เครื่องอ่านบัตรแม่เหล็กและการประยุกต์

นายธวัชชัย เอื้อมมนัสสกุล
นายนินนท์ จันทร์ดรี

โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตร์บัณฑิต
ภาควิชาฟิสิกส์ประยุกต์
คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2536

Magnetic Card Reader And Its Application

Mr.Thawatchai Iemmanassakul

Mr.Nipon Jantri

A Special Project Submitted in Partial Fulfillment of the

Requirement for the Degree of Bachelor of Science

Department of Applied Physics

Faculty of Science

King Mongkut's Institute of Technology Ladkrabang

1993

หัวข้อโครงงานพิเศษ	เครื่องอ่านบัตรแม่เหล็กและการประยุกต์
โดย	นายธวัชชัย เอื้อมมนัสสกุล
	นายนิพนธ์ จันทร์ตรี
ภาควิชา	ฟิสิกส์ประยุกต์
อาจารย์ที่ปรึกษา	ผศ.ดร.ปรีชา เทียนสมประสงค์

ภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง อนุมัติให้นับโครงงานพิเศษฉบับนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

ปรีชา

_____ หัวหน้าภาควิชาฟิสิกส์ประยุกต์

(ผศ.ดร.ปรีชา เทียนสมประสงค์)

คณะกรรมการโครงงานพิเศษ

_____ ประธานกรรมการ

(ผศ.ดร.ปรีชา เทียนสมประสงค์)

_____ กรรมการ

(ผศ.ดร.เสน่ห์ เอกะวิภาต)

_____ กรรมการ

(ผศ.สุวรรณ คูสำราญ)

หัวข้อโครงงานพิเศษ	เครื่องอ่านบัตรแม่เหล็กและการประยุกต์
นักศึกษา	นายธวัชชัย เอี่ยมมนัสสกุล 33504012
	นายนิพนธ์ จันทร์ตรี 33504016
อาจารย์ที่ปรึกษา	ผศ.ดร.ปรีชา เทียนสมประสงค์
ภาควิชา	ฟิสิกส์ประยุกต์
ปีการศึกษา	2536

บทคัดย่อ

โครงงานพิเศษนี้เป็นการศึกษาหลักการทำงานของเครื่องอ่านบัตรแม่เหล็ก และการประยุกต์ใช้ในด้านการรักษาความปลอดภัยของการสื่อสารข้อมูล โดยทำการสร้างเป็นระบบรักษาความปลอดภัยในการเข้าถึงข้อมูลโดยใช้บัตรแม่เหล็ก (Magnetic Login System) ในการดำเนินงานผู้จัดทำได้สร้าง เครื่องอ่านบัตรแม่เหล็กตามระบบมาตรฐาน โมเด็มเพื่อใช้ในการส่งข้อมูลผ่านทางสายโทรศัพท์ และ ระบบควบคุมการทำงานของอุปกรณ์โดยใช้ไมโครคอนโทรลเลอร์ 8031 ระบบนี้สามารถนำไปใช้งานเสริมระบบสื่อสารข้อมูลเดิมเพื่อสร้างความปลอดภัยให้เพิ่มขึ้น

Special Project Title	Magnetic card reader and its application
Name	Mr.Thawatchai Iemmanassakul 33504012
	Mr.Nipon Jantri 33504016
Special Project Advisor	Asst.Prof.Dr.Preecha Teansomprasong
Department	Applied Physics
Academic Year	1993

Abstract

This special project is the study of the magnetic card reader and its application in the data communication security. It is applied for the Magnetic Login System that allow priviledged user to access this system by using his magnetic card and password. Standard magnetic card reader, modem and control unit are constructed and assembled for this system. It can be used to add in conventional data communication system to enhance system security.

กิตติกรรมประกาศ

ในการจัดทำโครงงานพิเศษนี้ ผู้ทำโครงงานได้รับความอนุเคราะห์ช่วยเหลือในด้านต่างๆ ตั้งแต่เริ่มต้นจนกระทั่งสำเร็จลุล่วงไปด้วยดี จากบุคคลต่างๆ ดังรายนามต่อไปนี้

คุณพ่อและคุณแม่	ให้ความอุปการะในทุกๆ ด้านจนสำเร็จการศึกษา
ผศ.ดร.ปรีชา เทียนสมประสงค์	ให้คำแนะนำ และคอยดูแลเอาใจใส่ในการทำโครงงานพิเศษนี้
นายชานินทร์ จันทร์ตรี (พี่ตุ้ม)	ให้คำแนะนำปรึกษา ให้ความห่วงใยและให้ความช่วยเหลือในการติดตั้งอุปกรณ์ลงกล่อง
อ.วิชิต ศิริโชติ	ให้คำแนะนำและให้ยืม EPROM Emulator
อ.ธวัชชัย ขาวประเสริฐ	เอื้อเฟื้อสถานที่ในการทำโครงงาน
อ.สนิท หมอกมืด	ให้ยืมเครื่องมือและอุปกรณ์ต่างๆ ในการทำโครงงาน
นายสมภพ ภูริวิกรัยพงศ์ (พี่ภพ)	ให้คำแนะนำปรึกษา
นายบุญชัย พจนาสมสมาน	ให้คำแนะนำและให้ยืม EPROM Emulator
นายเอกราช โล่ห์เพชรัตน์	ช่วยถ่ายรูปและให้กำลังใจ
นายจักรกฤษณ์ จูเจริญ	ให้คำแนะนำช่วยเหลือด้านการจัดพิมพ์ และให้ความห่วงใยถามไถ่เสมอ
นายเสริมชัย เกียรติยศนากุล	ให้คำแนะนำ
เพื่อนๆ พี่ๆ น้องๆ ทุกคน	ให้กำลังใจและความช่วยเหลือในทุกๆ ด้าน
คณะกรรมการทุกท่าน	ให้คำแนะนำ
ภาควิชาฟิสิกส์ประยุกต์	ให้ความอนุเคราะห์ในด้านต่างๆ

ผู้จัดทำโครงงานขอขอบคุณบุคคลเหล่านี้อย่างจริงใจ

ธวัชชัย เอื้อมมนัสสกุล

นิพนธ์ จันทร์ตรี

สารบัญ

สารบัญรูป

หน้า

สารบัญตาราง

หน้า

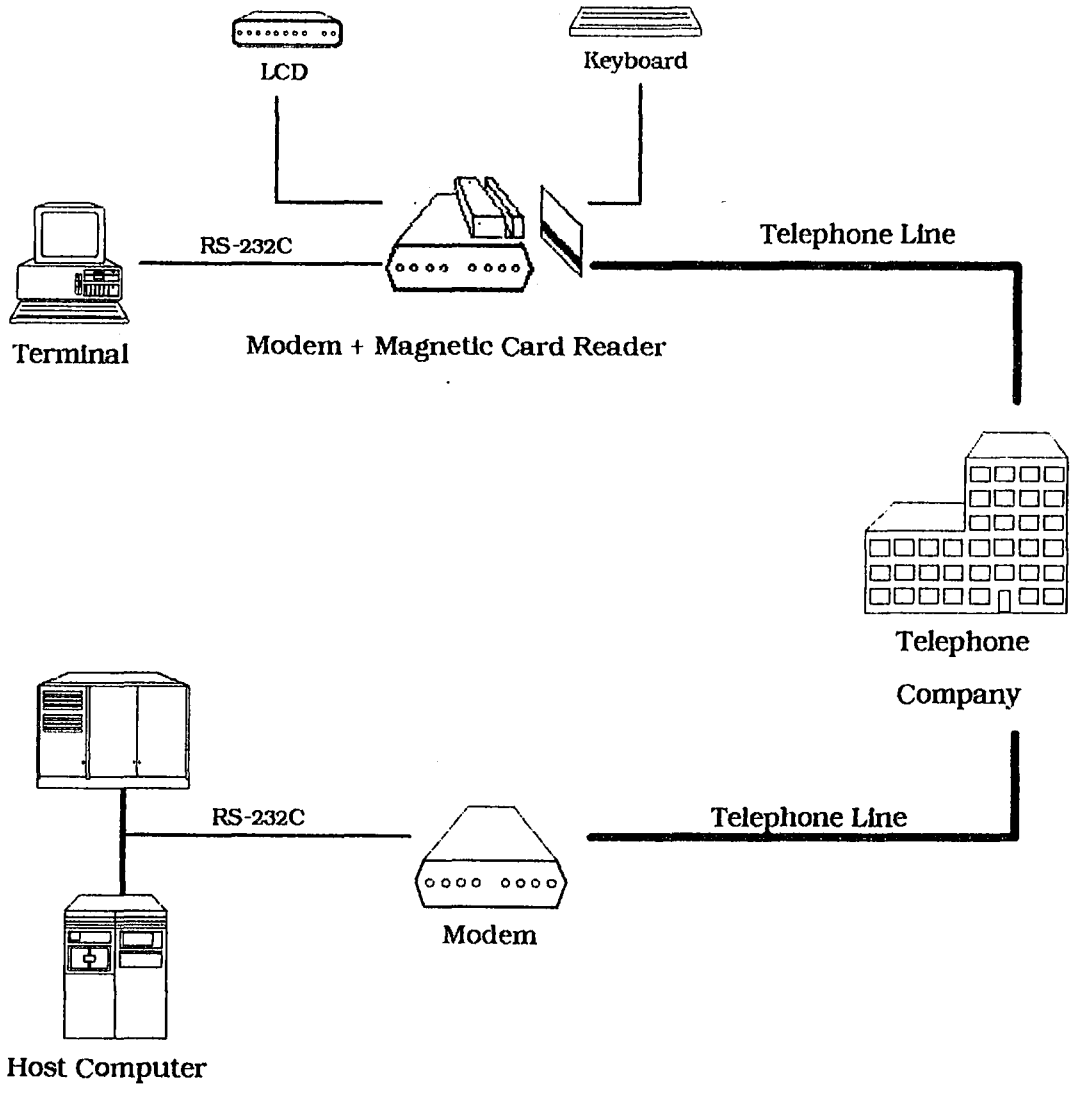
บทที่ 1
บทนำ

เทคโนโลยีทางด้านการสื่อสารที่ได้รุดหน้าตามความเจริญแห่งยุคที่ก้าวไปอย่างรวดเร็วอย่างในปัจจุบัน ทำให้เกิดอุปกรณ์ทางอิเล็กทรอนิคทางด้านการสื่อสารมากมาย ทั้งด้านอนาล็อกและดิจิตอล หนึ่งในอุปกรณ์ที่ใช้ในด้านการสื่อสารที่มีความสำคัญมาก ได้แก่ อุปกรณ์ที่ใช้ในการรับส่งข้อมูลทางดิจิตอล เช่น โมเด็ม แฟกซ์ จนกระทั่งถึงระบบเน็ตเวิร์ค แต่สิ่งที่ต้องคำนึงถึงในการแลกเปลี่ยนข้อมูลผ่านทางอุปกรณ์อิเล็กทรอนิคนั้น นอกจากความเร็วในการรับส่งข้อมูลและระยะทางที่ข้อมูลสามารถไปได้ไกลที่สุดแล้ว ยังควรจะพิจารณาถึงความปลอดภัยของระบบที่จะต้องอนุญาติให้ข้อมูลไปถึงมือผู้รับเฉพาะผู้ที่ต้องการข้อมูลซึ่งมีสิทธิ์ในข้อมูลดังกล่าวเท่านั้น จากแนวคิดดังกล่าวนี้นำมาซึ่งการรักษาความปลอดภัยโดยใช้ระบบการเข้าถึงข้อมูลโดยใช้บัตรแม่เหล็ก (Magnetic Login System)

Magnetic Login System จะมีลักษณะคล้ายกับระบบรับส่งข้อมูลแบบดิจิตอลผ่านทางสายโทรศัพท์ มีการใช้อุปกรณ์ที่เรียกว่า โมเด็ม (Modem) เป็นตัวแปลงสัญญาณทางดิจิตอลจากคอมพิวเตอร์ที่เป็นแหล่งข้อมูล (host) เป็นสัญญาณทางอนาล็อกผ่านสายโทรศัพท์ สัญญาณทางอนาล็อกนี้จะถูกแปลงกลับเป็นสัญญาณทางดิจิตอลโดยใช้โมเด็มอีกครั้ง เพื่อส่งให้คอมพิวเตอร์ที่ต้องการข้อมูล (terminal) รับข้อมูลตามที่ได้เรียกร้องไป แต่การเรียกร้องข้อมูลนี้จะไม่ได้รับการตอบสนองเลยถ้าผู้ใช้ไม่มีบัตรแม่เหล็กและรหัสผ่าน อันเป็นสิ่งที่ยืนยันว่าผู้ที่ต้องการข้อมูลนี้มีสิทธิ์ในการรับข้อมูลดังกล่าวตามที่ได้ตกลงกันไว้ล่วงหน้า จุดนี้เองที่จะช่วยรักษาความปลอดภัยของระบบมากกว่าการใช้เพียงรหัสผ่านซึ่งเป็นแบบที่ใช้กันอยู่ในปัจจุบัน

จากรูปที่ 1.1 จะเห็นว่า เมื่อผู้ใช้ต้องการข้อมูลจากศูนย์ข้อมูลจากศูนย์ข้อมูล (ในรูปนี้เราจะใช้ ระบบคอมพิวเตอร์ขนาดใหญ่ เพื่อแสดงให้เห็นว่าระบบนี้สามารถขยายได้ถึงระดับหนึ่ง) ผู้ใช้ก็จะเริ่มต้นรูดบัตรแม่เหล็กพร้อมกับกดรหัสผ่าน ข้อมูลที่เฉพาะในบัตรแม่เหล็กที่อ่านได้กับรหัสผ่านจะถูกส่งผ่านโมเด็มเพื่อไปตรวจสอบระดับของสิทธิ์ที่จะใช้ระบบ (privilege) ถ้าบัตรแม่เหล็กและรหัสถูกต้อง ผู้ใช้จะได้รับอนุญาตให้ใช้ terminal ในการติดต่อเพื่อรับข้อมูลตามที่ได้ตกลงกันไว้ หรือถ้าต้องการตรวจสอบหรือยืนยันข้อมูลเพียงเล็กน้อย เช่น ตรวจสอบว่ามีผู้ใช้ชื่อนี้อยู่ในฐานข้อมูลหรือไม่ ก็อาจติดต่อผ่านจอ LCD และคีย์บอร์ดขนาดเล็กเพื่อไม่ต้องใช้คอมพิวเตอร์ที่จะต้องเสียค่าใช้จ่ายมากกว่าถ้าใช้ในงานนี้เพียงอย่างเดียว

รูปที่ 1.1 แสดงผังการเชื่อมต่อทั้งหมดของระบบเข้าถึงข้อมูลโดยใช้บัตรแม่เหล็ก



1.1 ส่วนต่างๆ ของระบบเข้าถึงข้อมูลโดยใช้บัตรแม่เหล็ก

Magnetic Login System สามารถแบ่งเป็นส่วนๆ ได้ดังนี้

1. **Magnetic Card Reader** ทำการอ่านข้อมูลจากบัตรแม่เหล็ก ข้อมูลที่อ่านได้จากบัตรแม่เหล็กกับรหัสผ่าน จะเป็นสิ่งที่ตรวจสอบว่าผู้ที่กำลังเข้ามาใช้ระบบนี้มีสิทธิ์ถูกต้องหรือไม่ โดยการตรวจสอบจะเกิดขึ้นที่ host เท่านั้น ทั้งนี้เพื่อให้ระบบมีความปลอดภัยสูงขึ้น ในส่วนของ Magnetic Card Reader นี้จะมีเฉพาะทางฝ่ายของ terminal เพราะส่วนนี้จะเป็นส่วนที่ใช้เฉพาะการเริ่มต้นติดต่อเท่านั้น

2. **Modem** รับข้อมูลที่เป็นดิจิตอลแล้วแปลงเป็นสัญญาณทางอนาล็อก เพื่อจะได้ส่งผ่านทางสายโทรศัพท์ได้ ในส่วนของการรับข้อมูลก็จะกระทำในทางตรงกันข้าม คือ แปลงจากสัญญาณทางอนาล็อกเป็นดิจิตอล การรับส่งข้อมูลของโมเด็มนี้จะเป็นแบบ Full Duplex คือ สามารถรับส่งได้พร้อมกันทั้งสองด้าน แต่อัตราเร็วในการรับส่งข้อมูล (baud rate) ทั้งสองด้านจะไม่เท่ากัน ทางด้าน host จะส่งข้อมูลไปยัง terminal ด้วยอัตราเร็ว 1,200 bit/sec ในขณะที่ฝ่าย terminal จะตอบกลับด้วยอัตราเร็ว 150 bit/sec ซึ่งเพียงพอสำหรับใช้งานได้ในระดับหนึ่ง

3. **Control Unit** ทำหน้าที่ควบคุมการทำงานในแต่ละด้าน หัวใจของส่วนนี้ คือ Micro controller 8031 ซึ่งทำหน้าที่กำหนดการทำงานของส่วนต่าง ๆ ให้เป็นไปตามกำหนด ตัวอย่างเช่น ควบคุมโมเด็มให้ทำการยกหูโทรศัพท์ (off-hook) และ ทำการสร้างสัญญาณ DTMF เพื่อระบุจุดหมายปลายทางที่จะติดต่อ ควบคุมการทำงานของ LCD เพื่อแจ้งข้อมูลให้กับผู้ใช้ ควบคุมคีย์บอร์ดเพื่อรับข้อมูลในการทำงานอย่างเป็นขั้นตอน นอกจากนี้ยังควบคุมการทำงานของ Magnetic Card Reader ให้อ่านข้อมูลตามลำดับอย่างถูกต้อง แล้วทำการส่งข้อมูลนี้ไปยังส่วนอื่นๆ ต่อไป

ส่วนของ LCD และคีย์บอร์ดนี้ จะมีการใช้งานต่อเมื่อ terminal ไม่ได้ต่อกับคอมพิวเตอร์ การทำงานในลักษณะนี้จะใช้ LCD และคีย์บอร์ด ในการติดต่อและแสดงข้อมูลที่รับจากด้าน host แต่ถ้าระบบต่อกับคอมพิวเตอร์แล้ว การควบคุมทั้งหมดจะกระทำผ่านทางคอมพิวเตอร์ ทั้งนี้เพื่ออำนวยความสะดวกแก่ผู้ใช้เมื่อมีคอมพิวเตอร์ และทำให้ระบบยืดหยุ่นมากขึ้นสำหรับการควบคุมและแสดงผลทางคีย์บอร์ด และ LCD ในกรณีที่ไม่ได้ใช้คอมพิวเตอร์

1.2 ข้อดีของระบบเข้าถึงข้อมูลโดยใช้บัตรแม่เหล็ก

1. เป็นการประยุกต์ระบบอ่านบัตรแม่เหล็กมาใช้กับระบบรับส่งข้อมูลโดยใช้โมเด็มเพื่อสร้างระบบรวมที่มีความปลอดภัยสูงขึ้น

2. สามารถให้ผู้ใช้ที่มีแต่รหัสผ่านหรือบัตรแม่เหล็กอย่างใดอย่างหนึ่ง สามารถเข้าถึงข้อมูลเบื้องต้นที่ไม่ได้มีความสำคัญมากนักได้ เป็นการยืดหยุ่นให้ระบบกรณีที่ผู้ใช้ทำบัตรแม่เหล็กหายหรือลืมรหัสผ่าน

3. ผู้ใช้ไม่จำเป็นต้องมีการป้อนชื่อสำหรับการเข้าใช้ระบบ เพราะศูนย์ข้อมูลจะทราบชื่อได้จากบัตรแม่เหล็กอยู่แล้ว เป็นการอำนวยความสะดวกแก่ผู้ใช้ในด้านหนึ่ง

4. สามารถควบคุมและจำกัดจำนวนผู้ใช้ข้อมูลได้โดยง่าย

1.3 ข้อเสียของระบบเข้าถึงข้อมูลโดยใช้บัตรแม่เหล็ก

1. เสียค่าใช้จ่ายเพิ่มขึ้นในการสร้างระบบอ่านบัตรแม่เหล็กเพิ่มเข้าไปจากระบบเดิม

2. มีความซับซ้อนในการเขียนโปรแกรมควบคุมระบบ ซึ่งถ้าโปรแกรมที่เขียนไม่รัดกุมเพียงพอจะทำให้ระบบขาดความปลอดภัย

1.4 ประโยชน์ที่จะได้รับในการสร้างโครงงาน

1. ได้รับความรู้ต่างๆ เพิ่มขึ้นหลายด้าน โดยเฉพาะการสื่อสารข้อมูลผ่านทางโมเด็ม

2. เพิ่มประสบการณ์และทักษะในการสร้างและออกแบบวงจร

3. เข้าใจระบบไมโครโปรเซสเซอร์มากขึ้น

4. สามารถเขียนโปรแกรมควบคุมพอร์ตอนุกรมจากคอมพิวเตอร์ตระกูลไอบีเอ็มได้

5. รู้จักวางแผนการทำงานอย่างเป็นระบบ เพื่อให้งานดำเนินไปอย่างเป็นขั้นตอน จนกระทั่งสำเร็จ

6. เพิ่มทักษะในการแก้ปัญหาทั้งในส่วนของวงจรและโปรแกรม

1.5 ขั้นตอนในการดำเนินงาน

1. ศึกษาความเป็นไปได้ของโครงงาน ตั้งแต่ส่วนของระบบอ่านบัตรแม่เหล็ก ระบบโมเด็ม และระบบควบคุมการทำงานทั้งหมด

2. เลือกอุปกรณ์ที่จะนำมาใช้งาน โดยพิจารณาทีละระบบ เช่น การสร้างเครื่องอ่านควรจะสร้างระบบเองหรือใช้ไอซีสำเร็จรูป ส่วนของโมเด็มก็จะต้องเลือกไอซีที่ทำหน้าที่เป็นตัวมอดดูเลทและดีมอดดูเลท ซึ่งส่วนนี้จะมีความสำคัญต่อการออกแบบระบบต่อไป

3. ศึกษาการใช้งานของอุปกรณ์แต่ละตัว ข้อจำกัดในการใช้งาน และปัญหาที่อาจจะเกิดขึ้นในการใช้งานอุปกรณ์ตัวนั้น ซึ่งถ้าคาดว่าจะเกิดปัญหา หรือ อุปกรณ์ตัวนั้นอาจจะใช้ไม่ได้สำหรับการสร้างระบบนี้ ก็ต้องไปทำในขั้นตอนที่ 2 ใหม่

4. ออกแบบวงจร โดยจะเริ่มจากการออกแบบระบบอ่านบัตรแม่เหล็กก่อนเพื่อดูว่าจะต้องออกแบบระบบควบคุมในแนวทางไหน จากนั้นก็ออกแบบโมเด็มตามข้อกำหนดที่ได้ทำการศึกษาไว้แล้ว จากนั้นจึงทำการออกแบบส่วนควบคุมให้เป็นแนวทางเดียวกับระบบอื่นๆ ที่ได้ออกแบบไว้ก่อนหน้านี้แล้ว

5. ทำการทดลอง เนื่องจากในแต่ละระบบก็จะมีส่วนย่อยที่ต้องนำมาประกอบกันจึงจะทำงานเป็นระบบได้ จึงต้องทดลองทีละส่วน ดูว่ามีการทำงานเป็นไปตามที่ได้ออกแบบไว้หรือไม่ ถ้าไม่ถูกต้องควรจะต้องแก้ไขอย่างไร บางครั้งอาจจะต้องออกแบบระบบใหม่ เนื่องจากการทำงานไม่ถูกต้องหรือไม่สามารถเข้ากับส่วนที่เหลือได้ ส่วนของการทดลองจะต้องใช้เวลาในการทำงานมากที่สุด เนื่องจากบางครั้งการออกแบบทำไว้ไม่รัดกุม จึงต้องทำการออกแบบใหม่

6. เมื่อได้ทดลองและเห็นว่าส่วนย่อยต่าง ๆ ทำงานได้อย่างถูกต้อง ก็นำส่วนย่อยแต่ละส่วนมาสร้างเป็นระบบ การสร้างจะเริ่มจากส่วนอ่านบัตรแม่เหล็กก่อน แล้วจึงเป็นส่วนของโมเด็ม สุดท้ายจะเป็นการสร้างส่วนของระบบควบคุมทั้งหมด ทำการตรวจสอบข้อบกพร่องในการทำงานของแต่ละระบบดูว่ามีการทำงานเป็นปกติตามที่ต้องการหรือไม่

7. นำระบบทั้งหมดมารวมกันเพื่อตรวจสอบการทำงานอีกครั้งหนึ่ง

8. เขียนซอฟต์แวร์ควบคุมการทำงานของระบบ ให้เป็นไปตามที่ได้ออกแบบส่วนของฮาร์ดแวร์ไว้แล้ว

9. ทดสอบระบบอีกครั้งด้วยซอฟต์แวร์ที่เขียนขึ้น

10. ทำการปรับปรุงแก้ไขในส่วนที่บกพร่อง หรือทำการเพิ่มเติมส่วนของซอฟต์แวร์เพื่อให้ระบบมีความสมบูรณ์มากขึ้น

บทที่ 2
บัตรแม่เหล็ก

ในปัจจุบันมีการนำ บัตรแม่เหล็ก (MAGNETIC CARD) มาใช้งานกันอย่างกว้างขวาง เช่น บัตร ATM บัตรเครดิต บัตรประจำตัวพนักงาน บัตรสมาชิกศูนย์บริการบางแห่ง เป็นต้น นอกจากนี้ก็ยังจะมีการนำบัตรแม่เหล็กมาใช้ในโอกาสอื่นๆ อีกในอนาคต การแพร่หลายในการใช้งานของบัตรแม่เหล็กเนื่องจากข้อดีของบัตรแม่เหล็กมีหลายประการ เช่น

1. ความเชื่อถือได้ของข้อมูลที่ถูกบันทึกไว้และที่อ่านได้มีค่าสูง
2. เครื่องอ่านและเครื่องบันทึกบัตรแม่เหล็ก มีราคาถูกกว่าบัตรอื่น เมื่อเทียบกับความปลอดภัยที่เท่ากัน
3. บัตรแม่เหล็กมีราคาไม่แพง
4. มีความปลอดภัยสูงเนื่องจากข้อมูลที่บันทึกไว้ไม่สามารถมองเห็นได้ด้วยตาเปล่า จึงสามารถรักษาความลับได้ดี
5. สามารถพกพาและเก็บรักษาได้สะดวก

2.1 ชนิดและการใช้งานบัตรแม่เหล็ก

บัตรแม่เหล็กนั้นมีชนิดและการใช้งานมากมาย ดังแสดงในตารางที่ 2.1 ในยุคแรก ๆ บัตรแม่เหล็กจะมีเนื้อวัสดุเป็น พลาสติก PVC ชนิดแข็ง และ แบบที่มีเนื้อวัสดุเป็นกระดาษ แต่ในปัจจุบันได้พัฒนาเนื้อวัสดุของบัตรแม่เหล็กมาเป็น POLYETHYLENE TEREPHTHALATE หรือเรียกย่อๆ ว่า PET

ในตารางที่ 2.2 เป็นตัวอย่างการใช้งานบัตรแม่เหล็กชนิดต่างๆ ซึ่งบางตัวอย่างอาจจะยังไม่มีการใช้งานในเมืองไทย แต่อาจมีการนำมาใช้บ้างในอนาคต

ตารางที่ 2.1 ชนิดและการใช้งานบัตรแม่เหล็ก

ชนิด	เนื้อวัสดุ	ความหนากระดาษ	การใช้งาน
บัตรพลาสติก (PLASTIC CARD)	- PVC ชนิดแข็ง - PVCA ชนิดแข็ง	0.76 mm.	- บัตร ATM - บัตรเครดิต - บัตรประจำตัว (ID)
PET CARD	- POLYETHYLENE TEREPHTHALATE (PET)	0.2 - 0.4 mm.	- บัตรโทรศัพท์ - บัตรช็อปปิ้งต่างๆ (PREPAID CARD)
บัตรกระดาษ (COMPOSITE PAPER CARD)	- กระดาษอย่างดี - กระดาษเคลือบพลาสติก - กระดาษบันทึกข้อมูล	0.2 - 0.76 mm.	- ตั๋วรถไฟ - ตั๋วทางด่วน - ตั๋วเครื่องบิน - บัตรโรงแรม

ตารางที่ 2.2 ตัวอย่างการใช้งานบัตรแม่เหล็ก

การใช้งาน	ชนิดของบัตร	หมายเหตุ (อุปกรณ์, ระบบ)
การเงิน การธนาคาร	- บัตร ATM	- เครื่อง ATM, CD, AD, ระบบธนาคารอัตโนมัติ
	- บัตรเครดิต	- ระบบ CAT, POS
	- บัตรเติมน้ำมัน	- ระบบ POS ของปั๊มน้ำมัน
	- บัตรหลักทรัพย์	- การจ่ายเงินซื้อหลักทรัพย์, CASHING, CD
	- บัตรช็อปปิ้ง	- การซื้อสินค้าในห้างสรรพสินค้า, ร้านค้า

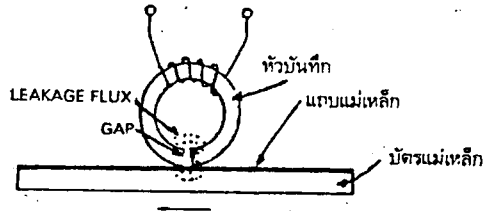
ตารางที่ 2.2 (ต่อ)

การใช้งาน	ชนิดของบัตร	หมายเหตุ (อุปกรณ์, ระบบ)
การคมนาคม	- ตั๋วรถไฟแม่เหล็ก - ตั๋วผ่านทางด่วน - ตั๋วเครื่องบิน - ตั๋วจอดรถ	- เครื่องขายตั๋วรถไฟอัตโนมัติ - ระบบเก็บเงินค่าผ่านทางด่วนอัตโนมัติ - ระบบขายตั๋วเครื่องบินอัตโนมัติ - ระบบควบคุมที่จอดรถ
การสื่อสาร	- บัตรโทรศัพท์	- โทรศัพท์สาธารณะแบบใช้บัตร
การสำนักงาน (OA, FA)	- บัตรประจำตัวพนักงาน - บัตรประจำตัวนักเรียน - บัตรเครื่องถ่ายเอกสาร - บัตรตรวจโรค - บัตรโรงแรม	- ระบบเช็คเวลาเข้า-ออกงาน - ระบบยืม-คืนหนังสือห้องสมุด - ระบบควบคุมเครื่องถ่ายเอกสาร - ระบบควบคุมโรค - ระบบเช็คอิน-เอาท์ของโรงแรม
การรักษาความปลอดภัย	- บัตรปิด-เปิดล็อค	- ระบบควบคุมการเข้า-ออก

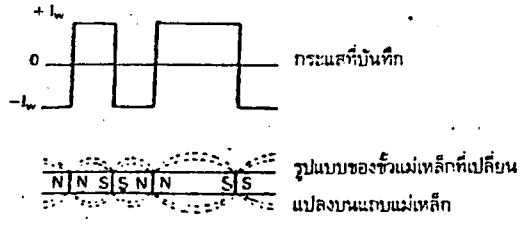
2.2 การบันทึกข้อมูลลงบนบัตรแม่เหล็ก

การบันทึกข้อมูลลงบนบัตรแม่เหล็กจะใช้วิธีการบันทึกแบบดิจิตอล ในลักษณะเดียวกับที่ใช้ในแผ่นฟล็อปปี้ดิสก์หรือเทปแม่เหล็กสำหรับเครื่องคอมพิวเตอร์ทั่วไป ในการบันทึกข้อมูลลงบนแถบแม่เหล็กนั้นจะต้องป้อนกระแสพัลซ์ซึ่งมีทั้งด้าน + และด้าน - พร้อมทั้งมีขนาดเพียงพอเข้าที่ขดลวดของหัวบันทึกซึ่งกดอยู่บนแถบแม่เหล็กที่เคลื่อนด้วยความเร็วคงที่ ดังในรูปที่ 2.1(ก) แถบแม่เหล็กจะถูกเปลี่ยนให้มีรูปแบบของขั้วแม่เหล็กตาม LEAKAGE FLUX จาก GAP ของหัวบันทึกดังในรูปที่ 2.1(ข) แถบแม่เหล็กจะเกิดเป็นแม่เหล็กถาวรขนาดจิ๋วเรียงตัวกันตามขั้ว + และ - ของพัลซ์และความกว้างของพัลซ์สัญญาณที่บันทึก เนื่องจากกระแสพัลซ์ที่ใช้ในการบันทึกมีขนาดเพียงพอ

ที่หัวบันทึก จะทำให้แถบแม่เหล็กมีสนามแม่เหล็กอิ่มตัวได้ ดังนั้นเมื่อทำการบันทึกข้อมูลตัวข้อมูลเดิมที่เคยมีอยู่จะถูกเขียนทับและหายไป เหลือเพียงข้อมูลใหม่เท่านั้น



(ก) การบันทึกลงบนบัตรแม่เหล็ก

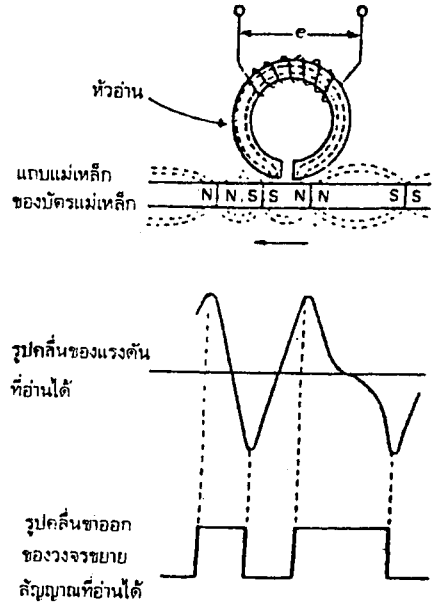


(ข) รูปแบบของขั้วแม่เหล็กที่บันทึก

รูปที่ 2.1 การบันทึกข้อมูล

2.3 การอ่านข้อมูลจากบัตรแม่เหล็ก

การอ่านข้อมูลจากบัตรแม่เหล็กทำได้โดยให้หัวอ่านสัมผัสกับแถบแม่เหล็กซึ่งเคลื่อนที่ด้วยความเร็วคงที่ดังในรูปที่ 2.2 FLUX ที่เกิดจากแม่เหล็กถาวรขนาดจิ๋วบนแถบแม่เหล็กจะผ่านจาก GAP ของหัวอ่านไปยังแกน(CORE) การเปลี่ยนแปลงของ FLUX ตามข้อมูลที่บันทึกจะทำให้เกิดแรงดันที่ขดลวดของหัวอ่าน จุดสูงสุด (PEAK) ของแรงดันที่อ่านได้นั้น จะตรงกับจุดที่สนามแม่เหล็กบนแถบแม่เหล็กกลับทิศพอดี ดังนั้นถ้าขยายแรงดันนี้ขึ้นและตรวจหาจุดสูงสุด (PEAK) ด้วยวิธี DIFFERENTIAL แล้วเปลี่ยนเป็นสัญญาณพัลซ์ ก็จะได้ข้อมูลที่บันทึกอยู่ในบัตรแม่เหล็กนั้น ดังรูปที่ 2.2



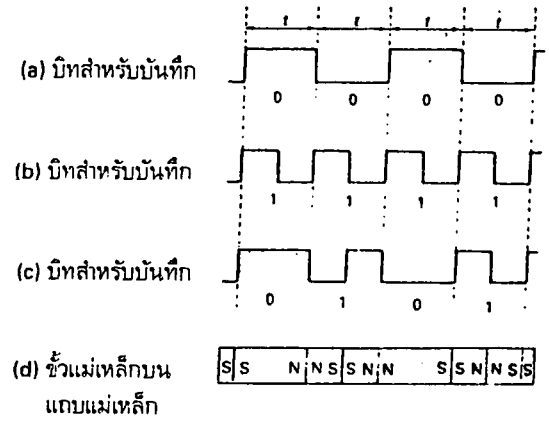
รูปที่ 2.2 การอ่านข้อมูล

2.4 รูปแบบการบันทึกข้อมูล

รูปแบบการบันทึกข้อมูลลงบนบัตรแม่เหล็ก ส่วนใหญ่จะใช้รูปแบบ F2F (เรียกทั่วไปว่าแบบ FM) F2F หมายถึง TWO FREQUENCY COHERENT PHASE ENCODING และ FM หมายถึง FREQUENCY MODULATION การบันทึกข้อมูลในลักษณะเช่นนี้จะบันทึกข้อมูล (DATA) และสัญญาณนาฬิกา (CLOCK) เข้าไว้ในแถบข้อมูล (TRACK) เดียวกัน นอกเหนือจากนี้ ยังมีการบันทึกที่ข้อมูลและสัญญาณนาฬิกาแยกกันคนละแถบข้อมูล เช่นแบบ NRZI (NON RETURN TO ZERO INVERTED RECORDING) ซึ่งมีความจุในการบันทึกต่ำ

โครงสร้างของบิทข้อมูลสำหรับแต่ละตัวเลขหรือตัวอักษรบนแถบแม่เหล็ก จะเป็นโครงสร้างแบบบิทที่มีค่านัยสำคัญต่ำสุด (LSB) จะถูกเข้ารหัสเป็นบิทแรกและบิทพาริตี้เป็นบิทสุดท้ายที่ถูกเข้ารหัสแล้วบันทึกลงบนบัตรแม่เหล็ก

การบันทึกแบบ F2F (FM)



รูปที่ 2.3 การบันทึกแบบ F2F (FM)

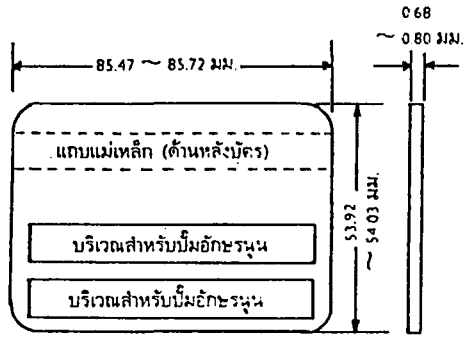
การบันทึกแบบ F2F (FM) จะมีรูปแบบดังแสดงในรูปที่ 2.3 หากสนามกลับทิศในช่วงเวลาคงที่ (t) จะเรียกระยะระหว่างการกลับทิศสนามแม่เหล็กที่อยู่ติดกัน 2 ด้านว่า BIT CELL ตำแหน่งตรงกลางของ BIT CELL ที่สนามแม่เหล็กกลับทิศทาง ดังในรูปที่ 2.3(ข) จะกำหนดให้มีลอจิกเป็น "0" ลักษณะการเปลี่ยนสนามแม่เหล็กที่บันทึกบนแถบแม่เหล็ก จะเป็นดังในรูปที่ 2.3(ง)

2.5 ขนาดมาตรฐานของบัตรแม่เหล็ก

บัตรแม่เหล็กแต่ละชนิดจะมีมาตรฐานประจำตัวเอง อย่างเช่น ขนาดความกว้าง-ยาว-หนาของบัตร ความหนาแน่นในการบันทึก มุมในการบันทึก และรหัสชุดตัวอักษรของแต่ละแถบข้อมูล เป็นต้น สำหรับมาตรฐานของบัตรเครดิตที่ใช้กันทั่วโลก คือ มาตรฐาน ISO 7810-7813 (ISO : INTERNATIONAL ORGANIZATION FOR STANDARDIZATION) ซึ่งเข้ากันได้กับมาตรฐาน JIS ชนิดที่ I ของญี่ปุ่น ส่วน JIS ชนิดที่ II เป็นมาตรฐานของบัตรแม่เหล็กที่ใช้กับธนาคารทั่วญี่ปุ่น จึงไม่เป็นมาตรฐานสากล

บัตรแม่เหล็กโดยทั่วไปจะมีขนาดดังนี้

ความกว้าง	ระหว่าง	53.92	ถึง	54.03	มิลลิเมตร
ความยาว	ระหว่าง	85.47	ถึง	85.72	มิลลิเมตร
ความหนา	ระหว่าง	0.68	ถึง	0.80	มิลลิเมตร

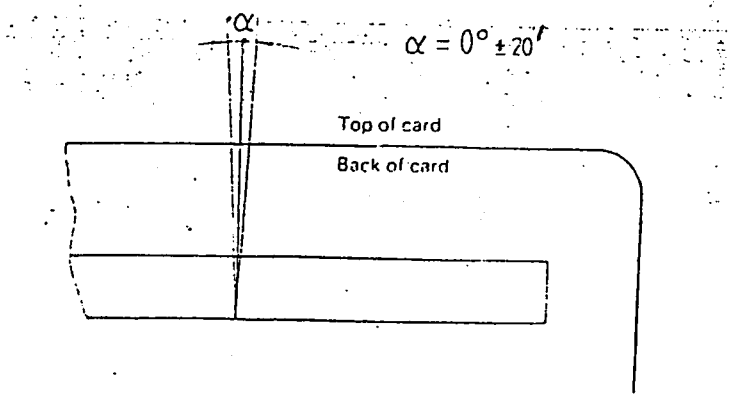


รูปที่ 2.4 ขนาดมาตรฐานของบัตรแม่เหล็ก

2.6 ความหนาแน่นและมุมในการบันทึก

ความหนาแน่นในการบันทึกจะกำหนดเป็น จำนวนบิทต่อหน่วยของความยาว BIT CELL เช่น 210 bpi (bit per inch) จะมีความยาว BIT CELL เป็น 0.121 mm.

มุมในการบันทึกจะต้องตั้งฉากกับขอบด้านบนของตัวบัตรที่อยู่ใกล้แถบแม่เหล็ก มุมในการบันทึกนั้นจะวัดจากมุมของ head gap ของหัวอ่านขณะเมื่อขนาด(amplitude) ของสัญญาณที่อ่านได้มีค่ามากที่สุด



รูปที่ 2.5 มุมของการบันทึก

ความหนาแน่นและมุมในการบันทึกของแต่ละแถบข้อมูลมีดังนี้

แถบข้อมูลที่ 1 สำหรับใช้อ่านข้อมูลอย่างเดียว

มีค่าความหนาแน่นในการบันทึก 8.3 บิทต่อมิลลิเมตร ± 5%

มุมในการบันทึก 0° ± 20′ (1° = 60′)

แถบข้อมูลที่ 2 สำหรับใช้อ่านข้อมูลอย่างเดียว

มีค่าความหนาแน่นในการบันทึก 3 บิทต่อมิลลิเมตร ± 3%

มุมในการบันทึก 0° ± 20′

แถบข้อมูลที่ 3 สำหรับใช้อ่านและบันทึกข้อมูลได้

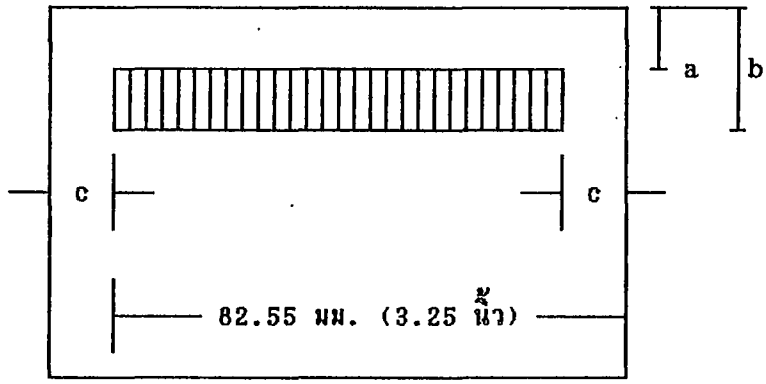
มีค่าความหนาแน่นในการบันทึก 8.3 บิทต่อมิลลิเมตร ± 8%

มุมในการบันทึก 0° ± 20′

บัตรแม่เหล็กถ้าแบ่งตามจำนวนแถบข้อมูลที่บันทึกข้อมูลอยู่บนแถบแม่เหล็ก จะมี 2 ชนิด คือ ชนิดที่มีแถบข้อมูล 2 แถบ และ ชนิดที่มีแถบข้อมูล 3 แถบ โดยมีรายละเอียด ดังนี้

2.7 บัตรแม่เหล็กชนิดที่มีแถบข้อมูล 2 แถบ

2.7.1 ตำแหน่งของแถบแม่เหล็กบนบัตรแม่เหล็ก



รูปที่ 2.6 ตำแหน่งของแถบแม่เหล็กบนบัตรแม่เหล็ก ชนิดที่มีแถบข้อมูล 2 แถบ

ระยะ a มีขนาด 5.54 มิลลิเมตร (0.218 นิ้ว) (ค่าสูงสุด)

ระยะ b มีขนาด 11.89 มิลลิเมตร (0.468 นิ้ว) (ค่าสูงสุด)

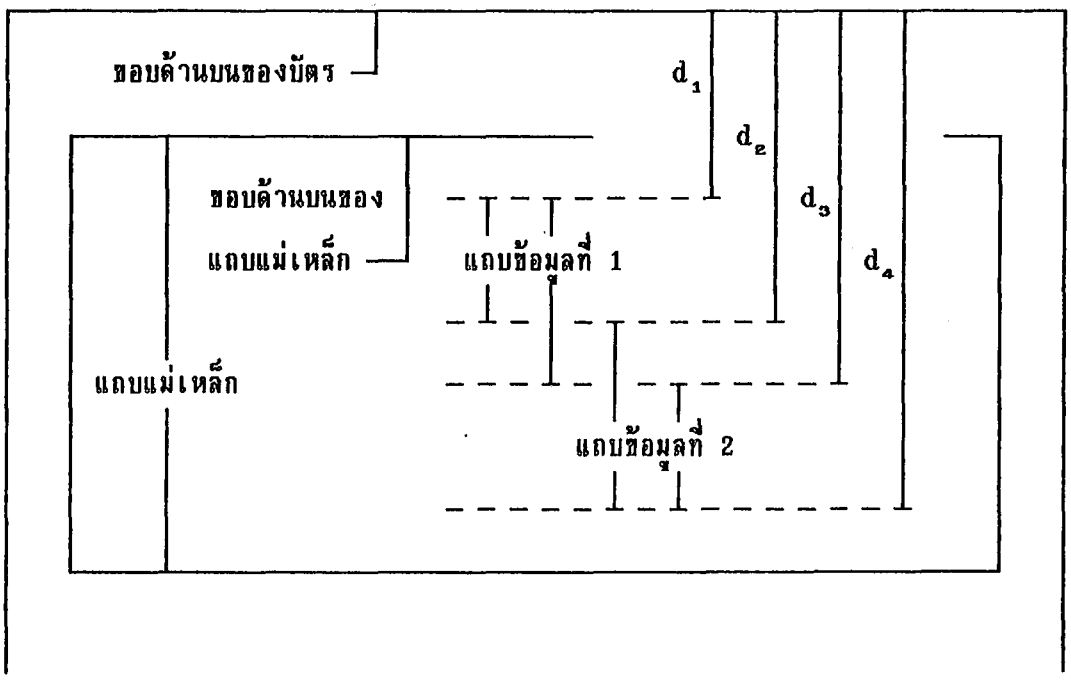
ระยะ c มีขนาด 2.92 มิลลิเมตร (0.115 นิ้ว) (ค่าสูงสุด)

โดย ระยะ a คือระยะห่างระหว่างขอบด้านบนของแถบแม่เหล็กกับขอบด้านบนของบัตรแม่เหล็ก

ระยะ b คือระยะห่างระหว่างขอบด้านล่างของแถบแม่เหล็กกับขอบด้านบนของบัตรแม่เหล็ก

ระยะ c คือระยะห่างระหว่างขอบด้านข้างของแถบแม่เหล็กกับขอบด้านข้างของบัตรแม่เหล็ก

2.7.2 ตำแหน่งของแถบข้อมูลบนแถบแม่เหล็ก



รูปที่ 2.7 ตำแหน่งของแถบข้อมูลบนแถบแม่เหล็กของบัตรแม่เหล็ก ชนิดที่มีแถบข้อมูล 2 แถบ

แถบข้อมูลแถบที่ 1 อยู่ระหว่างระยะ d_1 กับ ระยะ d_2 (แคบสุด) หรือ ระยะ d_3 (กว้างสุด)

แถบข้อมูลแถบที่ 2 อยู่ระหว่างระยะ d_2 (กว้างสุด) หรือ ระยะ d_3 (แคบสุด) กับ ระยะ d_4

โดย ระยะ d_1 มีขนาด 5.66 มิลลิเมตร (0.223 นิ้ว)

ระยะ d_2 มีขนาด 8.46 มิลลิเมตร (0.333 นิ้ว)

ระยะ d_3 มีขนาด 8.97 มิลลิเมตร (0.353 นิ้ว)

ระยะ d_4 มีขนาด 11.76 มิลลิเมตร (0.463 นิ้ว)

เพราะฉะนั้น

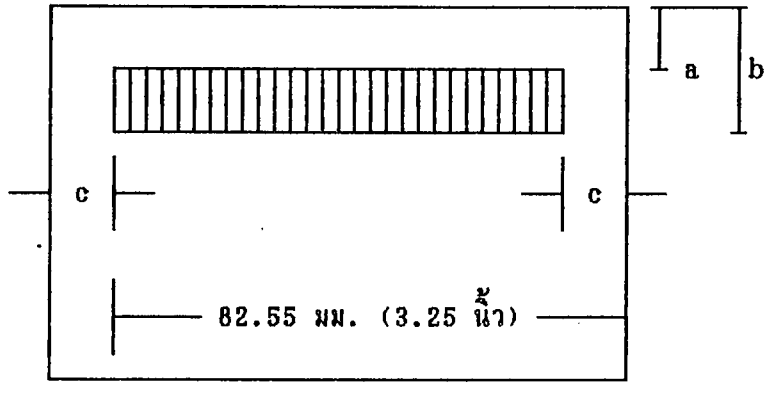
ความกว้างของแถบข้อมูลแถบที่ 1 คือ 2.80 ถึง 3.31 มิลลิเมตร (0.110 ถึง 0.130 นิ้ว)

ความกว้างของแถบข้อมูลแถบที่ 2 คือ 2.79 ถึง 3.30 มิลลิเมตร (0.110 ถึง 0.130 นิ้ว)

2.8 <u>บัตรแม่เหล็กชนิดที่มีแถบข้อมูล 3 แถบ</u>

สำหรับบัตรแม่เหล็กชนิดที่มี 3 แถบข้อมูล มีรายละเอียด ดังนี้

2.8.1 ตำแหน่งของแถบแม่เหล็กบนบัตรแม่เหล็ก



รูปที่ 2.8 ตำแหน่งของแถบแม่เหล็กบนบัตรแม่เหล็ก ชนิดที่มีแถบข้อมูล 3 แถบ

ระยะ a มีขนาด 5.54 มิลลิเมตร (0.218 นิ้ว) (ค่าสูงสุด)

ระยะ b มีขนาด 15.82 มิลลิเมตร (0.623 นิ้ว) (ค่าสูงสุด)

ระยะ c มีขนาด 2.92 มิลลิเมตร (0.115 นิ้ว) (ค่าสูงสุด)

โดย ระยะ a คือระยะห่างระหว่างขอบด้านบนของแถบแม่เหล็กกับขอบด้านบนของบัตรแม่เหล็ก

ระยะ b คือระยะห่างระหว่างขอบด้านล่างของแถบแม่เหล็กกับขอบด้านบนของบัตรแม่เหล็ก

ระยะ c คือระยะห่างระหว่างขอบด้านข้างของแถบแม่เหล็กกับขอบด้านข้างของบัตรแม่เหล็ก

2.8.2 ตำแหน่งของแถบข้อมูลบนแถบแม่เหล็ก

แถบข้อมูลแถบที่ 1 และแถบข้อมูลแถบที่ 2 จะเหมือนกับบนบัตรแม่เหล็กชนิดที่มี 2 แถบข้อมูล

แถบข้อมูลแถบที่ 3 อยู่ระหว่าง ระยะ d_5 (กว้างสุด) หรือ ระยะ d_6 (แคบสุด)

กับ ระยะ d_7 (แคบสุด) หรือ ระยะ d_8 (กว้างสุด)

ดังแสดงในรูปที่ 2.9

โดย ระยะ d_5 มีขนาด 12.01 มิลลิเมตร (0.473 นิ้ว)

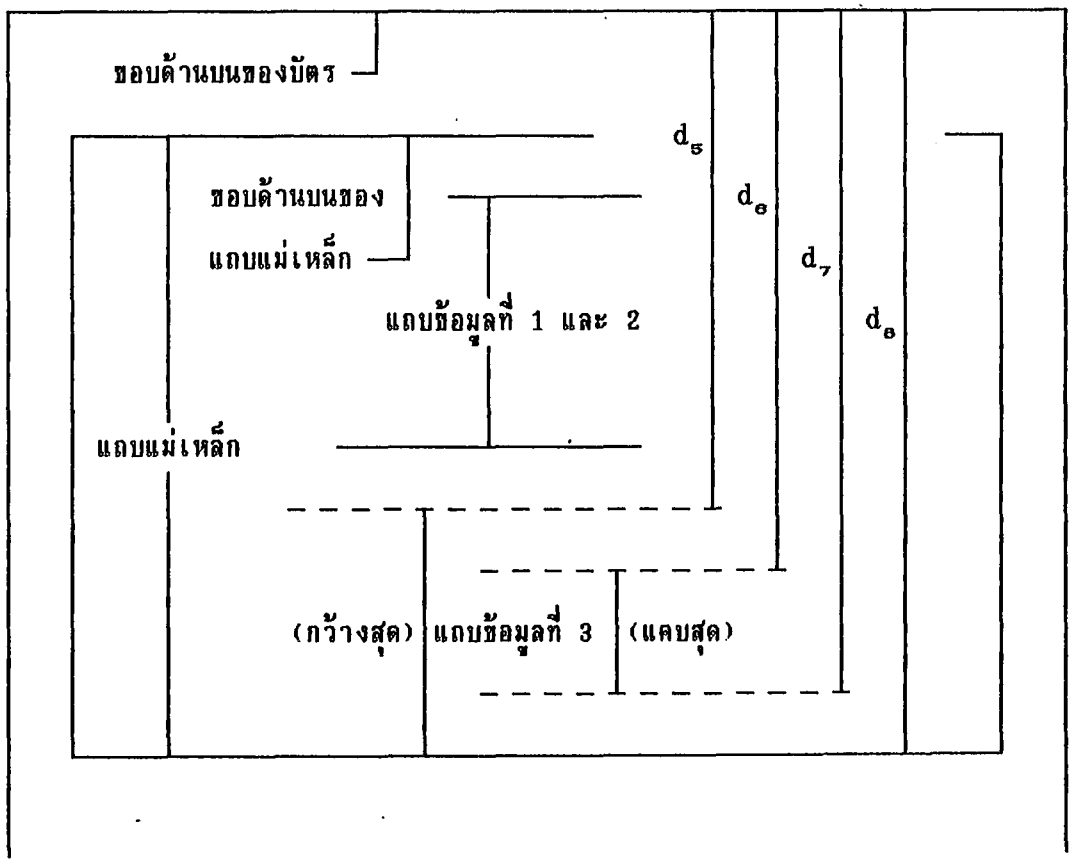
ระยะ d_6 มีขนาด 12.52 มิลลิเมตร (0.493 นิ้ว)

ระยะ d_7 มีขนาด 15.32 มิลลิเมตร (0.603 นิ้ว)

ระยะ d_8 มีขนาด 15.82 มิลลิเมตร (0.623 นิ้ว)

เพราะฉะนั้น

ความกว้างของแถบข้อมูลแถบที่ 3 คือ 2.80 ถึง 3.81 มิลลิเมตร (0.110 ถึง 0.130 นิ้ว)



รูปที่ 2.9 ตำแหน่งของแถบข้อมูลบนแถบแม่เหล็กของบัตรแม่เหล็ก ชนิดที่มีแถบข้อมูล 3 แถบ

2.9 ข้อมูลในแถบข้อมูลที่ 1

ข้อมูลในแถบข้อมูลที่ 1 นั้น จะเป็นตัวอักษรและตัวเลขซึ่งใช้บิทข้อมูลจำนวน 6 บิทต่อหนึ่งตัวอักษรหรือตัวเลข และใช้การตรวจสอบพาริตี้ (parity) แบบพาริตี้คี่ ตารางที่ 2.3 แสดงรหัสของตัวอักษร ตัวเลข และสัญลักษณ์ ที่ใช้ในแถบข้อมูลที่ 1 นี้

				b_6	0	0	1	1
				b_5	0	1	0	1
b_4	b_3	b_2	b_1	Row \ Column	0	1	2	3
0	0	0	0	0	SP	0	ⓐ	P
0	0	0	1	1	ⓐ	1	A	Q
0	0	1	0	2	ⓐ	2	B	R
0	0	1	1	3	ⓒ	3	C	S
0	1	0	0	4	$	4	D	T
0	1	0	1	5	% ⓓ	5	E	U
0	1	1	0	6	ⓐ	6	F	V
0	1	1	1	7	ⓐ	7	G	W
1	0	0	0	8	(	8	H	X
1	0	0	1	9	)	9	I	Y
1	0	1	0	10	ⓐ	ⓐ	J	Z
1	0	1	1	11	ⓐ	ⓐ	K	ⓑ
1	1	0	0	12	ⓐ	ⓐ	L	ⓑ
1	1.	0	1	13	–	ⓐ	M	ⓑ
1	1	1	0	14	,	ⓐ	N	ⓓ
1	1	1	1	15	/	? ⓓ	O	ⓐ

ตารางที่ 2.3 รหัสข้อมูลของแถบข้อมูลที่ 1

จากตารางที่ 2.3

a เป็นตำแหน่งของรหัสที่ใช้เฉพาะในการควบคุมทางฮาร์ดแวร์เท่านั้น

b เป็นตำแหน่งที่สงวนไว้สำหรับการเพิ่มอักขระเฉพาะของชาติใดๆ และไม่สามารถใช้ในระดับสากลได้

c เป็นตำแหน่งที่สงวนไว้สำหรับการเพิ่มสัญลักษณ์ทางกราฟิค

d เป็นสัญลักษณ์ที่มีความหมายเฉพาะสำหรับระบบบัตรแม่เหล็ก ดังนี้

ตำแหน่ง column 0 row 5 เป็นสัญลักษณ์การเริ่มต้นของข้อมูล (start sentinel)

ตำแหน่ง column 1 row 15 เป็นสัญลักษณ์การสิ้นสุดของข้อมูล (end sentinel)

ตำแหน่ง column 3 row 14 เป็นสัญลักษณ์ของตัวแยกข้อมูล (separator)

2.10 ข้อมูลในแถบข้อมูลที่ 2 และ 3

ข้อมูลในแถบข้อมูลที่ 2 และ 3 นั้น จะเป็นตัวเลขอย่างเดียว โดยใช้รหัส BCD ขนาด 4 บิท และใช้การตรวจสอบพาริตี้แบบพาริตี้คี่ ตารางที่ 2.4 แสดงรหัสของตัวเลขและสัญลักษณ์ที่ใช้ในแถบข้อมูลที่ 2 และ 3

Bits					Row	Character
P	b_4	b_3	b_2	b_1		
1	0	0	0	0	0	0
0	0	0	0	1	1	1
0	0	0	1	0	2	2
1	0	0	1	1	3	3
0	0	1	0	0	4	4
1	0	1	0	1	5	5
1	0	1	1	0	6	6
0	0	1	1	1	7	7
0	1	0	0	0	8	8
1	1	0	0	1	9	9
1	1	0	1	0	10	(a)
0	1	0	1	1	11	(b^1)
1	1	1	0	0	12	(c)
0	1	1	0	1	13	(b^2)
0	1	1	1	0	14	(d)
1	1	1	1	1	15	(b^3)

ตารางที่ 2.4 รหัสข้อมูลของแถบข้อมูลที่ 2 และ 3

จากตารางที่ 2.4

a เป็นตำแหน่งของรหัสที่ใช้เฉพาะในการควบคุมทางฮาร์ดแวร์เท่านั้น

b^1 เป็นสัญลักษณ์การเริ่มต้นของข้อมูล (start character)

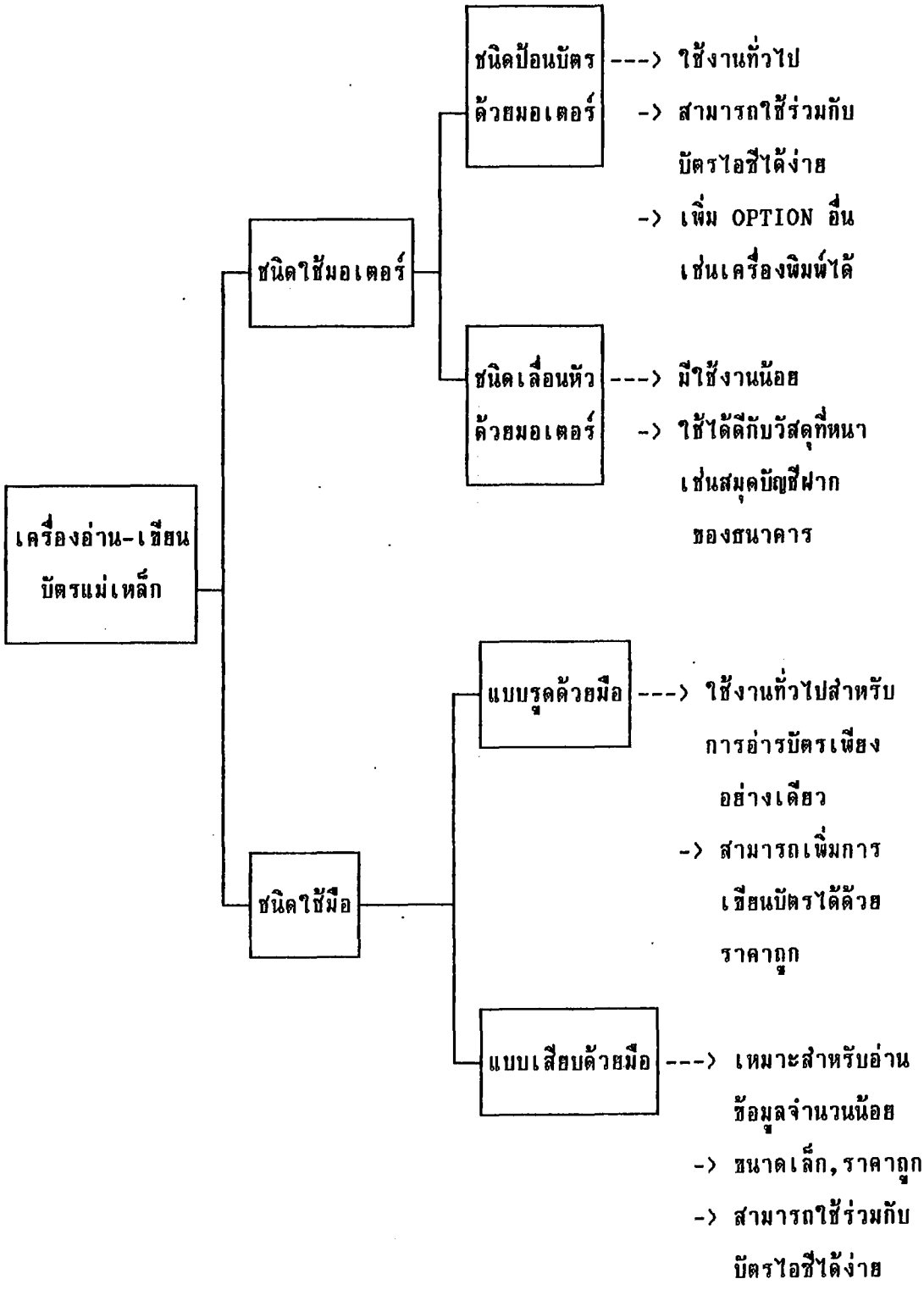
b^2 เป็นสัญลักษณ์ของตัวแยกข้อมูล (separator)

b^3 เป็นสัญลักษณ์การสิ้นสุดของข้อมูล (stop character)

2.11 ชนิดของเครื่องอ่าน-เขียนบัตรแม่เหล็ก

แผนผังในรูปที่ 2.10 แสดงชนิดของเครื่องอ่าน-เขียนบัตรแม่เหล็กโดยแบ่งตามวิธีการป้อนบัตรเข้าสู่ตัวเครื่อง นอกจากนี้การอินเทอร์เฟสเข้ากับอุปกรณ์อื่น (TTL, RS-232-C หรือ

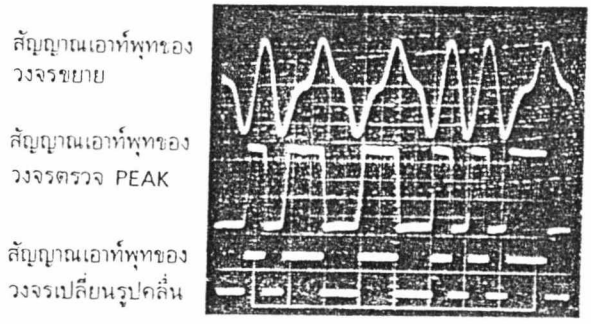
อื่นๆ) และลักษณะการใช้งาน (ประกอบเข้ากับเครื่องอื่น หรือวางตั้งโต๊ะตัวเดียว) ก็ทำให้โครงสร้างและวงจรของเครื่องอ่าน-เขียนบัตรแม่เหล็กแตกต่างกันไปบ้าง



รูปที่ 2.10 ชนิดของเครื่องอ่าน-เขียนบัตรแม่เหล็ก

2.12 วงจรอ่านบัตร

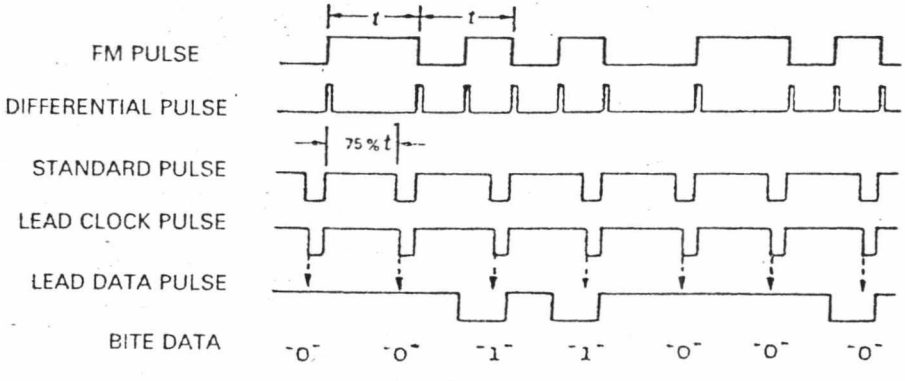
หัวอ่านข้อมูลที่ใช้สามารถใช้หัวเทปคาสเซททั่วไป อาจเป็นแบบ โมโน หรือ สเตอริโอ ก็ได้ โดยสัญญาณจากหัวอ่านจะถูกขยายที่วงจรขยายหัวอ่าน และตรวจจับ PEAK ที่วงจรตรวจจับ PAEK PEAK ที่ตรวจพบจะนำมาแปลงเป็นรูปพัลซ์ด้วยวงจรเปลี่ยนรูปคลื่นเพื่อให้ได้สัญลักษณ์พัลซ์ เหมือนกับที่บันทึกเอาไว้ ซึ่งมีลักษณะดังตัวอย่างที่แสดงในรูปที่ 2.11



รูปที่ 2.11 รูปคลื่นของการอ่านบัตรแม่เหล็ก

การถอดเฉพาะสัญญาณข้อมูลออกมาจากสัญญาณพัลซ์ที่ได้นี้ จะต้องตรวจดูว่ามีการกลับขั้วของพัลซ์ FM ในช่วง STANDARD TIME ซึ่งมีค่าประมาณ 75% ของความยาว 1 บิท (75%t) หรือไม่ ถ้าไม่มีการกลับขั้ว ค่าของข้อมูลจะเป็น "0" ถ้ามีการกลับขั้ว ค่าของข้อมูลจะเป็น "1"

การแยกสัญญาณข้อมูลออกจากสัญญาณนาฬิกา (CLOCK) จะใช้วงจร DEMODULATION โดยมี TIME CHART ของ FM DEMODULATION ดังในรูปที่ 2.12



รูปที่ 2.12 TIME CHART ของ FM DEMODULATION

การสร้าง STANDARD TIME เพื่อแยกข้อมูลออกมานั้นมี 2 วิธี คือ แบบความยาวคงที่ และแบบความยาวเปลี่ยนแปลงได้ แบบความยาวคงที่นั้นจะกำหนด STANDARD TIME ให้มีค่าคงที่ แล้วเปรียบเทียบกับทุกบิท ส่วนแบบที่ความยาวเปลี่ยนแปลงได้ จะวัดความยาวของบิทก่อนหน้า และสร้าง STANDARD TIME จากจุดนั้น แล้วนำไปเปรียบเทียบกับบิทถัดไป

สำหรับการสร้าง STANDARD TIME แบบเปลี่ยนแปลงความยาวได้นั้น ขณะที่กำลังอ่านบัตรอยู่ ถ้าความเร็วในการป้อนบัตรเปลี่ยนแปลง STANDARD TIME จะเปลี่ยนแปลงตามไปด้วย จึงใช้วิธีนี้กับเครื่องอ่านบัตรที่ป้อนด้วยมือ เนื่องจากความเร็วในการรูดด้วยมือมีค่าไม่แน่นอน

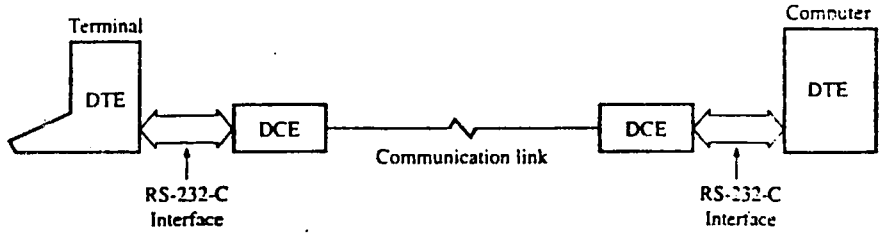
การแยกข้อมูลโดยตรวจหา PEAK ตามแบบ FM นั้น ยอมให้ระดับเอาท์พุทที่อ่านได้เปลี่ยนแปลงได้เล็กน้อย แต่สำหรับความยาวของบิทนั้นถ้าเปลี่ยนแปลงมาก (เรียกว่า JITTER) จะเป็นสาเหตุให้การอ่านข้อมูลผิดพลาดได้ JITTER ทั้งการอ่านและบันทึกข้อมูลนั้น เกิดสาเหตุหลายประการ ในการออกแบบใช้งานจะต้องให้ความสำคัญกับปัญหานี้อย่างมาก

บทที่ 3

มาตรฐาน RS-232-C

ในระบบการสื่อสารข้อมูลนั้น จะมีคำอยู่ 2 คำที่มักจะพบอยู่เสมอ คือ DTE และ DCE คำว่า DTE นั้นย่อมาจาก Data Terminal Equipment เป็นอุปกรณ์ทางดิจิตอลสำหรับส่งและ/หรือรับข้อมูล และใช้อุปกรณ์การสื่อสารสำหรับการส่งถ่ายข้อมูล หมายถึงตัวเครื่องที่เป็นต้นกำเนิดข้อมูลและตัวปลายทางที่รับข้อมูลนั่นเอง อุปกรณ์ DTE ที่พบอยู่เสมอมักจะเป็น เครื่องคอมพิวเตอร์ที่มีจอแสดงผล CRT เครื่องพิมพ์ เครื่องโทรพิมพ์ เป็นต้น อุปกรณ์เหล่านี้เป็นตัวอย่างของ DTE ส่วนคำว่า DCE นั้นย่อมาจาก Data Communication Equipment อุปกรณ์ DCE จะถูกต่อเข้ากับสายส่งทางการสื่อสาร ซึ่งก็มักจะเป็นสายโทรศัพท์ เพื่อส่งถ่าย (transfer)ข้อมูลจากจุดหนึ่งไปยังอีกจุดหนึ่งผ่านสายส่งนั้น อุปกรณ์ DTE นี้ ไม่ได้ประมวลผลหรือปฏิบัติการใดๆ กับข้อมูล แต่จะส่งถ่ายข้อมูลไป โดยไม่ได้คำนึงถึงสาระหรือเนื้อความของข้อมูล ข้อมูลจะถูกส่งเป็นแบบอนาล็อก (analog) หรือดิจิตอล (digital) ก็ขึ้นอยู่กับ การเชื่อมโยงกันทางการสื่อสาร (communication link) อุปกรณ์ DCE ที่พบกับบ่อยที่สุด คือ โมเด็ม (MODEM)

ผลิตภัณฑ์ DTE และ DCE มักจะถูกออกแบบมาโดยการผลิตที่แตกต่างกันไป ดังนั้นจึงจำเป็นที่จะต้องมีมาตรฐานการเชื่อมต่อ เพื่อที่อุปกรณ์เหล่านี้จะสามารถต่อเข้าด้วยกันได้โดยง่าย มาตรฐานของการเชื่อมต่อที่ใช้กันมากที่สุด คือ มาตรฐาน RS-232-C "RS" นั้นมาจาก Recommended Standard สำหรับ "232" เป็นหมายเลขของคณะกรรมการ (Committee number) ส่วน "C" แสดงถึง เวอร์ชัน (version) ในรูปที่ 3.1 แสดงการเชื่อมต่อกันระหว่าง DTE และ DCE มาตรฐานนี้ถูกประกาศโดยสมาคม Electronics Industries Association (EIA) EIA เป็นกลุ่มอุตสาหกรรม ซึ่งให้ความช่วยเหลือและเผยแพร่ ซึ่งเกี่ยวข้องกับด้านต่างๆ หลายด้านของอุตสาหกรรมอิเล็กทรอนิกส์ อย่างเช่น ส่วนประกอบต่างๆ อุปกรณ์ ระบบ การบรรจุหีบห่อ และการสื่อสาร มาตรฐานเดียวกันนี้แต่ใหม่กว่า RS-232-C รู้จักกันในนาม EIA-232-D แต่ถึงกระนั้นก็ตาม เราจะพบคำว่า "RS-232-C" บ่อยกว่า เพราะว่า ผู้ผลิตทางด้านการสื่อสารข้อมูลยังคงใช้มาตรฐาน RS-232-C กันอยู่ ดังนั้นคำว่า "RS-232" จะหมายรวมถึงทั้งสองมาตรฐาน โดยส่วนที่ต่างกันจะระบุไว้ให้ทราบ



รูปที่ 3.1 การเชื่อมต่อกันระหว่าง DTE and DCE

มีหลากหลายวิถีทางที่จะ ออกแบบมาตรฐานขึ้นมาใช้สำหรับการส่งถ่ายข้อมูลแบบอนุกรม RS-232-C นั้นเป็นมาตรฐานหนึ่งที่ถูกใช้อยู่โดยผู้ผลิตหลายราย แต่ EIA ไม่สามารถที่จะบังคับทางกฎหมายต่อผู้ผลิตให้ใช้มาตรฐานการเชื่อมต่อนี้ได้ เพราะว่า การใช้มาตรฐานนี้นั้น เป็นแบบตามแต่สมัครใจ เนื่องจากการเชื่อมต่อ (interface) ตามแบบ RS-232-C ไม่ครอบคลุมถึงทุกๆ ผลิตภัณฑ์หรือการใช้งานที่เป็นไปได้ ผู้ผลิตบางรายจึงปรับเปลี่ยนมัน ตามเงื่อนไขหรือสภาพเฉพาะของผู้ผลิตแต่ละราย อย่างไรก็ตาม มาตรฐานนี้ก็ได้เตรียมโครงร่างสำหรับให้สถานีปลายทาง (terminal) และอุปกรณ์การสื่อสารข้อมูลสามารถถูกต่อเข้าด้วยกันได้โดยง่าย ชื่ออย่างเป็นทางการของ การเชื่อมต่อแบบ RS-232-C คือ Interface Between Data Terminal Equipment and Data Communication Equipment Employing Serial Binary Data Interchange. ถึงแม้ว่า RS-232-C นั้นจะถูกใช้อย่างกว้างขวาง แต่มันก็มีข้อจำกัดของมันอยู่บ้าง

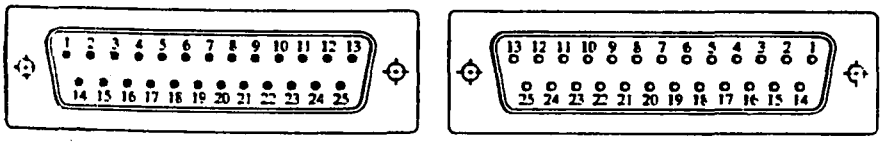
มาตรฐาน RS-232-C บรรยายถึงลักษณะเฉพาะ (characteristic) ทางด้านกลไก (mechanical) ด้านไฟฟ้า (electrical) และด้านหน้าที่การทำงาน (functional) การเชื่อมต่อแบบ RS-232-C สามารถถูกใช้ได้ทั้งสำหรับ การปฏิบัติการแบบ full-duplex และ half-duplex ทั้งสำหรับระบบจุดต่อจุด (point-to-point system) หรือ หลายจุด (multipoint) รวมทั้ง การสื่อสารแบบซิงโครนัส (synchronous communication) หรือแบบอะซิงโครนัส (asynchronous) ก็ได้ ดังนั้นมาตรฐาน RS-232-C จึงมีความยืดหยุ่นมาก และถูกใช้งานอย่างหลากหลายที่เดียว

3.1 ลักษณะทางกลไก (Mechanics Characteristic)

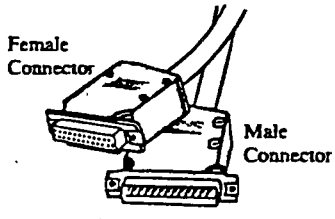
ดังที่ได้กล่าวมาแล้วตอนต้น มาตรฐาน RS-232-C นั้นเป็นมาตรฐานอ้างอิงสำหรับผู้ออกแบบอุปกรณ์ ซึ่งระบุถึงลักษณะเฉพาะต่างๆ ส่วนใหญ่แต่ไม่ทั้งหมด ลักษณะเฉพาะอย่างหนึ่ง

ที่ไม่ได้ระบุไว้ในเวอร์ชัน "C" คือ รูปทรงสัณฐานของหัวต่อ (connector) อย่างไรก็ตาม ตามปกติแล้วส่วนมากจะใช้หัวต่อแบบ DB-25 หัวต่อชนิดนี้ได้ถูกบรรยายไว้ในเวอร์ชัน "D" รูปที่ 3.2 แสดงรูปของหัวต่อชนิดนี้ หัวต่อแบบชนิดตัวผู้ คือ plug type (DB-25P) จะต่อเข้ากับ DTE หัวต่อแบบชนิดตัวเมีย คือ socket type (DB-25S) จะต่อเข้ากับ DCE ในรูปที่ 3.2 แสดงภาพด้านหน้าและการจัดเรียงหมายเลขของขา (pin) ของหัวต่อทั้งสองชนิด และแสดงรูปทรงสัณฐานของหัวต่อชนิดนี้ในรูปที่ 3.2(ค) สำหรับตารางที่ 3.1 แสดงรายชื่อของการต่อของแต่ละขา สำหรับในมาตรฐานนี้ทุกๆ สิ่งจะอ้างอิงกับ DTE ดังนั้นชื่อสำหรับขาหมายเลข 2 คือ Transmit Data มันถูกใช้สำหรับส่งข้อมูล (transmit data) จาก DTE ไปยัง DCE เช่นกัน สำหรับขาหมายเลข 3 คือ Receive Data มันถูกใช้โดย DTE สำหรับรับข้อมูลจาก DCE

ตามมาตรฐานนั้น แนะนำความยาวของสายเคเบิลที่ยาวที่สุด คือ 50 ฟุต (หรือ 15 เมตร) ที่ระหว่าง DTE และ DCE อย่างไรก็ตาม หลายๆ บริษัทใช้สายเคเบิลยาวกว่านั้น โดยให้ผลเป็นที่น่าพอใจ แต่ต้องปฏิบัติการที่อัตราเร็วของข้อมูลต่ำกว่าอัตราเร็วสูงสุด (20,000 บิทต่อวินาที) และ/หรือ ใช้สายเคเบิลที่มีค่าความจุไฟฟ้า (capacitance) ต่ำกว่า



(ก) Front view DB-25P connector (ข) Front view DB-25S connector



(ค) 3 dimensional view of 25 pin connectors

รูปที่ 3.2 (ก) รูปด้านหน้าของหัวต่อชนิดตัวผู้ (DB-25P)

(ข) รูปด้านหน้าของหัวต่อชนิดตัวเมีย (DB-25S)

(ค) รูปร่างของหัวต่อชนิด 25 ขา

ตารางที่ 3.1 แสดงการกำหนดขาของการเชื่อมต่อแบบ EIA's 232 และหมายเลขของวงจรเทียบเท่า V.24 ของ CCITT

Pin Number	EIA circuit	CCITT V.24 Circuit	Signal Description	To DCE	From DCE
1	AA	101	Protective ground (chassis)[1]		
2	BA	103	Transmitted data	X	
3	BB	104	Received data		X
4	CA	105	Request-to-send	X	
5	CB	106	Clear-to-send		X
6	CC	107	DCE ready (also known as Data Set Ready)		X
7	AB	102	Signal ground (common return)	X	X
8	CF	109	Received line signal detector		X
9	—		(Reserved for data set testing)		
10	—		(Reserved for data set testing)		
11			Unassigned		
12	SCF	122	Secondary received line signal detector		X
13	SCB	121	Secondary Clear-to-Send		X
14	SBA	118	Secondary transmitted data	X	
15	DB	114	Transmitter signal element timing (DCE source)		X

[1]*Not defined as part of the D version.* [2]*Unassigned in the C version.*

ตารางที่ 3.1 (ต่อ)

Pin Number	EIA circuit	CCITT V.24 Circuit	Signal Description	To DCE	From DCE
16	SBB	119	Secondary received data		X
17	DD	115	Receiver signal element timing (DCE source)		X
18	LL	141	Local loopback[2]	X	
19	SCA	120	Secondary request-to-send	X	
20	CD	108/2	Data terminal ready	X	
21	RL/CG	110	Remote loopback signal quality detector		X
22	CE	125	Ring calling indicator		X
23	CH	111	Data signal rate selector (DTE source)	X	
23	CI	112	Data signal rate selector (DCE source)		X
24	DA	113	Transmitter signal element timing (DTE)	X	
25	TM	142	Test mode[2]		X

[1]*Not defined as part of the D version.* [2]*Unassigned in the C version.*

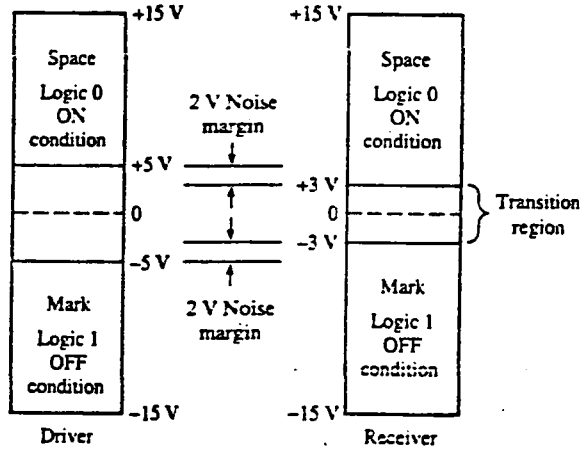
3.2 ลักษณะทางไฟฟ้า (Electrical characteristic)

ส่วนที่เกี่ยวกับทางด้านไฟฟ้าของมาตรฐาน RS-232-C ครอบคลุมถึงรายละเอียดของแรงดันไฟฟ้า (voltage) และ กระแสสำหรับแต่ละขา ระดับของแรงดันไฟฟ้าจะแสดงเป็นแบบ TTL (Transsistor-Transistor Logic level) มาตรฐาน RS-232-C นั้นจะใช้ลอจิกเป็นลบ (negative logic) นั่นคือ ลอจิก "0" จะมีศักย์ไฟฟ้าเป็นบวกมากกว่าศักย์ไฟฟ้าสำหรับลอจิก "1" นั่นคือระดับศักย์ไฟฟ้าของลอจิก "1" จะเป็นลบเมื่อเทียบกับระดับศักย์ไฟฟ้าของ ลอจิก "0" ดังแสดงในตารางที่ 3.2 และรูปที่ 3.3 ช่วงศักย์ไฟฟ้าระหว่าง +3 โวลต์ ถึง -3 โวลต์ นั้นเป็นย่านของทรานซิสเตอร์และระดับสัญญาณจะไม่ถูกกำหนดในช่วงนี้ รูปที่ 3.3 แสดงช่วงศักย์ไฟฟ้าของไดรเวอร์ (driver) หรือตัวส่ง (transmitter) และสถานีปลายทาง (terminator end) หรือ ตัวรับ (receiver) สภาวะ mark (mark condition : ลอจิก "1") ของการส่ง คือ ระหว่าง +5 โวลต์ ถึง -15 โวลต์ เทียบกับระดับสัญญาณพื้น (ground) นั่นคือ ตัวส่งจะส่งสัญญาณลอจิก "1" จะต้องจ่ายแรงดันไฟฟ้าระหว่าง -5 โวลต์ ถึง -15 โวลต์ ส่วนตัวรับนั้น ถ้าแรงดันไฟฟ้าที่ได้รับมีค่าระหว่าง -3 โวลต์ ถึง -15 โวลต์ เมื่อเทียบกับสัญญาณพื้น สัญญาณนั้นจะถูกพิจารณาเป็นสภาวะ mark สภาวะ space (space condition : ลอจิก "0") ของการส่ง คือ +5 โวลต์ ถึง +15 โวลต์ เทียบกับสัญญาณพื้น ถ้าแรงดันไฟฟ้าที่ตัวรับ รับได้มีค่าระหว่าง +3 โวลต์ ถึง +15 โวลต์ เมื่อเทียบกับสัญญาณพื้นแล้ว สัญญาณนั้นจะถูกพิจารณาเป็นสภาวะ space สำหรับมาตรฐาน RS-232-C นี้อนุญาตให้มีขอบเขตของสัญญาณรบกวน (noise margin) ขนาดไม่เกิน 2 โวลต์ สำหรับแต่ละสภาวะ

ตารางที่ 3.2 ช่วงแรงดันไฟฟ้าของตัวรับ

	$-15V < V_1 < -3V$	$3V < V_1 < 15V$
Logic state	1	0
Signal	mark	space
Control	off	on

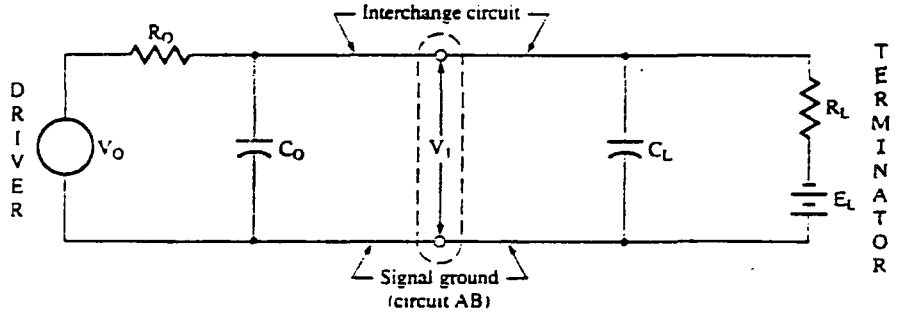
โดยปกติแล้วการเชื่อมต่อจะถูกสมมุติว่าใช้สำหรับส่งข้อมูลระหว่าง DTE และ DCE อย่างไรก็ตาม สายสัญญาณส่วนใหญ่ที่ระบุในมาตรฐานจะใช้สำหรับสัญญาณเวลาและสัญญาณควบคุม (timing and control signals) เพื่อที่สัญญาณจะถูกส่งแบบมีระเบียบ สายสัญญาณเวลาและสายสัญญาณควบคุมจะใช้คำว่า "ON" และ "OFF" แทน "space" และ "mark" ตามลำดับ ตารางที่ 3.2 และ รูปที่ 3.3 แสดงช่วงของศักย์ไฟฟ้าสำหรับสถานะ ON และ OFF



รูปที่ 3.3 ช่วงแรงดันไฟฟ้าของ RS-232-C

3.3 วงจรสมมูล (Equivalent Circuit)

จากรูปที่ 3.4 แสดงวงจรสมมูล (Equivalent Circuit) สำหรับการเชื่อมต่อแบบ RS-232-C ซึ่งตัวส่ง (Drive) คือ DTE และ ตัวรับ (Terminator) คือ DCE หรือกลับกัน



รูปที่ 3.4 แสดงวงจรสมมูลของการเชื่อมต่อ RS-232-C

จากรูปสามารถอธิบายได้ดังนี้

V_o เป็นแรงดันขณะเปิดวงจร (open circuit) ของตัวส่งเทียบกับกราวนด์ จะต้องมีขนาดไม่เกิน 25 โวลต์

R_o เป็นความต้านทานกระแสตรง (dc resistance) ภายใน ของตัวส่ง โดยขณะที่ปิดวงจร (short circuit) กระแสที่ไหลผ่านตัวต้านทานต้องไม่เกิน 0.5 แอมแปร์

C_o เป็นค่าผลรวมความจุไฟฟ้า (total effective capacitance) ของตัวส่งซึ่งคือ ผลรวมของค่าความจุไฟฟ้าของวงจรตัวส่งและค่าความจุไฟฟ้าของสายส่งทางด้านตัวส่ง

C_L เป็นค่าผลรวมความจุไฟฟ้า ของตัวรับ ซึ่งคือ ผลรวมของความจุไฟฟ้าของโหลดกับค่าความจุไฟฟ้าของสายส่งทางด้านตัวรับ โดย C_L จะต้องไม่เกิน 2,500 pF

R_L เป็นความต้านทานไฟฟ้ากระแสตรง (dc resistance) ของโหลดทางด้านตัวรับซึ่งเป็นโหลดในวงจรการเชื่อมต่อ มีค่าระหว่าง 3,000 ถึง 7,000 โอห์ม โดยที่ค่า R_L ต้องไม่น้อยกว่า 3,000 โอห์ม เมื่อป้อนแรงดันไฟฟ้าไม่เกิน 25 โวลต์ และ R_L จะต้องมีค่าไม่เกิน 7,000 โอห์ม เมื่อป้อนแรงดันไฟฟ้าภายในช่วง 3 ถึง 25 โวลต์

E_L เป็นแรงดันไฟฟ้าขณะเปิดวงจร (open circuit) ของตัวรับ และ จะต้องมีค่าไม่เกิน 2 โวลต์

V_i เป็นแรงดันไฟฟ้าที่จุดเชื่อมต่อ

ตามที่มาตรฐาน RS-232-C ระบุไว้นั้น แสดงถึงวงจรสมมูลที่มีเฉพาะตัวต้านทาน ตัวเก็บประจุ และแหล่งจ่ายแรงดันไฟฟ้า ที่เป็น passive components เท่านั้น อย่างไรก็ตาม ในการสร้างหรือต่อระบบแล้ว ผู้ออกแบบมักใช้ไอซีซึ่งเข้ากันได้กับมาตรฐาน RS-232-C และ EIA-232-D ตัวอย่างเช่น ไอซี MC1488 เป็น line driver และ ไอซี MC1489 เป็น line receiver เป็นต้น

3.4 <u>ลักษณะทางหน้าที่การทำงาน</u> (Funtional characteristic)

ตารางที่ 3.3 จัดแยกวงจรส่วนต่างๆ ของ RS-232 ตามหน้าที่การทำงานที่คล้ายกัน ตัวอักษรตัวแรกระบุถึงการจัดแยกประเภท และ ตัวอักษรตัวที่สองแทนวงจรเชื่อมต่อเฉพาะแต่ละส่วนภายในแต่ละประเภท การจัดแยกประเภทมีดังนี้

A - กราวนด์และสัญญาณกราวนด์ (Ground and common return)

B - วงจรข้อมูล (Data circuits)

C - วงจรควบคุม (Control circuits)

D - วงจรจังหวะเวลา (Timing circuits)

S - วงจรของ channel ที่สอง (secondary circuits)

ตารางที่ 3.3 RS-232 interchange circuits.

	Interchange circuit	Direction	
		DTE	DCE
AA	Protective ground (Does not exist in Version D)	- -	- -
AB	Signal ground	- -	- -
BA	Transmitted Data	---	-->
BB	Received Data	<--	---
CA	Request-to-send (RTS)	---	-->
CB	Clear-to-send (CTS)	<--	---
CC	DCE ready (also know as Data Set Ready)	<--	---
CD	Data terminal ready (DTR)	---	-->
CE	Ring/calling Indicator	<--	---
CF	Received Line Signal Detector (Data Carrier Detect)	<--	---
RL/CG	Remote Loop Back/Signal Quality Detector	<--	---
CH	Data Signal Rate Selector (DTE source)	---	-->
CI	Data Signal Rate Selector (DCE source)	<--	---

ตารางที่ 3.3 (ต่อ)

	Interchange circuit	Direction	
		DTE	DCE
DA	Transmitter Signal Element Timing (DTE source)	---	-->
DB	Transmitter Signal Element Timing (DCE source)	<--	---
DD	Receiver Signal Element Timing (DCE source)	<--	---
SBA	Secondary Transmitted Data	---	-->
SBB	Secondary Received Data	<--	---
SCA	Secondary Transmitted Data	---	-->
SCB	Secondary CTS	<--	---
SCF	Secondary Received Line Signal Detector	<--	---
LL	Local Loop back (D version only)		
TM	Test mode (D version only)		

การทำงานของแต่ละวงจร สรุปได้ดังนี้

3.4.1 กราวนด์ (Ground)

วงจร AA : Protective Ground (ขา 1)

ในการใช้งานวงจรนี้อาจนำไปต่อกับตัวถังอุปกรณ์เพื่อใช้เป็นสายดิน หรืออาจปล่อยลอยไว้โดยไม่มีการใช้งานก็ได้ ในมาตรฐาน RS-232-C จะกำหนดถึง protective ground นี้ไว้ แต่ใน EIA-233-D ไม่ได้ระบุไว้ เนื่องจากระบุรวมไว้ใน signal ground

วงจร AB : Signal Ground (ขา 7)

วงจรนี้นำไปใช้เป็นจุดอ้างอิงของสัญญาณที่ใช้ในวงจรต่างๆ จะต้องต่อไว้เสมอทั้งทางด้าน DTE และ DCE ไม่ว่าจะมีการประยุกต์ใช้งานแบบใด

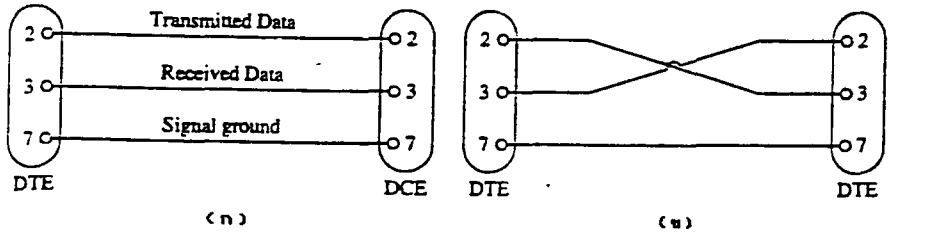
3.4.2 วงจรข้อมูล (Data Circuits)

วงจร BA : Transmitted Data (TxD, ขา 2) DTE --> DCE

วงจรนี้ใช้ในการส่งถ่ายข้อมูลจาก DTE ไปยัง DCE ของด้านตนเอง (local DCE) โดยในขณะที่ยังไม่มีการส่งข้อมูล DTE จะทำให้วงจร BA มีสถานะลอจิกเป็น "1" (mark) ตลอดเวลา ข้อมูลจะถูกส่งไปยัง DCE ก็ต่อเมื่อวงจรควบคุมอยู่ในสถานะ ON (ลอจิกเป็น "0" หรือสภาวะ space) ซึ่งวงจรควบคุมเหล่านี้ได้แก่ Request-to-send (RTS), Clear-to-send (CTS), Data Set Ready (DSR) และ Data Terminal Ready (DTR)

วงจร BB : Received Data (RxD, ขา 3) DTE <-- DCE

DTE จะรับข้อมูลจาก DCE (ของด้านตนเอง) โดยทางวงจร BB นี้ ข้อมูลจะถูกส่งจากสถานีอีกด้านที่อยู่ไกลออกไปมาทางช่องการสื่อสาร (communication channel) DCE ฝั่งนี้จะรับข้อมูลไว้ แล้วส่งให้ DTE โดยทางวงจร BB การอ้างอิงของชื่อขาสัญญาณหรือการรับส่งข้อมูลจะอ้างอิงกับตัว DTE ดังนั้น DTE จะส่งข้อมูลทางขา 2 และรับข้อมูลทางขา 3 ส่วน DCE จะส่งข้อมูลทางขา 3 และรับข้อมูลทางขา 2 ดังแสดงในรูปที่ 3.5(ก) จะเป็นการต่อแบบง่ายที่สุดสำหรับฟูลดูเพล็กซ์ (Full-Duplex) โดยที่ไม่มีวงจรหรือสายควบคุม รูปที่ 3.5(ข) เป็นการต่อ DTE เข้ากับ DTE โดยตรงโดยไม่มี DCE การต่อแบบนี้รู้จักกันในนาม Null Modem ในรูที่นั้นแสดงการต่อ null modem อย่างง่ายที่สุด สังเกตว่า จะต้องมีการไขว้สายสัญญาณระหว่างขา 2 และขา 3



รูปที่ 3.5 (ก) การต่อแบบง่ายที่สุดสำหรับฟูลดูเพล็กซ์ (ข) Null Modem

3.4.3 วงจรควบคุม (control circuits)

วงจร CA : Request to send (RTS, ขา 4) DTE --> DCE

วงจร RTS ถูกใช้ในการส่งสัญญาณจาก DTE ไปยัง DCE สัญญาณนี้ควบคุม local DCE ในการส่งและรับข้อมูลจากช่องการสื่อสาร สำหรับซิมเพล็กซ์(Simplex) และฟูลดูเพล็กซ์ (Full-Duplex) นั้น ถ้าสัญญาณ RTS เป็นสถานะ ON หมายถึง DCE จะอยู่ในโหมดการส่ง (transmit mode) ซึ่ง DCE รับข้อมูลจาก DTE และจะส่งผ่านไปทางช่องการสื่อสาร ถ้า RTS เป็น OFF แล้ว DCE จะอยู่ในโหมดไม่มีการส่ง (nontransmit mode) หมายถึง DCE จะไม่ส่งผ่านข้อมูลไปยังช่องการสื่อสาร

สำหรับฮาล์ฟดูเพล็กซ์ (Half-Duplex) แล้ว สถานะ ON จะทำให้ DCE อยู่ในโหมดการส่ง (transmit mode) และจะไม่รับข้อมูลจากช่องการสื่อสาร ส่วนสถานะ OFF จะทำให้ DCE อยู่ในโหมดการรับ (receive mode) คือ DCE จะรับข้อมูลจากช่องการสื่อสารแล้วส่งผ่านไปยัง DTE แต่ DCE จะไม่ส่งข้อมูลซึ่งรับจาก DTE ไปยังช่องการสื่อสาร ดังสรุปเป็นตารางได้ดังนี้

ตารางที่ 3.4 สรุปสัญญาณ RTS

สถานะ	ซิมเพล็กซ์และฟูลดูเพล็กซ์	ฮาล์ฟดูเพล็กซ์
ON	transmit mode	transmit mode
OFF	nontransmit mode	receive mode

วงจร CB : Clear to Send (CTS, ขา 5) DTE <-- DCE

DCE จะสร้างสัญญาณด้วยวงจรนี้แล้วส่งไปยัง DTE สถานะ ON เป็นการบอกแก่ DTE ว่าข้อมูลที่ส่งมายัง DCE นั้นจะถูกส่งไปทางช่องการสื่อสาร ในสถานะ OFF นั้น DTE จะไม่ส่งข้อมูลไปยัง local DCE สัญญาณ CTS จะอยู่ในสถานะ ON ก็ต่อเมื่อสัญญาณ RTS และ DSR จะต้องอยู่ในสถานะ ON ทั้งคู่ โดยที่ CTS จะมีสถานะ ON หลังจาก RTS มีสถานะ ON ไปแล้วเป็นเวลา 8 - 50 มิลลิวินาทีโดยทั่วไป หรืออาจใช้เวลานานถึง 200 มิลลิวินาทีก็ได้

วงจร CC : DCE Ready (Data Set Ready : DSR , ขา 6) DTE <-- DCE

สถานะ ON แสดงว่า DCE ได้ถูกต่อเข้ากับช่องการสื่อสารแล้วและพร้อมที่จะใช้ในการ

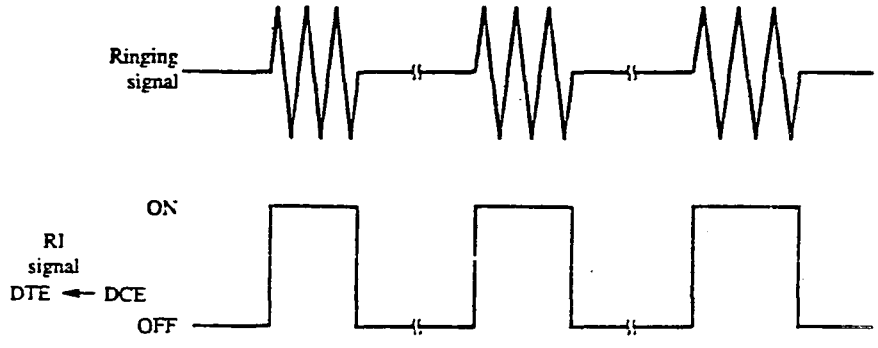
สื่อสาร และยังแสดงว่า local DCE ได้เสร็จสิ้นหน้าที่ทางด้านควบคุมหน้าที่ทางจังหวะเวลา (timing functions) แล้วด้วย สำหรับสถานะ OFF เป็นการบอกให้ DTE ให้ไม่สนใจต่อสัญญาณอื่นใด ยกเว้น สัญญาณแสดงกริ่ง (Ring Indicator (RI) : วงจร CF) ในมาตรฐาน RS-232-C นั้น วงจรนี้รู้จักกันในนาม Data Set Ready (DSR) ส่วนในมาตรฐาน EIA-232-D วงจรนี้รู้จักกันในนาม DCE Ready

วงจร CD : Data Terminal Ready (DTR, ขา 20) DTE --> DCE

วงจรนี้ใช้ในระบบ switched (dial-up) system สัญญาณของวงจรนี้จะส่งออกจาก DTE ไปยัง DCE สถานะ ON เป็นการบอกให้ DCE ให้เตรียมที่จะติดต่อกับช่องการสื่อสารได้ ซึ่งอาจจะโดย ระบบหมุนโทรศัพท์ด้วยมือ (manual dial) หรือจะโดย ระบบตอบรับอัตโนมัติ (automatic answer) ก็ได้ เนื่องจาก DTE ที่จะรับหรือส่งข้อมูลแล้ว เมื่อ DTR เปลี่ยนสถานะจาก ON เป็น OFF local DCE จะตัดการเชื่อมต่อกับ remote DCE

วงจร CE : Ring Indicator (RI, ขา 22) DTE <-- DCE

วงจรนี้ใช้ในระบบ switched (dial-up) network สัญญาณของวงจรนี้จะส่งออกจาก DCE ไปยัง DCE โดยที่สัญญาณ DTR ต้องมีสถานะเป็น ON เมื่อวงจร CE มีสถานะ ON แสดงว่า DCE กำลังได้รับสัญญาณเสียงกริ่ง (Ringing Signal) ที่มีเข้ามา เมื่อสถานะ OFF แสดงว่า DCE ไม่ได้รับสัญญาณเสียงกริ่ง วงจร CE นี้ทำให้ตัว DCE สามารถตอบรับต่อการเรียกโดยอัตโนมัติ (Automatic Answer)



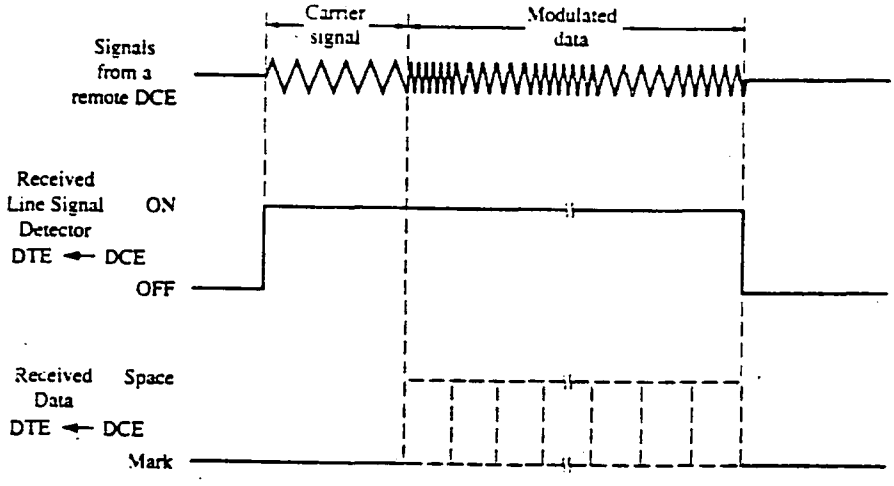
รูปที่ 3.6 ผลของสัญญาณ RI ที่ซ้อนอยู่กับสัญญาณเสียงกริ่ง

วงจร CF : Received Line Signal Detector หรือ

Data Carrier Detect (DCD, ขา 8) DTE <-- DCE

เมื่อวงจร CF มีสถานะ ON แสดงว่า DCE ได้รับสัญญาณพาหะ (Carrier Signal)

ที่เหมาะสมสำหรับการดีมอดดูเลท (Demodulate) มาจาก DCE ของอีกด้านหนึ่ง (remote DTE) แล้ว สถานะ OFF แสดงว่า DCE ไม่ได้รับสัญญาณใดๆ เข้ามา หรือสัญญาณที่เข้ารับมาได้ไม่สามารถทำการดีมอดดูเลทได้ สถานะ OFF ของวงจร CF นี้ ทำให้สาย Received Data (วงจร BB) มีสภาวะ mark ดังแสดงในรูปที่ 3.7



รูปที่ 3.7 แสดงสถานะของ DCD ในกรณีของฟูลดูเพล็กซ์

วงจร CG : Signal Quality Detector (ขา 21) DTE <-- DCE

วงจรนี้มักไม่ค่อยใช้กันแล้วในการออกแบบระบบ แต่ก็เป็นส่วนหนึ่งของมาตรฐาน เพื่อรองรับระบบที่ออกแบบมาก่อนหน้า สัญญาณนี้ใช้แสดงว่า มีความเป็นไปได้อย่างมากที่ขณะนั้นได้เกิดความผิดพลาดขึ้นแล้วในสาย Received Data สถานะ ON นั้นชี้ว่าไม่มีความผิดพลาดเกิดขึ้นหรือไม่น่าเชื่อว่าความผิดพลาดได้เกิดขึ้นจริง ส่วนสถานะ OFF นั้นชี้ว่ามีความเป็นไปได้สูงว่าได้เกิดความผิดพลาดขึ้นแล้ว ในการใช้งานบางอย่าง สถานะ OFF นี้จะเป็นตัวบอกว่าให้ส่งข้อมูลมาให้ใหม่โดยอัตโนมัติ

วงจร CH : Data Signal Rate Selector (ขา 23) DTE --> DCE หรือ

วงจร CI : Data Signal Rate Selector (ขา 23) DTE <-- DCE

เป็นวงจรที่เลือกใช้ตามแหล่งจ่าย (source) โดยถ้า DTE เป็นแหล่งจ่ายจะหมายถึงวงจร CH หรือถ้า DCE เป็นแหล่งจ่ายจะหมายถึงวงจร CI วงจรนี้ใช้ในการเลือกระหว่างอัตราการส่งข้อมูลที่ต่างกันสำหรับ DCE แบบซิงโครนัส (Synchronous) หรือให้อยู่ระหว่างขอบเขตสองค่าสำหรับ DCE แบบซิงโครนัส ในแต่ละกรณี สถานะ ON จะเลือกอัตราเร็วที่สูงกว่า

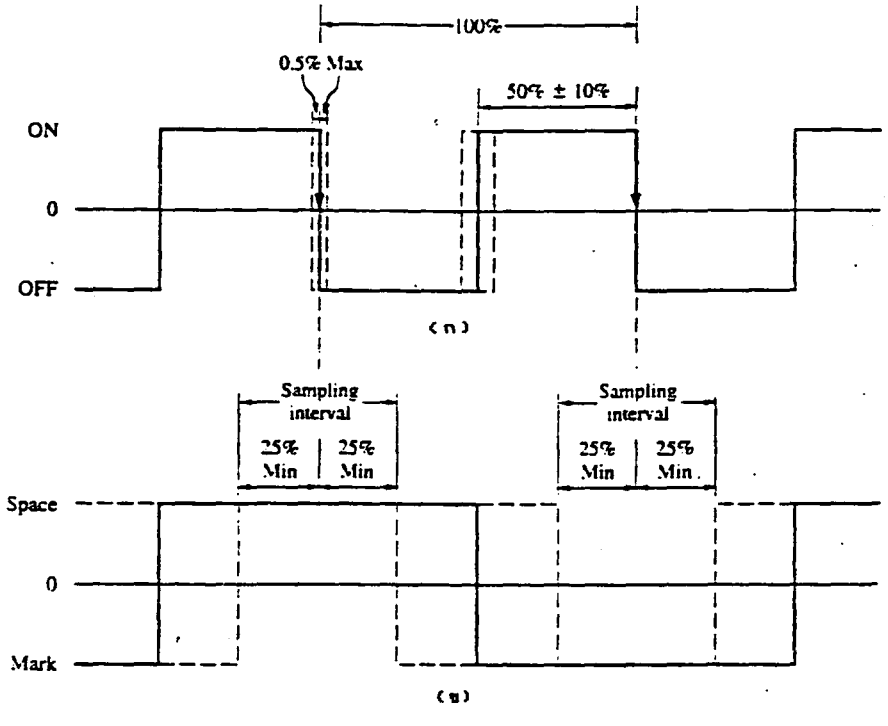
3.4.4 วงจรจังหวะเวลา (Timing Circuits)

เป็นวงจรที่ใช้ในการสื่อสารแบบซิงโครนัส (synchronous) มี 3 วงจร คือ

วงจร DA : Transmitting Signal Element Timing (ขา 24)

DTE --> DCE

วงจรนี้ถูกใช้โดย DTE เพื่อให้ข้อมูลของจังหวะเวลา (timing) สำหรับการส่งข้อมูลโดยวงจร BA (TxD) สัญญาณนี้จะเปลี่ยนสถานะจาก ON เป็น OFF ที่จุดกึ่งกลางของบิทข้อมูลที่ส่งออกไป ดังแสดงในรูปที่ 3.8 ตัวเลขที่อยู่ในรูปของเปอร์เซ็นต์แสดงข้อมูลของความคลาดเคลื่อนที่มากที่สุดที่ยอมให้เกิดขึ้นได้ของแต่ละส่วนของสัญญาณ

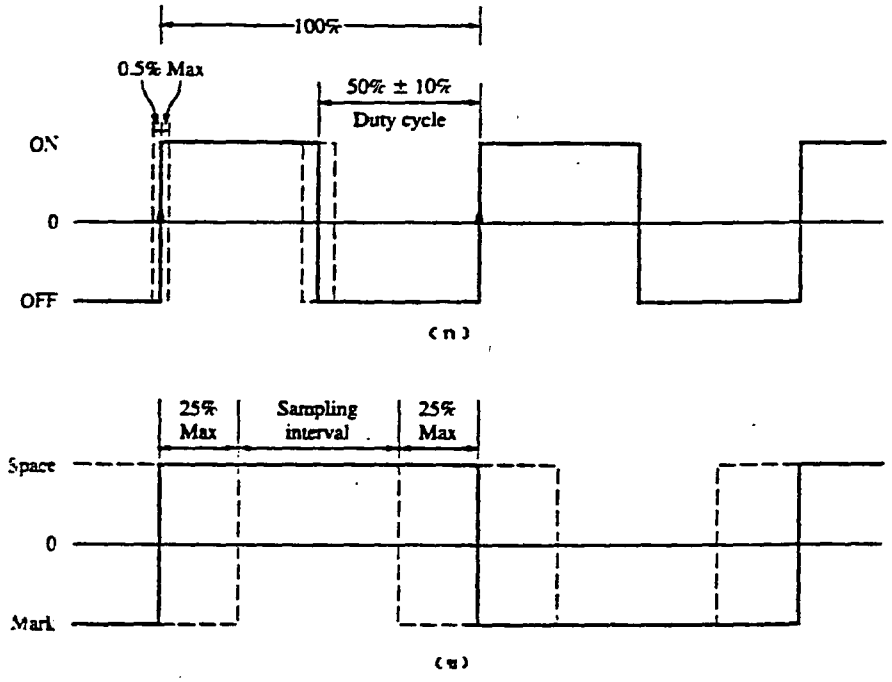


รูปที่ 3.8 (ก) รูปคลื่นของสัญญาณเวลาของวงจร DA
(ข) ข้อมูลที่ถูกส่งโดยวงจร BA

วงจร DB : Transmitting Signal Element Timing (ขา 15)

DTE <-- DCE

วงจรนี้คล้ายกับวงจร DA แต่ว่า DCE จะเป็นตัวส่งสัญญาณแทน โดย DTE จะส่งสัญญาณข้อมูลไปทางวงจร BA (TxD) ได้เมื่อสถานะของวงจร DB เปลี่ยนจาก OFF เป็น ON ดังรูปที่ 3.9

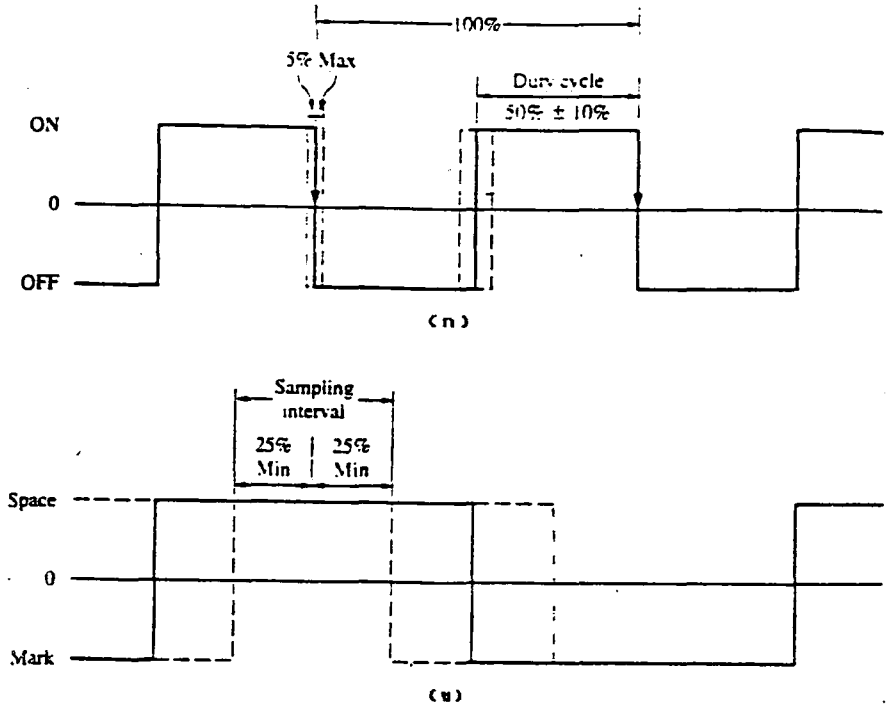


รูปที่ 3.9 (ก) สัญญาณจังหวะเวลาของวงจร DB

(ข) ข้อมูลที่ถูกส่งโดยวงจร BA

วงจร DD : Receiver Signal Element Timing (ขา 17) DTE <-- DCE

วงจร DD ถูกใช้สำหรับให้สัญญาณจังหวะเวลา (timing signal) ไปยัง DTE สถานะจะเปลี่ยนจาก ON เป็น OFF ที่จุดกึ่งกลางของบิทข้อมูลที่ถูกส่งมาทางวงจร BB (RxD) ดังรูปที่ 3.10 วงจรนี้ถูกใช้โดย DCE จะให้สัญญาณจังหวะเวลา อยู่ตลอดเวลาที่วงจร CF (DCD) มีสถานะเป็น ON ถ้า DCD มีสถานะ OFF สัญญาณจังหวะเวลาก็จะไม่ปรากฏ



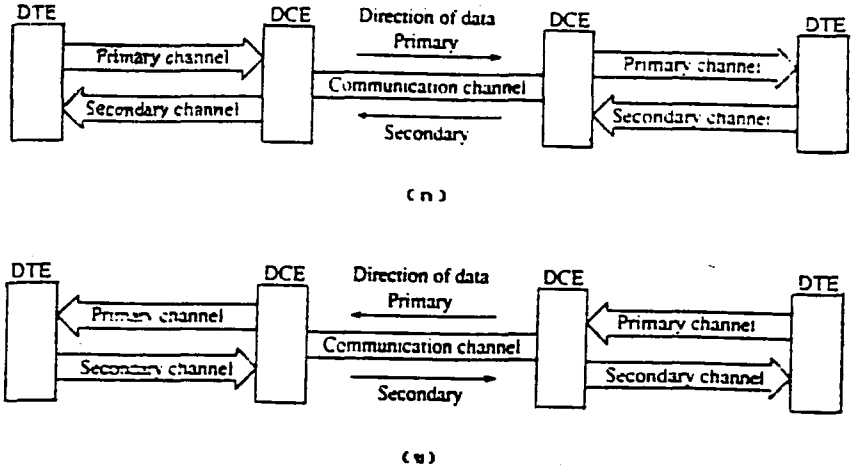
รูปที่ 3.10 (ก) สัญญาณจังหวะเวลาของวงจร DD

(ข) ข้อมูลที่รับได้ทางวงจร BB

3.4.5 วงจรของ Channel ที่สอง (Secondary Circuits)

ในวงจรต่างๆ ที่กล่าวมาก่อนหน้านี้เป็นการสื่อสารของ channel หลักหรือ channel ที่หนึ่ง (primary channel) แต่ใน RS-232-C ยังมี channel รองหรือ channel ที่สอง (secondary channel) ที่ใช้ในการสื่อสารด้วย โดยอาจจะใช้เป็นซิมเพล็กซ์ (Simplex) ฮาล์ฟดูเพล็กซ์ (Half-duplex) หรือ ฟูลดูเพล็กซ์ (Full-duplex) ก็ได้ แต่อัตราการส่งข้อมูลของ Channel ที่สองจะน้อยกว่าอัตราการส่งข้อมูลของ Channel ที่หนึ่ง Channel ที่สองนี้ แบ่งเป็น 2 ประเภท คือ Backward Channel และ Auxiliary Channel

Backward Channel จะมีทิศทางการสื่อสารตรงกันข้ามกับทิศทางของ Channel ที่หนึ่งเสมอ ดังแสดงในรูปที่ 3.11 ดังนั้น BackWard Channel จึงเรียกอีกชื่อว่า Reverse Channel คำว่า channel หลัก และ channel รอง หมายถึงวงจรในการเชื่อมต่อระหว่าง DTE กับ DCE ในบางระบบ Backward Channel จะไม่ใช้ในการส่งข้อมูลแต่ใช้สำหรับการควบคุมการขัดจังหวะ (interrupt control) โดยสัญญาณควบคุมการขัดจังหวะจะถูกส่งจากฝ่ายรับข้อมูลไปยังฝ่ายส่งข้อมูล เพื่อให้ยุติการส่งข้อมูลทาง channel ที่สอง



รูปที่ 3.11 ทิศทางของการสื่อสารของ channel ที่สอง
โดยใช้เป็น Backward Channel

Auxiliary Channel จะมีทิศทางการสื่อสารไม่ขึ้นกับทิศทางของ channel ที่หนึ่ง คือ มีทิศทางเดียวกัน หรือทิศทางตรงกันข้ามกับทิศทางของ channel ที่หนึ่งก็ได้

สำหรับวงจรของ Channel ที่สอง (Secondary Channel) มี 5 วงจรดังนี้

วงจร BBA : Secondary Transmitted Data (ขา 14) DTE --> DCE

การใช้ของวงจรนี้คล้ายกับวงจร BA (TxD) ซึ่งใช้กับ channel ที่หนึ่ง แต่วงจรนี้ใช้สำหรับ channel ที่สอง ข้อมูลจะถูกส่งจาก DTE ไป DCE วงจรนี้จะมีสภาวะ mark เมื่อไม่มีข้อมูลส่งมาและช่วงระหว่างตัวอักษร

วงจร SBB : Secondary Received Data (ขา 16) DTE <-- DCE

วงจรนี้ก็เหมือนวงจร BB (RxD) นั่นเอง แต่ใช้สำหรับ channel ที่สอง ในกรณีที่ channel ที่สอง ถูกใช้สำหรับอินเทอร์รัพท์ควบคุมการไหลของข้อมูลใน channel หลักแล้วนั้น วงจร SBB มักจะไม่ใช้ แต่จะใช้วงจร SCF แทน

วงจร SCA : Secondary RTS (ขา 19) DTE --> DCE

วงจรนี้คล้ายกับวงจร CA (RTS) แต่ใช้สำหรับ channel ที่สอง ในกรณีที่ใช้งาน channel ที่สองเป็นแบบ Backward Channel แล้ว สถานะของ RTS จะใช้ควบคุมวงจร SCA ให้ใช้ได้ หรือใช้ไม่ได้ ได้ด้วย โดยสถานะ ON ของวงจร CA (RTS) จะสามารถ ดีสเอเบิล (disable) วงจร SCA และสถานะ OFF ของวงจร CA (RTS) จะใช้ อีนาเบิล (enable) วงจร SCA ได้ ทั้งนี้เพื่อให้แน่ใจว่า channel ที่หนึ่ง และ channel ที่สอง จะไม่มีการส่งข้อมูลในทิศทางเดียวกัน

ในกรณี channel ที่สอง ถูกใช้เพียงเพื่อยืนยันหรือขัดจังหวะการไหลของข้อมูลใน channel ที่หนึ่งแล้ว วงจร SCA จะควบคุมสัญญาณพาหะที่ยังไม่ได้มอดดูเลตของ channel ที่สอง สถานะ ON ของวงจร SCA จะแสดงว่าวงจรมีความน่าเชื่อถือ (ไม่เกิดความผิดพลาดขึ้น) หรือให้สภาวะห้ามขัดจังหวะ (Noninterrupt Condition) และสถานะ OFF ของวงจร SCA จะแสดงว่า วงจรเกิดความผิดพลาดหรืออยู่ในสภาวะขัดจังหวะได้ (Interrupt Condition)

วงจร SCB : Secondary CTS (ขา 13) DTE <-- DCE

วงจรนี้เหมือนวงจร CB (CTS) แต่ใช้สำหรับ channel ที่สอง ในกรณี channel ที่สองถูกใช้เพียงเพื่อ การยืนยันวงจร (circuit assurance) หรือ ควบคุมการขัดจังหวะ (interrupt control) แล้ววงจร SCB นี้จะไม่ถูกใช้

วงจร SCF : Secondary Recieved Line Signal Detector (ขา 12)

DTE <-- DCE

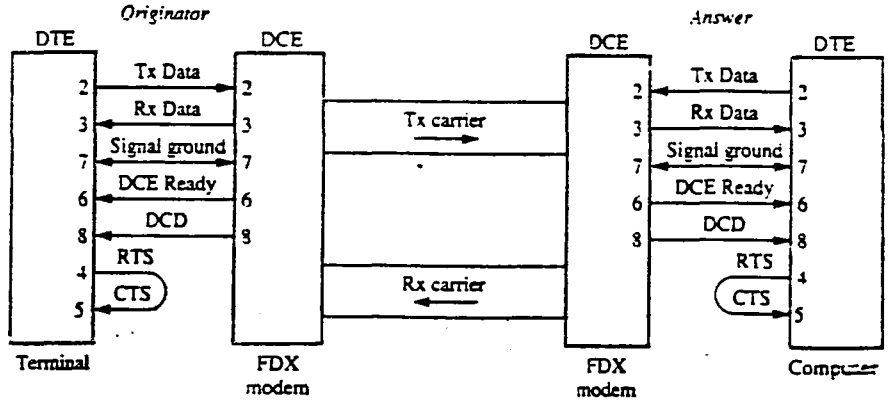
วงจรนี้จะแสดงการรับ สัญญาณพาหะ (Carrier Signal) สำหรับ channel ที่สอง ซึ่งจะมีหน้าที่คล้ายวงจร CF (DCD) ซึ่งใช้สำหรับ channel ที่หนึ่ง สถานะ ON ของวงจร SCF นี้ จะแสดงว่าสัญญาณพาหะของ channel ที่สองที่รับมานั้นมีความเหมาะสม (อยู่ในช่วงที่ใช้งานได้)

ในกรณีที่ channel ที่สองถูกใช้สำหรับ การยืนยันวงจรหรือสำหรับการขัดจังหวะแล้ว วงจร SCF จะแสดงสถานะของสถานีอีกฝ่ายที่อยู่ไกลออกไป (remote station) โดยสถานะ ON จะชี้ว่าวงจรมีความน่าเชื่อถือ หรืออยู่ในสถานะห้ามขัดจังหวะ (noninterrupt condition) สถานะ OFF จะชี้ว่า วงจรเกิดความผิดพลาดขึ้นหรืออยู่ในสภาวะขัดจังหวะได้ (interrupt condition)

3.5 การเชื่อมต่อโดยทั่วไปและสัญญาณควบคุม

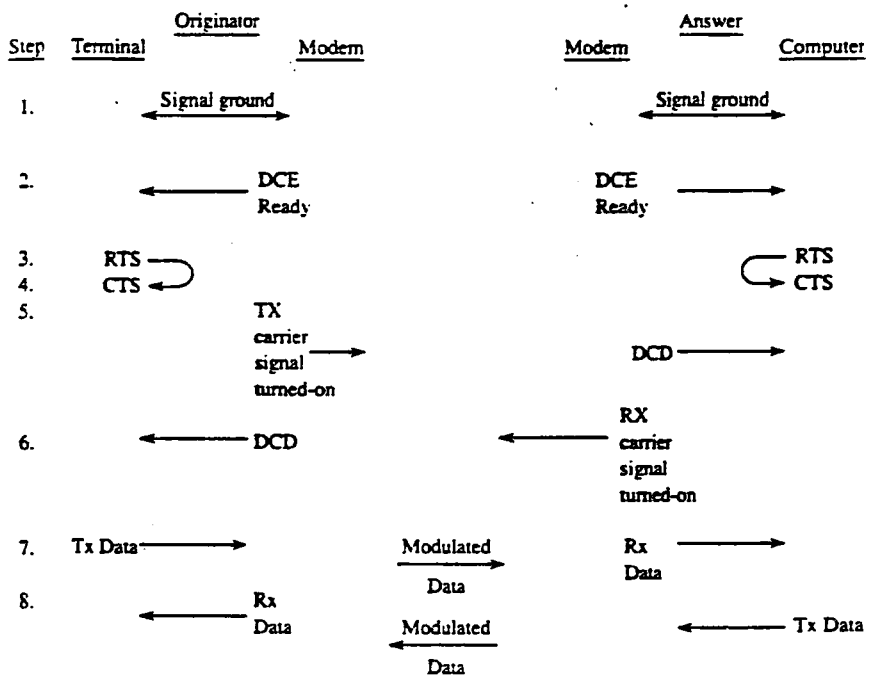
3.5.1 ระบบฟูลดูเพล็กซ์สายตรง (Full-Duplex Leased Line System)

การต่อ : รูปที่ 3.12 แสดงการต่อระบบฟูลดูเพล็กซ์สายตรงสี่สาย (Full Duplex Leased Four-Wire System)



รูปที่ 3.12 ระบบฟูลดูเพล็กซ์สายตรงสี่สาย

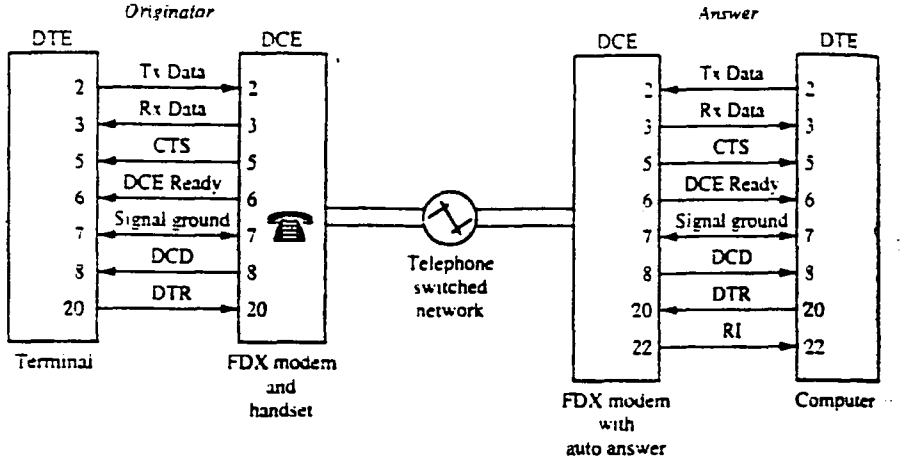
การติดต่อสื่อสาร : รูปที่ 3.13 แสดงขั้นตอนการส่งและตรวจสอบสัญญาณต่างๆ ในการติดต่อสื่อสารกันของระบบฟูลดูเพล็กซ์สายตรงสี่สาย



รูปที่ 3.13 การส่งและตรวจสอบสัญญาณของระบบฟูลดูเพล็กซ์สายตรงสี่สาย

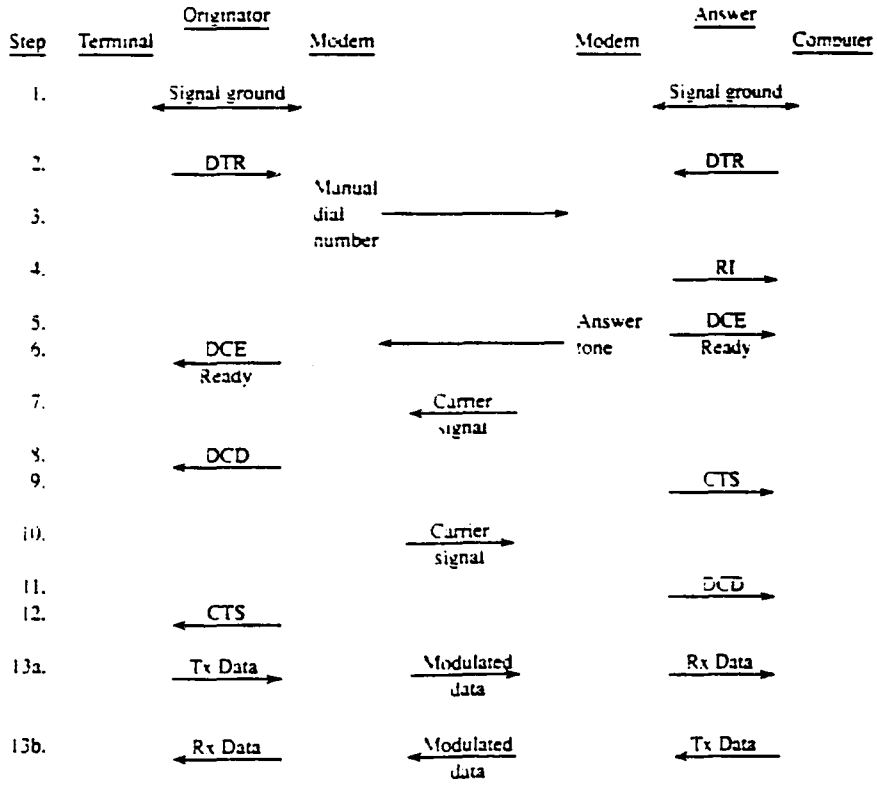
3.5.2 ระบบฟูลดูเพล็กซ์สายโทรศัพท์ (Full-Duplex Dial-Up System)

การต่อ : รูปที่ 3.14 แสดงการต่อระบบฟูลดูเพล็กซ์ แบบหมุนโทรศัพท์ด้วยมือและตอบรับอัตโนมัติ (Full-Duplex Manual Dial-Up and Auto Answer System)



รูปที่ 3.14 การต่อระบบฟลูเพล็กซ์แบบหมุนโทรศัพท์ด้วยมือและตอบรับอัตโนมัติ

การติดต่อสื่อสาร : รูปที่ 3.15 แสดงขั้นตอนการส่งและตรวจสอบสัญญาณต่างๆ ในการติดต่อสื่อสารกันของระบบฟลูเพล็กซ์ แบบหมุนโทรศัพท์ด้วยมือ

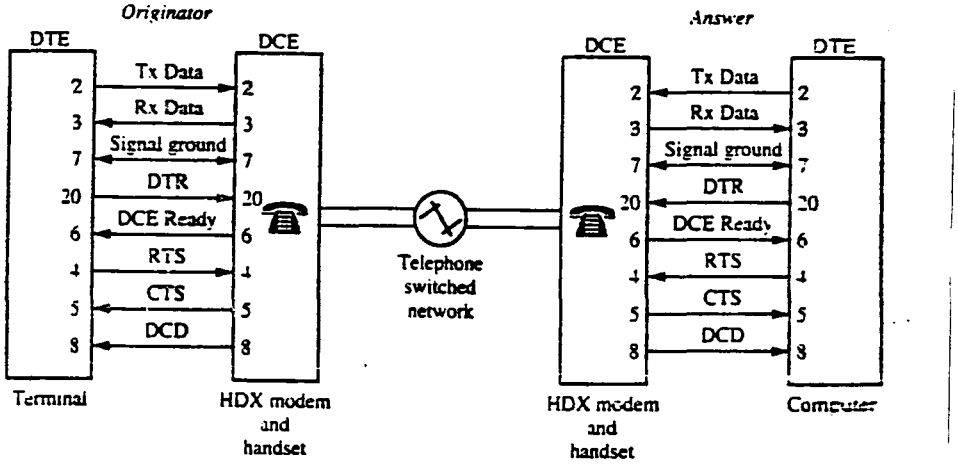


รูปที่ 3.15 ขั้นตอนการส่งและตรวจสอบสัญญาณของระบบฟลูเพล็กซ์แบบหมุนโทรศัพท์ด้วยมือ

3.5.3 ระบบฮาล์ฟดูเพล็กซ์สายโทรศัพท์ (Half-Duplex Dial-Up System)

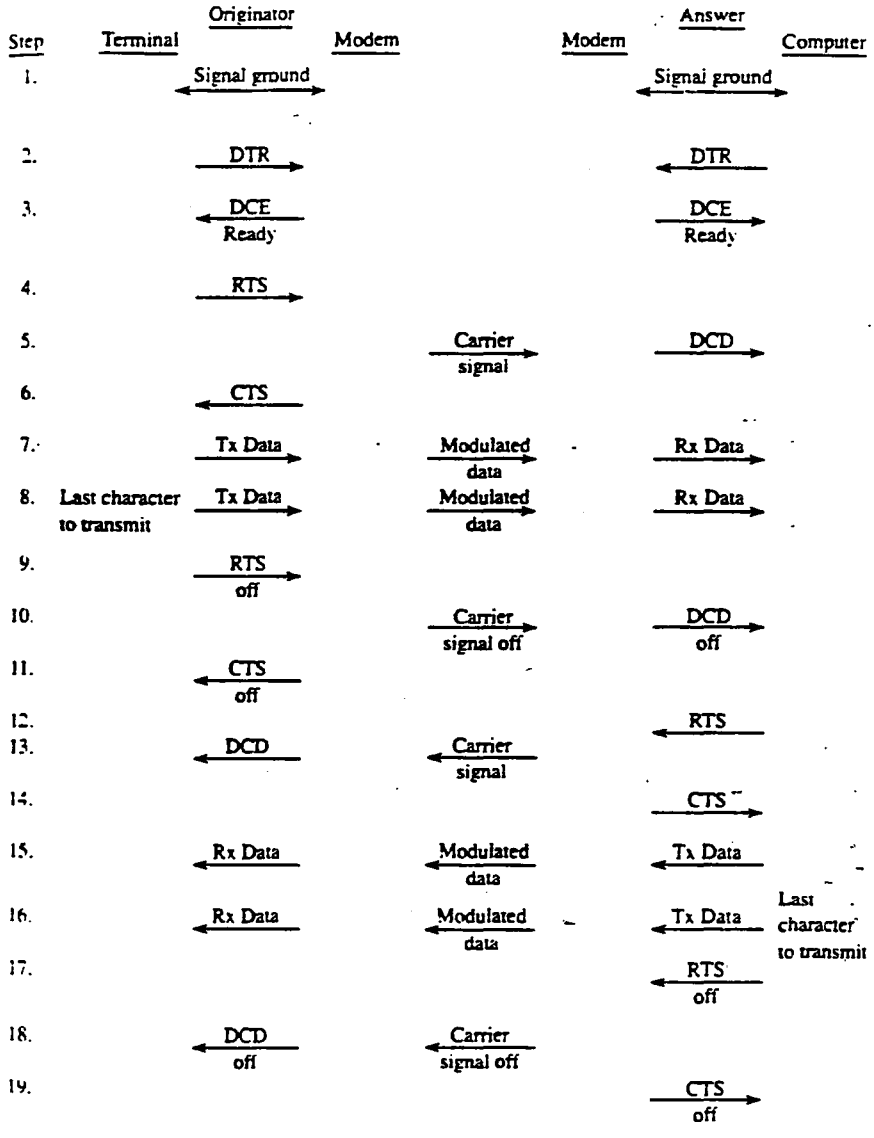
3.5.3.1 ระบบฮาล์ฟดูเพล็กซ์แบบตอบรับด้วยมือ (Half-Duplex Manual Answer System)

การต่อ : รูปที่ 3.16 แสดงการต่อระบบฮาล์ฟดูเพล็กซ์ แบบหมุนโทรศัพท์ด้วยมือและตอบรับด้วยมือ (Half-Duplex Manual Dial-Up Manual Answer System)



รูปที่ 3.16 การต่อระบบฮาล์ฟดูเพล็กซ์แบบหมุนโทรศัพท์ด้วยมือและตอบรับด้วยมือ

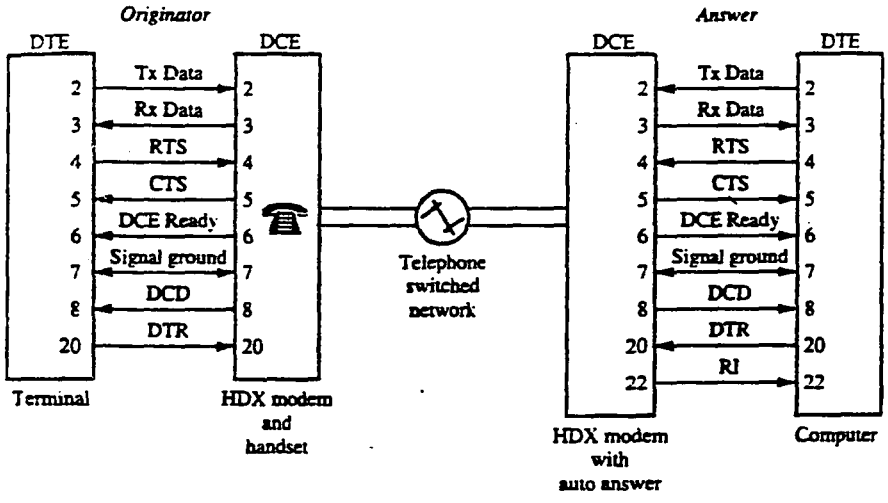
การติดต่อสื่อสาร : รูปที่ 3.17 แสดงขั้นตอนการส่งและตรวจสอบสัญญาณต่างๆ ในการติดต่อสื่อสารกันของระบบฮาล์ฟดูเพล็กซ์ แบบหมุนโทรศัพท์ด้วยมือและตอบรับด้วยมือ



รูปที่ 3.17 การส่งและตรวจสอบสัญญาณของระบบฮาล์ฟดูเพล็กซ์แบบหมุนโทรศัพท์และตอบรับด้วยมือ

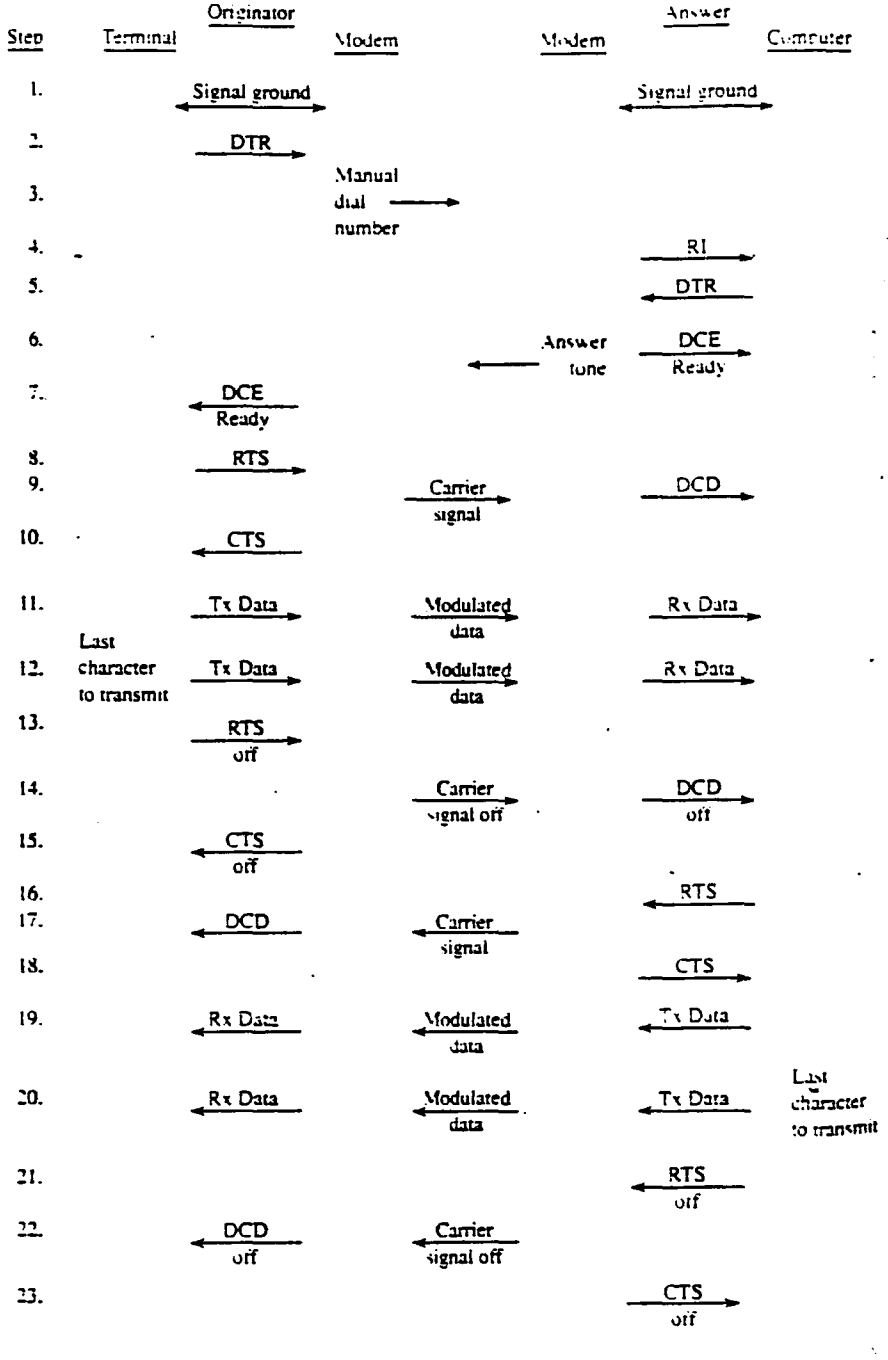
3.5.3.1 ระบบฮาล์ฟดูเพล็กซ์แบบตอบรับอัตโนมัติ (Half-Duplex Auto Answer System)

การต่อ : รูปที่ 3.18 แสดงการต่อระบบฮาล์ฟดูเพล็กซ์ แบบหมุนโทรศัพท์ด้วยมือและตอบรับอัตโนมัติ(Half-Duplex Manual Dial-Up and Auto Answer System)

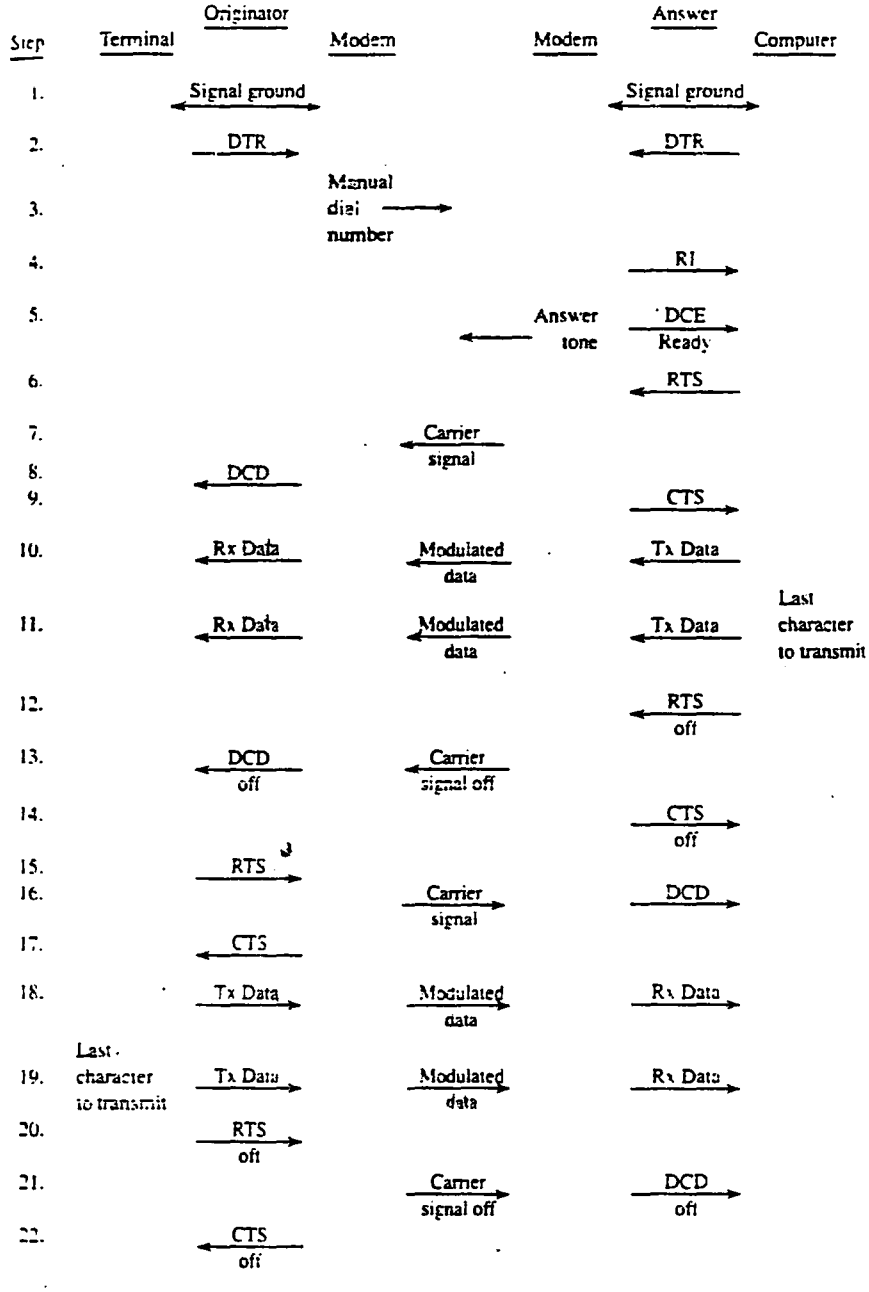


รูปที่ 3.18 การต่อระบบฮาล์ฟดูเพล็กซ์แบบหมุนโทรศัพท์ด้วยมือ และตอบรับอัตโนมัติ

การติดต่อสื่อสาร : รูปที่ 3.19 แสดงขั้นตอนการส่งและตรวจสอบสัญญาณต่างๆ ในการติดต่อสื่อสารกันของระบบ ในกรณีที่ฝ่ายเรียกไปเป็นผู้ที่จะส่งข้อมูลก่อน และ รูปที่ 3.20 เป็นกรณีที่ฝ่ายตอบรับจะเป็นผู้ส่งข้อมูลก่อน



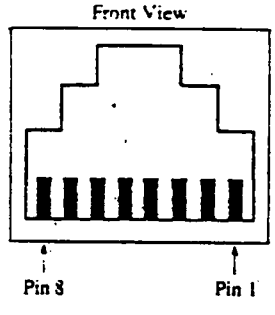
รูปที่ 3.19 แสดงขั้นตอนการส่งและตรวจสอบสัญญาณของระบบ
ในกรณีที่ฝ่ายเรียกไปเป็นผู้ที่จะส่งข้อมูลก่อน



รูปที่ 3.20 แสดงขั้นตอนการส่งและตรวจสอบสัญญาณของระบบ
ในกรณีที่ฝ่ายตอบรับจะเป็นผู้ที่จะส่งข้อมูลก่อน

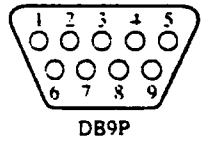
3.6 หัวต่อขนาดเล็ก (Miniature Connector)

ในกรณีที่การใช้งานไม่ต้องใช้สัญญาณการเชื่อมต่อที่กล่าวไปแล้วทั้งหมดทุกสัญญาณ โดยจะใช้เพียงบางส่วนที่จำเป็นเฉพาะบางขาของหัวต่อแล้วก็มักจะใช้หัวต่อขนาดเล็ก ซึ่งที่นิยมใช้กันมีสองชนิด คือ หัวต่อแบบ 8-pin modular connector ซึ่งใช้กับโทรศัพท์ โดยถูกกำหนดไว้ในมาตรฐาน EIA-561 และหัวต่อแบบ DB9 ซึ่งใช้กับเครื่องคอมพิวเตอร์ IBM PC โดยถูกกำหนดไว้ในมาตรฐาน EIA-574 สาเหตุที่หัวต่อดังกล่าวเป็นที่นิยม คือ ใช้งานง่าย ประหยัดพื้นที่ และมีฟังก์ชันที่จำเป็นสำหรับคอมพิวเตอร์ส่วนใหญ่ รูปที่ 3.21 แสดงหมายเลขของขาและชื่อของแต่ละขา



Pin Number	Designation
1	RI
2	Received Line Signal Dectector
3	DTE Ready
4	Signal ground
5	Received Data
6	Transmitted Data
7	CTS
8	RTS

(a)



Pin Number	Designation
1	Received Line Signal Detector
2	Transmitted Data
3	Received Data
4	DTE Ready
5	Signal ground
6	DCE Ready
7	RTS Ready for Receiving
8	CTS
9	RI

(b)

รูปที่ 3.21 หมายเลขของขาและชื่อแต่ละขาของหัวต่อชนิด

(ก) 8-Pin Modular Plug-and-Jack Connector

(ข) 9-Pin DB9 Connector

จากที่กล่าวมาทั้งหมดในบทนี้ จะเห็นว่ามาตรฐาน RS-232-C ซึ่งเป็นการรับส่งข้อมูลแบบอนุกรมนั้น ได้มีการกำหนดขึ้นมาเพื่อให้คอมพิวเตอร์ต่างยี่ห้อกัน หรืออุปกรณ์ต่อพ่วงแต่ละชนิดรับส่งข้อมูลกันได้ เมื่อทำตามมาตรฐานนี้ เราไม่จำเป็นต้องสนใจว่าอุปกรณ์หรือคอมพิวเตอร์นั้นจะผลิตมาจากที่ใด โดยมีการกำหนดรายละเอียดในการรับส่งข้อมูล เช่น หัวต่อ (connector)

ที่ใช้เป็นแบบใด มีสัญญาณที่ใช้กี่เส้น แต่ละสัญญาณทำหน้าที่อะไร และ ใช้ระดับแรงดันไฟฟ้าเท่าไรในการรับส่งข้อมูล ความเร็วในการรับส่งข้อมูลจะเป็นเท่าใดบ้าง ใช้ข้อมูลกี่บิทในการรับส่งข้อมูล ฯลฯ อุปกรณ์หรือคอมพิวเตอร์ที่ทำตามมาตรฐานนี้ เราจะสามารถเชื่อมต่อกันเพื่อรับส่งข้อมูลได้อย่างไม่มีปัญหา

บทที่ 4
โมเด็มกับการรับส่งข้อมูล

การรับส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์ในระยะทางไม่ไกลมากนัก เราอาจใช้การรับส่งแบบอนุกรม (RS-232-C) ซึ่งส่งข้อมูลดิจิตอลของคอมพิวเตอร์ไปตามสายจนถึงผู้รับได้ ในกรณีนี้ เราสามารถรับส่งข้อมูลได้ไกลถึง 35 เมตร ตามคุณสมบัติของ RS-232-C หรือ ถ้าสายเคเบิลที่ใช้มีคุณภาพดีอาจส่งได้ไกลถึง 150 เมตร ที่ความเร็ว 9,600 บิทต่อวินาที แต่สำหรับระยะทางที่ไกลมากๆ เช่น หลายสิบกิโลเมตร หลายร้อยกิโลเมตรจนถึงหลายพันกิโลเมตร การส่งข้อมูลแบบดิจิตอลออกไปโดยตรงจะไม่เหมาะสมหลายอย่าง ปัญหาที่สำคัญก็คือ คลื่นรูปสี่เหลี่ยมของสัญญาณดิจิตอล เมื่อส่งไปไกลจะเพี้ยนหรือมีรูปร่างผิดไปจากเดิมได้ง่าย ทำให้สายส่งและวงจรรับส่งสัญญาณดิจิตอลต้องถูกออกแบบมาเป็นอย่างดี ราคาของสายส่งสัญญาณแบบดิจิตอล จึงมีราคาแพงกว่าสายส่งสัญญาณแบบอนาล็อกมาก ในทางปฏิบัติ เราอาจรับส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์สองเครื่องโดยใช้สัญญาณดิจิตอลผ่านสายส่งได้ ซึ่งทั้งสายส่งและวงจรเชื่อมต่อทั้งหมดเป็นแบบดิจิตอล แต่ว่าค่าใช้จ่ายจะมีราคาแพงมากจนกระทั่งไม่ค่อยคุ้มที่จะทำเช่นนี้ วิธีหลีกเลี่ยงก็คือ หาทางส่งข้อมูลไปตามสายส่งในแบบอนาล็อกแทน การทำเช่นนี้เราจำเป็นต้องใช้อุปกรณ์เข้าช่วยแปลงสัญญาณในการรับส่งข้อมูลทั้งสองด้าน ซึ่งเป็นที่มาของโมเด็มนั่นเอง

โมเด็ม (MODEM) จะทำหน้าที่แปลงสัญญาณจากคอมพิวเตอร์ที่ส่งมาทาง RS-232-C ให้กลายเป็นสัญญาณอนาล็อกแล้วส่งออกไปตามสายส่ง กระบวนการนี้เราเรียกว่า การมอดดูเลท (Modulate) สัญญาณ เมื่อถึงปลายทาง โมเด็มก็จะแปลงสัญญาณอนาล็อกที่ได้รับ กลับมาเป็นสัญญาณดิจิตอล แล้วส่งให้คอมพิวเตอร์ต่อไปในรูปของสัญญาณดิจิตอลผ่านทาง RS-232-C เช่นกัน กระบวนการแปลงสัญญาณกลับนี้เรียกว่า การดีมอดดูเลท (Demodulation) ชื่อของโมเด็มก็ได้จากการทำงานทั้งสองแบบนี้นั่นเอง

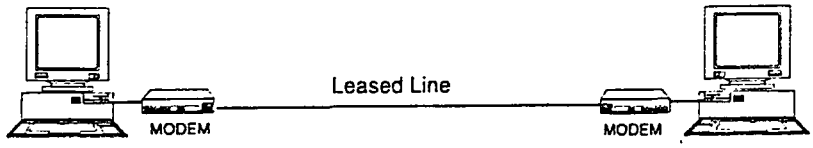
รูปที่ 4.1 โมเด็มช่วยให้คอมพิวเตอร์รับส่งข้อมูลผ่านสายโทรศัพท์ได้ โดยเปลี่ยนสัญญาณให้เป็นเสียงก่อน

สัญญาณอนาล็อกมีคุณสมบัติเหมาะที่จะส่งไปไกลๆ มากกว่าสัญญาณแบบดิจิตอล เพราะว่าสัญญาณอนาล็อกจะเพี้ยนหรือมีรูปร่างผิดจากเดิมยากกว่า และสูญเสียกำลังในสายส่งน้อยกว่าทำให้ส่งได้ระยะทางไกลมากขึ้น นอกจากนี้เรายังสามารถกรองสัญญาณรบกวนบางส่วนที่ไม่ต้องการออกได้อีกด้วย ราคาของสายส่งและอุปกรณ์เชื่อมต่อก็มีราคาถูก เราจึงจำเป็นต้องใช้โมเด็มในการรับส่งข้อมูลคอมพิวเตอร์ระยะทางไกลผ่านสายส่งอนาล็อก

จากการที่โมเด็มแปลงสัญญาณจากคอมพิวเตอร์ให้กลายเป็นสัญญาณอนาล็อกในการรับส่งข้อมูลนี้เอง ถ้าโมเด็มแปลงสัญญาณออกมาอยู่ในรูปของเสียงซึ่งเป็นสัญญาณอนาล็อกแบบหนึ่งเราก็สามารถรับส่งข้อมูลผ่านทางสายโทรศัพท์ได้ โมเด็มทั่วๆ ไปที่เราใช้งานจะเป็นโมเด็มที่แปลงสัญญาณจากคอมพิวเตอร์ให้อยู่ในรูปคลื่นเสียงทั้งนั้น มีโมเด็มบางชนิดที่แปลงสัญญาณดิจิตอลเป็นสัญญาณอนาล็อกความถี่สูง แต่โมเด็มแบบนี้มีใช้งานน้อย และส่งข้อมูลโดยใช้สายส่งพิเศษจะส่งผ่านสายโทรศัพท์ธรรมดาไม่ได้ เราจึงเน้นเฉพาะโมเด็มที่ทำงานในย่านความถี่เสียงเท่านั้น ไม่ว่าโมเด็มจะเป็นแบบไหนก็ตาม เมื่อได้รับข้อมูลดิจิตอลจากคอมพิวเตอร์ มันจะเปลี่ยนให้กลายเป็นสัญญาณอนาล็อก จากนั้นก็นำสัญญาณอนาล็อกที่ได้นี้มารวมเข้ากับสัญญาณพาหะ (Carrier Wave) แล้วส่งออกไปทางสายส่งข้อมูล สัญญาณพาหะหรือคลื่นพาหะนี้จะทำหน้าที่พาข้อมูลที่อยู่ในรูปสัญญาณอนาล็อกไปจนถึงปลายทาง

4.1 โมเด็มสำหรับสายตรงและสายโทรศัพท์

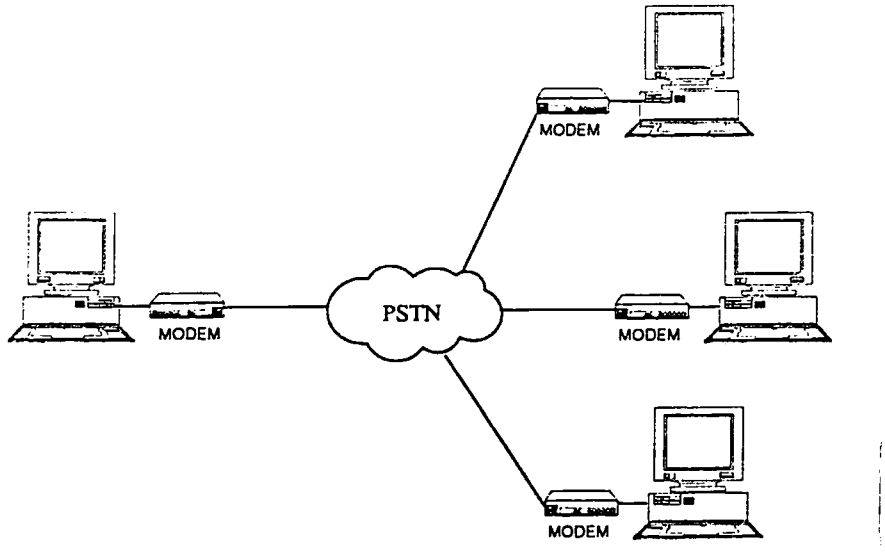
โมเด็มแบ่งตามการใช้งานได้สองแบบ คือ โมเด็มที่ใช้กับสายตรง (Leased Line) และ โมเด็มที่ใช้กับสายโทรศัพท์ (Dial Up Line) โมเด็มที่ใช้กับสายตรงหรือสายเช่าจะส่งข้อมูลด้วยความเร็วสูง (9,600 บิทต่อวินาทีหรือสูงกว่า) ผ่านสายไปยังจุดหมายปลายทางตายตัว ซึ่งเป็นการติดต่อในลักษณะจุดต่อจุด (Point to Point) จะต่อไปยังจุดอื่นๆ ไม่ได้ ส่วนมากจะเป็นการใช้ติดต่อส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์ ที่มีข้อมูลส่งไปมาเป็นจำนวนมาก เช่น เครือข่าย ATM ของธนาคาร เทอร์มินัลของคอมพิวเตอร์ขนาดใหญ่ซึ่งอาจจะเป็นมินิคอมพิวเตอร์หรือเมนเฟรม เป็นต้น การส่งข้อมูลมักจะส่งเป็นกลุ่มและมีซอฟต์แวร์ควบคุมการรับส่งโดยเฉพาะที่เราเรียกว่าการรับส่งแบบ Synchronous นั่นเอง ข้อดี ของโมเด็มแบบที่ใช้กับสายตรง คือ ส่งข้อมูลได้ที่ความเร็วสูง เนื่องจากสายส่งมีคุณภาพดีเหมาะกับงานส่งข้อมูลจำนวนมากระหว่างจุดสองจุด ข้อเสีย ก็คือ ไม่สามารถเปลี่ยนจุดรับข้อมูลไปที่ต่างๆ ได้ จึงขาดความคล่องตัว



รูปที่ 4.2 การรับส่งข้อมูลผ่านสายตรงจะเป็นการติดต่อระหว่างจุดต่อจุดตายตัว

โมเด็มแบบใช้กับสายโทรศัพท์ (Dial Up Line) จะรับส่งข้อมูลด้วยความเร็วตั้งแต่ 300 บิทต่อวินาที จนถึง 9,600 บิทต่อวินาที ผ่านเครือข่ายโทรศัพท์ที่เราใช้อยู่ เราสามารถรับส่งข้อมูลไปยังที่ต่างๆ ได้ตามต้องการ ไม่กำหนดจุดรับข้อมูลตายตัวเหมือนอย่างโมเด็มสายตรง เครื่องคอมพิวเตอร์ส่วนบุคคลและบริการรับส่งข้อมูลต่างๆ จะใช้โมเด็มแบบนี้ในการทำงานเกือบทั้งหมด การรับส่งข้อมูลจะเป็นการรับส่งทีละหนึ่งตัวอักษรไม่ส่งเป็นกลุ่ม เรียกว่า รับส่งแบบ Asynchronous ปกติแล้วเรารับส่งข้อมูลผ่านสายโทรศัพท์ด้วยความเร็ว 1,200 ถึง 2,400 บิทต่อวินาทีเท่านั้น

ความเร็วสูงสุดที่เชื่อถือได้คือ 9,600 บิทต่อวินาที ถ้าใช้โมเด็มความเร็วสูงกว่านี้รับส่งข้อมูลผ่านสายโทรศัพท์อาจทำได้ในบางจุด แต่ส่วนมากจะมีความผิดพลาดสูง จึงถือความเร็ว 9,600 บิทต่อวินาทีเป็นความเร็วสูงสุดสำหรับส่งข้อมูลผ่านสายโทรศัพท์ในปัจจุบัน ข้อดีของการใช้โมเด็มแบบนี้ก็คือ มีความคล่องตัวสูง สามารถรับส่งข้อมูลไปยังที่ต่างๆ ได้ไม่จำกัด และไม่จำเป็นต้องจัดหาวงจรสายตรงมาเป็นพิเศษเพื่อส่งข้อมูล แต่มีข้อเสียตรงที่ว่า ความเร็วในการส่งข้อมูลต่ำกว่าโมเด็มแบบสายตรง ข้อเสียอีกอันหนึ่งก็คือ ถ้ารับส่งข้อมูลเป็นจำนวนมากไปยังจุดปลายทางจุดเดียวในระยะทางไกล ค่าโทรศัพท์อาจจะแพงกว่าการเช่าวงจรสายตรงมาใช้ก็ได้ อันนี้ขึ้นกับเวลาที่ใช้ส่งข้อมูลเป็นหลัก



รูปที่ 4.3 การรับส่งข้อมูลผ่านสายโทรศัพท์ เราสามารถรับส่งข้อมูลกับจุดต่างๆ ได้หลายแห่งโดยหมุนเบอร์ปลายทางผ่านชุมสาย

โมเด็มที่มีขายในท้องตลาดปัจจุบัน บางรุ่นอาจทำงานได้เฉพาะกับสายตรงหรือบางรุ่นอาจทำงานได้กับสายโทรศัพท์เท่านั้น โมเด็มที่ใช้กับสายตรงได้เพียงอย่างเดียวจะนำมาต่อใช้กับสายโทรศัพท์ไม่ได้และโมเด็มที่ใช้กับสายโทรศัพท์ได้อย่างเดียวจะนำมาใช้งานต่อกับสายตรงไม่ได้เช่นกัน ยกเว้นโมเด็มบางชนิดซึ่งเลือกการทำงานในตัวได้ว่าจะต่อกับสายตรงหรือสายโทรศัพท์ โมเด็มที่ใช้งานได้ทั้งสองอย่างจะมีราคาแพงกว่าโมเด็มที่ใช้กับสายโทรศัพท์เพียงอย่างเดียว ถ้าการใช้งานของเราไม่ใช่การต่อคอมพิวเตอร์ส่วนบุคคลเข้ากับเมนเฟรมหรือมินิคอมพิวเตอร์แล้วละก็การใช้โมเด็มแบบต่อกับสายโทรศัพท์ได้อย่างเดียวก็นับว่าเพียงพอแล้วสำหรับรับส่งข้อมูลทั่วไป

4.2 โทรศัพท์แบบกดปุ่มและแบบหมุน

เมื่อเราใช้โมเด็มรับส่งข้อมูลผ่านสายโทรศัพท์ วิธีติดต่อไปยังปลายทางที่ต้องการก็คือหมุนโทรศัพท์ของเบอร์ปลายทางก่อนที่จะส่งข้อมูล การหมุนโทรศัพท์นี้อาจจะให้คนหมุนเองหรือใช้โมเด็มหมุนโทรศัพท์แบบอัตโนมัติก็ได้ โทรศัพท์ที่ใช้อยู่ทุกวันนี้มีสองแบบคือแบบกดปุ่มและแบบใช้มือหมุน โมเด็มจะต้องรู้ว่าโทรศัพท์ที่ต่อนั้นเป็นแบบไหนจึงจะสามารถหมุนโทรศัพท์ไปหาปลายทางได้ โทรศัพท์แบบกดปุ่ม (Tone Dialing) เป็นโทรศัพท์แบบที่เราใช้กันส่วนมาก การเรียกติดต่อไปยังเบอร์ปลายทางจะใช้วิธี กดปุ่มบนหมายเลขที่ต้องการส่งไปให้ชุมสายโทรศัพท์ สัญญาณในสายโทรศัพท์จะมีลักษณะเป็นเสียงความถี่ต่าง ๆ ตามเบอร์ที่เรากด สังเกตได้จาก เสียงที่เกิดขึ้น

เมื่อเรากดเบอร์โทรศัพท์ไปหาผู้อื่น โมเด็มที่หมุนโทรศัพท์แบบกดปุ่มจะทำหน้าที่ส่งสัญญาณเสียงตามเบอร์ที่เรากำหนดไปให้ชุมสาย แทนที่จะให้คนมานั่งกดเบอร์ต่างๆ ด้วยตัวเอง

ส่วนโทรศัพท์แบบใช้มือหมุน (Pulse Dialing) นั้น มักจะเป็นโทรศัพท์รุ่นเก่าใช้วิธีหมุนหน้าปัดแทนการกดปุ่ม เมื่อเราหมุนหน้าปัดโทรศัพท์จะส่งสัญญาณเป็นช่วงๆ ตามเบอร์ที่หมุนไปให้ชุมสาย เช่น เบอร์ "1" จะส่งสัญญาณไปหนึ่งครั้ง เบอร์ "2" จะส่งสัญญาณไปสองครั้ง เป็นต้น ขณะที่เราหมุนโทรศัพท์ไปยังเบอร์ปลายทางเราจึงได้ยินเสียง คลิ๊ก...คลิ๊ก...คลิ๊ก ตามเบอร์ต่างๆ เป็นจังหวะ การหมุนโทรศัพท์แบบนี้จึงช้ากว่าแบบกดปุ่มมาก

โมเด็มทั่วไปมักจะหมุนโทรศัพท์ได้ทั้งแบบกดปุ่มและแบบมือหมุน ที่เราต้องสั่งโมเด็มให้หมุนโทรศัพท์ให้ถูกต้องก็เนื่องมาจาก ถ้าโทรศัพท์ที่เราใช้เป็นแบบมือหมุน แต่สั่งให้โมเด็มหมุนโทรศัพท์แบบกดปุ่ม ชุมสายจะไม่รับสัญญาณการหมุนโทรศัพท์ของโมเด็มทำให้ติดต่อไปยังปลายทางไม่ได้ และทำนองกลับกัน ถ้าเราใช้โทรศัพท์แบบกดปุ่มแต่ส่งสัญญาณให้ชุมสายโทรศัพท์แบบมือหมุนก็จะต่อไปหาปลายทางไม่ได้เช่นกัน เนื่องจากสัญญาณการหมุนโทรศัพท์ไม่ถูกต้อง ดังนั้นเราจึงต้องสั่งให้โมเด็มหมุนโทรศัพท์ให้ตรงตามชนิดของโทรศัพท์ที่เรามีอยู่ วิธีการง่ายๆ ที่จะรู้ว่าโทรศัพท์ของเราเป็นแบบไหนก็คือฟังเสียงขณะที่เราหมุนโทรศัพท์ไปหาเบอร์ปลายทางตามปกติ ถ้ามีเสียงลักษณะสูงๆ ต่ำๆ ละก็ แสดงว่าโทรศัพท์ของเราเป็นแบบกดปุ่ม (Tone Dialing) แต่ถ้าได้ยินเสียงลักษณะเป็นช่วงๆ แสดงว่าโทรศัพท์ของเราเป็นแบบมือหมุน (Pulse Dialing)

บางคนอาจจะสงสัยว่าทำไมไม่ดูที่เครื่องโทรศัพท์ล่ะว่ามีปุ่มกดหรือมีหน้าปัดหมุน คำตอบก็คือ มันไม่แน่เสมอไป เพราะโทรศัพท์กดปุ่มบางแบบ จะเปลี่ยนสัญญาณที่เรากดปุ่ม ให้กลายเป็นสัญญาณ คลิ๊ก...คลิ๊ก ให้ใช้กับชุมสายโทรศัพท์แบบมือหมุนได้ การดูจากรูปร่างของเครื่องโทรศัพท์จึงบอกไม่ได้ว่าโทรศัพท์ของเราเป็นแบบกดปุ่มหรือแบบมือหมุน อย่างไรก็ตาม โทรศัพท์รุ่นใหม่ๆ จะเป็นแบบกดปุ่มกันทั้งนั้น แบบมือหมุนจะค่อยๆ ถูกเปลี่ยนไปจนหมด ตามโครงการขององค์การโทรศัพท์ฯ

4.3 คุณสมบัติของฟูลดูเพล็กซ์และฮาล์ฟดูเพล็กซ์

ในการรับส่งข้อมูลระหว่างกันนั้น อาจแบ่งตามลักษณะของการรับส่งได้เป็น 3 วิธีใหญ่ๆ คือ

- การรับหรือส่งทางเดียว (Simplex)
- การรับส่งแบบผลัดกันส่ง (Half Duplex)
- การรับส่งสวนทางได้พร้อมกัน (Full Duplex)

การติดต่อสื่อสารส่งข้อมูลกันแบบที่รับหรือส่งทางเดียวนั้นเป็นการสื่อสารแบบ ซิมเพล็กซ์ (Simplex) ตัวอย่างง่ายๆ ที่เห็นได้ชัดก็คือ การรับส่งโทรศัพท์และวิทยุกระจายเสียง สถานีโทรทัศน์จะเป็นตัวส่งและเครื่องรับทำหน้าที่รับแต่เพียงอย่างเดียวจะส่งข่าวหรือภาพกลับมายังสถานีส่งไม่ได้ การสื่อสารแบบซิมเพล็กซ์นี้ มักจะไม่ค่อยนำมาใช้ในการสื่อสารข้อมูล เนื่องจากว่าเราจำเป็นต้องโต้ตอบกันระหว่างการรับส่งข้อมูล หรือบางทีก็เปลี่ยนจากผู้รับเป็นผู้ส่ง ซึ่งทำให้สำหรับการติดต่อกันในแบบซิมเพล็กซ์นอกจากจะใช้สำหรับส่งโทรทัศน์ และวิทยุกระจายเสียงแล้ว เครื่องโทรพิมพ์ ตามสำนักพิมพ์บางชนิด อาจใช้การติดต่อแบบนี้เช่นกันในการรับข่าวสารจากที่อื่นๆ เพียงอย่างเดียว

การรับส่งแบบฮาล์ฟดูเพล็กซ์ (Half Duplex) มีคุณสมบัติสามารถรับรู้และส่งข้อมูลได้ แต่ว่าต้องสลับกันส่ง จะส่งพร้อมกันทั้งสองด้านไม่ได้ ตัวอย่างของอุปกรณ์ที่ใช้การติดต่อในแบบฮาล์ฟดูเพล็กซ์ ได้แก่วิทยุมือถือ (Walkie-talkie) และ INTERCOM เป็นต้น เมื่อฝ่ายใดฝ่ายหนึ่งส่งอีกฝ่ายก็จะทำหน้าที่รับ จนกระทั่งฝ่ายแรกส่งจบฝ่ายหลังจึงจะกลับเป็นผู้ส่งได้ และฝ่ายส่งในตอนแรกก็จะเป็นผู้รับสลับกันเช่นนี้เรื่อยไป ทั้งสองฝ่ายจะเป็นผู้ส่งพร้อมกันไม่ได้เพราะสัญญาณจะชนกันทำให้ฟังไม่รู้เรื่อง การรับส่งในแบบฮาล์ฟดูเพล็กซ์นับว่าซับซ้อนกว่าในแบบซิมเพล็กซ์ขึ้นมาหน่อย เพราะทั้งสองด้านสามารถทำหน้าที่รับและส่งได้ตามลำดับ อย่าส่งชนกันเป็นอันใช้ได้

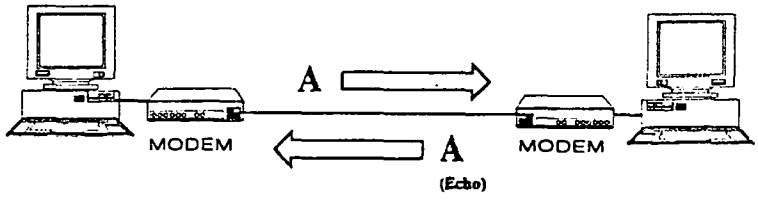
การรับส่งในแบบสวนทางได้พร้อมกัน เรียกว่า ฟูลดูเพล็กซ์ (Full Duplex) การรับส่งแบบนี้ ผู้รับและส่งสามารถรับและส่งพร้อมๆ กันในเวลาเดียวกันได้ ไม่จำเป็นต้องรอให้อีกฝ่ายหนึ่งส่งจบเสียก่อนอย่างในฮาล์ฟดูเพล็กซ์ ตัวอย่างเช่น การพูดโทรศัพท์ของเรา ถึงแม้ปกติเมื่อผู้หนึ่งพูดอีกฝ่ายจะคอยฟัง แล้วตอบกลับมาเมื่อฝ่ายแรกพูดจบซึ่งเป็นลักษณะของการติดต่อแบบฮาล์ฟดูเพล็กซ์ ก็ตาม แต่เราอาจจะพูดพร้อมๆ กันหรือพูดสวนกลับไปได้ทันทีโดยยังคงฟังอยู่เหมือนเดิม ลักษณะเช่นนี้เราเรียกว่าติดต่อกันในแบบฟูลดูเพล็กซ์ การสื่อสารข้อมูลระหว่างคอมพิวเตอร์สองเครื่อง มีใช้ทั้งแบบฮาล์ฟดูเพล็กซ์และฟูลดูเพล็กซ์ ขึ้นอยู่กับลักษณะของการเชื่อมต่อและงานของมัน

4.4 ลักษณะการรับส่งและการ Echo ของข้อมูล

เมื่อคอมพิวเตอร์รับส่งข้อมูลในแบบฮาล์ฟดูเพล็กซ์หรือฟูลดูเพล็กซ์ก็ตาม มันจะต้องใช้การรับส่งให้เหมือนกันทั้งสองด้าน การรับส่งข้อมูลแบบฮาล์ฟดูเพล็กซ์ เครื่องคอมพิวเตอร์ผลัดกันส่งข้อมูล และมีหน้าที่พิมพ์ข้อความที่ตัวเองส่งออกไปขึ้นแสดงบนจอภาพด้วย หรือมีคุณสมบัติที่เรียกว่า "Echo On" นั่นเอง ข้อความไม่ว่าจะส่งออกไปโดยการพิมพ์จากแป้นพิมพ์หรืออ่านจากแผ่นดิสก์ก็ตาม เครื่องคอมพิวเตอร์ที่ส่งข้อความจะต้องนำข้อความนั้นแสดงผลออกทางจอภาพด้วย

ตนเอง การรับส่งข้อมูลในแบบฮาล์ฟดูเพล็กซ์จึงต้องกำหนดให้เครื่องคอมพิวเตอร์หรือเทอร์มินัลที่ใช้ ทำงานในลักษณะ Echo On เสมอ มิฉะนั้นเราจะมองไม่เห็นข้อความที่เราส่งออกไป มองเห็นแต่เฉพาะข้อความที่อีกฝ่ายหนึ่งส่งมาเท่านั้น

เมื่อคอมพิวเตอร์รับส่งข้อมูลในแบบฟูลดูเพล็กซ์ มันก็สามารถรับและส่งข้อมูลสวนทางกันได้ในเวลาเดียวกัน เครื่องคอมพิวเตอร์ที่รับส่งแบบฟูลดูเพล็กซ์จะไม่พิมพ์ข้อความที่ตัวเองส่งออกไปขึ้นแสดงบนจอภาพ แต่จะรอรับข้อความจากอีกฝ่ายหนึ่งส่งกลับมาให้เท่านั้น เราเรียกว่า "Echo Off" ข้อความจากแป้นพิมพ์และจากแผ่นดิสก์ที่ส่งออกไป ปกติแล้วเราจะมองไม่เห็น เครื่องคอมพิวเตอร์อีกด้านหนึ่งจะส่งข้อความนั้นกลับมาให้ปรากฏบนจอภาพของเราเองเมื่อมีความจำเป็นการรับส่งข้อมูลแบบฟูลดูเพล็กซ์ จึงต้องกำหนดให้คอมพิวเตอร์ทำงานในลักษณะ Echo Off เสมอ ถ้ากำหนดผิด เราจะเห็นข้อความที่พิมพ์ออกไปกลายเป็นสองตัวซ้อนกันอย่างนี้บนจอภาพ "LLiikkee TThhiiss"



รูปที่ 4.4 การรับส่งแบบฟูลดูเพล็กซ์ ผู้รับจะต้องส่งข้อมูลกลับ (echo) ไปให้ผู้ส่งเสมอ

ในการเชื่อมต่อระหว่างคอมพิวเตอร์นั้น การรับส่งแบบฮาล์ฟดูเพล็กซ์จะมีประสิทธิภาพต่ำกว่าเนื่องจากต้องผลัดกันส่งข้อมูล แต่ก็มีข้อดี คือ ประหยัดสายส่งข้อมูลเพราะเราสามารถใช้สายเพียงคู่เดียวในการรับส่งข้อมูลแบบฮาล์ฟดูเพล็กซ์นี้ การรับส่งแบบฟูลดูเพล็กซ์จะต้องใช้สายสองคู่ คือ สำหรับส่งข้อมูลหนึ่งคู่และรับข้อมูลอีกหนึ่งคู่ แยกจากวงจรรับส่งให้เป็นอิสระออกจากกัน ประสิทธิภาพในการรับส่งข้อมูลจึงสูงกว่าในแบบฮาล์ฟดูเพล็กซ์ถึงสองเท่า

4.5 เทคนิคที่ใช้ส่งข้อมูลแบบฟูลดูเพล็กซ์ด้วยสายคู่เดียว

จากการที่ฟูลดูเพล็กซ์มีข้อดีหลายอย่างในการรับส่งข้อมูล แต่มีข้อเสียตรงที่ต้องใช้สายสองคู่หรือสายสี่เส้นเพื่อส่งข้อมูล จึงได้มีผู้พยายามพัฒนาลดจำนวนสายส่งลงให้เหลือเพียงหนึ่งคู่เท่ากับที่ใช้ในฮาล์ฟดูเพล็กซ์ ทั้งนี้เนื่องจากค่าใช้จ่ายของสายส่งระยะทางไกลๆ มีราคาแพง ถ้าต้องใช้สายถึงสองคู่ก็จะทำให้ค่าใช้จ่ายของสายส่งข้อมูลแพงมากจนเกินไป สำหรับการเชื่อมต่อ

ระยะใกล้ๆ เราอาจยอมใช้สายสองคู่รับส่งข้อมูลแยกจากกันได้ แต่การใช้งานระยะทางไกลแล้ว สายคู่เดียวจะประหยัดและสะดวกกว่ามาก โดยเฉพาะอย่างยิ่งในกรณีที่เรารับส่งข้อมูลด้วยความเร็วไม่สูงมากนัก การใช้สายเพียงคู่เดียวส่งข้อมูลในแบบฟูลดูเพล็กซ์เป็นสิ่งที่เป็นไปได้ และนับเป็นการใช้งานสายส่งข้อมูลอย่างคุ้มค่าอีกด้วย

คุณสมบัติของสายส่งข้อมูลนั้นจะสามารถส่งสัญญาณไฟฟ้าผ่านไปยังปลายทางได้โดยมีช่วงความถี่ช่วงหนึ่ง ช่วงความถี่ที่สายส่งข้อมูลส่งผ่านไปถึงปลายทางได้นี้เราเรียกว่า bandwidth ของสายส่ง อย่างเช่น สายโทรศัพท์ที่เราใช้กันอยู่ทุกวันนี้ รับส่งความถี่ได้ในช่วง 300 เฮิร์ตซ์ (Hertz) ถึง 3,000 เฮิร์ตซ์ ความถี่ที่สูงกว่าและต่ำกว่านี้จะถูกดูดกลืนหรือถูกกำจัดออกไป ทำให้ส่งไปถึงปลายทางไม่ได้

ในการรับส่งข้อมูลต่างๆ ที่ความเร็วไม่สูงนัก เราจะไม่ได้ใช้ความถี่ทั้งหมดของสายในการรับส่งข้อมูล ดังนั้นจึงมีผู้คิดขึ้นมาว่า ความถี่ที่ไม่ได้ใช้ของสายส่งที่เหลือน่าจะนำมาใช้ให้เป็นประโยชน์ได้ โดยแบ่งความถี่ของสายส่งออกเป็นสองส่วน ส่วนที่หนึ่งที่ใช้สำหรับส่งข้อมูลและส่วนที่สองใช้สำหรับรับข้อมูล เทคนิคอันนี้เรียกว่า Frequency Division เช่น ด้านส่งใช้ความถี่ 1,200 เฮิร์ตซ์ และด้านรับใช้ความถี่ 2,400 เฮิร์ตซ์ เป็นต้น เพียงเท่านี้เราก็สามารถรับส่งข้อมูลในแบบฟูลดูเพล็กซ์ผ่านสายเพียงคู่เดียวได้

ขีดจำกัดของการใช้เทคนิคแบ่งความถี่ของสายส่ง ก็คือ สายส่งจะต้องมีช่วงกว้างของความถี่ (bandwidth) มากพอที่จะแบ่งออกได้ โดยไม่รบกวนกันระหว่างความถี่รับและความถี่ส่ง ส่วนความเร็วในการส่งข้อมูลของแต่ละด้านสูงสุดนั้นขึ้นอยู่กับเทคนิคการเข้ารหัสที่ใช้ในการส่งเป็นหลัก ข้อจำกัดอีกอันหนึ่งของการแบ่งความถี่ ก็คือ จะนำมาใช้กับการรับส่งข้อมูลในแบบดิจิตอล (digital) ของคอมพิวเตอร์โดยตรงไม่ได้ เนื่องจากสัญญาณข้อมูลคอมพิวเตอร์ดิจิตอลเป็นรูปคลื่นแบบสี่เหลี่ยม ซึ่งประกอบด้วยความถี่หลายๆ ความถี่รวมกันขึ้นมา เราจึงแบ่งความถี่ของสายส่งเพื่อส่งรูปคลื่นสี่เหลี่ยมจากสัญญาณของคอมพิวเตอร์ไม่ได้ ต้องส่งโดยวิธีทางอ้อม คือ แปลงสัญญาณข้อมูลของคอมพิวเตอร์ ที่เป็นรูปสี่เหลี่ยม ให้กลายเป็นสัญญาณต่อเนื่อง ในแบบอนาล็อก (analog) เสียก่อน ถึงจะรับส่งข้อมูลผ่านสายคู่เดียวในแบบฟูลดูเพล็กซ์ได้ เทคนิคนี้เป็นเทคนิคที่โมเด็มใช้รับส่งข้อมูลผ่านสายโทรศัพท์ซึ่งมีสองสาย และส่งข้อมูลแบบฟูลดูเพล็กซ์ ที่เราใช้กันอยู่ทุกวันนี้นั่นเอง

4.6 พารามิเตอร์ทางการสื่อสาร

จากการที่โมเด็มต่อกับคอมพิวเตอร์ผ่านทาง RS-232-C และส่งข้อมูลไปให้เครื่องคอมพิวเตอร์อีกเครื่องหนึ่ง ดังนั้นองค์ประกอบในการรับส่งข้อมูล (communications parameter)

จึงต้องตรงกันตลอดเส้นทางที่รับส่ง องค์ประกอบดังกล่าว คือ ความเร็ว (speed) จำนวนบิทของข้อมูล (data bit) จำนวนบิทเริ่ม (start bit) การตรวจสอบพาริตี้บิท (parity bit) และเลือกใช้ echo ในการรับส่ง

ความเร็ว คือ อัตราการรับส่งข้อมูลเป็นจำนวนบิทต่อวินาที เราต้องใช้ความเร็วเดียวกันตลอดเส้นทางการรับส่งข้อมูล เช่นเดียวกันองค์ประกอบอื่นๆ ทั้งเครื่องคอมพิวเตอร์ที่ส่งข้อมูลออกไปโมเด็มที่ด้านส่ง โมเด็มที่ด้านรับและคอมพิวเตอร์เครื่องรับจะต้องมีความเร็วในการรับส่งข้อมูลเท่ากัน ความเร็วที่ใช้กันทั่วไปมีตั้งแต่ 300 บิทต่อวินาที 1200, 2400, 4800 ไปจนถึง 9600 บิทต่อวินาที

สำหรับจำนวนบิทของข้อมูลนั้น (data bit) คือ ในหนึ่งตัวอักษรจะใช้จำนวนข้อมูลกี่บิทในการส่งปกติจะเลือกได้สองแบบ คือ 7 บิทต่อหนึ่งตัวอักษร หรือ 8 บิทต่อหนึ่งตัวอักษร การรับส่งข้อมูลภาษาไทยเราจำเป็นต้องใช้แบบ 8 บิทต่อตัวอักษรเท่านั้น เนื่องจากภาษาไทยเราใช้รหัสครบทั้ง 8 บิท ถ้าหากส่งไปในแบบ 7 บิทต่อหนึ่งตัวอักษร รหัสภาษาไทยจะถูกตัดบิทที่ 8 ทิ้งไปกลายเป็นตัวภาษาอังกฤษซึ่งใช้งานไม่ได้ จำนวนบิทเริ่มและบิทจบนั้นกำหนดไว้ให้ตัวรับและตัวส่งแยกออกว่าข้อมูลจะเริ่มต้นเมื่อใดและจบลงเมื่อใด

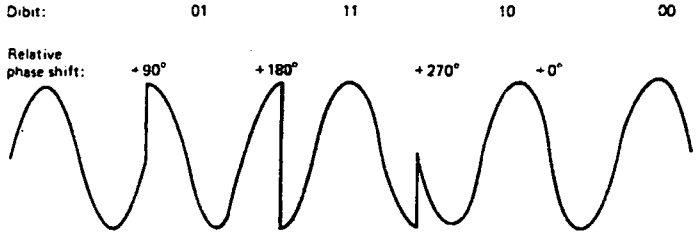
บิทเริ่มต้น (start bit) มักจะใช้หนึ่งบิทเสมอ ส่วนบิทจบข้อมูล (stop bit) จะมีสองแบบ คือ หนึ่งบิทหรือสองบิท การใช้งานทั่วไปมักจะใช้หนึ่งบิทจบเช่นกัน

การตรวจสอบพาริตี้ (parity bit) เป็นการตรวจสอบความถูกต้องของการรับส่งข้อมูลที่ส่งมามีจำนวนข้อมูลที่เป็น "1" เป็นจำนวนคู่ (even) หรือจำนวนคี่ (odd) หรือไม่ต้องตรวจสอบ (none) เช่น ถ้าส่งข้อมูล 11000001 ไปให้ผู้รับ และมีการตรวจสอบพาริตี้แบบจำนวนคู่ พาริตี้บิทในกรณีนี้จะมีค่าเป็น "1" เพื่อให้จำนวน "1" ทั้งหมด มีจำนวนเป็นเลขคู่ (even) ถ้ากำหนดการตรวจสอบพาริตี้เป็นจำนวนคี่ พาริตี้จะเป็น "0" เพื่อให้จำนวน "1" ของข้อมูลทั้งหมดเป็นจำนวนคี่ (odd) และถ้าไม่มีการตรวจสอบเลย เครื่องส่งก็จะส่งบิทจบ (stop bit) ปิดท้ายข้อมูลทันที ไม่มีการส่งพาริตี้ไปให้ผู้รับส่วนมากถ้ารับส่งข้อมูลแบบ 7 บิท เรามักจะตั้งการตรวจสอบพาริตี้เอาไว้ แต่ถ้าส่งรับข้อมูลแบบ 8 บิทก็จะไม่มีการตรวจสอบพาริตี้บิท สำหรับการเลือกใช้ echo ในการรับส่ง ก็เป็นการเลือกให้สอดคล้องกับฟูลดูเพล็กซ์หรือฮาล์ฟดูเพล็กซ์นั่นเอง ถ้าเป็นการรับส่งแบบฟูลดูเพล็กซ์ก็ต้องเลือกใช้ echo off และถ้าเป็นแบบฮาล์ฟดูเพล็กซ์เราเลือกใช้ echo on เมื่อองค์ประกอบทั้งหมดนี้ตรงกันการรับส่งข้อมูลก็จะทำได้ถูกต้อง ถ้าส่วนใดส่วนหนึ่งไม่ว่าจะเป็นคอมพิวเตอร์ด้านส่ง โมเด็มด้านส่ง โมเด็มด้านรับ หรือ คอมพิวเตอร์ด้านรับ เพียงส่วนเดียวใช้องค์ประกอบในการรับส่งข้อมูลผิดไป การรับส่งข้อมูลจะผิดพลาดได้ ทั้งผู้รับและผู้ส่งจึงต้องตกลงกันให้แน่นอนว่าจะใช้องค์ประกอบในการรับส่งข้อมูลแต่ละตัวอย่างไร

4.7 Bit Rate กับ Baud Rate

ความเร็วในการรับส่งข้อมูลของโมเด็มนั้นกำหนดเป็นบิทต่อวินาที (bit per second : bps) หรือ bit rate ซึ่งแตกต่างจากอัตราการเปลี่ยนแปลงของสัญญาณไฟฟ้าในสายส่งหรือที่เรียกกันว่า baud rate

ในสมัยก่อนการรับส่งข้อมูลใช้เทคนิคการผสมสัญญาณแบบง่าย ๆ เช่น การเปลี่ยนแปลงความถี่ตามข้อมูล "0" และ "1" ที่ได้รับ อัตราการส่งข้อมูล และอัตราการเปลี่ยนแปลงของสัญญาณในสายส่งมีค่าเท่ากัน เราจึงถือว่าอัตราการเปลี่ยนแปลงของสัญญาณในสายส่ง (baud rate) คือ อัตราการส่งข้อมูลนั่นเอง ต่อมาเทคนิคการผสมสัญญาณซับซ้อนมากขึ้น ทำให้เราสามารถส่งข้อมูลด้วยความเร็วสูง โดยอัตราการเปลี่ยนแปลงของสัญญาณในสายยังคงเท่าเดิม ดังนั้นเมื่อเราพูดว่าโมเด็มรับส่งข้อมูลที่ความเร็ว 1,200 baud เราจะไม่ทราบเลยว่าโมเด็มนั้นรับส่งข้อมูลได้กี่บิทต่อวินาที เนื่องจากว่าถ้าโมเด็มผสมสัญญาณ 1 บิทต่อหนึ่งลูกคลื่น โมเด็มนั้นจะรับส่งข้อมูลที่ความเร็ว 1,200 บิทต่อวินาที หรือถ้าโมเด็มผสมสัญญาณ 2 บิทต่อหนึ่ง ลูกคลื่นที่เปลี่ยนแปลงในสายส่ง โมเด็มจะรับส่งข้อมูลได้เร็วถึง 2,400 บิทต่อวินาที โดยยังคงมีอัตราการเปลี่ยนแปลงในสายส่งเท่ากับ 1,200 baud เท่าเดิม เพราะฉะนั้น เราจึงเลิกใช้คำว่า baud rate สำหรับบอกความเร็วการรับส่งข้อมูลของโมเด็มมาใช้คำว่า bit rate หรืออัตราการส่งข้อมูลเป็นบิทต่อวินาทีแทน ซึ่งสื่อความหมายได้เข้าใจตรงกันมากกว่า

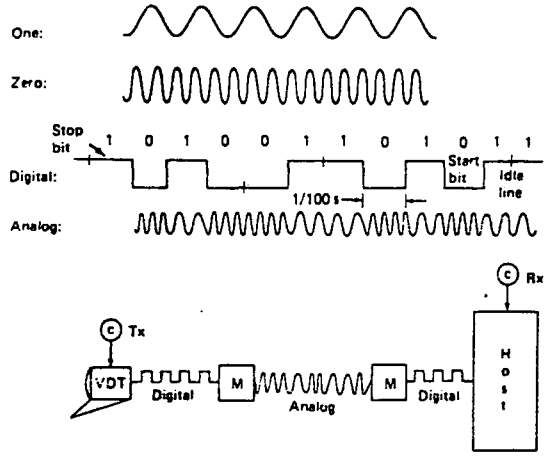


รูปที่ 4.5 การผสมสัญญาณ 2 บิทต่อหนึ่งลูกคลื่น ทำให้ Bit Rate มีค่ามากกว่า Baud Rate

4.8 การผสมสัญญาณแบบ FSK และ PSK

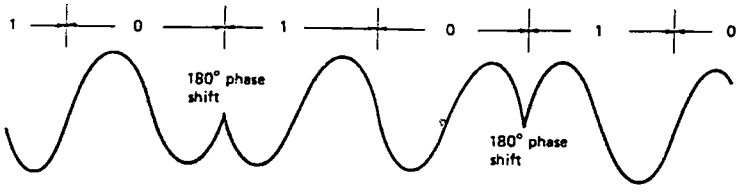
นอกจากมาตรฐานจะกำหนดความเร็วในการรับส่งข้อมูลให้ตรงกันแล้ว ยังกำหนดความถี่ของคลื่นพาหะที่ใช้ว่าผู้ส่งและผู้รับใช้ความถี่เท่าไร รวมทั้งมาตรฐานของการผสมสัญญาณอีกด้วย มาตรฐานการผสมสัญญาณที่ใช้กันมากในปัจจุบัน คือ Frequency Shift Keying (FSK), Phase shift (PSK) และ Quadrature Amplitude Modulation (QAM)

Frequency Shift Keying นั้นจะใช้ในโมเด็มความเร็วต่ำ โดยแทน "0" และ "1" ด้วยความถี่ต่างกัน ผู้ส่งจะใช้ความถี่สองความถี่แทน "0" และ "1" ของด้านส่ง ส่วนผู้รับก็ใช้ความถี่อีกสองความถี่แทน "0" และ "1" ของด้านรับเช่นกัน ทั้งหมดจึงใช้ความถี่รวม 4 ความถี่ การผสมสัญญาณแบบ FSK มักใช้กับโมเด็มความเร็วประมาณ 300 ถึง 600 บิทต่อวินาที รวมทั้งใช้กับโมเด็มแบบ Acoustic coupler ด้วย ความเร็วสูงสุดของโมเด็มที่ใช้เทคนิคการผสมสัญญาณแบบนี้อยู่ที่ 1,200 บิทต่อวินาที แต่ตามปกติแล้วโมเด็มในท้องตลาดสำหรับเครื่องคอมพิวเตอร์ส่วนบุคคลที่ใช้การผสมสัญญาณแบบ FSK จะรับส่งข้อมูลด้วยความเร็ว 300 บิทต่อวินาที การผสมสัญญาณแบบ FSK นี้อัตราการส่งข้อมูล (bit rate) จะเท่ากับ baud rate เสมอ ดังแสดงในรูปที่ 4.6



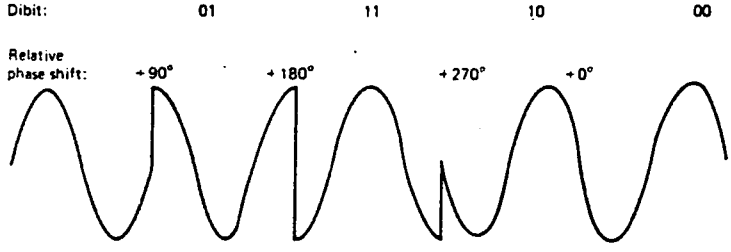
รูปที่ 4.6 แสดงการส่งข้อมูลโดยใช้ FSK

สำหรับการผสมสัญญาณแบบ Phase Shift Keying (PSK) นั้น ใช้หลักการที่ว่าแทนข้อมูล "0" และ "1" ด้วยการเปลี่ยนแปลงมุมของช่วงสัญญาณในสายส่ง (phase) เช่น โดยเราอาจกำหนดว่า "0" แทนด้วยมุมต่อเนื่องกันไป และ "1" แทนด้วยมุมที่เปลี่ยนไปจากเดิม 180 องศา ถ้าเราส่งข้อมูล 01010 รูปร่างคลื่นในสายส่งจะเป็นดังในรูปที่ 4.7



รูปที่ 4.7 การใช้ PSK ผสมสัญญาณแบบหนึ่งบิท

จากหลักการที่เราเปลี่ยนเฟส (phase) ในสัญญาณส่งนี้ ด้วยมุมทั้ง 360 องศาของคลื่นพาหะ ทำให้เราสามารถแบ่งการเปลี่ยนแปลงของเฟสให้เล็กลงไปได้อีก เช่น เปลี่ยนเฟสทีละ 90 องศา ซึ่งในหนึ่งลูกคลื่นจะเปลี่ยนได้ถึง 4 สภาวะ และกำหนดให้แต่ละสภาวะแทนด้วยข้อมูลสองบิทได้ เราสามารถกำหนด 00, 01, 10 และ 11 ให้มีการเปลี่ยนเฟสได้ครั้งละ 90 องศา เมื่อเราส่งข้อมูล เช่น 01 11 10 00 รูปร่างของคลื่นในสายส่งจะเป็นดังรูปที่ 4.8



รูปที่ 4.8 การใช้ PSK ผสมสัญญาณแบบ 2 บิท

ถ้าคลื่นพาหะมีความถี่ 600 เฮิร์ตซ์ข้อมูลที่ส่งออกไปจะมีค่าเท่ากับ 1,200 บิทต่อวินาที เนื่องจากการเปลี่ยนแปลงเฟสหนึ่งครั้งเท่ากับข้อมูลสองบิท

เราจะเห็นว่าการผสมสัญญาณแบบ PSK นี้อัตราการส่งข้อมูล (bit rate) จะมีค่าสูงกว่าอัตราการเปลี่ยนแปลงของสัญญาณในสายส่ง (baud rate) ได้ การผสมสัญญาณที่เราใช้กันมาก คือ ใช้ข้อมูล 2 บิท 3 บิท และ 4 บิท ในการเปลี่ยนเฟส โดยแบ่งมุมของคลื่นพาหะให้มีการเปลี่ยนแปลงเท่ากับ 90 องศา 45 องศา และ 22.5 องศา ตามลำดับ ความเร็วสูงสุดที่ใช้ PSK คือ 9,600 บิทต่อวินาที ซึ่งมีการเปลี่ยนแปลงของมุมครั้งละ 22.5 องศา การเปลี่ยนแปลงมุม 22.5 องศานี้ นับว่าน้อยมาก ทำให้ข้อมูลมักผิดพลาดอยู่เสมอ ส่วนมากจึงใช้งาน PSK ที่ความเร็ว 4,800 บิทต่อวินาทีแทน และที่ความเร็วสูงกว่านี้เราจะใช้เทคนิคการผสมสัญญาณที่ซับซ้อนขึ้นไปอีก

Quadrature Amplitude Modulation (QAM) เป็นการผสมสัญญาณที่ใช้ทั้งการเปลี่ยนเฟสและขนาดของสัญญาณควบคู่กันไปสำหรับใช้กับโมเด็มความเร็วสูง ซึ่งถ้าใช้การเปลี่ยนเฟสเพียงอย่างเดียวมุมที่เปลี่ยนแปลงจะมีค่าน้อยเกินไปทำให้เกิดข้อผิดพลาดได้ง่าย ถ้าเราใช้การเปลี่ยนเฟสและขนาดของสัญญาณ ค่าของมุมก็จะอยู่ห่างกันมากขึ้น ปกติที่ใช้กันอยู่จะมีเฟสต่างกัน 8 เฟสและขนาดของสัญญาณต่างกัน 4 ระดับ ใช้แทนข้อมูล 16 สถานะ ซึ่งในหนึ่งลูกคลื่นจะสามารถส่งข้อมูลได้คราวละ 4 บิท การผสมสัญญาณของ QAM บางแบบจะใช้เฟสต่างไปจากนี้ เช่น ใช้เฟสต่างกัน 12 เฟส และขนาดของสัญญาณ 3 ระดับ หรืออาจใช้เฟสต่างกัน 8 เฟสและขนาดของสัญญาณต่างกัน 2 ระดับก็ได้ ขึ้นอยู่กับการออกแบบ แต่ว่าโมเด็มในมาตรฐานเดียวกันนั้น จะต้องใช้การแบ่งเฟสและระดับสัญญาณเท่าเดิมเสมอ ความเร็วในการรับส่งข้อมูลของ QAM อยู่ที่ 9,600 บิทต่อวินาที โดยใช้ความถี่พาหะ 2,400 เฮิร์ตซ์ และในหนึ่งลูกคลื่นแทนข้อมูลได้คราวละ 4 บิท เทคนิคการผสมสัญญาณแบบ QAM นี้ อาจถูกนำไปใช้กับโมเด็มความเร็วปานกลางก็ได้ เช่น ที่ความเร็ว 2,400 บิทต่อวินาที เป็นต้น

นอกจากนี้ยังมีการผสมสัญญาณสำหรับโมเด็มความเร็วสูงอีกหลายแบบ แต่ส่วนมากยังไม่ได้รับการยอมรับให้เป็นมาตรฐานในปัจจุบัน การผสมสัญญาณในแบบ QAM มีความซับซ้อนมาก ดังนั้นวงจรทางด้านฮาร์ดแวร์ ไม่ว่าจะเป็นวงจรด้านส่ง วงจรด้านรับ และการแยกสัญญาณต่าง ๆ ยุ่งยากกว่าที่ใช้กับวงจรเปลี่ยนเฟสอย่างเดียว ราคาของโมเด็มที่ใช้เทคนิคแบบ QAM จึงสูงกว่าโมเด็มที่ใช้เทคนิค PSK อยู่มาก

4.9 มาตรฐานของโมเด็ม

โมเด็มจำเป็นต้องมีมาตรฐานเช่นเดียวกับอุปกรณ์อื่น ๆ เพื่อให้ผู้ผลิตแต่ละบริษัทผลิตโมเด็มออกมาใช้รับส่งข้อมูลระหว่างกันได้ มาตรฐานของโมเด็มจะแบ่งออกเป็นสองส่วน คือ มาตรฐานในส่วนของฮาร์ดแวร์ที่โมเด็มใช้ และอีกส่วนหนึ่งคือ มาตรฐานในส่วนของซอฟต์แวร์ที่ควบคุมการทำงานของโมเด็มหรือคำสั่งของโมเด็ม ในที่นี้จะกล่าวถึงเฉพาะมาตรฐานทางด้านฮาร์ดแวร์ของโมเด็ม โดยมาตรฐานนี้กำหนดขึ้นจากองค์การมาตรฐานสื่อสารสากล หรือ CCITT (International Telephone and Telegraph Consultation Committee) ซึ่งมาตรฐานนี้จะครอบคลุมเกี่ยวกับความเร็วในการรับส่งข้อมูล ความถี่ที่ใช้และเทคนิคการผสมสัญญาณในสาย ฯลฯ CCITT มีบทบาทมากในการกำหนดมาตรฐานทางการสื่อสาร เนื่องจากมีสมาชิกเกือบทุกประเทศทั่วโลก ผู้ผลิตรายใหญ่จึงปฏิบัติตามมาตรฐานของ CCITT ทำให้โมเด็มแต่ละยี่ห้อสามารถรับส่งข้อมูลกันได้ นอกจากมาตรฐานของ CCITT แล้ว มาตรฐานที่ใช้กำหนดลักษณะของโมเด็มยังมีมาตรฐานของเบลล์ (Bell Standard) ซึ่งในปัจจุบันได้รับความนิยมน้อยกว่า

4.10 มาตรฐานของโมเด็มตาม CCITT-V-Series

มาตรฐานของโมเด็มที่เราใช้อยู่ทุกวันนี้เป็นไปตามที่องค์การมาตรฐานสื่อสารสากลหรือ CCITT เป็นผู้กำหนดขึ้น โดยมีชื่อเรียกแต่ละมาตรฐานของโมเด็มขึ้นต้นด้วยตัวอักษร "V" และตามด้วยตัวเลข เราจึงเรียกมาตรฐานเหล่านี้อีกชื่อหนึ่งว่า V-Series นอกจากมาตรฐานของโมเด็มแล้ว CCITT ยังเป็นผู้กำหนดมาตรฐานทางการสื่อสารอื่น ๆ อีก เช่น มาตรฐานของการสื่อสารผ่านดาวเทียม มาตรฐานของโทรสาร (facsimile) มาตรฐานการสื่อสารข้อมูลต่างๆ ทั้งในแบบดิจิตอลและอนาล็อก รวมถึงมาตรฐานเกี่ยวกับระบบโทรศัพท์อีกด้วย มาตรฐานที่ CCITT เป็นผู้กำหนดได้รับการยอมรับกันทั่วโลก การติดต่อสื่อสารระหว่างประเทศจึงดำเนินไปได้อย่างไม่มีปัญหา เนื่องจากทุก ๆ คนต่างก็ทำตามมาตรฐานเดียวกัน

มาตรฐานที่ขึ้นต้นด้วยอักษร V นี้ ไม่ใช่มาตรฐานของโมเด็มทั้งหมด บางมาตรฐานอาจหมายถึงการเชื่อมต่อแบบอื่น ๆ ก็ได้ เช่น V.24 เป็นมาตรฐานการรับส่งข้อมูลแบบอนุกรมเทียบได้กับ RS-232-C นั่นเอง และ V.35 หมายถึง การรับส่งข้อมูลแบบอนุกรมความเร็วสูง เป็นต้น ในที่นี้จะกล่าวถึงมาตรฐานของโมเด็มแบบต่างๆ ที่ใช้กันมากตาม CCITT V-Series ตั้งแต่ความเร็วต่ำไปจนถึงความเร็วสูง ดังนี้

- V.21 เป็นมาตรฐานของโมเด็มความเร็ว 300 บิทต่อวินาที ใช้เทคนิคการผสมสัญญาณแบบ FSK (Frequency Shift Keying) รับส่งข้อมูลได้ในแบบฟูลดูเพล็กซ์ เป็นโมเด็มที่ใช้กับสายโทรศัพท์ ปัจจุบันนี้มีใช้กันน้อย เนื่องจากความเร็วในการรับส่งข้อมูลต่ำ

- V.22 รับส่งข้อมูลด้วยความเร็ว 1,200 บิทต่อวินาที หรือลดความเร็วลงมาที่ 600 บิทต่อวินาทีได้ การผสมสัญญาณใช้เทคนิคของ PSK (Phasse Shift Keying) รับส่งข้อมูลในแบบฟูลดูเพล็กซ์ ใช้กับสายโทรศัพท์หรือสายตรงได้ ขึ้นอยู่กับโมเด็มว่าถูกออกแบบมาให้ต่อใช้กับสายตรงหรือไม่ จัดเป็นโมเด็มความเร็วปานกลางที่ได้รับความนิยมอยู่ในปัจจุบัน

- V.22 bis รับส่งข้อมูลด้วยความเร็ว 2,400 บิทต่อวินาที หรือลดความเร็วลงมาที่ 1,200 บิทต่อวินาทีได้ การผสมสัญญาณใช้เทคนิคของโมเด็มความเร็วสูง คือ QAM รับส่งข้อมูลแบบฟูลดูเพล็กซ์ ใช้กับสายโทรศัพท์หรือสายตรงได้ V.22 bis เป็นมาตรฐานของโมเด็มความเร็วปานกลางที่จะเข้ามาแทนที่ V.22 ซึ่งมาตรฐาน V.22 bis นี้กำลังได้รับความนิยมมากเนื่องจากความเร็วสูงถึง 2,400 ต่อวินาที และราคาของโมเด็มไม่แพงจนเกินไป

- V.23 เป็นมาตรฐานที่คล้ายกับมาตรฐาน V.22 แต่รับส่งข้อมูลในแบบฮาล์ฟดูเพล็กซ์ คือ มีความเร็ว 1,200 บิทต่อวินาทีหรือลดความเร็วลงมาที่ 600 บิทต่อวินาทีได้ ใช้เทคนิคการผสมสัญญาณแบบ FSK ต่อใช้กับสายโทรศัพท์ก็ได้ มาตรฐาน V.23 นี้ เราไม่ค่อยได้ใช้งานเท่าไรนัก เพราะว่าประสิทธิภาพการของการรับส่งข้อมูลต่ำ และ เป็นการติดต่อแบบฮาล์ฟดูเพล็กซ์

จึงสู้มาตรฐานแบบ V.22 หรือ V.22 bis ไม่ได้

- V.26 เป็นมาตรฐานของโมเด็มสายตรง แบบใช้สาย 4 เส้น (4 Wires) รับส่งข้อมูลในแบบฟูลดูเพล็กซ์ ใช้เทคนิคการผสมสัญญาณชนิด PSK มีความเร็วในการรับส่งข้อมูล 2,400 บิทต่อวินาที จะนำมาใช้กับสายโทรศัพท์ไม่ได้ มาตรฐาน V.26 เราจึงไม่ค่อยได้พบเห็นกันนัก ปัจจุบันก็มีใช้น้อยเนื่องจากความเร็วต่ำเกินไปสำหรับสายตรง ส่วนมากจะเลือกใช้มาตรฐานอื่นที่ความเร็วสูงกว่านี้

_ V.26 bis เป็นมาตราฐานเหมือนกับ V.26 แต่สำหรับใช้กับสายโทรศัพท์แทน มีความเร็วในการรับส่งข้อมูลที่ 2,400 บิทต่อวินาที หรือลดความเร็วลงมาที่ 1,200 บิทต่อวินาที ได้การรับส่งข้อมูลเป็นแบบฮาล์ฟดูเพล็กซ์ ใช้เทคนิคการผสมสัญญาณแบบ PSK มาตราฐานนี้จึงสู้ V.22 bis ไม่ได้ โดยทั่วไปเราจะใช้ V.22 bis ที่ความเร็ว 2,400 บิทต่อวินาที และเป็นการติดต่อแบบฟูลดูเพล็กซ์มากกว่า

_ V.27 เป็นมาตราฐานสำหรับโมเด็มความเร็ว 4,800 บิทต่อวินาที ที่ใช้กับสายตรงเท่านั้น เทคนิคของการผสมสัญญาณเป็นแบบ PSK รับส่งข้อมูลในแบบฟูลดูเพล็กซ์ได้ ความเร็ว 4,800 บิทต่อวินาทีนี้ ถือว่าเป็นความเร็วการรับส่งข้อมูลสูงที่สุดของเทคนิคการผสมสัญญาณ PSK

_ V.27 bis คล้ายกับมาตราฐานแบบ V.27 แต่ว่าการรับส่งข้อมูลที่ 4,800 บิทต่อวินาที หรือลดความเร็วลงมาที่ 2,400 บิทต่อวินาทีได้ ใช้สำหรับสายตรงแบบ 4 Wires เท่านั้น การผสมสัญญาณก็เป็นแบบ PSK สามารถรับส่งข้อได้ทั้งในแบบฟูลดูเพล็กซ์และฮาล์ฟดูเพล็กซ์

- V.27 ter เป็นมาตราฐานโมเด็มความเร็ว 4,800 บิทต่อวินาที หรือลดความเร็วลงมาที่ 2,400 บิทต่อวินาที สำหรับใช้กับสายโทรศัพท์ การรับส่งข้อมูลเป็นแบบฮาล์ฟดูเพล็กซ์เท่านั้น เทคนิคการผสมสัญญาณชนิด PSK มาตราฐาน V.27 ter คล้ายกับ V.27 bis เพียงแต่ใช้กับสายโทรศัพท์แทนที่จะเป็นสายตรง

- V.29 จัดเป็นมาตรฐานของโมเด็มความเร็วสูงใช้กับสายตรงแบบ 4 wires เท่านั้น การรับส่งข้อมูลใช้ได้ทั้งฟูลดูเพล็กซ์และฮาล์ฟดูเพล็กซ์ สามารถรับส่งข้อมูลได้ตั้งแต่ 9,600 บิทต่อวินาที หรือลดความเร็วลงมาที่ 7,200 บิทต่อวินาที และ 4,800 บิทต่อวินาทีได้ ที่ความเร็ว 9,600 บิทต่อวินาที จะใช้เทคนิคการผสมสัญญาณแบบ QAM ส่วนที่ความเร็ว 7,200 และ 4,800 บิทต่อวินาที ใช้การผสมสัญญาณแบบ PSK มาตรฐาน V.29 นี้ มีใช้กันมากสำหรับการรับส่งข้อมูลผ่านสายตรงระหว่างคอมพิวเตอร์กับคอมพิวเตอร์

- V.32 เป็นมาตรฐานโมเด็มความเร็วสูงสำหรับใช้กับสายโทรศัพท์ สามารถรับส่งข้อมูลได้ที่ความเร็ว 9,600 บิทต่อวินาที ในแบบฟูลดูเพล็กซ์ หรือลดความเร็วลงมาที่ 4,800 บิทต่อวินาทีได้ มาตรฐาน V.32 นี้ยังใช้งานกับสายตรงแบบสาย 2 เส้น (2 wires) ได้อีกด้วย

เทคนิคการผสมสัญญาณเป็นแบบ QAM ทั้งที่ความเร็ว 9,600 และ 4,800 บิทต่อวินาที การรับส่งข้อมูลความเร็วสูงผ่านสาย 2 เส้นของ V.32 ใช้เทคนิค Echo Cancellation แทนที่จะใช้การแบ่งความถี่อย่างในโมเด็มความเร็วต่ำ

- V.32 bis คล้ายกับ V.32 แต่จะรับส่งข้อมูลด้วยความเร็ว 14,400 บิทต่อวินาทีในแบบฟูลดูเพล็กซ์

- V.33 เป็นมาตรฐานโมเด็มความเร็วสูง สำหรับใช้กับสายตรงเท่านั้น สามารถรับส่งข้อมูลได้ที่ความเร็ว 14,400 บิทต่อวินาที หรือลดความเร็วลงมาที่ 12,000 บิทต่อวินาทีได้ มาตรฐาน V.33 นี้ ใช้กับสายตรงแบบ 2 wires โดยส่งข้อมูลแบบฮาล์ฟดูเพล็กซ์หรือใช้กับสายตรงแบบ 4 wires โดยส่งข้อมูลแบบฟูลดูเพล็กซ์

ตารางที่ 4.1 สรุปมาตรฐานโมเด็ม V-Series ของ CCITT

มาตรฐานโมเด็มของ CCITT

Series Number	Line Speed	Channel Separation	FDX or HDX	Modulation Technique	Switch Lines	Leased Lines
V.21	300	FD	FDX	FSK	Yes	0
V.22	1200	FD	FDX	TSK	Yes	PP2W
V.22	600	FD	FDX	PSK	Yes	PP2W
V.22 bis	2400	FD	FDX	QAM	Yes	PP2W
V.22 bis	1200	FD	FDX	QAM	Yes	PP2W
V.23	600	NA	HDX	FMK	Yes	0
V.23	1200	NA	HDX	FMK	Yes	0
V.26	2400	4-Wire	FDX	PSK	No	PP MP4W
V.26 bis	2400	NA	HDX	PSK	Yes	No
V.26 bis	1200	NA	HDX	PSK	Yes	No
V.26 ter	2400	EC	Either	PSK	Yes	PP 2W
V.26 ter	1200	EC	Either	PSK	Yes	PP 2W
V.27	4800	ND	Either	PSK	No	Yes
V.27 bis	4800	4-Wire	Either	PSK	No	2W 4W
V.27 bis	2400	4-Wire	Either	PSK	No	2W 4W
V.27 ter	4800	None	HDX	PSK	Yes	No
V.27 ter	2400	None	HDX	PSK	Yes	No
V.29	9600	4-Wire	Either	QAM	No	PP 4W
V.29	7200	4-Wire	Either	PSK	No	PP 4W
V.29	4800	4-Wire	Either	PSK	No	PP 4W
V.32	9600	EC	FDX	QAM	Yes	PP 2W
V.32	9600	EC	FDX	TCM	Yes	PP 2W
V.32	4800	EC	FDX	QAM	Yes	PP 2W
V.32	14,400	4-Wire	FDX	TCM	FS	PP 4W

Also in preparation are:

V.34 a 14,400/19,200 half duplex asymmetrical duplex highspeed Modem for FAX applications. Unlikely to appear before 1992.

V.32 bis a 14,400 full duplex PSTN modem; under study but not being formally discussed.

ND = not defined NA = not applicable EC = echo cancellation
FD = frequency division FDX = full duplex HDX = half duplex
PP = point to point MP = multipoint FS = for further study

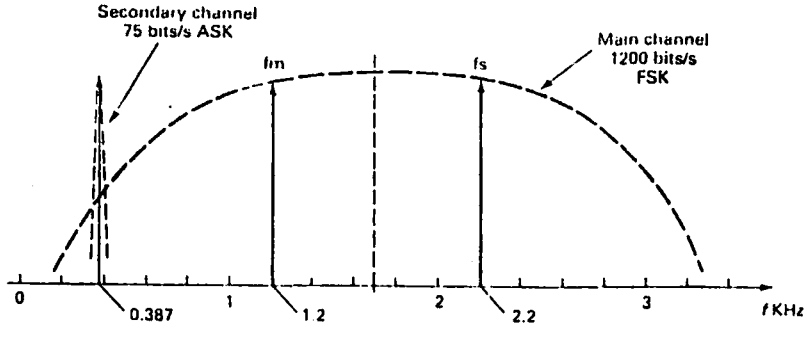
มาตรฐานของโมเด็ม V-Series ที่กล่าวถึงทั้งหมดนี้ เป็นมาตรฐานที่เราพบเห็นได้ทั่วไป ซึ่งยังมีบางมาตรฐานที่ไม่ได้กล่าวถึงในที่นี้เนื่องจากใช้งานพิเศษเฉพาะงานเท่านั้น ส่วนมาตรฐานของโมเด็มตามแบบสหรัฐอเมริกา หรือที่เราเรียกว่า Bell Standard ปัจจุบันค่อยๆ ลดความนิยมลง เนื่องจากประเทศต่างๆ ใช้ตามมาตรฐานของ CCITT เป็นหลัก และในประเทศไทยเราก็ใช้ตามมาตรฐานของ CCITT เช่นกัน

4.11 โมเด็มมาตรฐาน Bell 202

ในการใช้งานบางอย่าง เช่น การที่สถานีปลายทาง (Terminal) ที่อยู่ไกลมาก ๆ สามารถเข้าถึงฐานข้อมูลของคอมพิวเตอร์ตัวแม่ (Host Computer) ซึ่งต้องการช่องการสื่อสารที่มีความเร็วสูงจากตัวแม่ไปยังสถานีปลายทาง แต่ช่องการสื่อสารในทางกลับกันจากสถานีปลายทางไปยังคอมพิวเตอร์ตัวแม่ ซึ่งรับการกดคีย์บอร์ดเพียงนาน ๆ ครั้งนั้นสามารถใช้ความเร็วต่ำได้

โมเด็มตามมาตรฐาน Bell 202 จะมีคุณสมบัติดังกล่าว ซึ่งเป็นการสื่อสารแบบฟูลดูเพล็กซ์อย่างไม่สมมาตร (asymmetrical full-duplex communication) จะมีอัตราเร็วในทิศทางส่งไปและส่งกลับจะต่างกันแต่สามารถใช้ได้พร้อมๆกัน มักจะเรียกช่องการสื่อสารที่มีความเร็วสูงกว่าจากตัวแม่ไปยังสถานีปลายทางว่าช่องการสื่อสารหลัก (primary channel) และเรียกว่า ช่องการสื่อสารรอง (secondary channel) สำหรับช่องการสื่อสารที่มีความเร็วต่ำกว่า จากสถานีปลายทางไปยังตัวแม่

ระบบ Bell 202 จะมีความเร็ว 1,200 บิทต่อวินาที โดยใช้การผสมสัญญาณแบบ FSK สำหรับช่องการสื่อสารหลัก และมีความเร็ว 75 บิทต่อวินาที โดยใช้การผสมสัญญาณ แบบ ASK สำหรับช่องการสื่อสารรอง รูปที่ 4.9 แสดงย่านความถี่โทรศัพท์ที่ใช้สำหรับช่องการสื่อสารหลัก ความถี่ของสภาวะ mark (ลอจิก "1") คือ 1,200 เฮิร์ตซ์ และความถี่ของสภาวะ space (ลอจิก "0") คือ 2,200 เฮิร์ตซ์ สำหรับช่องการสื่อสารรองนั้นจะมีค่าแอมปลิจูดของพาหะ คือ 387 เฮิร์ตซ์

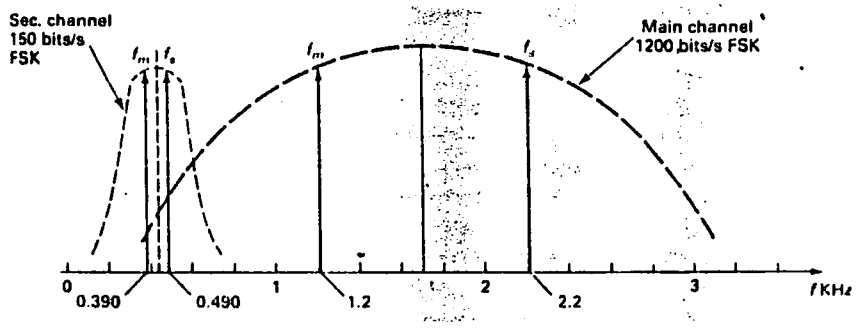


รูปที่ 4.9 แถบความถี่ใช้งานของโมเด็มมาตรฐาน Bell 202

4.12 โมเด็มมาตรฐาน Bell 202C

การใช้งานโดยทั่วไปที่เหมาะสมสำหรับ การสื่อสารแบบฟูลดูเพล็กซ์อย่างไม่สมมาตรนั้น คือ ใช้ใน videotex (การส่งภาพหน้าจอและข้อความ) สถานีปลายทางของ videotex ที่อยู่ไกลออกไปจะเรียกมายังคอมพิวเตอร์ตัวแม่ ซึ่งแฟ้มข้อมูลของข้อความและรูปภาพจะถูกส่งไปให้ยังสถานีปลายทาง แฟ้มข้อมูลจะถูกจัดอยู่ในรูปแบบเป็นหน้าจอแต่ละหน้า ซึ่งจะถูกเรียกจากผู้ใช้โดยการใส่หมายเลขหน้าโดยทางแป้นคีย์บอร์ดเล็กๆ ในกรณีนี้ช่องการสื่อสารที่มีความเร็วต่ำสามารถถูกใช้ในการถ่ายทอดคำสั่ง (สัญญาณการกดคีย์) ของผู้ใช้ไปยังคอมพิวเตอร์ตัวแม่ โดยการส่งถ่ายแฟ้มข้อมูลจะต้องใช้ช่องการสื่อสารหลักซึ่งมีความเร็วสูงกว่า

การปรับปรุงอุปกรณ์ใส่ข้อมูล (input device) ทางด้านผู้ใช้จากแป้นคีย์บอร์ดเล็ก ๆ มาเป็นคีย์บอร์ดที่สามารถรับตัวอักษรแอสกี้ได้ครบทุกตัว (full ASCII keyboard) นั้น ทำให้การติดต่อสื่อสารกันในแง่ของ videotex มีประสิทธิภาพยิ่งขึ้น แต่ต้องการช่องการสื่อสารรองที่มีอัตราเร็วสูงกว่า 75 บิทต่อวินาที ถึงจะให้ผลเป็นที่น่าพอใจ โมเด็มมาตรฐาน Bell 202C ได้ถูกพัฒนาขึ้นสำหรับความต้องการดังกล่าวซึ่งจะมีอัตราเร็ว 1,200 บิทต่อวินาที เท่าเดิมสำหรับช่องการสื่อสารหลัก แต่จะมีอัตราเร็ว 150 บิทต่อวินาที แบบ FSK สำหรับช่องการสื่อสารรองแทนที่จะเป็น 75 บิทต่อวินาที ดังในมาตรฐาน Bell 202 ธรรมดา รูปที่ 4.10 แสดงความถี่ใช้งานของ Bell 202C ความถี่ของสภาวะ mark (ลอจิก "1") และ สภาวะ space (ลอจิก "0") สำหรับช่องการสื่อสารรองที่มีความเร็ว 150 บิทต่อวินาทีนั้น คือ 390 เฮิร์ตซ์ และ 490 เฮิร์ตซ์ ตามลำดับ เนื่องจากมีการซ้อนทับกันของความถี่ด้านข้างของแต่ละช่องการสื่อสาร ดังในรูปที่ 4.10 ดังนั้น ตัวกรองความถี่ (filter) สำหรับเลือกและจำกัดย่านความถี่จึงเป็นส่วนที่สำคัญอย่างยิ่งที่ต้องมีในระบบของ Bell 202C

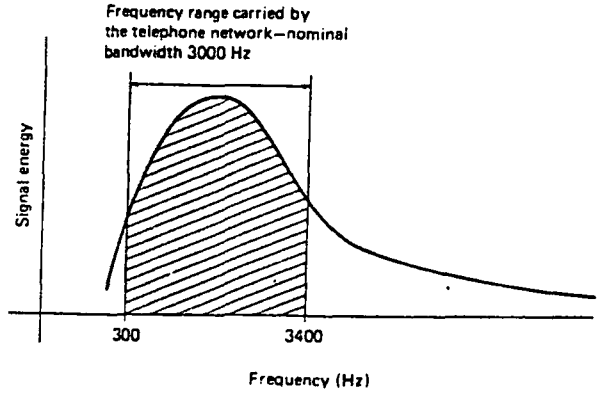


รูปที่ 4.10 แถบความถี่ใช้งานของโมเด็มมาตรฐาน Bell 202C

4.13 ขีดจำกัดของเครือข่ายโทรศัพท์

จะเห็นได้ว่าการส่งข้อมูลโดยใช้เสียงผ่านเครือข่ายโทรศัพท์นั้น เป็นวิธีที่น่าสนใจมาก เนื่องจากว่าโทรศัพท์มีอยู่ทุกหนทุกแห่ง ไม่ว่าจะเป็นที่บ้าน ที่ทำงาน หรือสถานที่ต่างๆ และเราสามารถใช้โทรศัพท์ติดต่อไปยังที่ต่างๆ ได้ทั่วโลก การรับส่งข้อมูลผ่านสายโทรศัพท์จึงทำได้ครอบคลุมพื้นที่กว้างขวางมากกว่าการใช้ระบบอย่างอื่นๆ แต่เนื่องจากระบบโทรศัพท์ไม่ได้ถูกออกแบบมาให้ใช้รับส่งข้อมูลคอมพิวเตอร์ หรือข้อมูลอื่นๆ นอกจากรับส่งเสียงพูดได้เท่านั้น ระบบโทรศัพท์จึงมีขีดจำกัดหลายอย่างที่ทำให้การรับส่งข้อมูลต้องทำด้วยความระมัดระวัง ขีดจำกัดของระบบโทรศัพท์ที่มีผลต่อการรับส่งข้อมูล ก็คือ ช่วงความถี่ของสัญญาณ (Bandwidth) ระบบกำจัดเสียงสะท้อนของโทรศัพท์ (Echo Suppressor) และเสียงรบกวนในสาย (Background Noise) ขีดจำกัดเหล่านี้ ทำให้การรับส่งข้อมูลทำได้ช้าและอาจผิดพลาดได้

ระบบการรับฟังเสียงของคนเราสามารถรับรู้เสียงความถี่ ได้ตั้งแต่ 20 เฮิร์ตซ์ จนถึง 20,000 เฮิร์ตซ์ โดยประมาณ ความถี่สูงกว่าและต่ำกว่านี้เราจะไม่ได้ยินเสียง เมื่อมีการคิดค้นระบบโทรศัพท์ขึ้นมา เทคโนโลยียังไม่สูงมากพอที่จะสร้างระบบโทรศัพท์ให้รับส่งเสียงได้เทียบเท่ากับที่หูคนเราได้ยิน การออกแบบจึงพิจารณาทางด้านเทคนิคและราคาที่พอยอมรับได้ในสมัยนั้น ผลที่ออกมา ก็คือ ระบบโทรศัพท์สามารถรับส่งสัญญาณเสียงได้ตั้งแต่ความถี่ 300 เฮิร์ตซ์ จนถึงประมาณ 3,400 เฮิร์ตซ์ ซึ่งความถี่ช่วงดังกล่าวนี้มากพอสำหรับการพูดคุยของคนทั่วไปด้วยคุณภาพของเสียงพอสมควร แต่มันไม่มากพอให้เราส่งข้อมูลคอมพิวเตอร์ได้อย่างสะดวกผ่านสายโทรศัพท์ เพราะในการรับส่งข้อมูลทั่วไปสายส่งจะมี bandwidth ยิ่งสูงเท่าใด ก็จะสามารถรับส่งข้อมูลได้เร็วมากขึ้นเท่านั้น



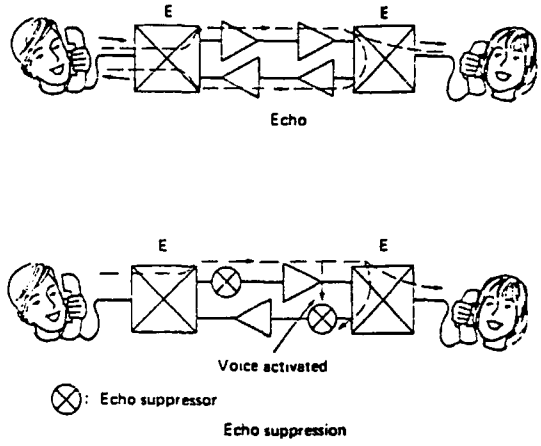
รูปที่ 4.11 สายโทรศัพท์รับส่งสัญญาณได้ในช่วงความถี่ 300 เฮิร์ตซ์ ถึง 3,400 เฮิร์ตซ์

ช่วงความถี่ 300 เฮิร์ตซ์ ถึง 3,400 เฮิร์ตซ์ ที่สายโทรศัพท์สามารถรับส่งสัญญาณได้นี้ นับว่าเป็นสายส่งที่มี bandwidth แคบมากเมื่อเรานำมาใช้รับส่งข้อมูลเพราะว่าตามธรรมดาแล้ว ความเร็วสูงสุดของการรับส่งข้อมูล จะมีค่าต่ำกว่าความถี่สูงสุดของสายส่งเสมอ ถ้าหากไม่ใช้เทคนิคพิเศษเข้าช่วย สายโทรศัพท์ก็จะรับส่งข้อมูลได้ไม่เกิน 3,000 บิทต่อวินาทีเท่านั้น ซึ่งเป็นตัวเลขที่ไม่น่าพอใจเท่าไร่ แรกเริ่มของการส่งข้อมูลผ่านสายโทรศัพท์ มีความเร็วเพียง 300 บิทต่อวินาที และพัฒนาจนรับส่งได้ 1,200 ถึง 2,400 บิทต่อวินาที ตามลำดับ เมื่อเทคโนโลยีก้าวหน้าขึ้น จากการที่ช่วงความถี่ของสัญญาณ (bandwidth) จำกัดมากในสายโทรศัพท์ในการรับส่งข้อมูลความเร็วสูงกว่า 2,400 บิทต่อวินาที ผู้ผลิตโมเด็มจำเป็นต้องใช้เทคนิคการเข้ารหัสและการผสมสัญญาณที่ซับซ้อนมากเพื่อแก้ปัญหาดังกล่าว ดังนั้นการรับส่งข้อมูลความเร็วสูงจึงถูกรบกวนได้ง่าย และมีอัตราการผิดพลาดในการรับส่งข้อมูลสูงขึ้นเป็นเงาตามตัว

วงจรสายตรงนับว่าได้เปรียบมากในข้อนี้เนื่องจากคุณภาพของสายส่งและวงจรมีคุณภาพดีกว่า ทำให้รับส่งข้อมูลความเร็วสูงได้อย่างไม่มีปัญหาด้านขีดจำกัดของช่วงความถี่เหมือนที่เกิดขึ้นกับสายโทรศัพท์

ขีดจำกัดลำดับที่สองของการส่งข้อมูลผ่านสายโทรศัพท์ ก็คือ วงจรกำจัดเสียงสะท้อน (Echo Suppressor) ของระบบโทรศัพท์ เนื่องจากการส่งสัญญาณไฟฟ้าผ่านสายส่งเป็นระยะทางไกลๆ นั้นสัญญาณที่ส่งออกไปจะมีส่วนหนึ่งสะท้อนกลับมายังผู้ส่งซึ่งเราเรียกว่าเกิด echo ขึ้น ปรากฏการณ์อันนี้เกิดขึ้นกับคลื่นทุกชนิด ไม่จำกัดว่าจะต้องเป็นสัญญาณไฟฟ้าเท่านั้น เมื่อมันเกิดขึ้นในสายโทรศัพท์ เราก็จะได้ยินเสียงตัวเองสะท้อนกลับมาในลักษณะของเสียงก้อง คล้าย ๆ กับเวลาที่เราพูดอยู่ในถ้ำนั่นเอง การเกิดเสียงก้องนี้นอกจากจะทำให้การรับฟังโทรศัพท์มีคุณภาพต่ำลงแล้วมันยังทำให้การสนทนาเป็นไปอย่างลำบากอีกด้วย

ระบบโทรศัพท์จึงใช้วงจรกำจัดเสียงสะท้อนเข้ามาช่วยแก้ปัญหาอันนี้ การทำงานของมันจะยอมให้สัญญาณจากผู้พูดเดินทางไปถึงผู้รับได้ทางเดียว สัญญาณที่สะท้อนกลับมาจะถูกกั้นเอาไว้ไม่ให้กลับมาหาผู้พูดได้ เมื่อผู้ฟังพูดสวนกลับไปวงจรกำจัดเสียงสะท้อนก็จะสลับทิศทางไปในทางตรงข้าม เพื่อให้สัญญาณจากผู้พูดถูกกั้นไม่ให้สะท้อนกลับมาหาตัวเองอีกเช่นเดียวกัน วิธีการนี้ทำให้เสียงสะท้อนถูกกำจัดออกไปได้ วงจรกำจัดเสียงสะท้อนจะรู้ว่าใครเป็นผู้พูด โดยดูจากความแรงของสัญญาณ ใช้กฎเกณฑ์ที่ว่าด้านที่พูดจะมีสัญญาณแรงกว่าด้านรับเสมอ และการสลับทิศทางของวงจรกำจัดเสียงสะท้อน ทำด้วยความเร็วสูงมากจนเราไม่รู้สึกผิดปกติเมื่อพูดและฟังตามธรรมดา ขณะที่เราสลับกันพูดโทรศัพท์ในการสนทนาทั่วไป วงจรกำจัดเสียงสะท้อนจะสลับทิศทางตลอดเวลาเพื่อป้องกันเสียงสะท้อนไม่ให้เกิดขึ้น



รูปที่ 4.12 วงจรป้องกันเสียงสะท้อนจะตัดเสียงผู้พูดไม่ให้สะท้อนกลับมา

4.14 วิธีป้องกันเสียงสะท้อนของโทรศัพท์

เมื่อเราส่งข้อมูลผ่านสายโทรศัพท์ วงจรกำจัดเสียงสะท้อนจะทำให้การรับส่งข้อมูลมีปัญหามาก เนื่องจากการสลับทิศทางของวงจรกำจัดเสียงสะท้อน จะทำให้สัญญาณขาดหายไปในช่วงนั้น ถึงแม้ว่าจะเป็นช่วงเวลาสั้นๆ แต่ก็อาจจะทำให้ข้อมูลบางส่วนหายไปได้ นอกจากนี้การสลับทิศทางของวงจรกำจัดเสียงสะท้อนยังทำให้เสียเวลาในการส่งข้อมูลอีกด้วย เพราะมันต้องใช้เวลาช่วงหนึ่งในการสลับทิศทางเสมอ สิ่งเหล่านี้มีผลมาก เมื่อเราส่งข้อมูลผ่านสายโทรศัพท์ด้วยความเร็วสูง

โมเด็มในปัจจุบันจะหยุดการทำงานของวงจรกำจัดเสียงสะท้อนได้ โดยส่งคลื่นเสียงความถี่ประมาณ 2,100 เฮิร์ตซ์ ไปยังชุมสายโทรศัพท์นานประมาณครึ่งวินาที วงจรกำจัดเสียงสะท้อนก็จะหยุดสลับทิศทาง ทำให้การรับส่งข้อมูลเป็นไปได้อย่างไม่มีปัญหา โมเด็มจะต้องหยุดการทำงานของวงจรกำจัดเสียงสะท้อนเสมอ เพราะว่า การรับส่งข้อมูลผ่านสายโทรศัพท์มักจะเป็นแบบส่งสวนทางกันได้ (ฟูลดูเพล็กซ์) โดยใช้เทคนิคแบ่งความถี่เข้าช่วย ซึ่งถ้าหากว่าวงจรกำจัดเสียงสะท้อนยังคงทำงานอยู่ข้อมูลบางส่วนอาจจะผิดพลาดหรือขาดหายไปได้ ในการรับส่งแบบฮาล์ฟดูเพล็กซ์เราก็ยังจำเป็นจะต้องหยุดการทำงานของวงจรกำจัดเสียงสะท้อนเช่นกัน เนื่องจากว่าสัญญาณของข้อมูลที่ส่งไปตามสายนั้นมักจะทำให้วงจรกำจัดเสียงสะท้อนหลงทางอยู่เสมอ มีผลทำให้ข้อมูลเดินทางไปไม่ถึงผู้รับ เพราะถูกกั้นสัญญาณเอาไว้ วงจรกำจัดเสียงสะท้อนจึงเป็นปัญหาที่โมเด็มต้องจัดการให้เรียบร้อยก่อนส่งข้อมูลผ่านสายโทรศัพท์เสมอ

สัญญาณรบกวนในสายโทรศัพท์เป็นปัญหาอีกอันหนึ่งในการรับส่งข้อมูล ปกติสัญญาณรบกวนจะมีอยู่ระดับหนึ่งตลอดเวลาในสาย ซึ่งอาจจะเกิดจากคลื่นแม่เหล็กไฟฟ้าจากที่อื่นถูกเหนี่ยวนำเข้า

มาในสาย หรือเกิดจากอุปกรณ์รับส่งของชุมสายโทรศัพท์เองก็ได้ เมื่อสัญญาณรบกวนมีขนาดไม่มากนักเทียบกับสัญญาณข้อมูลที่เราใช้รับส่งเราก็ยังคงแยกออกว่า ส่วนไหนคือข้อมูล และส่วนไหนคือสัญญาณรบกวน แต่ในระยะทางไกลๆ สัญญาณข้อมูลที่ส่งออกไปจะมีกำลังอ่อนลงตามระยะทาง ทำให้ความแตกต่างระหว่างสัญญาณรบกวนน้อยลง การรับข้อมูลจะทำได้ลำบากมากยิ่งขึ้น

โมเด็มที่รับส่งข้อมูลด้วยความเร็วไม่เกิน 2,400 บิทต่อวินาที มักไม่ค่อยมีปัญหาเรื่องของสัญญาณรบกวน เนื่องจากว่าความเร็วขนาดนี้การผสมสัญญาณในการรับส่งยังไม่ซับซ้อนมากนัก รูปคลื่นที่ส่งออกไปมีความแตกต่างกันชัดเจน เมื่อถูกรบกวนจากสัญญาณอื่น ๆ ก็ยังสามารถแยกออกได้ง่ายกว่าข้อมูลที่ส่งมาเป็นอย่างไร ปัญหาของสัญญาณรบกวนในสายจะมีผลกับ โมเด็มความเร็วสูง เช่น 4,800 หรือ 9,600 บิทต่อวินาทีค่อนข้างมาก เพราะที่ความเร็วสูงโมเด็มจะต้องใช้เทคนิคการผสมสัญญาณซับซ้อนมาก เพื่อส่งข้อมูลไปในสายโทรศัพท์ ซึ่งมี bandwidth 3,000 เฮิร์ตซ์เท่านั้น รูปคลื่นของสัญญาณที่ส่งออกไปจึงต้องมีความแม่นยำสูง ผู้รับถึงจะรับข้อมูลได้ถูกต้อง สัญญาณรบกวนจะทำให้รูปคลื่นผิดไปจากเดิมมีผลถึงการรับข้อมูลผิดพลาด โมเด็มความเร็วสูงจำเป็นต้องมีฮาร์ดแวร์พิเศษมาทำหน้าที่ตัดสัญญาณรบกวนในสายออกโดยเฉพาะ ทำให้ราคาของโมเด็มแพงมากขึ้น

4.15 ปัญหาของโมเด็มความเร็วสูง

ดังที่ได้กล่าวมาแล้วว่าเครือข่ายโทรศัพท์นั้นมีขีดจำกัดทำให้โมเด็มรับส่งข้อมูลด้วยความเร็วสูงได้ลำบาก ในโมเด็มรุ่นก่อนๆ รับส่งข้อมูลด้วยความเร็วต่ำ เราสามารถใช้เทคนิคการผสมสัญญาณอย่างไม่ซับซ้อนนักได้และสัญญาณรบกวนก็มีผลต่อการรับส่งไม่มาก เนื่องจากส่งข้อมูลที่ความเร็วต่ำ เมื่อเราพยายามรับส่งข้อมูลผ่านโมเด็มให้เร็วขึ้น ขีดจำกัดของระบบโทรศัพท์ที่มีอยู่เดิมจะยิ่งมีผลให้ความผิดพลาดในการรับส่งข้อมูลสูงขึ้น จนอาจถึงขนาดรับส่งข้อมูลกันไม่ได้เลย เทคนิคการผสมสัญญาณที่เราใช้ในโมเด็มความเร็วต่ำกับโมเด็มความเร็วปานกลาง ก็ไม่เพียงพอเสียแล้วเมื่อต้องการส่งข้อมูลที่ความเร็วสูง เพื่อให้โมเด็มรับส่งข้อมูลที่ความเร็วสูง เราจำเป็นต้องใช้เทคนิคหลายอย่างเข้าช่วยเพื่อให้โมเด็มรับส่งข้อมูลได้สูงขึ้น ไม่ว่าจะเป็นเทคนิคการผสมสัญญาณที่สลับซับซ้อนกว่าเดิมการกำจัดสัญญาณรบกวน และการใช้เทคนิคพิเศษเพื่อให้โมเด็มรับส่งข้อมูลสวนทางกันได้ที่ความเร็วสูง โดยใช้ความถี่ของสัญญาณไม่เกิน 3,400 เฮิร์ตซ์ ของโทรศัพท์

4.16 ขีดจำกัดของโมเด็มในปัจจุบัน

ขีดจำกัดของการรับส่งข้อมูลผ่านโมเด็มในปัจจุบันอยู่ที่ความเร็ว 2,400 บิทต่อวินาทีถ้าเราไม่ใช้เทคนิคพิเศษเข้าช่วย แต่เมื่อนำเทคนิคใหม่ ๆ เข้ามาใช้โมเด็มสามารถรับรู้ส่งข้อมูล

ได้สูงถึง 9,600 บิทต่อวินาที สวนทางกันได้ผ่านสายโทรศัพท์ ซึ่งที่ความเร็ว 9,600 บิทต่อวินาทีนี้ เราต้องแก้ปัญหาหลายอย่างที่เป็นขีดจำกัดของเครือข่ายโทรศัพท์เสียก่อน

ปัญหาหนักด่านแรกของโมเด็มความเร็วสูง คือ ช่วงกว้างของความถี่ ในระบบโทรศัพท์ที่มีเพียงประมาณ 3,000 เฮิร์ตซ์ เท่านั้น ยิ่งเราต้องการส่งข้อมูลด้วยความเร็วสูงมากขึ้นเท่าใด เราก็ต้องการช่วงกว้างของความถี่ (bandwidth) ของการรับส่งมากยิ่งขึ้นเท่านั้น และที่แย่ลงไปอีกก็คือ เมื่อเรารับส่งข้อมูลในแบบสวนทางกันได้ (ฟูลดูเพล็กซ์) ช่วงกว้างของความถี่ที่ต้องการก็เพิ่มมากขึ้นไปอีก

เทคนิคการแบ่งความถี่ ที่ใช้ในโมเด็มความเร็วปานกลาง ให้รับส่งข้อมูลสวนทางกันได้ผ่านสายโทรศัพท์นั้น จะนำมาใช้กับโมเด็มความเร็วสูงได้ยากหรือใช้ไม่ได้เลยเนื่องจากโมเด็มความเร็วปานกลางมีความเร็วในการรับส่งข้อมูลต่ำพอ ที่เราจะแบ่งย่านความถี่ออกเป็นสองส่วนให้อยู่ภายในช่วง 3,000 Hz ได้ โดยแต่ละฝ่ายก็รับส่งข้อมูลแยกจากกัน พอเรารับส่งข้อมูลด้วยความเร็วสูงถึง 9,600 บิทต่อวินาที ย่านความถี่ที่ใช้อย่างต่ำสำหรับส่งจะเท่ากับ 2,400 เฮิร์ตซ์ โดยแต่ละลูกคลื่นผสมสัญญาณเข้าไปครั้งละ 4 บิท (2,400 x 4 = 9,600 บิทต่อวินาที) และอีกย่านความถี่หนึ่งสำหรับรับจะอยู่ในช่วง 4,800 เฮิร์ตซ์ เพื่อแยกความถี่รับส่งออกจากกันไม่ให้มารบกวนกันได้ ซึ่งที่ย่านความถี่ 4,800 เฮิร์ตซ์ นี้ยังคงมีอัตราการเปลี่ยนแปลงของสัญญาณในสายเท่ากับ 2,400 baud เหมือนเดิม แต่เราจะเห็นว่าความถี่ย่าน 4,800 เฮิร์ตซ์นี้ เกินขีดจำกัดของระบบโทรศัพท์ไปเสียแล้ว เพราะความถี่สูงสุดที่รับส่งผ่านระบบโทรศัพท์มีค่าประมาณ 3,300 ถึง 3,400 เฮิร์ตซ์เท่านั้น ทำให้เราใช้เทคนิคการแบ่งความถี่กับโมเด็มความเร็วสูงเพื่อรับส่งข้อมูลในแบบฟูลดูเพล็กซ์ไม่ได้ จำเป็นต้องหาวิธีอื่นแทน

4.17 <u>เทคนิคที่ใช้โมเด็มความเร็วสูง</u>

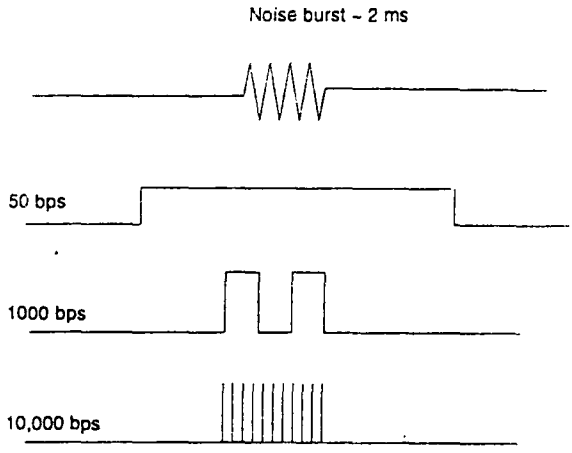
เมื่อช่วงกว้างของความถี่ไม่พอให้โมเด็มความเร็วสูงแบ่งเป็นด้านรับและด้านส่งได้ จึงมีการนำเทคนิคที่เรียกว่า Echo Cancellation มาใช้กับโมเด็มความเร็วสูง การทำงานของ Echo Cancellation ก็คือ ขณะที่โมเด็มส่งสัญญาณอนาล็อกออกไปทางสายส่ง จะจำรูปคลื่นที่ส่งออกไปเอาไว้เมื่อโมเด็มได้รับสัญญาณที่อีกด้านหนึ่งส่งมา จะนำรูปคลื่นที่ตัวเองส่งออกไปมาลบออกจากสัญญาณที่ได้รับ เพื่อให้เหลือแต่สัญญาณที่ต้องการจะรับเข้ามาเท่านั้น ด้วยวิธีนี้ทั้งด้านรับและด้านส่งสามารถใช้ย่านความถี่เดียวกันสำหรับส่งข้อมูลได้ โดยไม่รบกวนกัน เนื่องจากทั้งด้านรับและด้านส่งจะลบสัญญาณของตัวโมเด็มเองออกไปจากสัญญาณที่ได้รับเสมอ และมีผลทำให้สัญญาณสะท้อนถูกกำจัดออกไปด้วย Echo Cancellation นี้มีการทำงานที่ซับซ้อนมาก นอกจากจะต้องจำรูปคลื่นที่ตัวเองส่งออกไป เพื่อนำมาลบออกจากสัญญาณที่ได้รับแล้ว ยังจะต้องรู้ว่าควรจะลบ

สัญญาณออกด้วยความแรงเท่าไร และ เริ่มต้นลบสัญญาณที่ตรงไหน เพื่อให้สัญญาณที่ได้รับถูกต้อง เพราะในการส่งข้อมูลระยะทางไกลนั้นสัญญาณสะท้อนของตัวโมเด็มเองจะใช้เวลาเดินทางกลับมาถึงด้านส่งนานพอสมควร และแต่ละปลายทางก็ใช้เวลาไม่เท่ากัน ความแรงของสัญญาณสะท้อนก็ไม่แน่นอน โดยเฉพาะอย่างยิ่งการส่งข้อมูลข้ามทวีปผ่านดาวเทียม วงจร Echo Cancellation จึงมีราคาแพงมาก ปัจจุบันมีใช้ในโมเด็มตามมาตรฐาน V.23 ของ CCITT สำหรับส่งข้อมูลผ่านสายโทรศัพท์หรือสายตรงเท่านั้น

ปัญหาต่อไป ที่ผู้ผลิตโมเด็มความเร็วสูงต้องขบคิด ก็คือ เทคนิคการผสมสัญญาณ (modulation technic) ถ้าเราค้นพบวิธีผสมสัญญาณที่ดีมากขึ้น การรับส่งข้อมูลก็จะมีความเร็วสูงขึ้นตามไปด้วย เทคนิคที่ใช้กันมากทุกวันนี้ คือ QAM (Quadrature Amplitude Modulation) ซึ่งสามารถผสมสัญญาณได้ครั้งละ 4 บิทในหนึ่งลูกคลื่น หากเราผสมสัญญาณได้มากกว่านี้เช่น 5 บิท 6 บิท จนถึง 8 บิท ในหนึ่งลูกคลื่นโมเด็มก็สามารถรับส่งข้อมูลได้เร็วขึ้นอาจถึงสองเท่าจากปัจจุบัน แต่ว่าเป็นเรื่องที่ค่อนข้างเป็นไปได้ยาก เพราะว่าการผสมสัญญาณหลาย ๆ บิทเข้าไปในหนึ่งลูกคลื่นนั้น จะทำให้ความแตกต่างของสัญญาณแต่ละช่วงน้อยลงทุกที จนกระทั่งด้านรับอาจแปลงข้อมูลกลับมาผิดพลาดมากกว่าที่จะได้รับข้อมูลที่ถูกต้อง และทั้งการด้านรับและด้านส่งจะต้องมีความแม่นยำสูง ทำให้วงจรฮาร์ดแวร์ของโมเด็มซับซ้อนขึ้นไปอีก

สำหรับปัญหาพื้นฐานที่ยังคงมีผลถึงโมเด็มความเร็วสูงคือเรื่องของสัญญาณรบกวนนั่นเอง โมเด็มที่รับส่งข้อมูลด้วยความเร็วสูง ย่อมได้รับผลกระทบจากสัญญาณรบกวนมากกว่า โมเด็มความเร็วต่ำไม่ว่าจะเป็นด้านความไวที่จะถูกรบกวนได้ง่ายกว่า และจำนวนข้อมูลที่ผิดพลาดจากสัญญาณรบกวนที่เข้ามาในวงจร เช่น ถ้ามีสัญญาณรบกวนเกิดขึ้นในช่วงสั้น ๆ ประมาณ 1 ms (หนึ่งส่วนพันวินาที) ที่ความเร็ว 300 baud นั้น ซึ่งหนึ่งลูกคลื่นมีค่าเท่ากับ 3.33 ms จะเห็นว่าสัญญาณรบกวนแทบจะไม่มีผลใด ๆ เลยการรับส่งข้อมูลยังคงทำได้อย่างถูกต้อง เพราะสัญญาณรบกวนที่เกิดขึ้นนั้นมีค่าน้อยกว่า 50 เปอร์เซ็นต์ของหนึ่งลูกคลื่นเสียอีก

เมื่อสัญญาณรบกวนนี้เข้ารบกวนโมเด็มความเร็ว 1,200 baud ในหนึ่งลูกคลื่นของ 1,200 baud มีค่าเท่ากับ 0.83 ms ผลกระทบจะมากขึ้น เนื่องจากสัญญาณรบกวนมีผลมากกว่าหนึ่งลูกคลื่น ทำให้การรับส่งข้อมูลผิดพลาดได้ และ เมื่อความเร็วเพิ่มขึ้นเป็น 2,400 baud สัญญาณรบกวนช่วงเวลา 1 ms สามารถรบกวนสัญญาณที่เรารับส่งได้เกิน 2 ลูกคลื่นติดต่อกัน ถ้าหากว่าที่ 2,400 baud นี้ เราผสมสัญญาณลูกคลื่นละ 4 บิท ข้อมูลจะผิดพลาดอย่างน้อยถึง 8 บิทเดียว โมเด็มความเร็วสูง จึงได้รับผลกระทบจากสัญญาณรบกวนมากกว่าโมเด็มความเร็วปานกลางหลายเท่า



รูปที่ 4.13 สัญญาณรบกวนจะมีผลต่อโมเด็มความเร็วสูงมากกว่าที่ความเร็วต่ำ

ในแง่ของความไวที่จะถูกรบกวน โมเด็มความเร็วสูง ใช้การผสมสัญญาณซับซ้อนกว่าโมเด็มความเร็วต่ำ ดังนั้นระดับของสัญญาณที่เปลี่ยนแปลงไปเพียงเล็กน้อยเนื่องจากถูกรบกวนจะมีผลทำให้ข้อมูลผิดไปจากเดิมได้ง่าย ผู้ผลิตโมเด็มใช้วงจรฮาร์ดแวร์หลายอย่างเข้าช่วยแก้ปัญหาอันนี้ไม่ว่าจะเป็นวงจรกรองสัญญาณ วงจรควบคุมระดับสัญญาณอัตโนมัติ รวมถึงวงจรชดเชยสัญญาณความถี่ต่างๆ เพื่อให้สัญญาณที่ได้รับปราศจากสัญญาณรบกวน และมีรูปคลื่นถูกต้อง ตรงตามที่ด้านส่งส่งมาให้

4.18 มาตรฐานของโมเด็มความเร็วสูง

จากข้อจำกัดที่กล่าวมา มาตรฐานของโมเด็มความเร็วสูงจึงมีรายละเอียดมาก ตั้งแต่ความถี่ที่ใช้ในการรับส่งข้อมูล เทคนิคการกำจัดสัญญาณที่สะท้อนกลับมา (Echo Cancellation) เทคนิคการผสมสัญญาณ และเพื่อให้โมเด็มความเร็วสูงรับส่งข้อมูลได้ถูกต้อง ก็อาจรวมมาตรฐานของการตรวจสอบข้อผิดพลาดในการส่งเข้าไปด้วย นอกจากนี้เพื่อให้โมเด็มส่งข้อมูลได้เร็วยิ่งขึ้น ยังอาจใช้เทคนิคการลดขนาดข้อมูล (Data Compression) มาลดขนาดข้อมูลก่อนส่งไปยังปลายทางทำให้ความเร็วในการส่งข้อมูลเร็วขึ้นไปอีก มาตรฐานของโมเด็มความเร็วสูงจึงใช้เวลานานกว่าจะตกลงกันออกมาได้ ปัญหาของโมเด็มความเร็วสูงดังกล่าว ยังคงต้องมีการพัฒนาแก้ไขกันต่อไป แม้ว่าจะมีแนวโน้มยากขึ้นทุกที ๆ ตามความเร็วที่เพิ่มขึ้นมา ไม่อย่างนั้นแล้วการรับส่งข้อมูลไปใช้งานไม่ได้สิ่งนี้นับว่าท้าทายเทคโนโลยีสมัยใหม่มากสำหรับโมเด็มความเร็วสูง

4.19 การตรวจสอบความผิดพลาดและการลดขนาดข้อมูล

การรับส่งข้อมูลไม่ว่าจะใช้วิธีใดก็ตามจะเกิดการรับส่งผิดพลาดขึ้นเสมอ โดยมีปัจจัยอยู่สองอย่าง คือ ถ้าส่งข้อมูลที่ความเร็วสูงมากขึ้นเท่าใด ข้อมูลก็จะผิดพลาดมากขึ้นเท่านั้น หรือส่งข้อมูลไกลออกไปมากขึ้น โอกาสที่จะได้รับข้อมูลผิดก็จะเพิ่มขึ้นตามกันไป ดังนั้น ถ้าเราต้องการส่งข้อมูลความเร็วสูงผ่านระยะทางไกล ๆ ความผิดพลาดในการรับส่งข้อมูล ก็จะยิ่งมีมากขึ้นเป็นทวีคูณ เราไม่สามารถหลีกเลี่ยงข้อผิดพลาดนี้ได้ เนื่องจากเป็นข้อจำกัดของระบบการรับส่งข้อมูลเอง วิธีที่พอจะทำได้ ก็คือ พยายามตรวจสอบว่าการรับส่งข้อมูลผิดพลาดที่ส่วนใด แล้วส่งส่วนนั้นไปใหม่ ซึ่งวิธีนี้จะดีกว่าการส่งไปจนจบแล้วถ้าพบวว่ามีข้อมูลผิดค่อยทำการส่งใหม่อีกครั้งเพราะจะเสียเวลาและค่าใช้จ่ายสูงมาก

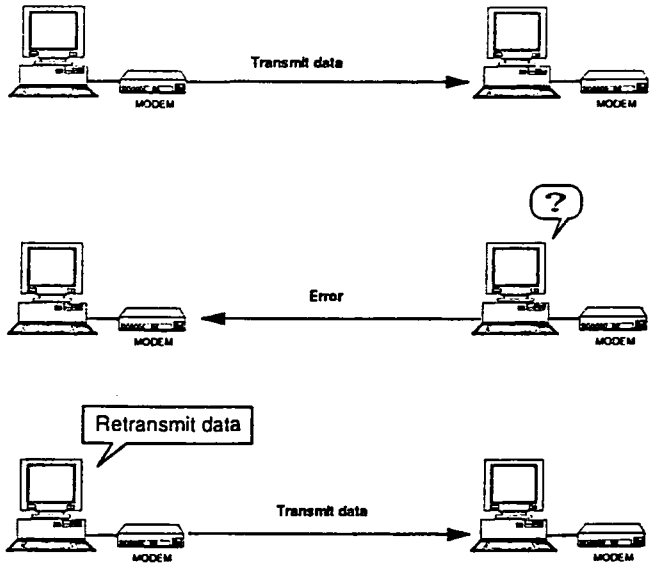
โมเด็มในรุ่นแรก ๆ จะรับส่งข้อมูลอย่างธรรมดา ไม่มีการตรวจสอบความถูกต้องของข้อมูลเลย ในปัจจุบันเมื่อโมเด็มมีความเร็วสูงขึ้น การรับส่งข้อมูลผิดพลาดก็มีมากขึ้นตามไปด้วย เราจึงได้เพิ่มส่วนของการตรวจสอบข้อผิดพลาดเข้ามาช่วยรับส่งข้อมูลผ่านโมเด็ม ซึ่งจะช่วยให้เรามั่นใจได้ว่าข้อมูลที่ได้รับถูกต้องและเชื่อถือได้ โดยการตรวจสอบความผิดพลาด แบ่งได้เป็นสองแบบ คือ การตรวจสอบความผิดพลาดที่ใช้ฮาร์ดแวร์ และ แบบที่ใช้ซอฟต์แวร์

4.20 วิธีตรวจสอบความผิดพลาดแบบต่างๆ

หลักการทำงานของการตรวจสอบก็ใช้หลักการง่ายๆ คือ แบ่งข้อมูลที่ต้องการส่งออกเป็นส่วน และนำแต่ละส่วนนั้นมาหาผลรวม (check sum) หรือใช้สูตรทางคณิตศาสตร์คำนวณหาค่าตัวเลขตรวจสอบหรือ CRC (Cyclic Redundancy Check) และส่งผลรวมหรือตัวเลขตรวจสอบนี้ติดไปกับข้อมูลแต่ละส่วนด้วย ทางด้านรับเมื่อได้รับข้อมูลมาแล้วก็จะหาผลรวมหรือตัวเลขตรวจสอบจากข้อมูลที่ได้รับอีกครั้งหนึ่ง แล้วจึงนำมาเปรียบเทียบกับผลรวมหรือตัวเลขตรวจสอบที่มาจากด้านส่งว่ามีค่าตรงกันหรือไม่ ถ้าตรงกันก็ถือว่าข้อมูลที่ได้รับนั้นถูกต้องและส่งสัญญาณบอกว่าข้อมูลถูกต้องไปให้ผู้ส่ง เพื่อให้ทำการส่งข้อมูลต่อไป ถ้าหากตัวเลขผลรวมหรือเลขตรวจสอบไม่ตรงกันจะถือว่าข้อมูลผิดพลาด ด้านรับจะส่งสัญญาณไปให้ผู้ส่งเพื่อทำการส่งซ้ำอีกครั้งหนึ่งจนกว่าจะถูกต้อง หลักการนี้ใช้กับทั้งการตรวจสอบข้อผิดพลาดแบบซอฟต์แวร์และฮาร์ดแวร์

ในการตรวจสอบข้อผิดพลาดแบบที่ใช้ฮาร์ดแวร์นั้น ที่เราพบกันมากได้แก่ ระบบ MNP และระบบ ARQ ซึ่งจะเป็นวงจรฮาร์ดแวร์สร้างขึ้นมาอยู่ภายในตัวโมเด็มเลย ดังนั้นการใช้งาน MNP หรือ ARQ เพื่อตรวจสอบข้อผิดพลาด โมเด็มทั้งสองด้านจะต้องมีวงจรฮาร์ดแวร์นี้อยู่ด้วย ถ้าโมเด็มด้านใดด้านหนึ่งเป็นโมเด็มธรรมดาก็จะใช้ไม่ได้ การใช้ฮาร์ดแวร์ตรวจสอบข้อผิดพลาดมีข้อดีคือ ความเร็วในการทำงานจะสูงกว่าแบบที่ใช้ซอฟต์แวร์ แต่ก็มีข้อเสีย คือ โมเด็มจะต้อง

มีฮาร์ดแวร์ดังกล่าวทั้งคู่จึงจะทำงานได้ และเนื่องจากมันเป็นส่วนของฮาร์ดแวร์ เราจึงเปลี่ยนแปลงแก้ไขอะไรไม่ได้ ทำให้ความคล่องตัวในการใช้งานนี้น้อยกว่าแบบที่ใช้ซอฟต์แวร์



รูปที่ 4.14 ในการรับส่งข้อมูลแบบที่มีการตรวจสอบความผิดพลาด เมื่อมีข้อผิดพลาดเกิดขึ้น ทางด้านส่งจะส่งข้อมูลซ้ำให้ใหม่

ปัจจุบันฮาร์ดแวร์ที่ใช้ตรวจสอบข้อผิดพลาดของโมเด็มที่ใช้อยู่ 2 แบบใหญ่ ๆ คือ ตามแบบของ MNP และตามแบบของ CCITT V-42 นับว่าใช้กันมากในการกำหนดมาตรฐานของฮาร์ดแวร์ตรวจสอบข้อผิดพลาดในการรับส่งข้อมูลของโมเด็ม เนื่องจาก V-42 มีการทำงานได้ 2 แบบ แบบแรกเรียกว่า Link Access Procedure for Modems (LAPM) และแบบที่สองจะมีการทำงานเหมือนกับ MNP Class 4 (MNP 4) ดังนั้น โมเด็มที่ใช้ฮาร์ดแวร์ตรวจสอบความผิดพลาดแบบ V-42 จึงรับส่งข้อมูลได้ทั้งกับ MNP และ V-42 เมื่อโมเด็มติดต่อกับปลายทางได้มันจะพยายามใช้การรับส่งแบบ LAPM ตรวจสอบความผิดพลาดก่อน ถ้าโมเด็มปลายทางไม่ได้ใช้ฮาร์ดแวร์ตามแบบ V-42 มันก็จะใช้การตรวจสอบความผิดพลาดแบบ MNP Class 4 รับส่งข้อมูลต่อไป และถ้าโมเด็มปลายทางไม่มีฮาร์ดแวร์ตรวจสอบความผิดพลาดอยู่ในตัวเลย มันก็จะทำการรับส่งข้อมูลแบบธรรมดาที่ไม่มีการตรวจสอบความผิดพลาดแทน ซึ่งแนวโน้มในอนาคตจะมีการใช้ฮาร์ดแวร์ตรวจสอบความผิดพลาดกันมากขึ้น

มาตรฐานการตรวจสอบความผิดพลาด V-42 นี้ผู้ผลิตโมเด็มเริ่มบรรจุเข้าไปในโมเด็มรุ่นใหม่ความเร็วสูงหลายยี่ห้อแล้ว เนื่องจากมันทำให้โมเด็มรับส่งข้อมูลได้ถูกต้อง และใช้งานได้กว้างกว่าแบบ MNP-4

ส่วนการตรวจสอบข้อผิดพลาดโดยใช้ซอฟต์แวร์นั้น มีอยู่หลายระดับด้วยกัน อย่างเช่น X modem, X modem CRC, Y modem, Kermit ฯลฯ ซึ่งการใช้งานทั้งฝ่ายส่งก็ต้องใช้ซอฟต์แวร์ที่มีระบบตรวจสอบข้อผิดพลาดเหมือนกันทั้งสองด้าน การใช้ซอฟต์แวร์ตรวจสอบข้อผิดพลาดนี้มีข้อดีคือเราสามารถเลือกใช้แบบไหนก็ได้ ตามแต่ผู้รับและผู้ส่งจะตกลงกัน ความคล่องตัวในการใช้งานมีสูง เพราะโมเด็มที่ใช้รับส่งข้อมูลเป็นโมเด็มแบบธรรมดา ๆ ซึ่งทำให้เราสามารถติดต่อกับใคร ๆ ได้ โดยยังคงมีระบบตรวจสอบข้อผิดพลาดอยู่ ไม่จำเป็นจะต้องใช้โมเด็มแบบพิเศษ สำหรับข้อเสียก็คือ เวลาที่ใช้รับส่งข้อมูลอาจจะเพิ่มขึ้นได้ เนื่องจากเราต้องส่งผลรวมหรือตัวเลขตรวจสอบติดไปกับข้อมูลด้วย และคอมพิวเตอร์ด้านรับจะเสียเวลาส่วนหนึ่งไปกับการตรวจสอบตัวเลขผลรวม ทำให้การทำงานช้าลงได้

การทำงานอย่างคร่าว ๆ ของแต่ละระบบเป็นดังนี้ คือ X modem จะแบ่งข้อมูลออกเป็นส่วน ๆ มีขนาด 128 ไบต์ หรือ 128 ตัวอักษร และคำนวณหาผลรวมของ 128 ไบต์ออกมาเพื่อส่งไปพร้อมกับข้อมูล ทางด้านรับก็จะคำนวณหาผลรวมจากข้อมูลที่ได้รับมาเปรียบเทียบกับด้านส่ง ถ้ามีข้อผิดพลาดก็จะขอให้ด้านส่ง ส่งข้อมูลซ้ำอีกครั้งหนึ่งจนกว่าจะถูกต้อง X modem นี้จัดว่าเป็นซอฟต์แวร์ตรวจสอบข้อผิดพลาดที่นิยมใช้กันมากแบบหนึ่งซึ่งมีความเชื่อถือได้สูงถึง 96 เปอร์เซ็นต์ สำหรับ X modem CRC นั้นจะมีคุณสมบัติเหมือนกับ X modem ทุกอย่าง ยกเว้นวิธีคิดหาผลรวมจะใช้สูตรคณิตศาสตร์แบบ Cyclic Redundaancy Check แทน ซึ่งการใช้ CRC เป็นตัวเลขตรวจสอบนี้จะทำให้การรับส่งข้อมูลเชื่อถือได้สูงถึง 99.6 เปอร์เซ็นต์ แต่ว่าจำนวนบิทที่ใช้ส่งเป็นตัวเลขตรวจสอบจะมากกว่าแบบที่ใช้ผลรวมธรรมดาเล็กน้อย และ Y modem นั้น มีการทำงานคล้ายกับ X modem มาก ผิดกันตรงที่ Y modem จะแบ่งข้อมูลออกเป็นส่วนละ 1,024 ไบต์ หรือ 1 กิโลไบต์ ซึ่งจะมีผลทำให้การรับส่งข้อมูลมีประสิทธิภาพมากขึ้น เนื่องจากจำนวนตัวเลขผลรวมที่ส่งไปกับข้อมูลลดลงมาก เมื่อเทียบกับ X modem ที่ต้องส่งผลรวมทุก ๆ 128 ไบต์

สำหรับ Kermit นั้นก็เป็นอีกระบบหนึ่งที่ได้รับความนิยมไม่แพ้ X modem แต่ว่า Kermit นั้นมีความคล่องตัวมากกว่า X modem หลายอย่าง เช่น ผู้ใช้สามารถกำหนดได้ว่าจะแบ่งข้อมูลออกเป็นส่วนละกี่ไบต์ เพื่อให้เหมาะกับงานที่ต้องการ และในบางกรณี Kermit สามารถรับส่งข้อมูลต่อได้เมื่อมีอะไรมาขัดจังหวะขณะรับส่งข้อมูลอยู่ เช่น มีสัญญาณรบกวนเข้ามา ซึ่งบางครั้ง X modem ทำไม่ได้ นอกจากนี้ Kermit ยังมีคำสั่งให้ใช้มากกว่าด้วย

4.21 การลดขนาดข้อมูลก่อนการส่ง

สิ่งที่น่าสนใจอีกอย่างหนึ่งที่เริ่มเข้ามาเกี่ยวข้องกับโมเด็ม ก็คือ การลดขนาดข้อมูลก่อนการส่งออกไปทางสายส่ง คุณสมบัติอันนี้มีผลทำให้โมเด็มดูเหมือนรับส่งข้อมูลได้เร็วขึ้น เช่น

โมเด็มธรรมดารับส่งข้อมูลที่ความเร็ว 9,600 บิทต่อวินาที เมื่อโมเด็มมีความสามารถลดขนาดข้อมูลก่อนส่งได้ 2 เท่า ความเร็วในการรับส่งข้อมูลโดยรวมจึงมีค่าเท่ากับ 19,200 บิทต่อวินาที โดยเราไม่ต้องเพิ่มความเร็วในการรับส่งข้อมูลของโมเด็มเลย นับเป็นการเพิ่มประสิทธิภาพของโมเด็มที่น่าสนใจมาก แน่นอนว่าคุณสมบัติการลดขนาดของโมเด็มจะต้องเป็นส่วนของฮาร์ดแวร์ที่สร้างขึ้นมาโดยเฉพาะ เพื่อให้มีความเร็วในการทำงานสูง ในขณะที่มีใช้อยู่ในฮาร์ดแวร์ระบบ MNP เรียกว่า MNP-5 ซึ่งสามารถลดขนาดข้อมูลได้เฉลี่ย 1.74 เท่า

ส่วน CCITT ได้ออกมาตรฐานการลดขนาดข้อมูลออกมาเหมือนกัน เรียกว่ามาตรฐาน V.42 bis ซึ่งสามารถลดขนาดข้อมูลได้ดีกว่า MNP Class 5 (MNP-5) โดยลดขนาดข้อมูลได้เฉลี่ย 2 ถึง 3 เท่า เมื่อนำมาใช้กับโมเด็มความเร็ว 9,600 บิทต่อวินาที เราจะรับส่งข้อมูลได้เร็วถึงประมาณ 20,000 ถึง 38,400 บิทต่อวินาทีเลยทีเดียว การลดขนาดข้อมูลจะลดได้มากน้อยแค่ไหนขึ้นอยู่กับชนิดของข้อมูล และการลดขนาดข้อมูลตามมาตรฐาน V.42 bis นี้จะต้องใช้ควบคู่กับการตรวจสอบความผิดพลาด V.42 เสมอจะใช้เดี่ยว ๆ ไม่ได้ คาดว่าโมเด็มรุ่นต่อ ๆ ไปจะเริ่มมีคุณสมบัติการลดขนาดข้อมูลรวมอยู่ในตัวเป็นมาตรฐานเพิ่มมากขึ้นเรื่อย ๆ

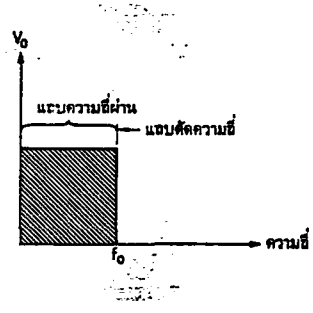
หลักการทำงานของการลดขนาดข้อมูลมีเทคนิคแตกต่างกันหลายอย่าง เช่น บางชนิดใช้การเข้ารหัสเข้าช่วยลดขนาดลง บางชนิดก็ส่งข้อมูลเท่าที่จำเป็นเท่านั้นเมื่อเจอตัวที่ซ้ำๆ กันก็จะส่งเป็นรหัสออกไป ทำให้จำนวนข้อมูลลดลง ข้อมูลแต่ละชนิด เช่น ข้อมูลตัวอักษร ข้อมูลจากเครื่องคอมพิวเตอร์ ข้อมูลจากโปรแกรมต่างๆ มีผลต่อการลดขนาดข้อมูลมาก เทคนิคบางอย่างอาจใช้ได้ดีกับข้อมูลพวกตัวอักษร แต่ใช้กับข้อมูลคอมพิวเตอร์แล้วขนาดกลับเพิ่มขึ้นจากเดิมก็ได้ มาตรฐานของโมเด็มรุ่นใหม่ต่อ ๆ ไป จะมีคุณสมบัติการลดขนาดข้อมูลรวมอยู่ด้วยอย่างแน่นอน

บทที่ 5

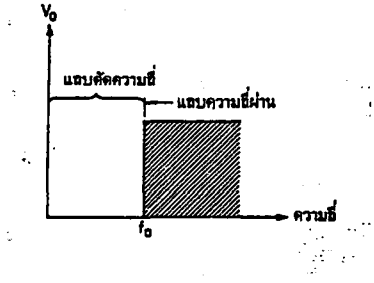
วงจรกรองความถี่

วงจรฟิลเตอร์ (Filter) หรือวงจรกรองความถี่ มีบทบาทที่สำคัญในกระบวนการทั้งอนาล็อกและดิจิตอล ใช้สำหรับกำจัดสัญญาณที่ไม่ต้องการออก เช่น ใช้ในการตัดเสียงฮัมในวงจรขยายเสียง แล้วทำการแยกสัญญาณที่มีความซับซ้อนออกมาเป็นส่วนๆเพื่อป้อนเข้า สู่วงจรต่าง ๆ ของระบบต่อไป เมื่อพิจารณาในด้านการใช้งานแล้ว วงจรกรองความถี่จะถูกแบ่งออกตามลักษณะใช้งานใน 4 ลักษณะ ได้แก่

1. วงจรกรองความถี่ต่ำผ่าน (Low Pass Filter) จะยอมให้สัญญาณที่มีความถี่ตั้งแต่ 0 เฮิร์ตซ์ ไปจนถึงความถี่กำหนด (f_o) ผ่านวงจรไปได้ ส่วนความถี่ตั้งแต่ความถี่ที่กำหนดขึ้นไปจนถึงความถี่อนันต์จะถูกตัดทิ้ง ดังรูปที่ 5.1 ซึ่งจะแสดงความสัมพันธ์ระหว่างระดับสัญญาณทางเอาท์พุท (V_o) ต่อความถี่ที่เปลี่ยนไปของวงจรความถี่ต่ำผ่าน



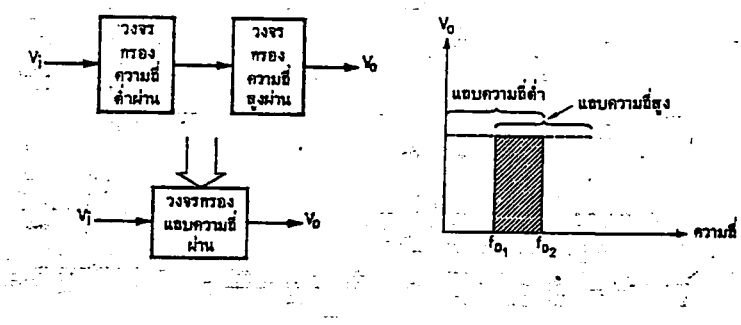
รูปที่ 5.1 แสดงลักษณะการใช้งานของวงจรกรองความถี่ต่ำผ่าน



รูปที่ 5.2 แสดงลักษณะการใช้งานของวงจรกรองความถี่สูงผ่าน

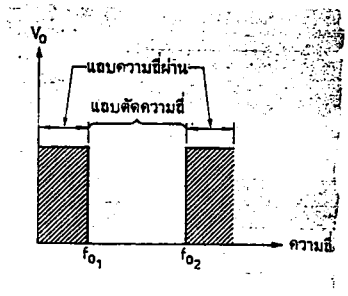
2. วงจรกรองความถี่สูงผ่าน (High Pass Filter) ให้ความถี่ผ่านได้ตั้งแต่ความถี่ที่กำหนด (f_o) ไปจนถึงความถี่ที่อนันต์ แสดงดังรูปที่ 5.2 ส่วนความถี่ที่ต่ำกว่า f_o ลงมาจะถูกตัดทิ้ง

3. วงจรกรองแถบความถี่ผ่าน (Band Pass Filter) เปรียบเสมือนการนำวงจรกรองความถี่ต่ำผ่านมาอนุกรมกับวงจรกรองความถี่สูงผ่าน ดังรูปที่ 5.3 ความถี่ที่จะผ่านได้จะต้องมีค่ามากกว่า f_{o1} และมีค่าน้อยกว่า f_{o2} ส่วนแถบความถี่ที่ต่ำกว่า f_{o1} และสูงกว่า f_{o2} จะถูกตัดทิ้ง พิจารณาดังรูปที่ 5.4

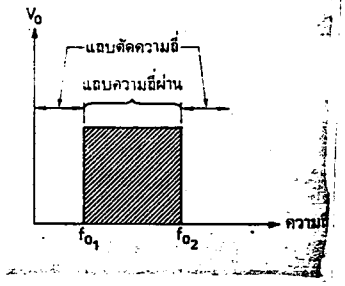


รูปที่ 5.3 วงจรกรองแถบความถี่ผ่านเชิงเปรียบเทียบ

4. วงจรตัดแถบความถี่ (Band Stop Filter) วงจรกรองลักษณะนี้จะตรงข้ามกับการใช้งานในลักษณะแถบความถี่ผ่าน ดังแสดงในรูปที่ 5.5 ความถี่ในช่วงที่สูงกว่า f_{o1} และต่ำกว่า f_{o2} จะถูกตัดทิ้งไป แต่ความถี่ที่ต่ำกว่า f_{o1} และสูงกว่า f_{o2} จะสามารถผ่านไปได้

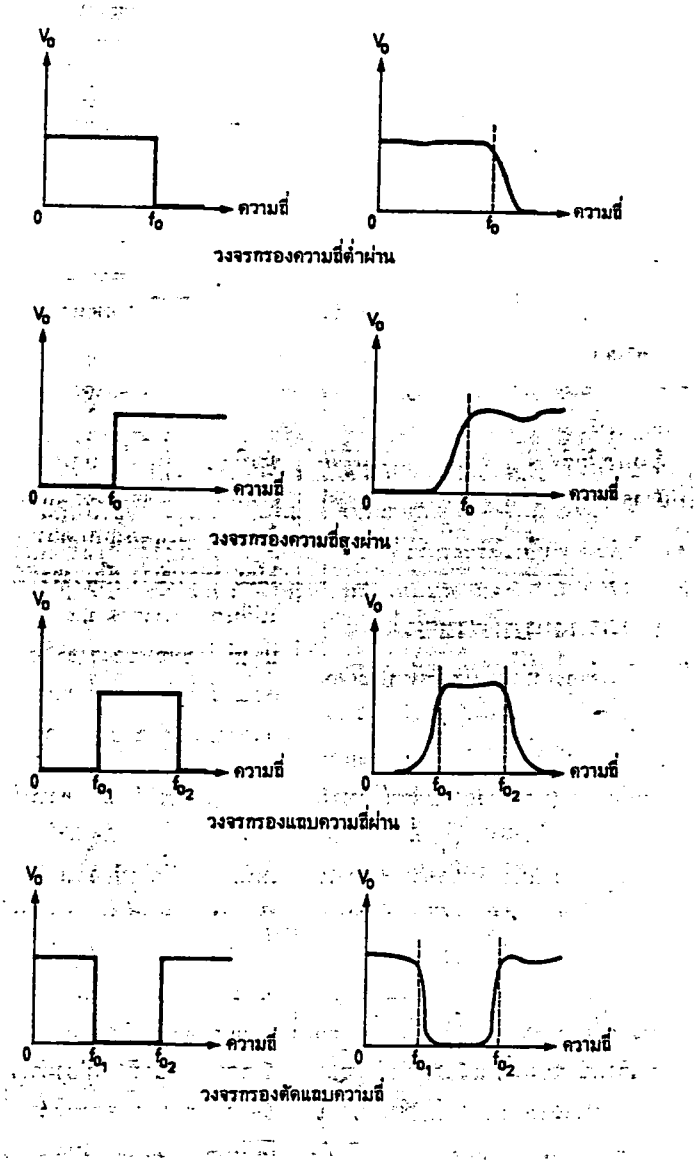


รูปที่ 5.4 แสดงลักษณะการใช้งานของวงจรกรองแถบความถี่ผ่าน



รูปที่ 5.5 แสดงลักษณะการใช้งานของวงจรกรองตัดแถบความถี่

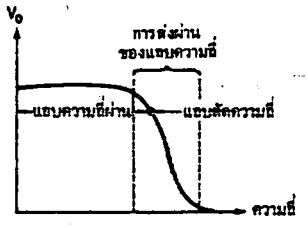
รูปทั้งห้า คือ รูปที่ 5.1 ถึงรูปที่ 5.5 ได้แสดงให้เห็นการใช้งานวงจรกรองความถี่ในลักษณะต่าง ๆ ซึ่งเป็นวงจรในอุดมคติ เพราะในการใช้งานจริง ๆ นั้นไม่สามารถใช้คุณสมบัติของวงจรกรองได้อย่างเต็มที่ ในรูปที่ 5.6 ได้เปรียบเทียบคุณลักษณะในอุดมคติ กับการใช้งานจริง จะเห็นได้ว่า ในการใช้งานจริงจะเกิดปัญหาอยู่ 2 ประการใหญ่ ๆ คือ เกิดการส่งผ่าน(Transition) ของความถี่ และเกิดช่วงการกระเพื่อม (ripple band) ในการใช้งาน



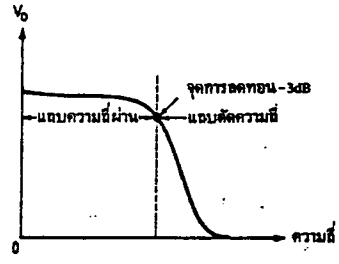
รูปที่ 5.6 แสดงการเปรียบเทียบวงจรกรองในอุดมคติกับการใช้งานจริงทั้ง 4 แบบ

5.1 การส่งผ่านของแถบความถี่

จากรูปที่ 5.7 เป็นตัวอย่างของวงจรกรองความถี่ต่ำผ่าน ซึ่งลักษณะการส่งผ่านความถี่ของแถบความถี่จะเป็นตัวบ่งบอกคุณสมบัติของวงจรกรอง ถ้าการส่งผ่านมีช่วงแคบและมีลักษณะชันแสดงว่าวงจรกรองนั้นมีคุณภาพดี แต่ถ้าช่วงการส่งผ่านความถี่มีช่วงกว้างและลาดมาก แสดงว่า วงจรกรองมีลักษณะการเลือกความถี่ที่เลว ดังนั้น ในการออกแบบจึงควรทำให้ช่วงการส่งผ่านชันและแคบให้ใกล้เคียงกับอุดมคติมากที่สุด



รูปที่ 5.7 แสดงการเกิดการส่งผ่านของแถบความถี่ในการใช้งานจริง

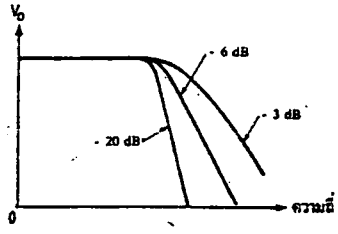


รูปที่ 5.8 แสดงความถี่ตัดซึ่งกำหนดไว้ที่จุดการลดทอน 3 dB

ก่อนที่จะเข้าเรื่องของค่าโรลออฟ (roll off) ซึ่งเป็นค่าที่แสดงลักษณะการส่งผ่านแถบความถี่ของวงจรกรอง เรามาทำความเข้าใจกับความถี่ตัด (cutoff frequency : f_c) กันก่อน ความถี่ตัดคือ ความถี่แรกสุดที่อยู่ในแถบตัดความถี่ เมื่อพิจารณาตามรูปที่ 5.8 ความถี่ตัดจะเป็นตัวแบ่งแถบความถี่และแถบตัดความถี่ออกจากกัน ซึ่งจะถูกกำหนดไว้ในช่วงการส่งผ่านที่มีการลดทอน (attenuation) เท่ากับ -3 dB (การลดทอนในที่นี้ จะหมายถึง อัตราการลดลงของสัญญาณจะมีค่าเท่ากับ $20\log(v_o/v_i)$ เช่น เมื่อ สัญญาณเอาท์พุทลดลงเป็น 1 ใน 10 ของอินพุท การลดทอนจะมีค่าเป็น $20\log 1/10$ ก็คือ -20 dB เป็นต้น)

โรลออฟ คือ อัตราการลดทอนของสัญญาณต่อจำนวนความถี่ที่เปลี่ยนไป หรืออีกนัยหนึ่งก็คือ ความชันของการส่งผ่านนั่นเอง มีหน่วยเป็น เดซิเบลต่อเดคเคด (dB/Decade) ซึ่งได้มาจาก $20\log_{10}(v_o/v_i)$ และ เดซิเบลต่อออกเทฟ (dB/Octave) ซึ่งได้มาจาก $20\log_2(v_o/v_i)$ โดยหน่วยที่มักจะพบบ่อยนั้นจะเป็นหน่วยหลังมากกว่า

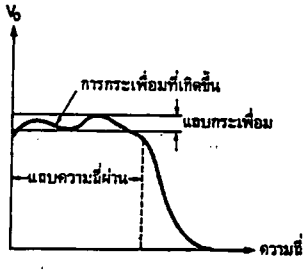
จากรูปที่ 5.9 แสดงการส่งผ่านที่ค่าโรลออฟต่างๆ กันเพื่อให้เห็นว่าการส่งผ่านที่ดีนั้นจะต้องมีค่าโรลออฟสูงๆ โดยค่าโรลออฟที่สูงนั้น นอกจากขึ้นอยู่กับชนิดของวงจรกรองแล้ว ยังขึ้นอยู่กับจำนวนตัวเหนี่ยวนำหรือตัวเก็บประจุที่อยู่ในวงจรมารวมกันและจะเรียกจำนวนนี้ว่า ลำดับ (order) เช่น ถ้ามีรวมกัน 4 ตัว ก็จะเรียกว่า วงจรกรองความถี่ลำดับ 4 (forth order filter) รายละเอียดของชนิดและลำดับของวงจรกรองจะได้กล่าวต่อไป



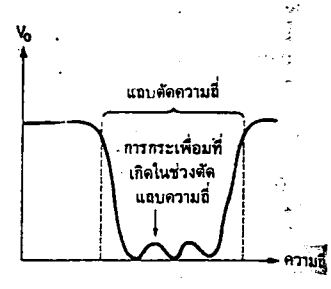
รูปที่ 5.9 แสดงการเปรียบเทียบลักษณะการส่งผ่านความถี่ต่อค่าโรลออฟที่เปลี่ยนไป

5.2 การเกิดการกระเพื่อมในแถบความถี่

เนื่องจากวงจรกรองในอุดมคตินั้น จะมีความแรงของสัญญาณเท่ากันตลอดทุกค่าความถี่ที่ยอมให้ผ่านได้ แต่ในทางปฏิบัติแล้ว จะให้วงจรกรองมีเอาท์พุทระดับเท่ากันตลอดนั้นเป็นไปไม่ได้ เพราะในความถี่บางค่าอาจมีผลกระทบที่ทำให้ค่าอิมพิแดนซ์ของวงจรเปลี่ยนไป การที่วงจรกรองมีความแรงของสัญญาณทางเอาท์พุทไม่เท่ากันนั้น ทำให้เกิดการกระเพื่อมขึ้นในช่วงแถบความถี่ (passband) ในรูปที่ 5.10 การกระเพื่อมนั้นไม่ได้มีแต่ในแถบความถี่เท่านั้น แต่ยังสามารถเกิดในช่วงตัดแถบความถี่ (stopband) ได้อีกด้วย ดังแสดงในรูปที่ 5.11 ซึ่งการกระเพื่อมในช่วงนี้ไม่ค่อยมีความสำคัญในงานใด ๆ นัก แต่จะเกิดปัญหาเมื่อใช้งานในลักษณะของวงจรตัดแถบความถี่



รูปที่ 5.10 แสดงการเกิดการกระเพื่อมในแถบความถี่



รูปที่ 5.11 แสดงการเกิดการกระเพื่อมในแถบตัดความถี่

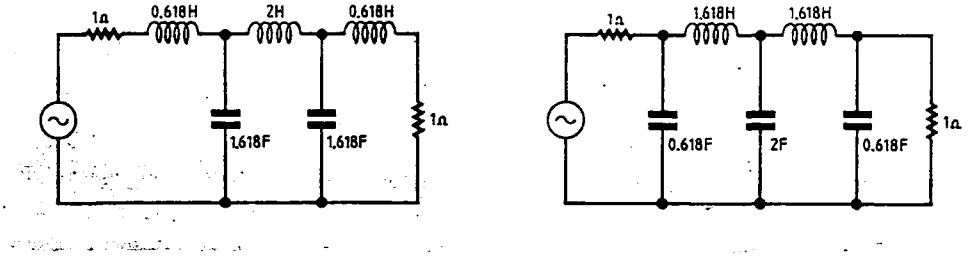
นอกจากปัญหา 2 ประการใหญ่ ๆ นี้แล้ว ก็ยังมีปัญหาที่สำคัญรองลงมาอีก นั่นคือ การผิดเพี้ยน (distortion) ของสัญญาณ อันเนื่องมาจากการเลื่อนเฟส (phase shift) และ

การหน่วง (delay) ซึ่งจะกล่าวถึงภายหลังในหัวข้อที่เกี่ยวกับการใช้วงจรกรองชนิดต่าง ๆ ในการแก้ปัญหา

วงจรกรองขั้นพื้นฐานจะมีอยู่ 2 แบบแยกตามชนิดของอุปกรณ์ที่ใช้ในวงจร ได้แก่ วงจรกรองความถี่แบบพาสซีฟ (passive filter) และวงจรกรองความถี่แบบแอกทีฟ (active filter) ซึ่งวงจรกรองทั้งสองแบบนี้ถูกนำมาใช้งานที่เกี่ยวกับสัญญาณเสียง เช่น การแยกทุ้มแหลมออกจากกัน และในการสื่อสารของมูลทางอิเล็คทรอนิกส์ต่าง ๆ

5.3 วงจรกรองแบบพาสซีฟ

วงจรกรองแบบพาสซีฟ จะประกอบไปด้วยอุปกรณ์แบบพาสซีฟ (passive device) เป็นหลัก ได้แก่ ตัวเก็บประจุและตัวเหนี่ยวนำ แต่มักจะมีตัวต้านทานประกอบอยู่ด้วย ดังรูปที่ 5.12 เนื่องจากวงจรกรองแบบนี้มีราคาแพง การออกแบบก็ซับซ้อนจึงไม่เป็นที่นิยมนัก ข้อดีของวงจรกรองแบบนี้ คือ สามารถตอบสนองความถี่ได้สูงมาก ปัจจุบันสามารถพัฒนาให้ตอบสนองความถี่ได้สูงถึง 500 MHz นอกจากนี้ข้อดีอีกอย่างหนึ่งก็คือ สามารถใช้งานได้โดยไม่ต้องใช้แหล่งจ่ายไฟใด ๆ ทั้งสิ้น



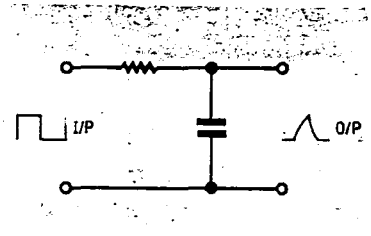
รูปที่ 5.12 ตัวอย่างของวงจรกรองแบบพาสซีฟซึ่งประกอบไปด้วย R, C และ L

ในความเป็นจริงแล้ว สัญญาณที่ออกมาจากเอาท์พุทของวงจรกรองชนิดนี้จะเกิด การสูญเสีย (loss) ขึ้น เนื่องจากค่าอิมพิแดนซ์ของวงจร และเมื่อพิจารณาถึงการส่งผ่านของแถบความถี่จะบ่งบอกอย่างชัดเจนได้ว่า เป็นวงจรที่มีการส่งผ่านได้ไม่ดีนัก อย่างไรก็ตามสามารถแก้ไขปรับปรุงค่าโรลออฟของวงจรได้โดยเพิ่มอุปกรณ์เข้าไป แต่สิ่งที่ตามมา ก็คือการออกแบบที่ซับซ้อนยุ่งยากมากขึ้น

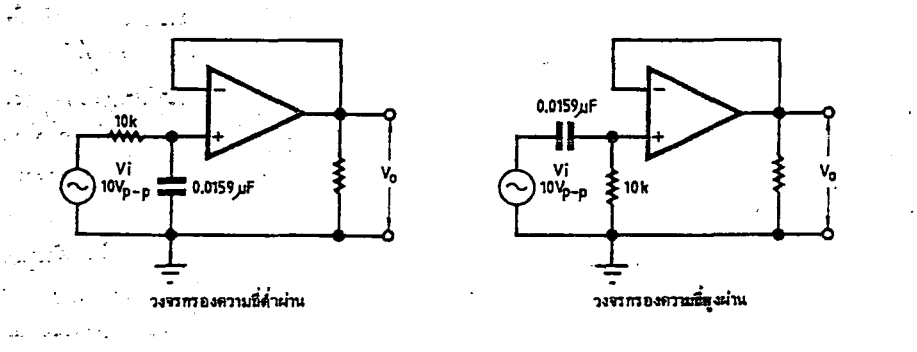
5.4 วงจรกรองแบบแอกทีฟ

ก่อนที่จะมาเข้าเรื่องของวงจรกรองแบบแอกทีฟ ขอแนะนำให้รู้จักกับโครงข่าย RC

(RC network) เสียก่อน โครงข่ายนี้สามารถนำมาใช้เป็นวงจรอินทิเกรเตอร์ซึ่งจัดได้ว่าเป็นวงจรกรองประเภทหนึ่งเหมือนกัน ดังแสดงในรูปที่5.13 ดังนั้น จึงนับได้ว่า วงจรอินทีเกรเตอร์เป็นวงจรกรองแบบพาสซีฟ แต่เนื่องจากการใช้โครงข่าย RC นี้เป็นวงจรกรองจะเกิดการผิดเพี้ยน (distortion) และการสูญเสียของสัญญาณสูงมาก จึงมีการนำอุปกรณ์แอกทีฟ (อุปกรณ์ที่แสดงคุณสมบัติเมื่อมีพลังงานมากระตุ้น เช่น ทรานซิสเตอร์ ไดโอด ออปแอมป์ เป็นต้น) จำพวกออปแอมป์มาช่วยลดความผิดเพี้ยนของสัญญาณ และยังสามารถเพิ่มอัตราการขยายให้แก่วงจรได้ด้วย ดังจะเห็นในตัวอย่างรูปที่ 5.14



รูปที่ 5.13 วงจรอินทิเกรเตอร์ที่ประกอบจากโครงข่าย RC อย่างง่าย ๆ



รูปที่ 5.14 แสดงตัวอย่างการใช้งานวงจรกรองแบบแอกทีฟหรือวงจรกรอง RC

การที่โครงข่าย RC ถูกต่อใช้งานร่วมกับออปแอมป์ จึงกลายเป็นวงจรกรองอีกแบบหนึ่ง นั่นคือ วงจรกรองแบบแอกทีฟ ซึ่งหมายถึง วงจรกรองที่จะทำงานได้ก็ต่อเมื่อมีพลังงานมากระตุ้น ซึ่งก็คือไฟเลี้ยงของออปแอมป์นั่นเอง ข้อดีของวงจรแบบนี้ คือ ปรับแต่งได้ง่าย การออกแบบก็ไม่ซับซ้อน มีเสถียรภาพในการทำงานสูงขนาดเล็กแถมยังราคาถูก นอกจากนี้ยังสามารถหลีกเลี่ยงการใช้อุปกรณ์พาสซีฟที่มีความยุ่งยากในการใช้งาน ซึ่งก็คือ ตัวเหนี่ยวนำ

(inductor) โดยการใช้ออปแอมป์ช่วยจ่ายพลังงานให้กับวงจรแทน ดังนั้นวงจรกรองแบบแอกทีฟนี้จึงเป็นที่นิยมใช้กันมาก ปัจจุบันก็ได้มีการผลิตวงจรกรองแบบนี้ในรูปของวงจรรวมหรือไอซีที่เรารู้จักกันดีให้มีขนาดเล็กลงไปอีก

แต่ในทางตรงกันข้าม วงจรกรองแบบแอกทีฟจะตอบสนองความถี่ได้ไม่สูงนักเมื่อเทียบกับวงจรแบบพาสซีฟ นอกจากนี้ยังต้องใช้ไฟเลี้ยงในการทำงาน อย่างไรก็ตามข้อเสียเหล่านี้ก็มิได้เป็นอุปสรรคในการใช้งานนัก

5.5 การเลือกใช้ชนิดของวงจรกรองในระบบต่างๆ

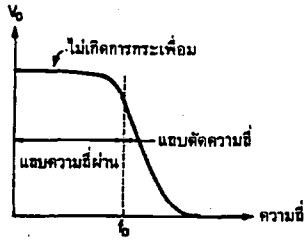
เนื่องจากในระบบงานต่างๆต้องการใช้คุณสมบัติของวงจรกรองที่แตกต่างกัน ดังนั้นจึงต้องมีวงจรกรองชนิดต่างๆ ไว้สำหรับแก้ไขปัญหาเฉพาะในงานแต่ละประเภท เช่น บางประเภทต้องอาศัยการส่งผ่านของแถบความถี่ที่ใช้ในระยะเวลาสั้น ก็ต้องเลือกใช้วงจรกรองชนิดชีบีเชฟ (Chebyshev filter) เพื่อให้ได้ค่าโรลออฟสูงๆ

งานบางประเภทต้องการความราบเรียบของแถบความถี่มาก ก็จะเลือกใช้วงจรกรองชนิดบัตเตอร์เวิร์ธ (Butterworth filter) ซึ่งจะทำให้ความราบเรียบสูง มีแถบกระเพื่อมต่ำ

สำหรับในงานที่ต้องการใช้วงจรกรองในลักษณะตัดแถบความถี่ (band stop) ก็จะเลือกใช้วงจรกรองชนิดอีเลปติก (eleptic filter) ซึ่งจะกำจัดการกระเพื่อมที่เกิดขึ้นในช่วงแถบตัดความถี่ นอกจากนี้ยังมีวงจรกรองความถี่อื่นๆ อีกมาก เช่น วงจรกรองชนิดพาราโบลิก (parabolic filter) ชนิดเบสเซล (Bessel filter) ชนิดกัสเซียน (Guasian filter) เป็นต้น แต่วงจรกรองที่นิยมใช้กันมากที่สุดมีอยู่ 3 ชนิด ได้แก่ วงจรกรองชนิดบัตเตอร์เวิร์ธ ชนิดชีบีชีฟ และชนิดเบสเซล ซึ่งจะหยิบยกมาพูดไว้ในที่นี้

5.6 วงจรกรองชนิดบัตเตอร์เวิร์ธ

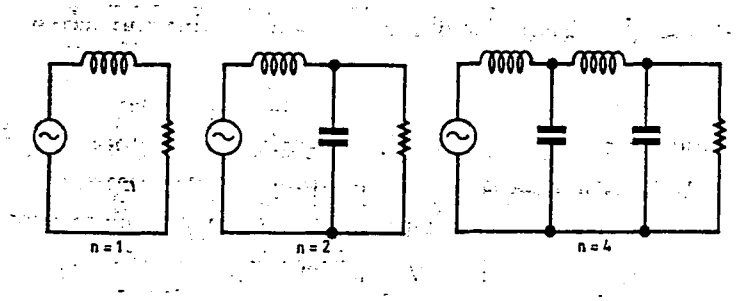
ลักษณะพิเศษของวงจรกรองชนิดนี้ คือ ให้อัตราขยายของสัญญาณเท่ากันทุกความถี่ให้ผ่านได้ ดังนั้นช่วงกระเพื่อมที่เกิดขึ้นจะมีค่าน้อยมาก ดูจากรูปที่ 5.15 นับได้ว่าวงจรกรองชนิดบัตเตอร์เวิร์ธมีความราบเรียบของแถบความถี่สูงที่สุด ในบรรดาวงจรกรองชนิดต่างๆ



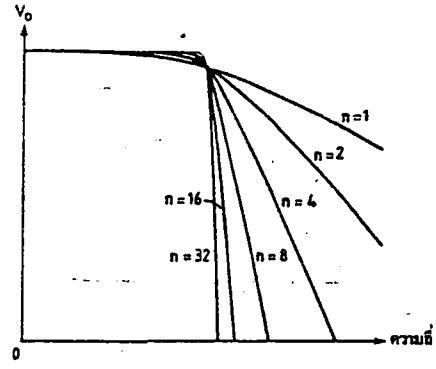
รูปที่ 5.15 แสดงรูปคลื่นของวงจรกรองชนิดบัตเตอร์เวิร์ธ จะเห็นได้ว่ามีความราบเรียบของแถบความถี่สูงมาก

ในทางตรงกันข้ามการส่งผ่านแถบความถี่ของวงจรกรองชนิดนี้ทำได้ไม่ดีนัก แต่สามารถแก้ไขได้โดยการเพิ่มจำนวนอุปกรณ์เข้าไปอีก

ถ้าให้จำนวน n แทนจำนวนอุปกรณ์ในวงจรกรอง โดยนับจากจำนวนของตัวเหนี่ยวนำหรือตัวเก็บประจุรวมกัน ดังตัวอย่างในรูปที่ 5.16 วงจรกรองมีค่า n เท่ากับ 1, 2 และ 4 ตามลำดับจะมีลักษณะการส่งผ่านไม่เหมือนกัน ดูจากรูปที่ 5.17 จะเห็นได้ว่า ค่า n ยิ่งมากเท่าใดการส่งผ่านก็จะมีลักษณะชันขึ้นเท่านั้น ค่าของ n ประจำวงจรนั้น ก็คือ ลำดับของการกรองนั่นเอง อย่างไรก็ตาม การเพิ่มจำนวนอุปกรณ์เข้าไปจะทำให้การออกแบบซับซ้อนขึ้นเป็นเงาตามตัว



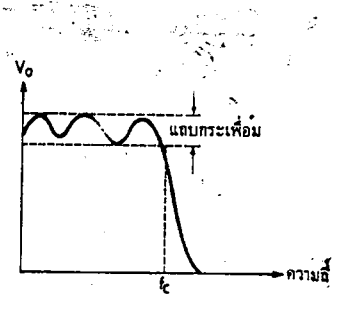
รูปที่ 5.16 แสดงค่า n แทนจำนวนอุปกรณ์เพื่อใช้ประกอบรูป 5.17



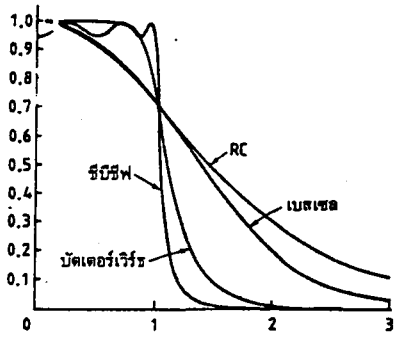
รูปที่ 5.17 แสดงลักษณะการส่งผ่านของวงจรกรองต่อจำนวนอุปกรณ์ที่เพิ่มเข้าไป

5.7 วงจรกรองชนิดซีบีซีฟ

ลักษณะของวงจรกรองชนิดนี้มีข้อดีตรงที่การส่งผ่านของแถบความถี่มีความชันมาก หรือมีค่าโรลออฟสูงนั่นเอง ดังรูปที่ 5.18 แต่ปัญหาที่เกิดขึ้น ก็คือ มีระดับของการกระเพื่อมสูงกว่าวงจรกรองชนิดอื่นๆ พิจารณาจากรูปที่ 5.19 จะเห็นได้ว่า วงจรกรองชนิดซีบีซีฟมีช่วงกระเพื่อมสูงสุด และ ชนิดบัตเตอร์เวิร์ธจะมีช่วงการกระเพื่อมต่ำสุด จนแทบจะไม่มีการส่งผ่านแถบความถี่ทำให้วงจรกรองชนิดนี้ได้ชื่อว่า มีค่าโรลออฟสูงที่สุดในบรรดาวงจรกรองชนิดอื่นๆ



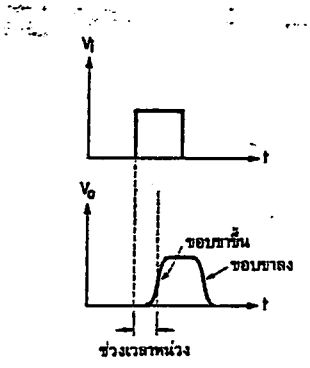
รูปที่ 5.18 แสดงลักษณะของวงจรกรองชนิดซีบีซีฟ ซึ่งให้ค่าโรลออฟสูงที่สุด



รูปที่ 5.19 เปรียบเทียบลักษณะการตอบสนองความถี่ของวงจรกรองชนิดต่างๆ

5.8 วงจรกรองชนิดเบสเซล

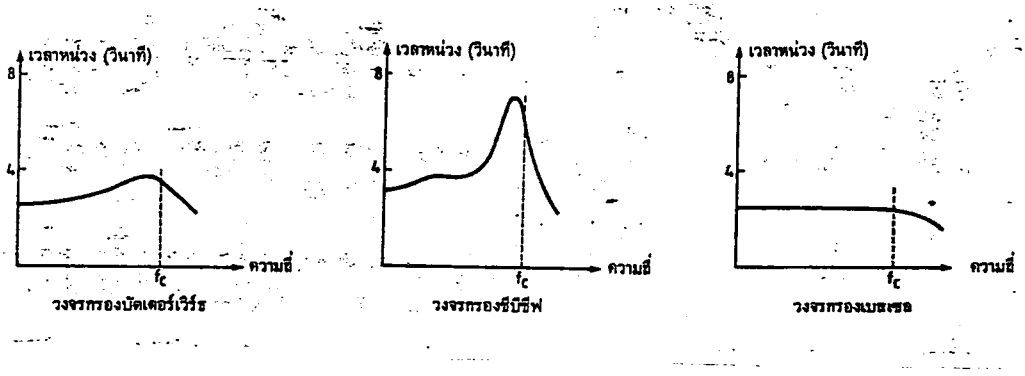
ก่อนที่จะกล่าวถึงลักษณะของวงจรกรองชนิดนี้ จะขออธิบายปัญหาอีกลักษณะหนึ่งของวงจรกรอง นั่นคือ การผิดเพี้ยนของสัญญาณอันเนื่องมาจากการเลื่อนเฟสและการหน่วง พิจารณาจากรูปที่ 5.20 เปรียบเทียบกับสัญญาณที่เข้ามาทางอินพุท (V_i) กับสัญญาณเอาท์พุท (v_o) ของวงจรกรองจะเห็นได้ว่า ขณะที่สัญญาณที่เข้าไปทางอินพุทและออก ไปทางเอาท์พุทของวงจรกรองจะเกิดช่วงเวลาหนึ่งเรียกว่า ช่วงเวลาหน่วง ช่วงเวลานี้เองที่ทำให้เฟสของสัญญาณเอาท์พุทเลื่อนไปและเกิดความผิดเพี้ยนของรูปคลื่นทางเอาท์พุท สังเกตได้จากช่วงขอบขาขึ้นและลงของพัลส์ทางเอาท์พุทจะมีช่วงเวลาในการขึ้นลงไม่เท่ากับสัญญาณทางอินพุท ถ้าสัญญาณมีความถี่สูงการผิดเพี้ยนเนื่องจากการเลื่อนเฟสจะมีค่ามากขึ้นตามไปด้วย



รูปที่ 5.20 แสดงการเกิดช่วงเวลาหน่วงของรูปคลื่นทางเอาท์พุท

ในการใช้งานกับความถี่ไม่สูงนัก การผิดเพี้ยนของสัญญาณเนื่องจากการเลื่อนเฟสมีเพียงเล็กน้อย ถ้าการเลื่อนเฟสนั้นมีช่วงที่เท่ากันตลอดทุกความถี่ ในการใช้งานจะไม่มีปัญหา แต่ถ้าการเลื่อนเฟสของวงจรกรองมีช่วงที่ไม่เท่ากัน ก็จะเกิดการผิดเพี้ยนของสัญญาณเนื่องจากการหน่วง (delay distortion) ซึ่งจะเป็นปัญหาที่ใหญ่มากในระบบงานสื่อสารข้อมูลทางดิจิตอล เพราะจะทำให้ข้อมูลเกิดการผิดพลาดได้

วงจรกรองชนิดเบสเซลจึงถูกนำมาใช้งานเพื่อแก้ปัญหาข้างต้น ดูจากลักษณะรูปคลื่นในรูปที่ 5.21 เปรียบเทียบช่วงเวลาหน่วงต่อความถี่ที่เปลี่ยนไปของสัญญาณในวงจรกรองชนิดต่าง ๆ จะเห็นได้ว่า วงจรกรองชนิดเบสเซลจะมีช่วงเวลาหน่วงสม่ำเสมอ ความผิดเพี้ยนของสัญญาณเนื่องจากการหน่วงจึงไม่เกิดขึ้น



รูปที่ 5.21 แสดงการเปรียบเทียบระหว่างวงจรกรองความถี่ชนิดต่าง ๆ

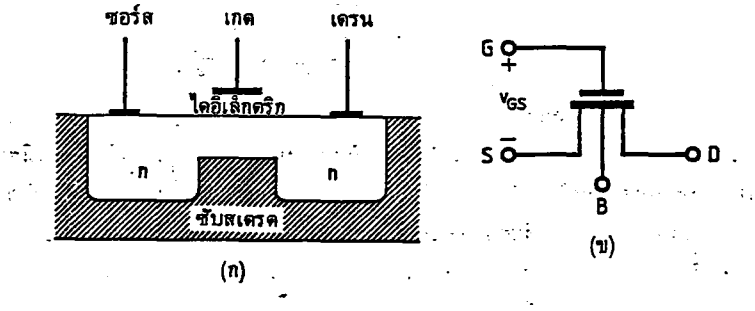
5.9 สวิตช์คาปาซิเตอร์ฟิลเตอร์ (Switched Capacitor Filter)

จากที่ได้กล่าวข้างต้นว่า วงจรกรองความถี่แบบแอกทีฟจะประกอบไปด้วย ตัวต้านทาน ตัวเก็บประจุ และส่วนที่สำคัญอย่างหนึ่งก็คือ ไอซีออปแอมป์ ค่าความถี่ตัด จะถูกกำหนดด้วยค่าความต้านทานและตัวเก็บประจุที่ประกอบขึ้น ถ้าค่าของอุปกรณ์ตัวใดตัวหนึ่งเกิดเปลี่ยนไป จะทำให้ค่าความถี่กลางก็จะเปลี่ยนไปด้วย และในการประกอบเป็นวงจรกรองความถี่ในแต่ละแบบมีข้อแตกต่างกันมาก ทำให้ยุ่งยากต่อการประกอบ แต่ด้วยเทคโนโลยีที่ทันสมัย ปัจจุบันนี้ได้มีไอซีทำหน้าที่กรองความถี่โดยเป็นการรวมเอา ตัวต้านทาน ตัวเก็บประจุ และออปแอมป์เข้าไว้ในตัวเดียวกัน สามารถนำมาทำวงจรกรองความถี่ได้หลายแบบ รวมทั้งสามารถเปลี่ยนค่าความถี่กลางได้โดยไม่ต้องเปลี่ยนค่าอุปกรณ์ใด ๆ ซึ่งอุปกรณ์ตัวนี้เรียกว่า สวิตช์คาปาซิเตอร์ฟิลเตอร์ (switched-capacitor filter)

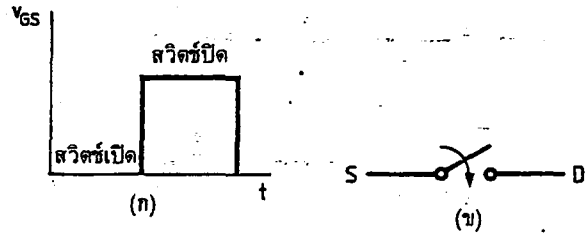
พื้นฐานการพัฒนาไอซีกรองความถี่แบบสวิตช์คาปาซิเตอร์นี้มาจากการใช้เทคโนโลยีของมอส (MOS : Metal Oxide Semiconductor) เทคโนโลยีนี้ใช้ความสัมพันธ์ง่าย ๆ ของตัวเก็บประจุกับสวิตช์ และรวมไปถึงออปแอมป์ด้วย เนื่องจากการจะสร้างตัวต้านทานบรรจุลงในตัวไอซีให้มีค่าถูกต้องแน่นอนนั้นทำได้ยาก แต่เป็นที่ทราบกันว่า ความต้านทานสามารถแทนด้วยการทำงานของมอสสวิตช์ (MOS switch) ซึ่งทำงานร่วมกับตัวเก็บประจุ เจ้าอุปกรณ์ทั้งสองตัวนี้นับว่าเป็นส่วนสำคัญของวงจรทั้งหมด และเป็นวงจรกรองความถี่แบบใหม่ล่าสุดนับตั้งแต่มีการปรับปรุงพัฒนาของไอซี

ก่อนอื่นขอให้ทำความเข้าใจถึงการทำงานของมอสสวิตช์เสียก่อน จากรูปที่ 5.22(ก) เป็นภาพตัดขวางของมอสที่เป็นทรานซิสเตอร์ชนิดหนึ่ง ที่เรียกกันว่า มอสเฟต (MOSFET) รูปที่ 5.22(ข) เป็นสัญลักษณ์ของทรานซิสเตอร์ชนิดนี้ ซึ่งประกอบด้วยขาเกต (Gate), ซอร์ส

(Source), เดรน (Drain) และ ซับสเตรต (Substate) ซึ่งก็คือตัวถังของมอสเฟต V_{GS} เป็นแรงดันระหว่างเกตกับซอร์ส ปกติจะมีค่าเป็นศูนย์หรือสูงกว่า V_{cr} (cutoff region : แรงดันที่ทำให้กระแสเดรนคงที่ ปกติมีค่า 1-2 โวลต์) V_{GS} นี้จะเป็นตัวควบคุมให้เฟตตัวนี้เปิดหรือปิด จึงเรียกว่า มอสสวิตช์ โดยส่วนที่เป็นสวิตช์คือซอร์ส และเดรน ซึ่งมีค่าความต้านทานเป็น R_{DS}(Drain-Source resistance) เมื่อมอสสวิตช์อยู่ในสภาวะจาก (off -mode: $V_{GS} < V_{cr}$) ความต้านทาน R_{DS} จะมีค่าสูงมากประมาณ 100 - 1000 เมกะโอห์ม ในขณะที่มอสสวิตช์เมื่ออยู่ในสภาวะต่อ (on-mode: $V_{GS} > V_{cr}$) ค่าความต้านทาน R_{DS} จะมีค่าลดลงมาถึง 10 กิโลโอห์ม (ค่านี้ขึ้นกับขนาดของมอส) อัตราส่วนของค่าความต้านทานทั้งสองนี้มีค่าประมาณ 10^{15} เท่า ตารางที่ 5.1 เป็นการเปรียบเทียบการทำงานของมอสสวิตช์ทั้งสองสภาวะ รูปที่ 5.23 แสดงรูปร่างแรงดัน V_{GS} กับการแทนมอสสวิตช์ขั้วเดียวทางเดียว รูปที่ 5.24 เป็น การป้อน V_{GS} ด้วยสัญญาณนาฬิกาที่มีคาบเวลาแน่นอน T_c สัญลักษณ์ของสัญญาณนาฬิกา คือ ϕ มอสสวิตช์จะเปิด-ปิดด้วยคาบเวลาคงที่ เมื่อสัญญาณนาฬิกาคงที่



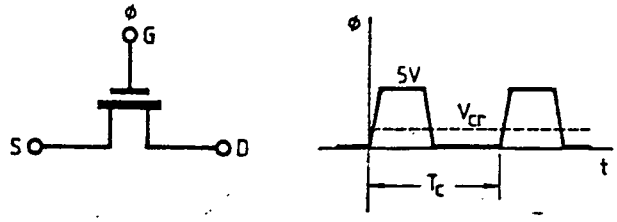
รูปที่ 5.22 โครงสร้างภายในของทรานซิสเตอร์



รูปที่ 5.23 แสดงมอสสวิตช์สภาวะปิดและเปิดที่ขึ้นกับแรงดัน V_{GS}

(ก) แรงดัน V_{GS}

(ข) แทนสภาวะด้วยสวิตช์

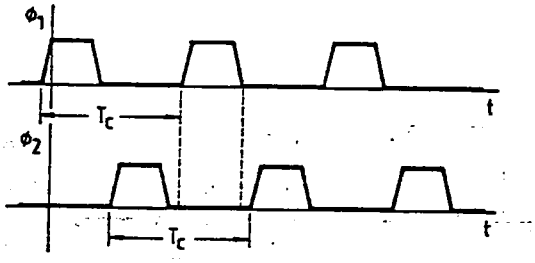


รูปที่ 5.24 แสดงรูปร่างสัญญาณนาฬิกาที่ใช้กระตุ้นมอสก็คือพัลส์นั่นเอง

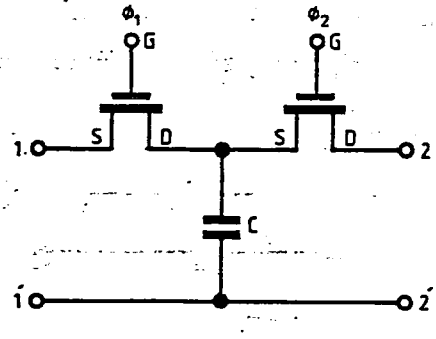
สภาวะ	สถานะทางลอจิก	ความต้านทานสมมูลย์	สถานะทางไฟฟ้า
$V_{GS} > V_{cr}$	on	10k	ลัดวงจร
$V_{GS} < V_{cr}$	off	100M	เปิดวงจร

ตารางที่ 5.1 แสดงการแทนสภาวะการทำงานของมอสด้วยตัวต้านทาน

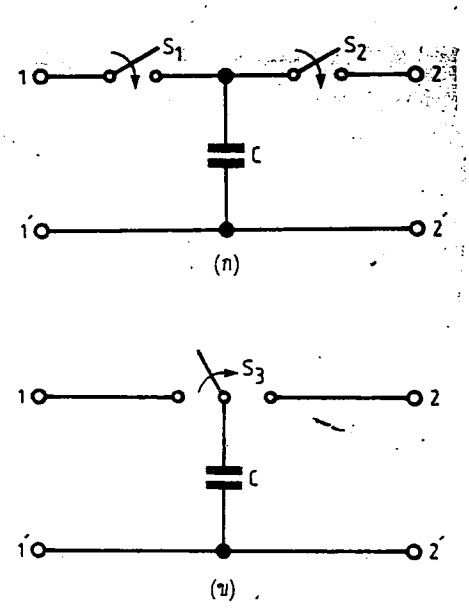
รูปที่ 5.25 แสดงสัญญาณนาฬิกา 2 สัญญาณ คือ ϕ_1 และ ϕ_2 โดยมีความถี่เท่ากันแต่มีรูปคลื่นล้าหลังกันอยู่ครึ่งคาบ และไม่มีส่วนที่เหลื่อมล้ำกัน (nonoverlap) คือ เมื่อ ϕ_1 อยู่ในสภาวะเปิด ϕ_2 จะอยู่ในสภาวะปิดสลับกันไปตลอด ถ้านำมอสสองตัวมาต่อกัน ดังรูปที่ 5.26 และควบคุมด้วยสัญญาณนาฬิกาของรูปที่ 5.25 จะได้ว่า ระหว่างจุดที่ 1 และจุดที่ 2 จะไม่มีการต่อถึงกันเลย เพราะจะมีมอสสวิตช์ตัวใดตัวหนึ่งเปิดในขณะที่อีกตัวปิด รูปที่ 5.27 เป็นการแทนมอสด้วยสวิตช์สองตัว รูปที่ 5.27(ก) นั้น สวิตช์ S_1 และ S_2 จะเปิดและปิดสลับกันตามสัญญาณนาฬิกา ส่วนรูปที่ 5.27(ข) เป็นการยุบสวิตช์ 2 ตัว ในรูป (ก) ให้เหลือตัวเดียวคือ S_3 โดยใช้สวิตช์ขั้วเดียวสองทาง



รูปที่ 5.25 แสดงรูปคลื่นสัญญาณนาฬิกา ทั้ง 2 สัญญาณ



รูปที่ 5.26 แสดงการต่อมอสกับตัวเก็บประจุ



รูปที่ 5.27 แสดงการทำงานของมอสด้วยสวิตช์

(ก) ใช้สวิตช์ทางเดียวแทนมอสแต่ละตัว

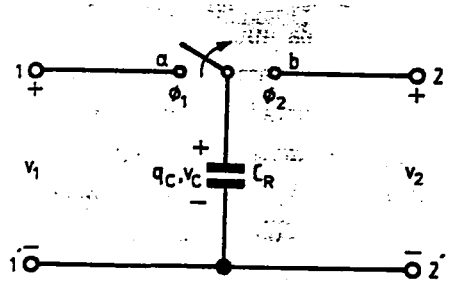
(ข) ใช้มอสขั้วเดียวสองทางแทนมอสทั้ง 2 ตัว

5.10 การทำงานของสวิตช์คาปาซิเตอร์

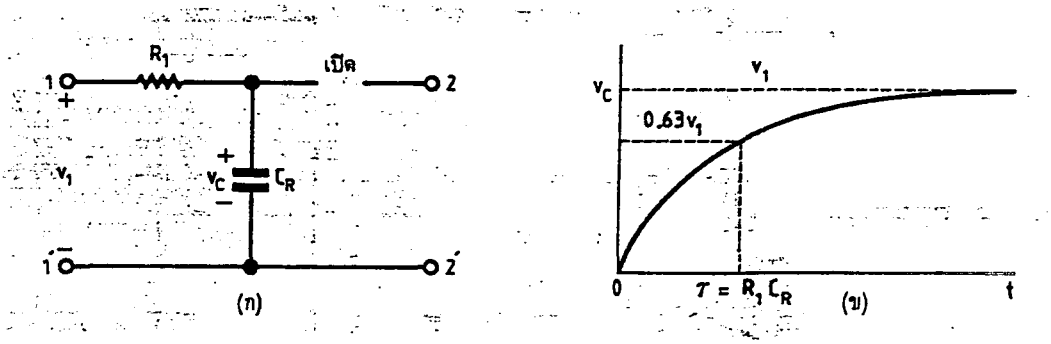
รูปที่ 5.28 เป็นรูปเดียวกับรูปที่ 5.27(ข) โดยมีตัวเก็บประจุ C_p และสวิตช์ขั้วเดียวสองทางโยกไปมาด้วยความถี่นาฬิกา $f_c = 1 / T_c$ V_1 คือสัญญาณที่ป้อน ส่วน V_2 คือแรงดันทางเอาท์พุท สมมุติว่า แรงดัน $V_1(t)$ เปลี่ยนแปลงไปตามเวลาหรือเป็นไฟสลับ เริ่มต้นให้สวิตช์โยกไปตำแหน่ง a เราสามารถจะแทนสภาพนี้ได้ด้วยรูปที่ 5.29(ก) R_1 คือ ความ

ต้านทานของมอสตัวที่ 1 ขณะอยู่ในสภาวะต่อ มอสตัวที่ 2 อยู่ในสภาวะจาก ทำให้มีความต้านทานสูงมากเหมือนกับเปิดวงจร ถ้า $V_1(t)$ เปลี่ยนแปลงช้ามากถือว่าคงที่ ตัวเก็บประจุ C_R จะถูกประจุไฟ ซึ่งมีลักษณะรูปร่างดังรูปที่ 5.29(ข) จากสูตรคาบเวลาคงที่ในการประจุ $T = R_1C_R$ ค่าความต้านทาน R_1 ประมาณ 10 กิโลโอห์ม ตัวเก็บประจุ C_R ปกติมีค่า 10^{-12} ฟารัด ค่าแรงดันที่ประจุได้ที่ 63 เปอร์เซ็นต์ จะใช้เวลา

$$T = R_1C_R = 10^4 \times 10^{-12}$$
$$= 10^{-8} \text{ วินาที}$$



รูปที่ 5.28 แสดงการทำงานของสวิตช์คาปาซิเตอร์



รูปที่ 5.29 แสดงการแทนการทำงานของสวิตช์คาปาซิเตอร์ด้วยตัวต้านทาน

(ก) วงจรสมมูลของสวิตช์คาปาซิเตอร์ขณะสวิตช์โยกไปที่ตำแหน่ง a

(ข) ค่าแรงดันที่ถูกประจุเข้าไปในตัวเก็บประจุ

เราจะสมมุติว่า ขณะที่ตัวเก็บประจุกำลังประจุไฟอยู่นั้น ค่า V_1 เปลี่ยนแปลงน้อยมาก ขณะเดียวกันเมื่อโยกไปทางตำแหน่ง b แรงดันคายประจุจะเป็น

$$q_c = C_R(V_1-V_2) \qquad \text{.....(1)}$$

คายเวลาที่คายประจุเป็น T_c จะได้กระแสเฉลี่ยเป็น

$$i_{av}(t) = \frac{q}{t}$$

$$= \frac{C_R(V_1-V_2)}{T_c} \qquad \text{.....(2)}$$

ขณะเดียวกันเราสามารถจะแทนด้วยค่าความต้านทานสมมูล $R = \frac{V}{I}$ จะได้

$$R_c = \frac{V_1-V_2}{i(t)}$$

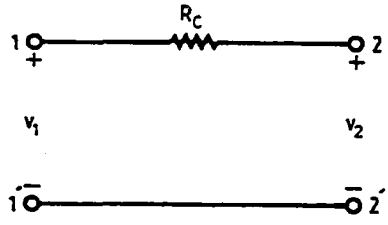
$$= \frac{T_c}{C_R}$$

$$= \frac{1}{f_c C_R} \qquad \text{.....(3)}$$

ดังนั้น จาก (2) และ (3) จะได้

$$i_{av}(t) = \frac{1}{R_c}(V_1-V_2) \qquad \text{.....(4)}$$

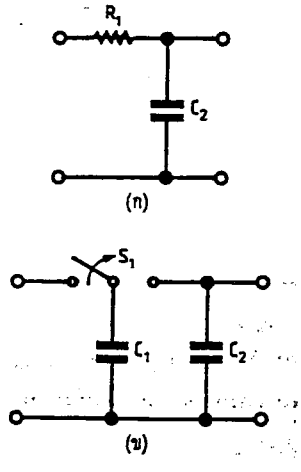
เราสามารถแทนค่าในสมการที่ 4 ได้ด้วยรูปที่ 5.30 หรือกล่าวได้ว่า สวิตช์คาปาซิเตอร์จากรูปที่ 5.28 นั้น สามารถแทนได้ด้วยความต้านทานสมมูลดังรูปที่ 5.30 สำหรับการประมาณค่าจากสมการที่ 3 นั้น จำเป็นต้องให้ค่าของความถี่นาฬิกามีค่าสูงกว่าความถี่ V_1 และ V_2 มากๆ แต่ค่าจะต้องอยู่ในช่วงที่ยอมรับได้ คือ C_R มีค่าประมาณ 10^{-12} ฟารัด และค่า R_c จะอยู่ในช่วง 10 เมกะโอห์ม



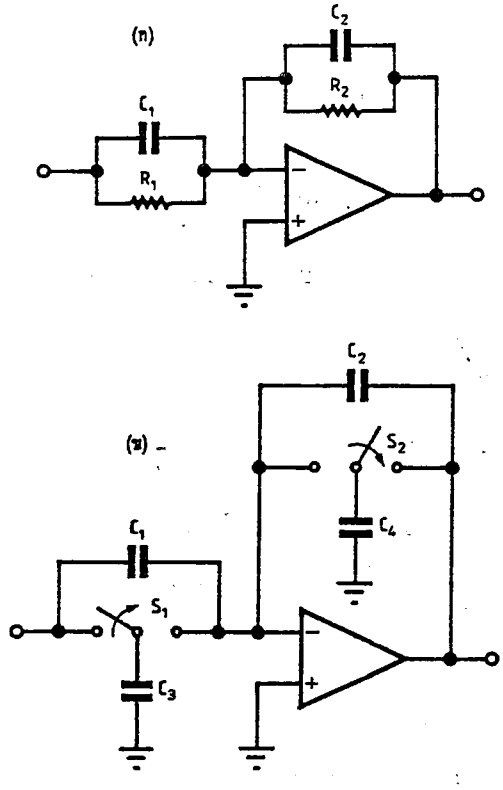
รูปที่ 5.30 แสดงวงจรสมมูลของสวิตช์คาปาซิเตอร์

5.11 วงจรกรองความถี่จากสวิตช์คาปาซิเตอร์

อย่างที่ได้กล่าวไว้ข้างต้นว่า สวิตช์คาปาซิเตอร์สามารถปฏิบัติตัวเหมือนเป็นตัวต้านทานตัวหนึ่งคือ $R=1/f_c C_R$ และค่าความต้านทานนี้ก็ยังเปลี่ยนแปลงตามค่าความถี่ของสัญญาณนาฬิกาในการประยุกต์เข้ากับวงจรกรองความถี่ จะทำได้โดยแทนตัวต้านทานด้วยสวิตช์คาปาซิเตอร์ ดังตัวอย่างในรูปที่ 5.31(ก) ซึ่งเป็นวงจรกรองแบบให้ความถี่ต่ำผ่าน โดยการใช้สวิตช์คาปาซิเตอร์แทนตัวต้านทาน R_1 ดังรูป 5.31(ข) ค่าความถี่กลางสามารถเปลี่ยนได้โดยปรับความถี่นาฬิกา ส่วนรูปที่ 5.32(ก) เป็นวงจรกรองแบบตัดความถี่ (bandstop) รูปที่ 5.32(ข) เป็นการใช้สวิตช์คาปาซิเตอร์แทนตัวต้านทาน R_1 และ R_2 (C_3 แทน R_1 และ C_4 แทน R_2) และกับวงจรอื่นๆ ก็สามารถแทนตัวต้านทานด้วยสวิตช์คาปาซิเตอร์ได้ แต่ค่าความต้านทานจะต้องอยู่ในช่วงที่กำหนดเท่านั้น



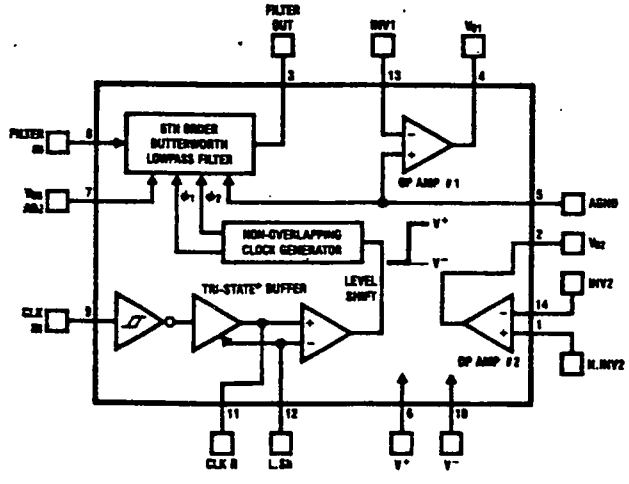
รูปที่ 5.31 แสดงการใช้สวิตช์คาปาซิเตอร์กับวงจรกรองความถี่ประเภทพาสซีฟ



รูปที่ 5.32 แสดงการใช้สวิตช์คาปาซิเตอร์กับวงจรกรองความถี่ประเภทแอกทีฟ

5.12 ไอซีสำเร็จรูป

จากรูปที่ 5.33 แสดงให้เห็นถึงรูปลักษณะของไอซีกรองความถี่ต่ำผ่านแบบสวิตช์คาปาซิเตอร์ เบอร์ MF6 เป็นชนิดบัตเตอร์เวิร์ธลำดับที่ 6 (forth order) ของ National Semiconductor ไอซี MF6 นี้มีอยู่ 2 รุ่นคือ MF6-50 และ MF6-100 ค่าความถี่กลางของวงจรสามารถกำหนดได้โดยการป้อนสัญญาณนาฬิกาจากภายนอก โดยค่าอัตราส่วนระหว่างความถี่ของสัญญาณนาฬิกาต่อความถี่กลางจะมีค่าเป็น 50:1 และ 100:1 ตามรุ่นที่ระบุ รายละเอียดของวงจรติดตามได้จากบทถัดไป



รูปที่ 5.33 แสดงผังการทำงานของ MF6

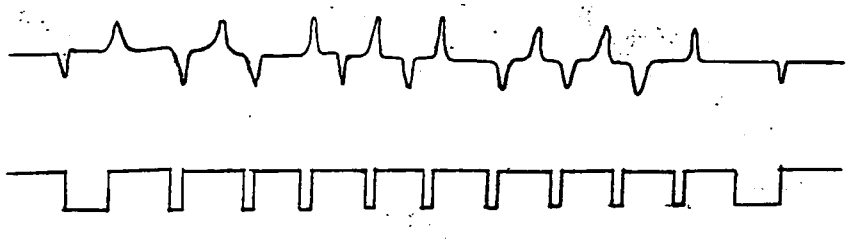
บทที่ 6

หลักการและโครงสร้างของระบบ

ในบทนี้ เราจะทำการศึกษาถึงหลักการและโครงสร้าง รวมทั้งอธิบายจุดประสงค์และการทำงานของแต่ละส่วนของวงจรต่างๆ และเพื่อให้ผู้อ่านเข้าใจได้ดีขึ้น จะทำการแยกหลักการของระบบเป็นส่วนๆ ดังนี้

6.1 หลักการและโครงสร้างของระบบอ่านบัตรแม่เหล็ก

ในส่วนนี้เราจะพิจารณากลไกพื้นฐานในการสร้างวงจรอ่านข้อมูลจากบัตรแม่เหล็ก รวมทั้งเทคนิคในการเขียนโปรแกรม เพื่อควบคุมการอ่านให้เป็นไปอย่างมีระเบียบและถูกต้อง ดังที่ได้กล่าวไว้แล้วในบทที่ 2 เกี่ยวกับเทคนิคที่ใช้ในการเข้ารหัสข้อมูลบนบัตรแม่เหล็ก ที่เรียกว่า Two Frequency, Coherent Phase Recording หรือ F2F โดยลักษณะการบันทึกข้อมูลลงบนแทร็คหนึ่งๆ นั้น จะเป็นการรวมกันระหว่างสัญญาณข้อมูลที่เป็นดิจิตอล และสัญญาณนาฬิกาสำหรับกำหนดจังหวะการอ่านสัญญาณข้อมูลนั้น ทำให้สัญญาณรวมที่ได้จากการอ่านโดยตรงจากหัวอ่านในขณะที่มีการรูดบัตรจะเป็นไปตามรูปที่ 6.1 เราอาจจะสังเกตได้ว่า ถ้าสัญญาณบริเวณใดมีการเปลี่ยนแปลงของฟลักซ์อย่างกระทันหันจากเหนือเป็นใต้ หรือจากใต้เป็นเหนือ ซึ่งช่วงดังกล่าวก็คือ ช่วงที่จุดที่สัญญาณมีจุดสูงสุดและต่ำสุดติดกันมาก กรณีนี้จะแทนการบันทึกด้วยข้อมูลที่เป็น "1" ในบริเวณที่ไม่ได้มีการเปลี่ยนแปลงฟลักซ์กระทันหัน ซึ่งก็คือ บริเวณที่ช่วงห่างระหว่างจุดที่มีสัญญาณสูงสุดกับต่ำสุดมีค่าน้อย จะแทนด้วยข้อมูลที่เป็น "0" จากลักษณะที่สังเกตได้ข้างต้นนี้เองเราจะนำไปใช้ในการสร้างวงจรถอดรหัสจากสัญญาณที่อ่านได้จากหัวอ่านไปเป็นข้อมูลที่เป็นดิจิตอลต่อไป



รูปที่ 6.1 แสดงสัญญาณจากหัวอ่านเทียบกับข้อมูลที่บันทึก

การถอดรหัสจากข้อมูลที่เป็นดิจิตอลนั้น จะต้องทำการสร้างวงจรตามลำดับดังนี้

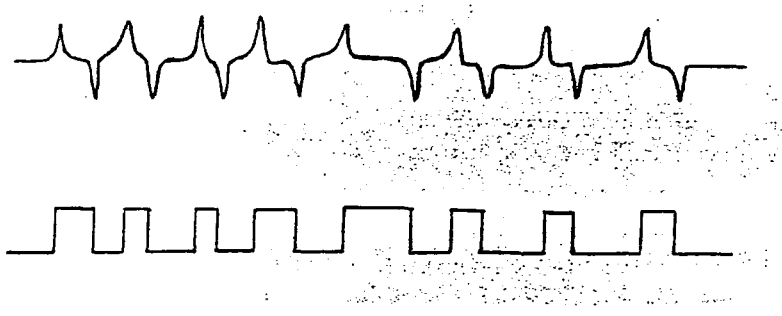
1. วงจรขยายสัญญาณแรงดันไฟฟ้า

สัญญาณแรงดันไฟฟ้าที่อ่านได้จากหัวอ่านแถบแม่เหล็กนี้จะมีค่าน้อยมาก คือ จะมีค่าประมาณ 5-10 mV เราจึงต้องนำมาขยายให้ได้ระดับแรงดันที่สูงขึ้น โดยจะยกให้มีระดับแรงดันในระดับของการประมวลผลด้วยดิจิตอล วงจรขยายที่ใช้จึงต้องมีค่าอัตราการขยายรวมประมาณ 1000 เท่า

2. วงจรสร้างสัญญาณรูปเหลี่ยม

เมื่อได้สัญญาณแรงดันที่ถูกขยายมาแล้ว จะนำมาสร้างเป็นสัญญาณรูปสี่เหลี่ยม ซึ่งทำได้โดยการตรวจสอบขอบสูงสุดและต่ำสุดของสัญญาณ (peak) เพื่อสร้างเป็นตำแหน่งขอบขาของสัญญาณดิจิตอล โดยแนวทางการสร้างวงจรนี้สามารถแบ่งออกเป็น 2 แบบคือ

2.1 ใช้การหน่วงสัญญาณข้อมูลในช่วงเวลาสั้นมาก ๆ แล้วนำข้อมูลจริงมาเปรียบเทียบกับข้อมูลที่ถูกหน่วงเพื่อสร้างเป็นขอบของสัญญาณ ดังรูป



รูปที่ 6.2 แสดงสัญญาณที่ได้จากวงจรสร้างรูปเหลี่ยมโดยใช้การหน่วงข้อมูล

2.2 ใช้วงจรดิฟเฟอร์เรนทิเอเตอร์ (differentiators) และวงจรเปรียบเทียบแรงดันแบบไม่กลับเฟส (Noninverting Voltage Level Detector with Hysteresis) โดยสัญญาณจากหัวอ่านที่เป็นรูปสามเหลี่ยม เมื่อป้อนเป็นอินพุทให้กับวงจรดิฟเฟอร์เรนทิเอเตอร์ สัญญาณเอาท์พุทที่ได้จะเป็นสัดส่วนกับค่าอนุพันธ์ของค่าอินพุท ทำให้ได้เป็นสัญญาณคลื่นรูปเหลี่ยม และใช้วงจรเปรียบเทียบสัญญาณแรงดันเป็นส่วนที่ใช้ปรับปรุงสัญญาณรูปสี่เหลี่ยมให้ได้สัดส่วนที่ดีขึ้น นอกจากนี้ยังช่วยจัดค่าแรงดันของสัญญาณรูปเหลี่ยมให้กับไอซีดิจิตอลที่จะต่อในภาคต่อไป

3. วงจรถอดรหัสข้อมูล

สัญญาณรูปเหลี่ยมที่ได้จะนำมาประมวลผลเพื่อตีความว่าข้อมูลจริงควรเป็นเช่นไร โดยทำการเปรียบเทียบความกว้างของสัญญาณเหลี่ยมที่ได้ ถ้าสัญญาณมีความกว้างน้อย จะแทนด้วยข้อมูล "1" แต่ถ้ารูปเหลี่ยมมีความกว้างมาก ก็จะแทนด้วยข้อมูลที่เป็น "0" ในการสร้างวงจร

จะใช้วงจรนับเพื่อวัดค่าความกว้างระหว่างของขาของสัญญาณแล้วมาเปรียบเทียบกัน จุดนี้เป็นปัญหามาก เนื่องจากถ้าอัตราเร็วในการรูดบัตรไม่เท่ากันตลอดช่วงแล้ว จะทำให้ข้อมูลที่อ่านได้มีค่าผิดพลาดมาก

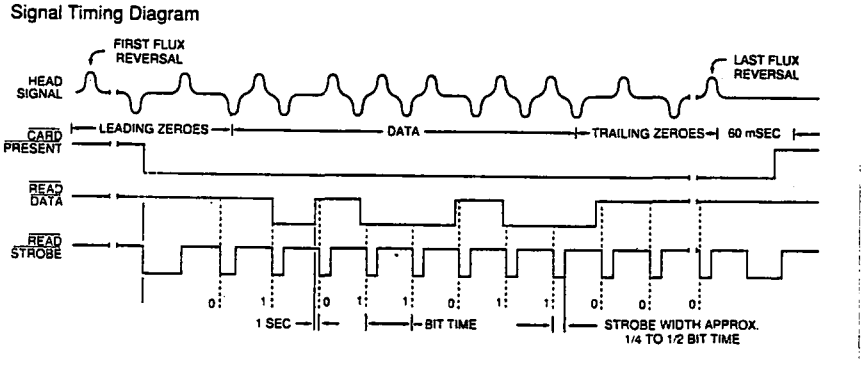
จะเห็นได้ว่า การสร้างวงจรที่จะถอดรหัสข้อมูลที่อ่านได้จากบัตรแม่เหล็กนี้มีความยุ่งยากเนื่องจากจะต้องใช้วงจรหลายส่วนมาประกอบกัน ซึ่งนอกจากจะทำให้วงจรทั้งหมดดูเทอะทะไม่เหมาะสำหรับการใช้งานจริงแล้ว ยังทำให้ของข้อมูลที่อ่านได้เกิดความผิดพลาดได้ง่าย จึงได้มีการสร้างไอซีที่ทำหน้าที่ถอดรหัสเพื่อให้เกิดความสะดวกในการใช้งานมากขึ้น โดยในโครงงานนี้ได้เลือกใช้ ไอซี TB54910P ของ OMRON เป็นหัวใจหลักในการสร้างวงจรถอดรหัสข้อมูล โดยวงจรได้แสดงไว้ในรูปที่ 6.4

จากวงจรดังรูป เราจะทำการป้อนสัญญาณที่อ่านได้จากหัวอ่านเข้าสู่ขา Head ทั้งสองสัญญาณเอาท์พุทที่จะต่อเข้าสู่ส่วนควบคุมมีอยู่ด้วยกัน 3 ขา ดังนี้

1. CARD PRESENT ขณะที่ยังไม่มีการรูดบัตร สัญญาณลอจิกที่ออกจากขานี้จะมีค่าเป็น "1" แต่เมื่อมีการรูดบัตรจะทำให้เกิดสัญญาณลอจิกเป็น "0" สัญญาณนี้จะถูกต่อเข้ากับส่วนควบคุมเพื่อทำการอ่านข้อมูลที่เก็บอยู่จริงบนบัตรแม่เหล็กจากขา DATA

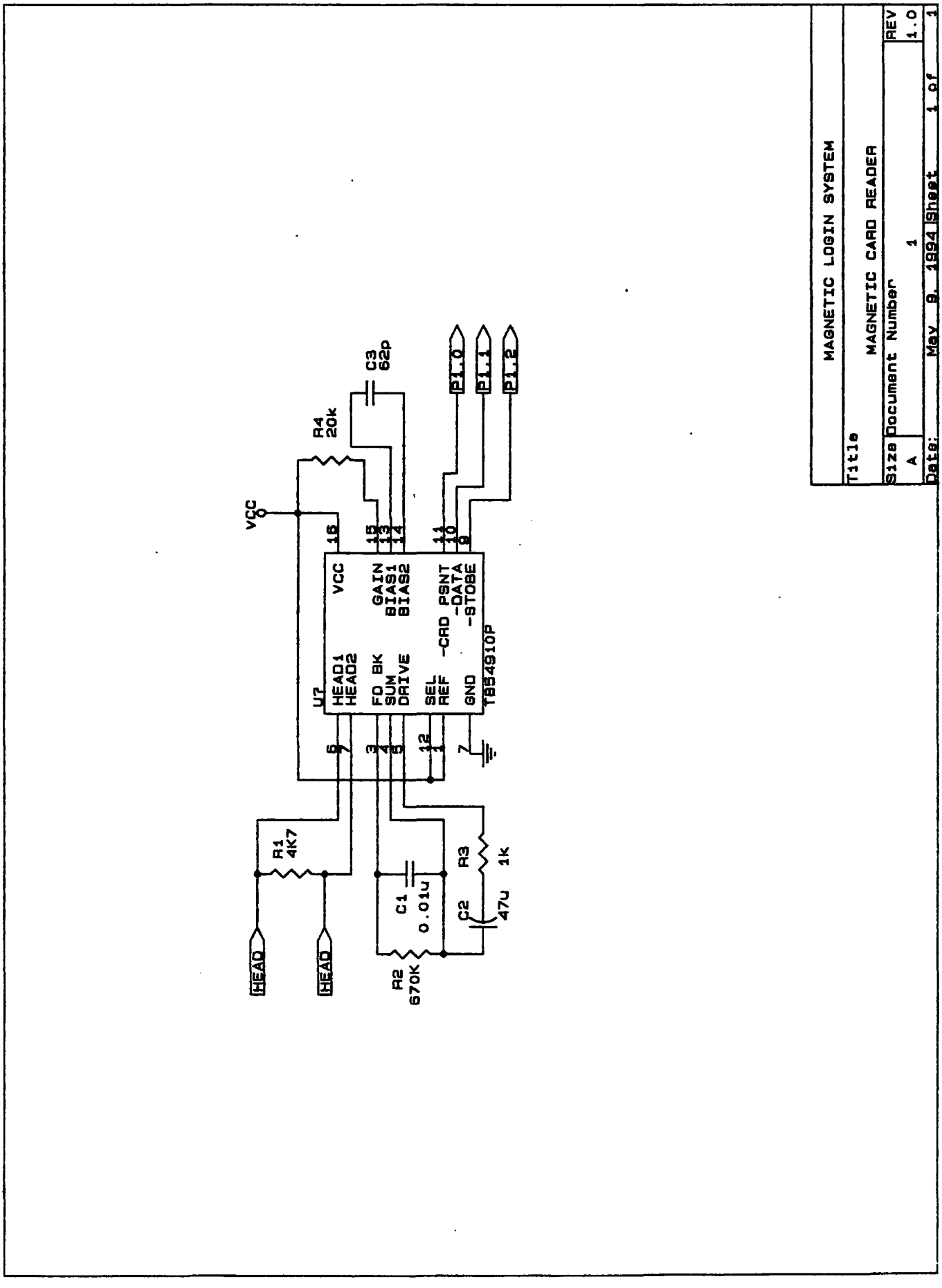
2. DATA จะให้สัญญาณข้อมูลแบบอนุกรม ส่วนควบคุมจะทำการตรวจสอบสัญญาณจากขา STROBE ว่ามีการเปลี่ยนแปลงระดับลอจิกจาก "1" เป็น "0" หรือเป็นสัญญาณนาฬิกาขาลงหรือไม่ ถ้าใช่จะทำการอ่านข้อมูลจากขา DATA นี้เป็นจำนวน 1 บิท แล้วจึงทำการรอจนกว่าจะเกิดสัญญาณนาฬิกาขาลงอีกครั้งจึงจะทำการอ่านข้อมูลในบิทต่อไป

3. STROBE เป็นขาที่ให้สัญญาณนาฬิกาสำหรับกำหนดจังหวะของการอ่านข้อมูลจากขา DATA



รูปที่ 6.3 เปรียบเทียบสัญญาณอินพุทและเอาท์พุทของ TB54910P

รูปที่ 6.4 แสดงวงจรอ่านบัตรแม่เหล็กโดยใช้ TB54910P



ในการที่จะเขียนโปรแกรมเพื่อควบคุมการอ่านข้อมูลจากบัตรแม่เหล็กให้ได้อย่างถูกต้องนั้น จำเป็นที่จะต้องทราบลักษณะในการจัดเก็บข้อมูล โดยส่วนแรกนี้จะสรุปถึงลักษณะที่แตกต่างและเหมือนกันในแต่ละแทร็ค และเหตุผลในการเลือกใช้งานในแทร็คต่างๆ

แทร็คที่ 1

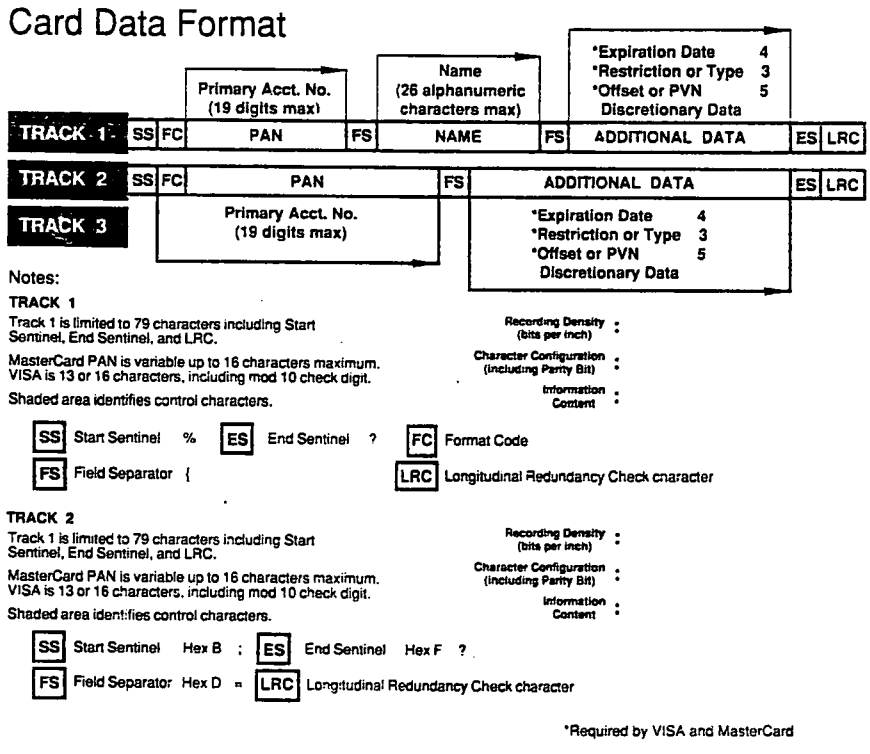
- ข้อมูลมีทั้งตัวอักษรและตัวเลข
- ความหนาแน่น 210 บิทต่อนิ้ว
- 1 ตัวอักษรหรือตัวเลข ประกอบด้วย 6 บิทข้อมูล + 1 บิทพาริตี้คี่
- ข้อมูลจะเริ่มบันทึกจากบิทที่มีนัยสำคัญต่ำสุดก่อน และจบแต่ละชุดด้วยพาริตี้บิท
- จำนวนตัวอักษรหรือตัวเลขทั้งหมดมีได้สูงสุด 79 ตัว

แทร็คที่ 2 และ 3

- ข้อมูลมีแต่ตัวเลข
- แทร็คที่ 2 มีความหนาแน่น 75 บิทต่อนิ้ว แทร็คที่ 3 มีความหนาแน่น 210 บิทต่อนิ้ว
- 1 ตัวเลขประกอบด้วย 4 บิท BCD บวกกับอีก 1 บิทพาริตี้คี่
- ข้อมูลจะเริ่มบันทึกจากบิทที่มีนัยสำคัญต่ำสุดก่อน และจบแต่ละชุดด้วยพาริตี้บิท
- จำนวนตัวเลขทั้งหมดในแทร็คที่ 2 มีได้สูงสุด 40 ตัว ในขณะที่แทร็คที่ 3 มีได้สูงสุดถึง 107 ตัว

ในการใช้งานทั่วๆ ไป แทร็คที่นิยมใช้งานมากที่สุดคือ แทร็คที่ 2 ซึ่งถึงแม้ว่าจำนวนข้อมูลที่เก็บได้ในแทร็คนี้จะมีค่าน้อยที่สุดก็ตาม แต่ก็เพียงพอสำหรับใช้ในการตรวจสอบและยืนยันตัวผู้ถือบัตร โดยในแทร็คที่ 1 นั้นจะใช้ในการเก็บชื่อเจ้าของบัตร ส่วนแทร็คที่ 3 มักจะใช้ในการเก็บข้อมูลที่จะมีการเปลี่ยนแปลงหรือต้องบันทึกใหม่อยู่เสมอ

นอกจากนี้แล้วยังมีการกำหนดตัวอักษรหรือตัวเลขบางตัวสำหรับใช้ในการกำหนดรูปแบบในการบันทึก ได้แก่ start sentinel, end sentinel และ LRC เพื่อทำให้รูปแบบการบันทึกเป็นไปตามมาตรฐานเดียวกัน ซึ่งจะเป็นดังรูปที่ 6.5

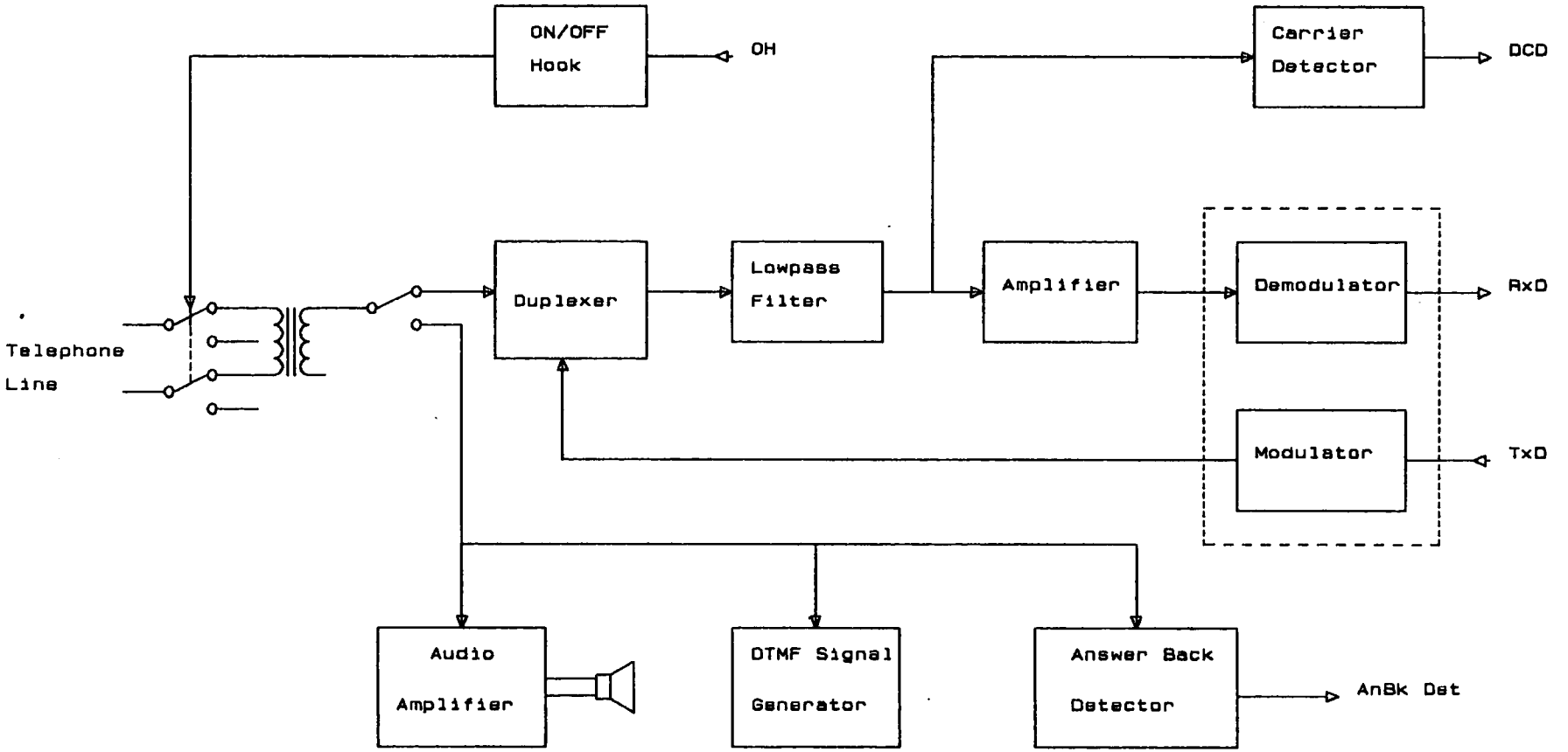


รูปที่ 6.5 แสดงรูปแบบมาตรฐานในการจัดเก็บข้อมูลบนบัตรแม่เหล็ก

ข้อมูลในแต่ละแทร็คจะเริ่มต้นแทร็คจากบิทนำ หรือ synchronize bit ซึ่งเป็นบิทที่มีลอจิกเป็น "0" ติดต่อกันมากกว่า 1 บิท บิทนำนี้นอกจากจะเป็นตัวบ่งบอกว่าผู้ใช้เริ่มทำการเริ่มรูดบัตรแล้ว ยังเป็นการจัดลำดับช่วงการอ่านให้เป็นไปอย่างถูกต้องสำหรับส่วนของวงจรอ่าน ถัดจากบิทนำจะมี start sentinel ซึ่งใช้เป็นตัวบ่งบอกว่าข้อมูลต่อจากนี้จะเป็นข้อมูลที่แท้จริง ตอนท้ายของข้อมูลจะมี end sentinel เป็นตัวแสดงว่าข้อมูลจริงจบเพียงเท่านี้ และติดตามด้วย LRC ที่ใช้ในการตรวจสอบว่า ข้อมูลที่อ่านไปทั้งหมดยกเว้นตัว LRC เองนั้นถูกต้องหรือไม่ การตรวจสอบทำได้โดยการเปรียบเทียบค่า LRC จากการคำนวณค่าพาริตี้ในแนวตั้ง ซึ่งก็คือการทำ exclusive OR กับข้อมูลทั้งหมดตั้งแต่ start sentinel จนถึง end sentinel กับ LRC ที่อ่านได้จากบัตร ถ้าตรงกันแสดงว่าข้อมูลที่ทำการอ่านไปแล้วนั้นมีค่าถูกต้อง ต่อจาก LRC จะมีบิทที่มีลอจิกเป็น "0" ถูกบันทึกจนเต็มส่วนที่เหลือทั้งหมดเหมือนกับกรณีเริ่มต้นที่มีบิทนำ

6.2 หลักการทำงานและโครงสร้างทางวงจรของระบบโมเด็ม

โครงสร้างทั้งหมดสามารถแบ่งออกเป็นบล็อคต่าง ๆ ดังรูปที่ 6.6 และ 6.7 ซึ่งในส่วนของ Terminal และ Host จะมีทั้งบล็อคการทำงานที่เหมือนและต่างกัน โดยวงจรส่วนต่างๆ ของ Terminal และ Host ที่จะอธิบายต่อไปนี้ จะประกอบรวมกันเป็นวงจรที่สมบูรณ์ของแต่ละเครื่อง ซึ่งได้แสดงไว้ในส่วนของภาคผนวก

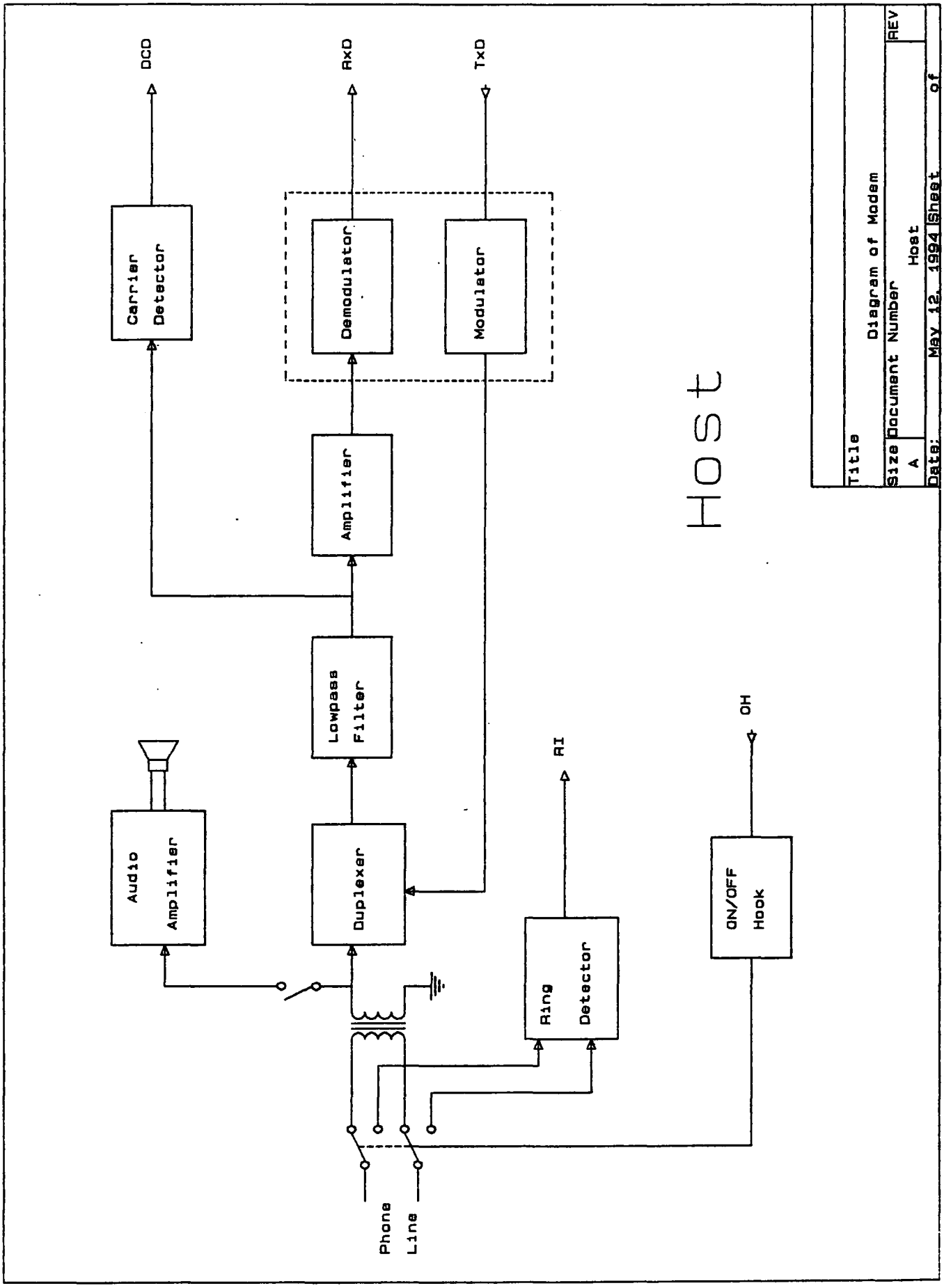


Terminal

Title: Diagram of MODEM		
Size: A	Document Number: Terminal	REV
Date: May 12, 1994	Sheet of	

Fig 6.6 Hardware Terminal

รูปที่ 6.7 แสดงผังการทำงานของ Host



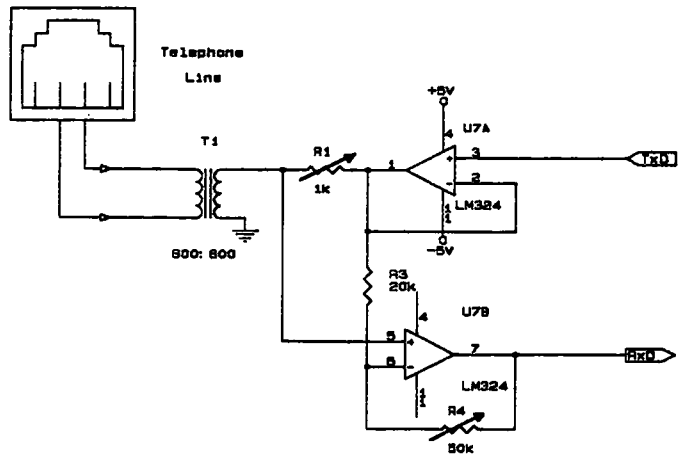
รายละเอียดของวงจรในบล็อกต่าง ๆ และหน้าที่ที่สำคัญ มีดังนี้

1. วงจรแมตชิ่งอิมพีแดนซ์ (Matching Impedance)

ใช้เพียงหม้อแปลง (Transformer) ที่มีค่าอิมพีแดนซ์ทั้งขาเข้าและขาออกเป็น 600 โอห์มเท่ากัน จุดประสงค์ที่ต้องใช้วงจรนี้ในส่วนแรกของโมเด็มคือ เพื่อที่จะทำให้สัญญาณขาเข้าที่ได้รับได้จากชุมสายมีค่ามากที่สุด ทั้งนี้เนื่องจากค่าอิมพีแดนซ์ของสายโทรศัพท์มีค่าประมาณ 600 โอห์ม จึงต้องมีค่าอิมพีแดนซ์ในส่วนรับที่เท่ากันจึงจะเกิดการรับสัญญาณได้ดีที่สุด

2. วงจรดูเพล็กเซอร์ (Duplexer)

เป็นวงจรที่ใช้แยกสัญญาณที่ผ่านมาจากสายโทรศัพท์ให้เป็นส่วนที่รับอย่างเดียว กับส่วนที่ส่งออกเพียงอย่างเดียว ทั้งนี้เนื่องจากโดยปกติแล้วสายโทรศัพท์จะมีเพียง 2 เส้น ทำให้สัญญาณที่ผ่านในสายโทรศัพท์จะมีทั้งสัญญาณพูดออก และสัญญาณที่รับจากคู่สนทนารวมกันอยู่ เมื่อสัญญาณรวมนี้ผ่านวงจรดูเพล็กเซอร์ วงจรนี้จะทำการแยกระหว่างสัญญาณส่วนส่งกับสัญญาณส่วนรับอย่างเด็ดขาด เพื่อให้สัญญาณส่วนที่รับนี้เข้าสู่วงจรดีมอดดูเลชั่นได้โดยไม่มีสัญญาณส่งมารบกวน



รูปที่ 6.8 แสดงวงจรดูเพล็กเซอร์

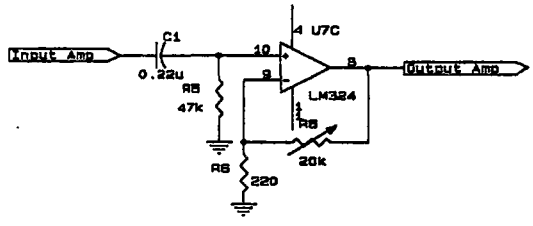
หลักการของวงจรดูเพล็กเซอร์นี้ คือ การกำจัดสัญญาณส่งออกจากสัญญาณรวมที่รับมาได้จากสายโทรศัพท์ การเปรียบเทียบสัญญาณโดยใช้ วงจรขยายส่วนต่าง (differential amplifier) จะทำให้สัญญาณที่ได้จากวงจรนี้เป็นสัญญาณรับเพียงอย่างเดียว

จากวงจร เมื่อสัญญาณส่งเข้ามาที่ขา 3 ของออปแอมป์ ซึ่งเป็นขา noninverting input ในขณะที่สัญญาณรับจะผ่านจากหม้อแปลงแมตชิ่งเข้ามาตกคร่อมที่ R1 ซึ่งถูกปรับให้มีค่า

ความต้านทานประมาณ 600 โอห์ม เพื่อแบ่งแรงดันของสัญญาณส่งออกเป็นส่วนที่ป้อนกลับและส่วนที่จะส่งออกไปอย่างละเท่า ๆ กัน สัญญาณส่งที่ถูกส่งออกไปจะรวมกับสัญญาณรับแล้วเข้าสู่ขา 5 ของออปแอมป์ สัญญาณนี้เขียนได้ง่ายๆ คือ Rx + Tx/2 นั่นเอง ส่วนสัญญาณที่ป้อนเข้าที่ขา 6 จะเป็นสัญญาณส่งที่ป้อนกลับ มีค่าเป็น Tx/2 ผลลัพธ์ของวงจรขยายส่วนต่างจะกลายเป็น Rx เพียงอย่างเดียว ในทางปฏิบัติเราจะพยายามปรับค่า R1 และ R2 อย่างเหมาะสม เพื่อให้สัญญาณที่รับและส่งมีค่าสมดุลกัน ส่วนออปแอมป์ที่ใช้คือ LM324 บรรจุออปแอมป์ไว้ 4 ตัวในไอซีตัวเดียว ทำให้สะดวกต่อการใช้งาน

3. วงจรขยายสัญญาณ

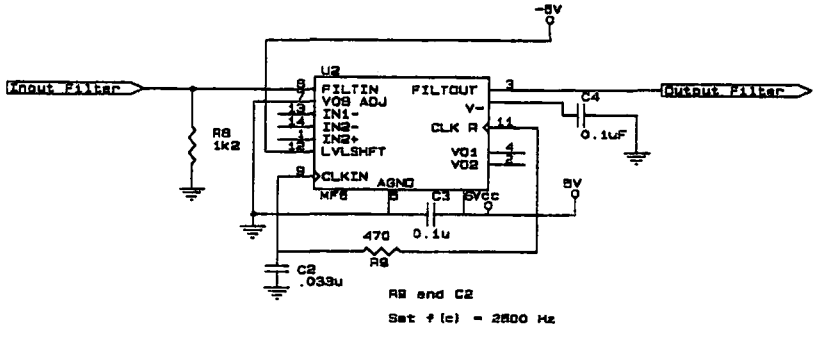
เป็นวงจรขยายสัญญาณแบบไม่กลับเฟส (noninverting amplifier) สามารถปรับค่าอัตราการขยายโดยใช้ตัวต้านทานปรับค่าได้ ทำการขยายสัญญาณที่รับมาจากวงจรดูเพล็กเซอร์ให้มีค่าเหมาะสมเพื่อป้อนให้แก่วงจรกรองความถี่ต่อไป



รูปที่ 6.9 แสดงวงจรขยายสัญญาณ

4. วงจรกรองความถี่แบบให้ความถี่ต่ำผ่าน (Lowpass Filter)

มีหน้าที่กรองความถี่ของสัญญาณอินพุทก่อนเข้าสู่วงจรดีมอดดูเลเตอร์โดยจะเลือกสัญญาณอินพุทในช่วงความถี่ที่สอดคล้องกับวงจรดีมอดดูเลเตอร์เท่านั้น โดยในที่นี้จะเลือกให้ผ่านเฉพาะค่าความถี่ที่ต่ำกว่า 2,500 Hz



รูปที่ 6.10 แสดงวงจรกรองความถี่โดยใช้ MF6

จากวงจรในรูปที่ 6.10 เราเลือกใช้ MF6 ของ National Semiconductor ซึ่งเป็น 6th order switched capacitor Butterworth Lowpass Filter การเลือกค่าความถี่ตัด (cutoff frequency) นั้น สามารถทำได้โดยการป้อนสัญญาณนาฬิกาให้เป็นจำนวนเท่าของความถี่นั้น ในโครงงานนี้ใช้ MF6-100 ดังนั้นจะต้องป้อนสัญญาณนาฬิกา 2,500 x 100 = 250 kHz แต่ไอซีตัวนี้อำนวยความสะดวกให้เราในการใช้งาน คือ แทนที่เราจะต้องป้อนสัญญาณนาฬิกาจากภายนอก ก็ใช้ตัวต้านทานและตัวเก็บประจุเพื่อกำหนดความถี่ให้วงจรออสซิเลตภายในได้ โดยค่าของความถี่กับตัวต้านทานและตัวเก็บประจุจะสัมพันธ์กันตามสมการ

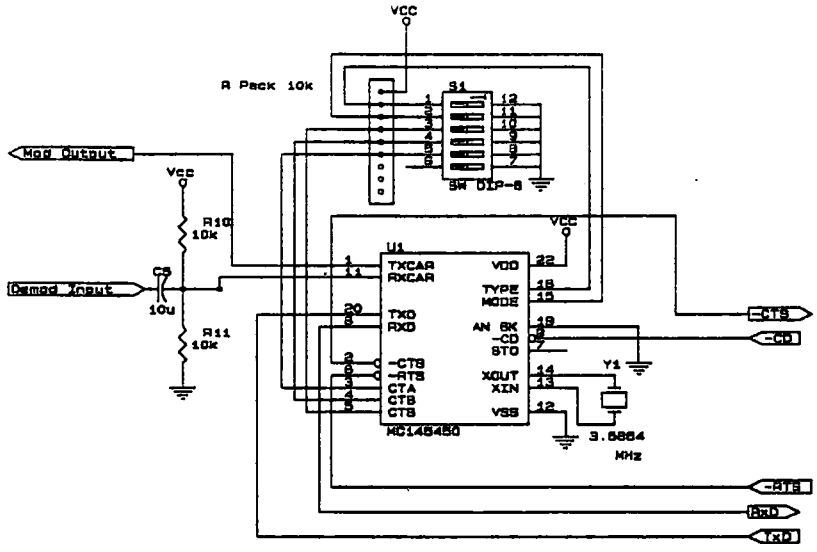
$$F_{clk} = \frac{1}{1.69RC}$$

ในวงจรจะกำหนดค่าตัวเก็บประจุเป็น 0.0015 μF และใช้ตัวต้านทานปรับค่าได้แทนเพื่อหลีกเลี่ยงข้อผิดพลาดอันเนื่องจากอุปกรณ์

เมื่อสร้างสัญญาณนาฬิกาเพื่อกำหนดค่าความถี่ตัดแล้ว เราจะสามารถป้อนสัญญาณอินพุทผ่านทางขา 8 และเอาท์พุทที่ออกทางขา 3 จะเป็นสัญญาณที่มีค่าเฉพาะบางช่วงความถี่ตามที่ต้องการส่วนเอาท์พุทนี้จะถูกป้อนเข้าสู่วงจรดีมอดดูเลเตอร์ต่อไป

5. วงจรมอดดูเลเตอร์ และวงจรดีมอดดูเลเตอร์ (Modulator and Demodulator)

วงจรทั้งสองนี้เป็นส่วนที่สำคัญที่สุดของโมเด็ม โดยวงจรมอดดูเลเตอร์จะทำหน้าที่แปลงจากสัญญาณทางดิจิตอลให้เป็นคลื่นพาหะ (Carrier) เพื่อส่งให้วงจรดูเพล็กซ์ป้อนเข้าสายโทรศัพท์ หรือ รับคลื่นพาหะจากวงจรกรองความถี่เข้าสู่วงจรดีมอดดูเลเตอร์ เพื่อแปลงให้เป็นสัญญาณรับดิจิตอล นอกจากนี้ยังทำหน้าที่อื่น เช่น สร้างสัญญาณตอบรับ (Answer Back Tone) รวมทั้งสามารถกำหนดมาตรฐานในการใช้งานและอัตราเร็วในการรับส่งข้อมูลได้อีกด้วย



รูปที่ 6.11 แสดงวงจรมอดดูเลเตอร์และดีมอดดูเลเตอร์

จากวงจร เราจะใช้ MC145450 เป็นไอซีที่ทำหน้าที่มอดดูเลตและดีมอดดูเลตแบบ FSK ในตัวเดียวกัน มาตรฐานที่สนับสนุน คือ Bell 202 และ CCITT V.23 โดยในโครงงานนี้จะใช้ Bell 202 เนื่องจากเป็นมาตรฐานที่ให้อัตราเร็วสำหรับการรับส่งใน reverse channel สูงกว่า

อินพุทและเอาท์พุทของไอซีที่สำคัญและใช้งาน ได้แก่

1. Xin และ Xout จะต่อเข้ากับคริสตอลความถี่มาตรฐานตามที่กำหนด คือ 3.6864 MHz และต้องมีตัวต้านทาน 15 เมกกะโอห์ม ต่อขนานกับคริสตอล โดยวงจรภายในจะนำความถี่นี้ไปสร้างเป็นฐานเวลาเพื่อใช้ในการกำเนิดความถี่ที่สำคัญในการส่งและรับข้อมูล

2. Type ใช้ในการเลือกมาตรฐานสำหรับการรับส่งข้อมูล ถ้ามีระดับลอจิกเป็น "1" จะเป็น Bell 202 และถ้าเป็น "0" จะเป็น CCITT V.23 ในที่นี้เราเลือกใช้ Bell 202

3. Mode เป็นขาที่กำหนดคู่ความถี่ที่จะใช้ในการดีมอดดูเลตและดีมอดดูเลต โดยถ้าเลือกให้ขานี้เป็น "0" จะแสดงว่าเป็นการใช้งานแบบ forward channel แต่ถ้าเป็น "1" จะหมายถึง reverse channel เมื่อสังเกตจากตารางที่ 6.1 จะเห็นได้ว่า ตามมาตรฐาน Bell 202 ถ้าเป็น Host เราจะเลือก Mode เป็น "0" หรือ forward channel เพื่อส่งข้อมูลด้วยอัตราเร็ว 1,200 บิทต่อวินาที และรับด้วยอัตรา 150 บิทต่อวินาที แต่ถ้ากรณีที่เป็น terminal จะกำหนดให้ Mode เป็น "1" หรือ reverse channel เพื่อส่งข้อมูลด้วยอัตราเร็ว 150 บิทต่อวินาที และรับด้วยอัตราเร็ว 1,200 บิทต่อวินาที

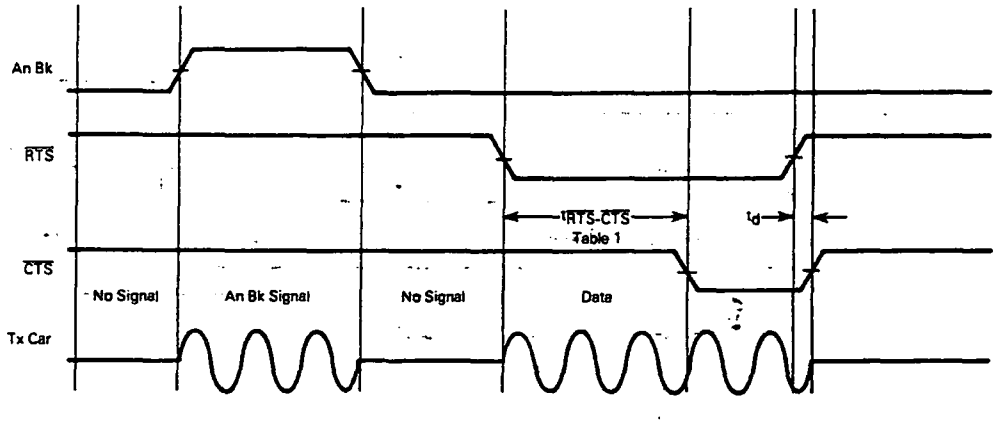
ตารางที่ 6.1 แสดงการเลือก mode ใช้งานไอซี MC145450

Type	Mode	Transmit Data	Transmit Frequency	Answer Back Tone	Application
0	0	0 1	2,100 1,300	2,100	CCITT V.23 75 baud Rx 1,200 baud Tx forward channel
0	1	0 1	450 390	2,100	CCITT V.23 1,200 baud Rx 75 baud Tx reverse channel
1	0	0 1	2,200 1,200	2,025	Bell 202 150 baud Rx 1,200 baud Tx forward channel
1	1	0 1	510 390	390	Bell 202 75 baud Rx 1,200 baud Tx reverse channel

4. RTS เป็นขาอินพุทจากส่วนควบคุมเพื่อร้องขอส่งข้อมูลมาให้วงจรมอดดูเลเตอร์ ซึ่งเมื่อวงจรมอดดูเลเตอร์ทราบจะตอบรับผ่านขา CTS ซึ่งเราสามารถกำหนดช่วงเวลาหน่วงก่อนที่จะมีการตอบรับที่ขา RTS ได้ โดยการกำหนดช่วงเวลาหน่วงระหว่าง RTS - CTS จากขา CTA, CTB และ CTC ซึ่งเวลาหน่วงจะเป็นไปตามตารางที่ 6.2 การร้องขอ RTS ทำโดยป้อนระดับลอจิกเป็น "0" ซึ่งการตอบกลับของ CTS คือเปลี่ยนจาก "1"เป็น "0" ดังรูปที่ 6.12

ตารางที่ 6.2 แสดงช่วงเวลาที่หน่วงระหว่าง RTS - CTS

CTC	CTB	CTC	Delay (ms)
0	0	0	0
0	0	1	26.7
0	1	0	40.0
0	1	1	60.0
1	0	0	133.3
1	0	1	213.3
1	1	0	266.7
1	1	1	426.6



รูปที่ 6.12 แสดงลำดับสัญญาณของ AnBk และ RTS-CTS

5. CD เป็นขาอินพุทเพื่อที่จะบอกวงจรดีมอดดูเลเตอร์ว่า จะให้ทำการดีมอดดูเลต สัญญาณพาหะที่เข้ามาหรือไม่ ถ้า CD เป็น "1" จะไม่มีการดีมอดดูเลต ทำให้เอาท์พุทที่เป็นดิจิตอลจากขา RxD เป็น "1" เสมอ แต่ถ้า CD เป็น "0" จะทำให้วงจรทำการดีมอดดูเลตสัญญาณอินพุท

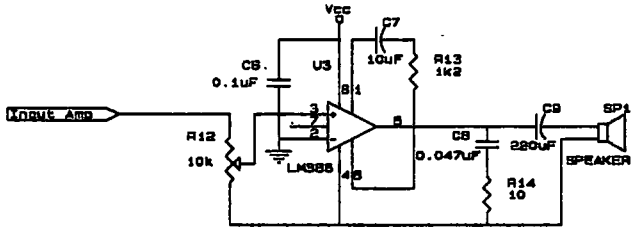
6. AnBk เป็นการกำหนดว่าจะให้วงจรมอดดูเลเตอร์ส่งสัญญาณตอบกลับหรือไม่ ถ้า AnBk เป็น "1" จะทำให้ขา TxCar ผลิตเอาท์พุทออกมาเป็นสัญญาณตอบกลับ ขณะเดียวกันก็จะทำให้ CTS เป็น "1" ดังรูปที่ 6.12

7. TxData สัญญาณอินพุทที่เป็นดิจิตอลจากวงจรควบคุมหรือคอมพิวเตอร์จะถูกป้อนผ่านขานี้ไปยังวงจรมอดดูเลเตอร์ โดยจะเกิดสัญญาณพาหะเป็นเอาท์พุทที่ขา TxCar สัญญาณพาหะที่เกิดขึ้นจะเป็นสัญญาณรูปไซน์ที่ถูกสังเคราะห์เชิงดิจิตอล 16 ระดับ มีค่าแอมปลิจูดประมาณ 0.5 Vp-p เอาท์พุทจะถูกป้อนผ่านตัวต้านทาน 10 กิโลโอห์มก่อนส่งให้กับวงจรดูเพล็กเซอร์ต่อไป

8. RxCar จะรับสัญญาณพาหะในรูปของ FSK จากวงจรกรองความถี่เข้าสู่วงจรดีมอดดูเลเตอร์ เอาท์พุทจะออกทางขา RxD ซึ่งเป็นสัญญาณดิจิตอลตามความถี่ที่ได้กำหนดไว้

6. วงจรขยายเสียง (Audio Amplifier)

เพื่อให้ผู้ใช้ได้ยินเสียงสัญญาณต่าง ๆ ซึ่งจะทำให้ทราบขั้นตอนการทำงานในขณะนั้นๆ สัญญาณที่จะผ่านเข้าสู่วงจรนี้ ได้แก่ สัญญาณ DTMF และสัญญาณตอบรับ

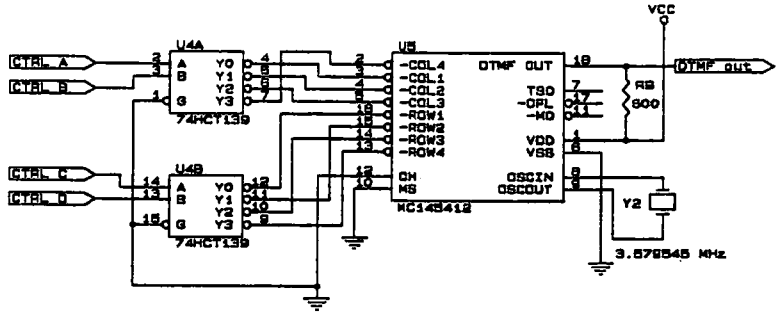


รูปที่ 6.13 แสดงวงจรขยายเสียง

เราจะใช้ LM386 ซึ่งเป็นออปแอมป์ที่ขยายสัญญาณช่วงความถี่เสียง สามารถนำไปขับลำโพงขนาด 8 โอห์มได้ ในวงจรนี้จะปรับค่าอัตราการขยายไว้ 20 เท่า ส่วนสัญญาณอินพุทจะถูกต่อผ่านตัวต้านทานปรับค่าได้ เพื่อปรับความดังค่อยของลำโพงได้

7. วงจรกำเนิดสัญญาณโทน (DTMF Generator)

ใช้สำหรับสร้างสัญญาณโทนเพื่อระบุหมายเลขโทรศัพท์ของเป้าหมายที่จะติดต่อด้วย



รูปที่ 6.14 แสดงวงจรกำเนิดสัญญาณโทน

จากรูป เราจะใช้ไอซี MC145412 สร้างสัญญาณโทน หรือที่เรียกว่า สัญญาณ DTMF (Dual Tone Multi Frequency) สัญญาณที่ใช้จะแทนการกดปุ่มแต่ละปุ่มจะประกอบด้วยความถี่ 2 ความถี่ที่แทนคอลัมน์และแถวของปุ่มหมายเลขนั้น ดังรูป

	1,209 Hz	1,336 Hz	1,477 Hz	High Group Frequency
697 Hz	1	2	3	
770 Hz	4	5	6	
852 Hz	7	8	9	
941 Hz	*	0	#	

Low Group Frequency

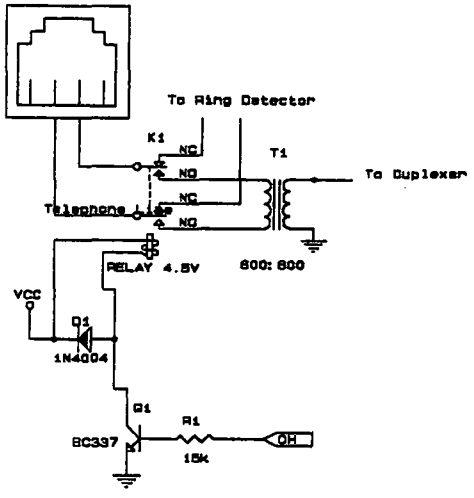
รูปที่ 6.15 แสดงความถี่ที่ใช้สำหรับคอลัมน์และแถวสำหรับ DTMF

เนื่องจากไอซีเบอร์นี้ เดิมออกแบบให้รับข้อมูลจากคีย์บอร์ดโดยตรง ขาอินพุทจึงเป็นแถวและคอลัมน์อย่างละ 4 ชุดของคีย์บอร์ด แต่เราต้องการเชื่อมต่ออินพุทดังกล่าวนี้เข้าสู่ส่วนควบคุมซึ่งเป็นเอาท์พุทพอร์ตขนาด 4 บิท จึงจำเป็นต้องต่อเอาท์พุทนี้ผ่าน 74139 โดยแบ่งเป็นส่วนละ 2 บิท สำหรับการระบุตำแหน่งคอลัมน์และอีก 2 บิทสำหรับระบุตำแหน่งแถว

ในการอ้างอิงความถี่ทั้งหมดจะใช้ คริสตอล 3.579545 MHz โดยจะต่อเข้าที่ขา OSCIN ของ MC145412 นอกจากนี้ยังต้องกำหนดสัญญาณเอาท์พุทให้เป็นโทนแทนที่จะเป็นพัลส์ โดยการป้อนลอจิก "0" ที่ขา MS (Mode Select) สัญญาณเอาท์พุท DTMF นี้จะออกจากขา 18 โดยเราต้องต่อขาเอาท์พุทผ่านตัวต้านทาน 600 โอห์มกับไฟบวกด้วย

8. วงจรควบคุมการยกและวางหูโทรศัพท์ (Telephone Line On/Off Hook)

ในสภาวะปกติ โทรศัพท์จะอยู่ในสภาวะวางหู (On hook) เพื่อที่จะรอสัญญาณกริ่งที่จะเข้ามา เมื่อมีสัญญาณกริ่งขึ้น วงจรนี้จะทำการยกหู (Off hook) เพื่อบอกชุมสายว่าพร้อมที่จะต่อกับคู่สนทนาแล้ว วงจรควบคุมการยกและวางหูโทรศัพท์นี้แสดงให้เห็นดังรูป



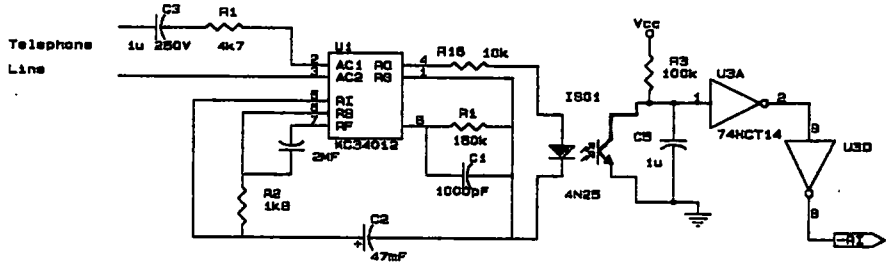
รูปที่ 6.16 แสดงวงจรควบคุมการยกและวางหูโทรศัพท์

การทำงานของวงจรจะเป็นการสั่งเปิดและปิดทรานซิสเตอร์ โดยถ้าต้องการยกหู จะป้อนลอจิก "1" ผ่านตัวต้านทาน 15k เพื่อจำกัดกระแสก่อนเข้าทรานซิสเตอร์ กระแสที่ไหลเข้าขาเบสนี้จะทำให้ BC337 เกิดสภาวะอิ่มตัว (Saturate) และจะทำให้กระแสไหลผ่านขดลวดเหนี่ยวนำของรีเลย์เข้าสู่ขาคอลเลคเตอร์และอิมิตเตอร์ จังหวะดังกล่าวจะทำให้รีเลย์ทำ

การย้ายหน้าสัมผัสจนสายโทรศัพท์ถูกต่อเข้ากับหม้อแปลงแมตชิ่ง ซึ่งเป็นการเริ่มต้นสภาวะการยกหู แต่ถ้าป้อนลอจิก "0" ผ่านตัวต้านทานเข้าสู่ทรานซิสเตอร์ จะทำให้ทรานซิสเตอร์ไม่ทำงาน หน้าสัมผัสของรีเลย์จะต่อสายโทรศัพท์เข้าสู่วงจรตรวจสอบสัญญาณกริ่งต่อไป

9. วงจรตรวจสอบสัญญาณกริ่ง (Ring Detector)

สัญญาณกริ่ง จะเป็นสัญญาณแรงดันไฟสลับที่มีค่าสูง (60-100 โวลต์) มีค่าความถี่ต่ำ โดยจะผ่านสายโทรศัพท์มาควบกับแรงดันไฟตรง 48 โวลต์ วงจรตรวจสอบกริ่งนี้จะสามารถตรวจสอบจับสัญญาณกริ่งที่มาเป็นช่วง ๆ แล้วให้เอาท์พุทที่เป็นลอจิกเข้าสู่ส่วนควบคุมได้

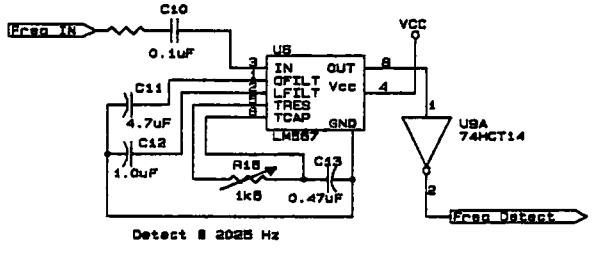


รูปที่ 6.17 แสดงวงจรตรวจสอบสัญญาณกริ่ง

สัญญาณกริ่งจะผ่านตัวเก็บประจุ 1 ไมโครฟารัด 250 โวลต์ เข้าสู่ขา 2 ของ MC34012 ซึ่งเป็นไอซีที่มีหน้าที่ตรวจสอบสัญญาณกริ่ง โดยจะให้เอาท์พุทที่เป็นไฟสลับสำหรับขับวงจรกระดิ่ง แต่ในวงจรเราต่อผ่านออปโตคัปเปอร์(Optocoupler) เพื่อทำการแยกระบบไฟจากสายโทรศัพท์ออกจากส่วนควบคุม เอาท์พุทจากออปโตคัปเปอร์จะเข้าสู่ส่วนควบคุม โดยลอจิก "1" หมายถึงไม่มีสัญญาณกริ่งมา และ "0" หมายถึงมีสัญญาณกริ่งเข้ามา

10. วงจรตรวจจับความถี่ (General Frequency Detector)

วงจรนี้จะใช้ในการตรวจจับสัญญาณที่เข้ามาในวงจรว่า มีค่าความถี่ตรงกับที่ต้องการหรือไม่ ความถี่ที่จะตรวจสอบนี้ก็คือความถี่ตอบรับและความถี่ที่เป็นสัญญาณพาหะ



รูปที่ 6.18 แสดงวงจรตรวจจับความถี่

ในการตรวจจับความถี่ เราจะใช้ LM567 ซึ่งเป็นไอซีเฟสล็อคลูป (PLL : Phase Lock Loop) ทำการตรวจจับความถี่ที่อยู่ในช่วงแคบๆ ช่วงความถี่หนึ่ง โดยเราสามารถกำหนดค่าความถี่กลางและความกว้างของช่วงความถี่ที่ตรวจจับได้จาก

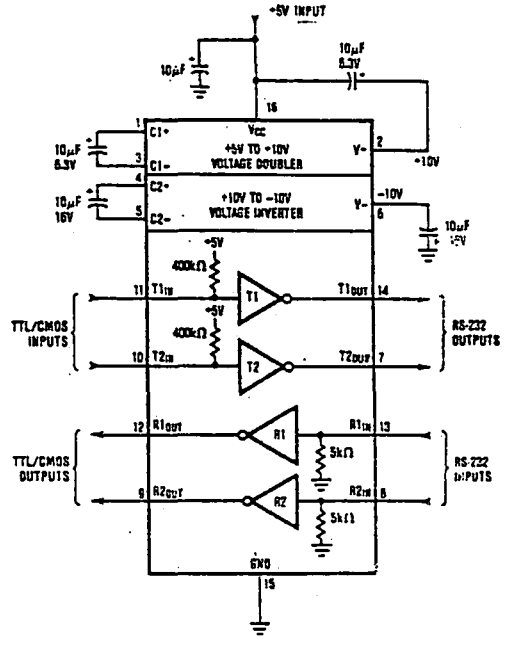
$$f_o = \frac{1}{1.1\ R_1 C_1}$$

และ $$BW = 1070 \sqrt{\frac{V_i}{f_o C_2}} \text{ in \% of } f_o$$

จากวงจร เราจะเลือกใช้ค่า C_1 และ C_2 คงที่ แต่จะปรับค่า R_1 เพื่อทำการเลือกความถี่แทน โดยถ้าวงจรตรวจจับความถี่นั้นได้จะเปลี่ยนเอาท์พุทลอจิกจาก "1" เป็น "0" สัญญาณจากเอาท์พุทนี้จะเข้าสู่ส่วนควบคุมเพื่อทำการตรวจสนองต่อสัญญาณที่ได้รับต่อไป

11. วงจรแปลงระดับสัญญาณระหว่าง TTL หรือ CMOS กับ RS-232

ในการติดต่อระหว่างคอมพิวเตอร์กับโมเด็ม จะใช้มาตรฐานการติดต่อผ่าน RS-232 โดยค่าระดับแรงดันที่แตกต่างกันระหว่างสัญญาณดังกล่าวจะต้องใช้ตัวแปลงระดับสัญญาณ ซึ่งเป็นไปตามวงจรดังรูป



รูปที่ 6.19 แสดงวงจรแปลงระดับสัญญาณระหว่าง TTL กับ RS-232

จากรูป จะใช้ MAX232 เพื่อทำหน้าที่แปลงระดับสัญญาณ โดย MAX232 แต่ละตัวจะประกอบด้วย วงจรขับแรงดันจาก TTL ให้เป็น RS-232 อยู่ 2 ชุด และ วงจรรับสัญญาณจาก RS-232 แปลงให้เป็น TTL อีก 2 ชุด ในรูปจะเห็นว่า เราต้องต่อตัวเก็บประจุเพื่อใช้ในการทวีคูณแรงดัน ซึ่งเป็นวงจรภายในเอง ซึ่งในกรณีที่เราป้อนแรงดันไฟเลี้ยงให้มีค่า 5 โวลต์ จะได้ค่าแรงดันจากวงจรขับเป็น 10 และ -10 โวลต์ ซึ่งอยู่ในช่วงของสัญญาณที่ยอมรับได้

6.3 หลักการของระบบควบคุม

ระบบควบคุมนี้ จะทำหน้าที่ในการควบคุมการทำงานของระบบทั้งหมด ให้เป็นไปตามจังหวะอย่างถูกต้อง โดยเราจะใช้ไมโครคอลโทรลเลอร์ 8031 ทำการจัดลำดับทั้งหมด ขั้นตอนการควบคุมที่ระบบนี้สร้างขึ้นสามารถอธิบายอย่างย่อ ๆ ได้ดังนี้

1. เริ่มต้นเมื่อทำการเปิดปุ่ม power เพื่อจ่ายไฟให้กับทุกระบบ ที่ฝาย Terminal ระบบควบคุมจะทำการตรวจสอบว่าขณะนั้นโมเด็มมีการต่อกับคอมพิวเตอร์หรือไม่ โดยการตรวจจากสัญญาณ DTE ว่า ON หรือไม่ ถ้าใช่จะบอกผู้ใช้ผ่านทางจอคอมพิวเตอร์ว่า ให้ทำการรูดบัตรและรับรหัสผ่าน

2. วงจรควบคุมจะทำการอ่านข้อมูลจากบัตรแม่เหล็กเก็บไว้ แล้วทำการรับรหัสผ่านจากคอมพิวเตอร์

3. ทำการเริ่มต้นติดต่อกับฝาย Host โดยเริ่มจากการทำให้ขาควบคุม OH เป็น "1" เพื่อทำการยกหูโทรศัพท์ แล้วส่งสัญญาณควบคุมไปยังวงจรกำเนิดสัญญาณโทนให้เริ่มสร้างสัญญาณโทน เพื่อระบุเป้าหมายที่จะติดต่อ

4. ฝาย Host จะตรวจสอบสัญญาณกริ่งจากวงจรตรวจสอบกริ่ง แล้วทำการยกหูโทรศัพท์ หลังจากนั้นจะส่งสัญญาณตอบรับไปให้กับ Terminal

5. เมื่อ Terminal ตรวจรับสัญญาณตอบรับได้จากวงจรตรวจจับสัญญาณตอบรับ ก็แสดงว่าการติดต่อระหว่างสองฝายได้เริ่มขึ้น Terminal จะทำการส่งข้อมูลจากยืนยันว่าเป็นสถานีลูกจริงข้อมูลที่อ่านได้จากบัตรแม่เหล็ก และรหัสผ่านไปให้ฝาย Host ทำการตรวจสอบ

6. Host จะทำการตรวจสอบจากข้อมูลที่รับได้ว่าถูกต้องหรือไม่ แล้วจึงตอบกลับมาว่าจะให้ผู้ใช้ทำการใช้ระบบเพื่อรับข้อมูลได้หรือไม่ ถ้าไม่ได้ ฝาย Terminal ก็จะแจ้งให้ผู้ใช้ทราบและอาจให้ผู้ใช้ได้ป้อนรหัสผ่านใหม่ แล้วจึงทำซ้ำในขั้นตอนที่ 5 ใหม่อีกครั้ง

7. เมื่อมาถึงขั้นตอนนี้ ผู้ใช้จะสามารถเข้าถึงระบบเพื่อรับข้อมูลได้แล้ว การควบคุมทั้งหมดจะโอนไปยังส่วนของคอมพิวเตอร์ทั้งสองที่จะรับส่งข้อมูลกัน

6.4 การปรับแต่งก่อนการใช้งาน

ก่อนที่จะทำการรับและส่งข้อมูลนั้น เราจำเป็นที่จะต้องทำการปรับแต่งในส่วนของวงจรบางส่วนเพื่อให้เหมาะสมสำหรับการใช้งาน โดยการปรับแต่งนี้จะทำโดยการปรับค่าความต้านทานตามส่วนต่างๆ ดังนี้

1. วงจร Duplexer ประกอบด้วยตัวต้านทานปรับค่า 2 ตัว แต่ละตัวจะมีหน้าที่ต่างกัน คือ ตัวต้านทานที่มีค่า 1k จะใช้ปรับค่าแรงดันที่จะส่งออกไป (Tx) โดยจะปรับให้มีค่าประมาณ 600 โอห์ม ซึ่งมีค่าใกล้เคียงกับค่าอิมพีแดนซ์ขาเข้าของวงจร ส่วนอีกตัวที่มีค่า 50k จะควบคุมแรงดันการป้อนกลับเพื่อนำไปเปรียบเทียบกันแรงดันที่จะรับเข้า โดยการปรับค่าความต้านทานทั้งสองจะต้องสัมพันธ์กันเพื่อให้สัญญาณที่จะรับและส่งมีความผิดเพี้ยนน้อยที่สุด

2. สัญญาณรับที่ผ่านจากวงจร Duplexer จะถูกขยายโดยออปแอมป์ มีอัตราการขยายสูงสุดประมาณ 100 เท่า แต่เราจะปรับให้แรงดันเอาท์พุทมีค่าประมาณ 5 โวลต์ ซึ่งเป็นแรงดันในระดับอ้างอิงแบบดิจิตอล เพื่อป้อนเข้าสู่วงจรดีมอดดูเลตในส่วนต่อไป

3. วงจรขยายสัญญาณเสียง ปรับตัวต้านทานค่า 10k เพื่อให้ได้ระดับเสียงที่เหมาะสมแก่การได้ยิน

4. วงจรตรวจจับความถี่ เนื่องจากสัญญาณความถี่ต่าง ๆ ที่ใช้มีค่าต่างกัน ประกอบกับค่าความต้านทานประจำตัวต้านทานแบบคาร์บอนที่ใช้ในโครงงานนี้จะมีค่าผิดพลาดในการใช้งานจึงต้องทำการปรับค่าของตัวต้านทานให้เหมาะสมสำหรับค่าความถี่ต่างๆ โดย Host จะปรับให้ตรวจจับความถี่ 390 Hz และ 510 Hz ส่วน Terminal จะปรับสำหรับตรวจจับความถี่ 1,200 Hz, 2,200 Hz และ 2,025 Hz โดยค่าความถี่ 4 ค่าแรก (390 Hz, 510 Hz, 1,200 Hz และ 2,200 Hz) จะเป็นความถี่พาหะ ส่วนความถี่ค่าสุดท้าย (2,025 Hz) จะเป็นค่าความถี่ตอบรับ ในทางปฏิบัติแล้ว เราจะปลดอินพุทของวงจร phase lock loop ออกจากวงจรหลักแล้วทำการป้อนสัญญาณอินพุทเป็นความถี่ที่ต้องการลงไป หลังจากนั้นจึงปรับค่าความต้านทานจนได้ค่าเอาท์พุทของวงจรเป็น "0" แสดงว่าวงจรพร้อมที่จะตรวจจับความถี่นั้นๆ แล้ว

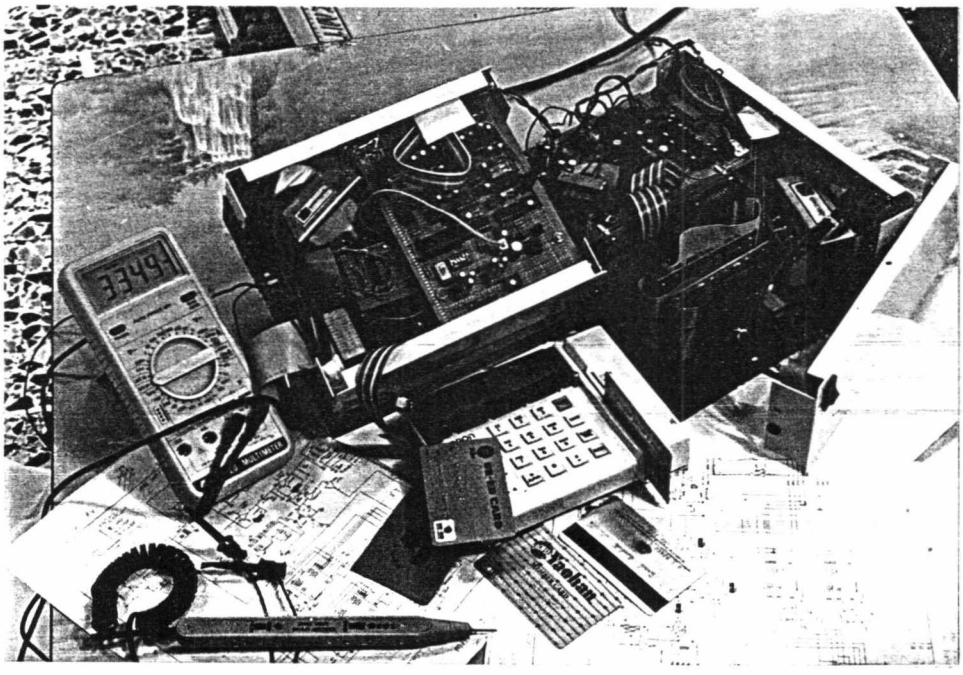
5. วงจรมอดดูเลตและดีมอดดูเลต จะสามารถปรับ mode การใช้งาน (Terminal หรือ Host) ชนิดของมาตรฐาน (Bell202 หรือ CCITT V.23) รวมทั้งช่วงเวลาหน่วงระหว่าง RTS-CTS ได้จาก dip switch ที่อยู่บนแผงวงจร โดยเราจะเลือก mode เป็น Terminal หรือ Host สำหรับแต่ละเครื่อง ปรับมาตรฐานเป็น Bell 202 ส่วนช่วงเวลาหน่วงจะปรับเป็น 0 ms (ไม่หน่วงเวลา) เพราะสัญญาณ RTS จากคอมพิวเตอร์จะถูกส่งผ่านส่วนควบคุมก่อนผ่านไปยังวงจรมอด-ดีมอดดูเลต จึงเหมือนกับ CTS ถูกควบคุมจะถูกควบคุมด้วยวงจรควบคุมแทน

บทที่ 7

ผลการทดลอง

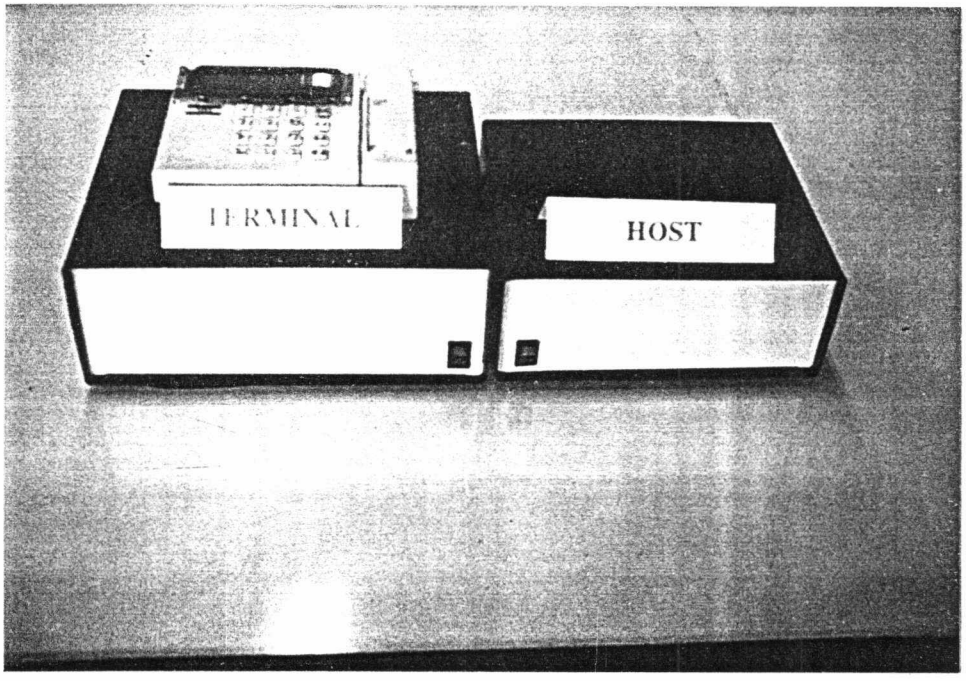
7.1 เครื่องที่ประกอบเสร็จสมบูรณ์แล้ว

เมื่อทำการประกอบวงจรทั้งหมดลงในกล่องทดลอง จะทำการจัดแผงวงจรดังรูปที่ 7.1 จากรูป ทางด้านซ้ายมือจะเป็นส่วนของ Terminal และส่วนของ Host จะอยู่ทางขวามือ

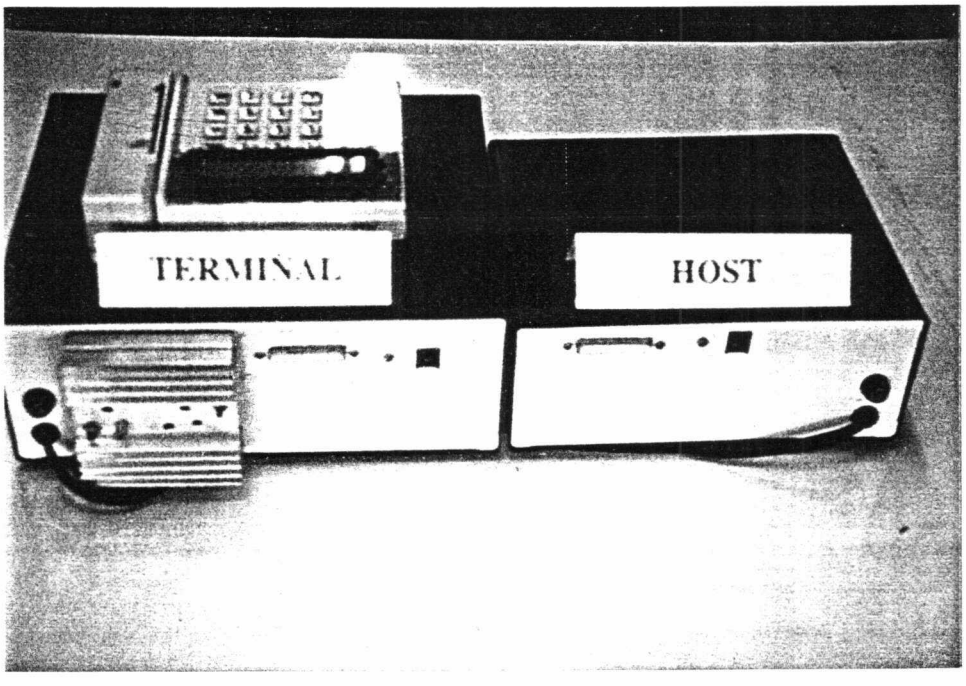


รูปที่ 7.1 แสดงรูปอุปกรณ์ทั้งหมดที่ประกอบรวมกันเป็นโครงงานนี้

รูปที่ 7.2 และ 7.3 คือ เครื่อง Terminal และ Host ที่ประกอบเสร็จสมบูรณ์แล้ว โดยในรูปที่ 7.2 จะเป็นด้านหน้าของเครื่อง และรูปที่ 7.3 แสดงด้านหลังของเครื่อง



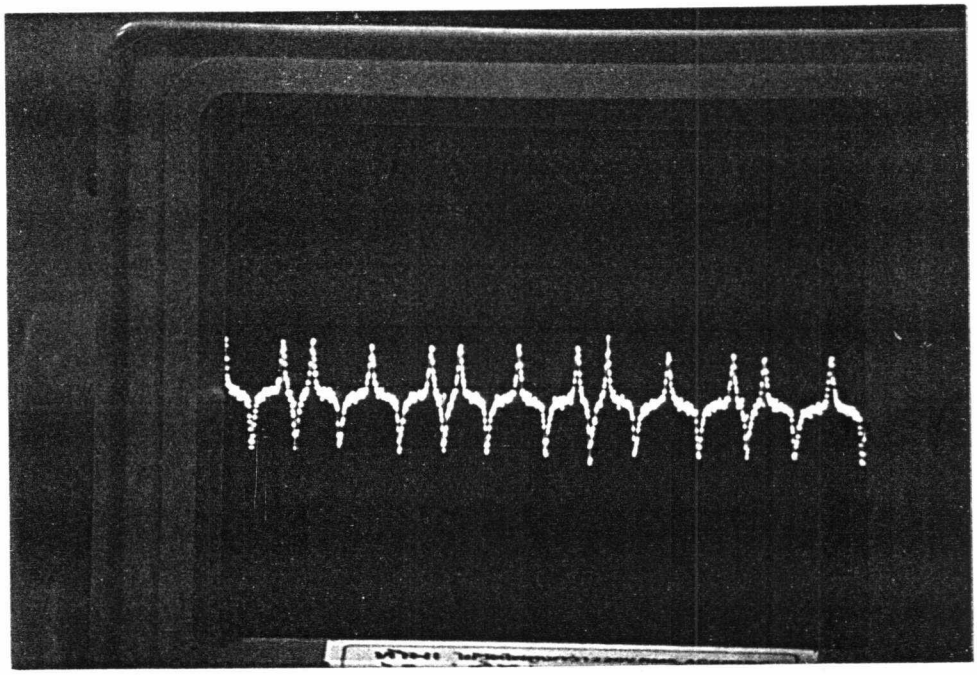
รูปที่ 7.2 ด้านหน้าของเครื่อง Terminal และ Host ที่ประกอบเสร็จแล้ว



รูปที่ 7.3 ด้านหลังของเครื่อง Terminal และ Host ที่ประกอบเสร็จแล้ว

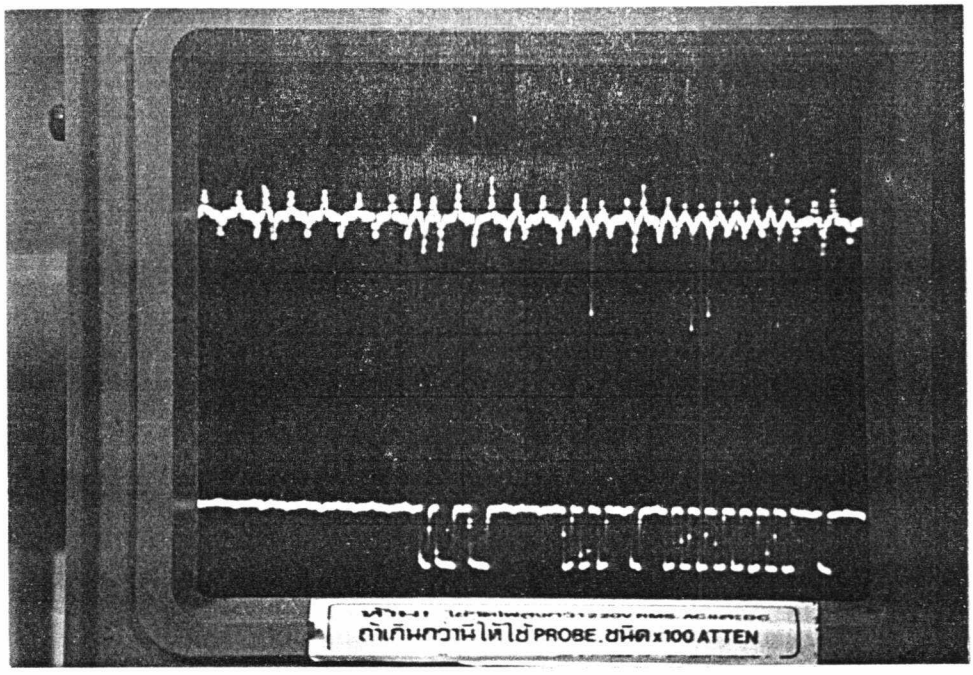
7.2 รูปแบบของสัญญาณต่างๆ ที่ได้จากการทดลอง

1. ในขั้นแรก เมื่อเราทำการรูดบัตรแม่เหล็กผ่านหัวอ่านแล้วทำการวัดค่าสัญญาณที่อ่านได้โดยตรง พบว่า สัญญาณจะมีลักษณะดังรูปที่ 7.4 โดยมีค่าระดับแรงดันประมาณ 20 mV



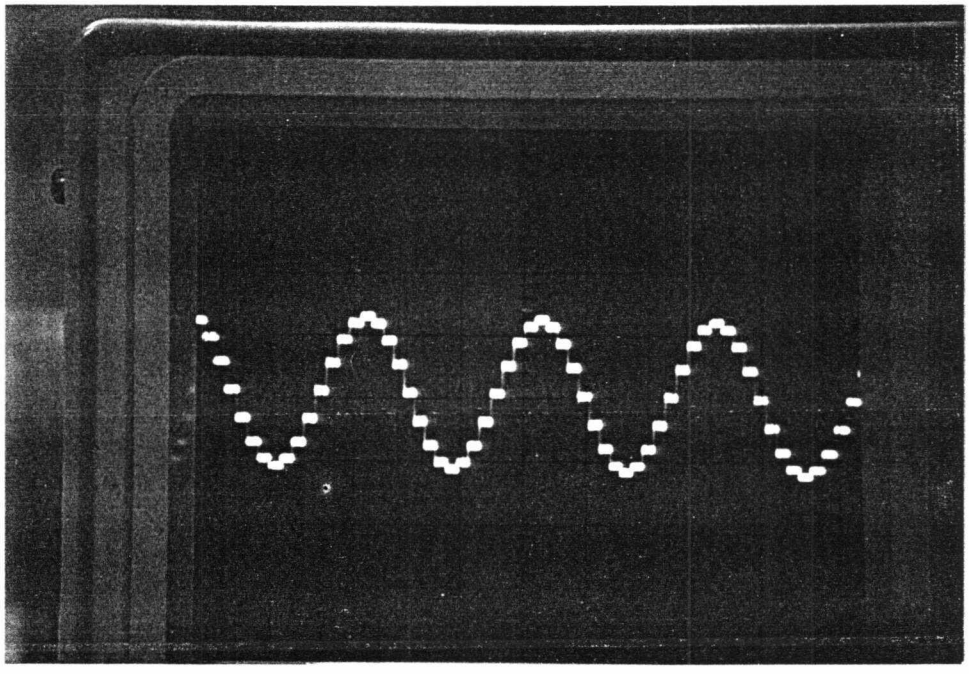
รูปที่ 7.4 แสดงรูปสัญญาณจากหัวอ่านบัตรแม่เหล็กเมื่อทำการรูดบัตร

2. เมื่อต่อป้อนสัญญาณจากหัวอ่านแม่เหล็กเข้าสู่วงจรอ่านบัตรแม่เหล็ก พบว่า สัญญาณจากเอาท์พุทของวงจรในขา DATA จะเป็นสัญญาณดิจิตอล มีระดับแรงดัน 5 และ 0 โวลต์ตามค่าของข้อมูลที่อ่านได้ ดังรูปที่ 7.5



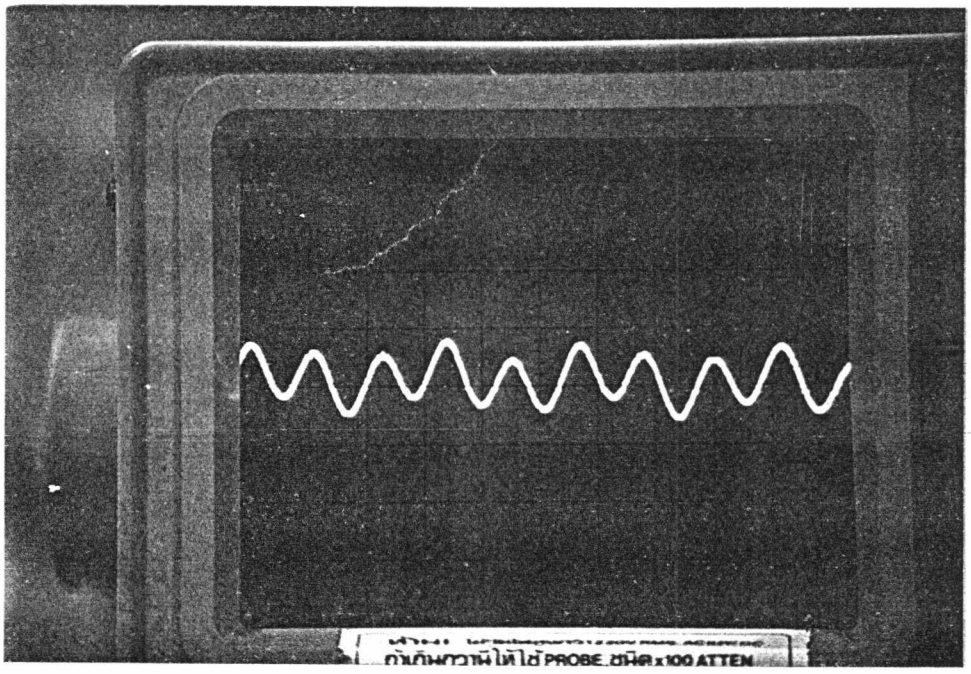
รูปที่ 7.5 แสดงรูปสัญญาณที่ได้จากวงจรอ่านบัตรแม่เหล็กเทียบกับสัญญาณจากหัวอ่านโดยตรง

3. วงจรมอดดูเลตจะสร้างสัญญาณพาหะจากข้อมูลที่จะส่งผ่านโมเด็ม โดยสัญญาณดังกล่าวจะเป็นสัญญาณที่สังเคราะห์ทางดิจิตอล ประกอบด้วยแรงดัน 16 ระดับประกอบขึ้นเป็นคลื่นสัญญาณไซน์ดังรูป มีค่าแอมปลิจูดประมาณ 0.5 Vp-p ความถี่ของสัญญาณจะขึ้นอยู่กับ mode และ type ของการทำงานที่ได้ตั้งไว้ ในที่นี้วงจรมอดดูเลตได้ผลิตสัญญาณความถี่ 1,200 Hz

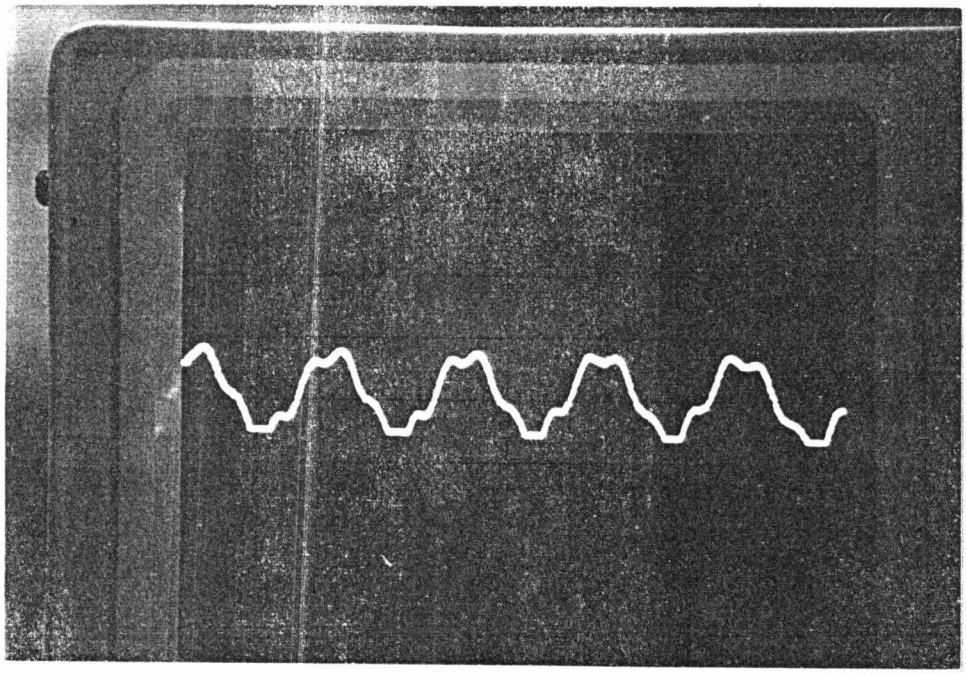


รูปที่ 7.6 แสดงรูปสัญญาณที่ได้จากวงจรมอดดูเลเตอร์

4. เมื่อต่อส่วนย่อยทั้งหมดของโมเด็มแล้ว ทำการวัดสัญญาณพาหะที่จะเข้าวงจรดีมอดดูเลตเพื่อทำการแปลงเป็นข้อมูลทางดิจิตอลภาครับ ในส่วนของ Host จะส่งสัญญาณความถี่ที่สูงกว่าเมื่อเทียบกับ Terminal ดังนั้นสัญญาณที่วัดได้จากวงจรดีมอดดูเลตที่ Terminal ตามรูปที่ 7.7 จะมีความถี่สูงกว่าเมื่อเทียบสัญญาณพาหะที่รับได้จาก Host ตามรูปที่ 7.8 จากรูปทั้งสองนี้จะเห็นว่า สัญญาณมีความผิดเพี้ยนไปจากสัญญาณที่ออกจากวงจรมอดดูเลตของฝั่งตรงข้ามไปบ้าง ทั้งนี้เนื่องจากวงจรดูเพล็กเซอร์ยังขาดความสมบูรณ์ในการตัดสัญญาณส่งออกจากสัญญาณรับ ทำให้สัญญาณรับบางส่วนแฝงเข้ามาในสัญญาณรับอยู่บ้าง



รูปที่ 7.7 แสดงสัญญาณที่รับได้ก่อนจะเข้าวงจรดีมอดดูเลเตอร์ของ Terminal



รูปที่ 7.8 แสดงสัญญาณที่รับได้ก่อนจะเข้าวงจรดีมอดดูเลเตอร์ของ Host

7.3 ผลการทดลองส่วนต่างๆ ของระบบ

ผลการทดลองในส่วนของวงจรอ่านบัตรแม่เหล็ก

เมื่อทำการประกอบวงจรอ่านบัตรแม่เหล็ก และต่อสัญญาณควบคุมทั้งสามเส้นเข้าสู่ระบบไมโครคอนโทรลเลอร์ แล้วเขียนโปรแกรมควบคุมการทำงาน สุดท้ายจะได้ข้อมูลที่เป็นรหัสแอสกี้ที่อ่านได้จากบัตรแม่เหล็ก มีขนาดของข้อมูลเป็น 40 ตัวอักษร ในการนำข้อมูลนี้ไปใช้ในการตรวจสอบเจ้าของบัตรนั้นจะใช้เพียง 20 ตัวแรก การทดสอบความถูกต้องของข้อมูลนั้นทำได้จากการเปรียบเทียบข้อมูลที่อ่านได้จากวงจรกับเลขรหัสที่ติดอยู่บนบัตร

บัตรที่นำมาใช้ในการทดสอบมีหลายชนิด ได้แก่ บัตรเอทีเอ็มของธนาคารกรุงเทพและกสิกรไทย บัตรสมาชิกร้านหนังสือและร้านค้า เป็นต้น เมื่อนำมาทดสอบพบว่า ข้อมูลที่ได้จากวงจรมีค่าตรงกับตัวเลขรหัสบนบัตรทุกตัว ทั้งนี้เนื่องจากรูปแบบการจัดเก็บของบัตรแม่เหล็กที่มีมาตรฐานในการตรวจสอบความผิดพลาดได้ในแต่ละตัวอักษรและข้อมูลทั้งแทร็ค แต่เมื่อนำบัตรที่มีรูปแบบการจัดเก็บที่ไม่ตรงกับมาตรฐานสากล เช่น บัตรผ่านเข้าออกประตู จะไม่สามารถอ่านข้อมูลได้ถูกต้อง โดยเครื่องจะเตือนว่าไม่สามารถอ่านข้อมูลได้

ผลการทดลองในส่วนของวงจรโมเด็ม

ผลการทำงานในส่วนต่างๆ ของวงจรโมเด็ม เป็นดังนี้

1. การรับส่งข้อมูลระหว่างคอมพิวเตอร์

ในการรับส่งข้อมูลระหว่าง Terminal กับ Host นั้น จะเกิดผ่านคีย์บอร์ดหรือโปรแกรมที่ทำงานบนคอมพิวเตอร์ ข้อมูลนี้จะถูกส่งมายังโมเด็มผ่านทาง RS-232 แล้วมอดดูเลตกับสัญญาณพาหะแล้วจึงเข้าสู่สายโทรศัพท์ โดยเราจะใช้อัตราเร็วในการส่งข้อมูลจาก Host ไปยัง Terminal เป็น 1,200 bps แล้วทำการปรับวงจร Duplexer จนทำให้ข้อผิดพลาดในการส่งข้อมูลในแต่ละตัวอักษรมีค่าน้อยที่สุด ปรากฏว่า ข้อผิดพลาดในการส่งข้อมูลแบบซิมเพล็กซ์มีค่าน้อยมาก ทั้งที่เนื่องจากเราใช้ช่วงความถี่ของสายโทรศัพท์ในการส่งข้อมูลทางเดียวทั้งหมด ส่วนการรับและส่งข้อมูลแบบฟูลดูเพล็กซ์ สามารถทำได้ที่อัตราเร็ว 300 bps ซึ่งที่อัตราเร็วนี้ความผิดพลาดที่เกิดขึ้นจะมีค่าน้อยมากหรือแทบไม่มีเลยเมื่อเป็นการผลัดกันรับและส่ง แต่เมื่อเราทำการรับและส่งข้อมูลพร้อมๆ กัน หรือสวนทางกันนั้น จะทำให้ข้อผิดพลาดในการรับและส่งข้อมูลมีค่ามากขึ้น ข้อผิดพลาดนี้เกิดจาก ความไม่สมบูรณ์ของวงจร Duplexer ในการแยกสัญญาณรับและส่งออกจากกันอย่างเด็ดขาด

นอกจากนี้ ยังต้องมีการจัดลำดับขั้นตอนสัญญาณควบคุมระหว่างคอมพิวเตอร์และโมเด็ม และต้องส่งสัญญาณให้ถูกต้องตรงกับสภาวะของระบบขณะนั้นๆ สัญญาณเหล่านั้นได้แก่ DTE, DSR, CTR, RTS, CD และ RI โดยสัญญาณสี่สัญญาณแรกเป็นการตรวจสอบความพร้อมของโมเด็มและคอมพิวเตอร์ จากการทดลองสามารถจัดให้ถูกต้องได้ตามมาตรฐาน ส่วน CD และ RI ซึ่งเป็นสัญญาณตรวจสอบพาหะและกริ่ง ก็สามารถตรวจสอบได้ถูกต้องเป็นไปตามมาตรฐานที่ได้อธิบายไปแล้วในบทที่ 3

2. สัญญาณเสียง

สัญญาณโทน DTMF ที่ได้จากวงจร DTMF generator สามารถให้สัญญาณความถี่ตรงตามมาตรฐาน ซึ่งทราบได้จาก การทดลองป้อนสัญญาณโทนที่ระบุหมายเลขที่จะติดต่อเข้าสู่สายโทรศัพท์เหมือนการเริ่มติดต่อของระบบโทรศัพท์ทั่วไป ปรากฏว่า เกิดสัญญาณกริ่งที่หมายเลขนั้น นั่นคือเราทำการติดต่อกับหมายเลขที่เป็นคู่สายได้ แสดงว่าสัญญาณที่ได้ถูกต้อง

สัญญาณโทนที่เกิดจากวงจร DTMF generator และ สัญญาณความถี่ตอบรับจากอีกฝ่ายนั้น จะถูกป้อนผ่านวงจรขยายเสียงและสามารถรับฟังได้จากลำโพง โดยระดับความเพี้ยนของเสียงที่รับฟังได้มีค่าน้อยมาก ทำให้ผู้รับฟังสามารถแยกแยะได้อย่างถูกต้องว่าเป็นความถี่ประเภทใด ส่วนของระดับความดังสูงสุดก็มีค่าพอสมควร แต่มีข้อสังเกตอยู่ว่า ถ้าปรับค่าความดัง

ถึงระดับสูงสุด จะเกิดความเพี้ยนมาก ดังนั้นการปรับระดับความดังที่เหมาะสมนอกจากจะทำให้ผู้ฟังไม่รำคาญแล้ว ยังต้องไม่ให้สัญญาณที่ได้ยินมีความผิดเพี้ยนมากจนเกินไปด้วย

3. การยกและวางหูโทรศัพท์

ในการควบคุมสภาวะของสายโทรศัพท์ผ่านเอาท์พุทพอร์ท 1 บิทนี้ สามารถทำได้อย่างถูกต้อง รีเลย์ทำงานตามจังหวะที่ต้องการ แต่มีข้อบกพร่องเล็กน้อยในส่วนเริ่มต้นการทำงานที่รีเลย์จะทำการยกหูขึ้นช่วงเวลาหนึ่ง ทั้งนี้เพราะวงจรควบคุมต้องใช้เวลาช่วงหนึ่งก่อนที่จะกำหนดพอร์ทให้ใช้งานได้อย่างถูกต้อง

4. การตรวจสอบสัญญาณกริ่ง

เมื่อมีสัญญาณกริ่งเข้ามา วงจรสามารถตรวจจับได้อย่างถูกต้องตามจังหวะของสัญญาณกริ่ง แต่จะมีการหน่วงเวลาบ้างเล็กน้อยเนื่องจากการเก็บประจุของคาร์ปาซิเตอร์ในวงจร

5. การตรวจจับความถี่

เมื่อมีสัญญาณพาหะหรือสัญญาณตอบรับที่มีค่าความถี่ตรงกับที่ได้ปรับเอาไว้ ผ่านเข้ามาในวงจร phase lock loop จะให้เอาท์พุทที่เป็น "0" แต่มีข้อพึงระวังอยู่ตรงที่วงจรนี้สามารถตรวจจับความถี่ที่ใกล้เคียงกับค่าที่ตั้งไว้ได้ ทำให้เกิดความยากลำบากที่จะตรวจจับความถี่ที่มีค่าใกล้เคียงกัน 2 ความถี่ขึ้นไป

7.4 สรุปและวิจารณ์ผลการทำงานของระบบ

จากการทดลอง การทำงานของระบบอ่านบัตรแม่เหล็ก การรับส่งข้อมูลและจัดลำดับการควบคุมระหว่างคอมพิวเตอร์และโมเด็ม การแสดงผลการทำงานทั้งทางจอคอมพิวเตอร์และทางจอ LCD ส่วนของระบบย่อยเหล่านี้สามารถทำงานได้อย่างถูกต้องตามโปรแกรม แต่ยังขาดความสมบูรณ์ของโปรแกรมในบางส่วน รวมทั้งการรับส่งข้อมูลยังมีข้อบกพร่องที่ควรจะต้องแก้ไขและยังมีบางจุดที่ควรจะเพิ่มเติม เพื่อให้ระบบมีความสมบูรณ์มากขึ้น

ในการพิจารณาว่า ระบบทั้งหมดนี้มีความถูกต้องตามที่ต้องการหรือไม่นั้น นอกจากจะต้องตรวจสอบการทำงานในส่วนของวงจรต่างๆ แล้ว ยังต้องรวมถึงการที่ระบบต่างๆ สามารถทำงานได้อย่างสัมพันธ์กันด้วย ซึ่งการทำงานที่สอดคล้องกันนี้เป็นหน้าที่ของส่วนควบคุมที่จะต้องจัดระบบโดยอาศัยโปรแกรมที่เขียนขึ้น

บทที่ 8

ปัญหาที่พบและข้อเสนอแนะ

8.1 ปัญหาที่พบในการสร้างโครงงาน

โครงงานนี้ถึงแม้จะถูกวางแผนการทำงานไว้อย่างเป็นลำดับขั้นตอน โดยเริ่มจากการหาข้อมูลในส่วนสำคัญต่างๆ ของระบบ ทำการศึกษาความเป็นไปได้ในการสร้าง รวมทั้งการทดลองและแก้ไขในส่วนต่างๆ จนสมบูรณ์ระดับหนึ่ง แต่เนื่องจากส่วนต่างๆ ของระบบนี้ มีความยุ่งยากซับซ้อนแตกต่างกัน ปัญหาที่เกิดขึ้นในการออกแบบและสร้างจึงแตกต่างกัน โดยเราสามารถแยกแยะปัญหาตามส่วนต่างๆ ของโครงสร้างระบบได้ดังนี้

ปัญหาที่พบในวงจรอ่านบัตรแม่เหล็ก

โดยทั่วไปแล้ว รายละเอียดและโครงสร้างของระบบอ่านบัตรแม่เหล็กนี้ค่อนข้างจำกัดแต่เฉพาะในกลุ่มอุตสาหกรรมเพื่อการผลิต ทำให้การศึกษาขั้นตอนและวิธีการอ่านบัตรแม่เหล็กนั้นทำได้ไม่ง่ายนัก ทั้งนี้เนื่องจากเราไม่สามารถค้นคว้าจากหนังสือวิชาการทั่ว ๆ ไปได้ รายละเอียดส่วนใหญ่จึงได้จากโครงงานที่ได้ทำการศึกษาไว้แล้ว รวมกับข้อมูลจาก ISO เท่าที่หามาได้ (ISO 7811/2) ซึ่งจะมีเฉพาะข้อมูลทางกายภาพ (physical characteristic) และข้อมูลที่เกี่ยวข้องกับการบันทึกของบัตรแม่เหล็ก เป็นส่วนใหญ่ แม้ว่ารายละเอียดทั้งหมดนี้เพียงพอที่จะทำให้ทราบโครงสร้างเกือบทั้งหมด แต่ในส่วนของวงจรซึ่งเป็นหัวใจของโครงงานนั้น ผู้สร้างได้มาจากเครื่องต้นแบบที่มีจำหน่ายในปัจจุบัน ทำให้ชิ้นงานจากวงจรนี้มีขนาดเล็กและมีความน่าเชื่อถือสูงขึ้น แม้ว่าหัวใจของวงจร ซึ่งก็คือ IC TB54910P นี้จะเป็นไอซีเฉพาะ (customer chip) ไม่มีวางจำหน่ายในท้องตลาดทั่วไป ซึ่งทำให้ข้อมูลในส่วนของโครงสร้างภายในหรือการใช้งานในบางจุดยังไม่ครบถ้วน แต่การใช้งานที่จำเป็น การต่อขาอินพุทและเอาท์พุท รวมทั้งการต่อตัวต้านทานและตัวเก็บประจุเพื่อไบอัสให้กับไอซีสามารถทำได้อย่างถูกต้อง ปัญหาในเรื่องของความคลุมเครือในการใช้งานจึงหมดไป

ในส่วนของการเขียนโปรแกรมควบคุมการอ่านบัตรแม่เหล็ก ปัญหาที่พบ ประกอบด้วย 2 ส่วนหลัก ส่วนแรก เกิดจากผู้สร้างยังไม่ชำนาญในการเขียนโปรแกรมในการควบคุมพอร์ทในระดับบิท ทำให้ต้องใช้เวลาในการศึกษาพอสมควร และอีกส่วนหนึ่งคือ การตรวจสอบข้อมูลที่อ่านได้ว่าถูกต้องตามมาตรฐานหรือไม่ ขณะเดียวกันก็ต้องแปลงข้อมูลดิบนี้ให้เป็นข้อมูลจริงที่เป็นตัวเลขโดยการตัดบิทควบคุมออก ซึ่งต้องใช้เวลาในการเขียนนานพอสมควร

ปัญหาที่พบในวงจรโมเด็ม

ในขั้นต้น ผู้สร้างประสบปัญหาบางประการที่ดูเหมือนจะไม่สำคัญเท่าใดนัก แต่สร้างความหนักใจให้พอสมควร คือ การเลือกใช้ไอซีที่ทำหน้าที่มอด-ดีมอดดูเลท ทั้งนี้เนื่องจากไอซีประเภทนี้มีจำหน่ายค่อนข้างน้อยในประเทศไทย และจำกัดอยู่ที่อัตราในการรับส่งข้อมูลที่ต่ำ เหตุผลแรกที่ตัดสินใจเลือกไอซี MC145450 นั้นเป็นเพราะมีอัตราเร็วในการรับส่งค่อนข้างสูงกว่าไอซีตัวอื่น นอกจากนี้ยังมีราคาที่ไม่แพงนัก แต่ปัญหาที่ตามมาในทันที ก็คือ ข้อมูลเกี่ยวกับไอซีตัวนี้มีน้อยมาก เพราะมีแค่ใน data sheet เพียง 6 หน้าเท่านั้น นอกจากนี้ในตัวอย่างการใช้งานก็เป็นการใช้งานกับสายเช่า (leased line) มากกว่าที่จะเป็นสายโทรศัพท์ทั่วไป ดังนั้นเมื่อตัดสินใจใช้ไอซีนี้เป็นหัวใจของวงจรโมเด็มแล้ว การออกแบบส่วนที่ทำงานร่วมกัน จึงต้องอาศัยแนวทางดังที่ได้กล่าวมา

ส่วนที่คิดว่าบกพร่องและมีปัญหามากที่สุด คือ วงจร Duplexer ที่จะต้องแยกแยะส่วนรับและส่วนส่งให้แยกจากกันเด็ดขาด ทั้งนี้เนื่องจากเมื่อพิจารณาตัวอย่างจากใน data sheet แล้ว การสร้างได้รองรับสายเช่า 4 เส้น โดยการรับและส่งเกิดขึ้นพร้อมกันอย่างละคู่สาย แต่เมื่อเรานำมาใช้กับสายโทรศัพท์ที่มีเพียง 2 เส้น ตามหลักการนั้นสามารถทำได้ แต่เนื่องจากความถี่ในการรับและส่งไม่ตรงกัน จึงมีข้อแม้ว่าจะต้องมีวงจร Duplexer ที่สมบูรณ์มากกว่าที่ใช้ในโครงงานนี้ จึงจะทำให้การรับส่งข้อมูลมีความผิดพลาดน้อยลง

8.2 ข้อเสนอแนะและแนวทางในการปรับปรุงโครงงาน

จากปัญหาและข้อบกพร่องที่เกิดขึ้นรวมทั้งเพื่อความสมบูรณ์ของโครงงานนี้ เราจึงควรพัฒนาส่วนต่างๆ ของโครงงาน แนวทางการปรับปรุงและแก้ไขมีดังนี้

1. เปลี่ยนแปลงส่วนของวงจร Duplexer ให้มีประสิทธิภาพสูงขึ้นในการแยกสัญญาณที่จะรับและส่ง ซึ่งจะทำให้ข้อผิดพลาดในการรับส่งในจังหวะที่มีการสวนกันมีค่าลดลง จนอาจจะไม่มีข้อผิดพลาดเลย การปรับปรุงนี้ทำได้โดยใช้อุปกรณ์ประเภท hybrid แทนวงจร Duplexer แต่จะมีข้อเสีย คือ จะทำให้วงจรมีขนาดใหญ่ แนวทางที่ดีที่สุดคือการหาอุปกรณ์ขนาดเล็กหรือไอซีที่ทำหน้าที่เป็นวงจร Duplexer เลย หรืออาจใช้ไอซีโมเด็มที่มีวงจร Duplexer ในตัว แต่จะทำให้ราคาสูงขึ้น

2. เพิ่มอัตราเร็วในการรับส่งข้อมูล แม้ว่าการเพิ่มอัตราเร็วจะทำได้ยาก เนื่องจากต้องเปลี่ยนระบบมอด-ดีมอดดูเลททั้งหมด แต่อัตราเร็วที่เพิ่มขึ้นนี้เป็นส่วนที่เพิ่มประสิทธิภาพของระบบให้สูงขึ้นเป็นอย่างมาก การเพิ่มอัตราเร็วทำได้โดยการเปลี่ยนระบบมอด-ดีมอดดูเลททั้งหมดให้เป็นแบบ PSK หรือ QPSK ขึ้นกับค่าอัตราเร็วที่ต้องการ แต่การออกแบบวงจรมอด-ดีมอด

ดูเลทแบบใหม่นี้จะมีความยุ่งยากมากกว่า FSK ซึ่งเป็นจุดที่ต้องคำนึงถึงอีกประการหนึ่งในการปรับปรุงอัตราเร็วด้วยวิธีนี้

3. เมื่อทำการปรับปรุงวงจร Duplexer และเพิ่มอัตราเร็วในการรับส่งข้อมูลให้พอเหมาะแล้ว สิ่งที่ควรเพิ่มเติม คือ ระบบรักษาความปลอดภัยที่สูงขึ้น จากเดิมที่เราเก็บบัญชีผู้มีสิทธิ์ใช้ระบบและตรวจสอบสิทธิ์ของผู้กำลังใช้ระบบไว้ที่ Terminal ทำให้ผู้ใช้ที่ไม่มีสิทธิ์แอบลอบเข้ามาใช้ระบบนี้และขโมยข้อมูลที่ส่งจาก Host ได้ ดังนั้นเพื่อสร้างให้ระบบมีความปลอดภัยสูงขึ้น บัญชีผู้ใช้และการตรวจสอบควรเกิดขึ้นที่ Host มากกว่า แต่จะทำให้การเข้าถึงระบบมีความล่าช้าในการตรวจสอบไปบ้างแต่ไม่มากนักถ้าอัตราเร็วในการรับส่งข้อมูลมีค่าสูงพอ

8.3 บทสรุป

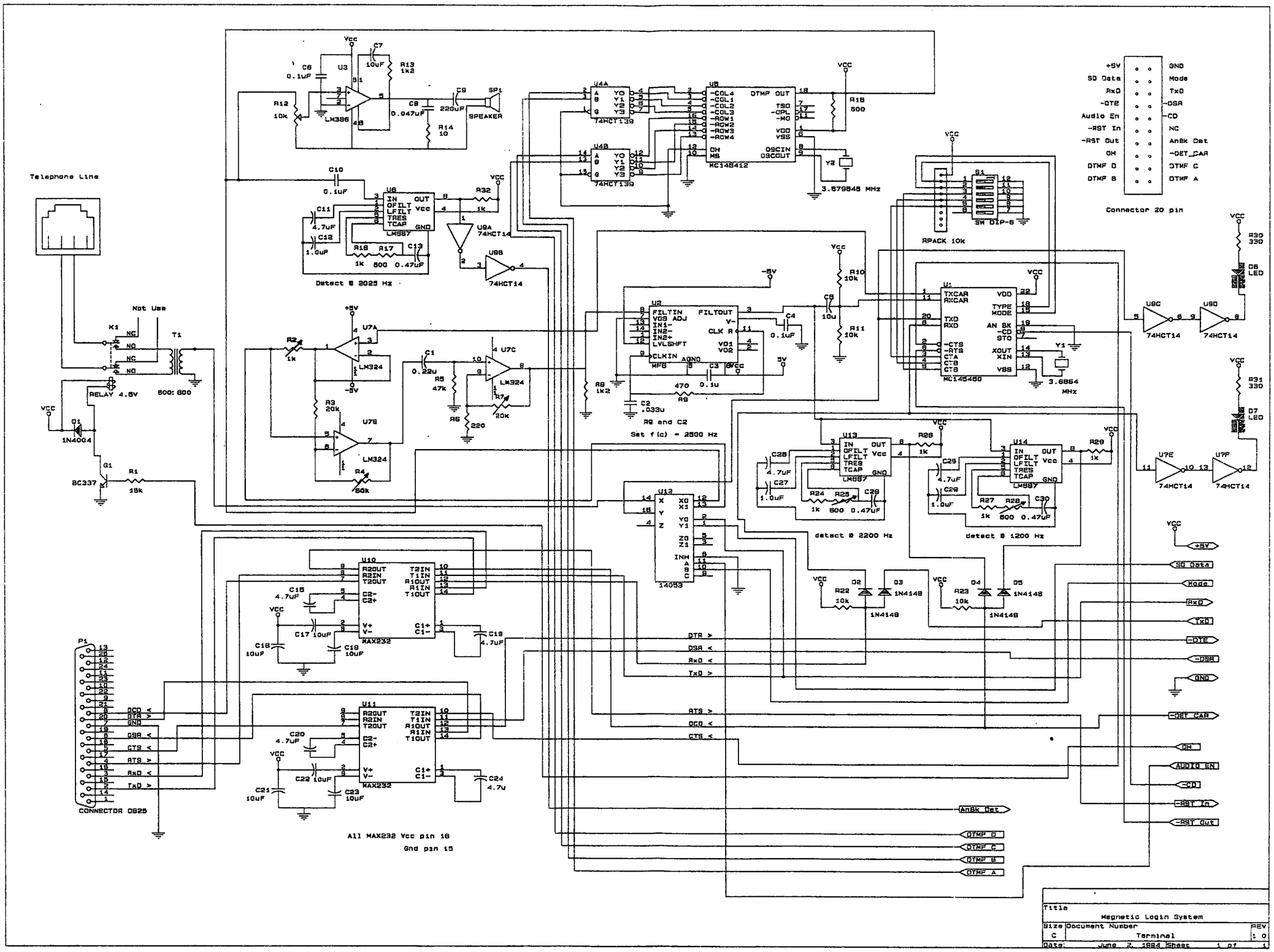
ในขณะที่เทคโนโลยีก้าวรุดหน้าไปอย่างรวดเร็วอย่างในปัจจุบัน สินค้าทางเทคโนโลยีที่มีจำหน่ายได้มีการพัฒนาประสิทธิภาพให้สูงขึ้นอย่างรวดเร็ว จนอาจทำให้โครงงานนี้ล้าหลังไปบ้าง เช่น ในส่วนของโมเด็ม ปัจจุบันนี้สามารถรับส่งข้อมูลผ่านเครือข่ายโทรศัพท์ด้วยความเร็วสูงถึง 28.8 kbps แต่ก็ต้องใช้เทคนิคหลายๆ อย่าง ทั้งการบีบอัดข้อมูล การประมวลผลเชิงดิจิตอล และอื่นๆ เพื่อให้ได้มาซึ่งความเร็วที่สูงขนาดนี้ โครงงานนี้จึงเป็นเพียงพื้นฐานสำหรับการพัฒนาโครงงานที่ใช้เทคโนโลยีที่สูงขึ้นต่อไป

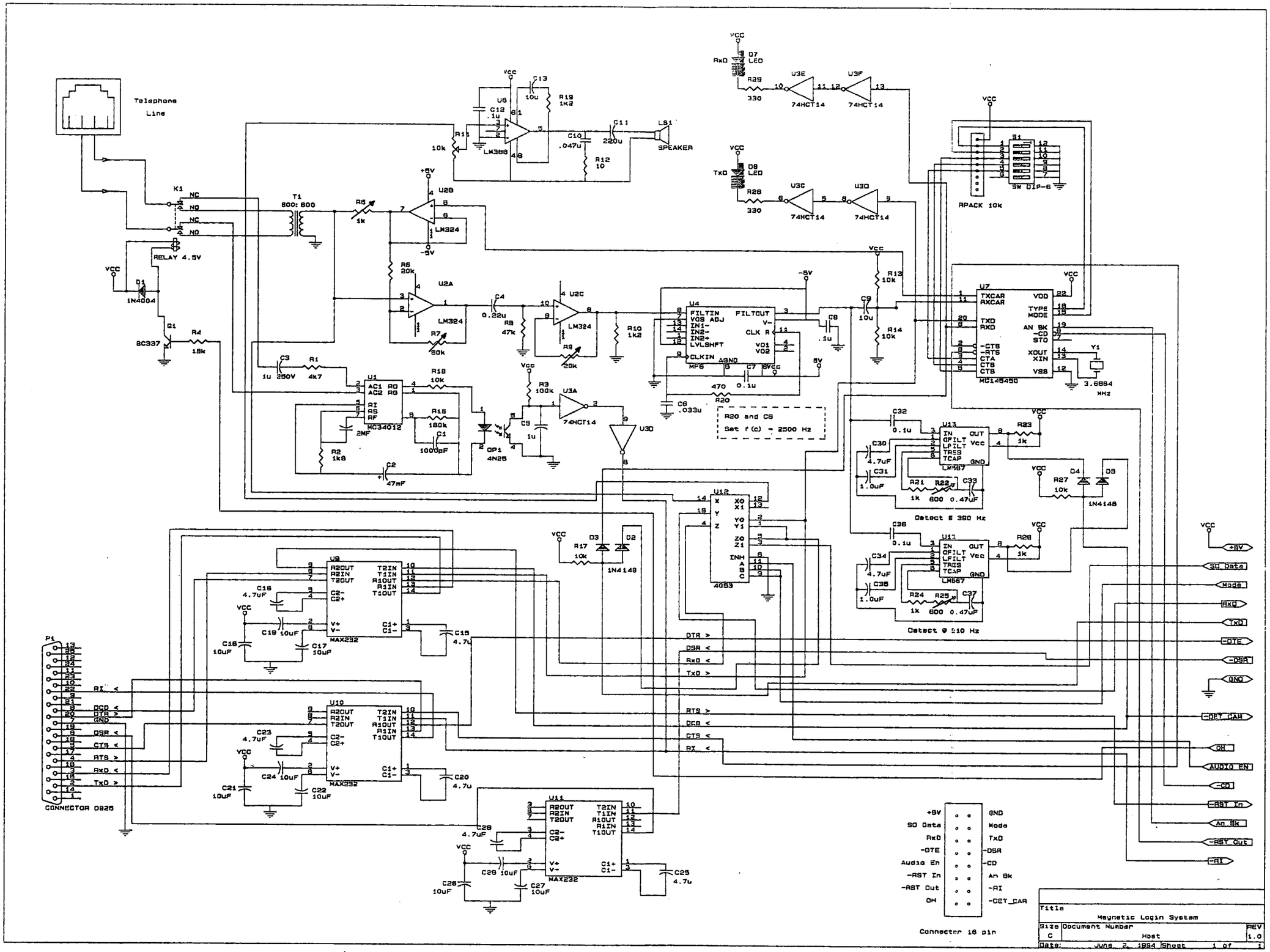
แม้ระบบนี้จะยังมีข้อบกพร่องอยู่หลายจุด รวมทั้งยังมีบางจุดที่ควรจะเพิ่มเติมขึ้นเพื่อให้ระบบมีความสมบูรณ์มากขึ้น แต่ด้วยเวลาและงบประมาณที่จำกัด อีกทั้งความรู้ความเข้าใจในระบบสื่อสารก็มีไม่มากนัก ทำให้โครงงานนี้ไม่สามารถสร้างตามจุดมุ่งหมายที่กำหนดไว้ในบทแรกได้ทั้งหมด แต่อย่างไรก็ตาม ปัญหาที่เกิดขึ้นในโครงงานนี้จะเป็นตัวส่งเสริมให้เกิดความรู้และความเข้าใจในระบบสื่อสารมากยิ่งขึ้น เพื่อที่จะใช้เป็นแนวทางในการพัฒนางานชิ้นอื่นๆ ที่ยากและซับซ้อนยิ่งขึ้นต่อไป

ภาคผนวก ก.

ผังวงจรสมบูรณ์ในส่วนของระบบโมเด็ม

ของ Terminal และ Host


Telephone Line
Not Use
K1
T1
RELAY 4.5V
600: 600
D1
1N4004
Q1
BC337
R1
15k
U3
LM386
C6
0.1uF
C7
10uF
R13
1k2
R12
10k
C8
0.047uF
C9
220uF
SP1
SPEAKER
R14
10
C10
0.1uF
U6
LM887
C11
4.7uF
C12
1.0uF
R16
1k
R17
600
C13
0.47uF
Detect @ 2025 Hz
U9A
74HCT14
R32
1k
U9B
74HCT14
U4A
U4B
74HCT139
U5
MC148412
3.579545 MHz
R15
500
+5V
U7A
LM324
R2
1k
C1
0.22u
U7C
R5
47k
R7
20k
R6
220
R3
20k
U7B
R4
50k
-5V
U2
FILTIN
VOS ADJ
IN1-
IN2-
IN2+
LVLSHFT
CLKIN
FILTOUT
V-
CLK R
VO1
VO2
AGND
MFS
C3
0.1u
C4
0.1uF
R9
470
R8
1k2
C2
.033u
R9 and C2
Set f(c) = 2500 Hz
R10
10k
C5
10u
R11
10k
U1
MC145480
TXCAR
RXCAR
TXD
RXD
-CTS
-RTS
CTA
CTB
CTB
VDD
TYPE
MODE
AN BK
-CD
STD
XOUT
XIN
VSS
Y1
3.6864
MHz
RPACK 10k
SW DIP-8
Connector 20 pin
+5V
SD Data
RxD
-DTE
Audio En
-RST In
-RST Out
OH
DTMF D
DTMF B
GND
Mode
TxD
-DSR
CD
NC
AnBk Dat
-DET_CAR
DTMF C
DTMF A
U8C
U8D
74HCT14
D6
LED
R30
330
U7E
U7F
D7
LED
R31
330
U13
LM887
C28
4.7uF
C27
1.0uF
R24
1k
R25
600
C29
0.47uF
R26
1k
detect @ 2200 Hz
U14
LM887
C29
4.7uF
C29
1.0uF
R27
1k
R28
600
C30
0.47uF
R29
1k
detect @ 1200 Hz
U12
14053
INH
A
B
C
X
Y
Z
XO
X1
YO
Y1
ZO
Z1
D2
D3
D4
D5
1N4148
R22
10k
R23
10k
U10
MAX232
C15
4.7uF
C17
10uF
C16
10uF
C18
10uF
C18
4.7uF
U11
MAX232
C20
4.7uF
C22
10uF
C21
10uF
C23
10uF
C24
4.7u
P1
CONNECTOR DB25
DCD <
DTR >
GND
DSR <
CTS <
RTS >
RxD <
TxD >
DTR >
DSR <
RxD <
TxD >
RTS >
DCD <
CTS <
All MAX232 Vcc pin 16
Gnd pin 15
AnBk Dat
DTMF D
DTMF C
DTMF B
DTMF A
Title
Magnetic Login System
Size
C
Document Number
Terminal
REV
1.0
Date:
June 2, 1994
Sheet
1 of 1

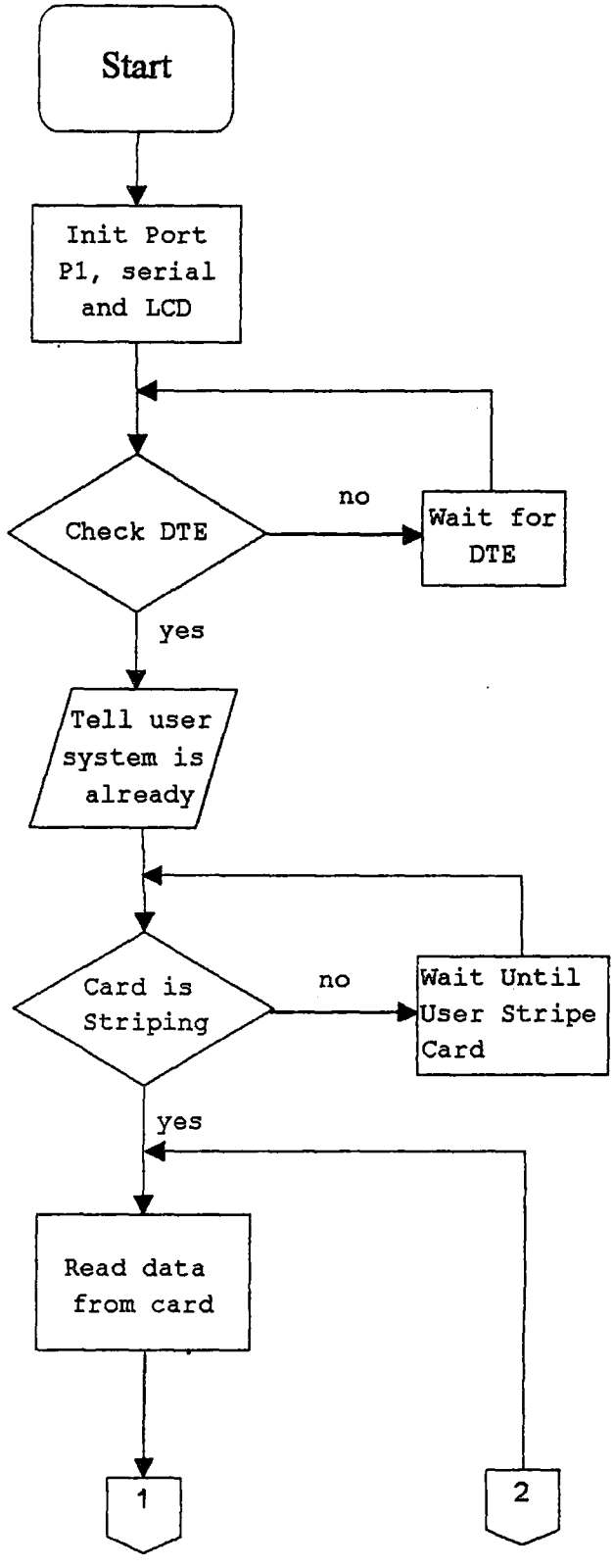

Telephone Line
K1
NC
NO
NC
NO
RELAY 4.5V
T1
600: 600
R5
1k
U2B
LM324
+5V
-5V
U2A
LM324
U2C
LM324
R6
20k
R7
50k
R8
47k
R9
20k
R10
1k2
C4
0.22u
D1
1N4004
Q1
BC337
R4
18k
C3
1u 250V
R1
4k7
U1
MC34012
2MF
R18
10k
R16
180k
C1
1000pF
R2
1k8
C2
47mF
R3
100k
U3A
74HCT14
C5
1u
OP1
4N26
U3D
U4
FILTIN
VOS ADJ
IN1-
IN2-
IN2+
LVLSHFT
CLKIN
MFS
FILTOUT
V-
CLK R
VO1
VO2
AGND
C7 .1u
VCC
C8
.033u
R20
470
0.1u
R20 and C5
Set f(c) = 2500 Hz
R12
10k
C9
10u
R14
10k
5V
U12
4053
X
Y
Z
XO
X1
YO
Y1
ZO
Z1
INH
A
B
C
U7
TXCAR
RXCAR
TXD
RXD
-CTS
-RTS
CTA
CTB
CTB
VDD
TYPE MODE
AN BK
-CD
STO
XOUT
XIN
VSS
MC145450
Y1
3.6864 MHz
U13
IN
OFILT
LFILT
TRES
TCAP
OUT
Vcc
GND
LM567
C32
0.1u
C30
4.7uF
C31
1.0uF
R21
1k
R22
600
C33
0.47uF
Detect @ 380 Hz
R23
1k
C36
0.1u
C34
4.7uF
C35
1.0uF
R24
1k
R25
600
C37
0.47uF
Detect @ 510 Hz
R28
1k
D7
LED
RxD
R29
330
U3E
U3F
74HCT14
D8
LED
TxD
R26
330
U3C
U3D
74HCT14
SW DIP-6
RPACK 10k
D4
D5
R27
10k
1N4148
D3
D2
R17
10k
1N4148
U8
U10
U11
MAX232
R2OUT
R2IN
T2OUT
T2IN
T1IN
R1OUT
R1IN
T1OUT
C2-
C2+
V+
V-
C1+
C1-
C18
4.7uF
C19
10uF
C16
10uF
C17
10uF
C15
4.7u
C23
4.7uF
C24
10uF
C21
10uF
C22
10uF
C20
4.7u
C28
4.7uF
C29
10uF
C26
10uF
C27
10uF
C25
4.7u
P1
CONNECTOR DB25
RI <
DCD <
DTR >
GND
DSR <
CTS <
RTS >
RxD <
TxD >
DTR >
DSR <
RxD <
TxD >
RTS >
DCD <
CTS <
RI <
+8V
SD Data
RxD
-DTE
Audio En
-RST In
-RST Out
OH
GND
Mode
TxD
-DSR
-CD
An Bk
-RI
-DET_CAR
Connector 18 pin
+8V
SD Data
Mode
RxD
TxD
-DTE
-DSR
GND
-DET_CAR
OH
AUDIO EN
-CD
-RST In
An Bk
-RST Out
-RI
Title
Magnetic Login System
Size C
Document Number
Host
REV 1.0
Date: June 2, 1994 Sheet 1 of 1

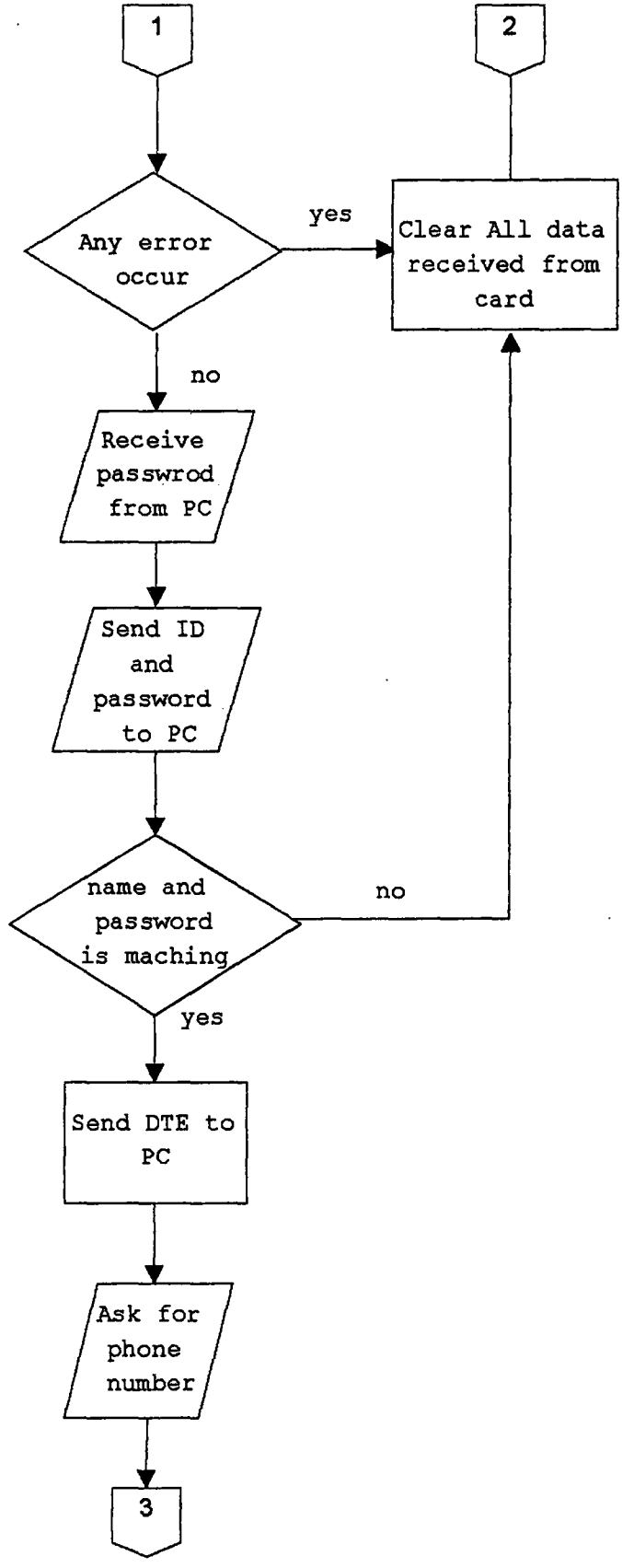
ภาคผนวก ข.

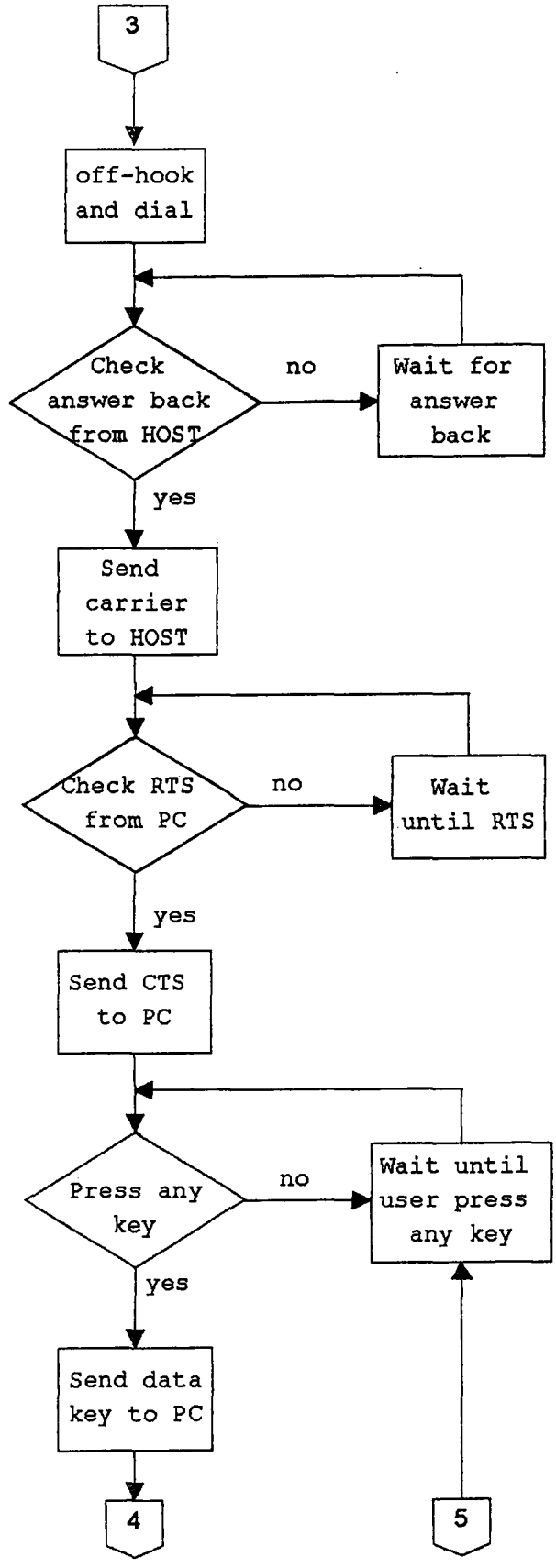
ผังแสดงลำดับขั้นตอนการทำงาน

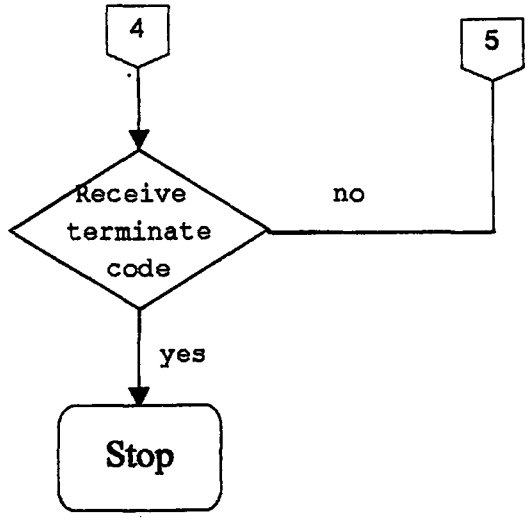
ของ Terminal และ Host

ผังแสดงลำดับขั้นตอนการทำงานของ Terminal

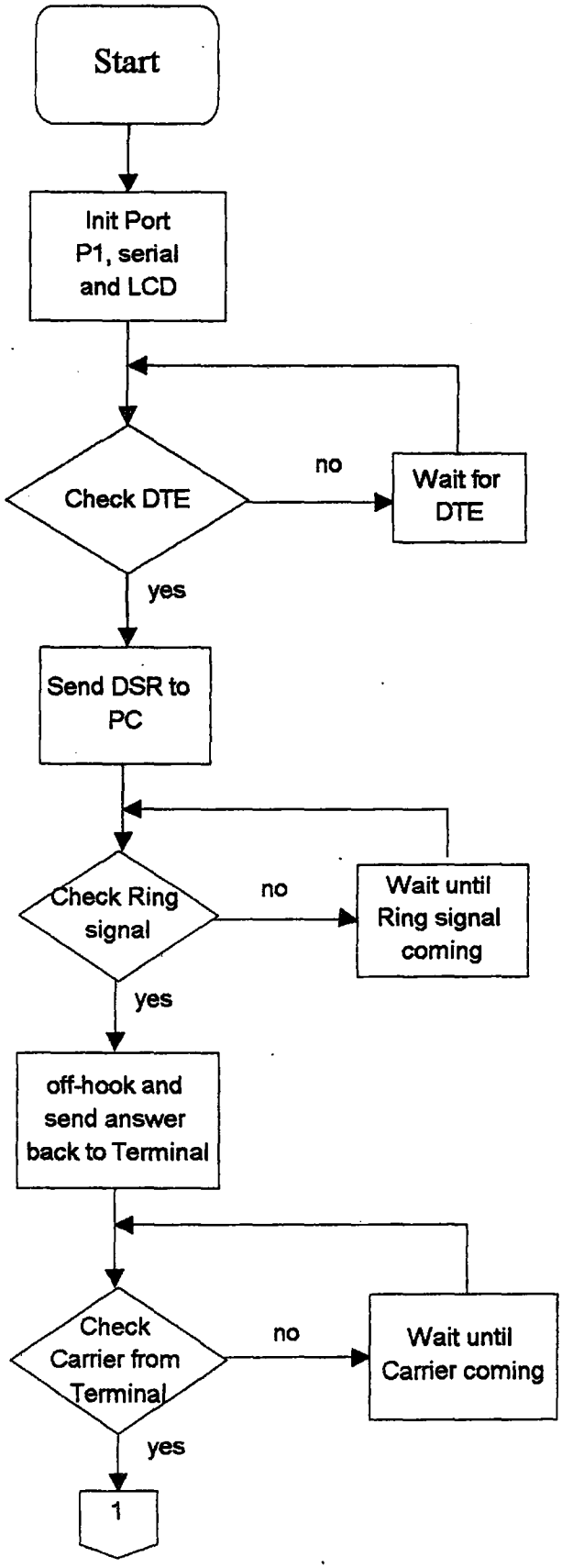


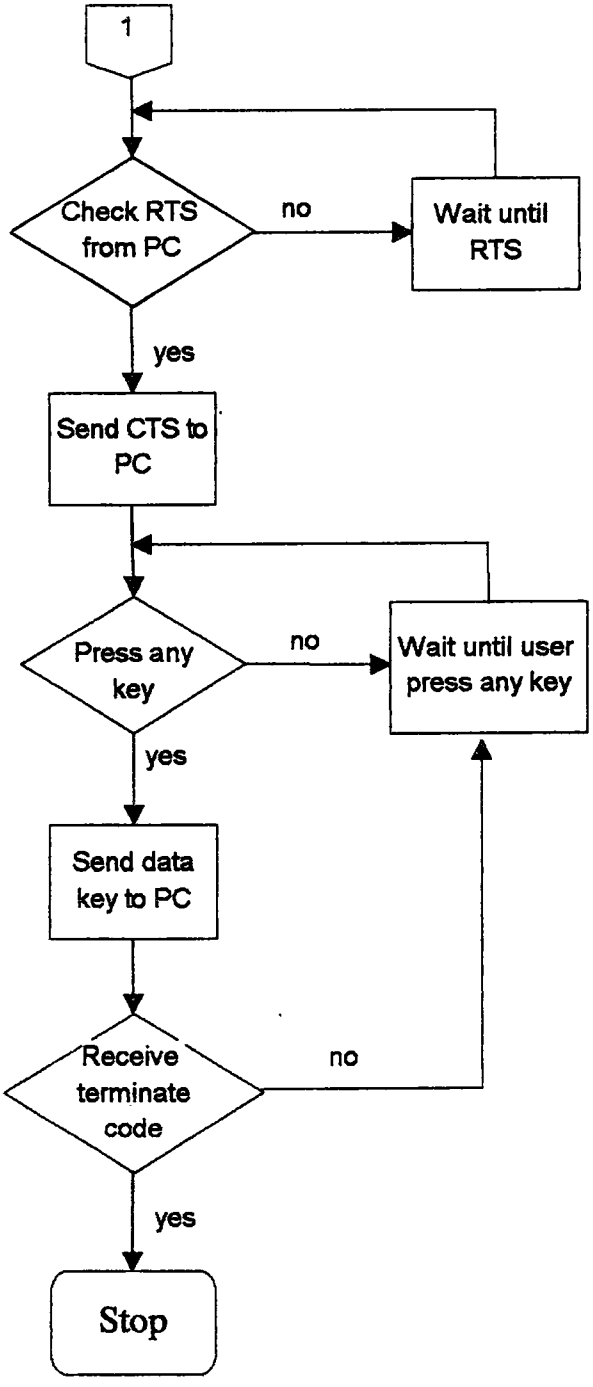

1
2
Any error occur
yes
Clear All data received from card
no
Receive passwrod from PC
Send ID and password to PC
name and password is maching
no
yes
Send DTE to PC
Ask for phone number
3


3
off-hook and dial
Check answer back from HOST
no
Wait for answer back
yes
Send carrier to HOST
Check RTS from PC
no
Wait until RTS
yes
Send CTS to PC
Press any key
no
Wait until user press any key
yes
Send data key to PC
4
5


4
5
Receive terminate code
no
yes
Stop

ผังแสดงลำดับขั้นตอนการทำงานของ Host




1
Check RTS from PC
no
Wait until RTS
yes
Send CTS to PC
Press any key
no
Wait until user press any key
yes
Send data key to PC
Receive terminate code
no
yes
Stop

ภาคผนวก ค.

โปรแกรมภาษาแอสเซมบลีของ 8031

```
;***************************** TERMINAL.ASM ******************************
;
;
; FILENAME        TERMINAL
; DESCRIPTION     TERMINAL ENBEDING CONTROL PROGRAM
; HARDWARE        Magnetic Login Terminal
; ASSEMBLER       SXA51.EXE
; PRODUCE BY      MR NIPON JANTRI  SCIENCE'9 APPLY PHYSIC 33504016
;
;*************************************************************************
; ## The following section is reserved for ALL VARIABLE which involve this
;    system  and constant
;*************************************************************************

;****************** Magnetic Read  VARIABLE ****************************
 S_BUF      EQU   8500H       ; sample buffer
 S_LIM      EQU   50          ; sample limit
 D_BUF      EQU   8550H       ; decode buffer
 PASSWD     EQU   8600H       ; password buffer
 RESULT_I   EQU   30H         ; result buffer in data area
 RESULT_E   EQU   8610H       ; result buffer in code area
 PHONE_NUMBER  EQU  8630H     ; telephone number buffer

 ;INPUT PORT DEFINE
 CP         EQU   90H         ; card present bit -> Port 1.0
 DQ         EQU   91H         ; clock bit       -> Port 1.1
 CLK        EQU   92H         ; data bit        -> Port 1.2

 ; CODE FOR TRACK 2
 M_SS       EQU   1011B         ; start sentinel 0Bh
 M_ES       EQU   1111B         ; end sentinel   0Fh

;********************** LCD  VARIABLE ******************************

LCDWRC      EQU      0FF80H          ; LCD Write Control
LCDRDC      EQU      0FF81H          ; LCD Read  Control
LCDWRD      EQU      0FF82H          ; LCD Write Data
LCDRDD      EQU      0FF83H          ; LCD Read  Data
LCDBUF      EQU      8000H           ; LCD RAM Buffer
```

```
;****************  Port Configuration for Terminal ***************
; ## Port Configuration is the designation of each pin for each port.
;    It is designed esspecially to support MLS hardware which is the
;    heart of this system
;*********************************************************************

;*****  8031 Port Configuration *****
;
;    P1.0     |
;    P1.1      >   reserve for magnetic card
;    P1.2     |
;    P1.3     -Sd DATA          send data if Mode is in transmit mode
;    P1.4     AnBk         0 for not Answer back        1  for Answer back
;    P1.5     -DTE         0 for IBM ready              1  for not ready
;    P1.6     -RTS IN      0 for IBM request to send    1  for not request
;    P1.7     -CAR DET     0 for Detect Carrier         1  for not detect
;
;******* 8255 Port Configuration *******
;
;    PA.0     Mode         0 for initial mode          1  for transmit mode
;    PA.1     OH           0 for ONHOOK               *1  for  OFFHOOK
;    PA.2     Audio En     0 for speaker ON            1  for connect to MC145450
;    PA.3     -CD         *0 for detect carrier        1  for not detect carrier
;    PA.4     -RTS out    *0 for set MC145450 ready    1  for not in trannit
;    PA.5     -DSR        *0 for tell IBM that Data Set Ready
;    PA.6     -
;    PA.7     -
;
;    PB.0     DTMF A
;    PB.1     DTMF B
;    PB.2     DTMF C
;    PB.3     DTMF D
;    PB.4 - PB.5   Available
;
;******************   PORT VARIABLE      ***************************
```

```
;WARNING !!!!   8255 Port start at FF00

PORTBASE   EQU   0FF00h
PORTA      EQU   PORTBASE + 0          ; Port A
PORTB      EQU   PORTBASE + 1          ; Port B
PORTC      EQU   PORTBASE + 2          ; Port C
CONP       EQU   PORTBASE + 3          ; 8255 Control Port
CONW       EQU   10001000B           ; Control Word: all port set to mode 0
                                     ;    Port A:               out
                                     ;    Port C high nibble:   in
     ;    Port B:               out
                                     ;    Port C low  nibble:   out
                                     ;* remark: see port configuration above

; Port Pin Constant

MODE       EQU   00000001b           ; Mode
OH         EQU   00000010b           ; On Hook
AUD_EN     EQU   00000100b           ; Audio Enable
CD_L       EQU   00001000b           ; - Carrier Detect
RTS_O_L    EQU   00010000b           ; - Request To Sent OUT
DSR_L      EQU   00100000b           ; - Data Set Ready

Sd_D       EQU   P1.3                ; Send Data
AnBk       EQU   P1.4                ; Answer Back
DTE_L      EQU   P1.5                ; - Data Terminal Equipment
RTS_I_L    EQU   P1.6                ; - Request To Send IN
CAR_D      EQU   P1.7                ; - Carrier Detect

; end of Port Constant
;*********************  END OF VARIABLE ******************************



;****************** Main Main Main Main Main Main  ***********************
;
;                  ** Main Program Start Here
;
;*************************************************************************
```

```
            org 0h              ; for EPROM or EPROM Emulator

            mov R2, #0ffh
            lcall delay
            mov R2, #0ffh
            lcall delay

 ; this area reserve for all interrupt which is used in the future


;*****************************************************************
;
; 1.  Initial All System Port which are 8031 port, serial port,
;     8255 port and LCD
;
;*****************************************************************

start:   ; Initialize All Port
         mov A, #11101111b    ; Card Present bit  input port latch "1"
      ; Clock Bit         "------------------"
                            ; Data Bit          "------------------"
      ; Sd Data  1 for mark state
      ; AnBk     0  for not answer
      ; -DTE      input port   latch "1"
      ; -RTS IN   "--------------------"
      ; -CAR DE   "--------------------"

 mov P1, A

         mov DPTR, #CONP
 mov A, #CONW
 movx @DPTR, A        ; Set All port to output except C High

 mov DPTR, #PORTA
         ;mov A, #00001111b
         mov A, #00111100b    ;initial state, wait for ring signal
      ; Mode        0  Initial
                            ; OH          0  ON-HOOK
```

```
                              ; Audio En     1  Connect to MC145450
                              ; -CD          1  not detect carrier
      ; -RTS Out     1  not in transmit
                              ; -DSR         1  Modem not ready

         movx @DPTR, A
 mov  DPTR, #PORTB
 mov A, #11111111b   ; set All DTMF control bit to 1
 movx  @DPTR, A



init_serial:
 mov  SCON, #01010010b   ;Set SCON (Serial Port Control Register)
 ;Mode 1
 ;REN  =  1 (Enable)
 ;Ti   =  1
 mov  TMOD, #00100010b   ;Set TMOD (Timer)
 ;Mode 2  (Auto Reload 8 bit)
 mov  TH1, #11101000b    ; Set Baud RATE = 1200
    ;mov  TH1,  #01000000b  ;                  150
    ;mov  TH1, #11111101b   ;                 9600
 mov  TH2, #01000000b    ; set Timer 2 to Baud Rate 150


 setb TR1                  ; set timer 1

 lcall initlcd

;-------------------   Initialize Complete ------------------->



;************** Check  RS232 Condition ************************
; 2. Handshake condition. check if IBM available
;    Wait Until DTE from IBM is "ON"
;    If IBM DTE is "ON" send information
;
;*****************************************************************
```

```
rs232_check:
 mov DPTR, #PORTA
 movx A, @DPTR
 orl A, #DSR_L          ; make sure that DSR is "OFF"
 movx @DPTR, A          ; until DTE is "ON" and system is access

wait_DTE:
 jnb DTE_L, tell_info   ; if DTE "ON" jmp to tell information
 mov DPTR, #NOTREADY
 lcall LCDLDP
 mov R2, #0ffh
 lcall delay
 jmp wait_DTE           ; wait until DTE is "ON"

; ** comment: timing check may be used here

tell_info:
 mov DPTR, #infor1      ;tell user that modem ready
 lcall tword
 mov DPTR, #infor2      ;and how to access this system
 lcall tword
 mov DPTR, #infor3
 lcall tword


;---------------------  IBM ready now ----------------------------->


;*********************** MAGNETIC ROUTINE  ***************************
;
;3. Magnetic Reading Subroutine.
;   This subroutine is used for supporting the magnetic reader hardware
;   that read your ID number from magnetic card ,  sending this
;   ID number to computer(see routine 4 below)  and showing this ID
;   number on LCD and monitor.
;
;*********************************************************************
```

```
MAG_ROUTINE:
      mov DPTR, #STRIPE
      lcall LCDLDP


START_READ:
     LCALL MAG_SAMPLE
     JZ    START_READ

     LCALL MAG_DECODE
     JZ  PRINT_ERROR

;            LCALL CLRLCD
;            MOV DPTR, #TEST_READ
;            LCALL LCDLDP
;            MOV R2,0FFH
;            LCALL DELAY
;            MOV R2, OFFH
;            LCALL DELAY

;     MOV R0, #20
;     MOV DPTR, #D_BUF
;LOOP:       MOVX A, @DPTR
;     LCALL BYTE_TO_LCD
;     INC DPTR
;     DJNZ R0, LOOP
     lcall clrlcd
     mov DPTR, #read_ok
            lcall LCDLDP
     LJMP get_passwd

PRINT_ERROR:
     LCALL CLRLCD
     CLR A
     MOV DPTR, #ERROR_MESSAGE
     LCALL LCDLDP
     SJMP START_READ

READ_OK:        DB   "Card Read OK.",0h
```

```
STRIPE:          DB   "*** MLS 1.0 Ready ***Please Stripe CARD*", 0h
NOTREADY:        DB   "***** MLS  1.0 ****** IBM is not ready *", 0h
ERROR_MESSAGE:   DB   "Card Read Error!!!!  Stripe AGAIN       ", 0h

Infor1:     db  'Welcome to MLS 1.0',0ah,0dh,00h
Infor2:     db  'This message send from Terminal to your IBM',0ah,0dh,00
Infor3:     db  'You must stripe the card to access this sytem.',0ah,0dh,00



;**************************************
;
;4. Tell Information and get password
;   then Send ID and password to IBM
;
;**************************************

get_passwd:
 mov DPTR, #Message1        ;ask for password
 lcall tword
 mov DPTR, #PASSWD
passwd_loop:
 lcall rx
 cjne A, #0dh, passwd_cont
 jmp passwd_exit
passwd_cont:
 movx @DPTR, A
 mov A, #'*'
 lcall tx
 inc DPTR
 jmp passwd_loop

passwd_exit:
 mov A, #0h                 ;Null Terminate
 movx @DPTR, A
 mov A, #0dh
        lcall tx
 mov A, #0ah
        lcall tx
```

```
send_ID:
 mov DPTR, #Message2            ;send ID to IBM
 lcall tword

 MOV R0, #10
 MOV DPTR, #D_BUF
Send_ID_loop:
 MOVX A, @DPTR
 LCALL send_byte
 INC DPTR
 DJNZ R0, Send_ID_loop

 mov DPTR, #PASSWD              ;send password to IBM
Send_Passwd:
 movx A, @DPTR
 lcall tx
 inc DPTR
 cjne A, #0h, Send_Passwd

 mov DPTR, #RESULT_E
 mov R0, #RESULT_I
get_answer:  ; receive processing result from IBM
 lcall rx
 mov @R0, A
 movx @DPTR, A
 inc R0
 inc DPTR
 cjne A, #0h, get_answer


 mov R2, #0ffh
 lcall delay                ;wait until IBM ok

 lcall CLRLCD
 mov DPTR, #RESULT_E
 lcall LCDLDD

 mov R2,#0ffh
```

```
 lcall delay
 mov R2,#0ffh
 lcall delay

 mov R0, #RESULT_I
 mov DPTR, #logok

check_answer:
        clr  A
        movc A, @A+DPTR
 cjne A, #0h, ch?as

 cjne @R0, #0h, ch?as
 jmp check_ans_ok          ; OK. two string match

ch?as: subb A, @R0
 jnz check_ans_exit
 inc DPTR
 inc R0
 sjmp check_answer


check_ans_exit:   ; two string not match

loop_again:
   mov DPTR, #logerr       ;tell user that modem ready
   lcall tword
   ljmp MAG_ROUTINE

check_ans_ok:

set_DSR: lcall clrlcd


 mov DPTR, #PORTA
 movx A, @DPTR
 mov R1, A              ; OK, DTR is "ON"
 mov A, #DSR_L
 cpl A
```

```
 anl A, R1              ; set DSR  "ON" respond to DTE
 movx @DPTR, A

        mov DPTR, #msg_dsr
        lcall LCDLDP

        mov DPTR, #msg_get_num
 lcall tword
 clr A              ; Null Terminate
 lcall tx

 mov DPTR, #Phone_Number
Get_Number:
 lcall rx
 movx @DPTR,A
 inc DPTR
 cjne A, #0h, Get_Number

 lcall clrlcd
 mov DPTR, #Phone_Number
 lcall LCDLDD
 mov R2, #0ffh
 lcall delay
        mov R2, #0ffh
 lcall delay
        mov R2, #0ffh
        lcall delay

 ;####### TEST ONLY
        ;ljmp wait_to_send

        jmp on_hook

msg_dsr:    db  'Send DSR',0h
Message1:   db  'Please type your password',0ah,0dh,00h
Message2:   db  'OK. send your password and ID to IBM',0ah,0dh,00h
table:      db  'Modem Ready',0ah,0dh,00h
table1:     db  'Transfer All Control to IBM',0ah,0dh,00h
logok:      db  'Login OK.',00h
```

```
logerr:     db  'Login Error. Please stripe card and type password again.',0ah,0dh,00h
msg_get_num:
            db 'Please enter host telephone number.',0dh,0ah,00h


;**********************************
;
;OFF HOOK Telephone Line and Dial
;
;**********************************

on_hook: mov DPTR, #PORTA
         movx A, @DPTR
 orl  A, #0H            ; OFF HOOK telephone line
 xrl  A, #AUD_EN        ; set speaker ON
 movx @DPTR, A

         mov dptr, #Phone_Number

dial:    movx A, @DPTR
         cjne a,#00h, cont_dial
         jmp next1

cont_dial:
            cjne a,#',', chk_asci
    mov R2, #0c0h
    lcall delay
    mov R2, #0c0h
            lcall delay
            jmp dial_loop

chk_asci:   subb a,#30h
    push dph
    push dpl
            push acc

    mov dptr, #phone_table
            movc a, @a+dptr
            lcall SET_PORTB
```

```
            mov R2,#35h
            lcall delay

    mov  a, #0ffh
    lcall SET_PORTB
    mov R2,#25
    lcall delay

            pop acc
            pop dpl
            pop dph
dial_loop:
    inc dptr
    jmp  dial


SET_PORTB:  push dph
            push dpl

            mov DPTR, #PORTB
            movx @DPTR, A

    pop  dpl
    pop  dph
    ret

phone_table:  db  0Dh,00d,01h,02h,04h,05h,06h,08h,09h,0Ah

NEXT1:      mov R2, #0ffh     ; wait for telephone line ready
    lcall delay
    mov R2, #0ffh
    lcall delay
    mov R2, #0ffh
    lcall delay
            mov   DPTR, #PORTA
            movx  A, @DPTR
            orl   A, #AUD_EN      ; connect input from line to MC145450
    movx  @DPTR, A
n?lp:       jb CAR_D, n?lp        ;wait for answer back tone
```

```
carier_back:
    movx A, @DPTR
    xrl A, #RTS_O_L       ;
    movx @DPTR, A
            mov R2, #0ffh
            lcall delay
    mov R2, 0ffh
    lcall delay

exit:

;*****************************************
;
; wait for RTS from IBM  if "ON" send CTS "ON" back
; then set speaker off and set MC145450 to receive carrier
;
;*****************************************

wait_to_send:
        lcall clrlcd
 mov DPTR,#test2
 lcall LCDLDP
 mov R2, #0ffh
 lcall delay

 mov DPTR, #PORTA
 movx A, @DPTR
wait_rts:
 jnb RTS_I_L, set_RTS   ; if RTS  "ON", jmp to set CTS
 orl A, #RTS_O_L        ; set CTS  "OFF"
 jmp wait_rts           ; wait and wait

TEST2:   DB  "Wait for RTS",0h

set_RTS:
        mov R1, A              ; Now RTS is "ON"
 mov A, #RTS_O_L
 cpl A
```

```
 anl A, R1              ; then set CTS  "ON" back
 movx @DPTR, A

 lcall clrlcd
 mov DPTR,#test3
 lcall LCDLDP
        mov R2, #0ffh
 lcall delay
        mov R2, #0ffh
        lcall delay


carrier_DE:
         mov DPTR, #PORTA
         movx A, @DPTR
         mov R1, A
         mov A, #CD_L
         cpl A
         anl A, R1              ; and set -CD to "0" for detecting carrier in

         movx @DPTR, A

;---------- Host Ready to send data ---------------->

;*****************************************************
;
; ~~~~5. send data to IBM
;  5.1 change mode


send_data:
 ;setb TR1           ;Enable Timer 2

 ;setb ET0           ; set Timer1 Int
        lcall clrlcd
        mov DPTR, #msg_txrx
        lcall LCDLDP

;test_tx:
```

```
;          mov DPTR, #PORTA
;          movx A, @DPTR
;          orl  A, #MODE
;          movx @DPTR, A          ; change mode to Transfer mode

st:      lcall rx
 lcall tx
 cjne A,#0dh,st
 mov A, #0ah
 lcall tx
 mov A, #0dh
 lcall tx
 sjmp st

test3:      db  'Send CTS',0h
msg_txrx:   db  'Transceive mode.',0h

;/*************************************************************************\
;/                                                                         \
;/                  SUBROUTINE OR MODULE AREA                              \
;/                                                                         \
;/*************************************************************************\


;*********************************************\
;                                             \
;1. serial communication subroutine           \
;                                             \
;                                             \

;***************************** SEND BYTE *********************************
;send 1 Byte Data to IBM with ascii character
;Input:  A  ->  Byte Data
;Return:   -
;Register used:  R2, , R4

SEND_BYTE:
    MOV R4, A
    LCALL HTOA
```

```
     MOV A, R2
     LCALL tx
     MOV A, R3
     LCALL tx
     MOV A, R4
     RET

;*****************************************************************
; Sub Program Send Data 1 word to RS-232                         \
;                                                                \
; REG   DPTR = table 1 word                                      \
; DESTROY      DPTR, A                                           \
;*****************************************************************\
tword:   clr A
 movc A, @A+DPTR
tw:      lcall tx                    ; send data
 inc DPTR                        ; Table = Table +1
 clr  A
 movc A, @A+DPTR
 cjne A, #00h, tw                ;If end word, end program
 ret                             ;and not send Null terminate

;--------------------------------
; Sub Program Send Data from RS-232
; A  -> Data to receive

rx:      jnb RI, $                       ;Check RI = 1
         clr RI                          ;Clear RI
 mov a, SBUF                      ;Input Data From SBUF
         ret


;--------------------------------
; Sub Program Send Data 1 Byte To RS-232
; A -> Data to Send
tx:      jnb TI, $                       ;Check TI = 1
         clr TI                          ;Clear TI
         mov SBUF, a                     ;Send Data to SBUF
 ret
```

```
;********************** End of Subroutine ******************************


;********************** LCD AND MAGNETIC READER MODULE *************
;
;


;*********************************************************************
; General purpose magnetic sampling routine
; INPUT:    -
; OUTPUT:   ACC  contains sample count
; USAGE REG.
;           DPTR: point to sampled bit storage area
;           R0:   8 bit counter
;           R1:   total bit counter
;*********************************************************************

MAG_SAMPLE:  MOV  DPTR, #S_BUF
     MOV  R1, #0              ; sample counter
L7MS1:       MOV  R0, #8              ; bit counter
L7MS2:       JB   CP, L7MS4           ; card not present, quit
             JB   CLK, L7MS2          ; waiting for zero bit
             MOV  C, DQ               ; get bit data
L7MS3:       JB   CP,L7MS4            ; card not present, quit
             JNB  CLK, L7MS3          ; waiting for one bit
             CPL  C
             RRC  A
             DJNZ R0, L7MS2
     MOVX @DPTR,A
             INC  DPTR
             INC  R1
             CJNE R1, #S_LIM, L7MS1
L7MS4:       MOV  A,R1
     RET


;*********************************************************************
; Bidirectional magnetic track 2 decode routine
```

```
; INPUT:   sample count in ACC
; OUTPUT:  ACC  contains character count
;          DPTR points to decoded data buffer
;
; USAGE REG.: for this routine (include subroutine) as follow
;          R5: Direction flag, 0 = forward
;          R4: Cumulative LRC register
;          R3: Decoded character counter
;          R2: Nondecrementing sample count
;          R1: Decrementing sample count
;          R0: bit synchronizing counter
;*************************************************************************

MAG_DECODE:  ;1st pass, setup for forward decode attempt
     MOV  R5, #0          ; indicate forward decode
          MOV  DPTR, #S_BUF    ; point to start of buffer
          MOV  R1, A           ; sample counter
     MOV  R2, A           ; sample count
          JMP  L7MD1

L7MD0:       ;2 nd pass setup for reserve decode atternpt
          MOV  R5, #1          ; indicate reverse decode
          MOV  DPTR, #S_BUF
          MOV  A, R2           ; sample count
          DEC  A
          ADD  A, DPL
          MOV  DPL, A
          CLR  A
     ADDC A, DPH
          MOV  DPH, A          ; point to end of buffer
     MOV  R1, 2           ; sample counter

L7MD1:       ;decode initialization
     MOV  R3, #0          ; character count
          MOV  R4, #0          ; initial LRC
          MOV  R0, #0          ; bit synchronizer
          CALL FIND_START      ; main decode loop
     JNC  L7MD5           ; start bit error
     CALL GET_CHAR        ; Start Sentinel
```

```
            JB   ACC.7, L7MD5      ; format error
    CJNE A, #M_SS, L7MD5   ; Start Sentinel error
L7MD2:                             ; data byte or End Sentinel
            CALL GET_CHAR
            JB   ACC.7, L7MD5      ; format error
            MOV  R6, A
            CJNE A, #M_ES, L7TEMP1 ; End Sentinel not found
            JMP L7MD4
L7TEMP1:    CALL GET_CHAR
            JB   ACC.7, L7MD5      ; format error
            CJNE A, #M_ES, L7MD3   ; End Sentinel not found
    JMP L7MD4              ; get LRC

L7MD3:      RL   A
            RL   A
            RL   A
            RL   A
            ORL  A, R6
    SWAP A
            CALL STORE_CHAR        ; data character
            JMP  L7MD2
L7MD4:                             ; LRC
            CALL GET_CHAR
            JB   ACC.7, L7MD5      ; format error
            MOV  A, R4
            JNZ  L7MD5             ; LRC error
            MOV  DPTR, #D_BUF      ; good return
    MOV  A, R3             ; final character count
            RET
L7MD5:                             ; decode error, check if 1st pass
            CJNE R5, #1, L7MD0     ; check direction
    CLR  A                 ; bad return
    RET

; Get the next bit from the sample buffer
; Output:  C contain data bit
GET_BIT:
    CJNE  R0, #0, L7GB1     ; bit synchronizer
    MOV   R0, #8
```

```
            PUSH  ACC
            MOVX  A, @DPTR
    CALL INDEX_PTR
            MOV  B,A
            POP  ACC
            DEC  R1                 ; sample counter .
L7GB1:      XCH  A, B
    CALL POSITION_BIT
            XCH  A, B
            DEC  R0                 ; bit synchronizer
            RET


; Find the first '1' bit in the sample buffer
; Output: C=1 if bit is found
FIND_START:
L7FS1:
     CJNE  R0, #0, L7FS2          ; check bit synchronizer
             MOV   R0, #8
     MOVX  A, @DPTR
             CALL  INDEX_PTR
             DJNZ  R1, L7FS2              ; sample counter
     JMP   L7FS4                  ; out of samples
L7FS2:       CALL  LOCATE_BIT             ; test for a '1' bit
             JC    L7FS3
             CALL  POSITION_BIT
             DEC   R0                     ; bit synchronizer
             JMP   L7FS1
L7FS3:                                    ; good return.
             MOV   B, A                   ; save copy in B
             SETB  C
     RET
L7FS4:       CLR   C
             RET

; Get the next 5 bit(4 data + 1 odd parity) element from sample buffer
; Output: ACC contain data element
;         ACC.7: error flag
GET_CHAR:
```

```
    MOV  A, R1                  ; sample counter
         JZ   L7GC2                     ; out of sample
         MOV  R7, #5                    ; bit counter
    CLR  A
L7GC1:   CALL GET_BIT                   ; next bit
         RRC  A
         DJNZ R7, L7GC1
         RR   A
         RR   A
         RR   A
         JNB  P, L7GC2                  ; odd parity error
    ANL  A, #0FH                ; discard parity
         PUSH ACC
    XRL  A, R4                  ; calculate LRC
         MOV  R4, A
         POP  ACC                       ; good return
    RET
L7GC2:   SETB ACC.7                     ; bad return
         RET

; Translate and store the data character
; Input: ACC contain data character
STORE_CHAR:
    PUSH  DPL
          PUSH  DPH
          PUSH  ACC
          MOV   DPTR, #D_BUF
          MOV   A, R3               ; character counter
          ADD   A, DPL
          MOV   DPL, A
    CLR   A
          ADDC  A, DPH              ; generate displacement
          MOV   DPH, A
    POP   ACC
          MOVX  @DPTR, A            ; store
    POP   DPH
          POP   DPL
          INC   R3                  ; character counter
          RET
```

```
; Index the sample pointer either forward or backword
INDEX_PTR:
            CJNE  R5, #0, L7IP1           ; check direction
     INC   DPTR                    ; forward
            RET
L7IP1:      PUSH  ACC                     ; backward
            DEC   DPL
            CJNE  A, #-1, L7IP2
            DEC   DPH
L7IP2:      POP   ACC
            RET

; Position bit is in ACC into C in either a right or left shift
POSITION_BIT:
     CJNE  R5, #0, L7PB1           ; check direction
            RRC   A                       ; forward
            RET
L7PB1:      RLC   A
            RET

; Locate a bit, either msb or lsb; output: C=1 if bit is a one
LOCATE_BIT:
            CJNE  R5, #0, L7LB1           ; check direction
     MOV   C, ACC.0                ; forward
            RET
L7LB1:      MOV   C, ACC.7                ; backward
            RET

;******************  END OF MAGNETIC ROUTINE  *************************

;*********************************************************************
;
; ### LCD display subroutine
;    show you varities of display
;
;*********************************************************************

;******************INIT & CLEAR LCD*********************
```

```
INITLCD:   MOV A, #00111000B          ; Function set
   LCALL LCDWI
           MOV A, #00001110B          ; Display ON/OFF
           LCALL LCDWI
CLRLCD:    MOV A, #01H                ; Clear
           LCALL LCDWI
           RET

;*************LCDWI SUB*********************
;LCD WRITE INSTRUCTION (RS=0)
;Input:  A

LCDWI:     PUSH   DPH
  PUSH   DPL
  MOV    DPTR, #LCDWRC
           MOVX   @DPTR, A
           MOV    DPTR, #LCDRDC
LCDWI1:    MOVX   A, @DPTR                    ; Wait for BF=0
           JB     ACC.7, LCDWI1
           POP    DPL
           POP    DPH
  RET


;*************LCDWD SUB*********************
;LDC WRITE DATA (RS=1)
;Input A

LCDWD:     PUSH  DPH
           PUSH  DPL
           MOV   DPTR, #LCDWRD
           MOVX  @DPTR, A
  MOV   DPTR, #LCDRDC
LCDWD1:    MOVX  A, @DPTR                 ;Wait for BF=0
  JB    ACC.7, LCDWD1
           POP   DPL
           POP   DPH
           RET
```

```
;************ LCDLDP SUB*********************
; LOAD PMEM TO LCD-MODULE
; Input :  DPTR START BLOCK
;

LCDLDP:    MOV A, #80H                ; Set address Line 1
   LCALL LCDLDPS
   MOV A, #0C0H                 ; Set Address Line 2
   LCALL LCDLDPS
   RET

LCDLDPS:   LCALL LCDWI                ; Load one line
   MOV R2, #20                  ; 20 characters
LCDLDPS1:  CLR A
   MOVC A, @A+DPTR              ; PMEM = MOVC
   CJNE A, #0h, LCDP_CONT
          SJMP LCDP_EXIT
LCDP_CONT:
          LCALL LCDWD                ; Write data
   INC DPTR
          DJNZ R2, LCDLDPS1
LCDP_EXIT:
   RET

;*******************LCDLDD SUB***************
; LOAD DMEM TO LCD-MODULE
; Input:  DPTR  START BLOCK

LCDLDD:    MOV A, #80H                 ; Set address Line 1
          LCALL LCDLDDS
          MOV A, #0C0H                ; Set Address Line 2
   LCALL LCDLDDS
          RET

LCDLDDS:   LCALL LCDWI                ; Load one line
   MOV R2, #20                  ; 20 characters
LCDLDDS1:  MOVX A, @DPTR                ; PMEM = MOVC
          CJNE A, #0h, LCDD_CONT
```

```
          SJMP LCDD_EXIT
LCDD_CONT:
          LCALL LCDWD                ; Write data
          INC DPTR
   DJNZ R2, LCDLDDS1
LCDD_EXIT:
          RET

;****************  DELAY SUB  ******************
; Input:  R2

DELAY:   MOV R3, #80H
DELAY1:  MOV A, #8
DELAY2:  DEC A
         JNZ DELAY2
         DJNZ R3, DELAY1
         DJNZ R2, DELAY
 RET

;****************** HEX TO ASCII ******************
; Input:  A  -> hex number
; Output:   R2  -> High byte
;           R3  -> Low byte

HTOA:     PUSH ACC
          SWAP A
  LCALL HTOAS
          MOV  R2, A
          POP ACC
          LCALL HTOAS
          MOV R3, A
  RET
HTOAS:    ANL A, #0FH
          CJNE A, #0AH, $+3
          JNC HTOAS1
          ORL A, #30H
          RET
HTOAS1:   SUBB  A, #9
          ORL  A, #40H
```

```
        RET

;************************** BYTE TO LCD ********************************
;Print  1 Byte Data to LCD
;Input:  A  ->  Byte Data
;Return:   -
;Register used:  R2, , R4

BYTE_TO_LCD:
            MOV R4, A
            LCALL HTOA
    MOV A, R2
            LCALL LCDWD
            MOV A, R3
            LCALL LCDWD
            MOV A, R4
            RET

;******************  END OF LCD SUBROUTINE *************************


        end
```

```
;****************************** HOST.ASM ******************************
;
;
;--------------------------------
;Port Configuration for  Host MLS
;--------------------------------
;*****  8031 Port Configuration *****
;
;    P1.0     -DTE         0 for IBM ready              1  for not ready
;    P1.1     -RTS IN      0 for IBM request to send    1  for not request
;    P1.2     -RI          0 for Ring Input             1  for no ring input
;    P1.3     -Sd DATA          send data if Mode is in transmit mode
;    P1.4     AnBk         0 for not Answer back        1  for Answer back
;    P1.5     -DCD     0 for carrier detect         1  for no carrier detect
;    P1.6      -
;    P1.7      -
;
;******* 8255 Port Configuration *******
;
;    PA.0     Mode         0 for initial mode         1  for transmit mode
;    PA.1     OH           0 for ONHOOK              *1  for  OFFHOOK
;    PA.2     Audio En     0 for contact to speaker   1  for contact to MC145450
;    PA.3     -CD         *0 for detect carrier       1  for not detect carrier
;    PA.4     -RTS out    *0 for set MC145450 ready   1  for not in tranmit
;    PA.5     -DSR        *0 for tell IBM that Data Set Ready
;    PA.6       -
;    PA.7      -
;-------------------------------------------------
PORTBASE   EQU   0H
PORTA      EQU   PORTBASE + 0
PORTB      EQU   PORTBASE + 1
PORTC      EQU   PORTBASE + 2
CONP       EQU   PORTBASE + 3
CONW       EQU   80h  ; 1 00 00 0 00


MODE       EQU   00000001b
OH         EQU   00000010b
AUD_EN     EQU   00000100b
```

```
CD_L        EQU   00001000b
RTS_O_L     EQU   00010000b
DSR_L       EQU   00100000b

DTE_L       EQU   P1.0
RTS_I_L     EQU   P1.1
RI_L        EQU   P1.2
Sd_D        EQU   P1.3
AnBk        EQU   P1.4
DCD_L       EQU   P1.5


P_BUF       EQU   30h


            org  0h
start:      mov r2, #0ffh
            lcall delay
            mov r2, #0ffh
            lcall delay
            mov r2, #0ffh
            lcall delay
            jmp init_port

            org 3h          ; IE0
            reti

            org 0Bh         ; TF0
            reti

            org 013h        ;IE1
            reti

            org 01Bh        ;TF1

            reti

            org 023h        ;RI&TI
            reti
```

```
;************************************************************
;
; 1. Initial All output and input port in 8255 PORT A and 8031
;


        org 30h
init_port:
        mov A, #11101111b    ; -DTE     input port   latch "1"
                             ; -RTS IN  "--------------------"
                             ; -RI      "--------------------"
                             ; Sd Data  1 for mark state
                             ; AnBk     0  for not answer
                             ; P1.5 - P1.7  avaiable
        mov P1, A
        mov DPTR, #CONP
        mov A, #CONW
        movx @DPTR, A        ; Set All port to Output

        mov DPTR, #PORTA
        mov A, #00111000b    ;initial state, wait for ring signal
                             ; Mode        0  Initial
                             ; OH          0  ON-HOOK
                             ; Audio En    0  connect line to Speaker
                             ; -CD         1  not detect carrier
                             ; -RTS Out    1  not in transmit
                             ; -DSR        1  Modem not ready
         movx @DPTR, A

init_serial:
        mov  SCON, #01010010b   ;Set SCON (Serial Port Control Register)
                                ;Mode 1
                                ;REN  =  1 (Enable)
                                ;Ti   =  1
        mov  TMOD, #00100010b   ;Set TMOD (Timer)
                                ;Mode 2  (Auto Reload 8 bit)
        mov  TH1, #11101000b    ; Set Baud RATE = 1200
           ;mov  TH1,  #01000000b  ;
```

```
            ;mov  TH1, #11111101b   ;                9600
          mov  TH2, #01000000b    ; set Timer 2 to Baud Rate 150


;------------------   Initialize Complete ------------------>



;*************** Check  RS232 Condition ************************
;
; 2. Wait Until  DTE from IBM is "ON"   and then send DSR  "ON"
;
;*********************************************************************

rs232_check:
        mov DPTR, #PORTA
        movx A, @DPTR

wait_DTE:
        jnb DTE_L, set_DSR      ; if DTE "ON" jmp to set DSR
        orl A, #DSR_L           ; if not , set DSR  "OFF"
        jmp wait_DTE            ; wait until DTE is "ON"

set_DSR: mov R1, A              ; OK, DTE is "ON"
        mov A, #DSR_L
        cpl A
        anl A, R1               ; set DSR  "ON" respond to DTE

        movx @DPTR, A

;---------------------  IBM ready ---------------------------->
;*********************************************************************
; 3. wait for ring signal to begin linking
;
;*********************************************************************

wait_ring:
        jnb p1.2, ring_in     ; wait for Ring signal
        jmp wait_ring         ; wait and wait
```

```
ring_in:
        movx A, @DPTR          ; Hey, ring come in
        orl  A, #0H            ; OFF-HOOK telephone quickly
        movx @DPTR, A
        setb AnBk              ; Anwer to Terminal that here is HOST

wait_car:
        jb DCD_L, wait_car     ; Terminal must send Carrier back
                               ; so, wait until DCD is "ON"
        clr AnBk               ; if DCD "ON", release AnBk


;------------ Now,  Connection is completed ----------------->
;------------ and it is ready to send data from ------------->
;------------ Host to Terminal          --------------------->


;*********************************************************************
;
;4. wait for RTS from IBM  if "ON" send CTS "ON" back
;   then set speaker off and set MC145450 to receive carrier
;
;*********************************************************************

routine_4:


        mov DPTR, #PORTA
        movx A, @DPTR
wait_rts:
        jnb RTS_I_L, set_RTS   ; if RTS  "ON", jmp to set CTS
        orl A, #RTS_O_L        ; set CTS  "OFF"
        jmp wait_rts           ; wait and wait
set_RTS: mov R1, A             ; Now RTS is "ON"
        mov A, #RTS_O_L
        cpl A
        anl A, R1              ; then set CTS  "ON" back
        movx @DPTR, A
```

```
carrier_DE:
        mov DPTR, #PORTA
        movx A, @DPTR
        orl  A, #AUD_EN         ; set line to MC145450 and OFF speaker
        mov R1, A
        mov A, #CD_L
        cpl A
        anl A, R1               ; and set -CD to "0" for detecting carrier in

        movx @DPTR, A

;---------- Host Ready to send data ---------------->

;*****************************************************
;
; 5. send data to IBM
;  5.1 change mode


send_data:
        setb TR1            ;Enable Timer 2

        setb ET0            ; set Timer1 Int
        mov DPTR, #table
        lcall tword
        mov DPTR, #table1
        lcall tword

;test_tx:
;         mov DPTR, #PORTA
;         movx A, @DPTR
;         orl  A, #MODE
;         movx @DPTR, A          ; change mode to Transfer mode

st:     lcall rx
        lcall tx
        cjne A,#0dh,st
        mov A, #0ah
```

```
        lcall tx
        mov A, #0dh
        lcall tx
        sjmp st

table:  db  '   Modem Ready',0ah
table1: db  'Transfer All Control to IBM', 0ah


;*********************************************************************
; Sub Program Send Data 1 word to RS-232
;
; REG   DPTR = table 1 word
; DESTROY      DPTR, A
;*****************************************

tword:  clr A
        movc A, @A+DPTR
tw:     lcall tx                    ; send data
        inc DPTR                    ; Table = Table +1
        clr  A
        movc A, @A+DPTR
        cjne A, #0ah, tw            ;If end word, end program
        mov A, #0ah                 ;LineFeed
        lcall tx
        mov A,#0dh                  ;Newline
        lcall tx
        ret

;--------------------------------
; Sub Program Send Data from RS-232
; A  -> Data to receive

rx:     jnb RI, $                   ;Check RI = 1
        clr RI                      ;Clear RI
        mov a, SBUF                 ;Input Data From SBUF
        ret
```

```
;--------------------------------
; Sub Program Send Data 1 Byte To RS-232
; A -> Data to Send
tx:     jnb TI, $                   ;Check TI = 1
        clr TI                      ;Clear TI
        mov SBUF, a                 ;Send Data to SBUF
        ret


;****************  DELAY SUB  ******************
; Input:  R2

DELAY:   MOV R3, #80H
DELAY1:  MOV A, #8
DELAY2:  DEC A
         JNZ DELAY2
         DJNZ R3, DELAY1
         DJNZ R2, DELAY
         RET

;********************* End of Subroutine ****************************



         end
```

ภาคผนวก ง.

โปรแกรมภาษาซีสำหรับ PC

```
/**********  ASCII.H  **********/

#define SOH 0x01
#define STX 0x02
#define EOT 0x04
#define ACK 0x06
#define BS  0x08
#define LF  0x0a
#define CR  0x0d
#define XON 0x11
#define XOFF 0x13
#define NAK 0x15
#define CAN 0x18
#define CTLZ 0x1a
#define ESC 0x1b
#define DEL 0x7f
```

```
/********** PCL4C.H **********/

#define PROTOTYPES 1

#if PROTOTYPES

/* External Library Function Prototypes */

extern int SioBaud(int, int);
extern int SioBrkKey();
extern int SioBrkSig(int, char);
extern int SioCrtWrite(char);
extern int SioCTS(int);
extern int SioDCD(int);
extern int SioDSR(int);
extern int SioDTR(int, char);
extern int SioDelay(int);
extern int SioDone(int);
extern int SioError(int);
extern int SioFIFO(int, int);
extern int SioFlow(int, int);
extern int SioGetc(int, int);
extern int SioInfo(char);
extern int SioIRQ(int, int);
extern int SioKeyPress();
extern int SioKeyRead();
extern int SioLine(int);
extern int SioLoopBack(int);
extern int SioModem(int, char);
extern int SioParms(int, int, int, int);
extern int SioPutc(int, char);
extern int SioRead(int, int);
extern int SioReset(int, int);
extern int SioRI(int);
extern int SioRTS(int, char);
extern int SioRxBuf(int, char *, int);
extern int SioRxFlush(int);
extern int SioRxQue(int);
extern long SioTimer();
```

```
extern int SioUART(int, int);
extern int SioUnGetc(int, char);

#else

/* External Library Functions */

extern int SioBaud();
extern int SioBrkKey();
extern int SioBrkSig();
extern int SioCrtWrite();
extern int SioCTS();
extern int SioDCD();
extern int SioDSR();
extern int SioDTR();
extern int SioDelay();
extern int SioDone();
extern int SioError();
extern int SioFIFO();
extern int SioFlow();
extern int SioGetc();
extern int SioInfo();
extern int SioIRQ();
extern int SioKeyPress();
extern int SioKeyRead();
extern int SioLine();
extern int SioLoopBack();
extern int SioModem();
extern int SioParms();
extern int SioPutc();
extern int SioRead();
extern int SioReset();
extern int SioRI();
extern int SioRTS();
extern int SioRxBuf();
extern int SioRxFlush();
extern int SioRxQue();
extern long SioTimer();
extern int SioUART();
```

```
extern int SioUnGetc();

#endif

/* Port Codes */

#define COM1 0
#define COM2 1
#define COM3 2
#define COM4 3

/* Baud Rate Codes */

#define Baud300 0
#define Baud600 1
#define Baud1200 2
#define Baud2400 3
#define Baud4800 4
#define Baud9600 5
#define Baud19200 6
#define Baud38400 7
#define Baud57600 8
#define Baud115200 9

#define NORESET -1

/* Parity Codes */

#define NoParity 0
#define OddParity  1
#define EvenParity 3
#define MarkParity 5
#define SpaceParity 7

/* Stop Bit Codes */

#define OneStopBit  0
#define TwoStopBits 1
```

```
/* Word Length Codes */

#define WordLength5  0
#define WordLength6  1
#define WordLength7  2
#define WordLength8  3

/* Buffer Size Codes */

#define Size8     0
#define Size16    1
#define Size32    2
#define Size64    3
#define Size128   4
#define Size256   5
#define Size512   6
#define Size1024  7
#define Size2048  8
#define Size4096  9
#define Size8192  10
#define Size16384 11
#define Size32768 12
#define Size1K    7
#define Size2K    8
#define Size4K    9
#define Size8K    10
#define Size16K   11
#define Size32K   12

/* Line Status Masks */

#define TransBufferEmpty 0x20
#define BreakDetect  0x10
#define FramingError 0x08
#define ParityError  0x04
#define OverrunError 0x02
#define DataReady    0x01

/* Modem Status Masks */
```

```
#define DCD 0x80
#define RI  0x40
#define DSR 0x20
#define CTS 0x10
#define DeltaDCD 0x08
#define DeltaRI  0x04
#define DeltaDSR 0x02
#define DeltaCTS 0x01

/* Break Signal Commands */

#define ASSERT 'A'
#define CANCEL 'C'
#define DETECT 'D'

/* SioDTR & SioRTS Commands */

#define SET   'S'
#define CLEAR 'C'
#define READ  'R'

/* FIFO level codes */
#define FIFO_OFF -1
#define LEVEL_1   0
#define LEVEL_4   1
#define LEVEL_8   2
#define LEVEL_14  3

/* Primary / Secondary IRQ codes */
#define PRIMARY   0
#define SECONDARY 1

/* SioInfo Commands */

#define VERSION 'V'
#define M_MODEL 'M'
```

```
/********** TERM.H **********/

/*
**  1) Set AT_COMMAND_SET to 1 if you are using a HAYES AT command set
**     compatible modem, else set to 0.
**  2) Set RTS_CTS_CONTROL to 1 if talking to a modem that requires flow
**     control.
**  3) If you are using a null modem cable for a direct PC to PC link, don't
**     set RTS_CTS_CONTROL to 1 unless you are absolutely sure that RTS and
**     CTS are switched in the null modem cable.
**  4) Don't reduce any of the timing constants.
*/

#define AT_COMMAND_SET  0
#define RTS_CTS_CONTROL 0

#define ONE_SECOND 18
#define SHORT_WAIT 3
#define LONG_WAIT 10
```

```
/********** CRC.C **********/

#include <stdio.h>
#define POLY 0x1021

unsigned short CRCtable[256];

/* initialize CRC table */

void InitCRC()
{
    int i;
    unsigned short CalcTable();

    for (i=0; i<256; i++) CRCtable[i] = CalcTable(i,POLY,0);
}

/* calculate CRC table entry */

unsigned short CalcTable(data,genpoly,accum)
unsigned short data;
unsigned short genpoly;
unsigned short accum;

{
    static int i;

    data <<= 8;
    for (i=8; i>0; i--)
    {
       if ((data^accum) & 0x8000)  accum = (accum << 1) ^ genpoly;
       else accum <<= 1;
       data <<= 1;
    }
    return(accum);
}

/* compute updated CRC */
```

```
unsigned short UpdateCRC(crc,byte)
unsigned short crc;
unsigned char  byte;
{
    return( (crc << 8) ^ CRCtable[ (crc >> 8) ^ byte ] );
}
```

```
/********** DIR.C **********/

#include <stdio.h>
#include <dos.h>

#define FALSE 0
#define TRUE !FALSE
#define WORD unsigned int


void setDTA(BufPtr)
WORD BufPtr;
{
    union REGS reg;
    reg.x.dx = (WORD) BufPtr;
    reg.h.ah = 0x1A;
    int86(0x21, &reg, &reg);
}

int FindFirst(FilePtr)
int FilePtr; /* file spec */
{
    union REGS reg;
    reg.x.dx = FilePtr;
    reg.h.ah = 0x4e;
    reg.x.cx = 0;
    int86(0x21, &reg, &reg);
    if (reg.x.cflag) return(FALSE);
    else return(TRUE);
}

int FindNext()
{
    union REGS reg;
    reg.h.ah = 0x4f;
    int86(0x21, &reg, &reg);
    if (reg.x.cflag) return(FALSE);
    else return(TRUE);
}
```

```
int ChangeDir(DirPtr)
int DirPtr;   /* directory */
{
    union REGS reg;
    reg.h.ah = 0x3b;
    reg.x.dx = DirPtr;
    int86(0x21, &reg, &reg);
    if (reg.x.cflag) return(FALSE);
    else return(TRUE);
}

int CurrentDir(DirPtr)
int DirPtr;   /* directory */
{
    union REGS reg;
    reg.h.ah = 0x47;
    reg.x.si = DirPtr;
    reg.h.dl = 0;
    int86(0x21, &reg, &reg);
    if (reg.x.cflag) return(FALSE);
    else return(TRUE);
}
```

```
/********** DOS.C **********/

#define BYTE unsigned char

#include <stdio.h>
#include <dos.h>

static BYTE cur_row  = 0;      /* current row */
static BYTE cur_col  = 0;      /* current col */

/* write character & attribute without advancing cursor */
void AttrWrite(ch,attr)
BYTE ch;
BYTE attr;
{
    union REGS  reg;
    reg.h.ah = 9;
    reg.h.bh = 0;        /* current text page */
    reg.h.al = ch;       /* character */
    reg.h.bl = attr;
    reg.x.cx = 1;
    int86(0x10, &reg, &reg);
}

/* position cursor at desired row & column */
void Position(row, col)
BYTE row, col;
{
    union REGS reg;
    reg.h.ah = 2;
    reg.h.bh = 0;
    reg.h.dh = row;
    reg.h.dl = col;
    int86(0x10, &reg, &reg);
    cur_row = row;
    cur_col = col;
}
```

```
/* returns the current row of the cursor */
int GetRow()
{
    union  REGS  reg;
    reg.h.ah = 3;
    reg.h.bh = 0;
    int86(0x10, &reg, &reg);
    return(reg.h.dh);
}

/* returns the current column of the cursor */
int GetCol()
{
    union  REGS  reg;
    reg.h.ah = 3;
    reg.h.bh = 0;
    int86(0x10, &reg, &reg);
    return(reg.h.dl);
}

/* scrolls area specified # rows */
void Scroll(urow, lcol, lrow, rcol, nrows, attr)
unsigned  urow;   /* upper row of area */
unsigned  lcol;   /* left column of area */
unsigned  lrow;   /* lower row of area */
unsigned  rcol;   /* right column of area */
int nrows;        /* # rows to scroll */
int attr;         /* attribute to use for blank lines */
{
    union REGS reg;
    reg.h.ah = 6;
    reg.h.ch = urow;
    reg.h.cl = lcol;
    reg.h.al = nrows;
    reg.h.bh = attr;
    reg.h.dh = lrow;
    reg.h.dl = rcol;
    int86(0x10, &reg, &reg);
}
```

```
/********** MODEM_IO.C **********/

#include <stdio.h>
#include "pcl4c.h"
#include "ascii.h"

#define FALSE 0
#define TRUE !FALSE
#define SECONDS 18

/*** send string to modem & get echo ***/

int SendTo(Port,String)
int Port;       /* port to talk to */
char *String;   /* string to send to mdoem */
{
    int i;
    char c;
    int Code;

    SioRxFlush(Port);
    SioDelay(SECONDS/4);
    for (i=0; i<strlen(String); i++)
    {
        /* User BREAK ? */
        if (BreakTest()) return(FALSE);
        /* fetch character */
        c = toupper( String[i] );
        switch(c)
        {
            case '!':
                /* replace ! with carriage return */
                c = CR;
                break;
            case '~':
                /* delay 1/2 second */
                SioDelay(SECONDS/2);
                c = ' ';
                break;
```

```
            case ' ':
                /* delay 1/4 second */
                SioDelay(SECONDS/4);
                break;
        } /* end switch */
        /* transmit as 7 bit character */
        PutChar(Port,(char)(0x7f & c));
        /* delay 3/18th of a second */
        SioDelay(3);
        /* wait up to 1 second for the echo */
        Code = GetChar(Port,SECONDS);
        if (Code>0) SioCrtWrite((char)Code);
    }
    return(TRUE);
} /* end SendTo */

/*** wait for incoming string from modem ***/

int WaitFor(Port,String)
int Port;       /* Port to talk to */
char *String;   /* string to wait for */
{
    int i, k;
    char c;
    int Code;

    SioCrtWrite(LF);
    /* flush leading LFs and CRs */
    while (1)
    {
        /* User BREAK ? */
        if (BreakTest()) return(FALSE);
        /* get next incoming character */
        Code = GetChar(Port,2*SECONDS);
        /* printf("{\n%x}",Code); */
        if (Code==-1) break;
        /* skip any leading CR or LF */
        if (((char)Code!=LF)&&((char)Code!=CR))
        {
```

```
            /* stuff character back & break out of loop */
            SioUnGetc(Port,(char)Code);
            break;
        }
        SioDelay(SECONDS/5);
    }
    /* wait for string */
    for (i=0; i<strlen(String); i++)
    {
        /* User BREAK ? */
        if (BreakTest()) return(FALSE);
        c = String[i];
        /* wait up to 1 second for next character */
        Code = GetChar(Port,SECONDS);
        /* printf("(\n%x)",Code);*/
        if (Code==-1) return(FALSE);
        SioCrtWrite((char)Code);
        if ((char)Code != c)
        {
            printf("\nExpecting %xH not %xH\n",c,Code);
            return(FALSE);
        }
    }
    /* another character ? */
    Code = GetChar(Port,SECONDS);
    if (Code>0) SioCrtWrite((char)Code);
    return(TRUE);
} /* end WaitFor */

int BreakTest()
{
    /* User BREAK ? */
    if(SioBrkKey())
    {
        printf("User BREAK\n");
        return(TRUE);
    }
    else return(FALSE);
}
```

```
/********** TERM_IO.C **********/

#include <stdio.h>
#include "pcl4c.h"
#include "term.h"
#include "ascii.h"

/*** Display status line ***/

#define INVERSE 0x70
#define RIGHTMOST 40

void DisplayLine(MsgPtr,UserPtr,UserLength)
char *MsgPtr;
char *UserPtr;
int UserLength;
{
    static int right = RIGHTMOST;
    int row;
    int col;
    int i;
    char c;
    int MsgLength;

    row = GetRow();
    col = GetCol();
    Position(24,0);
    /* display message */
    MsgLength = strlen(MsgPtr);
    for (i=0; i<MsgLength; i++)
    {
        AttrWrite(*MsgPtr++,INVERSE);
        Position(24,i+1);
    }
    /* blank rest of menu bar */
    for (i=MsgLength; i<right; i++)
    {
        AttrWrite(' ',INVERSE);
        Position(24,i+1);
```

```
}
right = MsgLength;
/* want response from user ? */
if (UserPtr!=NULL)
{
    right = RIGHTMOST;
    /* input text from user */
    i = 0;
    while (1)
    {
        Position(24,MsgLength+i);
        c = SioKeyRead();
        if ((c==ESC)||(c==CAN))
        {
            UserPtr[0] = '\0';
            break;
        }
        if ((c=='\r')||(MsgLength+i==RIGHTMOST))
        {
            UserPtr[i] = '\0';
            break;
        }
        if ((c==BS)&&(i>0))
        {
            i--;
            Position(24,MsgLength+i);
            AttrWrite(' ',INVERSE);
        }
        else
        {
            /* save character & display on status line */
            UserPtr[i++] = c;
            AttrWrite(c,INVERSE);
            if(i==UserLength)
            {
                /* user string done */
                UserPtr[i] = '\0';
                break;
            }
```

```
            }
        } /* end -- while */
    } /* end -- if */
    Position(row,col);
}

/*** output character to serial port ***/

int PutChar(Port,ch)
int Port;
char ch;
{
    int rc;

    /* transmit character */
    rc = SioPutc(Port,ch);
    if (rc<0)
    {
        printf("SioPutc Error: COM%d: ",1+Port);
        SioError(rc);
        SioDone(Port);
        exit(1);
    }
    return(rc);
}

/*** receive character from serial port ***/

int GetChar(Port,Timeout)
int Port;
int Timeout;
{
    int rc;

    rc = SioGetc(Port,Timeout);
    if (rc<-1)
    {
        printf("SioGetc Error: COM%d: ",1+Port);
        SioError(rc);
```

```
        exit(1);
    }
    return(rc);
}

/*** display the error text ***/

void SayError(Port,ptr)
int Port;
char *ptr;
{
    char temp[81];

    sprintf(temp,"ERROR! COM%d : %s",1+Port,ptr);
    DisplayLine(temp,NULL,0);
    printf("\n*** %s\n",temp);
    /* cancel remote */
    PutChar(Port,CAN);
    PutChar(Port,CAN);
    PutChar(Port,CAN);
} /* end SayError */
```

```
/********** XYMODEM.C **********/

#include <stdio.h>
#include <fcntl.h>
#include <sys\types.h>
#include <sys\stat.h>

#include "pc14c.h"
#include "ascii.h"

#define FALSE 0
#define TRUE !FALSE

extern int GetChar();
extern int PutChar();

int TxyModem(Port,Filename,Buffer,OneKflag,BatchFlag)
int Port;           /* COM port [0..3] */
char Filename[];    /* filename buffer */
char Buffer[];      /* 1024 byte data buffer */
int OneKflag;       /* if TRUE, use 1K blocks when possible */
int BatchFlag;      /* if TRUE, send filename in packet 0 */
{
    int i, k;
    int Code;
    int Handle;         /* file Handle */
    char c;
    int p;
    char PacketType;
    char PacketNbr;
    int  PacketSize;
    int  FirstPacket;
    unsigned short CheckSum;
    long filelength();
    int Number1K = 0;       /* total # 1K packets */
    int Number128 = 0;      /* total # 128 byte packets */
    char NCGchar = NAK;
    long FileSize;
    char temp[81];
```

```
int EmptyFlag = FALSE;

/* begin */
if (BatchFlag) if (Filename[0]=='\0') EmptyFlag = TRUE;
if (!EmptyFlag)
{
    /* Filename is not empty */
    EmptyFlag = FALSE;
    Handle = open(Filename,O_RDONLY|O_BINARY,S_IREAD);
    if (Handle<0)
    {
        strcpy(temp,"Cannot open ");
        strcat(temp,Filename);
        DisplayLine(temp,NULL,0);
        return(FALSE);
    }
}
DisplayLine("XYMODEM send: waiting for Receiver ",NULL,0);
while (SioKeyPress()) SioKeyRead();
/* compute # blocks */
if (!EmptyFlag)
{
    FileSize = filelength(Handle);
    if (OneKflag) Number1K = (int) (FileSize / 1024L);
    Number128 = 1 + (int) ((FileSize -1024L*(long)Number1K -1L) / 128L);
    sprintf(temp,"%d 1024 & %d 128 byte packets",Number1K,Number128);
    DisplayLine(temp,NULL,0);
}
else
{
    /* empty file */
    Number128 = 0;
    Number1K = 0;
    /*DisplayLine("Empty File",NULL,0);*/
}
/* clear comm port ( there may be several NAKs queued up ) */
SioRxFlush(Port);
/* get receivers start up NAK, 'C', or 'G' */
if (!TxStartup(Port,&NCGchar)) return(FALSE);
```

```
/* loop over all packets */
if (BatchFlag) FirstPacket = 0;
else FirstPacket = 1;
for (p=FirstPacket; p<=Number1K+Number128; p++)
{
    /* user aborts ? */
    if (SioKeyPress()) if((char)SioKeyRead()==CAN)
    {
        TxCAN(Port);
        DisplayLine("*** Canceled by USER ***",NULL,0);
        return(FALSE);
    }
    /* issue message */
    sprintf(temp,"Packet %d",p);
    DisplayLine(temp,NULL,0);
    /* load up Buffer */
    if (p==0)
    {
        /* Filename packet ! */
        PacketSize = 128;
        k = 0;
        for (i=0; i<strlen(Filename); i++) Buffer[k++] = Filename[i];
        Buffer[k++] = '\0';
        sprintf(temp,"%ld",FileSize);
        for (i=0; i<strlen(temp); i++) Buffer[k++] = temp[i];
        while (k<128) Buffer[k++] = '\0';
    }
    else /* p > 0 */
    {
        /* DATA Packet: use 1K or 128 byte block ? */
        if (p<=Number1K) PacketSize = 1024;
        else PacketSize = 128;
        /* read next block from disk */
        Code = read(Handle,Buffer,PacketSize);
        if (Code<=0)
        {
            SayError(Port,"Error on disk read");
            return(FALSE);
        }
```

```
            for (i=Code; i<PacketSize; i++) Buffer[i] = 0x1a;
        }
        /* send this packet */
        if (!TxPacket(Port,p,PacketSize,Buffer,NCGchar)) return(FALSE);
        SioDelay(5);
        /* must 'restart' after non null packet 0 */
        if (!EmptyFlag&&(p==0)) TxStartup(Port,&NCGchar);
    } /* end -- for(p) */
    /* done if empty packet 0 */
    if (EmptyFlag)
    {
        DisplayLine("Batch transfer complete",NULL,0);
        return(TRUE);
    }
    /* all done. send EOT up to 10 times */
    close(Handle);
    if (!TxEOT(Port))
    {
        SayError(Port,"EOT not acknowledged");
        return(FALSE);
    }
    DisplayLine("Transfer Complete",NULL,0);
    return(TRUE);
} /* end -- TxyModem */

int RxyModem(Port,Filename,Buffer,NCGchar,BatchFlag)
int Port;           /* COM port [0..3] */
char Filename[];    /* filename buffer */
char Buffer[];      /* 1024 byte data buffer */
char NCGchar;        /* NAK, 'C', or 'G' */
int BatchFlag;      /* if TRUE, get filename from packet 0 */
{
    int i;
    int Handle;         /* file Handle */
    int p;              /* packet index */
    int Code;           /* return code */
    int FirstPacket;
    char PacketNbr;
    int PacketSize;          /* 128 or 1024 */
```

```
long atol();
long FileSize;
char temp[81];
int  EOTflag = FALSE;

/* begin */
EOTflag = FALSE;
DisplayLine("XYMODEM Receive: Waiting for Sender ",NULL,0);
while (SioKeyPress()) SioKeyRead();
/* clear comm port */
SioRxFlush(Port);
/* Send NAKs, 'C's, or 'G's */
if (!RxStartup(Port,&NCGchar)) return(FALSE);
/* open file unless BatchFlag is on */
if (BatchFlag) FirstPacket = 0;
else
{
    /* start with packet 1 */
    FirstPacket = 1;
    /* open file passed in Filename[] for write */
    Handle = open(Filename,O_CREAT|O_TRUNC|O_WRONLY|O_BINARY,S_IWRITE);
    if (Handle<0)
    {
        strcpy(temp,"Cannot open ");
        strcat(temp,Filename);
        DisplayLine(temp,NULL,0);
        return(FALSE);
    }
}
/* get each packet in turn */
for (p=FirstPacket; ;p++)
{
    /* user aborts ? */
    if (SioKeyPress()) if ((char)SioKeyRead()==CAN)
    {
        TxCAN(Port);
        return(FALSE);
    }
    /* issue message */
```

```
sprintf(temp,"Packet %d",p);
DisplayLine(temp,NULL,0);
/* get next packet */
if (!RxPacket(Port,p,&PacketSize,Buffer,NCGchar,&EOTflag)) return(FALSE);
if (p==0)
{
    /* copy Filename */
    strcpy(Filename,Buffer);
    /* done if null packet 0 */
    if (Filename[0]=='\0')
    {
        DisplayLine("Batch Transfer Complete",NULL,0);
        return(TRUE);
    }
}
/* all done if EOT was received */
if (EOTflag)
{
    close(Handle);
    DisplayLine("Transfer Complete",NULL,0);
    return(TRUE);
}
/* process packet */
if (p==0)
{
    /* open file using filename in packet 0 */
    Handle = open(Filename,O_CREAT|O_TRUNC|O_WRONLY|O_BINARY,S_IWRITE);
    if (Handle<0)
    {
        strcat(Buffer," -- open failed");
        DisplayLine(Buffer,NULL,0);
        return(FALSE);
    }
    /* get file length */
    FileSize = atol(&Buffer[1+strlen(Buffer)]);
    /* must 'restart' after packet 0 */
    RxStartup(Port,&NCGchar);
}
else /* DATA packet */
```

```
        {
            /* write Buffer */
            if (BatchFlag)
            {
                if (FileSize<(long)PacketSize) i = (int) FileSize;
                else i = PacketSize;
                i = write(Handle,Buffer,i);
                FileSize -= (long)i;
            }
            else write(Handle,Buffer,PacketSize);
        } /* end -- else */
    } /* end -- for(p) */
} /* end - RxyModem */

int TxCAN(Port)
int Port;
{
    int i;

    for(i=0;i<6;i++) PutChar(Port,CAN);
    return(0);
}
```

```
/********** XYPACKET.c **********/

#include <stdio.h>
#include <fcntl.h>
#include <sys\types.h>
#include <sys\stat.h>
#include "pcl4c.h"
#include "term.h"
#include "ascii.h"

#define DEBUG 0
#define FALSE 0
#define TRUE !FALSE

#define MAXTRY 5
#define LIMIT 20

extern GetChar();
extern PutChar();
extern SayError();
extern DisplayLine();

int TxPacket(Port,PacketNbr,PacketSize,Buffer,NCGchar)
int Port;           /* COM port [0..3] */
int PacketNbr;      /* Packet # to send */
int PacketSize;     /* Packet size ( must be 128 or 1024 ) */
char Buffer[];      /* Data buffer */
char NCGchar;       /* NAK, 'C', or 'G' */
{
    int  i;
    int Code;
    unsigned short CheckSum;
    int Attempt;
    int PacketType;
    char temp[81];

    /* begin */
    #if DEBUG
        printf("[TxP: COM%d PacketNbr=%d PacketSize=%d NCGchar=%c]",
```

```
    1+Port,PacketNbr,PacketSize,NCGchar);
#endif
/* better be 128 or 1024 packet length */
if (PacketSize==1024) PacketType = STX;
else PacketType = SOH;
PacketNbr &= 0x00ff;
/* make up to MAXTRY attempts to send this packet */
for (Attempt=1; Attempt<=MAXTRY; Attempt++)
{
    /* send SOH/STX  */
    Code = PutChar(Port,PacketType);
    /* send packet # */
    Code = PutChar(Port,PacketNbr);
    /* send 1's complement of packet */
    Code = PutChar(Port,255-PacketNbr);
    /* send data */
    CheckSum = 0;
    for (i=0; i<PacketSize; i++)
    {
        Code = PutChar(Port,Buffer[i]);
        if (NCGchar==NAK) CheckSum += Buffer[i];
        else CheckSum = UpdateCRC(CheckSum, Buffer[i]);
    }
    /* send checksum */
    if (NCGchar==NAK)
    {
        Code = PutChar(Port,CheckSum & 0x00ff );
    }
    else
    {
        Code = PutChar(Port, (CheckSum>>8) & 0x00ff );
        Code = PutChar(Port, CheckSum & 0x00ff );
    }
    /* no ACK to wait for if 'G' */
    if (NCGchar=='G')
    {
        if (PacketNbr==0) SioDelay(SHORT_WAIT*ONE_SECOND/2);
        return(TRUE);
    }
```

```
        /* wait for receivers ACK */
        Code = GetChar(Port,LONG_WAIT*ONE_SECOND);
        if ((char)Code==CAN)
        {
            DisplayLine("*** Canceled by REMOTE ***",NULL,0);
            return(FALSE);
        }
        if ((char)Code==ACK) return(TRUE);
        if ((char)Code!=NAK)
        {
            PacketError(Port,PacketNbr,Attempt,"Out of sync");
            return(FALSE);
        }
        /* Attempt again */
    } /* end -- for(Attempt) */
    /* can't send packet ! */
    SayError(Port,"packet timeout");
    return(FALSE);
} /* end -- TxPacket */

int RxPacket(Port,PacketNbr,PacketSizePtr,Buffer,NCGchar,EOTptr)
int Port;           /* COM port [0..3] */
int PacketNbr;      /* Packet # expected */
int *PacketSizePtr; /* Pointer to PacketSize received ( 128 or 1024) */
char Buffer[];      /* 1024 byte data buffer */
char NCGchar;       /* NAK, C, or G */
int *EOTptr;        /* Pointer to EOT flag */
{
    int i;
    int Code;
    int Attempt;
    int RxPacketNbr;
    int RxPacketNbrComp;
    unsigned short CheckSum;
    unsigned short RxCheckSum;
    unsigned short RxCheckSum1, RxCheckSum2;
    /*char PacketType;*/
    char temp[81];
```

```
/* begin */
#if DEBUG
    printf("[RxP: COM%d PacketNbr=%d NCGchar=%c EOTflag=%d]",
    1+Port,PacketNbr,NCGchar,*EOTptr);
#endif
PacketNbr &= 0x00ff;
for (Attempt=1; Attempt<=MAXTRY; Attempt++)
{
    /* wait for SOH / STX */
    Code = GetChar(Port,SHORT_WAIT*ONE_SECOND);
    if (Code==-1)
    {
        PacketError(Port,PacketNbr,Attempt,"timed out waiting for SOH/STX");
        return(FALSE);
    }
    switch ((char)Code)
    {
        case SOH:
            /* 128 byte buffer incoming */
            /*PacketType = SOH;*/
            *PacketSizePtr = 128;
            break;
        case STX:
            /* 1024 byte buffer incoming */
            /*PacketType = STX;*/
            *PacketSizePtr = 1024;
            break;
        case CAN:
            /* sender has canceled ! */
            DisplayLine("*** Canceled by REMOTE ***",NULL,0);
            return(FALSE);
        case EOT:
            /* all packets have been sent */
            Code = PutChar(Port,ACK);
            *EOTptr = TRUE;
            return(TRUE);
        default:
            /* error ! */
            sprintf(temp,"Expecting SOH/STX/EOT/CAN not %xH",(char)Code);
```

```
            PacketError(Port,PacketNbr,Attempt,temp);
            return(FALSE);
        }
        /* receive packet # */
        Code = GetChar(Port,SHORT_WAIT*ONE_SECOND);
        if (Code==-1)
        {
            PacketError(Port,PacketNbr,Attempt,"timed out waiting for packet number");
            return(FALSE);
        }
        RxPacketNbr = 0x00ff & Code;
        /* receive 1's complement */
        Code = GetChar(Port,SHORT_WAIT*ONE_SECOND);
        if (Code==-1)
        {
            PacketError(Port,PacketNbr,Attempt,"timed out waiting for complement of packet #");
            return(FALSE);
        }
        RxPacketNbrComp = 0x00ff & Code;
        /* verify packet number */
        if (RxPacketNbr+RxPacketNbrComp!=255)
        {
            PacketError(Port,PacketNbr,Attempt,"Bad packet number");
            return(FALSE);
        }
        /* receive data */
        CheckSum = 0;
        for (i=0; i<*PacketSizePtr; i++)
        {
            Code = GetChar(Port,LONG_WAIT*ONE_SECOND);
            if (Code==-1)
            {
                PacketError(Port,PacketNbr,Attempt,"timed out waiting for data for packet #");
                return(FALSE);
            }
            Buffer[i] = Code;
            /* compute CRC or checksum */
            if (NCGchar!=NAK) CheckSum = UpdateCRC(CheckSum,Code);
            else CheckSum = (CheckSum + Code) & 0x00ff;
```

```
}
/* receive CRC/checksum */
if (NCGchar!=NAK)
{
    /* receive 2 byte CRC */
    Code = GetChar(Port,SHORT_WAIT*ONE_SECOND);
    if (Code==-1)
    {
        PacketError(Port,PacketNbr,Attempt,"timed out waiting for 1st CRC byte");
        return(FALSE);
    }
    RxCheckSum1 = Code & 0x00ff;
    Code = GetChar(Port,SHORT_WAIT*ONE_SECOND);
    if (Code==-1)
    {
        PacketError(Port,PacketNbr,Attempt,"timed out waiting for 2nd CRC byte");
        return(FALSE);
    }
    RxCheckSum2 = Code & 0x00ff;
    RxCheckSum = (RxCheckSum1<<8) | RxCheckSum2;
}
else
{
    /* receive one byte checksum */
    Code = GetChar(Port,SHORT_WAIT*ONE_SECOND);
    if (Code==-1)
    {
        PacketError(Port,PacketNbr,Attempt,"timed out waiting for checksum");
        return(FALSE);
    }
    RxCheckSum = Code & 0x00ff;
}
/* don't send ACK if 'G' */
if (NCGchar=='G') return(TRUE);
/* packet # and checksum OK ? */
if ((RxCheckSum==CheckSum)&&(RxPacketNbr==PacketNbr))
{
    /* ACK the packet */
    PutChar(Port,ACK);
```

```
            return(TRUE);
        }
        /* bad packet */
        DisplayLine("Bad Packet",NULL,0);
        Code = PutChar(Port,NAK);
    } /* end -- for(Attempt) */
    /* can't receive packet */
    SayError(Port,"RX packet timeout");
    return(FALSE);
} /* end -- RxPacket */

PacketError(Port,Packet,Attempt,MsgPtr)
int Port;
int Packet;
int Attempt;
char *MsgPtr;
{
    char temp[81];

    sprintf(temp,"Packet %d : Attempt %d : %s",Packet,Attempt,MsgPtr);
    return( SayError(Port,temp) );
}

int TxStartup(Port,NCGcharPtr)
int Port;
char *NCGcharPtr;
{
    int i;
    int Code;

    #if DEBUG
        printf("### TxStartup");
    #endif
    /* clear Rx buffer */
    SioRxFlush(Port);
    /* wait for receivers start up NAK, 'C', or 'G' */
    for (i=1; i<LIMIT; i++)
    {
        if (SioKeyPress())
```

```
        {
            SayError(Port,"Aborted by user");
            return(FALSE);
        }
        Code = GetChar(Port,SHORT_WAIT*ONE_SECOND);
        if (Code==-1) continue;
        /* received a byte */
        if ((char)Code==NAK)
        {
            *NCGcharPtr = NAK;
            #if DEBUG
                printf("(CS) OK\n");
            #endif
            return(TRUE);
        }
        if ((char)Code=='C')
        {
            *NCGcharPtr = 'C';
            #if DEBUG
                printf("(CRC) OK\n");
            #endif
            return(TRUE);
        }
        if ((char)Code=='G')
        {
            *NCGcharPtr = 'G';
            #if DEBUG
                printf("(G) OK\n");
            #endif
            return(TRUE);
        }
    } /* end -- for(i) */
    /* no response */
    SayError(Port,"No response from receiver");
    return(FALSE);
} /* end -- TxStartup */

int RxStartup(Port,NCGcharPtr)
int Port;
```

```
char *NCGcharPtr;
{
    int i;
    int Code;

    #if DEBUG
        printf("### RxStartup");
        printf(" %c[%xH]",*NCGcharPtr,*NCGcharPtr);
    #endif
    /* clear Rx buffer */
    SioRxFlush(Port);
    /* Send NAKs, 'C's, or 'G's */
    for (i=1; i<LIMIT; i++)
    {
        if (SioKeyPress())
        {
            DisplayLine("*** Canceled by USER ***",NULL,0);
            return(FALSE);
        }
        /* stop attempting CRC/'G' after 1st 4 tries */
        if ((*NCGcharPtr!=NAK)&&(i==5)) *NCGcharPtr = NAK;
        /* tell sender that I am ready to receive */
        Code = PutChar(Port,*NCGcharPtr);
        Code = GetChar(Port,SHORT_WAIT*ONE_SECOND);
        if (Code==-1) continue;
        /* no error -- must be incoming byte -- push byte back onto queue ! */
        SioUnGetc(Port,(char)Code);
        #if DEBUG
            printf("OK\n");
        #endif
        return(TRUE);
    }
    /* no response */
    SayError(Port,"No response from sender");
    return(FALSE);
} /* end -- RxStartup */

int TxEOT(Port)
int Port;
```

```
{
    int i;
    int Code;
    for (i=0; i<10; i++)
    {
        Code = PutChar(Port,EOT);
        /* await response */
        Code = GetChar(Port,SHORT_WAIT*ONE_SECOND);
        if ((char)Code==ACK) return(TRUE);
    }
    return(FALSE);
} /* end -- TxEOT */
```

```
/********** TERMINAL.C **********/
/*
** EXAMPLE CODE: Terminal emulator. Can transfer files using
** XMODEM, YMODEM, and YMODEM-G protocols.
**
** See TERM.H for configuration parameters.
**
** Link with TERM_IO, MODEM_IO, DIR, CRC, DOS, XYMODEM, and XYPACKET.
** See TERM makefiles.
**
** Do NOT select YMODEM-G when using a null modem cable unless you are
** certain that RTS & CTS are reversed ( this is usually not true ).
**
** This example program (not the PCL4C library) is donated to
** the Public Domain by MarshallSoft Computing, Inc. It is
** provided as an example of the use of the PCL4C.
*/
/***** TERMINAL.C was adapted for the Magnetic Login System. *****/

#include <stdio.h>
#include <process.h>
#include <stdlib.h>
#include <fcntl.h>
#include <sys\types.h>
#include <sys\stat.h>
#include "pcl4c.h"
#include "ascii.h"
#include "term.h"

#define FALSE 0
#define TRUE !FALSE
#define NORMAL 0x07
#define INVERSE 0x70
#define MESSAGE_POS 48

void MyCrtWrite();
void MyStart();
void ProcessESC();
void ShowProtocol();
```

```
void ShowMessage();
void MyExit();
void SayFilename();
void ErrorCheck();

struct dbase
    {
    char name[30];
    char code[21];
    char pass[10];
    };

/*** Global variables ***/

int Port;               /* current COM port [0..3] */
char Filename[15];      /* file name buffer */
char Buffer[1024];      /* block buffer */
char RxBuf[2048];       /* receive buffer */
char *BaudRate[10] =  {"300","600","1200","2400","4800","9600",
       "19200","38400","57600","115200"};
int BaudCode;           /* baud rate code ( index into BaudRate[] ) */
char *ModelText[4] = {"Small","Compact","Medium","Large"};
char DTAbuffer[128];
char NCGchar  = NAK;
int OneKflag  = FALSE;
int BatchFlag = FALSE;
char Protocol = 'X';
int numuser;
struct dbase data[10];

/*** main program ***/

main(argc,argv)
int argc;
char *argv[];
{
    int DataFlag = FALSE;
    char c;
    int i, k;
```

```
int n, rc;
int Delta;          /* delta port status */
int Status;         /* port status */
char Version;
char temp[81];
int slen;
FILE *user;
int  ctemp,j;
char codein[20];
char passwd[10];
char buffer[50];
int  pbuf =0;
int  passcount=0;

/* right number of parameters ? */
if (argc!=3)
{
    printf("Usage: 'TERM port baud' -- example 'TERM 1 9600'\n");
    exit(1);
}
/* get port number from command line */
Port = atoi(argv[1]) - 1;
if ((Port<0) || (Port>3))
{
    printf("Port must be 1 to 4\n");
    exit(1);
}
/* get baud rate from command line */
BaudCode = BaudMatch(argv[2]);
if (BaudCode<0)
{
    printf("Cannot recognize baud rate = %s\n",argv[2]);
    exit(1);
}
/* setup transmit & receive buffer */
ErrorCheck( SioRxBuf(Port,RxBuf,Size2048) );
/* set parms & reset (initialize) COM port */
ErrorCheck( SioParms(Port,NoParity,OneStopBit,WordLength8) );
MyStart(Port,BaudCode);
```

```
/* init CRC table */
InitCRC();
/* initialize screen */
Scroll(0,0,24,79,0,NORMAL);
/* display status message */
sprintf(temp," COM%d %s %c 'ESC for menu'  ",1+Port,BaudRate[BaudCode],Protocol);
ShowMessage(0,temp);
Position(1,0);
/* display some info */
puts("*****************************************");
puts("**      Magnetic Login System v1.0       **");
puts("**            Apply Physic#9             **");
puts("**                                       **");
puts("**  Nipon Jantri              33504016    **");
puts("**  Thawatchai Iemmanassakul  33504012    **");
puts("*****************************************\n");

if (!(LoadDbase(data)))  MyExit(0,"error!! user.dat not found.");
puts("Database OK.");
for (i=0; i<50; i++)  buffer[i] = 0;      /* clear buffer */
/* send DTR to Terminal */
puts("Sending  DTR");
SioDTR(Port,'S');
/* check for 3 times get password */
do
{
    while (strstr(buffer,"ID to IBM\n\r") == NULL)
    {
        /* Control-BREAK ? */
        if (SioBrkKey()) MyExit(0,"User pressed Ctrl-BREAK");
        /* was key pressed ? */
        if (SioKeyPress())
        {
            /* read key press */
            i = SioKeyRead();
            if ((char)i==ESC)
            {
                /* process user's request */
                ProcessESC();
```

```
            ShowMessage(strlen(BaudRate[BaudCode])+9,"'ESC for Menu' ");
            continue;
        }
        else PutChar(Port,i);
    }
    /* was break detected ? */
    if (SioBrkSig(Port,'D'))  DisplayLine("BREAK detected ",NULL,0);
    /* any incoming over serial port ? */
    i = GetChar(Port,0);
    if (i)-1)
    {
        /* good character */
        if (DataFlag&((i<0x20)||(i>0x7e)))
        {
            MyCrtWrite('^');
            MyCrtWrite('@'+i);
        }
        else
        {
            buffer[pbuf++] = i;
            MyCrtWrite(i);
            if (pbuf >= 49)
            {
                for (i=0; i<50; i++)
                buffer[i] = 0;              /* clear buffer */
                pbuf = 0;                   /* point to start of buffer */
            }
        }
    }
} /* end -- while */
for (i=0; i<50; i++)
buffer[i] = 0;              /* clear buffer */
i=0;
do
    if ((ctemp = GetChar(Port,0)) > -1)
    {
        codein[i] = (char)ctemp;
        i++;
    } while(i<20);
```

```
    codein[i] = 0x0;
    i=0;
    do
        if ((ctemp = GetChar(Port,0)) > -1)
        {
            passwd[i] = (char)ctemp;
            i++;
        } while(ctemp != 0);
    passcount++;
    if (passcount >= 3)
    MyExit(0,"It's more time to Login");
} /* end -- for */
while (!(checkIDpass(data,codein,passwd)));
printf("Waiting for Modem DSR.");
while (!SioDSR(Port))
{
    if (SioKeyPress()||SioBrkKey())  MyExit(0,"Aborted by user");
    SioDelay(18);
}
puts("  OK.\n");
/* Modem is ready now */
do
   if ((ctemp = GetChar(Port,0)) > -1)  putchar(ctemp);
while (ctemp != 0);
while ((ctemp = getch()) != '\r')
{
    PutChar(Port,ctemp);
    putch(ctemp);
}
puts("");
PutChar(Port,0x0);
puts("Sending RTS");
SioRTS(Port,'S');
printf("Waiting for Modem CTS.");
while (!SioCTS(Port))
{
    if (SioKeyPress()||SioBrkKey())  MyExit(0,"Aborted by user");
    SioDelay(18);
}
```

```
puts("  OK.\n");
/* see TERM.H for definition of AT_COMMAND_SET */
#if AT_COMMAND_SET
    /* wait for Modem to say its ready */
    printf("Waiting for Modem DSR.");
    while (!SioDSR(Port))
    {
        if (SioKeyPress()||SioBrkKey())  MyExit(0,"Aborted by user");
        putchar('.');
        SioDelay(18);
    }
    putchar('\n');
    /* initialize (Hayes compatible) modem */
    SendTo(Port,"!AT!!~");
    SendTo(Port,"!AT E1 S7=60 S11=60 V1 X1 Q0 S0=1!");
    if (WaitFor(Port,"OK")) printf("\nMODEM READY\n");
    else printf("\nWARNING: Expected OK not received\n");
#endif
/* enter terminal loop */
SioRxFlush(Port);
while (1)
{
    /* Control-BREAK ? */
    if (SioBrkKey())  MyExit(0,"User pressed Ctrl-BREAK");
    /* was key pressed ? */
    if (SioKeyPress())
    {
        /* read key press */
        i = SioKeyRead();
        if ((char)i==ESC)
        {
            /* process user's request */
            ProcessESC();
            ShowMessage(strlen(BaudRate[BaudCode])+9,"'ESC for Menu' ");
            continue;
        }
        else PutChar(Port,i);
    }
    /* was break detected ? */
```

```
        if (SioBrkSig(Port,'D'))  DisplayLine("BREAK detected ",NULL,0);
        /* any incoming over serial port ? */
        i = GetChar(Port,0);
        if (i>-1)
        {
            /* good character */
            if (DataFlag&((i<0x20)||(i>0x7e)))
            {
                MyCrtWrite('^');
                MyCrtWrite('@'+i);
            }
            else MyCrtWrite(i);
        }
        /* any change in DCD or DSR ? */
        Delta = SioModem(Port,DeltaDCD|DeltaDSR);
        if (Delta)
        {
            /* display new status */
            Status = SioModem(Port,(char)(DCD|DSR));
            if (!Status&DeltaDCD) MyExit(0,"Dropped DCD");
            if (!Status&DeltaDSR) MyExit(0,"Dropped DSR");
        }
    } /* end -- key pressed */
}

int LoadDbase(struct dbase *pdata)
{
    FILE *user;
    int  ctemp, j;

    /* tell user to wait for reading database */
    puts("Please Waiting.. reading User Database.");
    if ((user = fopen("USER.DAT", "rt")) == NULL)  return 0;
    /* Fill database from file to memory */
    for (numuser=1; ((EOF != fgetc(user)) && numuser <= 10); numuser++)
    {
        fseek(user, (long)(ftell(user) - 1), SEEK_SET);
        for (j=0; ((ctemp = fgetc(user)) != 0x20); j++)
        pdata[numuser-1].name[j] = ctemp;
```

```
        while ((ctemp = fgetc(user)) == 0x20);
        fseek(user, (long)(ftell(user) - 1), SEEK_SET);
        pdata[numuser-1].name[j++] = 0x20;
        for ( ;((ctemp = fgetc(user)) != 0x20); j++)
        pdata[numuser-1].name[j] = ctemp;
        pdata[numuser-1].name[j++] = 0x0;
        while ((ctemp = fgetc(user)) == 0x20);
        fseek(user, (long)(ftell(user) - 1), SEEK_SET);
        for (j=0; ((ctemp = fgetc(user)) != 0x20); j++)
            pdata[numuser-1].code[j] = ctemp;
        pdata[numuser-1].code[j++] = 0x0;
        while ((ctemp = fgetc(user)) == 0x20);
        fseek(user, (long)(ftell(user) - 1), SEEK_SET);
        for(j=0; ((ctemp != 0xa) && (ctemp = fgetc(user)) != 0x20); j++)
            pdata[numuser-1].pass[j] = ctemp;
        pdata[numuser-1].pass[--j] = 0x0;
        fseek(user, (long)(ftell(user) - 1), SEEK_SET);
        while ((ctemp = fgetc(user)) == 0x20);
        fseek(user, (long)(ftell(user) - 1), SEEK_SET);
        while ((ctemp = fgetc(user)) != 0xa);
    } /* end -- for */
    fclose(user);
    return 1;
} /* end -- LoadDbase */

int checkIDpass(struct dbase *pdata,char *codein,char *passwd)
{
    int i;

    /* check if ID and password match */
    i = 0;
    while ((i<numuser) && ((strcmp(pdata[i].code,codein) != 0)))
        i++;
    if (i < numuser)
    {
        if (!(strcmp(pdata[i].pass,passwd)))
        {
            char result[10] = "Login OK.";
            int j;
```

```
            for (j=0; (j<(strlen(result)+1)); j++)
            PutChar(Port, result[j]);
            printf("%s can access to this system\n", pdata[i].name);
            return 1;
        }
        /* password error */
        else
        {
            char result[16] = "Password Error.";

            for (i=0; (i<(strlen(result)+1)); i++)
            PutChar(Port, result[i]);
            return 0;
        }
    }
    else
    {
        char result[14] = "ID not found.";

        for (i=0; (i<(strlen(result)+1)); i++)
        PutChar(Port, result[i]);
        return 0;
    }
} /* end -- checkIDpass */

/*** write to screen except for bottom line ***/

void MyCrtWrite(ch)
char ch;
{
    /* write character */
    SioCrtWrite(ch);
    /* scroll all but bottom line */
    if(GetRow()==24)
    {
        Scroll(0,0,23,79,1,NORMAL);
        Position(23,0);
    }
}
```

```
/*** make multiple attempts to reset port ***/

void MyStart(Port,BaudCode)
int Port;
int BaudCode;
{
    int i, rc;

    /* try up to 3 times to reset COM port */
    for (i=0; i<3; i++)
    {
        printf("Resetting COM%d at %s baud\n",Port+1,BaudRate[BaudCode]);
        if ((rc = SioReset(Port,BaudCode))==0)  return;
        if (rc<0)  MyExit(rc,"Error resetting port");
        SioDone(Port);
        /* display errors */
        if (rc&OverrunError) puts("Overrun Error");
        if (rc&ParityError)  puts("Parity Error");
        if (rc&FramingError) puts("Framing Error");
        if (rc&BreakDetect)  puts("Break Detect");
    }
    exit(1);
}

/*** find baud rate string in table ***/

int BaudMatch(ptr)
char *ptr;
{
    int i;

    /* find baud rate in table */
    for (i=0; i<10; i++)  if (strcmp(BaudRate[i],ptr)==0) return(i);
    return(-1);
}

/*** user pressed Escape */
```

```
void ProcessESC()
{
    int i;
    int rc;
    int c1, c2;
    char Answer[2]; /* array for 1 char answer */
    int row, col;

    /* user pressed <ESC> */
    Answer[0] = '?';
    Answer[1] = '\0';
    DisplayLine("Q)uit P)rotocol S)end R)eceive: ",Answer,1);
    if (strlen(Answer))  switch (toupper(Answer[0]))
        {
            case 'P':
                #if RTS_CTS_CONTROL
                    DisplayLine("X) xmodem Y) ymodem G) ymodem-g: ",Answer,1);
                #else
                    DisplayLine("X) xmodem Y) ymodem: ",Answer,1);
                #endif
                if (strlen(Answer))  switch (toupper(Answer[0]))
                    {
                        case 'X':
                            Protocol = 'X';
                            ShowProtocol();
                            OneKflag = FALSE;
                            BatchFlag = FALSE;
                            NCGchar = NAK;
                            DisplayLine("Protocol = XMODEM",NULL,1);
                            break;
                        case 'Y':
                            Protocol = 'Y';
                            ShowProtocol();
                            OneKflag = TRUE;
                            BatchFlag = TRUE;
                            NCGchar = 'C';
                            DisplayLine("Protocol = YMODEM",NULL,1);
                            break;
```

```
            #if RTS_CTS_CONTROL
            case 'G':
                Protocol = 'G';
                ShowProtocol();
                OneKflag = TRUE;
                BatchFlag = TRUE;
                NCGchar = 'G';
                DisplayLine("Protocol = YMODEM-G",NULL,1);
                break;
            #endif
            default:
                DisplayLine("Must answer X, Y, or G",NULL,1);
                break;
        }
    break; /* end of case 'P' */
case 'Q':
    MyExit(0,"User pressed ESC");
    break;
case 'R':
    ShowMessage(strlen(BaudRate[BaudCode])+9,"'CTRL-X aborts'");
    /* XMODEM / YMODEM receive */
    if (BatchFlag)
    {
        do
        {
            /* receive files till get empty filename */
            RxyModem(Port,Filename,Buffer,NCGchar,BatchFlag);
            if (SioKeyPress()) break;
        } while (Filename[0]!='\0');
    }
    else /* not Batch */
    {
        DisplayLine("Enter filename:",Filename,15);
        if (strlen(Filename)==0) break;
        RxyModem(Port,Filename,Buffer,NCGchar,BatchFlag);
    }
    break;
case 'S':
    ShowMessage(strlen(BaudRate[BaudCode])+9,"'CTRL-X aborts'");
```

```
                DisplayLine("Enter filename:",Filename,15);
                if (strlen(Filename)==0)  break;
                if (BatchFlag)
                {
                    /* YMODEM send */
                    setDTA(DTAbuffer);
                    if (FindFirst(Filename))
                    {
                        SayFilename(&DTAbuffer[30]);
                        TxyModem(Port,&DTAbuffer[30],Buffer,OneKflag,BatchFlag);
                        while (FindNext())
                        {
                            SioDelay(4);
                            SayFilename(&DTAbuffer[30]);
                            TxyModem(Port,&DTAbuffer[30],Buffer,OneKflag,BatchFlag);
                        }
                        /* send empty filename */
                        Filename[0] = '\0';
                        SioDelay(5);
                        TxyModem(Port,Filename,Buffer,OneKflag,BatchFlag);
                    }
                }
                else
                {
                    /* XMODEM send */
                    TxyModem(Port,Filename,Buffer,OneKflag,BatchFlag);
                } /* end -- if(BatchFlag) */
                break;
            default:
                DisplayLine("Must answer Q, P, S, or R",NULL,0);
                break;
        } /* end switch */
}


/*** show protocol choosen ***/

void ShowProtocol()
{
```

```
    int SaveRow;
    int SaveCol;

    SaveRow = GetRow();
    SaveCol = GetCol();
    Position(24,MESSAGE_POS+strlen(BaudRate[BaudCode])+7);
    AttrWrite(Protocol,INVERSE);
    Position(SaveRow,SaveCol);
}

/*** show chosen message ***/

void ShowMessage(Pos,Msg)
int Pos;
char *Msg;
{
    int i;
    int SaveRow;
    int SaveCol;
    int Col;
    SaveRow = GetRow();
    SaveCol = GetCol();
    Col = MESSAGE_POS+Pos;
    for (i=0; i<strlen(Msg); i++)
    {
        Position(24,Col++);
        AttrWrite(Msg[i],INVERSE);
    }
    Position(SaveRow,SaveCol);
}

/*** exit program ***/

void MyExit(code,msg)
int code;
char *msg;
{
    int rc;
    if (code<0) SioError(code);
```

```
    /* Assert UART break & sign off */
    SioBrkSig(Port,'A');
    printf("\nTERMINATING: %s\n",msg);
    SioDelay(ONE_SECOND/2);
    SioDone(Port);
    exit(0);
}

/*** display file name in status area ***/

void SayFilename(Text)
char *Text;
{
    char Temp[40];
    strcpy(Temp,"Sending ");
    strcat(Temp,Text);
    DisplayLine(Temp,NULL,0);
    SioDelay(ONE_SECOND/2);
}

/*** check for error ***/

void ErrorCheck(Code)
int Code;
{
    /* trap PCL error codes */
    if (Code<0)
    {
        SioError(Code);
        exit(1);
    }
} /* end ErrorCheck */
```

```
/********** HOST.C **********/
/*
** EXAMPLE CODE: Terminal emulator. Can transfer files using
** XMODEM, YMODEM, and YMODEM-G protocols.
**
** See TERM.H for configuration parameters.
**
** Link with TERM_IO, MODEM_IO, DIR, CRC, DOS, XYMODEM, and XYPACKET.
** See TERM makefiles.
**
** Do NOT select YMODEM-G when using a null modem cable unless you are
** certain that RTS & CTS are reversed ( this is usually not true ).
**
** This example program (not the PCL4C library) is donated to
** the Public Domain by MarshallSoft Computing, Inc. It is
** provided as an example of the use of the PCL4C.
*/
/***** HOST.C was adapted for the Magnetic Login System. *****/

#include <stdio.h>
#include <fcntl.h>
#include <sys\types.h>
#include <sys\stat.h>
#include "pcl4c.h"
#include "ascii.h"
#include "term.h"

#define FALSE 0
#define TRUE !FALSE
#define NORMAL 0x07
#define INVERSE 0x70
#define MESSAGE_POS 48

void MyCrtWrite();
void MyStart();
void ProcessESC();
void ShowProtocol();
void ShowMessage();
void MyExit();
```

```
void SayFilename();
void ErrorCheck();

/*** Global variables ***/

int Port;               /* current COM port [0..3] */
char Filename[15];      /* file name buffer */
char Buffer[1024];      /* block buffer */
char RxBuf[2048];       /* receive buffer */
char *BaudRate[10] = {"300","600","1200","2400","4800","9600",
                      "19200","38400","57600","115200"};
int BaudCode;           /* baud rate code ( index into BaudRate[] ) */
char *ModelText[4] = {"Small","Compact","Medium","Large"};
char DTAbuffer[128];
char NCGchar  = NAK;
int OneKflag  = FALSE;
int BatchFlag = FALSE;
char Protocol = 'X';

/*** main program ***/

main(argc,argv)
int argc;
char *argv[];
{
    int DataFlag = FALSE;
    char c;
    int i, k;
    int n, rc;
    int Delta;          /* delta port status */
    int Status;         /* port status */
    char Version;
    char temp[81];
    int  dsr_c = 1;
    int  cts_c = 1;

    char Welcom[1000]  =  {"Welcome to Magnetic Login System.  \n \
       Here is host that fill with many and many information. \n \
       You can select some information you want to receive.   \n \
```

```
    and pay back once in times.\nPlease select\n \
      1. Data Bank          \n\
      2. Stock Analysis     \n\
      3. News               \n\
      4. quit               \n\
    Select? ");

int slen;


/* right number of parameters ? */
if (argc!=3)
{
    printf("Usage: 'TERM port baud' -- example 'TERM 1 9600'\n");
    exit(1);
}
/* get port number from command line */
Port = atoi(argv[1]) - 1;
if ((Port<0) || (Port>3))
{
    printf("Port must be 1 to 4\n");
    exit(1);
}
/* get baud rate from command line */
BaudCode = BaudMatch(argv[2]);
if (BaudCode<0)
{
    printf("Cannot recognize baud rate = %s\n",argv[2]);
    exit(1);
}
/* setup transmit & receive buffer */
ErrorCheck( SioRxBuf(Port,RxBuf,Size2048) );
/* set parms & reset (initialize) COM port */
ErrorCheck( SioParms(Port,NoParity,OneStopBit,WordLength8) );
MyStart(Port,BaudCode);
/* init CRC table */
InitCRC();
/* initialize screen */
Scroll(0,0,24,79,0,NORMAL);
```

```
/* display status message */
sprintf(temp," COM%d %s %c 'ESC for menu'  ",1+Port,BaudRate[BaudCode],Protocol);
ShowMessage(0,temp);
Position(1,0);
/* display some info */
puts("*******************************************");
puts("**        Magnetic Login System          **");
puts("**            Apply Physic#9             **");
puts("**                                       **");
puts("**  Nipon Jantri              33504016    **");
puts("**  Thawatchai Iemmanassakul  33504012    **");
puts("*******************************************\n");
/* set DTR and RTS */
puts("Sending  DTR");
SioDTR(Port,'S');
/* wait for Modem to say its ready */
printf("Waiting for Modem DSR.");
while (!SioDSR(Port))
{
    if (SioKeyPress()||SioBrkKey()) MyExit(0,"Aborted by user");
    putchar('.');
    SioDelay(18);
}
puts("  OK.\n");
/* puts("Waiting for DSR. ");
while (dsr_c)
{
    if (SioKeyPress()||SioBrkKey()) MyExit(0,"Aborted by user");
    if (SioDSR(Port))
    {
        puts("OK.\n");
        dsr_c = 0;
    }
}  */
puts("Sending  RTS");
SioRTS(Port,'S');
puts("Waiting for CTS. ");
while (cts_c)
{
```

```
    if (SioKeyPress()||SioBrkKey()) MyExit(0,"Aborted by user");
    if (SioCTS(Port))
    {
        puts("OK.\n");
        cts_c = 0;
    }
}
/* see TERM.H for definition of AT_COMMAND_SET */
#if AT_COMMAND_SET
    /* wait for Modem to say its ready */
    printf("Waiting for Modem DSR.");
    while ( !SioDSR(Port) )
    {
        if(SioKeyPress()||SioBrkKey()) MyExit(0,"Aborted by user");
        putchar('.');
        SioDelay(18);
    }
    putchar('\n');
    /* initialize (Hayes compatible) modem */
    SendTo(Port,"!AT!!~");
    SendTo(Port,"!AT E1 S7=60 S11=60 V1 X1 Q0 S0=1!");
    if(WaitFor(Port,"OK")) printf("\nMODEM READY\n");
    else printf("\nWARNING: Expected OK not received\n");
#endif
for (i=0; i<=100; i++)  PutChar(Port,'0'+i);
for (i=0; i<24; i++)  PutChar(Port,'a'+i);
for (i=0; i<24; i++)  PutChar(Port,'A'+i);
slen = strlen(Welcom);
for (i=0; i<=slen; i++)
{
    PutChar(Port,Welcom[i]);
    putchar(Welcom[i]);
}
/* enter terminal loop */
SioRxFlush(Port);
while (1)
{
    /* Control-BREAK ? */
    if (SioBrkKey()) MyExit(0,"User pressed Ctrl-BREAK");
```

```
        /* was key pressed ? */
        if (SioKeyPress())
        {
            /* read key press */
            i = SioKeyRead();
            if ((char)i==ESC)
            {
                /* process user's request */
                ProcessESC();
                ShowMessage(strlen(BaudRate[BaudCode])+9,"'ESC for Menu' ");
                continue;
            }
            else PutChar(Port,i);
        }
        /* was break detected ? */
        if (SioBrkSig(Port,'D'))  DisplayLine("BREAK detected ",NULL,0);
        /* any incoming over serial port ? */
        i = GetChar(Port,0);
        if (i>-1)
        {
            /* good character */
            if (DataFlag&((i<0x20)||(i>0x7e)))
            {
                MyCrtWrite('^');
                MyCrtWrite('@'+i);
            }
            else MyCrtWrite(i);
        }
        /* any change in DCD or DSR ? */
        Delta = SioModem(Port,DeltaDCD|DeltaDSR);
        if (Delta)
        {
            /* display new status */
            Status = SioModem(Port,(char)(DCD|DSR));
            if (!Status&DeltaDCD)  MyExit(0,"Dropped DCD");
            if (!Status&DeltaDSR)  MyExit(0,"Dropped DSR");
        }
    } /* end -- key pressed */
}
```

```
/*** write to screen except for botton line ***/

void MyCrtWrite(ch)
char ch;
{
    /* write character */
    SioCrtWrite(ch);
    /* scroll all but bottom line */
    if (GetRow()==24)
    {
        Scroll(0,0,23,79,1,NORMAL);
        Position(23,0);
    }
}

/*** make multiple attempts to reset port ***/

void MyStart(Port,BaudCode)
int Port;
int BaudCode;
{
    int i, rc;

    /* try up to 3 times to reset COM port */
    for (i=0;i<3;i++)
    {
        printf("Resetting COM%d at %s baud\n",Port+1,BaudRate[BaudCode]);
        if ((rc = SioReset(Port,BaudCode))==0) return;
        if (rc<0) MyExit(rc,"Error resetting port");
        SioDone(Port);
        /* display errors */
        if (rc&OverrunError) puts("Overrun Error");
        if (rc&ParityError)  puts("Parity Error");
        if (rc&FramingError) puts("Framing Error");
        if (rc&BreakDetect)  puts("Break Detect");
    }
    exit(1);
}
```

```
/*** find baud rate string in table ***/

int BaudMatch(ptr)
char *ptr;
{
    int i;

    /* find baud rate in table */
    for (i=0; i<10; i++)  if (strcmp(BaudRate[i],ptr)==0) return(i);
    return(-1);
}

/*** user pressed Escape */

void ProcessESC()
{
    int i;
    int rc;
    int c1, c2;
    char Answer[2]; /* array for 1 char answer */
    int row, col;

    /* user pressed <ESC> */
    Answer[0] = '?';
    Answer[1] = '\0';
    DisplayLine("Q)uit P)rotocol S)end R)eceive: ",Answer,1);
    if (strlen(Answer))  switch (toupper(Answer[0]))
        {
            case 'P':
            #if RTS_CTS_CONTROL
                DisplayLine("X) xmodem Y) ymodem G) ymodem-g: ",Answer,1);
            #else
                DisplayLine("X) xmodem Y) ymodem: ",Answer,1);
            #endif
            if (strlen(Answer)) switch( toupper(Answer[0]) )
                {
                    case 'X':
                        Protocol = 'X';
```

```
            ShowProtocol();
            OneKflag = FALSE;
            BatchFlag = FALSE;
            NCGchar = NAK;
            DisplayLine("Protocol = XMODEM",NULL,1);
            break;
        case 'Y':
            Protocol = 'Y';
            ShowProtocol();
            OneKflag = TRUE;
            BatchFlag = TRUE;
            NCGchar = 'C';
            DisplayLine("Protocol = YMODEM",NULL,1);
            break;
        #if RTS_CTS_CONTROL
        case 'G':
            Protocol = 'G';
            ShowProtocol();
            OneKflag = TRUE;
            BatchFlag = TRUE;
            NCGchar = 'G';
            DisplayLine("Protocol = YMODEM-G",NULL,1);
            break;
        #endif
        default:
            DisplayLine("Must answer X, Y, or G",NULL,1);
            break;
    }
    break; /* end of case 'P' */
case 'Q':
    MyExit(0,"User pressed ESC");
    break;
case 'R':
    ShowMessage(strlen(BaudRate[BaudCode])+9,"'CTRL-X aborts'");
    /* XMODEM / YMODEM receive */
    if (BatchFlag)
    {
        do
        {
```

```
            /* receive files till get empty filename */
            RxyModem(Port,Filename,Buffer,NCGchar,BatchFlag);
            if (SioKeyPress()) break;
        } while(Filename[0]!='\0');
    }
    else /* not Batch */
    {
        DisplayLine("Enter filename:",Filename,15);
        if (strlen(Filename)==0) break;
        RxyModem(Port,Filename,Buffer,NCGchar,BatchFlag);
    }
    break;
case 'S':
    ShowMessage(strlen(BaudRate[BaudCode])+9,"'CTRL-X aborts'");
    DisplayLine("Enter filename:",Filename,15);
    if (strlen(Filename)==0) break;
    if (BatchFlag)
    {
        /* YMODEM send */
        setDTA(DTAbuffer);
        if (FindFirst(Filename))
        {
            SayFilename(&DTAbuffer[30]);
            TxyModem(Port,&DTAbuffer[30],Buffer,OneKflag,BatchFlag);
            while (FindNext())
            {
                SioDelay(4);
                SayFilename(&DTAbuffer[30]);
                TxyModem(Port,&DTAbuffer[30],Buffer,OneKflag,BatchFlag);
            }
            /* send empty filename */
            Filename[0] = '\0';
            SioDelay(5);
            TxyModem(Port,Filename,Buffer,OneKflag,BatchFlag);
        }
    }
    else
    {
        /* XMODEM send */
```

```
                    TxyModem(Port,Filename,Buffer,OneKflag,BatchFlag);
                } /* end -- if(BatchFlag) */
                break;
            default:
                DisplayLine("Must answer Q, P, S, or R",NULL,0);
                break;
        } /* end switch */
}

/*** show protocol choosen ***/

void ShowProtocol()
{
    int SaveRow;
    int SaveCol;

    SaveRow = GetRow();
    SaveCol = GetCol();
    Position(24,MESSAGE_POS+strlen(BaudRate[BaudCode])+7);
    AttrWrite(Protocol,INVERSE);
    Position(SaveRow,SaveCol);
}

/*** show chosen message ***/

void ShowMessage(Pos,Msg)
int Pos;
char *Msg;
{
    int i;
    int SaveRow;
    int SaveCol;
    int Col;

    SaveRow = GetRow();
    SaveCol = GetCol();
    Col = MESSAGE_POS+Pos;
    for (i=0; i<strlen(Msg); i++)
    {
```

```
        Position(24,Col++);
        AttrWrite(Msg[i],INVERSE);
    }
    Position(SaveRow,SaveCol);
}

/*** exit program ***/

void MyExit(code,msg)
int code;
char *msg;
{
    int rc;

    if (code<0) SioError(code);
    /* Assert UART break & sign off */
    SioBrkSig(Port,'A');
    printf("\nTERMINATING: %s\n",msg);
    SioDelay(ONE_SECOND/2);
    SioDone(Port);
    exit(0);
}

/*** display file name in status area ***/

void SayFilename(Text)
char *Text;
{
    char Temp[40];

    strcpy(Temp,"Sending ");
    strcat(Temp,Text);
    DisplayLine(Temp,NULL,0);
    SioDelay(ONE_SECOND/2);
}

/*** check for error ***/

void ErrorCheck(Code)
```

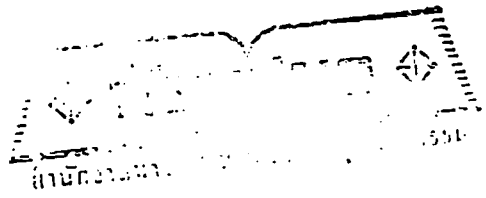
```
int Code;
{
    /* trap PCL error codes */
    if (Code<0)
    {
        SioError(Code);
        exit(1);
    }
} /* end ErrorCheck */
```

ภาคผนวก จ.

Data Sheets

International Standard 7811/2

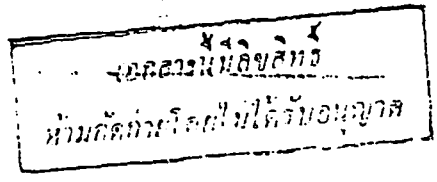
INTERNATIONAL ORGANIZATION FOR STANDARDIZATION•МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ•ORGANISATION INTERNATIONALE DE NORMALISATION



Identification cards — Recording technique — Part 2: Magnetic stripe

Cartes d'identification — Technique d'enregistrement — Partie 2: Magnétique

First edition — 1985-12-15



UDC 681.178.5 : 681.327.65 Ref. No. ISO 7811/2-1985 (E)

Descriptors : information interchange, data recording, identity cards, magnetic cards, magnetic recording, coded character sets, characteristics.

Identification cards — Recording technique — Part 2: Magnetic stripe

0 Introduction

This International Standard is one of a series of standards describing the parameters for identification cards as defined in clause 3 below and the use of such cards for international interchange.

1 Scope and field of application

This part of ISO 7811 specifies characteristics for a magnetic stripe (including any protective overlay) on an identification card, the encoding technique and coded character sets. The magnetic recordings are intended for machine reading.

2 References

ISO 7810, *Identification cards — Physical characteristics.*

ISO 7811, *Identification cards — Recording technique*

- *Part 4: Location of read-only magnetic tracks — Tracks 1 and 2.*
- *Part 5: Location of read-write magnetic track — Track 3.*

3 Definitions

For the purpose of this part of ISO 7811, the definition of "identification card" given in ISO 7810 and the following definitions apply.

3.1 primary standard: The NBS Master Standard Magnetic Tape (computer amplitude reference) kept in repository at the United States National Bureau of Standards (NBS).

NOTE — The relationship (correction factor) between the Master Standard and reference tape SRM 3200 is given by the NBS certificate supplied with the tape.

3.2 reference card[1]: A reference card, which shall be considered a secondary standard, comprises an ID card with a magnetic stripe consisting of secondary standard magnetic tape (computer amplitude reference) SRM 3200.

NOTE — The secondary reference card should be corrected to the master standard first using the correction factor provided by the supplier. Then the location of the window is calculated (see figure 5).

3.3 flux transition: The location of the maximum of the magnitude of the magnetic flux component normal to the surface of the magnetic stripe.

3.4 reference current (I_R): The minimum recording current amplitude (square wave) which causes on the reference card, under the given test conditions, a readback voltage amplitude equal to 80 % of the maximum amplitude (see figure 5) at a density of 8 ftpmm (flux transitions per millimetre) (200 ftpi (flux transitions per inch)).

3.5 test recording currents: Two test recording currents (square wave) at 350 % and 500 % of the reference current (I_R) shall be used.

3.6 average signal amplitude: The readback voltage, measured peak-to-peak, averaged over the total recording area of a card when recorded with the test recording current at the specified recording density.

3.7 reference signal amplitude: The maximum average signal amplitude of the reference card corrected to the master standard.

3.8 individual signal amplitude: The peak-to-peak amplitude of a single readback voltage signal.

3.9 test density: Densities of 8 ftpmm (200 ftpi) and 20 ftpmm (500 ftpi) which may be used for testing.

NOTE — When testing with the reference card, densities of 6 ftpmm (150 ftpi) and 16,6 ftpmm (420 ftpi) may be used. The correction factors are:

$$\frac{\text{amplitudes 6 ftpmm (150 ftpi)}}{\text{amplitudes 8 ftpmm (200 ftpi)}} \times 100 = 100\ \%$$

$$\frac{\text{amplitudes 16,6 ftpmm (420 ftpi)}}{\text{amplitudes 20 ftpmm (500 ftpi)}} \times 100 = \ldots$$

1) These cards can be ordered from Physikalisch-Technische Bundesanstalt, Lab. 1.41 — Bundesallee 100, D-3 300 Braunschweig, Germany, F.R. as long as available.

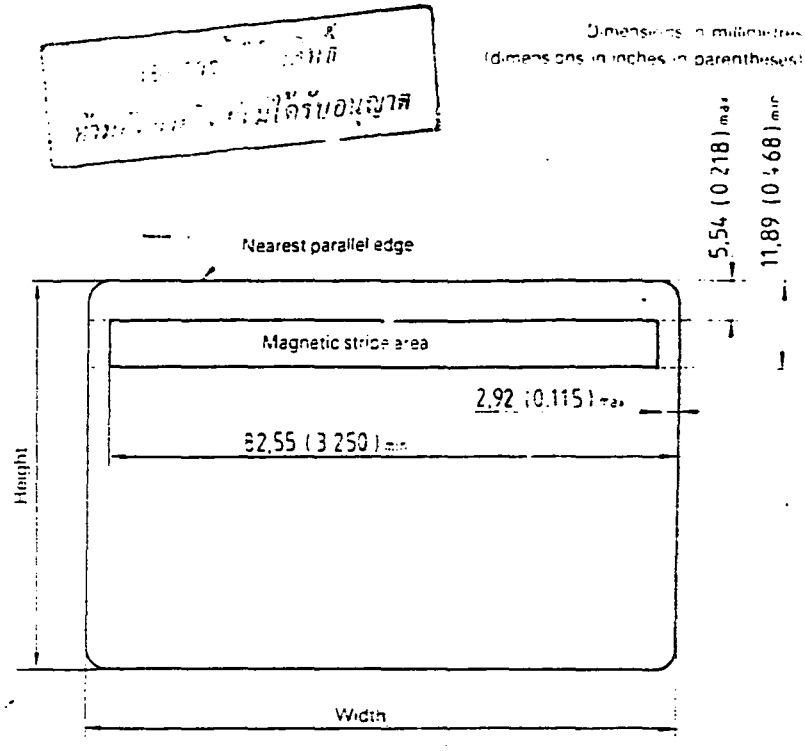


Figure 1 — Location of magnetic material for tracks 1 and 2 only on ID-1 type card

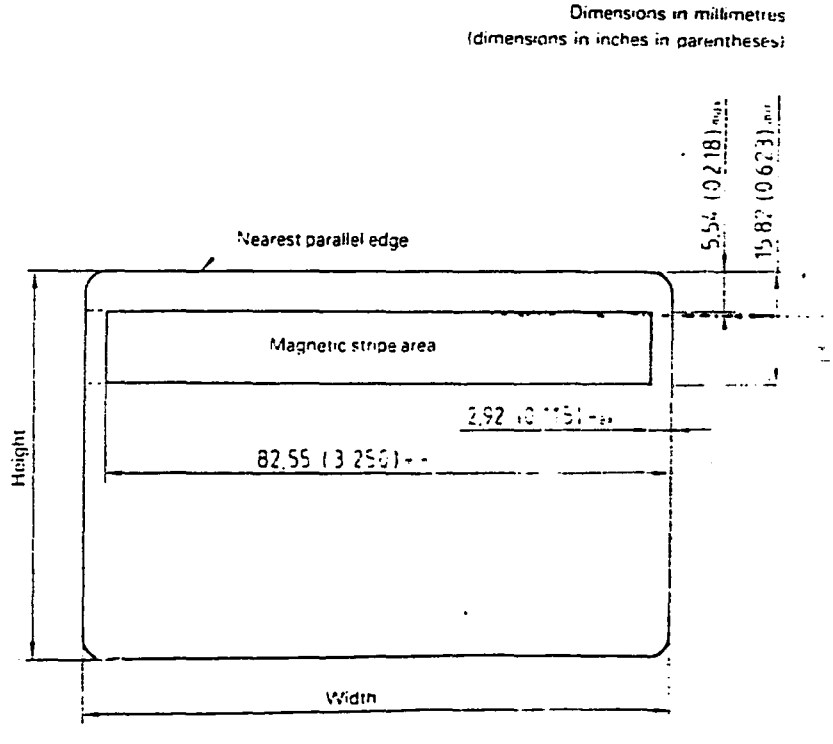


Figure 2 — Location of magnetic material for tracks 1, 2 and 3 on ID-1 type card

NOTE — While these dimensions state the maximum and minimum distance from the nearest parallel edge, the magnetic material areas are precluded from covering a greater area than indicated

4 Physical characteristics of the identification card

The identification card shall conform to the specifications given in ISO 7810.

WARNING — The attention of card issuers is drawn to the fact that information held on the magnetic stripe may be rendered ineffective through contamination by contact with dirt and certain commonly used chemicals including plasticizers. Further, exposure of the card to an intense magnetic field is likely to destroy the recorded data.

5 Physical characteristics of the magnetic material

5.1 Thickness

The height of the reading surface above the back surface of the card, when profiled with a stylus of between 0,38 mm (0.015 in) and 2,54 mm (0.100 in) radius, shall be

0 mm min. (0 in min.)

0,038 mm max. (0.001 5 in max.)

5.2 Surface roughness

The average surface irregularity of the magnetic surface shall not exceed 0,404 µm (15.9 µin) CLA (centreline average) in the longitudinal and transverse directions, when using a cut-off wavelength of 0,25 mm (0.01 in) or 0,76 mm (0.03 in) and a stylus of radius 2.54 µm max. (100 µin max.)

5.3 Surface profile

The average profile of the minimum magnetic stripe width (see figures 1 and 2) when measured parallel to the height of the card with a probe having a radius of 0,38 mm (0.015 in) to 2,54 mm (0.100 in) shall not show a vertical deviation from a straight line connecting the minimum strip width of more than 3,8 µm (150 µin) for every 2,54 mm (0.100 in) of stripe width (see figure 3).

During the measurement the back (magnetic stripe side) of the card shall be held parallel to the surface by means of a 2,2 N (0.51 lbf) load evenly distributed over the back of the measurement area

NOTES

1 Spiking in the profile caused by material "squirt-out" in hot stamping is not part of the stripe. It should not extend above the projected stripe surface (see figure 4).

2 [illegible] the magnetic material on top of printing is not recommended. When edge bleed ink covering the overlay seal is [illegible] [illegible] and delamination problems

3 It is necessary to have a defined method of measuring the surface profile in order to ensure uniformity. Methods of measurement are being evaluated and an agreed procedure will be added in this International Standard at the earliest opportunity.

5.4 Adhesion of stripe to card

The stripe shall not separate from the card under normal use.

6 Performance characteristics of the magnetic material for newly manufactured identification cards

6.1 General

This method uses a certified reference card whose magnetic material is traceable to a primary standard magnetic tape.

The test does not guarantee any minimum or maximum value of intrinsic coercivity H_{ci}. The specification of this parameter (if required) is left to the individual card purchasers. In general, higher coercivities provide greater resistance to erasure at increased cost.

6.2 Track 1, 2 and 3 (read only or read/write)

All measurements shall be made using the same device and under the same conditions.

6.2.1 Signal amplitude

When the magnetic material with any protective coatings in place is recorded with the test recording currents (see 3.5) at 8 ftpmm (200 ftpi) the signal amplitude shall be within 80 % to 130 % of the reference signal amplitude (see figure 5)

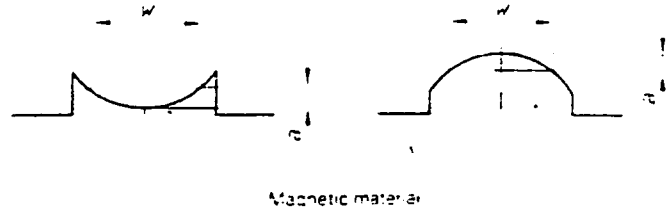
The signal amplitude obtained at this density, after recording with a test recording current of 500 % I_R, shall not exceed the signal amplitude obtained at the same density with a recording current of 350 % of I_R. The slope of the saturation curve shall never be positive between these two points.

When recorded with the same test recording currents (see 3.5) at 20 ftpmm (500 ftpi) and with all other parameters identical, the signal amplitude shall be not less than 70 % of that obtained at 8 ftpmm (200 ftpi) i.e.

$$\frac{\text{amplitude 20 ftpmm (500 ftpi)}}{\text{amplitude 8 ftpmm (200 ftpi)}} \times 100\ \% \geq 70\ \%$$

NOTE — The resolution of the read head sub-system (head amplifier) should be in the range of 90 % to 100 % when defined as

$$\frac{\text{amplitude 20 ftpmm (500 ftpi)}}{\text{amplitude 8 ftpmm ([illegible])}} \times 100\ \% = 9[illegible] \ldots 100\ \%$$



Magnetic material

$$a \leq 3.8 \left(\frac{w}{2.54}\right) \mu m \left[150 \left(\frac{w}{0.1}\right) \mu in\right]$$

where

w = minimum stripe width
= 6.35 mm (0.25 in) for tracks 1 and 2
= 10.28 mm (0.405 in) for tracks 1, 2 and 3

Figure 3 — Surface profile

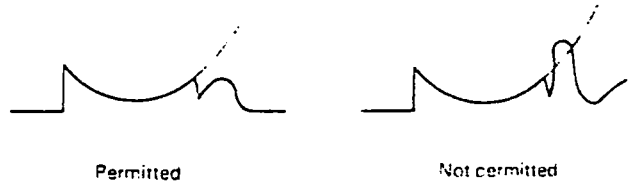


Figure 4 — Projected stripe surface

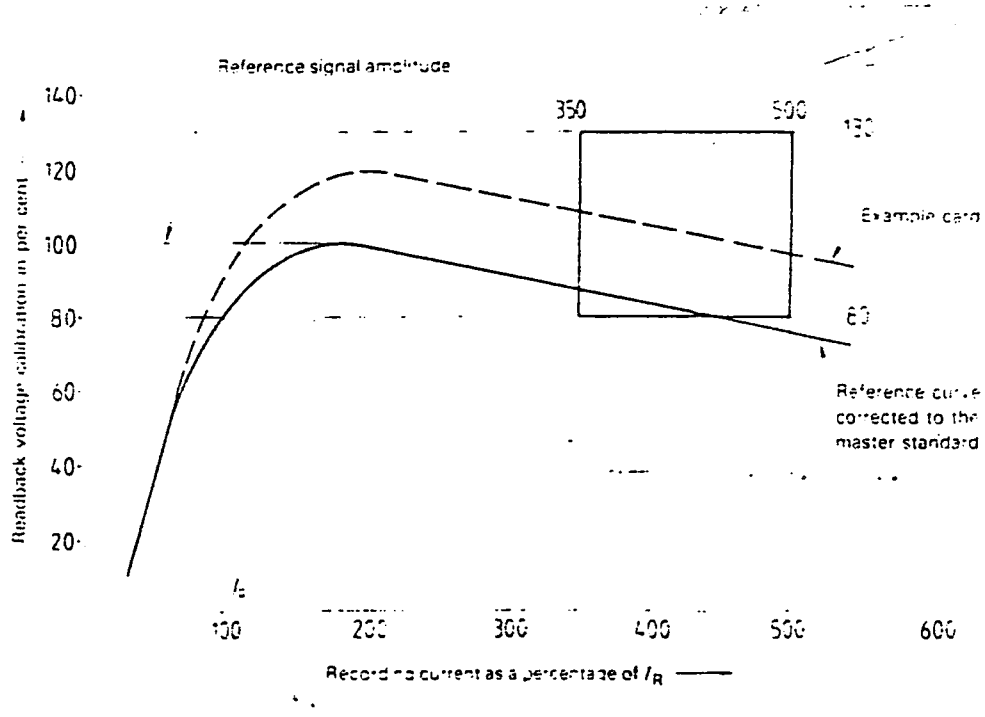


Figure 5 — Saturation curve of reference card and tolerance area at 8 ftpmm (200 ftpi)

NOTE — The corrected reference curve depicted above may not meet the specifications defined in 6.2. The curve defines the master signal response (on a card). The window parameters are defined to produce a card that will be functional in the machine readable environment.

6.2.2 Erasure

The magnetic material shall be capable of being erased by a DC write current equal to 350 % of I_R to a level of 3 % or less of the reference signal amplitude.

6.3 Testing and operating environment

The testing environment for signal amplitude measurements is 23 ± 3 °C (73 ± 5°F) and 40 % to 60 % relative humidity.

When tested under otherwise identical conditions, the signal amplitude from the magnetic stripe shall not deviate from its value in the above test environment by more than 15 % after 5 min of card exposure over the following operating environment range:

temperature: −35 to 50°C (−30 to 122 °F)

relative humidity: 5 % to 95 % with a maximum wet bulb temperature of 25 °C (77 °F)

6.4 Testing specifications

The read head used shall have a gap of 0,025 mm (0.001 in) or less.

When making the above measurements, the signal amplitude shall be measured after the encoding has stabilized. The stabilization criteria will be met if all measurements are taken under the same experimental conditions (i.e. taken after the same number of passages before magnetic head gap).

7 Encoding technique

The encoding technique is known as two-frequency coherent-phase recording. This method allows for serial recording of self-clocking data (on each track) (see figure 6).

The data comprise data and clocking bits together. A flux transition occurring between clocks signifies a "one", the absence of a flux transition signifies a "zero".

The data shall be recorded as a synchronous sequence of characters without intervening gaps.

Recording shall be in a saturation mode with magnetization parallel to a line in the plane of the track. The direction is determined by the recording angle.

8 Encoding specification, general

8.1 Angle of recording

The angle of recording shall be normal to the nearest edge of the card parallel to the magnetic stripe with the following tolerances:

Read-only track with 8,3 bpmm (210 bpi) (Track 1) ± 20

Read-only track with 3 bpmm (75 bpi) (Track 2) ± 20

Read-write track with 8,3 bpmm (210 bpi) (Track 3) ± 20

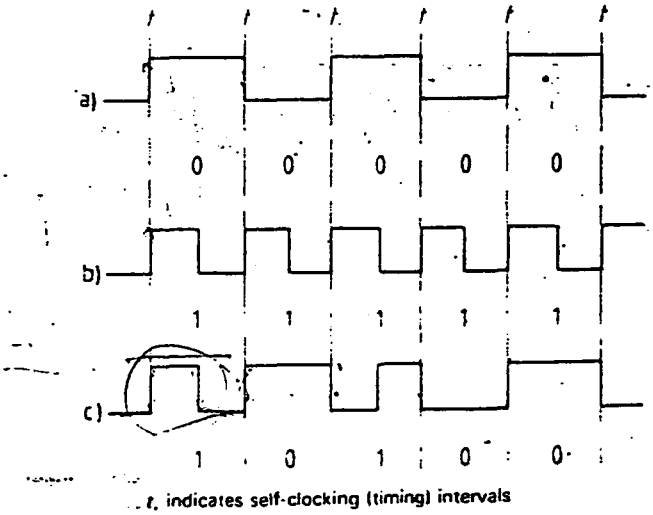
The angle of recording (α) is determined by measuring the angle of the head gap when the reading amplitude is maximum (see figure 7).

8.2 Bit configuration

In the bit configuration for each character on the magnetic area, the least significant bit (b_1) shall be encoded first and the parity bit last.

8.3 Direction of recording

The encoding shall begin from the right-hand side viewed from the side with the magnetic stripe and with the stripe at the top.



t, indicates self-clocking (timing) intervals

Figure 6 — Example of two-frequency coherent phase encoding

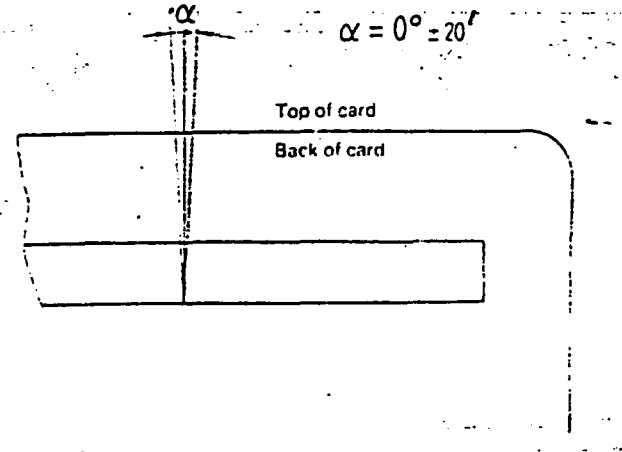


Figure 7 — Angle of recording

Clocking bits

The lead-in up to the first data bit shall be recorded with clocking bits (zeros) and the space from the last data bit to the end of the recording shall also be recorded with clocking bits (see the note).

NOTE — It is recognized that zeros prior to 2,9 mm (0,114 in) or after 82,55 mm (3,250 in) from the right edge of the card when viewed from the back may not meet the specifications given herein; however, only zeros should be encoded in this area.

9 Encoding specification for read-only tracks

In addition to the relevant part of clause 8 the following specifications apply for read-only tracks.

9.1 Alphanumeric track, Track 1

9.1.1 Bit density

The nominal bit density of the recorded signal shall be 8,3 bits per millimetre (210 bits per inch) measured along a line parallel to the longitudinal centreline of the track. The spacing between adjacent flux transitions shall be 0,121 ± 0,006 mm (4,762 ± 238 µin) ± 5 % for a "zero" and 0,06 ± 0,004 mm (2,381 ± 167 µin) ± 7 % for a "one". For a sequence of recorded "ones" the density corresponds to a nominal 16,5 ftpmm (420 ftpi).

9.1.2 Coded character set

The following character codes, which are alphanumeric and have 6 bit character sets with odd parity, shall be used:

Table 1 – Coded character set for track 1

				b_6	0	0	1	1
				b_5	0	1	0	1
b_4	b_3	b_2	b_1	Column / Row	0	1	2	3
0	0	0	0	0	SP	0	(a)	P
0	0	0	1	1	(a)	1	A	Q
0	0	1	0	2	(a)	2	B	R
0	0	1	1	3	(c)	3	C	S
0	1	0	0	4	$	4	D	T
0	1	0	1	5	% (d)	5	E	U
0	1	1	0	6	(a)	6	F	V
0	1	1	1	7	(a)	7	G	W
1	0	0	0	8	(	8	H	X
1	0	0	1	9	)	9	I	Y
1	0	1	0	10	(a)	(a)	J	Z
1	0	1	1	11	(a)	(a)	K	(b)
1	1	0	0	12	(a)	(a)	L	(b)
1	1	0	1	13	–	(a)	M	(b)
1	1	1	0	14	.	(a)	N	(d)
1	1	1	1	15	/	? (d)	O	(a)

(a) These character positions are available for hardware control purposes only and cannot contain information characters.

(b) These character positions are reserved for additional national characters when required. They are not to be used internationally

(c) This character position is reserved for optional additional graphic symbols

(d) These characters shall have the following meanings for this application:

Position 0/5 % represents "start sentinel".

1/15 ? represents "end sentinel"

3/14 ^ represents "separator"

9.1.3 Maximum number of characters for ID-1 type card

The data characters, the control characters and the longitudinal redundancy check character shall together not exceed 79 characters including start and end sentinels.

Numeric track, Track 2

Bit density

The nominal bit density of the recorded signal shall be 3 bits per millimetre (75 bits per inch) ± 3 % when measured along the line parallel to the longitudinal centreline of the track. The spacing between adjacent flux transitions shall be 0,339 ± 0,010 mm (13 333 ± 400 µin) i.e. ± 3 % for a "zero" and 0,169 ± 0,007 mm (6 667 ± 267 µin) i.e. ± 4 % for a "one". For a sequence of recorded "ones" the density corresponds to a nominal 6,0 ftpmm (150 ftpi).

9.2.2 Coded character set

The character code, which is numeric only, shall be a BCD 4 bit code with odd parity (P) as shown in table 2.

Table 2 — Coded character set for track 2 and 3

Bits					Row	Character
P	b_4	b_3	b_2	b_1		
1	0	0	0	0	0	0
0	0	0	0	1	1	1
0	0	0	1	0	2	2
1	0	0	1	1	3	3
0	0	1	0	0	4	4
1	0	1	0	1	5	5
1	0	1	1	0	6	6
0	0	1	1	1	7	7
0	1	0	0	0	8	8
1	1	0	0	1	9	9
1	1	0	1	0	10	(a)
0	1	0	1	1	11	(b1) Start
1	1	1	0	0	12	(a)
0	1	1	0	1	13	(b2) Sep
0	1	1	1	0	14	(a)
1	1	1	1	1	15	F (b3) End

(a) These character positions are available for hardware control purposes only and cannot contain information characters (data content).

(b1) Start sentinel (start character)

(b2) Separator

(b3) End sentinel (stop character)

9.2.3 Maximum number of characters for ID-1 type card

The data characters, the control characters and the longitudinal redundancy check character shall together not exceed 40 characters, including start and end sentinels.

10 Encoding specifications for read-write track, track 3

In addition to the relevant parts of clause 8, the following specifications apply to read-write track, track 3.

10.1 Bit density

The nominal bit density of the recorded signal shall be 8 bits per millimetre (210 bits per inch) ± 8 % when measured along the line parallel to the longitudinal centreline of the track. The spacing between adjacent flux transitions shall be 0,121 ± 0,010 mm (4 762 ± 381 µin) i.e. ± 8 % for a "zero" and 0,060 ± 0,006 mm (2 381 ± 238 µin) i.e. ± 10 % for a "one". For a sequence of recorded "ones" the density corresponds to a nominal 16,5 ftpmm (420 ftpi).

10.2 Coded character set

The numeric coded character set in 9.2.2 shall be used.

10.3 Maximum number of characters for ID-1-type card

10.3.1 ID-1-type card

The data characters, the control characters and the longitudinal redundancy check character shall together not exceed 107 characters, including start and end sentinels.

11 Error detection

Two techniques of error detection, as described below, shall be encoded. In both techniques, the clocking bits recorded are used for synchronization and shall not be regarded as data characters.

11.1 Parity

A parity bit for each encoded character shall be used. The value of the parity bit is defined such that the total quantity of one bits recorded, for a character, including the parity bit, shall be odd.

11.2 Longitudinal redundancy check (LRC)

A longitudinal redundancy check (LRC) character shall appear for each data message. The LRC character shall be encoded so that it immediately follows the end sentinel when the card is read in a direction giving the start sentinel first, followed by data and the end sentinel. The bit configuration of the LRC character shall be identical to the bit configuration of the data characters.

The LRC character shall be calculated using the following procedure:

The value of each bit in the LRC character, excluding the parity bit, is defined such that the total count of one bits encoded in the corresponding bit location of all characters of the data message, including the start sentinel, data, end sentinel, and LRC characters, shall be even.

The LRC characters parity bit is not a parity bit for the individual parity bits of the data message, but is only the parity bit for the LRC character encoded as described in 11.1.

National Semiconductor

LM124/LM224/LM324, LM124A/LM224A/LM324A, LM2902 Low Power Quad Operational Amplifiers

General Description

The LM124 series consists of four independent, high gain, internally frequency compensated operational amplifiers which were designed specifically to operate from a single power supply over a wide range of voltages. Operation from split power supplies is also possible and the low power supply current drain is independent of the magnitude of the power supply voltage.

Application areas include transducer amplifiers, DC gain blocks and all the conventional op amp circuits which now can be more easily implemented in single power supply systems. For example, the LM124 series can be directly operated off of the standard +5 V_{DC} power supply voltage which is used in digital systems and will easily provide the required interface electronics without requiring the additional ±15 V_{DC} power supplies.

Unique Characteristics

- In the linear mode the input common-mode voltage range includes ground and the output voltage can also swing to ground, even though operated from only a single power supply voltage.
- The unity gain cross frequency is temperature compensated.
- The input bias current is also temperature compensated.

Advantages

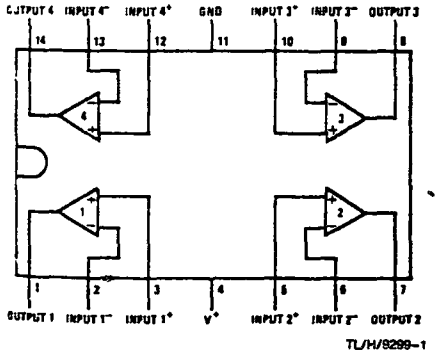
- Eliminates need for dual supplies
- Four internally compensated op amps in a single package
- Allows directly sensing near GND and V_{OUT} also goes to GND
- Compatible with all forms of logic
- Power drain suitable for battery operation

Features

- Internally frequency compensated for unity gain
- Large DC voltage gain 100 dB
- Wide bandwidth (unity gain) 1 MHz (temperature compensated)
- Wide power supply range:
  - Single supply 3 V_{DC} to 32 V_{DC}
  - or dual supplies ±1.5 V_{DC} to ±16 V_{DC}
- Very low supply current drain (700 μA)—essentially independent of supply voltage
- Low input biasing current 45 nA_{DC} (temperature compensated)
- Low input offset voltage 2 mV_{DC}
  and offset current 5 nA_{DC}
- Input common-mode voltage range includes ground
- Differential input voltage range equal to the power supply voltage
- Large output voltage swing 0 V_{DC} to V^+ − 1.5 V_{DC}

Connection Diagram

Dual-In-Line Package

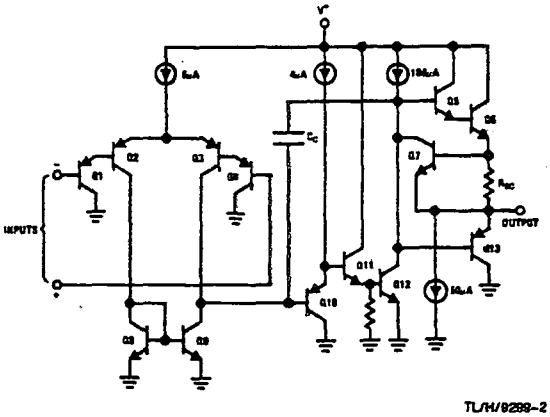


TL/H/9299–1

Top View

Order Number LM124J, LM124AJ, LM224J, LM224AJ, LM324J, LM324AJ, LM324M, LM324AM, LM2902M, LM324N, LM324AN or LM2902N
See NS Package Number J14A, M14A or N14A

Schematic Diagram (Each Amplifier)



TL/H/9299–2

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.
(Note 9)

	LM124/LM224/LM324 LM124A/LM224A/LM324A	LM2902
Supply Voltage, V^+	32 V_{DC} or ±16 V_{DC}	26 V_{DC} or ±13 V_{DC}
Differential Input Voltage	32 V_{DC}	26 V_{DC}
Input Voltage	−0.3 V_{DC} to +32 V_{DC}	−0.3 V_{DC} to +26 V_{DC}
Input Current (V_{IN} < −0.3 V_{DC}) (Note 3)	50 mA	50 mA
Power Dissipation (Note 1)		
Molded DIP	1130 mW	1130 mW
Cavity DIP	1260 mW	1260 mW
Small Outline Package	800 mW	800 mW
Output Short-Circuit to GND (One Amplifier) (Note 2) $V^+ \leq 15\ V_{DC}$ and $T_A = 25°C$	Continuous	Continuous
Operating Temperature Range		−40°C to +85°C
LM324/LM324A	0°C to +70°C	
LM224/LM224A	−25°C to +85°C	
LM124/LM124A	−55°C to +125°C	

	LM124/LM224/LM324 LM124A/LM224A/LM324A	LM2902
Storage Temperature Range	−65°C to +150°C	−65°C to +150°C
Lead Temperature (Soldering, 10 seconds)	260°C	260°C
Soldering Information		
Dual-In-Line Package		
Soldering (10 seconds)	260°C	260°C
Small Outline Package		
Vapor Phase (60 seconds)	215°C	215°C
Infrared (15 seconds)	220°C	220°C

See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" for other methods of soldering surface mount devices.

ESD Tolerance (Note 10)	250V	250V

Electrical Characteristics $V^+ = +5.0\ V_{DC}$, (Note 4), unless otherwise stated

Parameter	Conditions	LM124A		LM224A		LM324A		LM124/LM224		LM324		LM2902		Units
		Min	Typ Max	Min	Typ Max	Min	Typ Max	Min	Typ Max	Min	Typ Max	Min	Typ Max	
Input Offset Voltage	(Note 5) $T_A = 25°C$		±1 ±2		±1 ±3		±2 ±3		±2 ±5		±2 ±7		±2 ±7	mV_{DC}
Input Bias Current (Note 6)	$I_{IN(+)}$ or $I_{IN(-)}$, $V_{CM} = 0V$, $T_A = 25°C$		20 50		40 80		45 100		45 150		45 250		45 250	nA_{DC}
Input Offset Current	$I_{IN(+)} - I_{IN(-)}$, $V_{CM} = 0V$, $T_A = 25°C$		±2 ±10		±2 ±15		±5 ±30		±3 ±30		±5 ±50		±5 ±50	nA_{DC}
Input Common-Mode Voltage Range (Note 7)	$V^+ = 30\ V_{DC}$, (LM2902, $V^+ = 26\ V_{DC}$), $T_A = 25°C$	0	$V^+ - 1.5$	0	$V^+ - 1.5$	0	$V^+ - 1.5$	0	$V^+ - 1.5$	0	$V^+ - 1.5$	0	$V^+ - 1.5$	V_{DC}
Supply Current	Over Full Temperature Range $R_L = \infty$ On All Op Amps $V^+ = 30V$ (LM2902 $V^+ = 26V$) $V^+ = 5V$		1.5 3 0.7 1.2		1.5 3 0.7 1.2		1.5 3 0.7 1.2		1.5 3 0.7 1.2		1.5 3 0.7 1.2		1.5 3 0.7 1.2	mA_{DC}
Large Signal Voltage Gain	$V^+ = 15\ V_{DC}$, $R_L \geq 2\ k\Omega$, ($V_O = 1\ V_{DC}$ to 11 V_{DC}), $T_A = 25°C$	50	100	50	100	25	100	50	100	25	100	25	100	V/mV
Common-Mode Rejection Ratio	DC, $V_{CM} = 0V$ to $V^+ - 1.5\ V_{DC}$, $T_A = 25°C$	70	85	70	85	65	85	70	85	65	85	50	70	dB
Power Supply Rejection Ratio	DC, $V^+ = 5\ V_{DC}$ to 30 V_{DC}	65	100	65	100	65	100	65	100	65	100	50	100	dB

Parameter	Conditions	LM124A Min	LM124A Typ	LM124A Max	LM224A Min	LM224A Typ	LM224A Max	LM324A Min	LM324A Typ	LM324A Max	LM124/LM224 Min	LM124/LM224 Typ	LM124/LM224 Max	LM324 Min	LM324 Typ	LM324 Max	LM2902 Min	LM2902 Typ	LM2902 Max	Units
Power Supply Rejection Ratio	DC, $V^+ = 5\ V_{DC}$ to $30\ V_{DC}$ (LM2902, $V^+ = 5\ V_{DC}$ to $26\ V_{DC}$), $T_A = 25°C$	65	100		65	100		65	100		65	100		65	100		50	100		dB

Electrical Characteristics $V^+ = +5.0\ V_{DC}$, (Note 4) unless otherwise stated (Continued)

Parameter	Conditions	LM124A Min	LM124A Typ	LM124A Max	LM224A Min	LM224A Typ	LM224A Max	LM324A Min	LM324A Typ	LM324A Max	LM124/LM224 Min	LM124/LM224 Typ	LM124/LM224 Max	LM324 Min	LM324 Typ	LM324 Max	LM2902 Min	LM2902 Typ	LM2902 Max	Units
Amplifier-to-Amplifier Coupling (Note 8)	f = 1 kHz to 20 kHz, $T_A = 25°C$ (Input Referred)		−120			−120			−120			−120			−120			−120		dB
Output Current — Source	$V_{IN}^+ = 1\ V_{DC}$, $V_{IN}^- = 0\ V_{DC}$, $V^+ = 15\ V_{DC}$, $V_O = 2\ V_{DC}$, $T_A = 25°C$	20	40		20	40		20	40		20	40		20	40		20	40		mA_{DC}
Output Current — Sink	$V_{IN}^- = 1\ V_{DC}$, $V_{IN}^+ = 0\ V_{DC}$, $V^+ = 15\ V_{DC}$, $V_O = 2\ V_{DC}$, $T_A = 25°C$	10	20		10	20		10	20		10	20		10	20		10	20		mA_{DC}
Output Current — Sink	$V_{IN}^- = 1\ V_{DC}$, $V_{IN}^+ = 0\ V_{DC}$, $V^+ = 15\ V_{DC}$, $V_O = 200\ mV_{DC}$, $T_A = 25°C$	12	50		12	50		12	50		12	50		12	50		12	50		μA_{DC}
Short Circuit to Ground	(Note 2) $V^+ = 15\ V_{DC}$, $T_A = 25°C$		40	60		40	60		40	60		40	60		40	60		40	60	mA_{DC}
Input Offset Voltage	(Note 5)			±4			±4			±5			±7			±9			±10	mV_{DC}
Input Offset Voltage Drift	$R_S = 0\Omega$		±7	±20		±7	±20		±7	±30		±7			±7			±7		μV/°C
Input Offset Current	$I_{IN(+)} - I_{IN(-)}$, $V_{CM} = 0V$			±30			±30			±75			±100			±150		±45	±200	nA_{DC}
Input Offset Current Drift	$R_S = 0\Omega$		±10	±200		±10	±200		±10	±200		±10			±10			±10		pA_{DC}/°C
Input Bias Current	$I_{IN(+)}$ or $I_{IN(-)}$		40	100		40	100		40	200		40	300		40	500		40	500	nA_{DC}
Input Common-Mode Voltage Range (Note 7)	$V^+ = +30\ V_{DC}$ (LM2902, $V^+ = 26\ V_{DC}$)	0		V^+-2	0		V^+-2	0		V^+-2	0		V^+-2	0		V^+-2	0		V^+-2	V_{DC}
Large Signal Voltage Gain	$V^+ = +15\ V_{DC}$ (V_O Swing = $1\ V_{DC}$ to $11\ V_{DC}$) $R_L \geq 2\ k\Omega$	25			25			15			25			15			15			V/mV
Output Voltage Swing — V_{OH}	$V^+ = +30\ V_{DC}$, $R_L = 2\ k\Omega$	26			26			26			26			26			22			V_{DC}
Output Voltage Swing — V_{OH}	$R_L \geq 10\ k\Omega$ (LM2902, $V^+ = 26\ V_{DC}$)	27	28		27	28		27	28		27	28		27	28		23	24		V_{DC}
Output Voltage Swing — V_{OL}	$V^+ = 5\ V_{DC}$, $R_L \geq 10\ k\Omega$		5	20		5	20		5	20		5	20		5	20		5	100	mV_{DC}

Electrical Characteristics $V^+ = +5.0\ V_{DC}$ (Note 4) unless otherwise stated (Continued)

Parameter		Conditions		LM124A			LM224A			LM324A			LM124/LM224			LM324			LM2902			Units
				Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	Min	Typ	Max	
Output Current	Source	$V_O = 2\ V_{DC}$	$V_{IN}^+ = +1\ V_{DC}$, $V_{IN}^- = 0\ V_{DC}$, $V^+ = 15\ V_{DC}$	10	20		10	20		10	20		10	20		10	20		10	20		mA_{DC}
	Sink		$V_{IN}^- = +1\ V_{DC}$, $V_{IN}^+ = 0\ V_{DC}$, $V^+ = 15\ V_{DC}$	10	15		5	8		5	8		5	8		5	8		5	8		

Note 1: For operating at high temperatures, the LM324/LM324A, LM2902 must be derated based on a +125°C maximum junction temperature and a thermal resistance of 88°C/W which applies for the device soldered in a printed circuit board, operating in a still air ambient. The LM224/LM224A and LM124/LM124A can be derated based on a +150°C maximum junction temperature. The dissipation is the total of all four amplifiers—use external resistors, where possible, to allow the amplifier to saturate of to reduce the power which is dissipated in the integrated circuit.

Note 2: Short circuits from the output to V^+ can cause excessive heating and eventual destruction. When considering short circuits to ground, the maximum output current is approximately 40 mA independent of the magnitude of V^+. At values of supply voltage in excess of $+15\ V_{DC}$, continuous short-circuits can exceed the power dissipation ratings and cause eventual destruction. Destructive dissipation can result from simultaneous shorts on all amplifiers.

Note 3: This input current will only exist when the voltage at any of the input leads is driven negative. It is due to the collector-base junction of the input PNP transistors becoming forward biased and thereby acting as input diode clamps. In addition to this diode action, there is also lateral NPN parasitic transistor action on the IC chip. This transistor action can cause the output voltages of the op amps to go to the V^+ voltage level (or to ground for a large overdrive) for the time duration that an input is driven negative. This is not destructive and normal output states will re-establish when the input voltage, which was negative, again returns to a value greater than $-0.3\ V_{DC}$ (at 25°C).

Note 4: These specifications are limited to $-55°C \le T_A \le +125°C$ for the LM124/LM124A. With the LM224/LM224A, all temperature specifications are limited to $-25°C \le T_A \le +85°C$, the LM324/LM324A temperature specifications are limited to $0°C \le T_A \le +70°C$, and the LM2902 specifications are limited to $-40°C \le T_A \le +85°C$.

Note 5: $V_O \approx 1.4\ V_{DC}$, $R_S = 0\Omega$ with V^+ from $5\ V_{DC}$ to $30\ V_{DC}$; and over the full input common-mode range ($0\ V_{DC}$ to $V^+ - 1.5\ V_{DC}$) at 25°C; for LM2902, V^+ from $5\ V_{DC}$ to $26\ V_{DC}$.

Note 6: The direction of the input current is out of the IC due to the PNP input stage. This current is essentially constant, independent of the state of the output so no loading change exists on the input lines.

Note 7: The input common-mode voltage of either input signal voltage should not be allowed to go negative by more than 0.3V (at 25°C). The upper end of the common-mode voltage range is $V^+ - 1.5V$ (at 25°C), but either or both inputs can go to $+32\ V_{DC}$ without damage ($+26\ V_{DC}$ for LM2902), independent of the magnitude of V^+.

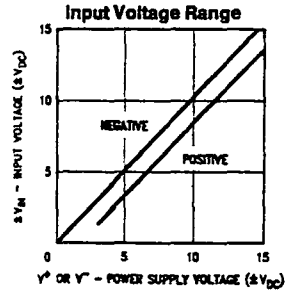
Note 8: Due to proximity of external components, insure that coupling is not originating via stray capacitance between these external parts. This typically can be detected as this type of capacitance increases at higher frequencies.

Note 9: Refer to RETS124AX for LM124A military specifications and refer to RETS124X for LM124 military specifications.

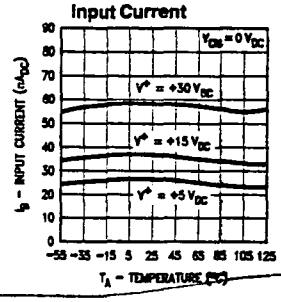
Note 10: Human body model, 1.5 kΩ in series with 100 pF.

Typical Performance Characteristics

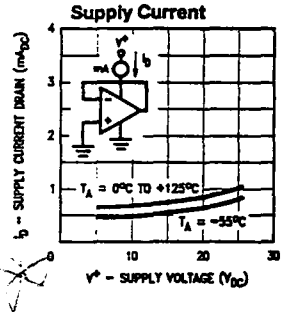
Input Voltage Range



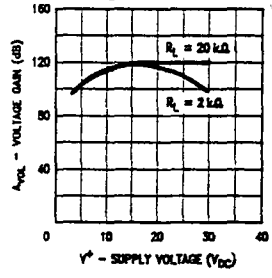
Input Current



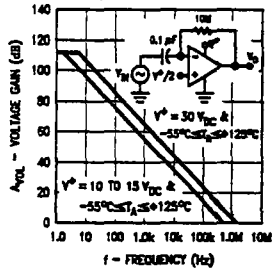
Supply Current



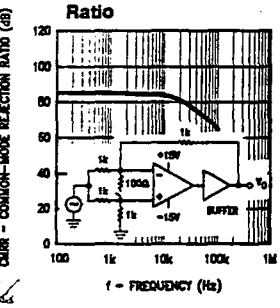
Voltage Gain



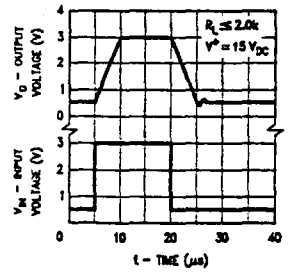
Open Loop Frequency Response



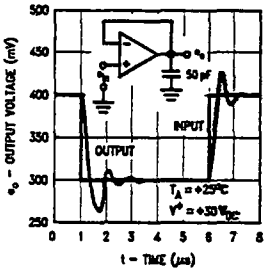
Common Mode Rejection Ratio



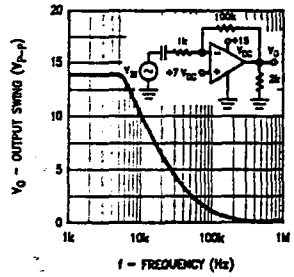
Voltage Follower Pulse Response



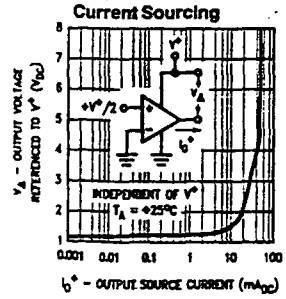
Voltage Follower Pulse Response (Small Signal)



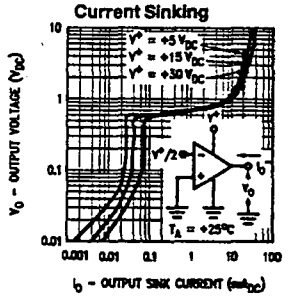
Large Signal Frequency Response



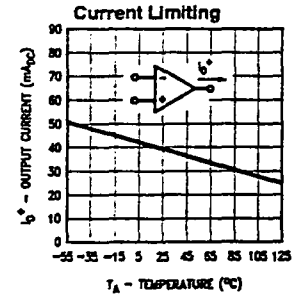
Output Characteristics Current Sourcing



Output Characteristics Current Sinking

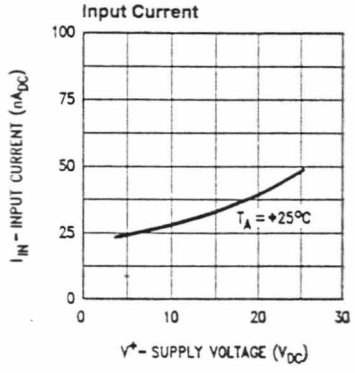


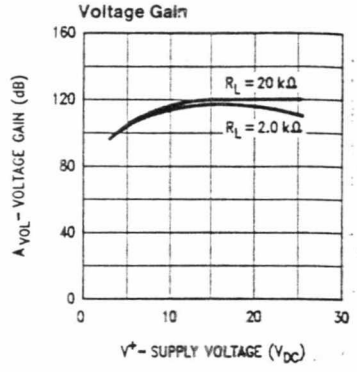
Current Limiting



TL/H/9299–3

Typical Performance Characteristics (LM2902 only)





TL/H/7894-4

Application Hints

The LM124 series are op amps which operate with only a single power supply voltage, have true-differential inputs, and remain in the linear mode with an input common-mode voltage of 0 V_{DC}. These amplifiers operate over a wide range of power supply voltage with little change in performance characteristics. At 25°C amplifier operation is possible down to a minimum supply voltage of 2.3 V_{DC}.

The pinouts of the package have been designed to simplify PC board layouts. Inverting inputs are adjacent to outputs for all of the amplifiers and the outputs have also been placed at the corners of the package (pins 1, 7, 8, and 14).

Precautions should be taken to insure that the power supply for the integrated circuit never becomes reversed in polarity or that the unit is not inadvertently installed backwards in a test socket as an unlimited current surge through the resulting forward diode within the IC could cause fusing of the internal conductors and result in a destroyed unit.

Large differential input voltages can be easily accommodated and, as input differential voltage protection diodes are not needed, no large input currents result from large differential input voltages. The differential input voltage may be larger than V^+ without damaging the device. Protection should be provided to prevent the input voltages from going negative more than -0.3 V_{DC} (at 25°C). An input clamp diode with a resistor to the IC input terminal can be used.

To reduce the power supply drain, the amplifiers have a class A output stage for small signal levels which converts to class B in a large signal mode. This allows the amplifiers to both source and sink large output currents. Therefore both NPN and PNP external current boost transistors can be used to extend the power capability of the basic amplifiers. The output voltage needs to raise approximately 1 diode drop above ground to bias the on-chip vertical PNP transistor for output current sinking applications.

For ac applications, where the load is capacitively coupled to the output of the amplifier, a resistor should be used, from the output of the amplifier to ground to increase the class A bias current and prevent crossover distortion.

Where the load is directly coupled, as in dc applications, there is no crossover distortion.

Capacitive loads which are applied directly to the output of the amplifier reduce the loop stability margin. Values of 50 pF can be accommodated using the worst-case noninverting unity gain connection. Large closed loop gains or resistive isolation should be used if larger load capacitance must be driven by the amplifier.

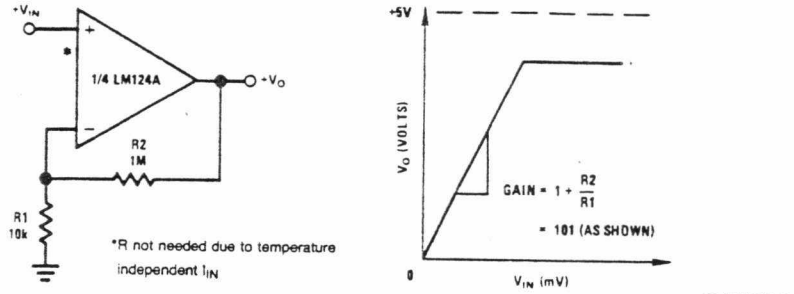
The bias network of the LM124 establishes a drain current which is independent of the magnitude of the power supply voltage over the range of from 3 V_{DC} to 30 V_{DC}.

Output short circuits either to ground or to the positive power supply should be of short time duration. Units can be destroyed, not as a result of the short circuit current causing metal fusing, but rather due to the large increase in IC chip dissipation which will cause eventual failure due to excessive junction temperatures. Putting direct short-circuits on more than one amplifier at a time will increase the total IC power dissipation to destructive levels, if not properly protected with external dissipation limiting resistors in series with the output leads of the amplifiers. The larger value of output source current which is available at 25°C provides a larger output current capability at elevated temperatures (see typical performance characteristics) than a standard IC op amp.

The circuits presented in the section on typical applications emphasize operation on only a single power supply voltage. If complementary power supplies are available, all of the standard op amp circuits can be used. In general, introducing a pseudo-ground (a bias voltage reference of $V^+/2$) will allow operation above and below this value in single power supply systems. Many application circuits are shown which take advantage of the wide input common-mode voltage range which includes ground. In most cases, input biasing is not required and input voltages which range to ground can easily be accommodated.

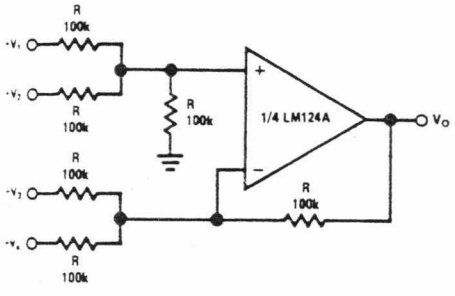
Typical Single-Supply Applications ($V^+ = 5.0\ V_{DC}$)

Non-Inverting DC Gain (0V Input = 0V Output)



TL/H/9299–5

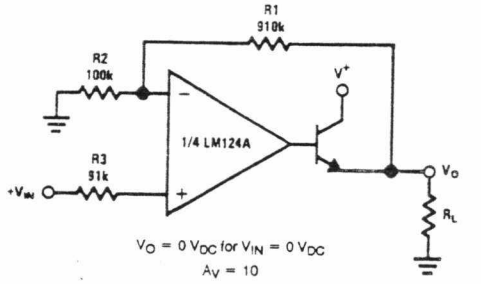
DC Summing Amplifier (V_{IN}'S ≥ 0 V_{DC} and V_O ≥ V_{DC})



TL/H/9299–6

Where: $V_O = V_1 + V_2 - V_3 - V_4$
$(V_1 + V_2) \geq (V_3 + V_4)$ to keep $V_O > 0\ V_{DC}$

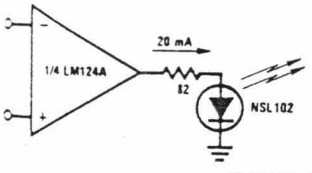
Power Amplifier



$V_O = 0\ V_{DC}$ for $V_{IN} = 0\ V_{DC}$
$A_V = 10$

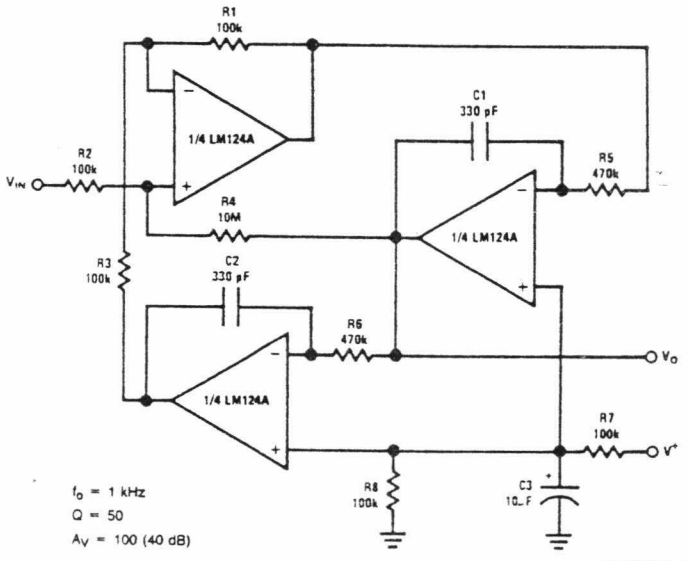
TL/H/9299–7

LED Driver



TL/H/9299–8

"Bi-QUAD" RC Active Bandpass Filter

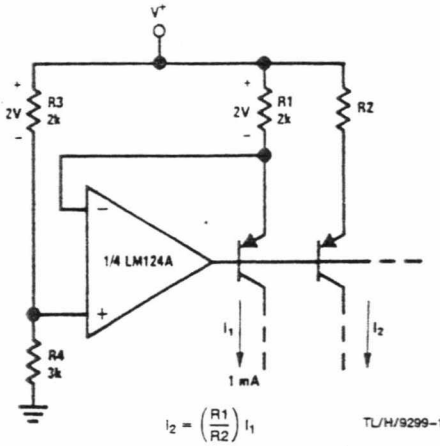


$f_o = 1$ kHz
$Q = 50$
$A_V = 100$ (40 dB)

TL/H/9299–9

Typical Single-Supply Applications (V^+ = 5.0 V_{DC}) (Continued)

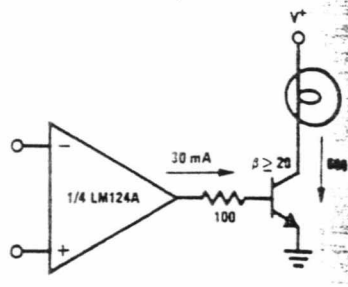
Fixed Current Sources



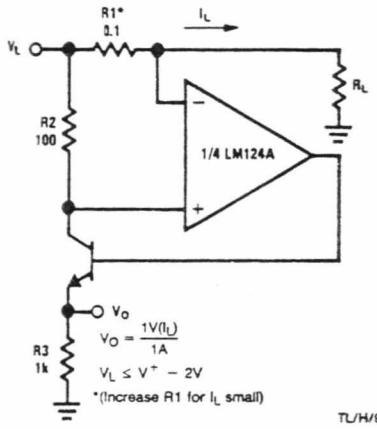
$$I_2 = \left(\frac{R1}{R2}\right) I_1$$

TL/H/9299–10

Lamp Driver

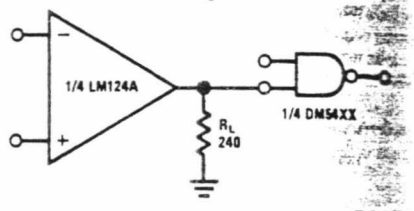


Current Monitor

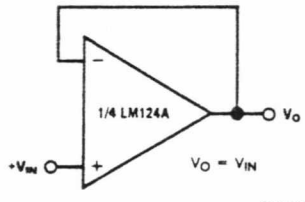


TL/H/9299–12

Driving TTL

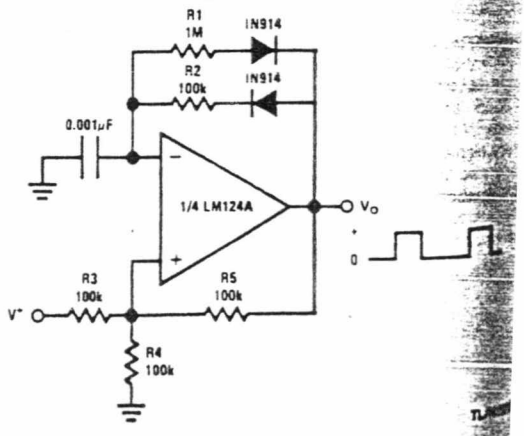


Voltage Follower



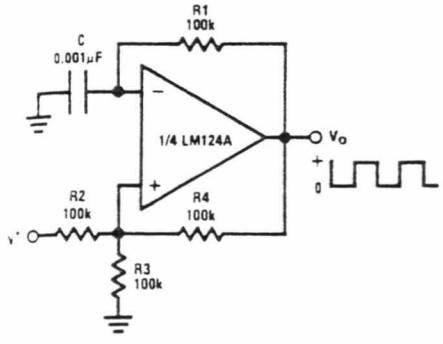
TL/H/9299–14

Pulse Generator



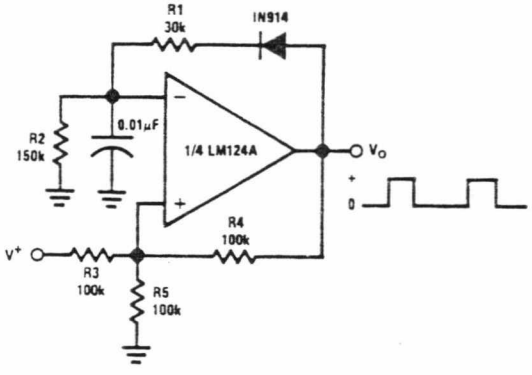
Typical Single-Supply Applications (V^+ = 5.0 V_{DC}) (Continued)

Squarewave Oscillator



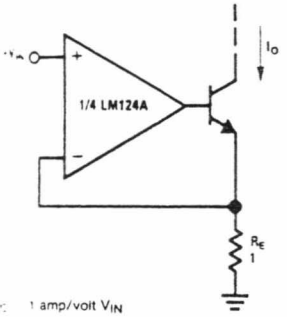
TL/H/9299–16

Pulse Generator



TL/H/9299–17

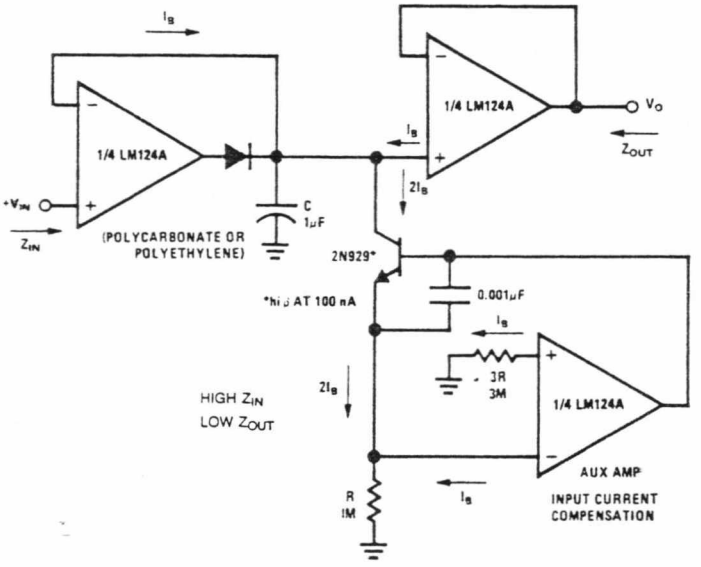
High Compliance Current Sink



1 amp/volt V_{IN}
ncrease R_E for I_O small)

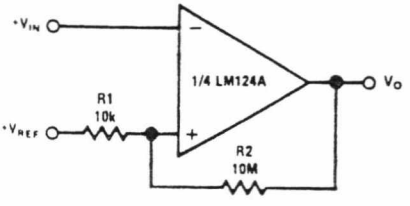
TL/H/9299–18

Low Drift Peak Detector



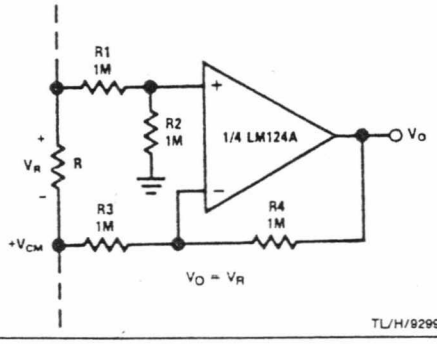
TL/H/9299–19

Comparator with Hysteresis



TL/H/9299–20

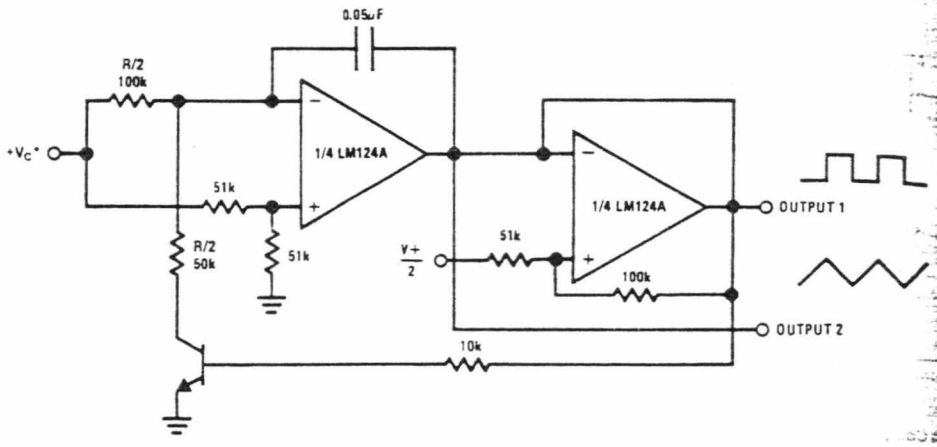
Ground Referencing a Differential Input Signal



TL/H/9299–21

Typical Single-Supply Applications (V^+ = 5.0 V_{DC}) (Continued)

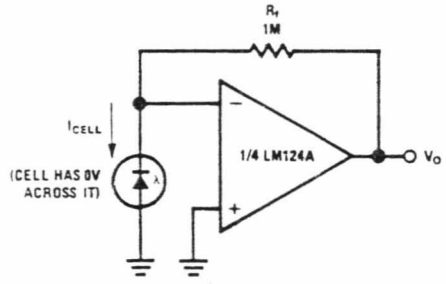
Voltage Controlled Oscillator Circuit



TL/H/9299-22

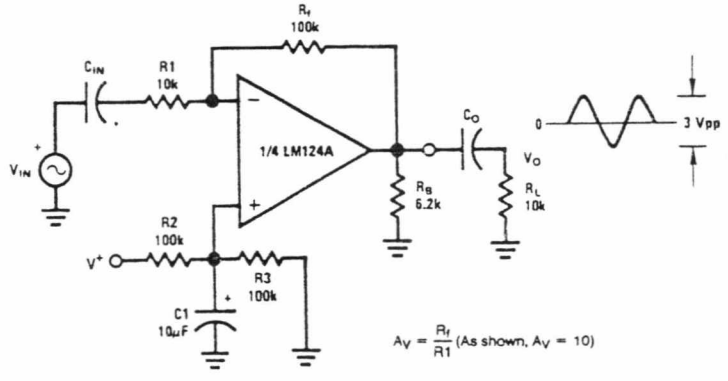
*Wide control voltage range: 0 $V_{DC} \leq V_C \leq 2\,(V^+ - 1.5\,V_{DC})$

Photo Voltaic-Cell Amplifier



TL/H/9299-23

AC Coupled Inverting Amplifier

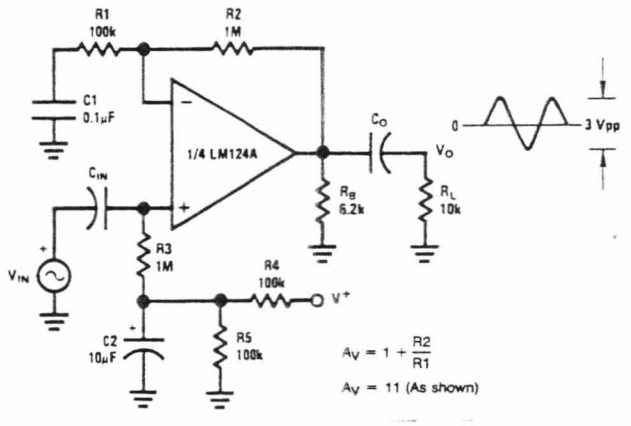


$A_V = \frac{R_f}{R1}$ (As shown, A_V = 10)

TL/H/9299-24

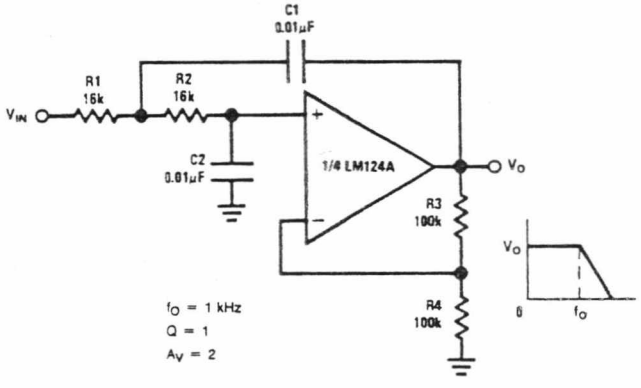
Typical Single-Supply Applications ($V^+ = 5.0\ V_{DC}$) (Continued)

AC Coupled Non-Inverting Amplifier



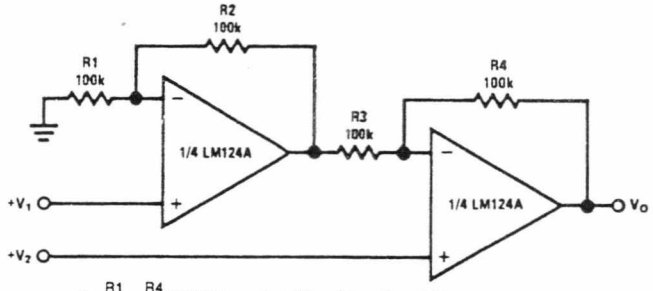
TL/H/9299–25

DC Coupled Low-Pass RC Active Filter



TL/H/9299–26

High Input Z, DC Differential Amplifier



TL/H/9299–27

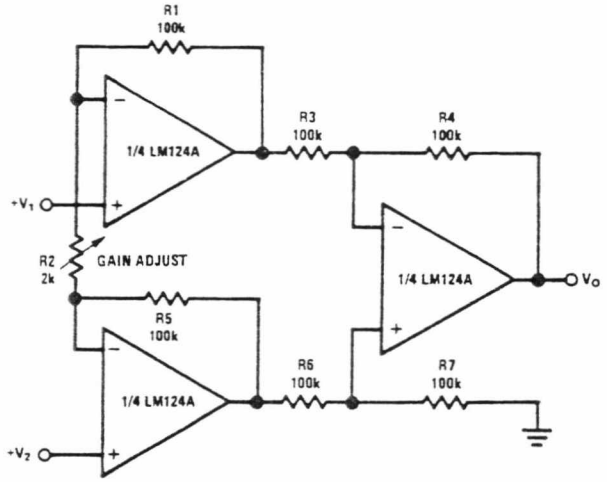
For $\frac{R1}{R2} = \frac{R4}{R3}$ (CMRR depends on this resistor ratio match)

$V_O = 1 + \frac{R4}{R3}(V_2 - V_1)$

As shown: $V_O = 2(V_2 - V_1)$

Typical Single-Supply Applications (V^+ = 5.0 V_{DC}) (Continued)

High Input Z Adjustable-Gain DC Instrumentation Amplifier



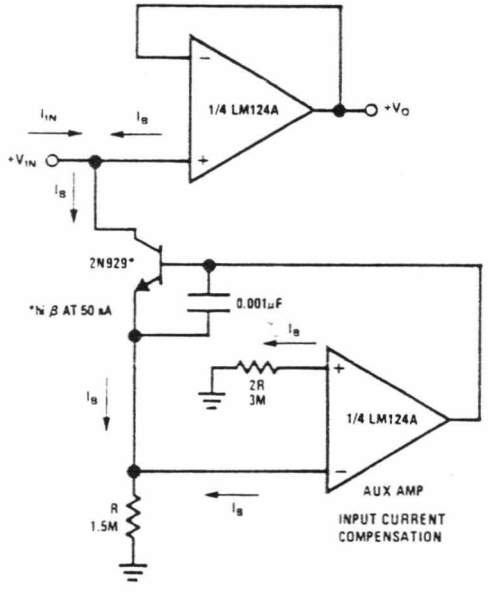
TL/H/9299–28

If R1 = R5 & R3 = R4 = R6 = R7 (CMRR depends on match)

$$V_O = 1 + \frac{2R1}{R2}(V_2 - V_1)$$

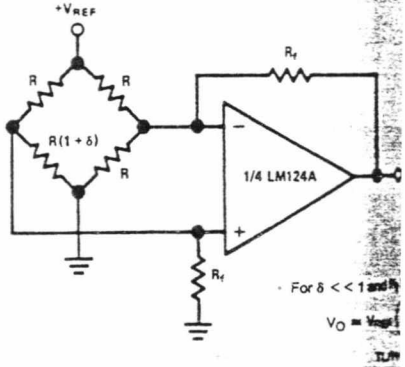
As shown $V_O = 101\,(V_2 - V_1)$

Using Symmetrical Amplifiers to Reduce Input Current (General Concept)



TL/H/9299–29

Bridge Current Amplifier

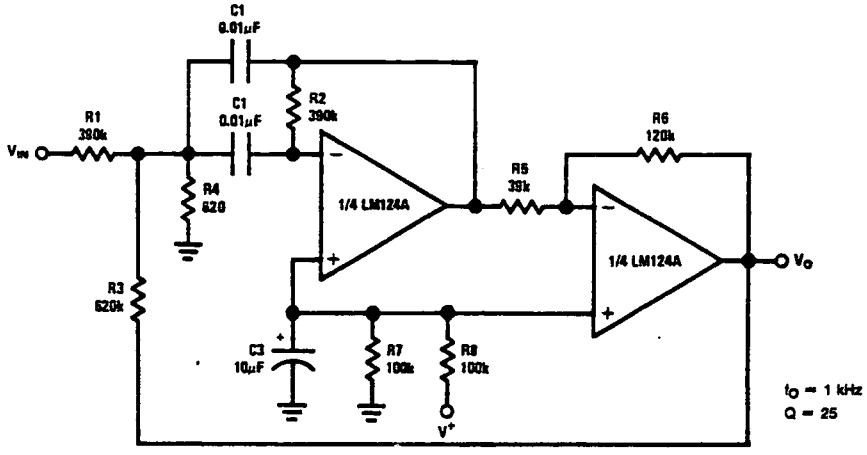


For $\delta << 1$ and

V_O =

Typical Single-Supply Applications (V^+ = 5.0 V_{DC}) (Continued)

Bandpass Active Filter



TL/H/9298–31

National Semiconductor

LM386 Low Voltage Audio Power Amplifier

General Description

The LM386 is a power amplifier designed for use in low voltage consumer applications. The gain is internally set to 20 to keep external part count low, but the addition of an external resistor and capacitor between pins 1 and 8 will increase the gain to any value up to 200.

The inputs are ground referenced while the output is automatically biased to one half the supply voltage. The quiescent power drain is only 24 milliwatts when operating from a 6 volt supply, making the LM386 ideal for battery operation.

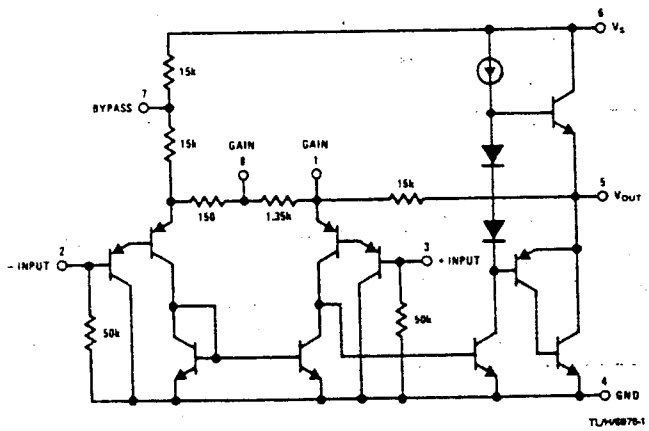
Features

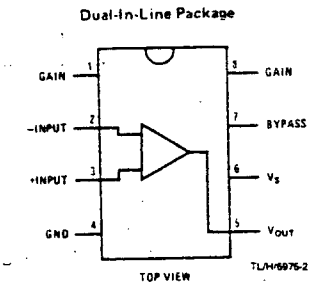
- Battery operation
- Minimum external parts
- Wide supply voltage range — 4V–12V or 5V–18V
- Low quiescent current drain — 4 mA
- Voltage gains from 20 to 200
- Ground referenced input
- Self-centering output quiescent voltage
- Low distortion
- Eight pin dual-in-line package

Applications

- AM-FM radio amplifiers
- Portable tape player amplifiers
- Intercoms
- TV sound systems
- Line drivers
- Ultrasonic drivers
- Small servo drivers
- Power converters

Equivalent Schematic and Connection Diagrams

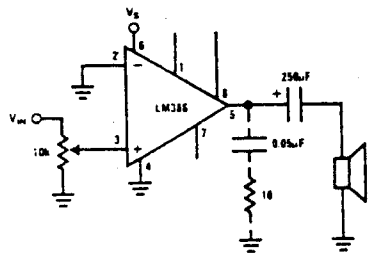




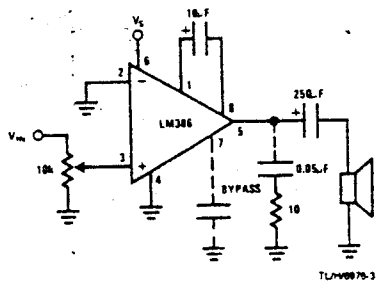
Order Number LM386N-1, LM386N-3 or LM386N-4
See NS Package N08E

Typical Applications

Amplifier with Gain = 20
Minimum Parts



Amplifier with Gain = 200



Absolute Maximum Ratings

Supply Voltage (LM386N)	15V	Storage Temperature	−65°C to +150°C
Supply Voltage (LM386N-4)	22V	Operating Temperature	0°C to +70°C
Package Dissipation (Note 1) (LM386N-4)	1.25W	Junction Temperature	+150°C
Package Dissipation (Note 2) (LM386)	660 mW	Lead Temperature (Soldering, 10 seconds)	+300°C
Input Voltage	±0.4V		

Electrical Characteristics $T_A = 25°C$

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
Operating Supply Voltage (V_S)					
LM386		4		12	V
LM386N-4		5		18	V
Quiescent Current (I_Q)	V_S = 6V, V_{IN} = 0		4	8	mA
Output Power (P_{OUT})					
LM386N-1	V_S = 6V, R_L = 8Ω, THD = 10%	250	325		mW
LM386N-3	V_S = 9V, R_L = 8Ω, THD = 10%	500	700		mW
LM386N-4	V_S = 16V, R_L = 32Ω, THD = 10%	700	1000		mW
Voltage Gain (A_V)	V_S = 6V, f = 1 kHz		26		dB
	10μF from Pin 1 to 8		46		dB
Bandwidth (BW)	V_S = 6V, Pins 1 and 8 Open		300		kHz
Total Harmonic Distortion (THD)	V_S = 6V, R_L = 8Ω, P_{OUT} = 125 mW f = 1 kHz, Pins 1 and 8 Open		0.2		%
Power Supply Rejection Ratio (PSRR)	V_S = 6V, f = 1 kHz, C_{BYPASS} = 10μF Pins 1 and 8 Open, Referred to Output		50		dB
Input Resistance (R_{IN})			50		kΩ
Input Bias Current (I_{BIAS})	V_S = 6V, Pins 2 and 3 Open		250		nA

Note 1: For operation in ambient temperatures above 25°C, the device must be derated based on a 150°C maximum junction temperature and a thermal resistance of 100°C/W junction to ambient.

Note 2: For operation in ambient temperatures above 25°C, the device must be derated based on a 150°C maximum junction temperature and a thermal resistance of 187°C junction to ambient.

GAIN

BYPASS

V_S

V_{OUT}

TL/H/6976-2

N-1,
N-4
8E

976-3

Application Hints

GAIN CONTROL

To make the LM386 a more versatile amplifier, two pins (1 and 8) are provided for gain control. With pins 1 and 8 open the 1.35 kΩ resistor sets the gain at 20 (26 dB). If a capacitor is put from pin 1 to 8, bypassing the 1.35 kΩ resistor, the gain will go up to 200 (46 dB). If a resistor is placed in series with the capacitor, the gain can be set to any value from 20 to 200. Gain control can also be done by capacitively coupling a resistor (or FET) from pin 1 to ground.

Additional external components can be placed in parallel with the internal feedback resistors to tailor the gain and frequency response for individual applications. For example, we can compensate poor speaker bass response by frequency shaping the feedback path. This is done with a series RC from pin 1 to 5 (paralleling the internal 15kΩ resistor). For 6 dB effective bass boost: R ≅ 15 kΩ, the lowest value for good stable operation is R = 10 kΩ if pin 8 is open. If pins 1 and 8 are bypassed then R as low as 2 kΩ can be used. This restriction is because the amplifier is only compensated for closed-loop gains greater than 9.

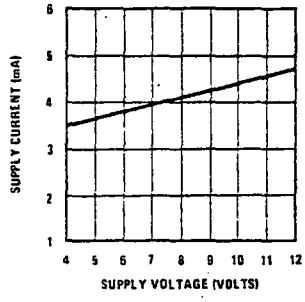
INPUT BIASING

The schematic shows that both inputs are biased to ground with a 50 kΩ resistor. The base current of the input transistors is about 250 nA, so the inputs are at about 12.5 mV when left open. If the dc source resistance driving the LM386 is higher than 250 kΩ it will contribute very little additional offset (about 2.5 mV at the input, 50 mV at the output). If the dc source resistance is less than 10 kΩ, then shorting the unused input to ground will keep the offset low (about 2.5 mV at the input, 50 mV at the output). For dc source resistances between these values we can eliminate excess offset by putting a resistor from the unused input to ground, equal in value to the dc source resistance. Of course all offset problems are eliminated if the input is capacitively coupled.

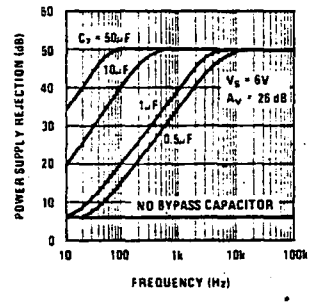
When using the LM386 with higher gains (bypassing the 1.35 kΩ resistor between pins 1 and 8) it is necessary to bypass the unused input, preventing degradation of gain and possible instabilities. This is done with a 0.1μF capacitor or a short to ground depending on the dc source resistance on the driven input.

Typical Performance Characteristics

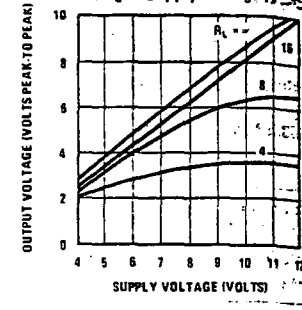
Quiescent Supply Current vs Supply Voltage



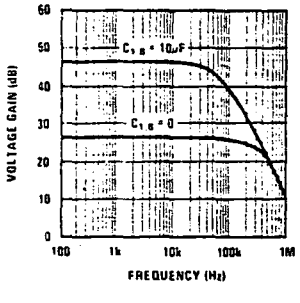
Power Supply Rejection Ratio (Referred to the Output) vs Frequency



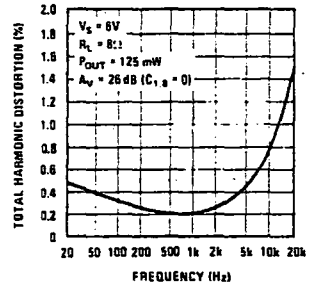
Peak-to-Peak Output Voltage Swing vs Supply Voltage



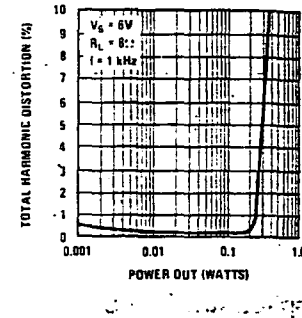
Voltage Gain vs Frequency



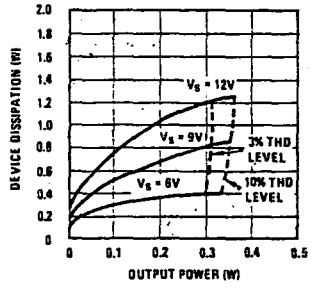
Distortion vs Frequency



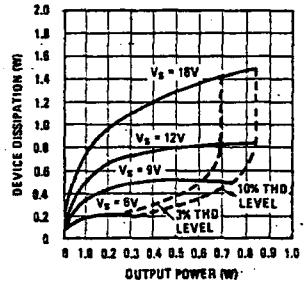
Distortion vs Output Power



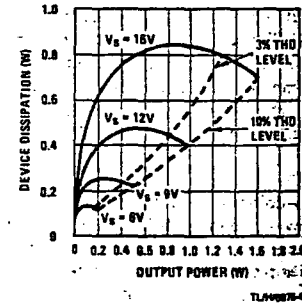
Device Dissipation vs Output Power—4Ω Load



Device Dissipation vs Output Power—8Ω Load

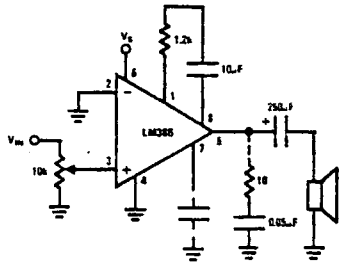


Device Dissipation vs Output Power—16Ω Load

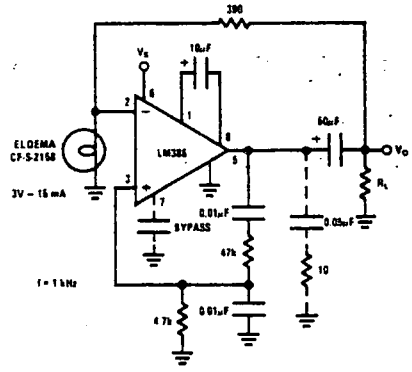


Typical Applications (Continued)

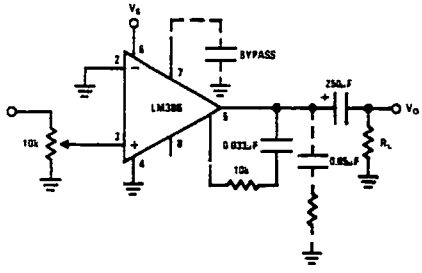
Amplifier with Gain = 50



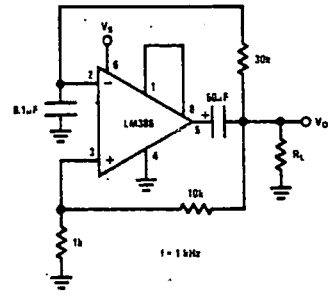
Low Distortion Power Wienbridge Oscillator



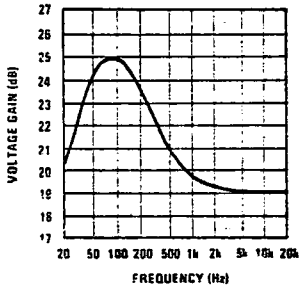
Amplifier with Bass Boost



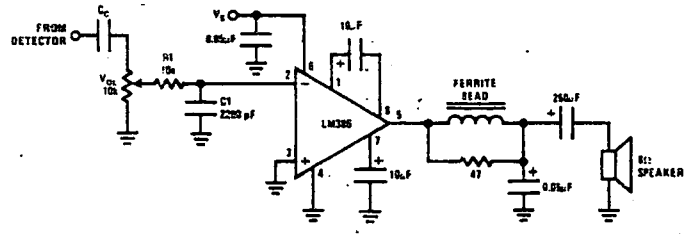
Square Wave Oscillator



Frequency Response with Bass Boost



AM Radio Power Amplifier



Note 1: Twist supply lead and supply ground very tightly.

Note 2: Twist speaker lead and ground very tightly.

Note 3: Ferrite bead is Ferroxcube K5-001-001/3B with 3 turns of wire.

Note 4: R1C1 band limits input signals.

Note 5: All components must be spaced very close to IC.

TL/H/5976-5

National Semiconductor

LM567/LM567C Tone Decoder

General Description

The LM567 and LM567C are general purpose tone decoders designed to provide a saturated transistor switch to ground when an input signal is present within the passband. The circuit consists of an I and Q detector driven by a voltage controlled oscillator which determines the center frequency of the decoder. External components are used to independently set center frequency, bandwidth and output delay.

Features

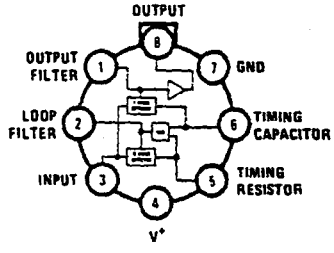
- 20 to 1 frequency range with an external resistor
- Logic compatible output with 100 mA current sinking capability
- Bandwidth adjustable from 0 to 14%
- High rejection of out of band signals and noise
- Immunity to false signals
- Highly stable center frequency
- Center frequency adjustable from 0.01 Hz to 500 kHz

Applications

- Touch tone decoding
- Precision oscillator
- Frequency monitoring and control
- Wide band FSK demodulation
- Ultrasonic controls
- Carrier current remote controls
- Communications paging decoders

Connection Diagrams

Metal Can Package

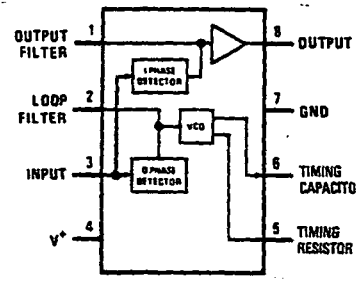


TL/H/6975–1

Top View

Order Number LM567H or LM567CH
See NS Package Number H08C

Dual-In-Line and Small Outline Packages



TL/H/6975–2

Top View

Order Number LM567CM
See NS Package Number M08A
Order Number LM567CN
See NS Package Number N08E

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage Pin	9V
Power Dissipation (Note 1)	1100 mW
V_8	15V
V_3	−10V
V_9	V_4 + 0.5V
Storage Temperature Range	−65°C to +150°C
Operating Temperature Range	
LM567H	−55°C to +125°C
LM567CH, LM567CM, LM567CN	0°C to +70°C

Soldering Information

Dual-In-Line Package	
Soldering (10 sec.)	260°C
Small Outline Package	
Vapor Phase (60 sec.)	215°C
Infrared (15 sec.)	220°C

See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" for other methods of soldering surface mount devices.

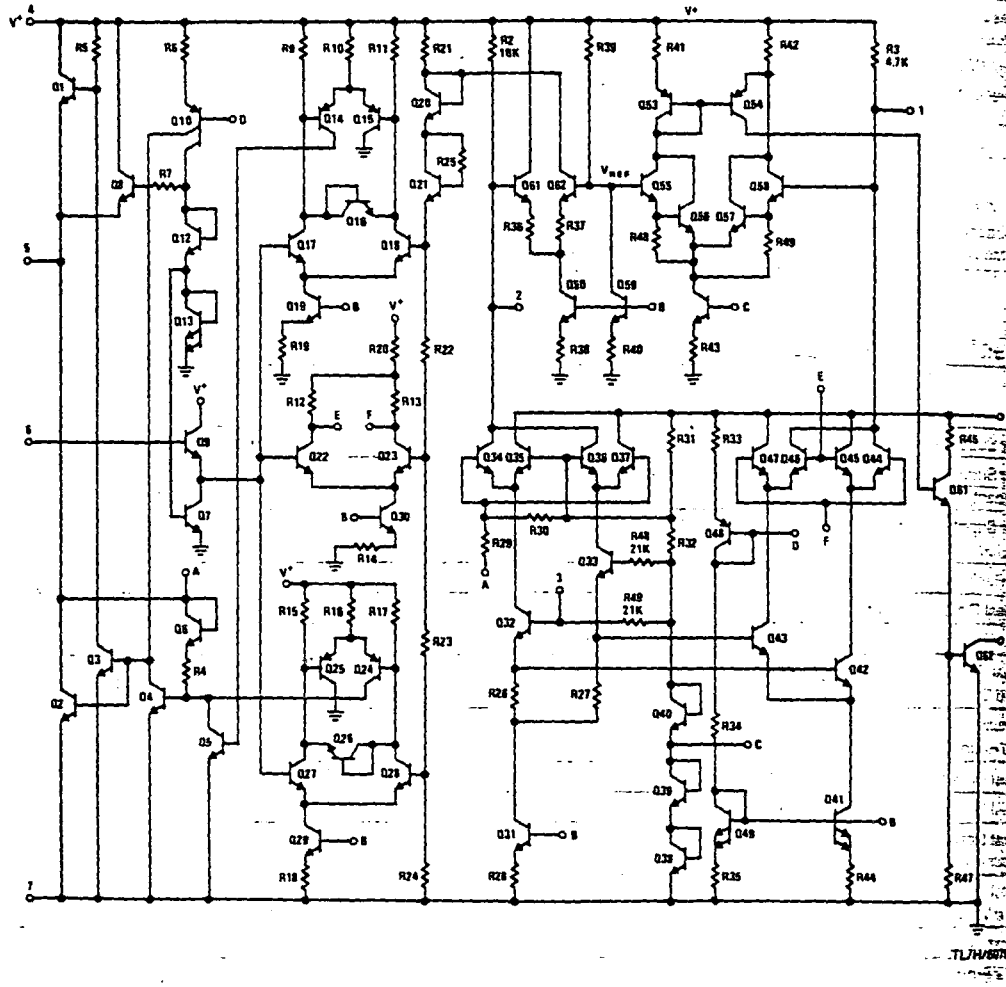
Electrical Characteristics AC Test Circuit, T_A = 25°C, V^+ = 5V

Parameters	Conditions	LM567			LM567C/LM567CM			Units
		Min	Typ	Max	Min	Typ	Max	
Power Supply Voltage Range		4.75	5.0	9.0	4.75	5.0	9.0	V
Power Supply Current Quiescent	R_L = 20k		6	8		7	10	mA
Power Supply Current Activated	R_L = 20k		11	13		12	15	mA
Input Resistance		18	20		15	20		kΩ
Smallest Detectable Input Voltage	I_L = 100 mA, $f_i = f_o$		20	25		20	25	mVrms
Largest No Output Input Voltage	I_C = 100 mA, $f_i = f_o$	10	15		10	15		mVrms
Largest Simultaneous Outband Signal to Inband Signal Ratio			6			6		dB
Minimum Input Signal to Wideband Noise Ratio	B_n = 140 kHz		−6			−6		dB
Largest Detection Bandwidth		12	14	16	10	14	18	% of f_o
Largest Detection Bandwidth Skew			1	2		2	3	% of f_o
Largest Detection Bandwidth Variation with Temperature			±0.1			±0.1		%/°C
Largest Detection Bandwidth Variation with Supply Voltage	4.75 − 6.75V		±1	±2		±1	±5	%V
Highest Center Frequency		100	500		100	500		kHz
Center Frequency Stability (4.75–5.75V)	0 < T_A < 70 −55 < T_A < +125		35 ± 60 35 ± 140			35 ±60 35 ± 140		ppm/°C ppm/°C
Center Frequency Shift with Supply Voltage	4.75V − 6.75V 4.75V − 9V		0.5	1.0 2.0		0.4	2.0 2.0	%/V %/V
Fastest ON-OFF Cycling Rate			f_o/20			f_o/20		
Output Leakage Current	V_8 = 15V		0.01	25		0.01	25	μA
Output Saturation Voltage	e_i = 25 mV, I_8 = 30 mA e_i = 25 mV, I_8 = 100 mA		0.2 0.6	0.4 1.0		0.2 0.6	0.4 1.0	V
Output Fall Time			30			30		ns
Output Rise Time			150			150		ns

Note 1: The maximum junction temperature of the LM567 and LM567C is 150°C. For operating at elevated temperatures, devices in the TO-5 package must be derated based on a thermal resistance of 150°C/W, junction to ambient or 45°C/W, junction to case. For the DIP the device must be derated based on a thermal resistance of 110°C/W, junction to ambient. For the Small Outline package, the device must be derated based on a thermal resistance of 160°C/W, junction to ambient.

Note 2: Refer to RETS567X drawing for specifications of military LM567H version.

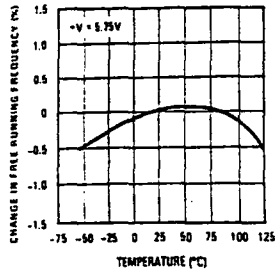
Schematic Diagram



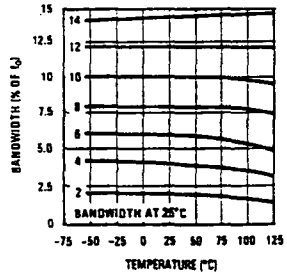
TL/H/6674-1

Typical Performance Characteristics

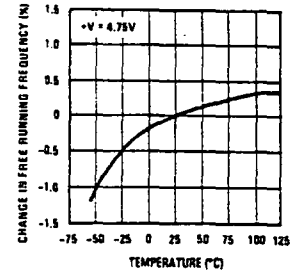
Typical Frequency Drift



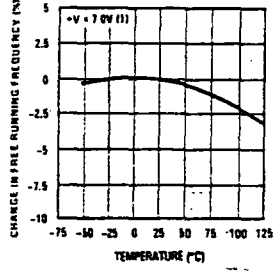
Typical Bandwidth Variation



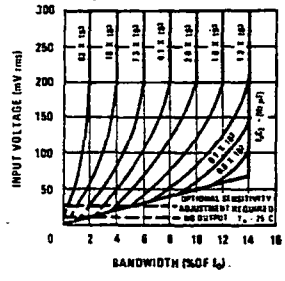
Typical Frequency Drift



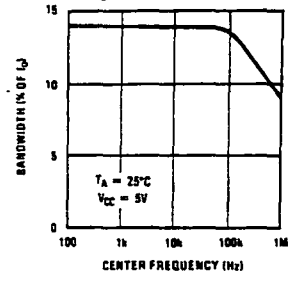
Typical Frequency Drift



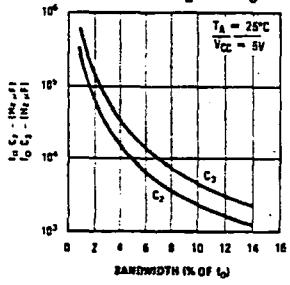
Bandwidth vs Input Signal Amplitude



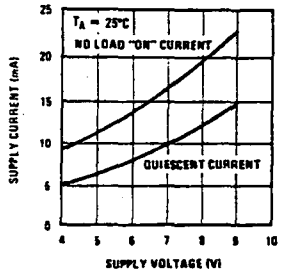
Largest Detection Bandwidth



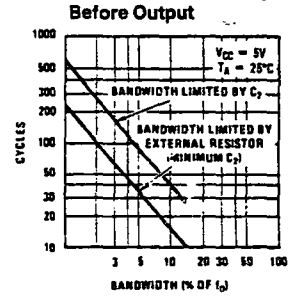
Detection Bandwidth as a Function of C_2 and C_3



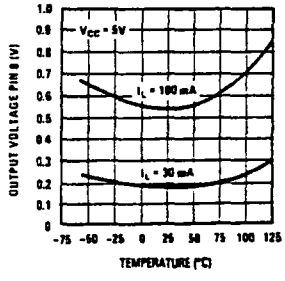
Typical Supply Current vs Supply Voltage



Greatest Number of Cycles Before Output



Typical Output Voltage vs Temperature

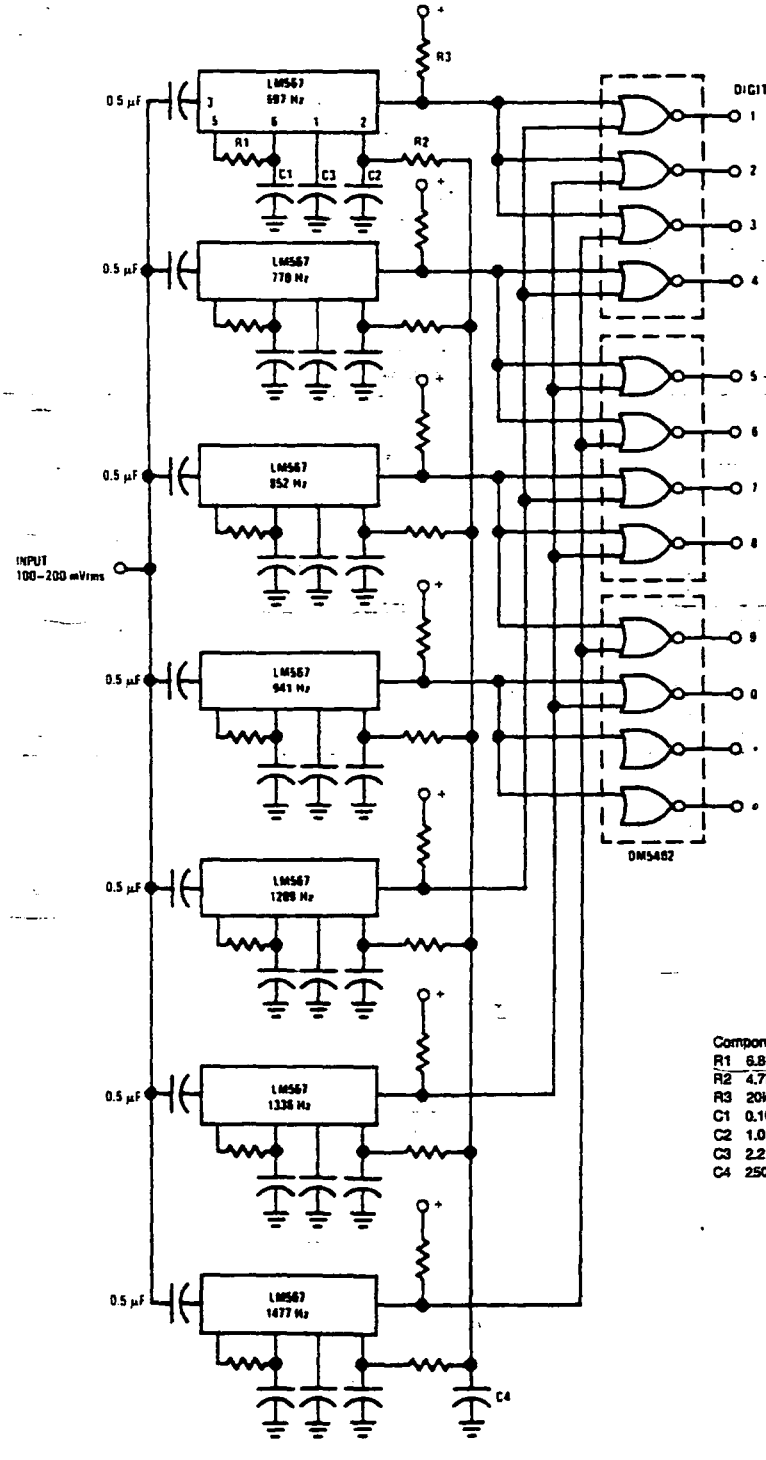


TL/H/6975–4

5

Typical Applications

Touch-Tone Decoder



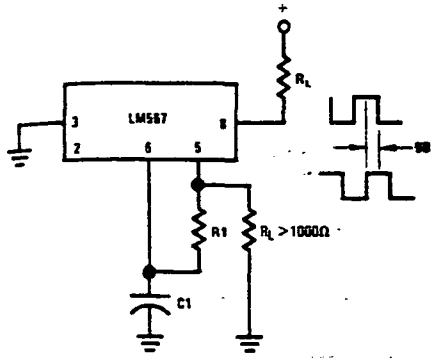
Component values (typ)

R1	6.8 to 15k
R2	4.7k
R3	20k
C1	0.10 mfd
C2	1.0 mfd 6V
C3	2.2 mfd 6V
C4	250 mfd 6V

TL/H/6975–5

Typical Applications (Continued)

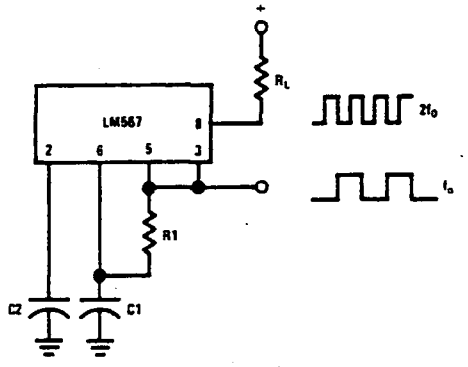
Oscillator with Quadrature Output



Connect Pin 3 to 2.8V to Invert Output

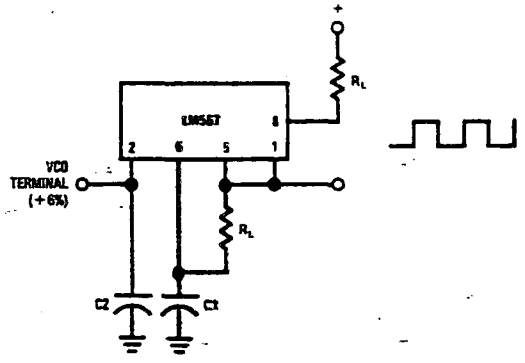
TL/H/6975–6

Oscillator with Double Frequency Output



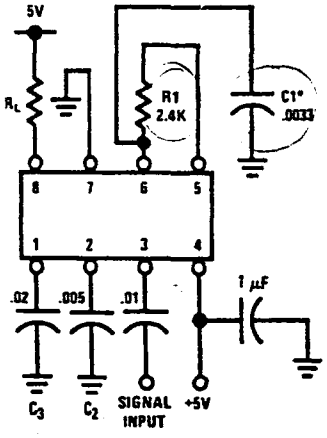
TL/H/6975–7

Precision Oscillator Drive 100 mA Loads



TL/H/6975–8

AC Test Circuit



TL/H/6975–9

f_i = 100 kHz + 5V
*Note: Adjust for f_o = 100 kHz.

Applications Information

The center frequency of the tone decoder is equal to the free running frequency of the VCO. This is given by

$$f_o \cong \frac{1}{1.1\,R_1C_1}$$

The bandwidth of the filter may be found from the approximation

$$BW = 1070\sqrt{\frac{V_i}{f_oC_2}} \text{ in \% of } f_o$$

Where:

V_i = Input voltage (volts rms), $V_i \leq 200$ mV

C_2 = Capacitance at Pin 2 (μF)

5

MC145412
MC145413
MC145512

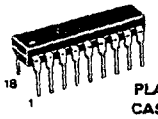
Advance Information

Pulse/Tone Repertory Dialer

Low Power Silicon-Gate CMOS

The MC145412/13 and MC145512 are silicon gate, monolithic CMOS integrated circuits which convert keyboard inputs into either pulse or DTMF outputs. They are packaged in a standard 18 pin (0.3" wide) plastic DIP.

- 3×4 or 4×4 Keyboard Compatibility Which Allows the Use of 2-of-7, 2-of-8, or Form A Type Keyboards
- MC145413 Adds Keyboard Selectable Pause Switch Function
- Single Pin Switchable Between DTMF, 10 pps and 20 pps
- 500 Hz Tone Signal Output in the Pulse Dialing Mode
- Memory Storage for Ten 18 Digit Numbers, Including Last Number Redial
- Uses 3.579545 MHz Colorburst Crystal
- Telephone Line Powered
- Silicon Gate CMOS Technology for 1.7–5.5 V Low Power Operation
- Stand Alone DTMF Dialer/Stand Alone Pulse Dialer
- Mute Output Used to Isolate Receiver from Dialing Output
- Memory Programming Options by Keyboard Configuration

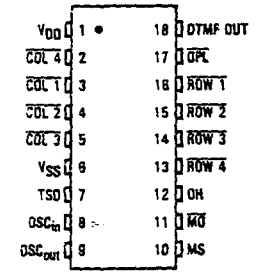


PLASTIC
CASE 707

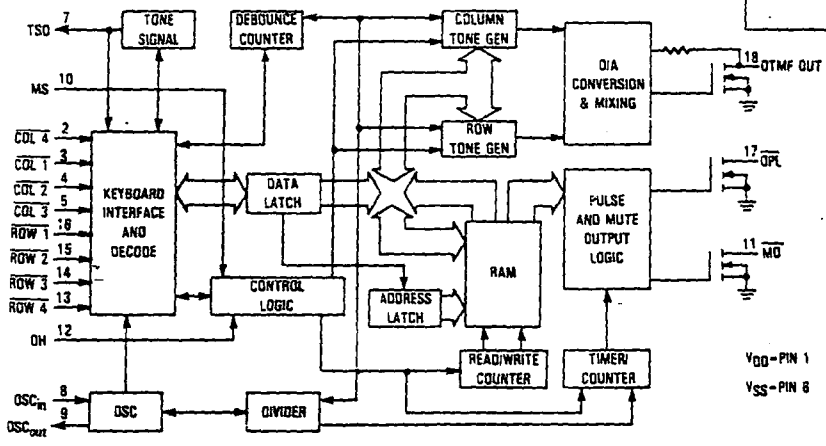
Ordering Information

MC145 X X X X
- P Plastic
- 4 40/60 M/B Ratio
- 5 32/68 M/B Ratio

PIN ASSIGNMENT



BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

2

ABSOLUTE MAXIMUM RATINGS (V_{SS} = 0 V)

Parameter	Symbol	Rating	Unit
DC Supply Voltage	V_{DD}	−0.5 to +8.0	V
Operating Temperature	T_A	−30 to +60	°C
Storage Temperature	T_{stg}	−65 to +150	°C
DC Current Drain Per Pin	I	10	mA
Maximum Voltage On Any Pin Relative to V_{SS} On Any Pin Relative to V_{DD}	 V_{in1} V_{in2}	 −0.5 +0.5	V

ELECTRICAL CHARACTERISTICS (T_A = −30 to 60°C, V_{DD} = 2.5 V, V_{SS} = 0 V, Unless Otherwise Noted)

Characteristics		Symbol	Min	Typ	Max	Unit
DC Supply Voltage	Pulse Mode DTMF Mode	V_{DD}	2.0 2.5	– –	5.5 5.5	V
Operating Current	Pulse Mode (MS = V_{DD}) DTMF Mode (MS = V_{SS})	I_{DD}	– –	0.25 1.0	0.7 2.0	mA
Memory Retention Voltage		V_{stby}	1.7	–	–	V
Memory Retention Current	(V_{DD} = 1.7 V) (V_{DD} = 2.5 V)	I_{stby}	– –	1.0 1.2	2.0 2.5	µA
Input Voltage, Row/Column/OH	"0" Level "1" Level	V_{IL} V_{IH}	– 0.8 V_{DD}	– –	0.2 V_{DD} –	V
Row/Column Input Impedance	To V_{DD} To V_{SS}	Z_{in}	– –	100 2	– –	kΩ
OH Pull-Up Resistance		R	–	50	–	kΩ
Input Capacitance (All Inputs)		C_{in}	–	10	–	pF
MS Pin Input Impedance		Z_{in}	50	200	–	kΩ
Output Sink Current	(V_{DD} = 2.5 V) TSO Pin $\overline{MO}$ Pin $\overline{OPL}$ Pin (V_{DD} = 4.0 V) $\overline{MO}$ Pin $\overline{OPL}$ Pin	I_{OL}	0.5 1.0 1.0 3.0 4.5	0.7 2.0 2.0 – –	– – – – –	mA
TSO Output Source Current (V_{out} = 2.0 V)		I_{OH}	0.5	0.7	–	mA
Output Leakage Current	$\overline{MO}$, $\overline{OPL}$ Pins	I_{lkg}	–	–	1.0	µA
DTMF Output Level Referenced to V_{DD}/2 (V_{DD} = 2.5 to 4.0 V, RL = 600 Ω to V_{DD})	Row Tone Column Tone	V_{out}	260 330	310 390	370 460	mV RMS
DTMF Output Tone Leakage (V_{DD} = 3.5, RL = 600 Ω, 300 to 4000 Hz)			–		−80	dBm
DTMF Output Tone Distortion (V_{DD} = 3.5, RL = 600 Ω, 300 to 4000 Hz)			–	–	5	%
Pre-Emphasis			1	2	2.5	dB
DTMF Output Leakage Current While Not Dialing Tones (V_{DD} = 2.5 V)			–	–	1.0	µA
DTMF Output Sink Current While Dialing Tones			20	–	–	µA

SWITCHING CHARACTERISTICS (T_A = 25°C, V_{DD} = 2.5 V, Osc. Freq. = 3.579545 MHz, Unless Otherwise Noted)

Characteristics		Symbol	Min	Typ	Max	Unit
Row/Column Scan Frequency		f	—	250	—	Hz
Key Debounce Time		t_{DB}	16	—	20	ms
DTMF Tone Duration for Keypad Dialing		t_{w1}	60	78	—	ms
DTMF Tone Duration for Memory Dialing		t_{w2}	90	102	110	ms
Inter-Digit Pause Time	DTMF (Memory Dialing)	t_{ID}	90	98	110	ms
	Pulse 10 pps		0.8	1.0	1.2	s
	20 pps		0.4	0.5	0.6	
MS-Pin Scan Rate		t_{rms}	—	1	—	kHz
Make/Break Ratio (MS = Open or V_{DD})	MC145412/13	MBR	—	40/60	—	%
	MC145512		—	32/68	—	
Outpulsing Rate	MS = Open	f_{OPL}	—	10	—	pps
	MS = V_{DD}		—	20	—	
MUTE Output ($\overline{MO}$) Overlap Time		t_{MO}	—	2	—	ms
TSO Output Frequency		f_{TSO}	—	500	—	Hz
TSO Output Duration		t_{TSO}	35	—	40	ms
DTMF Cycle Time	(Memory Dialing)		—	5	—	tones/s
	(Keypad Dialing)		—	10	—	
DTMF Frequency Deviation			—	—	+1.0	%
Predigit Mute	MC145412/13 Pulse 10 pps	t_d	—	40	—	ms
	20 pps		—	20	—	
	MC145512 Pulse 10 pps		—	32	—	
	20 pps		—	16	—	
	DTMF		—	1	—	

PIN DESCRIPTIONS

V_{DD}, V_{SS}—POWER SUPPLY (PIN 1, PIN 6)

DC power is supplied to the part on these two pins, with V_{DD} being the most positive. Permissible ranges are from 1.7 to 5.5 V.

$\overline{MS}$—MODE SELECT (PIN 10)

The MS pin is a three state input for switching between DTMF, 10 pps, and 20 pps dialing modes. Mode selection is done during the first key entry debounce period after the dialer has completed a dialing sequence or has just come off hook. When this pin is not scanned it is high impedance.

This pin is a combination input and weak output. The input circuitry has the capability to determine each of these three states. When the pin is open the weak driver will be able to clock the pin at 1 kHz. The relationship between pin input voltage and operating mode is shown in Table 1 below.

Table 1. Mode Select Options

MS	Dialing Mode
V_{DD}	20 pps Pulse Dialing
Open	10 pps Pulse Dialing
V_{SS}	DTMF Dialing

OH-ON-HOOK (PIN 12)

Connecting the OH pin to V_{DD}, or allowing it to float sets the device in the on-hook mode. Connecting this pin to V_{SS} selects the off-hook mode. When in the on-hook mode, repertory memory can be programmed without a dialing output.

TSO—TONE SIGNAL OUTPUT (PIN 7)

TSO emits 500 Hz tone signals after valid key inputs are accepted providing audio feedback for key depressions, except when DTMF tones are generated. This pin also outputs a tone during on-hook programming.

DTMF OUT—DUAL TONE MULTIFREQUENCY OUTPUT (PIN 18)

When the MS pin is set to V_{SS} the DTMF OUT pin outputs tones corresponding to the row and column of the key depressed. Simultaneously depressing two or more keys in a single row (or column) will generate the corresponding row (or column) tone on 4 × 4 keypad mode only.

In pulse dialing mode (MS = V_{DD} or float) and during on-hook programming this pin is high impedance. While outputting tones, this pin has a dc bias at ($V_{DD} - V_{SS}$)/2. DTMF OUT is an open drain output requiring an external pull-up to V_{DD}. This pull-up resistor must satisfy the instantaneous current requirements of the internal feedback network in addition to the load applied to the pin.

$\overline{OPL}$—OUTPULSING (PIN 17)

This pin outputs pulses at 10 pps (MS is open) or 20 pps (MS = V_{DD}). The MC145412/13 have a make/break ratio of 40/60, while the MC145512 has a make/break ratio of 32/68. In the DTMF dialing mode (MS = V_{SS}), this output is high impedance. During on-hook programming this pin will not outpulse. This pin is an open drain N-channel output which pulls low to break the loop current.

$\overline{MO}$—MUTE OUTPUT (PIN 11)

The Mute Output is an open drain N-channel output that pulls to V_{SS} during $\overline{OPL}$ outpulsing and during off-hook key depressions and memory dialing in DTMF mode.

KEYBOARD INPUTS—(PINS 2, 3, 4, 5, 13, 14, 15, 16)

The keyboard inputs allow either a single contact (Class A) keyboard, or a standard 2-of-8 or 2-of-7 keyboard with V_{SS} tied to common. A valid key entry occurs when either a single row is tied to a single column, or a single row and column are simultaneously connected to V_{SS}. Connecting pin 2, $\overline{COL\ 4}$, to V_{DD} sets the part to 3 × 4 keyboard mode. Keyboard mode selection is performed during application of power.

Typical keyboard configurations are shown in Figure 1.

OSC_{in}, OSC_{out} (PIN 8, PIN 9)

A 3.579545 MHz crystal is required as the frequency reference for the on-chip oscillator. Crystal biasing is accomplished by an internal resistor and capacitors.

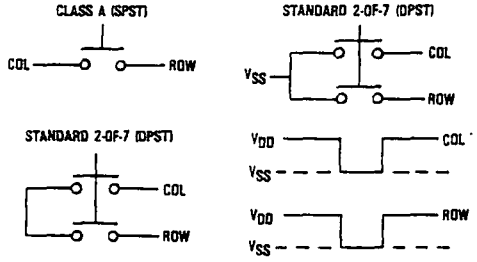


Figure 1. Keyboard Configurations

GENERAL DEVICE DESCRIPTION

The MC145412/MC145512 and the MC145413 provide users with switchable pulse and DTMF dialing functions. The MC145412/MC145512 change dialing modes via the MS pin. The MC145413 allows users to switch dialing modes via the keyboard in addition to the MS pin. All devices have 10 memories, LNR (last number redial) inclusive, each 18 digits long.

On application of power, there is a 64 ms initialization period during which the oscillator is enabled and the keyboard inputs are disabled. During initialization $\overline{COL\ 4}$ is scanned to set the keyboard mode. If the $\overline{COL\ 4}$ input is high (V_{DD}) the dialer is set to the 3 × 4 keypad mode, otherwise the 4 × 4 keypad mode is selected. Changing modes is not possible after this initialization period.

During normal dialing, the oscillator starts when a key is depressed. The key input is debounced for 32 ms. During this debounce period the RAM and dialing circuits are disabled, the mode select pin is scanned to determine the dialing mode (either 10 pps, 20 pps, or DTMF). After debounce, the keypad entry is checked and the input is latched into LNR memory followed by a stop code. This process continues until 18 digits have been entered. If a 19th digit is entered, it will over-write the first digit and will be followed by a stop code. When dialing, the device fetches data from memory until a stop code is encountered or 18 digits have been dialed.

During manual DTMF dialing, a minimum tone duration of 60 ms DTMF is output and will continuously output in 32 ms increments as long as the key is depressed. The DTMF OUT pin is designed to drive an external PNP transistor which can be used to modulate tip and ring voltage at the DTMF frequencies.

If the first key is for redial or recall, the device will respond accordingly, either redialing the last number entered, or recalling and dialing the number selected by a subsequent key depression. Responses to dialing sequences for 4 × 4 keyboards are shown in Figure 2, and 3 × 4 keyboard responses are shown in Figure 3.

The MC145412 series can be configured with an external battery to provide memory retention power and allow on-hook programming of the repertory memory. If the part is in the on-hook mode and a key is depressed, the oscillator will start and the key entry will be stored in the last number redial memory. Dialing outputs will not be activated while the device is in the on-hook condition. Dialing inputs will be stored in last number redial memory, as during off-hook operation. After the number has been entered in the on-hook mode, it can be stored in repertory memory. For the 4 × 4 keyboard, pressing the STORE key (* for 3 × 4 keyboard), followed by a digit (1 through 9) will store the number in the repertory memory location specified by the digit.

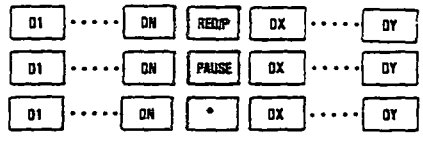
The RECALL key for the 4 × 4 keypad is used to recall and dial numbers stored in the repertory memory. The digit immediately following the RECALL key designates the memory location of the number to be auto-dialed. For the 4 × 4 keyboard, a last number redial can be accomplished if the RED/P key ($\overline{COL\ 4}$, $\overline{ROW\ 1}$) is the first key depressed after an on-hook to off-hook transition. Otherwise the RED/P key will effect a 4 second pause. If the pulse mode is selected, redial can be accomplished if the first key depressed on a transition to off-hook is #. For the 3 × 4 keyboard, redial occurs if the first key depressed is *,0.

The PAUSE key ($\overline{COL\ 4}$, $\overline{ROW\ 2}$) for the MC145412/MC145512 will cause a 4 second pause. The PAUSE/S key ($\overline{COL\ 4}$, $\overline{ROW\ 2}$) is a feature offered on the MC145413. Depressing this key will cause a 4 second delay, and will switch dialing modes. PAUSE (and PAUSE/S) is stored in memory for pauses (and mode switching) during auto-dialing.

1. MANUAL DIALING—OFF-HOOK (PULSE OR DTMF MODE)

ALL DIGITS ENTERED WILL BE STORED IN THE LAST NUMBER REDIAL REGISTER. PRESSING * OR # WILL DIAL OUT THE DTMF SIGNAL IN TONE MODE ONLY.

2. MANUAL DIALING WITH AUTO ACCESS PAUSE—OFF-HOOK (PULSE OR DTMF MODE)

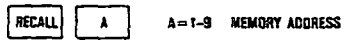


THE AUTO ACCESS PAUSE WILL NOT OCCUR DURING MANUAL DIALING IN DTMF MODE. IT IS RETRIEVED DURING RECALL OR REDIAL.

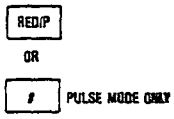
3. STORING NUMBERS INTO MEMORY—ON-HOOK/OFF-HOOK (PULSE OR DTMF MODE)

THIS OPERATION TRANSFERS THE DIGITS 01 TO ON FROM THE LAST NUMBER REDIAL REGISTER TO AN ADDRESS SPACE SPECIFIED BY "A". DIALING OUTPUTS ARE NOT ACTIVATED DURING ON-HOOK PROGRAMMING.

4. MEMORY REDIAL—OFF-HOOK (PULSE OR DTMF MODE)



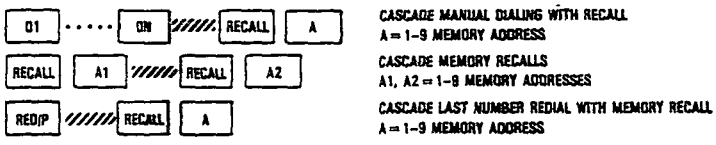
5. LAST NUMBER REDIAL—OFF-HOOK (PULSE OR DTMF MODE)



REDIALS THE NUMBER THAT WAS PREVIOUSLY ENTERED INTO THE LAST NUMBER REDIAL REGISTER.

6. PULSE-TO-TONE MODE SWITCH—OFF-HOOK (PULSE OR DTMF MODE)

7. CASCADED DIALING—OFF-HOOK (PULSE OR DTMF MODE)



////// WAIT UNTIL PREVIOUS REDIAL OR RECALL SIGNALS HAVE BEEN SENT BEFORE SUBSEQUENT ENTRIES ARE MADE.

8. SIGNALING * AND # TONES—OFF-HOOK (DTMF MODE ONLY)

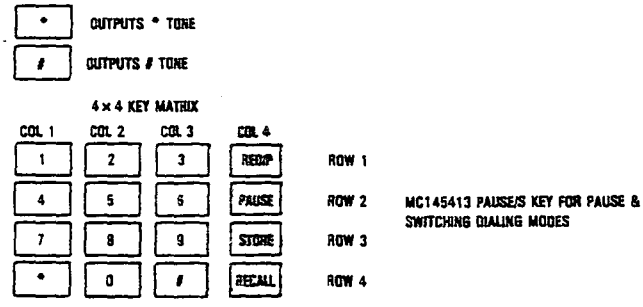
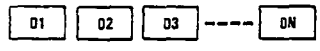


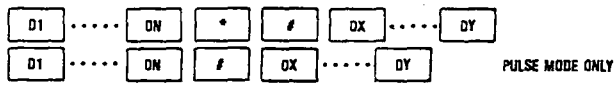
Figure 2. 4 × 4 Keyboard Dialing Sequences

1. **MANUAL DIALING—OFF-HOOK** (PULSE OF DTMF MODE)



ALL KEY ENTRIES, EXCEPT * AND #, WILL BE STORED IN THE LAST NUMBER REDIAL REGISTER. PRESSING * OR # WILL NOT DIAL OUT THE DTMF SIGNAL IN TONE MODE. FOR SIGNALING, * OR # SHOULD BE PRESSED TWICE.

2. **MANUAL DIALING WITH AUTO ACCESS PAUSE—OFF-HOOK** (PULSE OR DTMF MODE)

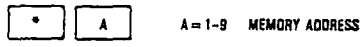


THE AUTO ACCESS PAUSE WILL NOT OCCUR ON MANUAL DIALING IN DTMF MODE. IT CAN ONLY BE RETRIEVED DURING RECALL OR REDIAL.

3. **STORING NUMBERS INTO MEMORY—ON-HOOK** (PULSE OR DTMF MODE)

THIS OPERATION TRANSFERS THE DIGITS D1 TO DN FROM THE LAST NUMBER REDIAL REGISTER TO AN ADDRESS SPACE SPECIFIED BY "A".

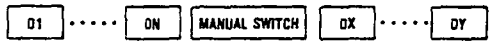
4. **MEMORY REDIAL—OFF-HOOK** (PULSE OR DTMF MODE)



5. **LAST NUMBER REDIAL—OFF-HOOK** (PULSE OR DTMF MODE)

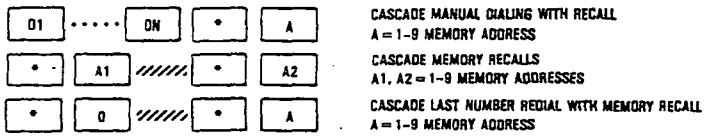
THIS OPERATION REDIALS THE LAST NUMBER ENTERED OFF-HOOK AND RETRIEVES DATA FROM MEMORY ADDRESS 0.

6. **PULSE-TO-TONE AND TONE-TO-PULSE SWITCHING—OFF-HOOK** (PULSE OR DTMF MODE)



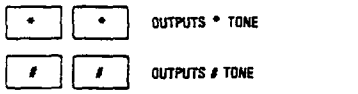
MODE SELECT (MS) PIN HAS TO BE MANUALLY SWITCHED TO DETERMINE THE DIALING MODE. DIALING MODE SELECTION WITH MANUAL SWITCH IS NOT PROGRAMMED INTO THE LAST NUMBER REDIAL MEMORY.

7. **CASCADED DIALING—OFF-HOOK** (PULSE OR DTMF MODE)



////// WAIT UNTIL PREVIOUS REDIAL OR RECALL SIGNALS HAVE BEEN SENT BEFORE SUBSEQUENT ENTRIES ARE MADE.

8. **SIGNALING * AND # TONES—OFF-HOOK** (DTMF MODE ONLY)



3 × 4 KEY MATRIX

COL 1	COL 2	COL 3	
1	2	3	ROW 1
4	5	6	ROW 2
7	8	9	ROW 3
*	0	#	ROW 4

Figure 3. 3 × 4 Keyboard Dialing Sequences

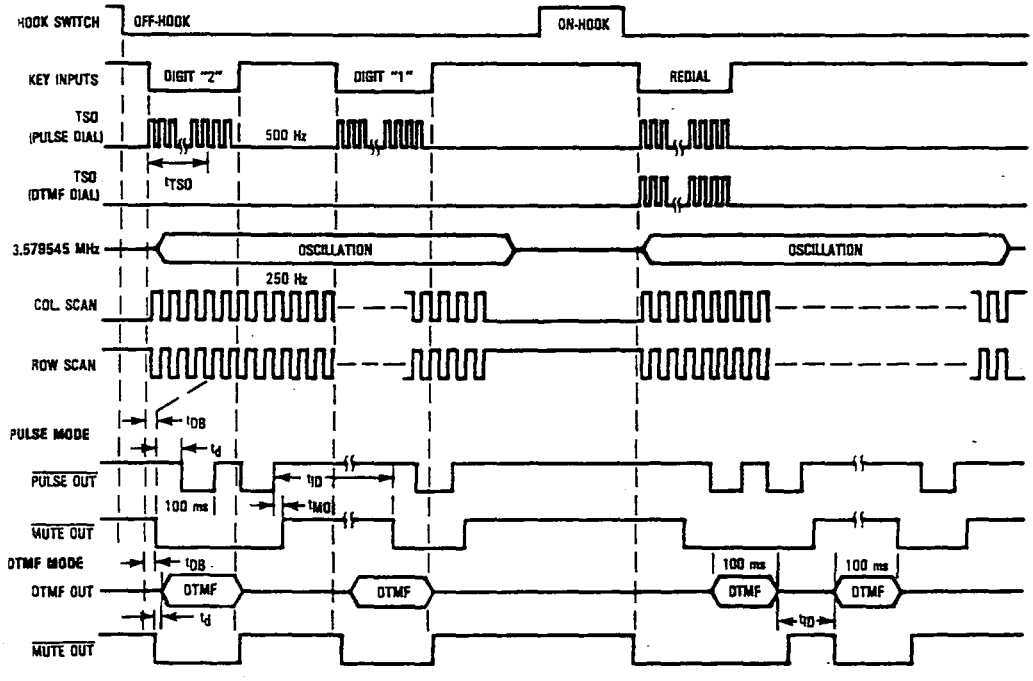


Figure 4. Timing Diagram

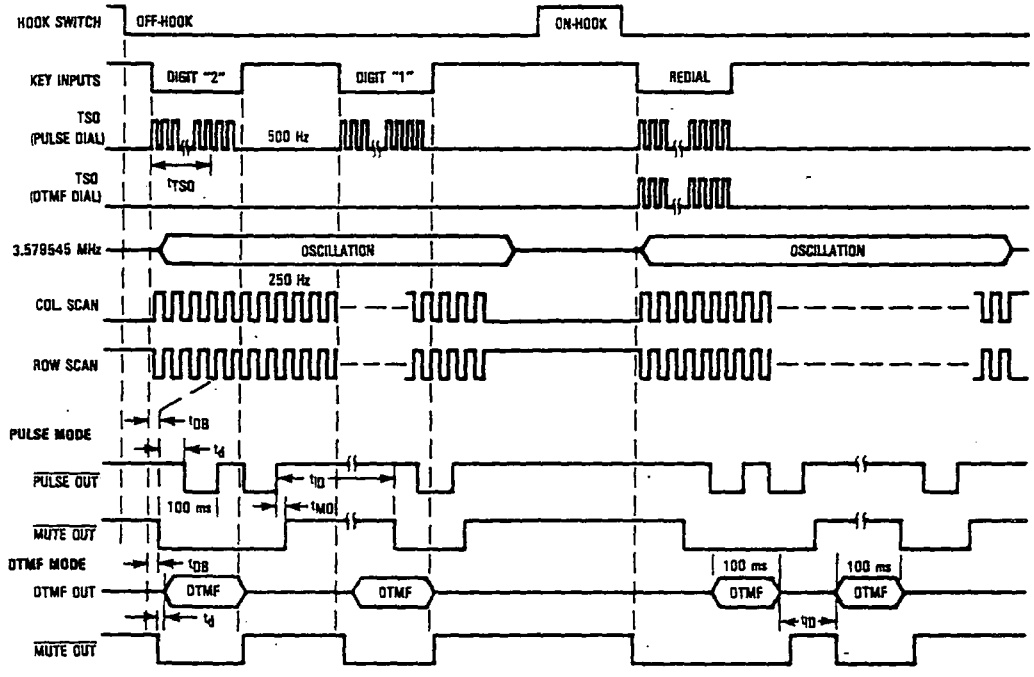


Figure 4. Timing Diagram

MC145450

Advance Information

CMOS

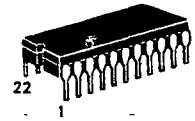
1200 BAUD
FSK MODEM

2

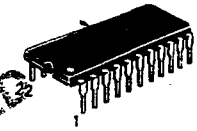
1200 BAUD FSK MODEM

The MC145450 is a silicon-gate CMOS frequency shift keying (FSK) modem intended for use in Bell 202 and CCITT V.23 applications. Features of the device include:

- Bell 202 Compatible 0 to 1800 Baud Main Channel
- 0 to 150 Baud Reverse Channel
- CCITT V.23 Mode 2 Compatible 0 to 1800 Main Channel
- CCITT V.23 0 to 75 Baud Compatible Reverse Channel
- TTL Compatible
- Eight Selectable $\overline{RTS}$-$\overline{CTS}$ Delay Options
- Soft Turn-Off Capability
- Answer Back Tone Generator (US and CCITT Tones)
- Carrier Detect Input
- 22 Pin Package

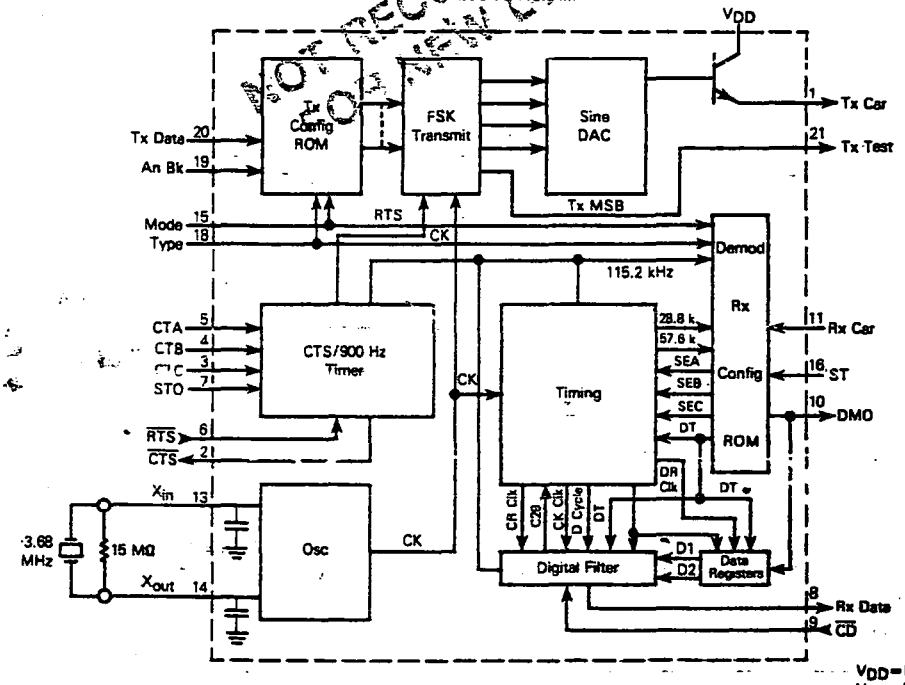


L SUFFIX
CERAMIC PACKAGE
CASE 738



P SUFFIX
PLASTIC PACKAGE
CASE 708

MC145450 1200 BAUD FSK MODEM
BLOCK DIAGRAM



This document contains information on a new product. Specifications and information herein are subject to change without notice.

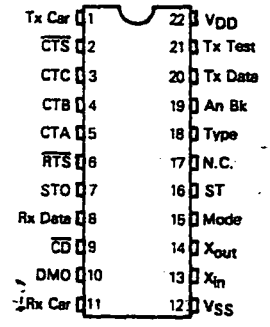
ABSOLUTE MAXIMUM RATINGS

Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	10	V
Input Voltages, All Inputs	V_{in}	$V_{SS}-0.5$ to $V_{DD}+0.5$	V
DC Current Drain per Pin Pin 3-6, 9, 15, 16, 18, 19, 20 Pins 2, 8	I_{out}	 10 35	mA
Operating Temperature Range	T_A	0 to +70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

RECOMMENDED OPERATING CONDITIONS

Parameter	Symbol	Min	Typ	Max	Unit
DC Supply Voltage	$V_{DD}-V_{SS}$	4.5	5.0	6.5	V

PIN ASSIGNMENTS



DC ELECTRICAL CHARACTERISTICS (V_{DD}=5.0 V ±5%, V_{SS}=0, T_A=0 to 70°C)

Characteristics	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}				V
Pins 3-7, 9, 15, 16, 18, 19, 20	—	2.8	—	—	
Pin 13, 11	—	4.0	—	—	
Input Low Voltage	V_{IL}				V
Pins 3-7, 9, 15, 16, 18, 19, 20		—	—	0.5	
Pin 13, 11		—	—	0.6	
Input Current	I_{in}				μA
All Inputs (V_{IL} 0 V)		—	—	−5.0	
All Inputs Except Pins 11, 13, (V_{IH}>2.8 V) (Note 1)		—	—	600	
Output High Current (V_{OH}=2.4 V)	I_{OH}				mA
Pins 2, 8 (Test Load A)		0.75	—	—	
Pins 10, 21 (Test Load B)		0.75	—	—	
Output Low Current (V_{OL}=0.4 V)	I_{OL}				mA
Pins 2, 8 (Test Load A)		1.2	—	—	
Pins 10, 21 (Test Load B)		0.6	—	—	
Operating Current	I_{DD}	—	2.5	6	mA
Input Capacitance	C_{in}				pF
All Except Pin 13		—	—	12	
Pin 13 (X_{in})		—	8	—	
Output Capacitance	C_{out}				pF
All Except Pin 14		—	—	12	
Pin 14 (X_{out})		—	13	—	
Transmit Audio Signal Level (Pin 1 R_L = 10 kΩ (Note 2)	—	0.428	0.5	0.578	Vp-p
Total Harmonic Distortion (2nd to 14th) (Note 2)	THD	—	−50	−40	dB

AC ELECTRICAL CHARACTERISTICS (V_{DD}=5.0 V ±5%, V_{SS}=0, T_A=0 to 70°C)

Characteristics		Symbol	Min	Typ	Max	Unit
Output Rise Time (Test Load A) (Pins 2, 8)		t_r	—	20	100	ns
Output Rise Time (Test Load B) (Pins 10, 14, 21)		t_r	—	20	100	ns
Output Fall Time (Test Load A) (Pins 2, 8)		t_f	—	20	100	ns
Output Fall Time (Test Load B) (Pins 10, 14, 21)		t_f	—	20	100	ns
Input Rise and Fall Times (Except Pin 13)		t_r, t_f	—	—	1000	μs
Delay From $\overline{RTS}$ to $\overline{CTS}$	STO = Low	$t_{d(low)}$	—	1	—	μs
Delay From $\overline{RTS}$ to $\overline{CTS}$	STO = High	$t_{d(high)}$	18.3	—	21.7	ms

NOTES:

1. Active pull-up devices are used on these inputs to allow interfacing to TTL devices. The I_{in} specified is a transitional load (not steady state) which is drawn when the input is brought up to 2.8 V until the internal pull-up device has raised the signal to the V_{DD} level.
2. Measured in any mode using HP-3555B dB meter (or equivalent) with 3 kHz flat filtering.

2

PIN DESCRIPTIONS

V_{DD}, POSITIVE POWER SUPPLY (PIN 22)
This is nominally 5.0 V.

V_{SS}, NEGATIVE POWER SUPPLY (PIN 12)
This is usually 0 volts.

Tx Car, TRANSMIT CARRIER (PIN 1)
The transmit carrier output is a 16 step digitally-synthesized sine wave with an amplitude of 0.1 V_{DD} (p-p) (±10%) and offset by a dc bias of 0.5 V_{DD} (±10%). The output load should be 10 kilohms or greater.

$\overline{CTS}$, CLEAR TO SEND (PIN 2)
The clear to send output goes low in response to a high-to-low translation of $\overline{RTS}$ following a selected delay (see CTA, CTB, CTC pin description). This output goes high immediately after loss of $\overline{RTS}$. During the time following activation of $\overline{RTS}$ and before the activation of $\overline{CTS}$, Tx Data should be held in the mark condition.

CTA, CLEAR TO SEND SELECT A (PIN 5)
CTB, CLEAR TO SEND SELECT B (PIN 4)
CTC, CLEAR TO SEND SELECT C (PIN 3)

For delay times for clear to send delay select inputs, see Table 1.

$\overline{RTS}$, REQUEST TO SEND (PIN 6)
The request to send input controls data transmission from the modulator. A low level enables the modulator output and a high level will disable the modulator. See Figure 1.

STO, SOFT TURN OFF INPUT (PIN 7)
Activation of STO causes a 900 Hz tone to be transmitted and $\overline{CTS}$ to remain active for 20 ms following the loss of $\overline{RTS}$. See Figure 5.

Rx Data, RECEIVE DATA (PIN 8)
The receive data output is the serial data output from the demodulator. Rx Data is clamped high when $\overline{CD}$ is not active.

$\overline{CD}$, CARRIER DETECT (PIN 9)
When carrier detect input is high (1), the Rx Data output will be clamped to a high state. When carrier detect is low (0), Rx Data output demodulates the Rx carrier input signal.

DMO, DEMODULATOR OUTPUT (PIN 10)
The demodulator output is the output of the differential delay detector. It is used for production testing of the demodulator. In normal operation, this pin should be left open.

Rx Car, RECEIVER CARRIER (PIN 11)
The receiver carrier input is the FSK input to the demodulator. This signal should be the hard-limited output of the receive filter, nominally 50%.

X_{in}, OSCILLATOR INPUT (PIN 13)
X_{out}, OSCILLATOR OUTPUT (PIN 14)
X_{in} should be driven from either an AT-cut crystal or a digital signal source at 3.6864 MHz ±0.01%. When driven by a crystal, a 15 megohm resisitor should be connected from X_{in} to X_{out} in parallel with the crystal.

MODE (PIN 15)
The mode pin selects the pair of frequencies used during modulation and demodulation. A "0" on this pin selects forward channel operation; i.e. high-speed transmit and low-speed receive. A "1" on this pin selects reverse channel operation; i.e. low-speed transmit and high-speed receive.

ST, SELF TEST (PIN 16)
When a high level is placed on this pin, the demodulator is switched to the modulator frequencies and baud rate (as determined by Mode and Type pins). The modulator should be looped back through the receive filter to the demodulator for self test (echo back).

N.C. NO CONNECTION (PIN 17)
This pin is not bonded internally and should be left open in normal operation.

TYPE (PIN 18)
This pin is used to select Bell 202 type operation and CCITT V.23 operation. When the type input pin is a "1", Bell operation is selected. When the type input pin is a "0", the CCITT standard is selected.

An Bk, ANSWER BACK (PIN 19)
The answer back input causes the answer back tone to be transmitted. The answer back tone is 2025 Hz for the Bell mode and 2100 Hz for the CCITT modes. When a high level is placed on the An Bk input pin, the Tx Car pin will output an answer back tone and $\overline{CTS}$ will go to a high state, regardless of the state of $\overline{RTS}$ (see Figure 1).

Tx Data, TRANSMIT DATA (PIN 20)
The transmit data input is the serial input to the modulator. A high level causes a mark frequency to be transmitted, a low level causes a space frequency to be transmitted.

Tx Test, TRANSMIT TEST (PIN 21)
The transmit test output is a square wave representation of the modulator transmit frequency. It is used for test purposes and should be left open in normal operation.

FIGURE 1 — An Bk AND $\overline{RTS}$-$\overline{CTS}$ TIMING

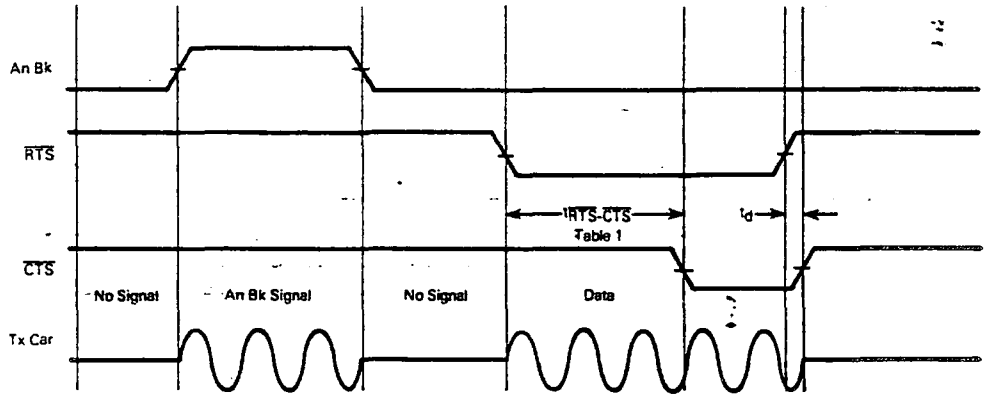


TABLE 1 — $\overline{RTS}$-$\overline{CTS}$ DELAY TIMES

CTC	CTB	CTA	Delay*
0	0	0	0 ms
0	0	1	26.7 ms
0	1	0	40.0 ms
0	1	1	60.0 ms
1	0	0	133.3 ms
1	0	1	213.3 ms
1	1	0	266.7 ms
1	1	1	426.6 ms

*All delays are ±1.7 ms.

TABLE 2 — OPERATING MODES

Type	Mode	Transmit Data	Transmit Frequency Spec	Transmit Frequency Actual	Answer Back Tone	Application
0	0	0	2100	2099.32	2100	CCITT V.23 75 Baud Receive 1200 Baud Transmit Forward Channel
		1	1300	1299.86		
0	1	0	450	450	2100	CCITT V.23 1200 Baud Receive 75 Baud Transmit Reverse Channel
		1	390	390.5		
1	0	0	2200	2199.52	2025	U.S. 150 Baud Receive 1200 Baud Transmit (Bell 202) Forward Channel
		1	1200	1200		
1	1	0	510	509.73	390	U.S. 1200 Baud Receive (Bell 202) 450 Baud Transmit Reverse Channel
		1	390	390.5		

Data = 0 = Space
= 1 = Mark

*Crystal Frequency = 3.6864 MHz

2

FIGURE 2 — STO TIMING

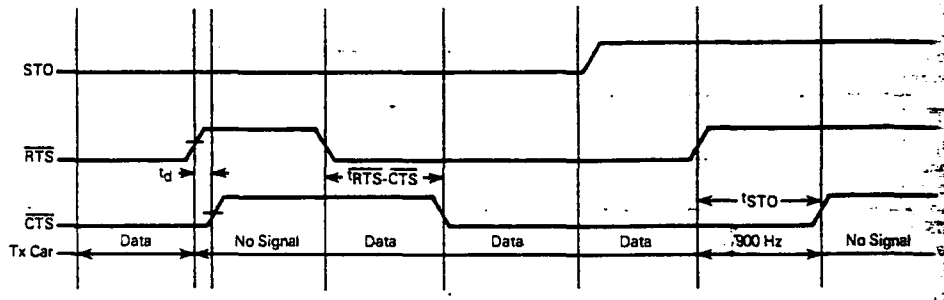
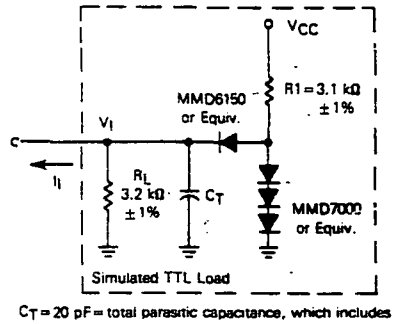
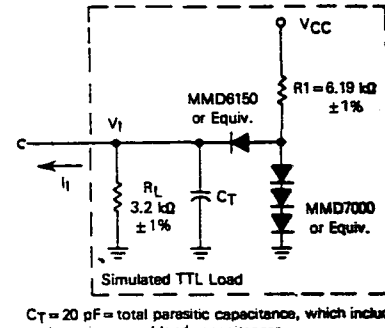


FIGURE 3 — OUTPUT TEST LOAD A



C_T = 20 pF = total parasitic capacitance, which includes probe, wiring, and load capacitances.

FIGURE 4 — OUTPUT TEST LOAD B



C_T = 20 pF = total parasitic capacitance, which includes probe, wiring, and load capacitances.

FIGURE 5 — TYPICAL MEDIUM-SPEED MODEM APPLICATION

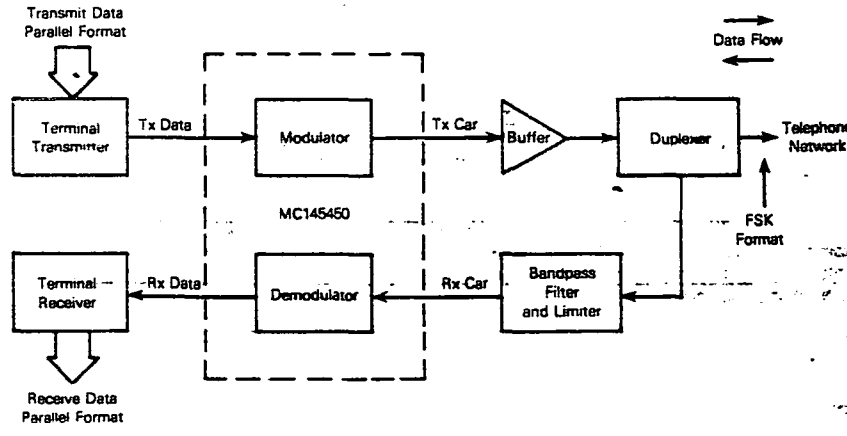
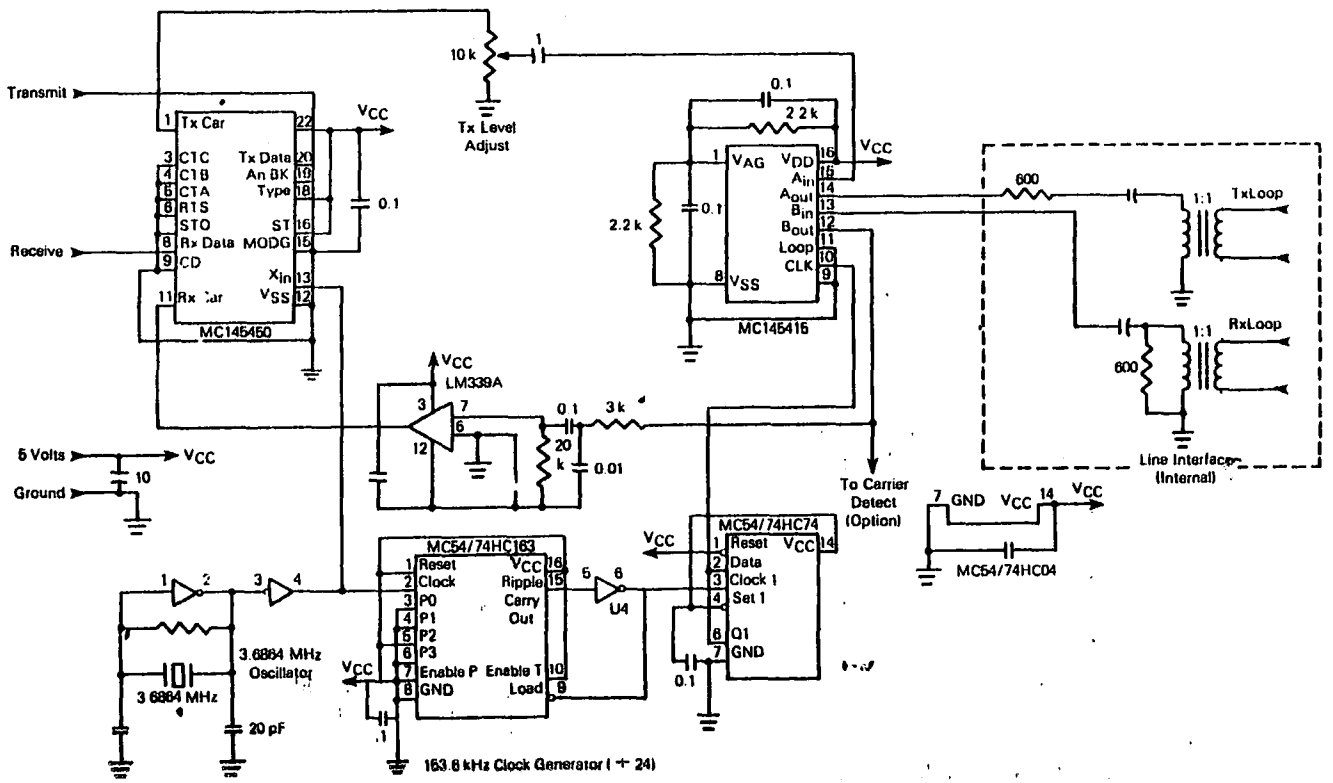


FIGURE 6 — TYPICAL 1200 BAUD 4 WIRE MODEM APPLICATION



MC34012-1
MC34012-2
MC34012-3

2

Advance Information

TELEPHONE TONE RINGER

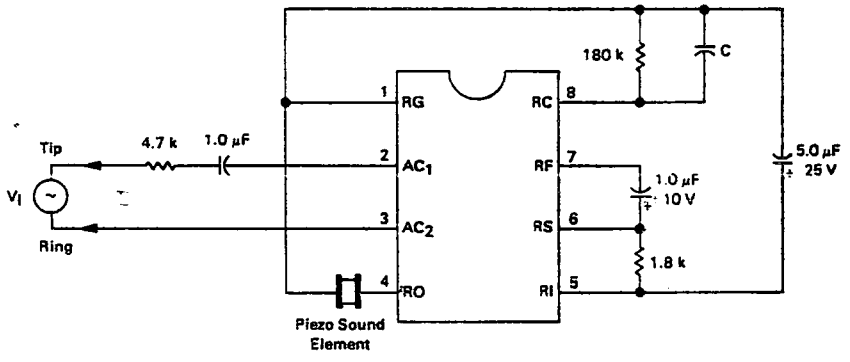
- Complete Telephone Bell Replacement Circuit with Minimum External Components
- On-Chip Diode Bridge and Transient Protection
- Direct Drive for Piezoelectric Transducers
- Base Frequency Options—MC34012-1: 1.0 kHz
  MC34012-2: 2.0 kHz
  MC34012-3: 500 Hz
- Input Impedance Signature Meets Bell and EIA Standards
- Rejects Rotary Dial Transients

TELEPHONE TONE RINGER

BIPOLAR LINEAR/I^2L

PLASTIC PACKAGE
CASE 626

APPLICATION CIRCUIT



MC34012-1: C = 1000 pF
MC34012-2: C = 500 pF
MC34012-3: C = 2000 pF

This document contains information on a new product. Specifications and information herein are subject to change without notice.

APPLICATION CIRCUIT PERFORMANCE

Characteristic	Typical Value	Units
Output Tone Frequencies MC34012-1 MC34012-2 MC34012-3 Warble Frequency	 832/1040 1664/2080 416/520 13	Hz
Output Voltage ($V_I \geq 60\ V_{rms}$, 20 Hz)	20	V_{p-p}
Output Duty Cycle	50	%
Ringing Start Input Voltage (20 Hz)	36	V_{rms}
Ringing Stop Input Voltage (20 Hz)	28	V_{rms}
Maximum ac Input Voltage (≤ 68 Hz)	150	V_{rms}
Impedance When Ringing $V_I = 40\ V_{rms}$, 15 Hz $V_I = 130\ V_{rms}$, 23 Hz	 20 10	kΩ
Impedance When Not Ringing $V_I = 10\ V_{rms}$, 24 Hz $V_I = 2.5\ V_{rms}$, 24 Hz $V_I = 10\ V_{rms}$, 5.0 Hz $V_I = 3.0\ V_{rms}$, 200–3200 Hz	 28 >1.0 55 >1.0	 kΩ MΩ kΩ MΩ
Maximum Transient Input Voltage (T ≤ 2.0 ms)	1500	V

PIN DESCRIPTIONS

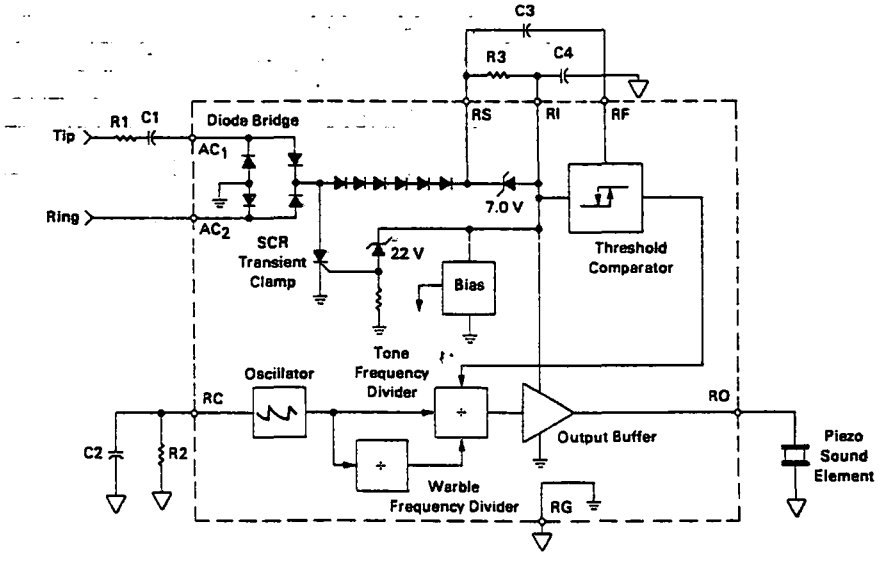
Name	Description
AC_1, AC_2	The input terminals to the full-wave diode bridge. The ac ringing signal from the telephone line energizes the ringer through this bridge.
RS	The positive output of diode bridge to which an external current sense resistor is connected.
RI	The positive supply terminal for the oscillator, frequency divider and output buffer circuits.
RF	The terminal for the filter capacitor used in detection of ringing input signals.
RO	The tone ringer output terminal through which the sound element is driven.
RG	The negative output of the diode bridge and the negative supply terminal of the tone generating circuitry.
RC	The oscillator terminal for the external resistor and capacitor which control the tone ringer frequencies.

2

ELECTRICAL CHARACTERISTICS (T_A = 25°C)

Characteristic	Test	Symbol	Min	Typ	Max	Units
Ringing Start Voltage ($V_{Start} = V_I$ @ Ring Start)						Vdc
$V_I > 0$	1a	$V_{Start(+)}$	31	34.5	38	
$V_I < 0$	1b	$V_{Start(-)}$	-31	-34.5	-38	
Ringing Stop Voltage ($V_{Stop} = V_I$ @ Ring Stop)	1c	V_{Stop}				Vdc
MC34012-1			16	20	25	
MC34012-2			13	18	22	
MC34012-3			16	20	25	
Output Frequencies (V_I = 50 V)	1d					Hz
MC34012-1 High Tone		f_H	967	1040	1113	
Low Tone		f_L	774	832	890	
Warble Tone		f_W	12	13	14	
MC34012-2 High Tone		f_H	1934	2080	2226	
Low Tone		f_L	1548	1664	1780	
Warble Tone		f_W	12	13	14	
MC34012-3 High Tone		f_H	967	1040	1113	
Low Tone		f_L	774	832	890	
Warble Tone		f_W	24	26	28	
Output Voltage (V_I = 50 V)	6	V_O	19	20	23	V_{p-p}
Output Short-Circuit Current	2	I_O	35	50	80	mA_{p-p}
Input Diode Voltage (I_I = 1.0 mA)	3	V_D	4.6	5.1	5.6	Vdc
Input Voltage—SCR Off (I_I = 30 mA)	4a	V_{off}	37	42	47	Vdc
Input Voltage—SCR On (I_I = 100 mA)	4b	V_{on}	3.2	4.2	6.0	Vdc
Threshold Filter Resistance $R_{RF} = 2.0\ V/I_{RF}$	5	R_{RF}	30	50	80	kΩ

BLOCK DIAGRAM



CIRCUIT DESCRIPTION

The MC34012 Tone Ringer derives its power supply by rectifying the ac ringing signal. It uses this power to activate a tone generator and drive a piezo-ceramic transducer. The tone generation circuitry includes a relaxation oscillator and frequency dividers which produce high and low frequency tones as well as the tone warble frequency. The relaxation oscillator frequency f_o is set by resistor R2 and capacitor C2 connected to pin RC. The oscillator will operate with f_o from 1.0 kHz to 10 kHz with the proper choice of external components (See Figure 1).

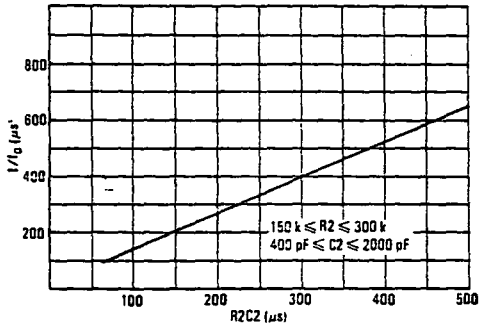
The frequency of the tone ringer output signal at pin RO alternates between $f_o/4$ to f_o 5. The warble rate at which the frequency changes is $f_o/320$ for the MC34012-1, $f_o/640$ for the MC34012-2, or $f_o/160$ for the MC34012-3. With a 4.0 kHz oscillator frequency, the MC34012-1 produces 800 Hz and 1000 Hz tones with a 12.5 Hz warble rate. The MC34012-2 generates 1600 Hz and 2000 Hz tones with a similar 12.5 Hz warble frequency from an 8.0 Hz oscillator frequency. The MC34012-3 will produce 400 Hz and 500 Hz tones with a 12.5 Hz warble rate from a 2.0 kHz oscillator frequency. The tone ringer output circuit can source or sink 20 mA with an output voltage swing of 20 volts peak-to-peak. Volume control is readily implemented by adding a variable resistance in series with the piezo transducer.

Input signal detection circuitry activates the tone ringer output when the ac line voltage exceeds programmed threshold level. Resistor R3 determines the ringing signal amplitude at which an output signal will be generated at RO. The ac ringing signal is rectified by the internal diode bridge. The rectified input signal produces a current through R3 which is input at terminal RI. The voltage across resistor R3 is filtered by capacitor C3 at the input to the threshold circuit. When the voltage on capacitor C3 exceeds 1.7 volts, the threshold comparator enables the tone ringer output. Line transients produced by pulse dialing telephones do not charge capacitor C3 sufficiently to activate the tone ringer output.

Capacitors C1 and C4 and resistor R1 determine the 10 volt, 24 Hz signature test impedance. C4 also provides filtering for the output stage power supply to prevent droop in the square wave output signal. Six diodes in series with the rectifying bridge provide the necessary non-linearity for the 2.5 volt, 24 Hz signature tests.

An internal shunt voltage regulator between the RI and RG terminals provides dc voltage to power output stage, oscillator, and frequency dividers. The dc voltage at RI is limited to approximately 22 volts in regulation. To protect the IC from telephone line transients, an SCR is triggered when the regulator current exceeds 50 mA. The SCR diverts current from the shunt regulator and reduces the power dissipation within the IC.

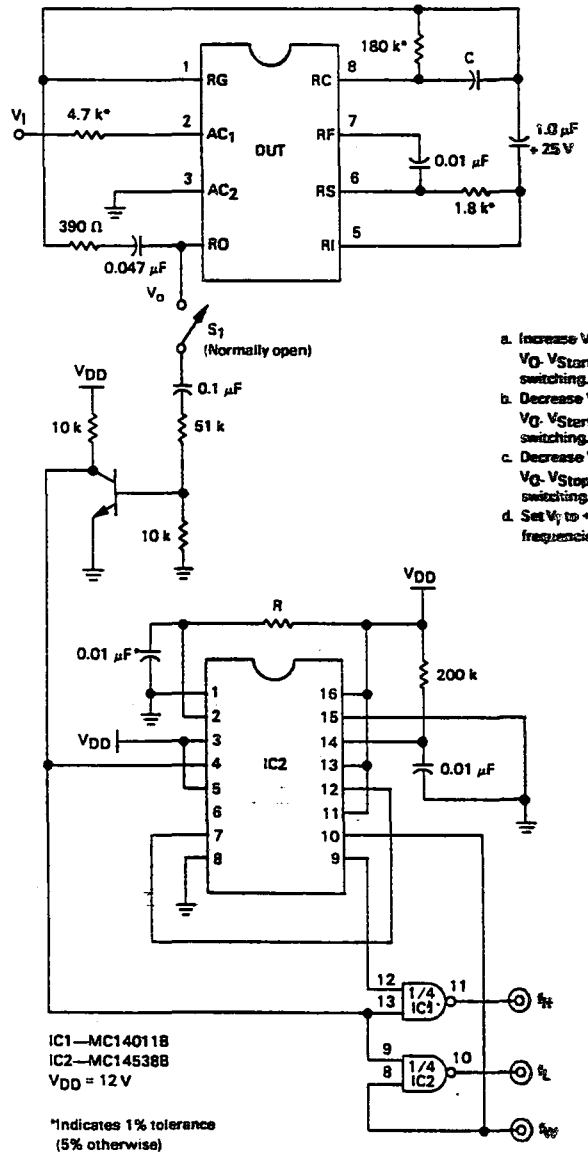
FIGURE 1 — OSCILLATOR PERIOD ($1/f_o$) versus OSCILLATOR R2 C2 PRODUCT



EXTERNAL COMPONENTS

R1	Line input resistor. R1 controls the tone ringer input impedance. It also influences ringing threshold voltage and limits current from line transients. (Range: 2.0 kΩ to 10 kΩ).
C1	Line input capacitor. C1 ac couples the tone ringer to the telephone line and controls ringer input impedance at low frequencies. (Range: 0.4 μF to 2.0 μF).
R2	Oscillator resistor. (Range: 150 kΩ to 300 kΩ).
C2	Oscillator capacitor. (Range: 400 pF to 2000 pF).
R3	Input current sense resistor. R3 controls the ringing threshold voltage. Increasing R3 decreases the ring-start voltage. (Range: 0.8 kΩ to 2.0 kΩ).
C3	Ringing threshold filter capacitor. C3 filters the ac voltage across R3 at the input of the ringing threshold comparator. It also provides dialer transient rejection. (Range: 0.5 μF to 5.0 μF).
C4	Ringer supply capacitor. C4 filters supply voltage for the tone generating circuits. It also provides an ac current path for the 10 V_{rms} ringer signature impedance. (Range: 1.0 μF to 10 μF).

FIGURE 2 — TEST ONE

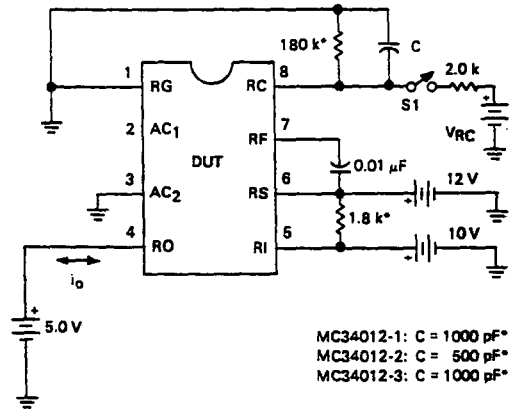


MC34012-1: C = 1000 pF*
MC34012-2: C = 500 pF*
MC34012-3: C = 1000 pF*

a. Increase V_I from +30 volts while monitoring V_O. $V_{Start}(+)$ equals V_I when V_O commences switching.
b. Decrease V_I from −30 volts while monitoring V_O. $V_{Start}(-)$ equals V_I when V_O commences switching.
c. Decrease V_I from +40 volts while monitoring V_O. V_{Stop} equals V_I when V_O ceases switching.
d. Set V_I to +50 volts. Close S1. Measure frequencies f_H, f_L, and f_W.

MC34012-1: R = 110 kΩ*
MC34012-2: R = 55 kΩ*
MC34012-3: R = 110 kΩ*

FIGURE 3 — TEST TWO

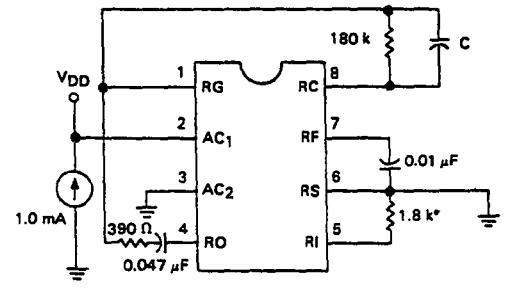


With V_{RC} = 3.5 volts, close S1. Measure output current at Pin 4 (I_{O1}). Repeatedly switch V_{RC} between 3.5 volts and 0 volts until Pin 4 current changes polarity. Measure this opposite polarity current (I_{O2}).
Calculate:

$$I_O = |I_{O1}| + |I_{O2}|$$

*Indicates 1% tolerance (5% otherwise)

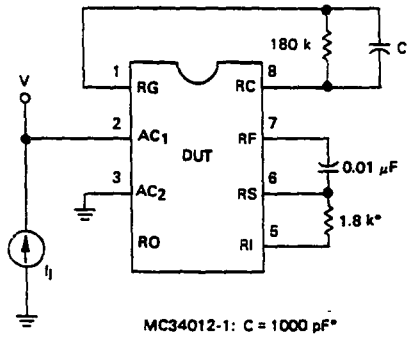
FIGURE 4 — TEST THREE



Measure voltage at Pin 2.

*Indicates 1% tolerance (5% otherwise)

FIGURE 5 — TEST FOUR

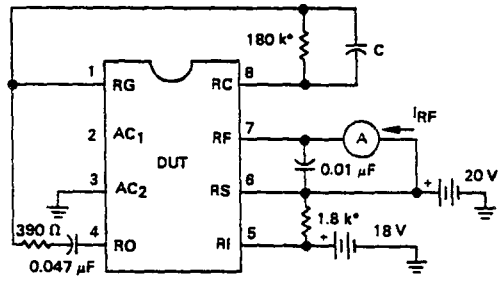


a. Set I_I to 30 mA. Measure voltage at Pin 2 (V_{off}).
b. Set I_I to 100 mA. Measure voltage at Pin 2 (V_{on}).

MC34012-1: C = 1000 pF*
MC34012-2: C = 500 pF*
MC34012-3: C = 1000 pF*

*Indicates 1% tolerance (5% otherwise)

FIGURE 6 — TEST FIVE



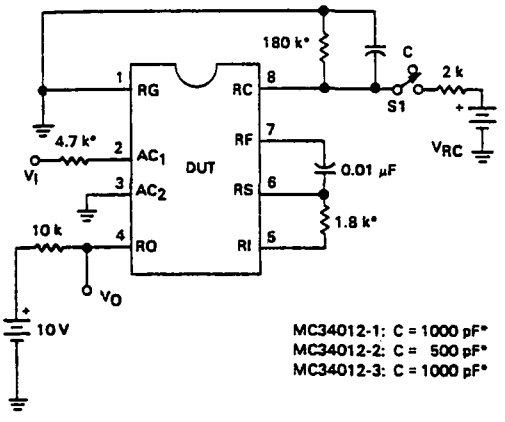
Measure current into Pin 7 (I_{RF}).
Calculate:

R_{RF} = 2 volts ÷ I_{RF}

MC34012-1: C = 1000 pF*
MC34012-2: C = 500 pF*
MC34012-3: C = 1000 pF*

*Indicates 1% tolerance (5% otherwise)

FIGURE 7 — TEST SIX



MC34012-1: C = 1000 pF*
MC34012-2: C = 500 pF*
MC34012-3: C = 1000 pF*

With V_{RC} = 3.5 volts, close S1. Set V_I to 50 volts. Measure dc voltage at Pin 4 (V_{O1}). Repeatedly change V_{RC} between 3.5 volts and 0 volts until Pin 4 changes state. Measure the new voltage at Pin 4 (V_{O2}). Calculate the peak-to-peak output voltage:

$$V_{Op\text{-}p} = |V_{O2} - V_{O1}|$$

*Indicates 1% tolerance (5% otherwise)

เอกสารอ้างอิง

1. ชัยพจน์ เนตรศิริ, "เทคโนโลยีของบัตร ATM" คอมพิวเตอร์อีเลคทรอนิคส์เวิลด์. 138 (2535) : 36-44.
2. สมภพ ภูริวิกรัยพงศ์, "เครื่องอ่านรหัสบัตรแม่เหล็ก" วิทยานิพนธ์วิศวกรรมศาสตรมหาบัณฑิต, คณะวิศวกรรมศาสตร์, สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง.
3. สุพจน์ ปุณณชัยยะ, MODEM. พิมพ์ครั้งที่ 3, กรุงเทพมหานคร : บริษัท ด่านสุทธาการพิมพ์ จำกัด, 2535.
4. Bates, Paul, Practical Digital and Data Communications with LSI Applications, Reston-Prentice Hall, Inc.,1987.
5. Driscoll, Frederick F., Data Communications, pp. 79-129. Saunders College Publishing, 1992.

ประวัติผู้ทำโครงงานพิเศษ

นายธวัชชัย เลี่ยมมนัสสกุล เกิดวันที่ ๑๒ พฤษภาคม ๒๕๑๕ สำเร็จการศึกษาระดับมัธยมศึกษาตอนต้นจาก โรงเรียนนิพัทธ์วิทยา และ มัธยมศึกษาตอนปลาย จากโรงเรียนวัดสุทธิวราราม จังหวัดกรุงเทพมหานคร และเข้าศึกษาต่อในระดับปริญญาตรี ภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง และสำเร็จการศึกษาในปีการศึกษา ๒๕๓๖

นายนิพนธ์ จันทร์ตรี เกิดวันพุธที่ ๒๘ มิถุนายน พ.ศ. ๒๕๑๕ สำเร็จการศึกษาระดับมัธยมศึกษาตอนต้นและมัธยมศึกษาตอนปลายจากโรงเรียนชลราษฎรอำรุง จังหวัดชลบุรี แล้วได้รับโควตาเข้าศึกษาต่อในระดับปริญญาตรี ภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง และสำเร็จการศึกษาในปีการศึกษา ๒๕๓๖