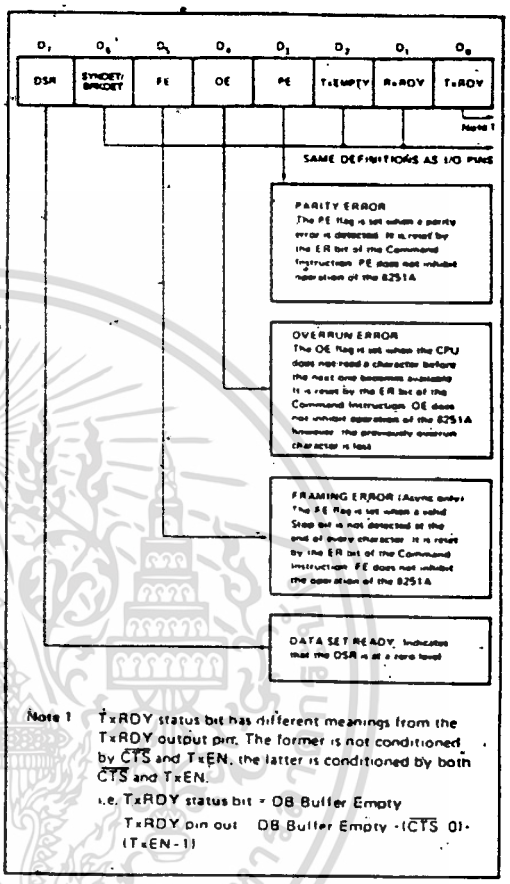
Note: Internal Reset on Power-up

When power is first applied, the 8251A may come up in the Mode, Sync character or Command format. To guarantee that the device is in the Command Instruction format before the Reset command is issued, it is safest to execute the worst-case initialization sequence (sync mode with two sync characters). Loading three 00Hs consecutively into the device with C/D = 1 configures sync operation and writes two dummy 00H sync characters. An Internal Reset command (40H) may then be issued to return the device to the "idle" state.



Note: Error Reset must be performed whenever RxEnable and Enter Hunt are programmed.

Figure 12. Command Instruction Format



Note 1 TxRDY status bit has different meanings from the TxRDY output pin. The former is not conditioned by $\overline{CTS}$ and TxEN, the latter is conditioned by both $\overline{CTS}$ and TxEN.

i.e. TxRDY status bit = DB Buffer Empty

TxRDY pin out = DB Buffer Empty · ($\overline{CTS}$ 0) · (TxEN-1)

Figure 13. Status Read Format

STATUS READ DEFINITION

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

A normal "read" command is issued by the CPU with C/$\overline{D}$ = 1 to accomplish this function.

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

Note that status update can have a maximum delay of 28 clock periods from the actual event affecting the status.

APPLICATIONS OF THE 8251A

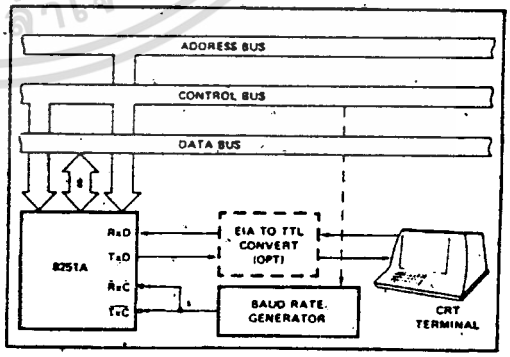


Figure 14. Asynchronous Serial Interface to CRT Terminal, DC—9600 Baud

205222-002

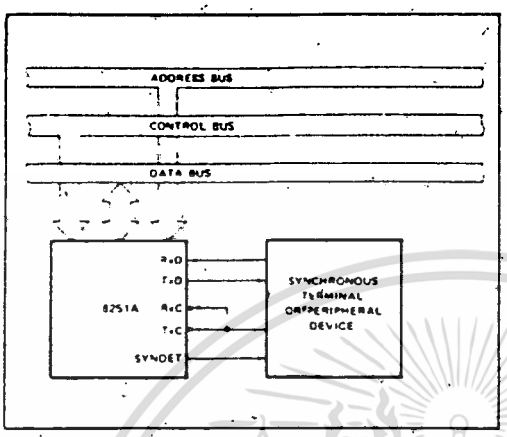


Figure 15. Synchronous Interface to Terminal or Peripheral Device

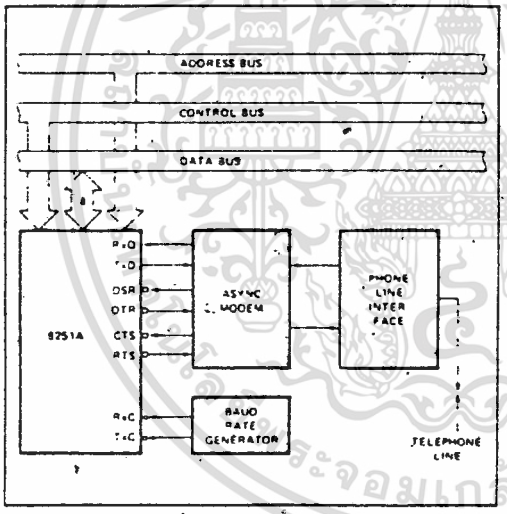


Figure 16. Asynchronous Interface to Telephone Lines

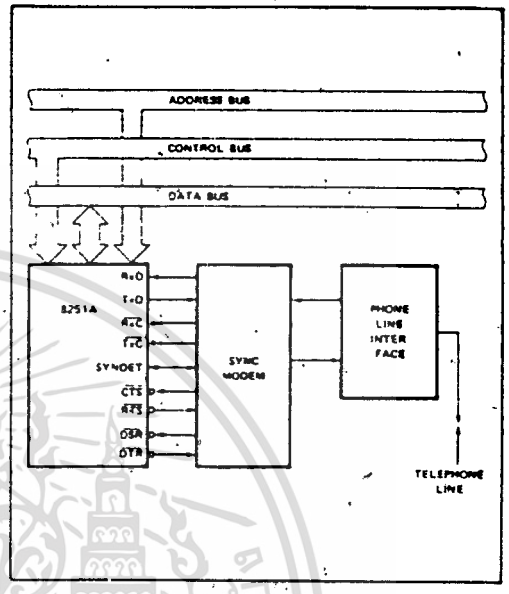


Figure 17. Synchronous Interface to Telephone Lines

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias ..... 0°C to 70°C
Storage Temperature ..... −65°C to +150°C
Voltage On Any Pin
With Respect To Ground ..... −0.5V to +7V
Power Dissipation ..... 1 Watt

*NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions 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.

D.C. CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = 5.0V ± 5%, GND = 0V)*

Symbol	Parameter	Min.	Max.	Unit	Test Conditions
V_{IL}	Input Low Voltage	−0.5	0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC}	V	
V_{OL}	Output Low Voltage		0.45	V	I_{OL} = 2.2 mA
V_{OH}	Output High Voltage	2.4		V	I_{OL} = −400 μA
I_{OFL}	Output Float Leakage		±10	μA	V_{OUT} = V_{CC} TO 0.45V
I_{IL}	Input Leakage		±10	μA	V_{IN} = V_{CC} TO 0.45V
I_{CC}	Power Supply Current		100	mA	All Outputs = High

CAPACITANCE (T_A = 25°C, V_{CC} = GND = 0V)

Symbol	Parameter	Min.	Max.	Unit	Test Conditions
C_{IN}	Input Capacitance		10	pF	fc = 1MHz
$C_{I/O}$	I/O Capacitance		20	pF	Unmeasured pins returned to GND

A.C. CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = 5.0V ±10%, GND = 0V)*

Bus Parameters (Note 1)

READ CYCLE

Symbol	Parameter	Min.	Max.	Unit	Test Conditions
t_{AR}	Address Stable Before $\overline{READ}$ ($\overline{CS}$, C/$\overline{D}$)	0		ns	Note 2
t_{RA}	Address Hold Time for $\overline{READ}$ ($\overline{CS}$, C/$\overline{D}$)	0		ns	Note 2
t_{RR}	$\overline{READ}$ Pulse Width	250		ns	
t_{RD}	Data Delay from $\overline{READ}$		200	ns	3. C_L = 150 pF
t_{DF}	$\overline{READ}$ to Data Floating	10	100	ns	

WRITE CYCLE

Symbol	Parameter	Min.	Max.	Unit	Test Conditions
t_{AW}	Address Stable Before $\overline{WRITE}$	0		ns	
t_{WA}	Address Hold Time for $\overline{WRITE}$	0		ns	
t_{WW}	$\overline{WRITE}$ Pulse Width	250		ns	
t_{DW}	Data Set-Up Time for $\overline{WRITE}$	150		ns	
t_{WD}	Data Hold Time for $\overline{WRITE}$	20		ns	
t_{RV}	Recovery Time Between WRITES	6		t_{CY}	Note 4

A.C. CHARACTERISTICS (Continued)

OTHER TIMINGS

Symbol	Parameter	Min.	Max.	Unit	Test Conditions
t_{CY}	Clock Period	320	1350	ns	Notes 5, 6
t_{ϕ}	Clock High Pulse Width	120	$t_{CY}-90$	ns	
$t_{\overline{\phi}}$	Clock Low Pulse Width	90		ns	
t_R, t_F	Clock Rise and Fall Time		20	ns	
t_{DTx}	TxD Delay from Falling Edge of $\overline{TxC}$		1	μs	
f_{Tx}	Transmitter Input Clock Frequency				
	1x Baud Rate	DC	64	kHz	
	16x Baud Rate	DC	310	kHz	
	64x Baud Rate	DC	615	kHz	
t_{TPW}	Transmitter Input Clock Pulse Width				
	1x Baud Rate	12		t_{CY}	
	16x and 64x Baud Rate	1		t_{CY}	
t_{TPD}	Transmitter Input Clock Pulse Delay				
	1x Baud Rate	15		t_{CY}	
	16x and 64x Baud Rate	3		t_{CY}	
f_{Rx}	Receiver Input Clock Frequency				
	1x Baud Rate	DC	64	kHz	
	16x Baud Rate	DC	310	kHz	
	64x Baud Rate	DC	615	kHz	
t_{RPW}	Receiver Input Clock Pulse Width				
	1x Baud Rate	12		t_{CY}	
	16x and 64x Baud Rate	1		t_{CY}	
t_{RPD}	Receiver Input Clock Pulse Delay				
	1x Baud Rate	15		t_{CY}	
	16x and 64x Baud Rate	3		t_{CY}	
t_{TxRDY}	TxRDY Pin Delay from Center of Last Bit		14	t_{CY}	Note 7
$t_{TxRDY\ CLEAR}$	TxRDY ↓ from Leading Edge of $\overline{WR}$		400	ns	Note 7
t_{RxRDY}	RxRDY Pin Delay from Center of Last Bit		26	t_{CY}	Note 7
$t_{RxRDY\ CLEAR}$	RxRDY ↓ from Leading Edge of $\overline{RD}$		400	ns	Note 7
t_{IS}	Internal SYNDET Delay from Rising Edge of $\overline{RxC}$		26	t_{CY}	Note 7
t_{ES}	External SYNDET Set-Up Time After Rising Edge of $\overline{RxC}$	18		t_{CY}	Note 7
$t_{TxEMPTY}$	TxEMPTY Delay from Center of Last Bit		20	t_{CY}	Note 7
t_{WC}	Control Delay from Rising Edge of WRITE (TxEn, $\overline{DTR}$, $\overline{RTS}$)		8	t_{CY}	Note 7
t_{CR}	Control to READ Set-Up Time ($\overline{DSR}$, $\overline{CTS}$)	20		t_{CY}	Note 7

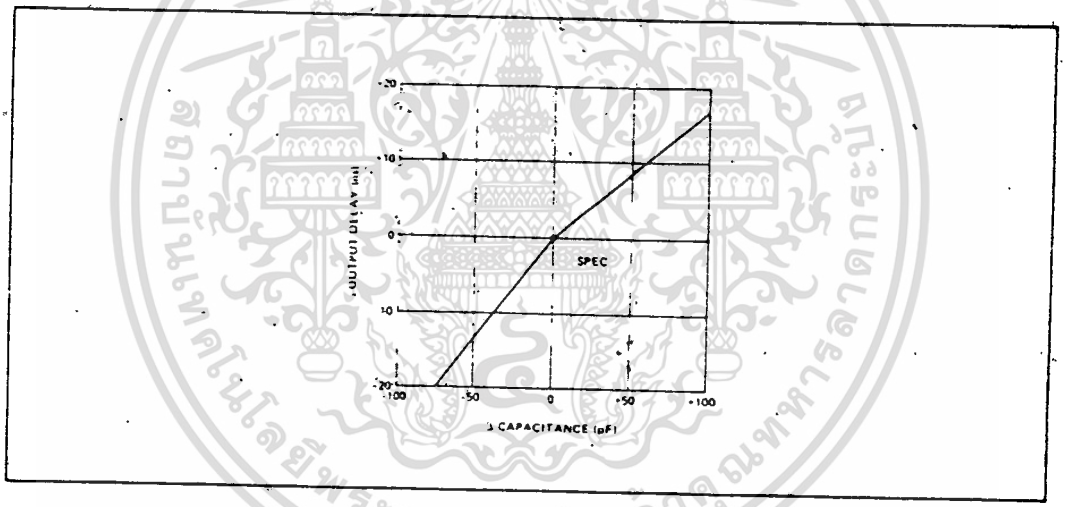
*NOTE:
1. For Extended Temperature EXPRESS, use M8251A electrical parameters.

A.C. CHARACTERISTICS (Continued)

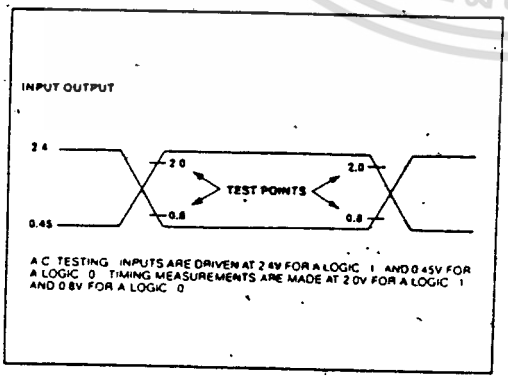
NOTES:

1. AC timings measured V_{OH} = 2.0 V_{OL} = 2.0, V_{OL} = 0.8, and with load circuit of Figure 1.
2. Chip Select (CS) and Command/Data (C/D) are considered as Addresses.
3. Assumes that Address is valid before $R_D\downarrow$.
4. This recovery time is for Mode Initialization only. Write Data is allowed only when TxRDY = 1. Recovery Time between Writes for Asynchronous Mode is 8 t_{CY} and for Synchronous Mode is 16 t_{CY}.
5. The TxC and RxC frequencies have the following limitations with respect to CLK: For 1x Baud Rate, f_{Tx} or $f_{Rx} \leq 1/(30\ t_{CY})$
   For 16x and 64x Baud Rate, f_{Tx} or $f_{Rx} \leq 1/(4.5\ t_{CY})$.
6. Reset Pulse Width = 6 t_{CY} minimum; System Clock must be running during Reset.
7. Status update can have a maximum delay of 28 clock periods from the event affecting the status.

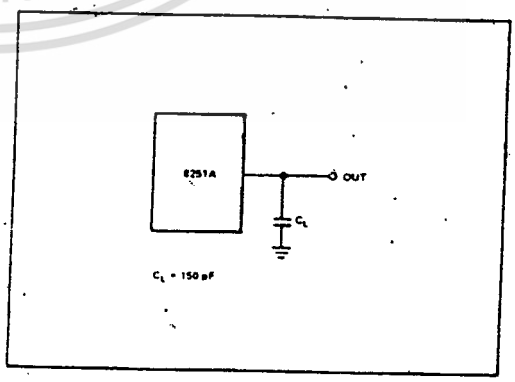
TYPICAL Δ OUTPUT DELAY VS. Δ CAPACITANCE (pF)



A.C. TESTING INPUT, OUTPUT WAVEFORM



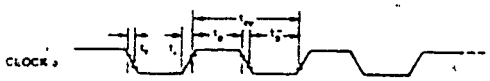
A.C. TESTING LOAD CIRCUIT



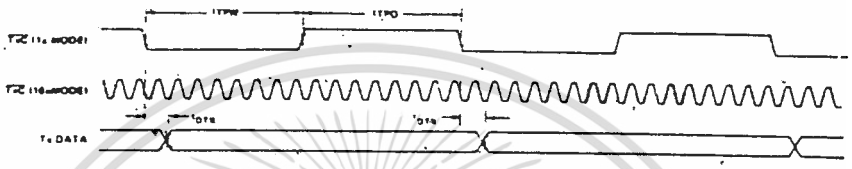
205222-002

WAVEFORMS

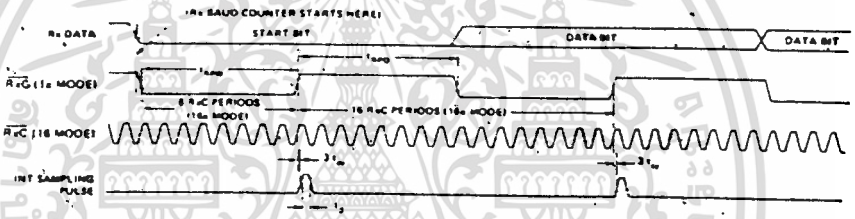
SYSTEM CLOCK INPUT



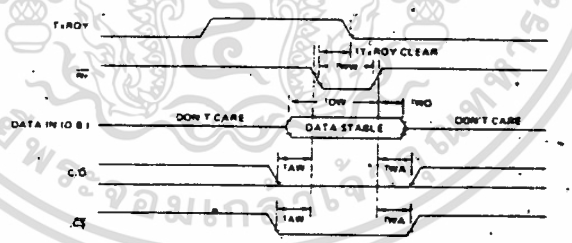
TRANSMITTER CLOCK AND DATA



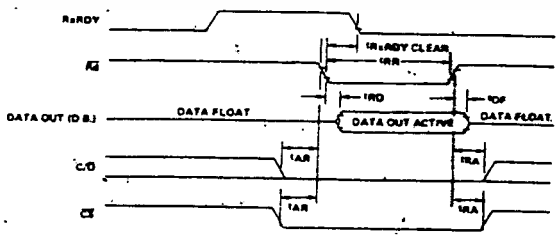
RECEIVER CLOCK AND DATA



WRITE DATA CYCLE (CPU → USART)



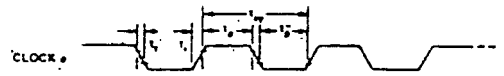
READ DATA CYCLE (CPU ← USART)



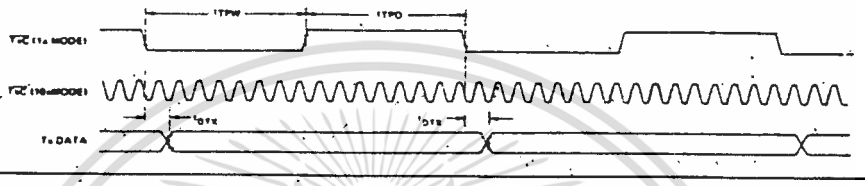
205222-002

WAVEFORMS

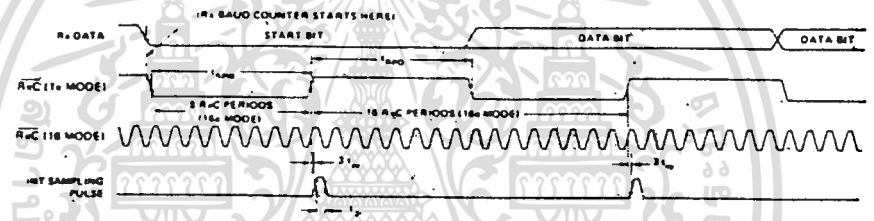
SYSTEM CLOCK INPUT



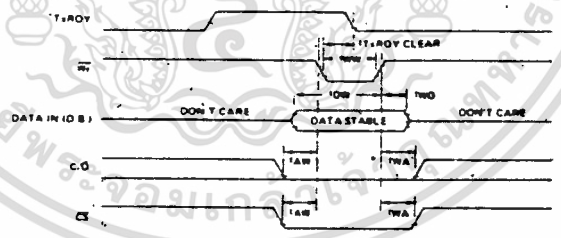
TRANSMITTER CLOCK AND DATA



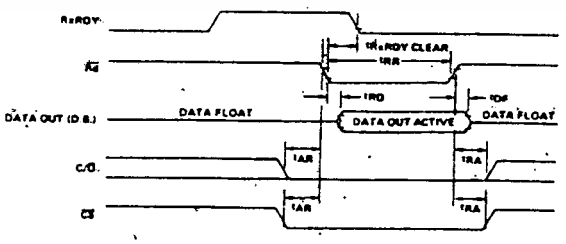
RECEIVER CLOCK AND DATA



WRITE DATA CYCLE (CPU → USART)



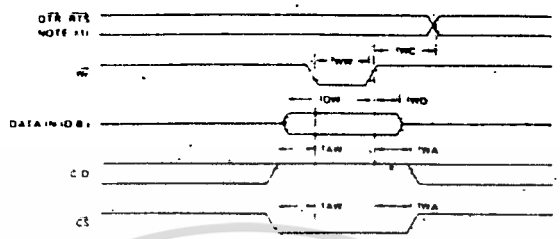
READ DATA CYCLE (CPU ← USART)



205222-002

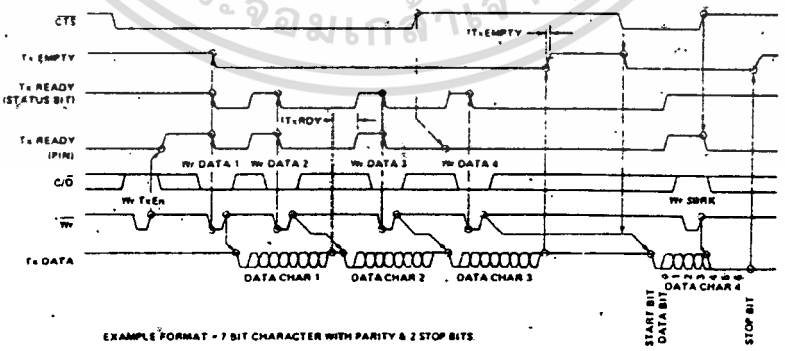
WAVEFORMS (Continued)

WRITE CONTROL OR OUTPUT PORT CYCLE (CPU → USART)



READ CONTROL OR INPUT PORT (CPU ← USART)

TRANSMITTER CONTROL AND FLAG TIMING (ASYNC MODE)



205222-002

intel

8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

- MCS-85™ Compatible 8255A-5
- 24 Programmable I/O Pins
- Completely TTL Compatible
- Fully Compatible with Intel® Microprocessor Families
- Improved Timing Characteristics
- Direct Bit Set/Reset Capability Easing Control Application Interface
- Reduces System Package Count
- Improved DC Driving Capability
- Available in EXPRESS
  - —Standard Temperature Range
  - —Extended Temperature Range

The Intel® 8255A is a general purpose programmable I/O device designed for use with Intel® microprocessors. It has 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. In the first mode (MODE 0), each group of 12 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 4 pins, 3 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 5 lines, borrowing one from the other group, for handshaking.

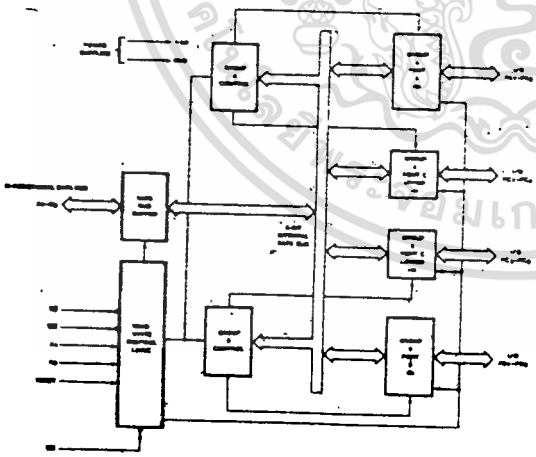


Figure 1. 8255A Block Diagram

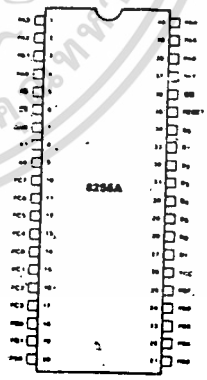


Figure 2. Pin Configuration

Order Number: 231308-001

8255A FUNCTIONAL DESCRIPTION

General

The 8255A is a programmable peripheral interface (PPI) device designed for use in Intel® microcomputer systems. Its function is that of a general purpose I/O component to interface peripheral equipment to the microcomputer system bus. The functional configuration of the 8255A 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 bidirectional 8-bit buffer is used to interface the 8255A to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the 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 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 communiction between the 8255A and the CPU.

($\overline{RD}$)

Read. A "low" on this input pin enables the 8255A to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 8255A.

($\overline{WR}$)

Write. A "low" on this input pin enables the CPU to write data or control words into the 8255A.

(A_0 and A_1)

Port Select 0 and Port Select 1. These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word registers. They are normally connected to the least significant bits of the address bus (A_0 and A_1).

8255A 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
X	X	1	1	0	DATA BUS → 3-STATE

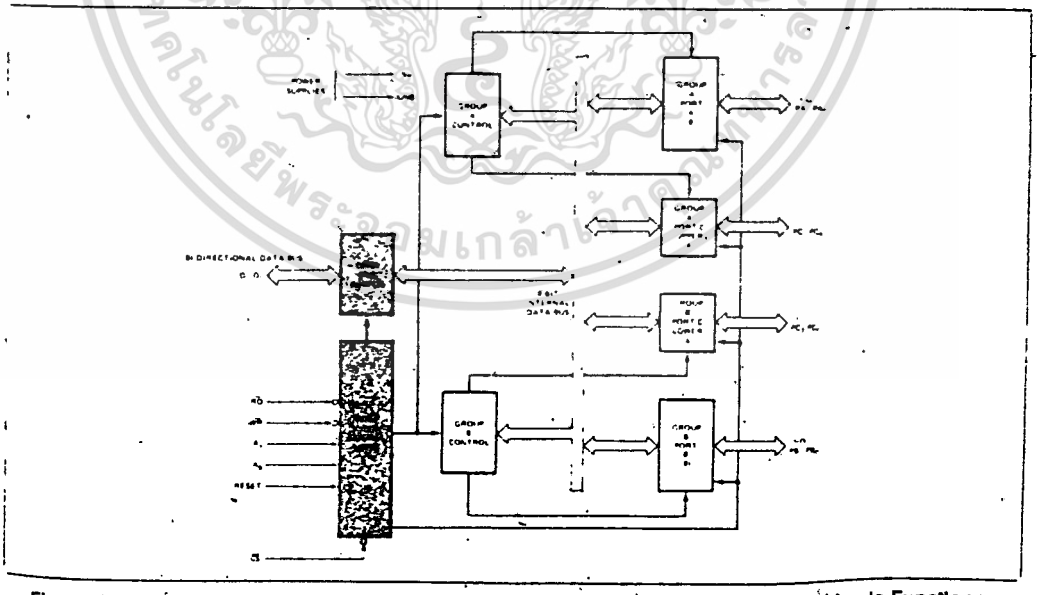


Figure 3. 8255A Block Diagram Showing Data Bus Buffer and Read/Write Control Logic Functions

231308-001

(RESET)

Reset. A "high" on this input clears 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 CPU "outputs" a control word to the 8255A. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 8255A.

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 8255A 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 8255A.

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.

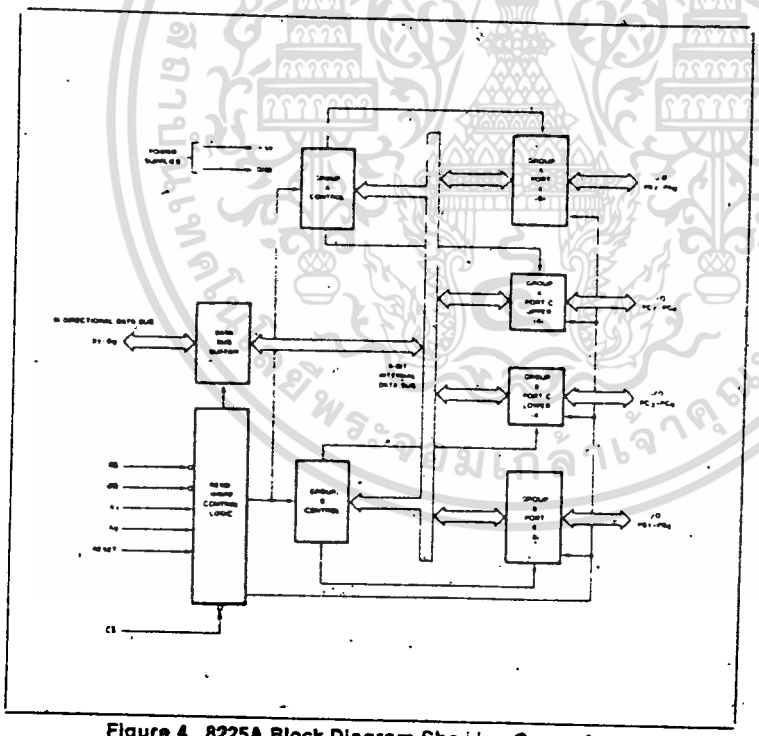
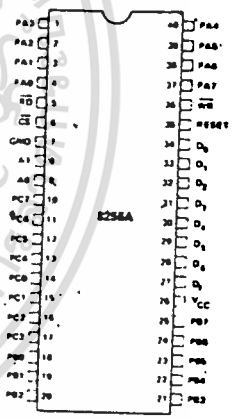


Figure 4. 8225A Block Diagram Showing Group A and Group B Control Functions

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

231308-001

8255A 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 8255A 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 8255A 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.

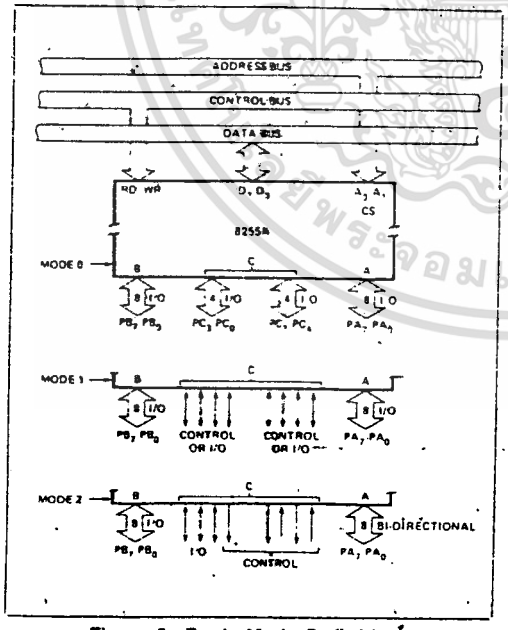


Figure 5. Basic Mode Definitions and Bus Interface

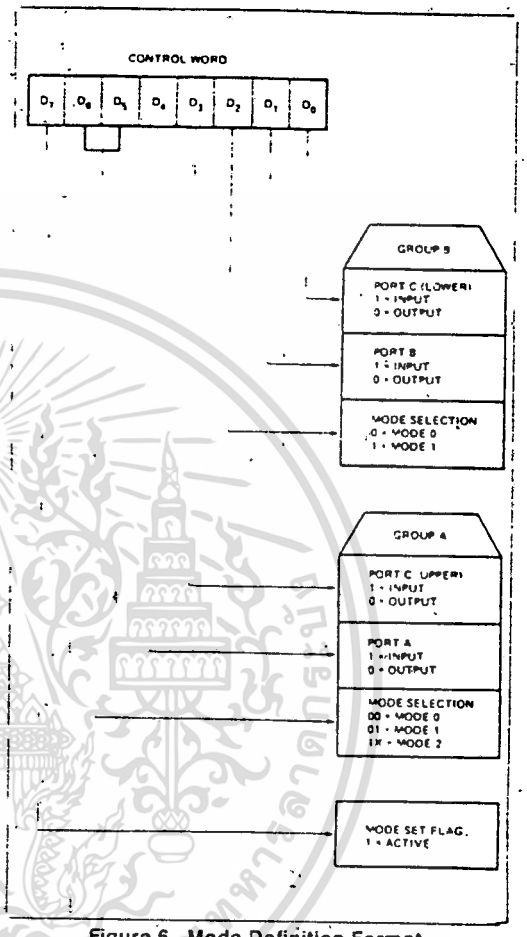


Figure 6. Mode Definition Format

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 8255A 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.

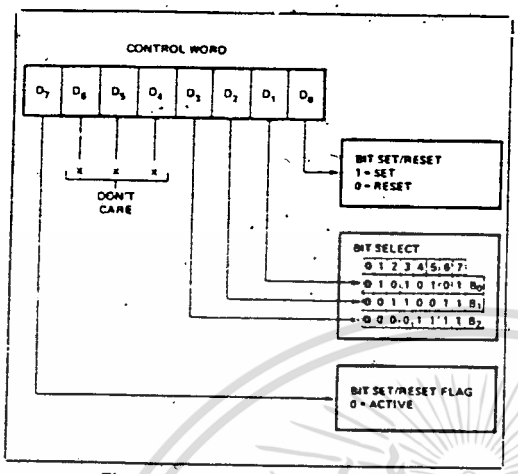


Figure 7. Bit Set/Reset Format

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 8255A 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 affecting 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

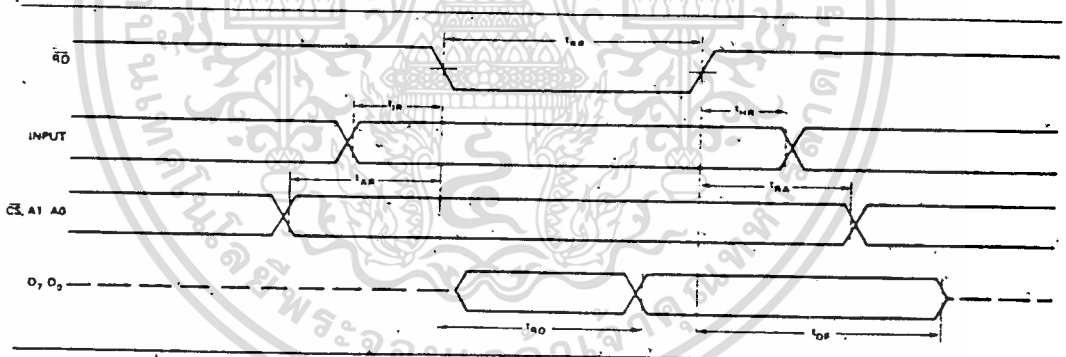
Note: All Mask flip-flops are automatically reset during mode selection and device Reset.

Operating Modes

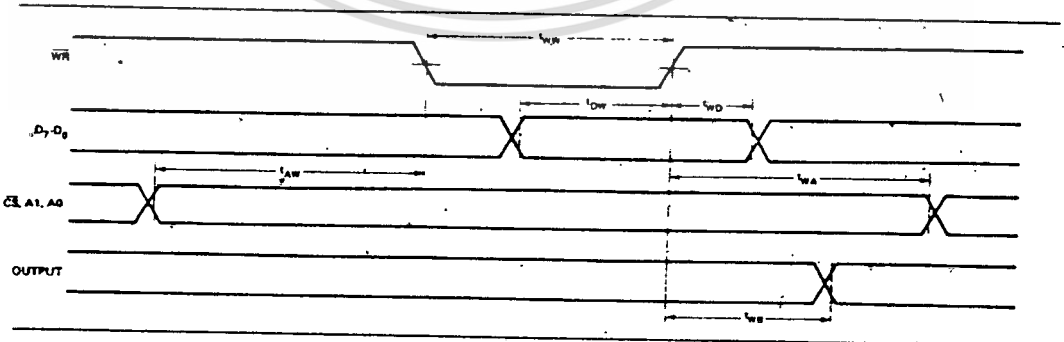
MODE 0 (Basic Input/Output). This functional configuration provides simple input and output operations for each of the three ports. No "handshaking" is required, data is simply written to or read from a specified port.

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 (Basic Input)



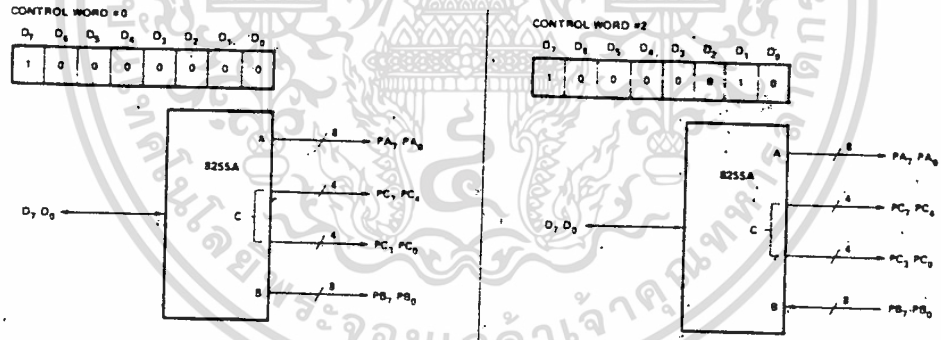
MODE 0 (Basic Output)

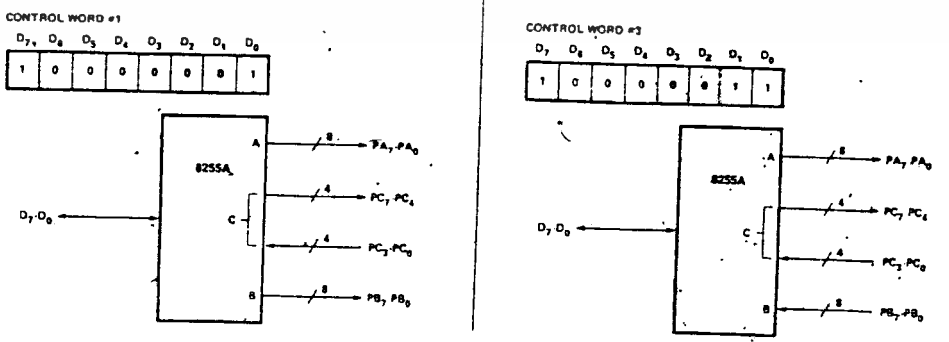
231308-001

MODE 0 Port Definition

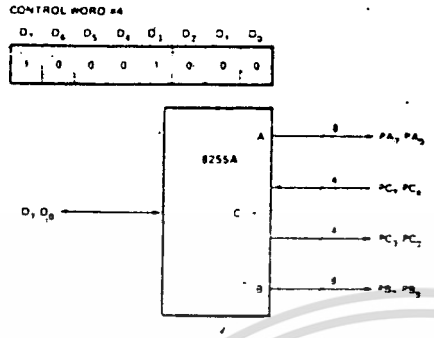
A		B		GROUP A			GROUP B	
D_4	D_3	D_1	D_0	PORT A	PORT C (UPPER)	#	PORT B	PORT C (LOWER)
0	0	0	0	OUTPUT	OUTPUT	0	OUTPUT	OUTPUT
0	0	0	1	OUTPUT	OUTPUT	1	OUTPUT	INPUT
0	0	1	0	OUTPUT	OUTPUT	2	INPUT	OUTPUT
0	0	1	1	OUTPUT	OUTPUT	3	INPUT	INPUT
0	1	0	0	OUTPUT	INPUT	4	OUTPUT	OUTPUT
0	1	0	1	OUTPUT	INPUT	5	OUTPUT	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

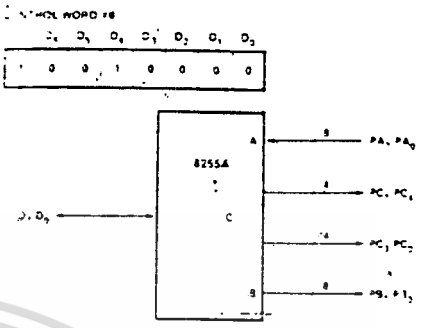
MODE 0 Configurations

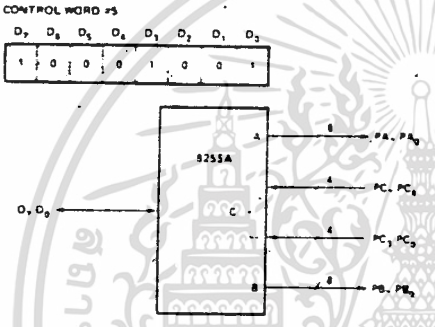


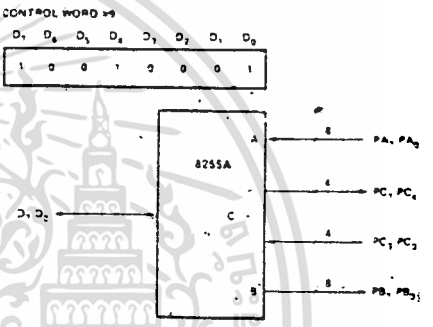


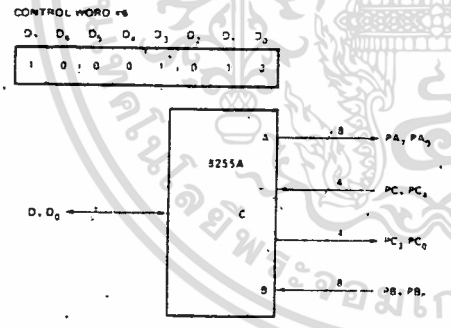
231308-001

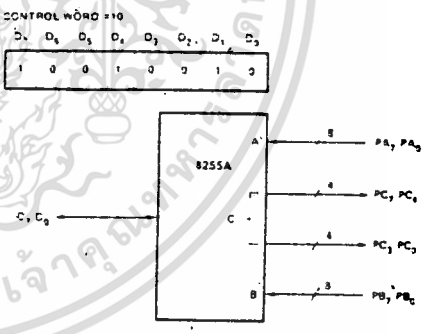


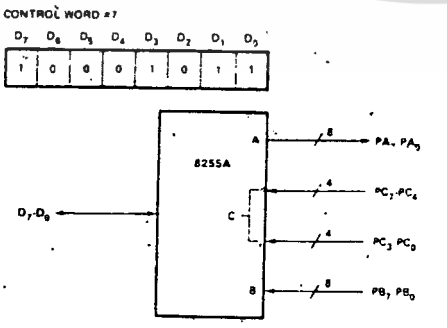


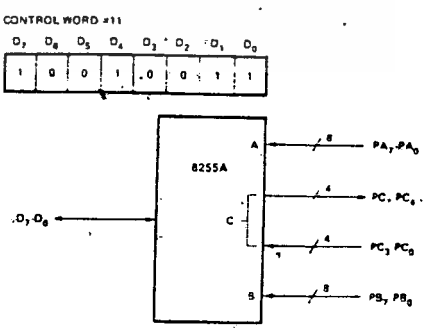


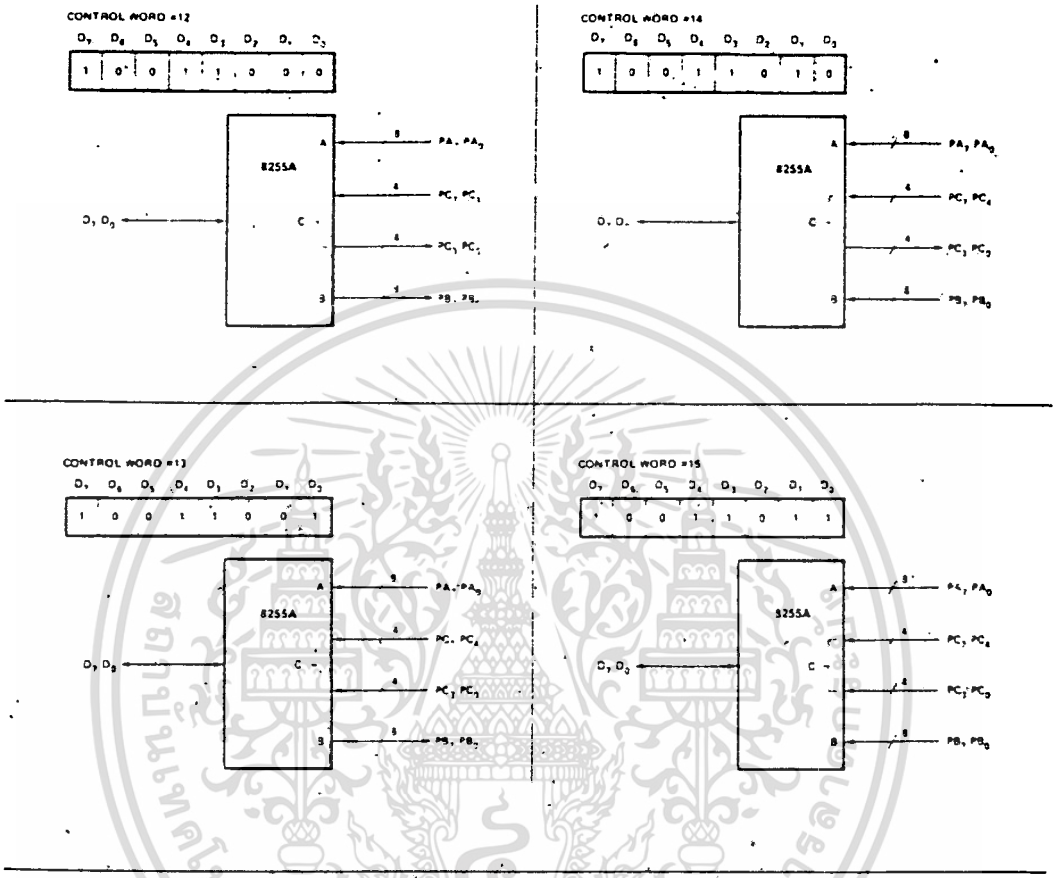












Operating Modes

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 STB input being low 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 $\overline{STB}$ is a "one", 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.

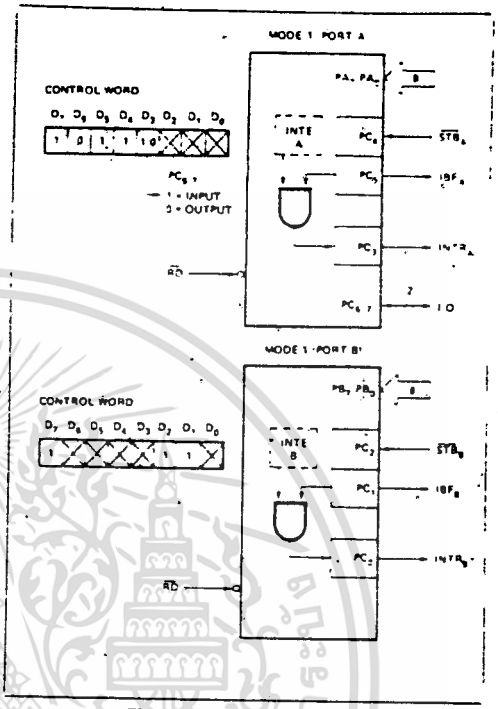


Figure 8. MODE 1 Input

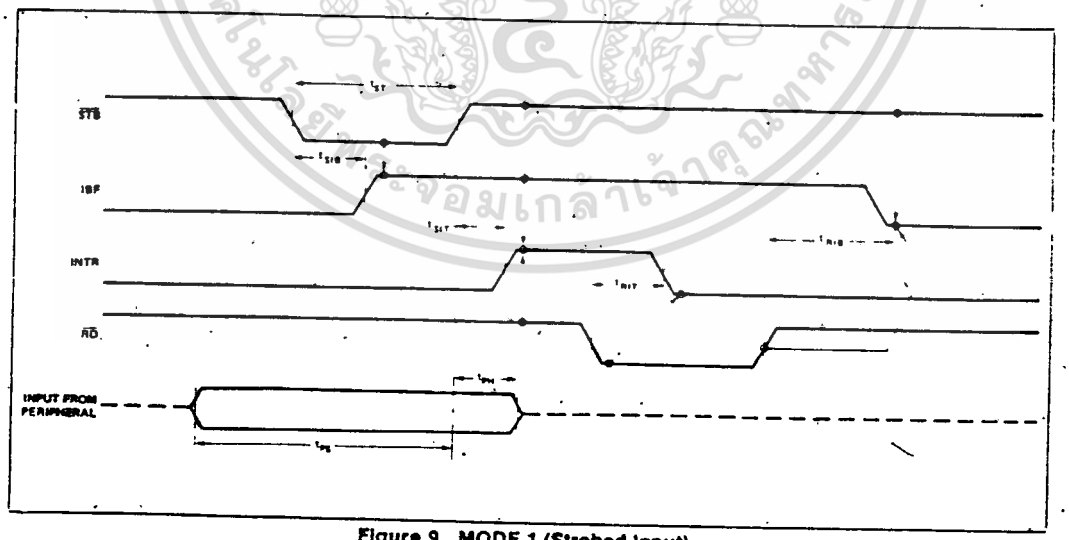


Figure 9. MODE 1 (Strobed Input)

Output Control Signal Definition

$\overline{OBF}$ (Output Buffer Full F/F). The 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 ACK input being low.

$\overline{ACK}$ (Acknowledge Input). A "low" on this input informs the 8255A that the data from port A or port B has been accepted. In essence, a response from the peripheral device indicating that it has recieved 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 when $\overline{ACK}$ is a "one", OBF is a "one", and INTE is a "one". It is reset by the falling edge of $\overline{WR}$.

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 when $\overline{ACK}$ is a "one", OBF is a "one", and INTE is a "one". It is reset by the falling edge of WR.

INTE A

Controlled by bit set/reset of PC_6.

INTE B

Controlled by bit set/reset of PC_2.

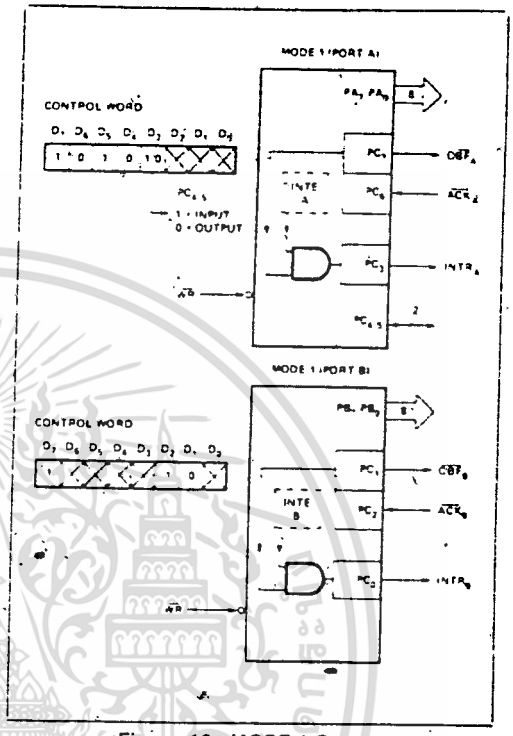


Figure 10. MODE 1 Output

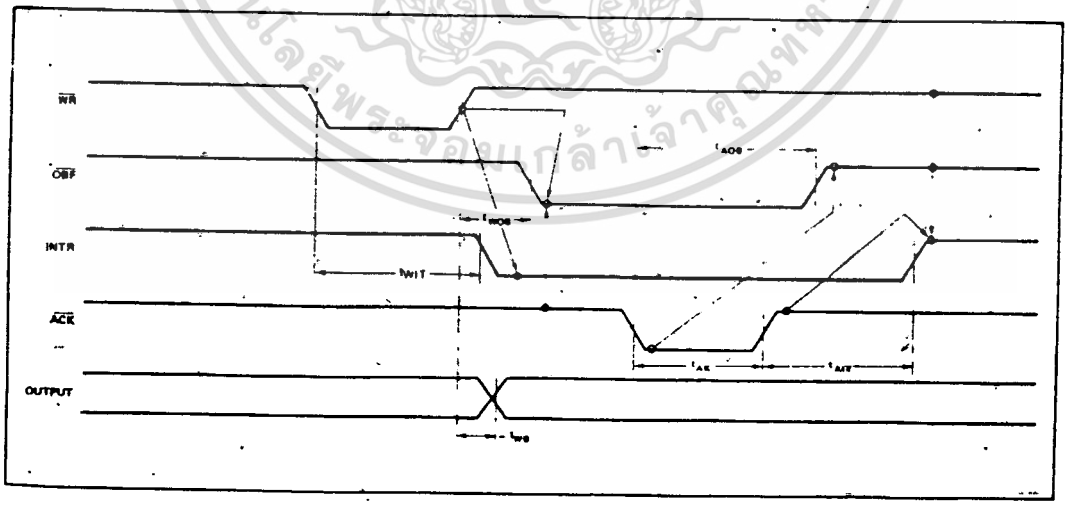


Figure 11. Mode 1 (Strobed Output)

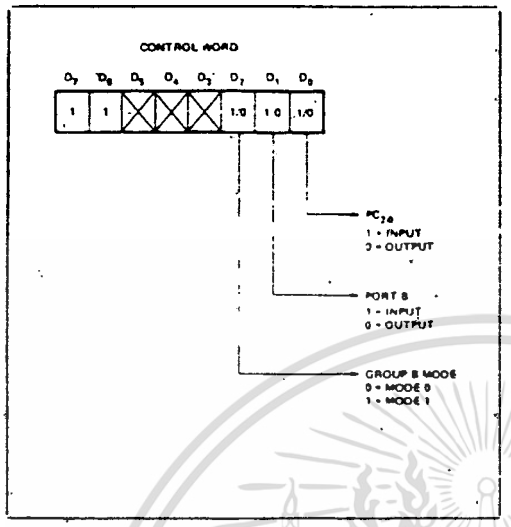


Figure 13. MODE Control Word

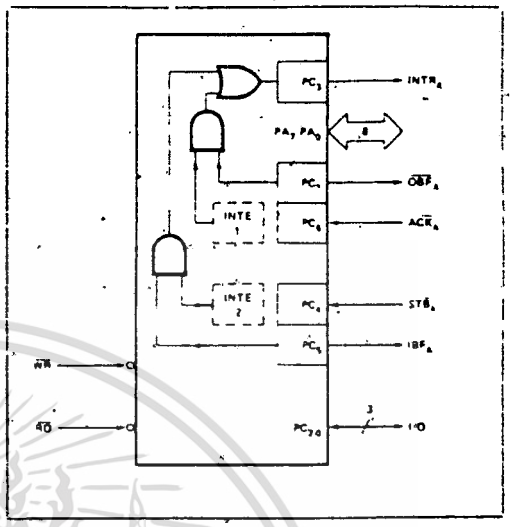


Figure 14. MODE 2

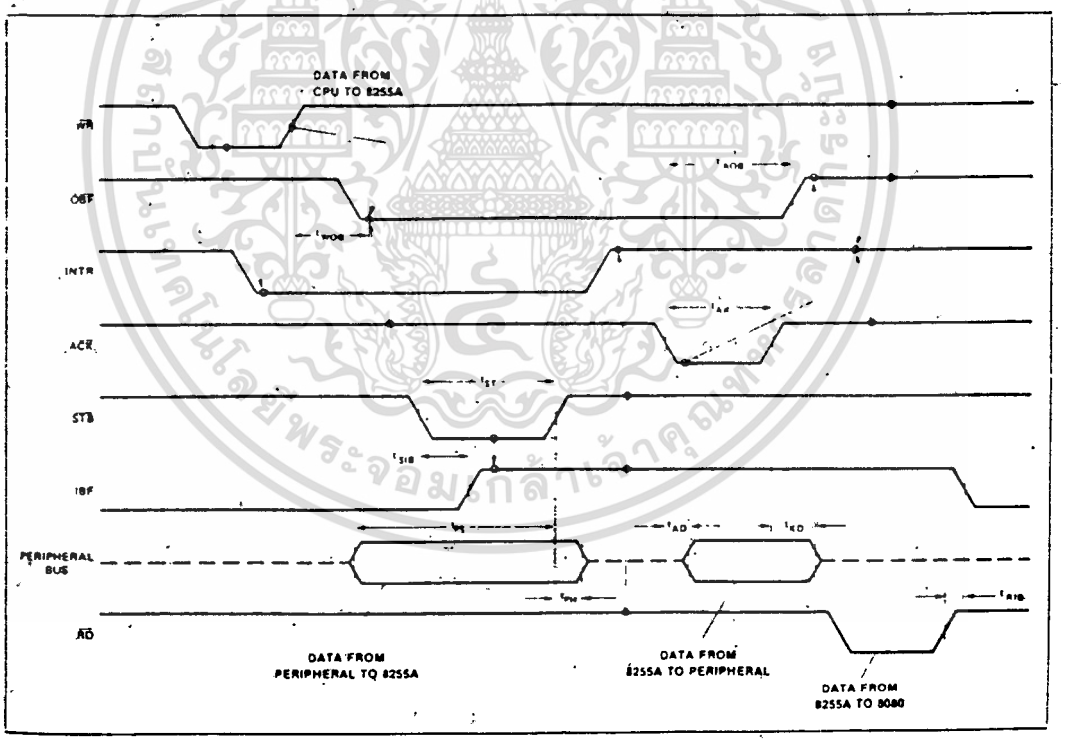


Figure 15. MODE 2 (Bidirectional)

NOTE Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$ and $\overline{STB}$ occurs before $\overline{RD}$ is permissible.
(INTR = IBF • $\overline{MASK}$ • $\overline{STB}$ • $\overline{RD}$ + $\overline{OBF}$ • $\overline{MASK}$ • $\overline{ACK}$ • $\overline{WR}$)

231308-001

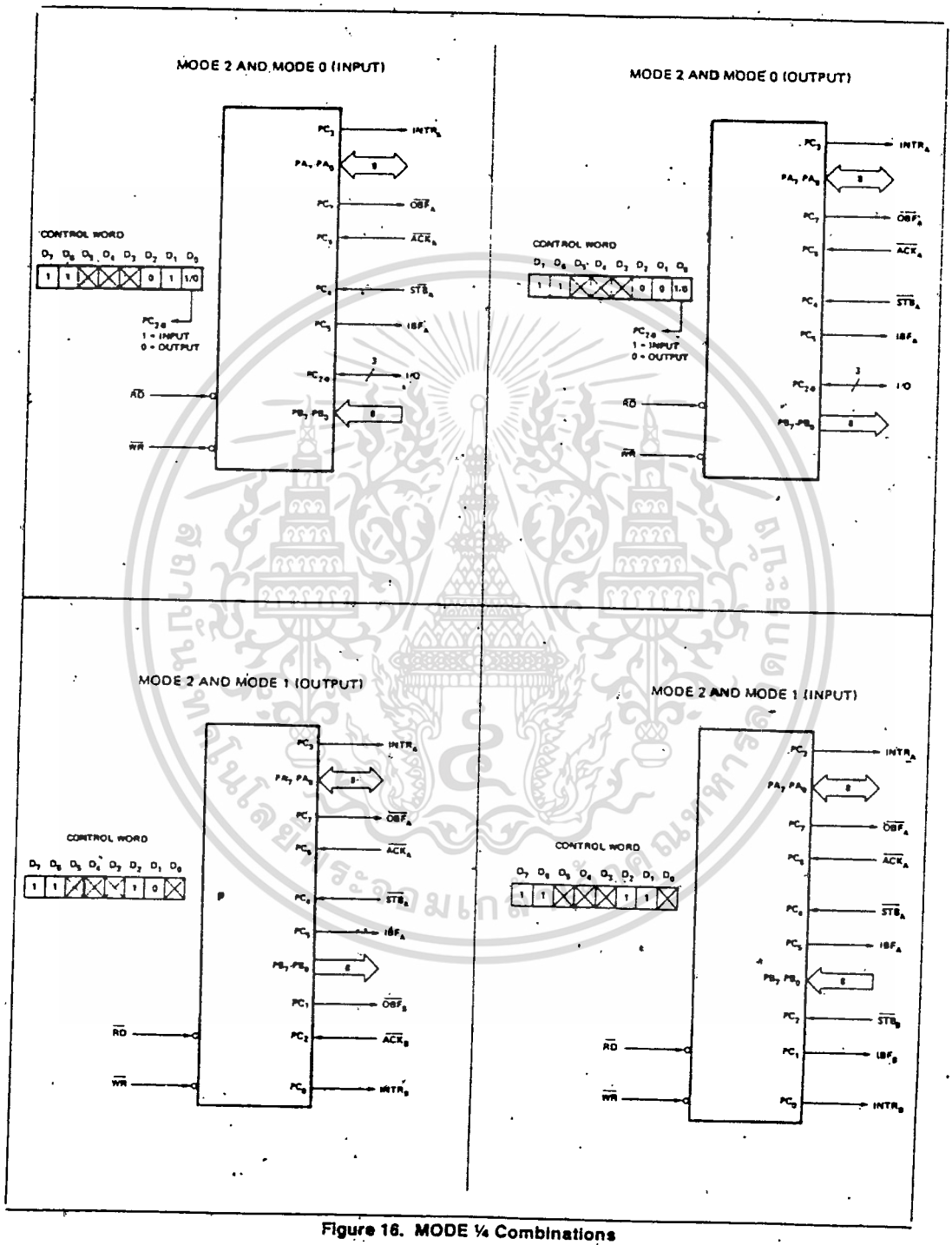


Figure 16. MODE ¼ Combinations

231308-001

Mode Definition Summary

	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.

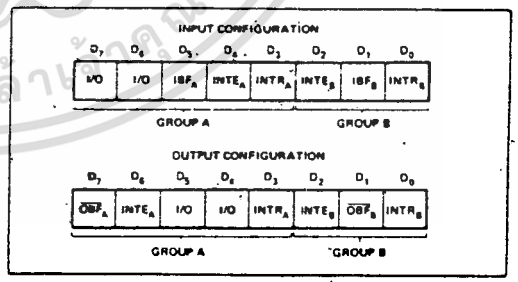


Figure 17. MODE 1 Status Word Format

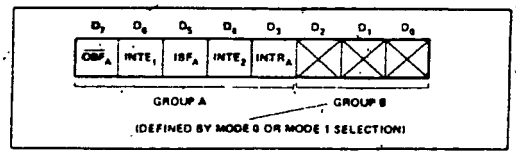


Figure 18. MODE 2 Status Word Format

APPLICATIONS OF THE 8255A

The 8255A is a very powerful tool for interfacing peripheral equipment to the microcomputer system. It represents the optimum use of available pins and is flexible enough to interface almost any I/O device without the need for additional external logic.

Each peripheral device in a microcomputer system usually has a "service routine" associated with it. The routine manages the software interface between the device and the CPU. The functional definition of the 8255A is programmed by the I/O service routine and becomes an extension of the system software. By examining the I/O devices interface characteristics for both data transfer and timing, and matching this information to the examples and tables in the detailed operational description, a control word can easily be developed to initialize the 8255A to exactly fit the application. Figures 19 through 25 present a few examples of typical applications of the 8255A.

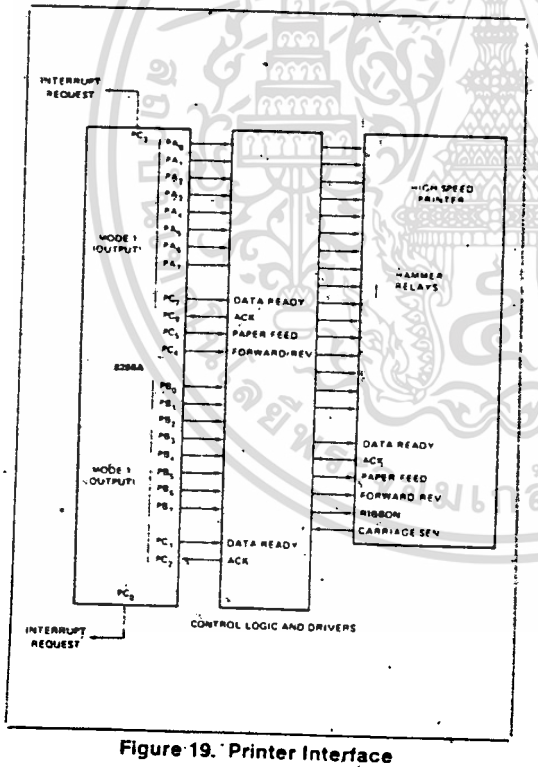


Figure 19. Printer Interface

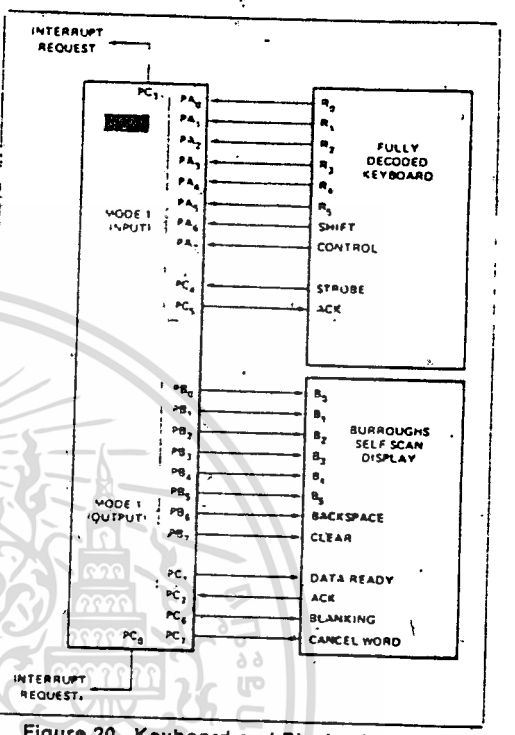


Figure 20. Keyboard and Display Interface

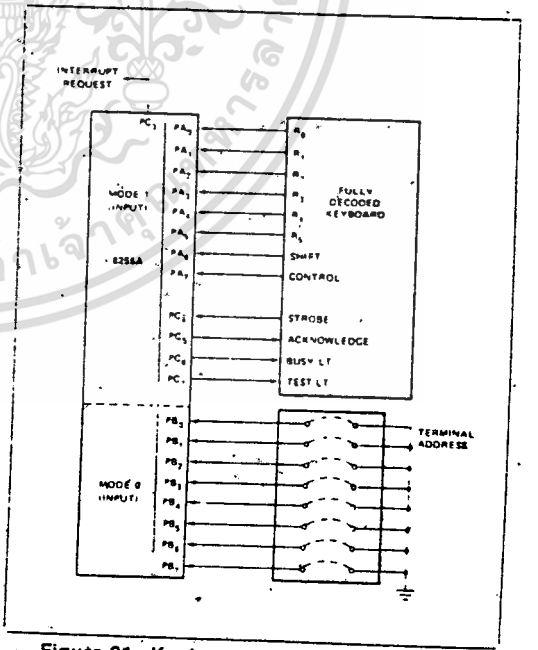


Figure 21. Keyboard and Terminal Address Interface

APPLICATIONS OF THE 8255A

The 8255A is a very powerful tool for interfacing peripheral equipment to the microcomputer system. It represents the optimum use of available pins and is flexible enough to interface almost any I/O device without the need for additional external logic.

Each peripheral device in a microcomputer system usually has a "service routine" associated with it. The routine manages the software interface between the device and the CPU. The functional definition of the 8255A is programmed by the I/O service routine and becomes an extension of the system software. By examining the I/O devices interface characteristics for both data transfer and timing, and matching this information to the examples and tables in the detailed operational description, a control word can easily be developed to initialize the 8255A to exactly fit the application. Figures 19 through 25 present a few examples of typical applications of the 8255A.

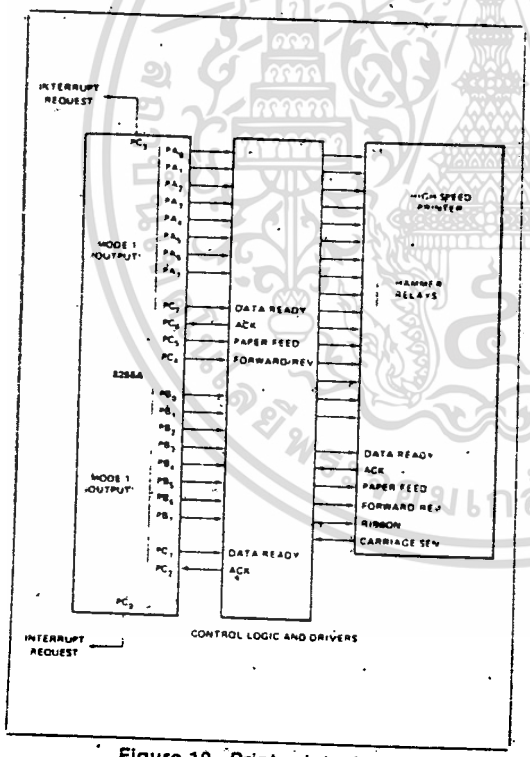


Figure 19. Printer Interface

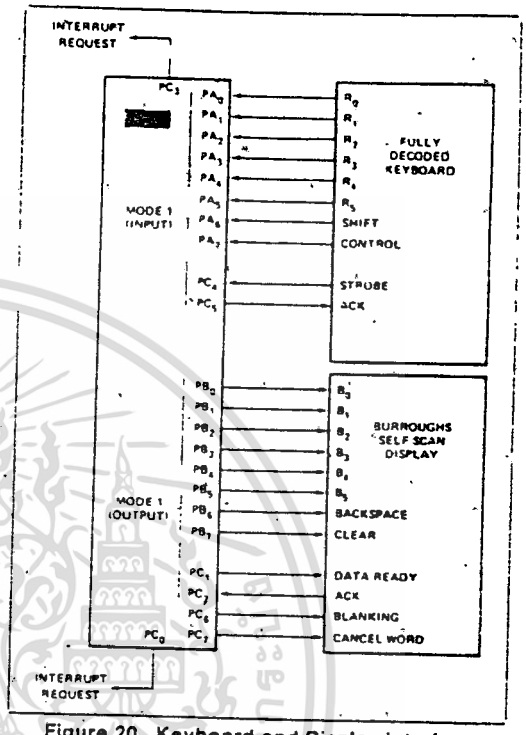


Figure 20. Keyboard and Display Interface

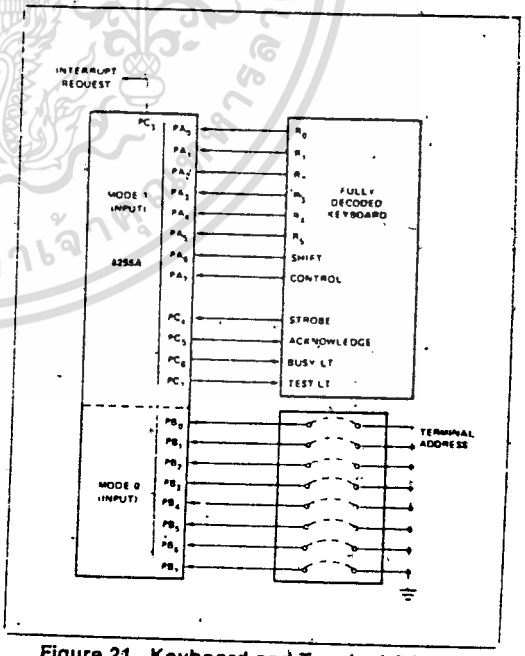


Figure 21. Keyboard and Terminal Address Interface

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias	0°C to 70°C
Storage Temperature	−65°C to +150°C
Voltage on Any Pin With Respect to Ground	−0.5V to +7V
Power Dissipation	1 Watt

*NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions 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.

D.C. CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = +5V ± 10%, GND = 0V)*

Symbol	Parameter	Min.	Max.	Unit	Test Conditions
V_{IL}	Input Low Voltage	-0.5	0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC}	V	
V_{OL} (DB)	Output Low Voltage (Data Bus)		0.45*	V	I_{OL} = 2.5mA
V_{OL}(PER)	Output Low Voltage (Peripheral Port)		0.45*	V	I_{OL} = 1.7mA
V_{OH}(DB)	Output High Voltage (Data Bus)	2.4		V	I_{OH} = -400μA
V_{OH}(PER)	Output High Voltage (Peripheral Port)	2.4		V	I_{OH} = -200μA
I_{DAR}[1]	Darlington Drive Current	-1.0	-4.0	mA	R_{EXT} = 750Ω; V_{EXT} = 1.5V
I_{CC}	Power Supply Current		120	mA	
I_{IL}	Input Load Current		±10	μA	V_{IN} = V_{CC} to 0V
I_{OFL}	Output Float Leakage		±10	μA	V_{OUT} = V_{CC} to .45V

NOTE:
1. Available on any 8 pins from Port B and C.

CAPACITANCE (T_A = 25°C, V_{CC} = GND = 0V)

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
C_{IN}	Input Capacitance			10	pF	fc = 1MHz
$C_{I/O}$	I/O Capacitance			20	pF	Unmeasured pins returned to GND

A.C. CHARACTERISTICS (T_A = 0°C to 70°C, V_{CC} = +5V ± 10%, GND = 0V)*

Bus Parameters

READ

Symbol	Parameter	8255A		8255A-5		Unit
		Min.	Max.	Min.	Max.	
t_{AR}	Address Stable Before READ	0		0		ns
t_{RA}	Address Stable After READ	0		0		ns
t_{RR}	READ Pulse Width	300		300		ns
t_{RD}	Data Valid From READ[1]		250		200	ns
t_{DF}	Data Float After READ	10	150	10	100	ns
t_{RV}	Time Between READs and/or WRITEs	850		850		ns

A.C. CHARACTERISTICS (Continued)

WRITE

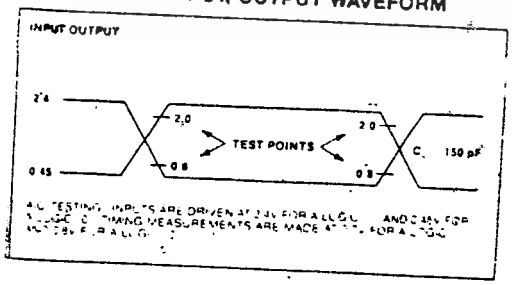
Symbol	Parameter	8255A		8255A-5		Unit
		Min.	Max.	Min.	Max.	
t_{AW}	Address Stable Before WRITE	0		0		ns
t_{WA}	Address Stable After WRITE	20		20		ns
t_{WW}	WRITE Pulse Width	400		300		ns
t_{DW}	Data Valid to WRITE (T.E.)	100		100		ns
t_{WD}	Data Valid After WRITE	30		30		ns

OTHER TIMINGS

Symbol	Parameter	8255A		8255A-5		Unit
		Min.	Max.	Min.	Max.	
t_{WB}	WR = 1 to Output[1]		350		350	ns
t_{IR}	Peripheral Data Before RD	0		0		ns
t_{HR}	Peripheral Data After RD	0		0		ns
t_{AK}	ACK Pulse Width	300		300		ns
t_{ST}	STB Pulse Width	500		500		ns
t_{PS}	Per. Data Before T.E. of STB	0		0		ns
t_{PH}	Per. Data After T.E. of STB	180		180		ns
t_{AD}	ACK = 0 to Output[1]		300		300	ns
t_{KD}	ACK = 1 to Output Float	20	250	20	250	ns
t_{WOB}	WR = 1 to OBF = 0[1]		650		650	ns
t_{AOB}	ACK = 0 to OBF = 1[1]		350		350	ns
t_{SIB}	STB = 0 to IBF = 1[1]		300		300	ns
t_{RIB}	RD = 1 to IBF = 0[1]		300		300	ns
t_{RIT}	RD = 0 to INTR = 0[1]		400		400	ns
t_{SIT}	STB = 1 to INTR = 1[1]		300		300	ns
t_{AIT}	ACK = 1 to INTR = 1[1]		350		350	ns
t_{WIT}	WR = 0 to INTR = 0[1,3]		450		450	ns

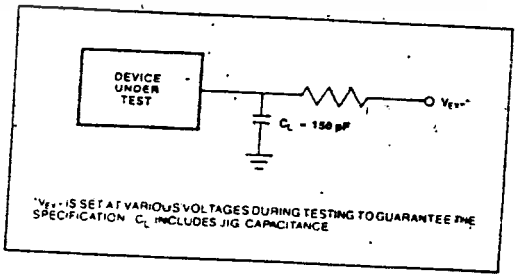
NOTES:
1. Test Conditions: C_L = 150 pF
2. Period of Reset pulse must be at least 50μs during or after power on. Subsequent Reset pulse can be 500 ns min.
3. INTR↑ may occur as early as $\overline{WR}$↓.
* For Extended Temperature EXPRESS use M8255A electrical parameters

A.C. TESTING INPUT, OUTPUT WAVEFORM



A.C. TESTING INPUTS ARE DRIVEN AT 2.4V FOR A LOGIC "1" AND 0.45V FOR A LOGIC "0". TIMING MEASUREMENTS ARE MADE AT 2.0V FOR A LOGIC "1" AND 0.8V FOR A LOGIC "0".

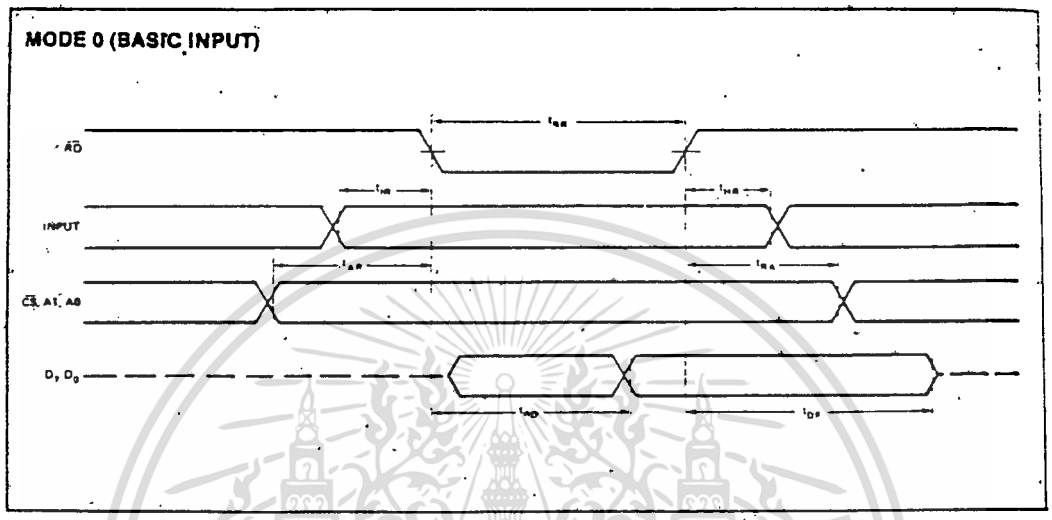
A.C. TESTING LOAD CIRCUIT



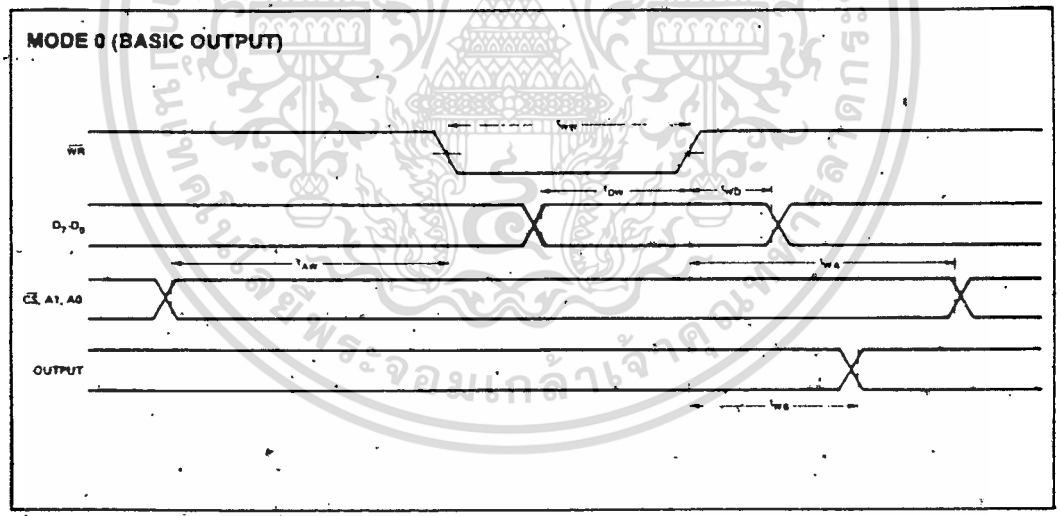
V_{CC}* IS SET AT VARIOUS VOLTAGES DURING TESTING TO GUARANTEE THE SPECIFICATION. C_L INCLUDES JIG CAPACITANCE.

WAVEFORMS

MODE 0 (BASIC INPUT)

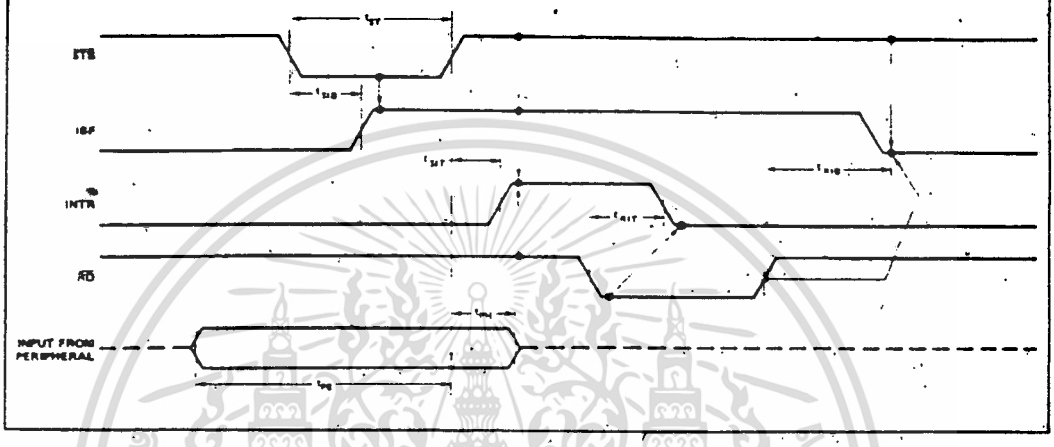


MODE 0 (BASIC OUTPUT)

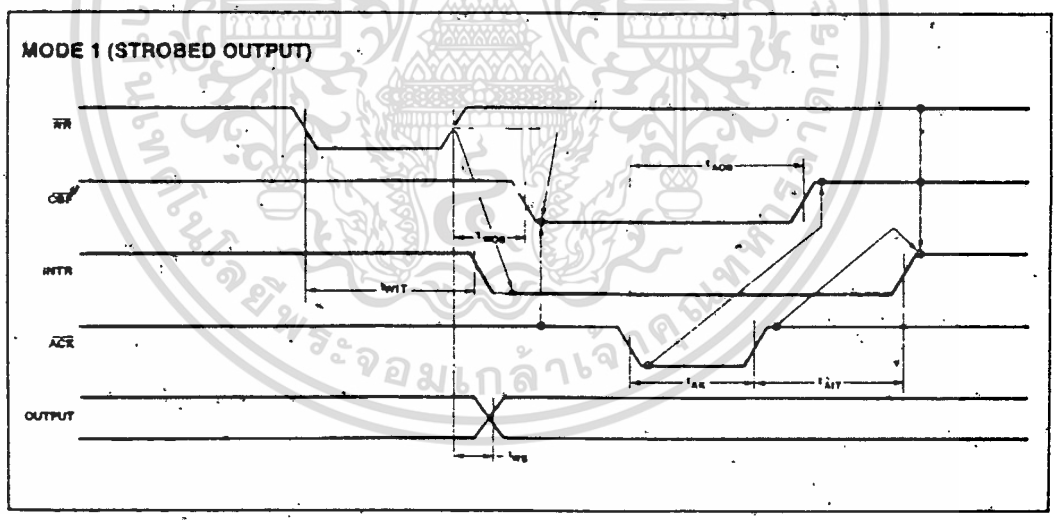


WAVEFORMS (Continued)

MODE 1 (STROBED INPUT)

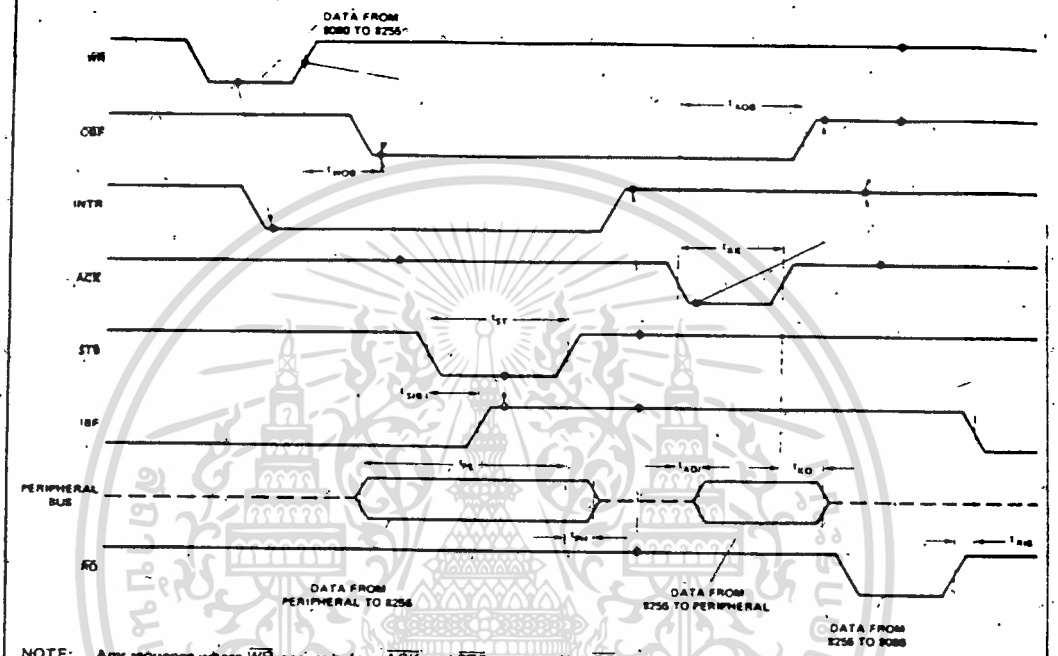


MODE 1 (STROBED OUTPUT)



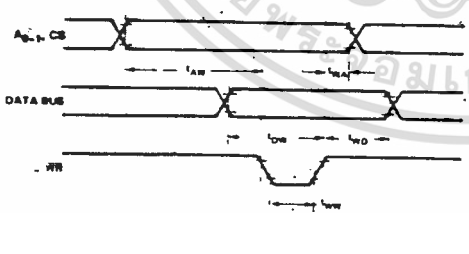
WAVEFORMS (Continued)

MODE 2 (BIDIRECTIONAL)



NOTE: Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$ and $\overline{STB}$ occurs before $\overline{RD}$ is permissible.
(INTR = IBF · MASK · $\overline{STB}$ · $\overline{RD}$ + $\overline{OBF}$ · MASK · $\overline{ACK}$ · $\overline{WR}$)

WRITE TIMING



READ TIMING