เครื่องอ่าน / เขียนข้อมูล โดยใช้ EEPROM

Read / Write Data by EEPROM

โดย

นาย ภูริน วีระวัฒนาเดช

นาย สิทธิพร อานามนารถ

นาย สุเชษฐ์ จุลนิธิ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตร์บัณฑิต

สาขา วิศวกรรมการวัดคุมทางอุตสาหกรรม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2541

ปริญญานิพนธ์ ปีการศึกษา 2541

ภาควิชา เทคโนโลยีการวัดคุมทางอุตสาหกรรม

สาขาวิชา วิศวกรรมการวัดคุมทางอุตสาหกรรม

คณะ วิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง Read / Write Data by EEPROM

ผู้จัดทำ

1. นาย ภูริน วีระวัฒนาเดช 39012099
2. นาย สิทธิพร อานามนารถ 39012111
3. นาย สุเชษฐ์ จุลนิธิ 39012112

..... อาจารย์ที่ปรึกษา

(อาจารย์ ทวีพล ซื่อสัตย์)

หัวข้อปริญญานิพนธ์	เครื่องอ่าน / เขียนข้อมูล PLC โดยใช้ EEPROM	
นักศึกษา	นาย ภูริน วีระวัฒนาเดช	39012099
	นาย สิทธิพร อานามนารถ	39012111
	นาย สุเชษฐ์ จุลนิธิ	39012112
อาจารย์ที่ปรึกษา	อาจารย์ ทวีพล ซื่อสัตย์	
ระดับการศึกษา	วิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมการวัดคุมทางอุตสาหกรรม	
ปีการศึกษา	พ.ศ. 2541	

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้ เป็นการศึกษาการทำงานและการใช้งานในเรื่องการติดต่อสื่อสารข้อมูลระหว่าง PLC (Programmable Logic Control) , ไมโครคอนโทรลเลอร์(MCS 51) สามารถแสดงผลและเปลี่ยนแปลงข้อมูลด้วยคอมพิวเตอร์ (Computer) ได้ ซึ่งในการติดต่อสื่อสารนี้จะเป็นการติดต่อสื่อสารทางพอร์ตอนุกรม(Serial Port) แบบ Asynchronous ในมาตรฐาน RS-232

การที่จะทำให้ PLC รับรู้ถึงการส่งนี้จะต้องมีการป้อนข้อมูลให้ PLC ทาง Host Link โดยส่งเข้า PLC เป็นบล็อกคำสั่ง(Command Block) จะทำให้ PLC ตอบสนองได้ซึ่งจากจุดนี้การเขียนบล็อกคำสั่ง(Command Block) จะต้องมีการระบุข้อมูลต่าง ๆ คือ ตำแหน่งที่ตั้ง(Unit number) , รหัสต้นทาง (Header Code) ,รหัสปลายทาง(ends code) ,รหัสตรวจสอบความถูกต้อง(Frame Check Sequence code (FCS)) และ ตำแหน่งปลายทาง(Terminator) ซึ่งจำเป็นต้องถูกต้อง PLC จึงจะตอบสนองต่อคำสั่ง การเขียนบล็อกคำสั่ง(Command Block) จะเป็นการเขียนโปรแกรม Virtual Basic บนคอมพิวเตอร์ ส่งออกไปทางพอร์ต COM1

การเขียนโปรแกรมเพื่อติดต่อกับ EEPROM โดยไมโครคอนโทรลเลอร์(MCS-51) จะเขียนเป็นภาษา C 51 ซึ่งเป็นโปรแกรมที่สามารถคอมไพล์ภาษา C (File.C) ให้เป็นภาษาเครื่อง (File.HEX) ได้ซึ่งจะสะดวกในการพัฒนาโปรแกรม และง่ายต่อการทำความเข้าใจ

Thesis	Read / Write Data by EEPROM		
Students	Mr. Purin	Veravattanadej	39012099
	Mr. Sittiporn	Arnarmnad	39012111
	Mr. Suchet	Junniti	39012112
Advisor	Taweepol	Suesut	
Education Level	Bachelor of industrial instrument engineering		
Education Year	1998		

Abstract

This Thesis education about data communication of PLC and other device . To stress in communication between PLC (Programmable Logic Control) and MCS 51 module and to be able to communicate with PC (computer). This communication is Asynchronous Serial type in RS-232 standard.

PLC can receive data by Host link. The data in command block has pattern from Unit number , Header code , Ends code with a Frame check sequence , (FCS) code and terminator. It will corrected command for PLC response to work.

The Program is written for connection between EEPROM and MCS-51 can by program written in C language it used complied by C-51 to (File.C) and make to machine this language (File.HEX) . C language has easy to understand and develop program.

กิตติกรรมประกาศ

โครงงานเครื่อง อ่าน/เขียน ข้อมูล โดย EEPROM คงไม่สามารถสำเร็จลงได้ถ้าไม่ได้รับความช่วยเหลือและสนับสนุนจาก ท่านอาจารย์ ทวีพล ซื่อสัตย์ และ ท่านคณะอาจารย์ภาควิชาเทคโนโลยีการวัดคุมทางอุตสาหกรรมทุกท่าน ที่ให้ความรู้และคำปรึกษาที่ดีตลอดมาจนโครงงานนี้สามารถทำได้สำเร็จตามวัตถุประสงค์จนกลายเป็นปริญญานิพนธ์ฉบับนี้ได้จนเสร็จสมบรูณ์ ขอขอบคุณเพื่อนๆจาก สจล. , สจพ. และ อีกหลายๆสถาบัน ทุกท่านที่ให้ความช่วยเหลือและเป็นกำลังใจให้ตลอดเวลา

กราบขอบพระคุณบุพการี ญาติพี่น้อง และ ท่านอาจารย์ที่ประศิตประศาสตร์วิชาทุกท่านที่ได้ให้ การเลี้ยงดู อุปถัมภ์ และ ความรู้ แก่ข้าพเจ้ามาโดยตลอด

ภูริน วีระวัฒนาเดช
สิทธิพร อานามนารถ
สุเชษฐ์ จุลนิธิ

สารบัญ

สารบัญรูป

หน้า

รูปที่

สารบัญรูป (ต่อ)

สารบัญตาราง

หน้า

ตารางที่

บทที่ 1
บทนำ

ความสำคัญและที่มาของโครงงาน

ระบบการควบคุมแบบอัตโนมัติได้เข้ามามีบทบาทในขบวนการผลิตทางอุตสาหกรรมในปัจจุบันอย่างมาก เนื่องจากระบบการควบคุมแบบอัตโนมัติมีความคล่องตัวต่อการควบคุม และยังสามารถแก้ไขหรือดัดแปลงแผนการผลิตได้รวดเร็วทำให้ขบวนการผลิตมีประสิทธิภาพในการผลิตสูง เป็นผลทำให้ผู้ประกอบการ สามารถควบคุมอัตราการผลิตได้ตามอัตราที่ต้องการนั่นหมายถึงการควบคุมต้นทุนจากการผลิตซึ่งมีผลต่อผลกำไรที่จะได้รับ

PLC เป็นรูปแบบหนึ่งของการควบคุมแบบอัตโนมัติที่นิยมใช้ในปัจจุบันเนื่องจากสามารถทำความเข้าใจเพื่อนำไปใช้งานได้ง่าย และสามารถนำไปใช้งานได้อย่างกว้างขวาง อย่างไรก็ตาม PLC ไม่สามารถคิดหรือควบคุมการทำงานได้ด้วยตัวเอง เพียงแต่ปฏิบัติตามคำสั่งหรือโปรแกรมที่ผู้ใช้งานระบุให้ โดยที่ภาษาที่ใช้ในการเขียนโปรแกรมมีหลายภาษาเพื่อให้สะดวกในการพัฒนาโปรแกรม

จากข้อมูลดังกล่าวข้างต้น จึงได้มีแนวความคิดที่จะประดิษฐ์ Module เคลื่อนที่ขึ้นมาเพื่อใช้เป็นตัวกลางในการแลกเปลี่ยนข้อมูลระหว่าง PLC, Module, PC (Computer) และ อุปกรณ์ที่มีการติดต่อสื่อสารแบบเดียวกันเช่น การสื่อสาร RS 232 เพื่อความคล่องตัวในการแลกเปลี่ยนข้อมูลในการทำงานโดยส่งผ่านข้อมูลจาก IC Card ซึ่งบรรจุคำสั่งที่ใช้ในการแลกเปลี่ยนข้อมูลไว้ภายใน เข้าสู่ PC และนำโปรแกรมจากภายใน EEPROM ไปควบคุมการทำงานของ PLC เพื่อให้ขบวนการผลิตเป็นไปตามที่ต้องการ และยังสามารถไปแสดงผลที่ PC และทำการแก้ไขข้อมูลได้

วัตถุประสงค์ของโครงงาน

1. เพื่อศึกษาและพัฒนา การติดต่อสื่อสารผ่านทาง พอร์ตอนุกรมโดยใช้มาตราฐาน RS 232
2. สามารถเข้าใจถึงการทำงานของ PLC ในการส่งข้อมูล
3. สามารถแสดงผลต่างๆในการติดต่อออกทาง คอมพิวเตอร์ได้
4. เพื่อความรวดเร็วในการส่งผ่านข้อมูล
5. เพื่อนำไปพัฒนางานอุตสาหกรรมให้เกิดความคล่องตัว
6. เพื่อแสดงเป็นแนวความคิดเริ่มต้นในการนำไปพัฒนา Module ที่ใช้ในงานประเภทเดียวกันและประยุกต์ใช้ในงานประเภทอื่นๆ

วิธีการดำเนินโครงงาน

การดำเนินงานแบ่งเป็นขั้นตอนดังนี้

1. ศึกษาหาข้อมูลของ EEPROM และโปรแกรมในการควบคุม
2. ศึกษาโปรแกรมการติดต่อสื่อสารข้อมูลแบบอนุกรมด้วยมาตราฐาน RS 232
3. ศึกษาการเขียนโปรแกรมแสดงผลที่ PC
4. ศึกษาการติดต่อสื่อสารข้อมูลโดยใช้เครื่อง PLC
5. ออกแบบวงจร และ จัดซื้ออุปกรณ์
6. ศึกษาโปรแกรมภาษา C และ Visaul Basic
7. ออกแบบโปรแกรม และ ตัว Module
8. ทดสอบการใช้งาน
9. สรุปผลการทำโครงงาน

ขอบเขตโครงงาน

1. ศึกษาการอ่าน / เขียนข้อมูลลงใน EEPROM โดยการควบคุมของ MCS-51
2. นำข้อมูลที่ได้มาแสดงผลในส่วนของ PC (computer)
3. สามารถส่งข้อมูลเข้า PLC โดยผ่านทาง PC และ PLC สามารถส่งข้อมูลกลับมาที่ IC CARD ได้
4. สามารถนำเครื่องมือนี้ไปประยุกต์กับอุปกรณ์อื่นๆได้

ประโยชน์ที่คาดว่าจะได้รับ

1. สามารถเขียนโปรแกรมที่จะทำการติดต่อสื่อสาร ตามแบบมาตรฐาน RS 232 ได้
2. สามารถทำการอ่าน / เขียนข้อมูลกับ EEPROM ซึ่งเป็นเบอร์มาตรฐานที่มีขายตามท้องตลาด จึงสามารถนำหลักการของโปรแกรมในโครงงานนี้ไปประยุกต์เพื่อให้สามารถใช้ได้กับ EEPROM เบอร์อื่น ๆ (ที่สามารถหา Data Sheet ได้) และนำไปใช้ประโยชน์ในงานอื่นๆ ที่ใช้ EEPROM ในการเก็บข้อมูลได้
3. สามารถทำการติดต่อสื่อสารเปลี่ยนแปลงข้อมูลให้ PLC ซึ่งเป็นอุปกรณ์ที่ใช้ในงานอุหสาหกรรมอย่างกว้างขวาง
4. ได้เครื่องต้นแบบอ่าน / เขียนข้อมูลจาก EEPROM เพื่อไปเปลี่ยนแปลงค่าในหน่วยความจำข้อมูลของเครื่อง PLC

บทที่ 2
ทฤษฎีและหลักการที่นำมาใช้

ในการทำโครงงานนี้ต้องทำการศึกษาทฤษฎีในหลาย ๆ อย่างเนื่องจาก Module ที่ต้องการสร้างต้องสามารถติดต่อกับ EEPROM , PLC และ PC โดยใช้มาตรฐานการสื่อสารแบบ RS-232 เป็นสื่อกลางได้ ดังนั้นจึงจะแบ่งส่วนของทฤษฎีออกเป็นหัวข้อใหญ่ ๆ ดังนี้คือ

1. PLC
2. ระบบการสื่อสารข้อมูล
3. มาตรฐาน RS-232
4. MCS-51

PLC/PC (Programmable Logic Controllers/Programmable Controller)

เป็นเครื่องควบคุมที่สามารถกำหนดการโปรแกรมได้ PLC/PC ที่ใช้ไมรโครโปรเซสเซอร์นอกจากจะใช้ในการควบคุมเครื่องจักรโดยทั่วไปแล้ว ยังมีการให้พัฒนาความสามารถและขอบเขตของงานได้กว้างขวางขึ้น เช่น มีฟังชั่นทางคณิตศาสตร์เพิ่มขึ้น ทำให้การควบคุมเป็นไปได้ทั้งแบบ ON-OFF หรือแบบอนาล็อก (Analog) เช่น PID (Proportional Integral Derivative) สามารถต่อเข้ากับอุปกรณ์วัดและควบคุม (Instrumentation) อื่นๆเช่นเครื่องอ่าน รหัสแถบเพื่อจำแนก หรือ จัดทำฐานข้อมูล การเชื่อมโยงข้อมูลลักษณะเครือข่ายท้องถิ่น (LAN) เป็นต้น หน่วยความจำก็ขยายได้มาก ทำให้ใช้กับระบบกระบวนการใหญ่ๆได้ และมีการจัดการเกี่ยวกับข้อมูลได้ตามต้องการ

สรุปข้อดีของ PLC/PC

ข้อดีของ PLC/PC สามารถแบ่งออกได้เป็นข้อๆดังนี้

1. สามารถจะใช้ควบคุมเครื่องจักรหรือระบบกระบวนการใดๆก็ได้ ถ้าเลือกขนาดของ PLC/PC เหมาะสม
2. การเปลี่ยนลำดับขั้นตอนหรือเงื่อนไขของการทำงานก็ทำได้ตามต้องการ เพราะใช้หลักของการโปรแกรม
3. ตัวตั้งเวลาและตัวนับจะเป็นซอฟท์แวร์ทำให้การกำหนดค่าต่างๆง่าย เปลี่ยนแปลงค่าได้ตลอดเวลาไม่ต้องมีฮาร์ดแวร์ร่วม และทำให้ราคาถูกลง
4. รีเลย์ภายใน (Internal relay) ก็เป็นซอฟท์แวร์เช่นเดียวกัน จึงลดค่าใช้จ่ายในการเดินสาย ลดฮาร์ดแวร์ และทำให้ขนาดเล็กลงด้วย
5. การติดตั้งทำได้ง่ายและสะดวก
6. การขยายระบบให้ใหญ่ขึ้นทำได้ง่าย
7. ความน่าเชื่อถือ (Reliability) ดีเพราะเป็นอุปกรณ์สารกึ่งตัวนำ ไม่มีการเดินสายมากไม่มีปัญหาเกี่ยวกับหน้าสัมผัส (Contact) แบบรีเลย์

8. มีการตรวจสอบหาที่ผิดพลาดด้วยตัวเอง การตรวจสอบแก้ไขเมื่อมีปัญหาจึงทำได้รวดเร็ว

9. การบำรุงรักษาทำได้ง่าย

10. เวลาในการทำงานเร็วกว่าระบบที่ใช้รีเลย์

11. มีฟังก์ชั่นทางคณิตศาสตร์ได้แก่ บวก ลบ คูณ หาร และอื่นๆทำให้สามารถใช้สำหรับการควบคุมแบบ ON-OFF หรือแบบอนาล็อก เช่น PID ได้

12. สามารถเชื่อมต่อกับอุปกรณ์การวัด เช่น เทอร์โมคับเปิ้ล (Thermocouple) และอื่นๆได้นอกเหนือจากอุปกรณ์ตรวจวัดที่เป็นสวิทช์

13. ต่อเข้ากับคอมพิวเตอร์เพื่อวัตถุประสงค์ใดๆ เช่น การเก็บข้อมูลการกระจายการควบคุม (Distributed Control) เป็นต้น

14. การโปรแกรมทำได้หลายแบบ เช่น คำสั่งในรูปของแลดเดอร์ไดอะแกรม คำสั่งบูลีน (Boolean Instruction) คำสั่งในรูปบล็อค (Block Instrution) หรือคำสั่งภาษาเบสิก

15. ใช้ในทุกสภาพแวดล้อมของงาน อุตสาหกรรม

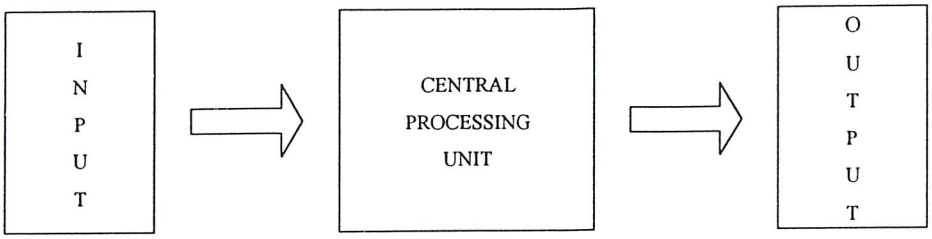
จากที่กล่าวมานี้จะเห็นว่า PLC/PC เป็นเครื่องควบคุมที่มีประสิทธิภาพและมีประโยชน์อย่างยิ่งต่องานควบคุมในปัจจุบันและในอนาคต

โครงสร้างและหลักการทำงานของ PLC

PLC/PC เป็นอุปกรณ์สารกึ่งตัวนำ ที่ใช้สำหรับควบคุมเครื่องจักร หรือระบบกระบวนการให้ทำงานได้ตามโปรแกรมคำสั่งของผู้ใช้ (user program)และข้อมูลต่างๆ ที่ได้รับจากอินพุท/เอาท์พุทของ PLC/ PC จะได้ทั้งการทำงานตามช่วงเวลา ตามลำดับขั้นตอนฟังก์ชั่นทางคณิตศาสตร์ และอื่นๆ PLC/PC มีสวนประกอบ 2 ส่วนคือ

1. หน่วยประมวลผลกลางหรือ CPU (Central Processing Unit)
2. หน่วยอินพุท/เอาท์พุท (Input/Output Unit)

นอกจากส่วนประกอบทั้งสองแล้ว PLC/PC ยังประกอบด้วยหน่วยป้อนโปรแกรม (Programming Unit) ผู้ใช้สามารถติดต่อกับ PLC/PC หรือเปลี่ยนแก้ไขโปรแกรมคำสั่งตรวจสอบสภาพการทำงานของเครื่องจักร หรือระบบกระบวนการ ตลอดจนตรวจสอบสภาพการทำงานของ PLC/PC ได้ทางหน่วยป้อนโปรแกรม เพื่อป้อนหรือเปลี่ยนแปลงแก้ไขโปรแกรม คำสั่งตรวจสอบสภาพการทำงานของเครื่องจักร หรือระบบกระบวนการตลอดจนตรวจสอบสภาพการทำงานของ PLC/PC รูปที่ 1แสดงส่วนประกอบที่สำคัญทั้ง 2 ส่วนที่สำคัญของ PLC/PC



รูปที่ 2.1 บล็อกแสดงไดอะแกรมของ PLC/PC

หน่วยประมวลผลกลาง

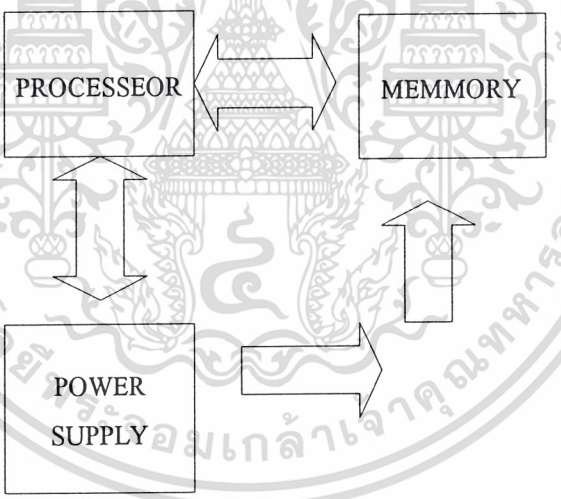
หน่วยประมวลผลกลางหรือ CPU ประกอบด้วย 3 ส่วนคือ

หน่วยประมวลผล (Processor)

หน่วยความจำ (Memory)

หน่วยจ่ายกำลัง (Power Supply)

รูปที่ 2.2 แสดงส่วนประกอบของ CPU โดยทั่วไป CPU จะใช้ไมรโครโปรเซสเซอร์ซึ่งเป็นวงจรรวม (IC: Integrated Circuit) ที่มีความสามารถทั้งการคำนวณทางคณิตศาสตร์ การจัดการข้อมูลและการตรวจสอบตัวเอง ในขณะที่วงจรควบคุมที่ใช้รีเลย์หรือไอซีพวกเกทต่างๆ ไม่สามารถจะทำได้ PLC/PC จึงมีข้อดีและประโยชน์มากกว่าระบบแบบเก่าอย่างเห็นได้ชัด



รูปที่ 2.2 บล็อคไดอะแกรมของ CPU

หน่วยประมวลผล

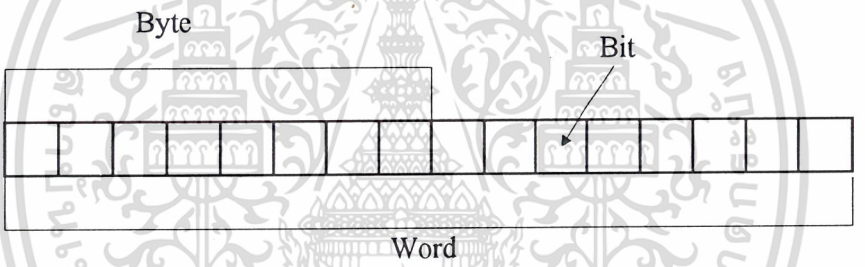
หน่วยประมวลผลของ CPU ทำหน้าที่ควบคุมและดูแลการทำงานของระบบทั้งหมด โดยรับข้อมูลจากหน่วยอินพุท มาทำการประมวลผลตามโปรแกรมคำสั่งของผู้ใช้ที่เก็บไว้ในหน่วยความจำ และส่งผลที่ได้ไปที่หน่วยเอาท์พุท ในปัจจุบันหน่วยประมวลผลของ CPU จะใช้ไมโครโปรเซสเซอร์ขนาดตั้งแต่ 4 บิท 8 บิท หรือ 16 บิท จำนวนหนึ่งตัวหรือหลายๆตัวมาทำงานร่วมกัน ในกรณีที่ใช้ไมโครโปรเซสเซอร์หลายๆ ตัวจะมีข้อดีคือ การทำงานของระบบจะเร็วขึ้น ทำให้ PLC/PC มีขีดความสามารถสูงขึ้น เช่น การใช้ชุดเชื่อมต่อพิเศษที่มีไมโครโปรเซสเซอร์และหน่วยความจำภายใน จะทำให้การทำงานควบคุมดียิ่งขึ้น

และเป็นอิสระไม่ขึ้นกับ CPU ตัวอย่างก็คือชุดควบคุม PID ที่ใช้สำหรับควบคุมระบบแบบลูฟปิด (Close-Loop Control) :ซึ่งทำงานอิสระจาก CPU

หน่วยความจำ

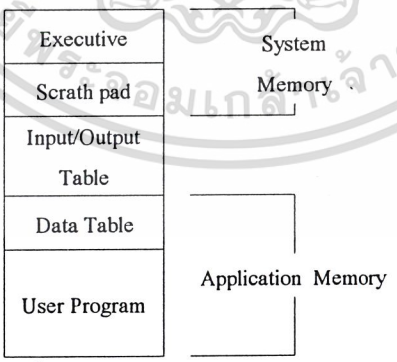
หน่วยความจำเป็นที่เก็บโปรแกรมและข้อมูลที่หน่วยประมวลผลใช้ในการควบคุมการทำงานของ PLC/PC โปรแกรมตามคำสั่งของผู้ใช้ ขนาดของหน่วยความจำจะแบ่งออกเป็นบิทข้อมูลภายในหน่วยความจำ 1 บิท จะมีค่าสภาวะทางลอจิก “1”หรือ “0”ข้อมูลขนาด 8บิท และ 16บิท รวมเรียกว่าไบท์ (Byte) และเวิร์ด(Word) ตามลำดับ

หน่วยความจำแบ่งออกเป็น 2 ประเภทใหญ่ๆ คือหน่วยความจำที่ข้อมูลสูญหายเมื่อไม่มีแหล่งจ่ายกำลัง (Volatile) และหน่วยความจำที่ข้อมูลคงอยู่แม้ไม่มีแหล่งจ่ายกำลัง (Non-Volatile) หรืออาจจะแบ่ง เป็น 2 ชนิดใหญ่ๆ ตามคุณลักษณะคือ RAM (Random Access Memory) หรือ NOVRAM (Nonvolatile RAM) และ ROM (Read Only Memory) นอกจากนี้ ROM ยังจำแนกออกได้เป็น PROM (Programmable ROM) EPROM (Erasable Programmable ROM) EAROM (Electrically Alterable ROM) และ EEPROM (Electrically Erasable Programmable ROM)



รูปที่ 2.3 ข้อมูลเป็นบิทไบท์และเวิร์ด

หน่วยความจำของ PLC/PC แบ่งออกเป็น 4 ส่วน ตามรูปที่ 4 เพื่อใช้เก็บข้อมูลและโปรแกรมต่างๆคือ



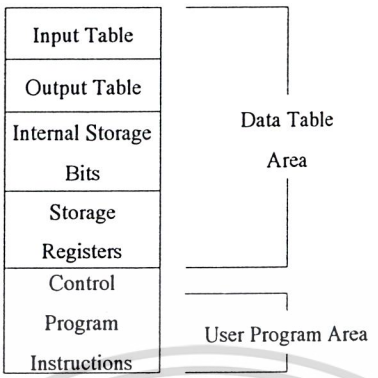
รูปที่ 2.4 หน่วยความจำของ PLC/PC

1.โปรแกรมจัดการ (Execute program) เป็นโปรแกรมที่ทำหน้าที่ควบคุม ดูแล และตรวจสอบการทำงานของ PLC/PC ทั้งหมด

2.ข้อมูลชั่วคราว(Processor Work Area หรือ Scratch pad) หน่วยความจำส่วนที่สองทำหน้าที่เก็บข้อมูลที่เกิดขึ้นระหว่าง PLC/PC ทำงานตามโปรแกรมจัดการและโปรแกรมคำสั่งของผู้ใช้

3.ตารางข้อมูล (Data table) หน่วยความจำส่วนที่สามทำหน้าที่เก็บ ค่าของอินพุท เอาท์พุท และตัวแปรต่างๆจากการทำงานตามโปรแกรมคำสั่งของผู้ใช้ทั้งที่เป็นค่าสภาวะทางลอจิกและตัวเลข

4.หน่วยความจำสำหรับเก็บโปรแกรมคำสั่งของผู้ใช้ (User Program Memory)



รูปที่ 2.5 หน่วยความจำสำหรับผู้ใช้

หน่วยความจำของผู้ใช้จะแตกต่างกันตามขนาดของ PLC/PC ใน PLC หรือ PC ขนาดใหญ่ ขนาดของหน่วยความจำดังกล่าวจะสามารถเปลี่ยนแปลงขนาดได้ตามต้องการ การเลือกใช้ PLC/PC จึงต้องคำนึงถึงขีดความสามารถและขีดจำกัดต่าง ๆ ด้วยประกอบด้วย เช่น ขนาดโปรแกรมคำสั่งของผู้ใช้ จำนวนอินพุท/เอาท์พุท ที่จะขยายได้สูงสุด ขนาดของหน่วยความจำภายในที่ใช้เก็บข้อมูล และฟังก์ชันพิเศษต่าง ๆ เช่น รีเลย์ภายใน ตัวตั้งเวลา ตัวนับ

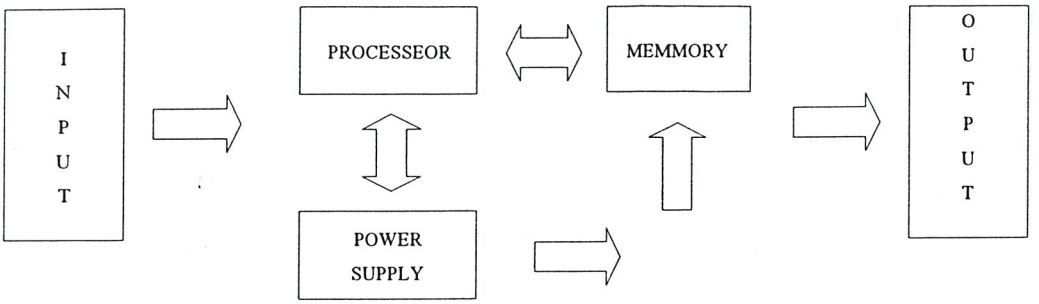
หน่วยจ่ายกำลัง

หน่วยจ่ายกำลังจะทำหน้าที่จ่ายและรักษาระดับแรงดันไฟฟ้ากระแสตรง (D.C. Voltage) ให้กับหน่วยประมวลผล หน่วยความจำ และหน่วยอินพุท/เอาท์พุท ตามความต้องการ และทำหน้าที่เตือนให้หน่วยประมวลผลทราบเมื่อเกิดปัญหา

หน่วยอินพุท/เอาท์พุท

หน่วยอินพุททำหน้าที่เชื่อมต่อระหว่าง CPU กับอุปกรณ์ภายนอกโดยรับค่าสภาวะ หรือปริมาณทางกายภาพต่าง ๆ จากอุปกรณ์ตรวจวัด (Sensor) ของเครื่องจักรหรือกระบวนการ เช่น ลิมิตสวิทช์ (Limit Switch) พร็อกซิมิตี้สวิทช์ (Proximity Switch) ตำแหน่ง อุณหภูมิ ระดับ แรงดัน กระแสไฟ และอื่น ๆ ส่งไปยัง CPU เพื่อประมวลผลตามโปรแกรมคำสั่งของผู้ใช้

หน่วยเอาท์พุททำหน้าที่รับค่าสภาวะหรือคำสั่งควบคุมที่ได้จาก CPU เพื่อส่งไปควบคุมอุปกรณ์ภายในเครื่องจักรหรือกระบวนการเช่น วาล์ว (Valve) มอเตอร์ (Motor) ปั๊ม (Pump) และอื่น ๆ ให้ทำงานตามสภาวะหรือคำสั่งที่ CPU ต้องการ



รูปที่ 2.6 บล็อคไดอะแกรมของหน่วยอินพุท/เอาท์พุท

ข้อตกลงในการติดต่อสื่อสาร (โปรโตคอล : Protocols)

ในการสื่อสารข้อมูลจะต้องมีกฎหรือข้อกำหนดในการสื่อสารข้อมูล หรือที่นิยมเรียกว่าโปรโตคอล (Protocols) ซึ่งเป็นส่วนที่กำหนดมาตรฐานในการควบคุมและจัดการระบบการสื่อสารข้อมูล

สำหรับรายละเอียดที่กล่าวในหัวข้อนี้จะเกี่ยวข้องเฉพาะในส่วนของโปรโตคอลการควบคุมการเชื่อมโยงข้อมูล (Data Link Control Protocols หรือ DLCP) ซึ่งจัดการในส่วนของขั้นตอนและหลักการต่างๆ คือโครงสร้างและรายละเอียดของข้อมูล วิธีในการสื่อสารข้อมูลการตรวจสอบแก้ไขได้ไขความผิดพลาดของข้อมูล และขบวนการในการควบคุมการติดต่อสื่อสาร โดย DLCP แบ่งได้ตามโครงสร้างของข้อมูล 2 แบบ คือ Byte-Oriented Protocols

1.ไบท์โอเรียนโปรโตคอล (Byte Oriented Protocols)

โปรโตคอลแบบนี้เป็รโปรโตคอลที่การสื่อสารข้อมูลและการควบคุมการทำงานจะทำโดยใช้ลักษณะข้อมูลที่เป็นตัวอักษร (Character) หรือไบท์ (Byte) หรืออาจเรียกว่า Character Oriented Protocls

1. อะซิงโครนัสโปรโตคอล (Asynchronous proyocols)

โปรโตคอลในการส่งข้อมูลนี้จะใช้การสื่อสารข้อมูลแบบ Half-Duplex ที่มีลักษณะการสื่อสารข้อมูลแบบอะซิงโครนัส ซึ่งเป็นการสื่อสารข้อมูลแบบพื้นฐานที่ใช้งานมาเป็นเวลานานแล้ว จึงมีรายละเอียดและขั้นตอนในการสื่อสารข้อมูลที่ทำให้มีโอกาสเกิดความผิดพลาดได้น้อย และยังมีข้อดีที่การสื่อสารข้อมูลแบบนี้มีในโครงการและการทำงานที่ง่าย อุปกรณ์ที่ใช้ในการสื่อสารข้อมูลก็ไม่สลับซับซ้อน และมีราคาถูก โปรโตคอลแบบนี้จึงเหมาะสมสำหรับใช้ในระบบขนานเล็ก

2. ไบนารี่ซิงโครนัสโปรโตคอล (Binary synchronous protocols)

โปรโตคอลแบบนี้จะมีลักษณะการทำงานที่ใช้งานข้อมูลเป็นลักษณะไบท์ และยังคงใช้การสื่อสารข้อมูลแบบซิงโครนัส มีรายละเอียดและขั้นตอนในการสื่อสารข้อมูลที่ให้ความน่าเชื่อถือมากกว่า อีกทั้งยังสามารถใช้อัตราเร็วในการสื่อสารข้อมูลที่สูงกว่าโดยตัวอย่างของการสื่อสารข้อมูลแบบนี้ที่ได้กำหนดเป็นมาตรฐานแล้วคือ การสื่อสารข้อมูลตามมาตรฐาน BSC (Binary Synchronous Communications) ซึ่งเป็นโปรโตคอลที่มีลักษณะของข้อมูลแบบไบท์ที่ได้รับความนิยมนำไปใช้งาน

2.บิทโอเรียนท์โปรโตคอล (Bit-Oriented protocols)

โปรโตคอลแบบนี้เป็นโปรโตคอลที่การสื่อสารข้อมูล และการควบคุมการทำงานจะทำโดยใช้ลักษณะข้อมูลที่เป็นบิท (bit) โดยมีตัวอย่างของการสื่อสารข้อมูลในลักษณะนี้ที่มีการกำหนดขึ้นเป็นมาตรฐานแล้วคือ HDLC (High-level Data Link Control) โดยมีโครงสร้างของข้อมูลแบบซิงโครนัส เช่นเดียวกับ BSC แต่ต่างกันที่มีลักษณะของข้อมูลเป็นแบบบิท ซึ่งโปรโตคอลแบบนี้มีข้อดีที่สามารถสื่อสารข้อมูลแบบ full-duplex ได้ทำให้การสื่อสารข้อมูลได้รวดเร็วกว่า แต่โปรโตคอลแบบนี้ก็มีรายละเอียดและโครงสร้างในการสื่อสารข้อมูลที่สลับซับซ้อนมาก ทำให้การควบคุมการทำงานทำได้ยากและต้องใช้อุปกรณ์ที่มีราคาสูง จึงไม่เหมาะที่จะนำไปใช้งานกับระบบขนาดเล็ก

2.1 แพ็กเก็ท (Packet of Information)

รูปแบบของแพ็กเก็ทในระบบโครงข่ายจะประกอบด้วยส่วนต่างๆ ดังนี้

1. Header จะประกอบด้วย
   - Peramble or start of packet indicator เป็นส่วนเริ่มแรกของแพ็กเก็ท และในบางระบบอาจใช้ในการซิงค์กับสัญญาณนาฬิกาของตัวส่งและตัวรับด้วย
   - Control information ส่วนนี้เป็นข้อมูลที่บอกถึงวัตถุประสงค์ของแพ็กเก็ทนั้นว่าใช้ทำอะไร เช่น เพื่อการจัดการระบบเพื่อดู Status ของ mode หรืออื่นๆ

   นอกจากส่วนต่างๆเหล่านี้แล้ว ในส่วน Header อาจจะมีส่วนที่เป็น Sequencial Number เป็นส่วนที่บอกให้ทราบถึงลำดับของแพ็กเก็ท ในกรณีที่ข้อมูลมีความยาวหลายแพ็กเก็ท
2. Information
   - Data field เป็นส่วนของข้อมูลจริงที่ต้องการจะส่ง
3. Tailer
   - Frame Check Sequence (FCS) เป็นส่วนที่ใช้ตรวจสอบความถูกต้องของข้อมูลซึ่งอาจเป็น parity bit, Check sum หรือ CRC เป็นต้น
   - End of Packet Indicator เป็นส่วนที่บอกให้ทราบว่าสิ้นสุดของข้อมูลแล้ว

PREAMBLE	DA	SA	CONTROL	INFFORMATION	FCS	STOP
HEADER			INFORMATION		TAILER	

รูปที่ 2.7 แสดงโครงสร้างของแพ็กเก็ท

การควบคุมความผิดพลาดในการส่งข้อมูล (Error Control)

การที่วงจรส่งข้อมูลขาดชั่วขณะ และผลกระทบต่อสัญญาณรบกวน (Noise) ทำให้ระดับแรงดันไฟฟ้าลดลงเป็นผลให้เกิดความผิดพลาดในการส่ง ดังนั้นต้องทำการค้นหาความผิดพลาดที่เกิดขึ้นระหว่างการส่งข้อมูลและแก้ไขความผิดพลาดให้ถูกต้อง

1. วิธีการเพิ่มบิตเข้าไปที่ข้อมูล (Parity Bit) เพื่อตรวจสอบความผิดพลาดมี 2 วิธี

1.1 วิธี Parity แนวนอน

จะเพิ่ม 1 บิต เข้าไปที่แต่ละตัวอักษรที่จะส่ง โดยจะกำหนดว่าเป็นการตรวจสอบแบบ Odd หรือ Even Number แล้วผลรวมของข้อมูลจะเป็น Odd Number ที่ด้านรับจะทำการตรวจสอบว่าผลรวมของข้อมูลจะเป็น Odd Number หรือ Even Number

1.2 วิธี Parity แนวตั้ง

จะตรวจสอบความผิดพลาดโดยการเพิ่ม 1 บิตเข้าไปที่ในแต่ละบล็อก

01101101	1	
01100110	0	
10111010	1	
10010111	1	
00101001	1	Parity แนวนอน
00010110	1	
10101110	1	
11110010	1	
01000101		
Parity แนวตั้ง		

รูปที่ 2.8 แสดงรูปแบบการตรวจสอบด้วย Parity Bite

2.วิธี Patrol Diffuse Inspection หรือการใช้ FCS : Frame Check Sequence

สำหรับข้อมูลอันหนึ่งจะสร้าง Error Inspection Sign (CRC Sign) ขนาด 2 Bite ที่คำนวณได้บนพื้นฐานของกฎที่กำหนดไว้ แล้วเพิ่มเครื่องหมายไปที่ข้อมูลอันนั้น ที่ทางด้านรับจะตรวจสอบข้อมูลโดย Inspection Sign ได้กำหนดไว้โดยสูตรการคำนวณที่เหมือนกัน (Blast Error ความผิดพลาดของข้อมูลที่ต่อเนื่อง) ก็สามารถตรวจสอบได้ ทำให้มีความน่าเชื่อถือสูง

ระบบการสื่อสารข้อมูล

การรับส่งเป็นรูปแบบหนึ่งของการสื่อสาร ซึ่งข้อมูลที่รับส่งกันนั้นเป็นสัญญาณไฟฟ้าในรูปเลขฐานสอง “0” หรือ “1” หากพิจารณาตามทิศทางการส่งข้อมูลภายในสายส่งแล้ว สามารถแบ่งการส่งข้อมูลออกได้เป็น 3 ชนิดคือ

1. การส่งแบบทิศทางเดียว (one-way transmission หรือ simplex transmission) คือข้อมูลจะไหลได้ทิศทางเดียว
2. การส่งแบบทิศทางใดทิศทางหนึ่ง (Either-way transmission หรือ half-duplex transmission) คือข้อมูลจะไหลได้สองทิศทาง แต่แต้องผลัดกันรับผลัดกันส่ง

3. การส่งแบบสองทิศทาง (both-way transmission หรือ full-duplex transmission) คือข้อมูลจะไหลได้สองทิศทางพร้อม ๆ กัน

ในยุคต้นๆ การสื่อสารมักจะเป็นแบบ Half-duplex อาจเป็นเพราะข้อจำกัดของข้อกำหนด (protocal) และฮาร์ดแวร์ ในการสื่อสารที่มีอยู่ในเวลานั้นในการสื่อสารแบบ half-duplex ข้อมูลจะสามารถเดินทางได้เพียงทิศทางเดียวในช่วงเวลาหนึ่ง ๆ ยกตัวอย่างเช่นในสายโทรศัพท์แบบมีสายสองเส้น เมื่อคอมพิวเตอร์เครื่องหนึ่งส่งข้อมูลออกมา อีกเครื่องหนึ่งก็จะทำการรับข้อมูล หรือกลับกัน ซอฟต์แวร์ที่ควบคุมการสื่อสารแบบ half-duplex จะทำงานร่วมกับฮาร์ดแวร์ในการกำหนดว่าปลายทางด้านไหนจะทำหน้าที่เป็นตัวส่งหรือเป็นตัวรับข้อมูล

ข้อกำหนดการสื่อสารแบบ Half-duplex อนุญาตให้มีการส่งและรับข้อมูลโดยไม่สนใจว่าด้านไหนกำลังรับและด้านไหนกำลังส่งในช่วงเวลาหนึ่ง ๆ เพื่อที่จะทำให้สามารถติดต่อสื่อสารแบบ half-duplex ได้ จึงจำเป็นต้องใช้สายโทรศัพท์แบบมีสายสัญญาณ 2 คู่โดยที่สัญญาณคู่ที่ใช่ส่งข้อมูลจากทางด้านต้นทางจะต่อเข้ากับสายสัญญาณที่ใช้รับข้อมูลที่อยู่ด้านปลายทางและสายสัญญาณคู่ที่ใช้ส่งข้อมูลที่ด้านปลายทาง

วิธีการส่งสัญญาณแบบ Half-duplex ที่ใช้สายตัวนำ 2 เส้น และแบบ full-duplex ที่ใช้สายตัวนำ 4 เส้น ก็ยังคงมีอยู่แม้จนกระทั่งทุกวันนี้ แต่ปัจจุบันนี้การสื่อสารแบบ full-duplex โดยใช้สายสัญญาณเพียง 2 เส้น จะใช้เทคนิคการรวมสัญญาณทั้งทางด้านความถี่และเฟส (frequency-and phase modulation) ที่พัฒนาขึ้นมาและอยู่ในโมเด็มสมัยใหม่

และถ้าพิจารณารูปแบบของข้อมูลที่ส่ง จะแยกได้เป็น 2 ชนิดคือ

1. แบบซิงโครนัส (synchronous transmission)
2. แบบอะซิงโครนัส (asynchronous transmission)

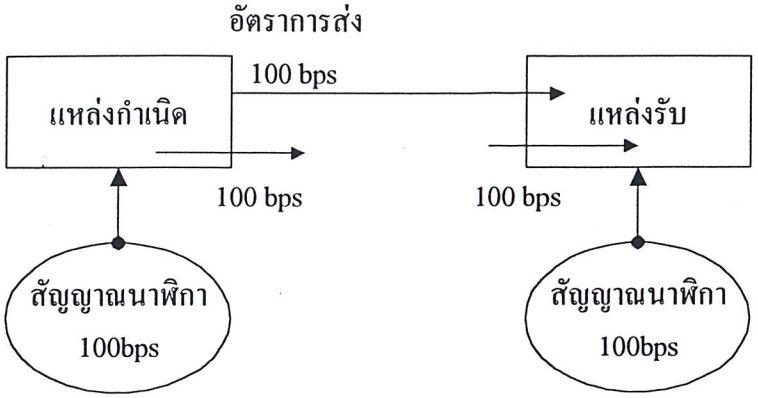
การส่งแบบซิงโครนัส

เราแบ่งได้เป็น 2 แบบ คือ แบบซิงโครนัสบิตและแบบซิงโครนัสอักขระ

ซิงโครนัสบิต

เป็นการับส่งที่ด้านรับต้องรู้ว่าจะรับบิตแรกจากสายส่งนั้นเมื่อใด และหลังจากรับมาแล้ว จะรับบิตที่ 2 , 3 ,...เมื่อใดซึ่งสามารถกระทำโดยเพิ่มสัญญาณนาฬิกา (clock) เข้าไปที่จุดปลายทางของระบบทั้งสองด้านดังรูปที่ 10

ซึ่งในบางระบบอาจใช้วิธีส่งสัญญาณนาฬิกาจากด้านส่งไปทางด้านรับด้วยซึ่งทางด้านรับจะนำสัญญาณที่ได้รับมาเป็นตัวขับสัญญาณนาฬิกาของตัวมันเองเพื่อให้สัญญาณนาฬิกาทางด้านรับและส่งเท่ากันพอดี

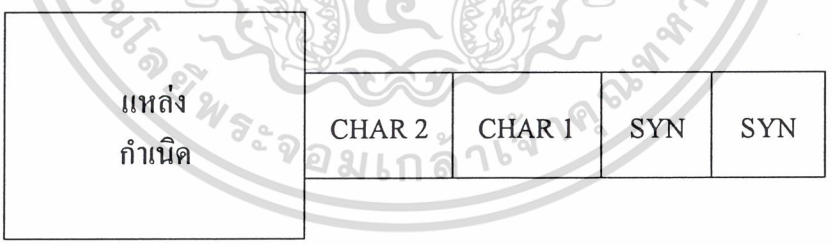


รูปที่ 2.9 การให้สัญญาณนาฬิกาเพื่อการรับส่งแบบซิงโครนัสบิต

ซิงโครนัสอักขระ

ในการรับส่งข้อมูลตามสายนั้นแม้ว่าจะมีการจัดการกับบิตได้ดีแล้วก็ตามแต่ก็ยังมีปัญหาอีก คือ การนำเอาบิตของอักขระ (Character) หลาย ๆ ตัวมารวมกันเป็นบล็อก ดังนั้นถึงแม้ว่าบิตต่าง ๆ จะได้รับมาอย่างถูกต้องแล้วก็ตาม เรายังต้องทราบว่ากลุ่มของบิตที่แสดงถึงตัวอักขระต่าง ๆ นั้นเริ่มต้นที่บิตใด เพื่อที่เราจะแก้ปัญหาได้ถ้าเราทราบว่าบิตใดเป็นบิตเริ่มต้นของตัวอักขระ และถ้าหากทราบว่าในตัวอักขระหนึ่งตัวนั้นมีกี่บิต รวมทั้งความเร็วของการสั่งของบิตต่าง ๆ โดยการนับจำนวนบิตที่ได้รับมาตามสายหลังจากทราบบิตแรกก็จะสามารถแยกตัวอักษรออกจากกันได้

เทคนิคการส่งข้อมูลแบบซิงโครนัสนั้น ใช้สำหรับส่งข้อมูลทั้งชุดไปครั้งเดียวโดยในการส่งแบบนี้ช่วงความกว้างของเวลาจะห่างบิตแต่ละบิตจะมีค่าเท่ากัน



รูปที่ 2.10 การส่งแบบซิงโครนัสที่มี SYN หลายตัว

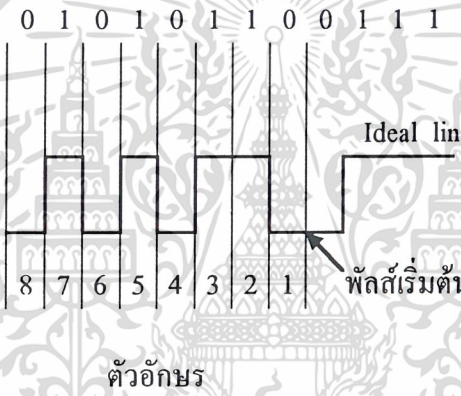
ในรูปที่ 11 แสดงให้เห็นถึงการส่งอักขระต่าง ๆ แบบซิงโครนัส โดยมีการส่งชุดข้อมูลชุดหนึ่งก่อนหน้าการส่งชุดข้อมูลตัวอักขระ เพื่อทำการหาบิตแรกของอักขระตัวแรกให้เป็นไปอย่างถูกต้อง ข้อมูลชุดนี้เรียกว่า SYN Transmission Control Character (TC) ซึ่งประกอบด้วยข้อมูลขนาด 8 บิต เช่น 00010110

(มีพาริตี้เป็นคี่) และทางด้านรับก็จะคอยมองหาชุดของบิตดังกล่าว (looking for SYNC) ซึ่งจะกระทำทึกครั้งเมื่อรับบิตใหม่เข้ามา จนกว่าจนกว่าจะได้รับข้อมูลตัวอักขระที่กล่าวมาแล้วข้างต้น

ในระบบนี้ส่วนมากมักจะมี SYN นำหน้าข้อมูล 3-4 ตัว เพื่อให้แน่ใจได้ว่าชุดของบิตตัวอักขระที่รับเข้าคือข้อมูลที่ถูกต้อง

การส่งแบบอะซิงโครนัส

เมื่อจะทำการส่งแบบอะซิงโครนัสตัวอักขระจะถูกส่งออกไปในเวลาใด ๆ ก็ได้ โดยไม่จำเป็นต้องมีความสัมพันธ์ระหว่างตัวอักขระ หรือจะต้องมีเวลาที่แน่นอนอย่างไร แล้วแต่ตัวอักขระแต่ละตัวอาจจะทิ้งช่วงห่างกันเท่าใดก็ได้ เพราะอักขระแต่ละตัวจะมีบิตที่คอยบอกการเริ่มต้น และการสิ้นสุดของอักขระตัวนั้น (start bit , stop bit) ซึ่งในทางปฏิบัติ ทางด้านรับจะทราบได้ว่ามีบิตเริ่มต้นเข้ามาแล้ว เริ่มด้วยการเปลี่ยนสถานะจาก “1” เป็น “0” หรือ จาก “0” เป็น “1” ก็ได้ (แต่ส่วนใหญ่มักให้เริ่มเปลี่ยนจาก “1” เป็น “0”) ดังแสดงในรูปที่ 12



รูปที่ 2.11 การส่งแบบอะซิงโครนัส

ทางด้านส่งจะเริ่มส่งบิตเริ่มต้นโดยการเปลี่ยนสถานะของค่าปกติที่เป็น “1” เป็น “0” หลังจากนั้นจะส่งบิตข่าวสารสำหรับอักขระหนึ่งตัวออกไป ในขณะที่ด้านรับทราบถึงการเปลี่ยนสถานะจาก “1” เป็น “0” เป็นการบอกจุดเริ่มต้นของการทำงานว่า จะเริ่มภายหลังเวลาหรือเท่ากับครึ่งเวลาของความกว้าง 1 บิต หลังจากนั้นทางด้านรับก็จะสุ่มตัวอย่างสถานะสายทุก ๆ ช่วงเวลา 1 บิต เพื่อหาค่ารหัสของตัวอักขระที่ส่งมาจนกว่าจะครบ 8 บิต (ถ้าเป็นรหัสแอสกี) และบิตถัดไปจะเป็นบิตสิ้นสุด หลังจากนั้นจะเว้นช่วงเวลาไปอีกนานเท่าไรก็ได้ในการเริ่มส่งตัวอักขระตัวใหม่ เพราะบิตเริ่มต้นจะคอยบอกอยู่แล้ว

ด้วยวิธีการส่งแบบมีบิตเริ่มต้นและจบด้วยบิตสิ้นสุดนี้ บางครั้งจึงเรียกการส่งแบบนี้ว่า start-stop transmission บิตสิ้นสุดนั้นจะแตกต่างกันไปในแต่ละระบบ เช่น การส่งรหัสแอสกี บิตสิ้นสุดจะมีขนาดความกว้างเท่ากับความกว้างของข้อมูล 1 หรือ 2 บิต ซึ่งเป็นช่วงเวลาที่ทางด้านรับทำงานหลังจากรับข้อมูลมาครบหนึ่งอักขระ เช่น การพิมพ์ในเครื่องพิมพ์การเจาะรูบนเทปของเครื่องเจาะรูเทป เป็นต้น ช่วงเวลาของบิตสิ้นสุดนี้ขึ้นอยู่กับการทำงานของเครื่องจักรกลซึ่งมีความล่าช้า โดยอาจต้องมีเวลาของ

บิตสิ้นสุดเท่ากับ 2 บิต แต่อุปกรณ์ประเภทเครื่องจักรไฟฟ้านั้น จะมีความเร็วในการทำงานมาก จึงอาจจะมีความกว้างของบิตดังกล่าวเพียง 1 บิตเท่านั้น

ในการส่งแบบอะซิงโครนัสนี้ เนื่องจากอักขระทุกตัวที่ส่งออกมาต่างเป็นอิสระต่อกัน และทุกอักขระจะมีบิตบอกการเริ่มต้นและสิ้นสุด ดังนั้นหากเกิดสัญญาณรบกวน จะก่อให้เกิดความผิดพลาดจนทำลายความถูกต้องของตัวอักขระ ความผิดพลาดนี้ถ้าเกิดขึ้นในการส่งแบบซิงโครนัสอาจจะทำลายข่าวสารหมดทั้งบล็อกก็ได้เพราะข่าวสารทั้งบล็อก (อักขระหลายตัว) จะมีความสัมพันธ์เพียงครั้งเดียว

การส่งข้อมูลในระบบดิจิตอลมีสองลักษณะคือ ส่งแบบอนุกรม (serial transmission) และแบบขนาน (parallel transmission)

ถ้าต้องการส่งอักษร "K" ในรหัส ASCII จะหมายถึงการส่งกลุ่มของข้อมูล "1001011" ไปในสายที่ขนานกันหลาย ๆ เส้น โดยแต่ละเส้นจะส่งข้อมูลแต่ละบิตในเวลาเดียวกัน ดังในรูปที่ 13(ก) สำหับตัวอย่างนี้สายเส้นที่ 8 เป็นสายสำหรับสัญญาณพาริตี้ (parity) ใช้ในการตรวจสอบความถูกต้องของข้อมูล และสายสัญญาฯที่สำคัญที่สุดของการส่งข้อมูลแบบขนานคือ สโตรบ (strobe) หรือสัญญาณนาฬิกา (clock) เป็นสัญญาณที่ใช้แจ้งให้ฝ่ายรับข้อมูล เมื่อข้อมูลทุกบิตพร้อมที่จะส่ง หรือแจ้งว่าฝ่ายรับเมื่อข้อมูลถูกส่งทุกบิตแล้วให้เตรียมอ่านข้อมูลนั้น

ในรูปที่ 13 (ข) เป็นการส่งข้อมูลตัวอักษร "K" ในรหัส ASCII โดยใช้การส่งแบบอนุกรม ซึ่งมีลักษณะการส่งแตกต่างกับแบบขนานอย่างสิ้นเชิง

การตรวจสองพาริตี้

วิธีตรวจสอบพาริตี้เป็นการตรวจสอบข้อมูลทางดิจิตอลอย่างง่ายที่นิยมใช้กันมาก การเพิ่มพาริตี้อีกหนึ่งบิตเข้าไปรวมกับข้อมูลที่จะส่งเพื่อให้ข้อมูลกลุ่มนั้นมีบิตที่เป็น "1" เป็นจำนวนคู่ เรียกว่า อีเวนพาริตี้ (Even parity) หรือจำนวนคี่เรียกว่า อ๊อดพาริตี้ (Odd parity) ดังตัวอย่างเป็นการส่งข้อมูลขนาด 7 บิต อีเวนพาริตี้ของอักษร "K" ในรหัส ASCII คือ "11010010" บิตสุดท้ายที่ส่งนี้จะเป็นพาริตี้บิต และทางฝ่ายรับข้อมูลจะตรวจดูว่าข้อมูลที่ส่งมานี้ยังเป็นอีเวนหรือมีจำนวนบิตที่เป็น "1" เป็นจำนวนคู่อยู่หรือไม่ถ้าไม่เป็น แสดงว่าข้อมูลนั้นผิดพลาดจะทำการส่งใหม่

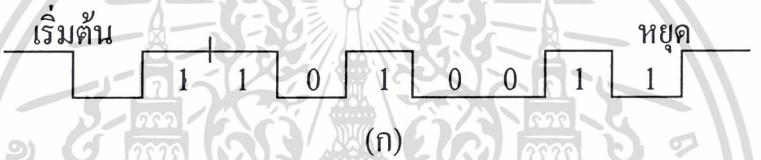
การสื่อสารข้อมูลแบบขนาน

การสื่อสารข้อมูลแบบขนานใช้กันในระบบเครื่องคอมพิวเตอร์ เช่น ระบบการติดต่อกับหน่วยความจำ ระบบการติดต่อระหว่างอินพุตเอาต์พุตพอร์ตต่าง ๆ และการติดต่อระหว่างพอร์ตขนาน ดังรูปที่ 14 (ก) สัญญาณ D0-D7 (ขา 2-9) จะเป็นข้อมูลโดยมีขา 2 เป็นบิตที่มีนัยสำคัญต่ำสุด (LSB) และขา 9 เป็นบิตที่มีนัยสำคัญสูงสุด (MSB) ขา 1 จะเป็นสัญญาณสโตรบ สำหรับบอกให้เครื่องพิมพ์รับข้อมูลเมื่อข้อมูลปรากฏบนขาข้อมูลในช่วงเวลา T1-T3 ดังแสดงในรูปที่ 14 (ข) สัญญาณข้อมูลที่ออกไปจะไม่มีการตรวจสอบแต่จะใช้สัญญาณ ACK ที่ขา 10 สำหรับส่งสัญญาณจากเครื่องพิมพ์ไปยังพอร์ตขนานโดยจะทำงานที่ระดับลอจิก "0" เมื่อเครื่องพิมพ์ได้อ่านข้อมูลไปแล้ว จะส่งสัญญาณไปยังวงจรควบคุมพอร์ตขนาน ให้รู้ว่าเครื่องพิมพ์พร้อมที่จะรับข้อมูลชุดใหม่แล้วส่วนระยะเวลาดูจากรูป 2 (ข) จะเป็นช่วงเวลาที่

โปรแกรมในเครื่องพิมพ์ทำการอ่านข้อมูลจากพอร์ตและทำการเก็บลงในหน่วยความจำจากนั้นเป็นสัญญาณ BUSY ที่ขา 11 จะทำงานที่ระดับลอจิก "1" โดยจะส่งสัญญาณให้วงจรควบคุมพอร์ตขนานเครื่องพิมพ์ไม่สามารถรับข้อมูลได้ ซึ่งอาจเกิดจากสาเหตุดังต่อไปนี้

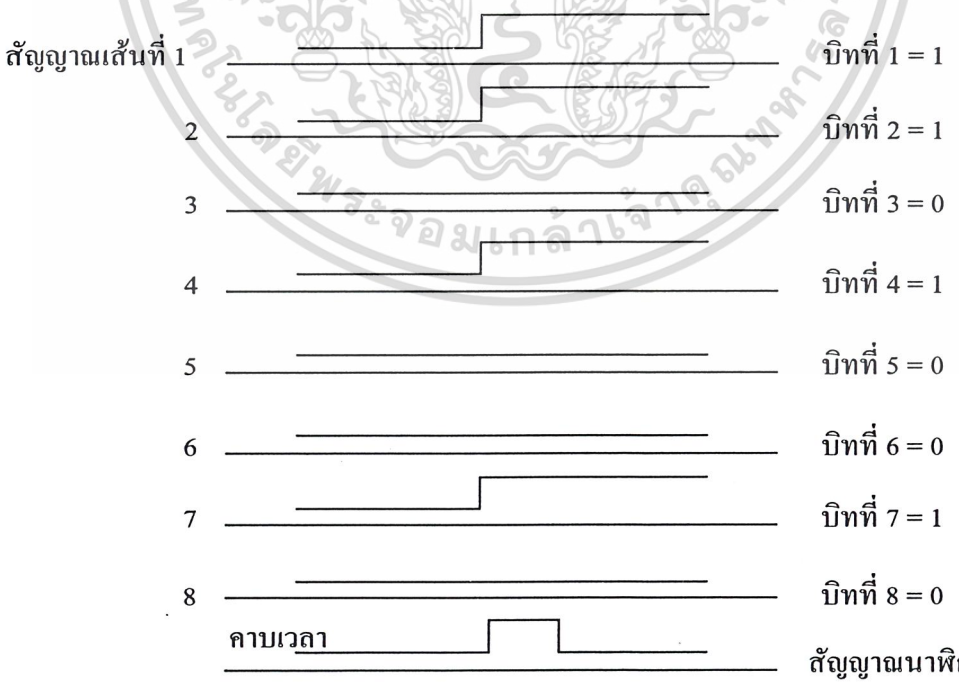
1. ข้อมูลกำลังเข้าเครื่องพิมพ์
2. เครื่องพิมพ์กำลังส่งข้อมูลเข้าไปเก็บในหน่วยความจำ
3. เกิดจาก offline state คือเครื่องพิมพ์อยู่ในสภาวะไม่สามารถสื่อสารข้อมูลได้ (ไฟแสดง online บนเครื่องพิมพ์ไม่ติด)
4. เกิดจากความผิดพลาดในการทำงานของเครื่องพิมพ์

สำหรับสัญญาณอื่น ๆ ที่เหลือบนพอร์ตขนานจะเป็นสัญญาณบอกสถานะของเครื่องพิมพ์ ข้อดีของระบบการสื่อสารแบบขนาน คือสามารถส่งข้อมูลได้รวดเร็วซึ่งมักใช้สื่อสารในระยะใกล้ ๆ เพราะถ้าระยะไกล ๆ การใช้สายไฟจำนวนมากจะทำให้ราคาในการผลิตสูงขึ้น และเกิดการสูญเสียสัญญาณในสายได้ง่าย ดังนั้นถ้าต้องการสื่อสารในระยะไกล ๆ จึงมักนิยมใช้การสื่อสารแบบอนุกรม



(ก)

รูปที่ 2.12 (ก) แผนภูมิเวลาของการส่งสัญญาณข้อมูลแบบอนุกรม

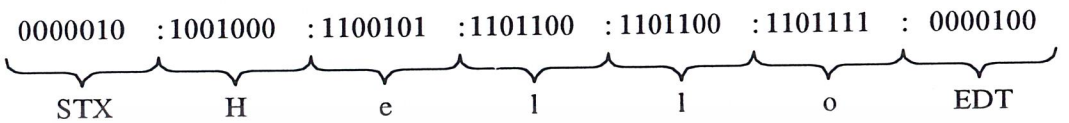


(ข)

รูปที่ 2.12 (ข) แผนภูมิเวลาของการส่งสัญญาณข้อมูลแบบขนาน

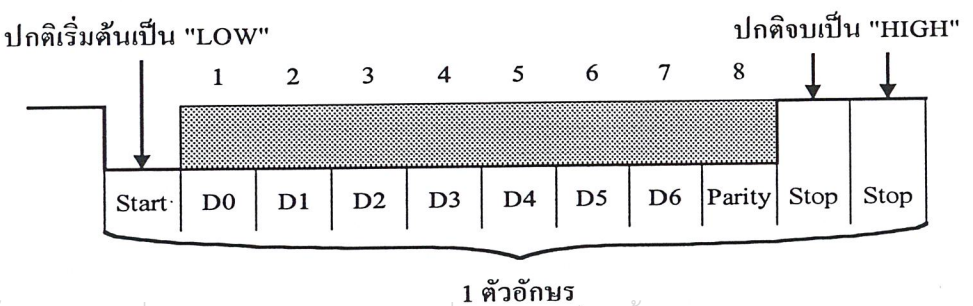
การสื่อสารข้อมูลแบบอนุกรม

การสื่อสารข้อมูลแบบอนุกรมสามารถส่งได้สองลักษณะคือ ส่งแบบซิงโครนัส (Synchronously) และอะซิงโครนัส (Asynchronously) การส่งแบบซิงโครนัสข้อมูลจะส่งเป็นกลุ่มบล็อก (block) โดยมีรหัสพิเศษที่เริ่มต้นของบล็อก และที่จุดสิ้นสุดของบล็อกค่าเวลาในการส่งแต่ละบิตจะคงที่ โดยฝ่ายรับจะทำการแยกข้อมูลแต่ละอักขระออก เช่น ระบบส่ง 7 บิต รหัส ASCII ข้อมูลคือ “Hello” โดยใช้รหัส STX (Start of Text) ส่งก่อน และ EOT (End of transmission) ปิดท้าย

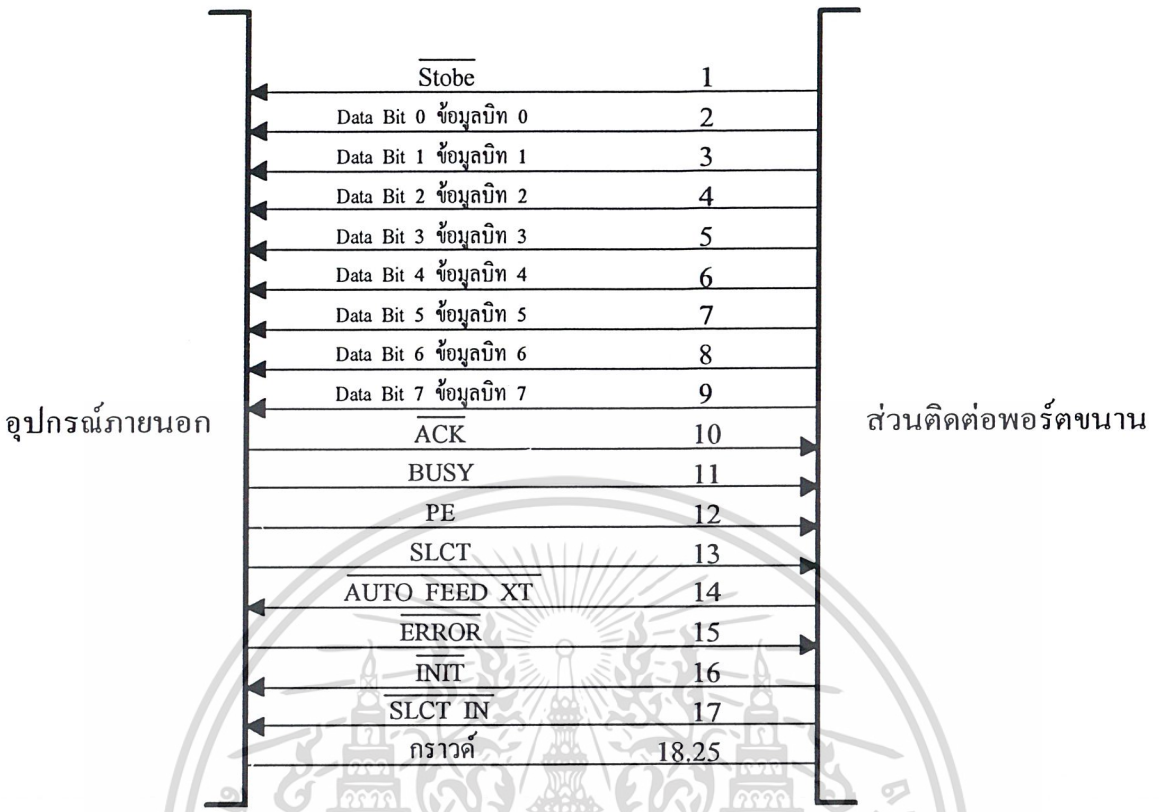


ทางฝ่ายรับก็จะทำการนับทีละ 7 บิต และแยกอักขระแต่ละตัวออกจากกัน สำหรับการส่งจริง ๆ นั้น รหัสพิเศษ อาจใช้ชุดของตัวอักษรหลาย ๆ ตัวประกอบกันก็ได้ เพื่อจะทำให้การแยกระหว่างข้อมูลและกรอบของข้อมูลเป็นไปอย่างถูกต้อง

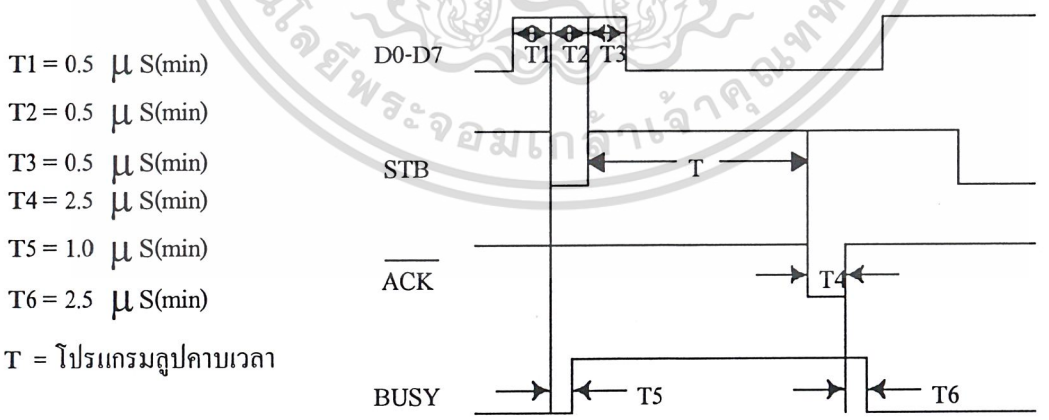
การส่งแบบอะซิงโครนัสเป็นการส่งข้อมูลโดยการเพิ่มบิตเริ่มต้น (start bit) และบิตสิ้นสุด (stop bit) 1-2 บิต เข้าไปในกลุ่มข้อมูลของแต่ละอักษร ซึ่งเรียกว่ากรอบของข้อมูล (Framing) ซึ่งจะทำให้สามารถส่งอักษรในเวลาใดก็ได้โดยทางฝ่ายรับจะทำการตรวจการเปลี่ยนจาก “1” เป็น “0” (บิตเริ่มต้น) ดังแสดงในรูปที่ 3 และต่อไปจะเป็นการสุ่มตัวอย่างในการอ่านข้อมูล โดยเริ่มทำการรีเซตวงจรสำหรับกำเนิดสัญญาณนาฬิกาที่ใช้ในการสุ่ม (Sampling) โดยจะเก็บข้อมูลในแต่ละคาบเวลา (timing) ของแต่ละบิตที่ส่งจะเท่ากับระยะห่างในการอ่านข้อมูลของทางฝ่ายรับในแต่ละบิต ดังรูปที่ 4 (ก) โดยจะทำให้การเก็บข้อมูลที่เกิดขึ้นที่กึ่งกลางของแต่ละบิต แต่ความเป็นจริงจะไม่สามารถปรับให้สัญญาณนาฬิกาหรือเวลาในการสุ่มข้อมูลของตัวรับกับตัวส่งให้เท่ากันพอดีได้ จะเกิดการเลื่อนดังรูปที่ 4 (ค) เกิดจากฝ่ายรับข้อมูลผิดพลาดเนื่องจากเวลาในการสุ่มข้อมูลช้าเกินไป เป็นที่สังเกตอีกอย่างหนึ่งว่า การใช้ข้อมูลที่มีจำนวนบิตมากจะทำให้ใช้เวลาในการสุ่มข้อมูลทั้งสองฝ่ายต่างกันเพียงเล็กน้อย ซึ่งจะทำให้ระบบบางระบบต้องใช้บิตของข้อมูลน้อยลง เช่น ระบบที่ใช้เครื่องจักรไฟฟ้าพวกมอเตอร์ ซึ่งการควบคุมความเร็วของมอเตอร์จะยากที่จะทำให้คงที่



รูปที่ 2.13 แสดงลักษณะของการส่งข้อมูลแบบอนุกรม (อะซิงโครนัส)



รูปที่ 2.14 (ก) ลักษณะพอร์ตขนานและขาสัญญาณต่างๆเมื่อเชื่อมต่อกับอุปกรณ์ภายนอก



(ข) ช่วงเวลาของการส่งข้อมูลแบบขนาน

RS – 232 มาตรฐานเพื่อการสื่อสาร

มาตรฐาน RS–232C ประกาศ และได้กำหนดการอินเตอร์เฟสระหว่างอุปกรณ์เทอร์มินัลของข้อมูล(Data Terminal Equipment , DTE) กับอุปกรณ์ส่วนปลายของวงจรข้อมูล (Data Circuit Terminating Equipment , DCE) อุปกรณ์ DTE ตามปกติแล้วจะเป็นอุปกรณ์เทอร์มินัลที่พูดไม่ได้ (dumb terminal) , เป็นอุปกรณ์ที่มีความฉลาด เช่นไมโครคอนโทรเลอร์หรือเครื่องคอมพิวเตอร์ส่วนบุคคลที่มีความฉลาดในการสร้างบิตข้อมูลเป็นแบบอนุกรมได้ อุปกรณ์ DCE จะรับบิตข้อมูลที่อุปกรณ์ DTE ส่งออกมา ผ่านทางการอินเตอร์เฟสแบบ RS-232 และแปลงให้อยู่ในรูปที่เหมาะสมสำหรับถ่ายทอดผ่านตัวกลางการสื่อสารจากระยะไกลเช่นผ่านทางสายโทรศัพท์

ถ้าติดตามมาตรฐาน RS-232C ต่อไปจะพบว่าในทางกายภาพแล้วขั้วต่อพอร์ตของอุปกรณ์ DTE จะเป็นข้อต่อตัวผู้และขั้วต่อพอร์ตของอุปกรณ์ DCE จะเป็นข้อต่อตัวเมีย พอร์ตอนุกรมของคอมพิวเตอร์ส่วนบุคคลปกติแล้วจะเป็นอุปกรณ์ DTE และพอร์ตของโมเด็มก็มักจะมีโครงสร้างเป็นอุปกรณ์ DCE

พอร์ตอนุกรมของเครื่องพีซีจะเป็นตัวผู้ไม่ว่าจะเป็นขั้วต่อแบบ 9 ขาหรือ 25 ขา ส่วนข้อต่อของโมเด็มส่วนมากแล้วจะเป็นขั้วต่อแบบตัวเมีย 25 ขา แม้ว่าขั้วต่อแบบ 9 ขาจะไม่ได้เป็นมาตรฐาน RS-232 ของ EIA แต่ในปัจจุบันนี้ก็มีการใช้อยู่ทั่วไป ในการอินเตอร์เฟสแบบ RS-232 การอินเตอร์เฟสที่ใช้ขั้วต่อแบบ 9 ขา พบจำหน่ายเป็นครั้งแรกในเครื่องพีซีรุ่น AT ในตอนต้นทศวรรษ 1980

มาตรฐาน RS-232C ยังกำหนดคุณลักษณะสำหรับการสื่อสารผ่านทางสายส่งสัญญาณเสียงด้วยอัตราความเร็วสูงถึง 9600 bps ซึ่งช่วยให้สามารถมีการสื่อสารแบบอะซิงโครนัสจากจุดหนึ่งไปยังอีกจุดหนึ่งผ่านทางเครือข่ายโทรศัพท์สาธารณะในขณะนั้นได้

มาตรฐาน RS-232C กำหนดวงจร 21 วงจรในการอินเตอร์เฟสนี้ตาราง 1 คือสรุปคำบรรยายหน้าที่ของขาในมาตรฐาน RS-232 A , B และ C

ในระหว่างการส่งข้อมูล สภาวะมีข้อมูล (mark condition) บ่งชี้ด้วยสถานะ “1” ในระบบเลขฐานสอง และสภาวะช่องว่าง (space condition) บ่งชี้ด้วยสถานะ “0” ในระบบเลขฐานสอง

สำหรับวงจรกำหนดจังหวะเวลาและควบคุมการแลกเปลี่ยนข้อมูลแล้วการทำงานของมันจะเป็น “on” เมื่อแรงดันไฟฟ้าเป็นบวกมากกว่า +3 โวลท์ และจะเป็น “off” เมื่อแรงดันไฟฟ้าเป็นลบมากกว่า –3 โวลท์ เมื่อเทียบกับกราวด์ การทำงานนี้จะไม่สามารถกำหนดได้ถ้าแรงดันไฟฟ้าอยู่ในช่วงของการเปลี่ยนแปลงระหว่าง –3 ถึง +3 โวลท์ เมื่อเทียบกราวด์

คำว่า “mark” และ “space” จะใช้อยู่ทั่วไปในเอกสารกำหนดคุณลักษณะ เพื่อบรรยายสภาวะของสายนำข้อมูลหรือสายสัญญาณควบคุมในระบบ RS-232 รูปที่ 16 กำหนดค่าแรงดันต่ำสุดและสูงสุดของข้อกำหนด RS-232D และ V.28 และข้อกำหนดที่จุดปลายของการอินเตอร์เฟสบางข้อ

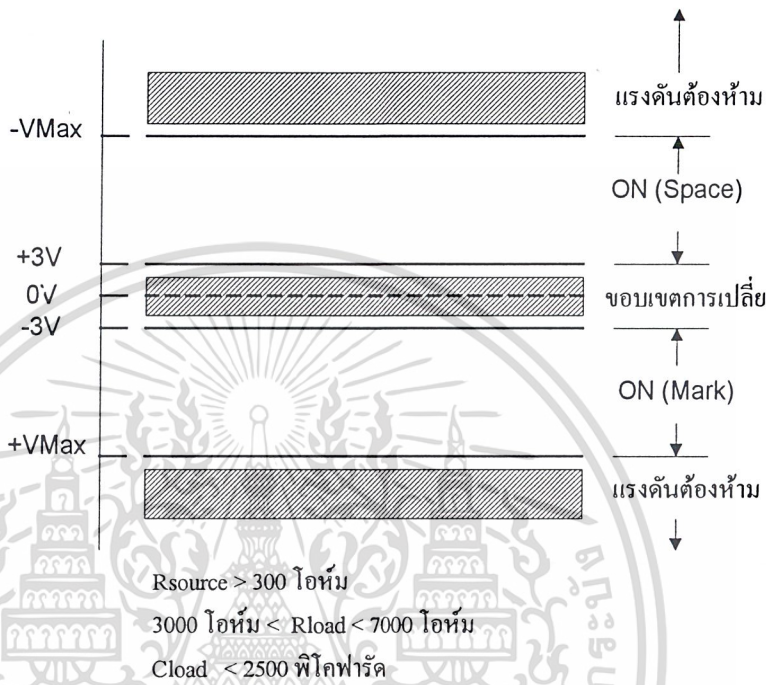
ตารางที่ 2.1 แสดงรายการคุณลักษณะเฉพาะสำหรับการอินเตอร์เฟสแบบ RS-232

หมายเลขขาสัญญาณ	ชื่อของสายสัญญาณ	ทิศทางของสัญญาณ
1	Positive Ground	N.A.
2	Transmitted Data	To DCE
3	Received Data	To DCE
4	Request To Send	To DCE
5	Clear To Send	To DCE
6	Data Set Ready	To DCE
7	Signal Ground	N.A.
8	Received Line Signal Detector (RS-232C) Data Carrier Detect (RS-232A/B)	To DTE
11	Select Standby	To DCE
12	Secondary Receive Line Signal Detector	To DTE
13	Secondary Clear To Send	To DTE
14	Secondary Transmitted Data	To DCE
14	New Sync	To DCE
15	Transmitter Signal Element Timing	To DTE
16	Secondary Received Data	To DTE
17	Receiver Signal Element Timing	To DTE
18	Test	To DCE
19	Secondary Request To Send	To DCE
20	Data Terminal Ready	To DCE
21	Signal Quality Detector	To DTE
22	Ring/Calling Indicator	To DTE
23	Data Signal Rate Selector	To DCE
23	Data Signal Rate Selector	To DTE
24	Transmitter Signal Element Timing	To DCE

ปัจจุบันนี้ อุปกรณ์ส่วนใหญ่ที่มีวางจำหน่ายอยู่จะเป็นไปตามมาตรฐาน RS-232C หรือ RS-232D (มาตรฐาน CCITT V.24 และ V.28 ก็ยังคงมีใช้อยู่ทั่วไปอย่างกว้างขวาง) วงจรของ RS-232

ส่วนมากไม่ได้ใช้ในการติดต่อสื่อสารระหว่างเครื่องเทอร์มินัล 2 เครื่องหรือเครื่องคอมพิวเตอร์ 2 เครื่องโดยตรง

ตาราง 4 และ 5 ซึ่งได้บรรยายถึงจุดประสงค์ของข้อต่อในการอินเตอร์เฟสข้อมูลโดยใช้มาตรฐาน RS-232 ที่พบได้บ่อยที่สุด



รูปที่ 2.15 แสดงคุณลักษณะของแรงดันไฟฟ้าต่ำสุดและสูงสุดสำหรับ RS-232D และมาตรฐาน V.28

คำบรรยายลักษณะวงจรของ RS-232

สิ่งแรกที่คุณอาจสังเกตเห็นก็คือ ตารางที่ 3 จะแสดงสายสัญญาณเพียง 11 เส้น จาก 25 เส้น ที่เป็นไปได้ของระบบ RS-232 ที่ต้องการใช้ในการทำการสื่อสารระหว่าง DTE ไปยัง DCE ให้สมบูรณ์ ส่วนมากแล้วคุณสามารถจะทิ้งสายวงจรตัวตรวจจับอัตราสัญญาณข้อมูล (Data Signal Rate Detector) และสายวงจรกราวด์ (Protective Ground) ออกไปได้ ทำให้เหลือสายสัญญาณที่ต้องต่อเพียง 9 เส้น

จงจำไว้ว่า RS-232 เป็นข้อกำหนดการอินเตอร์เฟสมาตรฐาน และสามารถใช้เพื่อจุดประสงค์อื่น ๆ ต่าง ๆ กันไป เช่นการสื่อสารแบบซิงโครนัส (synchronous communication) และรูปแบบการสื่อสารที่ต้องการสัญญาณนาฬิกาและสัญญาณกำหนดจังหวะเวลาเพิ่มเติมขึ้นมา ในความเป็นจริงแล้วคุณสามารถทำให้มีการสนทนากันจาก DTE ไปยัง DCE โดยใช้สายสัญญาณเพียง 3 เส้น จากจำนวน 11 เส้นที่แสดงในตารางที่ 3 ถ้าอุปกรณ์ DTE และ DCE ใช้ซอฟต์แวร์ที่เขียนขึ้นตามความต้องการของลูกค้า (custom-written software) ก็จะใช้เพียงสาย TD , RD และสายกราวด์สัญญาณเท่านั้นในการย้ายข้อมูลไปตามสายตัวนำ 3 เส้นนี้

ตารางที่ 2.2 แสดงรายละเอียดสำหรับการอินเตอร์เฟส RS-232 โดยใช้คอนเน็กเตอร์แบบ DB-25

หมายเลขขาสัญญาณ	ชื่อของขาสัญญาณ
1	Protective Ground
2	Transmitted Data
3	Received Data
4	Request To Send
5	Clear To Send
6	Data Set Ready
7	Signal Common
8	Received Line Signal Detect
9	Reserved For Testing
10	Reserved For Testing
11	Unassigned
12	Secondary Received Line Signal Detector
13	Secondary Clear To Send
14	Secondary Transmitted Data
15	Transmission Signal Element Timing
16	Secondary Received Data
17	Received Signal Element Timing
18	Unassigned
19	Secondary Request To Send
20	Data Terminal Ready
21	Signal Quality Detector
22	Ring Indicator
23	Data Signal Rate
24	Transmitter Signal Element Timing
25	Unassigned

ต่อไปนี้เป็นข้อกำหนดของขาสัญญาณ 11 ขา ที่แทบทุกคนนำไปใช้งาน :

- ขา 1 (Protective Ground Circuit , AA) ขานี้จะต่อเข้ากับตัวถังของอุปกรณ์ และสามารถต่อเข้ากับกราวด์ภายนอกถ้าอุปกรณ์อื่นๆ ต้องใช้ขานี้
- ขา 2 (Transmitted Data Circuit BA , TD) เป็นขาสัญญาณข้อมูลที่ออกมาจากอุปกรณ์ DTE กระแสบิตข้อมูลอนุกรมจากขานี้ คือข้อมูลที่จะถูกถอดรหัสโดยอุปกรณ์ DCE ที่มี
- ขา 3 (Received Data Circuit BB , RD) สัญญาณที่ขานี้จะถูกสร้างจากอุปกรณ์ DCE กระแสบิตข้อมูลอนุกรมนี้จะกำเนิดขึ้นที่อุปกรณ์ DTE ปลายทาง และเป็นผลผลิตของวงจรนับข้อมูลของอุปกรณ์ DCE สัญญาณนี้มักจะเป็นข้อมูลดิจิตอลที่ถูกสร้างขึ้นโดยอุปกรณ์ DCE ที่มีความฉลาดหรือจากวงจรถอดสัญญาณ (demodula□ion) ของโมเด็ม
- ขา 4 (Request to Send Circuit CA , RTS) สัญญาณนี้จะเตรียมพร้อมอุปกรณ์ DCE สำหรับการทำการส่งข้อมูล เมื่อสัญญาณ RTS นี้อยู่ในสภาวะ “ON” จะทำให้อุปกรณ์ DCE อยู่ในโหมดส่งข้อมูล (transmit mode) ในขณะที่สัญญาณนี้อยู่ในสภาวะ “OFF” จะทำให้อุปกรณ์ DCE อยู่ในโหมดรับข้อมูล (receive mode) อุปกรณ์ DCE ควรจะตอบสนองต่อสัญญาณ RTS ON โดยการทำให้สัญญาณ Clear to Send (CTS) อยู่ในสภาวะ “ON” ด้วยเมื่อสัญญาณ RTS อยู่ในสภาวะ “OFF” สัญญาณไม่ควรจะ “ON” ขึ้นอีก จนกว่าสัญญาณ CTS จะอยู่ในสภาวะ “OFF” เสียก่อน สัญญาณนี้จะถูกใช้ร่วมกับสัญญาณ DTR ,DSR และ DCD ขาสัญญาณ RTS จะถูกใช้อย่างมากในการควบคุมการไหลของข้อมูล
- ขา 5 (Clear To Send Circuit CB , CTS) สัญญาณนี้จะตอบรับกลับไปยังอุปกรณ์ DTE เมื่อได้รับสัญญาณ RTS และข้อมูลสามารถส่งออกไปได้ ข้อมูลจะถูกส่งไปตามตัวกลางที่ใช้สื่อสารได้ก็ต่อเมื่อสัญญาณ CTS นี้อยู่ในสภาวะ “ON” เท่านั้น สัญญาณนี้จะใช้ร่วมกับขา DTR , DSR และ DCD ขาสัญญาณ CTS นี้จะใช้ร่วมกับขา RTS สำหรับควบคุมการไหลของข้อมูล
- ขา 6 (Data Set Ready Circuit CC , DSR) สัญญาณ DSR จะบอกต่ออุปกรณ์ DTE ว่าอุปกรณ์ DCE ได้ต่อกับตัวกลางการสื่อสารที่ถูกต้องแล้วและในบางกรณีจะบ่งชี้ว่าสายโทรศัพท์อยู่ในสภาวะ “OFF HOOK” สภาวะ “OFF HOOK” นี้จะเป็นตัวบ่งชี้ว่าอุปกรณ์ DCE กำลังอยู่ในโหมด dialing หรือกำลังติดต่อกับอุปกรณ์ DCE อีกตัวหนึ่งอยู่ เมื่อสัญญาณ DSR นี้อยู่ในสภาวะ “OFF” อุปกรณ์ DTE ก็ควรจะถูกกำหนดให้ไม่สนใจสัญญาณอื่น ๆ ทั้งหมดจากอุปกรณ์ DCE ถ้าสัญญาณนี้ถูกทำให้อยู่ในสภาวะ “OFF” ก่อนอุปกรณ์ DTR แล้วอุปกรณ์ DTE ก็จะสรุปว่าการสื่อสารกันนั้นสิ้นสุดลง
- ขา 7 (Signal common Circuit , AB) สายตัวนี้จะให้สัญญาณอ้างอิงของกราวด์ร่วมกันสำหรับวงจรการแลกเปลี่ยนข้อมูลทั้งหมด ยกเว้นวงจร AA หรือ protective ground ข้อ

กำหนด RS-232 จะอนุญาตให้วงจรนี้ถูกต่อเพิ่มเติมเข้ากับ protective ground ภายในอุปกรณ์ DCE ได้ ถ้าจำเป็น

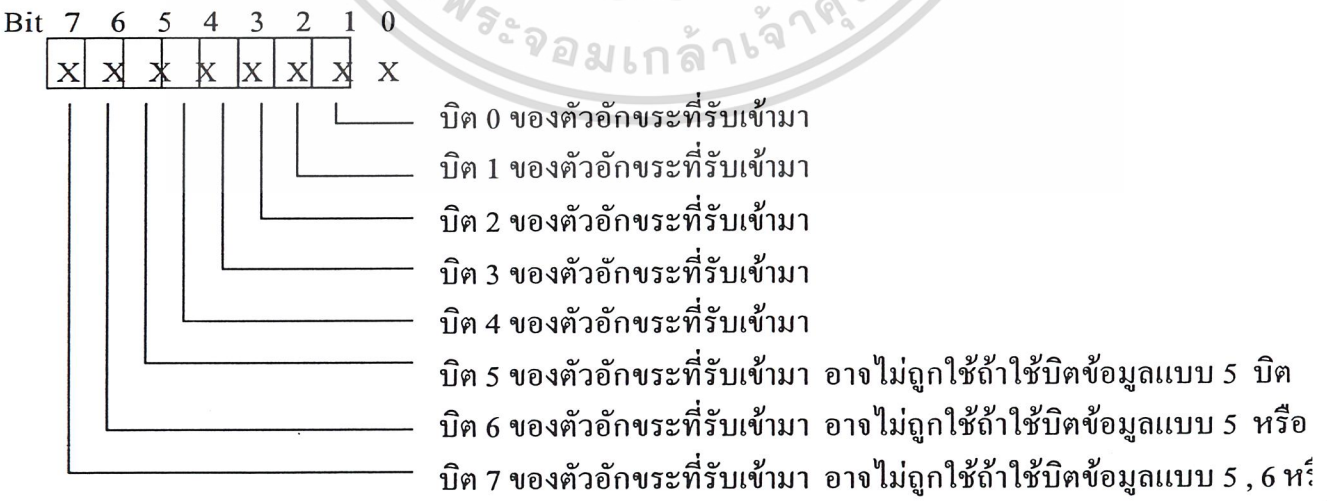
- ขา 8 (Data Carrier Detect Circuit CF , DCD) ขานี้ยังรู้จักกันในนามของ Received Line Signal Detect (RLSD) หรือขา Carrier Detect (CD) สัญญาณนี้จะแอกทีฟเมื่อเกิดสัญญาณพาหะที่เหมาะสมระหว่างอุปกรณ์ DCE ที่สถานีกับที่อยู่ในระยะไกลเมื่อสัญญาณนี้อยู่ในสภาวะ "OFF" สัญญาณที่ขา RD ควรจะถูกทำให้ค้างอยู่ในสภาวะ "Mark" (สถานะ "1" ในเลขฐานสอง)
- ขา 20 (Data Terminal Ready Circuit CD , DTR) สัญญาณ DTR ถูกควบคุมในการสวิทช์อุปกรณ์ DCE เข้ากับตัวกลางในการสื่อสารสัญญาณ DTR ON บ่งชี้ว่าอุปกรณ์ DCE ที่กำลังต่อเชื่อมกันอยู่ ก็ยังคงต่อเชื่อมกัน และถ้าไม่มีการต่อเชื่อมกันก็สามารถทำการต่อเชื่อมกันครั้งใหม่ได้ ปกติแล้วสัญญาณ DTR จะอยู่ในสภาวะ "OFF" เพื่อกระตุ้นให้เกิดสภาวะ ON HOOK (วางสาย) (hang up) อุปกรณ์ DCE โดยปกติแล้วจะตอบสนองต่อการกระตุ้นจากสัญญาณ DTR โดยการทำให้สัญญาณ DSR แอกทีฟ
- ขา 22 (Ring Indicator Circuit CE , RI) สภาวะ "ON" ของสัญญาณนี้จะบ่งชี้ว่าได้รับสัญญาณเรียกสายโทรศัพท์จากตัวกลางในการสื่อสาร (สายโทรศัพท์) ปกติแล้วจะขึ้นอยู่กับโปรแกรมควบคุม ในการที่จะทำให้เกิดสัญญาณนี้ขึ้นหรือไม่
- ขา 23 (Data Signal Rate Detector Circuit CH/CI , DSRD) วงจร CH เป็นส่วนประกอบของ DTE และวงจร CI เป็นส่วนประกอบของ DCE สัญญาณที่ขานี้ถูกใช้ในการเลือกค่าอัตราการส่งสัญญาณข้อมูลค่าใดค่าหนึ่งในสองค่าในกรณีที่ใช้โมเด็มที่มีอัตราการส่งข้อมูลได้ 2 ค่า (dual – rate modems) ถ้าสัญญาณที่ขานี้เป็น "ON" ก็จะเป็นการเลือกอัตราการส่งข้อมลที่มีค่าสูงที่สุดใน 2 ค่านั้น
- ตารางที่ 2.3 แสดงรายละเอียดของขาสัญญาณที่ต่อจาก DTE ไปยัง DCE โดยใช้คอนเน็กเตอร์แบบ DB-25

หมายเลขขาสัญญาณ	ชื่อของสายสัญญาณ
1	Protective Ground
2	Transmitted Data
3	Received Data
4	Request To Send
5	Clear To Send
6	Data Set Ready
7	Signal Common

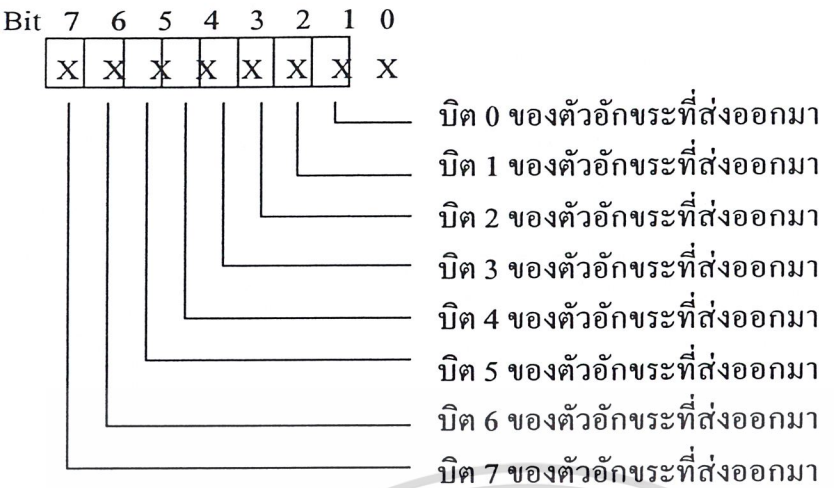
8	Data Carrier Detect
20	Data Terminal Ready
22	Ring Indicator
23	Data Signal Rate Detector

ตารางที่ 2.4 รายละเอียดการต่อแบบคอนเน็กเตอร์แบบ DB9 ตามมาตรฐาน RS-232

หมายเลขขาสัญญาณ	ชื่อของสายสัญญาณ
1	Data Carrier Detect
2	Received Data
3	Transmitted Data
4	Data Terminal Ready
5	Signal Common
6	Data Set Ready
7	Request To Send
8	Clear To Send
9	Ring Indicator



รูปที่ 2.16 แสดงรีจิสเตอร์พักข้อมูลที่รับเข้ามา (Receiver Buffer Register)



รูปที่ 2.17 แสดงรีจิสเตอร์ที่เก็บข้อมูลที่จะส่ง (Transmitter Holding Register)

ขั้นตอนการติดต่อระหว่างอุปกรณ์ DTE และ DCE

1. เมื่อจ่ายกำลังงานให้อุปกรณ์ DTE และอุปกรณ์นี้ส่งสัญญาณ DTR ออกมา
2. อุปกรณ์ DCE ถูกเปิดขึ้น และรับรู้ถึงสัญญาณ DTR ที่ส่งมาจากอุปกรณ์ DTE
3. อุปกรณ์ DCE ส่งสัญญาณ DSR ออกมา และโมเด็มก็กระทำกระบวนการ OFF-HOOK
4. ถ้าสายสัญญาณโทรศัพท์อยู่ในสภาพดีและปลายทางอีกด้านหนึ่งก็พร้อมจะรับข้อมูลแล้ว

   โดยจะตรวจจับพบสัญญาณพาหะ แล้วอุปกรณ์ DCE จะส่งสัญญาณ DCD ออกมา
5. อุปกรณ์ DTE ยกระดับสัญญาณ RTS ขึ้นสูง
6. อุปกรณ์ DCE จะสนองด้วยการส่งสัญญาณ CTS ออกมา
7. การติดต่อสื่อสารก็เริ่มขึ้น โปรแกรมควบคุมจะทำการส่งหรือรับข้อมูลส่วนลำดับขั้นในการ

   ตอบรับก็จะเป็นในทำนองนี้

1. อุปกรณ์ DTE จะส่งสัญญาณ DTR ออกมา
2. อุปกรณ์ DCE จะอยู่ในโหมดตอบรับอัตโนมัติ (auto-answer mode) โดยมีสัญญาณ DSR

   ออกมา
3. สถานีปลายทางส่งสัญญาณเรียกอุปกรณ์ DCE และอุปกรณ์ DCE ส่งสัญญาณ RI ออกมา
4. อุปกรณ์ DTE รับรู้ถึงสัญญาณ RI ที่ส่งมาจากเครื่องปลายทาง และอุปกรณ์ DCE ก็เข้า

สู่สภาวะ OFF-HOOK

5. อุปกรณ์ DCE ทำการแลกเปลี่ยนข้อมูลกับอุปกรณ์ DCE ที่อีกปลายทางหนึ่ง และมีการส่งสัญญาณ DCD ออกมา
6. อุปกรณ์ DCE จะส่งสัญญาณ RTS ออกมา หรืออาจจะรอรับข้อมูลก็ได้ขึ้นอยู่กับโปรแกรม

ที่ควบคุม

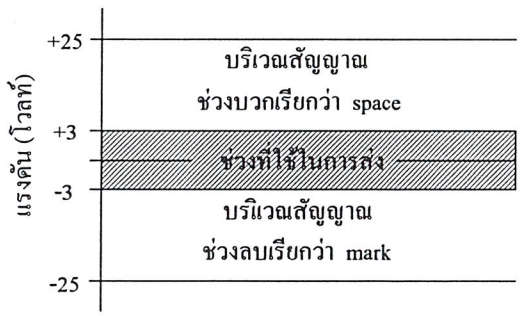
7. อุปกรณ์ DCE จะตอบสนองด้วยการส่งสัญญาณ CTS กลับออกมา
8. การติดต่อสื่อสารจะเริ่มขึ้น

ข้อจำกัดของ RS-232

ดูเหมือนว่าการที่ไม่มีวงจรควบคุมความต่อเนื่อง เป็นอุปสรรคที่ใหญ่อันหนึ่งของระบบ ES-232 แต่ก็มีวิธีสำหรับควบคุมความต่อเนื่องนั้น โดยกระทำผ่านซอฟต์แวร์จะดีกว่าฮาร์ดแวร์ ซึ่งเป็นสิ่งที่กระทำกันอยู่ในระบบใหม่ๆ โดยในอุปกรณ์ร่วมรุ่นเก่ายังไม่มีสายต่อ RS-232 ซึ่งคงใช้วิธีต่อตรงจากเทอร์มินัลเข้าสู่คอมพิวเตอร์ ไม่ว่าจะใช้โมเด็มด้วยหรือไม่ (แน่นอน เราต้องใช้สายและหัวต่อแบบ RS-232 แน่ๆ) และสำหรับการเชื่อมต่อแบบนี้ จะสามารถใช้สายยาวที่สุดได้ 50 ฟุต

ข้อจำกัดด้านระยะทาง

เครื่องส่ง RS-232 สามารถกำเนิดสัญญาณที่มีแรงดันระหว่าง +5 ถึง +25 โวลท์ สำหรับใช้ในสถานะใดสถานะหนึ่ง (space) และ -5 ถึง –25 โวลท์ สำหรับใช้ในสถานะตรงข้าม (mark) แต่โชคไม่ดีที่ระดับแรงดันเหล่านั้นมีค่าไม่เท่ากัน เช่น ในคอมพิวเตอร์กับจุดต่ออื่นๆ มักจะใช้มาตรฐาน TTL และ MOSFET ดังนั้นจึงต้องจ่ายแรงดันให้กับระบบ ดังแสดงในรูปที่ 9 เครื่องรับที่มีขั้วต่อ RS-232 จะใช้ระดับแรวดันตั้งแต่ +3 โวลท์ขึ้นไป เรียกว่า space และ –3 โวลท์ ลงมา เรียกว่า mark เมื่อสัญญาณเปลี่ยนจากสถานะหนึ่งไปยังสถานะตรงข้าม จะต้องใช้เวลาไม่เกิน 4 % ของเวลาที่ใช้ในแต่ละ 1 บิต ซึ่งต้องพิจารณาถึงค่า stray capacitance สูงสุดที่ยอมให้เกิดขึ้นได้ในสาย และค่าความจุจะจำกัดค่า rise time (เรียกว่า transition time) ของสัญญาณ



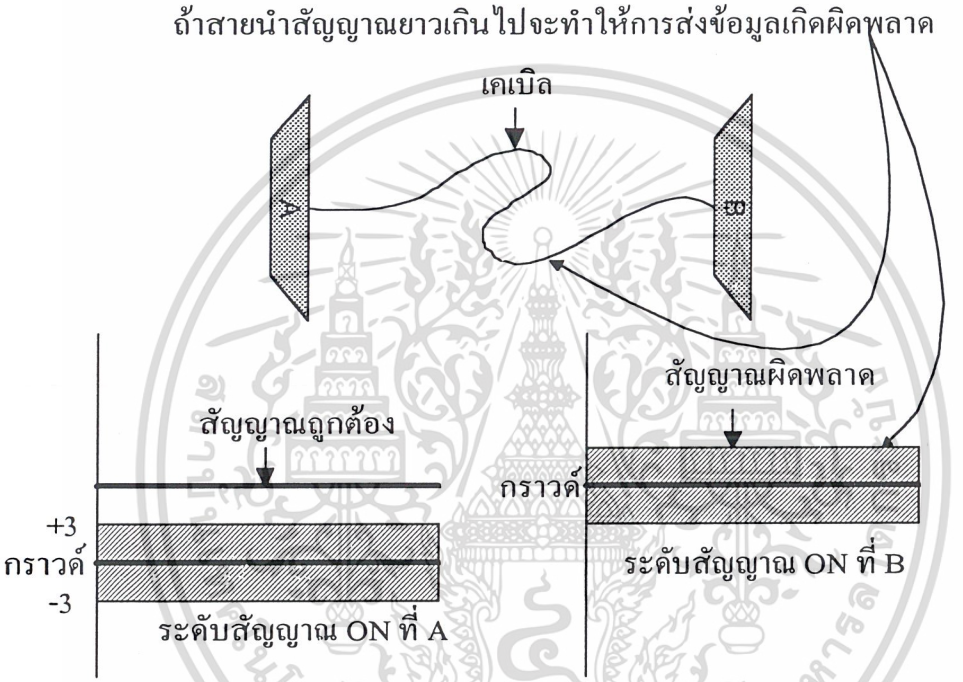
รูปที่ 2.18 ระดับสัญญาณ RS 232

ในระบบ RS-232 กำหนดค่าความจุในสายไม่เกิน 2,500 pF เพราะว่าสายที่ใช้กับระบบ RS-232 มีค่าความเก็บประจุประมาณ 40-50 pF/ฟุต ดังนั้นต้องทำการใช้สายยางไม่เกิน 2,500/50 =50 ฟุต

ข้อจำกัดด้านอัตราเร็ว

ข้อจำกัดอันที่ 2 ความเร็วของการส่งจะต้องไม่เกิน 20,000 บิตต่อวินาที แต่ส่วนใหญ่แล้วอัตราข้อมูลที่ใช้ร่วมกันระหว่างคอมพิวเตอร์และจุดต่ออื่น ๆ จะใช้ 19,200 บิตต่อวินาที เป็นอย่างมาก และมันยาก (และแพงเมื่อกล่าวถึงราคาของโมเด็ม) ที่จะส่งอัตราข้อมูลสูงขนาดใหญ่ผ่านไปตามชุมสายโทรศัพท์

ข้อจำกัดของสัญญาณกราวด์



รูปที่ 2.19 ปัญหาที่เกิดจากความต่างศักย์ระหว่างกราวด์ 2 จุดที่มีระยะห่างกัน

ข้อจำกัดอันที่ 3

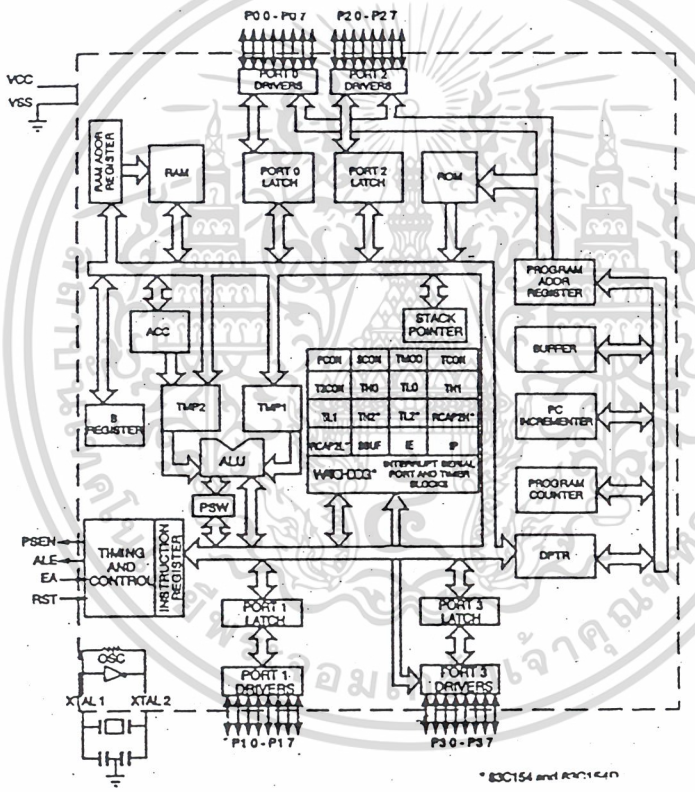
คือวิธีการต่อกราวด์เข้ากับตัวถัง ถึงแม้จะไม่ใช่ปัญหาสำคัญนัก เพื่อคุณภาพของสัญญาณข้อมูลและสัญญาณควบคุม จะต้องมีกราวด์อ้างอิงที่ตำแหน่งเดียวกัน คือขา 7 วิธีนี้เรียกว่าการส่งแบบไม่สมดุล (unbalanced transmission) ซึ่งสามารถใช้งานได้ดี แต่เมื่อความต่างศักดิ์เกิดขึ้นระหว่างกราวด์ของสายทั้งสองด้าน จะทำให้ช่องว่าง (transition region) ระหว่าง mark กับ space จะแคบลง เป็นผลทำให้การถอดรหัสสัญญาณผิดพลาด

ไมโครคอนโทรลเลอร์ MCS-51

1. ความรู้เบื้องต้นเกี่ยวกับไมโครคอนโทรลเลอร์ MCS-51

ปัจจุบันมีไมโครคอนโทรลเลอร์ MCS-51 ซึ่งเป็นไมโครคอมพิวเตอร์แบบชิพเดี่ยว (ไม่ต้องต่อกับอุปกรณ์ภายนอกก็สามารถทำงานได้) มีความสะดวกในการใช้งานและเขียนโปรแกรมควบคุมด้วยภาษาเบสิกได้โดยไม่ต้องศึกษาการทำงานของวงจรเหมือนกับภาษา แอสเซมบลี้หรือบางท่านที่ถนัดภาษาแอสแซมบลี้ก็ยังสามารถใช้ได้เช่นเดียวกัน

2. โครงสร้างภายในของไมโครคอนโทรลเลอร์ MCS-51

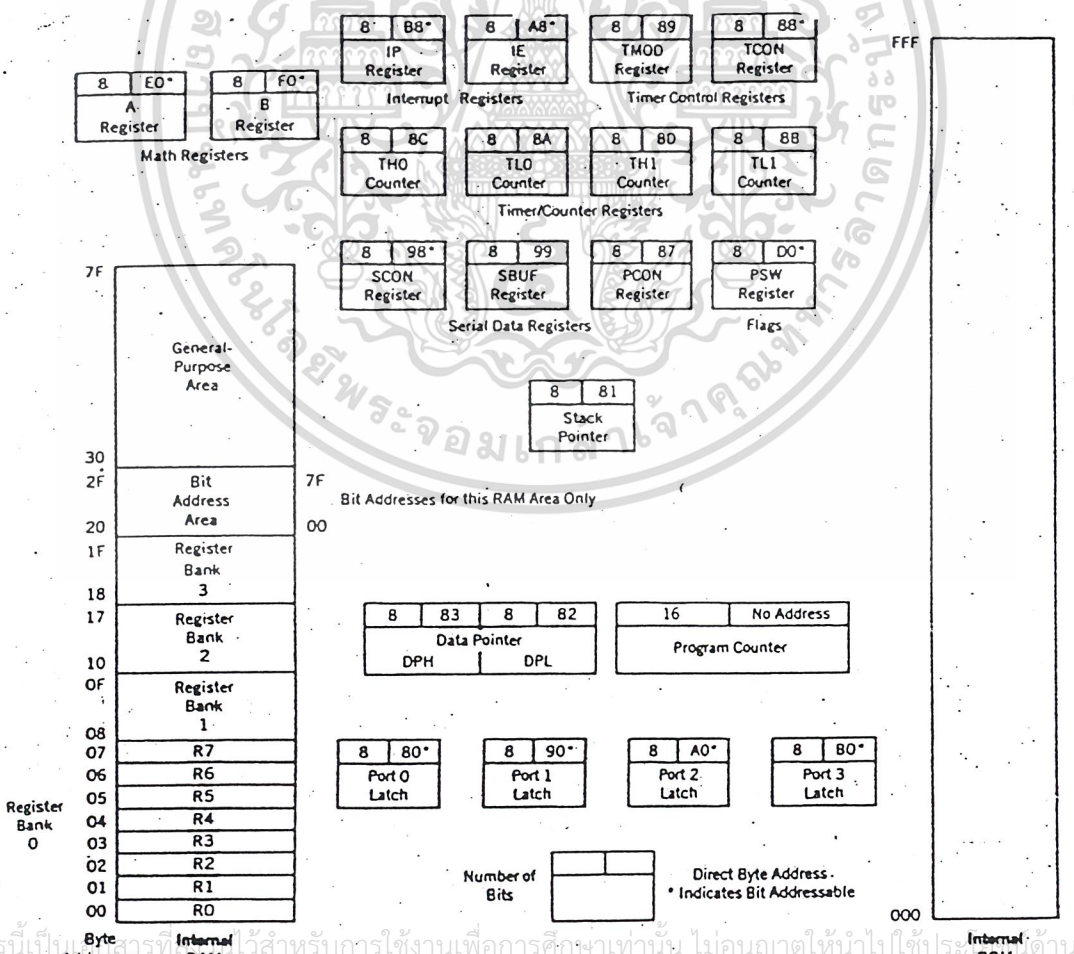


รูปที่2.20 แสดงโครงสร้างภายในชิพ MCS 51

3. การจัดการหน่วยความจำของไมโครคอนโทรลเลอร์ เบอร์ 8051

หน่วยความจำของ 8051 แบ่งออกไว้เป็น 2 แบบตามลักษณะของการใช้งาน คือ

3.1 หน่วยความจำโปรแกรม (Program Memory) เป็นหน่วยความจำที่ใช้เก็บคำสั่งในรูปรหัสภาษาเครื่อง (Machine Language) ซึ่งต้องการให้ 8051 ทำงาน เมื่อ 8051 ทำงานก็จะอ่านข้อมูลที่เก็บในหน่วยความจำประเภทนี้เข้าไปถอดรหัสแล้วสร้างสัญญาณควบคุมส่วนอื่นๆ ตามการทำงานของแต่ละคำสั่งนั้น หน่วยความจำแบบนี้จะต้องเป็นแบบหน่วยความจำอ่านได้ เท่านั้น (Read Only Memory (ROM)) และผู้ใช้ต้องเขียนข้อมูลในแต่ละตำแหน่งของหน่วยความจำเป็นรหัสภาษาเครื่องของ 8051 ตามลำดับการทำงานที่ต้องการ (หน่วยความจำแบบอ่านได้เท่านั้น ซึ่งเมื่อปิดไฟแล้วข้อมูลก็ไม่มีการสูญหาย) ในระหว่างการทำงานของ 8051 ผู้ใช้จะไม่สามารถใช้คำสั่งทำการเขียนข้อมูลลงในหน่วยความจำแบบนี้ได้ จำหนวนตำแหน่งสูงสุดของหน่วยความจำแบบนี้ที่ 8051 จะใช้งานได้คือ 65536 ตำแหน่ง ค่าของตำแหน่ง (Address) จะเขียนเป็นเลขฐาน 16 ได้ตั้งแต่ 0000H ถึง FFFFH หน่วยความจำตำแหน่ง 0000H ถึง 0FFFFH จำนวน 4 กิโลไบท์ ถ้าต้องการให้ 8051 ทำงานตามทำสั่งที่เก็บไว้ใน ROM ภายใน 8051 ก็ให้ป้อนสัญญาณสภาวะลอจิกสูง (1) เข้าที่ขา EA ของ 8051 แต่ถ้าต้องการให้ทำงานในโปรแกรมที่เก็บไว้ใน ROM ภายนอก 8051 ก็ให้ต่อลอจิกต่ำ (0) เข้าที่ขา /EA ของ 8051 ส่วนหน่วยความจำที่ตำแหน่ง 1FFFH ถึง FFFFH จะต้องต่ออยู่ภายนอก 8051 เสมอดังแสดงในแผนภูมิหน่วย-ความจำ (Memory Map)



รูปที่ 2.21 ตำแหน่งต่างๆของรีจิสเตอร์ต่างๆและหน่วยความจำ

หน่วยความจำภายใน (Internal Memory) หมายถึงหน่วยความจำนั้นอยู่ภายใน 8051ส่วนหน่วยความจำภายนอก (External Memory) หมายถึง หน่วยความจำนั้นอยู่ภายนอก 8051

3.2 หน่วยความจำของข้อมูล (Data Memory) เป็นหน่วยความจำที่ไมโคคอนโทรลเลอร์เบอร์ 8051 จะใช้สำหรับพัก, เก็บข้อมูล แล้วเรียกมาใช้ใหม่ในระหว่างการทำงานของไมโครคอนโทรลเลอร์เบอร์ 8051 การอ่านหรือเขียนข้อมูลจากหน่วยความจำจะกระทำ โดยคำสั่งที่เก็บไว้ในหน่วยความจำของโปรแกรม หน่วยความจำแบบนี้เป็นประเภท หน่วยความจำอ่านและเขียนได้ (Random Access Memory (RAM)) ถ้ามีไฟเลี้ยงอยู่ข้อมูลที่เก็บไว้จะไม่สูญหาย แต่ถ้าปิดเครื่องหรือไม่จ่ายไฟ ให้แก่น่วยความจำอ่านและเขียนได้แล้วข้อมูลในหน่วยความจำอ่านและเขียนได้ก็จะสูญหายไป หน่วยความจำของข้อมูลของไมโครคอนโทรลเลอร์เบอร์ 8051 จะมีอยู่ 2 ชุด ชุดหนึ่งอยู่ภายใน 8051 จำนวน 128 ไบท์ที่ตำแหน่ง 00H ถึง 7FH และอีกชุดหนึ่งจะต้องต่ออยู่ภายนอกของวงจรรวม 8051 มีได้สูงสุด 65536 ไบท์ (64 กิโลไบท์) อยู่ที่ตำแหน่ง 0000H ถึง FFFFH หน่วยความจำของข้อมูล ภายใน 8051 ที่ตำแหน่ง 08H ถึง FFH นั้นไม่ได้มีอยู่ทุกตำแหน่ง จะมีเฉพาะในบางตำแหน่ง ซึ่งเรียกหน่วยความจำบางตำแหน่งนี้ว่า รีจิสเตอร์พิเศษ (Special Function Register (SFR)) เพราะจะใช้หน่วยความจำเหล่านี้สำหรับงานพิเศษเท่านั้น

4. รีจิสเตอร์ภายในไมโครคอนโทรลเลอร์ MCS-51

ไมโครคอนโทรลเลอร์ MCS-51 มีรีจิสเตอร์ที่อำนวยความสะดวก ในการใช้งานตามคำสั่งต่างๆ ประกอบด้วยแอคคิวมูเลเตอร์ รีจิสเตอร์ B ที่ใช้ในการคูณและหาร รีจิสเตอร์สถานะ สแต็กพออยน์เตอร์ ข้อมูลพอยน์เตอร์ (2x8 บิท หรือ 1x16 บิท) พอร์ทหมายเลขศูนย์ถึงพอร์ทหมายเลขสามรีจิสเตอร์แบบคู่ ซึ่งใช้ส่งและรับข้อมูลชนิดอนุกรม รีจิสเตอร์ 16 บิท ที่เป็นวงจรตั้งเวลาและวงจรนับตัวที่ 3 รีจิสเตอร์ คำสั่งสำหรับหน้าที่พิเศษ (เช่น การอินเตอร์รัพท์ RTC:Read Time Clock) และอินพุท, เอาท์พุทแบบอนุกรม

5. ไทม์เมอร์/เคาน์เตอร์

ไมโครคอนโทรลเลอร์ในตระกูล MCS-51 มีรีจิสเตอร์พิเศษที่สามารถเลือกใช้งานเป็นไทม์เมอร์หรือเคาน์เตอร์อย่างใดอย่างหนึ่ง (นับจำนวนแมชชีนไซเคิลหรือนับจำนวนพลัส์ที่เกิดขึ้นภายนอกชิป) รีจิสเตอร์ประเภทนี้มีอยู่ด้วยกัน 2 ตัว แต่ละตัวมีขนาด 16 บิท โดยมีชื่อเรียกว่าไทม์เมอร์ 0 และไทม์เมอร์ 1 ตามลำดับรีจิสเตอร์ที่ใช้เป็นไทม์เมอร์ 0 ประกอบขึ้นจากรีจิสเตอร์ใช้งานเฉพาะ TL0, TH0 ส่วนรีจิสเตอร์ที่ใช้เป็นไทม์เมอร์ 1 ประกอบขึ้นจากรีจิสเตอร์ใช้งานเฉพาะ TL1, TH1 รีจิสเตอร์ไทม์เมอร์ 0 และไทม์เมอร์ 1 สามารถกำหนดการใช้งานเป็นไทม์เมอร์หรือเคาน์เตอร์อย่างใดอย่างหนึ่งได้ ในการทำงานเป็นไทม์เมอร์หรือเคาน์เตอร์จะมีรายละเอียดที่แตกต่างกันออกไปดังนี้

5.1 ไทม์เมอร์

ค่าในรีจิสเตอร์ที่เป็นไทม์เมอร์ที่ถูกเลือกใช้งานจะถูกเพิ่มค่าทุกๆ แมชชีนไซเคิล ดังนั้นจึงสามารถคิดว่าในขณะทำงานเป็นไทม์เมอร์หมายถึงใช้รีจิสเตอร์เป็นตัวนับจำนวนแมชชีนไซเคิลได้ และเนื่องจากใน 1 แมชชีนไซเคิลใดๆ ของไมโครคอนโทรลเลอร์ MCS-51 ประกอบไปด้วย 12 คาบสัญญาณออสซิลเลเตอร์ (oscillator period) ดังนั้นอัตราเร็วใน การนับ (count rate) จึงมีค่าเป็น 1/12 ของความถี่ออสซิลเลเตอร์ที่ใช้

5.2 เคาน์เตอร์

ค่าในรีจิสเตอร์ที่ใช้เป็นเคาน์เตอร์ที่ถูกเลือกใช้งานจะถูกเพิ่มค่าทีละหนึ่ง เมื่อมีการเปลี่ยนสถานะซึ่งตรวจจับได้จากขา T0, T1 หรือ T2 (ใน 8052) ขึ้นกับรีจิสเตอร์ที่ถูกเลือกใช้งานเป็นเคาน์เตอร์ในขณะนั้น การตรวจสอบการเปลี่ยนสถานะจะตรวจเฉพาะในขณะที่สัญญาณมีการเปลี่ยนค่าจาก 1 เป็น 0 โดยมีรายละเอียดในการตรวจสอบสัญญาณดังนี้

ไมโครคอนโทรลเลอร์ MCS-51 จะตรวจสอบสถานะสัญญาณที่ขา T0, T1 หรือ T2 (ใน 8052) โดยการตรวจสอบจะเกิดขึ้นในระหว่างสเตท 5 เฟส 2 ของแต่ละแมชชีนไซเคิลรายละเอียดในการตรวจสอบการเปลี่ยนสถานะสัญญาณที่แต่ละขา จะมีลักษณะที่เหมือนกัน ดังนี้ เมื่อขา T0, T1 หรือ T2 มีสถานะสัญญาณเป็น 1 ในขณะที่สเตท 5 เฟส 2 ของแมชชีนไซเคิลใดๆ และสเตท 5 เฟส 2 ของแมชชีนไซเคิลถัดไป หากสัญญาณที่ขา T0, T1 หรือ T2 มีค่าเปลี่ยนเป็น 0 จะทำให้รีจิสเตอร์ที่ถูกเลือกใช้งานเป็นเคาน์เตอร์ถูกเพิ่มค่าขึ้นอีก 1 ในช่วงสเตท 3 เฟส 1 ของแมชชีนไซเคิลซึ่งตรวจพบการเปลี่ยนสถานะของสัญญาณดังนั้นไมโครคอนโทรลเลอร์ MCS-51 จำเป็นจะต้องใช้เวลา 2 แมชชีนไซเคิล (24 คาบสัญญาณออสซิลเลเตอร์) เพื่อตรวจสอบการเปลี่ยนสถานะสัญญาณจาก 1 เป็น 0 ที่ขา T0, T1 หรือ T2 จึงทำให้อัตราการนับสูงสุดของเคาน์เตอร์ในไมโครคอนโทรลเลอร์ MCS-51 ถูกจำกัดความถี่ของสัญญาณไว้ที่ 1/24 ของความถี่ออสซิลเลเตอร์ที่ใช้โดยไม่มีข้อจำกัดในเรื่อง dutycycle (อัตราส่วนของช่วงเวลาที่สัญญาณมีค่าเป็น 1 ต่อคาบเวลาของสัญญาณ) ของสัญญาณ แต่เพื่อให้มั่นใจว่าสถานะสัญญาณจะถูกตรวจจสอบเข้ามาอย่างน้อย 1 ครั้งก่อนที่จะเปลี่ยนระดับจึงสมควรให้สถานะสัญญาณคงค่า 0 หรือ 1 อย่างน้อย 1 แมชชีนไซเคิลเต็ม

นอกจากจะสามารถเลือกการทำงานของรีจิสเตอร์ให้เป็นไทม์เมอร์หรือเคาน์เตอร์ได้แล้ว ในแต่ละการทำงานเป็นไทม์เมอร์หรือเคาน์เตอร์ของไทม์เมอร์ 0 หรือไทม์เมอร์ 1 ก็ยังมีการทำงานที่แยกย่อยลงไปอีกถึง 4 แบบ (โหมด 0, 1, 2 และ 3) ตามความเหมาะสมของการใช้งาน (ไทม์เมอร์ 0 และไทม์เมอร์ 1 มีให้เลือก 4 แบบ แต่ไทม์เมอร์ 2 มีให้เลือกเพียง 3 แบบ)การทำงานย่อยทั้ง 4 ประเภทของไทม์เมอร์ 0 และไทม์เมอร์ 1 มีรายละเอียดดังต่อไปนี้

5.3 ไทม์เมอร์ 0 และไทม์เมอร์ 1

ไทม์เมอร์ 0 และไทม์เมอร์ 1 ทั้ง 2 ตัว มีอยู่ในไมโครคอนโทรลเลอร์ MCS-51 ทุกเบอร์ ผู้ใช้สามารถเลือกการทำงานให้เป็นไทม์เมอร์หรือเคาน์เตอร์อย่างใดอย่างหนึ่ง โดยการกำหนดค่าบิท C/T ใน

รีจิสเตอร์ใช้งานเฉพาะ TMOD ดังแสดงในรูปที่ 2.22 (ไทม์เมอร์ 0 ใช้บิท 2 ส่วนไทม์เมอร์ 1 ใช้บิท 6) โดยหากบิทนี้มีค่าเป็น 0 หมายถึงเลือกให้รีจิสเตอร์ทำงานเป็น ไทม์เมอร์ (นับจำนวนแมชชีนไซคิล) ถ้าบิทนี้มีค่าเป็น 1 หมายถึงเลือกให้ทำงานเป็นเคาน์เตอร์ (นับจำนวนการเปลี่ยนสถานะจาก 1 เป็น 0 ที่ขา T0 หรือ T1)

รีจิสเตอร์ใช้งานเฉพาะ TMOD (Timer / Counter Mode Control Register) ไม่สามารถเข้าถึงข้อมูลในระดับบิทได้

บิท GATE

บิทเลือกการควบคุมให้รีจิสเตอร์สำหรับใช้เป็นไทม์เมอร์หรือเคาน์เตอร์ทำงานโดยควบคุมจากฮาร์ดแวร์หรือซอฟต์แวร์ ดังนี้

(1) เมื่อบิท TRx(TR0, TR1) และ GATE ถูกเซต ไทม์เมอร์หรือเคาน์เตอร์จะทำงานต่อเมื่อสถานะที่ขา INTx(INT0, INT1) มีค่าเป็น 1 (ควบคุมจากฮาร์ดแวร์)

(2) เมื่อบิท GATE ถูกเคลียร์ไทม์เมอร์หรือเคาน์เตอร์จะทำงานก็ต่อเมื่อบิท Trx ถูกเซตโดยไม่ขึ้นกับสถานะสัญญาณที่ขา INTx (ควบคุมจากซอฟต์แวร์)

บิท C/T

บิทเลือกการทำงานของรีจิสเตอร์สำหรับใช้เป็นไทม์เมอร์หรือเคาน์เตอร์ดังนี้

0 หมายถึงทำงานเป็นไทม์เมอร์ (นับจำนวนแมชชีนไซเคิล)

1 หมายถึงทำงานเป็นเคาน์เตอร์ (นับจำนวนพัลส์ภายนอกที่ขา Tx)

บิท M0 และ M1

M1 บิทสำหรับเลือกโหมดการทำงานของไทม์เมอร์ 0 หรือไทม์เมอร์ 1

M0 บิทสำหรับเลือกโหมดการทำงานของไทม์เมอร์ 0 หรือไทม์เมอร์ 1

0 0 โหมด 0 : ไทม์เมอร์หรือเคาน์เตอร์ขนาด 13 บิท

0 1 โหมด 1 : ไทม์เมอร์หรือเคาน์เตอร์ขนาด 16 บิท

1 0 โหมด 2 : ไทม์เมอร์หรือเคาน์เตอร์ขนาด 18 บิท ที่มีการโหมดค่าเองเมื่อเกิด โอเวอร์โฟลว์

1 1 โหมด 3 : ไทม์เมอร์ 0 รีจิสเตอร์ TL0 ใช้เป็นไทม์เมอร์หรือเคาน์เตอร์ขนาด 8 บิท ที่ควบคุมการทำงานได้จากบิทของไทม์เมอร์ 0 รีจิสเตอร์ TH0 ใช้เป็นไทม์เมอร์ขนาด 8 บิท ที่ควบคุมการทำงานได้ จากบิทของไทม์เมอร์ 1

1 1 โหมด 3 : ไทม์เมอร์ 1 หยุดทำงาน (หยุดนับ)

รีจิสเตอร์ที่ใช้งานเป็นไทม์เมอร์หรือเคาน์เตอร์ทั้ง 2 ตัวสามารถทำงานแตกต่างกันออกไป 4 แบบคือ โหมด 0, 1, 2 และ 3 โดยการเปลี่ยนค่าบิท M0 และ M1 ในรีจิสเตอร์ใช้งานเฉพาะ TMOD เช่นเดียวกับบิท C/T ดังในรูปที่ 2.22 การทำงานในโหมด 0, 1, 2 จะคล้ายๆ กัน สำหรับ

GATE	C/T	M1	M0	GATE	C/T	M1	M0

GATE When TRx (in TCON) is set and GATE = 1.TIMER/COUNTERx will run only while INTx

pin is high (hand ware control). When GATE=0.timer/counter will run only while TRx= 1 software control)

C/T TIMER or COUNTER selector.Cleared for TIMER operation (input on internal system clock). Set for Counter operation (inout from Tx input pin)

M1 Mode selector bit.(NOTE 1)

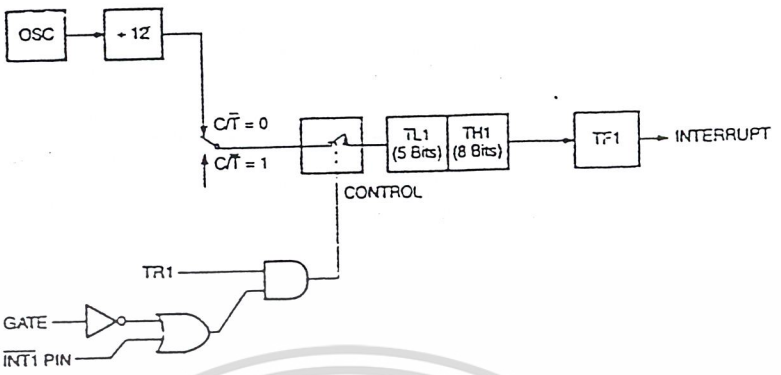
M0 Mode selector bit.(NOTE 1)

NOTE 1 :

M1	M2	Operating mode
0	0	0 13 bit TIMER
0	1	1 16 bit TIMER/COUNTER
1	0	2 8-bit Auto-Reload Timer/ counter
1	1	3 (TIME0) TLO is an 8-bit Timer/Counter controlled by the standard Timer 0 control bit . THO is an 8-bit Timer and is controlled by timer 1 control bits
1	1	3 (Timer1) Timer/Counter 1 stoped.

รูปที่ 2.22 รีจิสเตอร์ใช้ในงานเฉพาะ TMOD

ไทม์เมอร์ 0 และไทม์เมอร์ 1 แต่ในโหมด 3 ริสเตอร์ที่ใช้งานเป็นไทม์เมอร์หรือเคาน์เตอร์ทั้งสองตัว จะมีการทำงานที่ต่างออกไปจาก 3 โหมดแรก รายละเอียดการทำงานทั้ง 4 โหมดมีดังนี้



รูปที่ 2.23 การทำงานของไทม์เมอร์/เคาน์เตอร์ 1 โหมด 0 :13 bits counter

1.โหมด 0 สำหรับการทำงานเป็นไทม์เมอร์หรือเคาน์เตอร์ในโหมด 0 ของไมโครคอนโทรลเลอร์ MCS-51 ใช้รีจิสเตอร์ขนาด 8 บิทเป็นตัวนับ สัญญาณพัลส์ภายนอก โดยมีการเพิ่มค่าครั้งละ 1 ทุกครั้งที่นับสัญญาณได้ครบ 32 ครั้ง (สัญญาณอินพุทถูกหารด้วย 32)

ในการทำงานโหมดนี้ รีจิสเตอร์ที่ใช้นับไม่ว่าจะถูกกำหนดการทำงานเป็นไทม์เมอร์หรือเคาน์เตอร์จะถูกใช้เพียง 13 บิท เท่านั้น (8 บิท ในรีจิสเตอร์ TLx รวมกับ 5 บิท ในรีจิสเตอร์ THx) โดยในขณะที่ค่าในรีจิสเตอร์ถูกเปลี่ยนจากเดิมที่เป็น 1 ทั้งหมดเป็น 0 ทั้งหมด (เกิดโอเวอร์โฟลว์) จะทำให้บิท TFx (บิท TF0 หรือ TF1) ถูกเซต สัญญาณที่ใช้ในการนับจะผ่านเข้ามายังรีจิสเตอร์ที่ทำการนับได้ก็ต่อเมื่อ บิท TRx = 1 และ (บิท gate = 0 หรือสัญญาณที่ขา INTx (INT0, INT1) มีค่าเป็น 1) รายละเอียดการทำงานของโหมดนี้มีดังแสดงในรูปที่ 23

การเซตให้บิท Gate เป็น 1 จะเป็นการกำหนดให้รีจิสเตอร์ที่ใช้เป็นไทม์เมอร์หรือเคาน์เตอร์ถูกควบคุมการทำงานโดยสัญญาณที่ขา INTx (ควบคุมการนับด้วยฮาร์ดแวร์) ทั้งนี้เพื่อให้การนำไมโครคอนโทรลเลอร์ MCS-51 ไปใช้วัดความกว้างของพัลส์ทำได้ง่าย (บิท TRx ต้องเป็น 1) ส่วนการเคลียร์ให้บิท Gate เป็น 0 จะมีผลให้การควบคุมการทำงานของรีจิสเตอร์ที่ใช้เป็นไทม์เมอร์หรือเคาน์เตอร์ทั้งสองกระทำได้โดยการเซตหรือเคลียร์บิท TRx ด้วยคำสั่งในโปรแกรม (ควบคุมด้วยซอฟต์แวร์)

บิท TR0, TR1 จะอยู่ในรีจิสเตอร์ใช้งานเฉพาะ TCON ดังแสดงในรูปที่ 25 ส่วนบิท Gate อยู่ในรีจิสเตอร์ใช้งานเฉพาะ TMOD ดังแสดงในรูปที่ 23

รีจิสเตอร์ที่ใช้เป็นไทม์เมอร์หรือเคาน์เตอร์จะถูกใช้เพียง 13 บิท ที่ประกอบขึ้นจาก 8 บิท ในรีจิสเตอร์ใช้งานเฉพาะ THx และ 5 บิท ในรีจิสเตอร์ใช้งานเฉพาะ TLx โดย 3 บิท

บนของรีจิสเตอร์ใช้งานเฉพาะ TLx ไม่ถูกใช้งานในโหมดนี้ และการเซตค่าของบิท TRx (RUN Flag) ก็ไม่ได้เคลียร์ค่าในรีจิสเตอร์ทั้งสองแต่อย่างใด

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1	TCON.7	Timer 1 overflow flag. Set by handware whenThe Timer/Counter 1 Overfiows.Cleared by hardware as processor vectors to the interrrupt service routine
TR1	TCON.6	Timer 1 run control bit. Set/cleared by software to turn Timer /Counter 1 ON/OFF
TF0	TCON.5	Timer 0 overflowflag.Set by hardware when the Timer/Counter 0 overflows. Cleared by hardwar as processor vectors to service routine
TR0	TCON.4	Timer 0 run control bit. Set/Cleared by software to turnTimer/Counter 0 ON/OFF
IE1	TCON.3	External interrupt 1 edge flag.Set by hardware when External interrupt edge is detected. clear by hardware when interrupt is processed.
IT1	TCON.2	Interrupt 1 type control bit. Set/Cieared by software to specify failling edge/flow level triggered External interrupt.
IE0	TCON.1	External interrupt 0 edge flag.Set by hardware when External interrupt edge is detected. clear by hardware when interrupt is processed.
IT0	TCON.0	Interrupt 0 type control bit. Set/Cieared by software to specify failling edge/flow level triggered External interr

รูปที่ 2.24 รีจิสเตอร์ใช้งานเฉพาะ TCON

รีจิสเตอร์ TCON (Timer/Counter Control Register) เข้าถึงข้อมูลได้ในระดับบิท

บิท TF1 บิทแสดงการเกิดโอเวอร์โฟลว์ ของไทม์เมอร์ 1 ถูกเซตเองเมื่อไทม์เมอร์ 1 เกิด โอเวอร์โฟลว์ และถูกเคลียร์เองเมื่อซีพียูย้ายไปทำงานที่โปรแกรมบริการอินเตอร์-รัพท์

บิท TR1 บิทควบคุมการนับของไทม์เมอร์ 1 ควบคุมจากโปรแกรม

1 ไทม์เมอร์ 1 เริ่มการทำงานต่อ (นับต่อ)

0 ไทม์เมอร์ 1 หยุดทำงาน (หยุดนับสัญญาณนาฬิกา ภายในหรือนับจำนวนพัลส์ภายนอก)

บิท TF0 บิทแสดงการเกิดโอเวอร์โฟลว์ของไทม์เมอร์ 0 ถูกเซตเมื่อไทม์เมอร์ 0 เกิดโอเวอร์โฟลว์และถูกเคลียร์เอง เมื่อซีพียูย้ายไปทำงานที่โปรแกรมบริการอินเตอร์รัพท์

บิท TF0 บิทควบคุมการนับของไทม์เมอร์ 0 ควบคุมจากโปรแกรม

1 ไทม์เมอร์ 0 เริ่มทำงานต่อ (นับต่อ)

0 ไทม์เมอร์ 0 หยุดทำงาน

บิท IE1 บิทแสดงสถานะสัญญาณอินเตอร์รัพท์ภายนอกชนิดที่ 1 จะถูกเซตเองโดยฮาร์ดแวร์ เมื่อมีสัญญาณอินเตอร์รัพท์ และจะถูกเคลียร์เองโดยคำสั่ง RETI ที่อยู่ในโปรแกรมส่วนบริการอินเตอร์รัพท์ เมื่ออินเตอร์รัพท์ที่เกิดขึ้นได้มาจากการตรวจสอบจากการเปลี่ยนสถานะของสัญญาณ

บิท IT1 บิทเลือกประเภทการตรวจสอบสัญญาณอินเตอร์รัพท์ที่เกิดขึ้นที่ขา INT1 โดย

1 ตรวจสอบการเปลี่ยนระดับสัญญาณจาก 1 เป็น 0 ที่ขา INT1

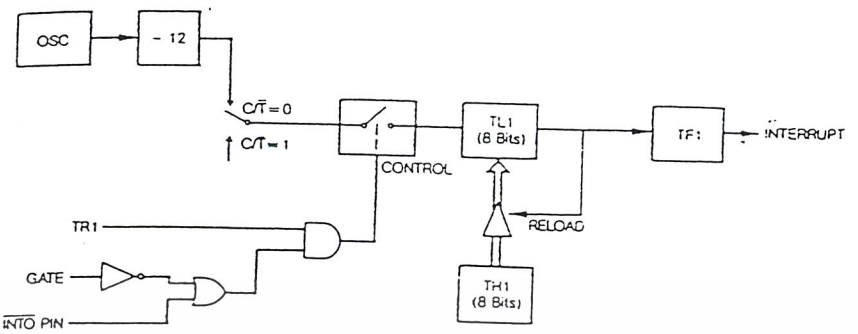
0 ตรวจสอบระดับของสัญญาณที่ขา INT1

บิท IE0 บิทแสดงสถานะสัญญาณอินเตอร์รัพท์ภายนอกชนิดที่ 0 จะถูกเซตเองโดยฮาร์ดแวร์ เมื่อมีสัญญาณอินเตอร์รัพท์ และจะถูกเคลียร์เองโดยคำสั่ง RETI ที่อยู่ในโปรแกรมส่วนบริการอินเตอร์รัพท์ เมื่ออินเตอร์รัพท์ที่เกิดขึ้นได้มาจากการตรวจสอบจากากรเปลี่ยนสถานะของสัญญาณ

บิท IT0 บิทเลือกประเภทการตรวจจสอบสัญญาณอินเตอร์รัพท์ที่เกิดขึ้นที่ขา INT1 เหมือนบิท IT1

(2) โหมด 1 การทำงานของรีจิสเตอร์ที่ใช้เป็นไทม์เมอร์หรือเคาน์เตอร์ในโหมด 1 จะเหมือนในโหมด 0 ทุกประการเว้นแต่ค่าในรีจิสเตอร์ที่ใช้เป็นไทม์เมอร์หรือเคาน์เตอร์ถูกใช้งานครบทั้ง 16 บิทนั่นคือ ไทม์เมอร์หรือเคาน์เตอร์ในโหมดนี้มีขนาด 16 บิท

(3) โหมด 2 การทำงานในโหมด 2 จะกำหนดให้รีจิสเตอร์ที่ใช้เป็นไทม์เมอร์หรือ เคาน์เตอร์ถูกใช้ในการนับเพียง 8 บิท (จากรีจิสเตอร์ใช้งานเฉพาะ TLx) ที่มีการโหลดค่าเองด้วยค่าในรีจิสเตอร์ใช้งานเฉพาะ THx เมื่อเกิดโอเวอร์โฟลว์ ในรีจิสเตอร์ TLx โดยค่าในรีจิสเตอร์ THx นี้สามารถกำหนดได้ล่วงหน้าโดยซอฟต์แวร์ และจะไม่เปลี่ยนแปลงเมื่อถูกโหลดไปไว้ในรีจิสเตอร์ TLx การทำงานในโหมด 2 มีดังแสดงในรูปที่ 2.25



รูปที่ 2.25 การทำงานของไทม์เมอร์เคาน์เตอร์ 1 โหมด 2 :8 bits autoreload

การทำงานโหมดนี้ มีไว้เพื่อใช้สร้างสัญญาณอินเตอร์รัพท์ที่มีคาบเวลาคงที่ หรือใช้สร้างฐานเวลาให้แก่ซีพียูในไมโครคอนโทรลเลอร์ MCS-51

(4) โหมด 3 ในการทำงานของรีจิสเตอร์ที่ใช้เป็นไทม์เมอร์หรือเคาน์เตอร์ในโหมด 3 ของไทม์เมอร์ 1 จะไม่มีการนับ ซึ่งมีผลเหมือนกับให้ค่าบิท TR1 = 0 แต่สำหรับไทม์เมอร์ 0 จะมีการทำงานดังต่อไปนี้

ไทม์เมอร์ 0 ในโหมด 3 จะบังคับให้รีจิสเตอร์ใช้งานเฉพาะ TL0 ของไทม์เมอร์ 0 ถูกใช้เป็นไทม์เมอร์หรือเคาน์เตอร์สำหรับบับจำนวนแมชชีนไซเกิลหรือจำนวนพัลส์ภายนอกขนาด 8 บิท โดยสามารถควบคุมการใช้งานรีจิสเตอร์ใช้งานเฉพาะ TL0 นี้ได้จากบิท C/T , Gate, TR0, INT0 และการเกิด โอเวอร์โฟลว์ของรีจิสเตอร์ TL0 จะมีผลไปเซตบิต TF0 ส่วนรีจิสเตอร์ใช้งานเฉพาะ TH0 ของไทม์เมอร์ 0 จะถูกบังคับให้ใช้งานเป็นไทม์เมอร์เพียงอย่างเดียว โดยสามารถควบคุมการทำงานได้จากบิท TR1 และ TF1 ของไทม์เมอร์ 1 และจะเซตบิท TF1 เมื่อเกิดโอเวอร์โฟลว์ นั่นคือ ขณะนี้รีจิสเตอร์ TH0 จะควบคุมการเกิดอินเตอร์รัปต์ของไทม์เมอร์ 1

6. พอร์ทสื่อสารข้อมูลแบบอนุกรมในไมโครคอนโทรลเลอร์ MCS-51

ไมโครคอนโทรลเลอร์ MCS-51 มีพอร์ทสำหรับสื่อสารข้อมูลแบบอนุกรม ที่สามารถรับและส่งข้อมูลแบบอนุกรม ได้โดยผู้ใช้ไม่จำเป็นต้องต่อชิพที่ทำหน้าที่รับหรือส่งข้อมูลแบบอนุกรมโดยเฉพาะเพิ่มแต่อย่างใดเลย การนำไมโครคอนโทรลเลอร์ MCS-51 ไปประยุกต์ใช้งานที่ต้องมีการติดต่อสื่อสารข้อมูลแบบอนุกรมกับวงจรภายนอกอื่นๆ จึงทำได้สะดวกและมีความคล่องตัวสูงมาก

พอร์ทสื่อสารข้อมูลแบบออนุกรมที่มีในไมโครคอนโทรลเลอร์ MCS-51 สามารถทำงานได้ในแบบ full duplex หมายความว่าไมโครคอนโทรลเลอร์ MCS-51 สามารถรับและส่งข้อมูลได้พร้อมๆ

กัน โดยในการรับข้อมูลจะมีการบัฟเฟอร์ ข้อมูลให้ด้วย จึงทำให้ไมโครคอนโทรลเลอร์ MCS-51 สามารถกำหนดการรับข้อมูลไบท์ที่สองซึ่งถูกส่งตามเข้ามาก่อนที่ไบท์แรกที่ได้รับเข้ามาจะถูกอ่านจากรีจิสเตอร์ใช้งานเฉพาะที่ใช้สำหรับข้อมูล (receive register) เพื่อนำไปเก็บไว้ในหน่วยความจำต่อไป

พอร์ทสื่อสารข้อมูลแบบอนุกรมในไมโครคอนโทรลเลอร์ MCS-51 ประกอบด้วยรีจิสเตอร์ขนาด 8 บิท จำนวนสองตัวแต่ละตัวมีชื่อเรียกตามหน้าที่ ดังนี้คือ

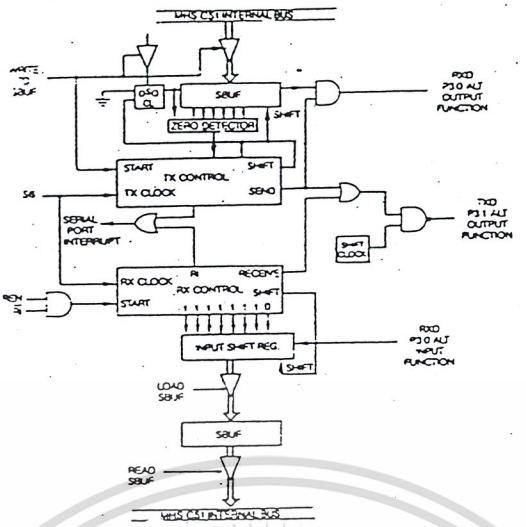
รีจิสเตอร์สำหรับข้อมูลใช้รับข้อมูลที่ส่งเข้ามาจากภายนอก

รีจิสเตอร์สำหรับส่งข้อมูล(transmit register)ใช้ส่งข้อมูลจากไมโครคอนโทรลเลอร์ MCS-51 ออกไปภายนอก

รีจิสเตอร์ทั้งสองมีตำแหน่งเดียวกันในรีจิสเตอร์ใช้งานเฉพาะ คือ ตรงกับตำแหน่งของรีจิสเตอร์ใช้งานเฉพาะ SBUF (ตำแหน่ง 99H) ในหน่วยความจำสำหรับเก็บข้อมูลภายในชิพที่ใช้เป็นรีจิสเตอร์ใช้งานเฉพาะ การเข้าถึงข้อมูลในรีจิสเตอร์แต่ละตัวไมโครคอนโทรลเลอร์ MCS-51 จะทราบเองว่าผู้ใช้ต้องการติดต่อกับบรีจิสเตอร์ตัวใดโดยตรวจสอบจากรหัสคำสั่ง ทั้งนี้เพราะในการเขียนข้อมูลไปไว้ในรีจิสเตอร์ใช้งานเฉพาะ SBUF หมายถึงการโหลดข้อมูลไปที่รีจิสเตอร์สำหรับส่งข้อมูลเพื่อส่งข้อมูลออกไปภายนอก ส่วนการอ่านข้อมูลจากรีจิสเตอร์ใช้งานเฉพาะ SBUF จะหมายถึงนำค่าที่รับเข้ามาได้จากภายนอก ที่เก็บไว้ในรีจิสเตอร์สำหรับรับข้อมูลมาใช้งาน

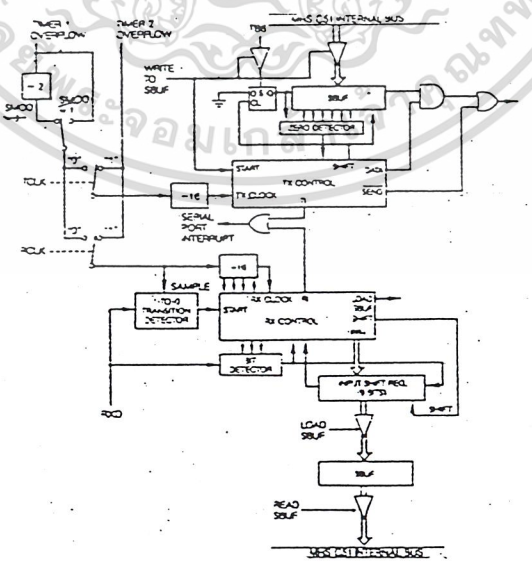
ผู้ใช้สามารถกำหนดการทำงานที่แตกต่างกันได้ถึง 4 ประเภท โดยสามารถกำหนดได้จากค่าของบิทในรรีจิสเตอร์ใช้งานเฉพาะ SCON การใช้งานที่แตกต่างกัน 4 ประเภทนี้มีจุดประสงค์เพื่อความคล่องตัวในการรับหรือส่งข้อมูลแบบอนุกรมแต่ละประเภทดังนี้

(1) โหมด 0 การทำงานของพอร์ทสื่อสารข้อมูลแบบอนุกรมในโหมด 0 ขา RXD จะใช้สำหรับรับและส่งข้อมูล ส่วนขา TXD มีไว้เพื่อใช้สร้างสัญญาณ shift clock เพื่อกำหนดจังหวะในการรับและส่งข้อมูล (ข้อมูลจะถูกรับหรือส่งตามจจังหวะของสัญญาณ shift clock) ในโหมดนี้การรับส่งข้อมูลจะเป็นแบบ 8 บิท (บิตข้อมูล 8 บิท) โดยเริ่มรับและส่งบิทต่ำสุดก่อน (LSB first) อัตราการรับส่งข้อมูลในการทำงานโหมด 0 ถูกกำหนดไว้ที่ 1/12 ของความถี่ออสซิลเล-เตอร์ที่ใช้ การทำงานของพอร์ทสื่อสารข้อมูลแบบอนุกรมในโหมด 0 จะไม่มีบิทเริ่มต้นของข้อมูล (start bit) และบิทสิ้นสุดของข้อมูล (stop bit) เพราะจังหวะการรับและส่งข้อมูลถูกกำหนดจากสัญญาณ shift clock แล้ว



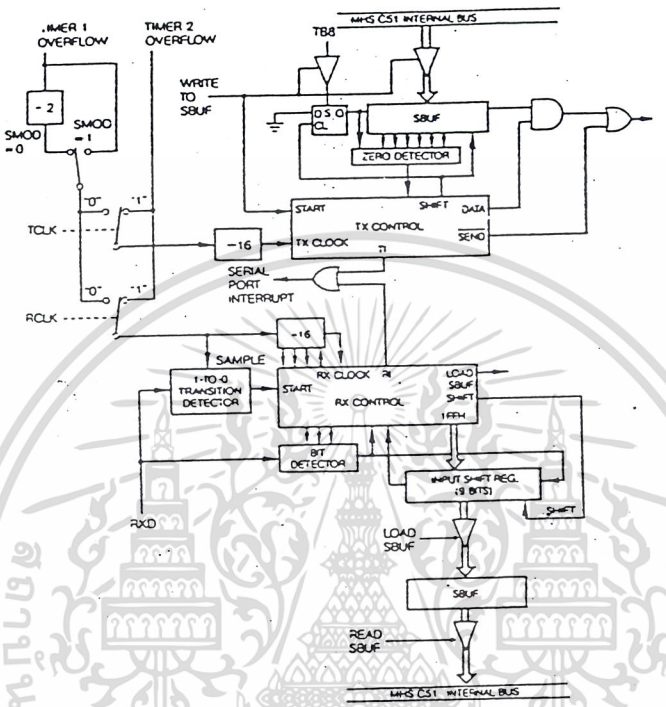
รูปที่ 2.26 การแสดงข้อมูลรับและส่งในการทำงานของพอร์ตสื่อสารอนุกรมโหมด 0

(2) โหมด 1 การทำงานแบบที่สองหรือการทำงานโหมด 1 นี้ มีการรับและส่งข้อมูลครั้งละ 10 บิท ข้อมูลจะถูกส่งออกไปภายนอกผ่านทางขา TXD และรับข้อมูลเข้ามาทางขา RXD ข้อมูลทั้ง 10 บิท ประกอบด้วยยบิทเริ่มต้นของข้อมูล 1 บิท (มีค่าเป็น 0 เสมอ) บิทข้อมูล 8 บิท (รับและส่งบิทต่ำสุดก่อน) และบิทสิ้นสุดของข้อมูลอีก 1 บิท (มีค่าเป็น 1 เสมอ) ในขณะทำการรับข้อมูล ค่าในบิทสิ้นสุดของข้อมูลที่รับได้จะไปอยู่ในบิท RB8 ของรีจิสเตอร์ใช้งานเฉพาะ SCON อัตราเร็วในการรับหรือส่งข้อมูลของพอร์ทสื่อสารข้อมูลแบบอนุกรมในโหมดนี้สามารถเปลี่ยนแปลงได้



รูปที่ 2.27 การแสดงข้อมูลรับและส่งในการทำงานของพอร์ตสื่อสารอนุกรมโหมด 1

(3) โหมด 2 การทำงานจะมีการรับและส่งข้อมูลครั้งละ 11 บิท ข้อมูลจะถูกส่งออกภายนอกผ่านทางขา TXD และรับเข้ามาผ่านทางขา RXD ข้อมูลที่รับและส่งทั้ง 11 บิทประกอบด้วยบิทเริ่มต้นของข้อมูล 1 บิท (มีค่า 0 เสมอ) บิทข้อมูล 8 บิท ตามด้วยบิทที่ 9 ซึ่งเป็นบิทที่สามารถกำหนดให้มีค่าเป็นศูนย์หรือหนึ่งได้และบิทสุดท้ายคือบิทสิ้นสุดของข้อมูล (มีค่าเป็น 1 เสมอ) ดังนั้นจำนวนบิทที่รับส่งทั้งหมด 11 บิท จะประกอบด้วยบิทต่างๆ ดังนี้



รูปที่ 2.28 การแสดงข้อมูลรับและส่งในการทำงานของพอร์ตสื่อสารอนุกรมโหมด 2,3

ในขณะทำการส่งข้อมูล บิทที่ 9 จะได้จากค่าในบิท TBB ของรีจิสเตอร์ใช้งานเฉพาะ SCON บิทนี้สามารถถูกกำหนดให้มีค่าเป็น 0 หรือ 1 อย่างไรก็ได้ ส่วนใหญ่ในการใช้งานจริงมักจะใช้บิทนี้สำหรับตรวจสอบความถูกต้องของข้อมูลที่รับหรือส่ง (parity bit) โดยจะนำบิท P (parity) ในรีจิสเตอร์ PSW ไปไว้ในบิท TB8 ส่วนในขณะรับข้อมูลบิทที่ 9 จะไปปรากฏอยู่ในบิท RB8 ของรีจิสเตอร์ SCON โดยไม่สนใจบิทสิ้นสุดของข้อมูลค่าอัตราเร็วในการรับหรือส่งข้อมูลโหมดนี้ถูกกำหนดไว้ที่ 1/32 หรือ 1/64 ของความถี่ออสซิลเลเตอร์ที่ใช้

(4) โหมด 3 การทำงานของพอร์ทสื่อสารข้อมูลแบบอนุกรมแบบสุดท้าย คือการทำงานในโหมด 3 ในการทำงานโหมดนี้ข้อมูลจำนวน 11 บิทถูกส่งผ่านขา TXD และถูกรับเข้ามาทางขา RXD ข้อมูลทั้ง 11 บิทประกอบด้วยบิทเริ่มต้นของข้อมูล 1 บิท (เป็น 0 เสมอ) บิทข้อมูล 8 บิท (รับและส่งบิทต่ำสุดก่อน) ตามด้วยบิทที่ 9 ซึ่งเป็นบิทที่สามารถกำหนดค่าได้เหมือนในโหมด 2 (programmable 9^{th} bit) และบิทสุดท้ายคือบิทสิ้นสุดของข้อมูล (เป็น 1 เสมอ) อัตราเร็วในการรับหรือส่งข้อมูลสามารถเปลี่ยนแปลงได้ดังจะศึกษาในรายละเอียดต่อไป ดังนั้นจะเห็นว่ารูปแบบการรับส่งข้อมูลใน

โหมด 2 ทุกอย่าง แต่ในโหมดนี้สามารถกำหนดค่าอัตราเร็วในการรับหรือส่งข้อมูลได้ตามความต้องการของผู้ใช้

การทำงานของพอร์ทสื่อสารข้อมูลแบบอนุกรมทั้ง 4 โหมดที่กล่าวมานี้

การส่งข้อมูลจะเริ่มทันทีเมื่อมีคำสั่งใดๆ ที่ใช้รีจิสเตอร์ใช้งานเฉพาะ SBUF เป็นรีจิสเตอร์ปลายทาง (destination register) เช่น

ส่วนในการรับข้อมูลจะเริ่มต้นโดยมีเงื่อนไขดังนี้

-ในโหมด 0 เริ่มเมื่อค่าในบิท R1 = 0 และบิท REN = 1

-ในโหมดอื่นๆ การรับข้อมูลเริ่มเมื่อไมโครคอนโทรลเลอร์ MCS-51 ได้รับบิทเริ่มต้นของข้อมูลเข้ามา โดยที่บิท REN ในขณะนั้นต้องมีค่า 1

7. อัตราเร็วในการรับและส่งข้อมูล

baud rate หมายความถึงอัตราเร็วในการรับหรือส่งข้อมูล โดยในไมโคร-คอนโทรลเลอร์ MCS-51 ค่าอัตราเร็วในการรับและส่งข้อมูลจะมีค่าเท่าใด ก็ขึ้นอยู่กับ
การทำงานในแต่ละโหมดของพอร์ทสื่อสารข้อมูลแบบอนุกรม ดังนี้

$$\text{baud rate โหมด 0} = \frac{\text{ความถี่ออสซิลเลเตอร์ที่ใช้}}{12}$$

หากใช้คริสตอลความถี่ 12 เมกะเฮิรตซ์ ค่า baud rate ของพอร์ทสื่อสาร ข้อมูลแบบอนุกรมในโหมด 0 จะมีค่าสูงถึง 1 เมกะเฮิรตซ์

ในโหมด 2 ค่า baud rate ขึ้นอยู่กับค่าของบิท SMOD ที่อยู่ในรีจิสเตอร์ใช้งานเฉพาะ PCON โดย

บิทSMOD = 0 ค่า baud rate จะเป็น 1/64 ของความถี่ออสซิลเลเตอร์ที่ใช้

บิทSMOD = 1 ค่า baud rate จะเป็น 1/32 ของความถี่ออสซิลเลเตอร์ที่ใช้

หลังจากการรีเซตไมโครคอนโทรลเลอร์ MCS-51 ค่าในบิท SMOD จะเป็น 0 เสมอ และเราสามารถเขียนสูตรสำหรับคำนวณค่า baud rate ได้ดังสมการนี้

$$\text{baud rate โหมด 2} = \frac{[2^{(\text{SMOD})} \times (\text{ความถี่ออสซิลเลเตอร์})]}{64}$$

เราสามารถที่จะสร้าง baud rate ค่าต่ำๆ ด้วยไทม์เมอร์ 1 ได้โดยปล่อยให้ไทม์เมอร์ 1 อินเตอร์รัพท์ซีพียูได้ และกำหนดการทำงานให้เป็นไทม์เมอร์ขนาด 16 บิท (โหมด 1) และใช้ไทม์เมอร์ 1 อินเตอร์รัพท์ซีพียูเพื่อโหลดค่าใหม่เองด้วยซอฟต์แวร์ขณะเกิดโอเวอร์โฟลว์ เนื่องจากในการทำงานโหมด 1 ของไทม์เมอร์ 1 ไม่สามารถโหลดค่าใหม่เองด้วยฮาร์ดแวร์ได้ (ไม่สามารถทำงานแบบ Auto-Reload)

ตารางที่ 2.5 ค่าที่นำไปไว้ในรีจิสเตอร์ของไทม์เมอร์ 1 เมื่อใช้ baud rate ค่ามาตราฐานต่างๆ

Baud Rate	Fosc	SMOD	TIMER 1		
			C/t	MODE	Reload Value
(MODE0) Max : 1 MHz	12 MHz	X	X	X	X
(MODE2) Max : 375	12 MHz	1	X	X	X
KHz	12 MHz	0	X	X	X
(MODE2) Min	12 MHz	1	0	2	FFH
:187.5KHz	11.059MHz	1	0	2	FDH
MODE 1,3 : 62.5	11.059MHz	0	0	2	FDH
kHz	11.059 MHz	0	0	2	FAH
19.2KHz	11.059MHz	0	0	2	F4H
9.6KHz	11.059MHz	0	0	2	E8H
4.8KHz	11.059MHz	0	0	2	1DH
2.4KHz	6 MHz	0	0	2	72H
1.2KHz	12 MHz	0	0	1	FEEBH
137.5					
110					
110					

จากตารางที่ 2.5 จะเห็นว่าในการทำงานของพอร์ทสื่อสารข้อมูลแบบอนุกรมโหมด 0 จะมีความเร็วในการส่งมากที่สุดเมื่อเปรียบเทียบกับโหมดอื่นที่ความถี่คริสตอลค่าเดียวกัน และจะเห็นว่าหากเลือกใช้คริสตอลความถี่ 11.059 เมกะเฮิรตซ์ จะสามารถตั้งค่า baud rate ในโหมด 1 และ 3 เป็นค่ามาตราฐานที่ใช้กันทั่วไปได้เช่น 1200, 2400, 4800, 9600, 19200 จึงเป็นเหตุผลสำคัญที่ในระบบควบคุมส่วนใหญ่เลือกใช้คริสตอลความถี่ 11.059 เมกะเฮิรตซ์มากกว่า 12 เมกะเฮิรตซ์

ในตารางที่ 2.5 นอกจากจะแสดงค่า baud rate ค่าต่างๆ เปรียบเทียบให้เห็นแล้ว ตารางนี้ยังแสดงค่าที่ต้องโหลดไปไว้ในรีจิสเตอร์ใช้งานเฉพาะ TH1 ที่ค่า baud rate มาตรฐานต่างๆ ให้ทราบอีกด้วย ผู้เขียนโปรแกรมสามารถนำค่านี้ไปใช้ได้เลย แต่หากต้องการใช้ค่า baud rate อื่นๆ ที่นอกเหนือไปจากตารางนี้ จะต้องคำนวณค่าที่ต้องโหลดให้รีจิสเตอร์ใช้งานเฉพาะ TH1 จากสมการที่แสดงไปแล้วเอง

8. โครงสร้างการอินเตอร์รัพท์ไมโครคอนโทรลเลอร์ MCS-51

ไมโครคอนโทรลเลอร์ MCS-51 สามารถรับสัญญาณอินเตอร์รัพท์ที่เกิดขึ้นได้อย่างน้อย 5 ชนิดด้วยกัน (ไมโครคอนโทรลเลอร์ MCS-51 บางเบอร์ในตระกูลนี้สามารถรับสัญญาณอินเตอร์รัพท์ได้มากกว่า 5 ชนิด เช่น เบอร์ 8052 สามารถรับได้ 6 ชนิด) แหล่งกำเนิดสัญญาณอินเตอร์รัพท์ทั้ง 5 ชนิดที่ไมโครคอนโทรลเลอร์ MCS-51 สามารถรับได้มีดังแสดงในรูปที่ 2.29

ลำดับ	ชื่อสัญญาณอินเตอร์รัปต์	Vector Address	Priority
1	INTO	0003H	Highest
2	TF0	000BH	
3	INT1	00013H	
4	TF1	001BH	
5	TL,RI	0023H	Lowest

รูปที่ 2.29 แหล่งกำเนิดสัญญาณอินเตอร์รัฟท์ทั้ง 5 ชนิด

อินเตอร์รัพท์แต่ละชนิดที่ไมโครคอนโทรลเลอร์ MCS-51 สามารถรับได้มีรายละเอียดดังต่อไปนี้

8.1 อินเตอร์รัพท์ที่เกิดจากภายนอก (External Interrupts) เป็นอินเตอร์รัพท์เกิดขึ้นจากภายนอก ไมโครคอนโทรลเลอร์ MCS-51 มี 2 ชนิดด้วยกันคือ

- อินเตอร์รัพท์ภายนอกชนิด 0 รับได้จากขา INT0
- อินเตอร์รัพท์ภายนอกชนิด 1 รับได้จากขา INT1

ผู้ใช้สามารถกำหนดให้ไมโครคอนโทรลเลอร์ MCS-51 ตรวจสอบสัญญาณอินเตอร์รัพท์ทั้งสองชนิดที่เกิดขึ้นที่ขา INT0, INT1 2 แบบด้วยกัน คือ

- ตรวจสอบจากระดับสัญญาณ (level-activated)
- ตรวจสอบจากการเปลี่ยนสถานะสัญญาณ (transition-activated)

การตรวจสอบสถานะของสัญญาณอินเตอร์รัพท์ภายนอกที่ขาทั้งสอง สามารถเลือกได้เพียงอย่างใดอย่างหนึ่งขึ้นอยู่กับการกำหนดค่าบิท IT0, IT1 ในรีจิสเตอร์ใช้งานเฉพาะ TCON ดังแสดงในรูปที่ 2.18 ในหัวข้อไทม์เมอร์

เมื่อไมโครคอนโทรลเลอร์ MCS-51 ตรวจพบสัญญาณอินเตอร์รัพท์จากภายนอก จะมีผลทำให้บิท IE0 (จากขา INT0) หรือ IE1 (จากขา INT1) ของรีจิสเตอร์ใช้งานเฉพาะ TCON ถูกเซต บิททั้งสองจะเป็นตัวบอกสถานะของสัญญาณอินเตอร์รัพท์ที่เกิดจากภายนอก โดยถูกเซตเมื่อเกิดสัญญาณอินเตอร์รัพท์และจะถูกเคลียร์โดยฮาร์ดแวร์ภายใน MCS-51 เอง เมื่อซีพียูย้ายไปทำงานที่โปรแกรมบริการอินเตอร์รัพท์ต่อเมื่อ สัญญาณอินเตอร์รัพท์ภายนอกที่เกิดขึ้นเป็นชนิดที่ตรวจสอบได้ จากการเปลี่ยนสถานะสัญญาณแต่ถ้าสัญญาณอินเตอร์รัพท์ ภายนอกที่เกิดขึ้นได้มาจากการตรวจสอบระดับสัญญาณ เมื่อซีพียูย้ายไปทำงานที่โปรแกรมบริการอินเตอร์รัพท์ จะไม่มีการเคลียร์บิท IE0 หรือ IE1 ให้ในกรณีนี้จะเป็นหน้าที่ของวงจรกำเนิดสัญญาณอินเตอร์รัพท์ภายนอกที่จะต้องทำหน้าที่ควบคุมสถานะของสัญญาณที่ขา INTx (INT0 หรือ INT1) ให้กลับสู่สภาพเดิมเอง มิฉะนั้นโปรแกรมหลักที่ทำงานอยู่จะถูกอินเตอร์รัพท์ไปเรื่อยๆ จนกระทั่งสัญญาณอินเตอร์รัพท์กลับมีค่าเป็น 1 อีกครั้ง

8.2 อินเตอร์รัพท์ของไทม์เมอร์ 0 และไทม์เมอร์ 1 อินเตอร์รัพท์ของไทม์เมอร์ 0 หรือไทม์เมอร์ 1 ถูกทำให้เกิดขึ้นโดยบิท TF0 หรือ TF1 ซึ่งถูกเซตเมื่อไทม์เมอร์ 0 หรือไทม์เมอร์หรือเคาน์เตอร์ของไทม์เมอร์ 0 หรือไทม์เมอร์ 1) ยกเว้นไทม์เมอร์ 0 ในโหมด 3 ซึ่งหยุดการทำงานเมื่อมีอินเตอร์รัพท์จากไทม์เมอร์เกิดขึ้น บิท TF0 และ TF1 จะถูกเคลียร์โดยฮาร์ดแวร์ภายในไมโครคอนโทรลเลอร์ MCS-51 เอง เมื่อซีพียูย้ายไปทำงานที่โปรแกรมบริการอินเตอร์รัพท์

8.3 อินเตอร์รัพท์ของพอร์ตสื่อสารอนุกรม (Serial Port Interrupt) พอร์ทสื่อสารอนุกรมของไมโครคอนโทรลเลอร์ MCS-51 สามารถทำให้เกิดสัญญาณอินเตอร์รัพท์ได้ สัญญาณอินเตอร์รัพท์ที่เกิดขึ้นได้มาจากบิท T1 หรือ R1 ที่นำมาผ่านเกตออร์ (ดังแสดงในรูปที่ 2.24) และบิทที่ควบคุมการอินเตอร์รัพท์ทั้งสองนี้จะไม่ถูกเคลียร์โดยฮาร์ดแวร์ในไมโครคอนโทรลเลอร์ MCS-51 เมื่อซีพียุไปทำงานในโปรแกรมบริการอินเตอร์รัพท์เพราะการเกิดอินเตอร์รัพท์ของพอร์ทสื่อสารอนุกรมอาจจะเกิดจากบิท RI หรือ TI ก็ได้ ดังนั้นโปรแกรมในส่วนบริการอินเตอร์รัพท์จะต้องตรวจสอบเองว่าสัญญาณอินเตอร์รัพท์ที่เกิดขึ้นได้มาจากบิท RI หรือ TI และบิททั้งสองจะถูกเคลียร์โดยซอฟต์แวร์เท่านั้น

8.4 อินเตอร์รัพท์ของไทม์เมอร์ 2 (Timer 2 Interrupt) ในเบอร์ 8052 จะมี รีจิสเตอร์สำหรับใช้เป็นไทม์เมอร์หรือเคาน์เตอร์เพิ่มขึ้นมาอีก 1 ตัวคือไทม์เมอร์ 2 โดยสามารถใช้สร้างสัญญาณอินเตอร์รัพท์ ได้เหมือนเช่นในไทม์เมอร์ 0 และไทม์เมอร์ 1 แต่การเกิดอินเตอร์รัปต์ของไทม์เมอร์ 2 จะแตกต่างออกไปจากไทม์เมอร์ 0 และไทม์เมอร์ 1 ดังนี้

อินเตอร์รัพท์ของไทม์เมอร์ 2 เกิดขึ้นโดยการนำบิท TF2 และ EXF2 มาผ่านเกตออร์ โดยบิท TF2 และ EXF2 ทั้งสองจะไม่ถูกเคลียร์โดยฮาร์ดแวร์เมื่อซีพียูไปทำงานในส่วนโปรแกรมบริการอินเตอร์รัพท์เหมือนในการเกิดอินเตอร์รัพท์ของพอร์ทสื่อสารอนุกรม ดังนั้นในโปรมแกรม

บริการอินเตอร์รัพท์จะต้องตรวจสอบเองว่าบิท TF2 และ EXF2 ที่เป็นสาเหตุทำให้เกิดการอินเตอร์รัพท์และบิทที่ทำให้เกิดอินเตอร์รัพท์จะต้องถูกเคลียร์โดยซอฟต์แวร์เองด้วย

อินเตอร์รัพท์แต่ละชนิดที่กล่าวไปแล้ว สามารถถูกควบคุมให้สามารถอินเตอร์รัพท์ไมโครคอนโทรลเลอร์ MCS-51 ได้หรือไม่ โดยการควบคุมจากบิทต่างๆ ในรีจิสเตอร์ใช้งานเฉพาะ IE ดังแสดงในรูปที่ 31

EA	X	ET2	ES	ET1	EX1	ET0	EX0

Symbol	Position	Function
EA	IE.7	disables all interrupts. If EA=0,no interrupt will be acknowledged .If EA=1,each in interrupt source is individually enabled all disabled by setting or clearing its enable bit.
-	IE.6	reserved
ET2	IE.5	enable or disables the Timer 2 overflow or capture interrupt .If ET2=0, the Timer2 interrupt is disabled.
ES	IE.4	enables or disables the serial Port interrupt.If ES=0, the serial port interrupt is disabled.
ET1	IE.3	enables or disables the timer overflow interrupt. 1f ET1=0,the Timer 1 interrupt is disabled.
EX1	IE.2	enables or disables External Interrupt 1 . If EX=0,External Interrupt1 is disabled
ET0	IE.1	enables or disables the timer overflow interrupt. If ET0 = 0,the Timer 0 interrupt is disabled.
EX0	IE.0	eables or disables External Interrupt 0 . If EX=0,External Interrupt1 is disabled

รูปที่ 2.30 รีจิสเตอร์ใช้ในงานเฉพาะ IE

บิท	ชื่อบิท	
IE.7	EA	ใช้ควบคุมการตอบสนองต่อสัญญาณอินเตอร์รัพท์ทั้งหมด 0: MCS-51 จะไม่ตอบสนองต่อสนองต่อสัญญาณอินเตอร์รัพท์ใดๆ ทั้งสิ้น 1: อินเตอร์รัพท์แต่ละชนิดจะถูกควบคุมการตอบสนองอย่างอิสระจากบิทในรีจิสเตอร์นี้

IE.6		ไม่ถูกกำหนดการใช้งาน
IE.5	ET2	ควบคุมการตอบสนองต่อสัญญาณอินเตอร์รัพท์ของไทม์เมอร์ 2 เมื่อเกิดโอเวอโฟลว์
IE.4	ES	ควบคุมการตอบสนองต่อสัญญาณอินเตอร์รัพท์ของพอร์ทสื่อสารอนุกรม
IE.3	ET1	ควบคุมการตอบสนองต่อสัญญาณอินเตอร์รัพท์ของไทม์เมอร์ 1 เมื่อเกิดโอเวอร์โฟลว์
IE.2	EX	ควบคุมการตอบสนองต่อสัญยาณอินเตอร์รัพท์ภายนอกชนิด 1
IE.1	ET0	ควบคุมการตอบสนองต่อสัญญาณอินเตอร์รัพท์ของไทม์เมอร์ 0 เมื่อเกิดโอเวอร์โฟลว์
IE.0	EX0	ควบคุมการตอบสนองต่อสัญญาณอินเตอร์รัพท์ภายนอกชนิด 0

การกำหนดให้บิทควบคุมการตอบสนองต่อสัญญาณอินเตอร์รัพท์แต่ละชนิดมีค่าเป็น 0 หมายถึงไม่ให้ MCS-51 ตอบสนองต่อสัญญาณอินเตอร์รัพท์ชนิดนั้น หากกำหนดให้บิทควบคุมการตอบสนองต่อสัญญาณอินเตอร์รัพท์แต่ละชนิดมีค่าเป็น 1 หมายถึงให้ไมโครคอนโทรลเลอร์ MCS-51 ตอบสนองต่อสัญญาณอินเตอร์รัพท์ชนิดนั้น (บิท EA ต้องถูกเซตไว้ก่อนด้วย)

บิท EA ในรีจิสเตอร์ใช้งานเฉพาะ IE สามารถควบคุมการอินเตอร์รัพท์ในไมโครคอนโทรลเลอร์ MCS-51 ได้ทั้งหมดหากบิทนี้มีค่าเป็น 0 สัญญาณอินเตอร์รัพท์ทุกชนิดที่เกิดขึ้นจะไม่สามารถอินเตอร์รัพท์ไมโครคอนโทรลเลอร์ MCS-51 ได้ แต่หากบิทนี้มีค่าเป็น 1 สัญญาณอินเตอร์รัพท์แต่ละชนิดจะถูกควบคุมให้อินเตอร์รัพท์ไมโครคอนโทรลเลอร์ MCS-51 ได้อย่างอิสระ (ควบคุมจากบิท IE.0-IE05)

บิท IE05-IE.6 ไม่ถูกใช้ใน 8051 เพราะถูกสงวนไว้ในไมโครคอนโทรลเลอร์ MCS-51 เบอร์อื่นๆ ที่สามารถรับอินเตอร์รัพท์ได้เพิ่มขึ้น ดังนั้นซอฟต์แวร์ของผู้ใช้ไม่ควรจะมีคำสั่งเขียนค่า 1 ลงไปในบิทเหล่านี้เพื่อให้โปรแกรมยังคงสามารถใช้กับชิพเบอร์ใหม่ๆ ในตระกูลนี้ได้

9. โครงสร้างระดับความสำคัญในการบริการอินเตอร์รัพท์ (Priority Level Structure)

อินเตอร์รัพท์แต่ละชนิดสามารถถูกเลือกระดับความสำคัญในการบริการได้ 2 ระดับ โดยการเซตหรือเคลียร์บิทในรีจิสเตอร์ใช้งานเฉพาะ IP ดังแสดงในรูปที่ 32

รีจิสเตอร์ใช้งานเฉพาะ IP (Interrupt Priority Register) เข้าถึงข้อมูลได้ในระดับบิต

PCT	X	PT2	PS	PT1	PX1	PT0	PX0

Symbol Position Funtion

PCT	IP.7	83C154/C154D only. Priority interrupt circuit control bit. The prority register contents are valid priority assigned Interrupt can be processed when this bit is "0" .When the bit is "1",the priority Interrupt circuit is stopd. And interrupt can only be controlled by the interrupt enabie register (IE).
-	IP.6	reserved
PT2	IP.5	defines the timer 2 interrupt priority level. PT2=1 program it to higher priority level.
PS	IP.4	defines the Serial port interrupt priority level. PS=1 program it to higher priority level.
PT1	IP.3	defines the timer 1 interrupt priority level. PT1=1 program it to higher priority level.
PX1	IP.2	defines the External interrupt 1 priority level. PX1=1 program it to higher priority level.
PT0	IP.1	defines the timer 0 interrupt priority level. PT0=1 program it to higher priority level .
PX0	IP.0	defines the External interrupt 0 priority level. PX0=1 program it to higher priority level

รูปที่ 2.31 รีจิสเตอร์ใช้ในงานเฉพาะ IP

บิท	ชื่อบิท	
IP.7	-	ไม่ถูกกำหนดการใช้งาน (สำรองไว้ใช้ในไมโครคอนโทรลเลอร์ MCS-51 เบอร์ใหม่ๆ ในอนาคต)
IP.6	-	ไม่ถูกกำหนดการใช้งาน (สำรองไว้ใช้ในไมโครคอนโทรลเลอร์ MCS-51 เบอร์ใหม่ๆ ในอนาคต)
IP.5	PT2	กำหนดลำดับความสำคัญในการตอบสนองต่อสัญญาณอินเตอร์รัพท์ของไทม์เมอร์ 2
IP.4	PS	กำหนดลำดับความสำคัญในการตอบสนองต่อสัญญาณอินเตอร์รัพท์ของพอร์ทสื่อสารอนุกรม
IP.3	PT1	กำหนดลำดับความสำคัญในการตอบสนองต่อสัญญาณอินเตอร์รัพท์ของไทม์เมอร์ 1

IP.2	PX1	กำหนดลำดับความสำคัญในการตอบสนองต่อสัญญาณอินเตอร์รัพท์ภายนอกชนิด 1
IP.1	PT0	กำหนดลำดับความสำคัญในการตอบสนองต่อสัญญาณอินเตอร์รัพท์ของไทม์เมอร์ 0
IP.0	PX0	กำหนดลำดับความสำคัญในการตอบสนองต่อสัญญาณอินเตอร์รัพท์ภายนอกชนิด 0

การให้บิทกำหนดลำดับความสำคัญของอินเตอร์รัพท์เป็น 0 หมายถึงให้อินเตอร์รัพท์ชนิดนั้นมีลำดับความสำคัญต่ำ ส่วนการให้บิทกำหนดลำดับความสำคัญของอินเตอร์รัพท์เป็น 1 หมายถึงให้อินเตอร์รัพท์ชนิดนั้นมีลำดับความสำคัญสูง

ระดับความสำคัญในการบริการอินเตอร์รัพท์ที่เกิดขึ้นมีได้ 2 ระดับ ได้แก่

9.1 อินเตอร์รัพท์ระดับความสำคัญต่ำ (Low Priority Interruput) : อินเตอร์รัพท์ชนิดนี้สามารถถูกอินเตอร์รัพท์จากสัญญาณอินเตอร์รัพท์ระดับความสำคัญสูงได้ แต่จะไม่สามารถถูกอินเตอร์รัพท์โดยสัญญาณอินเตอร์รัพท์ระดับความสำคัญต่ำตัวอื่นๆ ได้

9.2 อินเตอร์รัพท์ระดับความสำคัญสูง (High Priority Interrupt) : อินเตอร์รัพท์ประเภทนี้ไม่สามารถถูกอินเตอร์รัพท์โดยสัญญาณอินเตอร์รัพท์ชนิดอื่นๆ ได้เลย นั่นคือมีระดับความสำคัญสูงสุด

ถ้ามีสัญญาณอินเตอร์รัพท์เกิดขึ้นพร้อมกัน 2 ชนิด โดยมีระดับความสำคัญในการบริการอินเตอร์รัพท์ไม่เท่ากัน สัญญาณอินเตอร์รัพท์ที่มีระดับความสำคัญสูงกว่าจะได้รับการบริการก่อน แต่ถ้ามีการขออินเตอร์รัพท์พร้อมกัน 2 ชนิด ซึ่งมีระดับความสำคัญเท่าเทียมกัน ลำดับการบริการอินเตอร์รัพท์ภายในจะเป็นตัวกำหนดเองว่าอินเตอร์รัพท์ชนิดใดควรจะถูกบริการก่อน ดังนั้นภายในระดับความสำคัญของการบริการอินเตอร์รัพท์หนึ่งๆ จะมีระดับความสำคัญในการบริการอินเตอร์รัพท์ย่อยลงไปอีกหนึ่งระดับ

การจัดการกับสัญญาณอินเตอร์รัพท์ที่มีระดับความสำคัญเท่าเทียมกันที่เกิดขึ้นพร้อมกันถูกใช้เพียงเพื่อแก้ปัญหาสัญญาณอินเตอร์รัพท์ที่มีระดับความสำคัญเท่ากันที่เกิดขึ้นพร้อมกันเท่านั้น

รีจิสเตอร์ใช้งานเฉพาะ IP มีบิทที่ไม่ถูกใช้งานอยู่บางบิท คือ IP.7 และ IP.6 โดยไม่ถูกใช้ในไมโครคอนโทรลเลอร์ 8051 และ 8052 ในไมโครคอนโทรลเลอร์ 8051 จะมีบิทที่ว่างเพิ่มมาอีก 1 บิทคือ IP.5 ดังนั้นซอฟต์แวร์ของผู้ใช้ไม่ควรมีการเขียนค่า 1 ไปที่ตำแหน่งบิทเหล่านี้เพราะมันอาจถูกนำไปใช้ในไมโครคอนโทรลเลอร์ MCS-51 เบอร์ใหม่ๆ ในอนาคตต่อไป

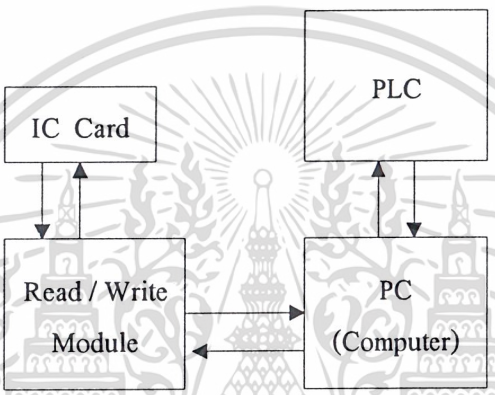
บทที่ 3

การออกแบบ Hardware และ Software

แนวทางการออกแบบและการสร้างเครื่องเราสามารถแยกออกแบบได้ 2 ส่วน ซึ่งแต่ละส่วนมีการทำงานร่วมกันโดยสามารถแบ่งได้ดังนี้

3.1 การออกแบบด้าน Hardware

เพื่อให้เข้าใจง่ายในการเข้าใจถึงตัวอุปกรณ์สามารถแสดงเป็นรูปได้ดังนี้

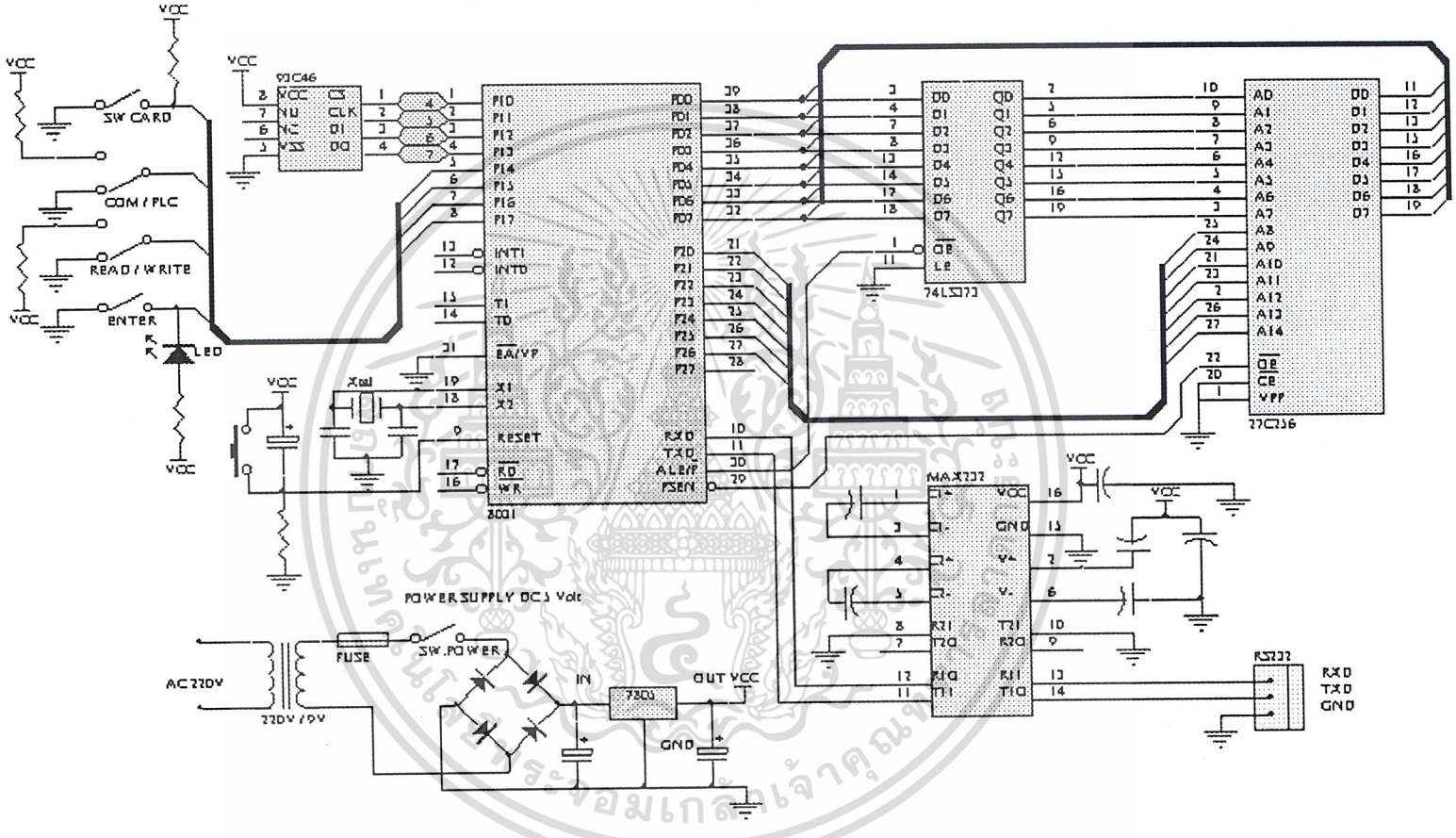


รูปที่ 3.1 แสดง Block Diagram ของเครื่องอ่านข้อมูลจาก IC Card

3.1.1 การออกแบบวงจรที่ใช้เป็น Module

วงจรที่ออกแบบจะใช้ไมโครคอนโทรลเลอร์เบอร์ 8031 เป็นตัวประมวลผลซึ่งเป็นไอซีที่มีขนาด 40 ขา ราคาถูกมีขายอยู่ทั่วไปทั้งคู่มือและอุปกรณ์เสริมอื่น ๆ และใช้ EPROM เบอร์ 27C256 เป็นหน่วยความจำภายนอก ซึ่งมีความจุขนาด 32 K โดยมี IC 74LS373 ทำหน้าที่ Latch ตำแหน่งในหน่วยความจำ 8 บิทล่างของ EPROM พอร์ทอนุกรมใช้ IC MAX232 เป็นตัว Convert แรงดันของ IC 8031 เป็นระดับแรงดันในมาตรฐาน RS232

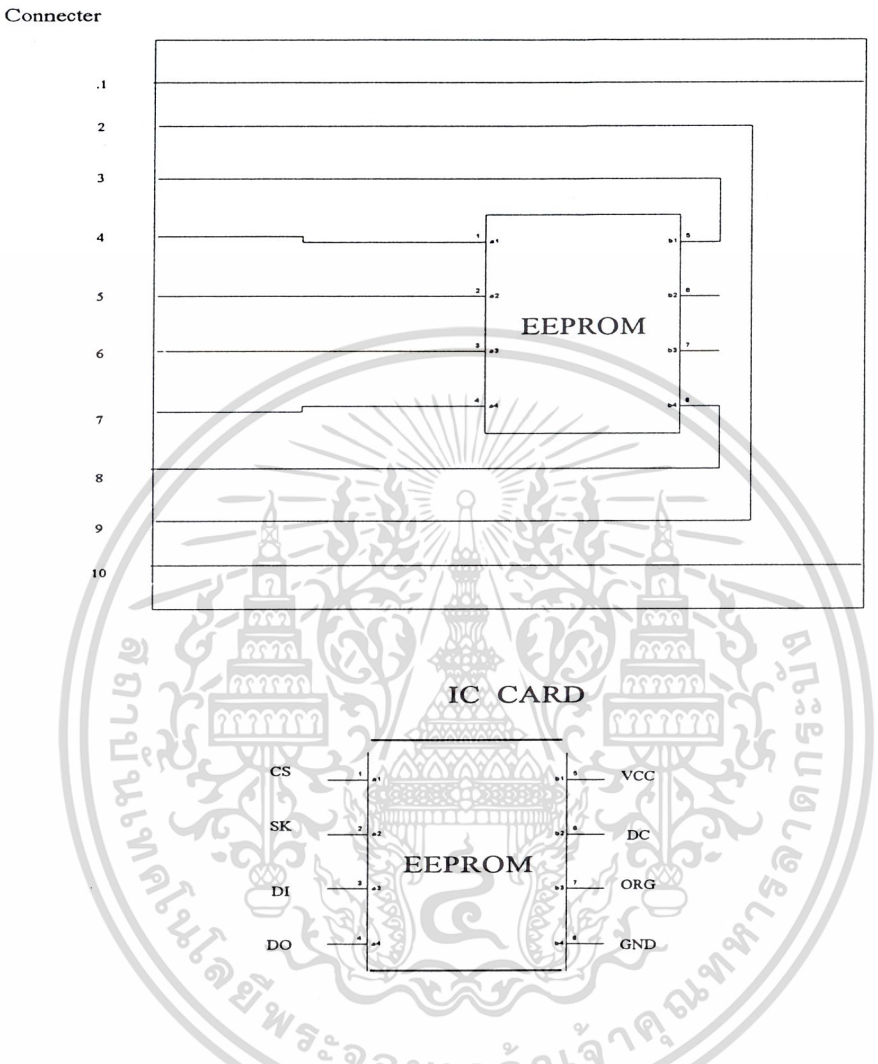
วงจรภาคจ่ายไฟนั้น INPUT 220v. ผ่านหม้อแปลง 220 / 9 เข้าวงจร Bridge เพื่อแปลงเป็นไฟกระแสตรง จากนั้นผ่าน IC 7805 ซึ่งเป็น IC Regulate จะได้ไฟ 9 โวล์ท และกรองกระแสให้เรียบขึ้นด้วย Capacitor 2 ตัว ตามรูปที่ 3.2



รูปที่ 3.2 แสดงภาพวงจรทั้งหมดของเครื่อง

3.1.2 การออกแบบและอ่าน / เขียนข้อมูลลงใน EEPROM

การออกแบบ จะทำการออกแบบบัตรโดยใช้ EEPROM เบอร์ 93C46 ซึ่งทำวงจรดังรูปที่ 3.3



รูปที่ 3.3 แสดงตำแหน่งสัญญาณขา IC Card

การเขียนข้อมูลลงใน EEPROM สามารถทำได้ตามขั้นตอนดังนี้คือ

1. Enable write protect
2. ทำการลบข้อมูลที่มีอยู่เดิม
3. เขียนขอมูลโดยจะเขียนตำแหน่งละ 16 บิท
4. Disable write protect

3.2 การออกแบบด้าน Soft Ware

เป็นการเขียนโปรแกรมการควบคุมการทำงานโดยแบ่งเป็นสองส่วนคือ

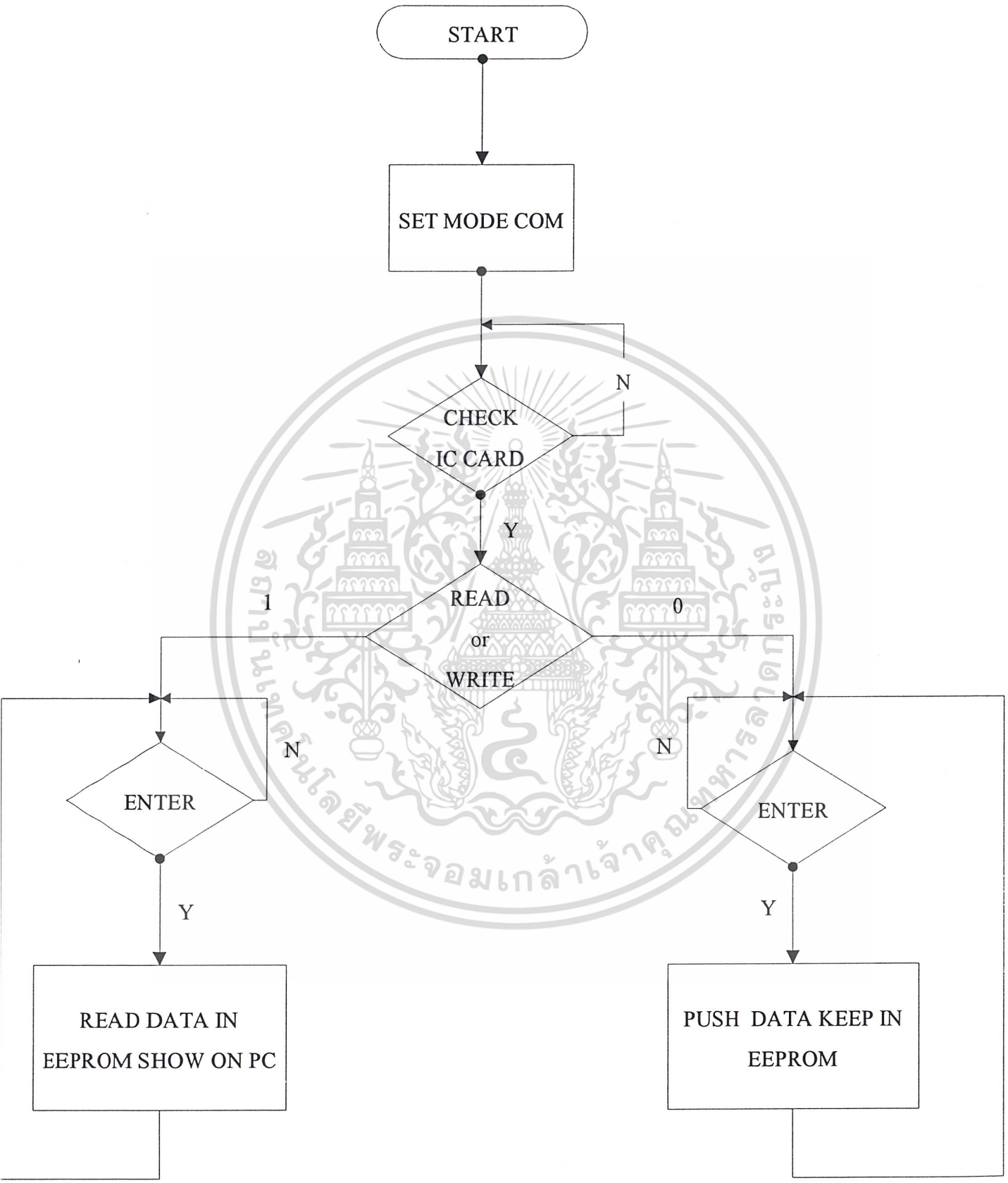
3.2.1 การออกแบบ soft ware ส่วนที่จะโปรแกรมลงใน EPROM ของ Module

ภาษาที่ใช้ในการเขียนโปรแกรมเป็นภาษา C51 ของ Frankin โดยจะมีคุณสมบัติที่ต่างจากภาษา C ธรรมดาคือ มีคำสั่งที่สามารถสั่งการไปยัง MCS-51 ได้ รูปแบบการใช้งานทั่วไปจะเหมือนกับ ANSI C แต่จะมีคำสั่งเพิ่มเติมในเรื่องของการอ้างอิง Address ของ Memory ใน MCS-51 โดยทั่งไปจะมีคำสั่ง sbit , xbyte เป็นคำสั่งในการเข้าถึง External Memory เมื่อเขียนโปรแกรมเสร็จแล้วจะได้เป็น File.C จากนั้นทำการคอมไพล์ดังนี้

- C51 File.C
- Bl51 File.OBJ
- OH51 File โปรแกรมจะสร้าง File.HEX ขึ้นมาโดยเป็นแบบ INTEL-HEX FILE

เมื่อได้ File.HEX แล้วสามารถทำการลองการทำงานด้วยโปรแกรม dScope จากนั้นต้องนำ File.HEX ที่ทดลองใช้งานได้แล้วไปอัดลงใน EPROM ซึ่งทำหน้าที่เป็นหน่วยความจำโปรแกรมภายนอกให้กับไมโครคอนโทรลเลอร์ โปรแกรมในส่วนนี้จะเป็นคำสั่งในการติดต่อ อ่าน / เขียน ข้อมูลลงบน EEPROM โดยมีหลักการการทำงานตาม Flow Chart รูปที่ 3.5

หลักการทำงานของโปรแกรมจะทำการตั้งค่าการติดต่อสื่อสารทางพอร์ทอนุกรมเพื่อให้สามารถติดต่อกับคอมพิวเตอร์ได้ จากนั้นจะทำการเช็คที่สวิทช์เพื่อให้ทราบว่าต้องไปเรียกโปรแกรมย่อยใดมาทำงาน โดยที่จะใช้พอร์ท 1.4 – 1.7 เป็นอินพุทพอร์ทเพื่อเช็คสวิทช์ เริ่มจากพอร์ท 1.4 จะเป็นสวิทช์เช็คว่ามีการใส่บัตรเข้ามาถูกต้องเรียบร้อยแล้วรึยัง จากนั้นทำการเลือกว่าจะทำการอ่าน / เขียนโดยสวิทช์ READ / WRITE (พอร์ท 1.6) และต้องกดสวิทช์ ENTER (พอร์ท 1.7) เพื่อเริ่มการทำงานตามโปรแกรมย่อย ถ้าต้องการอ่านข้อมูลจากบัตร ก็จะเรียกโปรแกรมย่อย re_data เพื่อทำการอ่านข้อมูลจากในบัตรมาเก็บไว้ในหน่วยความจำของเครื่อง หรือถ้าต้องการเขียนข้อมูลลงในบัตร ก็จะทำการเรียกโปรแกรมย่อย wr_data ซึ่งในโปรแกรมย่อยนี้ก็จะมีการเรียกโปรแกรมย่อยอีกคือ wr_disable , erase , wr_dat และ wr_enable เพื่อทำการเขียนข้อมูลลงบนบัตร



รูปที่ 3.5 Flow Chart แสดงการทำงานของโปรแกรม

3.2.1 โปรแกรมในส่วนที่เขียนบน Computer ด้วยภาษา Visual Basic 5

รายละเอียดของ Visual Basic 5

โปรแกรมไมโครซอฟต์ Visual Basic 5.0 เป็นโปรแกรมที่ช่วยในการสร้างแอปพลิเคชั่นบน Microsoft Windows การสร้างแอปพลิเคชั่นที่แสดงผลในหน้าต่างของเว็บบราวเซอร์ (Web Browser) ได้ รวมทั้งสร้างคอนโทรล ActiveX ได้อย่างรวดเร็ว และภายใน VB5 จะมีเครื่องมือช่วยในการสร้างแอปพลิเคชั่นอีกมากมาย

การสร้างแอปพลิเคชั่นโดยใช้ VB5 จะเป็นไปในลักษณะตามขั้นตอนต่อไปนี้

การออกแบบส่วนของการติดต่อกับผู้ใช้งาน โดยการเลือกคอนโทรลที่ตอบสนองการใช้งานตามที่ต้องการลงบนหน้าจอ จัดตำแหน่ง และขนาดตามความเหมาะสม

การเขียนคำสั่งโปรแกรมที่ต้องตอบสนองต่อเหตุการณ์ที่เกิดขึ้น เช่น การตอบสนองต่อผู้ใช้ เป็นต้น (เราเรียกว่าเป็นการเขียนโปรแกรมแบบ Event-driven) ใน VB5 นั้นได้ใช้ภาษา BASIC ซึ่งอาจจะมีรูปแบบคล้ายกับรูปแบบของภาษา BASIC แบบดั่งเดิม เนื่องจากภาษา BASIC เป็นภาษาที่ออกแบบมาสำหรับผู้เริ่มต้นใช้งาน แต่ยังคงความสามารถในการสร้างแอปพลิเคชั่นที่ ต้องการได้เป็นยังดี

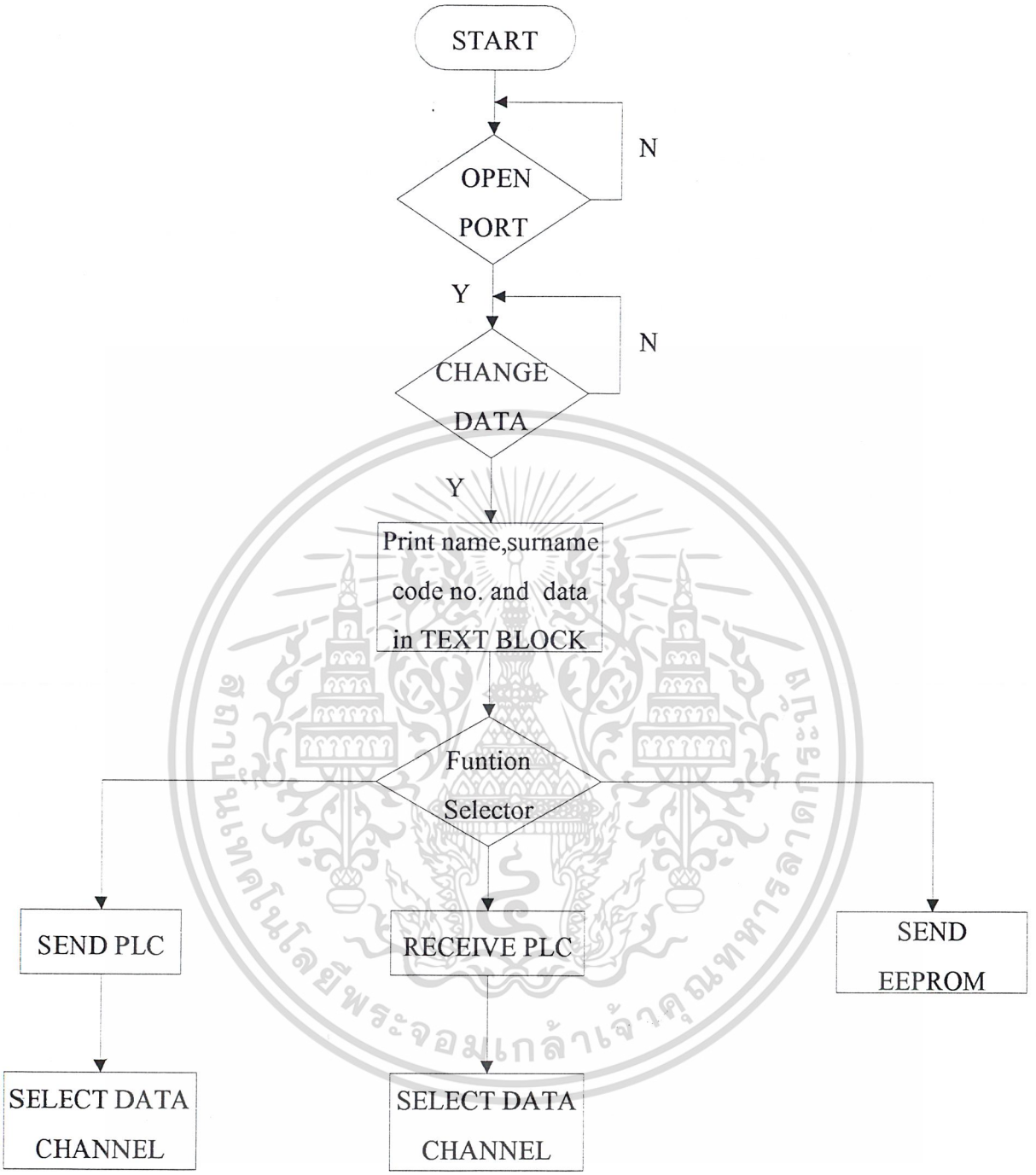
นอกจากนี้ VB5 ยังมีเครื่องมือในการตรวจสอบการทำงานของโปรแกรม เครื่องมือในการสร้างโปรแกรมติดตั้ง และเครื่องมือประเภท Wizard อยู่อย่างมากมาย รวมทั้งใช้งานได้ไม่ยาก ทำให้ VB5 เหมาะสำหรับการพัฒนาโปรแกรมในปัจจุบันมากที่สุด

ความต้องการของระบบ VB5

ความต้องการของระบบที่จะใช้ VB5 มีความต้องการดังต่อไปนี้

- ระบบปฏิบัติการ Microsoft Windows NT3.51 หรือ ใหม่กว่า
- หน่วยประมวลผลกลางรุ่น 80486 หรือสูงกว่า
- เนื้อที่ ฮาร์ดดิสก์อย่างน้อย 50 เมกกะไบต์
- ต้องการ CD –ROM ไดร์ฟ
- การ์ดจอ VGA หรือสูงกว่า
- ROM อย่างน้อย 16 เมกกะไบต์

หลักการของการทำงานเมื่อรันโปรแกรมแล้วจะเป็นดัง Flow Chart ดังรูปที่ 3.6

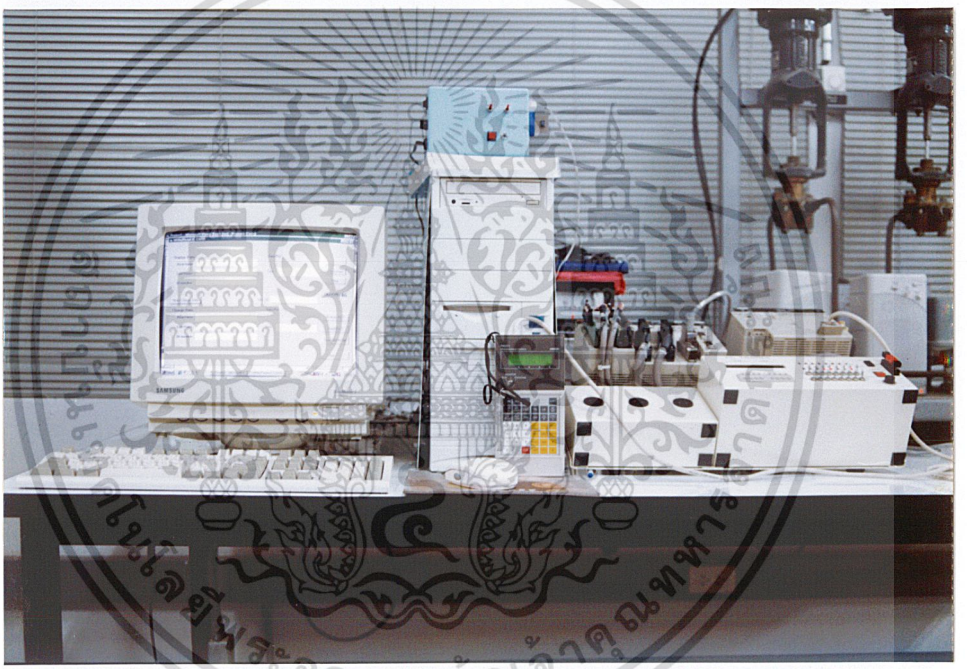


รูปที่ 3.6 Flow Chart หลักการทำงานของโปรแกรม

บทที่ 4
การทดลอง

การทดลองโปรแกรมในส่วนที่เขียนลงใน EPROM เพื่ออ่าน / เขียนข้อมูลลงในบัตรนั้น ได้ทำการทดลองโดยใช้โปรแกรมติดต่อทางพอร์ทอนุกรมกับ Computer คือโปรแกรม PCTOOLS เพื่อใช้ส่งข้อมูลออกมาทางพอร์ท COM 1 มาเข้าที่ตัวเครื่องและแสดงข้อมูลอ่านออกมาจากบัตร การทดลองในช่วงนี้จะยังไม่มีการส่งข้อมูล หรือ รับข้อมูลจาก PLC

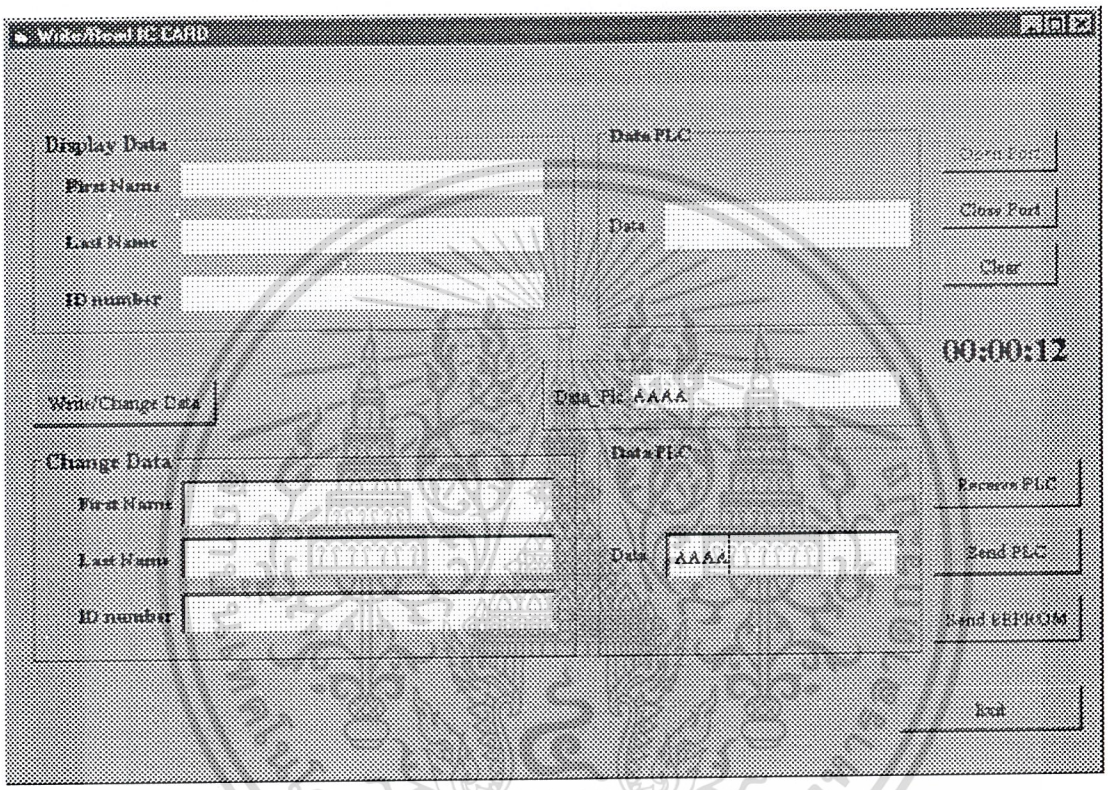
เมื่อโปรแกรมทั้งสองส่วนเสร็จจึงเริ่มทำการทดลองการใช้งานจริง เมื่อทำการทดลองต้องต่ออุปกรณ์ต่าง ๆ ให้เรียบร้อย เช็คสายสัญญาณให้ถูกต้องตามรูปที่4.1



รูปที่ 4.1 การต่ออุปกรณ์ทั้งหมดเมื่อทำการทดลอง

- ต่อสายสัญญาณ RS232 จากเครื่องเข้า Port COM 1 ของ PC
- ต่อสายสัญญาณ RS232 จาก PLC เข้า Port COM 2 ของ PC
- เปิดเครื่อง เสียบบัตร IC Card ใน Socket ให้ถูกต้อง
- เปิด PLC เข้าโหมดโปรแกรมและ Monitor ดูที่ Channel ที่ต้องการอ่าน หรือ เขียน

- เปิดคอมพิวเตอร์เข้าโปรแกรม PROJECT.EXE ที่เขียนด้วยภาษา Visual Basic จะเริ่มเข้าสู่หน้าต่างหลักของโปรแกรมดังรูปที่ 4.2



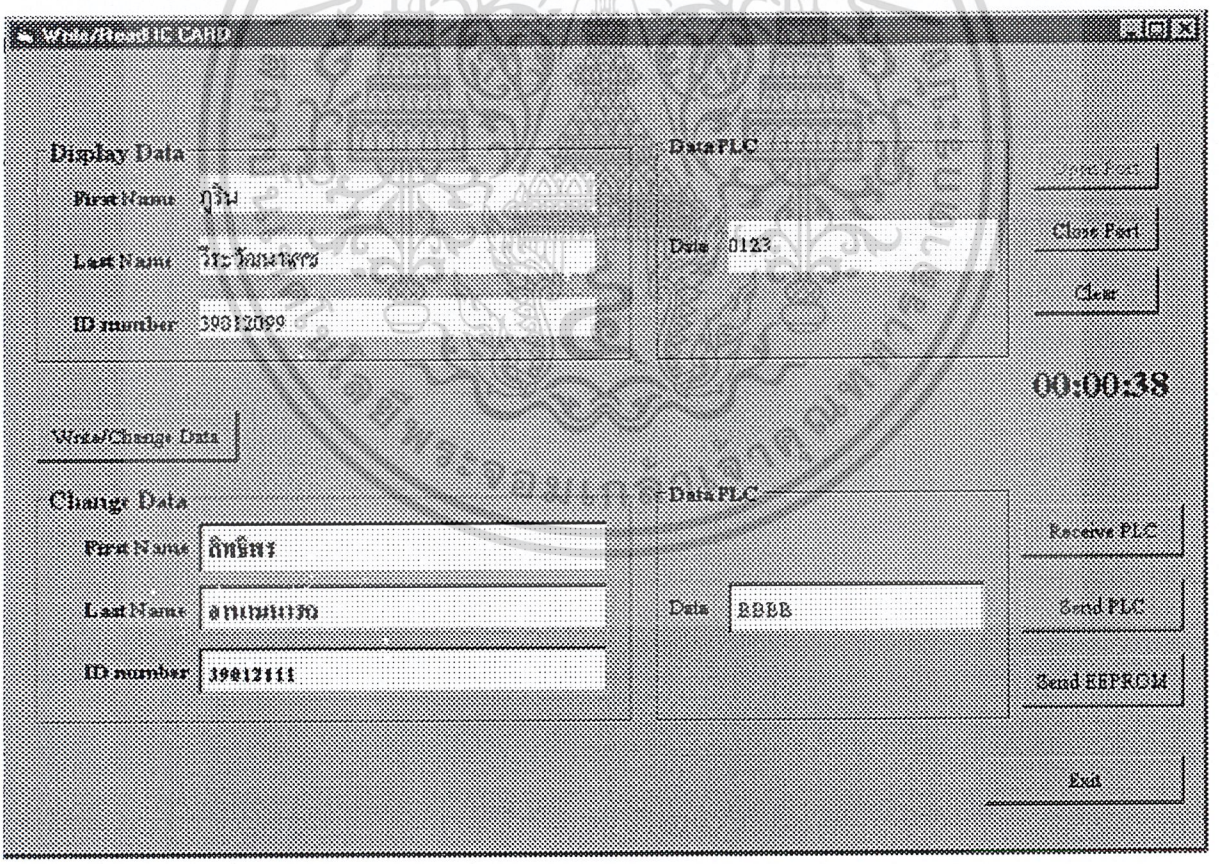
รูปที่ 4.2 หน้าต่างหลักของโปรแกรม PROJECT.EXE

- คลิกที่ปุ่ม Open Port จากนั้นจะสามารถเริ่มทำงานกับ PLC หรือ Module ได้ โปรแกรมจะเริ่มนับเวลาในการทำงานหรือใช้งาน การจับเวลามีไว้ในกรณีที่ถ้าต้องการกำหนดเวลาในการใช้งานของผู้ที่ถือบัตรแต่ละคน และจะสามารถเริ่มการทำงานในขั้นตอนต่อไปได้

- ถ้าคลิกที่ปุ่ม Write / Change Data หน้าต่างส่วนที่ทำการเปลี่ยนแปลงค่าจะปรากฏขึ้นดังรูปที่ 4.3

หากต้องการที่จะเขียนข้อมูลลงในบัตรก็ต้องตั้งสวิทช์ Read / Write บนตัวเครื่องไปที่ตำแหน่ง Write และกด Enter เครื่องจะรอรับค่าที่ส่งมาทางพอร์ท จากนั้นทำการป้อนข้อมูลลงในหน้าต่างส่วน Change Data ข้อมูลที่ป้อนลงในบัตรจะประกอบไปด้วย ชื่อ นามสกุล รหัส และ Data (ข้อมูลที่จะสามารถส่งให้ PLC ได้ ต้องเป็นเลขฐานสิบหกจำนวน 4 , 8 หรือ 12 Byte ตามจำนวน Channel ที่ต้องการป้อน ในรูปที่ 4.3 คือ ชื่อ : สิทธิพร , นามสกุล : อานามนารถ , รหัส : 39012111 , Data : BBBB) จากนั้นคลิกที่ปุ่ม Send EEPROM ข้อมูลทั้งหมดก็จะถูกเขียนลงในบัตร

หากต้องการอ่านข้อมูลในบัตรก็ต้องตั้งสวิทช์ Read / Write บนตัวเครื่องไปที่ตำแหน่ง Read และกด Enter เครื่องจะอ่านข้อมูลจากบัตรและส่งออกมาทางพอร์ท (ดังรูปที่ 4.3 คือ ชื่อ : ภูริน , นามสกุล : วีระวัฒนาเดช , รหัส : 39012099 Data : 0123) หน้าต่างส่วน Display Data ก็จะแสดงข้อมูลที่มีอยู่ในบัตรออกมา

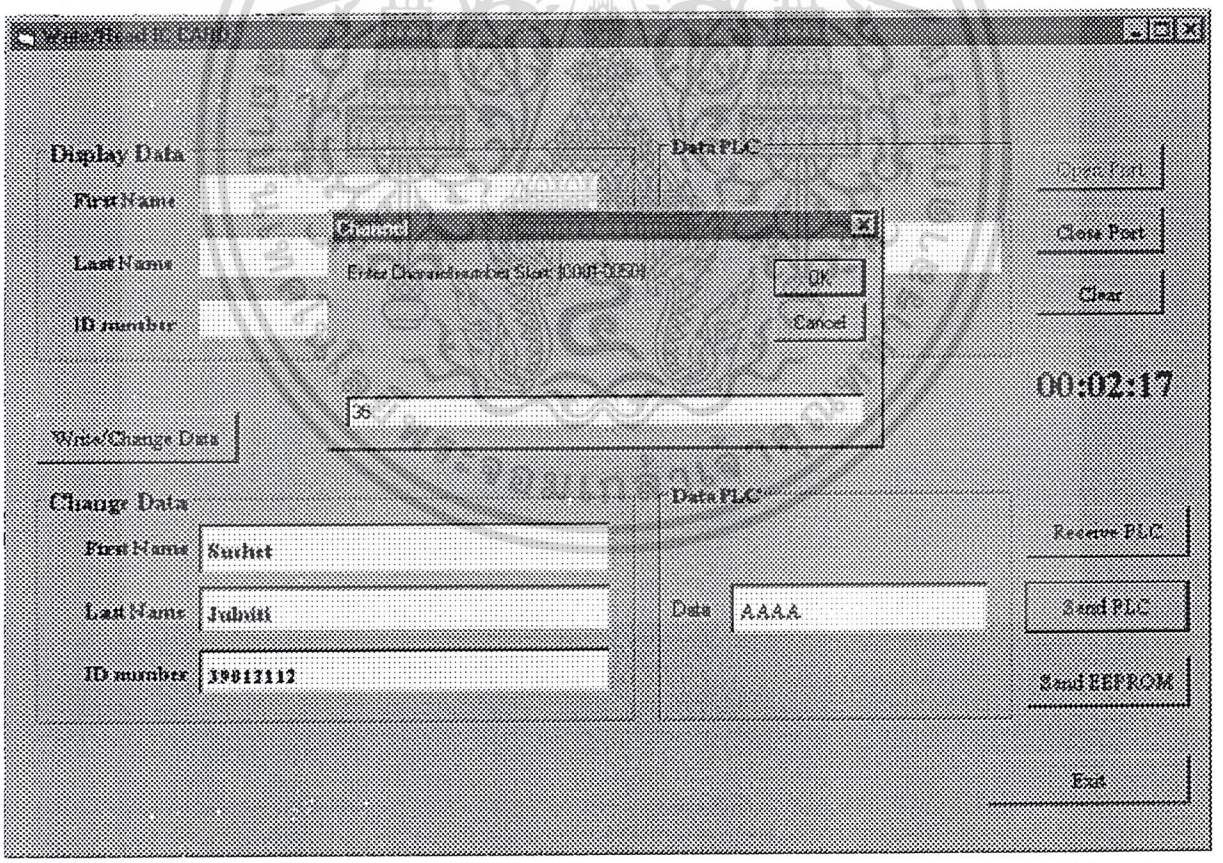


รูปที่ 4.3 ภาพหน้าต่างขณะทำการอ่าน / เขียนข้อมูลลงใน EEPROM

- การอ่าน / เขียนข้อมูลลงในหน่วยความจำของ PLC จะมี 2 กรณีคือ

หากต้องการนำข้อมูลส่วน Data เขียนลงในหน่วยความจำของ PLC ก็ทำได้โดยการคลิกที่ปุ่ม Send PLC จะปรากฏหน้าต่าง Channel ขึ้นเพื่อให้เลือกว่าต้องการส่ง Data ไปยังหน่วยความจำในตำแหน่งใด (จากรูปที่ 4.4 กำหนดให้ส่งไปที่ Channel 35) แล้วคลิก OK โปรแกรมจะนำเอา Data ที่กำหนด (จากรูปที่ 4.4 คือ AAAA) มารวมกับส่วนต่าง ๆ ในรูปแบบ Command Block ของ Protocal ที่กำหนด และส่งออกทางพอร์ท COM 2 Data จะถูกส่งออกไปเก็บไว้ยัง PLC Channel ละ 4 Byte ถ้า Data มาไม่ครบก็จะถูกตัดทิ้ง

หากต้องการอ่านค่าข้อมูลในหน่วยความจำของ PLC ก็คลิกที่ปุ่ม Receive PLC จะปรากกหน้าต่าง Channel เพื่อให้เลือกตำแหน่งหน่วยความจำที่ต้องการอ่านข้อมูล เมื่อใส่ค่าที่ต้องการแล้วคลิกที่ OK โปรแกรม จะส่งนำค่า Channel ที่กำหนดไปรวมกับส่วนต่าง ๆ ในรูปแบบ Command Block เพื่ออ่านข้อมูลในตำแหน่งที่กำหนด เมื่อ PLC ได้รับคำสั่งในการอ่านข้อมูลจะส่ง Respond Block ออกมาแล้ว จากนั้นโปรแกรมจะแยกเอาเฉพาะส่วนของข้อมูลออกมาแสดงผลที่หน้าต่าง Data_PLC ดังรูปที่ 4.2



รูปที่ 4.4 หน้าต่าง Channel เมื่อทำการติดต่อกับ PLC

บทที่ 5
สรุปผลการทดลองและข้อเสนอแนะ

สรุปผลการทดลอง

โครงงานและปริญญานิพนธ์เรื่อง เครื่องอ่าน / เขียนข้อมูล โดยใช้ EEPROM นี้จัดทำเพื่อเป็นเครื่องต้นแบบและใช้เป็นแนวทางพัฒนาให้มีคุณภาพและได้มาตรฐานสามารถนำไปประยุกต์ใช้งานให้เกิดประโยชน์ยิ่งขึ้นต่อไป โดยเครื่องที่สร้างขึ้นมานี้ได้ใช้การสื่อสารข้อมูลแบบอนุกรมด้วยมาตรฐาน RS-232 และทำการอ่านเขียนข้อมูลลงใน EEPROM # 93C46 โดยนำมาทำเป็นบัตร IC Card ซึ่งใช้ CPU # 8031 เป็นตัวทำการอ่านเขียนข้อมูลพร้อมทั้งรับส่งข้อมูลกับ PC และสามารถนำข้อมูลดังกล่าวไปเขียนลงในหน่วยความจำของ PLC โดยใช้โปรแกรม Visual Basic อยู่บน PC ซึ่งสามารถนำไปประยุกต์ใช้ในงานอุตสาหกรรมได้อย่างกว้างขวาง เช่น บัตรแต่ละใบก็จะเก็บข้อมูลเฉพาะไว้ เมื่อมีการนำบัตรใบ 1 มาใช้ก็จะเป็นการสั่งให้เครื่อง PLC ทำงานตามโปรแกรมย่อยที่ 1 ถ้าใช้บัตรอีกใบก็จะไปทำงานตามโปรแกรมย่อยอีกโปรแกรมหนึ่ง

ข้อเสนอแนะ

บัตรนี้ใช้ EEPROM ที่มีขายอยู่ทั่วไปสามารถหาซื้อ Data Sheet ได้ง่ายจึงสามารถนำไปอ่าน / เขียนข้อมูลได้ถ้ามีการป้อนสัญญาณที่ถูกต้องให้กับ IC ดังนั้นจึงยังไม่มีความปลอดภัยเพียงพอในการนำไปใช้งานที่ต้องการเก็บข้อมูลที่เป็นส่วนตัว หรือข้อมูลไม่อาจเปิดเผยได้ แต่ถ้าต้องความปลอดภัยในการเก็บข้อมูลก็ควรที่จะเขียนโปรแกรมที่เป็น Security เพื่อป้องกันการแก้ไขข้อมูลโดยบุคคลอื่น

บรรณานุกรม

1. รามินเดอร์ ศรีกิจจาภรณ์ , คู่มือการใช้ Visual / Basic สำหรับวินโดวส์ กรุงเทพมหานคร : บริษัท ซีเอ็ดยูเคชั่น จำกัด 2531
2. สุพรรณ กุลพาณิชย์ , Programmable Controller เทคนิคและการใช้งานเบื้องต้น พิมพ์ครั้งที่ 2 กรุงเทพมหานคร : บริษัท ออมร่อน ตรีศักดิ์ จำกัด 2533
3. ผศ. มนต์ชัย เทียนทอง , การโปรแกรมภาษา กรุงเทพมหานคร : โรงพิมพ์สถาบันเทคโนโลยีพระจอมเกล้าพระนครเหนือ 2535
4. สุเจตน์ จันทรังษ์ , ไมโครคอนโทรลเลอร์ชิพเดี่ยว 8051 พิมพ์ครั้งที่ 1 กรุงเทพมหานคร : วิทยาลัยมหานคร 2535

ภาคผนวก

รายละเอียดอุปกรณ์

โปรแกรมภาษา C ที่อัดลงใน EPROM

```
/*****************************************************************/
/*                        PROGRAM IC CARD 8031                   */
/*****************************************************************/
#pragma debug objectextend code
#include<reg51.h>
#include<absacc.h>
#include<stdio.h>
/*****************************************************************/
/*                        SET BIT PORT 1                         */
/*****************************************************************/
sbit clk =P1^0;
sbit cs  =P1^1;
sbit di  =P1^2;
sbit out =P1^3;
sbit plc_com=P1^5;
sbit sensor_card=P1^4;
sbit read_write=P1^6;
sbit enter=P1^7;
/*****************************************************************/
/*                        FUNCTION                               */
/*****************************************************************/
void serial(void);
void clock(void);
void re_data(void);
char re_dat(int);
void wr_data(void);
void wr_dat(int,char);
void wr_enable(void);
```

```
void wr_disable(void);
void erase(int);
int x;

/*****************************************************************************/
/*                                CLOCK                                      */
/*****************************************************************************/
void clock(void)
{
  clk=0;
  clk=1;
}
/*****************************************************************************/
/*                            READ DATA EEPROM                               */
/*****************************************************************************/
char re_dat(int address)
{
  int i,j;
  char cod;
  cod=0;
  di =0;
  clk=0;
  cs =0;
  out=1;
  cs =1;
  clock();
  di=1;
  clock();
  di=1;
  clock();
```

```
  di=0;
  clock();
  for(i=5;i>=0;i--)
  {
   di=(address>>i)&1;
   clk=0;
   clk=1;
  }
  clock();
  for(i=15;i>=0;i--)
  {
   clk=1;
   clk=0;
   j=out;
   cod|=j<<i;
  }
  clock();
  cs =0;
  clock();
  cs=1;
  di =0;
  clk=0;
  return(cod);
}
/*****************************************************************/
/*                        WRITE DATA EEPROM                      */
/*****************************************************************/
void wr_dat(int address,char value)
{
  int i;
```

```
 di =0;
 clk=0;
 cs =0;
 out=1;
 cs =1;
 clock();
 di =1;
 clock();
 di =0;
 clock();
 di =1;
 clock();
 for(i=5;i>=0;i--)
 {
  di=(address>>i)&1;
  clk=0;
  clk=1;
 }
 for(i=15;i>=0;i--)
 {
  di=(value>>i)&1;
  clk=0;
  clk=1;
 }
 cs =0;
 clock();
 clock();
 cs =1;
 clock();
 clock();
```

```
  while(!out)clock();
  clock();
  cs =0;
  clk=0;
  di =0;
  clock();
}
/*********************************************************************/
/*                          ERASE EEPROM                             */
/*********************************************************************/
void erase(int address)
{
   int i;
  di =0;
  clk=0;
  cs =0;
  out=1;
  cs =1;
  clock();
  di =1;
  clock();
  di =1;
  clock();
  di =1;
  clock();
  for(i=5;i>=0;i--)
  {
   di=(address>>i)&1;
   clk=0;
   clk=1;
```

```
  }
  clock();
  clock();
  cs=0;
  clock();
  cs=1;
  clock();
  while(!out);clock();
  cs=0;
  clk=0;
  di=0;
}
/*****************************************************************/
/*                    WRITE_ENABLE EEPROM                        */
/*****************************************************************/
void wr_enable(void)
{
  di =0;
  clk=0;
  cs =0;
  out=1;
  cs =1;
  clock();
  di =1;
  clock();
  di =0;
  clock();
  clock();
  di =1;
  clock();
```

```
   clock();
   clock();
   clock();
   clock();
   clock();
   cs =0;
   di =0;
   clk=0;
}
/*****************************************************************/
/*                    WRITE DISABLE EEPROM                       */
/*****************************************************************/
void wr_disable(void)
{
  di =0;
  clk=0;
  cs =0;
  out=1;
  cs =1;
  clock();
  di =1;
  clock();
  di =0;
  clock();
  clock();
  di =0;
  clock();
  clock();
  clock();
  clock();
```

```
   clock();
   clock();
   cs =0;
   clk=0;
}
/*************************************************************************/
/*                    THANSMITION DATA COMPUTER & 8031                   */
/*************************************************************************/
void wr_data(void)
{
  int i;
  char read_d[60];
  serial();
  for(i=0;i<x-1;i++)
   {
   read_d[i]=re_dat(i);
   TI=0;
   SBUF=read_d[i];
   while(!TI);
   }
}
/*************************************************************************/
/*                    RECEIVE DATA COMPUTER & 8031                       */
/*************************************************************************/
void re_data(void)
{
  int i;
  char cod[60];
  serial();
  x=0;
```

```
  do
  {
   while(!RI);
   cod[x]=SBUF;
   RI=0;
   x=x+1;
  }
  while(cod[x-1]!='*');
  wr_enable();
  for(i=0;i<x-1;i++)
  {
   erase(i);
   wr_dat(i,cod[i]);
  }
  wr_disable();
  while(!RI);
}

/*********************************************************************/
/*                          THANSMITION DATA PLC & 8031                */
/*********************************************************************/
/*void wr_dataPLC(void)
{
  int i;
  char read_PLC[60];
  Send[16]="@00WR01000120Fcs*";
  serial();
  for(i=0;i<16;i++)
   {
   TI=0;
```

```
    SBUF=Send[i];
    while(!TI);
   }
  for(i=0;i<x-1;i++)
   {
   read_PLC[i]=re_dat(i);
   TI=0;
   SBUF=read_PLC[i];
   while(!TI);
   }
} */
/*************************************************************************/
/*                          RECEIVE DATA PLC & 8031                      */
/*************************************************************************/
/*void re_dataPLC(void)
{
  int i;
  char Send[16]="@00RR01000002Fcs*";
  char codPLC[60];
  serial();
  x=0;
  for(i=0;i<16;i++)
   {
    TI=0;
    SBUF=Send[i];
    while(!TI);
   }
  do
  {
   while(!RI);
```

```
    cod[x]=SBUF;
    RI=0;
    x=x+1;
   }
   while(x<22);
   wr_enable();
   for(i=0;i<x-1;i++)
   {
    erase(i);
    wr_dat(i,cod[i]);
   }
   wr_disable();
   while(!RI);
} */
/*************************************************************************/
/*                              SERIAL PORT                              */
/*************************************************************************/
void serial(void)
{
  TMOD = 0x20;
  TCON = 0x41;
  TH1  = 0xFD;
  SCON = 0x52;
  TR1  =1;
}
/*************************************************************************/
/*                                 MAIN                                  */
/*************************************************************************/
 main()
 {
```

```
 out=1;
 di =0;
 cs =0;
 clk=0;
for(;;)
{
 if(sensor_card==0 /*&& plc_com==1*/ && read_write==1 && enter==0)
   {
    re_data();
   }
 if(sensor_card==0 /*&& plc_com==1*/ && read_write==0 && enter==0)
   {
    wr_data();
   }
 /*if(sensor_card==0 && plc_com==0 && read_write==1 && enter==0)
   {
    re_dataPLC();
   }
 if(sensor_card==0 && plc_com==0 && read_write==0 && enter==0)
   {
    wr_dataPLC();
   }                    */
 }
}
```

โปรแกรม Visual Basic ที่เขียนบน Computer

```
Dim StartTime As Date
Dim Data As Variant
Dim Data_ReceivePlc As Variant
Dim data_s As Variant
Dim FName As Variant
Dim LName As Variant
Dim ID As Variant
Dim DataPLC As Variant
Dim FName1 As Variant
Dim LName1 As Variant
Dim ID1 As Variant
Dim DataPLC1 As Variant
Dim Channel As String
Dim FCS As String

Private Sub ChangeData_Click()
  Frame2.Visible = True
  Frame4.Visible = True
  Send.Enabled = True
  Clear.Enabled = True
End Sub

Private Sub Clear_Click()
  Text1.Text = ""
  Text2.Text = ""
  Text3.Text = ""
  Text4.Text = ""
  Label2.Caption = ""
```

```
  Label3.Caption = ""
  Label4.Caption = ""
  Label5.Caption = ""
  Label15.Caption = ""
End Sub

Private Sub ClosePort_Click()
  StopTiming
  If MSComm1.PortOpen = True Then
    MSComm1.PortOpen = False
  End If
  ChangeData.Enabled = False
  Frame2.Visible = False
  Frame4.Visible = False
  Frame5.Visible = False
  Send.Enabled = False
  SendPLC.Enabled = False
  Receive.Enabled = False
  Openport.Enabled = True
  Label1.Caption = ""
End Sub

Private Sub Exit_Click()
  End
End Sub

Private Sub Form_Load()
  frmmain.Move (Screen.Width - Width) / 2, (Screen.Height - Height) / 2
End Sub
```

```
Private Sub MSComm1_OnComm()
 Dim a, b, c, w, x, y, z As Integer
 Dim d As Variant
 d = ""
 delay
 d = MSComm1.Input
 delay
 a = InStr(1, d, "$", 0)
If a <> 0 Then
 b = Mid(d, a + 1, 2)
 Data = Mid(d, a, b - 2)
 MSComm1.PortOpen = False
 x = 3
 y = InStr(1, Data, "!", 0)
 z = InStr(1, Data, "@", 0)
 w = InStr(1, Data, "#", 0)
If w < z Then
  w = z + 9
End If
If w > z > y > x Then
  Label2.Caption = Mid(Data, x + 1, y - x - 1)
  FName = Label2.Caption
  Label3.Caption = Mid(Data, y + 1, z - y - 1)
  LName = Label3.Caption
  Label4.Caption = Mid(Data, z + 1, w - z - 1)
  ID = Label4.Caption
  Label5.Caption = Mid(Data, w + 1, b - w - 1)
  DataPLC = Label5.Caption
  Openport.Enabled = True
  Timer2.Enabled = True
```

```
    Else
     MsgBox "Error  Data", vbOKOnly
    End If
  Else
   MsgBox "Error  Data", vbOKOnly
  End If
End Sub

Private Sub MSComm2_OnComm()
    Dim instring As String
    instring = MSComm2.Input
    Data_ReceivePlc = Mid(instring, 8, 12)
    MSComm2.PortOpen = False
    Frame5.Visible = True
    Label15.Caption = Data_ReceivePlc
    Receive.Enabled = True
    SendPLC.Enabled = True
End Sub

Private Sub Openport_Click()
  If MSComm1.PortOpen = False Then
    MSComm1.PortOpen = True
  End If
  MSComm1.RThreshold = 20
  StartTiming
  ChangeData.Enabled = True
  SendPLC.Enabled = True
  Receive.Enabled = True
  Openport.Enabled = False
End Sub
```

```
Private Sub Receive_Click()
  Dim word As String
  Channel = InputBox$("Enter Channel number Start: (0001-0050)", "Channel")
  If Channel = "" Then
    Channel = "0001"
  ElseIf Len(Channel) = 1 Then
    Channel = "000" + Channel
  ElseIf Len(Channel) = 2 Then
    Channel = "00" + Channel
  ElseIf Len(Channel) = 3 Then
    Channel = "0" + Channel
  End If
  If MSComm2.PortOpen = False Then
    MSComm2.PortOpen = True
  End If
  MSComm2.RThreshold = 23
  word = "@00RR" + Channel + "0003"
  FCSloop (word)
  MSComm2.Output = word + FCS + "*" + Chr$(13) + Chr$(10)
  Channel = ""
  Receive.Enabled = False
  SendPLC.Enabled = False

End Sub

Private Sub Send_Click()
  Dim x, y, z, inspect As Variant
  If MSComm1.PortOpen = False Then
    MSComm1.PortOpen = True
  End If
```

```
    MSComm2.Output = word + FCS + "*" + Chr$(13) + Chr$(10)
    Channel = ""
    MSComm2.PortOpen = False
  Else
    MsgBox "ERROR PLC DATA", vbOKOnly
  End If
  DataPLC = ""
  If Text4.Text <> "" Then
   DataPLC = Trim(Text4.Text)
  End If
End Sub

Private Sub Text1_Change()
 Send.Enabled = True
 FName1 = Trim(Text1.Text)
 If FName1 = "" Then
  FName1 = FName
 End If
End Sub

Private Sub Text2_Change()
 Send.Enabled = True
 LName1 = Trim(Text2.Text)
 If LName1 = "" Then
  LName1 = LName
 End If
End Sub
```

```
Private Sub Text3_Change()
 Send.Enabled = True
 ID1 = Trim(Text3.Text)
 If ID1 = "" Then
  ID1 = ID
 End If
End Sub

Private Sub Text4_Change()
 Send.Enabled = True
 DataPLC1 = Trim(Text4.Text)
 If DataPLC1 = "" Then
  DataPLC1 = DataPLC
 End If
 DataPLC = DataPLC1
End Sub

Private Sub Timer1_Timer()
 Label1.Caption = Format(Now - StartTime, "hh:nn:ss")
End Sub

Private Sub StartTiming()
 StartTime = Now
 Timer1.Enabled = True
End Sub

Private Sub StopTiming()
 Timer1.Enabled = False
End Sub
```

```
Private Sub FCSloop(word As String)
 Dim b, c As Integer
 b = Len(word)
 c = 0
 For b = 0 To b - 1
  c = c Xor Asc(Mid(word, b + 1, 1))
 Next b
 FCS = Hex(c)
 If Len(FCS) = 1 Then
  FCS = "0" + FCS
 End If
End Sub
Private Sub delay()
Dim x As Variant
For x = 0 To 150000
 Next x
End Sub

Private Sub Timer2_Timer()
 If MSComm1.PortOpen = False Then
   MSComm1.PortOpen = True
 End If
End Sub
```

8-Bit Microcontroller with 4 Kbytes Flash

AT89C51

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

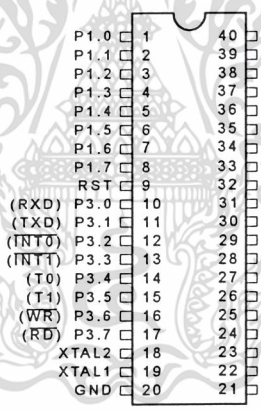
Description

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.

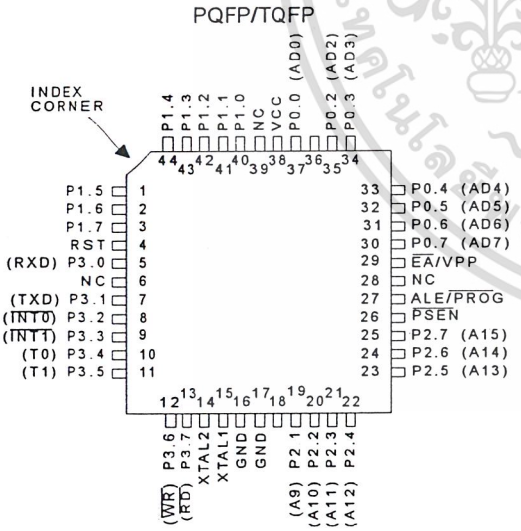
(continued)

Pin Configurations

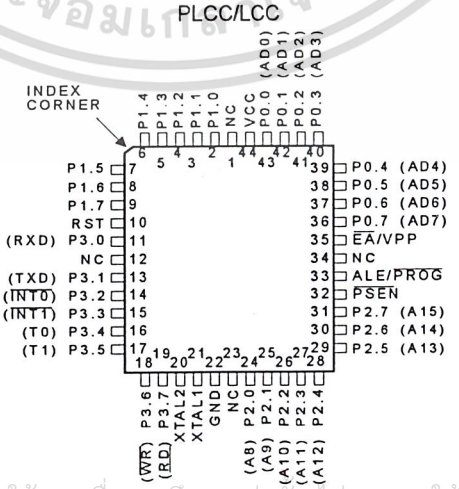
PDIP/Cerdip



PQFP/TQFP

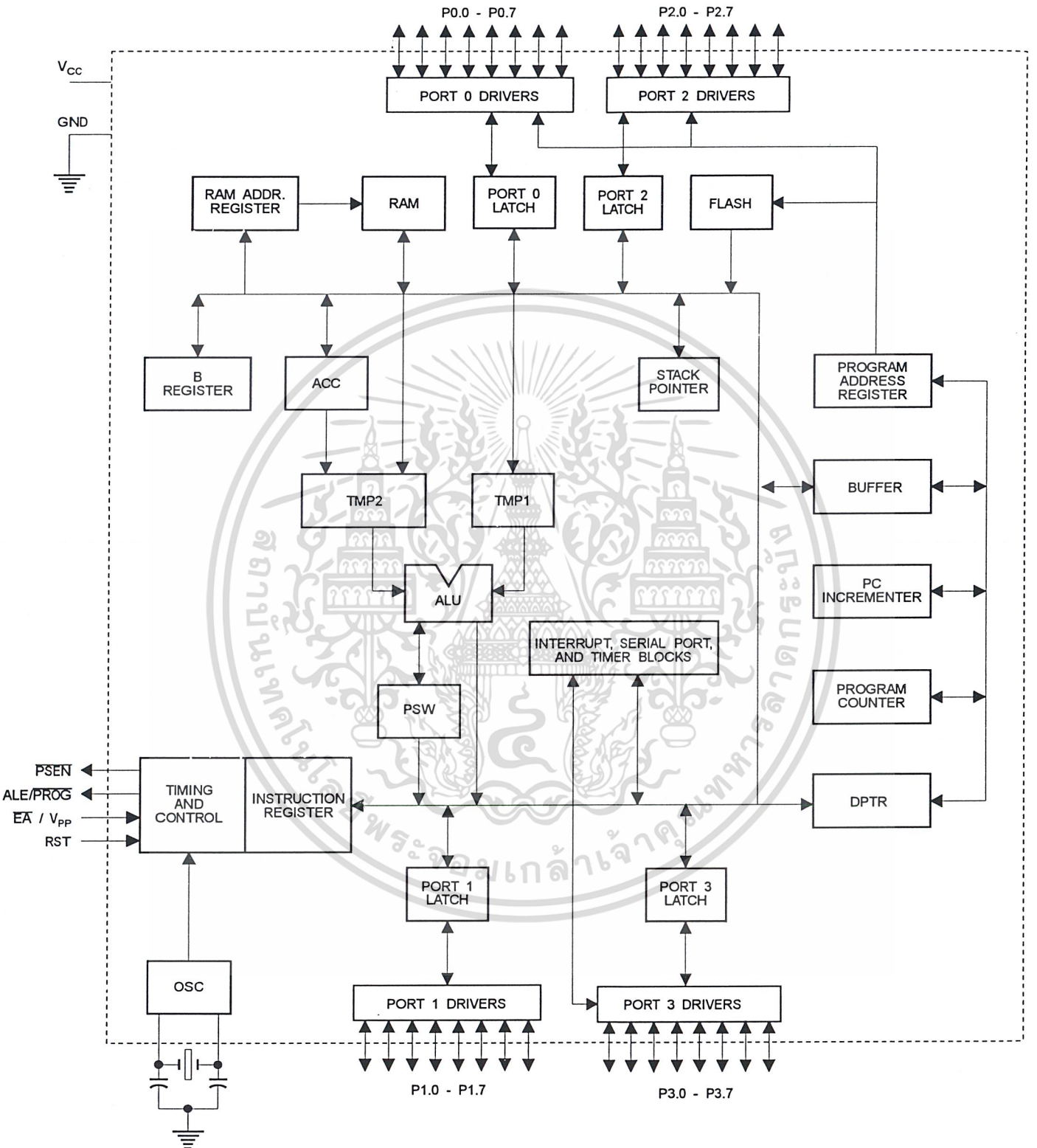


PLCC/LCC



0265E

Block Diagram



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)

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.

EA/VPP

External Access Enable. 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, EA will be internally latched on reset.

EA should be strapped to VCC for internal program executions.

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

XTAL1

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

XTAL2

Output from the inverting oscillator amplifier.

Oscillator Characteristics

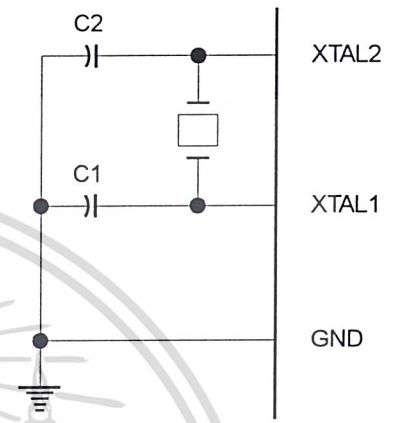
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected 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.

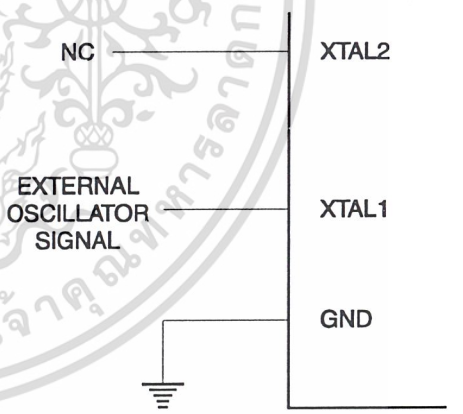
It should be noted that when idle is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hard-

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

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 V_{CC} 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, $\overline{EA}$ 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 (V_{CC}) 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-

(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/Busy: The progress of byte programming can also be monitored by the RDY/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/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	PSEN	ALE/PROG	EA/VPP	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 PROG pulse.

Figure 3. Programming the Flash

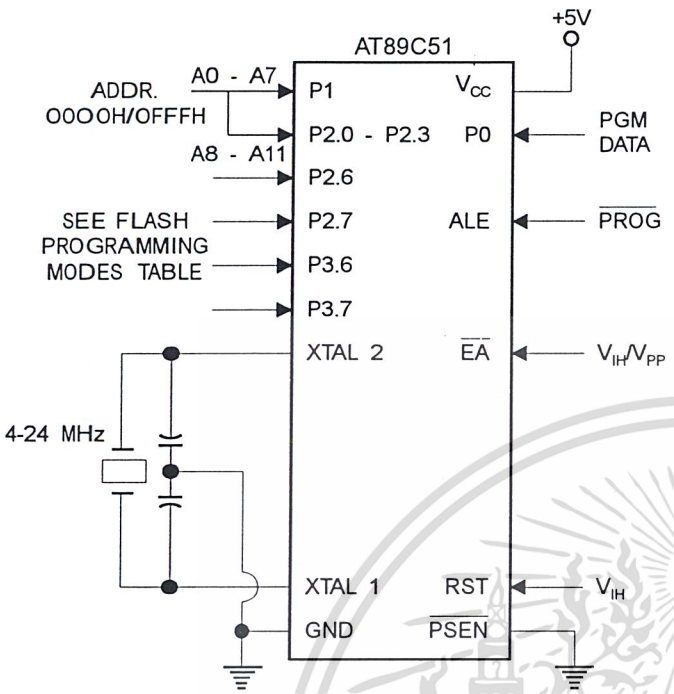
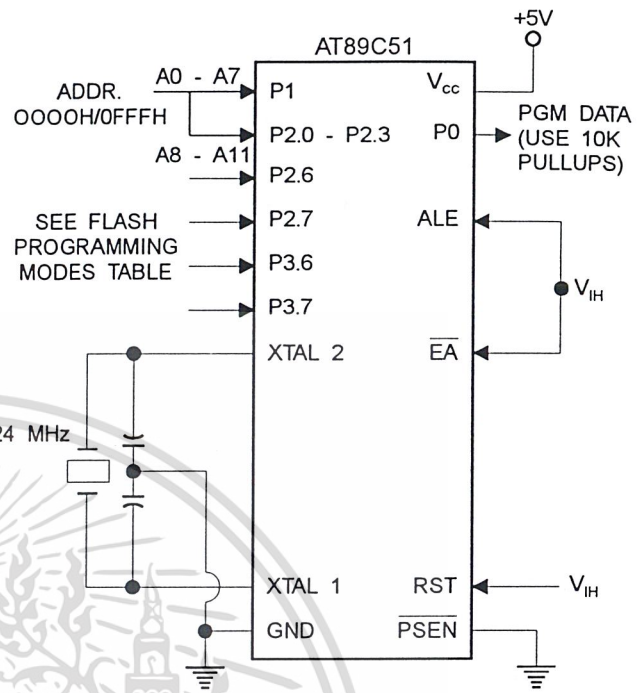


Figure 4. Verifying the Flash



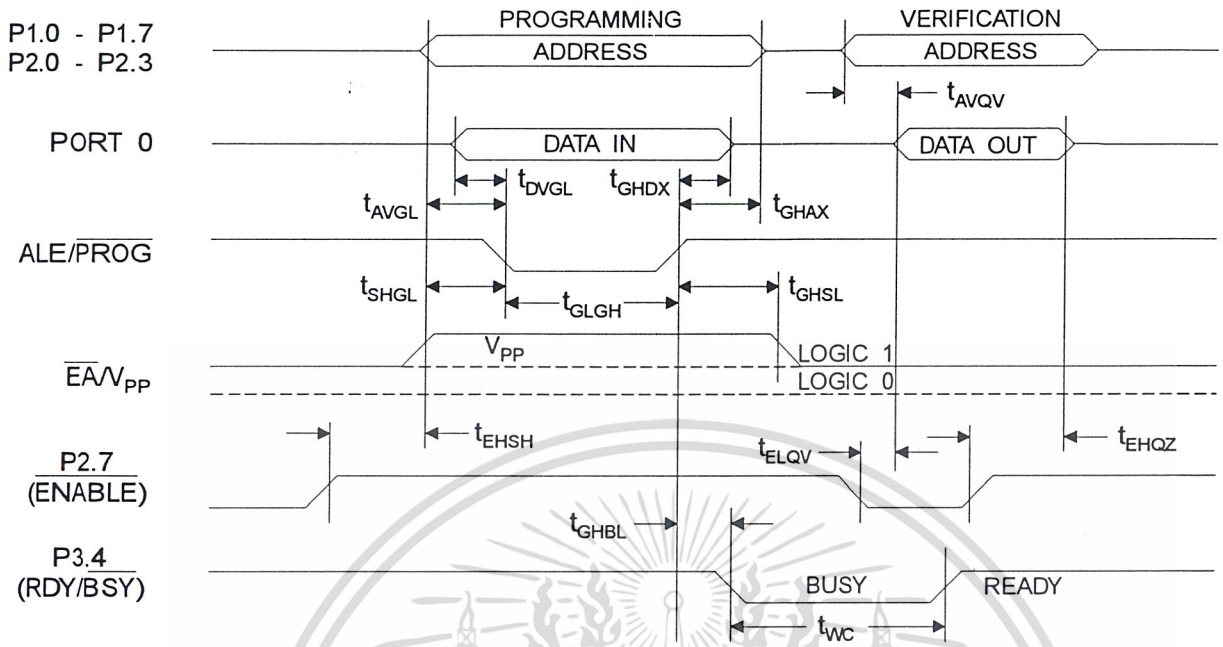
Flash Programming and Verification Characteristics

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

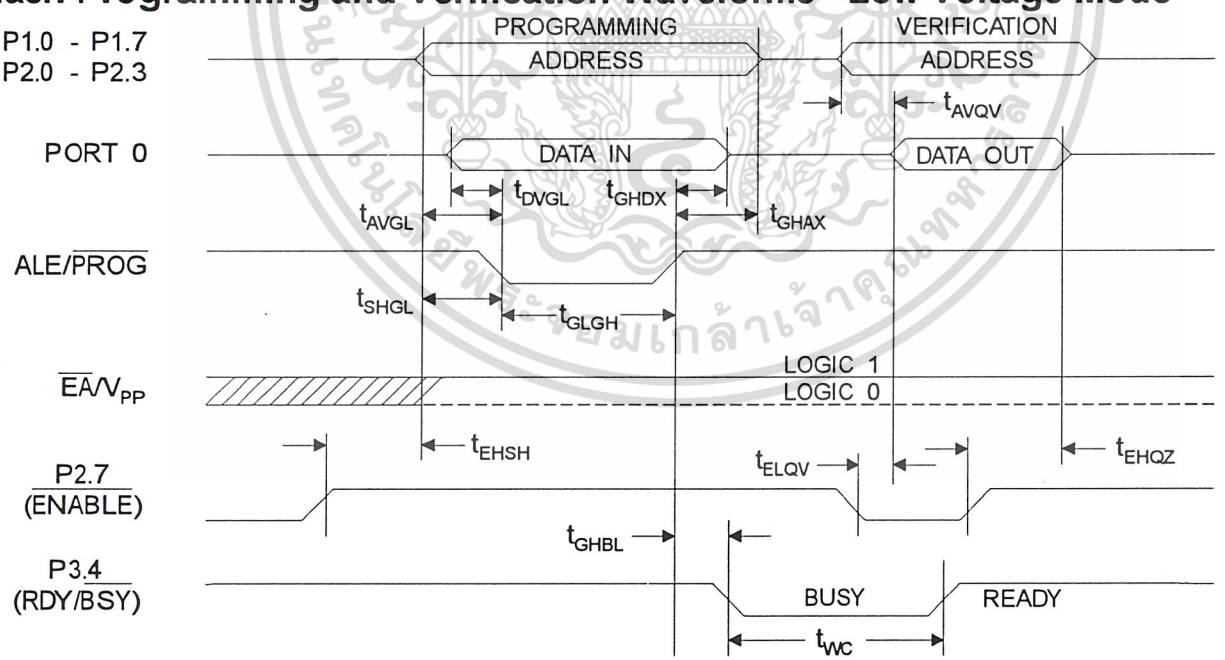
Symbol	Parameter	Min	Max	Units
V_{PP}(1)	Programming Enable Voltage	11.5	12.5	V
I_{PP}(1)	Programming Enable Current		1.0	mA
$1/t_{CLCL}$	Oscillator Frequency	4	24	MHz
t_{AVGL}	Address Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
t_{GHAX}	Address Hold After $\overline{PROG}$	$48t_{CLCL}$		
t_{DVGL}	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.

Flash Programming and Verification Waveforms - High Voltage Mode



Flash Programming and Verification Waveforms - Low Voltage Mode



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, $\overline{EA}$)	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.

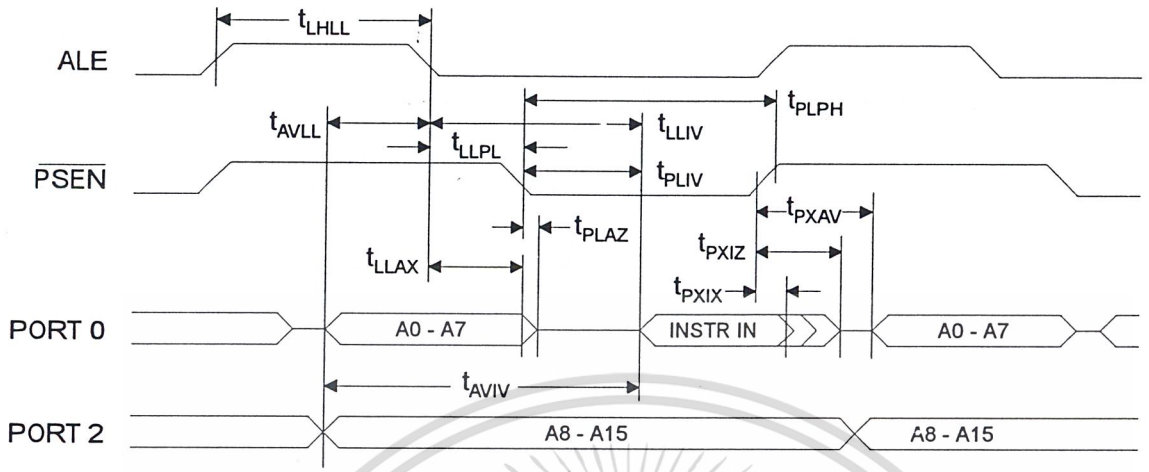
A.C. Characteristics

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

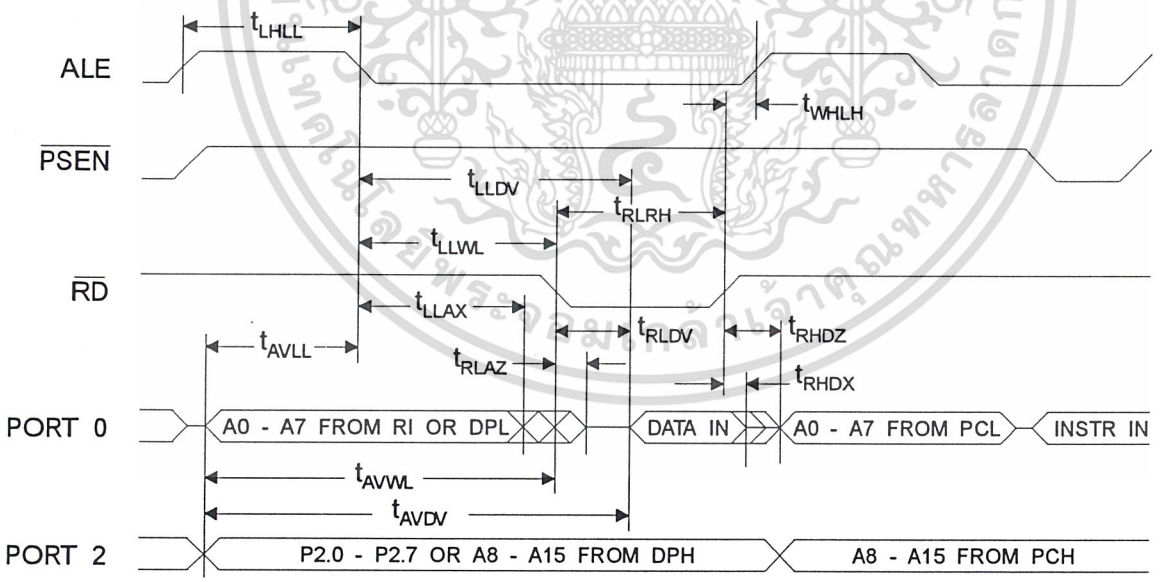
External Program and Data Memory Characteristics

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

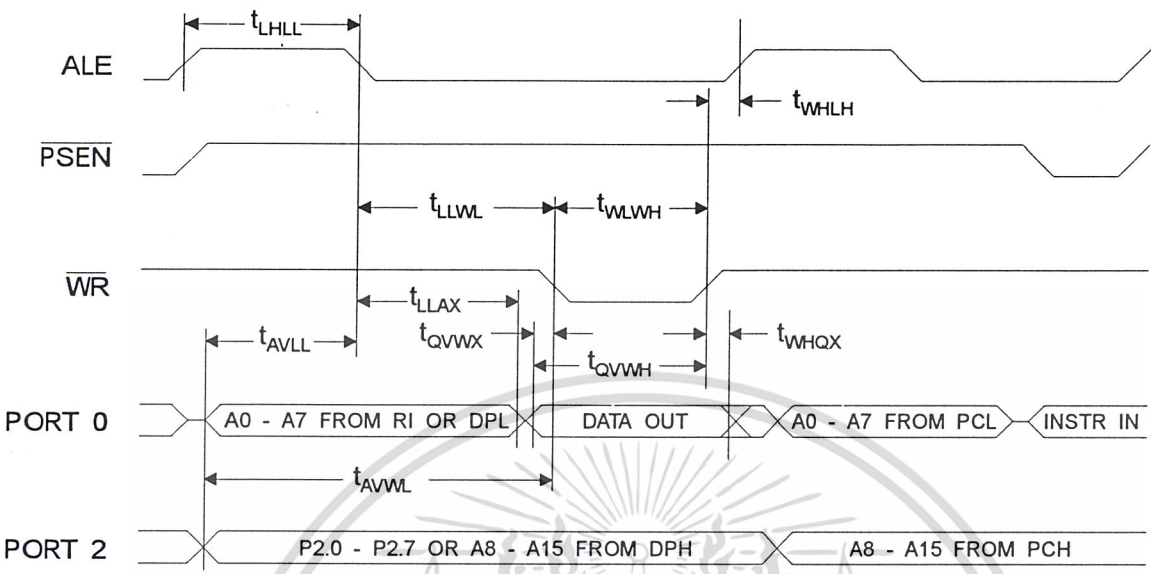
External Program Memory Read Cycle



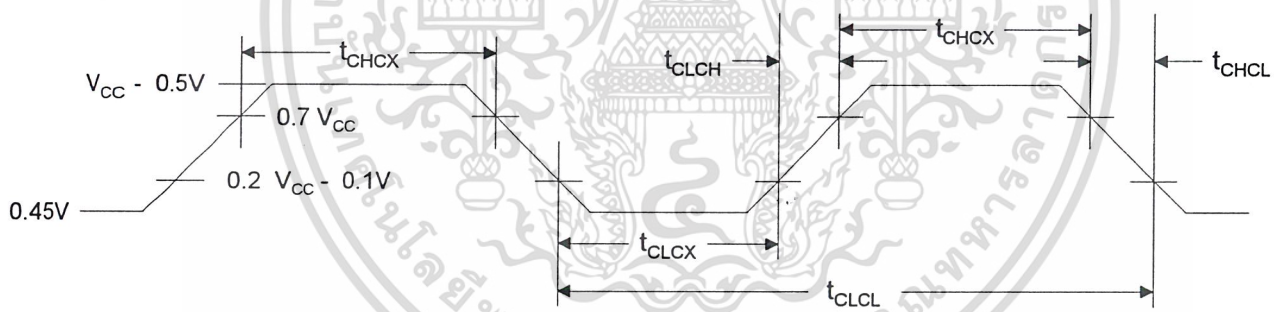
External Data Memory Read Cycle



External Data Memory Cycle



External Clock Drive Waveforms



External Clock Drive

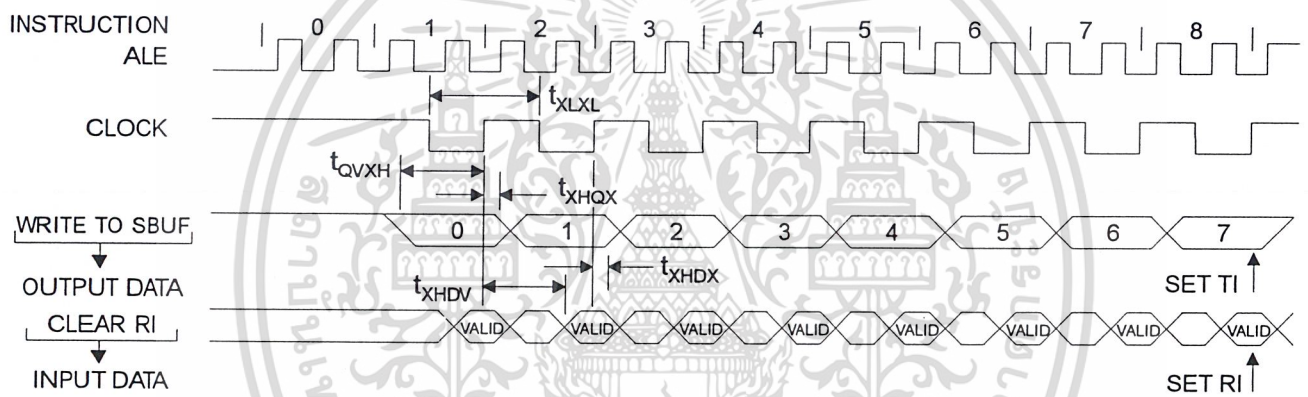
Symbol	Parameter	Min	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
t_{CLCL}	Clock Period	41.6		ns
t_{CHCX}	High Time	15		ns
t_{CLCX}	Low Time	15		ns
t_{CLCH}	Rise Time		20	ns
t_{CHCL}	Fall Time		20	ns

Serial Port Timing: Shift Register Mode Test Conditions

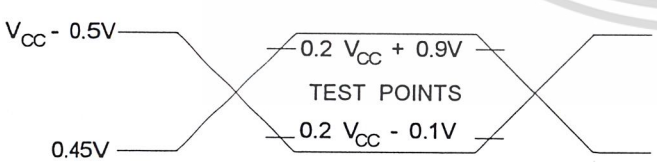
(V_{CC} = 5.0 V ± 20%; Load Capacitance = 80 pF)

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

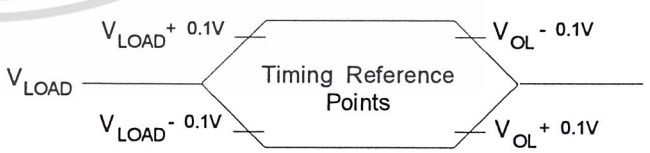


AC Testing Input/Output Waveforms[1]



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.

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)

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)

93LC46B/56B/66B

1K/2K/4K 2.0V CMOS Serial EEPROM

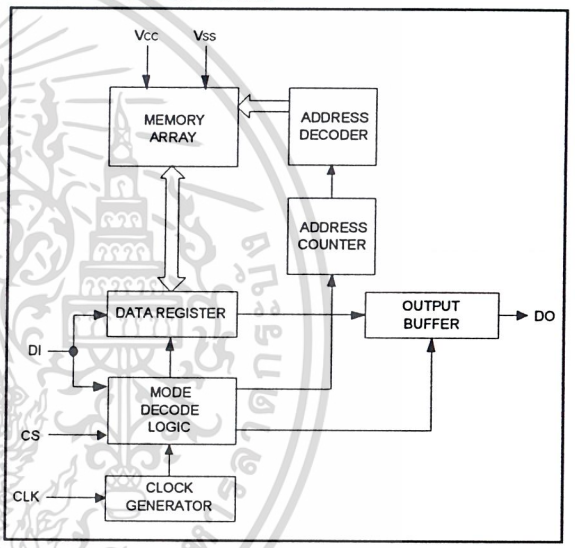
FEATURES

- Single supply with programming operation down to 2.0V (Commercial only)
- Low power CMOS technology
  - 1 mA active current typical
  - 5 μA standby current (typical) at 3.0V
- x16 bit organization
- 64x16 (93LC46B)
- 128x16 (93LC56B)
- 256x16 (93LC66B)
- Self-timed ERASE and WRITE cycles (including auto-erase)
- Automatic ERAL before WRAL
- Power on/off data protection circuitry
- Industry standard 3-wire serial I/O
- Device status signal during ERASE/WRITE cycles
- Sequential READ function
- **10,000,000 ERASE/WRITE cycles guaranteed on 93LC56B and 93LC66B**
- **1,000,000 E/W cycles guaranteed on 93LC46B***
- Data retention > 200 years
- 8-pin PDIP/SOIC and 14-pin SOIC package (SOIC in JEDEC and EIAJ standards)
- Available for extended temperature ranges:
  - Commercial: 0 C to +70 C
  - Industrial: -40 C to +85 C

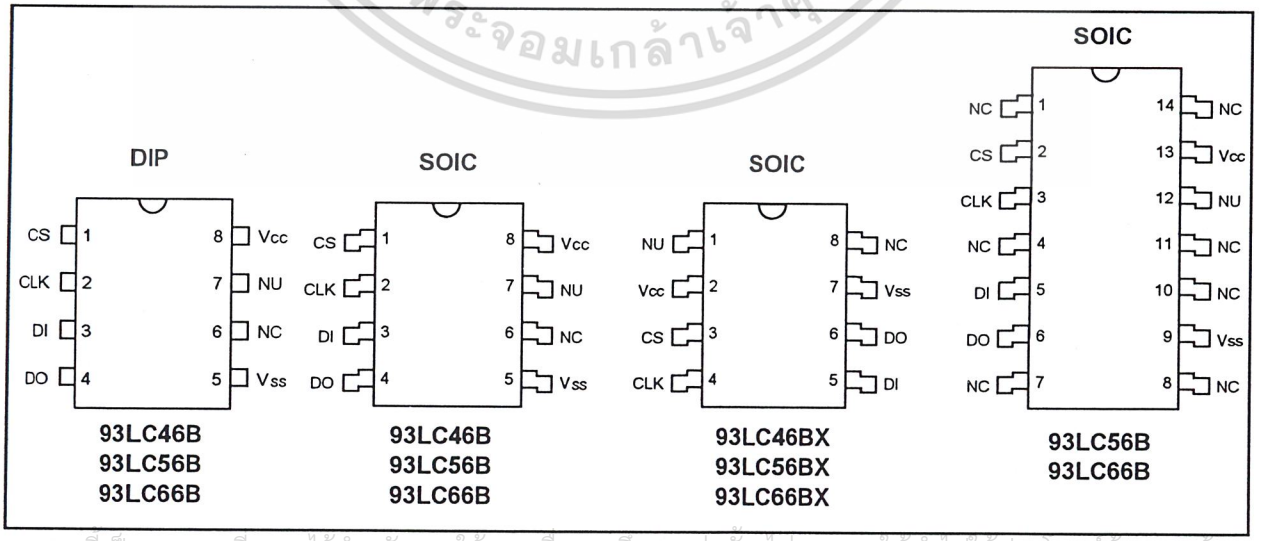
DESCRIPTION

The Microchip Technology Inc. 93LC46B/56B/66B are 1K, 2K and 4K low voltage serial Electrically Erasable PROMs. The device memory is configured as x16. Advanced CMOS technology makes these devices ideal for low power non-volatile memory applications. The 93LC Series is available in standard 8-pin DIP and 8/14-pin surface mount SOIC packages.

BLOCK DIAGRAM



PACKAGE TYPE



**Future: 10,000,000 E/W cycles guaranteed

1.0 ELECTRICAL CHARACTERISTICS

1.1 Maximum Ratings*

Vcc.....7.0V
All inputs and outputs w.r.t. Vss ....-0.6V to Vcc +1.0V
Storage temperature .....-65 C to +150 C
Ambient temp. with power applied .....-65 C to +125 C
Soldering temperature of leads (10 seconds) ...+300 C
ESD protection on all pins..... 4 kV

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

TABLE 1-1: PIN FUNCTION TABLE

Name	Function
CS	Chip Select
CLK	Serial Data Clock
DI	Serial Data Input
DO	Serial Data Output
Vss	Ground
NC	No Connect
NU	Not Utilized
Vcc	Power Supply

TABLE 1-2: DC AND AC ELECTRICAL CHARACTERISTICS

	Commercial Industrial	(C): Vcc = +2.0V to +6.0V (I): Vcc = +2.5V to +6.0V			(C): Tamb = 0 C to +70 C (I): Tamb = -40 C to +85 C
Parameter	**Symbol**	**Min**	**Max**	**Units**	**Conditions**
High level input voltage	VIH1	2.0	Vcc + 1	V	Vcc ≥ 2.7V
	VIH2	0.7 Vcc	Vcc + 1	V	Vcc < 2.7V
Low level input voltage	VIL1	-0.3	0.8	V	Vcc ≥ 2.7V
	VIL2	-0.3	0.2 Vcc	V	Vcc < 2.7V
Low level output voltage	VOL1	—	0.4	V	IOL = 2.1 mA; Vcc = 4.5V
	VOL2	—	0.2	V	IOL = 100 µA; Vcc = Vcc Min.
High level output voltage	VOH1	2.4	—	V	IOH = -400 µA; Vcc = 4.5V
	VOH2	Vcc-0.2	—	V	IOH = -100 µA; Vcc = Vcc Min.
Input leakage current	ILI	-10	10	µA	VIN = 0.1V to Vcc
Output leakage current	ILO	-10	10	µA	VOUT = 0.1V to Vcc
Pin capacitance (all inputs/outputs)	CIN, COUT	—	7	pF	VIN/VOUT = 0 V (Note 1 & 3) Tamb = +25 C, FCLK = 1 MHz
Operating current	ICC write	—	3	mA	FCLK = 2 MHz; Vcc = 6.0V (Note 3)
	ICC read	—	1 500	mA µA	FCLK = 2 MHz; Vcc = 6.0V FCLK = 1 MHz; Vcc = 3.0V
Standby current	ICCS	—	100 30	µA µA	CLK = CS = 0V; Vcc = 6.0V CLK = CS = 0V; Vcc = 3.0V
Clock frequency	FCLK	—	2 1	MHz MHz	Vcc ≥ 4.5V Vcc < 4.5V
Clock high time	TCKH	250	—	ns	
Clock low time	TCKL	250	—	ns	
Chip select setup time	TCSS	50	—	ns	Relative to CLK
Chip select hold time	TCSH	0	—	ns	Relative to CLK
Chip select low time	TCSL	250	—	ns	
Data input setup time	TDIS	100	—	ns	Relative to CLK
Data input hold time	TDIH	100	—	ns	Relative to CLK
Data output delay time	TPD	—	400	ns	CL= 100 pF
Data output disable time	TCZ	—	100	ns	CL = 100 pF (Note 3)
Status valid time	TSV	—	500	ns	CL= 100 pF
Program cycle time	TWC	—	10	ms	ERASE/WRITE mode (Note 2)
	TEC	—	15	ms	ERAL mode
	TWL	—	30	ms	WRAL mode

Note 1: This parameter is tested at tamb = 25 C and FCLK = 1 MHz.
Note 2: Typical program cycle time is 4 ms per word.
Note 3: This parameter is periodically sampled and not 100% tested.

TABLE 1-3: INSTRUCTION SET FOR 93LC46B

Instruction	SB	Opcode	Address	Data In	Data Out	Req. CLK Cycles
READ	1	10	A5 A4 A3 A2 A1 A0	—	D15 - D0	25
EWEN	1	00	1 1 X X X X	—	High-Z	9
ERASE	1	11	A5 A4 A3 A2 A1 A0	—	(RDY/$\overline{\text{BSY}}$)	9
ERAL	1	00	1 0 X X X X	—	(RDY/$\overline{\text{BSY}}$)	9
WRITE	1	01	A5 A4 A3 A2 A1 A0	D15 - D0	(RDY/$\overline{\text{BSY}}$)	25
WRAL	1	00	0 1 X X X X	D15 - D0	(RDY/$\overline{\text{BSY}}$)	25
EWDS	1	00	0 0 X X X X	—	High-Z	9

TABLE 1-4: INSTRUCTION SET FOR 93LC56B

Instruction	SB	Opcode	Address	Data In	Data Out	Req. CLK Cycles
READ	1	10	X A6 A5 A4 A3 A2 A1 A0	—	D15 - D0	27
EWEN	1	00	1 1 X X X X X X	—	High-Z	11
ERASE	1	11	X A6 A5 A4 A3 A2 A1 A0	—	(RDY/$\overline{\text{BSY}}$)	11
ERAL	1	00	1 0 X X X X X X	—	(RDY/$\overline{\text{BSY}}$)	11
WRITE	1	01	X A6 A5 A4 A3 A2 A1 A0	D15 - D0	(RDY/$\overline{\text{BSY}}$)	27
WRAL	1	00	0 1 X X X X X X	D15 - D0	(RDY/$\overline{\text{BSY}}$)	27
EWDS	1	00	0 0 X X X X X X	—	High-Z	11

TABLE 1-5: INSTRUCTION SET FOR 93LC66B

Instruction	SB	Opcode	Address	Data In	Data Out	Req. CLK Cycles
READ	1	10	A7 A6 A5 A4 A3 A2 A1 A0	—	D15 - D0	27
EWEN	1	00	1 1 X X X X X X	—	High-Z	11
ERASE	1	11	A7 A6 A5 A4 A3 A2 A1 A0	—	(RDY/$\overline{\text{BSY}}$)	11
ERAL	1	00	1 0 X X X X X X	—	(RDY/$\overline{\text{BSY}}$)	11
WRITE	1	01	A7 A6 A5 A4 A3 A2 A1 A0	D15 - D0	(RDY/$\overline{\text{BSY}}$)	27
WRAL	1	00	0 1 X X X X X X	D15 - D0	(RDY/$\overline{\text{BSY}}$)	27
EWDS	1	00	0 0 X X X X X X	—	High-Z	11

2.0 FUNCTIONAL DESCRIPTION

Instructions, addresses and write data are clocked into the DI pin on the rising edge of the clock (CLK). The DO pin is normally held in a high-Z state except when reading data from the device, or when checking the ready/busy status during a programming operation. The ready/busy status can be verified during an Erase/Write operation by polling the DO pin; DO low indicates that programming is still in progress, while DO high indicates the device is ready. The DO will enter the high-Z state on the falling edge of the CS.

2.1 START Condition

The START bit is detected by the device if CS and DI are both HIGH with respect to the positive edge of CLK for the first time.

Before a START condition is detected, CS, CLK, and DI may change in any combination (except to that of a START condition), without resulting in any device operation (READ, WRITE, ERASE, EWEN, EWDS, ERAL, and WRAL). As soon as CS is HIGH, the device is no longer in the standby mode.

An instruction following a START condition will only be executed if the required amount of opcode, address and data bits for any particular instruction is clocked in.

After execution of an instruction (i.e., clock in or out of the last required address or data bit) CLK and DI become don't care bits until a new start condition is detected.

2.2 DI/DO

It is possible to connect the Data In and Data Out pins together. However, with this configuration it is possible for a "bus conflict" to occur during the "dummy zero" that precedes the READ operation, if A0 is a logic HIGH level. Under such a condition the voltage level seen at Data Out is undefined and will depend upon the relative impedances of Data Out and the signal source driving A0. The higher the current sourcing capability of A0, the higher the voltage at the Data Out pin.

2.3 Data Protection

During power-up, all programming modes of operation are inhibited until Vcc has reached a level greater than 1.4V. During power-down, the source data protection circuitry acts to inhibit all programming modes when Vcc has fallen below 1.4V at nominal conditions.

The EWEN and EWDS commands give additional protection against accidentally programming during normal operation.

After power-up, the device is automatically in the EWDS mode. Therefore, an EWEN instruction must be performed before any ERASE or WRITE instruction can be executed.

3.0 READ

The READ instruction outputs the serial data of the addressed memory location on the DO pin. A dummy zero bit precedes the 16 bit (x16 organization) output string. The output data bits will toggle on the rising edge of the CLK and are stable after the specified time delay (TPD). Sequential read is possible when CS is held high. The memory data will automatically cycle to the next register and output sequentially.

4.0 ERASE/WRITE ENABLE AND DISABLE

The 93LC46B/56B/66B powers up in the Erase/Write Disable (EWDS) state. All programming modes must be preceded by an Erase/Write Enable (EWEN) instruction. Once the EWEN instruction is executed, programming remains enabled until an EWDS instruction is executed or Vcc is removed from the device. To protect against accidental data disturb, the EWDS instruction can be used to disable all Erase/Write functions and should follow all programming operations. Execution of a READ instruction is independent of both the EWEN and EWDS instructions.

5.0 ERASE

The ERASE instruction forces all data bits of the specified address to the logical "1" state. CS is brought low following the loading of the last address bit. This falling edge of the CS pin initiates the self-timed programming cycle.

The DO pin indicates the READY/$\overline{\text{BUSY}}$ status of the device if CS is brought high after a minimum of 250 ns low (TCSL). DO at logical "0" indicates that programming is still in progress. DO at logical "1" indicates that the register at the specified address has been erased and the device is ready for another instruction.

The ERASE cycle takes 4 ms per word (Typical).

6.0 WRITE

The WRITE instruction is followed by 16 bits of data which are written into the specified address. After the last data bit is put on the DI pin, CS must be brought low before the next rising edge of the CLK clock. This falling edge of CS initiates the self-timed auto-erase and programming cycle.

The DO pin indicates the READY/$\overline{\text{BUSY}}$ status of the device if CS is brought high after a minimum of 250 ns low (TCSL) and before the entire write cycle is complete. DO at logical "0" indicates that programming is still in progress. DO at logical "1" indicates that the register at the specified address has been written with the data specified and the device is ready for another instruction.

The WRITE cycle takes 4 ms per word (Typical).

7.0 ERASE ALL

The ERAL instruction will erase the entire memory array to the logical "1" state. The ERAL cycle is identical to the ERASE cycle except for the different opcode. The ERAL cycle is completely self-timed and commences at the falling edge of the CS. Clocking of the CLK pin is not necessary after the device has entered the self clocking mode. The ERAL instruction is guaranteed at Vcc = +4.5V to +6.0V.

The DO pin indicates the READY/$\overline{\text{BUSY}}$ status of the device if CS is brought high after a minimum of 250 ns low (TCSL) and before the entire write cycle is complete.

The ERAL cycle takes 15 ms maximum (8 ms typical).

8.0 WRITE ALL

The WRAL instruction will write the entire memory array with the data specified in the command. The WRAL cycle is completely self-timed and commences at the falling edge of the CS. Clocking of the CLK pin is not necessary after the device has entered the self clocking mode. The WRAL command does include an automatic ERAL cycle for the device. Therefore, the WRAL instruction does not require an ERAL instruction but the chip must be in the EWEN status. The WRAL instruction is guaranteed at Vcc = +4.5V to +6.0V.

The DO pin indicates the READY/$\overline{\text{BUSY}}$ status of the device if CS is brought high after a minimum of 250 ns low (TCSL).

The WRAL cycle takes 30 ms maximum (16 ms typical).

9.0 PIN DESCRIPTION

9.1 Chip Select (CS)

A HIGH level selects the device. A LOW level deselects the device and forces it into standby mode. However, a programming cycle which is already initiated and/or in progress will be completed, regardless of the CS input signal. If CS is brought LOW during a program cycle, the device will go into standby mode as soon as the programming cycle is completed.

CS must be LOW for 250 ns minimum (TCSL) between consecutive instructions. If CS is LOW, the internal control logic is held in a RESET status.

9.2 Serial Clock (CLK)

The Serial Clock is used to synchronize the communication between a master device and the 93LCXXB. Opcode, address, and data bits are clocked in on the positive edge of CLK. Data bits are also clocked out on the positive edge of CLK.

CLK can be stopped anywhere in the transmission sequence (at HIGH or LOW level) and can be continued anytime with respect to clock HIGH time (TCKH) and clock LOW time (TCKL). This gives the controlling master freedom in preparing opcode, address, and data.

CLK is a "Don't Care" if CS is LOW (device deselected). If CS is HIGH, but START condition has not been detected, any number of clock cycles can be received by the device without changing its status (i.e., waiting for START condition).

CLK cycles are not required during the self-timed WRITE (i.e., auto ERASE/WRITE) cycle.

After detection of a start condition the specified number of clock cycles (respectively LOW to HIGH transitions of CLK) must be provided. These clock cycles are required to clock in all required opcode, address, and data bits before an instruction is executed (see instruction set truth table). CLK and DI then become don't care inputs waiting for a new start condition to be detected.

Note:	CS must go LOW between consecutive instructions.

9.3 Data In (DI)

Data In is used to clock in a START bit, opcode, address, and data synchronously with the CLK input.

9.4 Data Out (DO)

Data Out is used in the READ mode to output data synchronously with the CLK input (TPD after the positive edge of CLK).

This pin also provides READY/$\overline{\text{BUSY}}$ status information during ERASE and WRITE cycles. READY/$\overline{\text{BUSY}}$ status information is available on the DO pin if CS is brought HIGH after being LOW for minimum chip select LOW time (TCSL) and an ERASE or WRITE operation has been initiated.

The status signal is not available on DO, if CS is held LOW or HIGH during the entire WRITE or ERASE cycle. In all other cases DO is in the HIGH-Z mode. If status is checked after the WRITE/ERASE cycle, a pull-up resistor on DO is required to read the READY signal.

FIGURE 9-1: SYNCHRONOUS DATA TIMING

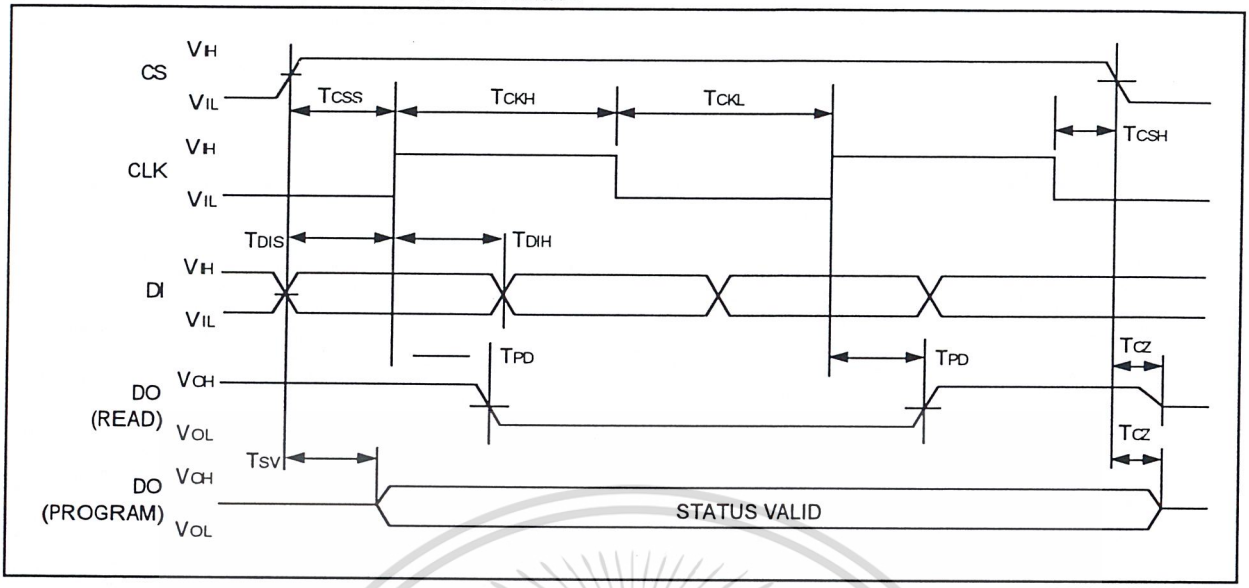


FIGURE 9-2: READ TIMING

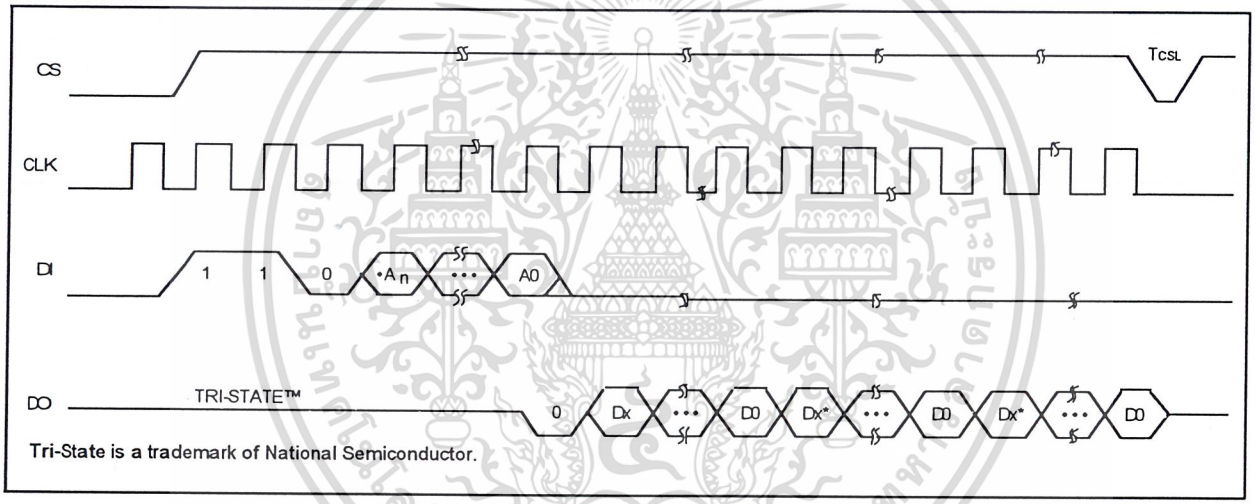


FIGURE 9-3: EWEN TIMING

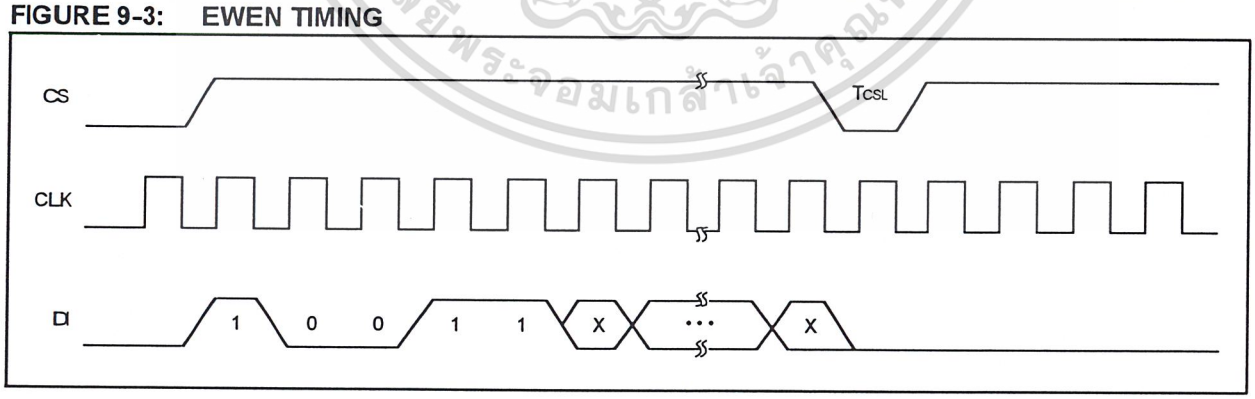


FIGURE 9-4: EWDS TIMING

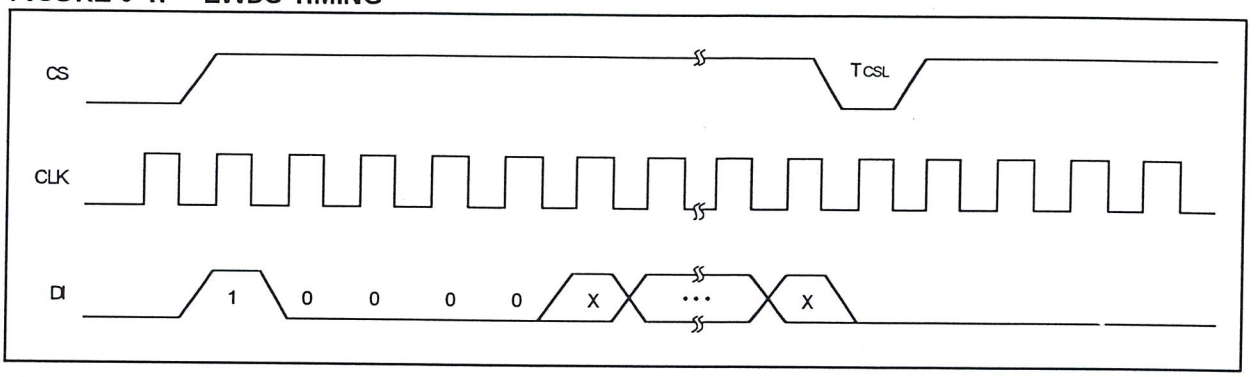


FIGURE 9-5: WRITE TIMING

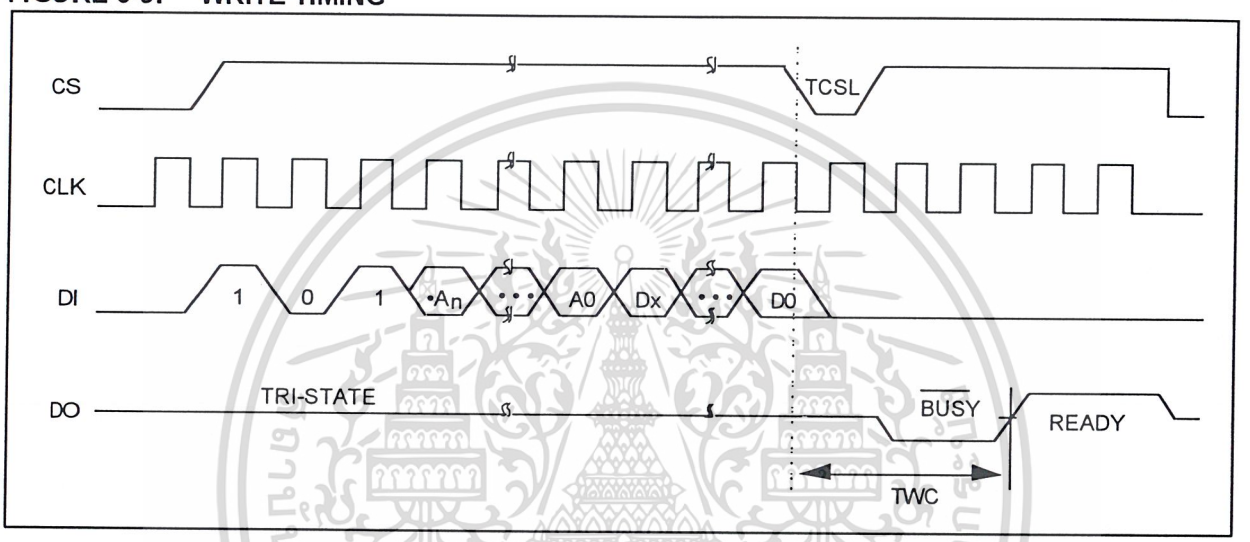
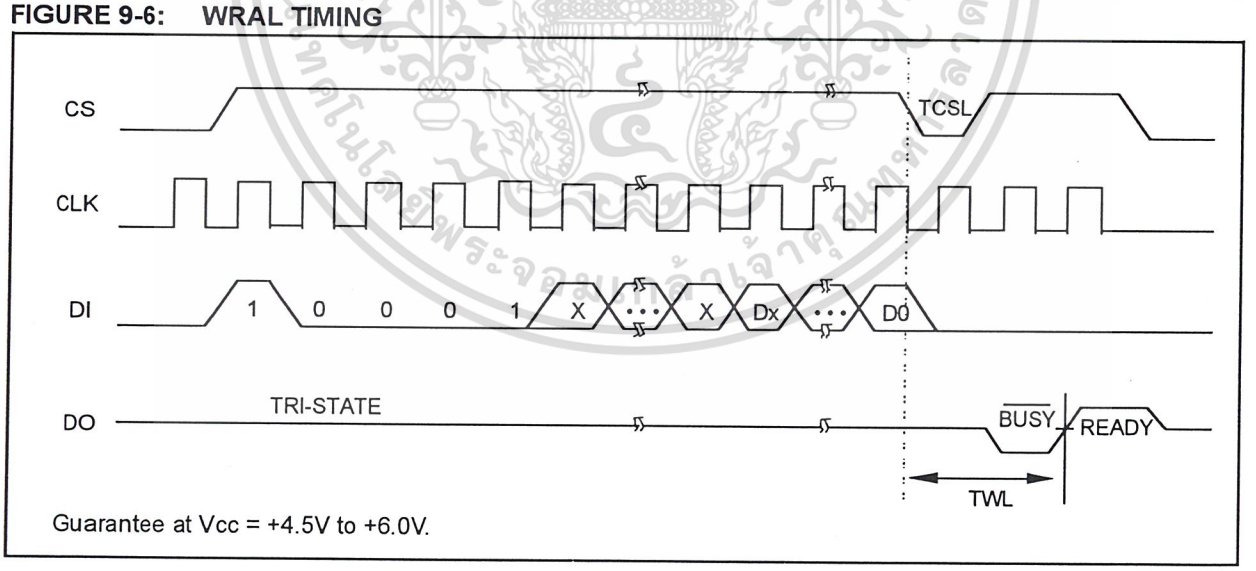


FIGURE 9-6: WRAL TIMING



Guarantee at Vcc = +4.5V to +6.0V.

FIGURE 9-7: ERASE TIMING

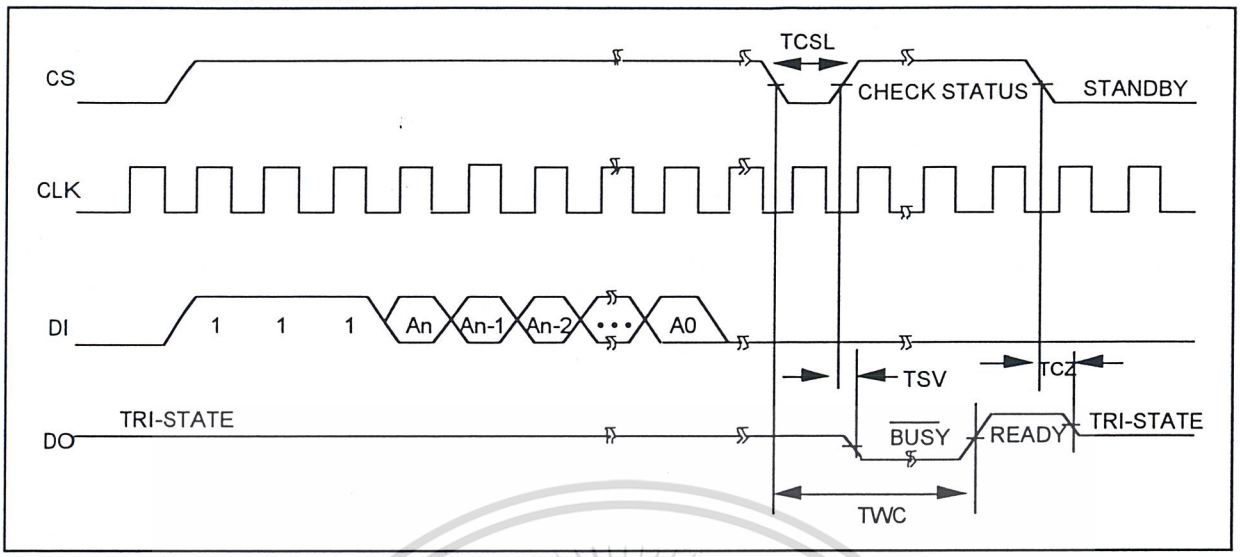
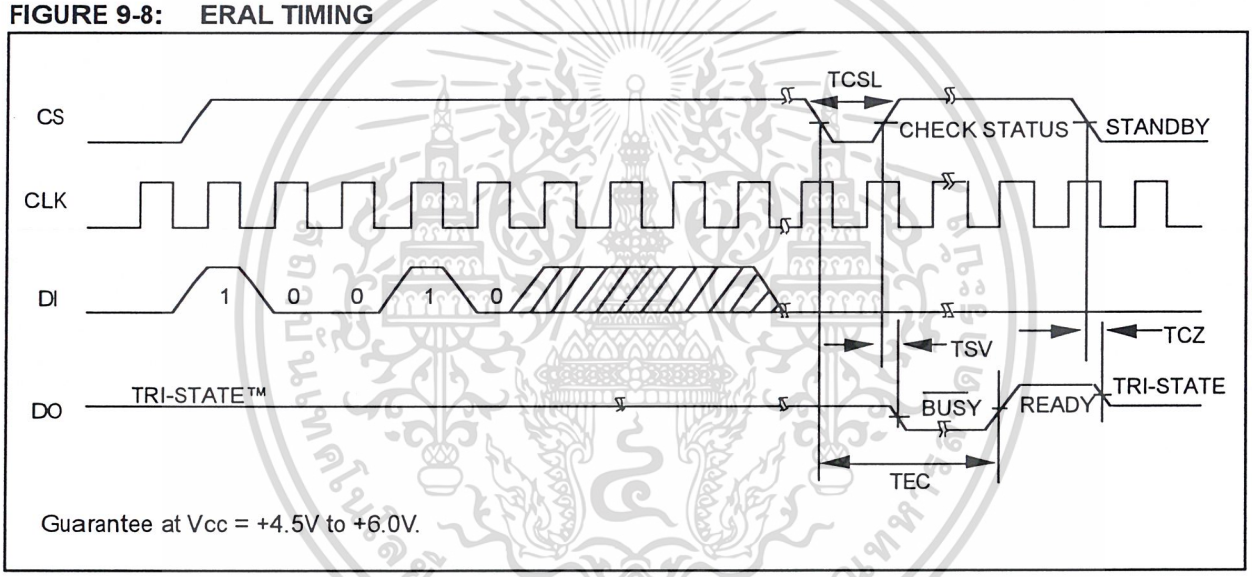


FIGURE 9-8: ERAL TIMING



93LC46B/56B/66B Product Identification System

To order or to obtain information, e.g., on pricing or delivery, please use the listed part numbers, and refer to the factory or the listed sales offices.

93LC46B/56B/66B - /P

Package:
- P = Plastic DIP (300 mil Body), 8-lead
- SN = Plastic SOIC (150 mil Body), 8-lead
- SM = Plastic SOIC (207 mil Body), 8-lead
- SL = Plastic SOIC (150 mil Body), 14-lead (93LC56B/93LC66B)

Temperature Range:
- Blank = 0°C to +70°C
- I = -40°C to +85°C

Device:

	Configuration
93LC46B/56B/66B	CMOS Serial EEPROM
93LC46BX/56BX/66BX	CMOS Serial EEPROM in alternate pinouts (SN package only)
93LC46BT/56BT/66BT	CMOS Serial EEPROM (Tape and Reel)
93LC46BXT/56BXT/66BXT	CMOS Serial EEPROM (Tape and Reel)

AMERICAS

Corporate Office
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 602 786-7200 Fax: 602 786-7277
Technical Support: 602 786-7627
Web: http://www.mchip.com/biz/mchip

Atlanta
Microchip Technology Inc.
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770 640-0034 Fax: 770 640-0307

Boston
Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508 480-9990 Fax: 508 480-8575

Chicago
Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 708 285-0071 Fax: 708 285-0075

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 214 991-7177 Fax: 214 991-8588

Dayton
Microchip Technology Inc.
35 Rockridge Road
Englewood, OH 45322
Tel: 513 832-2543 Fax: 513 832-2841

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 455
Irvine, CA 92715
Tel: 714 263-1888 Fax: 714 263-1338

New York
Microchip Technology Inc.
150 Motor Parkway, Suite 416
Hauppauge, NY 11788
Tel: 516 273-5305 Fax: 516 273-5335

AMERICAS (continued)

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408 436-7950 Fax: 408 436-7955

ASIA/PACIFIC

Hong Kong
Microchip Technology
Unit No. 3002-3004, Tower 1
Metroplaza
223 Hing Fong Road
Kwai Fong, N.T. Hong Kong
Tel: 852 2 401 1200 Fax: 852 2 401 3431

Korea
Microchip Technology
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku,
Seoul, Korea
Tel: 82 2 554 7200 Fax: 82 2 558 5934

Singapore
Microchip Technology
200 Middle Road
#10-03 Prime Centre
Singapore 188980
Tel: 65 334 8870 Fax: 65 334 8850

Taiwan
Microchip Technology
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886 2 717 7175 Fax: 886 2 545 0139

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
Unit 6, The Courtyard
Meadow Bank, Furlong Road
Bourne End, Buckinghamshire SL8 5AJ
Tel: 44 0 1628 851077 Fax: 44 0 1628 850259

France
Arizona Microchip Technology SARL
2 Rue du Buisson aux Fraises
91300 Massy - France
Tel: 33 1 69 53 63 20 Fax: 33 1 69 30 90 79

Germany
Arizona Microchip Technology GmbH
Gustav-Heinemann-Ring 125
D-81739 Muenchen, Germany
Tel: 49 89 627 144 0 Fax: 49 89 627 144 44

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Pegaso Ingresso No. 2
Via Paracelso 23, 20041
Agrate Brianza (MI) Italy
Tel: 39 039 689 9939 Fax: 39 039 689 9883

JAPAN

Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shin Yokohama
Kohoku-Ku, Yokohama
Kanagawa 222 Japan
Tel: 81 45 471 6166 Fax: 81 45 471 6122

9/5/95

Printed in the USA, 9/95