เครื่องจำลองการทำงานของอีพรอม แบบไร้สาย

WIRELESS EPROM EMULATOR

โดย

นายรุ่งโรจน์ เลิศรัตนศาสตร์

นายวีระยุทธ จีวรมงคล

นายศุภจิต พันกมลศิลป์

วัน เดือน ปี 1 ตค 2541

เลขทะเบียน 038300

เลขเรียกหนังสือ T 54320 .ว86ค

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2539

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2539

ภาควิชาวิศวกรรมคอมพิวเตอร์

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง เครื่องจำลองการทำงานของอีพรอม แบบไร้สาย

WIRELESS EPROM EMULATOR

ผู้จัดทำ นายรุ่งโรจน์ เลิศรัตนศาสตร์ 37013304

นายวีระยุทธ จีวรมงคล 37013311

นายศุภจิต พันกมลศิลป์ 37013314

..... อาจารย์ที่ปรึกษา

(อ.สมศักดิ์ มิตะถา)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เครื่องจำลองการทำงานของอีพรอม แบบไร้สาย

รุ่งโรจน์ เลิศรัตนศาสตร์
วีระยุทธ จีวรมงคล
ศุภจิต พันกมลศิลป์
อ.สมศักดิ์ มิตะถา **อาจารย์ที่ปรึกษา**
ปีการศึกษา 2539

บทคัดย่อ

โครงงานนี้นำเสนอเครื่องจำลองการทำงานของ EPROM แบบไร้สาย ซึ่งเป็นอุปกรณ์ที่ใช้ในการจำลองการทำงานของ EPROM ในระบบไมโครคอนโทรเลอร์เป็นส่วนใหญ่ โดยจะทำหน้าที่ในการเก็บโปรแกรมที่ใช้ในการควบคุม และควบคุมการทำงานของอุปกรณ์ต่างๆ นอกจากนี้การบรรจุโปรแกรมที่เขียนไว้บนไมโครคอมพิวเตอร์ ลงในเครื่องจำลองการทำงาน EPROM แบบไร้สายนี้ สามารถทำได้สะดวก จึงมีความเหมาะสมกับการทดลองโปรแกรมที่ต้องมีการแก้ไขและพัฒนาบ่อยครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Wireless EPROM Emulator

Roongrod Lestratanasast
Weerayut Cheewornmongkhol
Suphajit Pankamonsil
Somsak Mitatha **Advisor**
1996

Abstract

This project propose a Wireless EPROM Emulator that is mostly used in EPROM's function simulation of microcontroller system. It is used to store controlling program and control device functions

Additionally it is easy to download the program from microcomputer to this EPROM Emulator . So it is suitable for testing the frequently modified and developed program

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

บทที่ 1 บทนำ

ในปัจจุบันนี้ เทคโนโลยีทางด้านไมโครคอนโทรลเลอร์ ได้พัฒนาไปอย่างรวดเร็ว ซึ่งมีประสิทธิภาพที่แตกต่างกันออกไป รวมทั้งยังขึ้นอยู่กับการนำไปใช้งานเฉพาะทางอีกด้วย ซึ่งส่วนประกอบสำคัญส่วนหนึ่งที่ใช้ในการพัฒนาอุปกรณ์ด้วยการใช้ ไมโครคอนโทรเลอร์ ก็คือส่วนเก็บข้อมูลถาวรซึ่งใช้ในการเก็บส่วนของโปรแกรมที่ใช้ในการประมวลผลของไมโครคอนโทรเลอร์ และยังใช้ในการเก็บข้อมูลถาวรของอุปกรณ์นั้นด้วยๆ (เช่น อัตราในการเก็บค่าบริการของการใช้โทรศัพท์ ในเครื่องคิดค่าโทรศัพท์อัตโนมัติ) ซึ่งในปัจจุบันที่นิยมนำมาใช้ทำหน้าที่ดังกล่าว คือ EPROM (Erasable Programmable Read Only Memory) และในการที่จะใส่ข้อมูลลงไปเก็บไว้ในตัว EPROM ได้นั้นจะต้องทำการล้างข้อมูลเก่าที่เก็บอยู่ภายในออกเสียก่อน จึงจะใส่ข้อมูลใหม่ลงไปได้

การล้างข้อมูลภายในออก สามารถทำได้โดยการฉายแสง Ultraviolet ผ่านช่องกระจกบนตัว EPROM เป็นเวลามากกว่า 3 นาทีโดยประมาณ ซึ่งจะทำให้เสียเวลามาก หากต้องล้างข้อมูลและใส่ข้อมูลใหม่บ่อย ๆ เช่นในช่วงของการพัฒนาโปรแกรม จะต้องทำการแก้ไขโปรแกรมและทดสอบหลายต่อหลายครั้งจนกว่าจะได้ผลเป็นที่น่าพอใจ และตัว EPROM ยังมีข้อจำกัดอยู่ที่จำนวนครั้งของการโปรแกรม จากข้อเสียของการใช้ EPROM ดังกล่าวจะเห็นได้ว่า ในการพัฒนาโปรแกรมของอุปกรณ์ต่าง ๆ ที่ต้องใช้ EPROM เป็นตัวเก็บข้อมูล จะทำให้เสียทั้งเวลาและตัว EPROM จากการที่ต้องแก้ไขข้อมูลที่จะเก็บลงในตัว EPROM บ่อย ๆ และในปัจจุบันได้มีบริษัทที่ผลิตอุปกรณ์สนับสนุนการพัฒนาอุปกรณ์ที่ใช้ไมโครคอนโทรเลอร์ ได้ผลิตอุปกรณ์ที่สามารถใช้แทน EPROM โดยใช้ชื่อว่า EPROM Emulator ซึ่งในยุคแรก ๆ นั้นมักใช้การติดต่อกับเครื่องคอมพิวเตอร์ผ่านทางพอร์ตของเครื่องพิมพ์ ซึ่งมีข้อเสียที่เป็นพอร์ตแบบขนาน ถึงแม้จะมีความเร็วสูงแต่จะจำกัดที่ความยาวของสายที่เชื่อมระหว่าง EPROM Emulator กับเครื่องคอมพิวเตอร์ ซึ่งจะจำกัดที่ประมาณ 2.5 เมตร ซึ่งเป็นการไม่สะดวกหากอุปกรณ์ที่กำลังพัฒนาจำเป็นต้องอยู่ห่างจากเครื่องคอมพิวเตอร์เกินกว่า 2.5 เมตร

ในภายหลังได้มีการพัฒนาของ EPROM Emulator ไปอีกระดับหนึ่งคือใช้การติดต่อกับเครื่องคอมพิวเตอร์ผ่านทางพอร์ตแบบอนุกรม ซึ่งทำให้สามารถใช้สายเชื่อมระหว่าง EPROM Emulator กับเครื่องคอมพิวเตอร์ได้ยาวถึงประมาณ 15 เมตร ถึงแม้จะเสียในด้านความเร็วของการส่งข้อมูลไปบ้าง แต่ก็ยังอยู่ในระดับที่ยอมรับได้ แต่ EPROM Emulator แบบนี้ก็มีราคาสูงและหากมีความจำเป็นต้องทำงานกับอุปกรณ์ที่อยู่ห่างออกไปมาก ๆ หรืออุปกรณ์ที่ต้องมีการเคลื่อนที่ก็ยังคงเกิดปัญหาขึ้นกับการใช้สายเชื่อมโยง

ในโครงงานนี้จึงได้มีการพัฒนา EPROM Emulator แบบไม่ใช้สายขึ้นมาเพื่อความสะดวกในการใช้งานในกรณีที่ต้องเคลื่อนที่อุปกรณ์ไปมา หรืออุปกรณ์อยู่ห่างมากจากเครื่องคอมพิวเตอร์ ซึ่งใช้ความถี่ในย่านความถี่วิทยุเป็นสัญญาณเพื่อส่งให้กับ EPROM Emulator นอกจากนี้บอร์ด EPROM Emulator ก็ยังคงสามารถรับส่งข้อมูลแบบใช้สายได้ด้วย

เอกสารนี้สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2 ทฤษฎีและหลักการเบื้องต้น

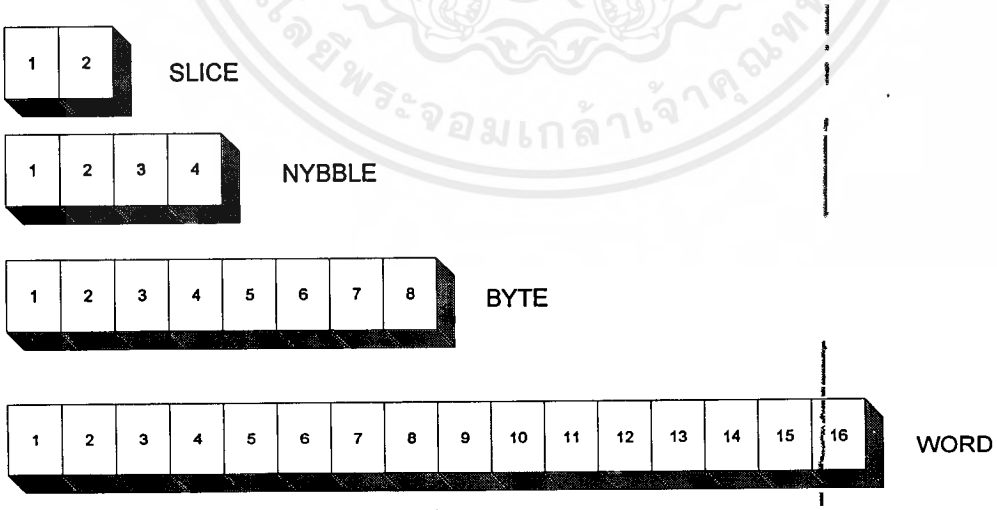
2.1 การสื่อสารข้อมูล

การสื่อสารข้อมูลคือ ขบวนการในการแลกเปลี่ยนข้อมูลหรือข่าวสาร ซึ่งในการสื่อสารกันนั้นต้องประกอบไปด้วย ผู้ส่ง (Sender) ผู้รับ (Receiver) และตัวกลางในการส่งข้อมูล (Medium) โดยที่ข้อมูลที่จะส่งนั้นเป็นสัญญาณดิจิตอล คืออยู่ในรูปของเลขฐานสอง ข้อมูลข่าวสารที่จะส่งจะเป็นรหัสตัวอักษร ตัวเลข หรือเครื่องหมายที่อยู่ในรูปเลขฐานสอง เช่น รหัสแอสกี (ASCII) หรือ รหัส (BCDIC)

ในการสื่อสารข้อมูลจะเป็นการรับส่งข้อมูลระหว่าง 2 จุดหมาย เช่นการเชื่อมโยงระหว่างคอมพิวเตอร์ 2 เครื่องเข้าด้วยกันในระยะทางใกล้ๆ ด้วยสายธรรมดา ข่าวสาร หรือ Encoded Information คือ ข้อมูลที่ถูกถ่ายโอนไปในรูปของสัญญาณไฟฟ้าจะถูกส่งออกไปโดยผู้ส่ง เช่นเมื่อเรากดตัวอักษร D บนคีย์บอร์ดซึ่งมีรหัสเลขฐานสองคือ 01000100 จะถูกส่งผ่านสายนำสัญญาณไปยังผู้รับ โดยจะทำการแปลงรหัสเลขฐานสองนี้ให้กลับเป็นตัวอักษร D ซึ่งข้อมูลที่ส่งนี้จะส่งในรูปของสัญญาณอะนาลอก หรือดิจิตอลก็ได้

2.1.1ประเภทของการส่งข้อมูล

โดยทั่วๆ ไปหลักใหญ่ของการส่งข้อมูลในคอมพิวเตอร์หรือระหว่างคอมพิวเตอร์ด้วยกันจะมีลักษณะการส่งข้อมูลอยู่ 2 แบบ คือ การส่งข้อมูลแบบขนาน และการส่งข้อมูลแบบอนุกรม คำสั่งหรือข้อมูลอยู่ในรูปของบิท คือ หลาย ๆ บิตประกอบกันเป็นคำ ๆ หนึ่ง (word) หรือคำสั่งหนึ่งๆ ดังในรูป 2.1 ได้แสดงถึงกลุ่มของบิทที่มีการใช้งานในไมโครคอมพิวเตอร์ จึงได้มีการกำหนดลักษณะมาตรฐานของข้อมูลคือ



รูปที่ 2.1 แสดงรูปแบบของข้อมูล

ถ้าข้อมูลหนึ่งตัวเมื่อแปลงให้อยู่ในรูปของเลขฐานสองแล้วประกอบด้วย 4 บิท เราเรียกว่า 4 บิทไมโครหรือ 1 นิบเบิล (nybble) และถ้าข้อมูลประกอบไปด้วยกลุ่มของบิทที่มี 8 บิท เราเรียกกว่าเป็น 1 ไบท์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

(byte) แต่ในระบบอื่นอาจจะมี 16 บิท หรือ 32 บิท เป็น 1 ไบท์ก็ได้ ดังนั้นเมื่อรู้ลักษณะของข้อมูลแล้ว ก็จะมาดูข้อแตกต่างของการส่งข้อมูลแบบขนานและแบบอนุกรม

2.1.1.1 การส่งข้อมูลแบบขนาน

ลักษณะของการส่งข้อมูลแบบขนาน หรือเรียกอีกอย่างหนึ่งว่า Parallel Interface ทำได้โดยการส่งข้อมูลออกมาทีละไบท์ ซึ่งถ้าใน 1 ไบท์มี 8 บิท ทั้ง 8 บิทจะถูกส่งออกจากอุปกรณ์ส่งไปอุปกรณ์รับพร้อมกัน และช่องสัญญาณในการส่งข้อมูลจะต้องมีอย่างน้อย 8 ช่องสัญญาณสำหรับสัญญาณแต่ละบิท พร้อมกับมีสัญญาณควบคุมอีกหลายเส้น ช่องสัญญาณในการส่งจะใช้สายเคเบิลแบบที่มีตัวนำหลายเส้น อาจจะเป็นสายเคเบิลชนิดแบนหรือกลมก็ได้ โดยที่ระยะทางระหว่างเครื่องทั้งสองไม่ควรมากเกินไป เนื่องจากสาเหตุต่าง ๆ หลายสาเหตุ เช่นการที่สัญญาณการที่สัญญาณถูกลดทอนไปกับความต้านทานของสาย การผิดเพี้ยนของสัญญาณเนื่องจากสภาพความเป็นตัวเก็บประจุภายในสาย การเดินทางมาถึงอุปกรณ์ไม่พร้อมกันในแต่ละบิท เพราะสภาพความไม่สมบูรณ์ของตัวนำในแต่ละเส้นภายในสายเคเบิล และการที่ระดับของกราวนด์ทางไฟฟ้าที่อุปกรณ์รับผิดไปจากอุปกรณ์ส่ง สาเหตุเหล่านี้อาจจะทำให้เกิดการผิดพลาดของข้อมูลขึ้นได้ นอกจากสายที่เป็นทางเดินของข้อมูลแล้ว อาจจะมีการเดินทางของสัญญาณควบคุมต่าง ๆ อีก เช่น สายที่ใช้ในการควบคุมในการโต้ตอบ (Handshake) ซึ่งการส่งข้อมูลแบบขนานจะใช้ในการส่งข้อมูลจากคอมพิวเตอร์ไปยังเครื่องพิมพ์เป็นส่วนใหญ่ มีข้อดีคือสามารถส่งข้อมูลได้อย่างรวดเร็วและเป็นจำนวนมาก ข้อเสียคือ ไม่เหมาะที่จะนำมาใช้ส่งข้อมูลระยะทางไกล ๆ เนื่องจากค่าใช้จ่ายในเรื่องของสายสำหรับส่งข้อมูลมีราคาแพงเกินไป

2.1.1.2 การส่งข้อมูลแบบอนุกรม

ในการส่งข้อมูลแบบอนุกรมนี้จะมีความแตกต่างจากการส่งข้อมูลแบบขนาน การส่งข้อมูลแบบอนุกรมนี้เรียกอีกอย่างหนึ่งว่า Serial Interface จะมีลักษณะของการส่งข้อมูลออกจากพอร์ตเรียงกันออกไปทีละบิทจากทางด้านส่ง ส่วนทางด้านรับนั้นจะรับข้อมูลเข้ามาทีละบิทแล้วรวมกันเป็นไบท์ ซึ่งทางด้านอุปกรณ์รับจะต้องคอยตรวจสอบว่าบิทใดเป็นบิทเริ่มแรกของไบท์นั้น การตรวจสอบขึ้นอยู่กับรูปแบบของรหัสของบิทที่ใช้

การส่งข้อมูลแบบอนุกรมมี 2 แบบคือ

ก.การส่งข้อมูลแบบอะซิงโครนัส (Asynchronous Transmission)

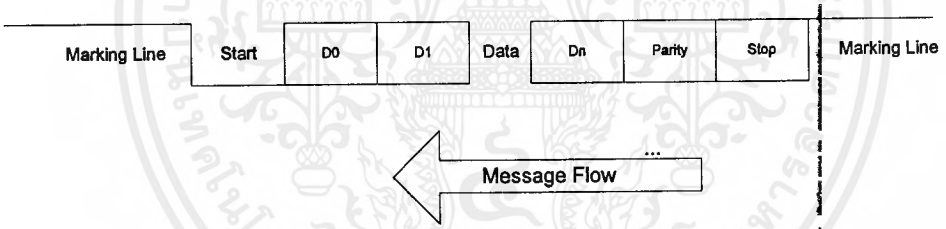
ในการส่งข้อมูลแบบอะซิงโครนัส การส่งข้อมูลแต่ละตัวอักษรไม่มีกำหนดเวลาที่แน่นอน คือ แต่ละตัวอักษรจะอยู่ห่างกันเท่าไหร่ก็ได้ หรือจะส่งติดกันไปตลอดก็ได้เช่นกันดังนั้นเพื่อให้ผู้รับแยกออกได้ว่าข้อมูลแต่ละตัวเริ่มต้นเมื่อใด ในการส่งข้อมูลแต่ละตัวหรือแต่ละไบท์นั้นจะมีสัญญาณสำหรับตรวจสอบบิทแรกภายในตัวของมันเอง โดยแต่ละไบท์จะถูกเพิ่มด้วยบิทเริ่มต้น (Start Bit) นำหน้าไบท์นั้น และบิทสิ้นสุด (Stop Bit) ตามหลังไบท์นั้น ซึ่งอาจจะมีการเพิ่มบิท พาริตี้ (Parity Bit) ก่อนบิทสิ้นสุดก็ได้ ดังนั้นระยะเวลาระหว่างข้อมูลแต่ละไบท์ก็ไม่จำเป็นต้องแน่นอน เพราะอุปกรณ์รับจะตรวจสอบบิทแรกที่ละไบท์เท่านั้น โดยขณะไม่มีการส่งข้อมูล สภาพลอจิกจะเป็น “1” อุปกรณ์รับจะคอยตรวจสอบการเปลี่ยนลอจิกจาก “1” เป็น “0” เมื่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

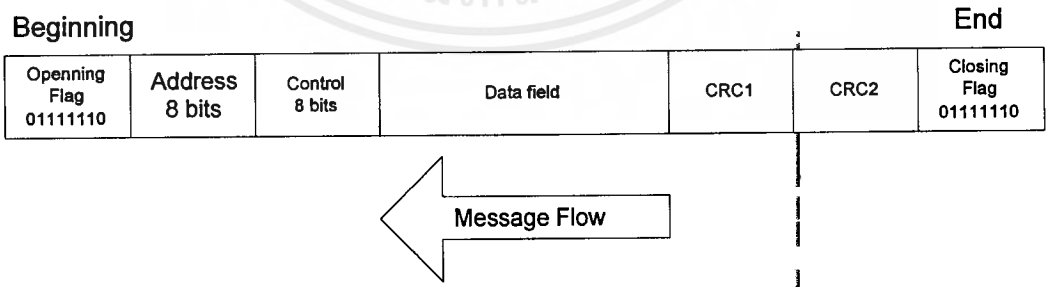
กำหนดให้บิทเริ่มต้นมีลอจิกเป็น "0" ซึ่งหมายถึงบิทที่ตามหลังมาคือบิทแรกของไบท์นั้น รูปแบบการจัดเรียงบิทในการส่งแบบอะซิงโครนัสแสดงดังรูป 2.2

ข.การส่งข้อมูลแบบซิงโครนัส (Syncronous Transmission)

การส่งข้อมูลแบบซิงโครนัสหมายถึง การส่งข้อมูลแบบอนุกรมที่มีการกำหนดจำนวนของอักขระที่จะส่งในแต่ละครั้งเป็นจำนวนที่แน่นอนเรียกว่า เฟรมข้อมูล การส่งข้อมูลแบบนี้จะต้องมีการส่งสัญญาณนาฬิกาไปพร้อมๆ กับสัญญาณข้อมูล ในการส่งข้อมูลระยะสั้นๆ สัญญาณนาฬิกาซึ่งใช้เป็นสัญญาณซิงค์อาจจะส่งแยกไปในสายส่งสัญญาณอีกเส้นหนึ่งไม่ส่งรวมไปในสายส่งข้อมูลก็ได้ แต่ถ้าเป็นการส่งในระยะไกลๆ แล้ว สัญญาณนาฬิกาจะถูกเข้ารหัสส่งรวมไปกับสัญญาณข้อมูลในสายส่งเส้นเดียวกัน การส่งแบบซิงโครนัสข้อมูลจะเรียงติดกันไปโดยไม่มีบิทเริ่มต้นและบิทสุดท้ายของข้อมูลบล็อกหนึ่งๆ (บล็อกหนึ่งๆ ประกอบด้วยข้อมูลหลายชุด) จะแสดงถึงจุดเริ่มต้นและจุดสิ้นสุดของข้อมูลเท่านั้น เพราะฉะนั้นถ้าเป็นการส่งอนุกรมแบบอะซิงโครนัสเราจะเพิ่ม Framing Bits รวมเข้าไปในแต่ละคาแรคเตอร์ และถ้าเป็นการส่งข้อมูลอนุกรมแบบซิงโครนัสเราจะเพิ่ม Framing Characters เข้าไปร่วมกับบล็อกข้อมูลแต่ละบล็อก ซึ่งแสดงความแตกต่างได้ดังรูปที่ 2.2 และ 2.3



รูปที่ 2.2 การส่งข้อมูลอนุกรมแบบอะซิงโครนัส



รูปที่ 2.3 การส่งข้อมูลอนุกรมแบบซิงโครนัส

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ตารางที่ 2.1 ข้อเปรียบเทียบระหว่างการส่งข้อมูลแบบขนานและแบบอนุกรม

	แบบขนาน	แบบอนุกรม
1.ระยะทาง	ปกติจะน้อยกว่า 100 ฟุต	ส่งได้ตั้งแต่ระยะทางสั้นๆ จนถึงระยะทางเป็นไมล์
2.ความเร็ว	อัตราความเร็วสูงมากในระยะทางที่ไม่ไกลมากนัก กำหนดได้เป็นจำนวนบิทต่อวินาที	อัตราความเร็วของข้อมูลที่ใช้กันอยู่ทั่วไปจะอยู่ในช่วง 0 ถึง 2 ล้านบิทต่อวินาที
3.ระดับของสัญญาณ	ในการอินเตอร์เฟซจะใช้ระดับของสัญญาณที่ใช้กับอุปกรณ์ TTL คือ สัญญาณลอจิก 1 และ 0 จะแทนด้วยระดับแรงดัน +5V และ 0V ตามลำดับ	ใช้มาตรฐานของ EIA-RS232C คือ มีระดับสัญญาณไฟฟ้าขนาด 12V หรืออาจจะใช้ มาตรฐาน 20 mA current loop หรืออาจจะใช้ระดับสัญญาณของ TTL ก็ได้ (ใช้กันน้อยมาก)
4.ความผิดพลาดของสัญญาณ	ถ้าส่งในระยะทางไกลๆความผิดพลาดของข้อมูลจะเกิดขึ้นได้ง่าย	การผิดพลาดของสัญญาณจะมีน้อยลง
5.ค่าใช้จ่าย	ถ้าส่งในระยะทางไกลๆ จะสิ้นเปลืองค่าใช้จ่ายมากเพราะต้องใช้สายส่งสัญญาณหลายเส้น	สิ้นเปลืองน้อยกว่าหลายเท่า ถึงแม้ว่าจะใช้อุปกรณ์เปลี่ยนสัญญาณของข้อมูลจากแบบขนานไปเป็นแบบอนุกรมแล้วส่งผ่านสายส่งใช้อุปกรณ์ในการแปลงสัญญาณกลับมาเป็นแบบขนานอีกยังลงทุนน้อยกว่า

2.1.2 ทิศทางของการส่งผ่านข้อมูล

ในการรับส่งข้อมูลระหว่างคอมพิวเตอร์นั้น อาจจะแบ่งตามลักษณะของการรับส่งได้เป็น 3 วิธีใหญ่ ๆ คือ

2.1.2.1 การรับส่งทางเดียว (Simplex)

เป็นการส่งข้อมูลแบบไปในทางเดียว เช่นการส่งข้อมูลจากคอมพิวเตอร์ A ไปยังเทอร์มินอล B ในกรณีนี้คอมพิวเตอร์จะต้องเป็นเครื่องส่ง และเทอร์มินอล B จะเป็นเครื่องรับเท่านั้น ตัวอย่างของการส่งข้อมูลแบบนี้คือ การส่งข้อมูลจากคอมพิวเตอร์ไปยังเครื่องพิมพ์ การส่งกระจายเสียงทางวิทยุหรือโทรทัศน์ การสื่อสารแบบ Simplex นี้เรามักจะไม่นำมาใช้ในการสื่อสารข้อมูล เนื่องจากว่าเราจำเป็นที่จะต้องมีการโต้ตอบกันในขณะทำการรับส่งข้อมูล หรือบางทีก็เปลี่ยนจากผู้รับเป็นผู้ส่งซึ่งทำไม่ได้ในการส่งแบบนี้

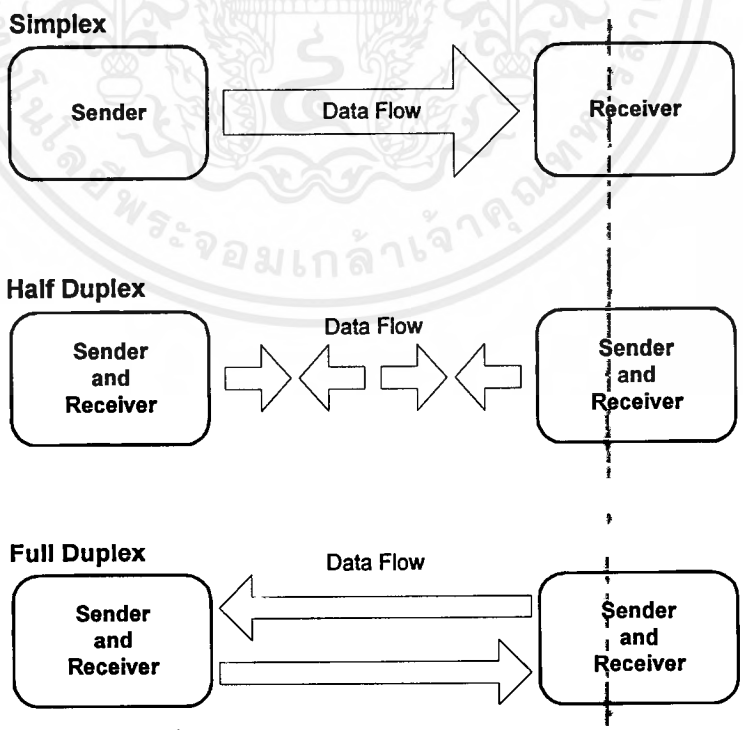
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.2.2 การรับส่งแบบผลัดกันส่ง (Half Duplex)

เป็นการรับส่งข้อมูลใน 2 ทิศทางสามารถทำหน้าที่เป็นเครื่องรับ และเครื่องส่งได้แต่ว่าต้องผลัดกันรับผลัดกันส่ง เมื่อฝ่ายใดฝ่ายหนึ่งส่งอีกฝ่ายหนึ่งจะทำหน้าที่รับจนกระทั่งฝ่ายแรกส่งจบ ฝ่ายหลังจึงจะทำหน้าที่เป็นฝ่ายส่งได้ และฝ่ายส่งในตอนแรกก็จะไปทำหน้าที่เป็นฝ่ายรับสลับกันเรื่อยไป ทั้งสองฝ่ายจะเป็นผู้ส่งพร้อมกันไม่ได้เพราะสัญญาณจะชนกัน อุปกรณ์ที่ใช้ในการติดต่อแบบ Half Duplex ได้แก่ ระบบวิทยุมือถือ (Walkie Talkie) ระบบ ATM และ Intercom เป็นต้น

2.1.2.3 การรับส่งสองทางได้พร้อมกัน (Full Duplex)

เป็นการรับส่งข้อมูลใน 2 ทิศทางพร้อมกัน เช่น คอมพิวเตอร์ A ส่งข้อมูลไปให้คอมพิวเตอร์ B ซึ่งเป็นเวลาเดียวกับที่คอมพิวเตอร์ B ส่งข้อมูลไปให้กับคอมพิวเตอร์ A ซึ่งทั้งคอมพิวเตอร์ A และ B ต่างก็ทำงานเป็นอิสระต่อกัน และสายส่งข้อมูลที่ใช้ในการติดต่อระหว่างคอมพิวเตอร์ A และ B จะประกอบไปด้วยสายส่ง 3 เส้นคือ สายสัญญาณรับ/ส่ง และสายกราวนด์ โดยระบบของคอมพิวเตอร์ A และ B จะใช้สายกราวนด์ร่วมกันจะเรียกว่าเป็นระบบฟูลดูเพล็กซ์ 4 เส้น (Full Duplex 4-wires) ส่วนอีกระบบหนึ่งจะใช้สายสัญญาณรับและส่งร่วมกัน และในระบบจะมีสายส่งสัญญาณเพียง 2 เส้น ในลักษณะนี้เรียกว่าเป็นระบบฟูลดูเพล็กซ์ 2 เส้น (Full Duplex 2-wires) แต่ในระบบฟูลดูเพล็กซ์ 2 เส้นนี้ จะต้องอาศัยเทคนิคการแบ่งความถี่เข้ามาช่วย คือ จะส่งความถี่ในช่วงหนึ่ง และจะรับความถี่ในช่วงหนึ่ง



รูปที่ 2.4 การส่งข้อมูลในลักษณะต่าง ๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.1.3 อัตราบิทและอัตราบอด (Bit Rate Versus Baud Rate)

อัตราบิท (Bite rate) เป็นการส่งข้อมูลแบบอนุกรมจากเครื่องคอมพิวเตอร์เครื่องหนึ่งไปยังอุปกรณ์ต่อพ่วง หรือไปยังเครื่องคอมพิวเตอร์อีกเครื่องหนึ่ง วัดเป็นจำนวนบิทต่อวินาที อาจมีค่าไม่เท่ากับอัตราการเปลี่ยนแปลงในสายส่ง (Baud Rate) ก็ได้

อัตราบอด (Baud Rate) เป็นอัตราการเปลี่ยนแปลงของสัญญาณอะนาล็อกในสายส่ง ซึ่งอาจจะเป็นอัตราการเปลี่ยนแปลงของความถี่ อัตราการเปลี่ยนแปลงขนาดของสัญญาณ หรืออัตราการเปลี่ยนช่วงต่อเนื่องของมุม (Phase) ในหนึ่งวินาทีก็ได้ และอัตราบอดไม่จำเป็นต้องมีค่าเท่ากับอัตราการส่งข้อมูล ซึ่งวัดเป็น บิทต่อวินาที (Bit Per Second) โดยอัตราบิทอาจจะมีค่าน้อยกว่าหรือมากกว่าอัตราการส่งข้อมูลก็ได้ ขึ้นอยู่กับเทคนิคการผสมสัญญาณที่ใช้

ในสมัยก่อน การรับส่งข้อมูลใช้เทคนิคการผสมสัญญาณแบบง่าย ๆ เช่นการเปลี่ยนแปลงความถี่ตามข้อมูล "0" และ "1" ที่ได้รับ อัตราการส่งข้อมูลและอัตราการเปลี่ยนแปลงของสัญญาณในสายส่งมีค่าเท่ากัน จึงถือว่าอัตราการเปลี่ยนแปลงของสัญญาณในสายส่ง (Baud Rate) คืออัตราการส่งข้อมูลนั่นเอง ต่อมาเทคนิคของการผสมสัญญาณซับซ้อนมากขึ้น ทำให้สามารถส่งข้อมูลด้วยความเร็วสูง โดยอัตราการเปลี่ยนแปลงในสายยังคงเท่าเดิม ดังนั้นเมื่อพูดว่าโมเด็มรับส่งข้อมูลที่ความเร็ว 9600 บ็อด จะไม่ทราบเลยว่า โมเด็มนั้นรับส่งข้อมูลได้กี่บิทต่อวินาที เนื่องจากว่าถ้าโมเด็มผสมสัญญาณ 1 บิทต่อหนึ่งลูกคลื่น โมเด็มนั้นจะรับส่งข้อมูลที่ความเร็ว 9600 บิทต่อวินาที หรือ ถ้าโมเด็มผสมสัญญาณ 2 บิทต่อหนึ่งลูกคลื่นที่เปลี่ยนแปลงในสายส่ง โมเด็มจะรับส่งข้อมูลได้เร็วถึง 19200 บิทต่อวินาที โดยยังมีอัตราการเปลี่ยนแปลงในสายส่งเท่ากับ 9600 บอดเหมือนเดิม ซึ่งจะสามารถหาอัตราบิทได้ดังนี้

อัตราบิท (Bit Rate) = อัตราบอด (Baud Rate) x บิทใน 1 บอด

2.2 พอร์ต RS-232

พอร์ต RS-232 นี้ทำหน้าที่รับส่งข้อมูลในแบบอนุกรม เรียกชื่อกันว่า Universal Asynchronous Adapter ซึ่งหน้าที่ของ พอร์ต RS-232 นี้ถ้าจะแบ่งเป็นการรับและส่งแล้วก็จะสามารถแบ่งหน้าที่ออกได้เป็น 2 อย่างคือ หน้าที่ในการรับข้อมูลและหน้าที่ในการส่งข้อมูล

หน้าที่ในการรับข้อมูล

- เปลี่ยนสัญญาณเข้ามาแบบอนุกรมให้เป็นแบบขนาน
- ตรวจสอบความผิดพลาดของสัญญาณที่รับ
- ตัดบิทสิ้นสุดและพาริตี้ออก
- ส่งสัญญาณให้ CPU รู้ว่ารับสัญญาณไว้แล้ว

หน้าที่ในการส่งข้อมูล

- เปลี่ยนสัญญาณแบบขนานจาก CPU ให้เป็นสัญญาณแบบอนุกรม

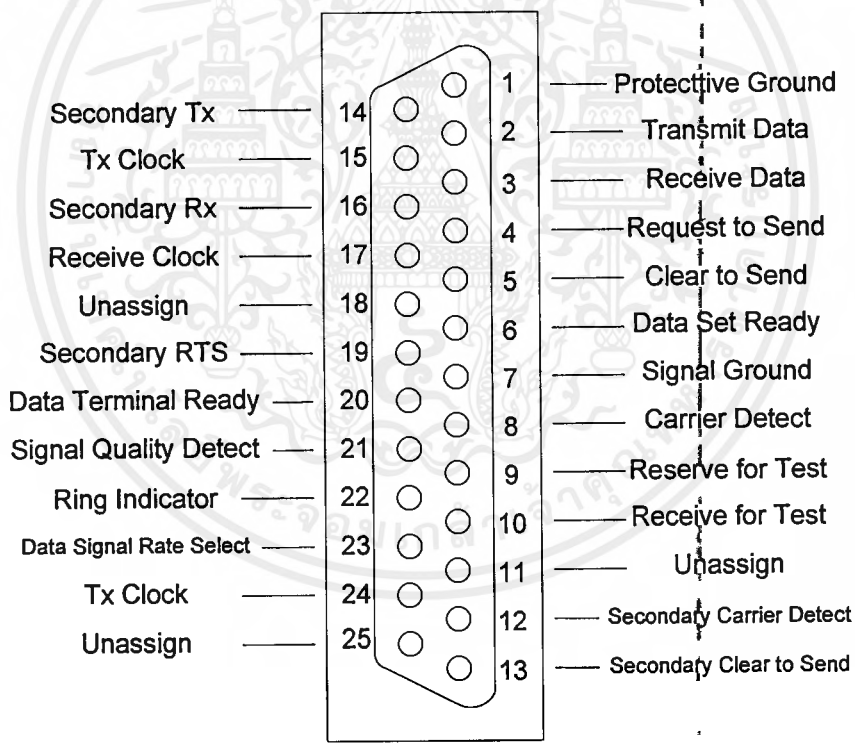
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- เพิ่มบิทสิ้นสุดและพาริตี้บิท
- เพิ่มสัญญาณควบคุมโมเด็มที่ต่อเชื่อม (ถ้ามี)

พอร์ต RS-232 สามารถเชื่อมต่อการโอนถ่ายข้อมูลได้จาก 0-20,000 บิทต่อวินาที ซึ่งเพียงพอสำหรับไมโครคอมพิวเตอร์ที่มีขนาดอัตราบอด 110 ถึง 9600 บอด ความยาวของสายเชื่อมต่อสัญญาณตามมาตรฐานของ RS-232 มีข้อจำกัดอยู่ 50 ฟุต

2.2.1 การกำหนดจุดข้อต่อของ RS-232

มาตราฐานของพอร์ต RS-232 กำหนดข้อต่อแบบ DB-25 ซึ่งตำแหน่งของแต่ละขากำหนดไว้ดังรูปที่ 2.5



รูปที่ 2.5 การกำหนดของข้อต่อ RS-232C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

หน้าที่ของแต่ละขาสัญญาณมีดังนี้

- *Transmit Data* (TD ขาที่ 2) เป็นสัญญาณที่ส่งออกมาจากคอมพิวเตอร์ ไปยังอุปกรณ์ตัวอื่น เมื่อไม่มีสัญญาณส่งออก สถานภาพของลอจิกที่เข้าขานี้จะมีค่าเท่ากับ "1" หรือเทียบเท่ากับบิทสิ้นสุด
- *Receive Data* (RD ขาที่ 3) เป็นขาของสัญญาณที่เข้าไปยัง DTE หรือไมโครคอมพิวเตอร์ เมื่อไม่มีสัญญาณเข้ามา ขานี้จะมีสภาพลอจิกเป็น "1"
- *Request to Send* (RTS ขาที่ 4) เป็นขาที่ใช้ร้องขอที่จะส่งสัญญาณ ซึ่งขานี้จะใช้คู่กับ CTS หากอุปกรณ์รับได้รับสัญญาณ RTS แล้วจะทำการตรวจสอบสถานภาพของตัวเองว่าพร้อมจะรับสัญญาณหรือไม่ หากพร้อมก็จะส่งสัญญาณมาที่สาย CTS
- *Clear to Send* (CTS ขาที่ 5) เมื่ออยู่ในสถานภาพทางลอจิกเป็น "1" แสดงว่าอุปกรณ์พร้อมที่จะรับข้อมูลแล้ว
- *Data Set Ready* (DSR ขาที่ 6) เมื่ออยู่ในสถานภาพทางลอจิกเป็น "0" เป็นการบอกฝ่ายส่งว่าพร้อมที่จะทำการส่งข้อมูลแล้ว
- *Signal Ground* (SG ขาที่ 7) ทำหน้าที่เป็นระดับแรงดันอ้างอิงสำหรับทุกๆ สายของสัญญาณจะมีแรงดันเป็น "0" เมื่อเทียบกับสัญญาณตัวอื่น
- *Carrier Detect* (CD ขาที่ 8) ทำหน้าที่ส่งสัญญาณไปบอกไมโครคอมพิวเตอร์ว่าได้รับสัญญาณจากอีกฝ่ายหนึ่งแล้ว สัญญาณจะอยู่ในสถานภาพลอจิก "0"
- *Data Terminal Ready* (DTR ขาที่ 20) เป็นสัญญาณที่คอมพิวเตอร์แสดงว่าพร้อมที่จะติดต่อกับอุปกรณ์

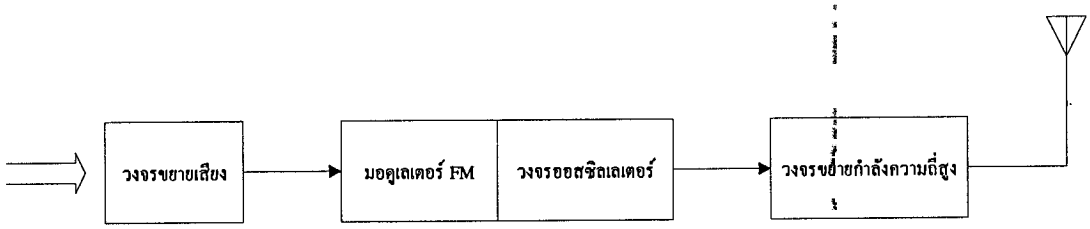
2.3 ทฤษฎีการสื่อสารแบบ Wireless

ในการสื่อสารข้อมูลแบบไม่ใช้สายนี้สามารถทำได้หลายวิธี เช่น ใช้แสง Infrared , ส่งผ่านความถี่วิทยุ เป็นต้น การใช้แสง Infrared นั้นไม่ค่อยสะดวกนักเนื่องจากจะถูกบังคับด้วยทิศทางของแสง และแสงยังไม่สามารถผ่านวัตถุทึบแสงได้อีกด้วย เพราะฉะนั้นการสื่อสารข้อมูลผ่านคลื่นความถี่วิทยุ จึงสะดวกต่อการใช้งานมากกว่า

2.3.1 เครื่องส่งสัญญาณวิทยุแบบ FM

จากแผนผังของเครื่องส่ง FM ในรูปที่ 2.6ก สัญญาณเสียงผ่านการขยายแล้วป้อนเข้าสู่มอดูเลเตอร์ วงจรมอดูเลเตอร์นี้จะทำการเปลี่ยนความถี่ของออสซิเลเตอร์ โดยมีช่วงความถี่เบี่ยงเบนและอัตราการเบี่ยงเบนขึ้นอยู่กับแอมพลิจูดและความถี่ของสัญญาณเสียงตามลำดับ พาหะ FM ที่ถูกมอดูเลตแล้วจะถูกขยายโดยภาคขยายกำลังสุดท้ายป้อนสู่สายอากาศเพื่อส่งออกอากาศต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.6ก แผนผังเครื่องส่ง FM อย่างง่าย

เครื่องส่งที่กล่าวมาข้างต้นอาจเกิดปัญหาเมื่อต้องการส่งออกอากาศที่ความถี่สูงๆ เช่น เครื่องส่งกระจายเสียง FM (ซึ่งมีความถี่อยู่ระหว่าง 88 ถึง 108 เมกะเฮิรตซ์) ทำงานที่ความถี่สูง ทำให้ยากต่อการควบคุมให้ความถี่คงที่ นอกจากนี้การควบคุมการเบี่ยงเบนความถี่ก็ทำได้ยากขึ้นด้วย วิธีแก้ปัญหาดังกล่าวสามารถทำได้หลายวิธีแตกต่างกันออกไป

รูปที่ 2.6ข. แผนผังเครื่องส่ง FM แบบคูณความถี่

ในรูปที่ 2.6ข แสดงการใช้ความถี่ของออสซิลเลเตอร์ 8 เมกะเฮิรตซ์ และมัลติพลาย (หรือคูณ) ความถี่ขึ้นไปเป็น 96 เมกะเฮิรตซ์ การคูณความถี่นี้สามารถทำได้โดยใช้วงจรมัลติพลาย หลักการของวงจรมัลติพลายก็คือ ใช้คุณสมบัติ *ความไม่ลิเนียร์ของวงจรขยาย* ซึ่งจะทำให้เกิดสัญญาณ ฮาร์มอนิกออกมาจำนวนมาก จากนั้น วงจรแทงค์ที่เอาท์พุตจะจูนเอาเฉพาะความถี่ฮาร์มอนิกที่ต้องการไปใช้ประโยชน์ โดยทั่วไปวงจรมัลติพลายมักเป็นชนิดคูณ 2 (เรียกว่าดับเบลอร์หรือ doubler) หรือชนิดคูณ 3 (เรียกว่าทริเปลอร์หรือ tripler) ในที่นี้เราจะใช้วงจรคูณ 3 จำนวน 1 วงจร และวงจรคูณ 2 อีก 2 วงจร นั่นคือ 3x2x2 = 12 เท่า ฉะนั้นความถี่เอาท์พุตจะเป็น 8 เมกะเฮิรตซ์ คูณ 12 จะได้ 96 เมกะเฮิรตซ์

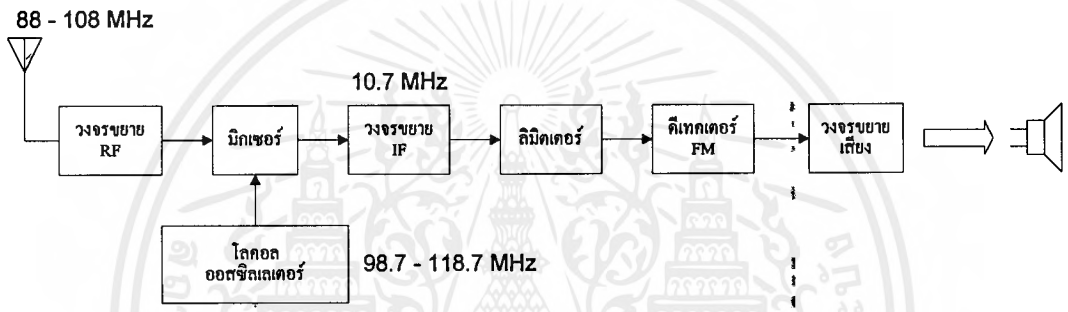
ช่วงความถี่เบี่ยงเบนของสัญญาณวิทยุกระจายเสียง FM เท่ากับ $\pm$ 75 กิโลเฮิรตซ์ ฉะนั้นเอาท์พุตจะต้องมีความถี่เบี่ยงเบนไปเท่ากับค่านี้ เมื่อสัญญาณเสียงมอดูเลต (แบบ FM) อย่างไรก็ดีการมัลติพลายความถี่จะทำให้ปริมาณความถี่เบี่ยงเบนถูกคูณให้กว้างขึ้นไปด้วย เช่น ออสซิเลเตอร์ 8 เมกะเฮิรตซ์ เบี่ยงเบนอยู่ระหว่าง 7.9 ถึง 8.1 เมกะเฮิรตซ์ ($\pm$ 0.1 เมกะเฮิรตซ์) เมื่อคูณ 12 เท่า พาหะมีความถี่กลางเป็น 96 เมกะเฮิรตซ์และเบี่ยงเบนอยู่ระหว่าง 94.8 ถึง 97.2 เมกะเฮิรตซ์ ($\pm$ 1.2 เมกะเฮิรตซ์) ดังนั้นถ้าหากเราต้องการให้ความถี่เบี่ยงเบนเป็น $\pm$ 75 กิโลเฮิรตซ์ที่เอาท์พุต ความถี่ออสซิลเลเตอร์จะต้องเบี่ยงเบนไปเท่ากับ $\pm$ 75 $\div$ 12 = 6.25 กิโลเฮิรตซ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ข้อดีอีกประการหนึ่งของระบบ FM ก็คือวงจรขยายกำลัง (Power Amplifier หรือ PA) สามารถทำงานในคลาส C ซึ่งมีประสิทธิภาพสูงกว่า ทั้งนี้เพราะแอมพลิจูดของสัญญาณ FM คงที่ไม่มีผลทำให้ข่าวสารเพี้ยนแม้จะมีการขลิบยอดสัญญาณ ข่าวสารนั้นอยู่ในช่วงความเปลี่ยนแปลงความถี่ของสัญญาณ FM เท่านั้น

2.3.2 เครื่องรับสัญญาณวิทยุแบบ FM

สำหรับเครื่องรับ FM แล้วมักจะใช้ความถี่ IF (Intermediate Frequency) 10.7 เมกะเฮิรตซ์ เพื่อกำจัดสัญญาณเงา และเพื่อให้แบนด์วิดท์ของวงจรกว้างพอที่จะรับสัญญาณ FM ได้ ความถี่เบี่ยงเบนของสัญญาณ FM ที่ส่งมาจากเครื่องส่งมีค่า $\pm$ 75 กิโลเฮิรตซ์ ดังนั้นแบนด์วิดท์ของเครื่องรับจะต้องมีค่า 150 กิโลเฮิรตซ์ เป็นอย่างน้อย ปกติมักจะเผื่อให้กว้างอีกเล็กน้อยเป็น 180 ถึง 200 กิโลเฮิรตซ์



รูปที่ 2.7 แผนผังเครื่องรับ FM

สมมติว่าปกติจูนเครื่องรับไว้ที่ 100 เมกะเฮิรตซ์ ลูกบิดหน้าปัดจะเลื่อนไปตรงกับความถี่ 100 เมกะเฮิรตซ์ (บนหน้าปัด) วงจรขยาย RF จะจูนไว้ที่ความถี่ 100 เมกะเฮิรตซ์ ส่วนออสซิลเลเตอร์จะจูนไว้ที่ 110.7 เมกะเฮิรตซ์ เมื่อผ่านกรรมวิธีเฮตเทอโรดายน์ในวงจรมิกเซอร์ ผลต่างความถี่จะปรากฏที่อินพุตของวงจรขยาย IF เท่ากับ 110.7 -100 เมกะเฮิรตซ์ = 10.7 เมกะเฮิรตซ์ สัญญาณความถี่ที่ IF นี้จะถูกขยายและกำจัดแบนด์วิดท์ให้กว้างพอที่จะรับสัญญาณ FM และแคบพอที่จะกำจัดสัญญาณที่ไม่ต้องการอื่นๆ ออกไป

ถ้าพาหะ FM ที่ส่งออกจากเครื่องมีความถี่เบี่ยงเบนเท่ากับ $\pm$ 50 กิโลเฮิรตซ์ โดยความถี่ FM เท่ากับ 100 เมกะเฮิรตซ์คงเดิม โลคอลออสซิลเลเตอร์คงเดิม และ IF คงเดิม สัญญาณ IF จะมีความถี่เบี่ยงเบนเท่ากับ $\pm$ 50 กิโลเฮิรตซ์ด้วย ฉะนั้นสัญญาณที่มอดูเลตมาบนพาหะจะยังคงอยู่ในสัญญาณ IF โดยไม่มีความเพี้ยน แม้ว่าสัญญาณ FM จะลดทอนจาก 100 เมกะเฮิรตซ์เหลือ 10.7 เมกะเฮิรตซ์

2.3.3 โมเด็มและมาตรฐานของโมเด็ม

การรับส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์ในระยะทางไม่ไกลมากนัก อาจใช้การรับส่งแบบอนุกรม (RS-232C) ซึ่งส่งข้อมูลดิจิตอลของคอมพิวเตอร์ไปตามสายจนถึงผู้รับได้ ในกรณีนี้สามารถรับส่งข้อมูลได้ไกลถึง 35 เมตรตามคุณสมบัติของ RS-232C หรือถ้าสายเคเบิลที่ใช้ดีอาจส่งได้ไกลถึง 150 เมตรที่ความเร็ว 9600 บิตต่อวินาที แต่สำหรับระยะทางที่ไกลมากๆ เช่น หลายสิบกิโลเมตร หรือหลายร้อยกิโลเมตร การส่งข้อมูลแบบดิจิตอลออกไปโดยตรงจะเกิดความไม่เหมาะสมหลายอย่าง ปัญหาที่สำคัญก็คือ คลื่นรูปสี่เหลี่ยมของสัญญาณดิจิตอล เมื่อส่งไปไกลๆ จะเพี้ยนหรือมีรูปร่างผิดไปจากเดิมได้ง่าย ทำให้สายส่งและวงจรรับสัญญาณ

เอกสารนี้เป็นเอกสารที่สงวนลิขสิทธิ์ ห้ามนำไปใช้ในการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ดิจิตอลต้องถูกออกแบบมาเป็นอย่างดี ราคาของสายส่งสัญญาณแบบดิจิตอลจึงมีราคาแพงกว่าสายส่งแบบอะนาล็อกมาก ในทางปฏิบัติอาจรับส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์สองเครื่องโดยใช้สัญญาณดิจิตอลผ่านสายได้ ซึ่งทั้งสายส่งและวงจรเชื่อมต่อทั้งหมดเป็นแบบดิจิตอลแต่ว่าค่าใช้จ่ายจะมีราคาแพงมาก จนกระทั่งไม่ค่อยคุ้มที่จะทำเช่นนั้น วิธีการหลีกเลี่ยงก็คือหาทางส่งข้อมูลไปตามสายส่งในแบบอนาล็อกแทน การทำเช่นนี้จำเป็นต้องใช้อุปกรณ์แปลงสัญญาณในการกับส่งข้อมูลทั้งสองด้านซึ่งเป็นที่มาของโมเด็มนั่นเอง

โมเด็ม (MODEM) จะทำหน้าที่แปลงสัญญาณจากคอมพิวเตอร์ที่ส่งมาทาง RS-232C ให้กลายเป็นสัญญาณอะนาล็อกแล้วส่งออกไปตามสายส่ง กระบวนการนี้เรียกว่าการ Modulate เมื่อสัญญาณถึงปลายทางโมเด็มก็จะแปลงสัญญาณอะนาล็อกที่ได้รับกลับมาเป็นสัญญาณดิจิตอลแล้วส่งให้คอมพิวเตอร์ต่อไปในรูปของสัญญาณดิจิตอลผ่านทาง RS-232C เช่นกัน กระบวนการแปลงสัญญาณกลับนี้เรียกว่า Demodulation

สัญญาณอะนาล็อกมีคุณสมบัติเหมาะที่จะส่งไปไกลๆ มากกว่าสัญญาณแบบดิจิตอล เพราะว่าสัญญาณอะนาล็อกจะเพี้ยนหรือมีรูปร่างผิดไปจากเดิมได้ยากกว่า และสูญเสียกำลังในการส่งน้อยกว่า ทำให้ส่งได้ระยะทางไกลมากขึ้น นอกจากนี้ยังสามารถกรองสัญญาณรบกวนบางส่วนที่ไม่ต้องการ (Filter) ออกได้อีกด้วย ราคาของสายส่งและอุปกรณ์เชื่อมต่อก็มีราคาถูก จึงจำเป็นต้องใช้โมเด็มในการรับส่งข้อมูลคอมพิวเตอร์ระยะทางไกลผ่านสายส่งอะนาล็อก

จากการที่โมเด็มแปลงสัญญาณจากคอมพิวเตอร์ให้เป็นสัญญาณอะนาล็อกในการรับส่งข้อมูลนี้เอง ถ้าโมเด็มแปลงสัญญาณออกมาอยู่ในรูปของเสียงซึ่งเป็นสัญญาณอะนาล็อกแบบหนึ่ง ก็สามารถรับส่งข้อมูลผ่านทางสายโทรศัพท์ได้ โมเด็มทั่วๆ ไปที่ใช้งานจะเป็นโมเด็มที่แปลงสัญญาณจากคอมพิวเตอร์ให้อยู่ในรูปของคลื่นเสียงทั้งหมด มีโมเด็มบางชนิดที่แปลงสัญญาณดิจิตอลเป็นอะนาล็อกความถี่สูง แต่โมเด็มแบบนี้มีใช้งานน้อยและใช้ส่งข้อมูลโดยใช้สายส่งพิเศษจะส่งผ่านสายโทรศัพท์ธรรมดาไม่ได้ ดังนั้นจึงเน้นเฉพาะโมเด็มที่ทำงานในย่านความถี่เสียงเท่านั้น ไม่ว่าโมเด็มจะเป็นแบบไหนก็ตามเมื่อได้รับข้อมูลดิจิตอลจากคอมพิวเตอร์มันจะเปลี่ยนให้กลายเป็นสัญญาณอะนาล็อก จากนั้นก็นำสัญญาณอะนาล็อกที่ได้นี้มารวมเข้ากับสัญญาณพาหะ (Carrier Wave) แล้วส่งออกไปทางสายส่งข้อมูล สัญญาณพาหะหรือคลื่นพาหะนี้จะทำหน้าที่พาข้อมูลที่อยู่ในรูปสัญญาณอะนาล็อกไปจนถึงปลายทาง

รูปที่ 2.8 โมเด็มช่วยในการรับส่งข้อมูลผ่านสายโทรศัพท์

2.3.4 มาตรฐานการอินเตอร์เฟซ

องค์กรต่างๆ ได้สร้างมาตรฐานการอินเตอร์เฟซของตนเองขึ้นมา ซึ่งก็มีอยู่มากมายได้แก่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3.4.1 EIA : THE ELECTRONIC INDUSTRIES ASSOCIATION เป็นมาตราฐานที่กำหนดโดยสมาคมโรงงานอุตสาหกรรมผู้ผลิตอิเลคทรอนิคส์แห่งอเมริกา มาตรฐานนี้ตั้งขึ้นมาใช้กำหนดมาตรฐานของเครื่องมือ อุปกรณ์อิเลคทรอนิคส์ต่างๆ การกำหนดมาตรฐานใช้รหัส RS เป็นหลัก เช่น มาตรฐาน RS-232C ใช้กันแพร่หลายในระบบสื่อสารข้อมูลคอมพิวเตอร์โดยจะกล่าวถึงมาตรฐานของสัญญาณไฟฟ้าในการอินเตอร์เฟซเทอร์มินอลเข้ากับโมเด็ม หรืออินเตอร์เฟซเทอร์มินอลเข้ากับไมโครคอมพิวเตอร์ หรือ อินเตอร์เฟซเครื่องพิมพ์เข้ากับคอมพิวเตอร์ นอกจากนี้ยังมีมาตรฐานอื่นๆ ที่มีการตั้งขึ้นมาใช้ประกอบด้วย EIA RS449 RS422A RS423A

2.3.4.2 CCITT : THE CONSULTATIVE COMMITTEE IN INTERNATIONAL TELEGRAPH AND TELEPHONE เป็นองค์กรสากลกำหนดมาตรฐานของระบบสื่อสารระหว่างประเทศทั้งโทรเลขและโทรศัพท์ สำหรับมาตรฐานของ CCITT ที่ใช้กันอย่างแพร่หลายในการสื่อสารข้อมูล เช่น V.28 (ใช้แทน RS-232C ได้) V.10 (ใช้แทน EIA RS423A ได้) V.11 (ใช้แทน EIA RS422A ได้) X.10 (ใช้แทน EIA RS449 ได้) เป็นต้น

2.3.4.3 ISO : THE INTERNATIONAL STANDARD ORGANIZATION เป็นองค์กรกำหนดมาตรฐานทางกายภาพของอุปกรณ์ต่างๆ ที่ใช้ในการสื่อสารโทรคมนาคมโดยเฉพาะที่เกี่ยวกับคอมพิวเตอร์และอินฟอร์เมชั่นโปรเซสซิ่ง องค์กรนี้จะประสานงานกับ CCITT อย่างใกล้ชิด ดังนั้นมาตรฐานของ IS\O สามารถใช้แทนมาตรฐานประเภทเดียวกันของ CCITT และ EIA ได้ เช่น ISO/2110 สามารถใช้แทน RS-232C และ RS-366A ได้ นอกจากนี้ ISO4902 ยังใช้แทน EIA RS-499 ได้ เป็นต้น

2.3.4.4 BELL SYSTEM เป็นมาตรฐานที่กำหนดโดยองค์กรทางโทรศัพท์ของบริษัท Bell Laboratory มาตรฐานของ Bell ถูกกำหนดขึ้นเพื่อใช้ควบคุมโรงงานผู้ผลิตสินค้าที่ต้องการใช้งานร่วมกับระบบของ Bell แต่ระยะหลัง Bell ก็เริ่มมีการแก้ข้อกำหนดของตนบางส่วนให้เข้ากับ CCITT ได้

มาตรฐานของโมเด็มแบบต่างๆ ที่ใช้กันมากนั้นมักจะใช้ตามมาตรฐานของ CCITT V-Series ตั้งแต่ความเร็วต่ำไปถึงความเร็วสูง ได้แก่

- *V.21* เป็นมาตรฐานของโมเด็มความเร็ว 300 บิทต่อวินาที ใช้เทคนิคการผสมสัญญาณแบบ FSK (Frequency Shift Keying) รับส่งข้อมูลได้ในแบบ Full Duplex เป็นโมเด็มที่ใช้กับสายโทรศัพท์ ปัจจุบันนี้มีใช้กันน้อย เนื่องจากความเร็วในการรับส่งข้อมูลต่ำ
- *V.22* รับส่งข้อมูลด้วยความเร็ว 1200 บิทต่อวินาที หรือลดความเร็วลงมาที่ 600 บิทต่อวินาทีได้ การผสมสัญญาณใช้เทคนิคของ PSK (Phase Shift Keying) รับส่งข้อมูลในแบบ Full Duplex ใช้กับสายโทรศัพท์หรือสายตรงก็ได้ขึ้นอยู่กับว่าโมเด็มถูกออกแบบมาให้ใช้กับสายตรงได้หรือไม่ ซึ่งจัดเป็นโมเด็มความเร็วปานกลางที่ได้รับความนิยมอยู่ในปัจจุบัน
- *V.22 bis* รับส่งข้อมูลด้วยความเร็ว 2400 บิทต่อวินาที หรือลดความเร็วลงมาที่ 1200 บิทต่อวินาทีได้ การผสมสัญญาณใช้เทคนิคของโมเด็มความเร็วสูง คือ QAM รับส่งข้อมูลแบบ Full

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Duplex ใช้กับสายโทรศัพท์หรือสายตรงก็ได้ V.22 bis เป็นมาตรฐานของโมเด็มความเร็วปานกลางที่จะมาแทนที่ V.22

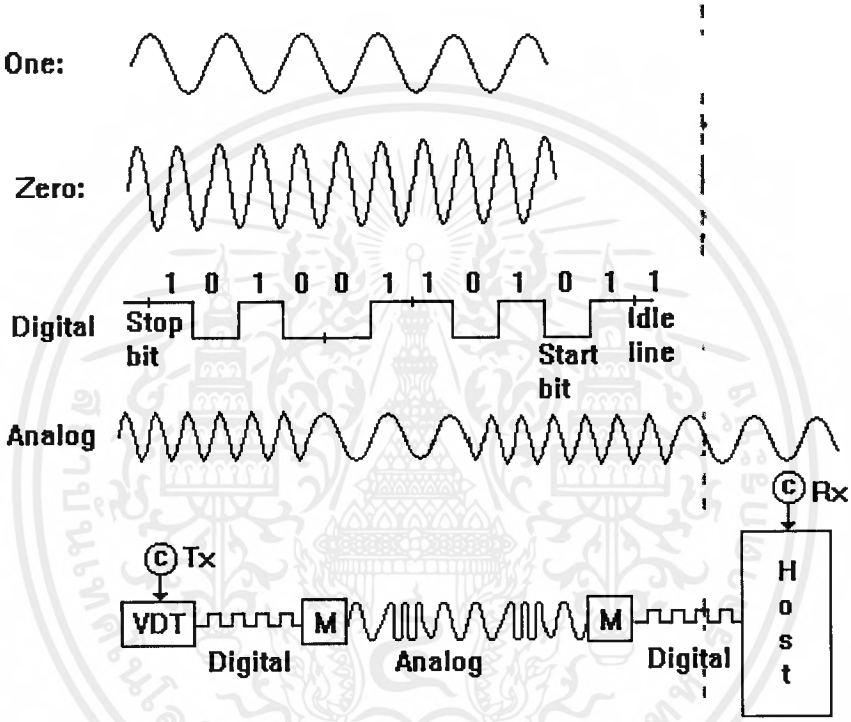
- *V.23* เป็นมาตรฐานที่คล้ายกับ V.22 แต่รับส่งข้อมูลในแบบ Half Duplex คือมีความเร็วที่ 1200 บิทต่อวินาที หรือลดความเร็วลงมาที่ 600 บิทต่อวินาที ใช้เทคนิคการผสมสัญญาณแบบ FSK สามารถต่อใช้กับสายโทรศัพท์ได้
- *V.26* เป็นมาตรฐานของโมเด็มสายตรงแบบใช้สาย 4 เส้น (4 wires) รับส่งข้อมูลในแบบ Full Duplex ใช้เทคนิคการผสมสัญญาณชนิด PSK มีความเร็วในการรับส่งข้อมูล 2400 บิทต่อวินาที จะนำมาใช้กับสายโทรศัพท์ไม่ได้
- *V.26 bis* เป็นมาตรฐานเหมือนกับ V.26 แต่สำหรับใช้สายโทรศัพท์แทน มีความเร็วในการรับส่งข้อมูลที่ 2400 บิทต่อวินาที หรือลดความเร็วลงมาที่ 1200 บิทต่อวินาทีได้ การรับส่งข้อมูลเป็นแบบ Half Duplex ใช้เทคนิคการผสมสัญญาณแบบ PSK
- *V.27* เป็นมาตรฐานสำหรับโมเด็มความเร็ว 4800 บิทต่อวินาทีที่ใช้กับสายตรงเท่านั้น เทคนิคของการผสมสัญญาณเป็นแบบ PSK รับส่งข้อมูลในแบบ Full Duplex ได้ มีความเร็ว 4800 บิทต่อวินาที ซึ่งถือได้ว่าเป็นความเร็วสูงสุดของเทคนิคการผสมสัญญาณแบบ PSK
- *V.27 bis* คล้ายกับมาตรฐานแบบ V.27 แต่ว่ารับส่งข้อมูลที่ 4800 บิทต่อวินาที หรือลดความเร็วลงมาที่ 2400 บิทต่อวินาทีได้ ใช้สำหรับสายตรงแบบ 4 Wires เท่านั้น การผสมสัญญาณก็เป็นแบบ PSK สามารถรับส่งข้อมูลได้ทั้งแบบ Half Duplex และ Full Duplex
- *V.27 ter* เห็นมาตรฐานโมเด็มความเร็ว 4800 บิทต่อวินาที หรือลดความเร็วลงมาที่ 2400 บิทต่อวินาทีได้สำหรับใช้กับสายโทรศัพท์ การรับส่งข้อมูลเป็นแบบ Half Duplex เท่านั้น ใช้เทคนิคการผสมสัญญาณชนิด PSK
- *V.29* จัดเป็นมาตรฐานของโมเด็มความเร็วสูงใช้กับสายตรงแบบ 4 Wires เท่านั้น การรับส่งข้อมูลใช้ได้ทั้ง Full Duplex และ Half Duplex สามารถรับส่งข้อมูลได้ตั้งแต่ 9600 บิทต่อวินาที หรือลดความเร็วลงมาที่ 7200 บิทต่อวินาที และ 4800 บิทต่อวินาที ที่ความเร็ว 9600 บิทต่อวินาที จะใช้เทคนิคการผสมสัญญาณแบบ QAM
- *V.32* เป็นมาตรฐานในโมเด็มความเร็วสูง สำหรับใช้กับสายโทรศัพท์ สามารถรับส่งข้อมูลได้ที่ความเร็ว 9600 บิทต่อวินาทีในแบบ Full Duplex หรือลดความเร็วลงมาที่ 4800 บิทต่อวินาทีได้ มาตรฐาน V.32 นี้ยังสามารถใช้งานกับสายตรงแบบ 2 Wires ได้อีกด้วย ใช้เทคนิคการผสมสัญญาณแบบ QAM ทั้งที่ความเร็ว 9600 และ 4800 บิทต่อวินาที

2.3.5 เทคนิคการมอดูเลทสัญญาณดิจิตอล

เทคนิคการมอดูเลทสัญญาณดิจิตอลที่ใช้กันมากในปัจจุบันมีรายละเอียดดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Frequency-Shift Keying (FSK) มีหลักการคือ ความถี่ของสัญญาณพาหะจะแปรเปลี่ยนตามสัญญาณดิจิตอลที่เข้ามา คือค่าสัญญาณมีระดับลอจิกเป็น 1 ความถี่ของสัญญาณพาหะจะสูงขึ้น แต่ค่าสัญญาณมีระดับลอจิกเป็น 0 ความถี่ของสัญญาณพาหะจะลดลง ดังรูป 2.9 โดยสามารถแยกความแตกต่างของ FSK จาก FM ได้ว่าในขบวน FSK คลื่นพาหะอาจจะมีความถี่ของคลื่นได้มากกว่า 2 ความถี่ แต่ใน FM จะมีความถี่ของคลื่นพาหะได้เพียง 1 ความถี่เท่านั้น จากรูปจะเห็นว่าสามารถแยกสัญญาณ FSK ออกเป็นสัญญาณ Amplitude Shift Keying (ASK) ได้สองสัญญาณ



รูปที่ 2.9 แสดงสัญญาณที่ได้จากเทคนิค FSK

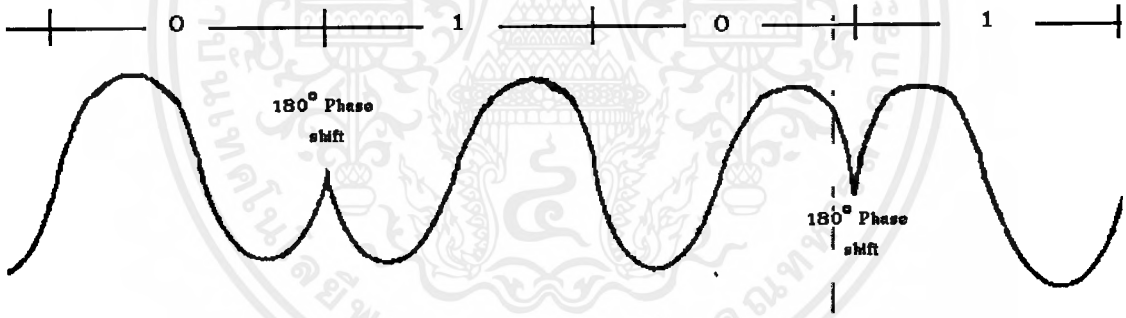
Phase Shift Keying (PSK) มีหลักการดังนี้ คือเฟสของสัญญาณพาหะจะเปลี่ยนไปตามสัญญาณดิจิตอลที่เข้ามาคือ ถ้าระดับลอจิกของสัญญาณเปลี่ยนจาก 0 ไปเป็น 1 เฟสของสัญญาณพาหะก็จะเปลี่ยนไป 90 องศา จากรูปที่ 2.10 จะเห็นว่าเฟสของสัญญาณหลังจาก 180 องศา แล้วควรจะเป็น 270 องศา แต่สัญญาณจะกลับไปเป็น 90 องศา แทน ในลักษณะเช่นนี้ ถ้าสามารถเลื่อนเฟสของสัญญาณไซน์ออกไปเป็นค่าต่างๆ เช่นอาจจะต่างกัน 4 ค่า ประสิทธิภาพของโมเด็มย่อมจะสูงขึ้น ซึ่งในโมเด็มชนิดที่ใช้เทคนิค PSK ก็ใช้หลักการนี้เช่นกัน เช่นในโมเด็ม Bell 201 B มีการเลื่อนสัญญาณไปเป็น 45°,135°,225° และ 315° ซึ่งค่าของเฟสที่เลื่อนไปจะสัมพันธ์กับการวนรอบของเฟสอื่นๆ ด้วย มุมเฟสต่างๆ กันทั้ง 4 ค่านี้จะใช้แทนบิทข้อมูลได้ 2 บิท (แทนที่จะเป็น 1 บิทเช่นใน FSK) คือเป็นค่า 00,01,10 และ 11 ตามลำดับ ซึ่งการเรียงบิทในลักษณะนี้เรียกว่า Dibits ซึ่งมีผลดีคือเป็นเทคนิคที่ทำให้สามารถเพิ่มอัตราบอดได้สูงยิ่งขึ้น เช่นถ้าโมเด็มสามารถส่งข้อมูลในอัตรา 600 dibits/วินาที (600 บอด) ก็จะมีค่าเท่ากับโมเด็มส่งข้อมูลด้วยอัตรา 1200 บิท/วินาที ในปัจจุบันขีดความ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สามารถของโมเด็มได้ถูกพัฒนาขึ้นเป็นอย่างมากสามารถส่งข้อมูลได้ถึง 4800 บิท/วินาที เช่นในกรณีที่มีอัตราของสัญญาณในสายส่ง (Line Signaling Rate) มีค่าเป็น 1600 บอด โมเด็มจะมีอัตราการส่งข้อมูลได้ถึง 4800 บิท/วินาที ซึ่งอาจเกิดความสับสนได้ระหว่างอัตราบิท, อัตราบอด และอัตราสัญญาณในสายส่ง จึงมีตารางเพื่อเปรียบเทียบดังนี้

ตารางที่ 2.2 เปรียบเทียบอัตราบิท, อัตราบอด และอัตราสัญญาณในสายส่ง

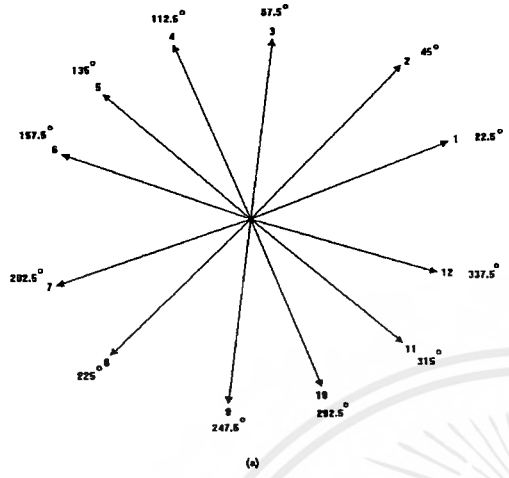
Line Signaling		
Modulation Technique	rate,bauds	Bit rate , bps
FSK	n(300)	n(300)
Dibits	n(1200)	2n(2400)
Tribits	n(1200)	8n(9600)
Quadbits	n(1200)	16n(19200)



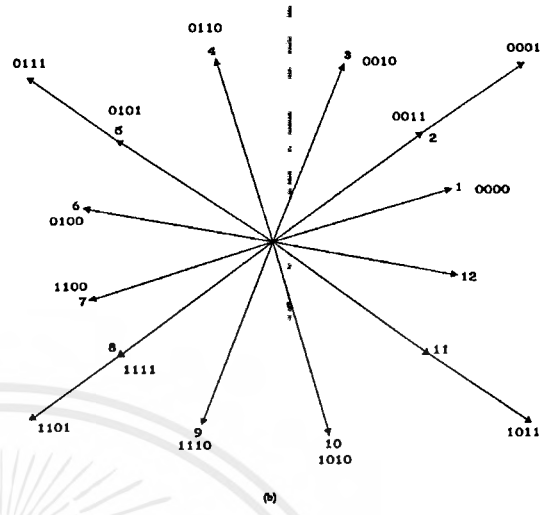
รูปที่2.10 แสดงสัญญาณที่ได้จากเทคนิค PSK

Phase Amplitude Modulation (PAM) เป็นเทคนิคที่เกิดจากการรวมเอาสัญญาณมอดูเลทที่เกิดจากเทคนิค PSK และ AM เข้าด้วยกัน ทำให้ได้อัตราการส่งสูงขึ้น คือสามารถส่งบิทข้อมูลได้ถึง 16 แบบต่างกัน ดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า



รูปที่ 2.11a PAM มุมเฟสต่างๆ กัน 12 มุม



รูปที่ 2.11b PAM การเข้ารหัสขนาด 4 บิท

ในรูปที่ 2.11a จะเห็นว่ามีการแบ่งมุมเฟสออกเป็น 12 มุม สำหรับใช้ในการมอดูเลทแบบ PAM และถ้าเพิ่มขนาดของแอมปลิจูดให้แก่มุมเฟส 4 มุมจากทั้งหมดทำให้มุมเฟส 4 มุมที่เพิ่มแอมปลิจูดเข้าไปนั้น มีค่าแอมปลิจูดถึง 2 ค่า ดังนั้นเกิดมุมเฟสขึ้นอีก 4 ค่ารวมเป็น 16 ค่า ดังรูป 2.11b ซึ่งมุมเฟสแต่ละค่าก็ถูกใช้ในการแทนข้อมูลกลุ่มหนึ่ง เพราะฉะนั้นจึงสามารถแทนกลุ่มข้อมูลได้ทั้งหมด 16 แบบต่างๆ กัน ดังนั้นเทคนิควิธีนี้จึงทำให้อัตราการส่งข้อมูลสูงขึ้นถึง 9600 บิทต่อวินาที ส่วนใหญ่โมเด็มประเภทนี้จะใช้กับการส่งข้อมูลแบบซิงโครนัส

Quadrature Amplitude Modulation (QAM) เป็นการผสมสัญญาณที่ใช้ทั้งการเปลี่ยนเฟสและขนาดของสัญญาณควบคู่กันไป สำหรับใช้กับโมเด็มความเร็วสูง ซึ่งถ้าใช้การเปลี่ยนเฟสเพียงอย่างเดียวมุมที่เปลี่ยนจะมีค่าน้อยเกินไป ทำให้เกิดข้อผิดพลาดได้ง่าย ถ้าใช้การเปลี่ยนเฟสและขนาดของสัญญาณค่าของมุมก็จะอยู่ห่างกันมากขึ้น ปกติที่มีใช้กันอยู่จะมีเฟสต่างกัน 8 เฟส และขนาดของสัญญาณต่างกัน 4 ระดับ ใช้การแทนข้อมูล 16 สถานะ ซึ่งในหนึ่งลูกคลื่นจะสามารถส่งข้อมูลได้คราวละ 4 บิท ความเร็วในการรับส่งข้อมูลของ QAM อยู่ที่ 9600 บิทต่อวินาที โดยใช้ความถี่พาหะ 2400 Hz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

2.4 สถาปัตยกรรมของไมโครคอนโทรลเลอร์แบบชิพเดียวตระกูล 51

2.4.1 บทนำ

ไมโครคอนโทรลเลอร์แบบชิพเดียว (Single Chip Microcontroller) คือไมโครคอมพิวเตอร์แบบที่มีขนาดเล็กโดยบรรจุไว้ในแผงวงจรรวม (Intergrated Ciruit) เพียงชิพเดียวที่เหมาะสมสำหรับงานควบคุมอุปกรณ์อื่นๆ แบบอัตโนมัติ เพราะผู้ใช้สามารถเขียนโปรแกรมควบคุมการทำงานได้ตามต้องการ ไมโครคอนโทรลเลอร์แบบชิพเดี่ยวตระกูล MCS-51 อันได้แก่ เบอร์ 8051 และ 8052 มีโครงสร้างและชุดคำสั่งแตกต่างกันเพียงเล็กน้อยดังตาราง

ตารางที่ 2.3 โครงสร้างของ MCS-51 แบบต่างๆ

Device	ROM less Version	EPROM Version	ROM Byte	RAM Byte	8 Bit I/O Ports	16 Bit Timer/ Counters	Programmable Counter Array (PCA)	UART	Serial Expansion Port (SEP)	Global Serial Channel (GSC)	DMA Channels	A/D Channels	Interrupt Sources/ Vectors	Power Down and Idle Modes
8051	8031	-	4K	128	4	2		√					6/5	
8051AH	8031AH	8751H 8751 BH	4K	128	4	2		√					6/5	
8052AH	8032AH	8752BH	8K	256	4	3		√					8/6	
80C51BH	80C31BH	87C51	4K	128	4	2		√					6/5	√
80C52	80C32	-	8K	256	4	3		√					8/6	√
83C51FA	80C51FA	87C51FA	8K	256	4	3	√	√					14/7	√
83C51FB	80C51FA	87C51FB	16K	256	4	3	√	√					14/7	√
83C152JA	80C152JA	-	3K	256	5	2		√		√	2		19/11	√
-	80C152JB	-	-	256	7	2		√		√	2		19/11	√
83C152JC	80C152JC	-	8K	256	5	2		√		√	2		19/11	√
-	80C152JD	-	-	256	7	2		√		√	2		19/11	√
83C452	80C452	87C452P	8K	256	5	2		√					9/8	√

จากตารางแต่ละคอลัมน์จะบอกถึงคุณสมบัติหรือโครงสร้างของไมโครคอนโทรลเลอร์แต่ละเบอร์ในตระกูล MCS-51 เช่นมี ROM หรือ RAM ภายในเท่าใด ถ้าเป็นรุ่นที่ไม่มี ROM อยู่ภายในจะเป็นเบอร์อะไร หรือถ้าเป็นรุ่นที่มีหน่วยความจำสำหรับโปรแกรมที่เป็นแบบ EPROM จะเป็นเบอร์อะไร นอกจากนี้ในตารางยังจะบอกว่าไมโครคอนโทรลเลอร์เบอร์นั้นมีพอร์ตสำหรับอ่านเขียนข้อมูลขนาด 8 บิทอยู่กี่ชุด (8 Bit I/O Port) มี Timer/Counter ขนาด 16 บิทกี่ชุด (16 Bit Timer/Counter) และยังบอกถึงคุณสมบัติอื่นๆ อีก ทำให้ผู้ใช้สามารถเลือกใช้ไมโครคอนโทรลเลอร์แต่ละเบอร์ให้เหมาะกับการใช้งานได้อย่างดีที่สุด MCS-51 ผลิตโดยบริษัท Intel มีการทำงานเป็นแบบ 8 บิท หมายความว่าส่วนที่ทำงานในการคำนวณ (Arithmetic Logic Unit, ALU) จะทำงานสูงสุดทีละ 8 บิท

2.4.2 MCS-51 มีข้อดีดังนี้

- สามารถนำข้อมูลมา AND, OR หรือทำ Complement ทั้งแบบทีละ 8 บิท และ 1 บิท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

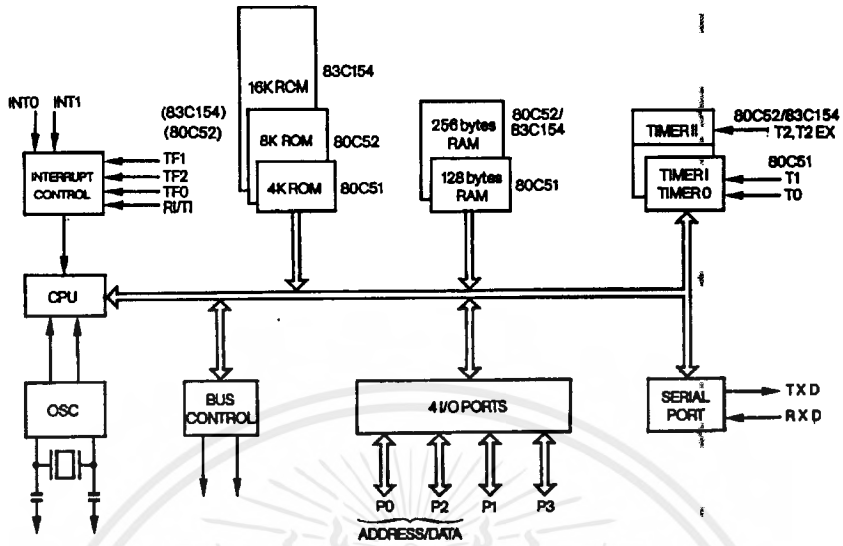
- สามารถใช้กับหน่วยความจำโปรแกรม (Program Memory) ซึ่งเป็นหน่วยความจำที่ใช้สำหรับเก็บชุดคำสั่งที่จะให้ MCS-51 ทำงานได้สูงสุด 64 กิโลไบท์ (64*1024) ทำให้เขียนโปรแกรมควบคุมการทำงานได้มาก
- สามารถต่อกับหน่วยความจำข้อมูล (Data Memory) ซึ่งเป็นหน่วยสำหรับเก็บข้อมูลในระหว่างการทำงานของโปรแกรมได้สูงสุด 64 กิโลไบท์
- ใน 8051 และ 8751 มีหน่วยความจำสำหรับโปรแกรมจำนวน 4 กิโลไบท์ (ใน 8052 และ 8752 มีหน่วยความจำสำหรับโปรแกรมจำนวน 8 กิโลไบท์) อยู่ภายวงจรรวม ทำให้ไม่ต้องต่อหน่วยความจำโปรแกรมอยู่ภายนอก ระบบรวมทั้งหมดจึงมีขนาดเล็กและสัญญาณรบกวนจากภายนอกจะทำให้ทำงานผิดพลาดได้ยาก
- มีพอร์ตแบบขนาน (Parallel Port) สำหรับข้อมูลเข้าและออกจำนวน 32 บิท ที่ข้อมูลแต่ละบิทเป็นอิสระต่อกัน
- มีวงจร Timer/Counter ขนาด 16 บิท 2 ชุด (8052 มี 3 ชุด) ที่ทำงานในโหมดต่างๆ ถึง 4 โหมด
- มี Univesal Asynchonus Receiver Transmitter (UART) สำหรับส่งข้อมูลอนุกรมแบบ Full duplex ที่สามารถเลือกรูปแบบการรับส่งข้อมูลได้ 4 แบบ
- มีแหล่งกำเนิดสัญญาณขอขัดจังหวะการทำงานของโปรแกรม (Interrupt Request Hignal) 6 แหล่ง ซึ่งสามารถทำการกระโดดไปทำงานตอบสนองการขัดจังหวะได้ต่างกัน 5 ตำแหน่ง
- สามารถเลือกการทำงานให้อยู่ในโหมดของ Idle และ Power Down ซึ่งจะประหยัดการใช้กำลังไฟในการทำงาน

ซึ่งจากข้อดีดังกล่าว จึงทำให้ MCS-51 เป็นที่นิยมนำมาใช้ในการควบคุมระบบอัตโนมัติ คุณสมบัติดังกล่าวบรรจุไว้ในวงจรรวมเดียว (Single Chip) ขนาด 40 ขา ดังนั้นจึงสามารถออกแบบให้ระบบทั้งหมดมีขนาดเล็ก และทำให้การตรวจสอบหาข้อผิดพลาดในระบบง่ายไม่ซับซ้อนรวมทั้งลดปัญหาเรื่องสัญญาณรบกวน แต่การที่จะนำ MCS-51 มาใช้งานได้จำเป็นที่จะต้องศึกษาและทำความเข้าใจถึงโครงสร้างและองค์ประกอบของ MCS-51 เสียก่อนแล้วถึงจะเขียนโปรแกรมเพื่อควบคุมการทำงานให้เป็นไปตามต้องการ

2.4.3 โครงสร้างของ 8051

ภายใน 8051 จะประกอบด้วยเกท (GATE) ต่างๆ เช่น AND, OR, NOT ซึ่งเกทเหล่านี้จะถูกนำมาออกแบบให้มีหน้าที่ต่างๆ เช่น วงจรถอดรหัสคำสั่ง (Instruction Decoder), วงจรสร้างสัญญาณนาฬิกา โครงสร้างภายในของ 8051 จะประกอบด้วยส่วนต่างๆ ดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า



รูปที่ 2.12 โครงสร้างภายใน 8051

2.4.3.1 ซีพียู (CPU) หรือตัวประมวลผลกลาง ส่วนนี้มีหน้าที่สร้างสัญญาณควบคุมในการติดต่อกับส่วนอื่นๆ เช่น สัญญาณสำหรับการติดต่อกับหน่วยความจำ, อุปกรณ์รับส่งข้อมูล, ส่วนควบคุมการขัดจังหวะ, ส่วนควบคุมบัส เป็นต้น สัญญาณควบคุมนี้จะถูกสร้างจากการถอดรหัสคำสั่งตามที่มีการกำหนดไว้ และสร้างสัญญาณขึ้นมาอ้างอิงกับสัญญาณนาฬิกาที่สร้างจากวงจรออสซิเลเตอร์ เพื่อให้ทุกๆ ส่วนทำงานประสานกันได้อย่างถูกต้อง

ในซีพียูยังประกอบด้วยส่วนย่อยอีกส่วนหนึ่งที่เรียกว่า ส่วนคณิตศาสตร์และลอจิก (Arithmetic Logic Unit) ซึ่งทำหน้าที่ในการประมวลผลข้อมูล เช่น การบวก, ลบ , คูณ, หาร แล้วนำผลลัพธ์ที่ได้ไปเก็บไว้ในรีจิสเตอร์ หรือหน่วยความจำที่ต้องการ

2.4.3.2 หน่วยความจำ (Memory) มีไว้สำหรับเก็บข้อมูล การนำข้อมูลไปไว้ในหน่วยความจำเรียกว่า การเขียน (Write) และการนำข้อมูลออกจากหน่วยความจำเรียกว่า การอ่าน (Read) การติดต่อกับหน่วยความจำจะต้องมีสัญญาณ 3 กลุ่ม คือ

2.4.3.2.1 แอสเดรสหรือค่าตำแหน่งที่ต้องการติดต่อ ใน 8051 จะติดต่อกับหน่วยความจำโปรแกรมและหน่วยความจำข้อมูลได้สูงสุดชนิดละ 65536 ตำแหน่ง ซึ่งการอ้างถึงตำแหน่งต่างๆ จะใช้บัสจำนวน 16 เส้น

2.4.3.2.2 ข้อมูลที่จะอ่าน หรือเขียนกับหน่วยความจำ

2.4.3.2.3 สัญญาณควบคุม ที่ส่งไปยังหน่วยความจำเพื่อบอกว่าจะอ่าน หรือเขียนข้อมูล สัญญาณเหล่านี้จะถูกสร้างมาจากวงจรถอดรหัสของคำสั่งที่อ่านจากหน่วยความจำโปรแกรม

2.4.3.3 อุปกรณ์อินพุทและเอาท์พุท เป็นส่วนที่ใช้ในการส่งข้อมูลเข้าหรือออกจาก 8051 เพื่อให้สามารถติดต่อกับภายนอกได้ อุปกรณ์เหล่านี้ได้แก่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

2.4.3.3.1 I/O Port เป็นที่สำหรับรับส่งข้อมูลเข้า หรืออุ่อกจากตัว 8051 มีทั้งหมด 4 พอร์ต แต่ละพอร์ตจะรับส่งข้อมูลได้ครั้งละ 8 บิท บางพอร์ตจะใช้งานได้มากกว่าุหนึ่งอย่าง เช่น P0, P2 จะใช้สำหรับส่งค่าตำแหน่งของหน่วยความจำที่ต้องการติดต่อ

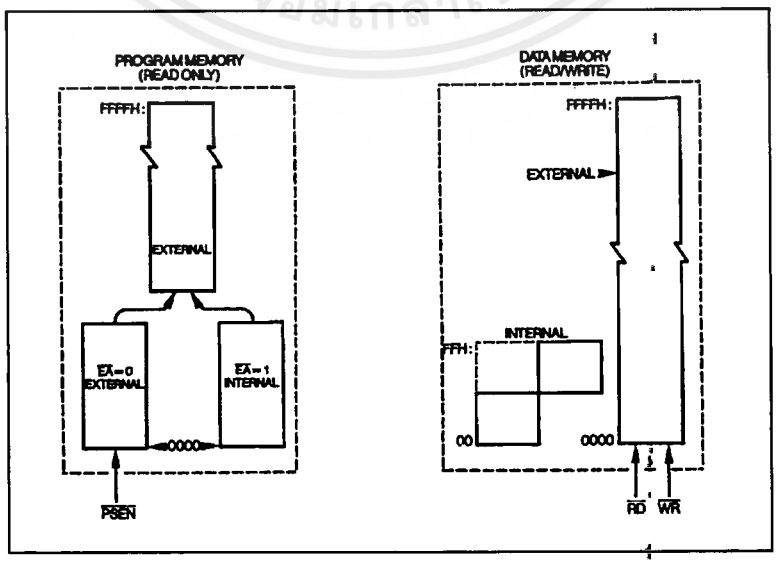
2.4.3.3.2 Timer 0, Timer 1 เป็นวงจรนับที่สามารถุกำหนดให้ทำการนับจำนวนไซเคิลที่ต่อจากภายนอก หรือจากภายใน 8051 ก็ได้

2.4.3.3.3 พอร์ตอนุกรม (Serial Port) การอ่านและเขียนข้อมูลของซีพียูกับพอร์ตอนุกรมจะเป็นแบบ 8 บิท แต่ข้อมูลจะถูกส่งออกจาก 8051 ที่ละบิททางขา TXD และในการรับข้อมูลก็จะรับที่ละบิททางขา RXD แล้วจัดเรียงเป็น 8 บิทเพื่อให้ซีพียูอ่านและนำไปใช้งานต่อไป

2.4.4 การจัดหน่วยความจำของ 8051

หน่วยความจำของ 8051 แบ่งตามลักษณะการใช้งานได้ 2 แบบ คือ

2.4.4.1 หน่วยความจำโปรแกรม (Program Memory) เป็นุส่วนที่ใช้เก็บคำสั่งในรูปของภาษาเครื่อง (Machine Language) ซึ่งต้องการให้ 8051 ทำงาน เมื่อ 8051 เริ่มทำงานก็จะอ่านข้อมูลที่เก็บในหน่วยความจำนี้เข้าไปถอดรหัสคำสั่งแล้วสร้างสัญญาณควบคุมส่วนต่างๆ ตามคำสั่งนั้น หน่วยความจำแบบนี้จะเป็นแบบรอม (ROM:Read Onry Memory) ซึ่งระหว่างการทำงานจะไม่สามารถใช้คำสั่งเขียนข้อมูลลงไปในหน่วยความจำแบบนี้ได้ ตำแหน่งของหน่วยความจำนี้จะเริ่มตั้งแต่ 0000H จนถึง 0FFFFH ซึ่งในตำแหน่ง 0000H - 0FFFH จำนวน 4 กิโลไบท์ สามารถเลือกได้ว่าจะใช้รอมที่อยู่ภายนอกหรือภายใน 8051 ถ้าต้องการใช้หน่วยความจำภายใน 8051 ก็ให้ป้อนลอจิก "1" เข้าที่ขา EA ของ 8051 แต่ถ้าต้องการใช้หน่วยความจำภายนอกก็ให้ต่อลอจิก "0" เข้าที่ขา EA ส่วนหน่วยความจำที่ตำแหน่ง 1FFFH - 0FFFFH จะต้องต่ออยู่ภายนอกเสมอดังรูป



รูปที่ 2.13 การจัดหน่วยความจำของ 8051

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

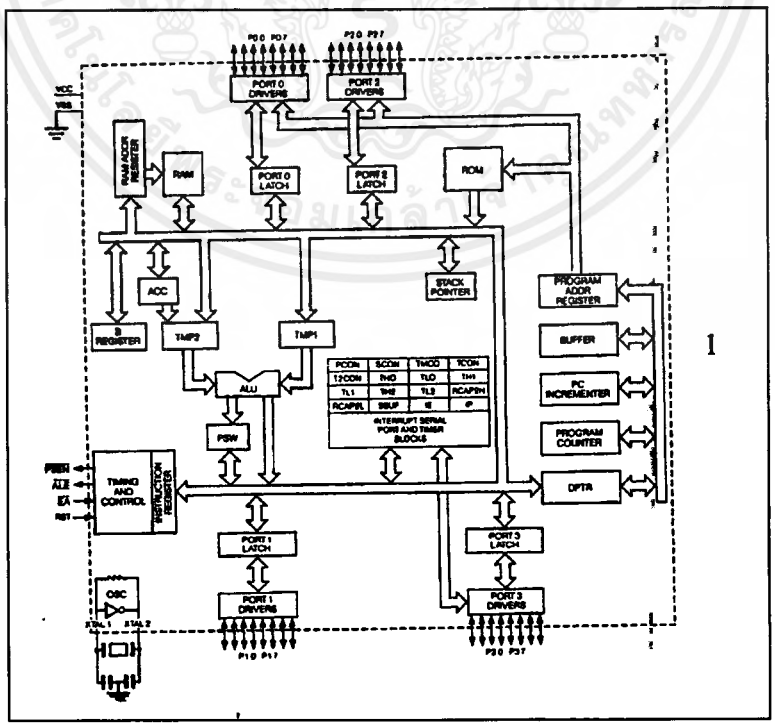
MCS-51 เบอร์ 8031, 8051, 8751 จะมีโครงสร้างและรหัสคำสั่งเหมือนกันทุกประการ แต่จะแตกต่างกันที่

- 8031 จะไม่มี ROM ขนาด 4 กิโลไบท์อยู่ภายใน จะต้องใช้หน่วยความจำโปรแกรมที่อยู่ภายนอกทั้ง 64 กิโลไบท์
- 8051 จะมี ROM ขนาด 4 กิโลไบท์อยู่ภายใน ถ้าต้องการใช้หน่วยความจำส่วนนี้จะต้องส่งโปรแกรมคำสั่งไปให้โรงงานผู้ผลิตทำการเขียนใส่ ROM ไว้ให้ตั้งแต่อยู่ในขั้นตอนการผลิต
- 8751 จะมี ROM ขนาด 4 กิโลไบท์แบบ EPROM อยู่ภายใน สามารถเขียนโปรแกรมคำสั่งลงไปได้เอง โดยใช้เครื่องโปรแกรม EPROM ซึ่งจะสะดวกมากสำหรับการพัฒนาโปรแกรม

2.4.4.2 หน่วยความจำข้อมูล (Data Memory) เป็นหน่วยความจำที่ใช้สำหรับพัก, เก็บข้อมูล แล้วเรียกมาใช้ใหม่ในระหว่างการทำงาน หน่วยความจำประเภทนี้จะเป็นแบบ Random Access Memory (RAM) ซึ่งข้อมูลจะสูญหายไปถ้าไม่มีไฟเลี้ยงต่ออยู่ หน่วยความจำแบบนี้จะมีอยู่ 2 ส่วน ส่วนแรกจะอยู่ในตัว 8051 ขนาด 128 ไบท์ที่ตำแหน่ง 00H-7FH ส่วนที่สองจะต่ออยู่ภายนอกมีขนาด 64 กิโลไบท์ ที่ตำแหน่ง 0000H-0FFFFH

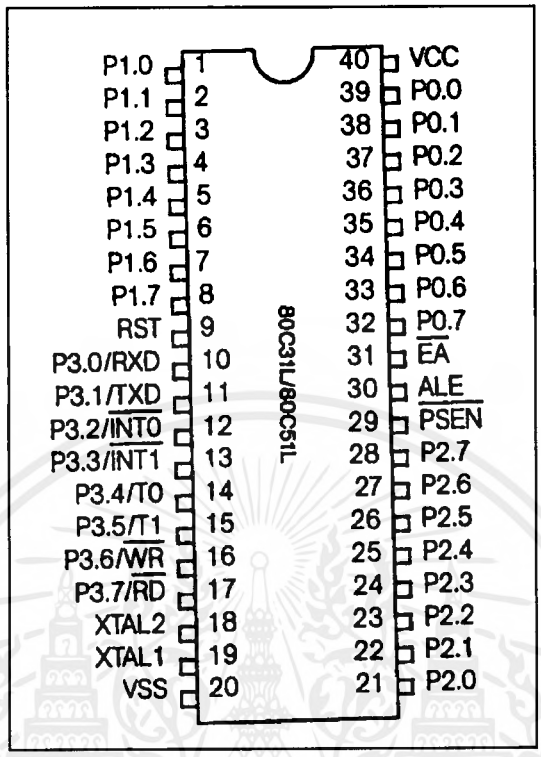
2.4.5 สถาปัตยกรรมของ 8051

8051 จะบรรจุอยู่ในวงจรรวมแบบ Dual Inline Package (DIP) สัญญาณจากภายในจะต่อออกสู่ภายนอกทางขาต่างๆ ของ 8051 ซึ่งมีอยู่ข้างละ 20 ขารวมทั้งหมด 40 ขา ดังรูป



รูปที่ 2.14 สถาปัตยกรรมของ 8051

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.15 ขาสัญญาณของ 8051

2.4.6 การทำงานของ 8051

คอมพิวเตอร์จะทำงานด้วยวงจรที่เรียกว่าฮาร์ดแวร์ (Hardware) ประกอบเพียงอย่างเดียวไม่ได้จะต้องมีโปรแกรมหรือคำสั่งที่จัดเรียงกันไว้ให้คอมพิวเตอร์ทำงานตามลำดับ ใน 8051 ก็เช่นกัน ผู้ใช้จะต้องเขียนโปรแกรมเป็นภาษาเครื่อง ซึ่งอยู่ในรูปของเลขฐาน 2 เก็บไว้ในหน่วยความจำประเภท Program Memory แต่คำสั่งของ 8051 อาจประกอบด้วย 1, 2หรือ 3 ไบท์แล้วแต่จะเป็นคำสั่งให้ทำงานอะไร

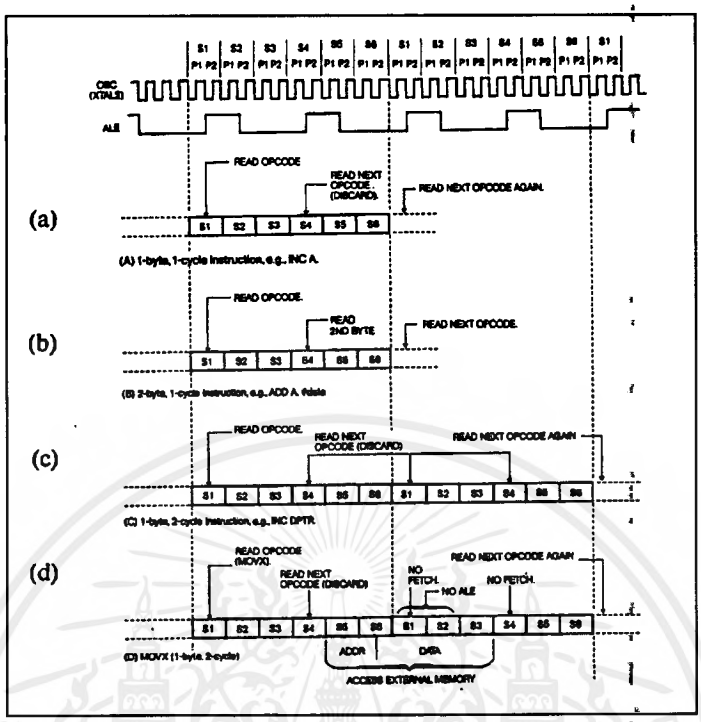
เมื่อเริ่มป้อนไฟเลี้ยงให้กับ 8051 ซึ่งมีวงจร Power on Reset ต่ออยู่จะมีการรีเซ็ทเกิดขึ้น การทำงานใน 8051 จะเริ่มจากบล็อก Program Counter ซึ่งจะเป็นวงจรนับ (Counter Circuit) ส่งค่าตำแหน่งหน่วยความจำโปรแกรมลงไปบนบัส (Bus) หมายเลข 1 บัสนี้มีขนาด 16 บิท ค่าตำแหน่งหน่วยความจำนี้ จะถูกส่งไปเก็บไว้ที่ Program ADDR Register ที่เป็นวงจรแลท (Latch) ข้อมูลซึ่งเป็นค่าตำแหน่งหน่วยความจำ จะปรากฏที่บนบัส 16 บิท หมายเลข 2 ถ้าเป็นค่าตำแหน่งหน่วยความจำแรกหลังจากรีเซ็ท ค่าตำแหน่งความจำเป็น 0000H หน่วยความจำสำหรับโปรแกรมจะเลือกไว้เป็น ROM ภายในหรือภายนอก 8051 โดยการป้อนสภาวะลอจิกเข้าไปที่ 8051 ทางขา EA ซึ่งต่ออยู่กับส่วน Timing and Control ทำหน้าที่เป็นวงจรถอดรหัส (Decoder) แล้วสร้างสัญญาณควบคุมต่อไปถ้าป้อนสัญญาณลอจิก 0 เข้าไปที่ขา EA จะเป็นการเลือกใช้ ROM ภายใน 8051 โดยมีวงจร Timing and Control จะสร้างสัญญาณไปยัง ROM ภายในให้ส่งข้อมูลที่เป็นคำสั่งจากตำแหน่งที่ถูกชี้ด้วยค่าตำแหน่งของบัสที่ถูกส่งมาทางบัสหมายเลข 2 ข้อมูลจาก ROM จะถูกส่งไปยังบัสหมายเลข 3 ที่เรียกว่า Internal Data Bus แล้วนำไปเก็บไว้ที่ Instruction Register (เป็นวงจร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Latch) เพื่อส่งต่อไปให้กับวงจร Timing and Control ทำการถอดรหัสแล้วควบคุมการทำงานส่วนอื่นๆ ต่อไปแล้วแต่ว่าเป็นคำสั่งให้ทำอะไร ในกรณีที่เลือก ROM ภายนอก 8051 โดยป้อนสัญญาณลอจิก 1 เข้าไปที่ ขา EA จะทำให้วงจร Timing and Control ส่งสัญญาณไปยังพอร์ต 0 และพอร์ต 2 เพื่อส่งค่าตำแหน่งหน่วยความจำบนบัสหมายเลข 2 ออกไปชี้หน่วยความจำภายนอก จากนั้นจะอ่านข้อมูลที่เป็นคำสั่งกลับเข้ามาทางพอร์ต 0 ไปยัง Internal Data Bus แล้วไปเก็บที่ Instruction Regirter เพื่อทำงานต่อไปเหมือนตอนอ่านคำสั่งจาก ROM ภายใน การทำงานในช่วงส่งค่าตำแหน่งหน่วยความจำไปยังหน่วยความจำแล้วอ่านข้อมูลที่เป็นคำสั่งกลับเข้ามาเก็บไว้ใน Instruction Register เรียกว่าเป็นช่วงของการ Fetch (Fetch Cycle) ช่วงต่อไปจะเป็นช่วงของการทำงานตามคำสั่ง เรียกว่า Execute Cycle เช่นถ้าเป็นคำสั่งในการให้บวกในรีจิสเตอร์ Accumulator จากข้อมูลจากหน่วยความจำ Data Memory ภายใน RAM ตำแหน่ง 23H วงจร Timing and Control ก็จะส่งสัญญาณให้ Instruction Register ส่งค่าตำแหน่งหน่วยความจำ 23H ลงไปยัง Internal Data Bus แล้วนำข้อมูลไปเก็บไว้ที่ RAM ADDR Register เพื่อใช้ชี้ตำแหน่งหน่วยความจำ RAM จากนั้น Timing and Control จะสั่งให้ RAM ส่งข้อมูลที่เก็บไว้ในหน่วยความจำตำแหน่ง 23H ลงมายัง Internal Data Bus แล้วนำข้อมูลไปเก็บไว้ที่ TMP1 (วงจร Latch) ขณะเดียวกันวงจร Timing and Control ก็จะส่งสัญญาณไปยัง ACC ให้ส่งข้อมูลมายัง TMP2 (วงจร Latch) วงจร ALU ซึ่งเป็นวงจรโครงสร้างการคำนวณทางคณิตศาสตร์ (บวก ลบ คูณ หาร) และยังสามารถทำงานทางลอจิก (AND, OR, NOT, XOR) จะทำการบวกเลขจาก TMP1 และ TMP2 เข้าด้วยกัน ผลลัพธ์ที่ได้จะส่งผ่าน Internal Data Bus กลับไปเก็บยัง ACC PSW (Program Status Word) จะทำหน้าที่เก็บสถานะผลลัพธ์ของการทำงานใน ALU เช่นผลลัพธ์การบวกมีค่าเป็น 8 บิท ก็จะทำให้บิทหนึ่งใน PSW ถูก SET เป็น 1

การทำงานที่กล่าวมาข้างต้นจะขึ้นกับสัญญาณควบคุมที่สร้างมาจากวงจร Timing and Control และสัญญาณที่สร้างขึ้นจะอ้างอิงกับสัญญาณนาฬิกาที่สร้างมาจากวง Oscillator ทำให้การทำงานต่างๆ เป็นไปตามลำดับที่ผู้ผลิตได้ออกแบบไว้ดังรูป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.16 สัญญาณการทำงานของคำสั่งแบบต่างๆ

คำสั่งแต่คำสั่งของ 8051 จะใช้เวลาทำงาน 1, 2 หรือ 3 ไซเคิลของเครื่องุ (Machine Cycle) แล้วแต่ว่าเป็นคำสั่งประเภทใด 1 ไซเคิลของเครื่องจะใช้เวลา 12 ไซเคิลของสัญญาณนาฬิกา ดังนั้นแต่ละคำสั่งของ 8051 จะใช้เวลาการทำงาน 12, 24 หรือ 36 ไซเคิลของสัญญาณนาฬิกานั้นเอง แต่ละไซเคิลของเครื่องจะถูกแบ่งออกเป็น 6 State คือ S1-S6 แต่ละ State จะประกอบด้วย 2 ไซเคิลของสัญญาณนาฬิกา ในไซเคิลแรกจะเรียกว่าเฟส 1 (P1) และไซเคิลที่ 2 เรียกเฟส 2 (P2) แต่ละเฟสจะนับขอบขาลงของสัญญาณนาฬิกาถึงขอบขาลงของสัญญาณนาฬิกาที่อยู่ถัดไปดังในรูป เมื่อ 8051 ทำงานเสร็จของ 1 ไซเคิลของเครื่อง วงจร Timing and Control จะสร้างสัญญาณ ALE ออกมา 2 ไซเคิลเพื่อ Fetch คำสั่งเข้าไป 2 ครั้งเสมอ ที่บริเวณขอบขาขึ้นของสัญญาณ ALE

2.4.7 ข้อมูลเฉพาะของชุดคำสั่ง

ในรูปเป็นสรุปชุดคำสั่งของ 8051 (MCS-51 Instruction Set Description) บล็อกที่ 1 เป็นรหัสคำสั่งช่วยจำ (Mnemonic Code) ทั้งหมดของ 8051 และในบล็อกที่ 2 จะเป็นคำอธิบายโดยย่อของการทำงานในแต่ละคำสั่งที่อยู่ข้างซ้ายของคำอธิบาย ในการเขียนโปรแกรมภาษาแอสแซมบลีจะไม่นำเอาส่วนของบล็อกที่ 2 มาเขียน จะมีเฉพาะบล็อกที่ 1 เท่านั้น รหัสคำสั่งช่วยจำเป็นประโยคที่สามารถอ่านแล้วเข้าใจการทำงานได้โดยตรง ในรหัสคำสั่งแต่ละคำสั่งจะประกอบด้วย 2 ส่วน คือ

2.4.7.1 รหัสดำเนินการ (Operation Code, OP-CODE) เป็นชุดของตัวอักษรที่บอกถึงการทำงานของไมโครโพรเซสเซอร์ รหัสดำเนินการนี้จะเป็นคำในภาษาอังกฤษความยาว 2-4 ตัวอักษรและคำเหล่านี้มีความหมายที่สามารถจดจำได้ง่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.7.2 ตัวถูกดำเนินการ (Operand) เป็นส่วนที่ 2 ของรหัสคำสั่งช่วยจำ ในส่วนนี้จะเป็นชุดของตัวอักษรที่บอกถึงส่วนที่จะถูกดำเนินการ ลักษณะการจัดวางข้อมูลโดยทั่วไปสำหรับการกระทำระหว่างรีจิสเตอร์กับรีจิสเตอร์ หรือรีจิสเตอร์กับหน่วยความจำใน MCS-51 จะมีรูปแบบ

Destination ,Source

การเขียนโปรแกรมภาษาแอสแซมบลีก็คือ การนำเอารหัสคำสั่งมาจัดเรียงกัน เพื่อให้คอมพิวเตอร์ทำงานตามคำสั่งนั้น แต่ก่อนที่จะนำเอาโปรแกรมดังกล่าวไปให้คอมพิวเตอร์ทำงานจะต้องเปลี่ยนรหัสคำสั่งให้เป็นภาษาเครื่องเสียก่อน รหัสคำสั่งแต่ละคำสั่งจะมีภาษาเครื่องต่างกันและไม่ซ้ำกัน

ARITHMETIC OPERATIONS

Mnemonic		Description	Byte	Cyc
ADD	A,Rn	Add register to Accumulator	1	1
ADD	A,direct	Add direct byte to Accumulator	2	1
ADD	A,@Ri	Add indirect RAM to Accumulator	1	1
ADD	A,#data	Add immediate data to Accumulator	2	1
ADDC	A,Rn	Add register to Accumulator with Carry	1	1
ADDC	A,direct	Add direct byte to A with Carry flag	2	1
ADDC	A,@Ri	Add indirect RAM to A with Carry flag	1	1
ADDC	A,#data	Add immediate data to A with Carry flag	2	1
SUBB	A,Rn	Subtract register from A with Borrow	1	1
SUBB	A,direct	Subtract direct byte from A with Borrow	2	1
SUBB	A,@Ri	Subtract indirect RAM from A with Borrow	1	1
SUBB	A,#data	Subtract immed. data from A with Borrow	2	1
INC	A	Increment Accumulator	1	1
INC	Rn	Increment register	1	1
INC	direct	Increment direct byte	2	1
INC	@Ri	Incriment indirect RAM	1	1
INC	DPTR	Incriment Data Pointer	1	2
DEC	A	Decrement Accumulator	1	1
DEC	Rn	Decrement register	1	1
DEC	direct	Decrement direct byte	2	1
DEC	@Ri	Decrement indirect RAM	1	1
MUL	AB	Multiply A & B	1	4
DIV	AB	Divide A by B	1	4
DA	A	Decimal Adjust Accumulator	1	1

LOGICAL OPERATIONS

Mnemonic		Destination	Byte	Cyc
ANL	A,Rn	AND register to Accumulator	1	1
ANL	A,direct	AND direct byte to Accumulator	2	1
ANL	A,@Ri	AND indirect RAM to Accumulator	1	1
ANL	A,#data	AND immediate data to Accumulator	2	1
ANL	direct,A	AND Accumulator to direct byte	2	1
ANL	direct,#data	AND immediate data to direct byte	3	2
ORL	A,Rn	OR register to Accumulator	1	1
ORL	A,direct	OR direct byte to Accumulator	2	1
ORL	A,@Ri	OR indirect RAM to Accumulator	1	1
ORL	A,#data	OR immediate data to Accumulator	2	1
ORL	direct,A	OR Accumulator to direct byte	2	1
ORL	direct,#data	OR immediate data to direct byte	3	2
XRL	A,Rn	Exclusive-OR register to Accumulator	1	1
XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	1
XRL	A,@Ri	Exclusive-OR indirect RAM to A	1	1
XRL	A,#data	Exclusive-OR immediate data to A	2	1
XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	1
XRL	direct,#data	Exclusive-OR immediate data to direct	3	2
CLR	A	Clear Accumulator	1	1
CPL	A	Complement Accumulator	1	1
RL	A	Rotate Accumulator Left	1	1
RLC	A	Rotate A Left through the Carry flag	1	1
RR	A	Rotate Accumulator Right	1	1
RRC	A	Rotate A Right through Carry flag	1	1
SWAP	A	Swap nibbles within the Accumulator	1	1

รูปที่ 2.17 ชุดคำสั่งของ 8051

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DATA TRANSFER

Mnemonic		Description	Byte	Cyc
MOV	A,Rn	Move register to Accumulator	1	1
MOV	A,direct	Move direct byte to Accumulator	2	1
MOV	A,@Ri	Move indirect RAM to Accumulator	1	1
MOV	A,#data	Move immediate data to Accumulator	2	1
MOV	Rn,A	Move Accumulator to register	1	1
MOV	Rn,direct	Move direct byte to register	2	2
MOV	Rn,#data	Move immediate data to register	2	1
MOV	direct,A	Move Accumulator to direct byte	2	1
MOV	direct,Rn	Move register to direct byte	2	2
MOV	direct,direct	Move direct byte to direct	3	2
MOV	direct,@Ri	Move indirect RAM to direct byte	2	2
MOV	direct,#data	Move immediate data to direct byte	3	2
MOV	@Ri,A	Move Accumulator to indirect RAM	1	1
MOV	@Ri,direct	Move direct byte to indirect RAM	2	2
MOV	@Ri,#data	Move immediate data to indirect RAM	2	1
MOV	DPTR,#data 16	Load Data Pointer with a 16-bit constant	3	2
MOVC	A,@A+DPTR	Move Code byte relative to DPTR to A	1	2
MOVC	A,@A+PC	Move Code byte relative to PC to A	1	2
MOVX	A,@Ri	Move External RAM (8-bit addr) to A	1	2
MOVX	A,@DPTR	Move External RAM (16-bit addr) to A	1	2
MOVX	@Ri,A	Move A to External RAM (8-bit addr)	1	2
MOVX	@DPTR,A	Move A to External RAM (16-bit addr)	1	2
PUSH	direct	Push direct byte onto stack	2	2
POP	direct	Pop direct byte form stack	2	2
XCH	A,Rn	Exchange register with Accumulator	1	1
XCH	A,direct	Exchange direct byte with Accumulator	2	1
XCH	A,@Ri	Exchange indirect RAM with A	1	1
XCHD	A,@Ri	Exchange low-order nibble ind RAM with A	1	1

BOOLEAN VARIABLE MANIPULATION

Mnemonic		Description	Byte	Cyc
CLR	C	Clear Carry flag	1	1
CLR	bit	Clear direct bit	2	1
SETB	C	Set Carry flag	1	1
SETB	bit	Set direct Bit	2	1
CPL	C	Complement Carry flag	1	1
CPL	bit	Complement direct bit	2	1
ANL	C,bit	AND direct bit to Carry flag	2	2
ANL	C,\ bit	AND complement of direct bit to Carry	2	2
ORL	C,bit	OR direct bit to Carry flag	2	2
ORL	C,\ bit	OR complement of direct bit to Carry	2	2
MOV	C,bit	Move direct bit to Carry flag	2	1
MOV	bit,C	Move Carry flag to direct bit	2	2

PROGRAM AND MACHINE CONTROL

Mnemonic		Description	Byte	Cyc
ACALL	addr 11	Absolute Subroutine Call	2	2
LCALL	addr 16	Long Subroutine Call	3	2
RET		Return from subroutine	1	2
RETI		Return from interrupt	1	2
AJMP	addr 11	Absolute Jump	2	2
LJMP	addr 16	Long Jump	3	2
SJMP	rel	Short Jump (relative addr)	2	2
JMP	@A+DPTR	Jump indirect relative to the DPTR	1	2
JZ	rel	Jump if Accumulator is Zero	2	2
JNZ	rel	Jump if Accumulator is Not Zero	2	2
JC	rel	Jump if Carry flag is set	2	2
JNC	rel	Jump if No Carry flag	2	2

PROGRAM AND MACHINE CONTROL (cont.)

Mnemonic		Description	Byte	Cyc
JB	bit,rel	Jump if direct Bit set	3	2
JNB	bit,rel	Jump if direct Bit Not set	3	2
JBC	bit,rel	Jump if direct Bit is set & Clear bit	3	2
CJNE	A,direct,rel	Compare direct to A & Jump if Not Equal	3	2
CJNE	A,#data,rel	Comp. immed. to A & Jump if Not Equal	3	2
CJNE	Rn,#data,rel	Comp. immed. to reg & Jump if Not Equal	3	2
CJNE	@Ri,#data. rel	Comp. immed. to ind. & Jump if Not Equal	3	2
DJNZ	Rn,rel	Decrement register & Jump if Not Zero	2	2
DJNZ	direct. rel	Decrement direct & Jump if Not Zero	3	2
NOP		No operation	1	1

Notes on data addressing modes:

Rn	– Working register R0-R7
direct	– 128 internal RAM locations, any I/O port, control or status register
@Ri	– Indirect internal RAM location addressed by register R0 or R1
#data	– 8-bit constant included in instruction
#data 16	– 16-bit constant included as bytes 2 & 3 of instruction
bit	– 128 software flags, any I/O pin, control or status bit

Notes on program addressing modes:

addr 16	– Destination address for LCALL & LJMP may be anywhere within the 64-k program memory address space
Addr 11	– Destination address for ACALL & AJMP will be within the same 2-k page of program memory as the first byte of the following instruction
rel	– SJMP and all conditional jumps include an 8-bit offset byte. Range is +127–128 bytes relative to first byte of the following instruction.

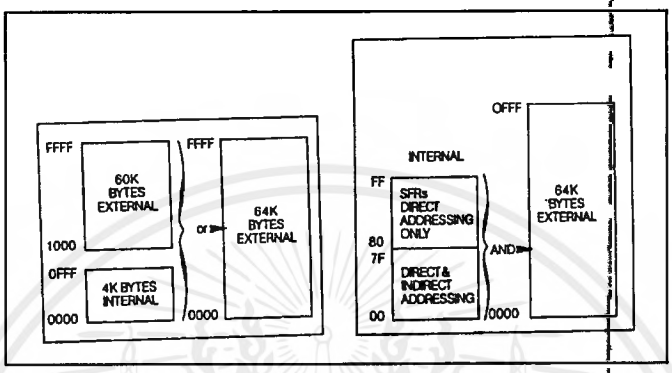
All mnemonics copyrighted © Intel Corporation 1979

รูปที่ 2.17 (ต่อ) ชุดคำสั่งของ 8051

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.8 รีจิสเตอร์ของ 8051

หน่วยความจำของ 8051 แบ่งออกเป็น 2 แบบ คือ หน่วยความจำโปรแกรม (Program Area) และหน่วยความจำสำหรับเก็บข้อมูล (Data Area) ดังแสดงในไดอะแกรม



รูปที่ 2.18 ไดอะแกรมหน่วยความจำของ 8051

หน่วยความจำโปรแกรมเป็นหน่วยความจำที่ใช้สำหรับเก็บโปรแกรมภาษาเครื่องที่จะทำงานเมื่อเริ่มป้อนไฟเลี้ยง หรือมีการรีเซ็ต (Reset) ซึ่งจะทำให้เริ่มการทำงานจากคำสั่งในตำแหน่งที่ 0000H เมื่อทำงาน 1 คำสั่ง ก็จะทำให้รีจิสเตอร์ PC ที่ชี้ตำแหน่งโปรแกรมมีค่าเพิ่มขึ้นเพื่อชี้ตำแหน่งของคำสั่งต่อไป ตำแหน่งสุดท้ายของหน่วยความจำ คือ 0FFFFH หน่วยความจำโปรแกรมนี้สามารถที่จะเลือกได้ว่าเป็นหน่วยความจำที่อยู่ภายในหรือภายนอก 8051 ก็ได้ หน่วยความจำในช่วงนี้สามารถอ่านข้อมูลได้อย่างเดียว ไม่สามารถเขียนข้อมูลเข้าไปได้ในระหว่างการทำงาน

หน่วยความจำข้อมูลเป็นหน่วยความจำที่ใช้สำหรับเก็บ หรือพักข้อมูลระหว่างที่ทำงาน หน่วยความจำสำหรับข้อมูลมี 2 แบบ แบบที่หนึ่งมีขนาด 128 ไบท์ อยู่ภายใน 8051 อีกแบบหนึ่งจะมีขนาด 64 กิโลไบท์ ต้องต่อเพิ่มเติมเข้าไปภายนอก หน่วยความจำภายในตำแหน่ง 0-7FH นี้ สามารถอ้างถึงได้โดยตรง คือมีการสั่งให้อ่านหรือเขียนข้อมูลไปยังตำแหน่งนั้นได้โดยตรง แต่หน่วยจำตำแหน่ง 80H-0FFH เป็นรีจิสเตอร์ฟังก์ชันพิเศษ (Special Function Register, SFR) หน่วยความจำช่วงนี้ใช้เป็นรีจิสเตอร์สำหรับงานเฉพาะอย่าง หน่วยความจำข้อมูลภายในช่วงตำแหน่ง 00-7FH สามารถแบ่งได้เป็น 3 กลุ่มคือ

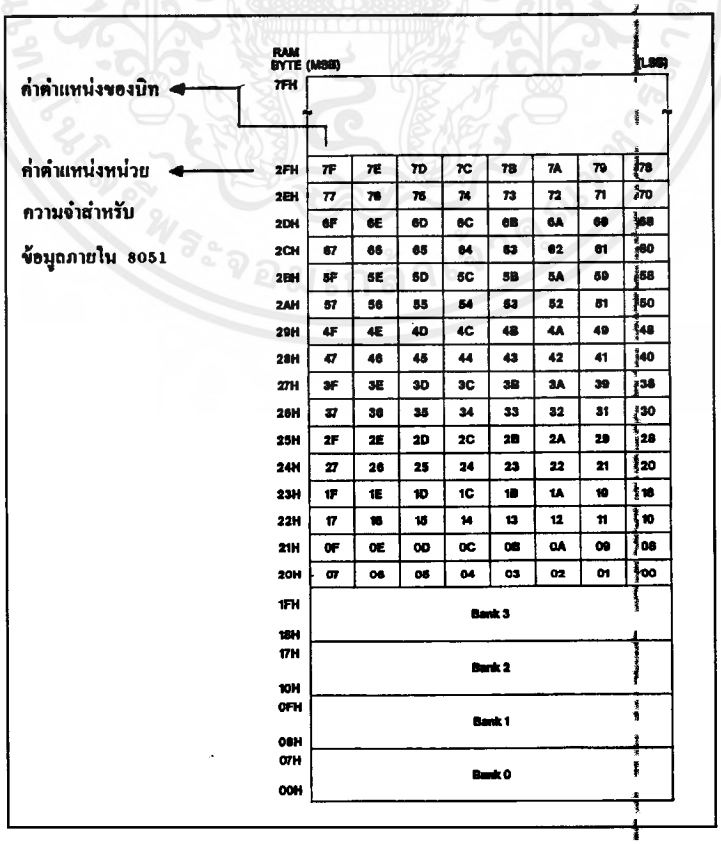
- Register Bank 0-3 อยู่ในตำแหน่งที่ 00-1FH หน่วยความจำนี้จะแบ่งออกเป็น 4 ชุด ชุดละ 8 ไบท์ แต่ละชุดเรียกว่า BANK ในแต่ไบท์จะมีชื่อของรีจิสเตอร์เป็น R0-R7 รีจิสเตอร์เหล่านี้จะเรียกใช้งานในระหว่างการทำงานของโปรแกรมได้อย่างสะดวก และรีจิสเตอร์เหล่าจะเป็นชื่อที่ซ้ำกันในทุก BANK การใช้งานจึงต้องเรียกใช้งานที่ละ BANK เท่านั้น โดยการกำหนดในรีจิสเตอร์ PSW เมื่อมีการรีเซ็ตการทำงานจะเริ่มใช้งานรีจิสเตอร์ R0-R7 ใน BANK 0 ซึ่งรีจิสเตอร์ในแต่ละ BANK จะอ้างอิงหน่วยความจำสำหรับข้อมูล 8051 ดังตาราง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2.4 การอ้างถึงหน่วยความจำของรีจิสเตอร์ BANK 0-3

รีจิสเตอร์	ตำแหน่งหน่วยความจำ			
	BANK 0	BANK 1	BANK 2	BANK 3
R0	0	8	10	18
R1	1	9	11	19
R2	2	A	12	1A
R3	3	B	13	1B
R4	4	C	14	1C
R5	5	D	15	1D
R6	6	E	16	1E
R7	7	F	17	1F

- Bit Address Area อยู่ในช่วงตำแหน่ง 20H-2FH โดยแต่บิทในช่วงของหน่วยความจำดังกล่าวจะสามารถตรวจสอบหรือตั้งค่าเป็น 0, 1 ได้โดยโปรแกรมภาษาเครื่อง แต่ละบิทของข้อมูลในหน่วยความจำช่วงนี้จะมีค่าของตำแหน่งดังในตารางเมโมรี่ โดยตัวเลขทางซ้ายเป็นค่าตำแหน่งของหน่วยความจำข้อมูลภายในซึ่งแต่บิทในตำแหน่งนั้นจะมีค่าเป็นเลขฐาน 16 ที่จะใช้เป็นค่าอ้างอิงในคำสั่งจัดการกับข้อมูลบิทนั้น

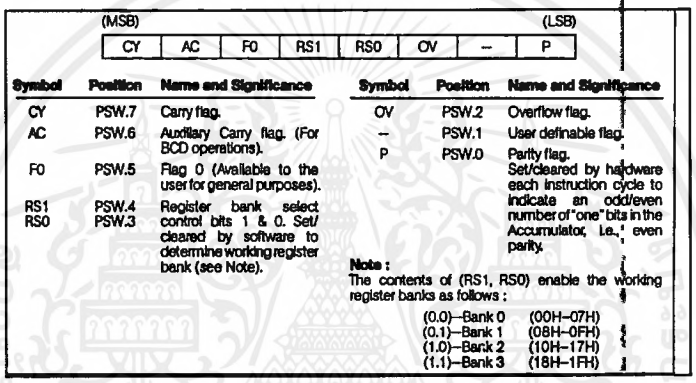


รูปที่ 2.19 ตำแหน่งบิทของข้อมูลในหน่วยความจำ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- Scratched Pod Area อยู่ในช่วงตำแหน่ง 30H-7FH ใช้สำหรับเก็บข้อมูลทั่วไป ถ้ารีจิสเตอร์ Stack Pointer ชี้มายังหน่วยความจำช่วงนี้จะต้องระวังไม่ให้เกิดการเขียนทับของข้อมูลซึ่งจะทำให้การทำงานของโปรแกรมผิดพลาดได้

คำสั่งของ 8051 เป็นคำสั่งที่มีประสิทธิภาพการทำงานสูงมาก ในขณะที่กำลังทำงานจะมีรีจิสเตอร์ตัวหนึ่งที่เก็บสภาวะ (Flag) ที่เกิดขึ้นระหว่างการคำนวณ เช่น ตัวทด (Carry) หรือจะใช้เลือก BANK ของรีจิสเตอร์ภายในก็ได้ รีจิสเตอร์นี้คือ Program Status Word (PSW) มีขนาด 8 บิทแต่ละบิทจะใช้เก็บสถานะการทำงานต่างๆ ไว้ดังรูป



รูปที่ 2.20 รีจิสเตอร์ PSW

- PWS.0 เรียกว่าบิทพาริตี้ (Parity) บิทนี้จะบอกว่าในรีจิสเตอร์ A มี 1 เป็นจำนวนคี่หรือคู่ ถ้าในรีจิสเตอร์ A มี 1 อยู่เป็นจำนวนคี่ ก็จะทำให้บิทนี้มีค่าเป็น 1
- PSW.1 บิทนี้ไม่มีการใช้งาน
- PSW.2 เรียกว่า Overflow Flag เป็นบิทที่บอกว่าการคำนวณนั้นเกิดตัวทดขึ้น ตัวทดนี้เป็นตัวทดที่เกิดจากบิท 6 ไปยัง บิท 7 จะมีประโยชน์เมื่อทำการคำนวณแบบ Signed Interger
- PSW.3, PSW.4 2 บิทนี้จะใช้งานร่วมกันเพื่อเป็นตัวบอกว่าขณะนี้ใช้รีจิสเตอร์ R0-R7 ใน BANK ใดดังตาราง

ตารางที่ 2.5 การเลือก BANK ของรีจิสเตอร์ R0-R7

บิท 4 (RB1)	บิท 3 (RB0)	Register Bank	Address
0	0	0	00H-07H
0	1	1	08H-0FH
1	0	2	10H-17H
1	1	3	18H-1FH

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- PSW.5 เป็นบิทที่ผู้ใช้สามารถใช้คำสั่งกำหนดให้เป็น 0 หรือ 1 ก็ได้ โดยการทงานของคำสั่งอื่นจะไม่ทำให้ค่าของบิทนี้เปลี่ยนแปลง บิทนี้ประโยชน์สำหรับการส่งสถานะของโปรแกรมระหว่างเรียกการทำงานของโปรแกรมย่อย
- PSW.6 เรียกว่า Auxiliary Carry Flag เป็นบิทที่ใช้สำหรับเก็บตัวทดที่เกิดขึ้นระหว่างการคำนวณโดยตัวทดนี้เกิดจาการคำนวณของบิทที่ 3 ข้ามไปยังบิท 4
- PSW.7 เรียกว่า Carry Flag เป็นบิทที่บอกสถานะการคำนวณทางคณิตศาสตร์ว่าผลลัพธ์ที่ได้ทำให้เกิดตัวทดขึ้นหรือไม่ เช่น การบวกเลขที่มีผลลัพธ์มากกว่า 255 ก็จะทำให้เกิดตัวทดขึ้น

2.4.9 แอดเดรสซิ่ง (Addressing)

2.4.9.1 Direct Addressing เป็นการกำหนดตำแหน่งที่อยู่ของหน่วยความจำที่จะติดต่อเข้าไปใน Operand ของคำสั่งโดยตรง วิธีการนี้ใช้ในการติดต่อหน่วยความจำข้อมูลในตัว 8051 จำนวน 256 ตำแหน่งเท่านั้น เช่น DEC 20H เป็นการลดค่าข้อมูลของหน่วยความจำข้อมูลภายในที่ตำแหน่ง 20H ลง 1

2.4.9.2 Indirect Addressing เป็นการกำหนดที่อยู่ของหน่วยความจำข้อมูลภายในโดยอ้อม วิธีนี้จะใช้รีจิสเตอร์ตัวหนึ่งเป็นตัวชี้ไปยังหน่วยความจำที่ต้องการ รีจิสเตอร์ที่ใช้ได้แก่ R1, R2, DPTR เป็นต้น คำสั่งนี้จะใช้สัญลักษณ์ @ นำหน้ารีจิสเตอร์ที่เป็นตัวชี้ เช่น DEC @R1 เป็นคำสั่งลดค่าข้อมูลในหน่วยความจำตำแหน่งที่ชี้ด้วยค่าในรีจิสเตอร์ R1 ลง 1

2.4.9.3 Register Instruction เป็นคำสั่งที่ใช้ติดต่อกับรีจิสเตอร์ R0-R7 ใน Operand จะมีชื่อของรีจิสเตอร์อยู่ ซึ่งเมื่อแปลเป็นภาษาเครื่องแล้วจะมีอยู่ 3 บิทที่เป็นตัวระบุ R0-R7 คำสั่งนี้จะเปลี่ยนเป็นภาษาเครื่องได้ 1 ไบท์ซึ่งจะแตกต่างกันที่ 3 บิทสุดท้ายเท่านั้น เช่น MOV A,R0

Register-Specific Instruction เป็นคำสั่งที่มีการทำงานเฉพาะกับรีจิสเตอร์บางตัว คำสั่งเหล่านี้จะไม่มีตำแหน่งของหน่วยความจำใน OP-CODE ที่จะไปชี้รีจิสเตอร์ใดๆ เพราะ CPU จะรู้ว่าเป็นรีจิสเตอร์โดยอัตโนมัติ เช่น CLR A

2.4.9.4 Immediate Constant เป็นคำสั่งเกี่ยวกับค่าคงที่โดยตรง คำสั่งนี้จะมีการกำหนดส่วนของค่าคงที่ใน Operand เช่น MOV A,#100 คำสั่งนี้เป็นการกำหนดค่า 100 ไปเก็บในรีจิสเตอร์ A เครื่องหมาย # ใช้สำหรับบอกว่าค่าที่ตามหลังมาเป็นค่าคงที่

2.4.9.5 Index Addressing เป็นการกำหนดเลขที่อยู่โดยตรรชนี การอ้างหน่วยความจำวิธีนี้ใช้ได้เฉพาะกับการติดต่อหน่วยความจำสำหรับโปรแกรมเท่านั้น ซึ่งจะใช้รีจิสเตอร์ DPTR หรือ Program Counter ขนาด 16 บิท บวกด้วยรีจิสเตอร์ A ขนาด 8 บิท แล้วนำผลลัพธ์ไปชี้ตำแหน่งหน่วยความจำโปรแกรมเพื่ออ่านข้อมูลออกมา คำสั่งนี้มีประโยชน์ในการอ่านข้อมูลที่เก็บไว้ในรูปแบบของตารางที่อยู่ในหน่วยความจำโปรแกรม เช่น MOVC A,@A+DPTR

2.4.10 ชุดคำสั่ง8051

ชุดคำสั่ง 8051 แบ่งออกได้เป็น 5 กลุ่ม ซึ่งจะใช้ตารางเป็นตัวอธิบายโดยประกอบด้วย 4 คอลัมน์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คอลัมน์ 1 คือ Mnemonic เป็นช่องที่บอกถึงรหัสคำสั่ง

คอลัมน์ 2 คือ Operation เป็นการกระทำที่เกิดขึ้นตามรหัสคำสั่ง

คอลัมน์ 3 คือ Addressing Mode แบ่งเป็นส่วนย่อยๆ คือ

Dir = Direct Addressing

Ind = Indirect Addressing

Reg = Register Addressing

Imm = Immediate Addressing

เครื่องหมาย X ในช่องนี้หมายความว่า รหัสคำสั่งมีเครื่องหมาย <byte> อยู่ในส่วน Operand สามารถอ้างอิงตำแหน่งของหน่วยความจำได้ด้วยวิธีดังกล่าว

คอลัมน์ 4 คือ Execution Time (μs) เป็นเวลาที่ใช้ในการทำคำสั่งนั้น มีหน่วยเป็นไมโครวินาที โดยจะใช้สัญญาณนาฬิกา 12 เมกกะเฮิรตซ์ ดังนั้นถ้าใช้สัญญาณนาฬิกาต่ำกว่านี้ก็จะทำให้ Execution Time ยาวนานกว่าในตาราง

2.4.10.1 คำสั่งคณิตศาสตร์ (Arithmetic Instruction)

8051 มีคำสั่งการกระทำทางคณิตศาสตร์ที่สามารถบวก, ลบ, คูณ และหารกันได้ ซึ่งดีกว่าไมโครโพรเซสเซอร์เบอร์อื่น ที่ไม่มีคำสั่งการคูณและหาร

ตารางที่ 2.6 คำสั่งสำหรับการกระทำทางคณิตศาสตร์

Mnemonic	Operation	Addressing Modes				Execution Time(μs)
		Dir	Ind	Reg	Imm	
ADD A,<byte>	A=A+<byte>	X	X	X	X	1
ADDC A,<byte>	A=A+<byte>+C	X	X	X	X	1
SUBB A,<byte>	A=A-<byte>-C	X	X	X	X	1
INC A	A=A+1	Accumulator only				1
INC <byte>	<byte>=<byte>+1	X	X	X	X	1
INC DPTR	DPTR=DPTR+1	Data Pointer only				2
DEC A	A=A-1	Accumulator only				1
DEC <byte>	<byte>=<byte>-1	X	X	X		1
MUL AB	B:A=BxA	ACC and B only				4
DIV AB	A=Int(A/B) B=Mod (A/B)	ACC and B only				4
DA A	Decimal Adjust	Accumulator only				1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.10.2 คำสั่งทางตรรกศาสตร์ (Logical Instruction)

คำสั่งกลุ่มนี้มีการทำงานเหมือน Boolean Operation ซึ่งได้แก่ AND, OR, EXCLUSIVE-OR, NOT คำสั่งเหล่านี้ทำงานแบบบิตต่อบิต คือบิต 0-7 ของข้อมูลชุดที่ 1 ก็จะกระทำกับบิต 0-7 ของข้อมูลอีกชุดหนึ่งในลักษณะบิต 0 ต่อบิต 0, บิต 1 ต่อบิต 1 จนถึงบิต 7 ต่อบิต 7

ตารางที่ 2.7 คำสั่งทางตรรกศาสตร์

Mnemonic	Operation	Addressing Modees				Execution Time (μs)
		Dir	Ind	Reg	Imm	
ANL A,<byte>	A=A.AND.<byte>	X	X	X	X	1
ANL ,byte>,A	<byte>=<byte>.AND.A	X				1
ANL,byte.,#data	<byte>=<byte>.AND.#data	X				2
ORL A,<byte>	A=A.OR<byte>	X	X	X	X	1
ORL <byte>,A	<byte>=<byte>.ORA	X				1
ORL<byte>.#data	<byte>=<byte>.OR#data	X				2
XRL A,<byte>	A=A.XOR<byte>	X	X	X	X	1
XRL<byte>,A	<byte>=<byte>.XOR.A	X				1
XRL<byte>,#data	<byte>=<byte>.XOR#data	X				2
CLR A	A=00H	Accumulator only				1
CPL A	A=.NOT.A	Accumulator only				1
RL A	Rotate ACC Left 1 bit	Accumulator only				1
RLC A	Rotate Left through Carry	Accumulator only				1
RR A	Rotate Acc Right 1 bit	Accumulator only				1
RRC A	Rotate Right through Carry	Accumulator only				1
SWAP A	Swap Nibbles in A	Accumulator only				1

2.4.10.3 คำสั่งเคลื่อนย้ายข้อมูล (Data Transfer)

เป็นกลุ่มคำสั่งที่ใช้กันมาก ในคำสั่งเหล่านี้จะมีคำสั่งที่ใช้เคลื่อนย้ายข้อมูลจากตำแหน่งหนึ่งไปยังอีกตำแหน่งหนึ่ง และคำสั่งที่ใช้ในการแลกเปลี่ยนข้อมูล คำสั่งในกลุ่มนี้จะไม่มีผลต่อ Flag ใดๆ ใน PSW

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 2.8 คำสั่งเคลื่อนย้ายข้อมูล

Mnemonic	Operation	Addressing Modes				Execution Time (μs)
		Dir	Ind	Reg	Imm	
MOV A,<src>	A=<src>	X	X	X	X	1
MOV <dest>,A	<dest>=A	X	X	X		1
MOV <dest>,<src>	<dest>=<src>	X	X	X	X	2
MOV DPTR,#data16	DPTR=16bit immediate constant				X	2
PUSH <src>	INC SP:MOV "@SP",<src>	X				2
POP <dest>	MOV <dest>,"@SP":DEC SP	X				2
XCH A,<byte>	ACC and <byte> exchange data	X	X	X		1
XCHD A,@Ri	ACC and @Ri exchange low nibbles		X			1

คำสั่งเคลื่อนย้ายข้อมูลกับหน่วยความจำสำหรับข้อมูลภายนอก 8051

คำสั่งเคลื่อนย้ายข้อมูลที่กล่าวมาแล้ว เป็นการเคลื่อนย้ายข้อมูลกับหน่วยความจำสำหรับข้อมูลภายใน 8051 เท่านั้น การติดต่อข้อมูลกับหน่วยความจำที่อยู่ภายนอก 8051 จะต้องใช้รีจิสเตอร์ RO, R1 หรือ DPTR เป็นตัวชี้ของตำแหน่งหน่วยความจำเท่านั้นการใช้ RO, R1 เป็นตัวชี้ตำแหน่งจะทำให้อ้างอิงตำแหน่งได้เพียง 256 ตำแหน่ง แต่การใช้ DPTR ชี้ตำแหน่ง ทำให้สามารถชี้ตำแหน่งได้ถึง 64*1024 ตำแหน่ง

ตารางที่ 2.9 คำสั่งสำหรับการติดต่อหน่วยความจำข้อมูลภายนอก

Address Width	Mnemonic	Operation	Execution Time (μs)
8 bits	MOVX A,@Ri	Read external RAM @Ri	2
8 bits	MOVX @Ri,A	Write external RAM @Ri	2
16 bits	MOVX A,@DPTR	Read external RAM @DPTR	2
16 bits	MOVX @DPTR,A	Write external RAM @ DPTR	2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำสั่งอ่านข้อมูลจากตาราง (Lookup Table)

คำสั่งในกลุ่มนี้จะมีเฉพาะการอ่านข้อมูลจากตารางซึ่งเป็นหน่วยความจำโปรแกรมมาเก็บไว้ยัง Accumulator ตารางนี้จะต้องเขียนไว้ในหน่วยความจำตั้งแต่ขั้นตอนของการเขียนโปรแกรมแล้ว

ตารางที่ 2.10 คำสั่งอ่านข้อมูลจากตาราง

Mnemonic	Operation	Execution Time (μs)
MOVC A, @A+DPTR	Read Pgm Memory at (A+DPTR)	2
MOVC A, @A+PC	Read Pgm Memory at (A+PC)	2

2.4.10.4 คำสั่งบูลีน

คำสั่งของ 8051 นอกจากจะมีการกระทำทีละ 8 บิตแล้ว หน่วยความจำข้อมูลภายใน 8051 ยังสามารถติดต่อ หรือมีการกระทำต่อข้อมูลทีละบิต คือแต่ละบิตของรีจิสเตอร์ SRF ในช่วงหน่วยความจำสำหรับข้อมูลตำแหน่ง 20H ถึง 2FH จำนวน 16 ไบท์ (มีจำนวนบิตที่สามารถติดต่อได้ 128 บิต หรือ 128 ตำแหน่ง)

ตารางที่ 2.11 คำสั่งบูลีน

Mnemonic	Operation	Execution Time (μs)
ANL C,bit	C=C,AND.bit	2
ANL C,/bit	C=C,AND..NOT.bit	2
ORL C,bit	C=C.OR.bit	2
ORL C,bit	C=C,ORNOT.bit	2
MOV C,bit	C=bit	1
MOV bit,C	bit=C	2
CLR C	C=0	1
CLR bit	bit=0	1
SETBC	C=1	1
SETB bit	bit=1	1
CPL C	C=.NOT.C	1
CPL bit	bit=.NOT.bit	1
JC rel	Jump ifC=1	2
JNC rel	Jump if C=0	2
JB bit,rel	Jump if bit=0	2
JNB bit,rel	Jump if bit=0	2
JBC bit,rel	Jump if bit=1:CLR bit	2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คำสั่ง ANL, ORL ,MOV, CRL จะมีการกระทำเหมือนกับในกลุ่มคำสั่งตรรกศาสตร์ทุกประการ แตกต่างกันที่การอ้างอิงตำแหน่งหน่วยความจำ คำสั่งในตารางจะมีคำว่า bit ปรากฏอยู่หมายความว่าที่ตำแหน่งนั้นของ Operand จะต้องป้อนค่าตำแหน่งหน่วยความจำที่เป็นแบบบิต เช่น บิต 0 ของหน่วยความจำตำแหน่ง 20H จะมีค่าหน่วยความจำแบบบิตเท่ากับ 00H และบิต 7 ของหน่วยความจำตำแหน่ง 20H จะมีตำแหน่งแบบหน่วยความจำเท่ากับ 07H ตัวอย่าง ต้องการAND ข้อมูลของ Carry Bit ใน PSW ด้วยข้อมูลบิต 7 ของหน่วยความจำตำแหน่ง 20H ใช้คำสั่ง

ANL C,7H

2.4.10.5 กลุ่มคำสั่งกระโดดข้าม (Jump Intruction)

กลุ่มคำสั่งกระโดดข้าม คือ กลุ่มของคำสั่งที่ทำให้การทำงานของโปรแกรมกระโดดข้ามไปยังตำแหน่งที่กำหนดโดยไม่ต้องทำตามลำดับของโปรแกรมเดิม

ตารางที่ 2.12 คำสั่งกระโดดข้ามแบบไม่มีเงื่อนไข

Memonic	Operation	Execution Time (μs)
JMP addr	Jump to addr	2
JMP @A+DPTR	Jump to A+DPTR	2
CALLaddr	Call subroutine at addr	2
RET	Return from subroutine	2
RETI	Return from interrupt	2
NOP	No operation	1

คำสั่งข้ามการทำงานแบบมีเงื่อนไข (Conditioned Jump Instruction)

คำสั่งให้ข้ามการทำงานข้างบนนั้นเป็นการข้ามแบบไม่มีเงื่อนไข คือไม่ต้องมีการตรวจสอบใดๆ ก่อนจะข้ามการทำงาน คำสั่งอีกกลุ่มหนึ่งจะต้องตรวจสอบเงื่อนไขก่อนการข้ามการทำงานมีคำสั่งดังนี้

ตารางที่ 2.13 คำสั่งกระโดดข้ามแบบมีเงื่อนไข

Memonic	Operation	Addressing Mode				Execution Time(μs)
		Dir	Ind	Reg	Imm	
JZ rel	Jump if A=0	Accumulator only				2
JNZ rel	Jump if A=0	Accumulator only				2
DJNZ<byte>,rel	Decrement and jump if not zero	X		X		2
CJNE a,<byte>,rel	Jump if A=<byte>	X			X	2
CJNE<byte>,#data,rel	Jump if <byte>=#data		X	X		2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.11 รีจิสเตอร์ฟังก์ชันพิเศษ (Special Function Register, SFR)

ใน 8051 จะใช้วิธีการกำหนดชื่อให้กับตำแหน่งของหน่วยความจำสำหรับข้อมูลภายใน (Internal Data Memory) ที่เรียกว่า Symbolize เช่นการให้ชื่อหน่วยความจำแต่ละตำแหน่งในแต่ละ BANK ซึ่งอยู่ในช่วงหน่วยความจำตำแหน่ง 00H-1FH แล้วในคำสั่งจะอ้างหน่วยความจำแต่ละตำแหน่งโดยการใช้ชื่อ RO, R1, R2, R3, R4, R5, R6 และ R7 หน่วยความจำตำแหน่งเหล่านี้ จะเรียกอีกอย่างหนึ่งว่าเป็นรีจิสเตอร์ซึ่งมีหน้าที่ในการเก็บหรือพักข้อมูล หรือใช้สำหรับการกระทำบางอย่าง รีจิสเตอร์กลุ่มหนึ่งใน 8051 ที่เรียกว่า (Special Function Register, SFR) เป็นรีจิสเตอร์ที่ใช้สำหรับงานเฉพาะ คือข้อมูลที่ถูกนำไปเก็บไว้ในรีจิสเตอร์เหล่านี้จะมีความหมายเฉพาะตัว ที่แต่ละตำแหน่งของ SFR อาจจะไม่ใช่เป็นหน่วยความจำ (RAM) แต่อาจเป็นตัวนับ (Count Register), Shift Register หรือ Latch ซึ่งการอ้างอิงในแต่ละตำแหน่งนั้น จะถือเสมือนว่าเป็นหน่วยความจำตำแหน่งหนึ่ง จึงเรียกการมองข้อมูลแต่ละตำแหน่งนี้ว่า Memory Map I/O รีจิสเตอร์กลุ่มนี้มีดังนี้

ตารางที่ 2.14 รีจิสเตอร์ฟังก์ชันพิเศษ

Symbol	Name	Address
*ACC	ACCULULATOR	0E0H
*B	B REGISTER	0F0H
*PSW	PROGRAM STATUS WORD	0D0H
SP	STACK POINTER	81H
DPTR	DATA POINTER 2 BYTE	-
DPL	LOW BYTE	82H
DPH	HIGHBYTE	83H
*P0	PORT 0	80H
*P1	PORT 1	90H
*P2	PORT 2	0A0H
*P3	PORT 3	0B0H
*IP	INTERRUPT PRIORITY CONTROL	0B8H
*IE	INTERRUPT ENABLE CONTROL	0ABH
TMOD	TIMER/COUNTER MODE CONTROL	89H
*TCON	TIMER/COUNTER CONTROL	88H
*+T2CON	TIMER/COUNTER 2 CONTROL	0C8H
TH0	TIMER/COUNTER 0 HIGH BYTE	8CH
TH1	TIMER/COUNTER 0 LOW BYTE	8AH

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TL0	TIMER/COUNTER 0 HIGH BYTE	8DH
TL1	TIMER/COUNTER 1 LOW BYTE	8BH
+TH2	TIMER/COUNTER 2 HIGH BYTE	0CDH
+TL2	TIMER/COUNTER 2 LOW BYTE	0CCH
+RCAP2H	T/C 2 CAPTURE REG HIGH BYTE	0CBH
+RCAP2L	T/C 2 CAPTURE REG LOW BYTE	0CAH
*SCON	SERIAL CONTROL	98H
SBUF	SERIAL DATA BUFFER	99H
PCON	POWER CONTROL	87H
*IOCON	IO CONTROL	0F8H

ช่อง Symbol ทางซ้ายจะเป็นสัญลักษณ์ของรีจิสเตอร์ ในช่องถัดมาคือชื่อของรีจิสเตอร์ตามสัญลักษณ์ที่อยู่ข้างซ้าย ในช่องขวาสุดจะเป็นตำแหน่งของหน่วยความจำสำหรับข้อมูลภายใน 8051 ที่แทนด้วยชื่อหรือสัญลักษณ์ทางซ้ายนั้นเอง เช่นในบรรทัดแรกคือรีจิสเตอร์ชื่อ Accumulator ที่มีสัญลักษณ์ ACC รีจิสเตอร์นี้คือหน่วยความจำข้อมูลภายใน ที่ตำแหน่ง 0E0H การอ่านหรือการเขียนรีจิสเตอร์กับข้อมูลเหล่านี้สามารถทำได้ โดยการใช้คำสั่งในกลุ่มการใช้ข้อมูล (เช่น MOV A, #25H หรือ MOV 0E0H,#25H) และรีจิสเตอร์ในกลุ่มนี้ยังสามารถใช้คำสั่งในกลุ่ม Boolean Instruction เพื่อการทำงานในแต่บิตในรีจิสเตอร์เหล่านี้ได้ จากตาราง รีจิสเตอร์ที่มีเครื่องหมาย * อยู่ข้างหน้าจะสามารถใช้คำสั่งในกลุ่ม Boolean Istruction จัดการกับแต่ละบิตได้ รีจิสเตอร์ที่มีเครื่องหมาย + นำหน้าหมายความว่า รีจิสเตอร์นั้นมีเฉพาะใน 80C52 และ 83C154 เท่านั้นไม่มีใน 8051

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Direct Byte Address	Bit Address (MSB)							(LSB)	Special Function Register Symbol
	WDT	T32	SERR	IZC	P3HZ	P2HZ	P1HZ	ALF	
0F8H	FF	FE	FD	FC	FB	FA	F9	F8	IOCON
0F0H	F7	F6	F5	F4	F3	F2	F1	F0	B
0E0H	E7	E6	E5	E4	E3	E2	E1	E0	ACC
	CY	AC	F0	RS1	RS0	OV	F1	P	
0D0H	D7	D6	D5	D4	D3	D2	D1	D0	PSW
0CDH	Not Bit Addressable								TH2
0CCH	Not Bit Addressable								TL2
0CBH	Not Bit Addressable								RCAP2H
0CAH	Not Bit Addressable								RCAP2L
	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2	CP/RL2	
0C8H	CF	CE	CD	CC	CB	CA	C9	C8	T2CON
	PCT		PT2	PS	PT1	PX1	PT0	PX0	
0B8H	BF	-	BD	BC	BB	BA	B9	B8	IP
0B0H	B7	B6	B5	B4	B3	B2	B1	B0	P3
	EA		ET2	ES	ET1	EX1	ET0	EX0	
0A8H	AF	-	AD	AC	AB	AA	A9	A8	IE
0A0H	A7	A6	A5	A4	A3	A2	A1	A0	P2
99H	Not Bit Addressable								SBUF
	SM0	SM1	SM2	REN	TB8	RB8	TI	RI	
98H	9F	9E	9D	9C	9B	9A	99	98	SCON
90H	97	96	95	94	93	92	91	90	P1
8DH	Not Bit Addressable								TH1
8CH	Not Bit Addressable								TH0
8BH	Not Bit Addressable								TL1
8AH	Not Bit Addressable								TL0
89H	Not Bit Addressable								TMOD
	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	
88H	8F	8E	8D	8C	8B	8A	89	88	TCON
87H	Not Bit Addressable								PCON
83H	Not Bit Addressable								DPH
82H	Not Bit Addressable								DPL
81H	Not Bit Addressable								SP
80H	87	86	85	84	83	82	81	80	P0

รูปที่ 2.21 รีจิสเตอร์ฟังก์ชันพิเศษ

ในช่องสี่เหลี่ยมเล็กๆ จะเป็นตำแหน่งของบิตนั้นในแต่ละรีจิสเตอร์ เช่น ในช่องซ้ายสุดของรีจิสเตอร์ TCON มีค่า 8FH ซึ่งเป็นค่าตำแหน่งบิต 7 ของหน่วยความจำตำแหน่ง 88H ถ้าต้องการให้บิตนั้นมีค่าเป็น 0 ก็สามารถทำได้โดยใช้คำสั่ง

CLR 8FH

หรือจะทำให้บิตนี้เป็น 1 ก็ทำได้โดยคำสั่ง

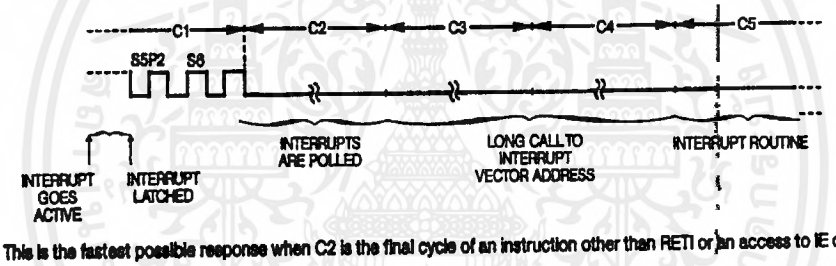
SETB 8FH

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.12 การขัดจังหวะ (Interrupt)

การขัดจังหวะคือสภาวะหนึ่งที่คอมพิวเตอร์ทำงานอยู่แล้วถูกขัดจังหวะด้วยสัญญาณหรือคำสั่งพิเศษที่ทำให้คอมพิวเตอร์ต้องละจากงานที่กำลังทำอยู่ ไปทำงานในโปรแกรมตอบสนองการขัดจังหวะนั้น เมื่อเสร็จแล้วก็จะกลับมาทำงานเดิมต่อไปได้ ใน 8051 จะสามารถขัดจังหวะการทำงานได้ 6 แหล่ง คือ

- INT0, INT1 เป็น 2 ขาของ 8051 ที่จะรับสัญญาณจากภายนอก การขัดจังหวะจะเกิดขึ้นถ้าสัญญาณที่ขาดังกล่าวมีสภาวะลอจิกเป็น 0 หรือเปลี่ยนจาก 1 เป็น 0 โดยการเลือกด้วยการกำหนดในบิต IT0 หรือ IT1 ในรีจิสเตอร์ TCON
- TF0, TF1 เป็นบิตหนึ่งที่จะบอกการทำงานของ Timer 0, Timer 1 เมื่อเกิด Overflow ขึ้น ใน Timer จะทำให้บิตนี้เป็น 1 และเกิดการขัดจังหวะการทำงานของ 8051 ได้
- TI, RI เป็น 2 บิต ในรีจิสเตอร์ SCON ถ้าบิตนี้ถูกเซต ให้เป็น 1 โดยฮาร์ดแวร์ อันเนื่องจากเสร็จสิ้นการส่งหรือรับข้อมูลจะสามารถทำให้เกิดการขัดจังหวะได้



รูปที่ 2.22 การทำงานเมื่อเกิดการขัดจังหวะ

8051 จะทำการอ่านสัญญาณจากทั้ง 6 แหล่งที่เวลา S5P2 ของทุกๆ ไซเคิลของเครื่อง (Machine Cycle) เข้ามาเก็บ และในช่วงของไซเคิลของเครื่องถัดไป ก็จะตรวจสอบสถานะของสัญญาณทั้ง 6 ที่เก็บเข้ามา ถ้าสัญญาณนั้นมีการขัดจังหวะที่ถูกต้อง 8051 ก็จะละทิ้งการทำงานเดิมไว้ชั่วคราวแล้วสร้างคำสั่ง LCALL ขึ้นมาภายใน 8051 เพื่อไปทำงานในโปรแกรมตอบสนองการขัดจังหวะแต่ละสัญญาณนั้น เมื่อทำงานในโปรแกรมตอบสนองการขัดจังหวะเสร็จแล้วก็สามารถกลับมาทำงานเดิมได้ โดยใช้คำสั่ง RETI เป็นคำสั่งสุดท้ายในโปรแกรมตอบสนองการขัดจังหวะสัญญาณขัดจังหวะจากแต่ละแหล่งจะมีตำแหน่งหน่วยความจำที่จะเก็บโปรแกรมตอบสนองการขัดจังหวะไว้ต่างกันดังนี้

สัญญาณที่ขอขัดจังหวะ		ตำแหน่งเริ่มต้นโปรแกรมตอบสนองการขัดจังหวะ
1	INT0	0003H
2	TF0	000BH
3	INT1	0013H
4	TF1	001BH
5	TI, RI	0023H

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตำแหน่งเริ่มต้นในโปรแกรมนี้ เป็นตำแหน่งใน Program area เช่น ถ้าสัญญาณของ INT0 เข้ามาแล้ว ตรวจสอบว่ามีการขอขัดจังหวะถูกต้อง ก็จะละทิ้งการทำงานเดิม แล้วไปทำงานที่โปรแกรมตอบสนองการขัดจังหวะที่มีตำแหน่งเริ่มต้นอยู่ที่ตำแหน่ง 0003H เมื่อเสร็จสิ้นการทำงานของโปรแกรมตอบสนองการขัดจังหวะจะต้องมีคำสั่ง RETI อยู่เพื่อกลับมาสู่การทำงานเดิมได้ 8051 จะทำการตรวจสอบสัญญาณดังกล่าวว่ามีสัญญาณใดของการขัดจังหวะมาบ้าง โดยใช้วิธี Polling คือการตรวจสอบเรียงตามลำดับจาก 1, 2, 3, 4 และ 5 ตามลำดับดังนั้นถ้ามีการขอการขัดจังหวะเข้ามาพร้อมๆ กัน 8051 ซึ่งตรวจสอบการขอขัดจังหวะแบบ Polling จะพบว่าสัญญาณมีการขอขัดจังหวะจากสัญญาณต้นๆ ก่อนจึงตอบสนองของการขอขัดจังหวะของสัญญาณต้นๆ ก่อน หรืออีกนัยหนึ่งก็คือสัญญาณขอขัดจังหวะที่1 จะมีลำดับความสำคัญสูงสุด (Highest Priority) และสัญญาณที่ 5 จะมีลำดับความสำคัญต่ำสุด (Lowest Priority) อย่างไรก็ตามสามารถที่จะจัดลำดับความสำคัญของสัญญาณขัดจังหวะนี้ใหม่เพื่อให้มีการตอบสนองการขัดจังหวะสัญญาณของการขัดจังหวะลำดับหลังได้ โดยการโปรแกรมในรีจิสเตอร์ IP (Interrupt Priority Register) และจะสามารถกำหนดว่าจะให้ทำโปรแกรมตอบสนองการขัดจังหวะ เมื่อมีสัญญาณขอการขัดจังหวะเข้ามาหรือไม่ก็ได้ โดยโปรแกรมในรีจิสเตอร์ IE (Interrupt Enable Register)

2.4.12.1 เงื่อนไขในการตอบสนองการขัดจังหวะ

เมื่อ 8051 ทำการตรวจสอบสัญญาณขอการขัดจังหวะที่เก็บเข้ามาเมื่อเวลา S5P2 แล้วพบว่ามีการขอขัดจังหวะนั้น แม้ว่ามีการ Enable ในรีจิสเตอร์ IE ถูกต้อง แต่จะต้องมีเงื่อนไขดังนี้ด้วย

2.4.12.1.1 ไม่ได้กำลังทำงานในโปรแกรมตอบสนองการขัดจังหวะของสัญญาณขัดจังหวะที่ลำดับความสำคัญสูงกว่าหรือเท่ากัน เช่น กำลังทำงานในโปรแกรมตอบสนองการขัดจังหวะของสัญญาณ INT0 อยู่ แล้วมีการขอขัดจังหวะจากสัญญาณ INT1 อีก จะไม่เกิดการทิ้งงานเดิม คือไม่มีการไปทำงานที่โปรแกรมตอบสนองการขัดจังหวะของสัญญาณ INT1

2.4.12.1.2 เนื่องจากการสุ่มสัญญาณเข้าไปตรวจสอบนั้นจะทำที่เวลา S5P2 ของไซเคิลสุดท้ายของคำสั่ง และคำสั่งที่อยู่ถัดมาจะต้องใช้เวลาทำงาน 2 ไซเคิลของเครื่อง ดังนั้นการตรวจสอบการกระทำในไซเคิลแรก แม้ว่าจะมีการขอการขัดจังหวะเข้ามาก็จะไม่ทำโปรแกรมตอบสนองขัดจังหวะ จะต้องอ่านสัญญาณที่เวลา S5P2 อีกครั้ง แล้วไปตรวจสอบบนไซเคิลที่ 2 ของคำสั่ง ถ้ามีการขอขัดจังหวะถูกต้อง จึงจะข้ามไปทำงานในโปรแกรมตอบสนองการขัดจังหวะ

2.4.12.1.3 คำสั่งที่กำลังทำงานอยู่ขณะที่ตรวจสอบสัญญาณขอขัดจังหวะ จะต้องไม่ใช่คำสั่ง RET หรือคำสั่งใดๆ ก็ตามที่พยายามเขียนข้อมูลไปยังรีจิสเตอร์ IE หรือ IP

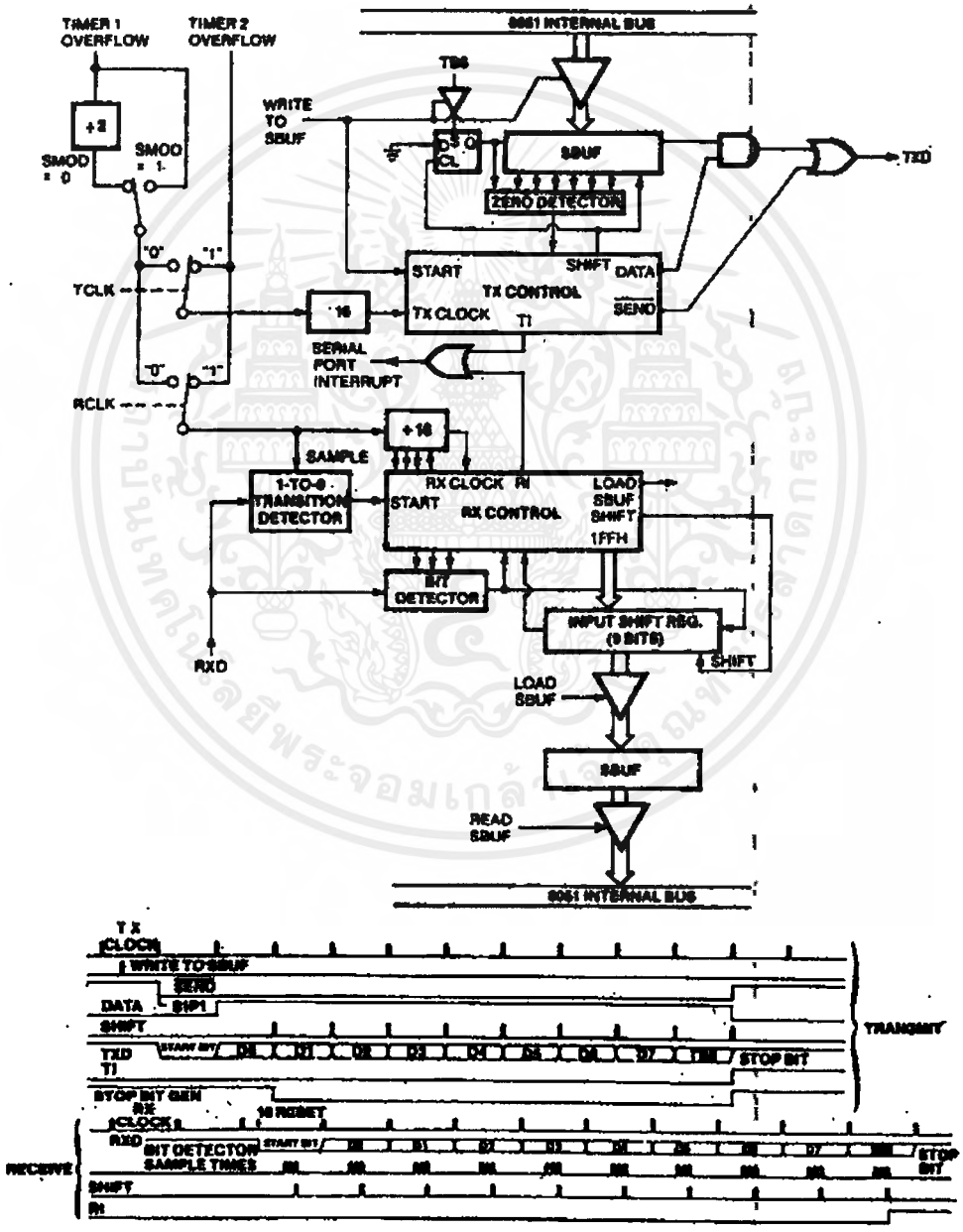
สัญญาณขอขัดจังหวะที่ถูกอ่านเข้าไปที่เวลา S5P2 นี้ ไม่ว่าได้รับการตอบสนองหรือไม่ ก็จะถูกทิ้งไปแล้วอ่านเข้าไปใหม่ทุกเวลา S5P2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.13 การรับส่งข้อมูลทางพอร์ตอนุกรม

ในการรับส่งข้อมูลแบบอนุกรมนั้น จะต้องมีการกำหนดโหมดการทำงานในรีจิสเตอร์ SCON และในบางโหมดของการทำงานจะสามารถกำหนดอัตราการส่งข้อมูลได้โดยการโปรแกรมในไทม์เมอร์ ข้อมูลที่จะส่งออกหรือรับเข้าจะอยู่ที่รีจิสเตอร์ SBUF การทำงานของแต่ละโหมดเป็นดังนี้

2.4.13.1 การทำงานโหมด 0



รูปที่ 2.23 การรับส่งข้อมูลทางพอร์ตอนุกรมโหมด 0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.13.1.1 การส่งข้อมูล

การส่งข้อมูลจะเริ่มจากการทำงานของคำสั่งเคลื่อนย้ายข้อมูลเข้ามายังรีจิสเตอร์ SBUF โดยจะมีสัญญาณ Write to SBUF ซึ่งเกิดขึ้นในช่วงเวลา S6P2 สัญญาณนี้จะทำให้ข้อมูลจากบัสภายในทางด้านบนถูกนำไปเก็บที่รีจิสเตอร์ SBUF และเซ็ตให้ D FLIP-FLOP ที่อยู่ทางด้านซ้ายของ SBUF มีสภาวะลอจิกเป็น 1 สัญญาณ Write to SBUF ซึ่งต่อไปยังขา Strat ของ TX Control จะบอกให้ TX Control เริ่มส่งข้อมูล โดยจะหน่วงเวลาไว้ 1 ไซเคิลของเครื่อง เมื่อผ่านเวลา 1 ไซเคิลของเครื่องไปแล้วขึ้นไซเคิลใหม่ สัญญาณ SEND จะเปลี่ยนสภาวะลอจิกจาก 0 เป็น 1 และเริ่มส่งข้อมูลบิท 0 จากนี้ในทุกเวลา S6P2 สัญญาณ SHIFT จะเปลี่ยนเป็น 1 ออกจากวงจร TX Control (ที่เวลาอื่นจะเป็น 0) ทำให้ข้อมูลใน SBUF ถูกเลื่อนออกไปทีละบิทตรงขอบขาลงของสัญญาณ SHIFT ในทุกไซเคิลของเครื่อง ขณะที่ข้อมูลถูกเลื่อนออกไปทางขวานั้น ข้อมูลจาก D FLIP-FLOP จะเลื่อนเข้ามาทางซ้าย ข้อมูลจาก D-FF บิทแรกที่เลื่อนเข้ามาจะเป็น 1 เพราะถูกเซ็ตไว้ในตอนแรก แต่บิทต่อมาจะเป็น 0 เพราะขา D ถูกต่อไว้ที่กราวด์ ขณะที่สัญญาณ SEND มีลอจิกเป็น 1 จะทำให้เอาท์พุทของ OR Gate มีลอจิกเป็น 1 ดังนั้นสัญญาณจากวงจร Shift Clock จะถูกส่งออกไปทางขา P3.1 (TXD) โดยสัญญาณนี้จะเป็น 1 ในช่วงเวลา S6P2 ของไซเคิลเครื่องถัดไป และจะเป็น 0 ในช่วงเวลา S3P1 ถึง S5P2 ในไซเคิลเครื่องเดียวกัน สัญญาณจากขา TXD ที่ส่งออกไปนี้ก็เพื่อให้อุปกรณ์ปลายทางสามารถรับข้อมูลได้ถูกต้องเพราะถูกส่งออกไปพร้อมกับข้อมูลใน SBUF ข้อมูลจำนวน 8 บิทจะถูกเลื่อนออกทางขา RXD จนครบทั้ง 8 บิท เมื่อข้อมูลบิทสุดท้ายถูกส่งออกไปจะทำให้ 1 ที่เกิดจาก D-FF ถูกเลื่อนมาอยู่ทางขวาสุดของรีจิสเตอร์ SBUF และทางซ้ายทั้งหมดเป็น 0 ทำให้วงจร Zero Detector ซึ่งตรวจสอบค่า 0 นี้ส่งสัญญาณไปบอกวงจร TX Control ว่าสิ้นสุดการส่งข้อมูลออก เมื่อสิ้นสุดการส่งข้อมูลสัญญาณ SEND จะเปลี่ยนสภาวะลอจิก 1 เป็น 0 ที่ขอบขาลงของสัญญาณ SHIFT และการส่งข้อมูลบิทสุดท้ายออกไปจะทำให้บิท T1 ในรีจิสเตอร์ SCON เปลี่ยนจาก 0 เป็น 1 บอกการสิ้นสุดของการส่งข้อมูล

2.4.13.1.2 การรับข้อมูล

การรับข้อมูลในโหมด 0 จะเริ่มต้นรับข้อมูลเข้ามาทางขา RXD ก็ต่อเมื่อมีการทำงานของคำสั่ง Set ค่าบิท REN เป็น 1 และเคลียร์บิท RI เป็น 0 ในรีจิสเตอร์ SCON ซึ่งการทำงานของคำสั่งนี้จะทำให้เริ่มรับข้อมูลที่เวลา S6P2 เสร็จแล้วจะหน่วงเวลาไปจนถึง S6P6 ในไซเคิลของเครื่องถัดไปก็จะทำให้สภาวะลอจิกของสัญญาณ Receive เปลี่ยนจาก 1 เป็น 0 เพื่อเริ่มส่งสัญญาณ Clock จากวงจร Shift Clock ถูกส่งออกไปทางขา TXD โดยสัญญาณ Clock ที่ขานี้จะมีลอจิกเป็น 1 ตั้งแต่ S6P1 จนถึง S6P6 ของไซเคิลถัดไป และมีสภาวะลอจิกเป็น 0 ในช่วงเวลา S3P1 ถึง S5P2 แต่ก่อนที่สัญญาณ Receive จะเปลี่ยนเป็น 1 นั้นวงจร RX Control จะเขียนข้อมูล 11111110B เข้าไปยัง Input shift Register จากนั้นในทุกๆ เวลา S6P2 ซึ่งสัญญาณ Shift มีสภาวะลอจิกเป็น 1 (ที่เวลาอื่นจะเป็น 0) จะเลื่อนข้อมูลใน Input shift Register ไปทางซ้ายทีละบิท เมื่อครบ 7 ครั้ง ข้อมูล 0 ที่อยู่ทางขวาสุดของรรีจิสเตอร์ในตอนเริ่มต้นจะเลื่อนมาอยู่ที่ตำแหน่งซ้ายสุด และบอกให้กับ RX Control ว่ามีข้อมูลบิทสุดท้ายอีก 1 บิทที่จะเข้ามา เมื่อข้อมูลสุดท้ายเข้ามาที่เวลา

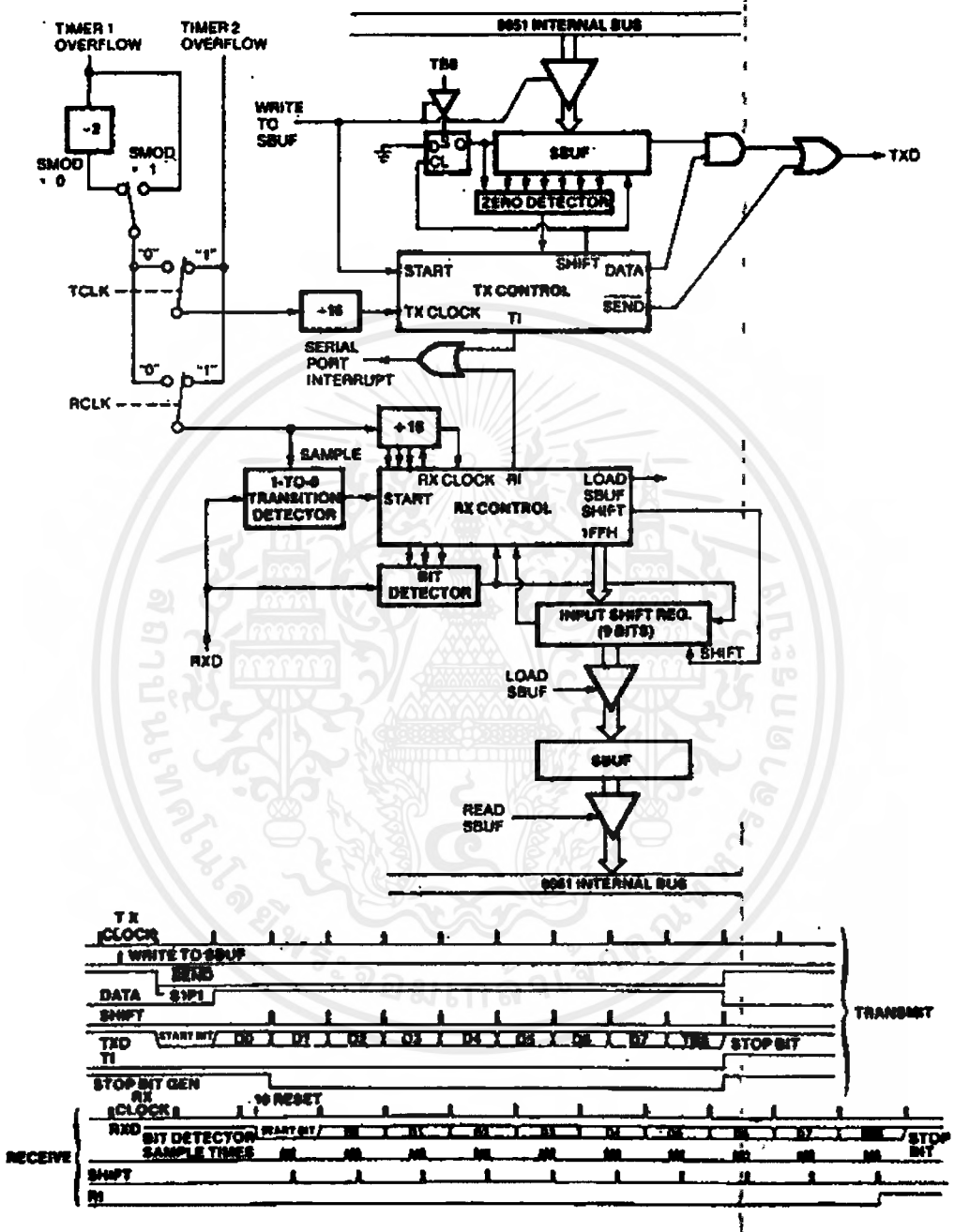
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

S5P2 ของไซเคิลของเครื่องลำดับที่ 9 นับตั้งแต่เริ่มรับข้อมูล จะทำให้ที่เวลา S1P1 ของไซเคิลของเครื่องที่ 10 นับตั้งแต่เริ่มมีสัญญาณ Write to SCON จะเกิดการส่งข้อมูลจาก Input shift Register ไปเก็บยังรีจิสเตอร์ SBUF ต่อไป และในขณะเดียวกันก็จะทำให้บิท RI ซึ่งถูกเคลียร์ตั้งแต่เริ่มต้นรับข้อมูลถูกเซ็ตให้เป็น 1 และจะทำให้สัญญาณ Receive เป็น 0 ทำให้ไม่มีสัญญาณ Clock ออกไป

ในโหมดนี้จะมีการส่งข้อมูลชุดละ 10 บิท คือ Stsrt bit 1 บิท ข้อมูล 8 บิท และ Stop bit 1 บิท อัตราของการส่งข้อมูลจะขึ้นกับอัตราการเกิด Overflow ใน Timer 1 ข้อมูลนี้จะถูกส่งออก 1 บิท ทุก 16 หรือ 32 ครั้งของการเกิดOverflow ใน Timer 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.4.13.2 การทำงานโหมด 1



รูปที่ 2.24 การรับส่งข้อมูลทางพอร์ตอนุกรมโหมด 1

2.4.13.2.1 การส่งข้อมูล

จากรูปที่ 2.24 บิท SMOD จะเป็นตัวเลือกว่าสัญญาณ Overflow ของ Timer 1 ที่ส่งไปยังวงจรหาร 16 จะถูกหาร 2 ก่อนหรือไม่ ถ้า SMOD เป็น 1 สัญญาณ Overflow ของ Timer 1 จะไม่ถูกหาร แต่ถ้าเป็น 0 สัญญาณ Overflow ของ Timer 1 จะถูกหาร 2 ก่อนเข้าวงจรหาร 16 การส่งข้อมูลจะเริ่มจากการมีคำสั่งเขียนข้อมูลไปยังรีจิสเตอร์ SBUF จะมีสัญญาณ Wirte to SBUF เกิดขึ้นเพื่อรับข้อมูลจาก Internal Bus

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ด้านบนไปเก็บยังรีจิสเตอร์ SBUF และทำให้เอาท์พุทของ D-FF ทางด้านซ้ายของ SBUF มีค่าเป็น 1 และเป็นบิทที่ 9 ของการส่งข้อมูล สัญญาณ Wirte to SBUF ยังส่งไปยัง TX Control ด้วย ขณะนี้ข้อมูลในวงจรหาร 16 ยังไม่ทราบค่าจึงรอจนกว่าข้อมูลในวงจรหาร 16 นับเพิ่มขึ้นจนถึงค่าสูงสุดแล้ววนกลับเป็น 0 คือเกิดการวนกลับทำให้เริ่มการส่งข้อมูลที่เวลา S1P1 ของไซเคิลถัดไป สัญญาณ SEND จาก TX Control เปลี่ยนสภาวะลอจิกเป็น 0 แล้วเริ่มส่งข้อมูลที่เป็น Start bit ออกไป เมื่อส่งออกไปแล้ววงจร TX Control ก็จะทำให้สัญญาณ DATA เป็น 1 เพื่อเลื่อนข้อมูลใน SBUF ออกไป เริ่มจากบิท 0 จนถึงบิท 7 การส่งข้อมูลนี้จะเกิดขึ้นเมื่อสัญญาณ TX Control เปลี่ยนสถานะจาก 0 เป็น 1 ขณะที่ข้อมูลถูกเลื่อนออกไปนั้นจะมี 0 ถูกเลื่อนเข้าไปทางซ้ายของรีจิสเตอร์ SBUF เมื่อข้อมูลถูกเลื่อนออกไปทั้ง 8 บิทแล้ว บิทที่ 9 ซึ่งเป็น 1 จะถูกเลื่อนมาอยู่ในตำแหน่งสุดท้ายทางขวาของรีจิสเตอร์ SBUF และทางซ้ายของหลักนี้จะมี 0 อยู่ทั้ง 8 บิททำให้ Zero Detector รู้ว่าเป็นข้อมูลบิทสุดท้ายแล้วที่ส่งออก โดยจะมีสัญญาณบอกกับวงจร TX Control ด้วย เมื่อ TX Control ส่งสัญญาณ Shift ออกไปเป็นการส่งข้อมูลบิทสุดท้ายออกไป ก็จะรออีก 1 TX Clock ก็จะทำให้ขา TXD ส่งข้อมูล Stop Bit ออกมา สัญญาณ DATA ซึ่งมีสภาวะลอจิกเป็น 1 มาตั้งแต่เริ่มต้นก็จะกลับเป็น 0 และบิท TI จะเป็น 1 เพื่อบอกการสิ้นสุดของการส่งข้อมูล

2.4.13.2..2 การรับข้อมูล

การรับข้อมูลจะขึ้นอยู่กับการเกิด Overflow ใน Timer 1 แล้วหาร 2 หรือไม่ขึ้นกับค่าของบิท SMOD สัญญาณนี้จะไปเข้าวงจรหาร 16 และเป็นตัวกำหนดอัตราการรับข้อมูล จะเริ่มจากวงจร 1-to-0 Transition Detector พบว่าสัญญาณที่ขา RXD เปลี่ยนจาก 1 เป็น 0 ซึ่งหมายถึงข้อมูล Start bit เข้ามา การตรวจสอบนี้จำกระทำด้วยอัตราเดียวกับสัญญาณที่เข้าวงจรหาร 16 เมื่อพบการเปลี่ยนสถานะลอจิกที่ขา RXD ก็จะเริ่มการรับข้อมูล ขณะนี้จะรีเซ็ตวงจรหาร 16 ให้มีค่าเป็น 0 เพื่อสร้างสัญญาณ RX Clock ให้เข้าจังหวะ (Synchronous) กับข้อมูลที่เข้าโดยสัญญาณ RX Clock จะเป็น 1 เมื่อการนับของวงจรหาร 16 มีค่าเป็น 15 ขณะที่วงจร 16 จนถึง 7, 8และ 9 จะมีการตรวจสอบข้อมูลที่เข้ามาทางขา RXD เพื่อเป็นการตรวจว่าข้อมูลนั้นเป็นอะไร ถ้าอย่างน้อย 2 ใน 3 เป็นค่าใดก็จะถือว่าข้อมูลที่เข้ามาเป็นค่านั้น ถ้าในการตรวจสอบ Start bit แล้วพบว่าผิดพลาด คือไม่เป็น 0 ก็จะรีเซ็ตการทำงานเพื่อไปตรวจสอบการเปลี่ยนสถานะจาก 1 เป็น 0 ของข้อมูลที่ขา RXD ใหม่ แต่ถ้าพบ Start bit ก็จะเก็บข้อมูลทั้งหมดที่เข้ามาโดยเลื่อข้อมูลเข้าไปยัง Input Shift Register ที่มีสัญญาณควบคุมการเลื่อนข้อมูล (Shift) ส่งมาจาก RX control ในตอนเริ่มต้นการรับข้อมูลและมีการเขียนข้อมูล 1FFH ไปเก็บใน Input Shift Register ขณะที่ข้อมูลถูกเลื่อนเข้าไปทางขวาของ Input Shift Register จะมี 1 ถูกเลื่อนออกไปทางซ้ายทุกครั้งที่มีข้อมูลเข้า เมื่อ Start bit ที่รับเข้ามาถูกเลื่อนไปถึงซ้ายสุดของ Input Shift Register ก็จะมีสัญญาณไปบอก RX control Clock หลังจากข้อมูลบิทสุดท้ายเข้ามาแล้วก็จะโหลด (Load) เอาข้อมูล 8 บิทไปเก็บในรีจิสเตอร์ SBUF พร้อมทั้งเซ็ตค่าในบิท RI และ RB8 และรีจิสเตอร์ SCON แต่การโหลดข้อมูลไปเก็บนี้จะเกิดขึ้นได้ก็ต่อเมื่อ

- RI=0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- SM2=0 หรือถ้า SM2=1 จะต้องได้รับ Stop bit เป็น 1

ถ้าไม่มีภาวะใดสภาวะหนึ่งดังกล่าวแล้ว ข้อมูลที่รับเข้ามาก็จะถูกทิ้งไปคือไม่โหลดไปเก็บในรีจิสเตอร์ SBUF ถ้ามีสภาวะดังกล่าวถูกต้อง Stop bit จะถูกนำไปเก็บในรีจิสเตอร์ SBUF และบิท RI จะเป็น 1 แต่ไม่ว่าทั้ง 2 กรณีจะเกิดหรือไม่ก็จะกลับไปสู่การตรวจสภาวะเปลี่ยนจาก 1 เป็น 0 ที่ขา RXD เพื่อรับข้อมูลต่อไป

ในการรับข้อมูลแบบอนุกรมโหมด 1 นี้อัตราการส่งข้อมูลแต่ละบิท (Baud Rate) จะขึ้นกับอัตราการเกิด Overflow ใน Timer 1 ดังสมการ

$$Baud\ Rate = 2^{SMOD} / 32 * (Timer\ 1\ Overflow\ Rate)$$

ในขณะที่ใช้ Timer 1 เป็นตัวกำหนด Baund Rate นี้จะต้อง Disable ไม่ให้เกิดการขัดจังหวะเนื่องมาจาก Overflow Timer 1 อาจใช้ในโหมดของ Timer หรือ Counter ก็ได้ ซึ่งเมื่อการนับในรีจิสเตอร์ตัวนับมีค่าสูงสุดแล้วกลับมาเป็น 0 ก็จะเกิด Overflow เช่นเดียวกัน แต่โดยปกติแล้วจะใช้ Timer 1 นี้ในโหมดของ Timer ที่มีการทำงานแบบ Auto Reload โหมด 2 เพื่อว่าเมื่อค่าในการนับโดยรีจิสเตอร์ TL1 ถึงค่าสูงสุดก็จะโหลดค่าในรีจิสเตอร์ TH1 มาไว้ใน TL1 สำหรับค่าเริ่มต้นการนับต่อไป ซึ่ง Baud rate จะมีค่า

$$Baud\ Rate = (2^{SMOD} / 32) * (Oscillator\ frequency / (12 * (256 - TH1)))$$

โดยที่ SMOD เป็นบิทหนึ่งในรีจิสเตอร์ PCON

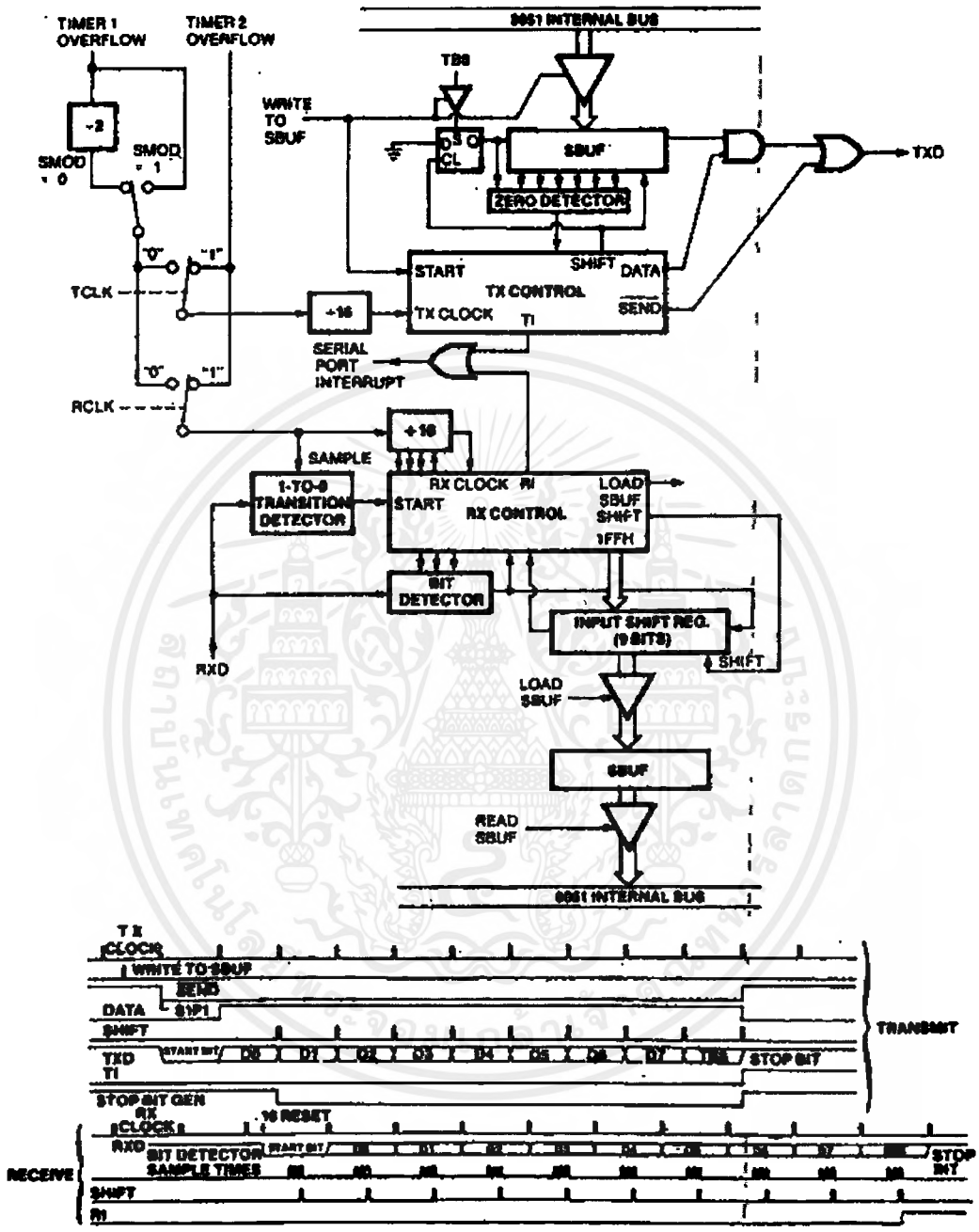
เช่นความถี่ของออสซิลเลเตอร์เท่ากับ 11.059 MHz บิท SMOD = 0 รีจิสเตอร์ TH1 มีค่า E8H, Timer 1 ทำงานในโหมด 2 จะได้อัตรการส่งรับขอมูลแบบอนุกรม

$$= (2^{0} / 32) * (11.059 * 10^{6} / (12 * (256 - 232)))$$

= 1200 บิท / วินาที

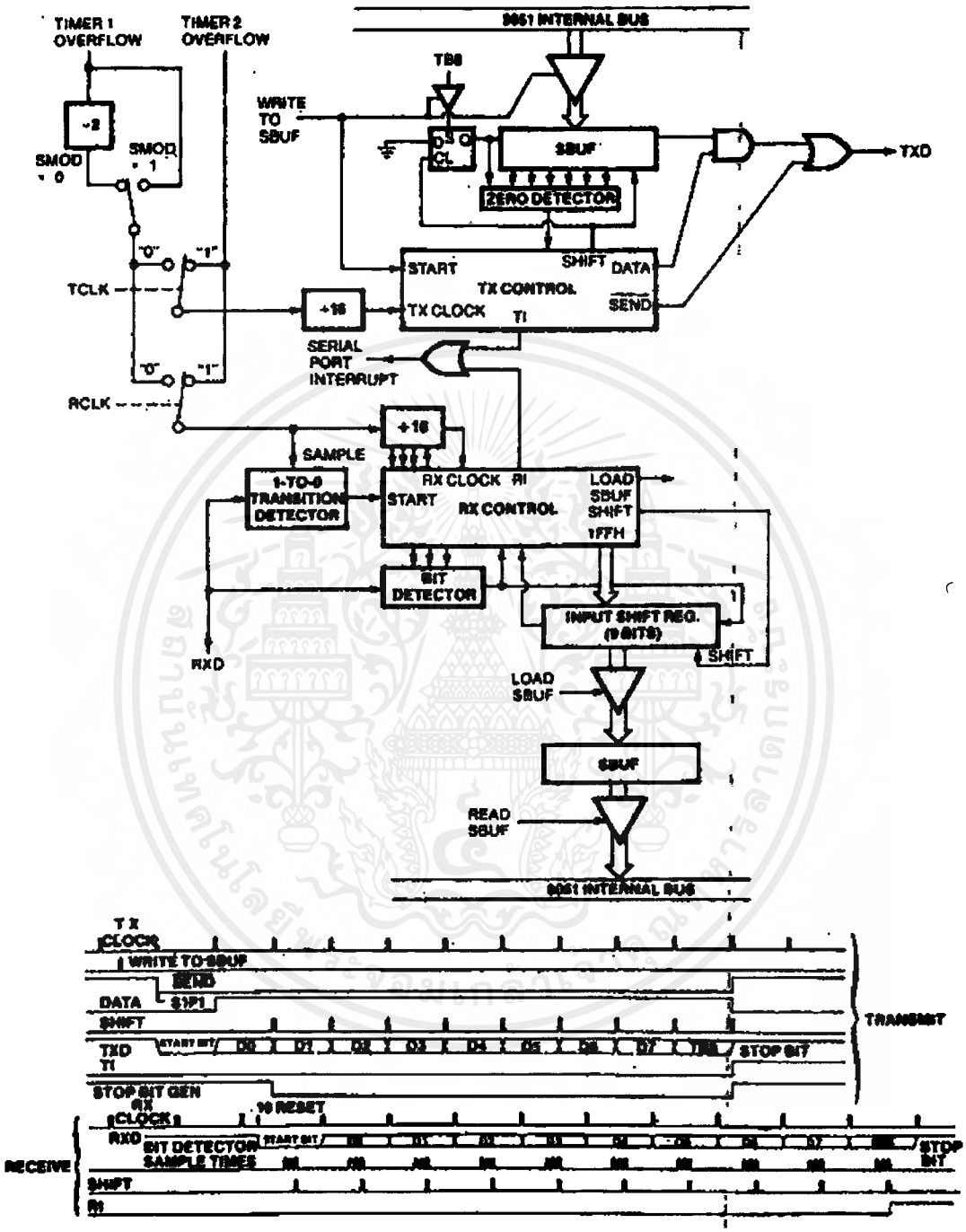
2.4.13.3 การทำงานโหมด 2 และโหมด 3

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.25 การรับส่งข้อมูลทางพอร์ตอนุกรมโหมด 2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.26 การรับส่งข้อมูลทางพอร์ตอนุกรมโหมด 3

โหมด 2 และโหมด 3 ของการรับส่งทางพอร์ตอนุกรมจะเหมือนกันแต่ต่างเฉพาะตรงที่อัตราการส่งข้อมูลเท่านั้น ในโหมด 2 จะเลือกอัตราการรับส่งข้อมูลได้เป็น 1/32 หรือ 1/64 เท่าของความถี่สัญญาณออสซิลเลเตอร์ ส่วนในโหมด 3 จะสามารถกำหนดอัตราการส่งข้อมูลได้โดยอัตราการส่งข้อมูลจะขึ้นกับอัตราการเกิด Overflow ของ Timer เช่นเดียวกับการรับส่งข้อมูลโหมด 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในรูปที่ 2.25 จะเห็นว่าสัญญาณจากเฟส 2 ของสัญญาณนาฬิกาซึ่งมีความถี่เท่ากับ 1/2 ของสัญญาณจากออสซิลเลเตอร์จะถูกเลือกโดยสวิตช์ SMOD ว่าจะให้หาร 2 หรือไม่ สวิตช์ SMOD นี้จะสามารถกำหนดตำแหน่งให้อยู่ข้างบนหรือข้างล่างโดยการกำหนดค่าของบิท SMOD ในรีจิสเตอร์ PCON ให้เป็น 1 หรือ 0 ตามลำดับ สัญญาณที่ออกจากการวงจรหาร 16 จะมีความถี่เป็น 1/32 หรือ 1/64 เท่าของสัญญาณจากออกซิลเลเตอร์

ในรูป 2.26 สัญญาณ Timer 1 Overflow อันเกิดจากการนับถึงค่าสูงสุดใน Timer1 แล้วค่าการนับจะกลับเป็น 0 ใหม่ สัญญาณนี้จะถูกเลือกโดยสวิตช์ SMOD ให้หาร 2 หรือไม่ก่อนที่จะส่งไปยังวงจรหาร 16 เพื่อสร้างเป็นสัญญาณควบคุมการเลื่อนข้อมูลเข้าหรือออกทางพอร์ตอนุกรมต่อไป การเลือกตำแหน่งของสวิตช์ SMOD รีจิสเตอร์ PCON ถ้าบิทนี้เป็น 1 สัญญาณ Timer 1 Overflow จะถูกส่งไปยังวงจรหาร 16 โดยตรง แต่ถ้าบิทเป็น 0 สัญญาณ Timer 1 Overflow จะถูกหาร 2 ก่อนแล้วจึงส่งเข้าไปยังวงจรหาร 16 การกำหนดอัตราการส่งของข้อมูลจะทำได้เหมือนกับในการรับ-ส่งข้อมูลพอร์ตอนุกรมโหมด 1 ที่กล่าวมาแล้วทุกประการ

การรับส่งข้อมูลในโหมด 2 และ 3 นั้น 1 ชุดของข้อมูลจะประกอบด้วย 1 Start Bit, 9 Data Bit 1 Stop Bit ทั้งหมดนี้จะส่งออกหรือรับเข้าทางพอร์ตอนุกรมทีละบิทเรียงตามลำดับ สามารถอธิบายการทำงานภายใน 8051 ในการส่งหรือรับข้อมูลได้ดังนี้

2.4.13.3.1 การส่งข้อมูล

การส่งข้อมูลจะเริ่มจากมีการเขียน (Write) ข้อมูลลงไปยังรีจิสเตอร์ SBUF ทำให้สัญญาณ Write to SBUF เกิดขึ้น แล้วข้อมูล 8 บิทจาก Internal Data Bus ของ 8051 ถูกเขียนลงไปในรีจิสเตอร์ SBUF และข้อมูลจากบิท TB8 ของรีจิสเตอร์ SCON จะเขียนลงไปยัง D FLIP-FLOP เช่นกัน รวมเป็น 9 เวลา S1P1 ของไซเคิลเครื่องแรกหลังจากสัญญาณ TX Clock นับครบ 1 รอบ จะทำให้สัญญาณ SEND ที่ออกจากวงจร TX Control เปลี่ยนสภาวะลอจิกเป็น 0 เพื่อเริ่มการส่งข้อมูล โดยจะเริ่มส่ง Start Bit (0) ออกไปทางขา TXD เมื่อสัญญาณ TX Clock เปลี่ยนเป็น 1 ครั้งต่อไปก็จะเริ่มส่งข้อมูลบิท 0 ออกไปและสัญญาณ DATA จะเป็น 1 ที่เวลานี้ด้วย TX Control จะส่งสัญญาณ Shift ไปยังรีจิสเตอร์ SBUF ทุกครั้งที่สัญญาณ TX Clock เป็น 1 เพื่อเลื่อนข้อมูลออกไปทางขวา และจะเลื่อน 0 เข้ามาทางซ้าย ขณะที่บิท TB8 อยู่ทางขวาสุด เตรียมส่งออกนั้นข้อมูลทางซ้ายทุกบิทจะเป็น 0 วงจร Zero Dedector จะมีสัญญาณไปบอก TX Control ในการส่งข้อมูลบิทสุดท้ายคือ TB8 ออกไปตามเวลาของสัญญาณ TX Clock (Bit time) เมื่อส่งข้อมูลบิท TB8 ออกไปเสร็จสิ้นแล้วก็จะรอเวลา 1 Bit Time สัญญาณ SEND ก็จะกลับเป็น 1 และบิท T1 ในรีจิสเตอร์ SCON จะเป็น 1 เพื่อบอกสิ้นสุดการส่งข้อมูลและสัญญาณ DATA ก็จะกลับเป็น 0 จากนั้นสัญญาณที่ออกจาก TXD ก็คือ Stop Bit นั่นเอง

2.4.13.3.2 การรับข้อมูล

การรับข้อมูลที่เข้ามาทางขา RXD จะเริ่มเมื่อวงจร 1 to 0 Transmistion Detector ทำการตรวจสอบข้อมูลที่ขา RXD 16 เท่าของอัตราการส่งข้อมูลแล้วพบว่าสัญญาณที่ขา RXD เปลี่ยนจาก 1 เป็น 0 ทำให้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

วงจรหาร 16 ถูกรีเซ็ตไปด้วย และจะเขียนข้อมูลในรีจิสเตอร์ SBUF เป็น 1FFH เมื่อ Counter ในวงจรหาร 16 นับถึง 7, 8 และ 9 จะมีการตรวจสอบข้อมูลที่เข้ามาทางขา RXD ถ้า 2 ใน 3 เท่ากันก็ถือว่าข้อมูลที่เข้ามามีค่านั้น ถ้าข้อมูลที่รับเข้ามาบิทแรกไม่เป็น 0 คือไม่ใช่ Start bit ก็จะเกิดการรีเซ็ตส่วนรับข้อมูลแล้วกลับไปเริ่มการตรวจสอบการเปลี่ยนสถานะจาก 1 เป็น 0 ใหม่ แต่ถ้าถูกต้องก็จะรับข้อมูลเข้ามาที่ละ 1 บิท ข้อมูลนี้จะถูกเลื่อนเข้าไปเก็บทางขวาของ Input Shift Register และ 1 จะถูกเลื่อนออกไปทางซ้ายโดยสัญญาณ Shift จนกระทั่งสัญญาณ Start Bit ซึ่งเป็น 0 ถูกเลื่อนเข้ามาถึงซ้ายสุดของ Input Shift Register จะบอกให้กับ RX Control รู้ว่าจะต้องมีข้อมูลบิทสุดท้ายเข้ามาอีก 1 บิท เมื่อข้อมูลบิทสุดท้ายเข้ามาจะเกิดการโหลดข้อมูลจาก Input Shift Register ไปยัง SBUF, RB8 และเซ็ต RI ให้เป็น 1 แต่สิ่งนี้จะเกิดขึ้นได้ต้องมีสภาวะดังนี้ด้วย

- บิท RI เป็น 0
- SM2 = 0 หรือถ้า SM2 = 1 ข้อมูลบิทที่ 9 จะต้องเป็น 1

ถ้าไม่มีสภาวะดังกล่าวทั้งสอง ข้อมูลที่รับมาจะถูกทิ้งไป ไม่นำมาเก็บที่ SBUF และ RI ก็จะไม่ถูกเซ็ต แต่ถ้าเกิดขึ้นทั้งสองกรณี ข้อมูลจะถูกนำไปเก็บที่รีจิสเตอร์ SBUF 8 บิท และบิทสุดท้ายจะนำไปเก็บที่บิท RB8 ของรีจิสเตอร์ SCON หลังจากนั้น 1 Bit Time ไม่ว่าจะมีการเก็บข้อมูลหรือไม่ ก็จะเข้าสู่การตรวจสอบสถานะการเปลี่ยนสัญญาณจาก 1 เป็น 0 เพื่อเตรียมรับข้อมูลต่อไป

2.5 หน่วยความจำ (Memory)

2.5.1 รอม (ROM:Read Only Memory)

หน่วยความจำแบบรอมจะใช้สำหรับเก็บข้อมูลโดยที่ข้อมูลเหล่านั้นจะไม่สูญหายถึงแม้จะไม่มีการจ่ายไฟเลี้ยงให้แก่ระบบ หน้าที่ของรอมในระบบไมโครโพรเซสเซอร์คือใช้เป็นที่เก็บโปรแกรมเพื่อควบคุมการทำงานของไมโครโพรเซสเซอร์ให้ทำงานได้ตามที่โปรแกรมไว้ ข้อมูลที่อยู่ภายในรอมสามารถอ่านออกมาได้แต่ไม่สามารถเขียนข้อมูลกลับเข้าไปได้ ซึ่งถ้าต้องการเขียนข้อมูลใส่ไว้ในรอมจะต้องใช้เครื่องมือพิเศษสำหรับการเขียนข้อมูลกับรอมโดยเฉพาะ เครื่องมือที่ใช้ในการเขียนข้อมูลให้กับบรอมก็แตกต่างกันไปตามชนิดของรอม ซึ่งมีอยู่หลายประเภท

2.5.1.1 การแบ่งชนิดของรอมสามารถแบ่งได้ดังนี้

- ROM (Read Only Memory) ข้อมูลทั้งหมดที่อยู่ภายในตัวรอมจะถูกโปรแกรมโดยผู้ผลิตหรือจากโรงงาน การใช้งานรอมก็ต่อเมื่อข้อมูลไม่มีการเปลี่ยนแปลงและมีความต้องการใช้งานเป็นจำนวนมาก
- PROM (Programable Read Only Memory) ข้อมูลที่ต้องการโปรแกรมสามารถโปรแกรมได้โดยผู้ใช้เอง โดยการป้อนพัลส์ที่มีแรงดันสูง (High Voltage Pulsed) เพื่อทำให้ Metel Strips หรือ Polycrstaline Silicon ที่มีอยู่ในตัว PROM ขาดออกจากกัน ทำให้เกิดลอจิก 1 หรือ 0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตามตำแหน่งที่กำหนดในหน่วยความจำนั้นๆ เมื่อ PROM ถูกโปรแกรมแล้วข้อมูลภายในจะไม่สามารถเปลี่ยนแปลงแก้ไขได้อีก

- EPROM (Erasable Programable Read Only Memory) ข้อมูลจะถูกโปรแกรมโดยผู้ใช้โดยการให้สัญญาณที่มีแรงดันสูง (High Voltage Signal) ผ่านเข้าไปในตัว EPROM ซึ่งเป็นวิธีเดียวกับที่ใช้ใน PROM แต่ข้อมูลที่อยู่ใน EPROM สามารถเปลี่ยนแปลงได้ โดยการลบข้อมูลเดิมที่อยู่ภายใน EPROM ออกก่อนแล้วจึงค่อยโปรแกรมเข้าไปใหม่ การลบข้อมูลนี้ทำได้ด้วยการฉายแสงอุลตร้าไวโอเลตเข้าไปในตัว EPROM โดยผ่านทางกระจกใสที่อยู่ด้านบนตัว EPROM เมื่อฉายแสงสักครู่หนึ่ง (ประมาณ 10-15 นาที) ข้อมูลที่อยู่ภายในก็จะถูกลบทิ้ง
- EAROM (Electrically Alterable Read Only Memory) ข้อมูลจะถูกโปรแกรมโดยผู้ใช้เหมือนใน EPROM แต่ข้อมูลของ EAROM สามารถลบได้ดดยทางไฟฟ้า ไม่ใช่โดยการฉายแสงแบบ EPROM

2.5.1.2 ขั้นตอนในการอ่านข้อมูลจากรอม

ลำดับขั้นตอนในการอ่านข้อมูลจากรอมในแต่ละครั้งมีลำดับขั้นตอนดังที่จะแสดงต่อไปนี้ ซึ่งเป็นขั้นตอนโดยทั่วๆ ไปที่ไม่คำนึงถึงชนิดของไมโครโพรเซสเซอร์ที่ใช้

2.5.1.2.1 ค่าแอสเดรสจะถูกป้อนเข้าไยังรอม ค่าแอสเดรสนี้จะกำหนดตำแหน่งของข้อมูลที่ต้องการอ่าน โดยข้อมูลจะถูกอ่านออกมาครั้งละ 1 ไบท์เท่านั้น

2.5.1.2.2 ไมโครโพรเซสเซอร์จะคอยอยู่ช่วงเวลาหนึ่ง (Wait State) เรียกว่า Access Time ประมาณ 100-300 nanoseconds ขึ้นอยู่กับชนิดของรอม ซึ่งเป็นเวลาที่ใช้ในการถอดรหัสของแอสเดรส (Decode Address) ของข้อมูลที่ต้องการอ่านออกมาที่เอาท์พุทของรอม

2.5.1.2.3 Chip Select Line จะถูกทำให้แอคทีฟ (Active) เพื่อให้ข้อมูลออกมาที่บัสข้อมูลของระบบได้ ต่อจากนั้นไมโครโพรเซสเซอร์จะสโตรป (Strobe) ข้อมูลเข้าไปเก็บไว้ในรีจิสเตอร์ภายใน

2.5.1.2.4 Chip Select Line จะถูกสั่งให้เลิกทำงาน (Inactive) เพื่อให้ข้อมูลที่อยู่บนบัสของระบบหายไป

2.5.2 แรม (RAM:Random Access Memory)

หน้าที่ของแรมจะคล้ายกับรอม คือเป็นที่เก็บโปรแกรมหรือข้อมูล แต่สิ่งที่แตกต่างกันก็คือแรมจะเก็บข้อมูลแบบชั่วคราว เมื่อไม่มีกระแสไฟฟ้าจ่ายในกับตัวแรม ข้อมูลที่อยู่ภายในก็จะสูญหายไป และแรมยังสามารถเขียนข้อมูลได้โดยง่ายไม่ต้องอาศัยเครื่องมือพิเศษใดๆ ช่วย

2.5.2.1 การทำงานของแรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แรมเป็นหน่วยความจำชนิดสารกึ่งตัวนำที่ใช้ในระบบไมโครโพรเซสเซอร์ ซึ่งสามารถอ่านข้อมูลและเขียนข้อมูลกลับเข้าไปได้ โดยไมโครโพรเซสเซอร์ต้องมีสัญญาณทางไฟฟ้าที่มีลักษณะเฉพาะสำหรับขบวนการเหล่านั้น ซึ่งประกอบด้วยขาสัญญาณดังต่อไปนี้

- ขาไฟเลี้ยงของแรมเป็นขาที่สำคัญขาหนึ่ง เนื่องจากแรมต้องมีไฟเลี้ยงจ่ายให้ตลอดเวลา มิเช่นนั้นข้อมูลที่อยู่ภายในก็จะสูญหายไป โดยทั่วไปแรมมักต้องการไฟเลี้ยงขนาด +5 โวลท์
- ขาข้อมูล (Data Line) เป็นส่วนที่ใช้ในการนำข้อมูลที่ต้องการเขียนเข้ามาในแรม และข้อมูลที่ต้องการอ่านออกจากแรม
- ขาแอสเดรส (Address Line) เป็นส่วนที่ใช้ทำหน้าที่สำหรับเลือกตำแหน่งของข้อมูลที่ต้องการเขียนหรืออ่าน
- ขาอ่านและเขียน (R/W) เป็นขาที่จะถูกใช้งานทุกครั้งที่มีการอ่านหรือเขียนข้อมูล

2.5.2.2 ขั้นตอนที่เกิดขึ้นเมื่อมีการอ่านแรม

2.5.2.2.1 สัญญาณระบุตำแหน่งของข้อมูลจะเข้ามาทางแอสเดรสบัส ในเวลาเดียวกันแรมจะนำสัญญาณนี้มาทำการถอดรหัสเพื่อหาตำแหน่งของข้อมูลภายใน

2.5.2.2.2 สัญญาณอ่าน/เขียน (R/W) จะถูกทำให้อยู่ในระดับของลอจิกที่ถูกต้อง ซึ่งจะเป็น 0 หรือ 1 ก็ขึ้นอยู่กับแรมชนิดนั้นๆ

2.5.2.2.3 ระบบจะคอยอยู่ช่วงเวลาหนึ่ง เรียกช่วงเวลานี้ว่า Read Access Time เพื่อให้วงจรภายในแรมถอดรหัสและเลือกข้อมูลในตำแหน่งนั้นๆ

2.5.2.2.4 หลังจากนั้นข้อมูลจะถูกส่งออกมาทางขาข้อมูล และถูกอ่านโดยไมโครโพรเซสเซอร์ของระบบ

2.5.2.3 ขั้นตอนที่เกิดขึ้นเมื่อทำการเขียนข้อมูลเข้าไปในแรม

2.5.2.3.1 ที่ขาแอสเดรสจะมีสัญญาณลอจิกที่ระบุถึงตำแหน่งภายในแรมซึ่งเป็นตำแหน่งที่ต้องการนำข้อมูลเข้าไปเขียน

2.5.2.3.2 ข้อมูลที่ต้องการจะถูกส่งมาที่ขาข้อมูล (Data Line)

2.5.2.3.3 รับบจะคอยอยู่ช่วงระยะเวลาหนึ่ง เรียกช่วงเวลานี้ว่า Write Access Time เพื่อที่จะให้วงจรถอดรหัสภายในทำงานอยู่ในสภาวะคงที่เสียก่อน

2.5.2.3.4 หลังจากคอยอยู่ช่วงเวลาหนึ่งแล้ว ขาสัญญาณอ่าน/เขียน ก็จะทำให้เป็นระดับของลอจิกที่ทำให้เกิดการเขียนขึ้น หรือเป็นพัลส์ (Pulse) เพื่อที่จะทำให้ข้อมูลถูกเขียนเข้าไปในแรม

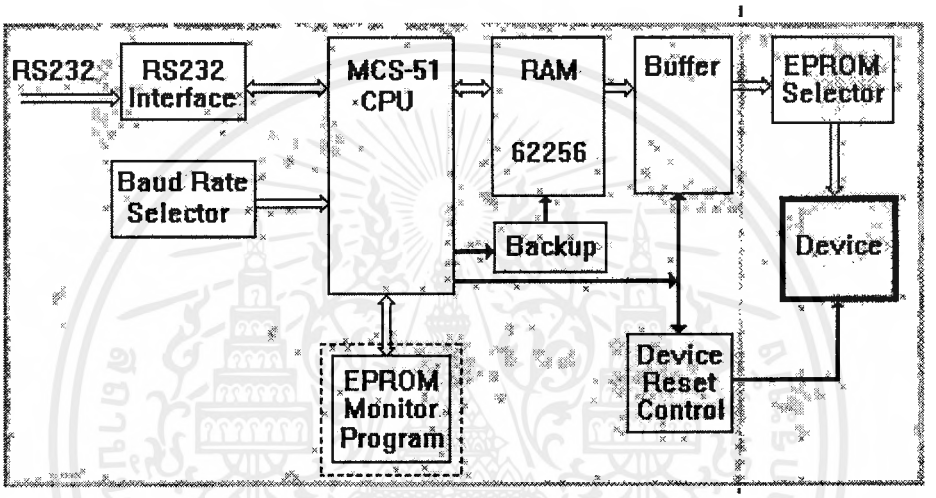
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3. การออกแบบและการสร้าง

3.1 การออกแบบและการสร้างในส่วนของฮาร์ดแวร์

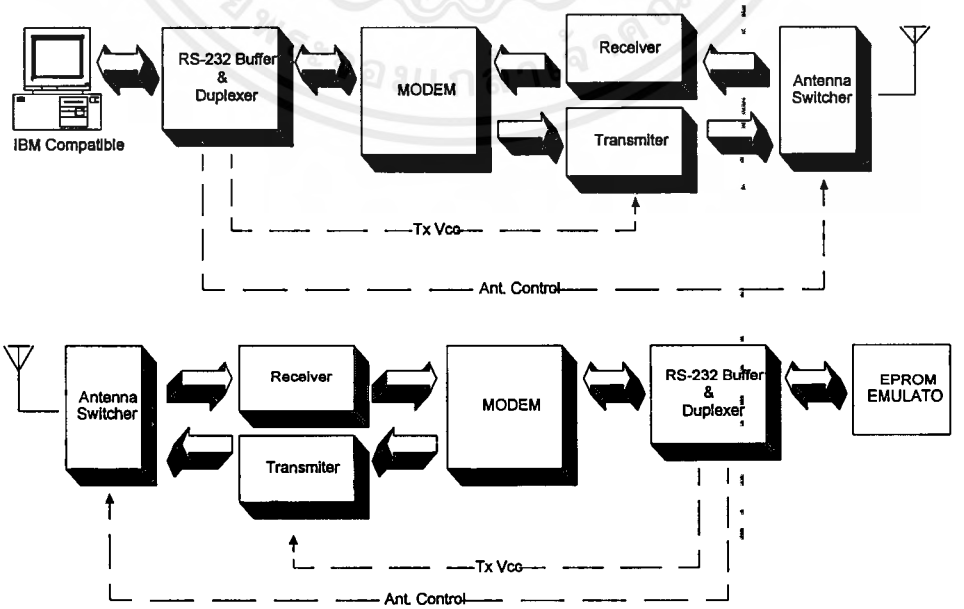
ในโครงงานเครื่องจำลองการทำงานของ EPROM แบบไร้สายนี้ จะสามารถแยกการทำงานออกเป็น 2 ส่วนใหญ่ๆ ได้ดังนี้คือ

- ส่วนของ EPROM Emulator ซึ่งใช้การติดต่อกับพอร์ตอนุกรม RS 232 ของเครื่องคอมพิวเตอร์ โดยใช้สายไฟเป็นสื่อกลาง



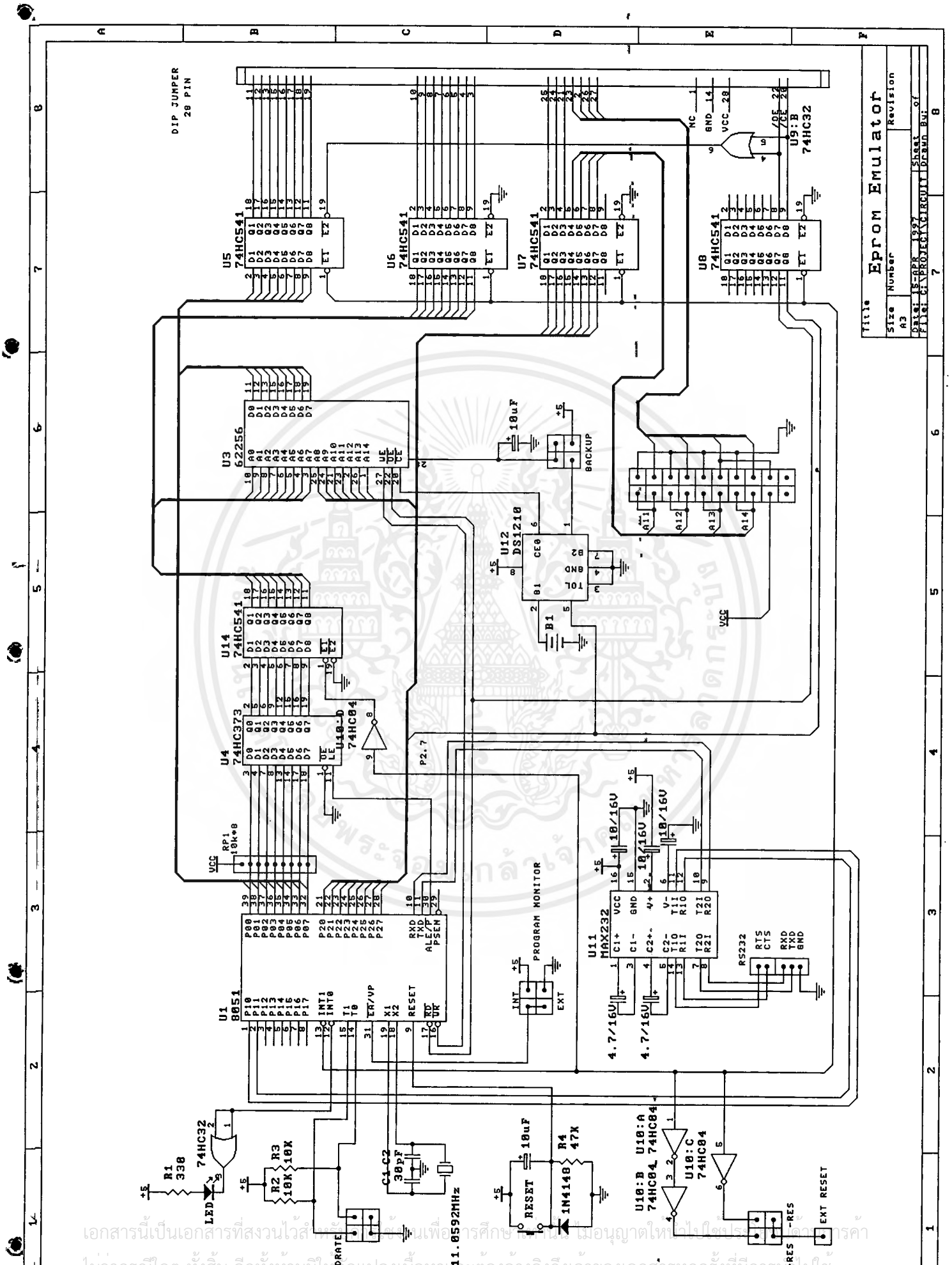
รูปที่ 3.1 Block Diagram ของ EPROM Emulator

- ส่วนที่ทำหน้าที่ติดต่อแบบไร้สาย (Wireless Link) ซึ่งจะใช้แทรกเข้าไประหว่าง พอร์ตอนุกรม RS 232 ของเครื่องคอมพิวเตอร์กับ พอร์ตอนุกรม RS 232 ของ วงจร EPROM Emulator

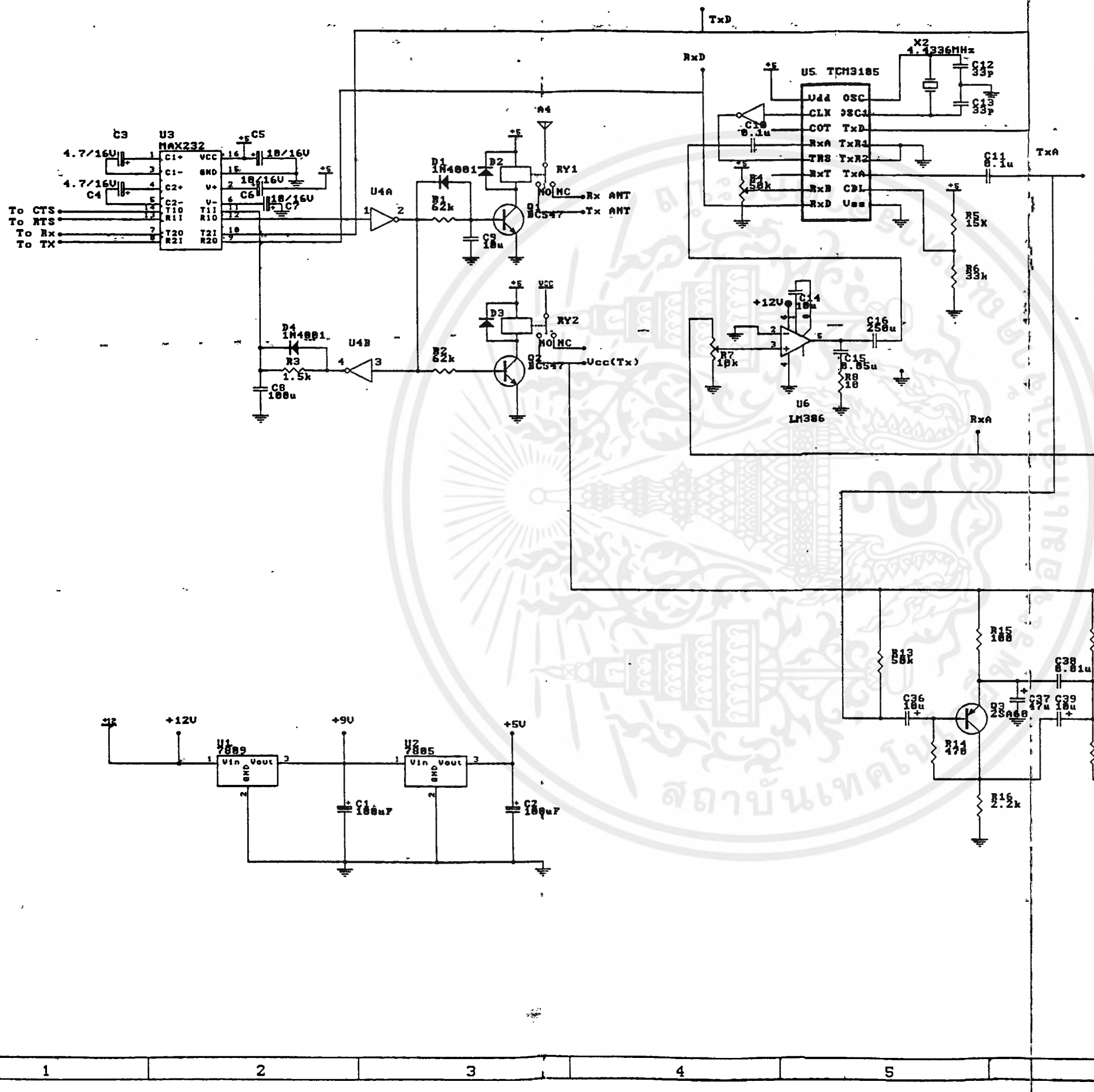


รูปที่ 3.2 Block Diagram ของส่วนเชื่อมต่อแบบไร้สาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


C3
U3
MAX232
C5
4.7/16V
18/16V
C4
C6
C7
To CTS
To RTS
To Rx
To TX
U4A
D1
1N4001
D2
RY1
NO NC
Rx ANT
Tx ANT
R1
62k
C9
10u
Q1
BC547
A4
TxD
RxD
US TCM3105
X2
4.1336MHz
C12
33p
C13
33p
C10
0.1u
R4
50k
C11
0.1u
TxA
R5
15K
R6
33k
+12V
C14
10u
C16
250u
C15
0.05u
R7
10k
R8
10
U6
LM386
RxA
D4
1N4001
U4B
R3
1.5k
C8
100u
R2
62k
D3
RY2
Q2
BC547
Vcc(Tx)
A2
L1
C21
220p
C20
0.047uF
C17
220p
C18
330p
C19
0.1u
C22
100p
C23
330p
U7
TDA7000
C24
0.002u
C25
0.02u
C26
0.01u
L2
C27
10p
C28
C29
0.003300p
R9
47k
C31
0.002u
C32
0.1u
R10
470
C38
100p
R11
56k
D5
C33
0.001u
C34
100u
C35
0.001u
R12
+12V
A3
R13
50k
R15
100
C36
10u
C37
47u
C38
0.01u
C39
10u
Q3
2SA68
R14
470
R16
2.2k
R17
1k
R18
4.7k
R19
1k
Q4
2SA68
C40
C45
0.01u
R22
220
C48
7pF
L4
C49
Q6
2SC945
C41
7p
C42
7p
L3
C43
C44
3p
R24
560
C50
0.01u
R20
3.3k
Q5
2SC945
R21
3k
C46
7p
R23
1k
C47
7p
+12
+12V
+9V
+5V
U1
7809
U2
7805
Vin
Vout
GND
C1
100uF
C2
100uF
Title
Size
A2
Number
Revision
Date: 18-APR-1997
Sheet of
File: FULL
Drawn By:

3.1.1 ส่วนของ EPROM Emulator

วงจรของ EPROM Emulator นั้นจะมีลักษณะดังแสดงในรูปที่ 3.3 ซึ่งในการทำงานนั้น จะใช้ซีพียู ซึ่งเป็นตัวประมวลผลหลัก เบอร์ 87C51 ซึ่งเป็นซีพียูที่มี EPROM สำหรับเก็บโปรแกรมการทำงาน (Monitor Program) อยู่ในตัวเองทำให้ในการออกแบบวงจรที่ใช้งานจริง สามารถตัดส่วนที่เป็นส่วนเก็บโปรแกรมที่ต่ออยู่ภายนอกออกได้ ทำให้ประหยัดพื้นที่บน Board EPROM Emulator ได้อีกด้วย

จากวงจรจะใช้ U1 เป็นส่วนควบคุมการทำงานทั้งหมดของวงจร ซึ่งในการติดต่อกับเครื่องคอมพิวเตอร์นั้น จะใช้ RS 232 Buffer เบอร์ MAX 232 เพื่อใช้ในการเปลี่ยนระดับแรงดันของ Logic จาก + 5 V , 0 V เป็น +15V,-15V ซึ่งในการติดต่อผ่าน Port RS 232 นี้ ตัว ซีพียู จะต้องใช้ Crystal 11.0592 MHz เพื่อใช้ในการควบคุมความเร็วในการรับส่งข้อมูลระหว่าง EPROM Emulator และคอมพิวเตอร์ โดยระดับความเร็วในการรับส่งข้อมูล จะขึ้นอยู่กับว่าโปรแกรมจะใส่ค่าลงใน Counter Register ให้เป็นค่าเท่าไร จากวงจรจะเห็นว่ามี Jumper 2 ตัว ต่ออยู่กับขา T0 และ T1 โดย Jumper ทั้ง 2 ตัวนี้จะใช้เป็นอินพุทของซีพียู เพื่อระบุว่าจะใช้ความเร็วเท่าไรในการรับส่งข้อมูล

ขา Int 0 จะใช้เป็นเอาท์พุทให้ LED เพื่อแสดงสถานะในการทำงานต่างๆ ส่วน Int 1 จะใช้เป็นตัวควบคุมว่าจะให้ Buffer ที่ต่ออยู่กับอุปกรณ์ภายนอกผ่าน Dip Jumper 28 Pins ยอมให้มีการรับส่งข้อมูลผ่านไปได้หรือไม่ และจะใช้เป็นเอาท์พุทส่งออกไปรีเซ็ตอุปกรณ์ภายนอกที่ต่ออยู่ในขณะที่มีการโหลดข้อมูล โดยจะมี U10:A, U10:B และ U10:C เป็นตัวเลือกว่าจะใช้การรีเซ็ตเป็นแบบ Active Low หรือ Active High

ส่วนในการรีเซ็ตของตัวซีพียูจะใช้การรีเซ็ตแบบ Power on Reset โดยในขณะเริ่มต้นแรงดันตกคร่อม C จะน้อยทำให้ขา 9 (Reset) ของซีพียูเป็น High และเมื่อเวลาผ่านไประยะหนึ่งแรงดันตกคร่อม C จะสูงขึ้นจนถึงระดับ Vcc มีผลทำให้ขา 9 มีสถานะเป็น Low (สถานะการทำงานปกติ)

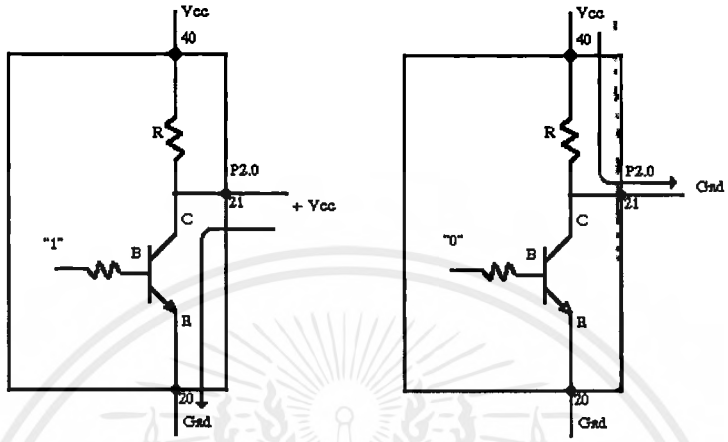
ส่วนการติดต่อกับหน่วยความจำจะใช้ Port P1 เป็น Address Byte Low และ Port P2 เป็น Address Byte High แต่จะใช้ P2.7 เป็น Chip Enable ของหน่วยความจำผ่าน U12 เพื่อระบุให้หน่วยความจำใช้งานอยู่ที่ตำแหน่ง 8000H-FFFFH เท่านั้นซึ่ง U12 จะใช้ทำหน้าที่ในการควบคุมการ Backup ข้อมูลของ U3 โดยจะทำการตรวจสอบระดับแรงดันของวงจร ว่าลดลงต่ำกว่า 90% หรือไม่ หากต่ำกว่า 90 % ก็จะเปลี่ยนมาใช้แหล่งจ่ายไฟจาก B1 ไปเลี้ยงเฉพาะส่วนหน่วยความจำ U3 แทน

และในขณะที่ Download ข้อมูลจากคอมพิวเตอร์ ขา Int 1 ของซีพียูจะต้องมีสถานะเป็น “1” เพื่อไม่ให้ U5-U8 ซึ่งเป็นบัฟเฟอร์ระหว่างอุปกรณ์ภายนอกกับวงจรทำงาน เพื่อป้องกันการอ่านข้อมูลที่ผิดพลาดและการชนกันของสัญญาณ ในขณะเดียวกันก็จะใช้สัญญาณนี้ส่งไปเป็นสัญญาณรีเซ็ตอุปกรณ์ภายนอกด้วย เพื่อหลีกเลี่ยงการที่อุปกรณ์ภายนอกจะพยายามอ่านข้อมูลจาก EPROM Emulator

ซึ่งหากพิจารณาจากข้อมูลของ ซีพียู MCS 51 แล้ว จะพบว่าพอร์ต P1,P2 และ P3 เป็นพอร์ตในลักษณะ Open Collector With Pull up Resistance คือจะมีตัวต้านทานต่ออยู่ระหว่างขา Collector ของ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Transistor และ Vcc ดังแสดงในรูปที่ 3.4 ส่วน Port P0 จะเป็นแบบ Open Collector แต่ไม่มี Pull up Resistance เราจะต้องใส่เองดังรูปวงจร จะมี Rp1 ต่ออยู่กับ Port P0

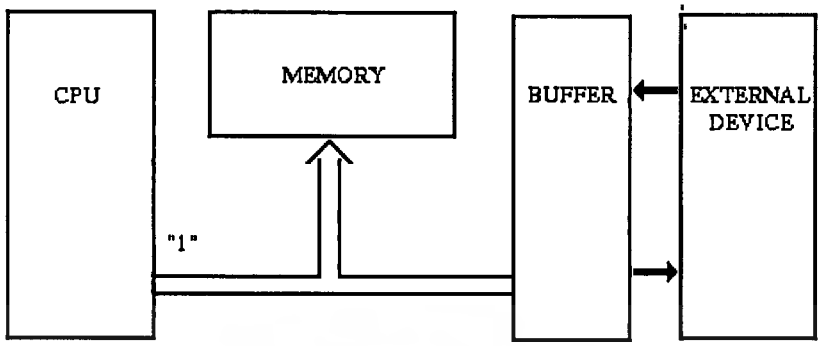


รูปที่ 3.4 พอร์ต P1,P2 และ P3

จากรูปที่ 3.4 จะเห็นได้ว่าหากต้องการให้ Output เป็น "0" จะต้องส่งสัญญาณ "1" ให้ขาเบสของทรานซิสเตอร์ เพื่อให้ทรานซิสเตอร์นำกระแส ซึ่งในขณะนี้หากมีลอจิก "0" ป้อนย้อนกลับเข้ามา ก็จะไม่เกิดปัญหาแต่อย่างใด แต่หากเป็น "1" ซึ่งเป็นไฟ +5V สำหรับ IC TTL และ อาจสูงถึง +30V หากเป็น IC CMOS ซึ่งกระแสไฟดังกล่าว จะไหลผ่านทรานซิสเตอร์จาก ขา Collector ไปยัง ขา Emitter แล้วลงกราวนด์ ซึ่งจะทำให้เกิดความเสียหายต่อพอร์ตของซีพียูได้

แต่หากเราต้องการให้เอาท์พุทเป็น "1" ก็จะต้องให้ขาเบสของทรานซิสเตอร์เป็น "0" เพื่อให้มีกระแสไหลจาก Vcc ผ่านตัวต้านทานออกไปที่เอาท์พุทได้ และในขณะนี้หากให้มีการป้อนย้อนกลับเข้ามาทางเอาท์พุทเป็นลอจิก "1" ก็จะไม่เกิดผลอะไรขึ้น แต่หากให้มีลอจิก "0" ป้อนกลับจากเอาท์พุทก็จะทำให้มีกระแสไหลผ่านตัวต้านทานออกไปที่เอาท์พุทมากขึ้น แต่ก็จะไม่ทำให้เกิดความเสียหายใดๆแก่วงจรเลย และจะใช้คุณสมบัติในข้อนี้ ในการทำงานของวงจร ซึ่งเมื่อพิจารณาจากวงจรจะเห็นว่า การที่ซีพียูจะทำงานกับหน่วยความจำ และการที่หน่วยความจำจะถูก Excute จากอุปกรณ์ภายนอก จะใช้บัสเดียวกัน ซึ่งตัวบัฟเฟอร์จะมีสัญญาณจาก Int 1 คอยควบคุมอยู่แล้ว แต่ตัวซีพียูจะต้องถูกควบคุมจากโปรแกรมดังรูปที่ 3.5 และเพื่อหลีกเลี่ยงไม่ให้เกิดความเสียหายขึ้น จึงต้องสั่งให้เอาท์พุทของซีพียูที่ติดต่อกับหน่วยความจำทั้งหมดเป็นลอจิก "1"ส่วน U9:B นั้นจะเป็นส่วนช่วยในการตรวจสอบว่ามีความต้องการในการอ่านข้อมูลจากหน่วยความจำหรือไม่ หากมีก็จะสั่งให้ U5 ซึ่งเป็น Buffer ของบัสข้อมูลทำงานได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



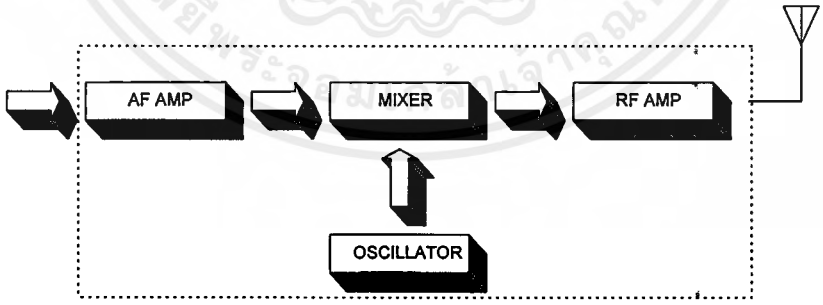
รูปที่ 3.5 การเข้าใช้หน่วยความจำ

แต่ในขณะทำการทดลองระหว่างการสร้าง เราจะมีการ ต่อ U2,U4,U13,U14 และ U15 ซึ่งเป็นส่วนของ Buffer และ EPROM สำหรับเก็บโปรแกรมในการทำงานของบอร์ด EPROM Emulator ซึ่ง U2 จะเป็น EPROM ที่ใช้ในการเก็บโปรแกรมที่ใช้ในการทำงาน ส่วน U4 จะเป็น Latch ซึ่งใช้กับกรณีที่ใช้พอร์ต 0 เป็นทั้ง Data Bus และ Address Byte Low และ U13,U14 และ U15 จะเป็นบัฟเฟอร์สำหรับป้องกันการชนกันของสัญญาณ ระหว่างการนำโปรแกรมเข้ามาทำงานภายในซีพียู และ การนำข้อมูลไปใส่ในหน่วยความจำซึ่งบัฟเฟอร์นี้จะต่ออยู่กับทั้ง Address Bus และ Data Bus

3.1.2 ส่วนประกอบของเครื่องส่ง

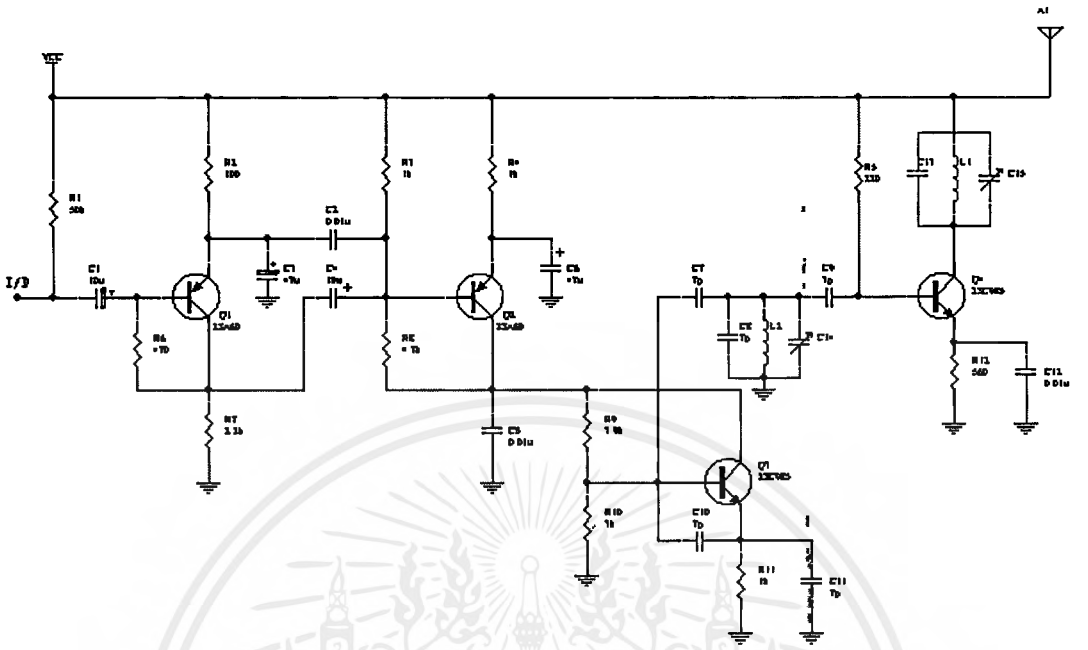
ในโครงงานนี้ได้เลือกใช้ภาคส่งชนิดเอฟเอ็มที่ใช้ทรานซิสเตอร์

<u>หลักการทำงาน</u>



รูปที่ 3.6 Block diagram ของภาคส่งแบบเอฟเอ็ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



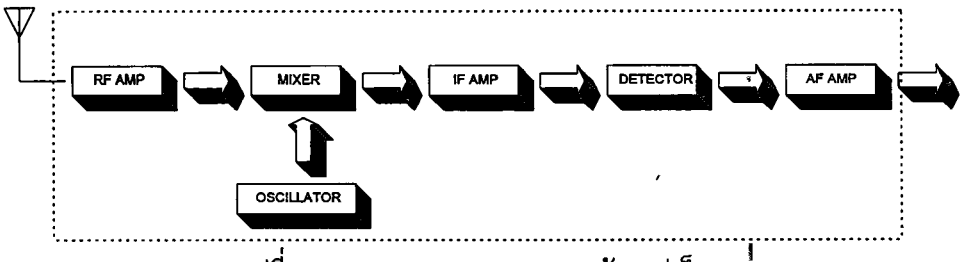
รูปที่ 3.7 วงจรภาคส่งแบบเอฟเอ็ม

จากวงจรในรูปที่ 3.2.2 Q1 จะรับสัญญาณจากโมเด็ม (Modem) และจะขยายสัญญาณส่งต่อให้กับ Q2 ทำหน้าที่ขยายสัญญาณอีกครั้งหนึ่ง แล้วส่งสัญญาณนี้ไปทำการมอดูเลทแบบเอฟเอ็มกับความถี่ที่ผลิตโดย C8, C14 และ L2 โดยใช้ Q3 ทำหน้าที่ในการมอดูเลทสัญญาณทั้งสอง เมื่อทำการมอดูเลทเรียบร้อยแล้ว ก็จะใช้ Q4 ขยายสัญญาณอีกครั้งหนึ่ง ก่อนที่จะทำการกรองความถี่อีกครั้งโดย C13, C15 และ L1 เพื่อส่งออกทางเสาอากาศ

3.1.3 ส่วนประกอบของเครื่องรับ

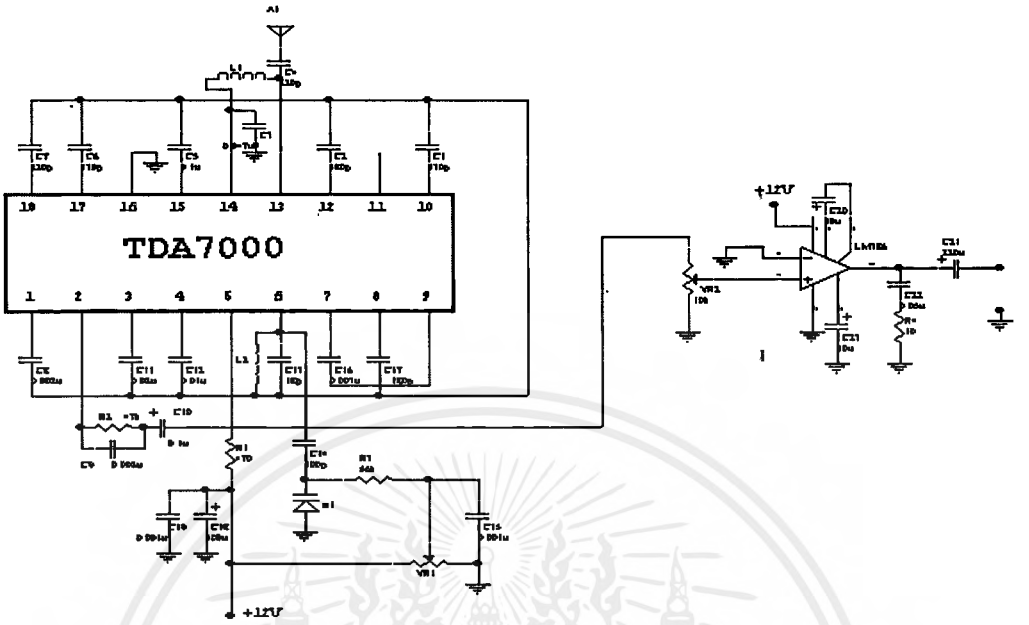
ได้เลือกใช้ระบบการรับส่งไร้สายแบบเอฟเอ็ม จึงเลือกใช้ไอซีภาครับเอฟเอ็มเบอร์ TDA7000 ซึ่งไอซีภาครับเอฟเอ็มเบอร์นี้ได้รวมเอาภาคการทำงานต่างๆ ของวิทยุเอฟเอ็มไว้อย่างครบครัน ตั้งแต่ภาคมิกเซอร์, ภาคไอเอฟ, ภาคกำเนิดความถี่ และ ภาคดีเทคเตอร์ จึงทำให้วงจรที่ใช้มีขนาดเล็กพอสมควร

<u>หลักการทำงาน</u>



รูปที่ 3.8 Block diagram ของภาครับเอฟเอ็ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า



รูปที่ 3.9 วงจรภาครับแบบเอฟเอ็ม

สัญญาณที่รับได้จากเสาอากาศจะถูกกรองโดย L1, C3 และ C4 เพื่อให้ได้สัญญาณที่ต้องการ ก่อนที่จะป้อนสัญญาณนี้ให้กับภาคมิกเซอร์ทางขา 13 และ 14 ของไอซี TDA7000 ที่ภาคมิกเซอร์นี้จะทำการผสมความถี่ที่ผลิตได้จากภาคผลิตความถี่ที่ขา 5 และ 6 ของ TDA7000 ซึ่งภาคผลิตความถี่ที่ใช้เป็นแบบ VCO (Voltage Control Oscillator) ซึ่งจะควบคุมแรงดันในการกำหนดความถี่โดย VR1 สัญญาณที่ได้จากภาคมิกเซอร์ จะถูกส่งต่อให้กับภาคไอเอฟ และส่งต่อไปยังภาคดีเทคเตอร์เพื่อให้ได้สัญญาณออกที่ขา 2 ของ TDA7000 หลังจากที่ได้สัญญาณเสียงจากขา 2 แล้วจะต้องทำการขยายอีกครั้งโดยไอซี LM386 ซึ่งเป็นไอซีที่ทำหน้าที่ในการขยายเสียงโดยเฉพาะ ซึ่งสัญญาณที่ได้จาก LM386 จะต้องถูกคัปปลิ้งโดย C21 ก่อนที่จะส่งต่อไปให้กับส่วนโมเด็มต่อไป

3.1.4 ส่วนประกอบของวงจร Duplexer

เนื่องจากว่าส่วนที่ทำการเชื่อมต่อแบบไร้สายจะต้องใช้ความถี่ในการออกอากาศเพียงความถี่เดียวเพื่อที่จะเป็นการสงวนไว้ไม่เกิดการใช้ช่องความถี่อย่างฟุ่มเฟือย เราจึงต้องใช้วงจร Duplexer เพื่อทำหน้าที่ในการเลือกว่า ณ เวลาใดๆ จะทำการส่งหรือไม่ และจะต่อเสาอากาศในการรับส่งเข้ากับภาคส่งหรือภาครับ

<u>หลักการทำงาน</u>

เนื่องจากในการส่งข้อมูลแบบ Half Duplex เป็นการติดต่อแบบผลัดกันรับส่งคนละจังหวะกัน กล่าวคือมีการผลัดกันส่งและผลัดกันรับโดยใช้เสาอากาศเพียงต้นเดียว วงจร Duplexer เป็นตัวเลือกการทำงานของวงจรว่าต้องการส่งข้อมูลหรือจะรับข้อมูล ซึ่งสามารถที่จะควบคุมการทำงานโดยใช้ซอฟท์แวร์ สัญญาณที่ใช้ควบคุมนี้จะออกมาทางพอร์ต RS232 ที่ขา RTS และ CTS โดยมีการทำงานอย่างละเอียดดังนี้คือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เมื่อเข้าโปรแกรมเพื่อที่จะทำการรับและส่งข้อมูล โปรแกรมจะต้องทำการสั่งให้พอร์ต RS232 ส่งสัญญาณ RTS ออกมา ซึ่งจะมีสถานะเปลี่ยนจาก -15V เป็น +15V จากนั้นจะนำสัญญาณ RTS นี้ไปผ่านวงจร RS232 Buffer ซึ่งจะได้สัญญาณที่ออกจากวงจร RS 232 Buffer เปลี่ยนระดับจาก "1" ไปเป็น "0" และจะนำสัญญาณนี้มาผ่าน Inverter ซึ่งใช้ไอซีเบอร์ 74HC04 เพื่อให้ได้สัญญาณที่เปลี่ยนจาก "0" ไปเป็น "1" ซึ่งจะแบ่งวงจรหลังจากนี้ออกเป็น 3 ส่วนย่อยๆ คือ

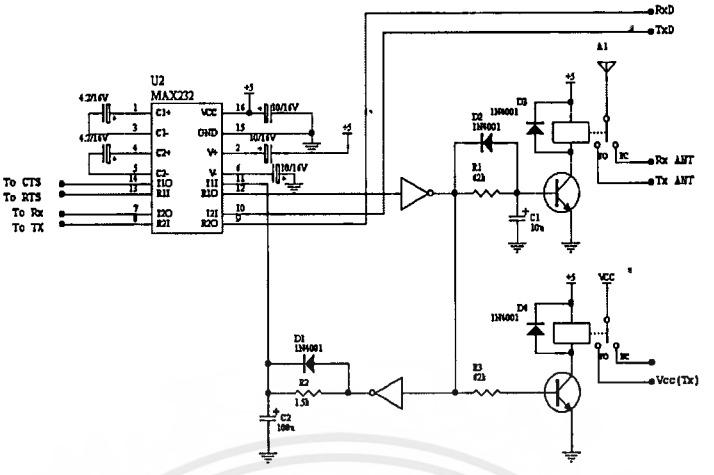
ส่วนที่ 1 ควบคุมการตัดต่อเสาอากาศเข้ากับภาครับหรือภาคส่ง โดยใช้ Coaxial Relay ก่อนเพื่อให้วงจรภาคส่งมีเสาต่อเป็น Load ก่อนที่จะทำการจ่ายไฟเลี้ยงให้กับภาคส่งเพื่อส่งสัญญาณออกอากาศ แต่เนื่องจาก Coaxial Relay มีราคาสูงอาจจะใช้ Relay ธรรมดาก็ได้ แต่จะทำให้ต้องสูญเสียกำลังส่งไปบ้าง หรืออาจจะใช้เสาอากาศ 2 ต้นสำหรับภาครับและภาคส่งโดยไม่ต้องผ่านวงจร Duplexer เลย

ส่วนที่ 2 ควบคุมการจ่ายไฟให้กับภาคส่ง โดยใช้ Relay และใช้ขั้ว NO (Normal Open) สำหรับการจ่ายไฟให้กับภาคส่งหลังจากที่ได้เชื่อมต่อเสาอากาศเข้ากับภาคส่งเรียบร้อยแล้ว โดยจะใช้ R, C เป็นตัวหน่วงเวลาในการไบอัสให้กับขาเบสของทรานซิสเตอร์เมื่อมีสัญญาณเป็น "1" เข้ามา แต่ถ้ามีสัญญาณเป็น "0" เข้ามาก็จะใช้ไดโอดทำการคายประจุออกจากตัว C ทันที

ส่วนที่ 3 ทำการควบคุมเพื่อให้ได้สัญญาณ CTS เมื่อมีการปฏิบัติตามส่วนที่ 1 และ 2 เรียบร้อยแล้ว โดยจะใช้วงจรหน่วงเวลาซึ่งประกอบด้วย R, C และ Diode ซึ่งจะต้องนำสัญญาณตัวเดียวกับที่จ่ายให้ส่วนที่ 1 และ 2 มาผ่าน Inverter ก่อน จะทำให้ในขณะที่ไม่มีสัญญาณ RTS ตัว Inverter จะให้สัญญาณออกเป็น "1" เพื่อป้อนกลับให้วงจร RS232 Buffer แปลงกลับเป็นแรงดันขนาด -10V เพื่อส่งให้ขา CTS ของ พอร์ต RS232 และในขณะเดียวกันก็จะทำการประจุกระแสให้กับ C อย่างทันทีทันใด แต่เมื่อมีสัญญาณ RTS เข้ามาจะทำให้ได้สัญญาณ "0" จากตัว Inverter ดังกล่าว จะทำให้เกิดการค่อยๆ คายประจุผ่าน R ด้วยเวลาค่าหนึ่งซึ่งมากกว่าเวลาที่ใช้ในส่วนที่ 2

ผลการทำงานก็คือ เมื่อมีสัญญาณ RTS ก็จะทำให้วงจรส่วนที่ 1 ทำงาน ตามด้วยส่วนที่ 2 และ ส่วนที่ 3 ตามลำดับ ซึ่งในจังหวะที่ 3 นี้วงจร Duplexer นี้จะส่งสัญญาณ CTS กลับไปให้กับพอร์ต RS232 เพื่อเป็น Hand Checking แสดงความพร้อมของ Hardware เมื่อ Program ตรวจพบ สัญญาณ CTS ก็จะทำการส่งข้อมูลออกไปทางขา TX ของพอร์ต RS232 และเมื่อส่งข้อมูลเรียบร้อยแล้ว โปรแกรมก็จะต้องสั่งให้เลิกส่งสัญญาณ RTS เมื่อไม่มีสัญญาณ RTS แล้ว วงจร Duplexer ก็จะทำการยกเลิก สัญญาณ CTS ตัดการจ่ายไฟให้กับภาคส่งและต่อเสาอากาศเข้ากับภาครับเพื่อพร้อมที่จะรับข้อมูลตลอดเวลา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

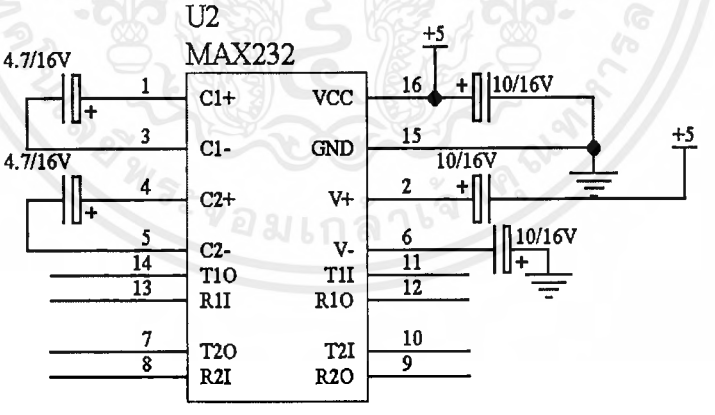


รูปที่ 3.10 วงจร Duplexer

3.1.5 ส่วนวงจร RS 232 Buffer

ในการเชื่อมต่อวงจรที่ทำงานกับระดับแรงดันแบบทีทีแอล (TTL) เข้ากับพอร์ต RS232 ของเครื่องคอมพิวเตอร์ ซึ่งมีระดับแรงดัน -15V ถึง +15V นั้น จะต้องมีวงจรพิเศษเพื่อทำการแปลงระดับแรงดันให้เหมาะสมซึ่งในที่นี้เราได้เลือกใช้ IC เบอร์ MAX 232 ซึ่งเป็น IC ที่ใช้อุปกรณ์ประกอบจากภายนอกน้อย คือ ใช้ C ชนิดอีเล็คโทรไลท์เพียง 5 ตัวเท่านั้น

<u>หลักการทำงาน</u>



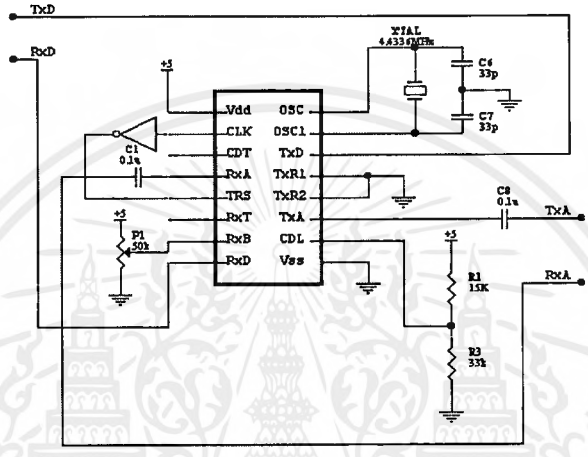
รูปที่ 3.11 วงจร RS 232 Buffer

จากวงจรจะใช้ C ที่ต่อระหว่างขา 1 กับ 3 , ระหว่างขา 4 กับ 5 , ขา ,2 และ ขา 6 เป็นตัวกำหนดระดับแรงดันที่จะใช้ในการเชื่อมต่อ โดยขา R1I และ R2I จะเป็นขาที่รับระดับแรงดัน -15V ถึง +15Vและแปลงออกเป็นแรงดัน 0V , +5V ออกที่ขา R1O และ R2O ส่วนขา T1I และ T2I ก็จะรับแรงดันที่เป็น 0V , +5V แปลงเป็นระดับแรงดัน -10V ถึง +10V ออกที่ขา T1O และ T2O

3.1.6 ส่วนวงจร MODEM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในการรับส่งข้อมูลผ่านระบบไร้สายนี้จำเป็นต้องมีการแปลงสัญญาณ "0" และ "1" ให้เป็นสัญญาณชนิดอื่นเพื่อใช้แทนสัญญาณ "0" กับ "1" ก่อน เช่น FSK , PSK , PAM , QAM แล้วจึงนำสัญญาณเหล่านี้ไปป้อนให้กับวงจรภาคส่งแบบ FM ในโครงงานนี้ได้เลือกใช้ IC เบอร์ TCM 3105 ซึ่งเป็น IC ที่ทำหน้าที่เป็น MODEM แบบ FSK ขนาดความเร็วไม่เกิน 1,200 บิทต่อวินาที ตามมาตรฐาน CCITT V.23 หรือ Bell 202 ซึ่งเราสามารถกำหนดความเร็ว และมาตรฐานได้จากการป้อนสัญญาณให้กับขา TXR1 ,TXR2 และ TRS ซึ่งการกำหนดความเร็วและมาตรฐานนี้ดูได้จากตารางในภาคผนวกส่วนของ IC TCM3105 ได้



รูปที่ 3.12 วงจร MODEM

<u>หลักการทำงาน</u>

จากวงจรกำหนดความเร็วเป็น 1,200 บิทต่อวินาที และใช้ มาตรฐาน BELL 202 ในการรับส่ง โดยการต่อขา TXR1 และ TXR2 ลงกราวด์ และต่อขา TRS ผ่าน Inverter จากขา CLK มา และใช้ X'TAL 4.4336 Mhz ในการกำเนิดความถี่ให้กับตัว TCM3105 และที่ขา TXA และ RXA มี C 0.1 uF คัปปลิ้ง เพื่อป้องกันแรงดันกระแสตรงผ่านเข้าไป ซึ่งจะใช้ขา RXA รับสัญญาณที่เป็นอะนาล็อก หากเป็นความถี่ 1,200 Hz ก็จะให้เอาท์พุทที่ขา RXD เป็น "1" และหากเป็นความถี่ 2,200 Hz ก็จะเป็น "0" และในทำนองเดียวกัน หากขา TXD เป็น "1" ก็จะให้เอาท์พุทเป็นความถี่ 1,200 Hz และหากเป็น "0" ก็จะให้เอาท์พุทเป็นความถี่ 2,200 Hz และจะใช้ตัวต้านทานปรับค่าได้ 50 KΩ ต่อที่ขา RXB เพื่อปรับไบอัสในการรับข้อมูลของ TCM3105

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 การออกแบบในส่วนของซอฟท์แวร์

3.2.1 การตรวจสอบข้อผิดพลาดในการส่งข้อมูล

ในการตรวจสอบความผิดพลาดของการส่งข้อมูล จะมีการตรวจสอบความผิดพลาดของการส่งข้อมูลดังนี้

3.2.1.1 ตรวจสอบพาริตี้

เป็นการตรวจสอบพาริตี้ของพอร์ตอนุกรม ซึ่งค่าพาริตี้นี้จะมีการกำหนดในขั้นตอนการ Initial พอร์ตการสื่อสาร ซึ่งหากค่าพาริตี้ผิดพลาด แล้วก็จะไม่มีการรับข้อมูล แต่ก็ไม่มีการแสดงผลใดๆ ออกมาว่าเกิดความผิดพลาดเกิดขึ้น

3.2.1.2 ตรวจสอบค่า Check SUM

ค่า Check SUM ของ Intel Hex Format File นี้คือการนำเอาข้อมูลตั้งแต่ Byte Counter เรื่อยมาจนถึงค่าข้อมูลไบท์สุดท้ายของบรรทัดมาบวกเข้าด้วยกัน จากนั้นจึงนำค่าที่ได้ไปทำ 2' Complement ก็จะได้ค่า Check SUM ออกมา ซึ่งบอร์ด EPROM Emulator ก็จะทำการบวกข้อมูล และทำ 2' Complement ออกมา แล้วนำมาเปรียบเทียบกับค่า Check SUM ที่รับได้จากพีซี ถ้าหากข้อมูลมีการผิดพลาดเกิดขึ้นแม้เพียงบิทเดียว ก็จะทำให้ค่า Check SUM ทั้งสองมีค่าไม่เท่ากัน แสดงว่าเกิดความผิดพลาดขึ้นในขั้นตอนการสื่อสาร ดังนั้นไฟ LED ที่บอร์ด EPROM Emulator ก็จะกระพริบถี่ๆ เพื่อบอกว่ามีความผิดพลาดในการส่งข้อมูลเกิดขึ้น

3.2.1.3 ตรวจสอบรูปแบบของไฟล์

โปรแกรมบนบอร์ด EPROM Emulator จะมีการตรวจสอบข้อมูลที่รับเข้ามาได้ โดยจะตรวจสอบข้อมูลตัวแรกว่าเป็นโคลอน ":" หรือไม่ ซึ่งถ้าข้อมูลที่ส่งไปไม่อยู่ในรูปแบบของ Intel Hex Format File ก็จะไม่มีการรับข้อมูล ส่วนในบอร์ดที่ได้ทำการพัฒนาขึ้นมานี้ ได้มีการเพิ่มเติมส่วนของการตรวจสอบความผิดพลาดโดยที่บอร์ด EPROM Emulator จะมีการส่งค่ารายงานผลการตรวจสอบค่า Check SUM กลับมาให้กับ PC เป็นจำนวน 1 ไบท์ ซึ่งหากพบว่ามีการผิดพลาดของข้อมูลเกิดขึ้นก็จะมีการส่งข้อมูลชุดนั้นกลับไปใหม่ ซึ่งจะทำแบบนี้จนครบ 3 ครั้ง จึงจะแจ้งผลของความผิดพลาดบนหน้าจอ ซึ่งการพยายามส่งข้อมูลไปใหม่ 3 ครั้งนี้เป็นการแก้ไขข้อผิดพลาด (Error Correction) โดยการส่งข้อมูลชุดเดิมทั้งชุด ส่วนของข้อมูลที่ใช้สื่อสารกันระหว่าง PC และบอร์ด EPROM Emulator นี้จะใช้ในส่วนของ Type ของ Intel Hex Format File ซึ่งแต่เดิมมีเพียง "00" และ "01" จึงได้เพิ่ม Type เข้าไปดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตารางที่ 3.1 ค่า TYPE สำหรับการรับส่งข้อมูล

Type	หน้าที่
00	Download
01	End of File
02	Download and Return Error Check
03	Upload and Return Error Check
04	No Error
05	Error

- *Type 02* เป็นการสั่งให้ทำการ Download ข้อมูลลงไปที่บอร์ด EPROM Emulator และให้บอร์ดส่ง Type 04 หรือ 05 กลับมาให้พีซี เพื่อใช้ในการตรวจสอบความผิดพลาดของข้อมูล
- *Type 03* เป็นการสั่งให้ทำการ Upload ข้อมูลจากบอร์ดมายังพีซี และให้ส่งค่าความผิดพลาดของข้อมูลกลับมาด้วย
- *Type 04* เป็น Type ที่บอร์ดส่งมาบอกพีซีว่า "ข้อมูลถูกต้อง"
- *Type 05* เป็น Type ที่บอร์ดส่งมาบอกพีซีว่า "ข้อมูลผิดพลาด"

3.2.2 ส่วนโปรแกรมควบคุมบอร์ดของเครื่องจำลองการทำงานของ EPROM

ส่วนโปรแกรมควบคุมเครื่องจำลองการทำงานของ EPROM เป็นการโปรแกรมโดยใช้ภาษาแอสแซมบลีของไมโครคอนโทรลเลอร์ตระกูล 8051 เนื่องจากส่วนของอุปกรณ์ที่ใช้ควบคุมการทำงานทั้งหมดเป็นไมโครคอนโทรลเลอร์ในตระกูล 8051 ส่วนของโปรแกรมจะอยู่ภายในตัวไมโครคอนโทรลเลอร์ ซึ่งจะมีที่เก็บโปรแกรมขนาด 4 กิโลไบท์อยู่ (สำหรับไมโครคอนโทรลเลอร์หมายเลข 8751 หรือ 8951)

ขั้นตอนในการพัฒนาโปรแกรมแบ่งได้ดังนี้

3.2.2.1 ศึกษาการทำงานของเครื่องจำลองการทำงานของ EPROM ว่ามีอะไรบ้าง ซึ่งจากการศึกษาสามารถแบ่งออกตามรูปแบบของงานได้ดังนี้

- การรับข้อมูลที่ส่งมาจากเครื่องคอมพิวเตอร์
- การส่งข้อมูลไปยังเครื่องคอมพิวเตอร์
- การรับข้อมูลและส่งผลของการตรวจสอบความผิดพลาดกลับไปยังเครื่องคอมพิวเตอร์

3.2.2.2 เขียนแผนภาพแสดงลำดับการทำงานของโปรแกรม ซึ่งจากการออกแบบและแก้ไขแล้ว จึงได้แผนภาพการทำงานของโปรแกรมดังรูป 3.18

3.2.2.3 เขียนคำสั่งที่ใช้สั่งงานจากแผนภาพที่ได้ออกแบบไว้ ด้วยภาษาแอสแซมบลี 8051 เครื่องมือที่ใช้ในการเขียนโปรแกรมสามารถใช้โปรแกรมที่เป็นโปรแกรมสำหรับพิมพ์และแก้ไขข้อความโปรแกรมใดก็ได้ เช่น โปรแกรม EDIT ที่มากับระบบปฏิบัติการดอส, โปรแกรม QEDIT หรือโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ประมวลผลคำที่มีอยู่บนระบบปฏิบัติการวินโดวส์ สำหรับโครงงานนี้เลือกใช้โปรแกรม QEDIT เนื่องจากเป็นโปรแกรมขนาดเล็กที่มีประสิทธิภาพสูง และใช้งานได้ง่าย (ภาคผนวก ค)

3.2.2.4 ทำการแอสแซมเบลอร์โปรแกรม และทดสอบการทำงานกับเครื่องจำลองการทำงานของ EPROM ว่าสามารถทำงานได้ถูกต้องตามต้องการหรือไม่ สำหรับโปรแกรมที่ใช้ในการแอสแซมเบลอร์นั้นก็มีอยู่หลายโปรแกรม เช่น CROSS32, SXA51 ในโครงงานนี้ใช้โปรแกรม SXA51 เป็นโปรแกรมสำหรับการแอสแซมเบลอร์ เนื่องจากเป็นโปรแกรมแอสแซมเบลอร์ของไมโครคอนโทรลเลอร์ตระกูล 51 โดยตรง และยังมีความสามารถต่างๆ ที่มากกว่าโปรแกรมอื่นๆ (ภาคผนวก ง)

3.2.2.5 แก้ไขโปรแกรมในส่วนที่ยังทำงานไม่ถูกต้อง จนกระทั่งโปรแกรมสามารถทำงานได้ถูกต้องตามต้องการ ซึ่งโปรแกรมที่สมบูรณ์จะอยู่ในภาคผนวก ก

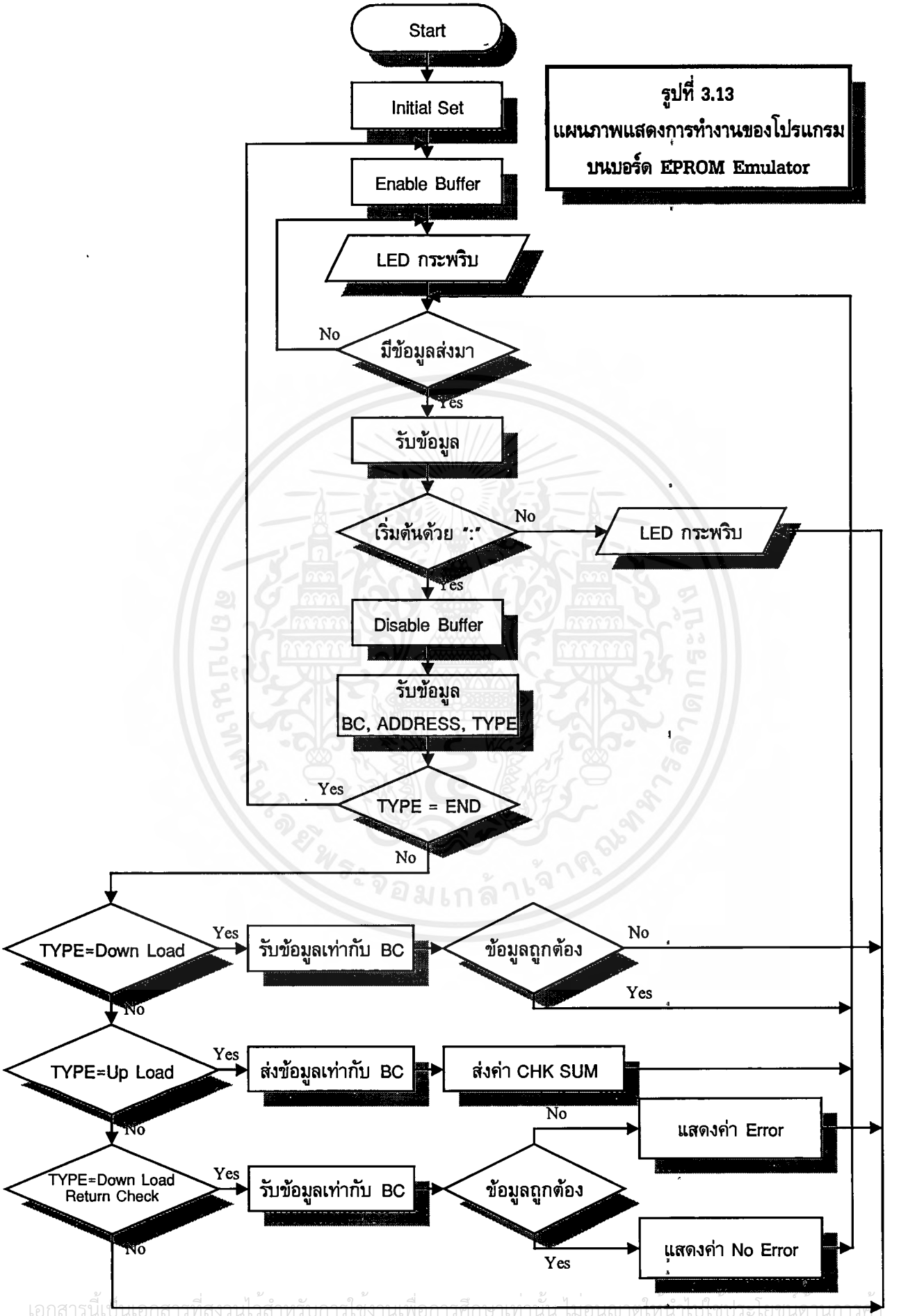
3.2.3 การทำงานของโปรแกรมส่วน MCS-51

โปรแกรมในส่วนนี้จะทำงานตามแผนภูมิโปรแกรมที่ได้ออกแบบไว้ดังรูป ,3.13 โดยเริ่มจากการหน่วงเวลาเพื่อให้เกิดการเพาเวอร์ออนรีเซ็ต (Power On Reset) จากนั้นก็จะทำการตั้งค่าของรีจิสเตอร์ต่างๆ ที่จำเป็นต้องใช้ในโปรแกรม เช่นการตั้งค่าของการอินเตอร์รัพท์ เมื่อตั้งค่าที่จำเป็นสำหรับการทำงานของโปรแกรมแล้วก็จะทำการตรวจสอบการตั้งค่าความเร็วในการสื่อสารข้อมูลกับเครื่องคอมพิวเตอร์ที่ขา INT0, INT1 ของ MCS-51 และทำการเซ็ตค่าตามที่ตรวจสอบได้ จากนั้นก็จะทำการพูลอัพ (Pull Up) ขาแอสเดรสของหน่วยความจำและอีนาเบิล (Enable) บัฟเฟอร์ที่อยู่ระหว่างซีพียูกับหน่วยความจำ ในส่วนที่กล่าวมาทั้งหมดเป็นส่วนของการตั้งค่าเริ่มต้น ซึ่งจะเกิดขึ้นทุกครั้งที่มีการรีเซ็ตบอร์ด หรือจ่ายกระแสไฟเข้าตัวบอร์ด

การทำงานในส่วนถัดมาจะเป็นการรอรับค่าที่ส่งเข้ามาทางพอร์ตสื่อสารที่ขา Rx ของ MSC-51 ว่าเป็นข้อมูลในรูปแบบของ Intel Hex Format หรือไม่ โดยตรวจสอบจากข้อมูลว่าเป็นเครื่องหมายโคลอน (:) หรือไม่ ถ้าไม่ใช่ก็จะรอรับข้อมูลต่อไป ซึ่งจะมีการแสดงผลทาง LED บนตัวบอร์ดด้วย โดยจะกระพริบเป็นจังหวะช้าๆ แต่ถ้าเป็นข้อมูลเครื่องหมายโคลอน โปรแกรมก็จะทำการรับข้อมูลในส่วนหัวของข้อมูลที่เป็นรูปแบบของ Intel Hex Format ซึ่งประกอบด้วยจำนวนข้อมูล, ตำแหน่งเริ่มต้น และชนิดของข้อมูล จากนั้นจะตรวจสอบชนิดของข้อมูลว่าเป็นการทำงานแบบใด เมื่อตรวจสอบได้แล้วโปรแกรมก็จะกระโดดไปทำงานในส่วนที่ตรงกับชนิดของข้อมูลที่ได้รับ

ถ้าชนิดของข้อมูลเป็นการรับข้อมูลมาเก็บยังตัวบอร์ด โปรแกรมก็จะทำการรับข้อมูลเป็นจำนวนเท่ากับค่าของตัวนับข้อมูลและเก็บในหน่วยความจำตามตำแหน่งที่ได้รับในตอนแรก เมื่อรับข้อมูลจนหมดก็จะมีการตรวจสอบความถูกต้องของข้อมูลด้วย โดยจะตรวจสอบจากค่าเช็คผลรวม (Check SUM) ที่ส่งมาหลังการส่งข้อมูล ซึ่งหากผลรวมที่ตรวจสอบไม่ถูกต้องก็จะมีการแสดงผลที่ LED โดยจะกระพริบเร็วกว่าปกติ แต่ถ้าการตรวจสอบข้อมูลถูกต้องก็จะกลับไปเริ่มรอรับข้อมูลใหม่อีกครั้ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.13
แผนภาพแสดงการทำงานของโปรแกรมบนบอร์ด EPROM Emulator

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ถ้าชนิดของข้อมูลเป็นการส่งข้อมูลกลับไปยังเครื่องคอมพิวเตอร์ โปรแกรมก็จะทำการส่งข้อมูลเป็นจำนวนเท่ากับค่าของตัวนับข้อมูลตามตำแหน่งในหน่วยความจำที่ได้รับในตอนแรก เมื่อส่งข้อมูลจนหมดก็จะมีส่งค่าเช็คผลรวม (Check SUM) ไปให้ครื่องคอมพิวเตอร์ จากนั้นก็จะกลับไปเริ่มรอรับข้อมูลใหม่อีกครั้ง

ถ้าชนิดของข้อมูลเป็นการรับข้อมูลมาเก็บยังตัวบอร์ด และให้มีการคืนค่าของการตรวจสอบข้อมูลกลับไปยังเครื่องคอมพิวเตอร์ โปรแกรมก็จะทำการรับข้อมูลเป็นจำนวนเท่ากับค่าของตัวนับข้อมูลและเก็บในหน่วยความจำตามตำแหน่งที่ได้รับในตอนแรก เมื่อรับข้อมูลจนหมดก็จะมีการตรวจสอบความถูกต้องของข้อมูล โดยจะตรวจสอบจากค่าเช็คผลรวม (Check SUM) ที่ส่งมาหลังการส่งข้อมูล ซึ่งหากผลรวมที่ตรวจสอบไม่ถูกต้องก็จะมีการแสดงผลที่ LED โดยจะกระพริบเร็วกว่าปกติ และจะคืนค่าแสดงความผิดพลาดไปที่เครื่องคอมพิวเตอร์ แต่ถ้าการตรวจสอบข้อมูลถูกต้อง ก็จะคืนค่าแสดงข้อมูลถูกต้องไปที่เครื่องคอมพิวเตอร์และกลับไปเริ่มรอรับข้อมูลใหม่อีกครั้ง

3.2.4 โปรแกรมส่วนการรับส่งทางพอร์ตอนุกรม

ส่วนที่สำคัญที่สุดของโปรแกรมก็คือส่วนของการรับส่งข้อมูลทางพอร์ตอนุกรม ซึ่งในส่วนนี้จะสัมพันธ์กับโครงสร้างทางสถาปัตยกรรมของไมโครคอนโทรลเลอร์ตระกูล 51 (หัวข้อ 2.4.13) การทำงานของโปรแกรมย่อยในส่วนรับส่งข้อมูลทางพอร์ตอนุกรมมีลักษณะคล้ายกันมาก โดยโปรแกรมรับข้อมูลจะเป็นการรอจนกระทั่งข้อมูลถูกรับเข้ามาหมดแล้ว ซึ่งตรวจสอบได้จากบิท RI จะถูกเซ็ตเป็น 1 จากนั้นจึงนำข้อมูลที่ได้รับซึ่งอยู่ในรีจิสเตอร์ SBUF ไปใช้งานและทำการเคลียร์บิท RI เพื่อรอรับข้อมูลต่อไป โปรแกรมส่วนการส่งข้อมูลจะเริ่มจากการรอจนกระทั่งข้อมูลก่อนหน้านี้ถูกส่งไปจนหมดเสียก่อนซึ่งก็ตรวจสอบได้จากรีจิสเตอร์ TI จะถูกเซ็ตเป็น 1 จากนั้นจึงนำข้อมูลที่ต้องการส่งไปใส่ไว้ในรีจิสเตอร์ SBUF และทำการเคลียร์บิท TI เพื่อรอการส่งต่อไป สำหรับโปรแกรมย่อยรับและส่งข้อมูลทางพอร์ตอนุกรมแสดงได้ดังนี้

- โปรแกรมย่อยรับข้อมูลทางพอร์ตอนุกรม

```
;RECEIVE DATA 1 BYTE
;O/P          A
R_BYTE: JNB      RI,$          ;WAIT FOR RECEIVE
        CLR      RI
        MOV      A,SBUF
        RET
```

- โปรแกรมย่อยส่งข้อมูลทางพอร์ตอนุกรม

```
;SEND 1 BYTE
;I/P          A
S_BYTE: JNB      TI,$          ;WAIT FOR SEND
        CLR      TI
        MOV      SBUF,A
        RET
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การรับส่งข้อมูลในโครงงานนี้เลือกใช้การรับส่งข้อมูลทางพอร์ตอนุกรมโหมด 1 (หัวข้อ 2.4.13.2) เนื่องจากสามารถควบคุมอัตราความเร็วในการรับส่งได้ง่าย โดยเพียงแค่ตั้งค่าความเร็วที่ต้องการที่รีจิสเตอร์ TH1 เท่านั้น สำหรับการตรวจสอบว่าต้องการค่าความเร็วในการรับส่งข้อมูลเป็นเท่าใด จะตรวจสอบโดยการรับค่าจากการเซ็ตที่ขา T0 และ T1 ของไมโครคอนโทรลเลอร์ 8051 ซึ่งกำหนดค่าไว้ดังนี้

ตาราง 3.2 แสดงการตั้งค่าที่ขา T0, T1 เพื่อกำหนดความเร็วในการรับส่งข้อมูล

T1	T0	อัตราความเร็วในการรับส่งข้อมูล
0	0	-
0	1	1200
1	0	9600
1	1	19200

การตั้งอัตราความเร็วในการรับส่งข้อมูลจะทำในตอนต้นของโปรแกรม ดังนั้นหากมีการตั้งค่าความเร็วการรับส่งข้อมูลที่ขา T0 ,T1 ใหม่จะต้องทำการรีเซ็ตบอร์ดเครื่องจำลองการทำงานของ EPROM ใหม่ทุกครั้ง เพื่อให้โปรแกรมทำการตั้งค่าของไทม์เมอร์ที่รีจิสเตอร์ TH1 ใหม่ตามต้องการ

3.2.5 ข้อตกลงในการรับส่งข้อมูล (Protocal)

รูปแบบของข้อมูลที่ใช้ในการรับส่งจะเป็นรูปแบบของ Intel Hex Format File ทั้งนี้เพื่อให้สามารถรองรับกับการส่งข้อมูลจากเครื่องคอมพิวเตอร์ด้วยคำสั่ง COPY ออกทางพอร์ตอนุกรมได้ รูปแบบของ Intel Hex Format File จะเป็นดังนี้

:BCAAAATTDDDDDD..... .....DDDDCK

:	เป็น Start Character ของแต่ละบรรทัด
BC	เป็น Byte Counter แสดงจำนวนของข้อมูลในบรรทัดนั้น เป็นตัวเลขฐาน 16 ถ้าเป็น “00” หมายถึง End of File
AAAA	เป็น Address เริ่มต้นของข้อมูลไบท์แรกในบรรทัดนั้น
TT	เป็น Type ของข้อมูล ถ้าเป็น “00” หมายถึง เป็นข้อมูล แต่ถ้าเป็น “01” หมายถึง เป็น End of File
DD	เป็นข้อมูลของไฟล์ที่ต้องการ ในกรณีที่ TT เป็น “00”
CS	เป็น Check SUM ของข้อมูลในบรรทัดนั้นทั้งหมด รวมถึง BC, AAAA และ TT ด้วย โดยเป็นผลบวกของข้อมูลทั้งหมดและทำเป็น 2’ Complement

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ในรูปแบบปกติของ Intel Hex Format File จะมีส่วนของ TYPE เพียง 2 แบบคือ 00 และ 01 ซึ่งในโครงงานนี้ถ้าเป็นการส่งข้อมูลจากเครื่องคอมพิวเตอร์ โดยใช้ไฟล์ข้อมูลโดยตรงก็จะมี TYPE เพียง 2 แบบเท่านั้น แต่ถ้าเป็นการทำงานในรูปแบบอื่นๆ เช่น การใส่ข้อมูลซ้ำกันเป็นชุด หรือการย้ายตำแหน่งข้อมูลภายในตัวบอร์ดจะใช้รูปแบบของ TYPE อื่นๆ ซึ่งกำหนดขึ้นเองสำหรับใช้เป็นข้อตกลงในการรับส่งข้อมูล นอกจากนี้การใช้งาน TYPE ที่กำหนดขึ้นเองยังเป็นตัวแสดงให้ทราบว่าจะต้องมีการคืนค่าจากการตรวจุ่สอบการส่งข้อมูลว่าถูกต้องหรือไม่ การตรวจสอบนี้จะอาศัยค่าของ CS หรือค่า Check SUM ของข้อมูลที่ส่งมาเปรียบเทียบกับค่า Check SUM ที่คำนวณจากข้อมูลที่ได้รับ สำหรับ TYPE ที่กำหนดเพื่อใช้งานมีดังนี้

00 ให้รับข้อมูลตามปกติ โดยตรวจสอบค่า Check SUM และแสดงผลทาง LED บนตัวบอร์ดเท่านั้น

01 หยุดการรับส่งข้อมูล

02 ให้ส่งข้อมูลมายังเครื่องคอมพิวเตอร์ ตามจำนวนและตำแหน่งที่กำหนด

03 ให้รับข้อมูลแบบมีการคืนค่าการตรวจสอบค่า Check SUM ว่าถูกต้องหรือไม่

04 ค่าข้อมูลสำหรับการคืนค่าการตรวจสอบ เพื่อแสดงว่าข้อมูลถูกต้อง

05 ค่าข้อมูลสำหรับการคืนค่าการตรวจสอบ เพื่อแสดงว่าข้อมูลไม่ถูกต้อง

3.2.6 การออกแบบโปรแกรมที่ทำงานบนเครื่องคอมพิวเตอร์

ในการออกแบบโปรแกรมที่ทำงานบนเครื่องคอมพิวเตอร์นั้น ได้มีการออกแบบให้เป็นลักษณะของ Object-Oriented ซึ่งใช้คอมไพเลอร์ของ C++ ในการเขียน และมีการกำหนดข้อมูลออกเป็นคลาสต่างๆ กัน ซึ่งข้อมูลที่ต่างประเภทกันแยกออกเป็นคนละคลาส และในแต่ละคลาสก็จะมีการทำงานที่แตกต่างกันออกไป โดยการแบ่งคลาสในโครงงานนี้ได้มีการแบ่งออกเป็น 4 คลาสดังนี้

- ข้อมูลบนบอร์ด EPROM Emulator ที่ไม่มีการตรวจสอบข้อผิดพลาด
- ข้อมูลบนบอร์ด EPROM Emulator ที่มีการตรวจสอบข้อผิดพลาด
- ข้อมูลประเภทไฟล์
- ข้อมูลที่ใช้ในการสื่อสารระดับพอร์ต

3.2.6.1. ข้อมูลบนบอร์ด EPROM Emulator ที่ไม่มีการตรวจสอบข้อผิดพลาด

จะเป็นคลาสของข้อมูลที่ใช้กระทำการใดๆ กับบอร์ด ซึ่งจะเป็นลักษณะของเครื่องคอมพิวเตอร์ส่งข้อมูลให้กับบอร์ดแต่เพียงอย่างเดียว โดยที่บอร์ดจะไม่มีการส่งค่าตรวจสอบความผิดพลาดกลับมาให้เครื่องคอมพิวเตอร์ ซึ่งถ้ามีความผิดพลาดเกิดขึ้นในระหว่างการส่งข้อมูลจากเครื่องคอมพิวเตอร์มายังบอร์ด แล้วทางบอร์ดจะเป็นผู้ตรวจสอบและรายงานความผิดพลาดเอง โดยถ้าเกิดการผิดพลาด ไฟ LED จะมีการกระพริบถี่ๆ ส่วนเครื่องคอมพิวเตอร์นั้นเมื่อส่งเสร็จแล้วก็แล้วกันไป จะเห็นว่าการทำงานในลักษณะเช่นนี้จะคล้ายกับการใช้คำสั่ง COPY ของดอสส่งออกพอร์ตการสื่อสาร เช่น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

C:\>COPY TEST.HEX COM1

ก็จะไม่มีการแสดงความผิดพลาดให้ทราบบนเครื่องคอมพิวเตอร์ สำหรับการทำงานกับข้อมูลชุดนี้จะมีการเติมข้อมูล (FILL) และการใส่ข้อมูล (ENTER) ลงไปในตำแหน่งของแอดเดรสที่ต้องการ ซึ่งการทำงานของทั้งสองคำสั่งมีดังนี้

FILL คำสั่ง FILL จะเป็นการนำค่าข้อมูลค่าหนึ่งที่รับจากผู้ใช้ไปเติมลงในช่วงของแอดเดรสที่ระบุเอาไว้ ซึ่งพารามิเตอร์ของคำสั่ง FILL มีอยู่ 3 ตัวดังนี้

- ค่าแอดเดรสเริ่มต้นที่จะนำข้อมูลไปเติม
- ค่าแอดเดรสสิ้นสุดของการเติม
- ค่าข้อมูลที่จะนำไปเติม

ENTER เป็นการนำข้อมูลที่รับจากผู้ใช้ไปใส่ลงในแอดเดรสที่ต้องการซึ่งจะมีพารามิเตอร์อยู่ 2 ตัว

- ค่าแอดเดรสที่จะนำข้อมูลไปใส่
- ค่าของข้อมูลที่จะนำไปใส่

3.2.6.2. ข้อมูลบนบอร์ด EPROM Emulator ที่มีการตรวจสอบข้อผิดพลาด

จะเป็นคลาสของข้อมูลที่มีการโต้ตอบกัน ระหว่างเครื่องคอมพิวเตอร์กับบอร์ด EPROM Emulator ซึ่งจะมีการตรวจสอบค่า Check SUM บนบอร์ดและมีการส่งค่าของผลการตรวจสอบนั้นกลับมาให้คอมพิวเตอร์ ซึ่งถ้าเกิดข้อผิดพลาดเกิดขึ้นก็จะมีการพยายามแก้ไขโดยการส่งข้อมูลบรรทัดที่ผิดพลาดนั้นลงไปอีกครั้ง การตรวจสอบนี้จะมีการกำหนดให้มีข้อผิดพลาดในการพยายามส่งข้อมูลใหม่ 3 ครั้ง ซึ่งถ้าเกินไปจากนี้ ก็จะมีข้อความบอกกับผู้ใช้ว่า การติดต่อสื่อสารผิดพลาด คำสั่งที่ใช้ในคลาสนี้มีดังนี้

FILL จะต่างกับแบบไม่ตรวจสอบความผิดพลาดตรงที่ เมื่อส่งจนครบหนึ่งบรรทัดแล้ว เครื่องคอมพิวเตอร์จะรอรับค่าของผลการตรวจสอบข้อมูลจากบอร์ด ซึ่งพารามิเตอร์มีอยู่ 3 ตัวคือ

- ค่าแอดเดรสเริ่มต้นที่จะนำข้อมูลไปเติม
- ค่าแอดเดรสสิ้นสุดของการเติม
- ค่าข้อมูลที่จะนำไปเติม

ENTER เป็นการนำข้อมูลที่รับจากผู้ใช้ไปใส่ลงในแอดเดรสที่ต้องการซึ่งจะมีพารามิเตอร์อยู่ 2 ตัว

- ค่าแอดเดรสที่จะนำข้อมูลไปใส่
- ค่าของข้อมูลที่จะนำไปใส่

COMPARE เป็นการเปรียบเทียบข้อมูลในช่วงหนึ่ง กับค่าข้อมูลอีกช่วงหนึ่งว่ามีตัวใดที่เหมือนกันบ้าง ซึ่งพารามิเตอร์มีดังนี้

- ค่าแอดเดรสเริ่มต้นของข้อมูลช่วงแรกที่จะทำการเปรียบเทียบ
- ค่าแอดเดรสสิ้นสุดของข้อมูลชุดแรก
- ค่าแอดเดรสเริ่มต้นของข้อมูลช่วงที่สองที่จะตรวจสอบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SEARCH เป็นการค้นหาข้อมูลตามที่ผู้ใช้ได้ใส่ค่าที่ต้องการจะค้นหา ซึ่งสามารถค้นหาได้ทีละหลายๆ ไบท์ และพารามิเตอร์ของการค้นหามีดังนี้

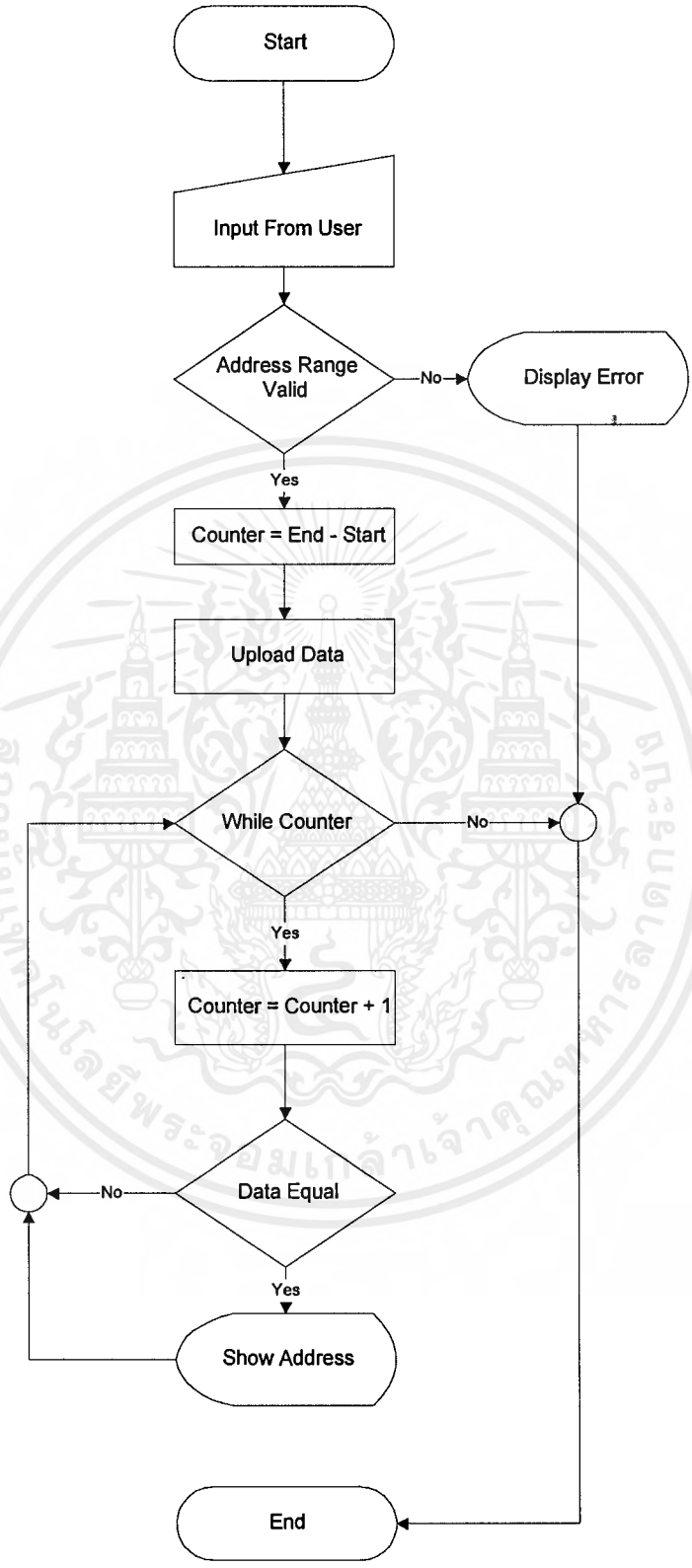
- ค่าแอดเดรสเริ่มต้นของข้อมูลช่วงแรกที่จะทำการค้นหา
- ค่าแอดเดรสสิ้นสุดของข้อมูล
- ค่าของข้อมูลที่ต้องการค้นหา

MOVE เป็นการทำสำเนาค่าข้อมูลจากชุดของตำแหน่งหนึ่งไปยังอีกตำแหน่งหนึ่ง ซึ่งมีพารามิเตอร์ดังนี้

- ค่าแอดเดรสเริ่มต้นของข้อมูลต้นทาง
- ค่าแอดเดรสสุดท้ายของข้อมูลต้นทาง
- ค่าแอดเดรสเริ่มต้นของข้อมูลปลายทาง

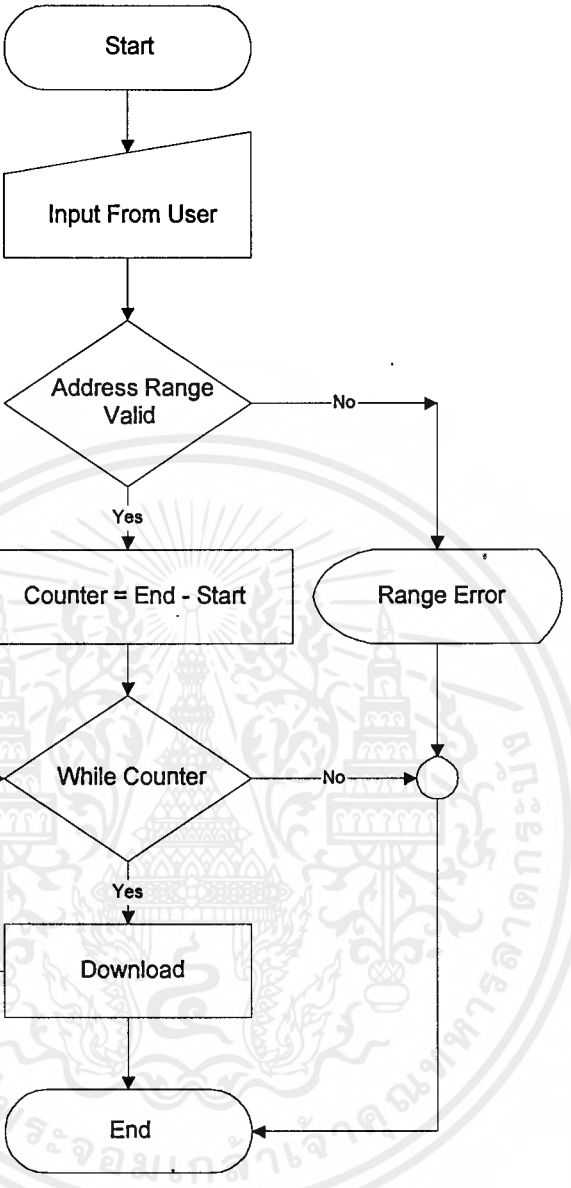
โฟลว์ชาร์ตของการทำงานคำสั่งต่าง ๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



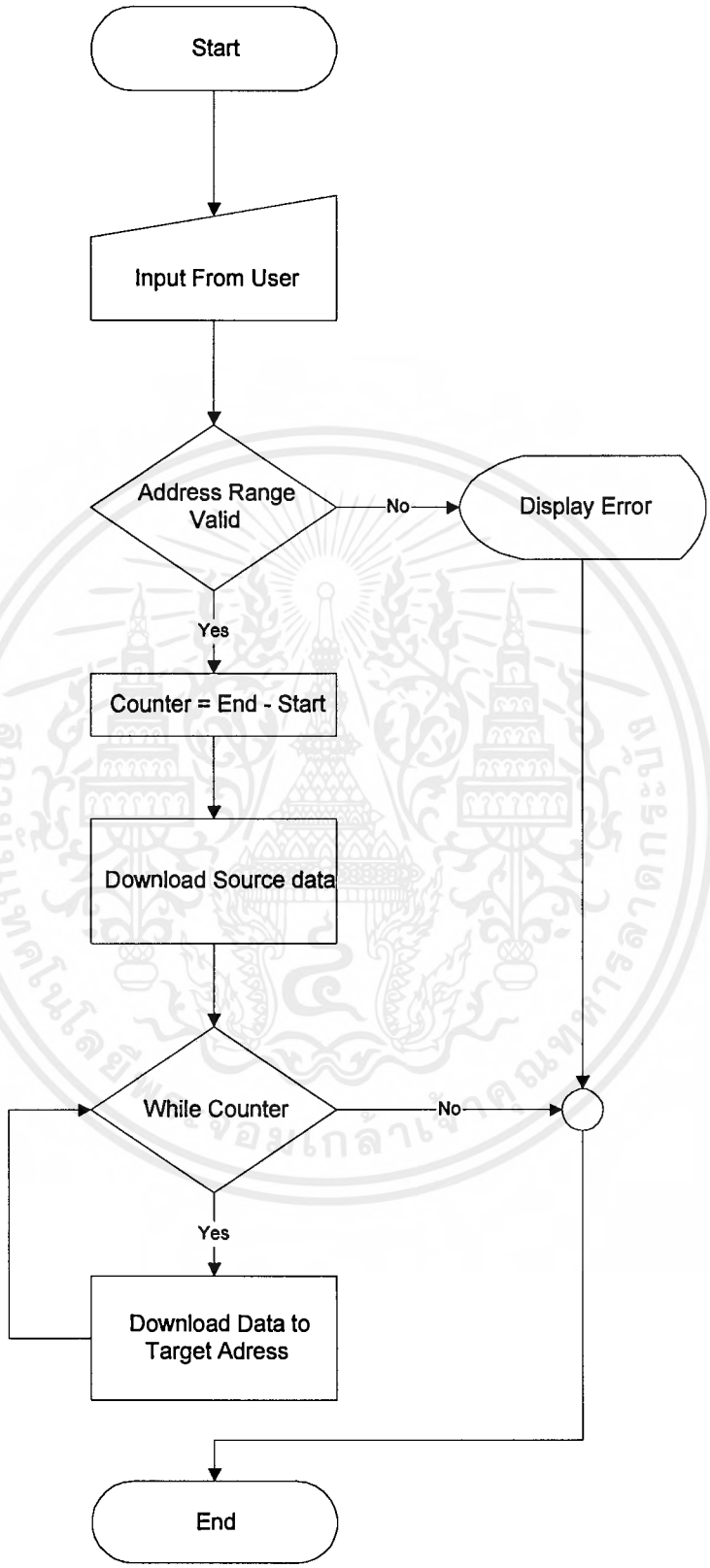
รูปที่ 3.14 แสดงโฟลว์ชาร์ตของคำสั่ง Compare

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



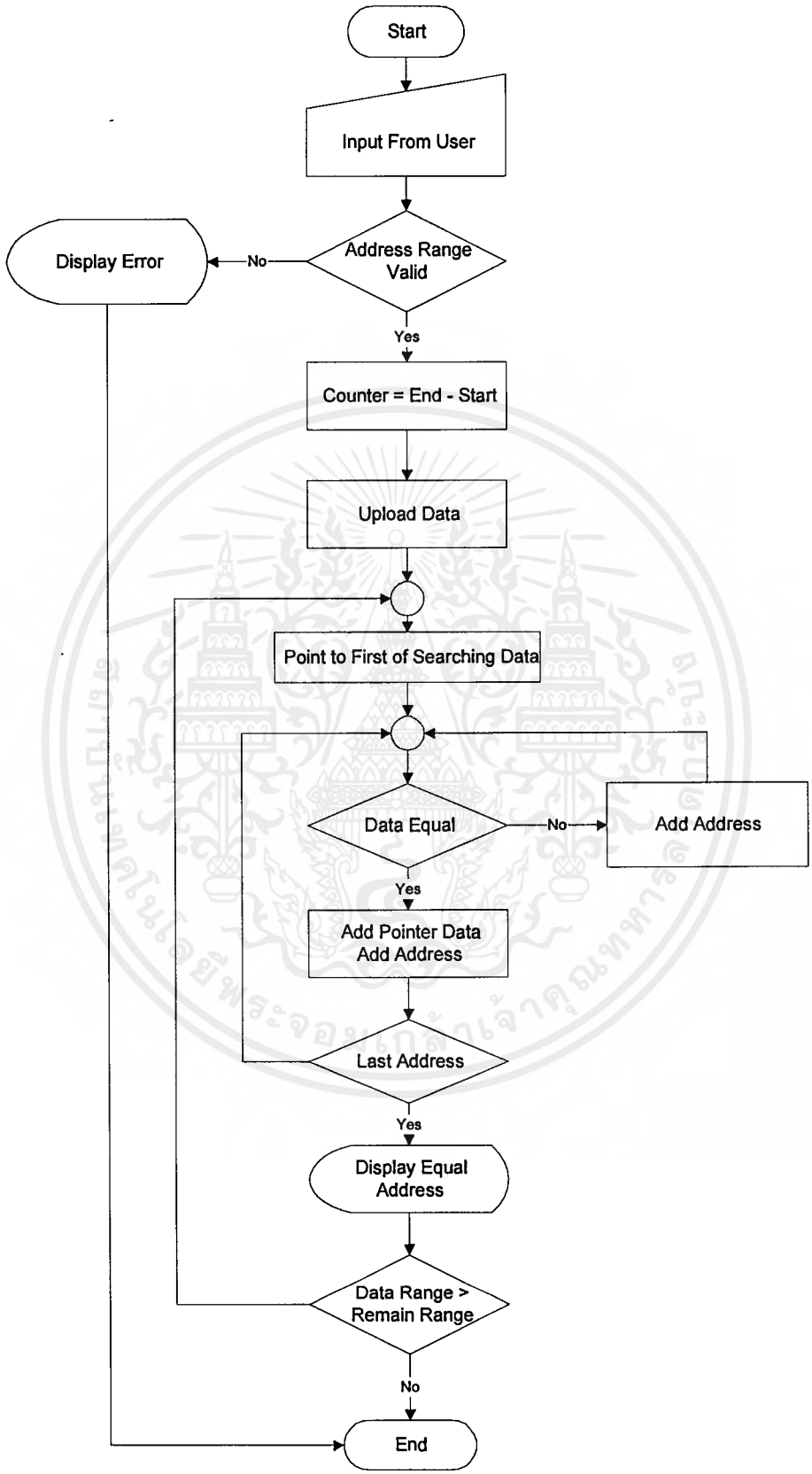
รูปที่ 3.15 แสดงโฟลว์ชาร์ตของคำสั่ง Fill

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.16 แสดงโฟลว์ชาร์ตการทำงานของคำสั่ง Move

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.17 แสดงโฟลว์ชาร์ตการทำงานของคำสั่ง Search

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2.6.3.ข้อมูลไฟล์

ข้อมูลประเภทไฟล์นี้จะเป็นข้อมูลของไฟล์ที่อยู่บนดิสก์ ซึ่งจะมีการเรียกใช้เมื่อต้องการขอดูข้อมูลในไฟล์ แบบ INTEL HEX FORMAT และการ DOWNLOAD ข้อมูลจากไฟล์ลงไปยังบอร์ด ซึ่งไฟล์ที่จะทำการดาวน์โหลดนั้นก็ต้องเป็นรูปแบบของ INTEL HEX FORMAT เท่านั้น เพราะในการขอดูข้อมูลในไฟล์จะมีการกำหนดลักษณะของการแสดงผลให้เป็นไปตามรูปแบบของ INTEL HEX FORMAT ซึ่งถ้าไม่ใช่รูปแบบนี้อาจจะทำให้การแสดงผลเกิดความผิดพลาดได้ และในส่วนของการดาวน์โหลดไฟล์ลงไปที่บอร์ดนั้นก็ต้องเป็นไฟล์รูปแบบของ INTEL HEX FORMAT เช่นกัน ทั้งนี้เนื่องจากโปรแกรมที่ทำงานบนบอร์ดได้มีการออกแบบให้ทำงานกับรูปแบบนี้เท่านั้น

3.2.6.4.ข้อมูลที่ใช้ในการติดต่อสื่อสารระดับพอร์ต

ในการติดต่อสื่อสารกับพอร์ตอนุกรมนั้นจะมีการเรียกใช้อินเตอร์รัพรูทีนของดอสเพื่อช่วยในการติดต่อโดยจะใช้อินเตอร์รัพหมายเลข 14h ของดอส โดยที่ค่าพารามิเตอร์จะมีอยู่ 3 ค่าคือ ค่า Port ,ค่า Code ซึ่งเป็นค่าของบอดเรทและค่าของพาริตี้ต่างๆ และค่าของตัวอักษรที่จะส่งจะมีการส่งผ่านค่าไปให้กับฟังก์ชั่นโดยค่า Code และ Port จะได้มาจากการเซ็ตค่าโดยผู้ใช้ผ่านรูทีนที่ทำงานเกี่ยวกับการเซ็ตค่าพอร์ตอนุกรม ซึ่งก็อยู่ในคลาสนี้เช่นกัน

ในการเรียกอินเตอร์รัพของดอสผ่าน C++ นี้จะมีรูปแบบการกำหนดเหมือนกับภาษา C คือจะมีการให้ตัวแปร 1 ตัวเป็นชุดของรีจิสเตอร์ดังนี้

```
union REGS r;
```

และในการอ้างถึงรีจิสเตอร์ก็จะมีรูปแบบเช่นนี้

```
r.h.ah = 0; อ้างถึงรีจิสเตอร์ขนาด 8 บิท
r.x.dx = 0; อ้างถึงรีจิสเตอร์ขนาน 16 บิท
```

และตัวอย่างในการส่งข้อมูลหนึ่งตัวอักษร เพื่อส่งออกสู่พอร์ตอนุกรมมีตัวอย่างดังนี้

```
void send(int port, int code)
 {
 r.h.ah =1;
 r.h.al = ch;
 r.x.dx =  ch;
 int86(0x014,&r,&r);
 }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4 การทดลองและผลการทดลอง

4.1 ส่วนของ Hardware

การทดลองในส่วนของฮาร์ดแวร์แบ่งได้ตามหัวข้อดังนี้

- Serial Link EPROM Emulator
- Buffer RS-232C
- Duplexer
- Modem
- ภาคส่ง FM และภาครับ FM
- ทดสอบรวมทั้งระบบ (ส่งแบบทางเดียว)
- ทดสอบรวมทั้งระบบ (ส่งแบบสองทาง)

4.1.1 Serial Link EPROM Emulator

<u>การทดลอง</u>

สร้าง Monitor Program สำหรับทดสอบ EPROM Emulator โดยการสั่งให้รับข้อมูลเข้ามาทางพอร์ต rx ของ MCS-51 ผ่าน RS-232 Buffer MAX-232 แล้วนำข้อมูลที่ได้รับไปเก็บไว้ในหน่วยความจำของบอร์ด EPROM Emulator แล้วทำการ Enable Buffer ส่วนที่ทำการ Interface กับ Dip Jumper แล้วใช้ ET Board ตรวจสอบความถูกต้องของข้อมูลที่ได้รับ

ในด้านการส่งข้อมูลนั้น จะใช้พีซีส่งข้อมูลที่เป็นรูปแบบของ Intel Hex Format File ผ่านทางพอร์ตอนุกรมของพีซี ไปสู่ Buffer RS-232 ของบอร์ด EPROM Emulator ซึ่งในรูปที่ 4.1.1.1 แสดงลักษณะของข้อมูลในไฟล์ที่เก็บในลักษณะของ Intel Hex Format

```
:10000000000102030405060708090A0B0C0D0E0F78
:10001000101112131415161718191A1B1C1D1E1F68
:00000001FF
```

รูปที่ 4.1 ลักษณะข้อมูลในไฟล์ที่เป็นแบบ Intel Hex Format (Test.hex)

<u>ผลการทดลอง</u>

ข้อมูลที่ได้รับแสดงในรูปที่ 4.2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

00006000: 0 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F :

00006010: 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

รูปที่ 4.2 แสดงข้อมูลที่อ่านได้จาก ET Board ที่ตำแหน่ง 6000-601F

4.1.2 RS-232 Buffer

การทดลอง

1.ให้ทำการป้อน Input ที่ขา R_{1I} และ R_{2I} ด้วยระดับแรงดัน +15 , +10, 0 ,-5 , -10 , -15 แล้วทำการวัดระดับแรงดันที่ได้จากขา R_{1O} และ R_{2O}

2. ให้ทำการป้อน Input ที่ขา T_{1I} และ T_{2I} ด้วยระดับแรงดัน +5V และ 0 V แล้วทำการวัดระดับแรงดันที่ขา T_{1O} และ T_{2O}

ผลการทดลอง

ตารางที่ 4.1 แสดงความสัมพันธ์ของแรงดันระหว่างขา R1I,R2I และ R1O,R2O

R1I,R2I	R1O	R2O
+15	0	0
+10	0	0
+5	0	0
0	+5	+5
-5	+5	+5
-10	+5	+5
-15	+5	+5

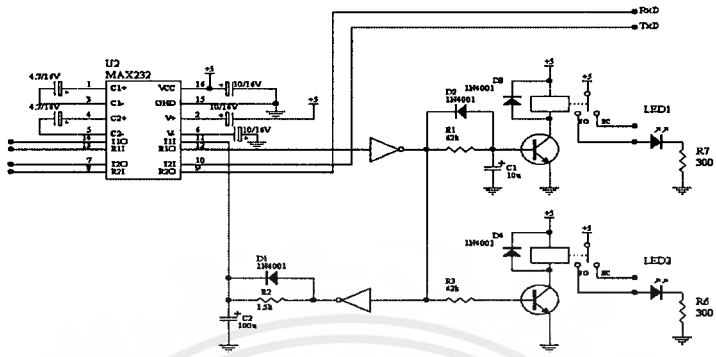
ตารางที่ 4.2 แสดงความสัมพันธ์ของแรงดันระหว่างขา T1I,T2I และ T1O,T2O

T1I,T2I	T1O	T2O
0	+10	+10
+5	-10	-10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.3 วงจร Duplexer

การทดลอง



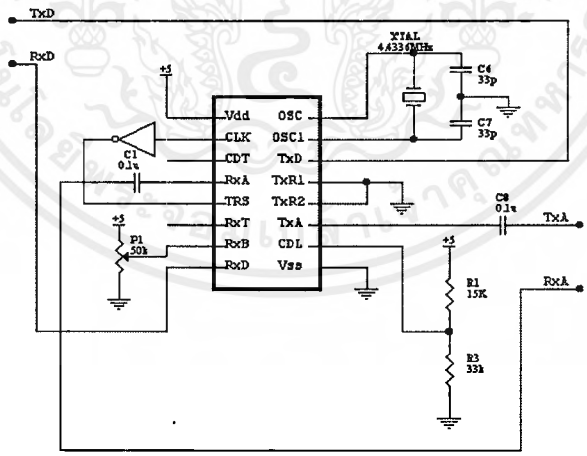
รูปที่ 4.3 แสดงการทดสอบวงจร Duplexer

ให้ใช้มิเตอร์วัดระดับแรงดันที่ขา T1O ไว้ แล้วทำการป้อนแรงดันขนาด +5 V ถึง + 15 V เข้าที่ขา R1I

ผลการทดลอง

จะสังเกตุเห็นว่า LED1 จะติด แล้วตามด้วน LED2 แล้วจึงตามด้วยมีแรงดันที่ขา Txo เปลี่ยนจากระดับ -10 V ไปเป็น + 10V ตามลำดับ

4.1.4 Modem



รูปที่ 4.4 แสดงวงจรของโมเด็ม

การทดลอง

1. ให้ป้อนแรงดัน +5 V และ 0 เข้าที่ Txo แล้วทำการทดลองวัดความถี่ที่ได้จาก TxA
2. ให้ป้อนความถี่ 1200 Hz และ 2700 Hz เข้าที่ขา RxA แล้วทดลองใช้มิเตอร์วัดแรงดันที่ได้จาก RxD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผลการทดลอง

Input TXD (Volt)	Output TXA (Hertz)
0	2200
+5V	1200

Input RXA (Hertz)	RAD Output (Volt)
2200	0
1200	+5V

4.1.5 ภาคส่งและภาครับแบบ FM

การทดลอง

ให้ปรับแต่งเครื่องส่งโดยการปรับความถี่ของเครื่องส่งและใช้เครื่องวัดความถี่ที่เสาอากาศของเครื่องส่งจนได้ความถี่ 78 MHz ตามที่ต้องการ แล้วป้อนสัญญาณความถี่ประมาณ 1200 Hz เข้าที่อินพุทของวงจร แล้วจึงปรับแต่งเครื่องรับจนรับสัญญาณเสียงนั้นๆ ได้

4.1.6 ทดสอบรวมทั้งระบบ (ส่งแบบทางเดียว)

การทดลอง

ให้ประกอบวงจรส่วนต่างๆทั้งหมดเข้าด้วยกันตาม Block diagram รวมของระบบ แล้วทำการทดสอบดูว่าระบบทำงานได้ตามวัตถุประสงค์หรือไม่ โดยทำการส่งข้อมูลในรูปแบบของ Intel Hex Format โดยใช้ข้อมูลขนาด 10,000 ไบท์ ที่ถูกกำหนดให้เป็น ตัวเลข 0 - 9 วนไปเรื่อยๆ แล้วทำการตรวจสอบดูว่าข้อมูลที่ได้รับถูกต้องหรือไม่ แล้วเพิ่มระยะห่างระหว่างเครื่องรับและเครื่องส่งให้มากขึ้นเรื่อยๆ แล้วตรวจสอบดูว่าข้อมูลที่ไดรับยังถูกต้องทั้งหมดหรือไม่

ผลการทดลอง

จากการทดลอง พบว่าความถูกต้องของข้อมูล และระยะห่างของเครื่องรับส่ง จะขึ้นอยู่กับสภาพแวดล้อมว่ามีสัญญาณรบกวนหรือไม่หากมีสัญญาณรบกวนมากจะทำให้ความถูกต้องของข้อมูลต่ำมากและระยะห่างของเครื่องรับส่งน้อยมาก ซึ่งบางครั้งระยะห่างก็ไม่ถึง 30 เซนติเมตรด้วย (ซึ่งในบางครั้งสัญญาณรบกวนนี้ก็เกิดจากเครื่องคอมพิวเตอร์เองด้วย) แต่หากอยู่ในสถานที่ที่ไม่มีสัญญาณรบกวนก็จะทำให้รับข้อมูลได้ถูกต้องทั้งหมดที่ระยะห่างประมาณ 15 เมตร และจากการทดลองนี้พบว่าเหตุที่ทำให้ได้ระยะห่างน้อยเนื่องจากประสิทธิภาพของเครื่องรับส่งไม่ดีพอทำให้เกิดสัญญาณรบกวนแทรกซ้อนได้ง่าย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.1.7 ทดสอบรวมทั้งระบบ (ส่งแบบสองทาง)

<u>การทดลอง</u>

ทำการทดลองโดยใช้ Program EZ ที่สร้างขึ้นซึ่งอยู่ในส่วนของภาคผนวก ทำการส่งข้อมูลไปยังเครื่องส่วน EPROM EMULATOR โดยสั่งให้ส่งสัญญาณตอบรับกลับมาด้วย โดยการทดสอบระยะทางไปด้วยในตัว

<u>ผลการทดลอง</u>

จากการทดลองพบว่าการสั่งให้ส่งสัญญาณตอบรับกลับจะทำได้ในระยะทางที่สั้นมากเนื่องจากภาคส่งชุดที่ใช้สำหรับการส่งกลับมีประสิทธิภาพต่ำซึ่งหากเป็นการส่งในระยะใกล้และมีสัญญาณรบกวนน้อยก็จะทำให้สามารถส่งข้อมูลได้ถูกต้องทั้งหมด

4.2 การทดลองส่วน Software

ในการเขียนโปรแกรมขึ้นมาเพื่อใช้ในการสื่อสารกับบอร์ด Eprom Emulator นี้ในขั้นตอนแรกได้มีการได้มีการเขียนโปรแกรมในลักษณะของ Procedural Programming โดยใช้ภาษา C ในการเขียนโปรแกรม จากนั้นเมื่อโปรแกรมมีส่วนของการทำงานต่างๆ ที่มากขึ้นและโปรแกรมก็มีขนาดและความซับซ้อนมากขึ้น อย่างเช่นในส่วนของการรับข้อมูลจากแป้นพิมพ์ได้มีการเขียนขึ้นมาใหม่ เนื่องจากไม่สามารถใช้คำสั่งที่มีอยู่ได้ เป็นต้น เมื่อมีความซับซ้อนของโปรแกรมมากขึ้น ทำให้การเขียนโปรแกรมไม่สามารถทำได้โดยสะดวก และในบางครั้งก็ไม่สามารถทำตามโครงสร้างที่ได้วางไว้ได้ จึงมีการจัดโครงสร้างของโปรแกรมขึ้นมาใหม่

การเขียนโปรแกรมขึ้นมาใหม่นี้ได้เขียนเป็นลักษณะของ Object Oriented Programming โดยใช้ภาษา C++ ในการเขียนซึ่งสามารถลดความซับซ้อนของข้อมูลต่างๆ และกรรมวิธีต่างๆ ลงไปได้มาก

ส่วนของการติดต่อกับพอร์ตอนุกรมนั้นได้มีการใช้อินเตอร์รัพหมายเลข 14H ของดอสและเขียนผ่าน C++ โดยใช้คำสั่ง int86 ซึ่งในบางครั้งก็ไม่สามารถทำตามความต้องการได้

ในการทำงานที่เกี่ยวกับไฟล์นั้นเมื่อทำการทดลองแล้วจะพบว่าเมื่อเปิดไฟล์ที่มีขนาดใหญ่มากๆ ขึ้นมาทำงานแล้ว จะทำให้ส่วนต่างของโปรแกรมมีความผิดเพี้ยนไปหรือบางครั้งอาจทำให้โปรแกรมไม่ทำงานได้ ซึ่งคาดว่าอาจจะเป็นผลของการ Over Flow ค่าใดค่าหนึ่งของการทำงานซึ่งแม้ว่าตัวแปรส่วนใหญ่ได้ทำการป้องกันโดยใช้ลักษณะของคลาสใน C++ เป็นตัวป้องกันแล้วก็ตาม

การทดลองคำสั่งต่าง ๆ

- การทดลองคำสั่ง Fill ในการทดลองคำสั่ง Fill นี้ได้มีการทดลองส่งข้อมูลตามขนาดของข้อมูล กล่าวคือ จะมีการทดลอง Fill ข้อมูลขาดเล็กก่อนซึ่งมีขนาดของข้อมูลประมาณไม่เกิน 16 ไบท์ ซึ่งไม่พบปัญหาใด ๆ จากนั้นจึงเพิ่มขนาดของข้อมูลให้ใหญ่ขึ้นเรื่อย ๆ ในช่วงแรกของการพัฒนานั้นจะพบปัญหาอยู่ที่ข้อมูลไบท์สุดท้ายจะไม่มีการ FILL ข้อมูลลงในบอร์ด เมื่อพบปัญหา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และตรวจดูก็พบว่าเป็นความสับสนในการตรวจสอบจำนวนครั้งของการทำงานจากคำสั่ง While ซึ่งเมื่อแก้ไขปัญหาดังกล่าวแล้วก็สามารถใช้งานได้ปกติ

- การทดลองใช้คำสั่ง Enter ในการใช้คำสั่ง Enter นั้นไม่มีการพบปัญหาเนื่องจากว่าเป็นการส่งข้อมูลไบท์เดียวซึ่งไม่มีความซับซ้อนมาก
- การทดลองใช้คำสั่ง Move ในการทดลองใช้คำสั่ง Move นั้นจะพบปัญหาคล้าย ๆ กับการ Fill กล่าวคือ จะมีการสับสนเรื่องของจำนวนของข้อมูลซึ่งปัญหานี้ได้รับการแก้ไขแล้วจากการทดลองใช้คำสั่ง Fill จึงง่ายต่อการทดลองและแก้ไขส่วนอื่นที่เหลือ
- การทดลองใช้คำสั่ง Compare ในการใช้คำสั่ง Compare นี้จะพบปัญหาเช่นเดียวกับที่พบมาในเรื่องของขนาดข้อมูลซึ่งสามารถแก้ไขได้โดยง่าย และจะพบปัญหาอื่นที่ตามมาอีกคือในการ Compare นี้ถ้าพบข้อมูลที่ไม่เท่ากันแล้วจะมีการแสดงผลออกที่หน้าจอ ถ้าข้อมูลไม่มากก็ไม่มีปัญหาแต่ถ้าข้อมูลที่ไม่เท่ากันมีจำนวนมากกว่าบรรทัดของหน้าจอ ที่สามารถแสดงได้ก็จะเกิดข้อผิดพลาดขึ้นที่บอร์ด เนื่องจากว่าในการแสดงผลที่เกินหน้าจอนั้นจะมีการใช้คำสั่ง getch() เพื่อรอรับข้อมูลการกดแป้นพิมพ์จากผู้ใช้ และในขณะที่รอนี้จะไม่มีการส่งข้อมูลเพื่อบอกว่าข้อมูลสิ้นสุดแล้ว ดังนั้นในช่วงนี้หากมีสัญญาณรบกวนเข้ามา บอร์ดก็จะมีการตรวจสอบค่า Sum ซึ่งจะไม่ตรงกับความเป็นจริงทำให้เกิดความผิดพลาดได้ ซึ่งได้มีการวางโครงสร้างข้อมูลของส่วนนี้ขึ้นใหม่ซึ่งแทนที่จะพิมพ์เมื่อพบข้อมูลไม่เท่ากันออกสู่หน้าจอเลย ก็จะมีการเก็บค่าเอาไว้ก่อนซึ่งเมื่อจบขนาดของข้อมูลแล้วจึงจะมีการแสดงผลออกทางหน้าจอ ซึ่งเมื่อได้ทดลองส่วนนี้แล้วพบว่าสามารถใช้งานได้จริง
- การทดลองใช้คำสั่ง Search ในการทำงานของคำสั่ง Search นี้เป็นส่วนที่ซับซ้อนและยากมากของส่วนซอฟท์แวร์ในโครงงานนี้เพราะจะมีการตรวจสอบหลายขั้นตอนและจะมีการเก็บค่าและคืนค่าต่าง ๆ ของฟังก์ชั่นการทำงานมากมาย ซึ่งในจุดนี้พบปัญหาหลายครั้งและหลายรูปแบบซึ่งถึงแม้ว่าจะเขียนโปรแกรมตามผังที่ได้วาดไว้ก็จะพบว่าเป็นการยากต่อการ Implement ออกมาให้เป็นภาษาคอมพิวเตอร์เพราะจะมีการคาบเกี่ยวกันของ Loop ในบางส่วน การทดลองและพัฒนาในส่วนของคำสั่ง Search นี้จึงใช้เวลานานพอสมควรจึงสามารถทำงานได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5 สรุปและวิจารณ์

จากผลงานของโครงงานนี้เมื่อทำสำเร็จออกมาแล้วนั้นยังมีข้อบกพร่องอยู่หลายอย่างดังนี้

1. แผ่นวงจรพิมพ์ของโครงงานในส่วนที่เป็น EPROM Emulator เมื่อเสร็จแล้วจะมีขนาดใหญ่พอสมควร ซึ่งเมื่อนำมาเปรียบเทียบกับขนาดของชิ้นงานเดิมที่มีขายอยู่ โครงงานนี้จะมีขนาดใหญ่กว่าประมาณเกือบเท่าตัว ถ้าจะนำมาใช้งานจริงจะมีจำกัดในเรื่องเนื้อที่ ดังนั้นขนาดของแผ่นวงจรพิมพ์ควรจะออกแบบให้มีขนาดที่เล็กกว่าที่เป็นอยู่
2. โครงงานในส่วนของการสื่อสารไร้สายก็จะมีปัญหาคล้ายๆ กับแผ่นวงจรพิมพ์ กล่าวคือจะมีขนาดใหญ่มากกว่าที่ควรจะเป็น ดังนั้นควรออกแบบให้มีขนาดเล็กลงกว่านี้
3. ในเรื่องของสัญญาณในการสื่อสารกันนั้นมีการใช้เสาอากาศด้านส่งค่อนข้างใหญ่ แต่เสาอากาศทางด้านรับก็ใหญ่พอสมควร ซึ่งควรจะให้มีขนาดเล็กลงกว่านี้ และลดเสาอากาศจาก 2 ต้น มาเป็นเสาอากาศเสาเดียว แต่จะมีปัญหาในเรื่องการใช้ Relay ที่ใช้ตัดต่อในย่านความถี่สูงซึ่งมีราคาแพงจึงไม่สามารถทำให้เล็กลงไปได้
4. วงจรที่ใช้ในการรับส่งมีประสิทธิภาพไม่ดีพอทำให้เกิดสัญญาณรบกวนแทรกซ้อนได้ง่ายมาก จึงควรมีการออกแบบวงจรสำหรับการรับส่งใหม่ให้มีประสิทธิภาพสูงกว่าที่เป็นอยู่ เพื่อให้ได้ระยะทางในการรับส่งได้ไกลขึ้น
5. ในขณะนี้ยังทำการส่งข้อมูลได้เพียงทางเดียว เนื่องจากวงจรภาครับส่งที่ใช้ทั้งสองชุดมีประสิทธิภาพไม่เหมือนกัน แต่ถ้าเป็นการทำงานโดยใช้สายสัญญาณ RS232 จะสามารถทำงานทั้งรับและส่งได้เป็นอย่างดี
6. ในส่วนของโปรแกรมนั้นไม่ค่อยพบปัญหาเท่าใดนัก เนื่องจากส่วนใหญ่แล้วจะเป็นการทำงานในเรื่องของโครงสร้างข้อมูล ซึ่งสามารถหาหนังสือมาอ่านประกอบการทำงานได้ แต่ในการโปรแกรมส่วนอื่นๆ เช่นในการรับข้อมูลจากแป้นพิมพ์โดยรับเป็นลักษณะตามโครงสร้างของอักขระที่ได้วางเอาไว้ ไม่มีหนังสือใดที่เป็นแนวทางในการเขียน ทำให้พัฒนาในเรื่องของการรับข้อมูลจากผู้ใช้ค่อนข้างนาน และยังไม่ค่อยดีเท่าที่ควร จึงควรมีการพัฒนาโปรแกรมในส่วนของการรับข้อมูลจากผู้ใช้ให้ดีขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ก

โปรแกรมสำหรับควบคุมเครื่องจำลองการทำงานของ EPROM แบบไร้สาย

```
;PROGRAM RECEIVE-SEND FOR WIRELESS EPROM EMULATOR
;BY 3P

;************** CONSTAND ***************
LED             EQU     INT0
BUFF541         EQU     INT1
BAUD1           EQU     T0
BAUD2           EQU     T1
RATE12          EQU     0E8H
RATE24          EQU     0F4H
RATE48          EQU     0FAH
RATE96          EQU     0FDH
PORT2           EQU     P2
PORT1           EQU     P1
PORT0           EQU     P0
LED_TIME        EQU     08000H          ;DELAY FOR LED ON-OFF
TLED_ERR        EQU     0B000H          ;DELAY FOR LED ERROR


;************** VARIABLE ******************
                ORG     0
                DS      8
STACK_:         DS      24

;************** CODE *****************
                ORG     0
RESET_:         MOV     R1,#40H         ;POWER ON DELAY
                DJNZ    R1,$

                MOV     TMOD,#20H       ;T1 MODE2
                MOV     SCON,#52H       ;MODE 8 BIT VART
                SETB    TR1             ;T1 ON
                MOV     A,PCON
                CLR     ACC.7
                MOV     PCON,A          ;CLR SMOD
                MOV     SP,#STACK_      ;SET STACK ADDRESS

                SETB    BAUD1           ;CLR BAUD RATE
                SETB    BAUD2
                CLR     A
                MOV     C,BAUD1         ;GET BAUD RATE JUMPPER
                MOV     ACC.0,C
                MOV     C,BAUD2
                MOV     ACC.1,C
                CJNE    A,#01,NOT_12
                MOV     TH1,#RATE12     ;BAUD RATE 1200
                LJMP    END_BRATE       ;NOT USE 2400
NOT_12:         CJNE    A,#02,NOT_48
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                MOV     TH1,#RATE48         ;BAUD RATE 4800
                LJMP    END_BRATE
NOT_48:         CJNE    A,#03,NOT_96
                MOV     TH1,#RATE96         ;BAUD RATE 9600
                LJMP    END_BRATE
NOT_96:         MOV     TH1,#RATE96         ;BAUD RATE 19200
                MOV     A,PCON
                SETB    ACC.7
                MOV     PCON,A              ;SET SMOD FOR DUBBEL B-RATE
END_BRATE:      MOV     DPTR,#0             ;SET INITIAL ADDRESS
                MOV     R5,#0               ;CLR ERROR COUNTER

;ENABLE BUFFER
EN_BUFF:        MOV     A,#0FFH             ;PULL UP RAM
                MOV     PORT0,A
                MOV     PORT2,A
                SETB    PORT1
                CLR     BUFF541             ;ENABLE 541 BUFFER
                CLR     LED

;STAND BY
STAND_BY:       LCALL   RW_CHAR             ;RECEIVE AND WAIT FOR CHAR
                CJNE    A,#':',STAND_BY     ;DATA_ERR
                LJMP    R_DATA              ;TO RECEIVE DATA
DATA_ERR:       LJMP    LED_ERR

;RECEIVE DATA
R_DATA:         SETB    BUFF541             ;DISABLE 541 BUFFER
                MOV     R7,#0               ;CLR CHK SUM
                LCALL   RNUM_CHK            ;RECEIVE NUMBER AND CHK SUM
                MOV     R6,A                ;GET BYTE COUNTER
                LCALL   RNUM_CHK
                MOV     DPH,A               ;RAM HIGH ADDRESS
                LCALL   RNUM_CHK
                MOV     DPL,A               ;RAM LOW ADDRESS
                LCALL   RNUM_CHK
                CJNE    A,#00,NOT_D_LD      ;CHK TYPE
                LJMP    DOWN_LD             ;TYPE 00 = DOWN LOAD
NOT_D_LD:       CJNE    A,#01,NOT_END
                LJMP    EN_BUFF             ;TYPE 01 = END FOR DATA
NOT_END:        CJNE    A,#02,NOT_U_LD
                LJMP    UP_LD               ;TYPE 02 = UP LOAD
NOT_U_LD:       CJNE    A,#03,DATA_ERR
                LJMP    D_RETURN            ;TYPE 03 = DONW LOAD
                                            ; AND RETURN CHK SUM

;DOWN LOAD NO RETURN CHK SUM
DOWN_LD:        SETB    PORT1.0
DOWN_LD1:       LCALL   RNUM_CHK
                MOVX    @DPTR,A
                INC     DPTR
                MOV     C,ACC.0
                MOV     LED,C               ;TURN ON-OFF LED
                DJNZ    R6,DOWN_LD
                MOV     A,R7                ;CHK SUM
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                CPL     A
                INC     A               ;2'COMPLEMENT CHK SUM
                MOV     B,A
                LCALL   RNUM_CHK        ;RECEIVE CHK SUM DATA
                CJNE    A,B,LED_ERR     ;IF ERROR THEN DISPLAY LED
                LJMP    STAND_BY

;UP LOAD TO PC
UP_LD:          CLR     PORT1.0
                JB      PORT1.1,$
UP_LD1:         MOVX    A,@DPTR         ;READ DATA FROM RAM
                LCALL   SNUM_CHK        ;SEND NUMBER AND CHK SUM
                INC     DPTR
                MOV     C,ACC.0
                MOV     LED,C           ;TURN ON-OFF LED
                DJNZ    R6,UP_LD1       ;R6 = BYTE COUNTER
                MOV     A,R7
                CPL     A
                INC     A               ;2'COMPLEMENT CHK SUM
                LCALL   SNUM_CHK
                LJMP    STAND_BY

;DOWN LOAD AND RETURN CHK SUM
D_RETURN:       SETB    PORT1.0
D_RETURN1:      LCALL   RNUM_CHK
                MOVX    @DPTR,A
                INC     DPTR
                MOV     C,ACC.0
                MOV     LED,C           ;TURN ON-OFF LED
                DJNZ    R6,D_RETURN1
                MOV     A,R7            ;CHK SUM
                CPL     A
                INC     A               ;2'COMPLEMENT CHK SUM
                MOV     B,A
                LCALL   RNUM_CHK        ;RECEIVE CHK SUM DATA
                CJNE    A,B,RET_ERR     ;IF ERROR THEN RETURN ERROR
                CLR     PORT1.0
                JB      PORT1.1,$
                MOV     A,#':'          ;SEND ':' FOR RETURN
                MOV     R0,#0FFH
                DJNZ    R0,$
                LCALL   SNUM_CHK
                MOV     A,#04           ;RETURN 04 = NO ERROR
                MOV     R0,#0FFH
                DJNZ    R0,$
                LCALL   SNUM_CHK
                MOV     R5,#0           ;CLR ERROR COUNTER
IN_LIMIT:       LJMP    STAND_BY
RET_ERR:        CLR     PORT1.0
                JB      PORT1.1,$
                MOV     A,#':'          ;SEND ':' FOR RETURN
                MOV     R0,#0FFH
                DJNZ    R0,$
                LCALL   SNUM_CHK
                MOV     A,#05           ;RETURN 05 = ERROR
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                MOV     R0,#0FFH
                DJNZ    R0,$
                LCALL   SNUM_CHK
                INC     R5
                CJNE    R5,#03,IN_LIMIT         ;CHK 3 ERROR ?
                LJMP    LED_ERR

;********************* PROCEDURE AREA *****************

;DISPLAY LED ERROR
LED_ERR:        MOV     R0,#HIGH TLED_ERR
ERR1:           MOV     R1,#LOW TLED_ERR
                DJNZ    R1,$
                DJNZ    R0,ERR1
                JNB     LED,LED_OFF
                CLR     LED
                LJMP    LED_ERR
LED_OFF:        SETB    LED
                LJMP    LED_ERR

;RECEIVE AND WAIT FOR 1 CHAR
;O/P            A

RW_CHAR:        SETB    PORT1.0
                MOV     R0,#HIGH LED_TIME       ;SET TIME FOR LED ON-OFF
SET_R1:         MOV     R1,#LOW LED_TIME
RECEV:          JB      RI,END_RW               ;WAIT UNTIL DATA RECEIVE
                MOV     A,#6
REC_DEL:        DEC     A
                JNZ     REC_DEL
                DJNZ    R1,RECEV
                DJNZ    R0,SET_R1
                JB      LED,CLR_BIT
                SETB    LED
                LJMP    RW_CHAR
CLR_BIT:        CLR     LED
                LJMP    RW_CHAR
END_RW:         CLR     RI
                MOV     A,SBUF                  ;MOVE DATA TO A
                RET

;RECEIVE ASCII 2 BYTE CONVERT TO HEX 1 BYTE
; AND ADD TO CHK SUM
;O/P            DATA = A
;               CHK SUM = R7
RNUM_CHK:       LCALL   R_BYTE                  ;RECEIVE 1 BYTE
                MOV     R2,A
                LCALL   R_BYTE
                MOV     R3,A
                LCALL   ASCI2HEX                ;CONVERT ASCII TO HEX, O/P IN A
                XCH     A,R7
                ADD     A,R7                    ;ADD CHK SUM
                XCH     A,R7
                RET
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;RECEIVE DATA 1 BYTE
;O/P            A
R_BYTE:         JNB     RI,$                ;WAIT FOR RECEIVE
                CLR     RI
                MOV     A,SBUF
                RET

;CONVERT ASCII TO HEX
;I/P            R2,R3
;O/P            A
ASCI2HEX:       MOV     A,R2
                LCALL   ATOHS
                SWAP    A
                MOV     R2,A
                MOV     A,R3
                LCALL   ATOHS
                ORL     A,R2
                RET

ATOHS:          CJNE    A,#'A',$+3
                JC      ATOHS1
                ADD     A,#9
ATOHS1:         ANL     A,#0FH
                RET

;SEND ASCII 2 BYTE CONVERT FROM HEX 1 BYTE
; AND ADD TO CHK SUM
;I/P            A
SNUM_CHK:       XCH     A,R7
                ADD     A,R7                ;ADD TO CHK SUM
                XCH     A,R7
                LCALL   HEX2ASCI            ;CONVERT HEX TO ASCII, O/P = R2,R3
                MOV     A,R2
                LCALL   S_BYTE
                MOV     A,R3
                LCALL   S_BYTE
                RET

;SEND 1 BYTE
;I/P            A
S_BYTE:
                JNB     TI,$                ;WAIT FOR SEND
                CLR     TI
                MOV     SBUF,A
                RET

;CONVERT HEX 1 BYTE TO ASCII 2 BYTE
;I/P            A
;O/P            R2,R3
HEX2ASCI:       PUSH    ACC
                SWAP    A
                LCALL   HTOAS
                MOV     R2,A
                POP     ACC
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                LCALL   HTOAS
                MOV     R3,A
                RET

HTOAS:          ANL     A,#0FH
                CJNE    A,#0AH,$+3
                JNC     HTOAS1
                ORL     A,#30H
                RET
HTOAS1:         SUBB    A,#9
                ORL     A,#40H
                RET

END
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ข.

โปรแกรมที่ใช้เพื่อติดต่อสื่อสารผ่านทางพอร์ตอนุกรมนี้ ได้พัฒนาโดยการใช้ภาษา C++ ซึ่งใช้ Compiler ของ Turbo C++ Version 3.0 ของบริษัท Borland และได้แยกไฟล์ออกตามหน้าที่การทำงานได้ 7 ไฟล์ซึ่งแต่ละไฟล์มีหน้าที่ดังนี้

1. ZHEADER.H เป็นไฟล์ที่เก็บส่วนของการประกาศค่าต่าง ๆ เช่นค่าของการกำหนดคีย์ของโปรแกรม เป็นต้น
2. EZ.CPP เป็นไฟล์ที่เป็นส่วนของเมนูหลักของการทำงานทั้งหมด ซึ่งการ Compile และ Link เพื่อเป็นไฟล์ใช้งานจะทำผ่านไฟล์นี้ และใช้ชื่อไฟล์นี้เป็นชื่อโปรแกรมที่ใช้ Execute ด้วย
3. ZBLOCK.CPP เป็นไฟล์ที่เก็บคำสั่งของการทำงานในโหมดที่ไม่มีการรอการส่งค่าตรวจสอบความผิดพลาดกลับมาให้เครื่องคอมพิวเตอร์
4. ZERROR.CPP เป็นไฟล์ที่เก็บคำสั่งของการทำงานในโหมดที่รอรับการส่งค่าตรวจสอบความผิดพลาดจากบอร์ดมาให้กับเครื่องคอมพิวเตอร์
5. ZBUFFER.CPP เป็นไฟล์ที่เก็บคำสั่งในการติดต่อสื่อสารผ่านพอร์ตอนุกรม
6. ZFILE.CPP เป็นไฟล์ที่จัดการเกี่ยวกับไฟล์ต่าง ๆ ที่เรียกขึ้นมาเพื่อส่งให้กับบอร์ด
7. ZSCREEN.CPP เป็นไฟล์ที่จัดการเกี่ยวกับหน้าจอที่ใช้ติดต่อกับผู้ใช้

ซึ่งแต่ละไฟล์จะมี Source Code ดังต่อไปนี้

```
//////////////////////////////////////////
//    WIRELESS EPROM EMULATOR           //
//  SOFTWARE SUPPORT : EZ.EXE           //
//  USING TURBO C++ VERSION 3.0         //
//  THIS FILE IS EZ.CPP (MAIN ROUTINE)  //
/////////////// by ////////////////////////

#include<iostream.h>
#include<bios.h>
#include<math.h>
#include<process.h>
#include"zbuffer.cpp"
#include"zheader.h"
#include"screen.cpp"
#include"zfile.cpp"
#include"zblock.cpp"
#include"zerror.cpp"

int stack[16][1024];  // Global tempolary
void main()
  {
  int ch,redraw,no,error,mode;
  int i,j;
  Block a;              //
  ChkError c;           //   Class Declaration
  Buffer forset;        //
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
File b;                 //
drawscreen();
redraw=1;
forset.port=0;forset.code=224;
forset.xbaud=224;forset.xparity=0;
forset.xstop=0;forset.xlength=3;
mode=0;          // Not check error mode
forset.code=forset.xbaud+forset.xpartiy+forset.xstop+
          forset.xlength;
// Initial default communication
init_port(forset.port,forset.code);
for(i=0;i<16;i++)
  for(j=0;j<1024;j++)
    stack[i][j] = 0x0ff;
do
  {
  _setcursortype(_NORMALCURSOR);
  if(redraw==1)
    {
    cls();cmdscreen();
    forset.printstatus(mode);
    textcolor(LIGHTGREEN);
    textbackground(BLUE);
    gotoxy(1,24);cprintf("COMMAND :");gotoxy(11,24);
    }
  ch=bioskey(0);
    switch(ch)      // Main menu
    {
    case _ENTER   : cls();
                  error=init_port2(forset.port,forset.code);
                  if(error&128)
                    {
                    //If initial FAIL
                    portfail(error);getch();
                    break;
                    }
                  //Else Success,Follow by Mode
                  if(mode==0)a.enter(forset.port);
                  else c.enter(forset.port);
                  redraw=1;
                  break;
    case _FILL    : cls();
                  error=init_port2(forset.port,forset.code);
                  if(error&128)
                    {
                    portfail(error);getch();
                    break;
                    }
                  if(mode==0)a.fill(forset.port);
                  else c.fill(forset.port);
                  redraw=1;
                  break;
    case _MOVE    :
                  error=init_port2(forset.port,forset.code);
                  if(error&128)
                    {
                    portfail(error);getch();
                    break;
                    }
                  if(mode==0)
                    {
                    dosmode();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                getch();
                }
              else c.move();
              redraw=1;
              cls();
              break;
case _SEARCH  :
              error=init_port2(forset.port,forset.code);
              if(error&128)
                {
                portfail(error);getch();
                break;
                }
              if(mode==0)
                {
                dosmode();
                getch();
                }
              else c.search();
              redraw=1;
              cls();
              break;
case _COMPARE :
              error=init_port2(forset.port,forset.code);
              if(error&128)
                {
                portfail(error);getch();
                break;
                }
              if(mode==0)
                {
                dosmode();
                getch();
                }
              else c.compare();
              redraw=1;
              cls();
              break;
case _LISTF   : cls();
              redraw=1;
              b.getname();
              no=b.readfile();
              if(no==0)
                b.listfile();
              break;
case  SETTING : cls();
              forset.setport();
              error=init_port2(forset.port,forset.code);
              if(error&128)
                {
                portfail(error);getch();
                break;
                }
              break;
case  _DUMP    : //cls();
              error=init_port2(forset.port,forset.code);
              if(error&128)
                {
                portfail(error);getch();
                break;
                }
              if(mode==0)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    {
                    dosmode();
                    getch();
                    }
                  else
                    {
                    cls();
                    c.dump(forset.port);
                    }
                  redraw=1;
                  cls();
                  break;
    case  PAGEDOWN: cls();
                  b.getname();
                  error=init_port2(forset.port,forset.code);
                  if(error&128)
                    {
                    portfail(error);getch();
                    break;
                    }
                  no=b.ReadFile(forset.port);
                  redraw=1;
                  break;
    case F5           : switch(mode)
                    {
                    case 0:mode=1;break;
                    case 1:mode=0;break;
                    }
                  redraw=1;
                  break;
    default           : redraw=0;
    };
  }while(ch!=EXIT);
goodbye();
clrscr();
cout<<"Thank you...";
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
////////////////////////////////////
//   ZBLOCK.CPP (COMMAND ROUTINE)  //
////////////////////////////////////
#include<iostream.h>
#include<bios.h>
#include<math.h>
#include"zheader.h"

//////// ---- CLASS DECLARATION
class Block:public Buffer  //-- INHERITANCE FROM BUFFER
  {
  friend Buffer;  //--- SHARE VARIABLE WITH  BUFFER
  private:
    int start,end,i,data;
    int counts,counte,countt,countd,sum;
    int pgx;
    char startaddr[5],endaddr[5],targetaddr[5];
    char bdata[2],victim[4],sdata[20];
    char hunter[3000],prey[3000];
  public:
    void enter(int);
    void fill(int);
    void search();
    void dump();
    void move();
    void compare();
    void getaddr1();
    void getaddr2();
    void gettarget();
    void getdataSet(int);
    void legaladdr();
    void legaldata();
    void progress(unsigned long int);
    void pgbar();
    void addSum(char,char);
    int  sendSum(int);
    int  checkSum(int,char,char);
    void getdata(int);
    int  hex2dec(char,char,char,char,int);
    int  hex2dec2(int,char);
    void int2hex(int);
    void convert(int);
  };

//////// FILL COMMAND ////////////
void Block::fill(int port)
  {
  int error=0,r_err=0,count=0,remain;
  int countline=0,pg=0,temp;
  textcolor(LIGHTGREEN);textbackground(BLUE);
  gotoxy(20,4);cprintf("FILL DATA   :Dos mode");
  getaddr2();
  range=end-start+1;
  remain=range;
  if(range<0)
    {
    fail("Address Range Error!",20);
    r_err=1;
    }
  else
    {
    getdata(2);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
legaladdr();
sum = 0;                // reset Sum
count=0;
pgbar();
do
  {
  if(count==0)
  {
  sum=0;
  error=transmit(port,':'); //--START COMMUNICATE
  if(error&128)goto out1;
  if(countline>0)
    remain=remain-16;
  if(remain<16)     // CHECK FOR BYTE REMAIN
    {
    int2hex(remain);
    error=transmit(port,victim[1]); // Byte counter
    if(error&128)goto out1;
    error=transmit(port,victim[0]);
    if(error&128)goto out1;
    addSum(victim[1],victim[0]);
    }
  else
    {
    error=transmit(port,'1'); // Send Byte counter
    if(error&128)goto out1;
    error=transmit(port,'0');
    if(error&128)goto out1;
    addSum('1','0');
    }
  countline++;
  int2hex(start+pg);

  error=transmit(port,victim[3]);  // Send Address
  if(error&128)goto out1;
  error=transmit(port,victim[2]);
  error=transmit(port,victim[1]);
  if(error&128)goto out1;
  error=transmit(port,victim[0]);
  if(error&128)goto out1;
  addSum(victim[3],victim[2]);
  if(error&128)goto out1;
  addSum(victim[1],victim[0]);
  if(error&128)goto out1;
  error=transmit(port,'0');
  if(error&128)goto out1;
  error=transmit(port,'0');    // Send Type
  if(error&128)goto out1;
  addSum('0','0');
  }
  error=transmit(port,bdata[0]);
  if(error&128)goto out1;
  error=transmit(port,bdata[1]);  // send data
  if(error&128)goto out1;
  addSum(bdata[0],bdata[1]);
  progress(pg);

  count++;pg++;
  if(count==16)
  {
  // Send Check Sum
  error=sendSum(sum);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        if(error&128)goto out1;
        count=0;
        // Send Lind Feed and Carriage Return
        error=transmit(port,0x0D);
        if(error&128)goto out1;
        error=transmit(port,0x0A);
        if(error&128)goto out1;

        }
        }while(pg<range);

      error=sendSum(sum);
      error=transmit(port,0x0D);
      error=transmit(port,0x0A);
      if(error&128)goto out1;
      else
        {
        // --  end of Transmit :00 0000 01 FF
        error=transmit(port,':');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;

        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;

        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'1');if(error&128)goto out1;

        error=transmit(port,'F');if(error&128)goto out1;
        error=transmit(port,'F');if(error&128)goto out1;
        error=transmit(port,0x0D);if(error&128)goto out1;
        error=transmit(port,0x0A);if(error&128)goto out1;
        progress(range);
        complete("FILL Complete",13);
        }
      goto out2;
      }
      if(!r_err==1)
        {
        out1:
        fail("Communication Error",19);
        }
      out2:getch();
    }

  /////// Enter Command /////////
  void Block::enter(int port)
    {
    int error=0,i;
    textcolor(LIGHTGREEN);textbackground(BLUE);
    gotoxy(20,4);cprintf("ENTER DATA");
    getaddr1();
    getdata(1);
    if(error==0)
      {
      legaladdr();
      sum = 0;                  // reset Sum
      error=transmit(port,':');
      if(error&128)goto out1;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    error=transmit(port,'0'); // Byte counter = 1
    if(error&128)goto out1;
    error=transmit(port,'1');
    if(error&128)goto out1;
    addSum('0','1');

    for(i=0;i<=3;i++)      // Send address
      error=transmit(port,startaddr[i]);
    addSum(startaddr[0],startaddr[1]);
    addSum(startaddr[2],startaddr[3]);

    error=transmit(port,'0');
    error=transmit(port,'0');    // Type
    addSum('0','0');

    error=transmit(port,bdata[0]);
    error=transmit(port,bdata[1]);  // send data
    addSum(bdata[0],bdata[1]);
    // Send Check Sum to Port
    error=sendSum(sum);

    // Send end of transmit
    error=transmit(port,0x0D);
    error=transmit(port,0x0A);
    error=transmit(port,':');
    error=transmit(port,'0');
    error=transmit(port,'0');

    error=transmit(port,'0');
    error=transmit(port,'0');
    error=transmit(port,'0');
    error=transmit(port,'0');

    error=transmit(port,'0');
    error=transmit(port,'1');

    error=transmit(port,'F');
    error=transmit(port,'F');
    error=transmit(port,0x0D);
    error=transmit(port,0x0A);

    complete("ENTER Complete",14);
    goto out2;
    }
    out1:
      fail("Communication Error",18);

    out2:getch();
  }

////// Search Command //////
void Block::search()
  {
  int r_err=0,error=0;
  int walk=0;
  textcolor(LIGHTGREEN);textbackground(BLUE);
  gotoxy(20,4);cprintf("SEARCH DATA");
  getaddr2();
  range=end-start;
  if(range<0)
    {
    fail("Address Range Error!",20);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    r_err=1;
    }
  else
    {
    getdataSet(3);   // Get address
    legaladdr();
    int i=range;
    sum = 0;                // reset Sum
    error=transmit(port,':');
    if(error&128)goto out1;
    int2hex(range);

    error=transmit(port,victim[1]); // Byte counter
    if(error&128)goto out1;
    error=transmit(port,victim[0]);
    if(error&128)goto out1;
    addSum(victim[1],victim[0]);

    for(i=0;i<=3;i++)       // Send address
      {
      error=transmit(port,startaddr[i]);
      if(error&128)goto out1;
      }
    addSum(startaddr[0],startaddr[1]);
    addSum(startaddr[2],startaddr[3]);

    error=transmit(port,'0');
    if(error&128)goto out1;
    error=transmit(port,'2');    // Type
    if(error&128)goto out1;
    addSum('0','2');
    i++;   // add for last byte is CHKSUM
    for(; i>=0 ; i--)
      {
      error=receive(port);       // Receive data
      if(error&128)goto out1;
      else if(error==13)
      {
      fail("EPROM-EMU. Not Respone",22);
      getch();
      return;
      }
      else
      {
      hunter[walk]=(char)error;
      walk++;
      }
      error=receive(port);
      if(error&128)goto out1;
      else if(error==13)
      {
      fail("EPROM-EMU. Not Respone",22);
      getch();
      return;
      }
      else
      {
      hunter[walk]=(char)error;
      walk++;
      }
      if(i!=0)  // If not last;Add Sum
      addSum(hunter[walk-2],hunter[walk-1]);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      }
    error=checkSum(sum,hunter[walk-2],hunter[walk-1]);
    if(error&128)
      {
      fail("Checksum Error!",15);
      r_err=1;
      goto out1;
      }
//------------ BEGIN SEARCH
    i=range;
    int tempwalk=0;
    int targetwalk=0;
    int loop=0;
    walk=0;
    do
      {
      targetwalk=0;
      tempwalk=walk;
      do
      {
      // --- If head is equal then check in range
      if(sdata[targetwalk]==hunter[tempwalk])
        {
        tempwalk++;
        targetwalk++;
        loop=1;
        }
      else  // If head not equal then shift pointer
        {
        walk++;walk++;
        loop=0;
        }
      }while(sdata[targetwalk]!='\0');
      if(loop==1)
      {
      complete("FOUND!",6);
      //equal
      }

      }while(i>0);  // still range
    if(loop==0)
    complete("Not Found",13);
    goto succ;
    }

  out1:
  if(r_err!=1)
    fail("Communication Error!",18);
  succ:getch();
  }
////////////// Move Command //////////
void Block::move()
  {
  int error=0,r_err=0;
  int walk=0;
  textcolor(LIGHTGREEN);textbackground(BLUE);
  gotoxy(20,4);cprintf("MOVE DATA");
  getaddr2();
  range=end-start;
  if(range<0)
    {
    fail("Address Range Error!",20);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    r_err=1;
    }
  else
    {
    gettarget();  // get address
    legaladdr();
    int i=range;
    sum = 0;                // reset Sum
    error=transmit(port,':');
    if(error&128)goto endmove;
    int2hex(range);

    error=transmit(port,victim[1]); // Byte counter
    if(error&128)goto endmove;
    error=transmit(port,victim[0]);
    if(error&128)goto endmove;
    addSum(victim[1],victim[0]);

    for(i=0;i<=3;i++)       // Send address
      {
      error=transmit(port,startaddr[i]);
      if(error&128)goto endmove;
      }
    addSum(startaddr[0],startaddr[1]);
    addSum(startaddr[2],startaddr[3]);

    error=transmit(port,'0');
    if(error&128)goto endmove;
    error=transmit(port,'2');    // Type
    if(error&128)goto endmove;
    addSum('0','2');

    i=range;
    i++;   // add for last byte is CHKSUM
    for(; i>=0 ; i--)
      {
      error=receive(port);       // Receive data
      if(error&128)goto endmove;
      else if(error==13)
      {
      fail("EPROM-EMU. Not Respone",22);
      getch();
      return;
      }
      else
      {
      hunter[walk]=(char)error;
      walk++;
      }
      error=receive(port);
      if(error&128)goto endmove;
      else if(error==13)
      {
      fail("EPROM-EMU. Not Respone",22);
      getch();
      return;
      }
      else
      {
      hunter[walk]=(char)error;
      walk++;
      }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        if(i!=0)  // If not last;Add Sum
        addSum(hunter[walk-2],hunter[walk-1]);
        }
      error=checkSum(sum,hunter[walk-2],hunter[walk-1]);
      if(error&128)
        {
        fail("SUM Error:PC «- EPROM",21);
        r_err=1;
        goto endmove;
        }
//-------^--- Upload Source data
      else     // Checksum not error ; Send Block
        {
        sum = 0;                // reset Sum
        int error=transmit(port,':');
        if(error&128)goto endmove;
        int2hex(range);

        error=transmit(port,victim[1]); // Byte counter
        if(error&128)goto endmove;
        error=transmit(port,victim[0]);
        if(error&128)goto endmove;
        addSum(victim[1],victim[0]);

        for(i=0;i<=3;i++)       // Send address
        {
        error=transmit(port,targetaddr[i]);
        if(error&128)goto endmove;
        }
        addSum(targetaddr[0],targetaddr[1]);
        addSum(targetaddr[2],targetaddr[3]);

        error=transmit(port,'0');
        if(error&128)goto endmove;
        error=transmit(port,'3');    // Type
        if(error&128)goto endmove;
        addSum('0','3');

        for(i=0;i<=range;i++)
        {
        error=transmit(port,hunter[i]);
        if(error&128)goto endmove;
        i++;
        error=transmit(port,hunter[i]);
        if(error&128)goto endmove;
        addSum(hunter[i-1],hunter[i]);
        }
        error=sendSum(sum);
        if(error&128)
        {
        fail("SUM Error:PC -» EPROM",21);
        goto endmove2;
        }
        }
      }
    endmove:if((error&128)&&(r_err==0))
      fail("Communication Error",19);
    endmove2:getch();
    }

//------- Dump ......
void Block::dump()
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
int error=0,r_err=0,y;
int walk=0,countbyte,addrwalk;
textcolor(LIGHTGREEN);textbackground(BLUE);
gotoxy(20,4);cprintf("DUMP DATA");
getaddr2();
range=end-start;
if(range<0)
  {
  fail("Address Range Error!",20);
  r_err=1;
  }
else
  {
  legaladdr();
  int i=range;
  sum = 0;                  // reset Sum
  error=transmit(port,':');
  if(error&128)goto enddump;
  int2hex(range);

  error=transmit(port,victim[1]); // Byte counter
  if(error&128)goto enddump;
  error=transmit(port,victim[0]);
  if(error&128)goto enddump;
  addSum(victim[1],victim[0]);

  for(i=0;i<=3;i++)       // Send address
    {
    error=transmit(port,startaddr[i]);
    if(error&128)goto enddump;
    }
  addSum(startaddr[0],startaddr[1]);
  addSum(startaddr[2],startaddr[3]);

  error=transmit(port,'0');
  if(error&128)goto enddump;
  error=transmit(port,'2');     // Type
  if(error&128)goto enddump;
  addSum('0','2');

  i=range;
  i++;   // add for last byte is CHKSUM

  for(; i>=0 ; i--)
    {
    error=receive(port);        // Receive data
    if(error&128)goto enddump;
    else if(error==13)
    {
    fail("EPROM-EMU. Not Respone",22);
    getch();
    return;
    }
    else
    {
    hunter[walk]=(char)error;
    walk++;
    }
    error=receive(port);
    if(error&128)goto enddump;
    else if(error==13)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        {
        fail("EPROM-EMU. Not Respone",22);
        getch();
        return;
        }
        else
        {
        hunter[walk]=(char)error;
        walk++;
        }
       'if(i!=0)  // If not last;Add Sum
        addSum(hunter[walk-2],hunter[walk-1]);
        }
      error=checkSum(sum,hunter[walk-2],hunter[walk-1]);
      if(error&128)
        {
        fail("SUM Error:PC «- EPROM",21);
        r_err=1;
        goto enddump;
        }
        //-------^--- Upload Source data
      i=0;
      int2hex(start);
      addrwalk=start;
      walk=0;y=2;countbyte=0;
      do
        {
        int2hex(addrwalk);
        if(countbyte==0)
        {
        gotoxy(15,y);
        cprintf("%c%c%c%c : ",victim[3],victim[2],victim[1],victim[0]);
        }
        if(countbyte==8)
        cprintf("%c%c - ",hunter[walk],hunter[walk+1]);
        else
        cprintf("%c%c ",hunter[walk],hunter[walk+1]);
        if(countbyte==16)
        {y++;countbyte=0;}
        else countbyte++;
        if(y>23)
        {
        cout<<"Press Anykey";getch();
        cls();
        y=2;
        }
        walk+=2;i++;
        addrwalk++;


        }while(i<range);
      }

    enddump:if(r_err==0)fail("Communication Error",19);
    enddump2:getch();
    }

  void Block::compare()
    {
  //  int r_err=0;
    textcolor(LIGHTGREEN);textbackground(BLUE);
    gotoxy(20,4);cprintf("COMPARE DATA");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  getaddr2();
  range=end-start;
  if(range<0)
    {
    fail("Address Range Error!",20);
//    r_err=1;
    }
  else
    {
    gettarget();
    pgx=0;
    range=end-start;
    pgbar();
    for(int i=0;i<=range;i++)
    progress(i);
    }
  getch();
  }

void Block::convert(int type)
      {
      //   Type 0 for Address start
      //        1 for Address end
      //        2 for Taget Address
      //        3 for Data
      if(type==0)
      start = hex2dec(startaddr[0],startaddr[1],
            startaddr[2],startaddr[3],counts);
      else if(type==1)
      end   = hex2dec(endaddr[0],endaddr[1],
            endaddr[2],endaddr[3],counte);
      else if(type==2)
      range = end-start;
      }



int Block::hex2dec2(int bite,char ch)
  {
  int number;
  switch(ch)
    {
    case '0' : number = 0; break;
    case '1' : number = 1*(pow(16,bite)); break;
    case '2' : number = 2*(pow(16,bite)); break;
    case '3' : number = 3*(pow(16,bite)); break;
    case '4' : number = 4*(pow(16,bite)); break;
    case '5' : number = 5*(pow(16,bite)); break;
    case '6' : number = 6*(pow(16,bite)); break;
    case '7' : number = 7*(pow(16,bite)); break;
    case '8' : number = 8*(pow(16,bite)); break;
    case '9' : number = 9*(pow(16,bite)); break;
    case 'a' : number = 10*(pow(16,bite)); break;
    case 'b' : number = 11*(pow(16,bite)); break;
    case 'c' : number = 12*(pow(16,bite)); break;
    case 'd' : number = 13*(pow(16,bite)); break;
    case 'e' : number = 14*(pow(16,bite)); break;
    case 'f' : number = 15*(pow(16,bite)); break;
    case 'A' : number = 10*(pow(16,bite)); break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      case 'B' : number = 11*(pow(16,bite)); break;
      case 'C' : number = 12*(pow(16,bite)); break;
      case 'D' : number = 13*(pow(16,bite)); break;
      case 'E' : number = 14*(pow(16,bite)); break;
      case 'F' : number = 15*(pow(16,bite)); break;
      default  : number = 0; break;
      }
    return number;
    }

int Block::hex2dec(char c3,char c2,char c1,char c0,int count)
    {
    int t0,t1,t2,t3,num=0;

    if(count==4)
      {
      t0=hex2dec2(0,c0);
      t1=hex2dec2(1,c1);
      t2=hex2dec2(2,c2);
      t3=hex2dec2(3,c3);
      num=t0+t1+t2+t3;
      }
    else if(count==3)
      {
      t0=hex2dec2(0,c1);
      t1=hex2dec2(1,c2);
      t2=hex2dec2(2,c3);
      num=t0+t1+t2;
      }
    else if(count==2)
      {
      t0=hex2dec2(0,c2);
      t1=hex2dec2(1,c3);
      num=t0+t1;
      }
    else if(count==1)
      {
      t0=hex2dec2(0,c3);
      num=t0;
      }
    return num;
    }

void Block::legaladdr()
    {
    if(counts==3)                  // Legal Start address
      {
      startaddr[4]='\0';
      startaddr[3]=startaddr[2];
      startaddr[2]=startaddr[1];
      startaddr[1]=startaddr[0];
      startaddr[0]='0';
      }
    else if(counts==2)
      {
      startaddr[4]='\0';
      startaddr[3]=startaddr[1];
      startaddr[2]=startaddr[0];
      startaddr[1]=startaddr[0]='0';
      }
    else if(counts==1)
      {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    startaddr[4]='\0';
    startaddr[3]=startaddr[0];
    startaddr[2]=startaddr[1]=startaddr[0]='0';
    };
  if(counte==3)                    // Legal end address
    {
    endaddr[4]='\0';
    endaddr[3]=endaddr[2];
    endaddr[2]=endaddr[1];
    endaddr[1]=endaddr[0];
    endaddr[0]='0';
    }
  else if(counte==2)
    {
    endaddr[4]='\0';
    endaddr[3]=endaddr[1];
    endaddr[2]=endaddr[0];
    endaddr[1]=endaddr[0]='0';
    }
  else if(counte==1)
    {
    endaddr[4]='\0';
    endaddr[3]=endaddr[0];
    endaddr[2]=endaddr[1]=endaddr[0]='0';
    };
  if(countt==3)                    // Legal target address
    {
    targetaddr[4]='\0';
    targetaddr[3]=targetaddr[2];
    targetaddr[2]=targetaddr[1];
    targetaddr[1]=targetaddr[0];
    targetaddr[0]='0';
    }


  else if(countt==2)
    {
    targetaddr[4]='\0';
    targetaddr[3]=targetaddr[1];
    targetaddr[2]=targetaddr[0];
    targetaddr[1]=targetaddr[0]='0';
    }
  else if(countt==1)
    {
    targetaddr[4]='\0';
    targetaddr[3]=targetaddr[0];
    targetaddr[2]=targetaddr[1]=targetaddr[0]='0';
    }
  }
void Block::legaldata()
  {
  int i=0,j;

  do
    {
    if(sdata[i]==' ')
      {
      j=i;
      do
        j++;
      while(sdata[j]==' ');
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      sdata[i]=sdata[j];
      sdata[j]=' ';
      }
    i++;
    }while(sdata[i-1]!='\0');
  countt=i;
  }

void Block::getaddr2()
  {
  int x=35,y=6,para=0;
  int ch;
  textbackground(BLUE);
  textcolor(WHITE);
  gotoxy(20,6);  cprintf("Address Start: ");

  textbackground(CYAN);
  gotoxy(35,6);  cprintf("    ");

  _setcursortype(_NORMALCURSOR);
  i=0;
  do{
    gotoxy(x+i,y);
    ch = bioskey(0);
    if((((char)ch >= '0')&&((char)ch <= '9'))||
       (((char)ch >= 'a')&&((char)ch <= 'f'))||
       (((char)ch >= 'A')&&((char)ch <= 'F')))
       {                          // if HEX number
       if(i<4)
       {
       cprintf("%c",(char)ch);
       switch(para)
         {
         case 0:startaddr[i]=(char)ch;break;
         case 1:endaddr[i]=(char)ch;break;
         }
       i++;
       }
       }
    else if(ch==BKSP)
      {
      gotoxy(x+i-1,y);cprintf(" ");
      if(i>0)
      i--;
      if(x>35)
      x--;
      }
    else if(ch==ENTER)
      {
      switch(para)
      {
      case 0:startaddr[i]='\0';counts=i;break;
      case 1:endaddr[i]='\0';counte=i;break;
      }
      para++;
      y+=2;
      x=35;
      i=0;
      textbackground(BLUE);
      switch(para)
      {
      case 1:gotoxy(20,8);cprintf("Address End  : ");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                textbackground(CYAN);
                gotoxy(35,8);  cprintf("    ");break;
          case 2:goto Out;
          }
          }
      }while(ch!=ESCAPE);
    Out:;
    if(para<2)
      return;
    else
      {
      convert(0);
      convert(1);
      }

    }

  void Block::getaddr1()
    {
    int x=35;
    int ch;
    textbackground(BLUE);
    textcolor(WHITE);
    gotoxy(26,6);  cprintf("Address: ");

    textbackground(CYAN);
    gotoxy(35,6);  cprintf("    ");

    _setcursortype(_NORMALCURSOR);
    i=0;
    do{
      gotoxy(x+i,6);
      ch = bioskey(0);
      if((((char)ch >= '0')&&((char)ch <= '9'))||
         (((char)ch >= 'a')&&((char)ch <= 'f'))||
         (((char)ch >= 'A')&&((char)ch <= 'F')))
         {
         if(i<4)
         {                          // if HEX number
         cprintf("%c",(char)ch);
         startaddr[i]=(char)ch;
         i++;
         }
         }
      else if(ch==BKSP)
        {
        gotoxy(x+i-1,6);cprintf(" ");
        if(i>0)
        i--;
        if(x>35)
        x--;
        }
      else if(ch==ENTER)
        {
        startaddr[i]='\0';counts=i;
        x=35;
        i=0;
        textbackground(BLUE);
        goto Out;
        }
      }while(ch!=ESCAPE);
    Out:;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  if(ch==ENTER)
    convert(0);
  }


void Block::gettarget()
  {
  int x=35;
  int ch;
  textbackground(BLUE);
  textcolor(WHITE);
  gotoxy(20,10);  cprintf("Target Address:");

  textbackground(CYAN);
  gotoxy(35,10);  cprintf("    ");

  _setcursortype(_NORMALCURSOR);
  i=0;
  do{
    gotoxy(x+i,10);
    ch = bioskey(0);
    if((((char)ch >= '0')&&((char)ch <= '9'))||
       (((char)ch >= 'a')&&((char)ch <= 'f'))||
       (((char)ch >= 'A')&&((char)ch <= 'F')))
       {
       if(i<4)
       {                          // if HEX number
       cprintf("%c",(char)ch);
       targetaddr[i]=(char)ch;
       i++;
       }
       }
    else if(ch==BKSP)
      {
      gotoxy(x+i-1,10);cprintf(" ");
      if(i>0)
      i--;
      if(x>35)
      x--;
      }
    else if(ch==ENTER)
      {
      targetaddr[i]='\0';
      countt=i;
      goto out;
      }
    }while(ch!=ESCAPE);
  out:
  if(ch==ENTER)
    convert(3);
  }

void Block::getdata(int rr)
  {
  int x=35,y;
  int ch;
  switch(rr)
    {
    case 1:y=8;break;
    case 2:y=10;break;
    case 3:y=12;break;
    };
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  textbackground(BLUE);
  textcolor(WHITE);
  gotoxy(26,y);  cprintf("Data    : ");

  textbackground(CYAN);
  gotoxy(35,y);  cprintf("  ");

  _setcursortype(_NORMALCURSOR);
  i=0;

  do{
    gotoxy(x+i,y);
    ch = bioskey(0);
    if((((char)ch >= '0')&&((char)ch <= '9'))||
       (((char)ch >= 'a')&&((char)ch <= 'f'))||
       (((char)ch >= 'A')&&((char)ch <= 'F')))
       {                          // if HEX number
       if(i<2)
       {
       cprintf("%c",(char)ch);
       bdata[i]=(char)ch;
       i++;
       }
       }
    else if(ch==BKSP)
      {
      gotoxy(x+i-1,y);cprintf(" ");
      if(i>0)
      i--;
      if(x>35)
      x--;
      }
    else if(ch==ENTER)
      {
      bdata[i]='\0';countd=i;
      x=35;
      i=0;
      textbackground(BLUE);
      goto Out;
      }
    }while(ch!=ESCAPE);
  Out:;
  }

void Block::getdataSet(int rr)
  {
  int x=35,y;
  int ch;
  switch(rr)
    {
    case 1:y=8;break;
    case 2:y=10;break;
    case 3:y=12;break;
    };
  textbackground(BLUE);
  textcolor(WHITE);
  gotoxy(26,y);  cprintf("Data    : ");

  textbackground(CYAN);
  gotoxy(35,y);  cprintf("                    ");

  _setcursortype(_NORMALCURSOR);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  i=0;

  do{
    gotoxy(x+i,y);
    ch = bioskey(0);
    if((((char)ch >= '0')&&((char)ch <= '9'))||
       (((char)ch >= 'a')&&((char)ch <= 'f'))||
       (((char)ch >= 'A')&&((char)ch <= 'F'))||
       ((char)ch==' '))
       {                          // if HEX number
       if(i<20)
       {
       if((sdata[i-1]==' ')&&((char)ch==' '))
         goto xtemp;
       cprintf("%c",(char)ch);
       sdata[i]=(char)ch;
       i++;
       }
       }
    else if(ch==BKSP)
      {
      gotoxy(x+i-1,y);cprintf(" ");
      if(i>0)
      i--;
      if(x>35)
      x--;
      }
    else if(ch==ENTER)
      {
      sdata[i]='\0';countd=i;
      x=35;
      i=0;
      textbackground(BLUE);
      legaldata();
      goto Out;
      }
    xtemp:
    }while(ch!=ESCAPE);
  Out:;
  }

void Block::int2hex(int val)
{
//   extern char *_addr[4];
   int temp,i;
   for(i=3;i>=0;i--)
   {
       temp = val / ((int)pow(16,i));
       switch(temp)
       {
       case 0  : victim[i]='0';break;
       case 1  : victim[i]='1';break;
       case 2  : victim[i]='2';break;
       case 3  : victim[i]='3';break;
       case 4  : victim[i]='4';break;
       case 5  : victim[i]='5';break;
       case 6  : victim[i]='6';break;
       case 7  : victim[i]='7';break;
       case 8  : victim[i]='8';break;
       case 9  : victim[i]='9';break;
       case 10 : victim[i]='A';break;
       case 11 : victim[i]='B';break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        case 12 : victim[i]='C';break;
        case 13 : victim[i]='D';break;
        case 14 : victim[i]='E';break;
        case 15 : victim[i]='F';break;
        }
        val = val % ((int)pow(16,i));
    }
}

void Block::addSum(char high,char low)
  {
  int thigh,tlow;

  thigh=hex2dec2(1,high);
  tlow =hex2dec2(0,low);
  sum = sum+thigh+tlow;
  }

int Block::sendSum(int sum)
  {
  unsigned int temp;
  int ret;
  temp = ((~sum)&0x00ff) + 1;     // 2' complement
  int2hex(temp);
  ret=transmit(port,victim[1]);  // high
    if(ret&128) return ret;
  ret=transmit(port,victim[0]);  // low
    return ret;
  }
int Block::checkSum(int sum,char high,char low)
  {
  unsigned int temp;
  int ret;
  temp = ((~sum)&0x00ff) + 1;    // 2'
  int2hex(temp);
  if(victim[1]!=high)
    return 1;
  if(victim[0]!=low)
    return 1;
  else return 0;
  }
void Block::progress(unsigned long int xx)
  {
  int i;
  if(range>0)
  _setcursortype(_NOCURSOR);
  xx=((xx*100)/range);
  gotoxy(20,18);
  for(i=0;i<=(40*xx)/100;i++)
    {
    textcolor(YELLOW);
    cprintf("");
    }
  textcolor(WHITE);textbackground(BLUE);
  gotoxy(38,17);cprintf("%d",xx);cprintf("%");
  delay(20);

  }
void Block::pgbar()
  {
  gotoxy(20,18);
  textcolor(CYAN);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
textbackground(BLUE);
cprintf("                                                                    ");
}
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//////////////////////////////////////
//   Zbuffer.cpp (Communication routine)
//////////////////////////////////////
#include<dos.h>
#include<conio.h>
#include<bios.h>
#include<list.h>
#include<queue.h>
#include"zheader.h"

class Buffer   /// class decalration
  {
  protected:
    int range;
  public:
    int xstop,xparity,xbaud,xlength;
    int port,code;
    void upload();
    void download();
    void setport();
    void printstatus(int);
  };

int transmit(int,char);
int receive(int);
int check_state(int);
void init_port(int,unsigned char);
int init_port2(int,unsigned char);

void init_port(int port,unsigned char code)
{
  union REGS r;
  r.x.dx = port;
  r.h.ah = 0;           // function 0 initial serial port
  r.h.al = code;
  int86(0x14,&r,&r);
}

int init_port2(int port,unsigned char code)
{

  union REGS r;
  code=0x0e3;
  r.x.dx = port;
  r.h.ah = 0;           // function 0 initial serial port
  r.h.al = code;
  int86(0x14,&r,&r);
  return r.h.ah;
}



int transmit(int port,char c)
{
  union REGS r;
  r.x.dx = port;
  r.h.al = c;
  r.h.ah = 1;           //function 01 Transmit data to port
  int86(0x14,&r,&r);
  return r.h.ah;        //return 0 if success
                //       1 if fail
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
int receive(int port)
{
  int i=0;
  union REGS r;

/*  asm{
    mov al,0
    mov dx,port
    mov ah,2
    int 14h
    }*/

  r.x.ax = (r.x.ax & 0);      // clear register
  while( ! (check_state(port)&256))  // loop if not return 0
    if(!kbhit())
      {
      i++;//cprintf("-");
      if(i>2000)
      return 13;  // No respone from EPROM
      }
  r.h.al = 55;
  r.x.dx = port;
  r.h.ah = 2;
  int86(0x14,&r,&r);
  if(!(r.h.ah & 128))     // if receive correct
    {
    cprintf("%c",r.h.al);
    return r.h.al;
    }
  else                    // if incorrect
  return 13;
}

int check_state(int port)
{
  union REGS r;
  r.x.ax = r.x.ax & 0 ; // clear
  r.x.dx = port;
  r.h.ah = 3;           // function 3 read ready status from port
  int86(0x14,&r,&r);
  return r.x.ax;
     //  0    - data ready
     //  1..7 - error
}


void Buffer::setport(void)
{
//  extern int cando;
  struct {
    int max,value,walk;
    }s[5];
  int kb,y,i;//,xport,xbaud,xparity,xstop;
  i=0;y=4;
  s[0].max = 1;s[1].max=3;s[2].max=2;s[3].max=1;s[4].max=1;
  s[0].walk = s[1].walk = s[2].walk = s[3].walk= s[4].walk=0;

//  window(14,1,80,13);
  textcolor(YELLOW);
  textbackground(BLUE);
  gotoxy(42,16); cprintf("Cursor Move : Up,Down");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
gotoxy(42,17); cprintf("Change      : PgUp,PgDn");

textcolor(YELLOW);
textbackground(BLUE);
gotoxy(16,2); cprintf("          SET COMMUNICATION PORT");
textcolor(WHITE);
gotoxy(34,4);  cprintf("Port          : ");
gotoxy(34,6);  cprintf("Baud Rate     : ");
gotoxy(34,8);  cprintf("Parity Bit    : ");
gotoxy(34,10); cprintf("Data Length   : ");
gotoxy(34,12); cprintf("Stop Bit      : ");
textbackground(CYAN);
gotoxy(49,4);  cprintf("    ");
gotoxy(49,6);  cprintf("    ");
gotoxy(49,8);  cprintf("    ");
gotoxy(49,10); cprintf(" ");
gotoxy(49,12); cprintf(" ");
_setcursortype(_NORMALCURSOR);
do
{
  textcolor(WHITE);
  gotoxy(49,4);
  switch(s[0].walk){
    case 0 : cprintf("COM1");port=0;break;
    case 1 : cprintf("COM2");port=1;break;
  }
  gotoxy(49,6);
  switch(s[1].walk){
    case 0 : cprintf("9600");xbaud=224;break;
    case 1 : cprintf("4800");xbaud=192;break;
    case 2 : cprintf("2400");xbaud=160;break;
    case 3 : cprintf("1200");xbaud=128;break;
  }
  gotoxy(49,8);
  switch(s[2].walk){
    case 0 : cprintf("NONE");xparity=0;break;
    case 1 : cprintf(" ODD");xparity=8;break;
    case 2 : cprintf("EVEN");xparity=24;break;
  }
  gotoxy(49,10);
  switch(s[3].walk){
    case 0 : cprintf("8");xlength=3;break;
    case 1 : cprintf("7");xlength=2;break;
  }
  gotoxy(49,12);
  switch(s[4].walk){
    case 0 : cprintf("1");xstop=0;break;
    case 1 : cprintf("2");xstop=4;break;
  }

  gotoxy(49,y);
  kb = bioskey(0);          //scan keyboard
  switch(kb){
    case UP       : y-=2;i--;
                if(i<0)
                {
                y = 12;
                i = 4;
                }
                break;
    case DOWN     : y+=2;i++;
                if(i>4)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                    {
                    y = 4;
                    i = 0;
                    }
                    break;
        case PAGEUP   : s[i].walk++;
                    if(s[i].walk > s[i].max)
                    s[i].walk = 0;
                    break;
        case PAGEDOWN : s[i].walk--;
                    if(s[i].walk < 0)
                    s[i].walk = s[i].max;
                    break;
        case ESCAPE   : goto out;
      } // end switch

    }while(kb != ENTER);   // end do
    code = xbaud + xparity + xstop + xlength;
    out:
    _setcursortype(_NOCURSOR);
}

void Buffer::printstatus(int mode)
    {
    textcolor(WHITE);textbackground(BLUE);
    gotoxy(20,20);cprintf("Config :>");
    gotoxy(20,19);cprintf("Mode   :>");
    textcolor(YELLOW);
    textbackground(BLUE);
    gotoxy(30,20);
    switch(port)
      {
      case 0: cprintf("COM1 - ");break;
      case 1: cprintf("COM2 - ");break;
      };
    switch(xbaud)
      {
      case 224:cprintf("9600 - ");break;
      case 192:cprintf("4800 - ");break;
      case 160:cprintf("2400 - ");break;
      case 128:cprintf("1200 - ");break;
      };
    switch(xparity)
      {
      case  0:cprintf("NONE - ");break;
      case  8:cprintf("ODD  - ");break;
      case 24:cprintf("EVEN - ");break;
      };
    switch(xlength)
      {
      case 2:cprintf("7 - ");break;
      case 3:cprintf("8 - ");break;
      };

    switch(xstop)
      {
      case 0:cprintf("1");break;
      case 4:cprintf("2");break;
      };

    gotoxy(30,19);
    switch(mode)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  {
  case 0:cprintf("Not Check Error #DOS copy mode");break;
  case 1:cprintf("Check Error *(Project mode)*   ");break;
  };
}
```


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//////////////////////////////////////////
//  Zerror.cpp (Command routine with check error)
//////////////////////////////////////////
#include<iostream.h>
#include<bios.h>
#include<math.h>
#include"zheader.h"


class ChkError:public Buffer
  {
  friend Buffer;
  private:
    int start,end,i,data;
    int counts,counte,countt,countd,sum;
    int pgx;
    char startaddr[5],endaddr[5],targetaddr[5];
    char bdata[2],victim[4],sdata[20];
    char hunter[3000],prey[3000];
  public:
    void enter(int);
    void fill(int);
    void search();
    void dump(int port);
    void move();
    void compare();
    void getaddr1();
    void getaddr2();
    void gettarget();
    void getdataSet(int);
    void legaladdr();
    void legaldata();
    void progress(unsigned long int);
    void pgbar();
    void addSum(char,char);
    int  sendSum(int);
    int  checkSum(int,char,char);
    void getdata(int);
    int  hex2dec(char,char,char,char,int);
    int  hex2dec2(int,char);
    void int2hex(int);
    void convert(int);
  };

void ChkError::fill(int port)
  {
  int error=0,r_err=0,count=0,remain;
  int counterror;
  int countline=0,pg=0,temp,rangetemp;
  char Z1,Z2;
  textcolor(LIGHTGREEN);textbackground(BLUE);
  gotoxy(20,4);cprintf("FILL DATA    :Check error");
  getaddr2();
  range=end-start+1;
  remain=range;
  if(range<0)
    {
    fail("Address Range Error!",20);
    r_err=1;
    }
  else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    {
    getdata(2);
    legaladdr();
    counterror=0;

    sum = 0;                // reset Sum
    count=0;                // set byte counter
    pgbar();
    do
      {
re_load:              // Correctness label
       //   cprintf("error:%d ",counterror);
      if(counterror==3)
        {
        gotoxy(1,2);textcolor(WHITE);
      printf("A=%d ",Z2);
        fail("Check Sum Error",15);
        goto out2;
        }
      ////////////////////
      if(count==0)
      {
      sum=0;// set check sum

      error=transmit(port,':');
      if(error&128)goto porterr;
      if(countline>0)
        remain=remain-16;
      if(remain<16)
        {
        int2hex(remain);
        error=transmit(port,victim[1]); // Byte counter
        if(error&128) goto porterr;
        error=transmit(port,victim[0]);
        if(error&128) goto porterr;
        addSum(victim[1],victim[0]);
        }
      else
        {
        error=transmit(port,'1'); // Byte counter
        if(error&128) goto porterr;
        error=transmit(port,'0');
        if(error&128) goto porterr;
        addSum('1','0');
        }
      countline++;
      int2hex(start+pg);

      error=transmit(port,victim[3]);  // Address
      if(error&128) goto porterr;
      error=transmit(port,victim[2]);
      if(error&128) goto porterr;
      error=transmit(port,victim[1]);
      if(error&128) goto porterr;
      error=transmit(port,victim[0]);
      if(error&128) goto porterr;

      addSum(victim[3],victim[2]);
      if(error&128) goto porterr;
      addSum(victim[1],victim[0]);
      if(error&128) goto porterr;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  error=transmit(port,'0');
  if(error&128) goto porterr;
  error=transmit(port,'3');    // Type
  if(error&128) goto porterr;
  addSum('0','3');

  }

  error=transmit(port,bdata[0]);
  if(error&128) goto porterr;
  error=transmit(port,bdata[1]);  // send data
  if(error&128) goto porterr;
  addSum(bdata[0],bdata[1]);
  progress(pg);
  if(counterror==0)
  {
  count++;pg++;
  }
  if(count==16)           // end of line
  {
  error=sendSum(sum);
  if(error&128) goto porterr;

  //----- CORRECTNESS -------
  Z2=(char)receive(port);

  if(Z2=='4')
    {
    counterror=0;
    }
    else if(Z2=='5')
      {
      counterror++;
      goto re_load;
      }
    else
      {
      counterror++;
      goto re_load;
      }
  count=0;
   //////////////
  }
  }while(pg<range);

if(remain==0)
{
  error=sendSum(sum);
  //----- CORRECTNESS -------
  Z2=(char)receive(port);


  if(Z2=='4')
    {
    counterror=0;
    }
    else if(Z2=='5')
      {
      counterror++;
      goto re_load;
      }
    else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
            {
            counterror++;
            goto re_load;
            }
      }
/*      if(error&128) goto porterr;
      error=transmit(port,0x0D);
      if(error&128) goto porterr;
      error=transmit(port,0x0A);
      if(error&128)goto out1;*/
//      else
        {
        // --  end of Transmit :00 0000 01 FF
        error=transmit(port,':');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;

        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'0');if(error&128)goto out1;

        error=transmit(port,'0');if(error&128)goto out1;
        error=transmit(port,'1');if(error&128)goto out1;

        error=transmit(port,'F');if(error&128)goto out1;
        error=transmit(port,'F');if(error&128)goto out1;
        error=transmit(port,0x0D);if(error&128)goto out1;
        error=transmit(port,0x0A);if(error&128)goto out1;
        progress(range);
        complete("FILL Complete",13);
        }
      goto out2;
      }
      if(!r_err==1)
        {

        out1:
        fail("Communication Error",19);
        }
      goto out2;
      porterr:
        fail("Port Error",10);
      out2:getch();
    }

  void ChkError::enter(int port)
    {
    int error=0,i;
    int counterror;
    char Z1,Z2;
    textcolor(LIGHTGREEN);textbackground(BLUE);
    gotoxy(20,4);cprintf("ENTER DATA");
    getaddr1();
    getdata(1);
    if(error==0)
      {
      counterror=0;
      legaladdr();

  re_load:
      if(counterror==3)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  {
  fail("Check Sum Error",15);
  goto out2;
  }
sum = 0;                // reset Sum
error=transmit(port,':');
if(error&128)goto out1;

error=transmit(port,'0'); // Byte counter = 1
if(error&128)goto out1;
error=transmit(port,'1');
if(error&128)goto out1;
addSum('0','1');

for(i=0;i<=3;i++)       // Send address
  error=transmit(port,startaddr[i]);
addSum(startaddr[0],startaddr[1]);
addSum(startaddr[2],startaddr[3]);

error=transmit(port,'0');
error=transmit(port,'0');     // Type
addSum('0','0');

error=transmit(port,bdata[0]);
error=transmit(port,bdata[1]);  // send data
addSum(bdata[0],bdata[1]);

error=sendSum(sum);
//----- CORRECTNESS -------
  Z1=(char)receive(port);
  Z2=(char)receive(port);

  if((Z1=='0')&&(Z2=='4'))   // ---- NO ERROR
    {
    counterror=0;
    }
    else if((Z1=='0')&&(Z2=='5'))  //---- ERROR
      {
      counterror++;
      goto re_load;
      }
    else
      {
      counterror++;
      goto re_load;
      }

error=transmit(port,0x0D);
error=transmit(port,0x0A);
error=transmit(port,':');
error=transmit(port,'0');
error=transmit(port,'0');
error=transmit(port,'0');
error=transmit(port,'0');
error=transmit(port,'0');
error=transmit(port,'0');

error=transmit(port,'0');
error=transmit(port,'1');

error=transmit(port,'F');
error=transmit(port,'F');
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    error=transmit(port,0x0D);
    error=transmit(port,0x0A);

    complete("ENTER Complete",14);
    goto out2;
    }
    out1:
      fail("Communication Error",18);

    out2:getch();
  }
void ChkError::search()
  {
  int r_err=0,error=0;
  int walk=0;
  textcolor(LIGHTGREEN);textbackground(BLUE);
  gotoxy(20,4);cprintf("SEARCH DATA");
  getaddr2();
  range=end-start;
  if(range<0)
    {
    fail("Address Range Error!",20);
    r_err=1;
    }
  else
    {
    getdataSet(3);
    legaladdr();
    int i=range;
    sum = 0;                    // reset Sum
    error=transmit(port,':');
    if(error&128)goto out1;
    int2hex(range);

    error=transmit(port,victim[1]); // Byte counter
    if(error&128)goto out1;
    error=transmit(port,victim[0]);
    if(error&128)goto out1;
    addSum(victim[1],victim[0]);


    for(i=0;i<=3;i++)       // Send address
      {
      error=transmit(port,startaddr[i]);
      if(error&128)goto out1;
      }
    addSum(startaddr[0],startaddr[1]);
    addSum(startaddr[2],startaddr[3]);

    error=transmit(port,'0');
    if(error&128)goto out1;
    error=transmit(port,'2');     // Type
    if(error&128)goto out1;
    addSum('0','2');
    i++;   // add for last byte is CHKSUM
    for(; i>=0 ; i--)
      {
      error=receive(port);        // Receive data
      if(error&128)goto out1;
      else if(error==13)
      {
      fail("EPROM-EMU. Not Respone",22);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        getch();
        return;
        }
        else
        {
        hunter[walk]=(char)error;
        walk++;
        }
        error=receive(port);
        if(error&128)goto out1;
        else if(error==13)
        {
        fail("EPROM-EMU. Not Respone",22);
        getch();
        return;
        }
        else
        {
        hunter[walk]=(char)error;
        walk++;
        }
        if(i!=0)  // If not last;Add Sum
        addSum(hunter[walk-2],hunter[walk-1]);
        }
      error=checkSum(sum,hunter[walk-2],hunter[walk-1]);
      if(error&128)
        {
        fail("Checksum Error!",15);
        r_err=1;
        goto out1;
        }

  //------------ BEGIN SEARCH
      i=range;
      int tempwalk=0;
      int targetwalk=0;
      int loop=0;
      walk=0;
      do
        {
        targetwalk=0;
        tempwalk=walk;
        do
        {
        if(sdata[targetwalk]==hunter[tempwalk])
          {
          tempwalk++;
          targetwalk++;
          loop=1;
          }
        else
          {
          walk++;walk++;
          loop=0;
          }

        }while(sdata[targetwalk]!='\0');
        if(loop==1)
        {
        complete("FOUND!",6);
        //equal
        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      }while(i>0);  // still range
    if(loop==0)
    complete("Not Found",13);
    goto succ;
    }

  out1:
  if(r_err!=1)
    fail("Communication Error!",18);
  succ:getch();
  }
void ChkError::move()
  {
  int error=0,r_err=0;
  int walk=0;
  textcolor(LIGHTGREEN);textbackground(BLUE);
  gotoxy(20,4);cprintf("MOVE DATA");
  getaddr2();
  range=end-start;
  if(range<0)
    {
    fail("Address Range Error!",20);
    r_err=1;
    }
  else
    {
    gettarget();
    legaladdr();
    int i=range;
    sum = 0;                    // reset Sum
    error=transmit(port,':');
    if(error&128)goto endmove;
    int2hex(range);

    error=transmit(port,victim[1]); // Byte counter
    if(error&128)goto endmove;
    error=transmit(port,victim[0]);
    if(error&128)goto endmove;
    addSum(victim[1],victim[0]);

    for(i=0;i<=3;i++)       // Send address
      {
      error=transmit(port,startaddr[i]);
      if(error&128)goto endmove;
      }
    addSum(startaddr[0],startaddr[1]);
    addSum(startaddr[2],startaddr[3]);

    error=transmit(port,'0');
    if(error&128)goto endmove;
    error=transmit(port,'2');     // Type
    if(error&128)goto endmove;
    addSum('0','2');

    i=range;
    i++;   // add for last byte is CHKSUM
    for(; i>=0 ; i--)
      {
      error=receive(port);        // Receive data
      if(error&128)goto endmove;
      else if(error==13)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        {
        fail("EPROM-EMU. Not Respone",22);
        getch();
        return;
        }
        else
        {
        hunter[walk]=(char)error;
        walk++;
        }
        error=receive(port);
        if(error&128)goto endmove;
        else if(error==13)
        {
        fail("EPROM-EMU. Not Respone",22);
        getch();
        return;
        }
        else
        {
        hunter[walk]=(char)error;
        walk++;
        }
        if(i!=0)  // If not last;Add Sum
        addSum(hunter[walk-2],hunter[walk-1]);
        }
    error=checkSum(sum,hunter[walk-2],hunter[walk-1]);
    if(error&128)
        {
        fail("SUM Error:PC «- EPROM",21);
        r_err=1;
        goto endmove;
        }
//-------^--- Upload Source data
    else     // Checksum not error ; Send ChkError
        {
        sum = 0;                  // reset Sum
        int error=transmit(port,':');
        if(error&128)goto endmove;
        int2hex(range);

        error=transmit(port,victim[1]); // Byte counter
        if(error&128)goto endmove;
        error=transmit(port,victim[0]);
        if(error&128)goto endmove;
        addSum(victim[1],victim[0]);

        for(i=0;i<=3;i++)       // Send address
        {
        error=transmit(port,targetaddr[i]);
        if(error&128)goto endmove;
        }
        addSum(targetaddr[0],targetaddr[1]);
        addSum(targetaddr[2],targetaddr[3]);

        error=transmit(port,'0');
        if(error&128)goto endmove;
        error=transmit(port,'3');     // Type
        if(error&128)goto endmove;
        addSum('0','3');

        for(i=0;i<=range;i++)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      {
      error=transmit(port,hunter[i]);
      if(error&128)goto endmove;
      i++;
      error=transmit(port,hunter[i]);
      if(error&128)goto endmove;
      addSum(hunter[i-1],hunter[i]);
      }
      error=sendSum(sum);
      if(error&128)
      {
      fail("SUM Error:PC -» EPROM",21);
      goto endmove2;
      }
      }
    }
  endmove:if((error&128)&&(r_err==0))
    fail("Communication Error",19);
  endmove2:getch();
  }

void ChkError::dump(int port)
  {
  int error=0,r_err=0,y,tt;
  int walk=0,countbyte,addrwalk;
  char aa[3000];
  textcolor(LIGHTGREEN);textbackground(BLUE);
  gotoxy(20,4);cprintf("DUMP DATA");
  getaddr2();
  for(i=0;i<=3000;i++)
    aa[i]='G';
  range=end-start;
  if(range<0)
    {
    fail("Address Range Error!",20);
    r_err=1;
    }
  else
    {
    legaladdr();
    int i=range;
    sum = 0;                    // reset Sum
    error=transmit(port,':');
    if(error&128)goto enddump;
    int2hex(range);

    error=transmit(port,victim[1]); // Byte counter
    if(error&128)goto enddump;
    error=transmit(port,victim[0]);
    if(error&128)goto enddump;
    addSum(victim[1],victim[0]);

    for(i=0;i<=3;i++)       // Send address
      {
      error=transmit(port,startaddr[i]);
      if(error&128)goto enddump;
      }
    i=0;
    addSum(startaddr[0],startaddr[1]);
    addSum(startaddr[2],startaddr[3]);
    i=0;
    error=transmit(port,'0');
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    error=transmit(port,'2');     // Type
    addSum('0','2');

//    do{
//      aa[i]=(char)receive(port);//cprintf("%d",aa[i]);
//      if((aa[i]=receive(port))==128){r_err=1;goto enddump;}else
//      i++;
//      }while(i<range);
//****************
    i=range;
    i++;   // add for last byte is CHKSUM
    (char)error='$';
    for(; i>=0 ; i--)
      {
      error=receive(port);        // Receive data
      if(error&128)goto enddump;
      else if(error==13)
      {
      fail("EPROM-EMU. Not Respone 1",22);
      getch();
      return;
      }
      else
      {
//    cprintf("%c",(char)error);
      hunter[walk]=(char)error;
      walk++;
//    gotoxy(10,5);cprintf("%d",walk);
      }
      (char)error='$';
      error=receive(port);
      if(error&128)goto enddump;
      else if(error==13)
      {
      fail("EPROM-EMU. Not Respone 2",22);
      getch();
      return;
      }
      else
      {
//    cprintf("%c ",(char)error);
      hunter[walk]=(char)error;
      walk++;
//    gotoxy(10,5);cprintf("%d",walk);
      }
      if(i!=0)  // If not last;Add Sum
      {
      addSum(hunter[walk-2],hunter[walk-1]);
      }
      }
    error=checkSum(sum,hunter[walk-2],hunter[walk-1]);
/*    if(error==1)
      {
      fail("SUM Error:PC «- EPROM",21);
      goto enddump2;
      }*/

//    for(i=0;i<=20;i+=2)
//      cprintf("%c%c ",aa[i],aa[i+1]);
    error=transmit(port,0x0D);
    error=transmit(port,0x0A);
    error=transmit(port,':');
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    error=transmit(port,'0');
    error=transmit(port,'0');

    error=transmit(port,'0');
    error=transmit(port,'0');
    error=transmit(port,'0');
    error=transmit(port,'0');

    error=transmit(port,'0');
    error=transmit(port,'1');

    error=transmit(port,'F');
    error=transmit(port,'F');
    error=transmit(port,0x0D);
    error=transmit(port,0x0A);
      //-------^--- Upload Source data

      getch();
    i=0;
    int2hex(start);
    addrwalk=start;
    walk=0;y=2;countbyte=0;
    cls();
    do
      {
      int2hex(addrwalk);
      if(countbyte==0)
      {
      gotoxy(3,y);
      cprintf("%c%c%c%c: ",victim[3],victim[2],victim[1],victim[0]);
      }
      if(countbyte==7)
      {
      cprintf("%c%c - ",hunter[walk],hunter[walk+1]);

      }
      else
      {
      cprintf("%c%c ",hunter[walk],hunter[walk+1]);
      }
      if(countbyte==15)
      {
      y++;countbyte=0;
      }
      else
      {
      countbyte++;
      }
      if(y>22)
      {
      cout<<"Press Anykey";getch();
      cls();
      y=2;
      }
      walk+=2;i++;
      addrwalk++;
      }while(i<range);
    }
enddump:if(r_err==1)fail("Communication Error",19);
enddump2:getch();
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
void ChkError::compare()
  {
//  int r_err=0;
  textcolor(LIGHTGREEN);textbackground(BLUE);
  gotoxy(20,4);cprintf("COMPARE DATA");
  getaddr2();
  range=end-start;
  if(range<0)
    {
    fail("Address Range Error!",20);
//    r_err=1;
    }
  else
    {
    gettarget();
    pgx=0;
    range=end-start;
    pgbar();
    for(int i=0;i<=range;i++)
    progress(i);
    complete("YO!",3);
    }
  getch();
  }

void ChkError::convert(int type)
      {
      //   Type 0 for Address start
      //        1 for Address end
      //        2 for Taget Address
      //        3 for Data
      if(type==0)
      start = hex2dec(startaddr[0],startaddr[1],
        .     startaddr[2],startaddr[3],counts);
      else if(type==1)
      end   = hex2dec(endaddr[0],endaddr[1],
            endaddr[2],endaddr[3],counte);
      else if(type==2)
      range = end-start;
      }
int ChkError::hex2dec2(int bite,char ch)
  {
  int number;
  switch(ch)
    {
    case '0' : number = 0; break;
    case '1' : number = 1*(pow(16,bite)); break;
    case '2' : number = 2*(pow(16,bite)); break;
    case '3' : number = 3*(pow(16,bite)); break;
    case '4' : number = 4*(pow(16,bite)); break;
    case '5' : number = 5*(pow(16,bite)); break;
    case '6' : number = 6*(pow(16,bite)); break;
    case '7' : number = 7*(pow(16,bite)); break;
    case '8' : number = 8*(pow(16,bite)); break;
    case '9' : number = 9*(pow(16,bite)); break;
    case 'a' : number = 10*(pow(16,bite)); break;
    case 'b' : number = 11*(pow(16,bite)); break;
    case 'c' : number = 12*(pow(16,bite)); break;
    case 'd' : number = 13*(pow(16,bite)); break;
    case 'e' : number = 14*(pow(16,bite)); break;
    case 'f' : number = 15*(pow(16,bite)); break;
    case 'A' : number = 10*(pow(16,bite)); break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    case 'B' : number = 11*(pow(16,bite)); break;
    case 'C' : number = 12*(pow(16,bite)); break;
    case 'D' : number = 13*(pow(16,bite)); break;
    case 'E' : number = 14*(pow(16,bite)); break;
    case 'F' : number = 15*(pow(16,bite)); break;
    default  : number = 0; break;
    }
  return number;
  }

int ChkError::hex2dec(char c3,char c2,char c1,char c0,int count)
  {
  int t0,t1,t2,t3,num=0;

  if(count==4)
    {
    t0=hex2dec2(0,c0);
    t1=hex2dec2(1,c1);
    t2=hex2dec2(2,c2);
    t3=hex2dec2(3,c3);
    num=t0+t1+t2+t3;
    }
  else if(count==3)
    {
    t0=hex2dec2(0,c1);
    t1=hex2dec2(1,c2);
    t2=hex2dec2(2,c3);
    num=t0+t1+t2;
    }
  else if(count==2)
    {
    t0=hex2dec2(0,c2);
    t1=hex2dec2(1,c3);
    num=t0+t1;
    }
  else if(count==1)
    {
    t0=hex2dec2(0,c3);
    num=t0;
    }
  return num;
  }

void ChkError::legaladdr()
  {
  if(counts==3)                  // Legal Start address
    {
    startaddr[4]='\0';
    startaddr[3]=startaddr[2];
    startaddr[2]=startaddr[1];
    startaddr[1]=startaddr[0];
    startaddr[0]='0';
    }
  else if(counts==2)
    {
    startaddr[4]='\0';
    startaddr[3]=startaddr[1];
    startaddr[2]=startaddr[0];
    startaddr[1]=startaddr[0]='0';
    }
  else if(counts==1)
    {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      startaddr[4]='\0';
      startaddr[3]=startaddr[0];
      startaddr[2]=startaddr[1]=startaddr[0]='0';
      };
    if(counte==3)                    // Legal end address
      {
      endaddr[4]='\0';
      endaddr[3]=endaddr[2];
      endaddr[2]=endaddr[1];
      endaddr[1]=endaddr[0];
      endaddr[0]='0';
      }
    else if(counte==2)
      {
      endaddr[4]='\0';
      endaddr[3]=endaddr[1];
      endaddr[2]=endaddr[0];
      endaddr[1]=endaddr[0]='0';
      }
    else if(counte==1)
      {
      endaddr[4]='\0';
      endaddr[3]=endaddr[0];
      endaddr[2]=endaddr[1]=endaddr[0]='0';
      };
    if(countt==3)                    // Legal target address
      {
      targetaddr[4]='\0';
      targetaddr[3]=targetaddr[2];
      targetaddr[2]=targetaddr[1];
      targetaddr[1]=targetaddr[0];
      targetaddr[0]='0';
      }
    else if(countt==2)
      {
      targetaddr[4]='\0';
      targetaddr[3]=targetaddr[1];
      targetaddr[2]=targetaddr[0];
      targetaddr[1]=targetaddr[0]='0';
      }
    else if(countt==1)
      {
      targetaddr[4]='\0';
      targetaddr[3]=targetaddr[0];
      targetaddr[2]=targetaddr[1]=targetaddr[0]='0';
      }
    }
  void ChkError::legaldata()
    {
    int i=0,j;

    do
      {
      if(sdata[i]==' ')
        {
        j=i;
        do
          j++;
        while(sdata[j]==' ');
        sdata[i]=sdata[j];
        sdata[j]=' ';
        }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    i++;
    }while(sdata[i-1]!='\0');
  countt=i;
  }

void ChkError::getaddr2()
  {
  int x=35,y=6,para=0;
  int ch;
  textbackground(BLUE);
  textcolor(WHITE);
  gotoxy(20,6);  cprintf("Address Start: ");

  textbackground(CYAN);
  gotoxy(35,6);  cprintf("    ");

  _setcursortype(_NORMALCURSOR);
  i=0;
  do{
    gotoxy(x+i,y);
    ch = bioskey(0);
    if((((char)ch >= '0')&&((char)ch <= '9'))||
       (((char)ch >= 'a')&&((char)ch <= 'f'))||
       (((char)ch >= 'A')&&((char)ch <= 'F')))
       {                          // if HEX number
       if(i<4)
       {
       cprintf("%c",(char)ch);


       switch(para)
         {
         case 0:startaddr[i]=(char)ch;break;
         case 1:endaddr[i]=(char)ch;break;
         }
       i++;
       }
       }
    else if(ch==BKSP)
      {
      gotoxy(x+i-1,y);cprintf(" ");
      if(i>0)
      i--;
      if(x>35)
      x--;
      }
    else if(ch==ENTER)
      {
      switch(para)
      {
      case 0:startaddr[i]='\0';counts=i;break;
      case 1:endaddr[i]='\0';counte=i;break;
      }
      para++;
      y+=2;
      x=35;
      i=0;
      textbackground(BLUE);
      switch(para)
      {
      case 1:gotoxy(20,8);cprintf("Address End  : ");
             textbackground(CYAN);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
                gotoxy(35,8);  cprintf("    ");break;
        case 2:goto Out;
        }
        }
      }while(ch!=ESCAPE);
    Out:;
    if(para<2)
      return;
    else
      {
      convert(0);
      convert(1);
      }

    }

  void ChkError::getaddr1()
    {
    int x=35;
    int ch;
    textbackground(BLUE);
    textcolor(WHITE);
    gotoxy(26,6);  cprintf("Address: ");

    textbackground(CYAN);
    gotoxy(35,6);  cprintf("    ");

    _setcursortype(_NORMALCURSOR);
    i=0;
    do{
      gotoxy(x+i,6);
      ch = bioskey(0);
      if((((char)ch >= '0')&&((char)ch <= '9'))||
         (((char)ch >= 'a')&&((char)ch <= 'f'))||
         (((char)ch >= 'A')&&((char)ch <= 'F')))
         {
         if(i<4)
         {                          // if HEX number
         cprintf("%c",(char)ch);
         startaddr[i]=(char)ch;
         i++;
         }
         }
      else if(ch==BKSP)
        {
        gotoxy(x+i-1,6);cprintf(" ");
        if(i>0)
        i--;
        if(x>35)
        x--;
        }
      else if(ch==ENTER)
        {
        startaddr[i]='\0';counts=i;
        x=35;
        i=0;
        textbackground(BLUE);
        goto Out;
        }
      }while(ch!=ESCAPE);
    Out:;
    if(ch==ENTER)
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    convert(0);
  }


void ChkError::gettarget()
  {
  int x=35;
  int ch;
  textbackground(BLUE);
  textcolor(WHITE);
  gotoxy(20,10);  cprintf("Target Address:");

  textbackground(CYAN);
  gotoxy(35,10);  cprintf("    ");

  _setcursortype(_NORMALCURSOR);
  i=0;
  do{
    gotoxy(x+i,10);
    ch = bioskey(0);
    if((((char)ch >= '0')&&((char)ch <= '9'))||
       (((char)ch >= 'a')&&((char)ch <= 'f'))||
       (((char)ch >= 'A')&&((char)ch <= 'F')))
       {
       if(i<4)
       {                          // if HEX number
       cprintf("%c",(char)ch);
       targetaddr[i]=(char)ch;
       i++;
       }
       }
    else if(ch==BKSP)
      {
      gotoxy(x+i-1,10);cprintf(" ");
      if(i>0)
      i--;
      if(x>35)
      x--;
      }
    else if(ch==ENTER)
      {
      targetaddr[i]='\0';
      countt=i;
      goto out;
      }
    }while(ch!=ESCAPE);
  out:
  if(ch==ENTER)
    convert(3);
  }

void ChkError::getdata(int rr)
  {
  int x=35,y;
  int ch;
  switch(rr)
    {
    case 1:y=8;break;
    case 2:y=10;break;
    case 3:y=12;break;
    };
  textbackground(BLUE);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  textcolor(WHITE);
  gotoxy(26,y);  cprintf("Data    : ");

  textbackground(CYAN);
  gotoxy(35,y);  cprintf("  ");

  _setcursortype(_NORMALCURSOR);
  i=0;

  do{
    gotoxy(x+i,y);
    ch = bioskey(0);
    if((((char)ch >= '0')&&((char)ch <= '9'))||
       (((char)ch >= 'a')&&((char)ch <= 'f'))||
       (((char)ch >= 'A')&&((char)ch <= 'F')))
       {                          // if HEX number
       if(i<2)
       {
       cprintf("%c",(char)ch);
       bdata[i]=(char)ch;
       i++;
       }
       }
    else if(ch==BKSP)
      {
      gotoxy(x+i-1,y);cprintf(" ");
      if(i>0)
      i--;
      if(x>35)
      x--;
      }
    else if(ch==ENTER)
      {
      bdata[i]='\0';countd=i;
      x=35;
      i=0;
      textbackground(BLUE);
      goto Out;
      }
    }while(ch!=ESCAPE);
  Out:;
  }

void ChkError::getdataSet(int rr)
  {
  int x=35,y;
  int ch;
  switch(rr)
    {
    case 1:y=8;break;
    case 2:y=10;break;
    case 3:y=12;break;
    };
  textbackground(BLUE);
  textcolor(WHITE);
  gotoxy(26,y);  cprintf("Data    : ");

  textbackground(CYAN);
  gotoxy(35,y);  cprintf("                    ");

  _setcursortype(_NORMALCURSOR);
  i=0;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  do{
    gotoxy(x+i,y);
    ch = bioskey(0);
    if((((char)ch >= '0')&&((char)ch <= '9'))||
       (((char)ch >= 'a')&&((char)ch <= 'f'))||
       (((char)ch >= 'A')&&((char)ch <= 'F'))||
       ((char)ch==' '))
       {                          // if HEX number
       if(i<20)
       {
       if((sdata[i-1]==' ')&&((char)ch==' '))
         goto xtemp;
       cprintf("%c",(char)ch);
       sdata[i]=(char)ch;
       i++;
       }
       }
    else if(ch==BKSP)
      {
      gotoxy(x+i-1,y);cprintf(" ");
      if(i>0)
      i--;
      if(x>35)
      x--;
      }
    else if(ch==ENTER)
      {
      sdata[i]='\0';countd=i;
      x=35;
      i=0;
      textbackground(BLUE);
      legaldata();
      goto Out;
      }
    xtemp:
    }while(ch!=ESCAPE);
  Out:;
  }


void ChkError::int2hex(int val)
{
//   extern char *_addr[4];
   int temp,i;
   for(i=3;i>=0;i--)
   {
       temp = val / ((int)pow(16,i));
       switch(temp)
       {
       case 0  : victim[i]='0';break;
       case 1  : victim[i]='1';break;
       case 2  : victim[i]='2';break;
       case 3  : victim[i]='3';break;
       case 4  : victim[i]='4';break;
       case 5  : victim[i]='5';break;
       case 6  : victim[i]='6';break;
       case 7  : victim[i]='7';break;
       case 8  : victim[i]='8';break;
       case 9  : victim[i]='9';break;
       case 10 : victim[i]='A';break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        case 11 : victim[i]='B';break;
        case 12 : victim[i]='C';break;
        case 13 : victim[i]='D';break;
        case 14 : victim[i]='E';break;
        case 15 : victim[i]='F';break;
        }
        val = val % ((int)pow(16,i));
    }
}

void ChkError::addSum(char high,char low)
  {
  int thigh,tlow;

  thigh=hex2dec2(1,high);
  tlow =hex2dec2(0,low);
  sum = sum+thigh+tlow;
  }


int ChkError::sendSum(int sum)
  {
  unsigned int temp;
  int ret;
  temp = ((~sum)&0x00ff) + 1;      // 2' complement
  int2hex(temp);

  ret=transmit(port,victim[1]);  // high
    if(ret&128) return ret;
  ret=transmit(port,victim[0]);  // low
    return ret;
  }
int ChkError::checkSum(int sum,char high,char low)
  {
  unsigned int temp;
  int ret;
  temp = ((~sum)&0x00ff) + 1;   // 2'
  int2hex(temp);
  if(victim[1]!=high)
    return 1;
  if(victim[0]!=low)
    return 1;
  else return 0;
  }
void ChkError::progress(unsigned long int xx)
  {
  int i;
  if(range>0)
  _setcursortype(_NOCURSOR);
  xx=((xx*100)/range);
  gotoxy(20,18);
  for(i=0;i<=(40*xx)/100;i++)
    {
    textcolor(YELLOW);
    cprintf("");
    }
  textcolor(WHITE);textbackground(BLUE);
  gotoxy(38,17);cprintf("%d",xx);cprintf("%");
  delay(20);

  }
void ChkError::pgbar()
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
{
gotoxy(20,18);
textcolor(CYAN);
textbackground(BLUE);
cprintf("                                                  ");
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
////////////////////////////////////
//  Zfile.cpp (File Managenent)
////////////////////////////////////
#include"zheader.h"
#include<bios.h>
#include<iostream.h>
#include<fstream.h>
#include<stdio.h>

//void sounderror();

class File
  {
  private:
    char fname[40],fvictim[4];
    int sum;
  public:
    FILE *fp;
    ifstream infile;
    int filefail;
//    int rangefile,filefail;//,sum;
    long int rangefile;
    char data[3000];
    void getname();
    int readfile();
    void file2eprom();
    void ffail();
    int  ReadFile(int);
    void listfile();
    void sendFile(int);
    void send2(int,int);
    void int2hex(int);
    int  checkSum(int ,char ,char );
    int  sendSum(int ,int);
    void addSum(char ,char );
    int  hex2dec(char c3,char c2,char c1,char c0,int count);
    int  hex2dec2(int bite,char ch);
    void progress(unsigned long int xx);
    void pgbar();
  };

void File::getname()
  {
  int ch,i=0,x=26;

  textcolor(YELLOW);textbackground(BLUE);
  gotoxy(20,3);cprintf("Open File for Download to EPROM");
  gotoxy(15,6);cprintf("File Name:");
  textbackground(CYAN);
  gotoxy(25,6);cprintf("                                        ");
  textcolor(BLUE);_setcursortype(_NORMALCURSOR);
  do
    {
    gotoxy(x,6);
    ch = bioskey(0);
    if(((char)ch > ' ')&&((char)ch <= 'z'))     // if character
      {
      cprintf("%c",(char)ch);
      fname[i]=(char)ch;
      x++;
      i++;
      }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      else if(ch==BKSP)
        {
        gotoxy(x-1,6);cprintf(" ");
        if(x>26)
        x--;
        if(i>0)
        i--;
        }
      }while((ch != ESCAPE)&&(ch != ENTER));
    if(ch == ENTER)
      fname[i]='\0';
    }

  int File::ReadFile(int port)
    {
    char ch,ch1,ch2;
    int i=0,first=0,error=0,dontcare=0;
    int line=0,item=0;
    filefail=0;
    if ((fp = fopen(fname, "rt"))== NULL)
      {
      fail("Can Not Open File!",18);
      getch();
      filefail = 0;
      goto ex;
      }
    i=0;
    while(!feof(fp))
      {
      if(first==0)
        {
        ch=fgetc(fp);
        if(ch==':')
          {
          first=1;
          dontcare++;
          i++;
          }
        }
      else if(first==1)
        {
        ch=fgetc(fp);
        dontcare++;
        if(dontcare>=9)
          {
  //      stack[line][item] =
          }
        i++;
        }
      }
    fclose(fp);
    rangefile=i;
    pgbar();
    if ((fp = fopen(fname, "rt"))== NULL)
      {
      fail("Can Not Open File!",18);
      filefail = 1;
      goto out;
      }

    i=0;first=0;
    while (!feof(fp))
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    {
    if(first==1)
      {
      ch=fgetc(fp);
      data[i]=ch;
      error=transmit(port,ch);
      if(error&128)
      {
      fail("Port Error",10);
      getch();
      goto ex;
      }
      i++;
      progress(i);
      }
    else if(first==0)
      {
      ch=fgetc(fp);
      if(ch==':')
      {
      error=transmit(port,ch);
      if(error&128)
        {
        fail("Port Error",10);
        getch();
        goto ex;
        }
      first=1;
      }
      }
    }
  rangefile=i;
  out:
//  error=transmit(port,0x0D);
//  error=transmit(port,0x0A);
  fclose(fp);
  error=transmit(port,0x0D);
  error=transmit(port,0x0A);
  error=transmit(port,':');
  error=transmit(port,'0');
  error=transmit(port,'0');

  error=transmit(port,'0');
  error=transmit(port,'0');
  error=transmit(port,'0');
  error=transmit(port,'0');

  error=transmit(port,'0');
  error=transmit(port,'1');

  error=transmit(port,'F');
  error=transmit(port,'F');
  error=transmit(port,0x0D);
  error=transmit(port,0x0A);


  if(filefail==1)
    {
    fail("Download ERROR",14);
    getch();
    return 1;
    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  else
    {
    progress(rangefile);
    complete("Download Success",15);
    getch();
    return 0;
    }
  ex:return 1;
  }

int File::readfile()
  {
  int i=0;
  filefail=0;
  infile.open(fname);  // Open file
  if(infile.fail())
    {
    fail("Can Not Open File!",18);
    filefail = 1;
    goto out;
    }
  while(!infile.eof())
    {
//    data[i]=fgetc(infile);
    infile.get(data[i]);
    i++;
    }
  rangefile=i;
  out:
  infile.close();
  if(filefail==1)
    {
    getch();
    return 1;
    }
  else return 0;
  }

void File::listfile()
  {
  int i=0,ch,count=1;
  int x=8,y=2;
  if(filefail!=1)
    {
    cls();
    do
      {
      i++;
      }while(data[i]==':');

    for( ; i<rangefile;i++)
      if(data[i]=='\n')
      {
      x=8;y++;
      count=1;
      if(y>22)
        {
        cout<<" Press anykey to nextpage";
        ch=bioskey(0);
        if(ch==ESCAPE)
          goto out;
        cls();
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          y=2;
          }
        i++;
        }
        else
        {
        textcolor(LIGHTGREEN);textbackground(BLUE);
        gotoxy(x,y);x++;
        cprintf("%c",data[i]);
        if((count==2)||(count==6)||(count==8))
          {
          gotoxy(x,y);x++;
          cprintf("-");
          }
        count++;
        }
      }
    getch();
    out:
    }

void File::int2hex(int val)
{
   int temp,i;
   for(i=3;i>=0;i--)
   {
       temp = val / ((int)pow(16,i));
       switch(temp)
       {
       case 0  : fvictim[i]='0';break;
       case 1  : fvictim[i]='1';break;
       case 2  : fvictim[i]='2';break;
       case 3  : fvictim[i]='3';break;
       case 4  : fvictim[i]='4';break;
       case 5  : fvictim[i]='5';break;
       case 6  : fvictim[i]='6';break;
       case 7  : fvictim[i]='7';break;
       case 8  : fvictim[i]='8';break;
       case 9  : fvictim[i]='9';break;
       case 10 : fvictim[i]='A';break;
       case 11 : fvictim[i]='B';break;
       case 12 : fvictim[i]='C';break;
       case 13 : fvictim[i]='D';break;
       case 14 : fvictim[i]='E';break;
       case 15 : fvictim[i]='F';break;
       }
       val = val % ((int)pow(16,i));
   }
}



void File::progress(unsigned long int xx)
  {
  int i;
  if(rangefile>0)
  _setcursortype(_NOCURSOR);
  xx=((xx*100)/rangefile);
  gotoxy(20,18);
  for(i=0;i<=(40*xx)/100;i++)
    {
    textcolor(YELLOW);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    cprintf("");
    }
  textcolor(WHITE);textbackground(BLUE);
  gotoxy(38,17);cprintf("%d",xx);cprintf("%");
  delay(1);

  }
void File::pgbar()
  {
  gotoxy(20,18);
  textcolor(CYAN);
  textbackground(BLUE);
  cprintf("");
  }

void File::addSum(char high,char low)
  {
  int thigh,tlow;

  thigh=hex2dec2(1,high);
  tlow =hex2dec2(0,low);
  sum = sum+thigh+tlow;
  }

int File::sendSum(int port,int sum)
  {
  unsigned int temp;
  int ret;
  temp = ((~sum)&0x00ff) + 1;     // 2' complement
  int2hex(temp);
  ret=transmit(port,fvictim[1]);  // high
    if(ret==1) return ret;
  ret=transmit(port,fvictim[0]);  // low
    return ret;
  }
int File::checkSum(int sum,char high,char low)
  {
  unsigned int temp;
  int ret;
  temp = ((~sum)&0x00ff) + 1;    // 2'
  int2hex(temp);
  if(fvictim[1]!=high)
    return 1;
  if(fvictim[0]!=low)
    return 1;
  else return 0;
  }


int File::hex2dec2(int bite,char ch)
  {
  int number;
  switch(ch)
    {
    case '0' : number = 0; break;
    case '1' : number = 1*(pow(16,bite)); break;
    case '2' : number = 2*(pow(16,bite)); break;
    case '3' : number = 3*(pow(16,bite)); break;
    case '4' : number = 4*(pow(16,bite)); break;
    case '5' : number = 5*(pow(16,bite)); break;
    case '6' : number = 6*(pow(16,bite)); break;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
      case '7' : number = 7*(pow(16,bite)); break;
      case '8' : number = 8*(pow(16,bite)); break;
      case '9' : number = 9*(pow(16,bite)); break;
      case 'a' : number = 10*(pow(16,bite)); break;
      case 'b' : number = 11*(pow(16,bite)); break;
      case 'c' : number = 12*(pow(16,bite)); break;
      case 'd' : number = 13*(pow(16,bite)); break;
      case 'e' : number = 14*(pow(16,bite)); break;
      case 'f' : number = 15*(pow(16,bite)); break;
      case 'A' : number = 10*(pow(16,bite)); break;
      case 'B' : number = 11*(pow(16,bite)); break;
      case 'C' : number = 12*(pow(16,bite)); break;
      case 'D' : number = 13*(pow(16,bite)); break;
      case 'E' : number = 14*(pow(16,bite)); break;
      case 'F' : number = 15*(pow(16,bite)); break;
      default  : number = 0; break;
      }
    return number;
    }
int File::hex2dec(char c3,char c2,char c1,char c0,int count)
    {
    int t0,t1,t2,t3,num=0;
    if(count==4)
      {
      t0=hex2dec2(0,c0);
      t1=hex2dec2(1,c1);
      t2=hex2dec2(2,c2);
      t3=hex2dec2(3,c3);
      num=t0+t1+t2+t3;
      }
    else if(count==3)
      {
      t0=hex2dec2(0,c1);
      t1=hex2dec2(1,c2);
      t2=hex2dec2(2,c3);
      num=t0+t1+t2;
      }
    else if(count==2)
      {
      t0=hex2dec2(0,c2);
      t1=hex2dec2(1,c3);
      num=t0+t1;
      }
    else if(count==1)
      {t0=hex2dec2(0,c3);
      num=t0;   }
    return num;
    }
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
//////////////////////////////////////////
//  SCREEN.CPP ( USER INTERFACE ROUTINE ) //
//////////////////////////////////////////
#include"zheader.h"
void drawscreen(void);
void goodbye(void);
void complete(char,int);
void fail(char,int);
void fail2(char,int);
void portfail(int);
void dosmode(void);

//------ DRAW BARE SCREEN  AND OUTTER BORDER-----//
void drawscreen(void)
  {
  int i,j;
  textcolor(YELLOW);
  textbackground(BLUE);
  for(i=1;i<=25;i++)
    for(j=1;j<=80;j++)
      {
      gotoxy(j,i);cprintf(" ");
      }
  for(i=2;i<=79;i++)
    {
    gotoxy(i,1);cprintf("═");
    gotoxy(i,23);cprintf("═");
    }
  for(i=2;i<=22;i++)
    {
    gotoxy(1,i);cprintf("║");
    gotoxy(80,i);cprintf("║");
    }
  _setcursortype(_NOCURSOR);
  }

//-------- DRAW COMMAND ON SCREEN ------------//
void cmdscreen()
  {
  gotoxy(1,1);cprintf("╔");
  gotoxy(80,1);cprintf("╗");
  gotoxy(1,23);cprintf("╚");
  gotoxy(80,23);cprintf("╝");
  textcolor(WHITE);
  gotoxy(25,6);cprintf(": Download File to EPROM");
  gotoxy(25,7);cprintf(": Enter ");
  gotoxy(25,8);cprintf(": Compare");
  gotoxy(25,9);cprintf(": Fill");
  gotoxy(25,10);cprintf(": Move");
  gotoxy(25,11);cprintf(": Search");
  gotoxy(25,12);cprintf(": Dump");
  gotoxy(25,13);cprintf(": Configurate");
  gotoxy(25,14);cprintf(": View Data in File");
  gotoxy(25,15);cprintf(": <Check Error> or <Not Check>");
  gotoxy(25,16);cprintf(": Exit");
  textcolor(YELLOW);
  gotoxy(25,3); cprintf("EASY LINK Version 2.01");
  gotoxy(23,4); cprintf("By. OveR handsome");
  gotoxy(20,6); cprintf("PgDn");
  gotoxy(20,7); cprintf("E");
  gotoxy(20,8); cprintf("C");
  gotoxy(20,9);cprintf("F");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  gotoxy(20,10);cprintf("M");
  gotoxy(20,11);cprintf("S");
  gotoxy(20,12);cprintf("D");
  gotoxy(20,13);cprintf("ALT+S");
  gotoxy(20,14);cprintf("ALT+L");
  gotoxy(20,15);cprintf("F5");
  gotoxy(20,16);cprintf("F10");
  }

//---------- CLEAR SCREEN IN USED  AREA -----//
void cls(void)
  {
  textcolor(YELLOW);
  textbackground(BLUE);
  for(int i=2;i<=22;i++)
    for(int j=2;j<=78;j++)
      {
      gotoxy(j,i);cprintf(" ");
      }
  }

//----------- FADE IN FOR EXIT -------//
void goodbye(void)
  {
  int xmin,xmax,ymin,ymax,x,y,xtemp,ytemp;
  xmin=1;xmax=80;ymin=1;ymax=25;
  textcolor(LIGHTGRAY);textbackground(BLACK);
  do
    {
    x=xmin;y=ymin;
    for(;;)
      {
      gotoxy(x,y);cprintf(" ");
      if(y<ymax)
      y++;
      else
      {
      xmin++;
      goto ex1;
      }
      }
    ex1:
    x=xmin;y=ymax;
    for(;;)
      {
      gotoxy(x,y);cprintf(" ");
      if(x<xmax)
      x++;
      else
      {
      ymax--;
      goto ex2;
      }
      }
    ex2:
    x=xmax;y=ymax;
    for(;;)
      {
      gotoxy(x,y);cprintf(" ");
      if(y>ymin)
      y--;
      else
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
        {
        xmax--;
        goto ex3;
        }
        }
      ex3:
      x=xmax;y=ymin;
      for(;;)
        {
        gotoxy(x,y);cprintf(" ");
        if(x>xmin)
        x--;
        else
        {
        ymin++;
        goto ex4;
        }
        }
      ex4:
      delay(20);
      }while((ymin!=ymax)&&(xmin!=xmax));
    }

  //--------- DIALOGUE FOR SUCCESS ACTIVITIES -----//
  void complete(char pass[26],int r)
    {

    textcolor(BLUE);textbackground(LIGHTGREEN);
    gotoxy(25,12);cprintf("▄▄▄▄");
    gotoxy(25,13);cprintf("▀▀▀▀");
    delay(30);
    gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄");
    gotoxy(25,13);cprintf("▀▀▀▀▀▀▀▀");
    delay(30);
    gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄▄▄▄▄");
    gotoxy(25,13);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀");
    gotoxy(25,14);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀");
    delay(30);
    gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄");
    gotoxy(25,13);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀");
    gotoxy(25,14);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀");
    delay(30);
    gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄");
    gotoxy(25,13);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀");
    gotoxy(25,14);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀");
    delay(30);
    gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄");
    gotoxy(25,13);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀");
    gotoxy(25,14);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀");
    textcolor(WHITE+BLINK);
    gotoxy(((26/2)-(r/2))+25,13);cprintf("%s",pass);
    textcolor(BLACK);textbackground(BLUE);
    gotoxy(53,13);cprintf("▀");
    gotoxy(53,14);cprintf("▀");
    gotoxy(53,15);cprintf("▀");

    for(int i=26;i<=53;i++)
      {
      gotoxy(i,15);cprintf("▀");
      }
    for(i=0;i<=8;i++)
      {
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
    sound(2000);delay(40);
    sound(1000);delay(20);
    }
  nosound();
  _setcursortype(_NOCURSOR);
  }

//------- DIALOGUE FOR FAIL ACTIVITIES -------//
void fail(char pass[26],int r)
  {

  textcolor(LIGHTRED);textbackground(RED);
  gotoxy(25,12);cprintf("▄▄▄▄");
  gotoxy(25,13);cprintf("████");
  delay(30);
  gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄");
  gotoxy(25,13);cprintf("████████");
  delay(30);
  gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄▄▄▄▄");
  gotoxy(25,13);cprintf("████████████");
  gotoxy(25,14);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀");
  delay(30);
  gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄");
  gotoxy(25,13);cprintf("████████████████");
  gotoxy(25,14);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀");
  delay(30);
  gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄");
  gotoxy(25,13);cprintf("████████████████████");
  gotoxy(25,14);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀");
  delay(30);
  gotoxy(25,12);cprintf("▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄");
  gotoxy(25,13);cprintf("████████████████████████");
  gotoxy(25,14);cprintf("▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀");
  textcolor(WHITE+BLINK);
  gotoxy(((26/2)-(r/2))+25,13);cprintf("%s",pass);
  textcolor(BLACK);textbackground(BLUE);
  gotoxy(53,13);cprintf("▄");
  gotoxy(53,14);cprintf("▄");
  gotoxy(53,15);cprintf("▄");

  for(int i=26;i<=53;i++)
    {
    gotoxy(i,15);cprintf("▄");
    }
  for(i=0;i<=8;i++)
    {
    sound(200);delay(40);
    sound(100);delay(20);
    }
  nosound();
  _setcursortype(_NOCURSOR);
  }


//---------- DIALOGUE FOR FAIL ACTIVITIES NUBER 2 ---//
void fail2(char pass[60],int r)
  {

  textcolor(LIGHTRED);textbackground(RED);
  gotoxy(17,10);cprintf("▄▄▄▄");
  gotoxy(17,11);cprintf("████");
  delay(30);
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
  gotoxy(17,10);cprintf("╔═══╗");
  gotoxy(17,11);cprintf("╚═══╝");
  delay(30);
  gotoxy(17,10);cprintf("╔═══════╗");
  gotoxy(17,11);cprintf("╚═══════╝");
  gotoxy(17,12);cprintf("");
  delay(30);
  gotoxy(17,10);cprintf("╔═══════════╗");
  gotoxy(17,11);cprintf("╚═══════════╝");
  gotoxy(17,12);cprintf("");
  delay(30);
  gotoxy(17,10);cprintf("╔═══════════════╗");
  gotoxy(17,11);cprintf("╚═══════════════╝");
  gotoxy(17,12);cprintf("");
  delay(30);
  gotoxy(17,10);cprintf("╔═══════════════════╗");
  gotoxy(17,11);cprintf("╚═══════════════════╝");
  gotoxy(17,12);cprintf("");
  delay(30);
  gotoxy(17,10);cprintf("╔═══════════════════════╗");
  gotoxy(17,11);cprintf("╚═══════════════════════╝");
  gotoxy(17,12);cprintf("");
  delay(30);
  gotoxy(17,10);cprintf("╔═══════════════════════════╗");
  gotoxy(17,11);cprintf("╚═══════════════════════════╝");
  gotoxy(17,12);cprintf("");
  textcolor(WHITE);textbackground(BLUE);
  gotoxy(17,9); cprintf("                    PORT ERROR");
  textcolor(WHITE+BLINK);textbackground(RED);
  gotoxy(((26/2)-(r/2))+25,11);cprintf("%s",pass);
  textcolor(BLACK);textbackground(BLUE);
  gotoxy(63,11);cprintf("▄");
  gotoxy(63,12);cprintf("▄");
  gotoxy(63,13);cprintf("▀");

  for(int i=18;i<=63;i++)
    {
    gotoxy(i,13);cprintf("▀");
    }
  for(i=0;i<=8;i++)
    {
    sound(200);delay(40);
    sound(100);delay(20);
    }
  nosound();
  _setcursortype(_NOCURSOR);
  }

//------ DIALOGUE FOR TELL USER THAT "YOU CAN'T USE THIS
//  BECAUSE IT IS  NOT DOS MODE "
void dosmode(void)
  {

  textcolor(WHITE);textbackground(WHITE);
  gotoxy(25,10);cprintf("╔═══╗");
  gotoxy(25,11);cprintf("╚═══╝");
  delay(30);
  gotoxy(25,10);cprintf("╔═══════╗");
  gotoxy(25,11);cprintf("╚═══════╝");
  delay(30);
  gotoxy(25,10);cprintf("╔═══════════╗");
  gotoxy(25,11);cprintf("╚═══════════╝");
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
gotoxy(25,11);cprintf("╚═══════╝");
delay(30);
gotoxy(25,10);cprintf("╔═══════╗");
gotoxy(25,11);cprintf("╚═══════╝");
gotoxy(25,12);cprintf("╔═══════╗");
delay(30);
gotoxy(25,10);cprintf("╔═══════════╗");
gotoxy(25,11);cprintf("╚═══════════╝");
gotoxy(25,12);cprintf("╔═══════════╗");
delay(30);
gotoxy(25,10);cprintf("╔═══════════════╗");
gotoxy(25,11);cprintf("╚═══════════════╝");
gotoxy(25,12);cprintf("╔═══════════════╗");
delay(30);
gotoxy(25,10);cprintf("╔═══════════════════╗");
gotoxy(25,11);cprintf("╚═══════════════════╝");
gotoxy(25,12);cprintf("╔═══════════════════╗");
textcolor(BLUE+BLINK);
gotoxy(27,11);cprintf(" DOS mode,can't do this");
textcolor(BLACK);textbackground(BLUE);
gotoxy(53,11);cprintf("▄");
gotoxy(53,12);cprintf("▀");
gotoxy(53,13);cprintf("▄");

for(int i=26;i<=53;i++)
  {
  gotoxy(i,13);cprintf("▀");
  }
for(i=0;i<=8;i++)
  {
  sound(200);delay(40);
  sound(100);delay(20);
  }
nosound();
_setcursortype(_NOCURSOR);
}
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
////////////////////////////////
///    Header file
////////////////////////////////
#define TOP 0
#define BOTTOM  9
#define UP  0x4800
#define DOWN       0x5000
#define LEFT       0x4b00
#define RIGHT      0x4d00
#define ESCAPE     0x011b
#define ENTER      0x1c0d
#define ALT_F4  0x6b00
#define DEL 0x5300
#define BKSP       0x0e08
#define SETTING 0x1f00  // ALT+S
#define PAGEUP  0x4900
#define PAGEDOWN 0x5100
#define EXIT     0x4400     // F10
#define _LOAD    0x266c     // L
#define _ENTER   0x1265     // E
#define _COMPARE 0x2e63     // C
#define _FILL    0x2166     // F
#define _MOVE    0x326d     // M
#define _SEARCH  0x1f73     // S
#define _DUMP    0x2064     // D
#define _LISTF   0x2600     // AKT+L
#define F5   0x3f00    // f5

#include<conio.h>
//#include<stdio.h>
#include<dos.h>
#include<stdlib.h>
#include<string.h>
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ค

รายละเอียดโปรแกรม QEDIT ซึ่งใช้ในการเขียนโปรแกรมแอสเซมบลี 8051 ข้อมูลทั้งหมดได้มาจากไฟล์ READ.ME ของโปรแกรม QEDIT

The SemWare Editor Junior (TSE Jr.) - version 4.0
Copyright 1985 - 1995 SemWare Corporation.
All rights reserved worldwide.

QEdit(R) has adopted a new name, The SemWare(R) Editor Junior (TSE Jr.). As you begin to explore this new version, you will discover that it is still the same editor that you have always known. In this version, there have been dozens of features added which bring the familiar QEdit more in line with its sister product - and now its namesake - The SemWare Editor Professional (TSE Pro).

TABLE OF CONTENTS

0 INTRODUCTION

This version of TSE Jr. is NOT public domain or free software, but is being distributed as "shareware" for EVALUATION PURPOSES ONLY.

SemWare grants a limited license to individuals to use this shareware software for a 30-day evaluation period on a private, non-commercial basis, for the express purpose of determining whether TSE Jr. is suitable for their needs. At the end of this 30-day evaluation period, the individual must either purchase a license from SemWare for continued use of the program, or discontinue using TSE Jr.

Many hours of work have gone into the development of TSE Jr - over 11

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

work-years, as of this version - and that does not include all the time spent on our users guide/reference manual. In addition, we provide full-time support, via phone during normal business hours, or mail, fax, or BBS if you prefer.

TSE Jr. is a professionally developed, packaged, and supported product. Our product has received excellent reviews, and our manual and product support are highly regarded by our licensed users. We offer discounts on updates to our registered users, and we regularly incorporate suggested changes into the product as we are able.

The only way we can continue to provide products like TSE Jr. and to offer outstanding technical support, is to stay in business; and the only way we can stay in business is for those who use TSE Jr. to purchase a license for it. We like the shareware concept and would like to stay a part of it. But we can do so only through paid licenses for TSE Jr. Please don't take this the wrong way - you may try TSE Jr. for 30 days (subject to the conditions and restrictions stated in TSEJR.DOC), under no obligation to pay to use it during that time. However, if you continue to use it after that 30-day evaluation period, you must purchase a license to do so. THANKS!

The SemWare Editor Junior software is owned by SemWare Corporation, or its suppliers, and is protected by United States copyright laws and international treaty provisions.

SemWare is a registered trademark of SemWare Corporation. All other trademarks and registered trademarks referenced in this file are the property of their respective owners. The QEdit registered trademark is used under license from Robelle Consulting Ltd.

For LICENSE, WARRANTY, U.S. GOVERNMENT RESTRICTED RIGHTS, EXPORT LAW
ASSURANCES, and GOVERNING LAW AND GENERAL PROVISIONS information, see the
documentation accompanying this software.

I SEMWARE NEWS

The SemWare Editor (in both its Professional and Junior versions) is extremely popular, with over 130,000 licensed customers throughout the U.S. and in more than 95 countries!

The SemWare Editor Junior for OS/2 is available with Long Filename and

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Extended Attribute support! If you want THE FASTEST editor available for OS/2, give SemWare a call!!

The SemWare Editor Junior / Mem-Res is AVAILABLE! For those of you who would
like to have access to your favorite editor at virtually anytime, in the middle of anything else, we at SemWare are proud to offer you a memory resident version of The SemWare Editor Junior. And best of all, if you have expanded memory (192k minimum) or extended memory (384k minimum), the memory
resident version will require only 9k of DOS memory! See the reference manual for more details.

QEdit/TSE Jr. is the winner of the Data Based Advisor Readers Choice Award for best text/program editor for 1989, 1991 and 1992, and received an honorable mention in 1994. We thank our users for this vote of confidence and continued support!

There are GERMAN versions available of The SemWare Editor products. All text displayed by the program (including the StatusLine!) is in German, and the reference manual was painstakingly translated by a native German. For information on ordering the German version, contact our German distributor, Manfred Luft, directly:

Manfred Luft SoftWare
Waldstr. 20a
D-79194 Gundelfingen
GERMANY

CIS: 100016,3715
Internet: 100016.3715@compuserve.com

	within Germany	from the US
Telephone:	0761-58 05 26	011-49-761-580526
Fax:	0761-58 05 47	011-49-761-580547
BBS:	0761-58 05 22	011-49-761-580522

II CONTENTS OF TSE JR DISTRIBUTION DISK

File	Description
Q.EXE	The editor program. This is the only file required to run TSE Jr.
QCONFIG.EXE	The configuration program.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

QCONFIG.DAT The standard Keyboard Definition File.
QHELP.TXT The standard Help text.

READ.ME The file you are currently reading.
TSEJR.DOC The documentation for the shareware version of TSE Jr.
MACRO.DOC Macro Reference guide.
HELPME.DOC Common Questions and Answers regarding TSE Jr.
QTIPS.DOC Helpful TSE Jr. tips compiled by SemWare Tech support.
FILE_ID.DIZ TSE Jr. description for bbs.
VENDOR.DOC Restrictions/Authorizations for Disk Vendor distribution.
ORDERFRM.DOC Handy order from.

Notes

The ONLY file required to run TSE Jr.is Q.EXE - the others are not required by the program.

III TECHNICAL SUPPORT

Because of the tremendous popularity of The SemWare Editor Junior (formerly QEdit), we can give technical support ONLY to licensed, registered users. Please have your SERIAL NUMBER handy when you call.

Please! Before you call us with a problem, make sure you have read this entire READ.ME file to determine if it solves your problem. If you should need to contact us, please provide or have available the following information:

- The SemWare Editor Junior Version number and Serial number
- OS/2, Windows, and DOS versions
- Computer model
- Amount of RAM
- The names of any memory-resident programs you have loaded

You may obtain technical support directly from SemWare via any of the following:

1. Internet

FTP ftp.semware.com
WWW http://www.semware.com

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. Send email to:

   InterNet tech.support@semware.com
   CIS 75300,2710
   Fido 1:133/314
   RIME ->SemWare (QEdit or RIME Admin)
   ->330 (QEdit or RIME Admin)

   On InterNet, we now have full FTP access so that macros, tips, demos, and many other utilities can be downloaded. We are building TSE-specific mailing lists that will allow us to deliver future product information directly to your email mailbox.

   We can also be reached in the SemWare/QEdit echoes on the following networks:

   Fido, ILink, Intellec, MetroLink, FringeNet, U'NI-net, and W-Net.

3. Call our remote electronic bulletin board service. The board is operational 24 hours a day, 7 days a week. Modem settings are <N81>, 14.4K bps. The BBS number is: (770) 641-8968.

4. CompuServe (GO SEMWARE, Section 6)

5. Send us a Fax: (770) 640-6213

6. Write to us:

   SemWare Corporation
   Attn: TSE Jr. Technical Support
   4343 Shallowford Road, Suite C3A
   Marietta, GA 30062-5022 USA

7. Call us directly at SemWare (770) 641-9002. Support hours are 9 am to 5 pm (ET), Monday through Friday.

SemWare is committed to supporting registered users. However, we request that you reserve telephone support for questions or problems requiring immediate attention.

As always, if you have a suggestion that you think would improve our products, do not hesitate to let us know about it.

IV VERSION 4.0 - RELEASE NOTES

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Important Notes

ù Multi-line macros are supported by QCONFIG, in the QCONFIG.DAT file. Also, the MacroBegin statement is optional. Because of these additional features, a couple of rules must be followed:

- Key assignments must begin in column 1.

- Lines two and greater of a multi-line macro must start in column 2 or greater.

- Text starting in column 1 (other than a comment) is assumed to be the start of a key assignment.

ù To take advantage of the commands assigned to the enhanced keyboard keys (^CursorUp, ^CursorDown, f11, f12, etc), the A)dvanced configuration option, "Test for presence of enhanced keyboard" must be set to "Y" by the QCONFIG program.

New Features in TSE Jr. 4.0

þ Integrated Mouse Support

Mouse support is now built-in, so QM.COM is no longer needed.

QConfig: (under Advanced Options)

Enable mouse processing (Y/N)? [Y] : Y
You must have a mouse driver loaded to use the mouse. (This option is not necessary with the OS/2 version)

Left-handed mouse (Y/N)? [N] : N
Swaps the outside buttons on the mouse.

Mouse hold time (in 1/18th sec) [1..36] [9] : 9
This is the number of clock-ticks (1/18th of a second) in which the mouse button must be down to be considered "HELD".

Mouse repeat-delay factor [0..32767] [600] : 600
This is a machine dependent value that can be adjusted to speed up or

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

slow down the repeat rate of the mouse buttons.

NOTE: You must also set the "Display Boxed" option (under Colors/screen) to TRUE, to enable scroll-bars on the editing windows.

Left-Click on top line of the screen invokes the MainMenu

In text area of current window:
Double-Left-Click marks the word
Triple-Left-Click marks the line
Click-Drag mouse to mark character block
Hold Alt-key while click-drag to mark a line block
Hold Ctrl-key while click-drag to mark a column block
Right-Click invokes the "Mouse Menu"

Resize windows by click-drag on StatusLine.

þ MouseMenu <Ctrl F10>
Pops up a menu of Block-related commands. This command provides access to a list of Block-related commands for easy selection using a mouse. If the mouse is not used, this menu can still be accessed by pressing <Ctrl F10>.

The commands available are:

```
Ú---------- Mouse Menu ----------¿
³ Cut                            ³
³ Copy                           ³
³ Paste                          ³
³ Paste Over                     ³
³ UnMark                         ³
Ã--------------------------------´
³ Copy to Windows Clipboard      ³
³ Paste from Windows Clipboard   ³
Ã--------------------------------´
³ Copy Block                     ³
³ Copy Over Block                ³
³ Move Block                     ³
³ Delete Block                   ³
Ã--------------------------------´
³ Fill Block                     ³
³ Sort                           ³
Ã--------------------------------´
³ Upper                          ³
³ Lower                          ³
³ Flip                           ³
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

À----------Ù

þ WinClipCopy, WinClipPaste
These two new commands allow The SemWare Editor Junior to access the Microsoft Windows or the OS/2 Clipboard. WinClipCopy will copy the current block to the Windows or OS/2 Clipboard, and WinClipPaste will paste the contents of the Windows or OS/2 Clipboard into the file at the current cursor location.

þ DirTree <Esc><F><I>
This command displays a pick list of all the drives on your system, along with a directory tree for the current drive. To switch to a different directory, move the cursor bar to the desired entry in the tree, and press <Enter>. To change to a different drive, move the cursor bar to the desired drive, and press <Enter>. The selected drive becomes the current drive, and the directory tree for that drive is displayed in the pick list.

þ LocateFile <Esc><F><A>
This command searches an entire drive for a specified filename. A list of all matching filenames is displayed, allowing you to select an appropriate file to edit.

When you execute this command, it prompts you for the filename for which to search. A complete name or an ambiguous name (with DOS wildcard characters) can be specified. By default, the current drive is searched. However, a different drive can be searched by prefacing the filename with the drive name (such as, d:foo). The default or specified drive is then searched, and all matching filenames are displayed in a pick list. To select the desired file to edit, position the cursor bar on that filename in the pick list, and press <Enter>.

If you assign DirTree or LocateFile to a key, you can execute these commands within any file-related prompt, such as "File(s) to edit:".

þ CUA-style block marking
QConfig options:
Use CUA-style (shift cursor) block marking (Y/N)? [Y] :
When ON, shifted keypad keys will perform CUA-style block marking commands.

Should Blocks remain marked after CUA marking (Y/N)? [N] :
Determines whether a block marked using the CUA-style marking keys will remain marked after you type a non-CUA-style block marking or command key.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Commands:

ToggleCUAMarking

Interactively toggles CUA-style block marking mode.

SetCUAMarking

Interactively turn ON CUA-style block marking mode.

NOTE: CUA-style block marking mode only activates the shifted cursor keys. If you wish to have the Cut, Copy, and Paste commands tied to the CUA keys, #del, ^ins, and #ins, you will need to manually modify your QCONFIG.DAT file to make the appropriate assignments.

þ IncrementalSearch <Ctrl I>

This is a specialized search command. It causes the editor to begin searching for a string while you type. As you enter each new character, the editor attempts to locate a string in the text that matches the incremented search string.

As you type characters, a case-insensitive, forward search is performed, beginning at the text position where the cursor was located when you invoked IncrementalSearch. As each matching string is found, the string is highlighted in the text.

þ New Find and FindReplace options:

These options are placed in the "Options:" prompt, not in the search string prompt.

[^] Anchor the search string to the beginning of the line (or Block, if the [L] option is also selected).

[$] Anchor the search string to the end of the line (or Block, if the [L] option is also selected).

þ Synchronized Scrolling

ToggleSyncScroll <Ctrl O><Y>

This command switches Synchronized Scrolling mode ON and OFF. If switched ON, and there are multiple windows on the screen, all other windows will be scrolled when the current window is scrolled. When Synchronized Scrolling mode is ON, an "S" appears on the StatusLine.

þ MaximizeWindow <Ctrl O><M>

This command causes the current window to be made as large as possible, by making all other windows as small as possible.

þ NextEqualIndent, PrevEqualIndent

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

These commands move the cursor to the next or previous line (as appropriate) whose text starts in the same column as the current line. If the current line is blank, the cursor moves to the next or previous blank line.

þ SwapChars
This command will swap the character at the cursor position with the character preceding the cursor. The cursor column position does not change.

þ SwapWords
This command will swap the word in which the cursor is positioned with the previous word on the current line. Any "non-word" characters between the affected words are preserved.

þ SwapLines <Ctrl F2> or <Esc><E><P>
This command will swap the current line with the line immediately following it. The cursor does not change its position relative to the screen.

þ isWord
Returns TRUE if the character at the cursor position in the file is in the current word set. See the AltWordSet command for more information on word sets.

þ SetPrintAddFF
þ TogglePrintAddFF
Toggles (ON or OFF) the option to automatically send a formfeed character to the printer at the end of each print operation.

þ SetVGA28
þ ToggleVGA28
Toggles a VGA monitor between 25-line and 28-line mode. This command has no effect if the video card is not a VGA type.

þ SetPromptForEAs
þ TogglePromptForEAs
Toggles (ON and OFF) prompting by the editor for a .TYPE EA when saving a file that does not currently have any Extended Attributes assigned. This includes newly-created files, and existing files without Extended Attributes. If ON, the editor prompts you for a .TYPE EA when you save a file that has no Extended Attributes assigned. If OFF, the editor does not prompt for a .TYPE EA.

þ GotoFirstLine and GotoLastLine
Positions the cursor on the beginning line, or the last line of the current

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

file. The column position is not changed.

þ Some additional commands:

- SetBakups Sets ON file Backup mode.

- SetBoxDraw Sets ON Box Drawing mode.

- SetCenterFinds Sets ON Find centering (to center found text, located by the Find and FindReplace commands, vertically on the screen).

- SetEnterMatching Sets ON EnterMatching mode.

- SetSwap Sets ON the option to swap to EMS or Disk when the Shell and DOS commands are executed. (This command is NOT available in the memory-resident or OS/2 versions.)

- SetTabsExpand Sets ON Physical Tab Expansion mode.

- SetTabsOut Sets ON Tabs Out mode.

þ The AsciiChart will now highlight the character under the cursor. If the cursor is on or past the end of line, then the first character in the chart (ASCII 0) will be highlighted.

þ The Match command will now find matching "<" and ">" characters.

þ The BegFile and EndFile commands will now keep the cursor in the same column if a column block is in progress.

þ New SwapPath which includes path as well as drive for swapping.
In QConfig, instead of just a swap drive, you can now specify a complete path for swapping.

þ Added the characters '»' (129) through '¥' (165) to both the DefaultWordSet and the AltWordSet.

þ Maximum line length increased to 2032.

þ Help text now compressed to allow more text for help screen.
The QuickHelp command will still only display one screen full. But the compression should allow you to put more info into the larger screen modes (such as 50-line mode).

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

þ New method of searching for macros.
When a macro is loaded, the editor searches in the following places:

Current directory
"Supplemental Files Path" from QConfig
Directory from which editor was loaded

þ CopyBlock command works in prompts.
CopyBlock MUST be the only command assigned to the desired key.

þ Two new attributes:

Menu Quick Letter - this is the letter to press to select the menu item
Menu Select Quick Letter - this is the quick letter of the currently selected menu item

þ New keys now available: (from the keypad)

NOTE: Any assignments to these keys are ignored when the CUA-style block marking option is ON. The CUA-style block marking can be set in QConfig.

grey/
#1
#2
#3
#4
#6
#7
#8
#9

Miscellaneous Usage Notes

Methods to Speed Up Keyboard Delays:

If you are running DOS 5 or greater, you can use the following DOS command:

mode con: rate=32 delay=1

See your DOS reference manual for additional details.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภาคผนวก ง

การใช้งานโปรแกรม SXA51 ASSEMBLER

1. รายละเอียดทั่วไปของ SXA51 ASSEMBLER

SXA51เป็นโปรแกรม 8051/52 ครอสแอสเซมเบลอร์ ใช้สำหรับทำการแปล SOURCE PROGRAM ซึ่งเขียนเป็นภาษาแอสเซมบลี้ของ MCS-51 (FILE.ASM) และได้เอาท์พุทไฟล์เป็น INTEL HEX FILE (FILE.HEX) การเขียน SOURCE PROGRAM ให้กระทำจากโปรแกรม EDITOR ใดๆ ก็ได้ จากนั้นจึงนำมาแปลด้วย SXA51 ซึ่งทำได้โดยการเรียกใช้โปรแกรม SXA51.EXE และตามด้วยชื่อ SOURCE PROGRAM ที่เขียนขึ้น การแปลของ SXA51 จะกระทำในแบบ 2 PASS ASSEMBLER ซึ่งเมื่อมีข้อผิดพลาดใดๆ SXA51 ก็จะแสดงให้ทราบในขณะที่แปล ลักษณะของ SOURCE PROGRAM จะเป็นไปตามมาตราฐานของการเขียนภาษา ASSEMBLY ทั่วไป

2. การใช้งานโปรแกรม SXA51 ASSEMBLER

โปรแกรมแอสเซมเบลอร์ที่ใช้งานคือ SXA51.EXE โดยแอสเซมเบลอร์จะทำการแปล SOURCE FILE ซึ่งถูกกำหนดไว้เป็น FILE.ASM เมื่อทำการแปลแล้ว จะได้ไฟล์เอาท์พุทเป็น INTEL-HEX FILE ซึ่งถูกกำหนดไว้เป็น FILE.HEX โดย FILE.HEX นี่เอง ที่ผู้ใช้สามารถนำไปโหลดเข้าบอร์ดต่างๆ หรืออัดเข้าตัว EPROM เพื่อนำไปทดสอบโปรแกรมตามที่เขียนได้ นอกจากนี้ SXA51 ยังสามารถสร้าง LISTING FILE ต่างๆ ได้ตามรูปแบบของ OPTION ดังนี้

A>SXA51 <-OPTION> SOURCEFILE

SOURCEFILE : คือชื่อไฟล์ของโปรแกรม กรณีที่ไม่ใส่นามสกุลของไฟล์ ตัว SXA51 จะกำหนดให้ .ASM โดยอัตโนมัติ

<-OPTION> : เป็นส่วนของการกำหนด OPTION ต่างๆ ประกอบด้วยเครื่องหมาย "-" และตามด้วยอักษรเพียงตัวเดียว การกำหนด OPTION นี้สามารถทำได้มากกว่า 1 ตัวในครั้งเดียว รายละเอียดของ OPTION จะมีดังนี้

- -L หมายถึงให้สร้างไฟล์แสดงรายละเอียด (LISTING FILE) โดยไฟล์เอาท์พุทที่ได้คือ FILE.LST
- -N หมายถึงให้ทำการคอมไพล์ โดยไม่สร้างไฟล์เอาท์พุทใดๆ ทั้งนี้เพื่อการตรวจสอบความถูกต้องของโปรแกรมอย่างรวดเร็ว
- -C หมายถึงให้สร้างไฟล์แสดงรายละเอียด เช่นเดียวกับ -L โดยจะเพิ่ม symbol cross-reference ไว้ที่ท้ายของไฟล์ ทั้งนี้เพื่อความสะดวกในการค้นหาตำแหน่งต่างๆ ตามชื่อ LABEL

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- -D ให้แสดง PROCESS ขณะที่กำลังทำการ ASSEMBLER โดยโปรแกรมจะแสดงเลขที่บรรทัดที่กำลังแปลอยู่

ตัวอย่างการใช้คำสั่ง

A>SXA51 J31EX4

หมายถึง SOURCE : J31EX4.ASM
LISTING : ไม่สร้าง
HEX : สร้าง HEX FILE ชื่อ J31EX4.HEX

A>SXA51 -L J31EX4.ASM

หมายถึง SOURCE : J31EX4.ASM
LISTING : สร้าง LISTING FILE ชื่อ J31EX4.LST
HEX : สร้าง HEX FILE ชื่อ J31EX4.HEX

A>SXA51 -N J31EX4.ASM

หมายถึง SOURCE : J31EX4.ASM
LISTING : ไม่สร้าง
HEX : ไม่สร้าง

3. ข้อกำหนดและรูปแบบของ source file

3.1ในแต่ละบรรทัดต้องมีคำสั่งในภาษาแอสเซมบลี้ หรือ ASSEMBLER DIRECTIVE เพียงหนึ่งคำสั่งเท่านั้น

3.2 LABEL DECLARATION ทุกเลเบลต้องปิดท้ายด้วย ":"

3.3 โปรแกรม SXA51 จะมองตัวอักษรตัวเล็กหรือตัวใหญ่เหมือนกัน เช่น hh: จะเหมือนกับ HH:

3.4 บรรทัดว่างในโปรแกรมต้นแบบจะยังคงปรากฏอยู่ในไฟล์แสดงรายละเอียด (FILE.LST) แต่ในไฟล์แสดงรหัสภาษาเครื่อง (FILE.HEX) จะถูกตัดทิ้งไป

3.5 SOURCE FILE ในแต่ละบรรทัดต้องประกอบด้วยตัวอักษรไม่เกิน 128 ตัวอักษร ซึ่งในแต่ละบรรทัดต้องมีรูปแบบดังนี้

[LABEL:] OPERATION [OPERAND1][,OPERAND2] [;COMMENT]

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- LABEL เป็นกลุ่มของตัวอักษรและตัวเลขที่จะใช้เป็นตัวแปร หรือตำแหน่งหน่วยความจำสำหรับอ้างอิงในโปรแกรม LABEL ต้องขึ้นต้นด้วยตัวอักษร A ถึง Z หรือ "_" "." "?" และต้องปิดท้ายด้วยเครื่องหมาย ":" เสมอ
- OPERATION ส่วนนี้จะเป็นคำสั่งมาตรฐานของ 8051 เช่นคำสั่ง MOV INC DEC ADD
- OPERAND เป็นกลุ่มของตัวอักษรและตัวเลขซึ่งอาจจะอยู่ในรูปของการกระทำทางคณิตศาสตร์หรือการกระทำทางด้าน LOGIC

COMMENT หมายถึงประโยคหรือข้อความที่ตามหลังเครื่องหมาย ";" โดยทั่วไปมักจะใช้อธิบายการทำงานของโปรแกรมสั้นๆ เพื่อให้ทำความเข้าใจได้ง่ายขึ้นเครื่องหมาย ";" จะอยู่ที่ใดก็ได้ข้อความที่ตามหลังเครื่องหมาย ";" จะไม่มีผลต่อการแปลของ ASSEMBLER

4. Assembler Directives

DB expression [,expression...]

DB (DEFINE BYTE) คำสั่งนี้ใช้กำหนดค่าข้อมูลเป็นไบท์ลงในหน่วยความจำโปรแกรม สามารถใส่ข้อความรหัสแอสกี้ (ASCII STRING) ลงตำแหน่งในหน่วยความจำได้ โดยข้อความนั้นต้องอยู่ในเครื่องหมาย QUOTES "'" และยังกำหนดค่าข้อมูลที่เป็นตัวเลขและตัวอักษรในบรรทัดเดียวกันได้โดยการคั่นด้วยเครื่องหมาย "," เช่น

```
POINTER DB 'This is a string',0
```

DS expression

DS (DEFINE SPACE) ใช้สำหรับกำหนดที่ว่างเป็นจำนวนไบท์สำหรับเก็บค่าตัวแปรหรือใช้เป็นส่วนของ BUFFER

DW expession [,expression...]

DW (DEFINE WORD) ใช้ในการเขียนข้อมูล 16 บิท ของหน่วยความจำโปรแกรมโดยการกำหนดค่าข้อมูลไบท์ต่ำก่อนแล้วตามด้วยข้อมูลไบท์สูง โดยข้อมูลแต่ละ WORD ต้องคั่นด้วยเครื่องหมาย "," เช่นเดียวกับคำสั่ง DB

ORG expression

ORG (ORIGIN) ใช้สำหรับกำหนดค่าเริ่มต้นของโปรแกรม (ORIGIN PROGRAM LOCATION) โดยกำหนดค่าแอดเดรสของหน่วยความจำ

END

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

END คำสั่งนี้มีไว้เพื่อบอกให้ ASSEMBLER ทราบถึงจุดสิ้นสุดของโปรแกรมว่าอยู่ที่ตำแหน่งใด คำสั่งนี้จำเป็นต้องใส่ไว้ท้ายโปรแกรมด้วย เพราะถ้าไม่ใส่อาจทำให้ผิดพลาดในขั้นตอนการแปลได้

SYMBOL_NAME EQU expression
SYMBOL_NAME BIT expression
สำหรับการกำหนดค่าให้กับชื่อ SYMBOL_NAME (หรือที่เรียกกันว่า LABEL) โดยสามารถใช้คำสั่ง EQU หรือคำสั่ง BIT ก็ได้ ซึ่งจะทำงานเหมือนกันทุกประการ

SYMBOL_NAME SET expression
คำสั่ง SET จะทำงานเหมือนกับ EQU ทุกประการ แต่จะสามารถกำหนดให้ SYMBOL_NAME มีค่าได้หลาย ๆ จุดตามที่ต้องการ

5. Expressions

การกำหนดค่าใน expression ของแต่ละคำสั่งในภาษา ASSEMBLER นั้น สามารถใส่ตัว OPERATORต่างๆ เพื่อความสะดวกในการใช้งานได้ รวมทั้งยังกำหนดเป็นเลขฐานต่างๆ ได้ด้วย โดยสรุปได้ดังนี้

- โดยปกติตัวเลขจะถือว่าเป็นเลขฐานสิบเสมอ (DECIMAL) แต่ถ้าต้องการกำหนดเป็นเลขฐานสิบหก (HEXADECIMAL) จะต้องเขียนนำหน้าด้วยตัวเลขและลงท้ายด้วยอักษร H เช่น 12H, 0F7H และถ้าต้องการกำหนดเป็นเลขฐานแปด (OCTAL) ให้ลงท้ายด้วยอักษร O หรือ Q และถ้าต้องการเป็นเลขฐานสอง (BINARY) ให้ลงท้ายด้วยอักษร B
- สัญญาลักษณ์ $ จะหมายถึงการอ้างตำแหน่งของ ADDRESS ในบรรทัดนั้นๆ
- สามารถใช้คำว่า HIGH หรือ LOW เพื่อกำหนดค่าเฉพาะส่วนสูงหรือต่ำได้ เช่นถ้ามีการกำหนดค่าให้ชื่อ TEST EQU 1234H จะสามารถเรียกค่าส่วนสูงได้คือ MOV A,#HIGH TEST ซึ่งจะทำให้ได้ค่า 12H ตามต้องการ
- สามารถกำหนด OPERATOR ในการคำนวนได้คือ + - * / ^ (exponentation)
- สามารถกระทำทางด้าน LOGIC ได้คือ MOD SHR SHL AND OR XOR

6. Bit Operations

SXA51 มีความสามารถในการอ้างค่าตัวแปรในระดับ BIT ได้ โดยใช้หลักการ n.m ซึ่ง n คือค่า BYTE ADDRESS และ m คือหมายเลข BIT ทั้งนี้จะต้องใช้กับค่าที่กำหนดเป็น BIT ADDRESSABLE เท่านั้น เช่น 22H.3 จะมีค่าเท่ากับ 13H รวมทั้งจะใช้กับชื่อมาตราฐานได้ด้วย เช่น ACC.7 P1.2

7. SXA51 Error Meassages

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Bype

B Value out-of-bounds Must be between 0 and 255

Control

C1 A primary control must be used before any code generation

C2 Unrecognized control

Expression

E1 Unbalanced parentheses

E2 connot translate variable name

E3 Devide by zero

E4 Incorrect bit address

E5 General expression error

E6 Radix error

E7 Bit.n: n must be between 0 and 7

Include

I Include nesting too deep

Multiply defind

M1 Label multiply defined

M2 EQU or SET multiply defined

Offset error

O1 ACALL or AJMP must be inside 4096 byte 'page'

O2 SJMP must be within +127 or-128

Syntex

S General

S1 First char. on line must be alphabetic

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

S2 Label error: propably a reserved word

S3 first char. (after label) must be alphabelic

S4 Reserved word in EQU or SET

S5 Incorrect use of opcode

S6 Cannot evaluate ORG expression

S7 Incorrect register or data type

S8 Cannot DEC DPTR

S9 Expected 'A'

S10 Expected comma

S11 Expected 'AB'

S12 Expected '@A+DPTR'

S13 Expected '@A+PC' or '@A+DPTR'

S14 Expected 'A,@DPTR','A,@R0' or 'A,@R0'

S15 Expected '@R0' or '@R1'

S16 SETB operand must be Bit or C

S17 String must be enclosed in quotes or is too long

Unrecognized

U General

U0 Opcode

8. ข้อแนะนำการใช้งาน SXA51

BUG ของ SXA51 ที่ต้องหลีกเลี่ยง ในกรณีที่ผู้ใช้ใช้คำสั่ง MOV BUFFER,#35H โดยชื่อ BUFFER คือส่วนหนึ่งในโปรแกรมที่ได้ทำการกำหนดเอาไว้ด้วยคำสั่งเทียม DS หรือคำสั่งเทียม EQU ในกรณีนี้ถ้าการกำหนดชื่ออยู่ก่อนการใช้อ้างอิงก็จะไม่มีปัญหาอะไร แต่ถ้าอยู่หลังจากการใช้อ้างอิงจะเกิดปัญหาการแปลไม่ถูกต้องและที่สำคัญคือ ไม่มีการบอก ERROR ให้ผู้ใช้ได้ทราบ ดังนั้นขอให้จำหลักการง่ายๆ ดังนี้ คือ การกำหนดค่าให้กับ LABEL ไม่ว่าจะจากคำสั่งเทียม DS หรือ EQU ก็ตาม ให้กระทำที่ต้นโปรแกรมเสมอ ซึ่งก็จะใช้งานไปได้อย่างดี

เทคนิคการใช้ตัวแปรสตริง $ คือตัวแปรที่ใช้แทนค่า ADDRESS ของบรรทัดนั้นๆ ซึ่งจะทำให้ผู้ใช้สามารถอ้างอิงถึง ADDRESS ในระยะใกล้ๆ ได้ โดยที่ไม่ต้องกำหนด LABEL ให้มากเกินความจำเป็น

ตัวอย่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
CJNE    A,#1,$+6        ;ถ้า Aไม่เท่ากับ 1 ให้ JUMP ไปอีก 6 BYTE
LJMP    MAIN1           ;คือไปทำคำสั่ง CJNE ต่อไปนั่นเอง
CJNE    A,#2,$+6
LJMP    MAIN2
CJNE    A,#3,$+6
LJMP    MAIN3
LJMP    MAIN4
```

ความสามารถในการอ้าง LABEL เฉพาะส่วนกรณีที่ผู้ใช้กำหนด LABEL ดังนี้ TEST EQU 147FH ชื่อ TEST จะมีค่าตัวแปรเป็น 16 BIT และถ้าผู้ใช้ต้องการอ้างอิงแต่ละ BYTE ของชื่อนี้ ก็จะสามารถทำได้ดังนี้

```
MOV  R2,#HIGH TEST      ;จะได้ค่าเท่ากับ 14H
MOV  R3,#LOW TEST       ;จะได้ค่าเท่ากับ 7FH
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Features

- Compatible with MCS-51™ Products
- 4 Kbytes of In-System Reprogrammable Flash Memory
  Endurance: 1,000 Write/Erase Cycles
- Fully Static Operation: 0 Hz to 24 MHz
- Three-Level Program Memory Lock
- 128 x 8-Bit Internal RAM
- 32 Programmable I/O Lines
- Two 16-Bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial Channel
- Low Power Idle and Power Down Modes

Description

The AT89C51 is a low-power, high-performance CMOS 8-bit microcomputer with 4 Kbytes of Flash Programmable and Erasable Read Only Memory (PEROM). The device is manufactured using Atmel's high density nonvolatile memory technology and is compatible with the industry standard MCS-51™ instruction set and pinout. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C51 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications.

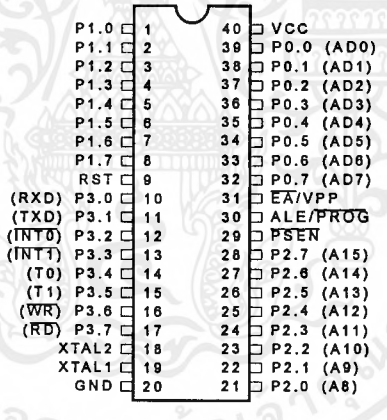
(continued)

8-Bit Microcontroller with 4 Kbytes Flash

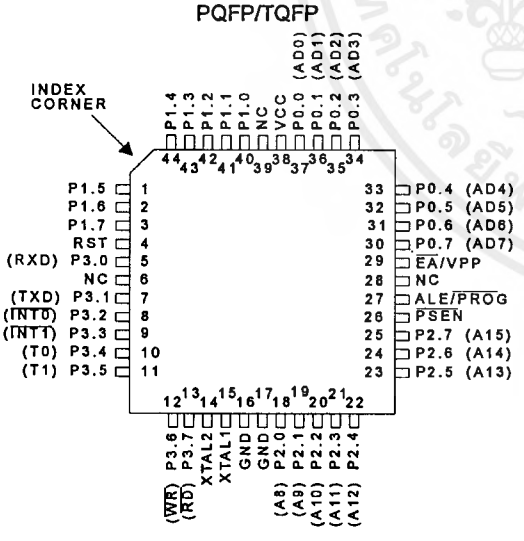
AT89C51

Pin Configurations

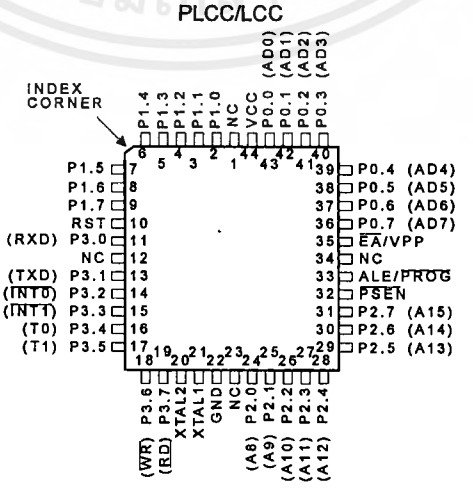
PDIP/Cerdip



PQFP/TQFP



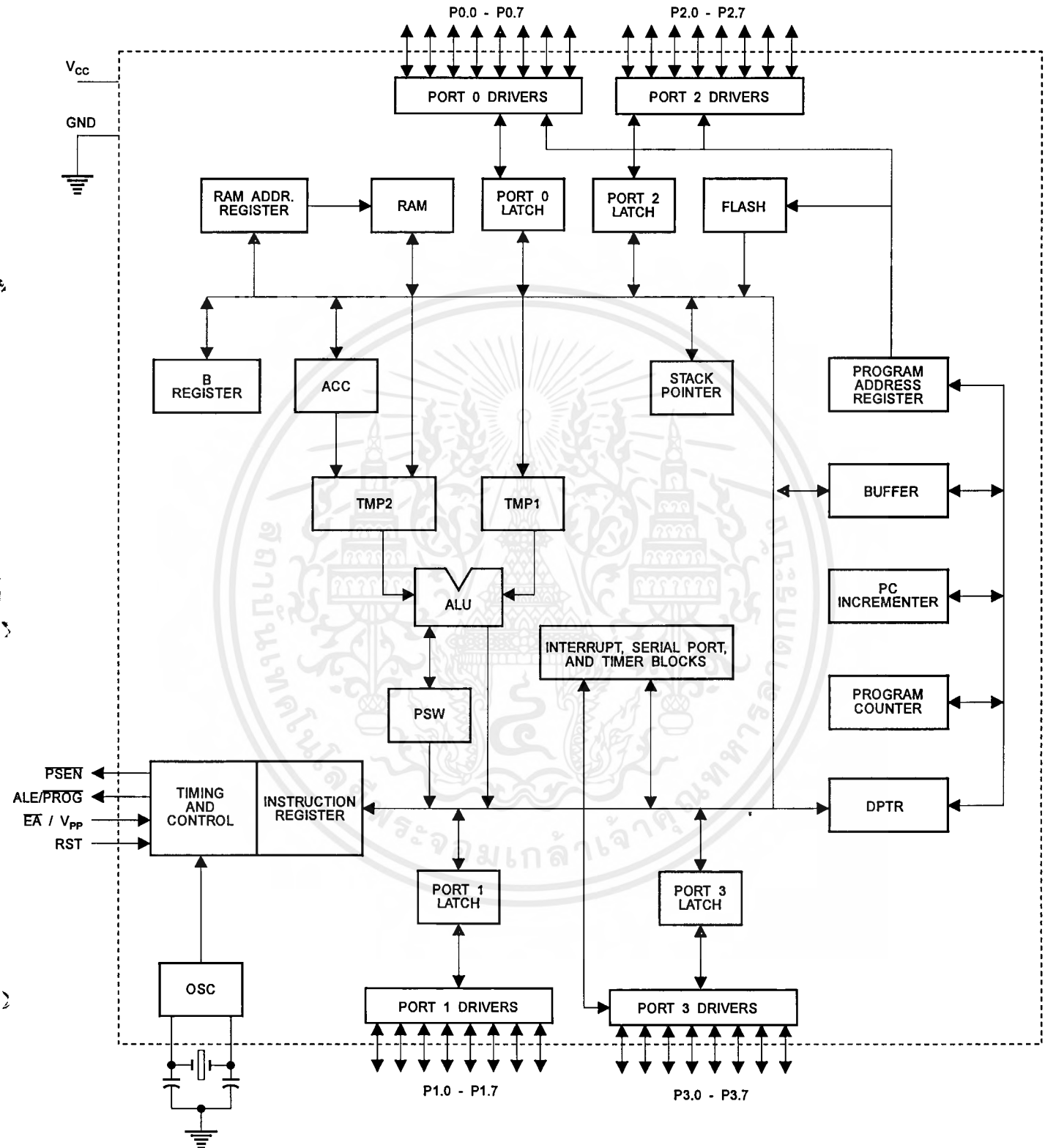
PLCC/LCC



0265E

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Block Diagram



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Description (Continued)

The AT89C51 provides the following standard features: 4 Kbytes of Flash, 128 bytes of RAM, 32 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator and clock circuitry. In addition, the AT89C51 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The Power Down Mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

Pin Description

V_{CC}

Supply voltage.

GND

Ground.

Port 0

Port 0 is an 8-bit open drain bidirectional I/O port. As an output port each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

Port 0 may also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode P0 has internal pullups.

Port 0 also receives the code bytes during Flash programming, and outputs the code bytes during program verification. External pullups are required during program verification.

Port 1

Port 1 is an 8-bit bidirectional I/O port with internal pullups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

Port 1 also receives the low-order address bytes during Flash programming and program verification.

Port 2

Port 2 is an 8-bit bidirectional I/O port with internal pullups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application it uses strong internal pullups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3

Port 3 is an 8-bit bidirectional I/O port with internal pullups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pullups.

Port 3 also serves the functions of various special features of the AT89C51 as listed below:

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (extenal interrupt 0)
P3.3	$\overline{INT1}$ (extenal interrupt 1)
P3.4	T0 (timer 0 extenal input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{WR}$ (extenal data memory write strobe)
P3.7	$\overline{RD}$ (external data memory read strobe)

Port 3 also receives some control signals for Flash programming and programming verification.

RST

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device.

ALE/$\overline{PROG}$

Address Latch Enable output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input ($\overline{PROG}$) during Flash programming.

In normal operation ALE is emitted at a constant rate of 1/6 the oscillator frequency, and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external Data Memory.

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcrontroller is in external execution mode.

$\overline{PSEN}$

Program Store Enable is the read strobe to external program memory.

(continued)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Pin Description (Continued)

When the AT89C51 is executing code from external program memory, PSEN is activated twice each machine cycle, except that two PSEN activations are skipped during each access to external data memory.

$\overline{EA}$/V_{PP}

External Access Enable. $\overline{EA}$ must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, $\overline{EA}$ will be internally latched on reset.

$\overline{EA}$ should be strapped to V_{CC} for internal program executions.

This pin also receives the 12-volt programming enable voltage (V_{PP}) during Flash programming, for parts that require 12-volt V_{PP}.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier.

Oscillator Characteristics

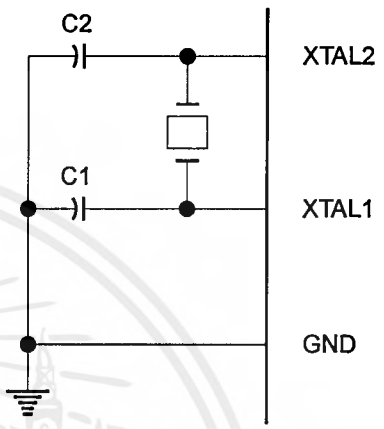
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 2. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

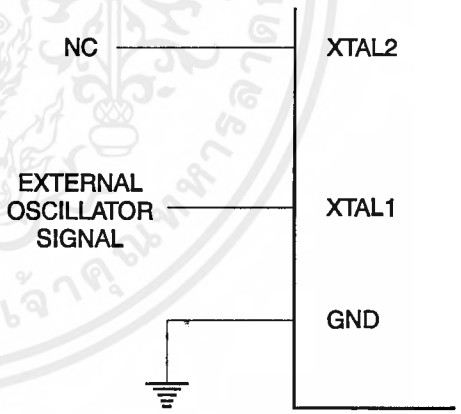
It should be noted that when idle is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hard-

Figure 1. Oscillator Connections



Notes: C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 2. External Clock Drive Configuration



Status of External Pins During Idle and Power Down

Mode	Program Memory	ALE	PSEN	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power Down	Internal	0	0	Data	Data	Data	Data
Power Down	External	0	0	Float	Data	Data	Data

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when Idle is terminated by reset, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external memory.

Power Down Mode

In the power down mode the oscillator is stopped, and the instruction that invokes power down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the power down mode is terminated. The only exit from power down is a hardware reset. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before Vcc is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

Program Memory Lock Bits

On the chip are three lock bits which can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the table below:

When lock bit 1 is programmed, the logic level at the $\overline{EA}$ pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value, and holds that value until reset is activated. It is necessary that the latched value of $\overline{EA}$ be in agreement with the current logic level at that pin in order for the device to function properly.

Lock Bit Protection Modes

Program Lock Bits				
	LB1	LB2	LB3	Protection Type
1	U	U	U	No program lock features.
2	P	U	U	MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory, EA is sampled and latched on reset, and further programming of the Flash is disabled.
3	P	P	U	Same as mode 2, also verify is disabled.
4	P	P	P	Same as mode 3, also external execution is disabled.

Programming the Flash

The AT89C51 is normally shipped with the on-chip Flash memory array in the erased state (that is, contents = FFH) and ready to be programmed. The programming interface accepts either a high-voltage (12-volt) or a low-voltage (Vcc) program enable signal. The low voltage programming mode provides a convenient way to program the AT89C51 inside the user's system, while the high-voltage programming mode is compatible with conventional third party Flash or EPROM programmers.

The AT89C51 is shipped with either the high-voltage or low-voltage programming mode enabled. The respective top-side marking and device signature codes are listed in the following table.

	V_{PP} = 12 V	V_{PP} = 5 V
Top-Side Mark	AT89C51 xxxx yyww	AT89C51 xxxx-5 yyww
Signature	(030H)=1EH (031H)=51H (032H)=FFH	(030H)=1EH (031H)=51H (032H)=05H

The AT89C51 code memory array is programmed byte-by-byte in either programming mode. *To program any non-blank byte in the on-chip Flash Memory, the entire memory must be erased using the Chip Erase Mode.*

Programming Algorithm: Before programming the AT89C51, the address, data and control signals should be set up according to the Flash programming mode table and Figures 3 and 4. To program the AT89C51, take the following steps.

1. Input the desired memory location on the address lines.
2. Input the appropriate data byte on the data lines.
3. Activate the correct combination of control signals.
4. Raise $\overline{EA}/V_{PP}$ to 12 V for the high-voltage programming mode.
5. Pulse ALE/$\overline{PROG}$ once to program a byte in the Flash array or the lock bits. The byte-write cycle is self-timed and typically takes no more than 1.5 ms. Repeat steps 1 through 5, changing the address and data for the entire array or until the end of the object file is reached.

Data Polling: The AT89C51 features $\overline{Data}$ Polling to indicate the end of a write cycle. During a write cycle, an at-

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(continued)

Programming the Flash (Continued)

tempted read of the last byte written will result in the complement of the written datum on PO.7. Once the write cycle has been completed, true data are valid on all outputs, and the next cycle may begin. Data Polling may begin any time after a write cycle has been initiated.

Ready/$\overline{\text{Busy}}$: The progress of byte programming can also be monitored by the RDY/$\overline{\text{BSY}}$ output signal. P3.4 is pulled low after ALE goes high during programming to indicate BUSY. P3.4 is pulled high again when programming is done to indicate READY.

Program Verify: If lock bits LB1 and LB2 have not been programmed, the programmed code data can be read back via the address and data lines for verification. The lock bits cannot be verified directly. Verification of the lock bits is achieved by observing that their features are enabled.

Chip Erase: The entire Flash array is erased electrically by using the proper combination of control signals and by holding ALE/$\overline{\text{PROG}}$ low for 10 ms. The code array is written with all "1"s. The chip erase operation must be executed before the code memory can be re-programmed.

Reading the Signature Bytes: The signature bytes are read by the same procedure as a normal verification of locations 030H,

031H, and 032H, except that P3.6 and P3.7 must be pulled to a logic low. The values returned are as follows.

(030H) = 1EH indicates manufactured by Atmel
(031H) = 51H indicates 89C51
(032H) = FFH indicates 12 V programming
(032H) = 05H indicates 5 V programming

Programming Interface

Every code byte in the Flash array can be written and the entire array can be erased by using the appropriate combination of control signals. The write operation cycle is self-timed and once initiated, will automatically time itself to completion.

All major programming vendors offer worldwide support for the Atmel microcontroller series. Please contact your local programming vendor for the appropriate software revision.

Flash Programming Modes

Mode		RST	$\overline{\text{PSEN}}$	ALE/$\overline{\text{PROG}}$	EA/V_{PP}	P2.6	P2.7	P3.6	P3.7
Write Code Data		H	L	⎍	H/12V(1)	L	H	H	H
Read Code Data		H	L	H	H	L	L	H	H
Write Lock	Bit - 1	H	L	⎍	H/12V	H	H	H	H
	Bit - 2	H	L	⎍ (2)	H/12V	H	H	L	L
	Bit - 3	H	L	⎍	H/12V	H	L	H	L
Chip Erase		H	L	⎍	H/12V	H	L	L	L
Read Signature Byte		H	L	H	H	L	L	L	L

Notes: 1. The signature byte at location 032H designates whether V_{PP} = 12 V or V_{PP} = 5 V should be used to enable programming.

2. Chip Erase requires a 10 ms $\overline{\text{PROG}}$ pulse.

เอกสารนี้เป็นเอกสาร... ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Figure 3. Programming the Flash

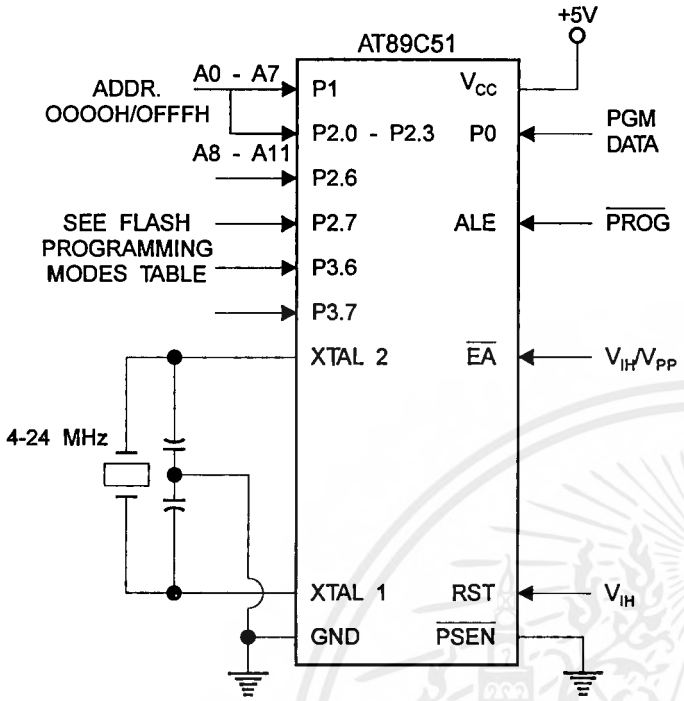
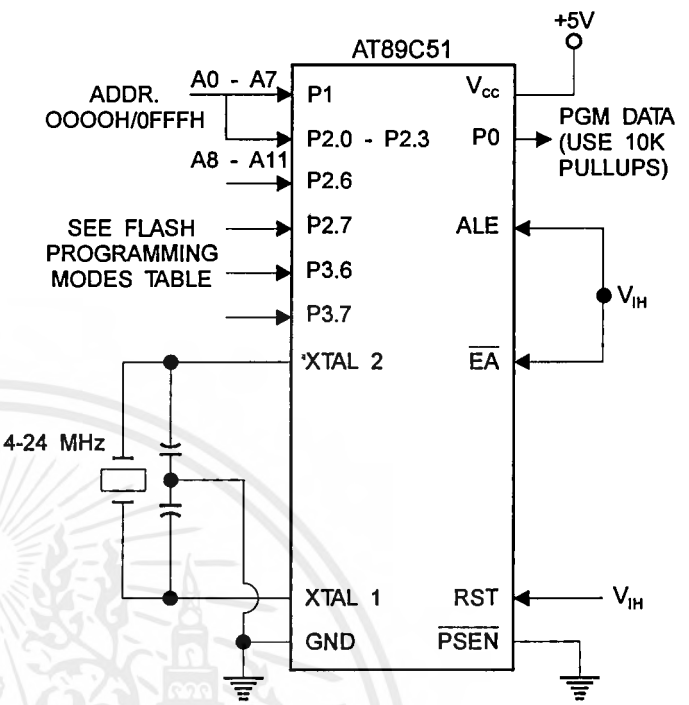


Figure 4. Verifying the Flash



Flash Programming and Verification Characteristics

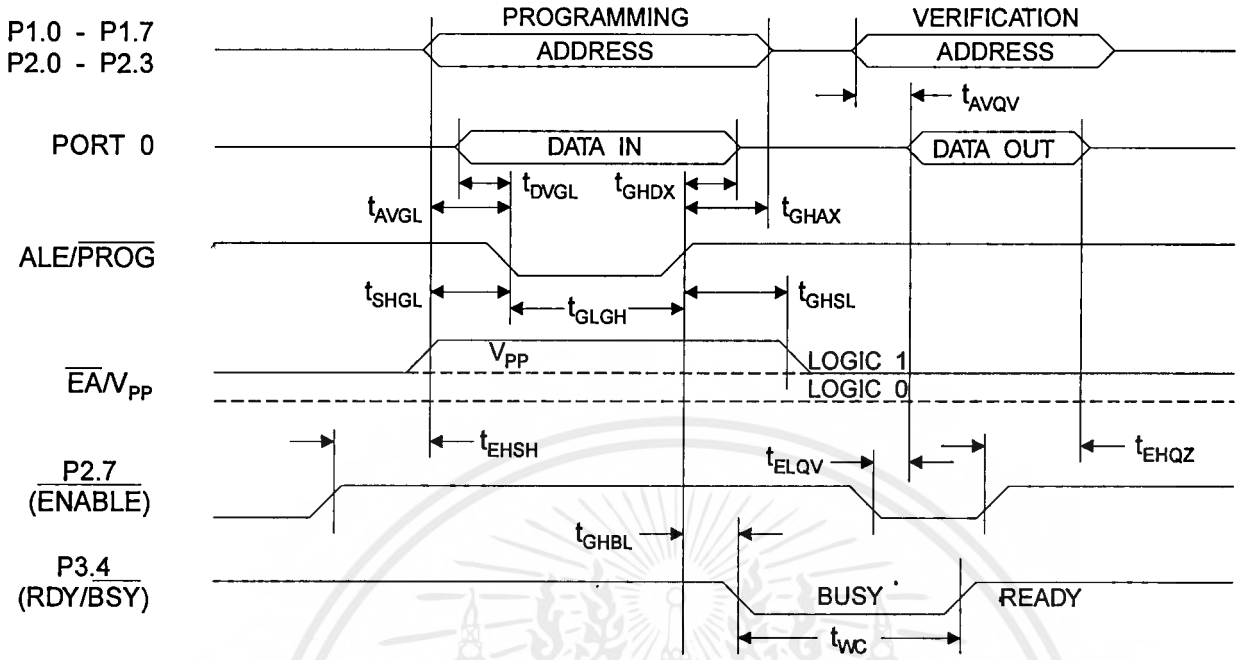
T_A = 21°C to 27°C, V_{CC} = 5.0 ± 10%

Symbol	Parameter	Min	Max	Units
V_{PP}(1)	Programming Enable Voltage	11.5	12.5	V
I_{PP}(1)	Programming Enable Current		1.0	mA
$1/t_{CLCL}$	Oscillator Frequency	4	24	MHz
t_{AVGL}	Address Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
t_{GHAX}	Address Hold After $\overline{PROG}$	$48t_{CLCL}$		
t_{DVGL}	Data Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
t_{GHDX}	Data Hold After $\overline{PROG}$	$48t_{CLCL}$		
t_{EHSH}	P2.7 ($\overline{ENABLE}$) High to V_{PP}	48tCLCL		
t_{SHGL}	V_{PP} Setup to $\overline{PROG}$ Low	10		µs
t_{GHSL}(1)	V_{PP} Hold After $\overline{PROG}$	10		µs
t_{GLGH}	$\overline{PROG}$ Width	1	110	µs
t_{AVQV}	Address to Data Valid		48tCLCL	
t_{ELQV}	$\overline{ENABLE}$ Low to Data Valid		$48t_{CLCL}$	
t_{EHQV}	Data Float After $\overline{ENABLE}$	0	$48t_{CLCL}$	
t_{GHBL}	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	µs
t_{WC}	Byte Write Cycle Time		2.0	ms

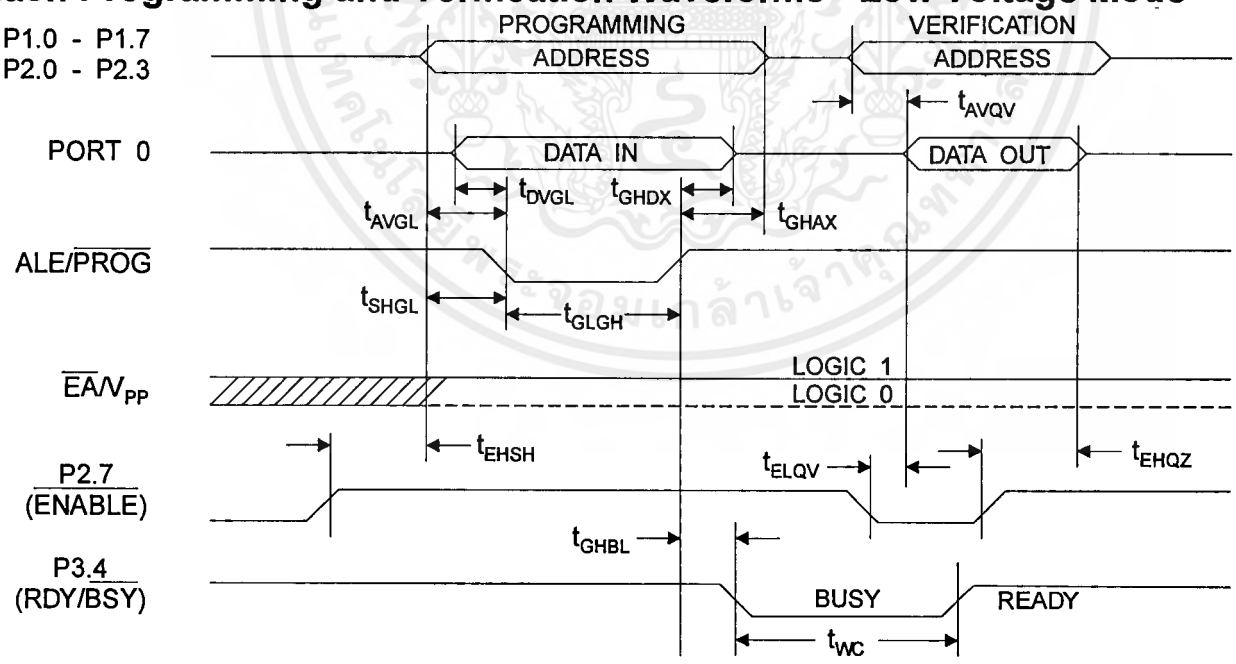
Note: 1. Only used in 12-volt programming mode.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Flash Programming and Verification Waveforms - High Voltage Mode



Flash Programming and Verification Waveforms - Low Voltage Mode



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin with Respect to Ground	-1.0 V to +7.0 V
Maximum Operating Voltage	6.6 V
DC Output Current	15.0 mA

*NOTICE: Stresses beyond 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 beyond 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 = -40°C to 85°C, V_{CC} = 5.0 V ± 20% (unless otherwise noted)

Symbol	Parameter	Condition	Min	Max	Units
V_{IL}	Input Low Voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC}-0.1	V
V_{IL1}	Input Low Voltage ($\overline{EA}$)		-0.5	0.2 V_{CC}-0.3	V
V_{IH}	Input High Voltage	(Except XTAL1, RST)	0.2 V_{CC}+0.9	V_{CC}+0.5	V
V_{IH1}	Input High Voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC}+0.5	V
V_{OL}	Output Low Voltage(1) (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
V_{OL1}	Output Low Voltage(1) (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
V_{OH}	Output High Voltage (Ports 1,2,3, ALE, $\overline{PSEN}$)	I_{OH} = -60 µA, V_{CC} = 5 V ± 10%	2.4		V
		I_{OH} = -25 µA	0.75 V_{CC}		V
		I_{OH} = -10 µA	0.9 V_{CC}		V
V_{OH1}	Output High Voltage (Port 0 in External Bus Mode)	I_{OH} = -800 µA, V_{CC} = 5 V ± 10%	2.4		V
		I_{OH} = -300 µA	0.75 V_{CC}		V
		I_{OH} = -80 µA	0.9 V_{CC}		V
I_{IL}	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45 V		-50	µA
I_{TL}	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2 V		-650	µA
I_{LI}	Input Leakage Current (Port 0, EA)	0.45 < V_{IN} < V_{CC}		±10	µA
RRST	Reset Pulldown Resistor		50	300	KΩ
C_{IO}	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
I_{CC}	Power Supply Current	Active Mode, 12 MHz		20	mA
		Idle Mode, 12 MHz		5	mA
	Power Down Mode(2)	V_{CC} = 6 V		100	µA
		V_{CC} = 3 V		40	µA

Notes: 1. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
Maximum I_{OL} per port pin:10 mA
Maximum I_{OL} per 8-bit port:
Port 0:26 mA
Ports 1,2, 3:15 mA
Maximum total IOL for all output pins:71 mA
If IOL exceeds the test condition, VOL may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.
2. Minimum VCC for Power Down is 2 V.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

A.C. Characteristics

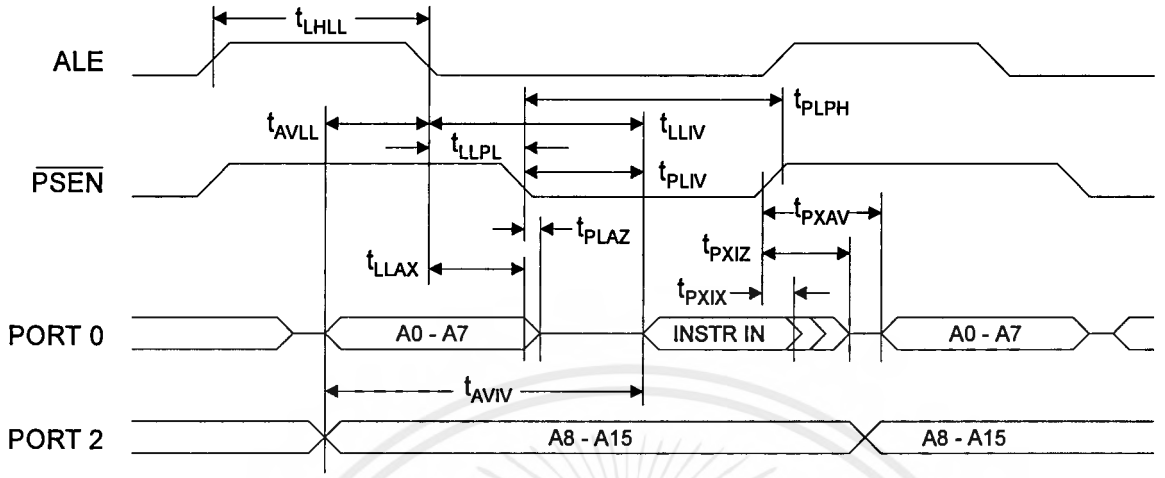
(Under Operating Conditions; Load Capacitance for Port 0, ALE/$\overline{PROG}$, and $\overline{PSEN}$ = 100 pF; Load Capacitance for all other outputs = 80 pF)

External Program and Data Memory Characteristics

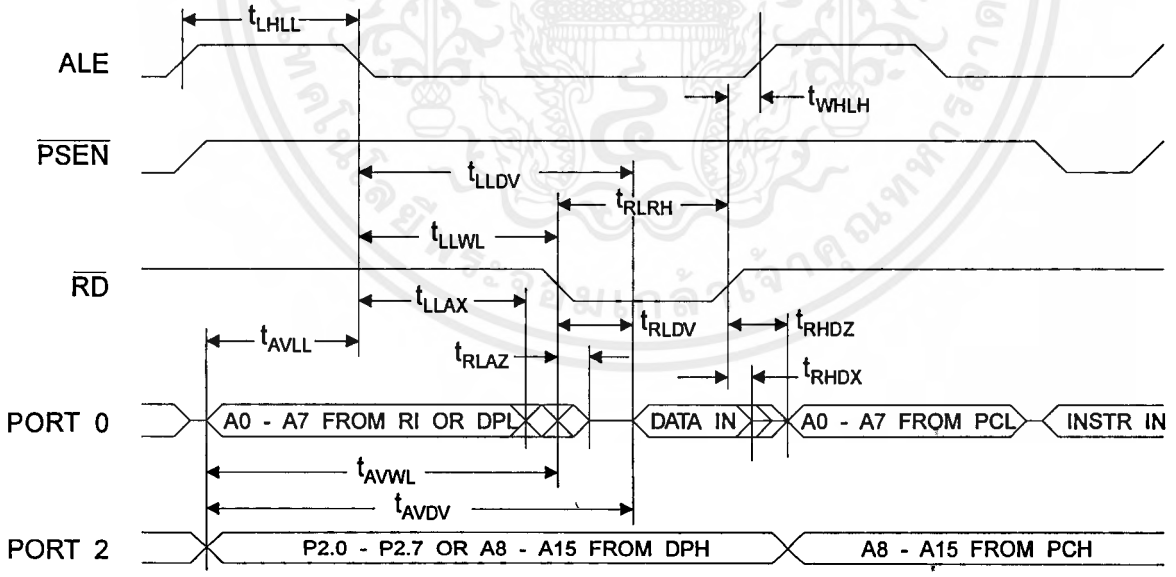
Symbol	Parameter	12 MHz Oscillator		16 to 24 MHz Oscillator		Units
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator Frequency			0	24	MHz
t_{LHLL}	ALE Pulse Width	127		$2t_{CLCL}$-40		ns
t_{AVLL}	Address Valid to ALE Low	28		tCLCL-13		ns
t_{LLAX}	Address Hold After ALE Low	48		tCLCL-20		ns
t_{LLIV}	ALE Low to Valid Instruction In		233		4tCLCL-65	ns
t_{LLPL}	ALE Low to $\overline{PSEN}$ Low	43		t_{CLCL}-13		ns
t_{PLPH}	$\overline{PSEN}$ Pulse Width	205		$3t_{CLCL}$-20		ns
t_{PLIV}	$\overline{PSEN}$ Low to Valid Instruction In		145		$3t_{CLCL}$-45	ns
t_{PXIX}	Input Instruction Hold After $\overline{PSEN}$	0		0		ns
t_{PXIZ}	Input Instruction Float After $\overline{PSEN}$		59		t_{CLCL}-10	ns
t_{PXAV}	$\overline{PSEN}$ to Address Valid	75		t_{CLCL}-8		ns
t_{AVIV}	Address to Valid Instruction In		312		$5t_{CLCL}$-55	ns
t_{PLAZ}	$\overline{PSEN}$ Low to Address Float		10		10	ns
t_{RLRH}	$\overline{RD}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{WLWH}	$\overline{WR}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{RLDV}	$\overline{RD}$ Low to Valid Data In		252		$5t_{CLCL}$-90	ns
t_{RHDX}	Data Hold After $\overline{RD}$	0		0		ns
t_{RHDZ}	Data Float After $\overline{RD}$		97		$2t_{CLCL}$-28	ns
t_{LLDV}	ALE Low to Valid Data In		517		8tCLCL-150	ns
t_{AVDV}	Address to Valid Data In		585		9tCLCL-165	ns
t_{LLWL}	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	200	300	$3t_{CLCL}$-50	$3t_{CLCL}$+50	ns
t_{AVWL}	Address to $\overline{RD}$ or $\overline{WR}$ Low	203		$4t_{CLCL}$-75		ns
t_{QVWX}	Data Valid to $\overline{WR}$ Transition	23		t_{CLCL}-20		ns
t_{QVWH}	Data Valid to $\overline{WR}$ High	433		$7t_{CLCL}$-120		ns
t_{WHQX}	Data Hold After $\overline{WR}$	33		t_{CLCL}-20		ns
t_{RLAZ}	$\overline{RD}$ Low to Address Float		0		0	ns
t_{WHLH}	$\overline{RD}$ or $\overline{WR}$ High to ALE High	43	123	t_{CLCL}-20	t_{CLCL}+25	ns

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

External Program Memory Read Cycle

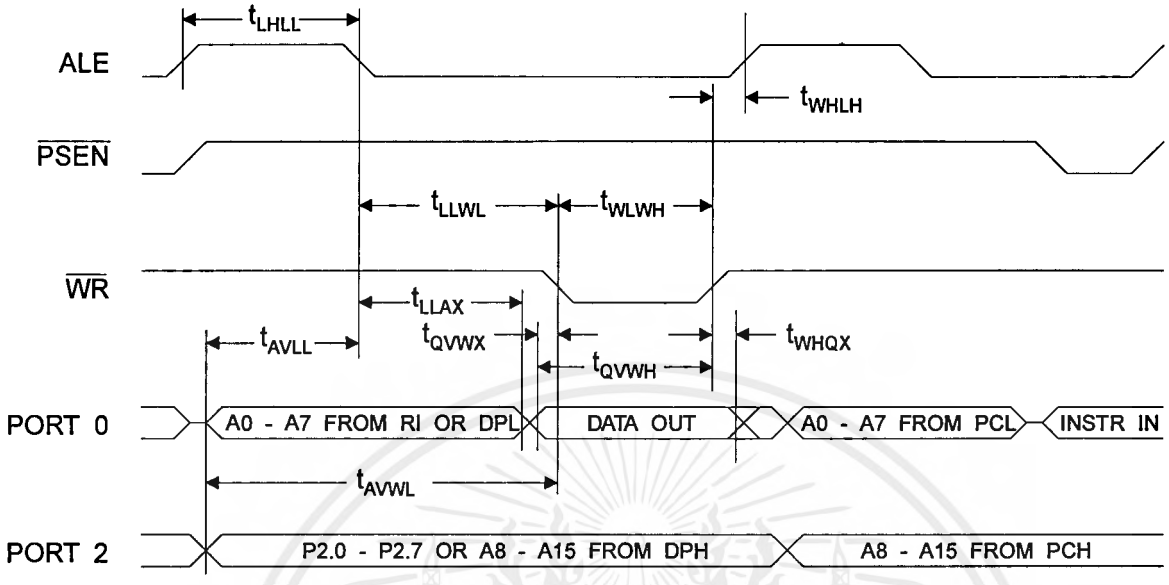


External Data Memory Read Cycle

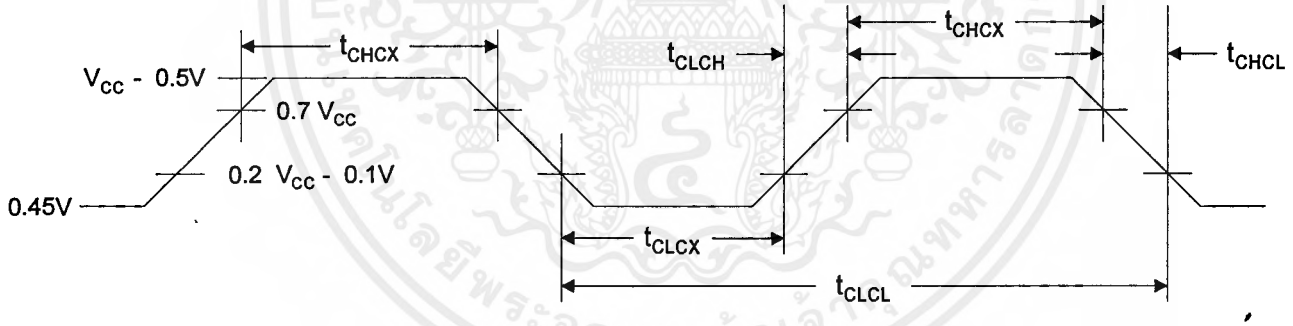


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

External Data Memory Cycle



External Clock Drive Waveforms



External Clock Drive

Symbol	Parameter	Min	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
t_{CLCL}	Clock Period	41.6		ns
t_{CHCX}	High Time	15		ns
t_{CLCX}	Low Time	15		ns
t_{CLCH}	Rise Time		20	ns
t_{CHCL}	Fall Time		20	ns

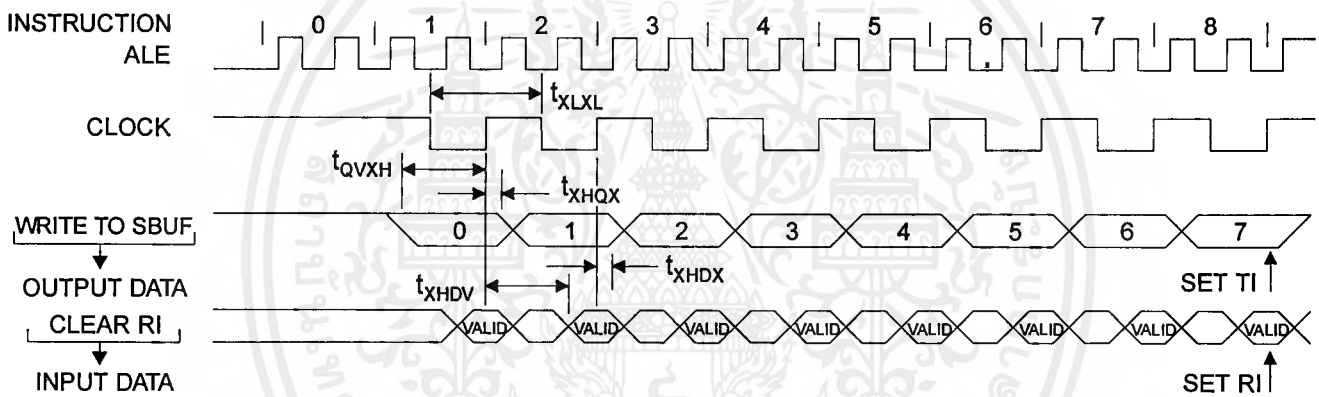
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Serial Port Timing: Shift Register Mode Test Conditions

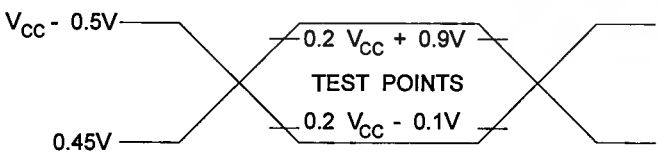
(V_{CC} = 5.0 V ± 20%; Load Capacitance = 80 pF)

Symbol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
t_{XLXL}	Serial Port Clock Cycle Time	1.0		$12t_{CLCL}$		µs
t_{QVXH}	Output Data Setup to Clock Rising Edge	700		$10t_{CLCL}$-133		ns
t_{XHQX}	Output Data Hold After Clock Rising Edge	50		$2t_{CLCL}$-33		ns
t_{XHDX}	Input Data Hold After Clock Rising Edge	0		0		ns
t_{XHDV}	Clock Rising Edge to Input Data Valid		700		10tCLCL-133	ns

Shift Register Mode Timing Waveforms

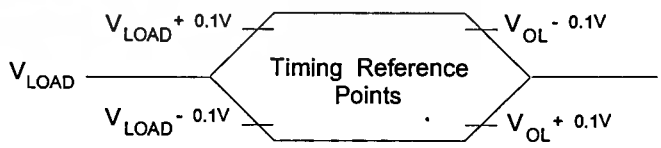


AC Testing Input/Output Waveforms[1]



Note: 1. AC Inputs during testing are driven at V_{CC} - 0.5 V for a logic 1 and 0.45 V for a logic 0. Timing measurements are made at V_{IH} min. for a logic 1 and V_{IL} max. for a logic 0.

Float Waveforms[1]



Note: 1. For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
12	5 V ± 20%	AT89C51-12AC AT89C51-12JC AT89C51-12PC AT89C51-12QC	44A 44J 40P6 44Q	Commercial (0°C to 70°C)
		AT89C51-12AI AT89C51-12JI AT89C51-12PI AT89C51-12QI	44A 44J 40P6 44Q	Industrial (-40°C to 85°C)
		AT89C51-12AA AT89C51-12JA AT89C51-12PA AT89C51-12QA	44A 44J 40P6 44Q	Automotive (-40°C to 125°C)
	5 V ± 10%	AT89C51-12DM AT89C51-12LM	40D6 44L	Military (-55°C to 125°C)
		AT89C51-12DM/883 AT89C51-12LM/883	40D6 44L	Military/883C Class B, Fully Compliant (-55°C to 125°C)
16	5 V ± 20%	AT89C51-16AC AT89C51-16JC AT89C51-16PC AT89C51-16QC	44A 44J 40P6 44Q	Commercial (0°C to 70°C)
		AT89C51-16AI AT89C51-16JI AT89C51-16PI AT89C51-16QI	44A 44J 40P6 44Q	Industrial (-40°C to 85°C)
		AT89C51-16AA AT89C51-16JA AT89C51-16PA AT89C51-16QA	44A 44J 40P6 44Q	Automotive (-40°C to 125°C)
20	5 V ± 20%	AT89C51-20AC AT89C51-20JC AT89C51-20PC AT89C51-20QC	44A 44J 40P6 44Q	Commercial (0°C to 70°C)
		AT89C51-20AI AT89C51-20JI AT89C51-20PI AT89C51-20QI	44A 44J 40P6 44Q	Industrial (-40°C to 85°C)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Ordering Information

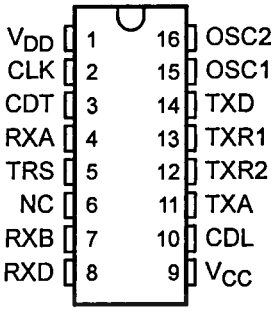
Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	5 V ± 20%	AT89C51-24AC AT89C51-24JC AT89C51-24PC AT89C51-24QC	44A 44J 44P6 44Q	Commercial (0°C to 70°C)
		AT89C51-24AI AT89C51-24JI AT89C51-24PI AT89C51-24QI	44A 44J 44P6 44Q	Industrial (-40°C to 85°C)

Package Type	
44A	44 Lead, Thin Plastic Gull Wing Quad Flatpack (TQFP)
40D6	40 Lead, 0.600" Wide, Non-Windowed, Ceramic Dual Inline Package (Cerdip)
44J	44 Lead, Plastic J-Leaded Chip Carrier (PLCC)
44L	44 Pad, Non-Windowed, Ceramic Leadless Chip Carrier (LCC)
40P6	40 Lead, 0.600" Wide, Plastic Dual Inline Package (PDIP)
44Q	44 Lead, Plastic Gull Wing Quad Flatpack (PQFP)

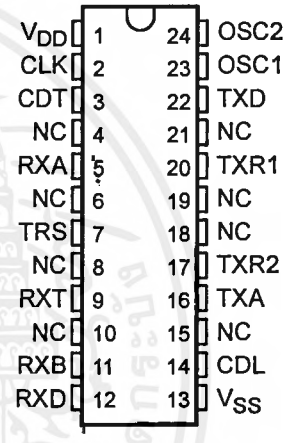
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **Single-Chip Frequency-Shift-Keying (FSK) Modem**
- **Meet Both Bell 202 and CCITT V23 Specifications**
- **Transmit Modulation at 75, 150, 600, and 1200 Baud**
- **Receive Demodulation at 5, 75, 150, 600, and 1200 Baud**
- **Half-Duplex Operation Up to 1200 Baud Transmit and Receive**
- **Full-Duplex Operation Up to 1200 Baud Transmit and 150 Baud Receive**
- **On-Chip Group Equalization and Transmit/Receive Filtering**
- **Carrier-Detect-Level Adjustment and Carrier-Fail Output**
- **Single 5-V Power Supply**
- **Low Power Consumption**
- **Reliable CMOS Silicon-Gate Technology**

J OR N PACKAGE
(TOP VIEW)



DW PACKAGE
(TOP VIEW)



NC – No internal connection

D package are available taped and reeled. Add the R suffix to device type (e.g., YCM3105DWLR).

description

The TCM3105 is a single-chip asynchronous frequency-shift-keying (FSK) voice-band modem that uses silicon-gate CMOS technology to implement a switched-capacitor architecture. It is pin selectable (TXR1, TXR2, and TRS) for a wide range of transmit/receive baud rates and is compatible with the applicable BELL 202 or CCITT V23 standards. Operation is fully reversible, thereby allowing both forward and backward channels to be used simultaneously.

The transmitter is a programmable frequency synthesizer that provides two output frequencies (on TXA), representing the marks and spaces of the digital signal present on TXD.

The receive section is responsible for the demodulation of the analog signal appearing at the RXA input and is based on the principle of frequency-to-voltage conversion. This section contains a group delay equalizer (to correct phase distortion), automatic gain control, carrier-detect-level adjustment, and bias-distortion adjustment, thereby optimizing performance and giving the lowest possible bit error rate.

Carrier-detect information is given to the system by means of the carrier-detect circuits, which set a flag on the CDT output if the level of received in-band energy falls below a value set on the CDL input for a specified minimum duration.

The TCM3105JE and TCM3105NE are characterized for operation from –40°C to 85°C. The TCM3105DWL, TCM3105JL, and TCM3105NL are characterized for operation from 0°C to 70°C.

Caution. These devices have limited built-in protection. The leads should be shorted together or the device placed in conductive foam during storage or handling to prevent electrostatic damage to the MOS gates.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

Copyright © 1994, Texas Instruments Incorporated

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

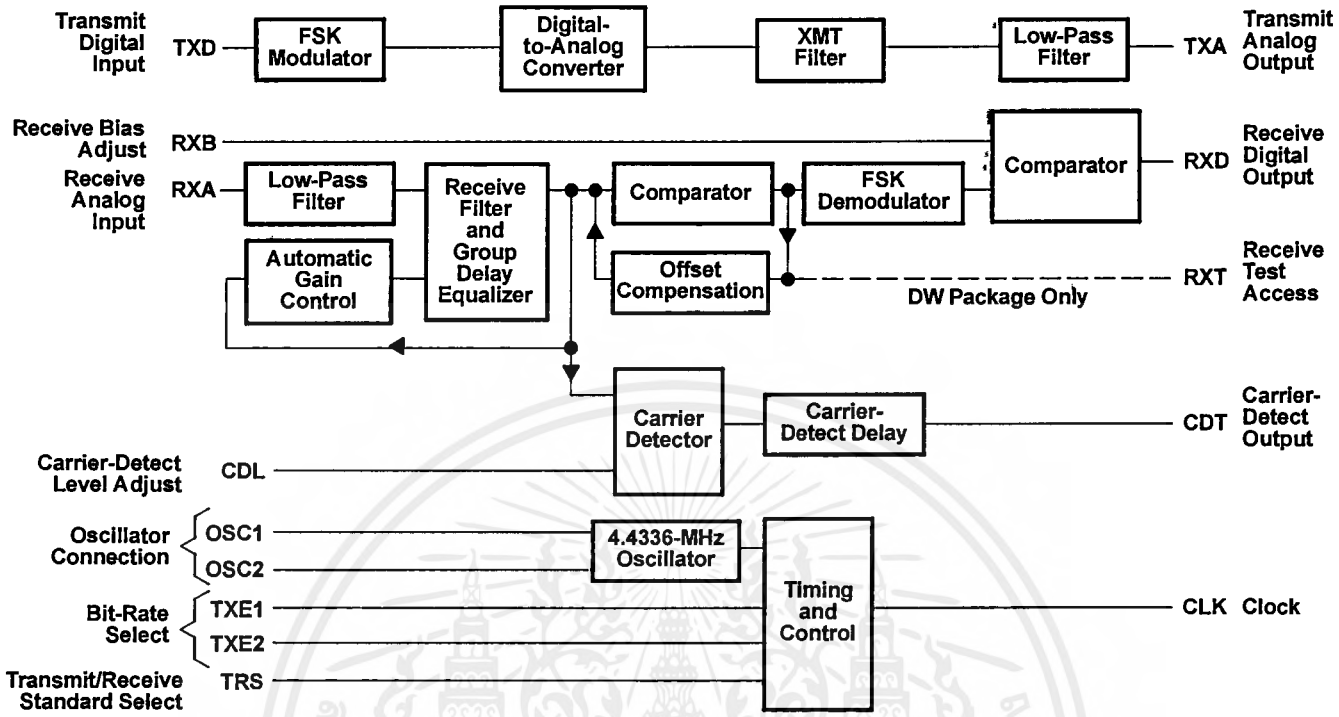
Terminal Functions

TERMINAL			DESCRIPTION
NAME	NO.		
	DW	J OR N	
CDL	14	10	Carrier-detect-level adjust for external adjustment of carrier-detect threshold
CDT	3	3	Carrier-detect output. A low-level output indicates carrier failure
CLK	2	2	Output for a continuous clock signal at 16 times the highest selected (transmit or receive) bit rate
NC	4, 6, 8, 10, 15, 18, 19, 21	6	No internal connection
OSC1, OSC2	23, 24	15, 16	Oscillator connections. The crystal (typically 4.4336 MHz) is connected to OSC1 AND OSC2. If an external clock is used, OSC2 is left open and the clock is connected to OSC1.
RXA	5	4	Receive analog input to which the received line signal must be ac coupled
RXB	11	7	Receive bias adjust for external adjustment of the decision threshold of the comparator to minimize bias distortion
RXD	12	8	Receiver digital output for the demodulated received data in positive logic. The high logic level is a mark and the low logic level is a space.
RXT	9	–	Receive test access. Output of limiter is available on RXT. (DW only)
TRS	7	5	Transmit/receive standard select input, which with TXR1 and TXR2, sets the standard bit rates and mark/space frequencies
TXA	16	11	Transmit analog output for the modulation signal, which must be ac coupled
TXD	22	14	Transmit digital input for data to the transmitter in positive logic. The high logic level is a mark, and the low logic level is a space. The data can be accepted at any speed from zero to the selected speed and may be totally asynchronous.
TXR1	20	13	Bit-rate select 1 input which along with TXR2 and TRS, sets the bit rates and mark/space frequencies
TXR2	17	12	Bit rate select 2 input, which along with TXR1 and TRS, sets the bit rates and mark/space frequencies
V_{DD}	1	1	Positive supply voltage
V_{SS}	13	9	Most negative supply voltage (normally ground); connected to substrate

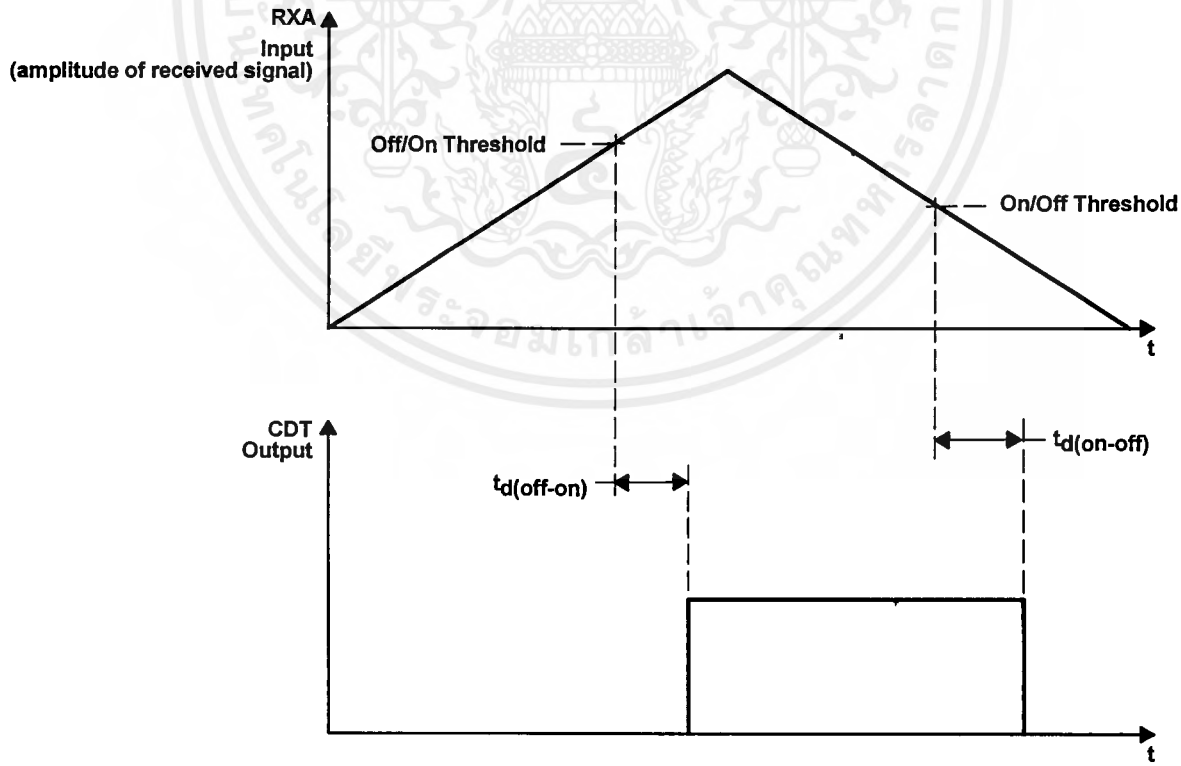
TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265
POST OFFICE BOX 1443 • HOUSTON, TEXAS 77251–1443

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

functional block diagram



timing diagram



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)

Supply voltage range, V_{DD} (see Note 1) ..... −0.3 V to 10 V
Input voltage range, V_I (any input) ..... −0.3 V to V_{DD}
Operating free-air temperature range, T_A: TCM3105DWL, TCM3105JL, TCM3105NL ..... −10°C to 70°C
TCM3105JE, TCM3105NE ..... −55°C to 85°C
Storage temperature range: ..... −55°C to 150°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds: DW or N package ..... 260°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds: J package ..... 300°C

NOTE 1: Voltage values are with respect to V_{SS}.

recommended operating conditions

	TCM3105JE TCM3105NE			TCM3105DWL TCM3105JL TCM3105NL			UNIT
	MIN	NOM	MAX	MIN	NOM	MAX	
Supply voltage, V_{DD}	4	5	5.5	4	5	5.5	V
High-level input voltage, V_{IH}	2		V_{DD}	2		V_{DD}	V
Low-level input voltage, V_{IL}	0		0.8	0		0.8	V
Analog input level, peak to peak (ac coupled)		0.3	0.78		0.3	0.78	V
Clock frequency, f_{clock}	4.4334	4.4336	4.4338	4.4334	4.4336	4.4338	MHz
Analog load impedance at TXA	50			50			kΩ
Operating free-air temperature range, T_A	−40		85	0		70	°C

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER			TEST CONDITIONS		TCM3105JE TCM3105NE			TCM3105DWL TCM3105JL TCM3105NL			UNIT
					MIN	TYP†	MAX	MIN	TYP†	MAX	
V_{OH}	High-level output voltage	RXD, CDT, CLK	I_{OH} = –100 µA		2.4		V_{DD}	2.4		V_{DD}	V
V_{OL}	Low-level output voltage	RXD, CDT, CLK	I_{OL} = 1.6 mA		V_{SS}		0.4	V_{SS}		0.4	V
	Analog output voltage level, peak to peak	TXA	V_{DD} = 4 V	R_L = 50 kΩ, R_L = 100 pF		1.55			1.55		V
			V_{DD} = 5 V		1.4	1.9	2.3	1.4	1.9	2.3	
			V_{DD} = 5.5 V			2.1			2.1		
	Adjust voltage	RXB	V_{DD} = 5 V		2.3	2.7	3.1	2.3	2.7	3.1	V
		CDL			2.8	3.3	3.9	2.8	3.3	3.9	
	Analog output dc offset	TXA				$V_{DD}/2$			$V_{DD}/2$		V
	Digital input current	TXD, TRS, TRX1, TRX2	V_I = 0 to V_{DD}				±1			±1	µA
	Analog input current	RXA					±15			±15	µA
	Bias input current	RXB, CDL	V_I = 3 V				±150			±150	µA
I_{DD}	Supply current		V_{DD} = 4 V			3	6		3	5	mA
			V_{DD} = 5 V			5	10		5	8	
			V_{DD} = 5.5 V			8	16		8	12	
C_i	Input capacitance, all inputs		f = 1 MHz			10			10		pF
C_o	Output capacitance, all inputs		f = 1 MHz			10			10		pF
	Phase jitter						200			200	µs
	Bias distortion‡						±15%			±15%	
	Carrier-detect threshold, off/on§				–45.5		–43	–45.5		–43	dBm
	Carrier-detect threshold, on/off§				–48		–45.5	–48		–45.5	dBm
	Carrier-detect hysteresis				2.5	2.8		2.5	2.8		dBm

switching characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS	TCM3105JE TCM3105NE			TCM3105DWL TCM3105JL TCM3105NL			UNIT
			MIN	TYP†	MAX	MIN	TYP†	MAX	
$t_{d(off\text{-}on)}$	Carrier-detect off-to-on delay time	RX = 600 or 1200 b/s	12		25	12		25	ms
		RX = 5, 75, or 150 b/s	48		80	48		80	
$t_{d(on\text{-}off)}$	Carrier-detect on-to-off delay time	RX = 600 or 1200 b/s	12		20	12		20	ms
		RX = 5, 75, or 150 b/s	48		75	48		75	
	Transmit frequency deviation from assignment (see Table 1)	f_{clock} = 4.4336 MHz		±1			±1		Hz

† All typical are at V_{CC} = 5 V, T_A = 25°C.
‡ Bias distortion is the departure from a 50% duty cycle when a series of alternating mark and space tones is received.
§ This is the threshold with the CDL input properly adjusted.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

PRINCIPLES OF OPERATION

The TCM3105 FSK modem is made up of four functional circuits. The circuits are the transmitter, the receiver, a carrier detector, and control and timing (see Figure 1).

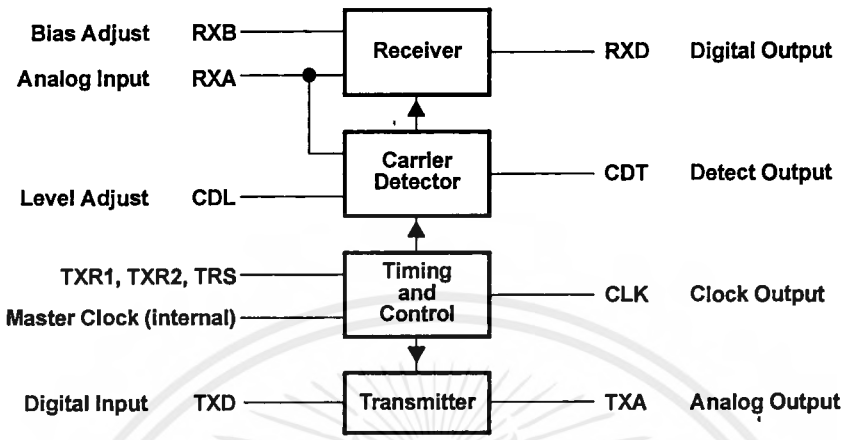


Figure 1. TCM3105 System Partitioning

transmitter

The transmitter comprises a phase-coherent FSK modulator, a transmit filter, and a transmit amplifier. The modulator is a programmable frequency synthesizer that drives the output frequencies by variable division of the oscillator frequency (4.4336 MHz). The division ratio is set by the states of the transmit/receive standard input (TRS), the bit-rate select inputs (TXR1 and TXR2), and the digital data input (TXD).

A switched-capacitor low-pass filter limits the harmonics and noise outside the transmit band, and the characteristics of this filter are set by the frequency-select inputs as previously described. The harmonics introduced by the transmit filter clock are removed by a continuous low-pass filter.

The transmitter output level varies with power supply voltage and so must be compensated in the 2-wire to 4-wire converter to give a constant output level to the line.

receiver

A continuous low-pass antialiasing filter is followed by the receiver amplifier, which automatically controls the gain to give a constant output level from the receiver filter. The receiver filter limits the bandwidth of the signal presented to the demodulator reducing out-of-band interference and has very high rejection of the transmit channel frequencies. These are typically present at much higher levels than the received signal.

The group delay equalizer is a switched-capacitor network that compensates the delay introduced by the receiver filter and the network. The output from the equalizer is then limited to give an FSK modulated squarewave that is presented to the demodulator.

The demodulator is an edge-triggered multivibrator that triggers off positive- and negative-going edges. The output of the demodulator is a stream of constant-length pulses at a frequency that is double the frequency of the limited input signal. The dc component of this signal is proportional to the received frequency and is extracted by a switched-capacitor, low-pass, post-demodulator filter.

The variation of dc level with received frequency is presented to a comparator that slices at a level externally fixed by the RXB bias-adjustment pin. This voltage depends on received bit rate and internal offsets. The comparator output is then the received data at RXD.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

carrier detect

The carrier-detect circuits comprise an energy detector and digital delay. The energy detector compares the total signal level at the output of the receive filter to an externally set threshold level on the CDL input. The comparator has a 2.5-dB hysteresis and a delay to allow for momentary signal loss and to prevent oscillation. The output detector is available on CDT where a high level indicates that a carrier is present. The data output is clamped to a mark condition when the carrier-detect output switches off at the end of transmission.

control and timing

An on-chip oscillator runs from an external 4.4336-MHz crystal connected between OSC1 and OSC2 or an external signal driving OSC1. A clock signal equal to 16 times the highest selected bit rate (transmit or receive) is available on the CLK output.

The single-supply rail means that all analog functions are referenced to an internally generated reference. All analog inputs and outputs must be ac coupled.

transmit and receive modes

The various modes of operation of the TCM3105 are given in Table 1. The data convention is that a logic high is a mark and a logic low is a space.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

Table 1. Modes of Operation

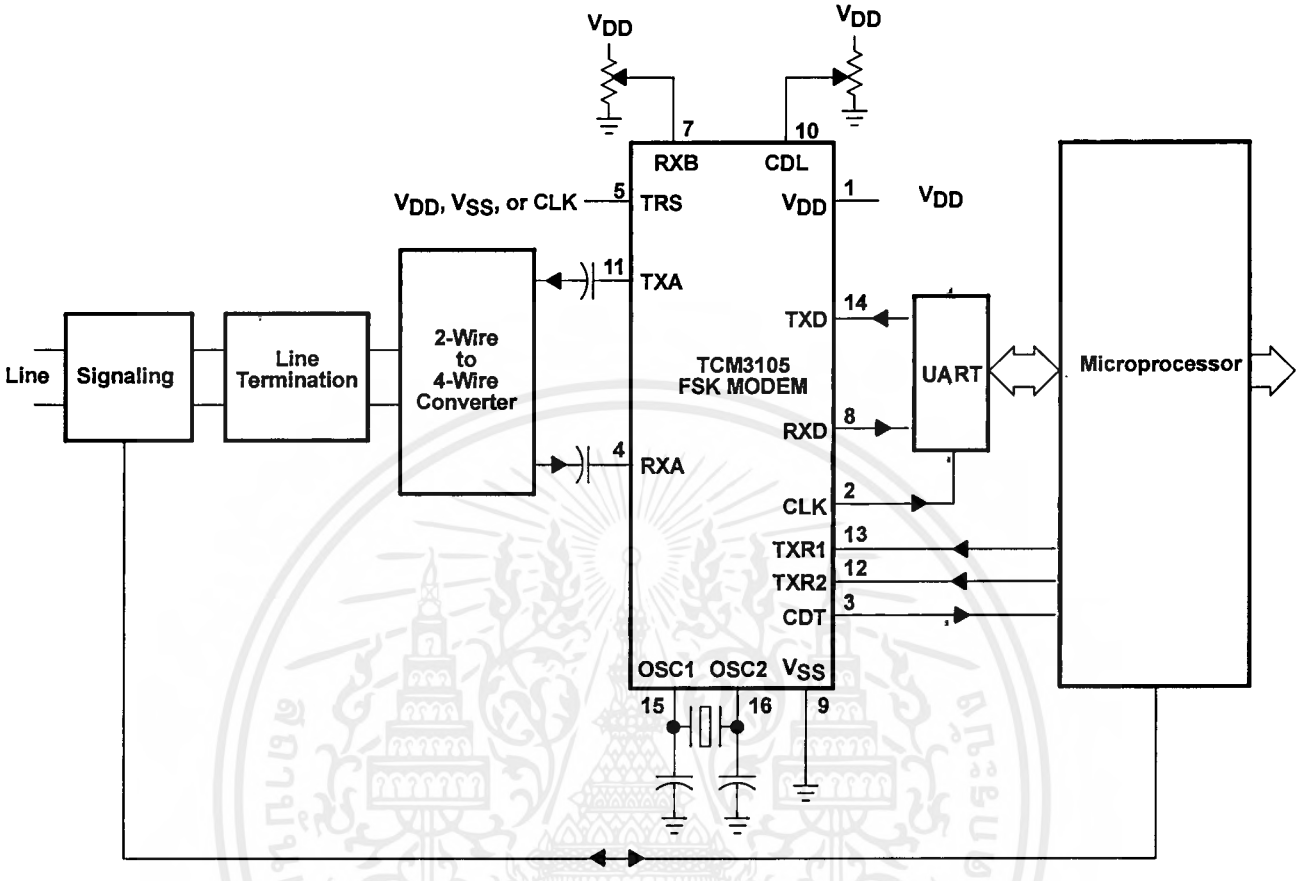
STANDARD	TRS	TXR1	TXR2	TRANSMITTED BAUD RATE	RECEIVED BAUD RATE	TRANSMIT FREQUENCY ASSIGNMENTS (Hz)	RECEIVE FREQUENCY ASSIGNMENTS (Hz)	CLK FREQUENCY (kHz)
CCITT V.23	L	L	L	1200	1200	M 1300 S 2100	M 1300 S 2100	19.11
	H	L	L	1200	75	M 1300 S 2100	M 390 S 450	19.11
	L	L	H	600	75	M 1300 S 1700	M 390 S 450	9.56
	H	L	H	600	600	M 1300 S 1700	M 1300 S 1700	9.56
	L	H	L	75	1200	M 390 S 450	M 1300 S 2100	19.11
	H	H	L	75	600	M 390 S 450	M 1300 S 1700	9.56
	L	H	H	75	75	M 390 S 450	M 390 S 450	1.19
BELL 202	$\overline{CLK}$	L	L	1200	1200	M 1200 S 2200	M 1200 S 2200	19.11
	$\overline{CLK}$/8	L	H	1200	150	M 1200 S 2200	M 387 S 487	19.11
	$\overline{CLK}$/8	L	H	1200	5	M 1200 S 2200	M 387 S 0	19.11
	CLK	H	L	150	1200	M 387 S 487	M 1200 S 2200	19.11
	CLK	H	H	150	150	M 387 S 487	M 387 S 487	2.39
	CLK†	H†	L†	5	1200	M 387	M 1200 S 2200	19.11
	H†	H†	H†			S 0		
	H	H	H	Transmit Disabled	1200	Transmit Disabled	M 1200 S 2200	19.11

H = high level, L = low level
† In these modes, the modulation is controlled by TRS and TXR2. TXD is tied high.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265
POST OFFICE BOX 1443 • HOUSTON, TEXAS 77251–1443

APPLICATION INFORMATION

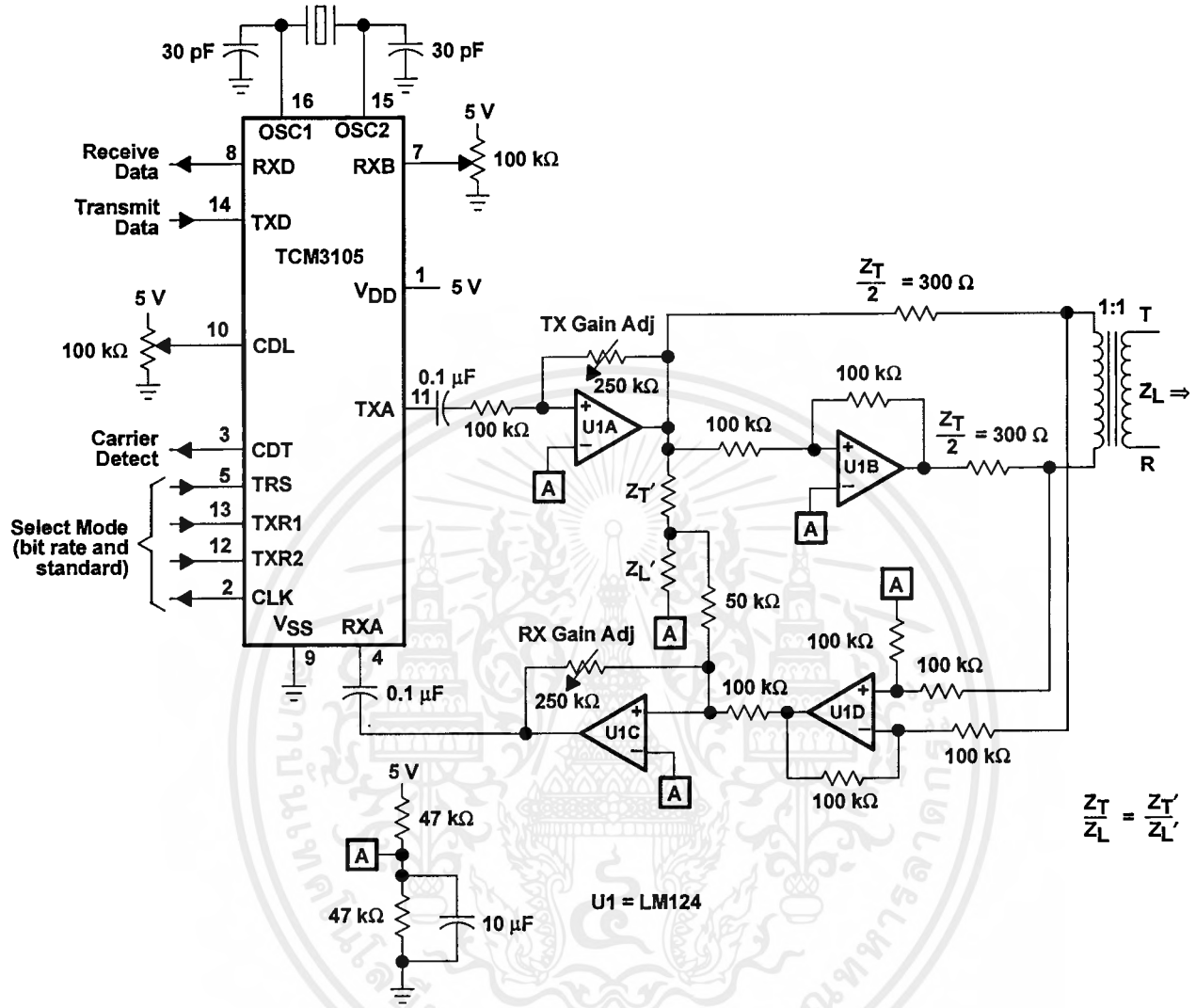


Pin numbers shown are for the J and N packages.

Figure 2. Typical System Configuration

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

APPLICATION INFORMATION



Pin numbers shown are for the J and N packages.

Figure 3. Telephone Line Interface Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเอกสารทุกครั้งที่มีการนำไปใช้

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265
POST OFFICE BOX 1443 • HOUSTON, TEXAS 77251–1443

APPLICATION INFORMATION

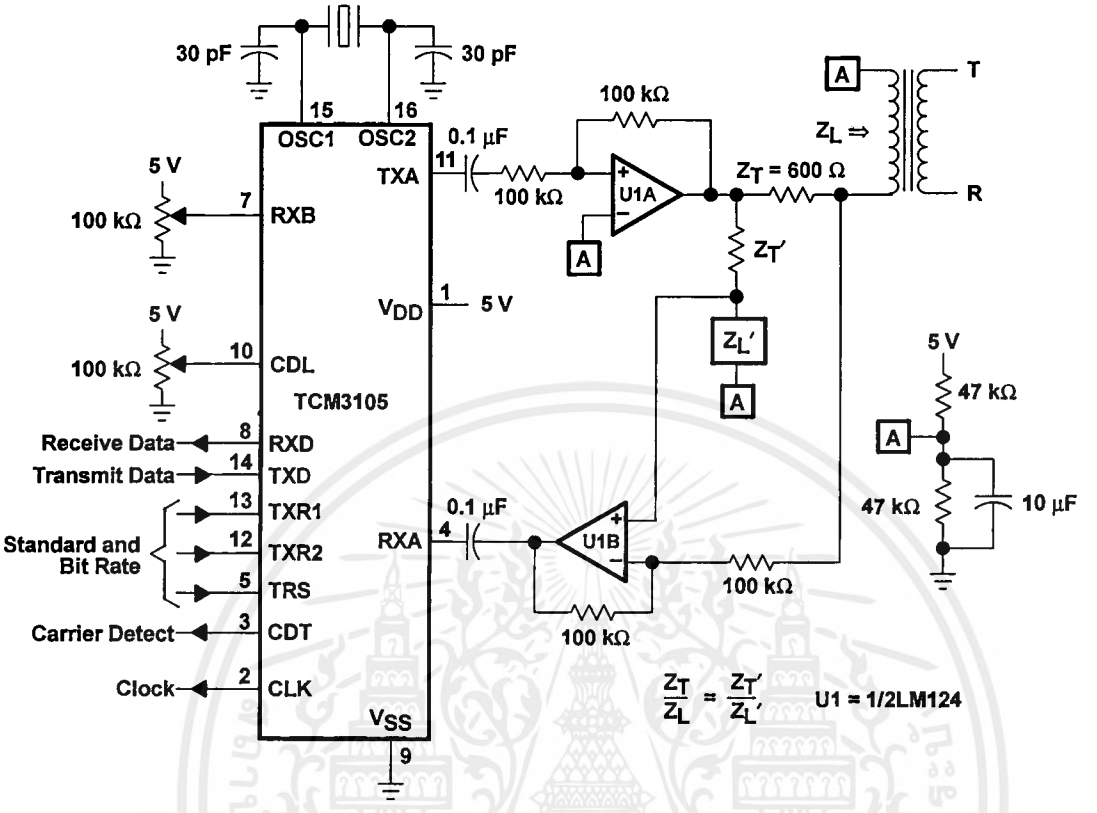


Figure 4. Simplified Telephone Line Interface Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

19-4323; Rev 6; 3/96

MAXIM

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

General Description

The MAX220-MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, and in particular, for those applications where ±12V is not available.

These parts are particularly useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than 5µW. The MAX225, MAX233, MAX235, and MAX245-MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

Applications

Portable Computers

Low-Power Modems

Interface Translation

Battery-Powered RS-232 Systems

Multi-Drop RS-232 Networks

Features

Superior to Bipolar

- **Operate from Single +5V Power Supply (+5V and +12V—MAX231/MAX239)**
- **Low-Power Receive Mode in Shutdown (MAX223/MAX242)**
- **Meet All EIA/TIA-232E and V.28 Specifications**
- **Multiple Drivers and Receivers**
- **3-State Driver and Receiver Outputs**
- **Open-Line Detection (MAX243)**

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220C/D	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO
MAX220EJE	-40°C to +85°C	16 CERDIP
MAX220MJE	-55°C to +125°C	16 CERDIP

Ordering Information continued at end of data sheet.

**Contact factory for dice specifications.*

Selection Table

Part Number	Power Supply (V)	No. of RS-232 Drivers/Rx	No. of Ext. Caps	Nominal Cap. Value (µF)	SHDN & Three-State	Rx Active in SHDN	Data Rate (kbps)	Features
MAX220	+5	2/2	4	4.7/10	No		120	Ultra-low-power, industry-standard pinout
MAX222	+5	2/2	4	0.1	Yes		200	Low-power shutdown
MAX223 (MAX213)	+5	4/5	4	1.0 (0.1)	Yes	✔	120	MAX241 + receivers active in shutdown
MAX225	+5	5/5	0	–	Yes	✔	120	Available in SO
MAX230 (MAX200)	+5	5/0	4	1.0 (0.1)	Yes		120	5 drivers with shutdown
MAX231 (MAX201)	+5 and +7.5 to +13.2	2/2	2	1.0 (0.1)	No		120	Standard +5/+12V or battery supplies; same functions as MAX232
MAX232 (MAX202)	+5	2/2	4	1.0 (0.1)	No		120 (64)	Industry standard
MAX232A	+5	2/2	4	0.1	No		200	Higher slew rate, small caps
MAX233 (MAX203)	+5	2/2	0	–	No		120	No external caps
MAX233A	+5	2/2	0	–	No		200	No external caps, high slew rate
MAX234 (MAX204)	+5	4/0	4	1.0 (0.1)	No		120	Replaces 1488
MAX235 (MAX205)	+5	5/5	0	–	Yes		120	No external caps
MAX236 (MAX206)	+5	4/3	4	1.0 (0.1)	Yes		120	Shutdown, three state
MAX237 (MAX207)	+5	5/3	4	1.0 (0.1)	No		120	Complements IBM PC serial port
MAX238 (MAX208)	+5	4/4	4	1.0 (0.1)	No		120	Replaces 1488 and 1489
MAX239 (MAX209)	+5 and +7.5 to +13.2	3/5	2	1.0 (0.1)	No		120	Standard +5/+12V or battery supplies; single-package solution for IBM PC serial port
MAX240	+5	5/5	4	1.0	Yes		120	DIP or flatpack package
MAX241 (MAX211)	+5	4/5	4	1.0 (0.1)	Yes		120	Complete IBM PC serial port
MAX242	+5	2/2	4	0.1	Yes	✔	200	Separate shutdown and enable
MAX243	+5	2/2	4	0.1	No		200	Open-line detection simplifies cabling
MAX244	+5	8/10	4	1.0	No		120	High slew rate
MAX245	+5	8/10	0	–	Yes	✔	120	High slew rate, int. caps, two shutdown modes
MAX246	+5	8/10	0	–	Yes	✔	120	High slew rate, int. caps, three shutdown modes
MAX247	+5	8/9	0	–	Yes	✔	120	High slew rate, int. caps, nine operating modes
MAX248	+5	8/8	4	1.0	Yes	✔	120	High slew rate, selective half-chip enables
MAX249	+5	6/10	4	1.0	Yes	✔	120	Available in quad flatpack package

For free samples & the latest literature: http://www.maxim-ic.com, or phone 1-800-998-8800

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX220/222/232A/233A/242/243

Supply Voltage (V_{CC}) ..... -0.3V to +6V
Input Voltages
T_{IN} ..... -0.3V to (V_{CC} - 0.3V)
R_{IN} ..... ±30V
T_{OUT} (Note 1) ..... ±15V
Output Voltages
T_{OUT} ..... ±15V
R_{OUT} ..... -0.3V to (V_{CC} + 0.3V)
Driver/Receiver Output Short Circuited to GND ..... Continuous
Continuous Power Dissipation (T_A = +70°C)
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C) 842mW
18-Pin Plastic DIP (derate 11.11mW/°C above +70°C) 889mW
20-Pin Plastic DIP (derate 8.00mW/°C above +70°C) 440mW
16-Pin Narrow SO (derate 8.70mW/°C above +70°C) ...696mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
18-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C) 800mW
20-Pin SSOP (derate 8.00mW/°C above +70°C) 640mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C) 800mW
18-Pin CERDIP (derate 10.53mW/°C above +70°C) 842mW
Operating Temperature Ranges
MAX2_ _AC_ _, MAX2_ _C_ _ ..... 0°C to +70°C
MAX2_ _AE_ _, MAX2_ _E_ _ ..... -40°C to +85°C
MAX2_ _AM_ _, MAX2_ _M_ _ ..... -55°C to +125°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10sec) ..... +300°C

Note 1: Input voltage measured with T_{OUT} in high-impedance state, $\overline{SHDN}$ or V_{CC} = 0V.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243

(V_{CC} = +5V ±10%, C1-C4 = 0.1µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 TRANSMITTERS						
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to GND		±5	±8		V
Input Logic Threshold Low				1.4	0.8	V
Input Logic Threshold High			2	1.4		V
Logic Pull-Up/Input Current	Normal operation			5	40	µA
	$\overline{SHDN}$ = 0V, MAX222/242, shutdown			±0.01	±1	
Output Leakage Current	V_{CC} = 5.5V, $\overline{SHDN}$ = 0V, V_{OUT} = ±15V, MAX222/242			±0.01	±10	µA
	V_{CC} = $\overline{SHDN}$ = 0V, V_{OUT} = ±15V			±0.01	±10	
Data Rate	Except MAX220, normal operation			200	116	kbits/ sec
	MAX220			22	20	
Transmitter Output Resistance	V_{CC} = V+ = V- = 0V, V_{OUT} = ±2V		300	10M		Ω
Output Short-Circuit Current	V_{OUT} = 0V			±22		mA
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range					±30	V
RS-232 Input Threshold Low	V_{CC} = 5V	Except MAX243 $R2_{IN}$	0.8	1.3		V
		MAX243 $R2_{IN}$ (Note 2)	-3			
RS-232 Input Threshold High	V_{CC} = 5V	Except MAX243 $R2_{IN}$		1.8	2.4	V
		MAX243 $R2_{IN}$ (Note 2)		-0.5	-0.1	
RS-232 Input Hysteresis	Except MAX243, V_{CC} = 5V, no hyst. in shdn.		0.2	0.5	1	V
	MAX243			1		
RS-232 Input Resistance			3	5	7	kΩ
TTL/CMOS Output Voltage Low	I_{OUT} = 3.2mA			0.2	0.4	V
TTL/CMOS Output Voltage High	I_{OUT} = -1.0mA		3.5	V_{CC} -0.2		V
TTL/CMOS Output Short-Circuit Current	Sourcing V_{OUT} = GND		-2	-10		mA
	Shrinking V_{OUT} = V_{CC}		10	30		
TTL/CMOS Output Leakage Current	$\overline{SHDN}$ = V_{CC} or $\overline{EN}$ = V_{CC} ($\overline{SHDN}$ = 0V for MAX222), 0V ≤ V_{OUT} ≤ V_{CC}			±0.05	±10	µA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243 (continued)

(V_{CC} = +5V ±10%, C1-C4 = 0.1µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
$\overline{EN}$ Input Threshold Low	MAX242			1.4	0.8	V
$\overline{EN}$ Input Threshold High	MAX242		2.0	1.4		V
POWER SUPPLY						
Operating Supply Voltage			4.5		5.5	V
V_{CC} Supply Current ($\overline{SHDN}$ = V_{CC}), Figures 5, 6, 9, 19	No load	MAX220		0.5	2	mA
		MAX222/232A/233A/242/243		4	10	
	3kΩ load both inputs	MAX220		12		
		MAX222/232A/233A/242/243		15		
Shutdown Supply Current	MAX222/242	T_A = +25°C		0.1	10	µA
		T_A = 0° to +70°C		2	50	
		T_A = -40° to +85°C		2	50	
		T_A = -55° to +125°C		35	100	
$\overline{SHDN}$ Input Leakage Current	MAX222/242				±1	µA
$\overline{SHDN}$ Threshold Low	MAX222/242			1.4	0.8	V
$\overline{SHDN}$ Threshold High	MAX222/242		2.0	1.4		V
AC CHARACTERISTICS						
Transition Slew Rate	C_L = 50pF to 2500pF, R_L = 3kΩ to 7kΩ, V_{CC} = 5V, T_A = +25°C, measured from +3V to -3V or -3V to +3V	MAX222/232A/233A/242/243	6	12	30	V/µs
		MAX220	1.5	3	30	
Transmitter Propagation Delay TLL to RS-232 (normal operation), Figure 1	t_{PHLT}	MAX222/232A/233A/242/243		1.3	3.5	µs
		MAX220		4	10	
	t_{PLHT}	MAX222/232A/233A/242/243		1.5	3.5	
		MAX220		5	10	
Receiver Propagation Delay RS-232 to TLL (normal operation), Figure 2	t_{PHLR}	MAX222/232A/233A/242/243		0.5	1	µs
		MAX220		0.6	3	
	t_{PLHR}	MAX222/232A/233A/242/243		0.6	1	
		MAX220		0.8	3	
Receiver Propagation Delay RS-232 to TLL (shutdown), Figure 2	t_{PHLS}	MAX242		0.5	10	µs
	t_{PLHS}	MAX242		2.5	10	
Receiver-Output Enable Time, Figure 3	t_{ER}	MAX242		125	500	ns
Receiver-Output Disable Time, Figure 3	t_{DR}	MAX242		160	500	ns
Transmitter-Output Enable Time ($\overline{SHDN}$ goes high), Figure 4	t_{ET}	MAX222/242, 0.1µF caps (includes charge-pump start-up)		250		µs
Transmitter-Output Disable Time ($\overline{SHDN}$ goes low), Figure 4	t_{DT}	MAX222/242, 0.1µF caps		600		ns
Transmitter + to - Propagation Delay Difference (normal operation)	t_{PHLT} - t_{PLHT}	MAX222/232A/233A/242/243		300		ns
		MAX220		2000		
Receiver + to - Propagation Delay Difference (normal operation)	t_{PHLR} - t_{PLHR}	MAX222/232A/233A/242/243		100		ns
		MAX220		225		

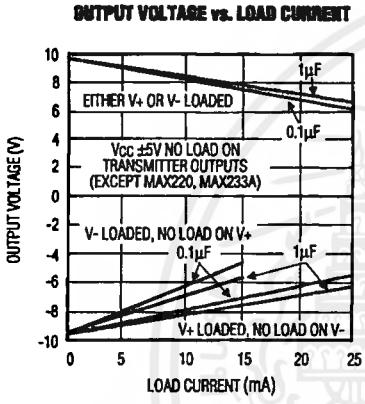
Note 2: MAX243 $R2_{OUT}$ is guaranteed to be low when $R2_{IN}$ is ≥ 0V or is floating.

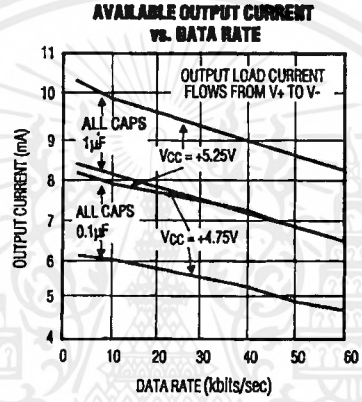
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

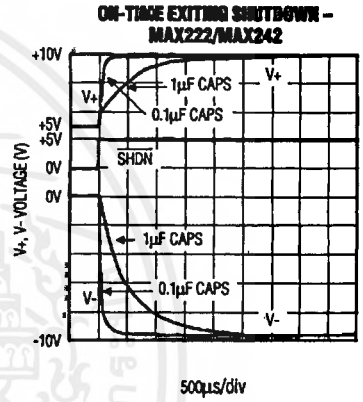
+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

Typical Operating Characteristics

MAX220/MAX222/MAX232A/MAX233A/MAX242/MAX243







เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ABSOLUTE MAXIMUM RATINGS—MAX223/MAX230-MAX241

V_{CC} ..... -0.3V to +6V
V+ ..... (V_{CC} - 0.3V) to +14V
V- ..... +0.3V to -14V
Input Voltages
T_{IN} ..... -0.3V to (V_{CC} + 0.3V)
R_{IN} ..... ±30V
Output Voltages
T_{OUT} ..... (V+ + 0.3V) to (V- - 0.3V)
R_{OUT} ..... -0.3V to (V_{CC} + 0.3V)
Short-Circuit Duration, T_{OUT} ..... Continuous
Continuous Power Dissipation (T_A = +70°C)
14-Pin Plastic DIP (derate 10.00mW/°C above +70°C)....800mW
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C)....842mW
20-Pin Plastic DIP (derate 11.11 mW/°C above +70°C)...889mW
24-Pin Narrow Plastic DIP
(derate 13.33mW/°C above +70°C) 1.07W
24-Pin Plastic DIP (derate 9.09mW/°C above +70°C).....500mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C).....762mW
20-Pin Wide SO (derate 10 00mW/°C above +70°C).....800mW
24-Pin Wide SO (derate 11.76mW/°C above +70°C).....941mW
28-Pin Wide SO (derate 12.50mW/°C above +70°C) 1W
44-Pin Plastic FP (derate 11.11 mW/°C above +70°C) 889mW
14-Pin CERDIP (derate 9.09mW/°C above +70°C).....727mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C).....800mW
20-Pin CERDIP (derate 11.11mW/°C above +70°C).....889mW
24-Pin Narrow CERDIP
(derate 12.50mW/°C above +70°C) 1W
24-Pin Sidebraze (derate 20.0mW/°C above +70°C).....1.6W
28-Pin SSOP (derate 9.52mW/°C above +70°C).....762mW
Operating Temperature Ranges
MAX2_ _ C_ _ ..... 0°C to +70°C
MAX2_ _ E_ _ ..... -40°C to +85°C
MAX2_ _ M_ _ ..... -55°C to +125°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10sec) ..... +300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX223/MAX230-MAX241

(MAX223/230/232/234/236/237/238/240/241 V_{CC} = +5V ±10%, MAX233/MAX235 V_{CC} = 5V ±5%, C1-C4 = 1.0µF MAX231/MAX239 V_{CC} = 5V ±10%, V+ = 7.5V to 13.2V, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to ground		±5.0	±7.3		V
V_{CC} Power-Supply Current	No load, T_A = +25°C	MAX232/233		5	10	mA
		MAX223/230/234-238/240/241		7	15	
		MAX231 /239		.4	1	
V+ Power-Supply Current		MAX231		1.8	5	mA
		MAX239		5	15	
Shutdown Supply Current	T_A = +25°C	MAX223		15	50	µA
		MAX230/235/236/240/241		1	10	
Input Logic Threshold Low	T_{IN}; EN, $\overline{SHDN}$ (MAX223), $\overline{EN}$, SHDN (MAX230/235-241)				0.8	V
Input Logic Threshold High	T_{IN}		2.0			V
	EN, $\overline{SHDN}$ (MAX223), $\overline{EN}$, SHDN (MAX230/235/236/240/241)		2.4			
Logic Pull-Up Current	T_{IN} = 0V			1.5	200	µA
Receiver Input Voltage Operating Range			-30		30	V

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

ELECTRICAL CHARACTERISTICS—MAX223/MAX230-MAX241 (continued)

(MAX223/230/232/234/236/237/238/240/241 V_{CC} = +5V ±10%, MAX233/MAX235 V_{CC} = 5V ±5%, C1-C4 = 1.0µF MAX231/MAX239 V_{CC} = 5V ±10%, V+ = 7.5V to 13.2V, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS			MIN	TYP	MAX	UNITS
RS-232 Input Threshold Low	T_A = +25°C, V_{CC} = 5V	Normal operation $\overline{SHDN}$ = 5V (MAX223) SHDN = 0V (MAX235/236/240/241)		0.8	1.2		V
		Shutdown (MAX223) $\overline{SHDN}$ = 0V, EN = 5V ($R4_{IN}$, $R5_{IN}$)		0.6	1.5		
RS-232 Input Threshold High	T_A = +25°C, V_{CC} = 5V	Normal operation $\overline{SHDN}$ = 5V (MAX223) SHDN = 0V (MAX235/236/240/241)			1.7	2.4	V
		Shutdown (MAX223) $\overline{SHDN}$ = 0V, EN = 5V ($R4_{IN}$, $R5_{IN}$)			1.5	2.4	
RS-232 Input Hysteresis	V_{CC} = 5V; no hysteresis in shutdown			0.2	0.5	1.0	V
RS-232 Input Resistance	T_A = +25°C, V_{CC} = 5V			3	5	7	kΩ
TTL/CMOS Output Voltage Low	I_{OUT} = 1.6mA (MAX231-233 I_{OUT} = 3.2mA)					0.4	V
TTL/CMOS Output Voltage High	I_{OUT} = -1mA			3.5	V_{CC} - 0.4		V
TTL/CMOS Output Leakage Current	0V ≤ R_{OUT} ≤ V_{CC}; EN = 0V (MAX223); $\overline{EN}$ = V_{CC} (MAX235-241)				0.05	±10	µA
Receiver Output Enable Time	Normal operation	MAX223			600		ns
		MAX235/236/239/240/241			400		
Receiver Output Disable Time	Normal operation	MAX223			900		ns
		MAX235/236/239/240/241			250		
Propagation Delay	RS-232 IN to TTL/CMOS OUT, C_L = 150pF	Normal operation			0.5	10	µs
		$\overline{SHDN}$ = 0V (MAX223)	t_{PHLS}		4	40	
			t_{PLHS}		6	40	
Transition Region Slew Rate	MAX223/MAX230/MAX234-241 T_A = +25°C, V_{CC} = 5V, R_L = 3kΩ to 7kΩ, C_L = 50pF to 2500pF, measured from +3V to -3V or -3V to +3V			3	5.1	30	V/µs
	MAX231/MAX232/MAX233 T_A = +25°C, V_{CC} = 5V, R_L = 3kΩ to 7kΩ, C_L = 50pF to 2500pF, measured from +3V to -3V or -3V to +3V				4	30	
Transmitter Output Resistance	V_{CC} = V+ = V- = 0V, V_{OUT} = ±2V			300			Ω
Transmitter Out Short-Circuit Current					±10		mA

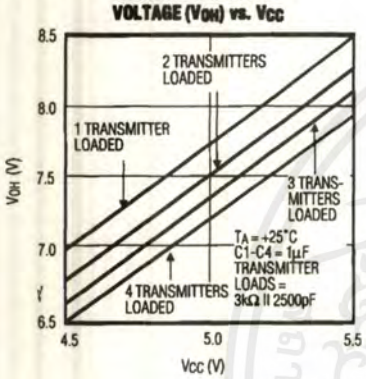
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

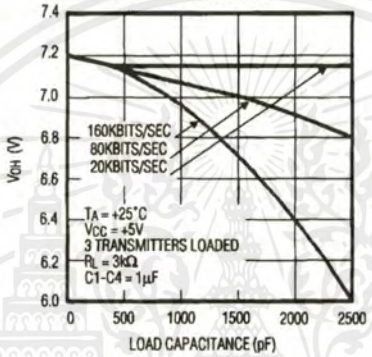
Typical Operating Characteristics

MAX223/MAX230-MAX241

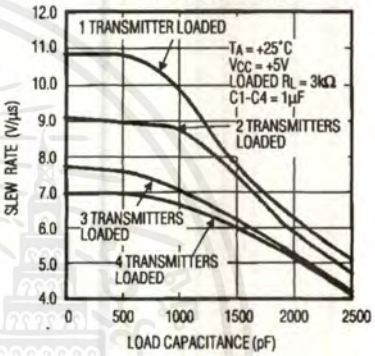
TRANSMITTER OUTPUT VOLTAGE (V_{OH}) vs. V_{CC}



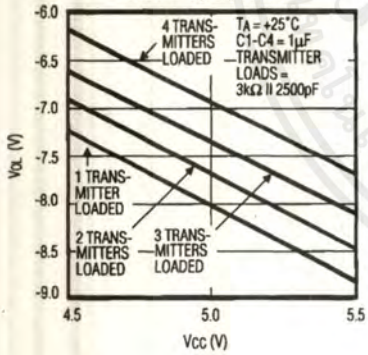
TRANSMITTER OUTPUT VOLTAGE (V_{OH}) vs. LOAD CAPACITANCE AT DIFFERENT DATA RATES



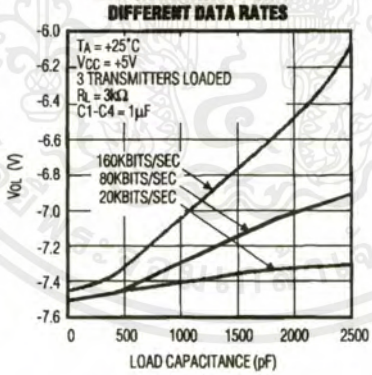
TRANSMITTER SLEW RATE vs. LOAD CAPACITANCE



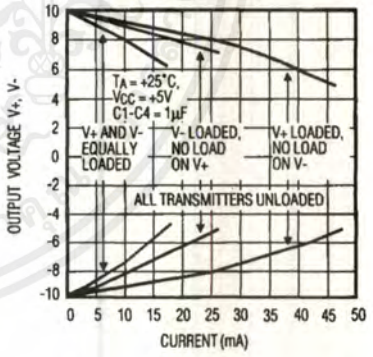
TRANSMITTER OUTPUT VOLTAGE (V_{OL}) vs. V_{CC}



TRANSMITTER OUTPUT VOLTAGE (V_{OL}) vs. LOAD CAPACITANCE AT DIFFERENT DATA RATES



TRANSMITTER OUTPUT VOLTAGE V+, V- vs. LOAD CURRENT



V+, V- WHEN EXITING SHUTDOWN (1µF CAPACITORS)

*SHUTDOWN POLARITY IS REVERSED FOR THE MAX241

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX225/MAX244-MAX249

Supply Voltage (V_{CC}) ..... -0.3V to +6V
Input Voltages
T_{IN}, $\overline{ENA}$, $\overline{ENB}$, $\overline{ENR}$, $\overline{ENT}$, $\overline{ENRA}$, $\overline{ENRB}$, $\overline{ENTA}$, $\overline{ENTB}$ ..... -0.3V to (V_{CC} + 0.3V)
R_{IN} ..... ±25V
T_{OUT} (Note 3) ..... ±15V
R_{OUT} ..... -0.3V to (V_{CC} + 0.3V)
Short Circuit (one output at a time)
T_{OUT} to GND ..... Continuous
R_{OUT} to GND ..... Continuous

Continuous Power Dissipation (T_A = +70°C)
28-Pin Wide SO (derate 12.50mW/°C above +70°C) 1W
40-Pin Plastic DIP (derate 11.11mW/°C above +70°C) ... 611mW
44-Pin PLCC (derate 13.33mW/°C above +70°C) 1.07W
Operating Temperature Ranges
MAX225C_ _, MAX24 C_ _ ..... 0°C to +70°C
MAX225E_ _, MAX24_E_ _ ..... -40°C to +85°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10sec) ..... +300°C

Note 3: Input voltage measured with transmitter output in a high-impedance state, shutdown, or V_{CC} = 0V.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX225/MAX244-MAX249

(MAX225 V_{CC} = 5.0V ±5%; MAX244-MAX249 V_{CC} = +5.0V ±10%, external capacitors C1-C4 = 1µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 TRANSMITTER						
Input Logic Threshold Low				1.4	0.8	V
Input Logic Threshold High			2	1.4		V
Logic Pull-Up/Input Current	Tables 1a-1d	Normal operation		10	50	µA
		Shutdown		±0.01	±1	
Data Rate	Tables 1a-1d, normal operation			120	64	kbits/sec
Output Voltage Swing	All transmitter outputs loaded with 3Ωk to GND		±5	±7.5		V
Output Leakage Current (shutdown)	Tables 1a-1d	$\overline{ENA}$, $\overline{ENB}$, $\overline{ENT}$, $\overline{ENTA}$, $\overline{ENTB}$ = V_{CC}, V_{OUT} = ±15V		±0.01	±25	µA
		V_{CC} = 0V, V_{OUT} = ±15V		±0.01	±25	
Transmitter Output Resistance	V_{CC} = V+ = V- = 0V, V_{OUT} = ±2V (Note 4)		300	10M		Ω
Output Short-Circuit Current	V_{OUT} = 0V		±7	±30		mA
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range					±25	V
RS-232 Input Threshold Low	V_{CC} = 5V		0.8	1.3		V
RS-232 Input Threshold High	V_{CC} = 5V			1.8	2.4	V
RS-232 Input Hysteresis	V_{CC} = 5V		0.2	0.5	1.0	V
RS-232 Input Resistance			3	5	7	kΩ
TTL/CMOS Output Voltage Low	I_{OUT} = 3.2mA			0.2	0.4	V
TTL/CMOS Output Voltage High	I_{OUT} = -1.0mA		3.5	V_{CC} - 0.2		V
TTL/CMOS Output Short-Circuit Current	Sourcing V_{OUT} = GND		-2	-10		mA
	Shrinking V_{OUT} = V_{CC}		10	30		
TTL/CMOS Output Leakage Current	Normal operation, outputs disabled, Tables 1A-1D, 0V ≤ V_{OUT} ≤ V_{CC}, $\overline{ENR}$_ = V_{CC}			±0.05	±0.10	µA

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ELECTRICAL CHARACTERISTICS—MAX225/MAX244-MAX249 (continued)

(MAX225 V_{CC} = 5.0V ±5%; MAX244-MAX249 V_{CC} = +5.0V ±10%, external capacitors C1-C4 = 1µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
POWER SUPPLY AND CONTROL LOGIC						
Operating Supply Voltage		MAX225	4.75		5.25	V
		MAX244-MAX249	4.5		5.5	
V_{CC} Supply Current (normal operation)	No load	MAX225		10	20	mA
		MAX244-MAX249		11	30	
	3kΩ loads on all outputs	MAX225		40		
		MAX244-MAX249		57		
Shutdown Supply Current	T_A = +25°C			8	25	µA
	T_A = T_{MIN} to T_{MAX}				50	
Control Input	Leakage current				±1	µA
	Threshold low			1.4	0.8	V
	Threshold high		2.4	1.4		
AC CHARACTERISTICS						
Transition Slew Rate	C_L = 50pF to 2500pF, R_L = 3kΩ to 7kΩ, V_{CC} = 5V, T_A = +25°C, measured from +3V to -3V or -3V to +3V		5	10	30	V/µs
Transmitter Propagation Delay TLL to RS-232 (normal operation), Figure 1	t_{PHLT}			1.3	3.5	µs
	t_{PLHT}			1.5	3.5	
Receiver Propagation Delay TLL to RS-232 (normal operation), Figure 2	t_{PHLR}			0.6	1.5	µs
	t_{PLHR}			0.6	1.5	
Receiver Propagation Delay TLL to RS-232 (low-power mode), Figure 2	t_{PHLS}			0.6	10	µs
	t_{PLHS}			3.0	10	
Transmitter + to - Propagation Delay Difference (normal operation)	t_{PHLT} - t_{PLHT}			350		ns
Receiver + to - Propagation Delay Difference (normal operation)	t_{PHLR} - t_{PLHR}			350		ns
Receiver-Output Enable Time, Figure 3	t_{ER}			100	500	ns
Receiver-Output Disable Time, Figure 3	t_{DR}			100	500	ns
Transmitter Enable Time	t_{ET}	MAX246-MAX249 (excludes charge-pump start-up)		5		µs
		MAX225/MAX245-MAX249 (includes charge-pump start-up)		10		ms
Transmitter Disable Time, Figure 4	t_{DT}			100		ns

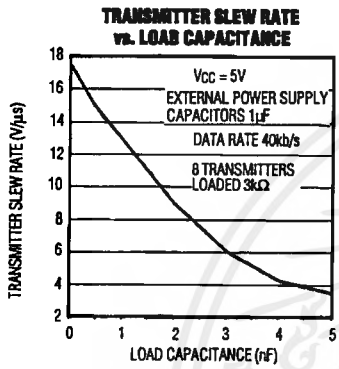
Note 4: The 300Ω minimum specification complies with EIA/TIA-232E, but the actual resistance when in shutdown mode or V_{CC} = 0 is 10MΩ as is implied by the leakage specification.

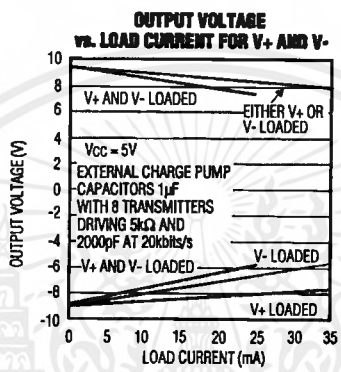
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

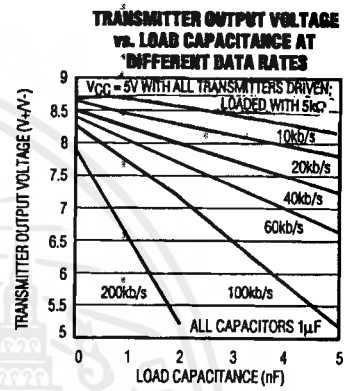
+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

Typical Operating Characteristics

MAX225/MAX244-MAX249







เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

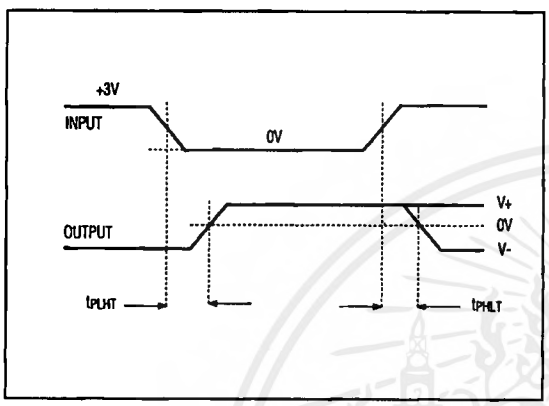


Figure 1. Transmitter Propagation Delay Timing

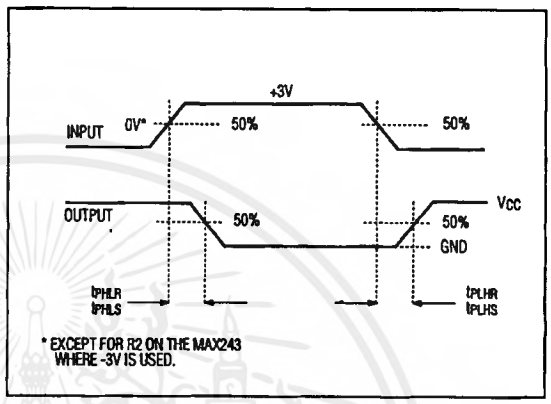


Figure 2. Receiver Propagation Delay Timing

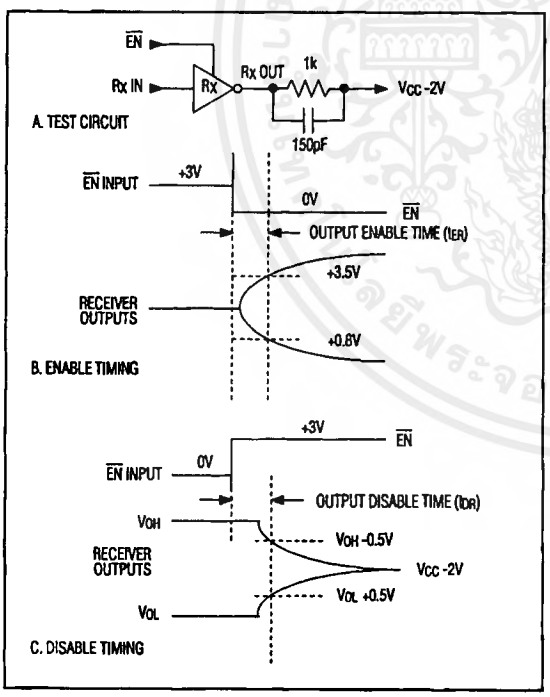


Figure 3. Receiver-Output Enable and Disable Timing

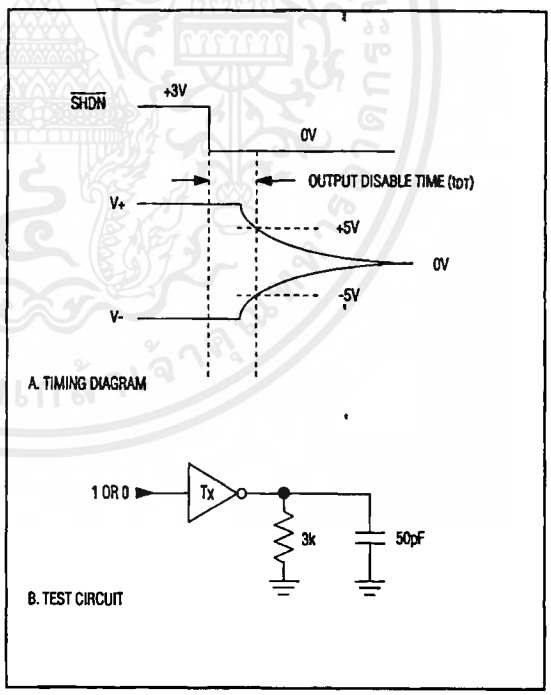


Figure 4. Transmitter-Output Disable Timing

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

Table 1a. MAX225 Control Pin Configurations

$\overline{ENT}$	$\overline{ENR}$	OPERATION STATUS	TRANSMITTERS	RECEIVERS
0	0	Normal Operation	All Active	All Active
0	1	Normal Operation	All Active	All 3-State
1	0	Shutdown	All 3-State	All Low-Power Receive Mode
1	1	Shutdown	All 3-State	All 3-State

Table 1b. MAX245 Control Pin Configurations

$\overline{ENT}$	$\overline{ENR}$	OPERATION STATUS	TRANSMITTERS		RECEIVERS	
			TA1-TA4	TB1-TB4	RA1-RA5	RB1-RB5
0	0	Normal Operation	All Active	All Active	All Active	All Active
0	1	Normal Operation	All Active	All Active	RA1-RA4 3-State RA5 Active	RB1-RB4 3-State RB5 Active
1	0	Shutdown	All 3-State	All 3-State	All Low Power Receiver Mode	All Low Power Receiver Mode
1	1	Shutdown	All 3-State	All 3-State	RA1-RA4 3-State RA5 Low-Power Receiver Mode	RB1-RB4 3-State RA5 Low-Power Receiver Mode

Table 1c. MAX246 Control Pin Configurations

$\overline{ENA}$	$\overline{ENB}$	OPERATION STATUS	TRANSMITTERS		RECEIVERS	
			TA1-TA4	TB1-TB4	RA1-RA5	RB1-RB5
0	0	Normal Operation	All Active	All Active	All Active	All Active
0	1	Normal Operation	All Active	All 3-State	All Active	RB1-RB4 3-State RB5 Active
1	0	Shutdown	All 3-State	All Active	RA1-RA4 3-State RA5 Active	All Active
1	1	Shutdown	All 3-State	All 3-State	RA1-RA4 3-State RA5 Low-Power Receiver Mode	RB1-RB4 3-State RA5 Low-Power Receiver Mode

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Table 1d. MAX247/248/249 Control Pin Configurations

ENTA	ENTB	ENRA	ENRB	OPERATION STATUS		TRANSMITTERS		RECEIVERS	
					MAX247	TA1–TA4	TB1–TB4	RA1–RA4	RB1–RB5
					MAX248	TA1–TA4	TB1–TB4	RA1–RA4	RB1–RB4
					MAX249	TA1–TA3	TB1–TB3	RA1–RA5	RB1–RB5
0	0	0	0	Normal Operation		All Active	All Active	All Active	All Active
0	0	0	1	Normal Operation		All Active	All Active	All Active	All 3-State, except RB5 stays active on MAX247
0	0	1	0	Normal Operation		All Active	All Active	All 3-State	All Active
0	0	1	1	Normal Operation		All Active	All Active	All 3-State	All 3-State, except RB5 stays active on MAX247
0	1	0	0	Normal Operation		All Active	All 3-State	All Active	All Active
0	1	0	1	Normal Operation		All Active	All 3-State	All Active	All 3-State, except RB5 stays active on MAX247
0	1	1	0	Normal Operation		All Active	All 3-State	All 3-State	All Active
0	1	1	1	Normal Operation		All Active	All 3-State	All 3-State	All 3-State, except RB5 stays active on MAX247
1	0	0	0	Normal Operation		All 3-State	All Active	All Active	All Active
1	0	0	1	Normal Operation		All 3-State	All Active	All Active	All 3-State, except RB5 stays active on MAX247
1	0	1	0	Normal Operation		All 3-State	All Active	All 3-State	All Active
1	0	1	1	Normal Operation		All 3-State	All Active	All 3-State	All 3-State, except RB5 stays active on MAX247
1	1	0	0	Shutdown		All 3-State	All 3-State	Low-Power Receive Mode	Low-Power Receive Mode
1	1	0	1	Shutdown		All 3-State	All 3-State	Low-Power Receive Mode	All 3-State, except RB5 stays active on MAX247
1	1	1	0	Shutdown		All 3-State	All 3-State	All 3-State	Low-Power Receive Mode
1	1	1	1	Shutdown		All 3-State	All 3-State	All 3-State	All 3-State, except RB5 stays active on MAX247

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

Detailed Description

The MAX220-MAX249 contain four sections: dual charge-pump DC-DC voltage converters, RS-232 drivers, RS-232 receivers, and receiver and transmitter enable control inputs.

Dual Charge-Pump Voltage Converter

The MAX220-MAX249 have two internal charge-pumps that convert +5V to ±10V (unloaded) for RS-232 driver operation. The first converter uses capacitor C1 to double the +5V input to +10V on C3 at the V+ output. The second converter uses capacitor C2 to invert +10V to -10V on C4 at the V- output.

A small amount of power may be drawn from the +10V (V+) and -10V (V-) outputs to power external circuitry (see Typical Operating Characteristics), except on the MAX225 and MAX245-MAX247, where these pins are not available. V+ and V- are not regulated, so the output voltage drops with increasing load current. Do not load V+ and V- to a point that violates the minimum ±5V EIA/TIA-232E driver output voltage when sourcing current from V+ and V- to external circuitry.

When using the shutdown feature in the MAX222, MAX225, MAX230, MAX235, MAX236, MAX240, MAX241, and MAX245-MAX249 avoid using V+ and V- to power external circuitry. When these parts are shut down, V- falls to 0V, and V+ falls to +5V. For applications where a +10V external supply is applied to the V+ pin (instead of using the internal charge pump to generate +10V), the C1 capacitor must not be installed and the $\overline{SHDN}$ pin must be tied to V_{CC}. This is because V+ is internally connected to V_{CC} in shutdown mode.

RS-232 Drivers

The typical driver output voltage swing is ±8V when loaded with a nominal 5kΩ RS-232 receiver and V_{CC} = +5V. Output swing is guaranteed to meet the EIA/TIA-232E and V.28 specification, which calls for ±5V minimum driver output levels under worst-case conditions. These include a minimum 3kΩ load, V_{CC} = +4.5V, and maximum operating temperature. Unloaded driver output voltage ranges from (V+ -1.3V) to (V- +0.5V).

Input thresholds are both TTL and CMOS compatible. The inputs of unused drivers can be left unconnected since 400kΩ input pull-up resistors to V_{CC} are built-in. The pull-up resistors force the outputs of unused drivers low because all drivers invert. The internal input pull-up resistors typically source 12µA, except in shutdown mode where the pull-ups are disabled. Driver outputs turn off and enter a high-impedance state—where leakage current is typically microamperes (maximum 25µA)—when in shutdown mode, in three-state mode, or when device power is removed. Outputs can be driven to ±15V. The power-supply current typically drops to 8µA in shutdown mode.

The MAX239 has a receiver 3-state control line, and the MAX223, MAX225, MAX235, MAX236, MAX240, and MAX241 have both a receiver 3-state control line and a low-power shutdown control. The receiver TTL/CMOS outputs are in a high-impedance, 3-state mode whenever the 3-state $\overline{EN}$able line is high, and are also high-impedance whenever the shutdown control line is high.

When in low-power shutdown mode, the driver outputs are turned off and their leakage current is less than 1µA with the driver output pulled to ground. The driver output leakage remains less than 1µA, even if the transmitter output is backdriven between 0V and (V_{CC} + 6V). Below -0.5V, the transmitter is diode clamped to ground with 1kΩ series impedance. The transmitter is also zener clamped to approximately V_{CC} + 6V, with a series impedance of 1kΩ.

The driver output slew rate is limited to less than 30V/µs as required by the EIA/TIA-232E and V.28 specifications. Typical slew rates are 24V/µs unloaded and 10V/µs loaded with 3Ω and 2500pF.

RS-232 Receivers

EIA/TIA-232E and V.28 specifications define a voltage level greater than 3V as a logic 0, so all receivers invert. Input thresholds are set at 0.8V and 2.4V, so receivers respond to TTL level inputs as well as EIA/TIA-232E and V.28 levels.

The receiver inputs withstand an input overvoltage up to ±25V and provide input terminating resistors with nominal 5kΩ values. The receivers implement Type 1 interpretation of the fault conditions of V.28 and EIA/TIA-232E.

The receiver input hysteresis is typically 0.5V with a guaranteed minimum of 0.2V. This produces clear output transitions with slow-moving input signals, even with moderate amounts of noise and ringing. The receiver propagation delay is typically 600ns and is independent of input swing direction.

Low-Power Receive Mode

The low-power receive-mode feature of the MAX223, MAX242, and MAX245-MAX249 puts the IC into shutdown mode, but still allows it to receive information. This is important for applications where systems are periodically awakened to look for activity. Using low-power receive mode, the system can still receive a signal that will activate it on command and prepare it for communication at faster data rates. This operation conserves system power.

Negative Threshold—MAX243

The MAX243 is pin compatible with the MAX232A, differing only in that RS-232 cable fault protection is removed on one of the two receiver inputs. This means that control lines such as CTS and RTS can either be driven or left floating without interrupting communication. Different cables are not needed to interface with different pieces of equipment.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

The input threshold of the receiver without cable fault protection is -0.8V rather than +1.4V. Its output goes positive only if the input is connected to a control line that is actively driven negative. If not driven, it defaults to the 0 or "OK to send" state. Normally, the MAX243's other receiver (+1.4V threshold) is used for the data line (TD or RD), while the negative threshold receiver is connected to the control line (DTR, DTS, CTS, RTS, etc.).

Other members of the RS-232 family implement the optional cable fault protection as specified by EIA/TIA-232E specifications. This means a receiver output goes high whenever its input is driven negative, left floating, or shorted to ground. The high output tells the serial communications IC to stop sending data. To avoid this, the control lines must either be driven or connected with jumpers to an appropriate positive voltage level.

Shutdown—MAX222-MAX242

On the MAX222, MAX235, MAX236, MAX240, and MAX241, all receivers are disabled during shutdown. On the MAX223 and MAX242, two receivers continue to operate in a reduced power mode when the chip is in shutdown. Under these conditions, the propagation delay increases to about 2.5µs for a high-to-low input transition. When in shutdown the receiver acts as a CMOS inverter with no hysteresis. The MAX223 and MAX242 also have a receiver output enable input ($\overline{EN}$) that allows receiver output control independent of $\overline{SHDN}$. With all other devices, $\overline{SHDN}$ also disables the receiver outputs.

The MAX225 provides five transmitters and five receivers, while the MAX245 provides ten receivers and eight transmitters. Both devices have separate receiver and transmitter-enable controls. The charge pumps turn off and the devices shut down when a logic high is applied to the ENT input. In this state, the supply current drops to less than 25µA and the receivers continue to operate in a low-power receive mode. Driver outputs enter a high-impedance state (three-state mode). On the MAX225, all five receivers are controlled by the $\overline{ENR}$ input. On the MAX245, eight of the receiver outputs are controlled by the $\overline{ENR}$ input, while the remaining two receivers (RA5 and RB5) are always active. RA1-RA4 and RB1-RB4 are put in a three-state mode when $\overline{ENR}$ is a logic high.

Receiver and Transmitter Enable Control Inputs

The MAX225 and MAX245-MAX249 feature transmitter and receiver enable controls.

The receivers have three modes of operation: full-speed receive (normal active), three-state (disabled), and low-power receive (enabled receivers continue to function at lower data rates). The receiver enable inputs control the full-speed receive and three-state modes. The transmitters have two modes of operation: full-speed transmit (normal active) and three-state (disabled). The transmitter enable inputs also control the shutdown mode. The device enters shutdown mode when all transmitters are disabled. Enabled receivers function in the low-power receive mode when in shutdown.

Tables 1a-1d define the control states. The MAX244 has no control pins and is not included in these tables.

The MAX246 has ten receivers and eight drivers with two control pins, each controlling one side of the device. A logic high at the A-side control input ($\overline{ENA}$) causes the four A-side receivers and drivers to go into a three-state mode. Similarly, the B-side control input ($\overline{ENB}$) causes the four B-side drivers and receivers to go into a three-state mode. As in the MAX245, one A-side and one B-side receiver (RA5 and RB5) remain active at all times. The entire device is put into shutdown mode when both the A and B sides are disabled ($\overline{ENA}$ = $\overline{ENB}$ = +5V).

The MAX247 provides nine receivers and eight drivers with four control pins. The $\overline{ENRA}$ and $\overline{ENRB}$ receiver enable inputs each control four receiver outputs. The $\overline{ENTA}$ and $\overline{ENTB}$ transmitter enable inputs each control four drivers. The ninth receiver (RB5) is always active. The device enters shutdown mode with a logic high on both $\overline{ENTA}$ and $\overline{ENTB}$.

The MAX248 provides eight receivers and eight drivers with four control pins. The $\overline{ENRA}$ and $\overline{ENRB}$ receiver enable inputs each control four receiver outputs. The $\overline{ENTA}$ and $\overline{ENTB}$ transmitter enable inputs control four drivers each. This part does not have an always-active receiver. The device enters shutdown mode and transmitters go into a three-state mode with a logic high on both $\overline{ENTA}$ and $\overline{ENTB}$.

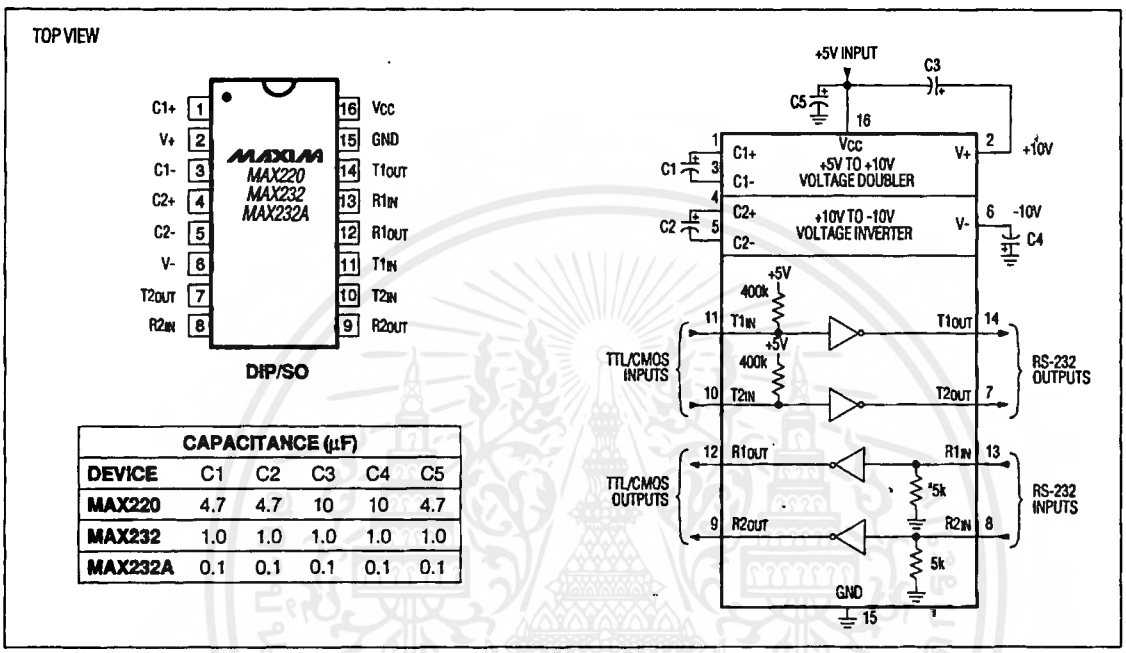
The MAX249 provides ten receivers and six drivers with four control pins. The $\overline{ENRA}$ and $\overline{ENRB}$ receiver enable inputs each control five receiver outputs. The $\overline{ENTA}$ and $\overline{ENTB}$ transmitter enable inputs control three drivers each. There is no always-active receiver. The device enters shutdown mode and transmitters go into a three-state mode with a logic high on both $\overline{ENTA}$ and $\overline{ENTB}$. In shutdown mode, active receivers operate in a low-power receive mode at data rates up to 20kbits/s.

Applications Information

Figures 5 through 25 show pin configurations and typical operating circuits. In applications that are sensitive to power-supply noise, V_{CC} should be decoupled to ground with a capacitor of the same value as C1 and C2 connected as close as possible to the device.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figure 5. MAX220/232/232A Pin Configuration and Typical Operating Circuit

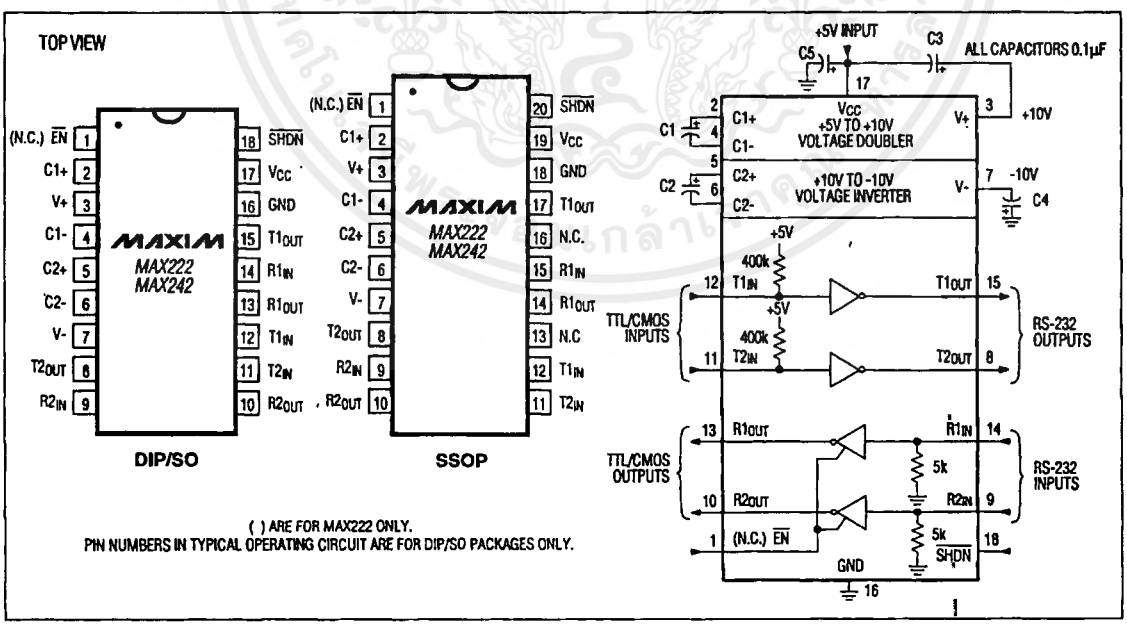


Figure 6. MAX222/MAX242 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

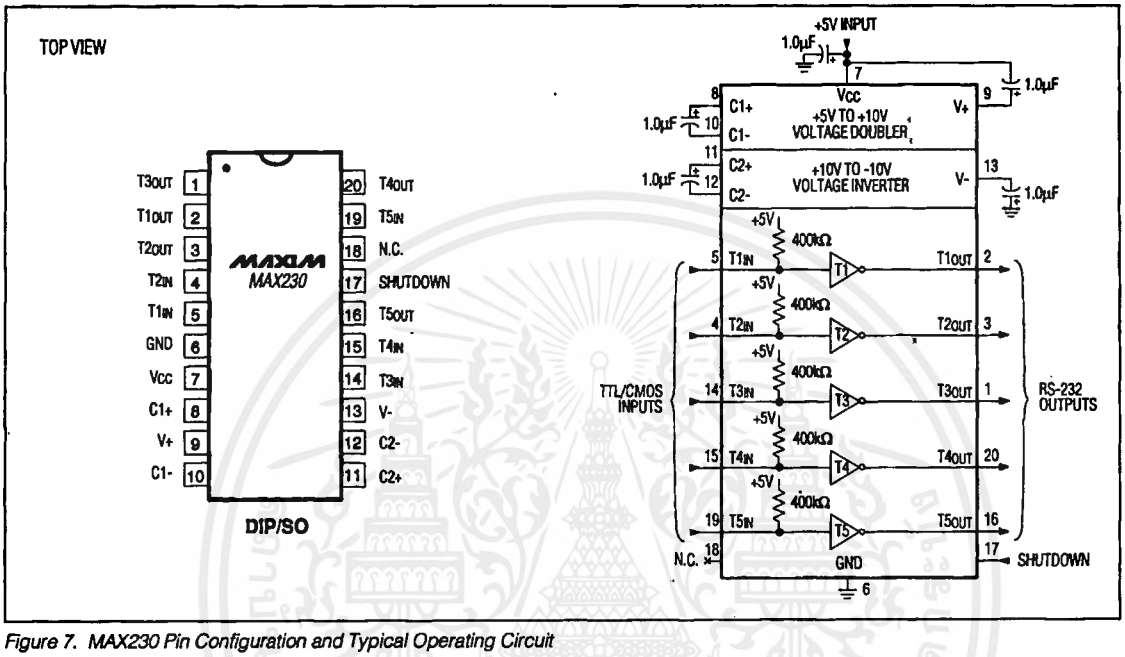


Figure 7. MAX230 Pin Configuration and Typical Operating Circuit

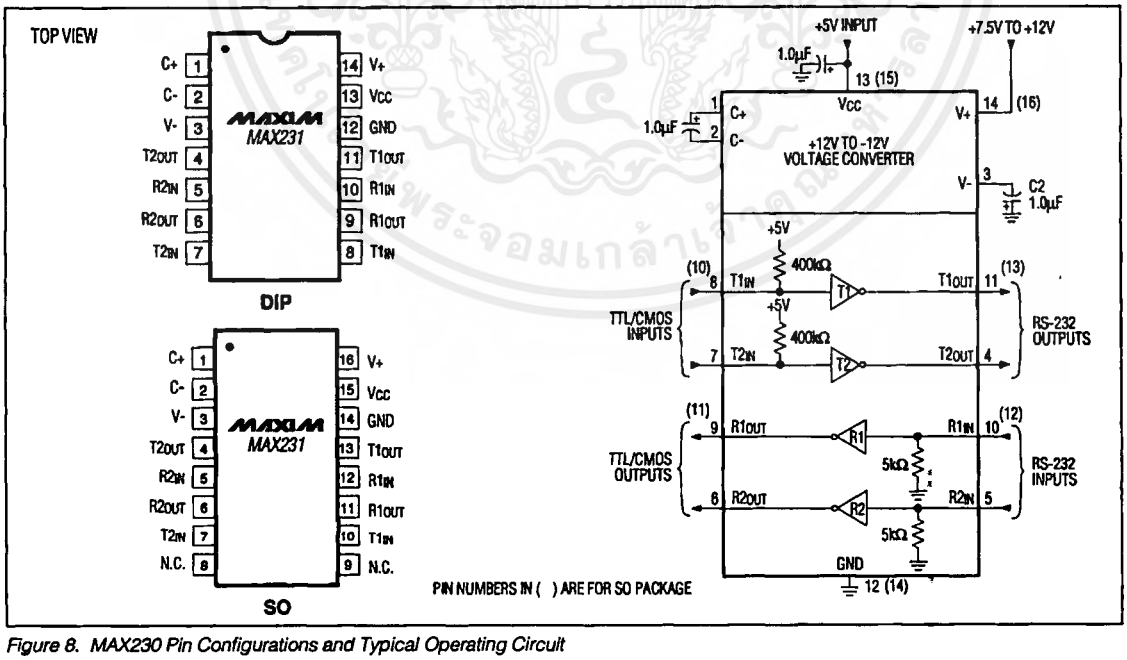


Figure 8. MAX230 Pin Configurations and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

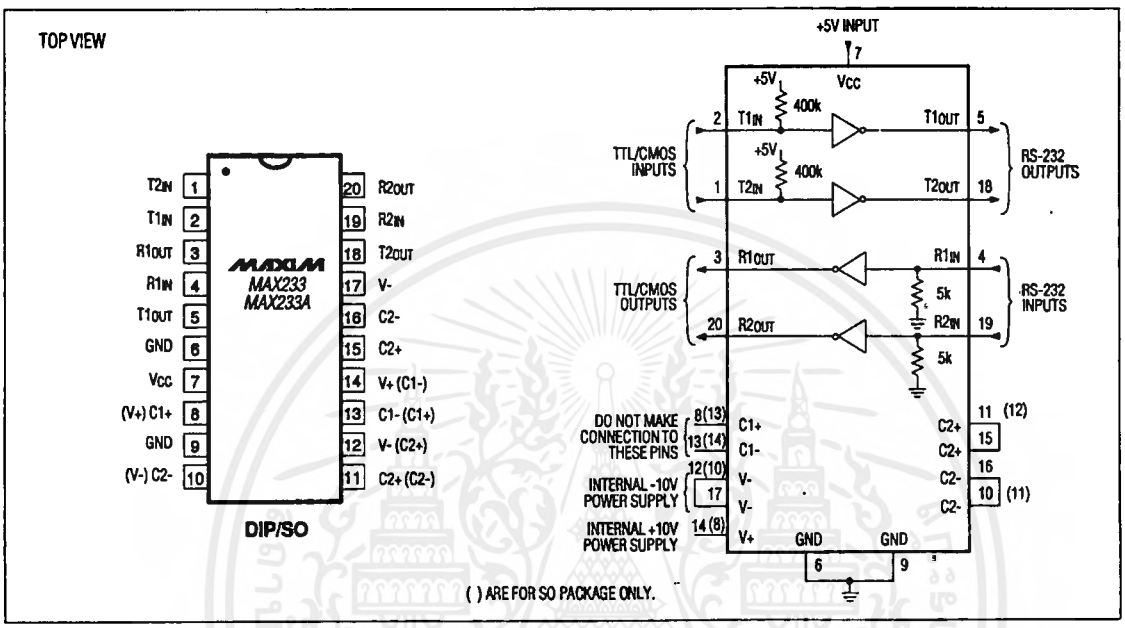


Figure 9. MAX233/MAX233A Pin Configuration and Typical Operating Circuit

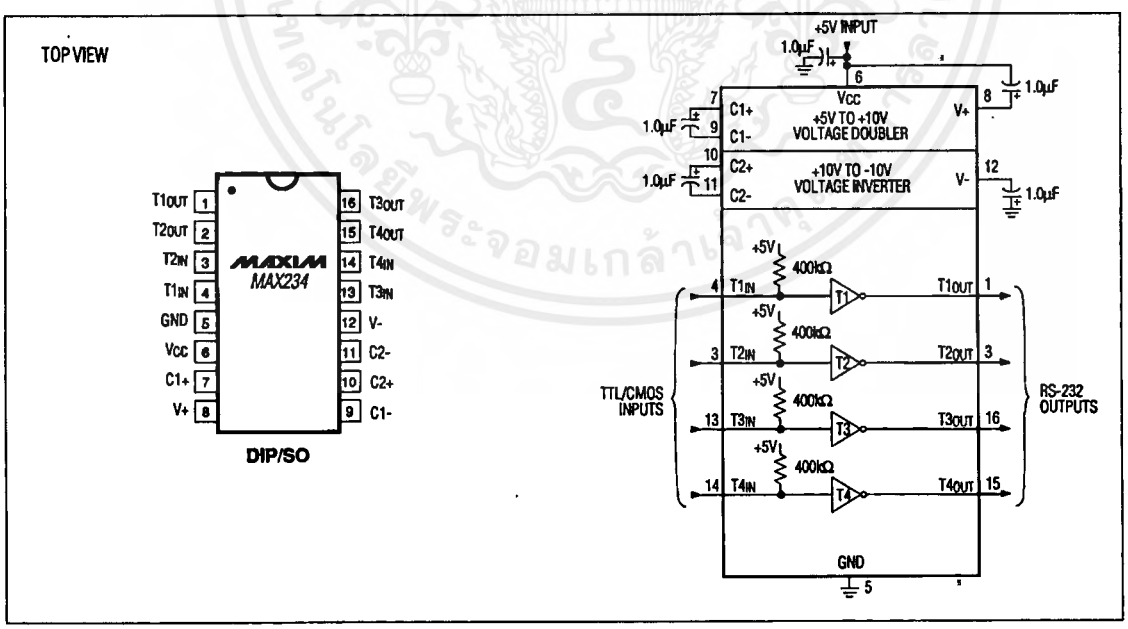


Figure 10. MAX234 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

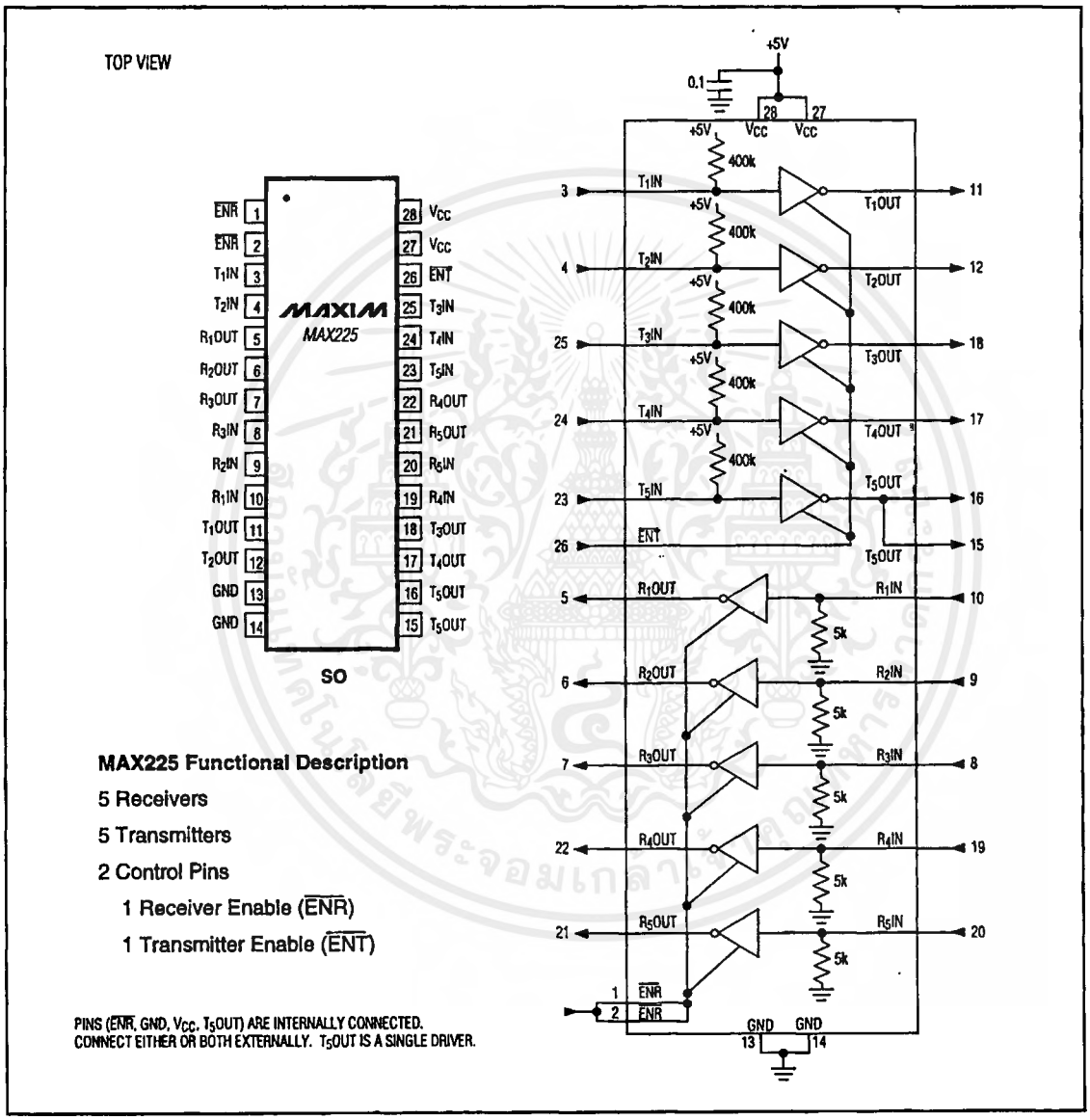


Figure 11. MAX225 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

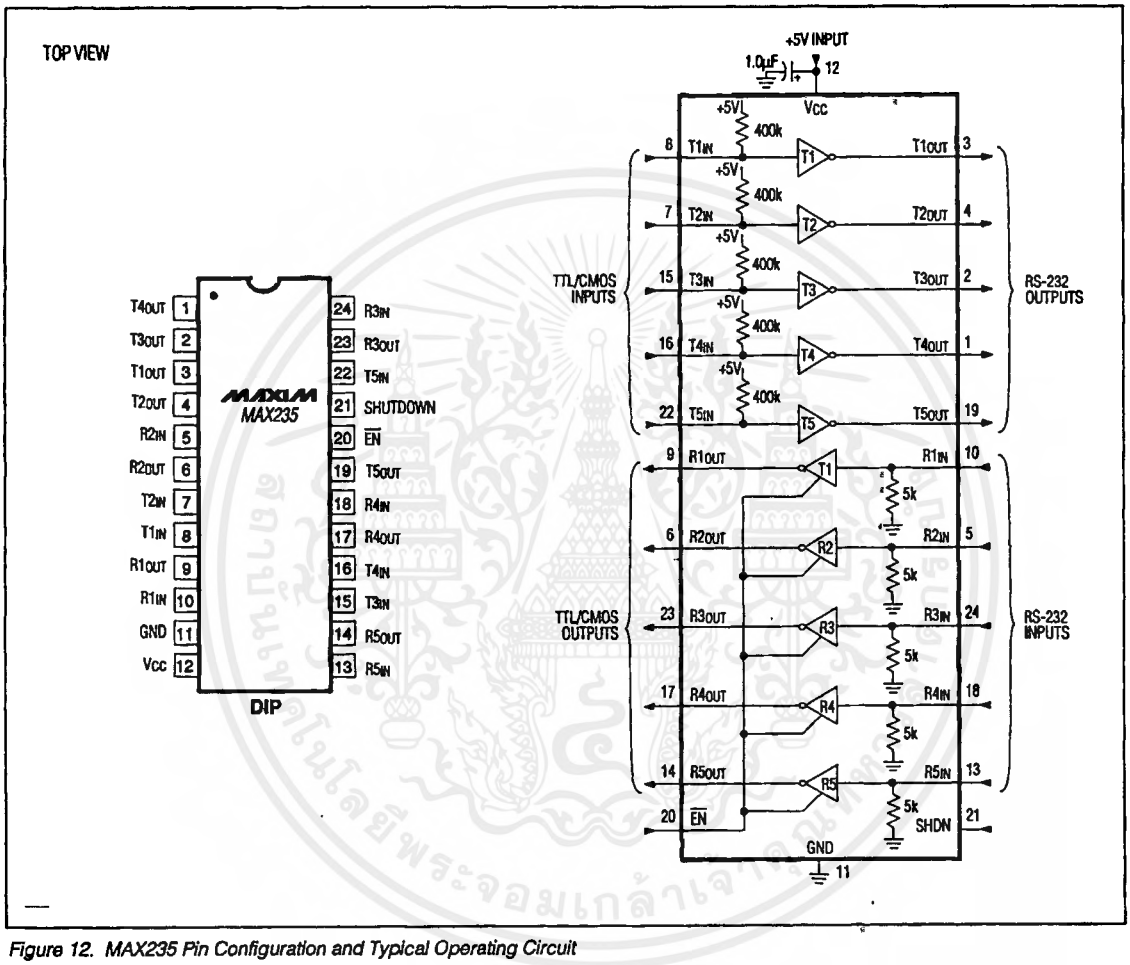


Figure 12. MAX235 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

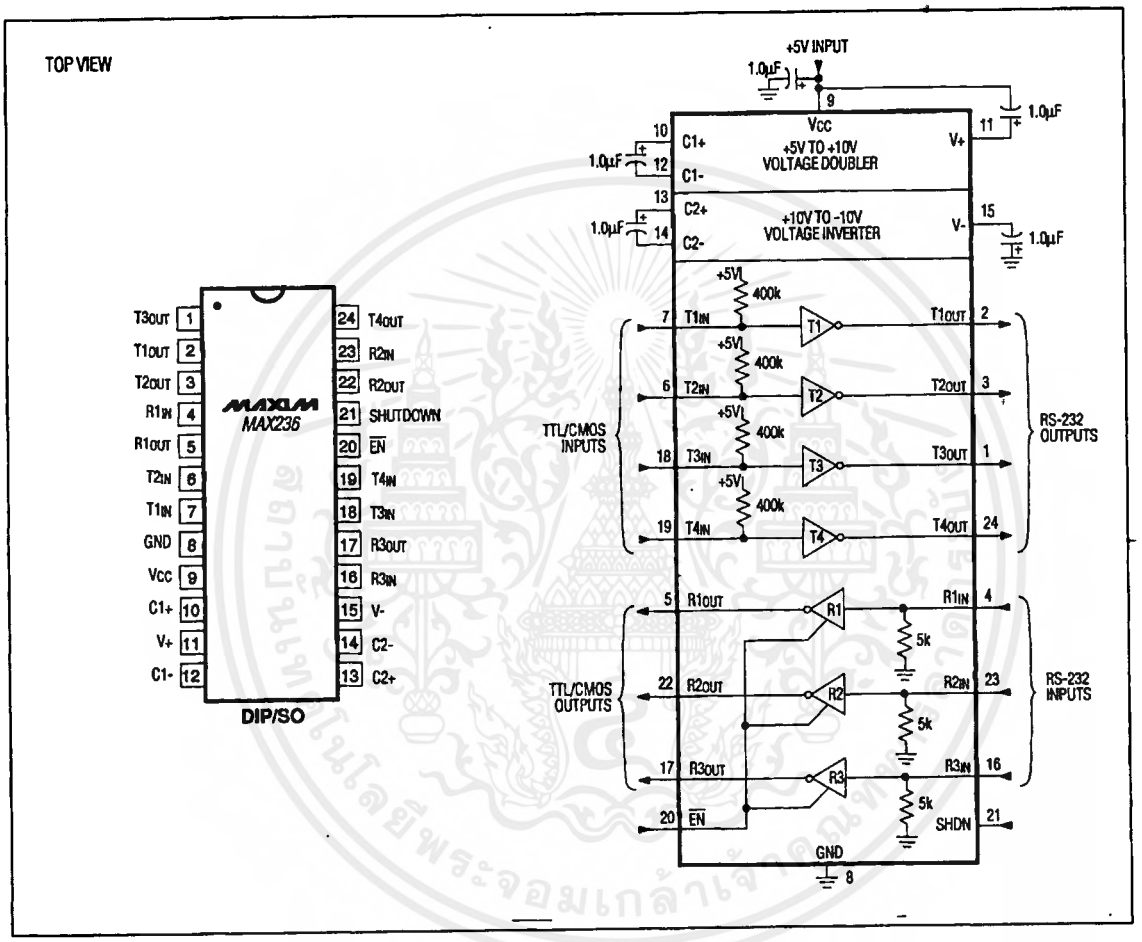


Figure 13. MAX236 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

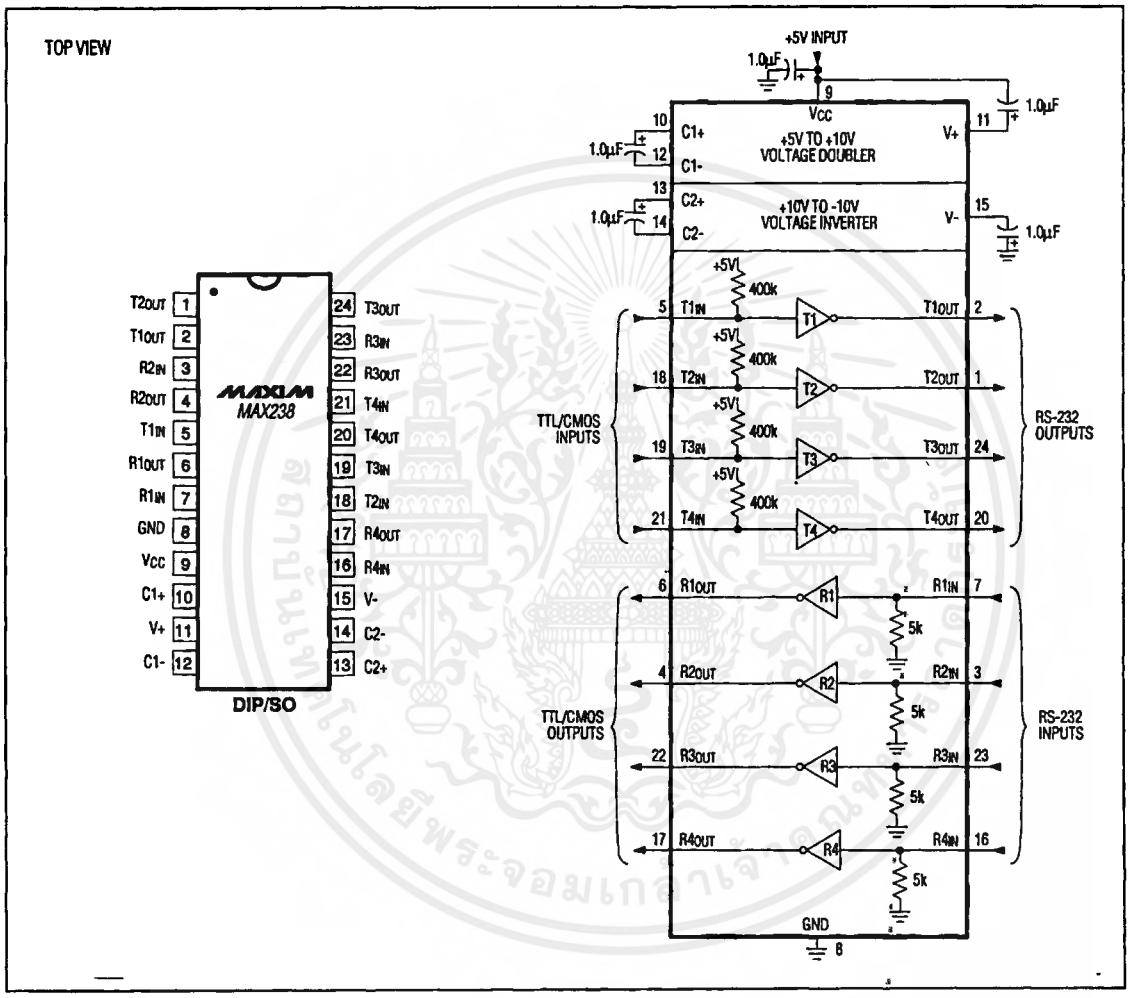


Figure 15. MAX238 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

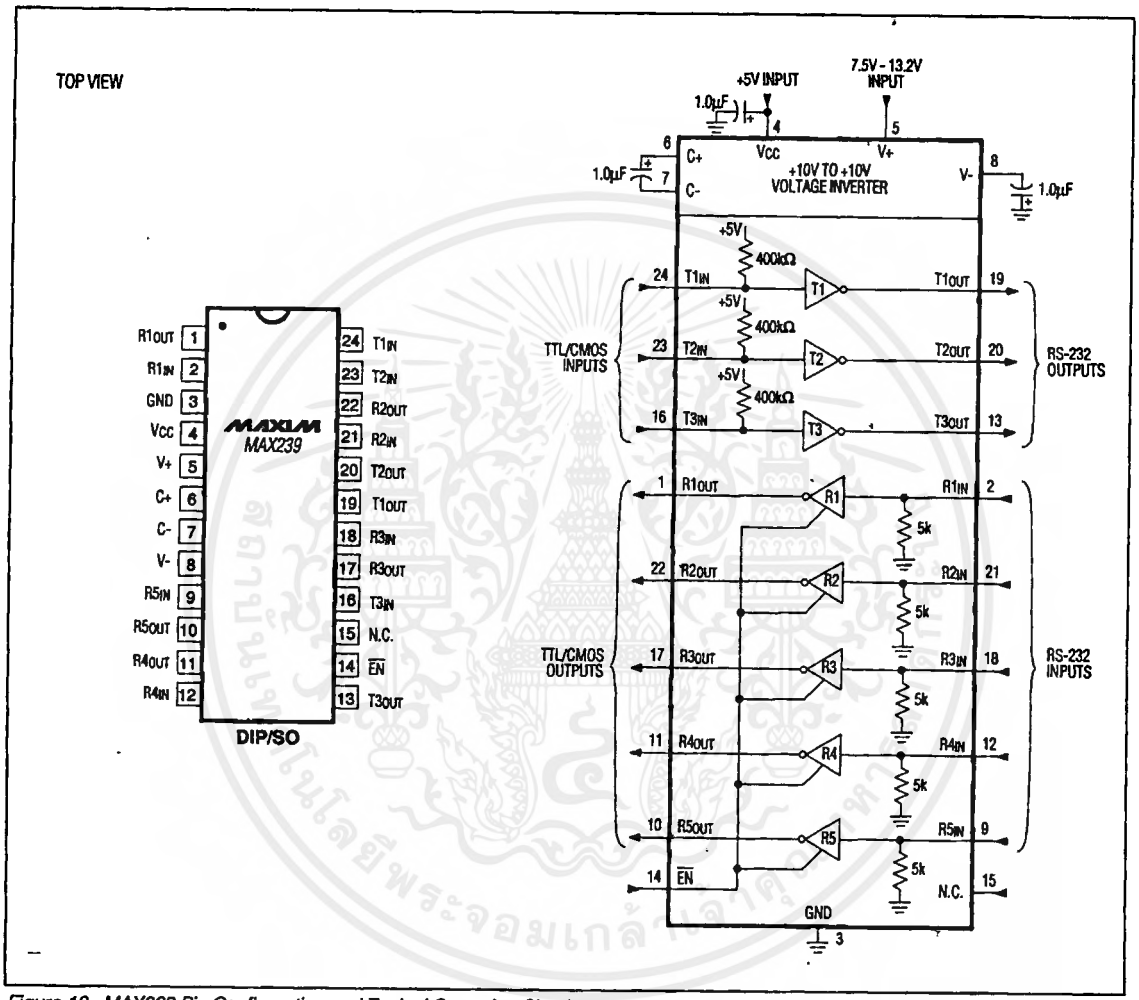


Figure 16. MAX239 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

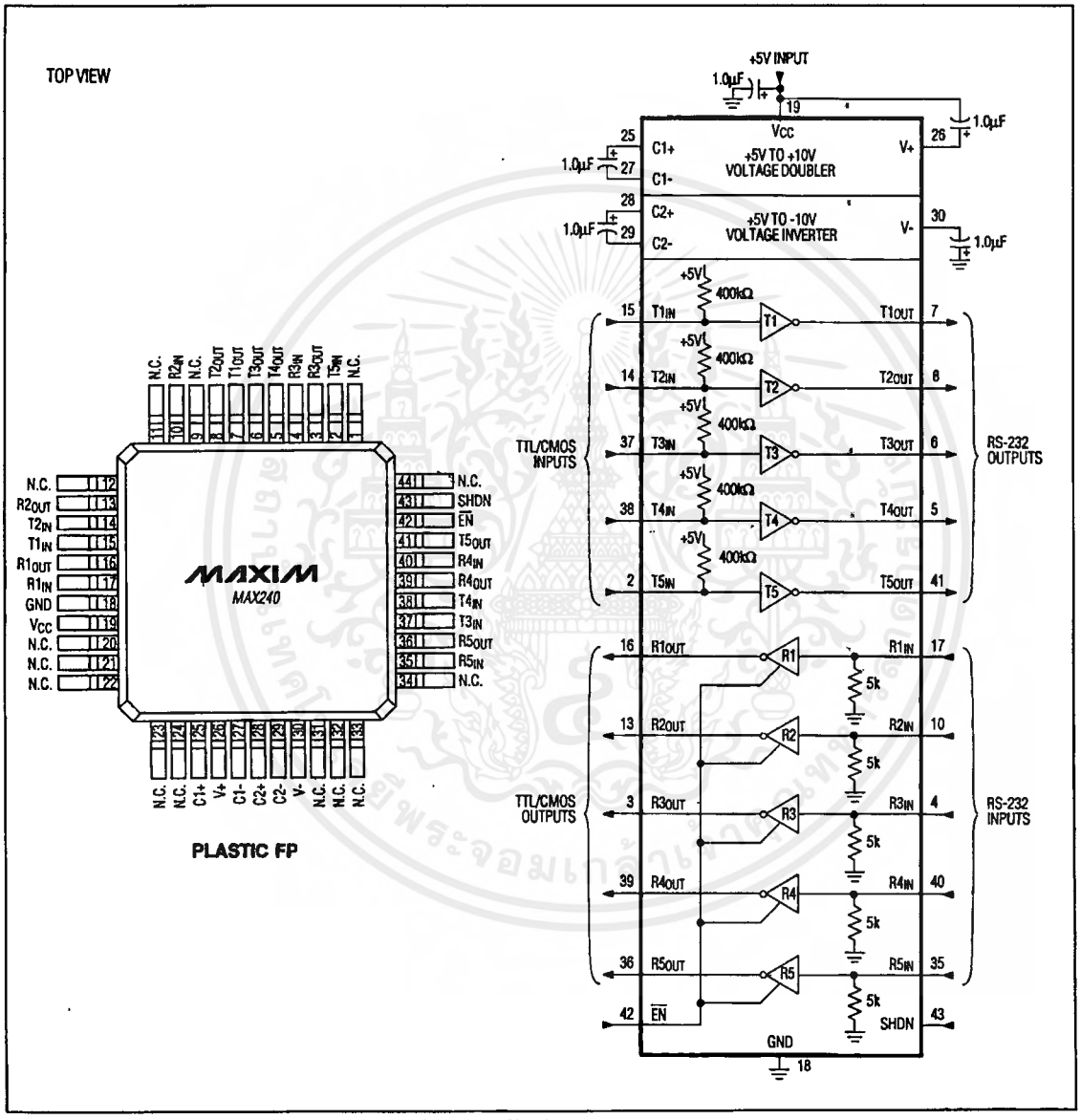


Figure 17. MAX240 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

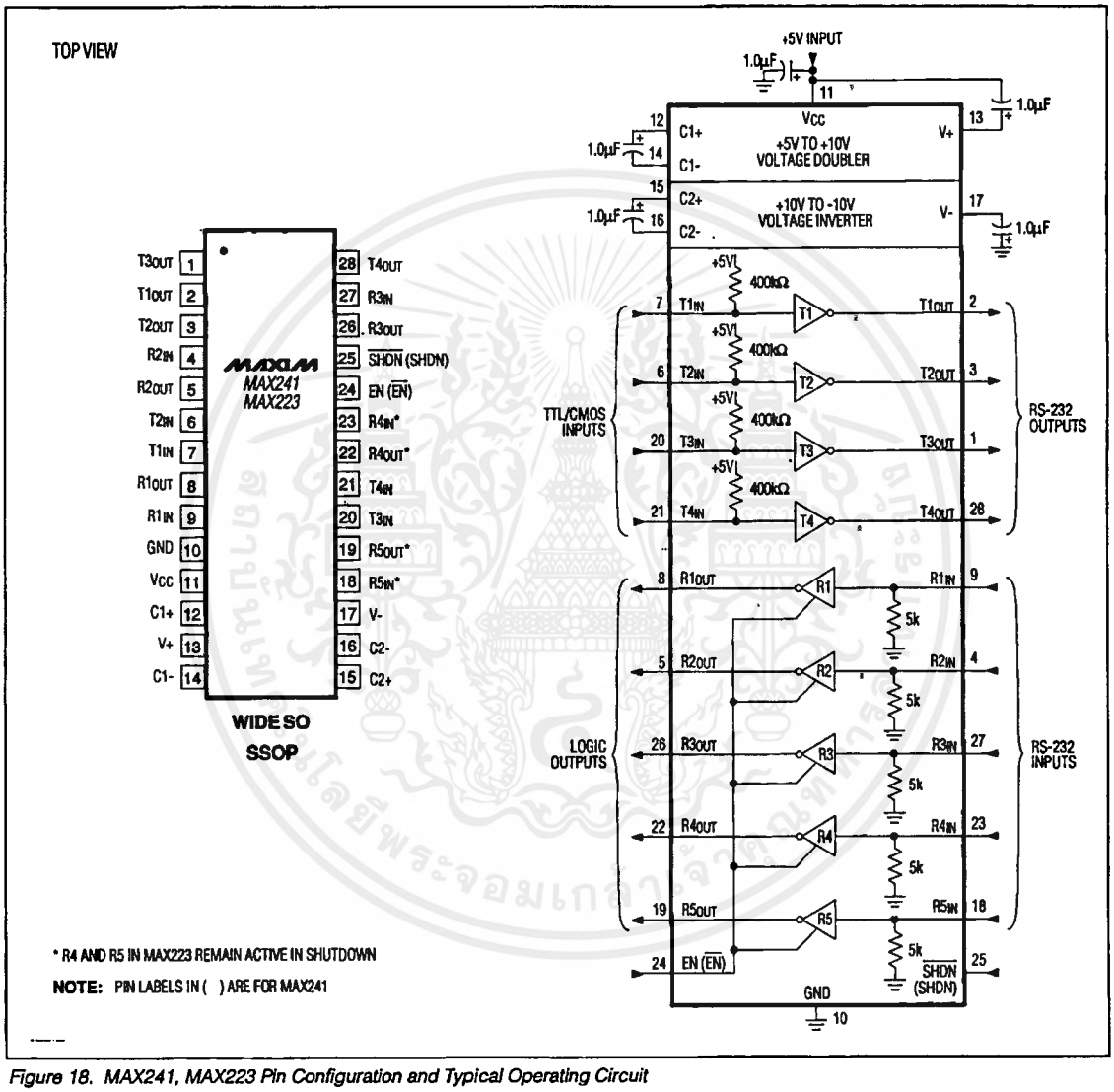


Figure 18. MAX241, MAX223 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

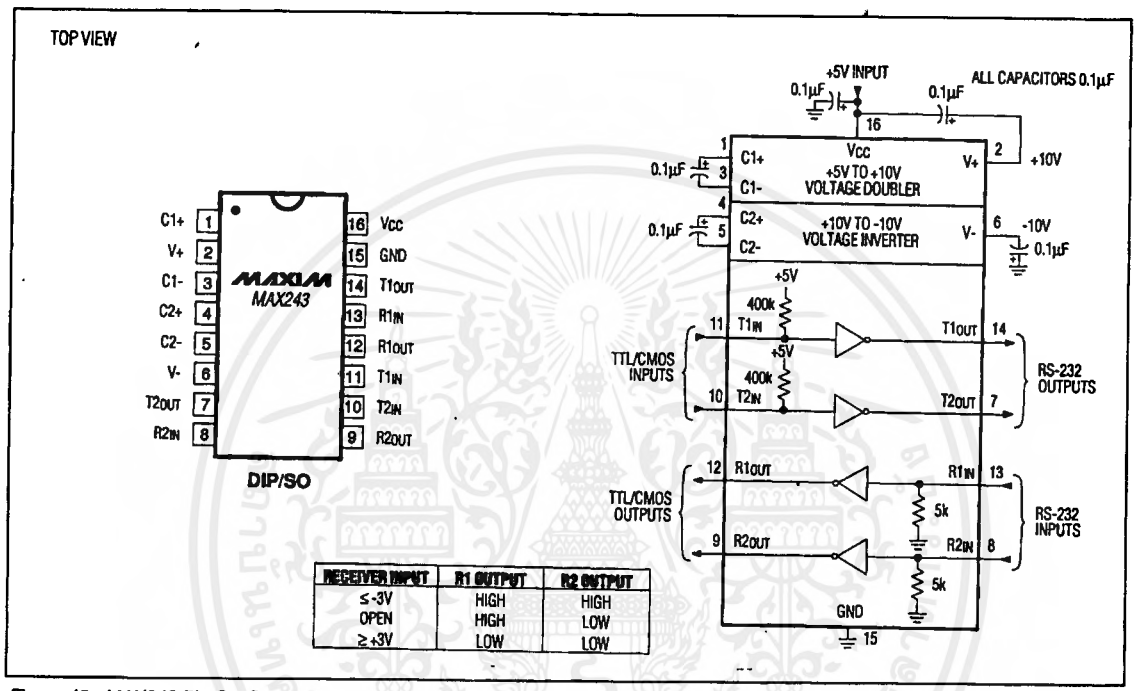
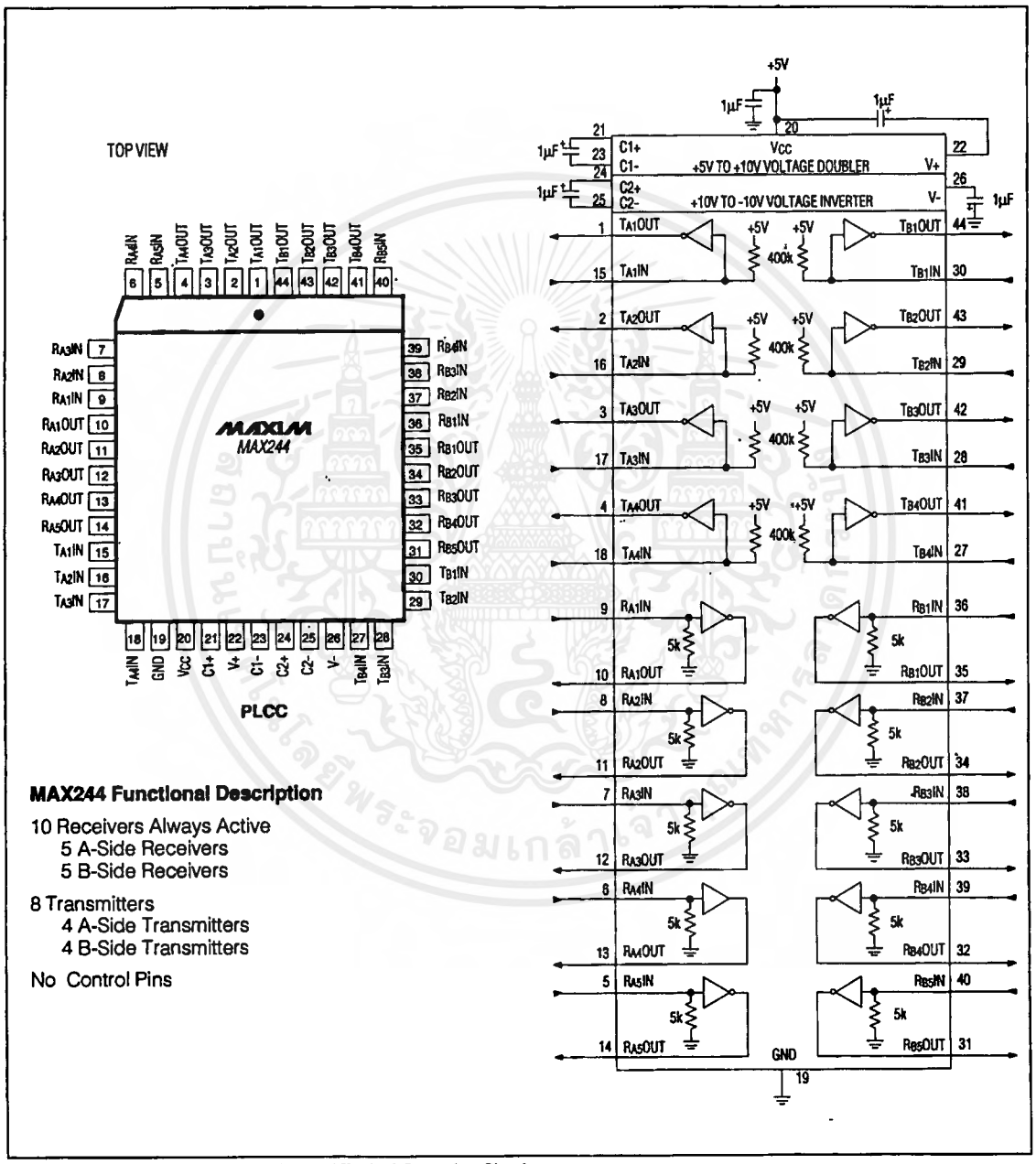


Figure 19. MAX243 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers



MAX244 Functional Description

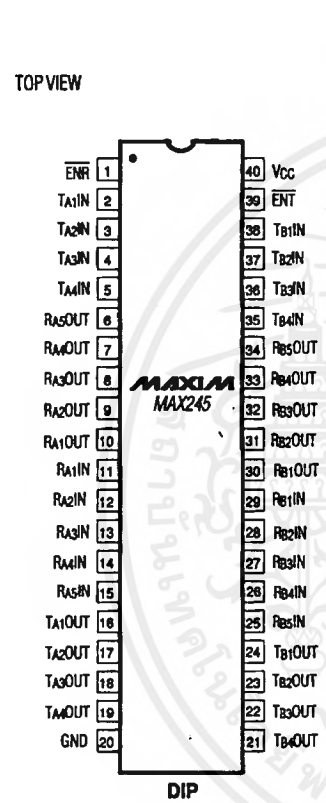
10 Receivers Always Active
- 5 A-Side Receivers
- 5 B-Side Receivers

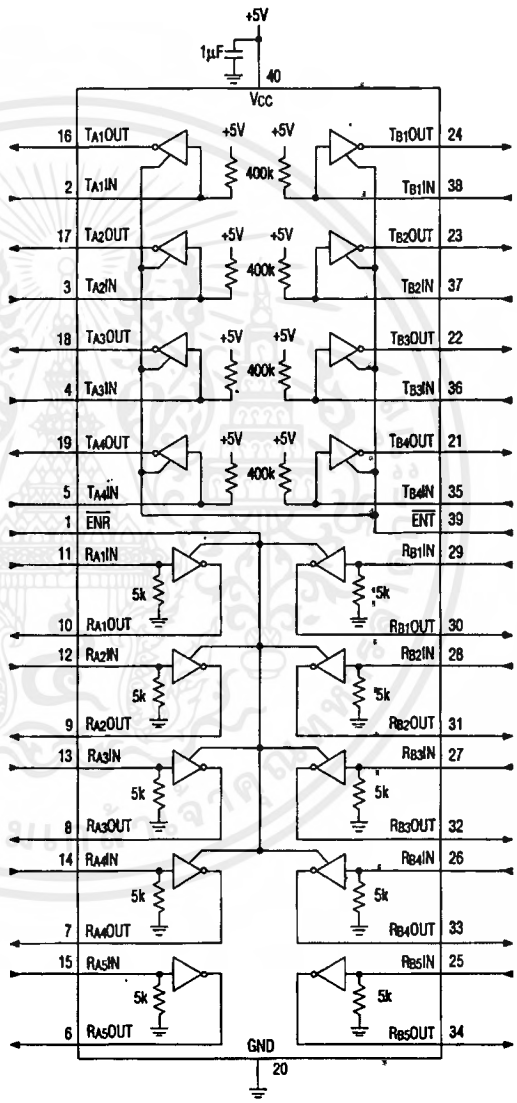
8 Transmitters
- 4 A-Side Transmitters
- 4 B-Side Transmitters

No Control Pins

Figure 20. MAX244 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้





MAX245 Functional Description

10 Receivers
- 5 A-Side Receivers (R_{A5} always active)
- 5 B-Side Receivers (R_{B5} always active)

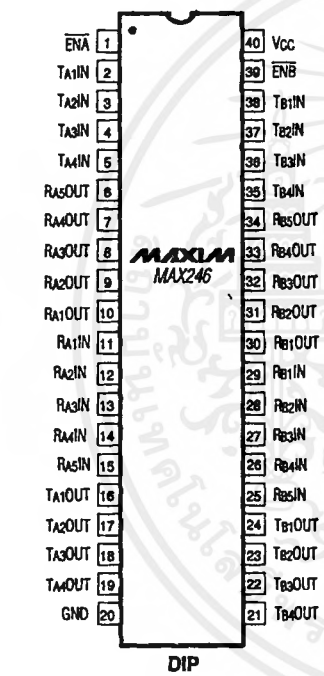
8 Transmitters
- 4 A-Side Transmitters
- 4 B-Side Transmitters

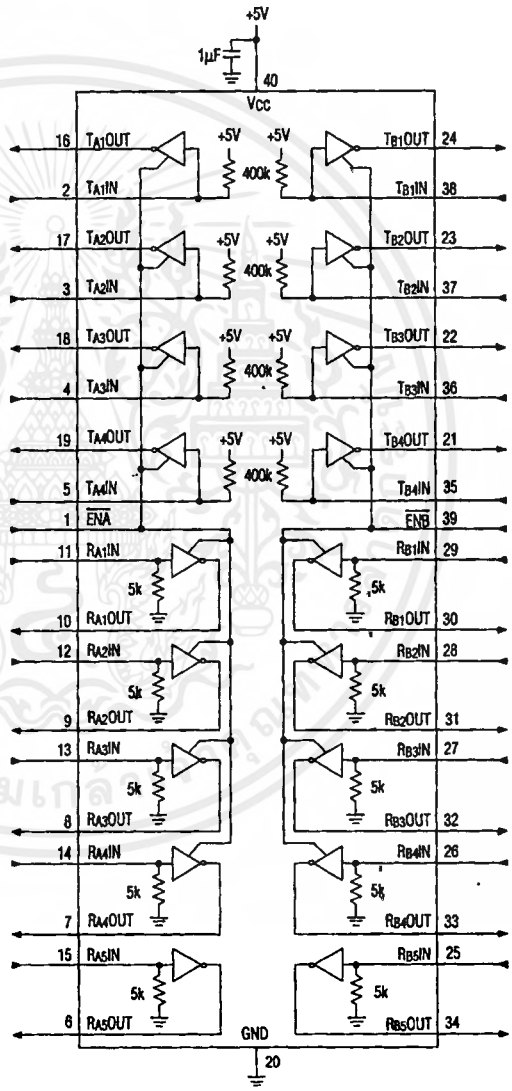
2 Control Pins
- 1 Receiver Enable ($\overline{ENR}$)
- 1 Transmitter Enable ($\overline{ENT}$)

Figure 21. MAX245 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers





MAX246 Functional Description

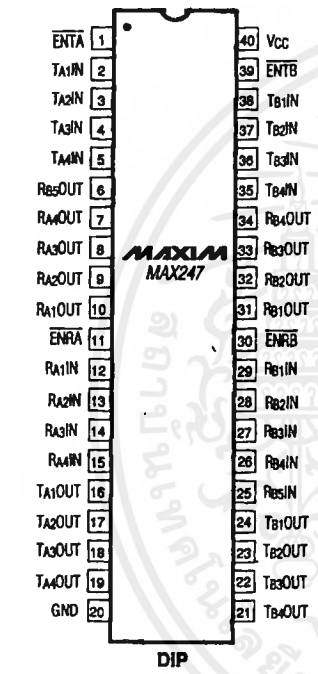
10 Receivers
- 5 A-Side Receivers (R_{A5} always active)
- 5 B-Side Receivers (R_{B5} always active)

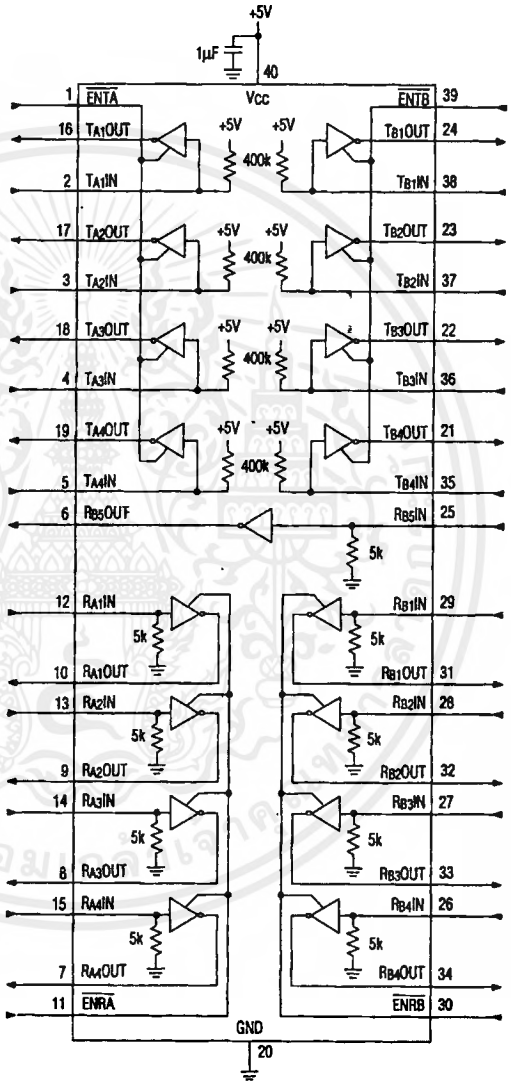
8 Transmitters
- 4 A-Side Transmitters
- 4 B-Side Transmitters

2 Control Pins
- Enable A-Side ($\overline{ENA}$)
- Enable B-Side ($\overline{ENB}$)

Figure 22. MAX246 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้





MAX247 Functional Description

9 Receivers
- 4 A-Side Receivers
- 5 B-Side Receivers (R_{B5} always active)

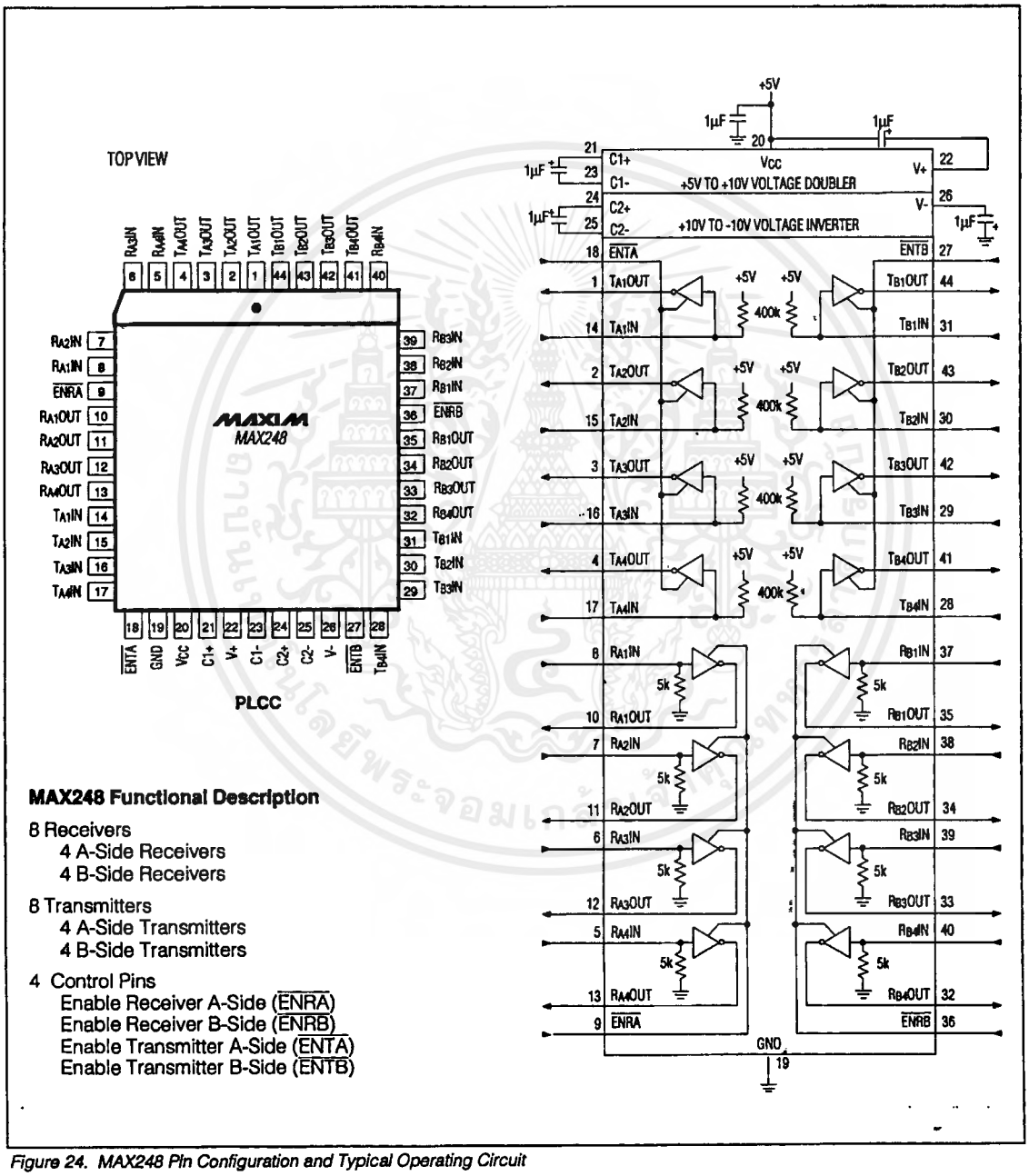
8 Transmitters
- 4 A-Side Transmitters
- 4 B-Side Transmitters

4 Control Pins
- Enable Receiver A-Side ($\overline{ENRA}$)
- Enable Receiver B-Side ($\overline{ENRB}$)
- Enable Transmitter A-Side ($\overline{ENTA}$)
- Enable Transmitter B-Side ($\overline{ENTB}$)

Figure 23. MAX247 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers



MAX248 Functional Description

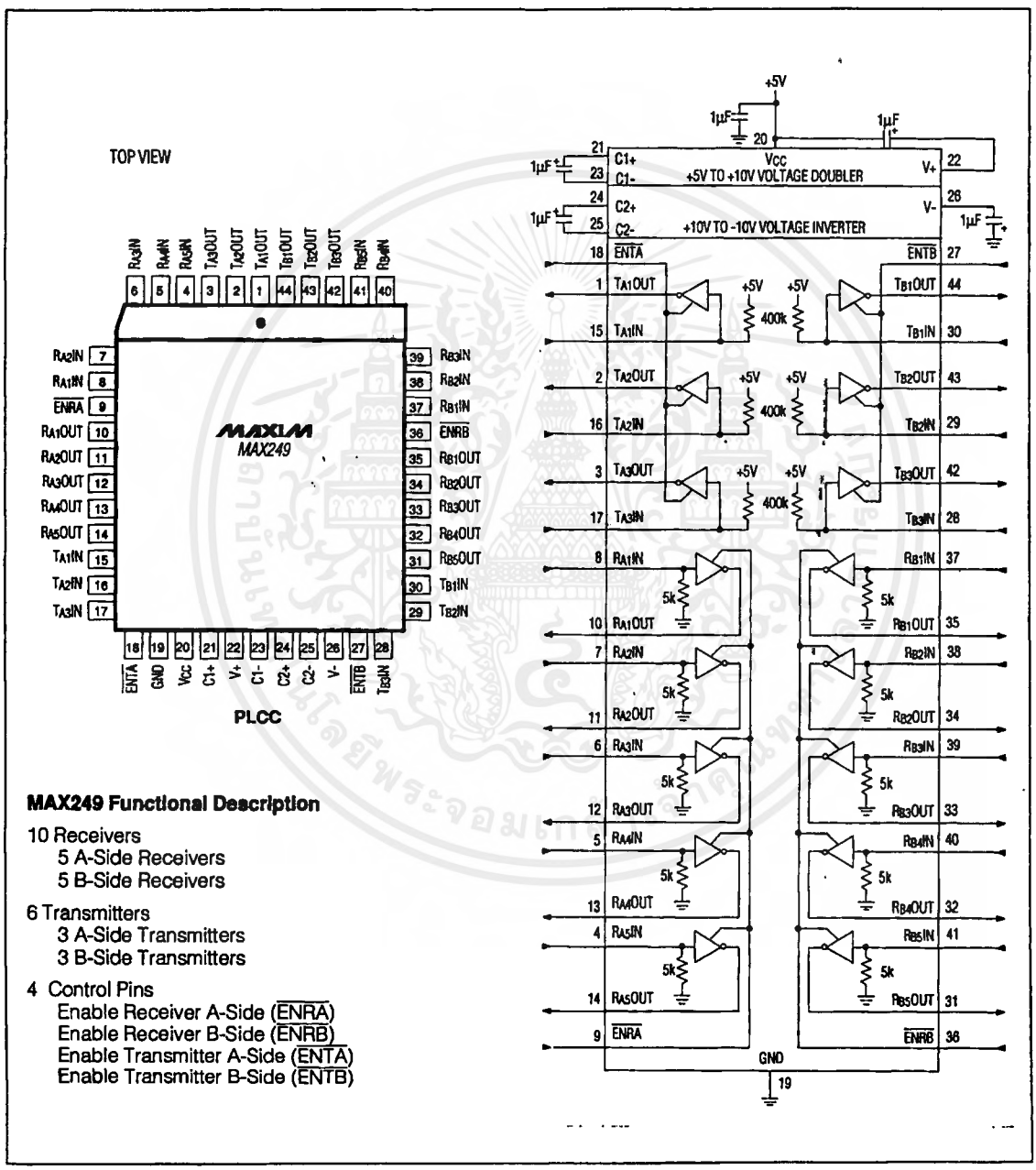
8 Receivers
- 4 A-Side Receivers
- 4 B-Side Receivers

8 Transmitters
- 4 A-Side Transmitters
- 4 B-Side Transmitters

4 Control Pins
- Enable Receiver A-Side ($\overline{ENRA}$)
- Enable Receiver B-Side ($\overline{ENRB}$)
- Enable Transmitter A-Side ($\overline{ENTA}$)
- Enable Transmitter B-Side ($\overline{ENTB}$)

Figure 24. MAX248 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



MAX249 Functional Description

10 Receivers
- 5 A-Side Receivers
- 5 B-Side Receivers

6 Transmitters
- 3 A-Side Transmitters
- 3 B-Side Transmitters

4 Control Pins
- Enable Receiver A-Side ($\overline{ENRA}$)
- Enable Receiver B-Side ($\overline{ENRB}$)
- Enable Transmitter A-Side ($\overline{ENTA}$)
- Enable Transmitter B-Side ($\overline{ENTB}$)

Figure 25. MAX249 Pin Configuration and Typical Operating Circuit

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

Ordering Information (continued)

PART	TEMP. RANGE	PIN-PACKAGE
MAX222CPN	0°C to +70°C	18 Plastic DIP
MAX222CWN	0°C to +70°C	18 Wide SO
MAX222C/D	0°C to +70°C	Dice*
MAX222EPN	-40°C to +85°C	18 Plastic DIP
MAX222EWN	-40°C to +85°C	18 Wide SO
MAX222EJN	-40°C to +85°C	18 CERDIP
MAX222MJN	-55°C to +125°C	18 CERDIP
MAX223CAI	0°C to +70°C	28 SSOP
MAX223CWI	0°C to +70°C	28 Wide SO
MAX223C/D	0°C to +70°C	Dice*
MAX223EAI	-40°C to +85°C	28 SSOP
MAX223EWI	-40°C to +85°C	28 Wide SO
MAX225CWI	0°C to +70°C	28 Wide SO
MAX225EWI	-40°C to +85°C	28 Wide SO
MAX230CPP	0°C to +70°C	20 Plastic DIP
MAX230CWP	0°C to +70°C	20 Wide SO
MAX230C/D	0°C to +70°C	Dice*
MAX230EPP	-40°C to +85°C	20 Plastic DIP
MAX230EWP	-40°C to +85°C	20 Wide SO
MAX230EJP	-40°C to +85°C	20 CERDIP
MAX230MJP	-55°C to +125°C	20 CERDIP
MAX231CPD	0°C to +70°C	14 Plastic DIP
MAX231CWE	0°C to +70°C	16 Wide SO
MAX231CJD	0°C to +70°C	14 CERDIP
MAX231C/D	0°C to +70°C	Dice*
MAX231EPD	-40°C to +85°C	14 Plastic DIP
MAX231EWE	-40°C to +85°C	16 Wide SO
MAX231EJD	-40°C to +85°C	14 CERDIP
MAX231MJD	-55°C to +125°C	14 CERDIP
MAX232CPE	0°C to +70°C	16 Plastic DIP
MAX232CSE	0°C to +70°C	16 Narrow SO
MAX232CWE	0°C to +70°C	16 Wide SO
MAX232C/D	0°C to +70°C	Dice*
MAX232EPE	-40°C to +85°C	16 Plastic DIP
MAX232ESE	-40°C to +85°C	16 Narrow SO
MAX232EWE	-40°C to +85°C	16 Wide SO
MAX232EJE	-40°C to +85°C	16 CERDIP
MAX232MJE	-55°C to +125°C	16 CERDIP
MAX232MLP	-55°C to +125°C	20 LCC
MAX232ACPE	0°C to +70°C	16 Plastic DIP
MAX232ACSE	0°C to +70°C	16 Narrow SO
MAX232ACWE	0°C to +70°C	16 Wide SO
MAX232AC/D	0°C to +70°C	Dice*
MAX232AEPE	-40°C to +85°C	16 Plastic DIP
MAX232AESE	-40°C to +85°C	16 Narrow SO
MAX232AEWE	-40°C to +85°C	16 Wide SO
MAX232AEJE	-40°C to +85°C	16 CERDIP
MAX232AMJE	-55°C to +125°C	16 CERDIP
MAX232AMLP	-55°C to +125°C	20 LCC
MAX233CPP	0°C to +70°C	20 Plastic DIP
MAX233EPP	-40°C to +85°C	20 Plastic DIP
MAX233ACPP	0°C to +70°C	20 Plastic DIP
MAX233ACWP	0°C to +70°C	20 Wide SO
MAX233AEPP	-40°C to +85°C	20 Plastic DIP
MAX233AEWP	-40°C to +85°C	20 Wide SO
MAX234CPE	0°C to +70°C	16 Plastic DIP
MAX234CWE	0°C to +70°C	16 Wide SO
MAX234C/D	0°C to +70°C	Dice*
MAX234EPE	-40°C to +85°C	16 Plastic DIP
MAX234EWE	-40°C to +85°C	16 Wide SO
MAX234EJE	-40°C to +85°C	16 CERDIP
MAX234MJE	-55°C to +125°C	16 CERDIP
MAX235CPG	0°C to +70°C	24 Wide Plastic DIP
MAX235EPG	-40°C to +85°C	24 Wide Plastic DIP
MAX235EDG	-40°C to +85°C	24 Ceramic SB
MAX235MDG	-55°C to +125°C	24 Ceramic SB
MAX236CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX236CWG	0°C to +70°C	24 Wide SO
MAX236C/D	0°C to +70°C	Dice*
MAX236ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX236EWG	-40°C to +85°C	24 Wide SO
MAX236ERG	-40°C to +85°C	24 Narrow CERDIP
MAX236MRG	-55°C to +125°C	24 Narrow CERDIP
MAX237CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX237CWG	0°C to +70°C	24 Wide SO
MAX237C/D	0°C to +70°C	Dice*
MAX237ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX237EWG	-40°C to +85°C	24 Wide SO
MAX237ERG	-40°C to +85°C	24 Narrow CERDIP
MAX237MRG	-55°C to +125°C	24 Narrow CERDIP
MAX238CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX238CWG	0°C to +70°C	24 Wide SO
MAX238C/D	0°C to +70°C	Dice*
MAX238ENG	-40°C to +85°C	24 Narrow Plastic DIP

** Contact factory for dice specifications.*

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

Ordering Information (continued)

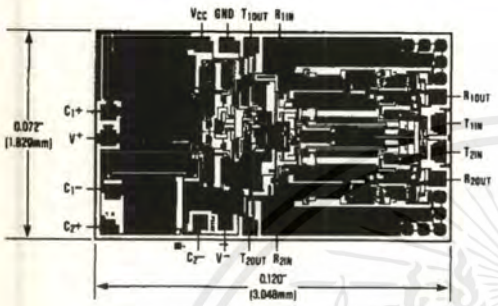
PART	TEMP. RANGE	PIN-PACKAGE
MAX238EWG	-40°C to +85°C	24 Wide SO
MAX238ERG	-40°C to +85°C	24 Narrow CERDIP
MAX238MRG	-55°C to +125°C	24 Narrow CERDIP
MAX239CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX239CWG	0°C to +70°C	24 Wide SO
MAX239C/D	0°C to +70°C	Dice*
MAX239ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX239EWG	-40°C to +85°C	24 Wide SO
MAX239ERG	-40°C to +85°C	24 Narrow CERDIP
MAX239MRG	-55°C to +125°C	24 Narrow CERDIP
MAX240CMH	0°C to +70°C	44 Plastic FP
MAX240C/D	0°C to +70°C	Dice*
MAX241CAI	0°C to +70°C	28 SSOP
MAX241CWI	0°C to +70°C	28 Wide SO
MAX241C/D	0°C to +70°C	Dice*
MAX241EAI	-40°C to +85°C	28 SSOP
MAX241EWI	-40°C to +85°C	28 Wide SO
MAX242CAP	0°C to +70°C	20 SSOP
MAX242CPN	0°C to +70°C	18 Plastic DIP
MAX242CWN	0°C to +70°C	18 Wide SO
MAX242C/D	0°C to +70°C	Dice*
MAX242EPN	-40°C to +85°C	18 Plastic DIP
MAX242EWN	-40°C to +85°C	18 Wide SO
MAX242EJN	-40°C to +85°C	18 CERDIP
MAX242MJN	-55°C to +125°C	18 CERDIP
MAX243CPE	0°C to +70°C	16 Plastic DIP
MAX243CSE	0°C to +70°C	16 Narrow SO
MAX243CWE	0°C to +70°C	16 Wide SO
MAX243C/D	0°C to +70°C	Dice*
MAX243EPE	-40°C to +85°C	16 Plastic DIP
MAX243ESE	-40°C to +85°C	16 Narrow SO
MAX243EWE	-40°C to +85°C	16 Wide SO
MAX243EJE	-40°C to +85°C	16 CERDIP
MAX243MJE	-55°C to +125°C	16 CERDIP
MAX244CQH	0°C to +70°C	44 PLCC
MAX244C/D	0°C to +70°C	Dice*
MAX244EQH	-40°C to +85°C	44 PLCC
MAX245CPL	0°C to +70°C	40 Plastic DIP
MAX245C/D	0°C to +70°C	Dice*
MAX245EPL	-40°C to +85°C	40 Plastic DIP
MAX246CPL	0°C to +70°C	40 Plastic DIP
MAX246C/D	0°C to +70°C	Dice*
MAX246EPL	-40°C to +85°C	40 Plastic DIP
MAX247CPL	0°C to +70°C	40 Plastic DIP
MAX247C/D	0°C to +70°C	Dice*
MAX247EPL	-40°C to +85°C	40 Plastic DIP
MAX248CQH	0°C to +70°C	44 PLCC
MAX248C/D	0°C to +70°C	Dice*
MAX248EQH	-40°C to +85°C	44 PLCC
MAX249CQH	0°C to +70°C	44 PLCC
MAX249EQH	-40°C to +85°C	44 PLCC

** Contact factory for dice specifications.*

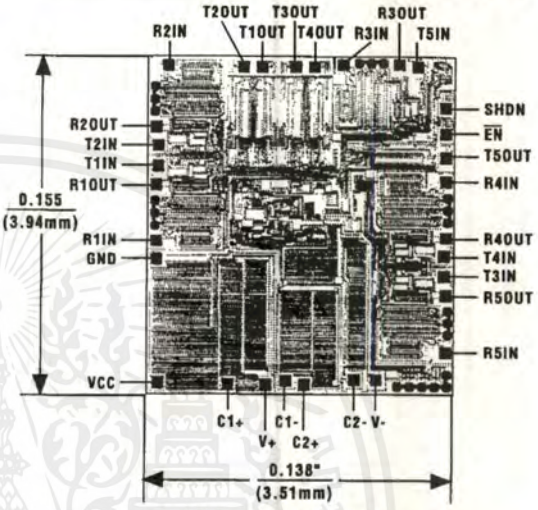
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

Chip Topographies

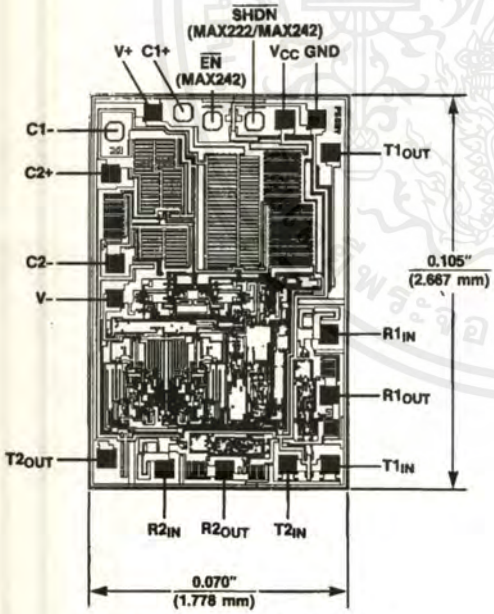


MAX231, MAX232 and MAX233



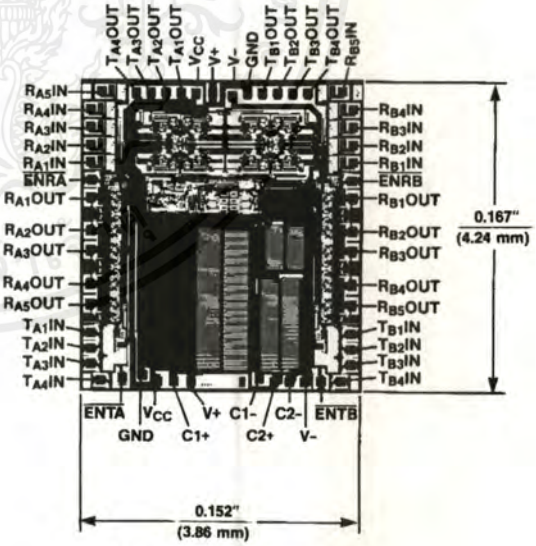
MAX230/MAX234-MAX241

SUTDOWN PIN OF MAX234, MAX237, MAX238, MAX239, MAX240 AND MAX241 ARE INTERNALLY CONNECTED TO GROUND.



MAX220/222/223A/233A/242/243

CONNECT SUBSTRATE TO V+



MAX244/245/246/247/248/249

Maxim cannot assume responsibility for use of any circuitry other than circuitry entirely embodied in a Maxim product. No circuit patent licenses are implied. Maxim reserves the right to change the circuitry and specifications without notice at any time.

© 1996 Maxim Integrated Products Printed USA MAXIM is a registered trademark of Maxim Integrated Products.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

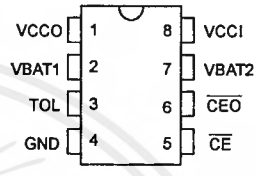
DALLAS SEMICONDUCTOR

DS1210
Nonvolatile Controller Chip

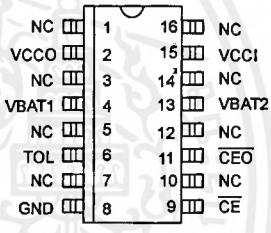
FEATURES

- Converts CMOS RAMs into nonvolatile memories
- Unconditionally write protects when V_{CC} is out of tolerance
- Automatically switches to battery when power fail occurs
- Space saving 8-pin DIP
- Consumes less than 100 nA of battery current
- Tests battery condition on power up
- Provides for redundant batteries
- Optional 5% or 10% power fail detection
- Low forward voltage drop on the V_{CC} switch
- Optional 16-pin SOIC surface mount package
- Optional industrial temperature range of -40°C to +85°C.

PIN ASSIGNMENT



DS1210 8-Pin DIP (300 MIL)



DS1210S 16-Pin SOIC (300 MIL)

PIN DESCRIPTION

V_{CCO}	– RAM Supply
V_{BAT1}	– + Battery 1
TOL	– Power Supply Tolerance
GND	– Ground
$\overline{CE}$	– Chip Enable Input
$\overline{CEO}$	– Chip Enable Output
V_{BAT2}	– + Battery 2
V_{CCI}	– + Supply
NC	– No Connect

DESCRIPTION

The DS1210 Nonvolatile Controller Chip is a CMOS circuit which solves the application problem of converting CMOS RAM into nonvolatile memory. Incoming power is monitored for an out-of-tolerance condition. When such a condition is detected, chip enable is inhibited to accomplish write protection and the battery is switched on to supply the RAM with uninterrupted power. Special circuitry uses a low-leakage CMOS process which affords precise voltage detection at extremely low battery consumption. The 8-pin DIP package keeps PC board real estate requirements to a minimum. By combining the DS1210 Nonvolatile Controller Chip with a CMOS memory and batteries, nonvolatile RAM operation can be achieved.

©Copyright 1995 by Dallas Semiconductor Corporation.
All Rights Reserved. For important information regarding patents and other intellectual property rights, please refer to Dallas Semiconductor databooks.

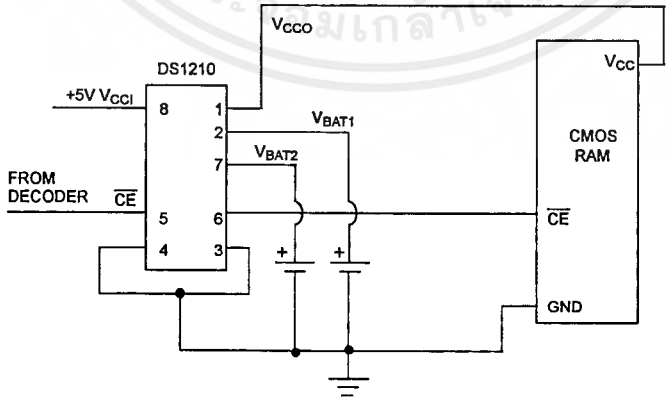
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

OPERATION

The DS1210 nonvolatile controller performs five circuit functions required to battery back up a RAM. First, a switch is provided to direct power from the battery or the incoming supply (V_{CCI}) depending on which is greater. This switch has a voltage drop of less than 0.3V. The second function which the nonvolatile controller provides is power fail detection. The DS1210 constantly monitors the incoming supply. When the supply goes out of tolerance a precision comparator detects power fail and inhibits chip enable ($\overline{CEO}$). The third function of write protection is accomplished by holding the $\overline{CEO}$ output signal to within 0.2 volts of the V_{CCI} or battery supply. If $\overline{CE}$ input is low at the time power fail detection occurs, the $\overline{CEO}$ output is kept in its present state until $\overline{CE}$ is returned high. The delay of write protection until the current memory cycle is completed prevents the corruption of data. Power fail detection occurs in the range of 4.75 volts to 4.5 volts with the tolerance Pin 3 grounded . If Pin 3 in connected to V_{CCO}, then power fail detection occurs in the range of 4.5 volts to 4.25 volts. During nominal supply conditions $\overline{CEO}$ will follow $\overline{CE}$ with a maximum propagation delay of 20ns. The fourth function the DS1210 performs is a battery status warning so that potential data loss is avoided. Each time that the circuit is powered up the battery voltage is checked with a precision comparator. If the battery voltage is less than 2.0 volts, the second memory cycle is inhibited. Battery status can, therefore, be determined by performing a read cycle after power-up to any location in memory, verifying that memory location content. A subsequent write cycle can then be executed to the same memory location altering the data. If the next read cycle fails to verify the written data, then the batteries are less than 2.0V and data is in danger of being corrupted. The fifth function of the nonvolatile controller provides for battery redundancy. In many applications, data integrity is paramount. In these applications it is often desirable to use two batteries to ensure reliability. The DS1210 controller provides an internal isolation switch which allows the connection of two batteries. During battery backup operation the battery with the highest voltage is selected for use. If one battery should fail, the other will take over the load. The switch to a redundant battery is transparent to circuit operation and to the user. A battery status warning will occur when the battery in use falls below 2.0 volts. A grounded V_{BAT2} pin will not activate a battery fail warning. In applications where battery redundancy is not required, a single battery should be connected to the BAT1 pin. The BAT2 battery pin must be grounded. The nonvolatile controller contains circuitry to turn off the battery back-up. This is to maintain the battery(s) at its highest capacity until the equipment is powered up and valid data is written to the SRAM. While in the freshness seal mode the $\overline{CEO}$ and V_{CCO} will be forced to V_{OL}. When the batteries are first attached to one or both of the V_{BAT} pins, V_{CCO} will not provide battery back-up until V_{CCI} exceeds V_{CCTP} as set by the T_{OL} pin, and then falls below V_{BAT}.

Figure 1 shows a typical application incorporating the DS1210 in a microprocessor-based system. Section A shows the connections necessary to write protect the RAM when V_{CC} is less than 4.75 volts and to back up the supply with batteries. Section B shows the use of the DS1210 to halt the processor when V_{CC} is less than 4.75 volts and to delay its restart on power- up to prevent spurious writes.

SECTION A – BATTERY BACKUP Figure 1



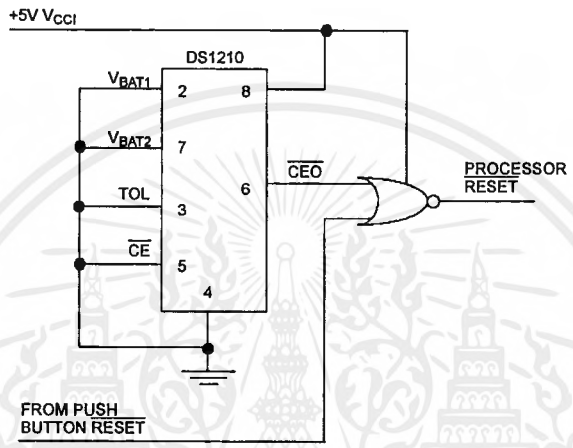
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

BATTERY BACKUP CURRENT DRAIN EXAMPLE

CONSUMPTION

DS1210 I_{BAT}	100 nA
RAM I_{CCO2}	10 µA
Total Drain	10.1 µA

SECTION B - PROCESSOR RESET



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ABSOLUTE MAXIMUM RATINGS*

Voltage on any Pin Relative to Ground	-0.3V to +7.0V
Operating Temperature	0°C to 70°C
Storage Temperature	-55°C to +125°C
Soldering Temperature	260°C for 10 seconds

* This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.

RECOMMENDED DC OPERATING CONDITIONS

(0°C to 70°C)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Pin 3 = GND Supply Voltage	V_{CCI}	4.75	5.0	5.5	V	1
Pin 3 = V_{CCO} Supply Voltage	V_{CCI}	4.5	5.0	5.5	V	1
Logic 1 Input	V_{IH}	2.2		V_{CC}+0.3	V	1
Logic 0 Input	V_{IL}	-0.3		+0.8	V	1
Battery Input	V_{BAT1}, V_{BAT2}	2.0		4.0	V	1,2

DC ELECTRICAL CHARACTERISTICS

(0°C to 70°C; V_{CCI} = 4.75V to 5.5V, PIN 3 = GND)
(V_{CCI} = 4.5 to 5.5V, PIN 3 = V_{CCO})

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Supply Current	I_{CCI}			5	mA	3
Supply Voltage	V_{CCO}	V_{CC}-0.2			V	1
Supply Current	I_{CCO1}			80	mA	4
Input Leakage	I_{IL}	-1.0		+1.0	µA	
Output Leakage	I_{LO}	-1.0		+1.0	µA	
$\overline{CEO}$ Output @2.4V	I_{OH}	-1.0			mA	5
$\overline{CEO}$ Output @0.4V	I_{OL}			4.0	mA	5
V_{CC} Trip Point (TOL=GND)	V_{CCTP}	4.50	4.62	4.74	V	1
V_{CC} Trip Point (TOL=V_{CCO})	V_{CCTP}	4.25	4.37	4.49	V	1

(0°C to 70°C; V_{CCI} = < V_{BAT})

$\overline{CEO}$ Output	V_{OHL}	V_{BAT}-0.2			V	7
V_{BAT1} or V_{BAT2} Battery Current	I_{BAT}			100	nA	2,3
Battery Backup Current @ V_{cco} = V_{BAT} - 0.3V	I_{CCO2}			50	µA	6,7

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

CAPACITANCE

(T_A = 25°C)

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
Input Capacitance	C_{IN}			5	pF	
Output Capacitance	C_{OUT}			7	pF	

AC ELECTRICAL CHARACTERISTICS

(0°C to 70°C; V_{CCI} = 4.75V to 5.5V, PIN 3 = GND)
(V_{CCI} = 4.5 to 5.5V, PIN3 = V_{CCO})

PARAMETER	SYMBOL	MIN	TYP	MAX	UNITS	NOTES
$\overline{CE}$ Propagation Delay	t_{PD}	5	10	20	ns	5
$\overline{CE}$ High to Power Fail	t_{PF}			0	ns	

(0°C to 70°C; V_{CCI} < 4.75V, PIN 3 = GND; V_{CCI} < 4.5, PIN 3 = V_{CCO})

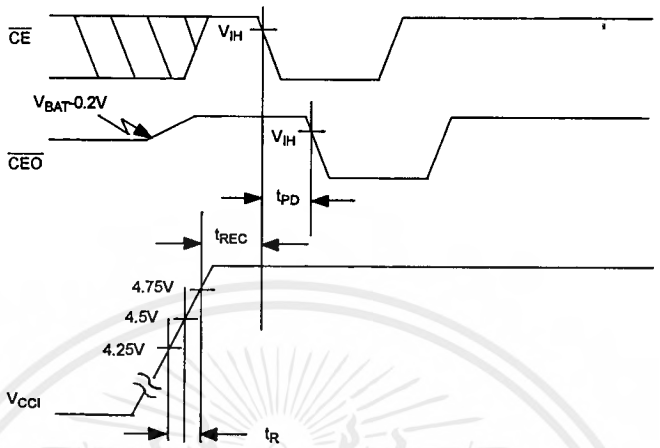
Recovery at Power Up	t_{REC}	2	80	125	ms	
V_{CC} Slew Rate Power Down	t_F	300			µs	
V_{CC} Slew Rate Power Down	t_{FB}	10			µs	
V_{CC} Slew Rate Power Up	t_R	0			µs	
$\overline{CE}$ Pulse Width	t_{CE}			1.5	µs	8

NOTES:

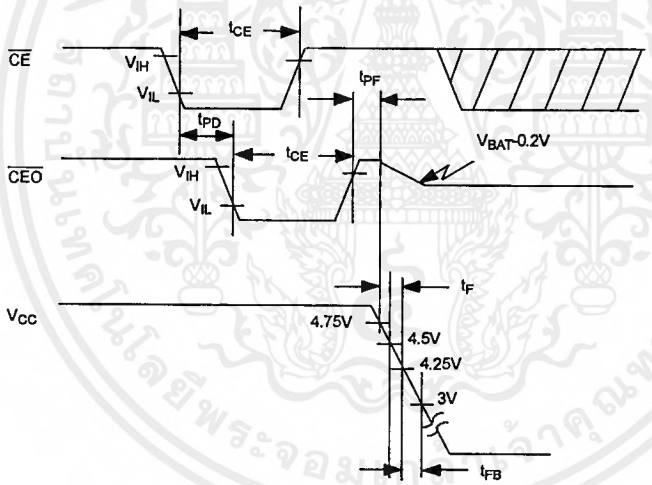
1. All voltages are referenced to ground.
2. Only one battery input is required. Unused battery inputs must be grounded.
3. Measured with V_{CCO} and $\overline{CEO}$ open.
4. I_{CCO1} is the maximum average load which the DS1210 can supply to the memories.
5. Measured with a load as shown in Figure 2.
6. I_{CCO2} is the maximum average load current which the DS1210 can supply to the memories in the battery backup mode.
7. t_{CE} max. must be met to ensure data integrity on power loss.
8. $\overline{CEO}$ can only sustain leakage current in the battery backup mode.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TIMING DIAGRAM: POWER UP

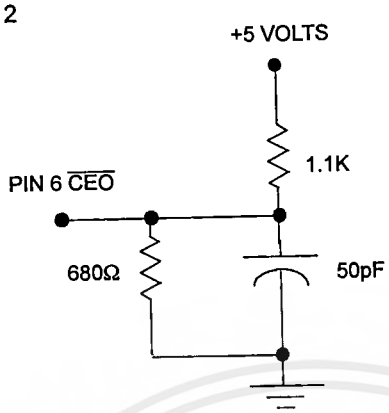


TIMING DIAGRAM: POWER DOWN



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

OUTPUT LOAD Figure 2



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

National Semiconductor

December 1994

LM386 Low Voltage Audio Power Amplifier

General Description

The LM386 is a power amplifier designed for use in low voltage consumer applications. The gain is internally set to 20 to keep external part count low, but the addition of an external resistor and capacitor between pins 1 and 8 will increase the gain to any value up to 200.

The inputs are ground referenced while the output is automatically biased to one half the supply voltage. The quiescent power drain is only 24 milliwatts when operating from a 6 volt supply, making the LM386 ideal for battery operation.

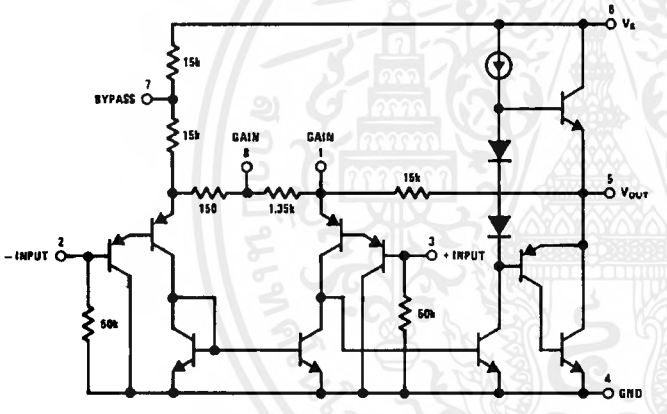
Features

- Battery operation
- Minimum external parts
- Wide supply voltage range — 4V–12V or 5V–18V
- Low quiescent current drain — 4 mA
- Voltage gains from 20 to 200
- Ground referenced input
- Self-centering output quiescent voltage
- Low distortion
- Eight pin dual-in-line package

Applications

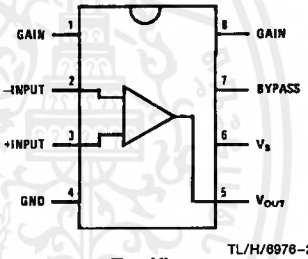
- AM-FM radio amplifiers
- Portable tape player amplifiers
- Intercoms
- TV sound systems
- Line drivers
- Ultrasonic drivers
- Small servo drivers
- Power converters

Equivalent Schematic and Connection Diagrams



TL/H/6976–1

Dual-In-Line and Small Outline Packages



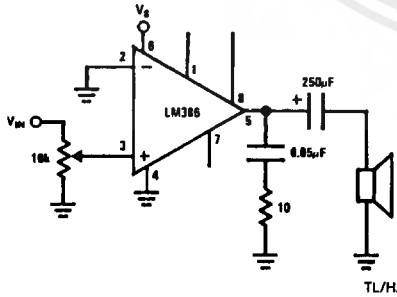
TL/H/6976–2

Top View

Order Number LM386M-1, LM386N-1, LM386N-3 or LM386N-4
See NS Package Number M08A or N08E

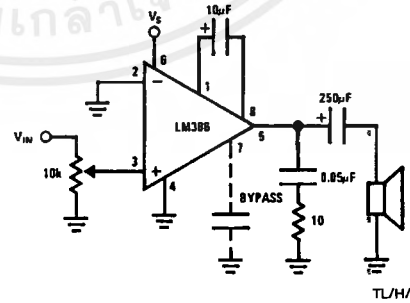
Typical Applications

Amplifier with Gain = 20
Minimum Parts



TL/H/6976–3

Amplifier with Gain = 200



TL/H/6976–4

©1995 National Semiconductor Corporation TL/H/6976

RRD-B30M75/Printed in U. S. A.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage (LM386N-1, -3, LM386M-1)	15V
Supply Voltage (LM386N-4)	22V
Package Dissipation (Note 1) (LM386N)	1.25W
(LM386M)	0.73W
Input Voltage	±0.4V
Storage Temperature	−65°C to +150°C
Operating Temperature	0°C to +70°C
Junction Temperature	+150°C

Soldering Information

Dual-In-Line Package	
Soldering (10 sec)	+260°C
Small Outline Package	
Vapor Phase (60 sec)	+215°C
Infrared (15 sec)	+220°C

See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" for other methods of soldering surface mount devices.

Thermal Resistance

θ_{JC} (DIP)	37°C/W
θ_{JA} (DIP)	107°C/W
θ_{JC} (SO Package)	35°C/W
θ_{JA} (SO Package)	172°C/W

Electrical Characteristics $T_A = 25°C$

Parameter	Conditions	Min	Typ	Max	Units
Operating Supply Voltage (V_S)					
LM386N-1, -3, LM386M-1		4		12	V
LM386N-4		5		18	V
Quiescent Current (I_Q)	$V_S = 6V$, $V_{IN} = 0$		4	8	mA
Output Power (P_{OUT})					
LM386N-1, LM386M-1	$V_S = 6V$, $R_L = 8\Omega$, THD = 10%	250	325		mW
LM386N-3	$V_S = 9V$, $R_L = 8\Omega$, THD = 10%	500	700		mW
LM386N-4	$V_S = 16V$, $R_L = 32\Omega$, THD = 10%	700	1000		mW
Voltage Gain (A_V)	$V_S = 6V$, f = 1 kHz		26		dB
	10 μF from Pin 1 to 8		46		dB
Bandwidth (BW)	$V_S = 6V$, Pins 1 and 8 Open		300		kHz
Total Harmonic Distortion (THD)	$V_S = 6V$, $R_L = 8\Omega$, $P_{OUT} = 125$ mW f = 1 kHz, Pins 1 and 8 Open		0.2		%
Power Supply Rejection Ratio (PSRR)	$V_S = 6V$, f = 1 kHz, $C_{BYPASS} = 10\ \mu F$ Pins 1 and 8 Open, Referred to Output		50		dB
Input Resistance (R_{IN})			50		kΩ
Input Bias Current (I_{BIAS})	$V_S = 6V$, Pins 2 and 3 Open		250		nA

Note 1: For operation in ambient temperatures above 25°C, the device must be derated based on a 150°C maximum junction temperature and 1) a thermal resistance of 80°C/W junction to ambient for the dual-in-line package and 2) a thermal resistance of 170°C/W for the small outline package.

Application Hints

GAIN CONTROL

To make the LM386 a more versatile amplifier, two pins (1 and 8) are provided for gain control. With pins 1 and 8 open the 1.35 kΩ resistor sets the gain at 20 (26 dB). If a capacitor is put from pin 1 to 8, bypassing the 1.35 kΩ resistor, the gain will go up to 200 (46 dB). If a resistor is placed in series with the capacitor, the gain can be set to any value from 20 to 200. Gain control can also be done by capacitively coupling a resistor (or FET) from pin 1 to ground.

Additional external components can be placed in parallel with the internal feedback resistors to tailor the gain and frequency response for individual applications. For example, we can compensate poor speaker bass response by frequency shaping the feedback path. This is done with a series RC from pin 1 to 5 (paralleling the internal 15 kΩ resistor). For 6 dB effective bass boost: R ≅ 15 kΩ, the lowest value for good stable operation is R = 10 kΩ if pin 8 is open. If pins 1 and 8 are bypassed then R as low as 2 kΩ can be used. This restriction is because the amplifier is only compensated for closed-loop gains greater than 9.

INPUT BIASING

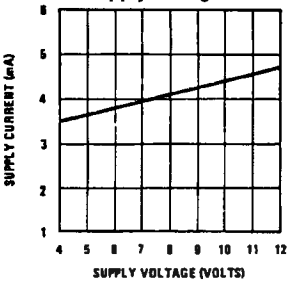
The schematic shows that both inputs are biased to ground with a 50 kΩ resistor. The base current of the input transistors is about 250 nA, so the inputs are at about 12.5 mV when left open. If the dc source resistance driving the LM386 is higher than 250 kΩ it will contribute very little additional offset (about 2.5 mV at the input, 50 mV at the output). If the dc source resistance is less than 10 kΩ, then shorting the unused input to ground will keep the offset low (about 2.5 mV at the input, 50 mV at the output). For dc source resistances between these values we can eliminate excess offset by putting a resistor from the unused input to ground, equal in value to the dc source resistance. Of course all offset problems are eliminated if the input is capacitively coupled.

When using the LM386 with higher gains (bypassing the 1.35 kΩ resistor between pins 1 and 8) it is necessary to bypass the unused input, preventing degradation of gain and possible instabilities. This is done with a 0.1 μF capacitor or a short to ground depending on the dc source resistance on the driven input.

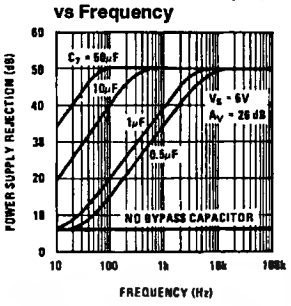
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Typical Performance Characteristics

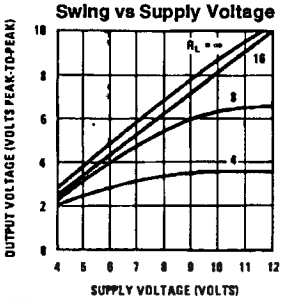
Quiescent Supply Current vs Supply Voltage



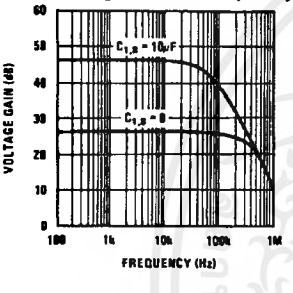
Power Supply Rejection Ratio (Referred to the Output) vs Frequency



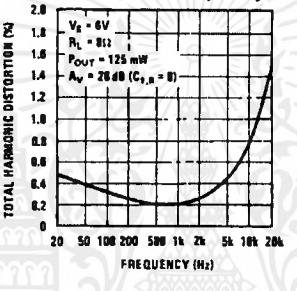
Peak-to-Peak Output Voltage Swing vs Supply Voltage



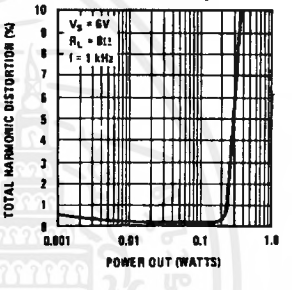
Voltage Gain vs Frequency



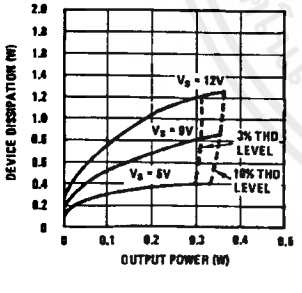
Distortion vs Frequency



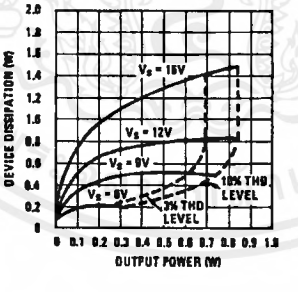
Distortion vs Output Power



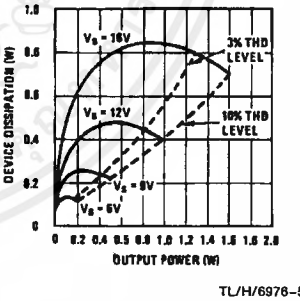
Device Dissipation vs Output Power—4Ω Load



Device Dissipation vs Output Power—8Ω Load



Device Dissipation vs Output Power—16Ω Load

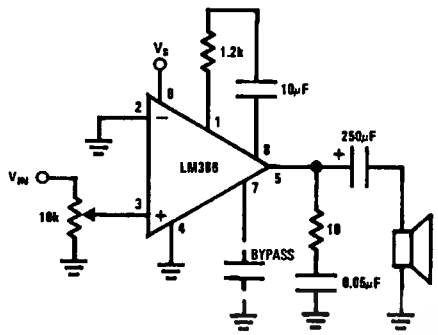


TL/H/6976–5

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

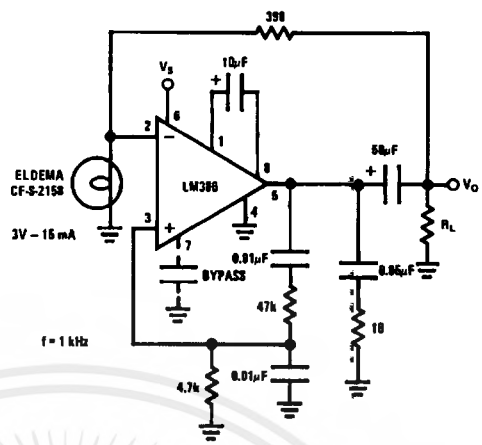
Typical Applications (Continued)

Amplifier with Gain = 50



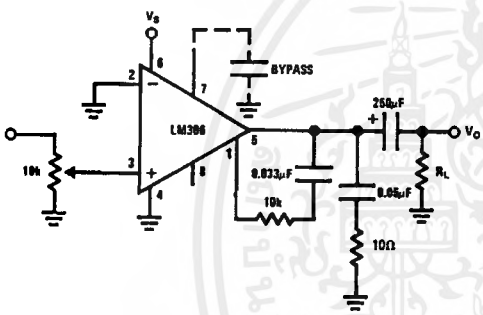
TL/H/6976–6

Low Distortion Power Wienbridge Oscillator



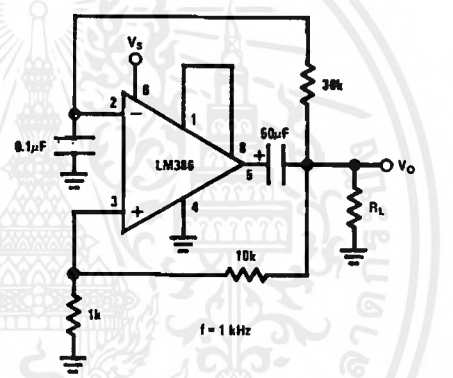
TL/H/6976–7

Amplifier with Bass Boost



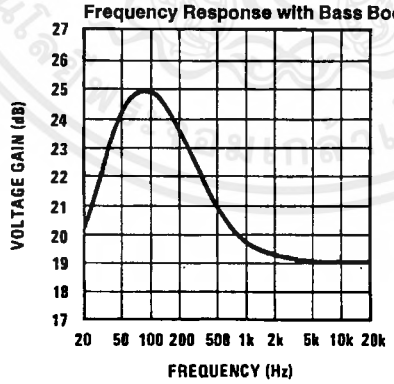
TL/H/6976–8

Square Wave Oscillator



TL/H/6976–9

Frequency Response with Bass Boost

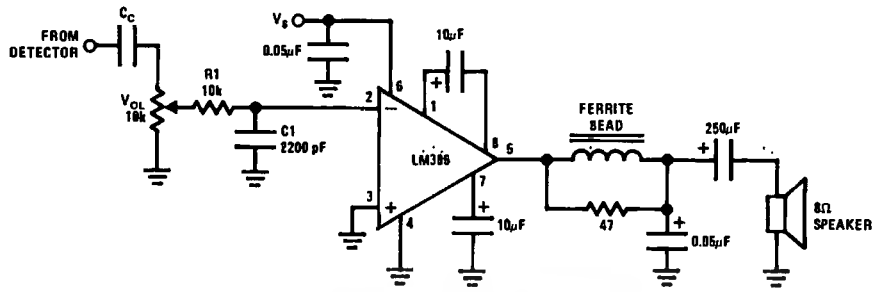


TL/H/6976–10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Typical Applications (Continued)

AM Radio Power Amplifier



TL/H/6976–11

Note 1: Twist supply lead and supply ground very tightly.

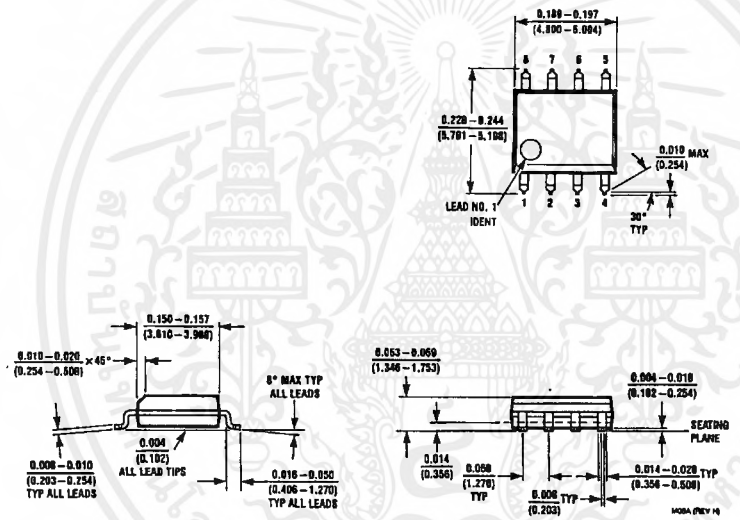
Note 2: Twist speaker lead and ground very tightly.

Note 3: Ferrite bead is Ferroxcube K5-001-001/3B with 3 turns of wire.

Note 4: R1C1 band limits input signals.

Note 5: All components must be spaced very close to IC.

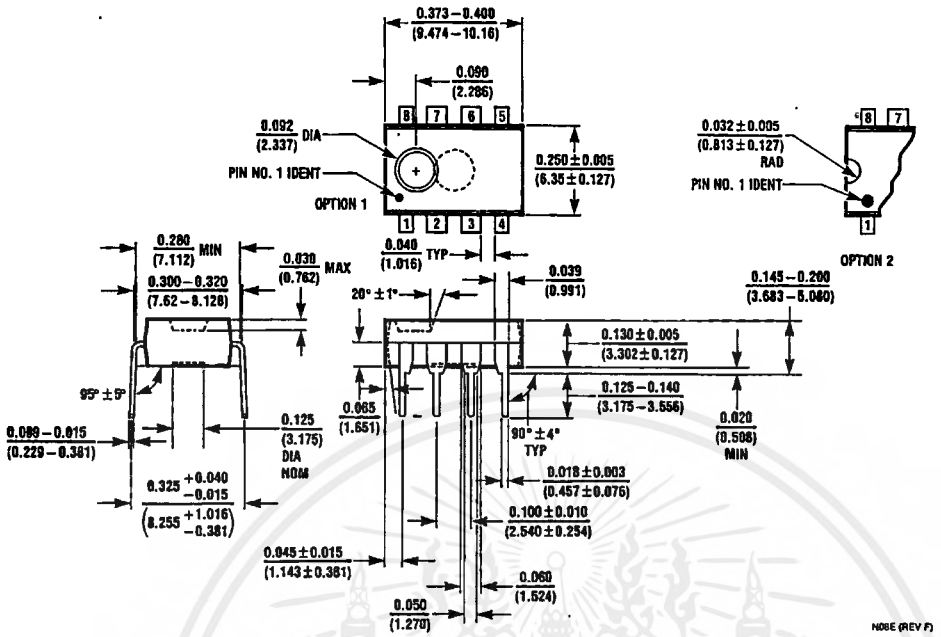
Physical Dimensions inches (millimeters)



SO Package (M)
Order Number LM386M-1
NS Package Number M08A

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Physical Dimensions inches (millimeters) (Continued)



Dual-In-Line Package (N)
Order Number LM386N-1, LM386N-3 or LM386N-4
NS Package Number N08E

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
Tel: 1(800) 272-9959
TWX: (910) 339-9240

National Semiconductor GmbH
Livry-Gargan-Str. 10
D-82256 Fürstenfeldbruck
Germany
Tel: (81-41) 35-0
Telex: 527649
Fax: (81-41) 35-1

National Semiconductor Japan Ltd.
Sumitomo Chemical
Engineering Center
Bldg. 7F
1-7-1, Nakase, Mihama-Ku
Chiba-City,
Ciba Prefecture 261
Tel: (043) 299-2300
Fax: (043) 299-2500

National Semiconductor Hong Kong Ltd.
13th Floor, Straight Block,
Ocean Centre, 5 Canton Rd.
Tsimshatsui, Kowloon
Hong Kong
Tel: (852) 2737-1600
Fax: (852) 2736-9960

National Semiconductores Do Brazil Ltda.
Rua Deputado Lacorda Franco
120-3A
Sao Paulo-SP
Brazil 05418-000
Tel: (55-11) 212-5066
Telex: 391-1131931 NSBR BR
Fax: (55-11) 212-1181

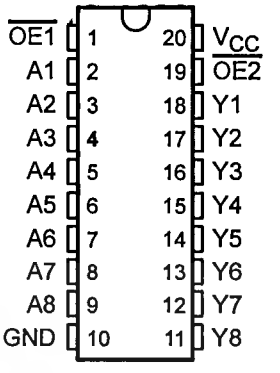
National Semiconductor (Australia) Pty, Ltd.
Building 16
Business Park Drive
Monash Business Park
Nottinghill, Melbourne
Victoria 3168 Australia
Tel: (3) 558-9999
Fax: (3) 558-9998

National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and National reserves the right at any time without notice to change said circuitry and specifications.

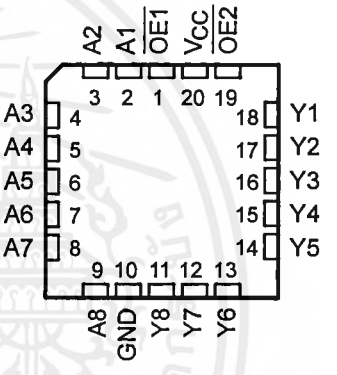
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- **High-Current 3-State Outputs Drive Bus Lines Directly or up to 15 LSTTL Loads**
- **Data Flow-Through Pinout (All Inputs on Opposite Side From Outputs)**
- **Package Options Include Plastic Small-Outline (DW), Thin Shrink Small-Outline (PW), and Ceramic Flat (W) Packages, Ceramic Chip Carriers (FK), and Standard Plastic (N) and Ceramic (J) 300-mil DIPs**

SN54HC541 . . . J OR W PACKAGE
SN74HC541 . . . DW, N, OR PW PACKAGE
(TOP VIEW)



SN54HC541 . . . FK PACKAGE
(TOP VIEW)



description

These octal buffers and line drivers feature the performance of the popular 'HC240 series and offer a pinout with inputs and outputs on opposite sides of the package. This arrangement greatly enhances printed-circuit-board layout.

The 3-state control gate is a 2-input NOR. If either output-enable ($\overline{OE1}$ or $\overline{OE2}$) input is high, all eight outputs are in the high-impedance state. The 'HC541 provide true data at the outputs.

The SN54HC541 is characterized for operation over the full military temperature range of –55°C to 125°C. The SN74HC541 is characterized for operation from –40°C to 85°C.

FUNCTION TABLE
(each buffer/driver)

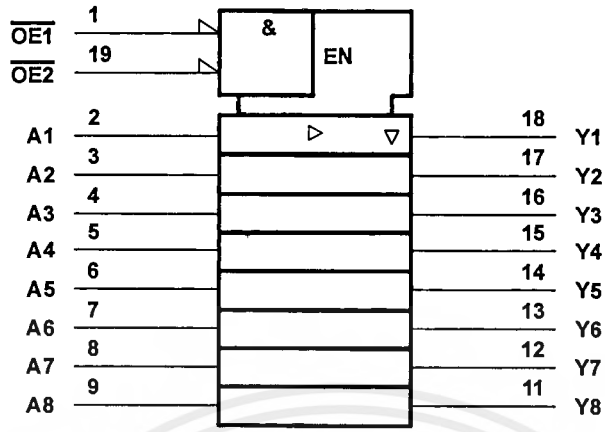
INPUTS			OUTPUT
$\overline{OE1}$	$\overline{OE2}$	A	Y
L	L	L	L
L	L	H	H
H	X	X	Z
X	H	X	Z

Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

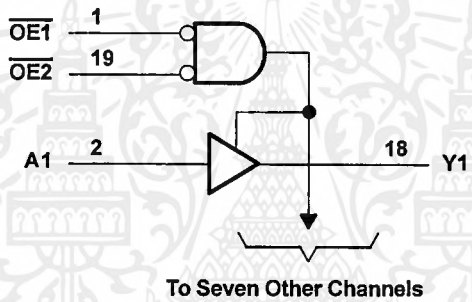
Copyright © 1996, Texas Instruments Incorporated

logic symbol†



† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

logic diagram (positive logic)



absolute maximum ratings over operating free-air temperature range‡

Supply voltage range, V_{CC}	–0.5 V to 7 V
Input clamp current, I_{IK} ($V_I < 0$ or $V_I > V_{CC}$) (see Note 1)	±20 mA
Output clamp current, I_{OK} ($V_O < 0$ or $V_O > V_{CC}$) (see Note 1)	±20 mA
Continuous output current, I_O (V_O = 0 to V_{CC})	±35 mA
Continuous current through V_{CC} or GND	±70 mA
Maximum power dissipation at T_A = 55°C (in still air) (see Note 2): DW package	1.6 W
N package	1.3 W
PW package	0.7 W
Storage temperature range, T_{stg}	–65°C to 150°C

‡ Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTES: 1. The input and output voltage ratings may be exceeded if the input and output current ratings are observed.
2. The maximum package power dissipation is calculated using a junction temperature of 150°C and a board trace length of 750 mils, except for the N package, which has a trace length of zero.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

recommended operating conditions

			SN54HC541			SN74HC541			UNIT
			MIN	NOM	MAX	MIN	NOM	MAX	
V_{CC}	Supply voltage		2	5	6	2	5	6	V
V_{IH}	High-level input voltage	V_{CC} = 2 V	1.5			1.5			V
		V_{CC} = 4.5 V	3.15			3.15			
		V_{CC} = 6 V	4.2			4.2			
V_{IL}	Low-level input voltage	V_{CC} = 2 V	0		0.5	0		0.5	V
		V_{CC} = 4.5 V	0		1.35	0		1.35	
		V_{CC} = 6 V	0		1.8	0		1.8	
V_I	Input voltage		0		V_{CC}	0		V_{CC}	V
V_O	Output voltage		0		V_{CC}	0		V_{CC}	V
t_t	Input transition (rise and fall) time	V_{CC} = 2 V	0		1000	0		1000	ns
		V_{CC} = 4.5 V	0		500	0		500	
		V_{CC} = 6 V	0		400	0		400	
T_A	Operating free-air temperature		–55		125	–40		85	°C

electrical characteristics over recommended operating free-air temperature range (unless otherwise noted)

PARAMETER	TEST CONDITIONS		V_{CC}	T_A = 25°C			SN54HC541		SN74HC541		UNIT
				MIN	TYP	MAX	MIN	MAX	MIN	MAX	
V_{OH}	V_I = V_{IH} or V_{IL}	I_{OH} = –20 μA	2 V	1.9	1.998		1.9		1.9		V
			4.5 V	4.4	4.499		4.4		4.4		
			6 V	5.9	5.999		5.9		5.9		
		I_{OH} = –6 mA	4.5 V	3.98	4.3		3.7		3.84		
		I_{OH} = –7.8 mA	6 V	5.48	5.8		5.2		5.34		
V_{OL}	V_I = V_{IH} or V_{IL}	I_{OL} = 20 μA	2 V		0.002	0.1		0.1		0.1	V
			4.5 V		0.001	0.1		0.1		0.1	
			6 V		0.001	0.1		0.1		0.1	
		I_{OL} = 6 mA	4.5 V		0.17	0.26		0.4		0.33	
		I_{OL} = 7.8 mA	6 V		0.15	0.26		0.4		0.33	
I_I	V_I = V_{CC} or 0		6 V			±0.1		±1000		±1000	nA
I_{OZ}	V_O = V_{CC} or 0		6 V			±0.01		±10		±5	μA
I_{CC}	V_I = V_{CC} or 0,	I_O = 0	6 V			8		160		80	μA
C_i			2 V to 6 V		3	10		10		10	pF

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

switching characteristics over recommended operating free-air temperature range, C_L = 50 pF (unless otherwise noted) (see Figure 1)

PARAMETER	FROM (INPUT)	TO (OUTPUT)	V_{CC}	T_A = 25°C			SN54HC541		SN74HC541		UNIT
				MIN	TYP	MAX	MIN	MAX	MIN	MAX	
t_{pd}	A	Y	2 V		40	115		171		144	ns
			4.5 V		12	23		34		29	
			6 V		10	20		29		25	
t_{en}	$\overline{OE}$	Y	2 V		80	150		224		188	ns
			4.5 V		17	30		45		38	
			6 V		15	26		38		32	
t_{dis}	$\overline{OE}$	Y	2 V		40	150		224		188	ns
			4.5 V		18	30		45		38	
			6 V		17	26		38		32	
t_t		Y	2 V		28	60		90		75	ns
			4.5 V		8	12		18		15	
			6 V		6	10		15		13	

switching characteristics over recommended operating free-air temperature range, C_L = 150 pF (unless otherwise noted) (see Figure 1)

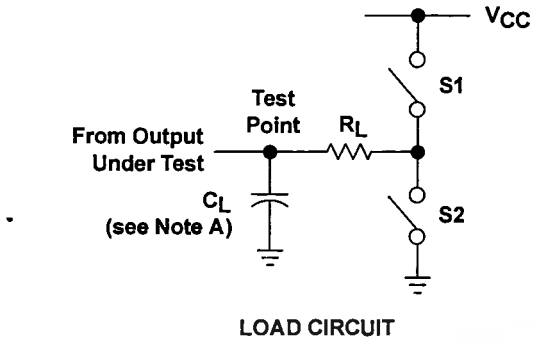
PARAMETER	FROM (INPUT)	TO (OUTPUT)	V_{CC}	T_A = 25°C			SN54HC541		SN74HC541		UNIT
				MIN	TYP	MAX	MIN	MAX	MIN	MAX	
t_{pd}	A	Y	2 V		65	165		246		206	ns
			4.5 V		16	33		49		41	
			6 V		14	28		42		35	
t_{en}	$\overline{OE}$	Y	2 V		100	200		298		250	ns
			4.5 V		20	40		60		50	
			6 V		17	34		51		43	
t_t		Y	2 V		45	210		315		265	ns
			4.5 V		17	42		63		53	
			6 V		13	36		53		45	

operating characteristics, T_A = 25°C

PARAMETER		TEST CONDITIONS	TYP	UNIT
C_{pd}	Power dissipation capacitance per buffer/driver	No load	35	pF

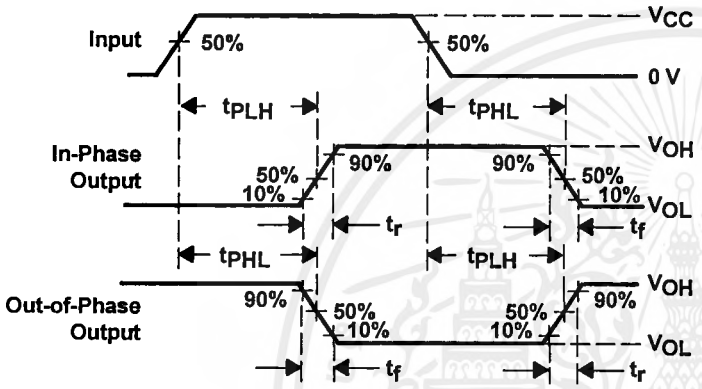
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

PARAMETER MEASUREMENT INFORMATION



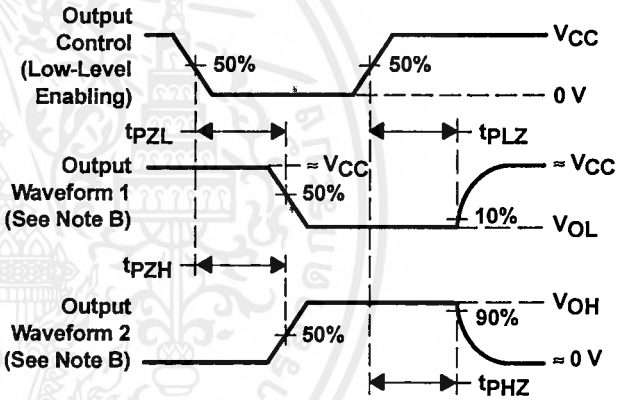
LOAD CIRCUIT

PARAMETER		R_L	C_L	S1	S2
t_{en}	t_{PZH}	1 kΩ	50 pF or 150 pF	Open	Closed
	t_{PZL}			Closed	Open
t_{dis}	t_{PHZ}	1 kΩ	50 pF	Open	Closed
	t_{PLZ}			Closed	Open
t_{pd} or t_t		—	50 pF or 150 pF	Open	Open



VOLTAGE WAVEFORMS
PROPAGATION DELAY AND OUTPUT TRANSITION TIMES

VOLTAGE WAVEFORM
INPUT RISE AND FALL TIMES



VOLTAGE WAVEFORMS
ENABLE AND DISABLE TIMES FOR 3-STATE OUTPUTS

NOTES: A. C_L includes probe and test-fixture capacitance.
B. Waveform 1 is for an output with internal conditions such that the output is low except when disabled by the output control. Waveform 2 is for an output with internal conditions such that the output is high except when disabled by the output control.
C. Phase relationships between waveforms were chosen arbitrarily. All input pulses are supplied by generators having the following characteristics: PRR ≤ 1 MHz, Z_O = 50 Ω, t_r = 6 ns, t_f = 6 ns.
D. The outputs are measured one at a time with one input transition per measurement.
E. t_{PLZ} and t_{PHZ} are the same as t_{dis}.
F. t_{PZL} and t_{PZH} are the same as t_{en}.
G. t_{PLH} and t_{PHL} are the same as t_{pd}.

Figure 1. Load Circuit and Voltage Waveforms

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

This Data Sheet is Applicable to All TMS27C256s and TMS27PC256s Symbolized With Code "B" as Described on Page 157.

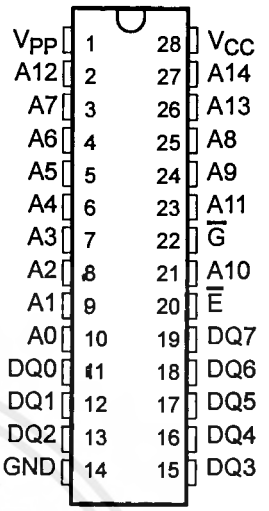
- **Organization . . . 32K × 8**
- **Single 5-V Power Supply**
- **Pin Compatible With Existing 256K MOS ROMs, PROMs, and EPROMs**
- **All Inputs/Outputs Fully TTL Compatible**
- **Max Access/Min Cycle Time**

  V_{CC} ± 10%

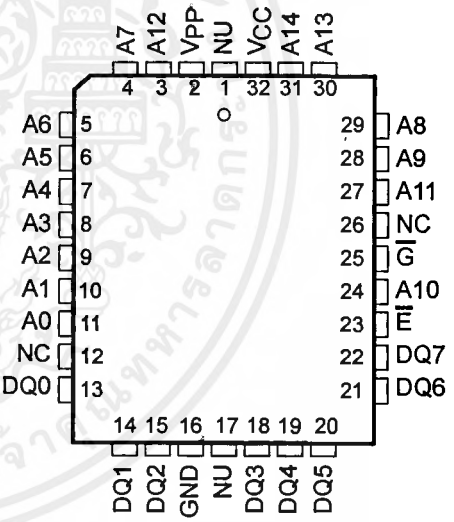
'27C/PC256-10	100 ns
'27C/PC256-12	120 ns
'27C/PC256-15	150 ns
'27C/PC256-17	170 ns
'27C/PC256-20	200 ns
'27C/PC256-25	250 ns

- **Power Saving CMOS Technology**
- **Very High-Speed SNAP! Pulse Programming**
- **3-State Output Buffers**
- **400-mV Minimum DC Noise Immunity With Standard TTL Loads**
- **Latchup Immunity of 250 mA on All Input and Output Lines**
- **Low Power Dissipation (V_{CC} = 5.5 V)**
  - **Active . . . 165 mW Worst Case**
  - **Standby . . . 1.4 mW Worst Case (CMOS Input Levels)**
- **PEP4 Version Available With 168-Hour Burn-In, and Choices of Operating Temperature Ranges**
- **256K EPROM Available With MIL-STD-883C Class B High Reliability Processing (SMJ27C256)**

J AND N PACKAGES
(TOP VIEW)



FM PACKAGE
(TOP VIEW)



description

The TMS27C256 series are 262144-bit, ultraviolet-light erasable, electrically programmable read-only memories.

The TMS27PC256 series are 262144-bit, one-time electrically programmable read-only memories.

PIN NOMENCLATURE	
A0–A14	Address Inputs
DQ0–DQ7	Inputs (programming)/Outputs
$\overline{E}$	Chip Enable/Powerdown
$\overline{G}$	Output Enable
GND	Ground
NC	No Internal Connection
NU	Make No External Connection
V_{CC}	5-V Power Supply
V_{PP}	13-V Power Supply

PRODUCTION DATA Information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

Copyright © 1995, Texas Instruments Incorporated

escription (continued)

These devices are fabricated using power-saving CMOS technology for high speed and simple interface with MOS and bipolar circuits. All inputs (including program data inputs) can be driven by Series 74 TTL circuits without the use of external pull-up resistors. Each output can drive one Series 74 TTL circuit without external resistors.

The data outputs are three-state for connecting multiple devices to a common bus. The TMS27C256 and the TMS27PC256 are pin compatible with 28-pin 256K MOS ROMs, PROMs, and EPROMs.

The TMS27C256 EPROM is offered in a dual-in-line ceramic package (J suffix) designed for insertion in mounting-hole rows on 15,2-mm (600-mil) centers. The TMS27PC256 OTP PROM is offered in a dual-in-line plastic package (N suffix) designed for insertion in mounting-hole rows on 15,2-mm (600-mil) centers. The TMS27PC256 OTP PROM is also supplied in a 32-lead plastic leaded chip-carrier package using 1,25-mm (50-mil) lead spacing (FM suffix).

The TMS27C256 and TMS27PC256 are offered with two choices of temperature ranges of 0°C to 70°C (JL, NL, and FML suffixes) and – 40°C to 85°C (JE, NE, and FME suffixes). The TMS27C256 and the TMS27PC256 are also offered with 168-hour burn-in on both temperature ranges (JL4, FML4, JE4, and FME4 suffixes); see table below.

All package styles conform to JEDEC standards.

EPROM AND OTP PROM	SUFFIX FOR OPERATING TEMPERATURE RANGES WITHOUT PEP4 BURN-IN		SUFFIX FOR PEP4 168-HR. BURN-IN VS TEMPERATURE RANGES	
	0°C TO 70°C	– 40°C TO 85°C	0°C TO 70°C	– 40°C TO 85°C
TMS27C256-XXX	JL	JE	JL4	JE4
TMS27PC256-XXX	NL	NE	NL4	NE4
TMS27PC256-XXX	FML	FME	FML4	FME4

These EPROMs and OTP PROMs operate from a single 5-V supply (in the read mode), thus are ideal for use in microprocessor-based systems. One other 13-V supply is needed for programming . All programming signals are TTL level. These devices are programmable by the SNAP! Pulse programming algorithm. The SNAP! Pulse programming algorithm uses a V_{PP} of 13 V and a V_{CC} of 6.5 V for a nominal programming time of four seconds. For programming outside the system, existing EPROM programmers can be used. Locations can be programmed singly, in blocks, or at random.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

operation

The seven modes of operation are listed in the following table. The read mode requires a single 5-V supply. All inputs are TTL level except for V_{PP} during programming (13 V for SNAP! Pulse), and 12 V on A9 for the signature mode.

FUNCTION	MODE†							
	READ	OUTPUT DISABLE	STANDBY	PROGRAMMING	VERIFY	PROGRAM INHIBIT	SIGNATURE MODE	
$\overline{E}$	V_{IL}	V_{IL}	V_{IH}	V_{IL}	V_{IH}	V_{IH}	V_{IL}	
$\overline{G}$	V_{IL}	V_{IH}	X	V_{IH}	V_{IL}	X	V_{IL}	
V_{PP}	V_{CC}	V_{CC}	V_{CC}	V_{PP}	V_{PP}	V_{PP}	V_{CC}	
V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	V_{CC}	
A9	X	X	X	X	X	X	V_H‡	V_H‡
A0	X	X	X	X	X	X	V_{IL}	V_{IH}
DQ0–DQ7	Data Out	Hi-Z	Hi-Z	Data In	Data Out	Hi-Z	CODE	
							MFG	DEVICE
							97	04

†X can be V_{IL} or V_{IH}.
‡V_H = 12 V ± 0.5 V.

read/output disable

When the outputs of two or more TMS27C256s or TMS27PC256s are connected in parallel on the same bus, the output of any particular device in the circuit can be read with no interference from the competing outputs of the other devices. To read the output of a single device, a low-level signal is applied to the $\overline{E}$ and $\overline{G}$ pins. All other devices in the circuit should have their outputs disabled by applying a high-level signal to one of these pins. Output data is accessed at pins DQ0 through DQ7.

latchup immunity

Latchup immunity on the TMS27C256 and TMS27PC256 is a minimum of 250 mA on all inputs and outputs. This feature provides latchup immunity beyond any potential transients at the P.C. board level when the devices are interfaced to industry-standard TTL or MOS logic devices. Input-output layout approach controls latchup without compromising performance or packing density.

power down

Active I_{CC} supply current can be reduced from 30 mA to 500 μA (TTL-level inputs) or 250 μA (CMOS-level inputs) by applying a high TTL or CMOS signal to the $\overline{E}$ pin. In this mode all outputs are in the high-impedance state.

erasure (TMS27C256)

Before programming, the TMS27C256 EPROM is erased by exposing the chip through the transparent lid to a high intensity ultraviolet light (wavelength 2537 Å). EPROM erasure before programming is necessary to assure that all bits are in the logic high state. Logic lows are programmed into the desired locations. A programmed logic low can be erased only by ultraviolet light. The recommended minimum exposure dose (UV intensity × exposure time) is 15-W•s/cm^2. A typical 12-mW/cm^2, filterless UV lamp erases the device in 21 minutes. The lamp should be located about 2.5 cm above the chip during erasure. It should be noted that normal ambient light contains the correct wavelength for erasure. Therefore, when using the TMS27C256, the window should be covered with an opaque label.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

nitializing (TMS27PC256)

The one-time programmable TMS27PC256 PROM is provided with all bits in the logic high state, then logic lows are programmed into the desired locations. Logic lows programmed into an OTP PROM cannot be erased.

SNAP! Pulse programming

The 256K EPROM and OTP PROM are programmed using the TI SNAP! Pulse programming algorithm illustrated by the flowchart in Figure 1, which programs in a nominal time of four seconds. Actual programming time varies as a function of the programmer used.

Data is presented in parallel (eight bits) on pins DQ0 to DQ7. Once addresses and data are stable, $\overline{E}$ is pulsed.

The SNAP! Pulse programming algorithm uses initial pulses of 100 microseconds (μs) followed by a byte verification to determine when the addressed byte has been successfully programmed. Up to 10 (ten) 100-μs pulses per byte are provided before a failure is recognized.

The programming mode is achieved when V_{PP} = 13 V, V_{CC} = 6.5 V, $\overline{G} = V_{IH}$, and $\overline{E} = V_{IL}$. More than one device can be programmed when the devices are connected in parallel. Locations can be programmed in any order. When the SNAP! Pulse programming routine is complete, all bits are verified with $V_{CC} = V_{PP}$ = 5 V.

program inhibit

Programming can be inhibited by maintaining a high level input on the $\overline{E}$ pin.

program verify

Programmed bits can be verified with V_{PP} = 13 V when $\overline{G} = V_{IL}$ and $\overline{E} = V_{IH}$.

signature mode

The signature mode provides access to a binary code identifying the manufacturer and type. This mode is activated when A9 is forced to 12 V ± 0.5 V. Two identifier bytes are accessed by A0; i.e., A0 = V_{IL} accesses the manufacturer code, which is output on DQ0–DQ7; A0 = V_{IH} accesses the device code, which is output on DQ0–DQ7. All other addresses must be held at V_{IL}. The manufacturer code for these devices is 97, and the device code is 04.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

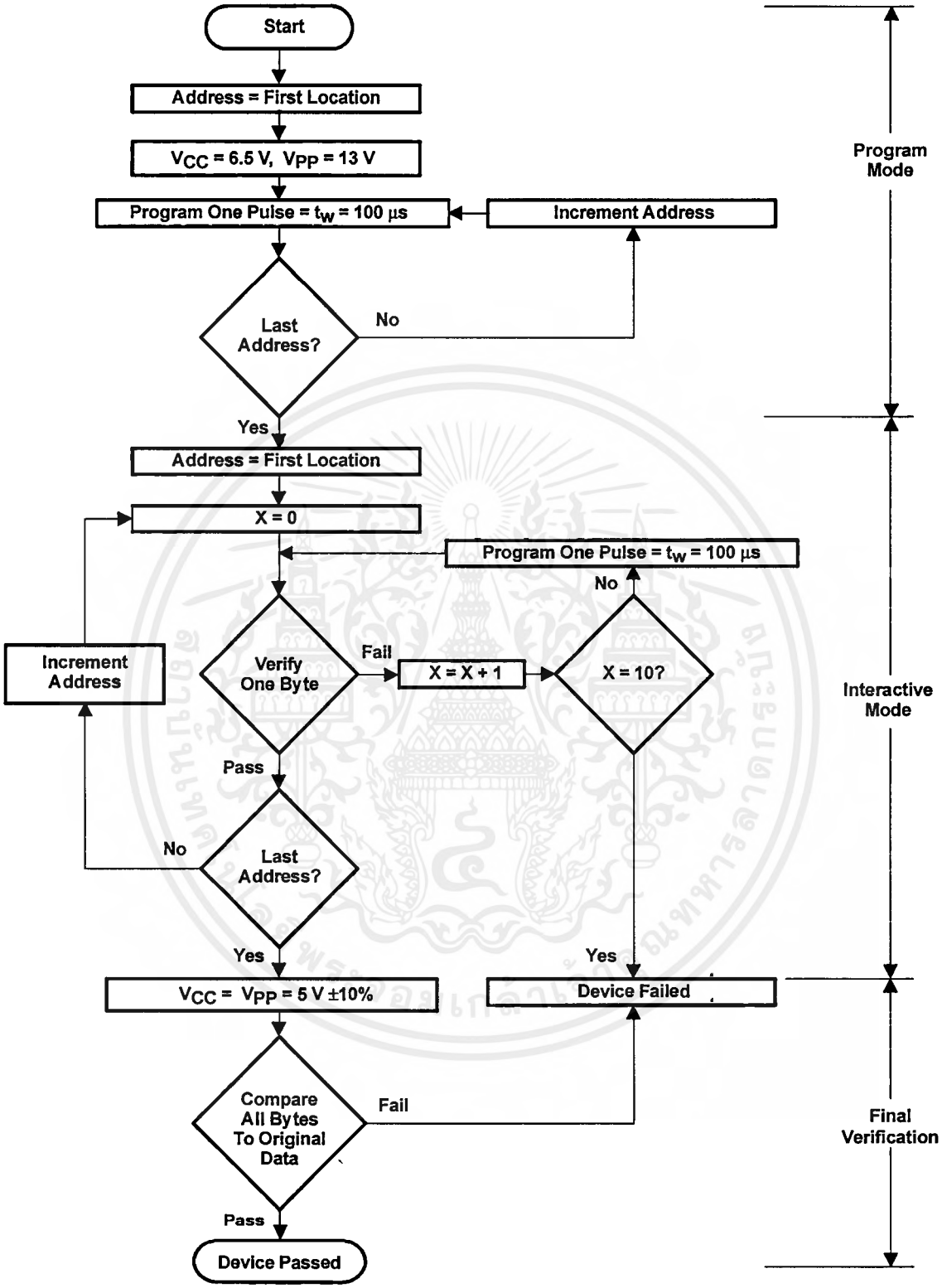
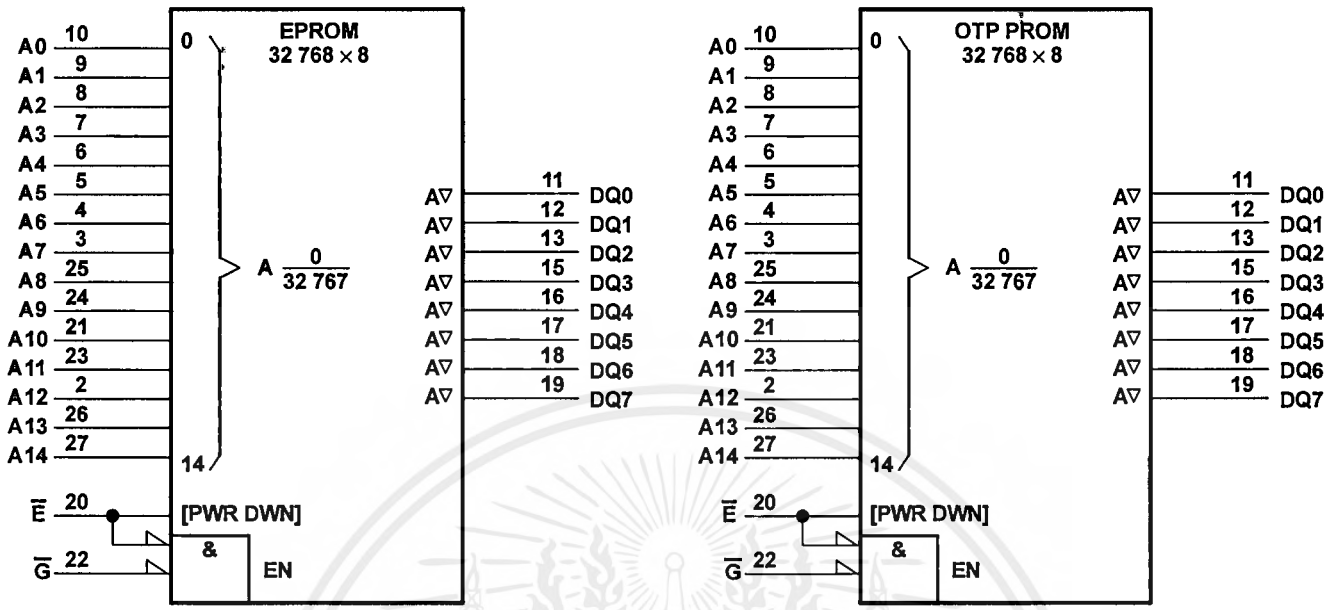


Figure 1. SNAP! Pulse Programming Flowchart

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

logic symbol†



† These symbols are in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.
Pin numbers shown are for J and N packages.

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)‡

Supply voltage range, V_{CC} (see Note 1) ..... −0.6 V to 7 V
Supply voltage range, V_{PP} ..... −0.6 V to 14 V
Input voltage range (see Note 1): All inputs except A9 ..... −0.6 V to V_{CC} + 1 V
A9 ..... −0.6 V to 13.5 V
Output voltage range (see Note 1) ..... −0.6 V to V_{CC} + 1 V
Operating free-air temperature range ('27C256-_ _JL and JL4, '27PC256-_ _NL, NL4, FML, and FML4) ..... 0° C to 70°C
Operating free-air temperature range ('27C256-_ _JE and JE4, '27PC256-_ _NE, NE4, FME, and FME4) ..... − 40° C to 85°C
Storage temperature range, T_{stg} ..... −65°C to 150°C

‡ Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: All voltage values are with respect to GND.

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265
POST OFFICE BOX 1443 • HOUSTON, TEXAS 77251–1443
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

recommended operating conditions

			MIN	NOM	MAX	UNIT
V_{CC}	Supply voltage	Read mode (see Note 2)	4.5	5	5.5	V
		SNAP! Pulse programming algorithm	6.25	6.5	6.75	
V_{PP}	Supply voltage	Read mode	$V_{CC}-0.6$		$V_{CC}+0.6$	V
		SNAP! Pulse programming algorithm	12.75	13	13.25	
V_{IH}	High-level dc input voltage	TTL	2		$V_{CC}+1$	V
		CMOS	$V_{CC}-0.2$		$V_{CC}+1$	
V_{IL}	Low-level dc input voltage	TTL	−0.5		0.8	V
		CMOS	−0.5		0.2	
T_A	Operating free-air temperature	'27C256-_ _JL, JL4 '27PC256-_ _NL, NL4, FML, FML4	0		70	°C
T_A	Operating free-air temperature	'27C256-_ _JE, JE4 '27PC256-_ _NE, NE4, FME, FME4	−40		85	°C

NOTE 2: V_{CC} must be applied before or at the same time as V_{PP} and removed after or at the same time as V_{PP}. The device must not be inserted into or removed from the board when V_{PP} or V_{CC} is applied.

electrical characteristics over recommended ranges of operating conditions

PARAMETER			TEST CONDITIONS	MIN	TYP†	MAX	UNIT
V_{OH}	High-level dc output voltage		I_{OH} = −2.5 mA	3.5			V
			I_{OH} = −20 µA	$V_{CC}-0.1$			
V_{OL}	Low-level dc output voltage		I_{OL} = 2.1 mA			0.4	V
			I_{OL} = 20 µA			0.1	
I_I	Input current (leakage)		V_I = 0 V to 5.5 V			±1	µA
I_O	Output current (leakage)		V_O = 0 V to V_{CC}			±1	µA
I_{PP1}	V_{PP} supply current		V_{PP} = V_{CC} = 5.5 V		1	10	µA
I_{PP2}	V_{PP} supply current (during program pulse)		V_{PP} = 13 V		35	50	mA
I_{CC1}	V_{CC} supply current (standby)	TTL-input level	V_{CC} = 5.5 V, $\overline{E}$ = V_{IH}		250	500	µA
		CMOS-input level	V_{CC} = 5.5 V, $\overline{E}$ = V_{CC}		100	250	
I_{CC2}	V_{CC} supply current (active)		V_{CC} = 5.5 V, $\overline{E}$ = V_{IL}, t_{cycle} = minimum cycle time, outputs open		15	30	mA

capacitance over recommended ranges of supply voltage and operating free-air temperature, f = 1 MHz‡

PARAMETER		TEST CONDITIONS	MIN	TYP†	MAX	UNIT
C_i	Input capacitance	V_I = 0, f = 1 MHz		6	10	pF
C_o	Output capacitance	V_O = 0, f = 1 MHz		10	14	pF

† Typical values are at T_A = 25°C and nominal voltages.
‡ Capacitance measurements are made on a sample basis only.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

switching characteristics over recommended range of operating conditions

PARAMETER		TEST CONDITIONS (SEE NOTES 3 AND 4)	'27C256-10 '27PC256-10		'27C256-12 '27PC256-12		'27C256-15 '27PC256-15		UNIT
			MIN	MAX	MIN	MAX	MIN	MAX	
$t_{a(A)}$	Access time from address	C_L = 100 pF, 1 Series 74 TTL Load, Input $t_r \le 20$ ns, Input $t_f \le 20$ ns		100		120		150	ns
$t_{a(E)}$	Access time from chip enable			100		120		150	ns
$t_{en(G)}$	Output enable time from $\overline{G}$			55		55		75	ns
t_{dis}	Output disable time from $\overline{G}$ or $\overline{E}$, whichever occurs first†		0	45	0	45	0	60	ns
$t_{v(A)}$	Output data valid time after change of address, $\overline{E}$, or $\overline{G}$, whichever occurs first†		0		0		0		ns

PARAMETER		TEST CONDITIONS (SEE NOTES 3 AND 4)	'27C256-17 '27PC256-17		'27C256-20 '27PC256-20		'27C256-25 '27PC256-25		UNIT
			MIN	MAX	MIN	MAX	MIN	MAX	
$t_{a(A)}$	Access time from address	C_L = 100 pF, 1 Series 74 TTL Load, Input $t_r \le 20$ ns, Input $t_f \le 20$ ns		170		200		250	ns
$t_{a(E)}$	Access time from chip enable			170		200		250	ns
$t_{en(G)}$	Output enable time from $\overline{G}$			75		75		100	ns
t_{dis}	Output disable time from $\overline{G}$ or $\overline{E}$, whichever occurs first†		0	60	0	60	0	60	ns
$t_{v(A)}$	Output data valid time after change of address, $\overline{E}$, or $\overline{G}$, whichever occurs first†		0		0		0		ns

† Value calculated from 0.5 V delta to measured level. This parameter is only sampled and not 100% tested.

switching characteristics for programming: V_{CC} = 6.50 V and V_{PP} = 13 V (SNAP! Pulse), T_A = 25°C (see Note 3)

PARAMETER		MIN	MAX	UNIT
$t_{dis(G)}$	Output disable time from $\overline{G}$	0	130	ns
$t_{en(G)}$	Output enable time from $\overline{G}$		150	ns

NOTES: 3. For all switching characteristics the input pulse levels are 0.4 V to 2.4 V. Timing measurements are made at 2 V for logic high and 0.8 V for logic low). (Reference page 9.)
4. Common test conditions apply for the t_{dis} except during programming.

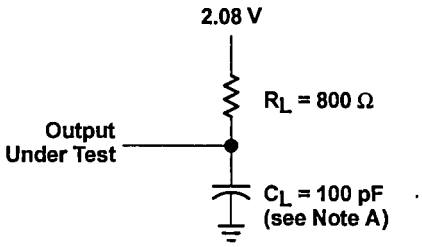
recommended timing requirements for programming: V_{CC} = 6.5 V and V_{PP} = 13 V, T_A = 25°C (see Note 3)

		MIN	NOM	MAX	UNIT
$t_{h(A)}$	Hold time, address	0			µs
$t_{h(D)}$	Hold time, data	2			µs
$t_{w(IPGM)}$	Pulse duration, initial program	95	100	105	µs
$t_{su(A)}$	Setup time, address	2			µs
$t_{su(G)}$	Setup time, $\overline{G}$	2			µs
$t_{su(E)}$	Setup time, $\overline{E}$	2			µs
$t_{su(D)}$	Setup time, data	2			µs
$t_{su(VPP)}$	Setup time, V_{PP}	2			µs
$t_{su(VCC)}$	Setup time, V_{CC}	2			µs

NOTE 3: For all switching characteristics the input pulse levels are 0.4 V to 2.4 V. Timing measurements are made at 2 V for logic high and 0.8 V for logic low). (Reference page 9.)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PARAMETER MEASUREMENT INFORMATION



NOTE A: C_L includes probe and fixture capacitance.

Figure 2. AC Testing Output Load Circuit

AC testing input/output wave forms

A.C. testing inputs are driven at 2.4 V for logic high and 0.4 V for logic low. Timing measurements are made at 2 V for logic high and 0.8 V for logic low for both inputs and outputs.

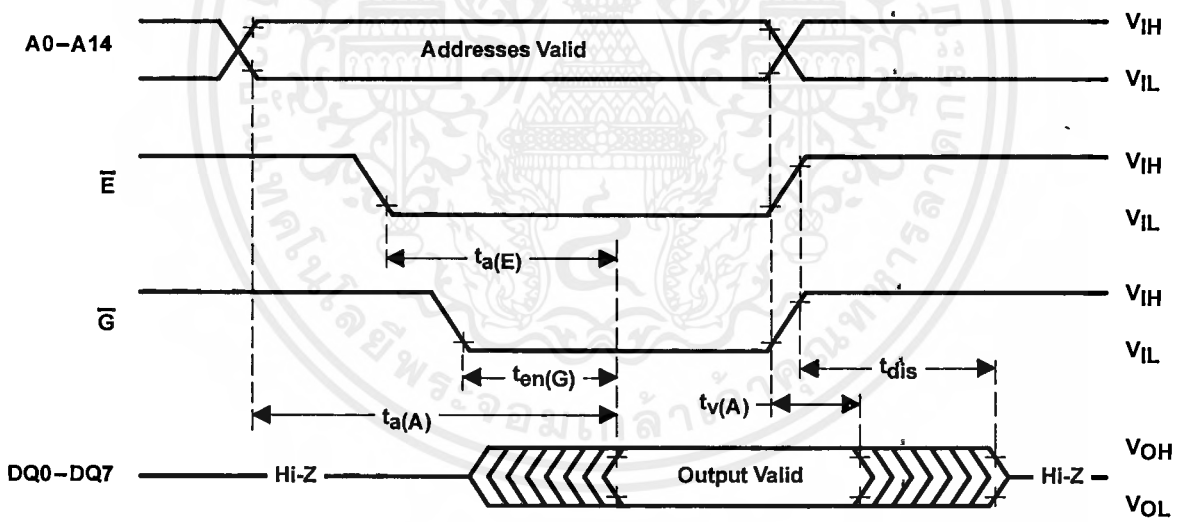
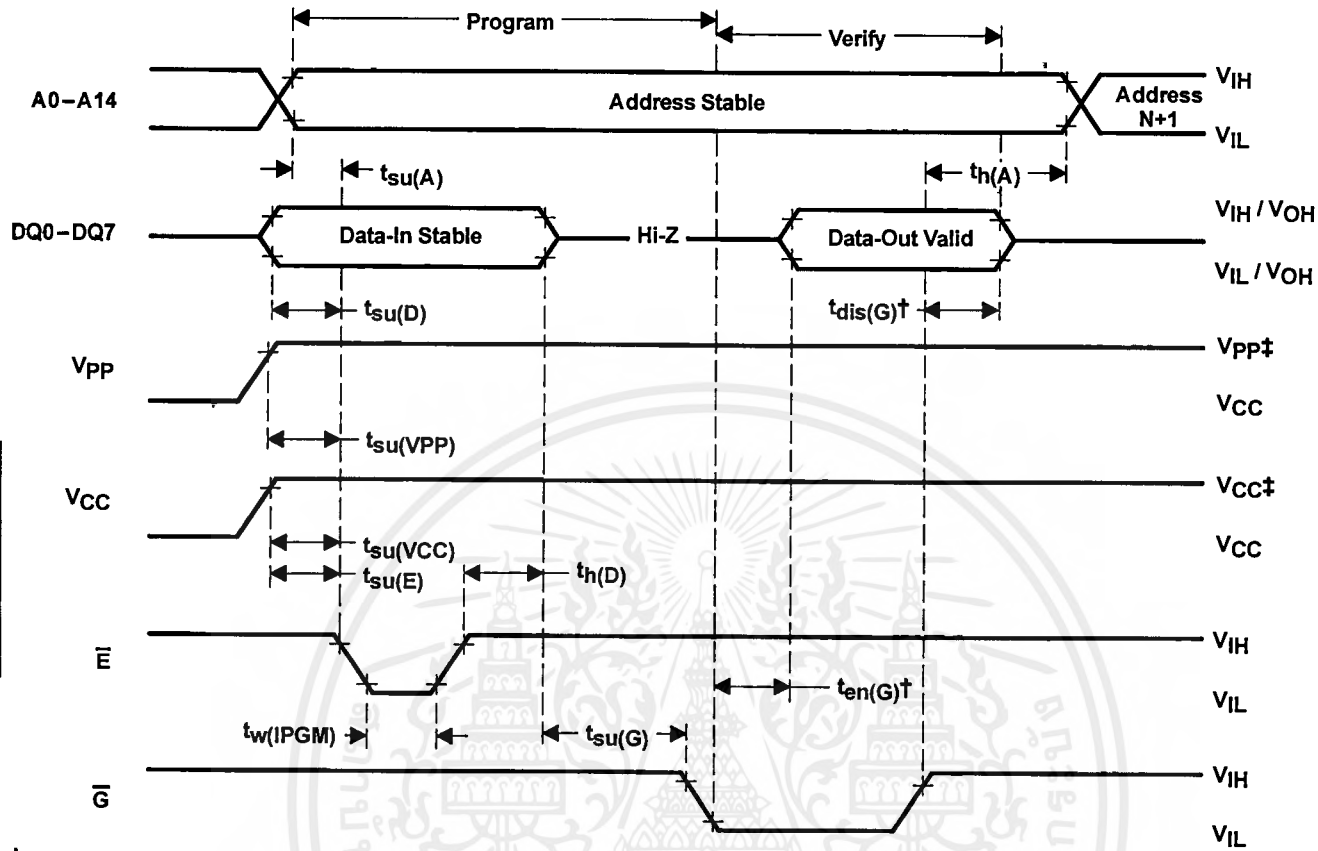


Figure 3. Read-Cycle Timing

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PARAMETER MEASUREMENT INFORMATION



† $t_{dis(G)}$ and $t_{en(G)}$ are characteristics of the device but must be accommodated by the programmer
‡ 13-V V_{PP} and 6.5-V V_{CC} for SNAP! Pulse programming

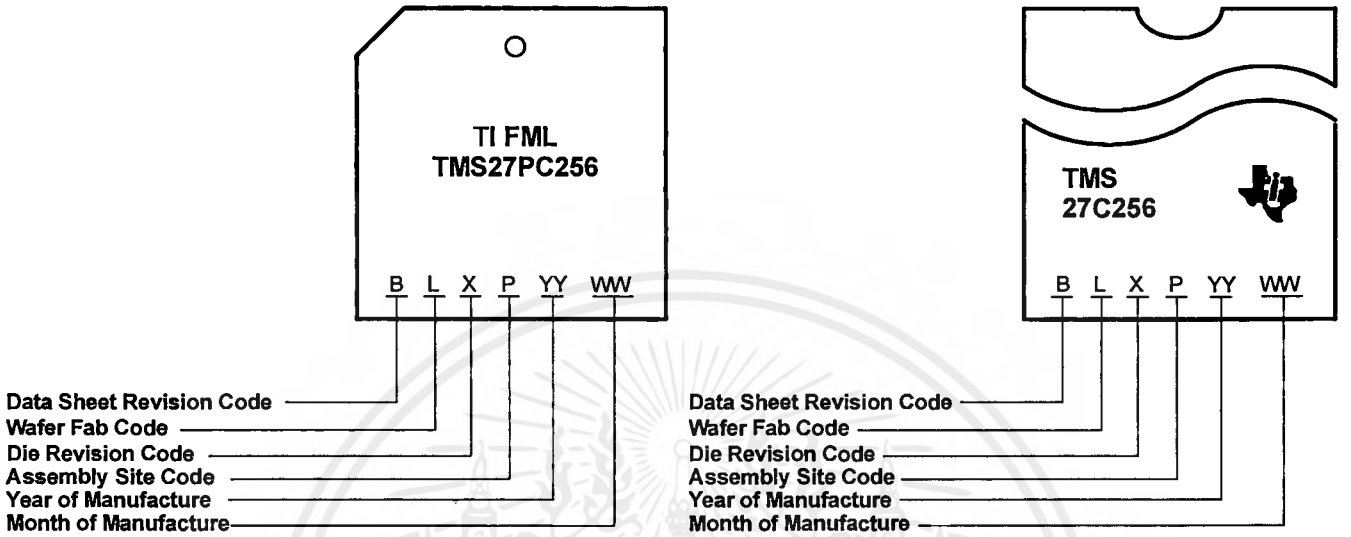
Figure 4. Program-Cycle Timing (SNAP! Pulse Programming)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงลิขสิทธิ์ของเอกสารทุกครั้งที่มีการนำไปใช้

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265
POST OFFICE BOX 1443 • HOUSTON, TEXAS 77251–1443

device symbolization

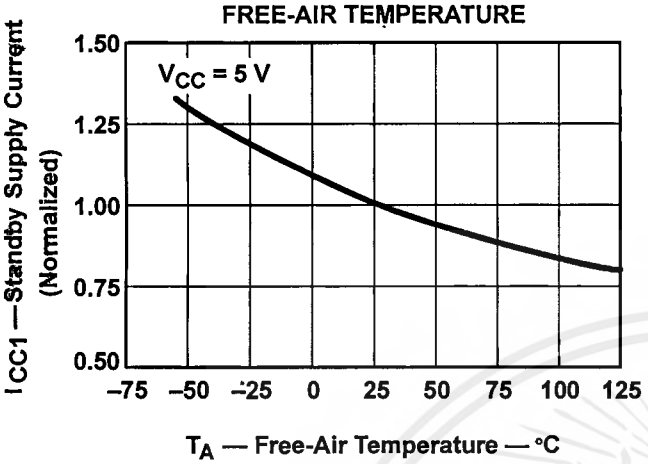
This data sheet is applicable to all TI TMS27C256 CMOS EPROMs and TMS27PC256 CMOS OTP PROMs with the data sheet revision code "B" as shown below.



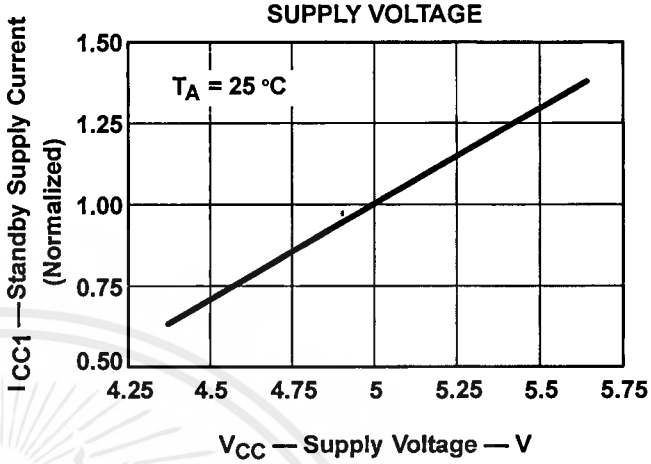
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลง เผยแพร่ หรือทำซ้ำ ก่อนได้รับอนุญาตจากเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TYPICAL TMS27C/PC256 CHARACTERISTICS

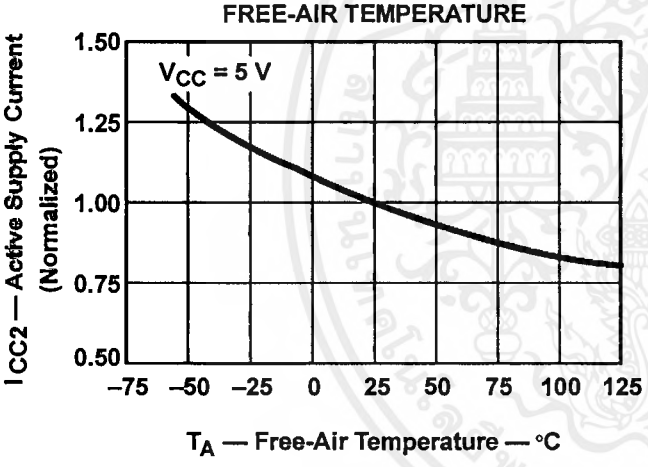
STANDBY SUPPLY CURRENT
vs
FREE-AIR TEMPERATURE



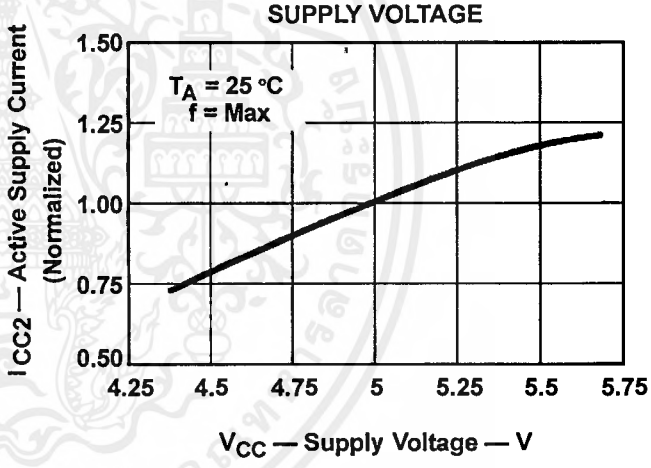
STANDBY SUPPLY CURRENT
vs
SUPPLY VOLTAGE



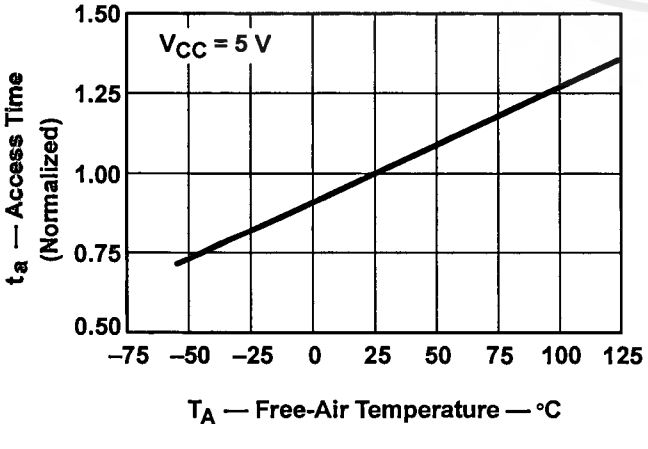
ACTIVE SUPPLY CURRENT
vs
FREE-AIR TEMPERATURE



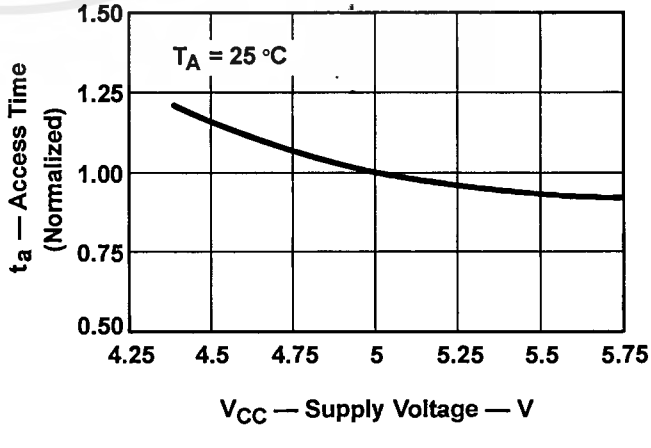
ACTIVE SUPPLY CURRENT
vs
SUPPLY VOLTAGE



ACCESS TIME
vs
FREE-AIR TEMPERATURE



ACCESS TIME
vs
SUPPLY VOLTAGE



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 • DALLAS, TEXAS 75265
POST OFFICE BOX 1443 • HOUSTON, TEXAS 77251–1443

กิตติกรรมประกาศ

- โครงงานนี้สำเร็จลุล่วงได้ด้วยดี จากความรู้ที่อาจารย์ทุกท่านในภาควิชาได้ถ่ายทอดความรู้ทางด้านวิชาการให้ รวมถึงอาจารย์ทุกๆ ท่านที่ได้ให้ความรู้ในทุกระดับการศึกษา มิฉะนั้นโครงงานนี้ก็ไม่อาจสำเร็จได้ และโดยเฉพาะอย่างยิ่งขอขอบคุณ อาจารย์สมศักดิ์ มิตะถา อาจารย์ที่ปรึกษาที่ให้คำแนะนำด้านต่างๆ และคอยดูแลควบคุมการทำงานของสมาชิกในกลุ่ม
- ขอบคุณ คุณจาตุรนต์ พวงนาค ที่ให้ความอนุเคราะห์เครื่องคอมพิวเตอร์มาใช้ในการทำโครงงานและใช้ในการนำเสนอโครงงาน
- ขอขอบคุณเพื่อนๆ และน้องๆ ทุกคนที่ให้กำลังใจในการทำโครงงานนี้
- ขอขอบคุณการรถไฟแห่งประเทศไทย และ ขสมก. ที่เป็นยานพาหะนะในการเดินทางเพื่อไปซื้ออุปกรณ์ในการประกอบโครงงาน
- ขอขอบคุณร้านค้าอุปกรณ์อิเลคทรอนิคส์ย่านบ้านหม้อ
- ขอขอบคุณ ร้านพี่ยุทธเชิงสะพานฉลองกรุง (ติดโรงงาน FBT) , ร้าน Seven Eleven และร้านข้าวแกงหัวตะเข้ ที่ขายอาหารยามค่ำคืน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เอกสารอ้างอิง

1. ชูชัย ธนสารตั้งเจริญ, ดนัย แสงสุริยศิลป์, ทินกร ดุ๊ก, ธงชัย อุดมกิจโดศล, ธานินทร์ ถาวรศาสนวงศ์, "การใช้งาน Z80", ฟิสิกส์เซนเตอร์, 284 หน้า
2. ชูชัย ธนสารตั้งเจริญ, ทินกร ดุ๊ก, "การสื่อสารข้อมูล", ฟิสิกส์เซนเตอร์, 203 หน้า
3. ธันวา ศรีประโมง, "การเขียนโปรแกรมภาษา C สำหรับวิศวกรรม", มหาวิทยาลัยเทคโนโลยีมหานคร, 739 หน้า, 2537
4. ปรเมษฐ์ ประนยานนท์, ปิยพงศ์ เผ่าวณิช, "คู่มือและการประยุกต์ใช้งานไมโครคอนโทรเลอร์ MCS-51", ซีเอ็ดยูเคชั่น, 380 หน้า
5. พงษ์ระพี เตชพาหุพงษ์, "แอดวานซ์ เอ็มเอสดอส", ซีเอ็ดยูเคชั่น, 297 หน้า, 2529
6. มโน มงคลธนานนท์, มงคล มงคลธนานนท์, "คัมภีร์โปรแกรมเมอร์", อินฟอเมติก บิซิเนส พับลิเคชั่น, 518 หน้า
7. ราบินเดอร์ ศรีกิจจาภรณ์, "การเขียนโปรแกรมแบบโอโอพีด้วยเทอร์โบและบอร์แลนด์ C++", ซีเอ็ดยูเคชั่น, 787 หน้า
8. สุเจตน์ จันทรังษ์, "ไมโครคอนโทรเลอร์ชิพเดี่ยว 8051", มหาวิทยาลัยเทคโนโลยีมหานคร, 187 หน้า, 2535
9. สุชาติ กังวารจิตต์, "เครื่องรับวิทยุและระบบวิทยุสื่อสาร", ซีเอ็ดยูเคชั่น, 386 หน้า, 2536
10. Aaron M. Tenenbaum, Moshe J. Augensteia, "Data Structure Using Pascal", Prentice Hall INC., Upper Saddle River, New Jersey, 774 p.,1981

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้