สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

โมดูลบันทึกข้อมูลระยะไกล

นายบุญชัย พจนาสมสมาน
นายสมนึก วิเศษพานิชกิจ

วพ.
ป426ม
2536

เลขหมู่.....
เลขทะเบียน.....
วัน,เดือน,ปี.....

b12555393

โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต
ภาควิชาฟิสิกส์ประยุกต์
คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

2526

REMOTE DATA ACQUISITION MODULE

Mr.Boonchai Photchanasomsaman

Mr.Somnuk Wisadpanichkij

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 Monkut's Institute of Technology Ladkrabang

1993

หัวข้อโครงงานพิเศษ	Remote Data Acquisition Module
โดย	นายบุญชัย พจนาสมสมาน
	นายสมนึก วิเศษพานิชกิจ
ภาควิชา	ฟิสิกส์ประยุกต์
อาจารย์ที่ปรึกษา	อาจารย์วิชิต ศิริโชติ

ภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ได้อนุมัติให้นับรายงานโครงงานพิเศษฉบับนี้ เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

..... หัวหน้าภาควิชาฟิสิกส์ประยุกต์

(ผศ. ปรีชา เทียนสมประสงค์)

คณะกรรมการโครงงานพิเศษ

..... กรรมการ

(อ.วิชิต ศิริโชติ)

..... กรรมการ

(ผศ.เครือวัลย์ ศีตะจิตต์)

..... กรรมการ

(อ. ดร.ฐิตินัย แก้วแดง)

ลิขสิทธิ์ของภาควิชาฟิสิกส์ประยุกต์

คณะวิทยาศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

หัวข้อโครงงานพิเศษ	Remote Data Acquisition Module
นักศึกษา	นายบุญชัย พจนาสมสมาน
	นายสมนึก วิเศษพานิชกิจ
ภาควิชา	ฟิสิกส์ประยุกต์
อาจารย์ที่ปรึกษา	อาจารย์วิชิต ศิริโชติ
ปีการศึกษา	2536

บทคัดย่อ

Remote Data Acquisition Module (RDAM) เป็นส่วนที่ออกแบบและสร้างขึ้นมาเพื่อการเก็บบันทึกสัญญาณอะนาล็อก โดย RDAM ประกอบไปด้วย ไมโครโปรเซสเซอร์, radio modem และเครื่องรับส่งสัญญาณวิทยุ ซึ่งสัญญาณอะนาล็อกจะถูกแปลงด้วยตัว ICL7109 ซึ่งเป็น A/D ขนาด 12-bit และมีช่องรับสัญญาณอินพุทได้ทั้งหมด 8 ช่อง ส่วนตัวไมโครโปรเซสเซอร์เป็นของบริษัท intel โดยใช้เบอร์ 8751 และใช้ร่วมกับ NVSRAM ขนาด 32 กิโลไบต์สำหรับเป็นที่เก็บข้อมูล ซึ่งข้อมูลที่เก็บรวบรวมได้จะส่งไปยังคอมพิวเตอร์ศูนย์กลาง โดยใช้ชิปโมเด็มเบอร์ MC 145442 (CCITT V.21) ซึ่งต่อร่วมกับเครื่องรับส่งสัญญาณวิทยุ โปรโตคอลที่ได้ออกแบบได้นั้น จะใช้คำสั่งเป็นตัวอักษรตัวเดียว และต่าแหน่งของสถานีย่อยซึ่งอ้างได้ถึง 256 สถานี ในส่วนของการประยุกต์ใช้งานของ RDAM จะใช้ในงาน Centralized Remote Monitoring System (CRMS) สำหรับการตรวจสอบคุณภาพของน้ำ

Special Project Title : Remote Data Acquisition Module

Name : Mr.Boonchai Photchanasomsaman

: Mr.Somnuk Wisadpanichkij

Special Project Advisor : Mr.Wichit Sirichote

Department : Applied Physics

Acadamic Year : 2536

Abstract

A Remote Data Acquicition Module,RDAM, is designed and built for recording an analog signal remotely. The RDAM consists of a micro power data logger,a radio MODEM and a 2-meter transceiver. The analog signal converter, ICL7109, provides a 12-bit resolution and eight channels input signal. The micropower data logger is the intel 8751 based with a 32 KB NVSRAM for data retentions. Recorded data is sent to the central computer by using a single chip 300 baud MODEM, MC 145442 CCITT V2.1, connected to the 2-meter FM transceiver. The designed protocal uses a simple single character command and provides the RDAM station upto 256 stations. Application of the RDAM will be used in the Centralized Remote Monitoring System, CRMS,for monitoring of water quality.

กิตติกรรมประกาศ

โครงงานพิเศษฉบับนี้สำเร็จขึ้นมาได้ด้วยความช่วยเหลือของบุคคลหลาย ๆ ฝ่ายดังต่อไปนี้

อ. วิชิต ศิริโชติ

ที่ได้กรุณาให้คำปรึกษา และคำแนะนำ ตลอดจนให้ความรู้เกี่ยวกับโครงงานพิเศษ

ขอขอบคุณ

คุณกิตติชัย อภินทนาพงศ์
คุณพงษ์พันธ์ ศรจิตติ
คุณธิปไตย ตันท์ประพันธ์
ชุมนุมวิทยุสมัครเล่น คณะวิทยาศาสตร์ สจล.

ที่ได้ให้ความเอื้อเฟื้อทางด้านเครื่องมือสื่อสารที่จำเป็นสำหรับใช้ในการทดลอง

รวมทั้งเพื่อน ๆ พี่ ๆ น้อง ๆ ทุกคนที่ได้ให้ความช่วยเหลือมาโดยตลอด

และบุคคลที่สำคัญที่สุดของผู้เขียน คือ พ่อและแม่ ที่ได้ให้กำเนิด และคอยเลี้ยงดู อบรมสั่งสอน จนประสบความสำเร็จในการศึกษา ผู้เขียนขอกราบขอบพระคุณอย่างสูง

บุญชัย พจนาสมสมาน
สมนึก วิเศษพานิชกิจ
18 มีนาคม 2537

สารบัญ

หน้า

บทคัดย่อโครงงานพิเศษภาษาไทย ..... ก

บทคัดย่อโครงงานพิเศษภาษาอังกฤษ ..... ข

กิตติกรรมประกาศ ..... ค

สารบัญตาราง ..... ง

สารบัญรูป ..... จ

บทที่ 1 บทนำ 1

บทที่ 2 ทฤษฎี 3

2.1 ระบบการสื่อสารข้อมูล 3

2.1.1 การส่งแบบซิงโครนัส 3

2.1.2 การส่งแบบอะซิงโครนัส 6

2.1.3 การส่งข้อมูลแบบขนานและอนุกรม 7

2.2 การสื่อสารแบบดิจิตอล 10

2.3 มาตรฐานสากลในการอินเตอร์เฟส 11

2.4 โปรโตคอลและเน็ทเวิร์คฟังก์ชัน 17

2.5 มาตรฐาน RS-232C 19

2.5.1 คุณสมบัติทางไฟฟ้าของ RS-232C 19

2.5.2 สัญญาณและการอินเตอร์เฟสทางแมคคานิคของ RS-232C 27

2.5.3 โปรโตคอลสำหรับการแลกเปลี่ยนข้อมูลของ RS-232C 28

2.6 PACKET RADIO-INTEGRATED DIGITAL COMMUNICATION 30

2.7 MODEM 38

2.7.1 โมเด็มทรานสมิทเตอร์ 38

2.7.2 โมเด็มรีซีฟเวอร์ 39

บทที่ 3 การออกแบบและการสร้าง 51

3.1 ส่วนของ packet modem 51

3.2 ส่วนควบคุมย่อย 62

บทที่ 4 การทดลอง 76
4.1 การทดลองติดต่อสื่อสารระหว่างไมโครคอมพิวเตอร์ 2 ชุด 76
4.1.1 การรับส่งข้อความ 77
4.1.2 การรับส่งแบบไฟล์ 79
4.2 การทดลองการติดต่อสื่อสารระหว่างส่วนควบคุมย่อยกับไมโครคอมพิวเตอร์ 80
บทที่ 5 สรุปและข้อเสนอแนะ 82
ภาคผนวก 84
บรรณานุกรม 85

สารบัญตาราง

หน้า

ตารางที่ 2.1 แสดงการเปรียบเทียบของการส่งข้อมูลแบบขนานและแบบอนุกรม 9
ตารางที่ 2.2 แสดงคุณลักษณะทางไฟฟ้าของมาตรฐาน RS-232C, RS-422A,RS-423A 26
ตารางที่ 2.3 แสดงสัญญาณทั้ง 25 ของมาตรฐานการอินเตอร์เฟสของ RS-232C 27
ตารางที่ 2.4 แสดงอัตราบิท อัตราโบด และอัตราสัญญาณในสายส่ง 49
ตารางที่ 3.1 แสดงค่า Transmit level 53
ตารางที่ 3.2 แสดงค่าความถี่ของ mode ต่าง ๆ ของ MC 145442 53
ตารางที่ 3.3 แสดงโหมดการทำงานต่าง ๆ ของ MC 145442 55
ตารางที่ 3.4 แสดงหน้าที่การทำงานต่าง ๆ ของ ICL7109 64
ตารางที่ 3.5 แสดงเวลาของขนาดของสัญญาณต่าง ๆ ของโหมดโดยตรง 68

สารบัญรูป

หน้า

รูปที่ 2.1 การให้สัญญาณนาฬิกาเพื่อการรับส่งแบบซิงโครนัสบิต 4
รูปที่ 2.2 การส่งแบบซิงโครนัสที่มี SYN หลายตัว 5
รูปที่ 2.3 การส่งแบบอะซิงโครนัส 6
รูปที่ 2.4 แสดงรูปร่างของคอนเนคเตอร์แบบ DB-25 20
รูปที่ 2.5 แสดงการเปรียบเทียบระดับลอจิกแบบ TTL และ แบบ RS-232C 21
รูปที่ 2.6 แสดงการติดต่อสื่อสารระหว่าง DTE และ DCE โดยผ่านข่ายสายโทรศัพท์ .. 22
รูปที่ 2.7 แสดงการเปลี่ยนระดับลอจิกของ RS ไปเป็นแบบ TTL 24
รูปที่ 2.8 แสดงการเปรียบเทียบระดับลอจิกของ RS-422A และ RS-423A 25
รูปที่ 2.9 แสดงการอินเตอร์เฟส RS-232C อย่างง่าย 29
รูปที่ 2.10 แสดงแบนด์วิดท์ของความถี่ที่ใช้ในโมเด็มชนิด Bell_103 35
รูปที่ 2.11 แสดงสัญญาณของการโมดูเลท 36
รูปที่ 2.12 โครงสร้างในการกำเนิดสัญญาณ FSK 38
รูปที่ 2.13 โครงสร้างในการทำงานของโมเด็มรีซีฟเวอร์ 39
รูปที่ 2.14 การทำงานของดีเทคเตอร์ 41
รูปที่ 2.15 การต่อตัวต้านทานฟีดแบ็ค 42
รูปที่ 2.16 แสดงลักษณะของสัญญาณตามวิธีแอมปลิจูดโมดูเลชั่น 43
รูปที่ 2.17 ฟรีเควนซี่โมดูเลชั่น 45
รูปที่ 2.18 หลักการมุมเฟสและ PM 46
รูปที่ 2.19 แสดงสัญญาณที่ได้จากเทคนิค FSK 47
รูปที่ 2.20 PAM 49
รูปที่ 3.1 แสดง block diagram ของ MC 145442 52
รูปที่ 3.2 แสดงวงจรภายในของ driver summing 54
รูปที่ 3.3 เป็นวงจร carrier detect 56
รูปที่ 3.4 แสดงวงจรภายในส่วน filter bias 57
รูปที่ 3.5a โหมด originate 57
รูปที่ 3.5b โหมด answer 57

รูปที่ 3.5c โหมด originate และ analog loopback 58
รูปที่ 3.6 วงจรของ MC 145442 59
รูปที่ 3.7 วงจรบัฟเฟอร์ 60
รูปที่ 3.8 วงจร MAX 232 60
รูปที่ 3.9 วงจรควบคุมรีเลย์และส่งสัญญาณ CTS กลับ 61
รูปที่ 3.10 แสดง block diagram ส่วนควบคุมย่อย 62
รูปที่ 3.11 แสดงการต่อวงจร test circuit ของ ICL7109 63
รูปที่ 3.12 แสดงส่วน digital section 66
รูปที่ 3.13 เป็น timing ของ status 67
รูปที่ 3.14 แสดงการทำงานของ RUN/HOLD 67
รูปที่ 3.15 แสดง timing diagram ของการติดต่อโดยตรง 68
รูปที่ 3.16 แสดง timign digaram ของการติดต่อแบบ handshake 69
รูปที่ 3.17 แสดงการต่อวงจร oscilator แบบใช้ RC 69
รูปที่ 3.18 แสดงการต่อวงจร oscilator แบบใช้คริสตอล 70
รูปที่ 3.19 แสดงการต่อไมโครคอนโทรลเลอร์เบอร์ 8751 กับแรม 71
รูปที่ 3.20 แสดงการต่อ ICL7109 71
รูปที่ 3.21 แสดงการต่อกับ dip switch 72
รูปที่ 3.22 วงจรส่วน multiplex analog switch เบอร์ 4051 72
รูปที่ 3.23 แสดงส่วน supply 73
รูปที่ 3.24 แสดงวงจรส่วน TNC 73
รูปที่ 4.1 แสดงการต่ออุปกรณ์ในการรับส่งข้อมูล 77
รูปที่ 4.2 แสดงลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่ส่งข้อความ 78
รูปที่ 4.3 แสดงลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่รับข้อความ 78
รูปที่ 4.4 แสดงลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่ส่งมูลแบบเป็นไฟล์(Text) 79
รูปที่ 4.5 แสดงลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่รับมูลแบบเป็นไฟล์(Text) 79
รูปที่ 4.6 แสดงการต่ออุปกรณ์ทั้งหมดของส่วนควบคุมย่อยที่ทดลอง 80
รูปที่ 4.7 แสดงลักษณะบนจอมอนิเตอร์ที่เกิดขึ้นในการส่งคำสั่งให้กับส่วนควบคุมย่อย ... 81
รูปที่ 4.8 แสดงลักษณะบนจอมอนิเตอร์หลังจากได้รับข้อมูลที่ส่งกลับจากส่วนควบคุมย่อย . 81

บทที่ 1

บทนำ

จากความเจริญก้าวหน้าทางด้านเทคโนโลยีในปัจจุบัน ทำให้คอมพิวเตอร์เข้ามามีบทบาทมากขึ้น ทำให้ความต้องการในการติดต่อระหว่างคอมพิวเตอร์ด้วยกันเป็นที่ต้องการมากขึ้น โดยการติดต่อที่ใช้มีด้วยกัน 2 แบบคือ การติดต่อสื่อสารแบบอนุกรมกับแบบขนาน ซึ่งการติดต่อสื่อสารแบบอนุกรมนั้นใช้จำนวนสายนำสัญญาณน้อย ทำให้ประหยัด และสะดวกสบายกว่า แต่ให้อัตราเร็วในการส่งน้อยกว่าแบบขนานมาก ในการติดต่อสื่อสารแบบอนุกรม มีอุปกรณ์อย่างหนึ่ง ที่ช่วยเพิ่มประสิทธิภาพของการรับส่งข้อมูลแบบอนุกรม ก็คือ โมเด็ม ซึ่งใช้ในการติดต่อสื่อสารระหว่างคอมพิวเตอร์ด้วยกันผ่านทางสายโทรศัพท์ จะต้องมีการใช้สายนำสัญญาณ ซึ่งได้มีการพัฒนา โมเด็มให้สามารถติดต่อกันโดยผ่านทางคลื่นวิทยุแทน เรียกอุปกรณ์นี้ว่า packet modem

ในการพัฒนาอุปกรณ์นี้ใช้หลักการเดียวกันกับโมเด็ม คือทำการเปลี่ยนสัญญาณดิจิตอลให้เป็นสัญญาณเสียงแล้วทำการโมดูเลทเข้ากับคลื่นวิทยุ เพื่อทำการส่งออกอากาศ ซึ่งสามารถนำไปประยุกต์ เพื่อใช้ในการติดต่อกับเครื่องมือวัดที่อยู่ในบริเวณต่าง ๆ ได้ โดยไม่ต้องใช้สายนำสัญญาณ โดยการติดต่อสื่อสารแบบนี้จะเป็นแบบ Half-Duplex เมื่อใช้ความถี่ในการติดต่อเพียงความถี่เดียว ข้อได้เปรียบของการติดต่อสื่อสารโดยใช้คลื่นวิทยุนี้สามารถติดต่อกับอุปกรณ์ได้เป็นจำนวนมากและได้ค่อนข้างไกล

วัตถุประสงค์ของโครงงาน

1. เพื่อศึกษาการติดต่อสื่อสารข้อมูลระหว่างไมโครคอมพิวเตอร์ 16 บิต
2. เพื่อศึกษาการเชื่อมต่อไมโครคอมพิวเตอร์เข้ากับวิทยุสมัครเล่น
3. เพื่อศึกษาการควบคุมอุปกรณ์โดยใช้ไมโครคอนโทรลเลอร์
4. เพื่อศึกษาการติดต่อ และควบคุมบอร์ดไมโครคอนโทรลเลอร์ โดยใช้ไมโครคอมพิวเตอร์ 16 บิต ซึ่งผ่านทางวิทยุสมัครเล่น

ขอบเขตของโครงงาน

1. สามารถสร้าง packet modem ที่มีอัตราการรับส่งข้อมูลขนาด 300 บิตต่อวินาที
2. สามารถติดต่อระหว่างไมโครคอมพิวเตอร์ 2 ชุดได้
3. สามารถสร้างส่วนควบคุมย่อยซึ่งใช้อ่านข้อมูลจากตัวเซ็นเซอร์ที่ต่อจากภายนอกได้
4. สามารถควบคุมและรับส่งข้อมูลระหว่างส่วนควบคุมย่อยกับไมโครคอมพิวเตอร์ได้

ประโยชน์ที่ได้รับจากโครงงาน

1. นำ packet modem ที่สร้างขึ้น มาใช้ในการติดต่อสื่อสารระหว่างไมโครคอมพิวเตอร์ โดยผ่านทางวิทยุสมัครเล่น
2. นำบอร์ดควบคุมที่สร้างขึ้น มาใช้ในการรับส่งข้อมูลกับไมโครคอมพิวเตอร์ โดยผ่านทางวิทยุสมัครเล่น

ขั้นตอนการดำเนินงาน

1. ศึกษาลักษณะของโครงงาน และหาข้อมูลเกี่ยวกับระบบที่ต้องการใช้
2. ทดลองออกแบบวงจรที่ใช้ในระบบ โดยแบ่งออกเป็น
   2.1 วงจรของ packet modem
   2.2 วงจรของ ส่วนควบคุมย่อย
3. ทดสอบวงจรต่าง ๆ
4. เขียนโปรแกรมสื่อสารที่ใช้กับระบบของ packet modem
5. เขียนโปรแกรมควบคุมการทำงานของ ส่วนควบคุมย่อย
6. ออกแบบลายวงจรต่าง ๆ บนแผ่นทองแดง และประกอบเป็นเครื่องสำเร็จรูป
7. ทดสอบระบบรวม และประยุกต์ใช้งาน

บทที่ 2

ทฤษฎี

ความรู้พื้นฐานเกี่ยวกับ Remote Data Acquisition Module

2.1 ระบบการสื่อสารข้อมูล

การรับส่งเป็นรูปแบบหนึ่งของการสื่อสาร ซึ่งข้อมูลที่รับส่งกันนั้นเป็นสัญญาณไฟฟ้าในรูปเลขฐานสอง "0" หรือ "1" หากพิจารณาตามทิศทางการส่งข้อมูลภายในสายส่งแล้ว สามารถแบ่งการส่งข้อมูลออกได้เป็น 3 ชนิด

1.การส่งแบบทิศทางเดียว (one-way transmission หรือ simplex transmission) คือข้อมูลจะไหลได้ทิศทางเดียว

2.การส่งแบบทิศทางใดทิศทางหนึ่ง (either-way transmission หรือ half-duplex transmission) คือข้อมูลจะไหลได้สองทิศทาง แต่ต้องผลัดกันรับผลัดกันส่ง

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

และถ้าพิจารณารูปแบบของข้อมูลที่ส่ง จะแยกได้เป็น 2 ชนิดคือ

1.แบบซิงโครนัส (synchronous transmission)

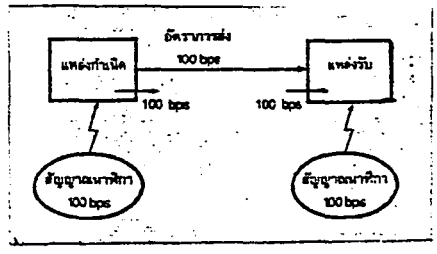
2.แบบอะซิงโครนัส (asymchronous transmission)

2.1.1 การส่งแบบซิโครนัส

เราแบ่งได้เป็น 2 แบบ คือ แบบซิงโครนัสบิตและแบบซิงโครนัสอักขระ

ซิงโครนัสบิต

เป็นการรับส่งที่ด้านรับต้องรู้ว่า จะรับบิตแรกจากสายส่งนั้นเมื่อใด และหลังจากรับมาแล้ว จะรับบิตที่ 2,3,...เมื่อใด ซึ่งสามารถกระทำโดยเพิ่มสัญญาณนาฬิกา (clock) เข้าไปที่จุดปลายทางของระบบทั้งสองดังรูปที่ 2.1



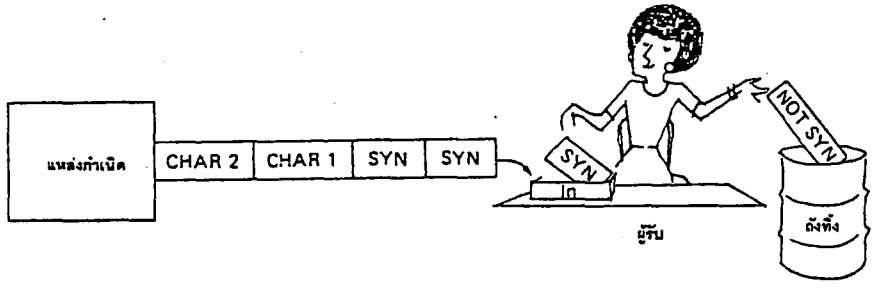
รูปที่ 2.1 การให้สัญญาณนาฬิกาเพื่อการรับส่งแบบซิงโครนัสบิต

ซึ่งในบางระบบอาจใช้วิธีส่งสัญญาณนาฬิกาจากด้านส่งไปทางด้านรับด้วย ซึ่งทางด้านรับจะนำสัญญาณที่ได้รับ มาเป็นตัวขับสัญญาณนาฬิกาของตัวมันเอง เพื่อให้สัญญาณนาฬิกาทางด้านรับและส่งเท่ากันพอดี

ซิงโครนัสอักขระ

ในการรับส่งข้อมูลตามสายนั้น แม้ว่าจะมีการจัดการกับบิตได้ดีแล้วก็ตาม แต่ก็ยังมีปัญหาอีก คือ การนำบิตของอักขระ (character) หลาย ๆ ตัวมารวมกันเป็นบล็อก ดังนั้นถึงแม้ว่าบิตต่าง ๆ จะได้รับมาอย่างถูกต้องแล้วก็ตาม เรายังต้องทราบว่า กลุ่มของบิตที่แสดงถึงตัวอักขระต่าง ๆ นั้น เริ่มต้นที่บิตใด เพื่อที่เราจะแก้ปัญหาได้ ถ้าเราทราบว่า บิตใดเป็นบิตเริ่มต้นของตัวอักขระ และหากทราบว่าในตัวอักขระหนึ่งตัวนั้นมีกี่บิต รวมทั้งความเร็วของการส่งของบิตต่าง ๆ โดยการนับจำนวนบิตที่ได้รับมาตามสาย หลังจากทราบบิตแรก ก็จะสามารถแยกตัวอักษรออกจากกันได้

เทคนิคการส่งข้อมูลแบบซิงโครนัสนั้น ใช้สำหรับส่งข้อมูลทั้งชุดไปครั้งเดียว โดยในการส่งแบบนี้ช่วงความกว้างของเวลาระหว่างบิตแต่ละบิตจะมีค่าเท่ากัน



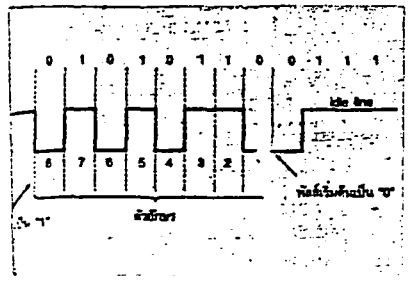
รูปที่ 2.2 การส่งแบบซิงโครนัสที่มี SYN หลายตัว

ในรูปที่ 2.2 แสดงให้เห็นถึงการส่งอักขระต่าง ๆ แบบซิงโครนัส โดยมีการส่งชุดข้อมูลชุดหนึ่งก่อนหน้าการส่งชุดข้อมูลตัวอักขระ เพื่อทำการหาบิตแรกของอักขระตัวแรกให้เป็นไปอย่างถูกต้อง ข้อมูลชุดนี้เรียกว่า SYN Transmission control Character (TC) ซึ่งประกอบด้วยข้อมูลขนาด 8 บิต เช่น 00010110 (มีพาริตี้เป็นคี่) และทางด้านรับก็จะคอยมองหาชุดของบิตดังกล่าว (looking for SYNC) ซึ่งจะกระทำทุกครั้งเมื่อรับบิตใหม่เข้ามาจนกว่าจะได้รับข้อมูลตัวอักขระที่กล่าวมาแล้วข้างต้น

ในระบบนี้ส่วนมากมักจะมี SYN นำหน้าข้อมูล 3-4 ตัว เพื่อให้แน่ใจได้ว่าชุดของบิตตัวอักขระที่รับเข้าคือข้อมูลที่ถูกต้อง

2.1.2 <u>การส่งแบบอะซิงโครนัส</u>

เมื่อจะทำการส่งแบบอะซิงโครนัส ตัวอักขระจะถูกส่งออกไปในเวลาใด ๆ ก็ได้ โดยไม่จำเป็นต้องมีความสัมพันธ์ระหว่างตัวอักขระหรือจะต้องมีเวลาที่แน่นอนอย่างไร แล้วแต่ตัวอักขระแต่ละตัว อาจจะทิ้งช่วงห่างกันเท่าใดก็ได้ เพราะอักขระแต่ละตัวจะมีบิตที่คอยบอกการเริ่มต้นและการสิ้นสุดของอักขระตัวนั้น (start bit,stop bit)ซึ่งในทางปฏิบัติ ทางด้านรับจะทราบได้ว่า มีบิตเริ่มต้นเข้ามาแล้ว เริ่มด้วยการเปลี่ยนสถานะจาก "1" เป็น "0" หรือจาก "0" เป็น "1" ก็ได้ (แต่ส่วนใหญ่มักให้เริ่มเปลี่ยนจาก "1" เป็น "0") ดังแสดงในรูปที่ 2.3



รูปที่ 2.3 การส่งแบบอะซิงโครนัส

ทางด้านส่ง จะเริ่มส่งบิตเริ่มต้น โดยการเปลี่ยนสถานะของค่าปกติที่เป็น "1" เป็น "0" หลังจากนั้นจะส่งบิตของข่าวสารสำหรับอักขระหนึ่งตัวออกไป ในขณะที่ด้านรับทราบถึงการเปลี่ยนสถานะจาก "1" เป็น "0" เป็นการบอกจุดเริ่มต้นของการทำงานว่า จะเริ่มภายหลังเวลาหรือเท่ากับครึ่งเวลาของความกว้าง 1 บิต หลังจากนั้น ทางด้านรับก็จะสุ่มตัวอย่างสถานะของสายทุก ๆ ช่วงเวลา 1 บิต เพื่อหาค่ารหัสของตัวอักขระที่ส่งมาจนกว่าจะครบ 8 บิต (ถ้าเป็นรหัสแอสกี) บิตถัดไปจะเป็นบิตสิ้นสุด หลังจากนั้น จะเว้นช่วงเวลาไปอีกนานเท่าไรก็ได้ ในการเริ่มส่งตัวอักขระตัวใหม่เพราะบิตเริ่มต้นจะคอยบอกอยู่แล้ว

ด้วยวิธีการส่งแบบมีบิตเริ่มต้น และจบด้วยบิตสิ้นสุดนี้ บางครั้งจึงเรียกการส่งแบบนี้ว่า start-stop transmission บิตสิ้นสุดนั้นจะแตกต่างกันไปในแต่ละระบบ เช่น การส่งรหัสแอสกี บิตสิ้นสุดจะมีขนาดความกว้างเท่ากับความกว้างของข้อมูล 1 หรือ 2 บิต ซึ่งจะเป็นช่วงเวลาที่ทางด้านรับทำงานหลังจากรับข้อมูลมาครบหนึ่งอักขระ เช่น การพิมพ์ในเครื่องพิมพ์ การเจาะรูบนเทปของเครื่องเจาะเทป เป็นต้น ช่วงเวลาของบิตสิ้นสุดนี้ ขึ้นอยู่กับการทำงานของเครื่องจักรกลซึ่งมีความเร็วช้า โดยอาจต้องมีเวลาของบิตสิ้นสุดเท่ากับ 2 บิต แต่ในอุปกรณ์ประเภทเครื่องจักรไฟฟ้านั้น จะมีความเร็วในการทำงานมาก ซึ่งอาจจะมีความกว้างของบิตดังกล่าวเพียง 1 บิตเท่านั้น

ในการส่งแบบอะซิงโครนัสนี้ เนื่องจากอักขระทุกตัวที่ส่งออกมาต่างเป็นอิสระต่อกัน และทุก ๆ อักขระจะมีบิตบอกการเริ่มต้นและสิ้นสุด ดังนั้นหากเกิดสัญญาณรบกวน จะก่อให้เกิดความผิดพลาดจนทำลายความถูกต้องของตัวอักขระ ความผิดพลาดนี้ถ้าเกิดขึ้นในการส่งแบบซิงโครนัส อาจจะทำลายข่าวสารหมดทั้งบล็อก (อักขระหลายตัว) จะมีความสัมพันธ์เพียงครั้งเดียว

2.1.3 <u>การส่งข้อมูลแบบขนานและแบบอนุกรม</u>

โดยทั่ว ๆ ไปหลักใหญ่ของการส่งข้อมูลในคอมพิวเตอร์ หรือระหว่างคอมพิวเตอร์ด้วยกัน จะมีลักษณะของการส่งข้อมูลอยู่ 2 แบบ คือ ส่งแบบขนาน และแบบอนุกรม

-การส่งข้อมูลแบบขนาน : ข้อมูลทุก ๆ บิทจะถูกส่งออกไปพร้อม ๆ กันในครั้งเดียว เช่น ถ้าข้อมูลเป็น 1010 ทั้งสี่บิทนี้จะถูกส่งออกไปพร้อมกัน โดยผ่านสายส่งข้อมูลที่มี 4 เส้น โดยแต่ละบิทจะส่งในสายส่งคนละเส้น

-การส่งข้อมูลแบบอนุกรม : ข้อมูลแต่ละบิทจะถูกส่งเรียงกันออกไปเป็นลำดับ ต่อเนื่องกันทีละบิท เช่น ถ้าข้อมูลเป็น 1010 : เลข 0 ทางขวามือสุดซึ่งเป็น LSB (Least Significant bit) ส่งออกไปก่อน ตามด้วยบิทที่สอง คือเลข 1 และบิทที่ 3 คือเลข 0 และบิทสุดท้ายคือเลข 1 ซึ่งเป็น MSB (Most Singnificant bit) ตามลำดับ โดยสายส่งข้อมูลจะมีอยู่เพียงเส้นเดียวเท่านั้น

ในระบบการสื่อสารข้อมูลนั้น ข้อมูลที่ส่งออกไปจะอยู่ในรูปของสัญญาณไฟฟ้า (เป็นสัญญาณอะนาล็อก) วิ่งผ่านไปตามสายส่ง ซึ่งมักจะเกิดปัญหาเรื่องเกิดความผิดพลาดของสัญญาณขึ้นมา ซึ่งปัจจัยอย่างหนึ่งที่เป็นสาเหตุก็คือ ระยะทางที่ส่งผ่านข้อมูล ถ้าระยะทางยิ่งไกลความผิดพลาดของข้อมูลก็จะเพิ่มมากขึ้นเป็นเงาตามตัว ฉะนั้นในการแก้ไขจุดบกพร่องนี้วิธีต่าง ๆ ที่ใช้แก้ไขปัญหานี้ก็คือ ส่งสัญญาณข้อมูลที่มีความแรงมาก ๆ ออกไปหรือมีการใช้วงจรขยายสัญญาณข้อมูลเป็นช่วง ๆ เพื่อให้ความแรงของสัญญาณคงที่ตลอดเวลา เป็นต้น ซึ่งการขยายสัญญาณของข้อมูลที่เป็นแบบอนุกรม จะยุ่งยากน้อยกว่าแบบขนานมาก นอกจากนี้ปัญหาอื่น ๆ ที่พบอยู่เสมอก็ได้แก่ เฟสของสัญญาณ และปัญหาเกี่ยวกับการหน่วงเวลาของสัญญาณข้อมูล ซึ่งล้วนแต่มีผลให้ข้อมูลที่รับได้ทางปลายทางผิดพลาดได้ และในการส่งข้อมูลแบบขนาน มักจะพบกับปัญหาการบิดเบือนของสัญญาณ (Skewing) เสมอ

ข้อเปรียบเทียบระหว่างการส่งข้อมูลแบบขนานและแบบอนุกรม

	แบบขนาน	แบบอนุกรม
1. ระยะทาง	ปกติจะน้อยกว่า 100 ฟุต	ส่งได้ตั้งแต่ระยะสั้น ๆ จนถึงระยะทางเป็นไมล์
2. ความเร็ว	อัตราความเร็วสูงมากในระยะที่ไม่ไกลมากนัก กำหนดได้เป็นจำนวนบิทต่อวินาที	อัตราความเร็วของข้อมูลที่ใช้กันอยู่ทั่วไปจะอยู่ในช่วง 0 ถึง 2 ล้านบิทต่อวินาที
3. ระดับของสัญญาณ	ในการอินเตอร์เฟสจะใช้ระดับสัญญาณที่ใช้กับอุปกรณ์ TTL คือ สัญญาณลอจิก 1 และ 0 จะแทนด้วยระดับแรงดัน +5V และ 0V ตามลำดับ	ใช้มาตรฐานของ EIA-RS 232C คือมีระดับสัญญาณไฟฟ้าขนาด 12 V หรืออาจจะใช้มาตรฐาน 20 mV current loop หรืออาจจะใช้ระดับสัญญาณของ TTL ก็ได้(ใช้กันน้อย)
4. ความผิดพลาดของสัญญาณ	ถ้าส่งในระยะทางไกล ๆ ความผิดพลาดของข้อมูลจะเกิดขึ้นง่าย	การผิดพลาดของสัญญาณจะมีน้อยลง
5. ค่าใช้จ่าย	ถ้าส่งในระยะทางไกล ๆ จะสิ้นเปลืองค่าใช้จ่ายมาก เพราะต้องใช้สายส่งสัญญาณหลายเส้น	สิ้นเปลืองน้อยกว่าหลายเท่า ถึงแม้ว่า จะใช้อุปกรณ์เปลี่ยนสัญญาณของข้อมูลจากแบบขนานไปเป็นแบบอนุกรม แล้วส่งผ่านสายส่งใช้อุปกรณ์ในการแปลงสัญญาณกลับมาเป็นแบบขนานอีก ก็ยังลงทุนน้อยกว่า

ตารางที่ 2.1

2.2 การสื่อสารแบบดิจิตอล

มีการพัฒนาการส่งจากแบบอะนาลอกเปลี่ยนไปเป็นการสื่อสารข้อมูลแบบดิจิตอล ไม่จำกัดเพียงคอมพิวเตอร์ที่ใช้ภาษาทางดิจิตอลเท่านั้น การส่งสัญญาณแบบดิจิตอลได้เริ่มขึ้นในแบบที่เป็นอะนาล็อก และถึงผู้ใช้ในลักษณะอะนาล็อก

อะนาล็อกของการสื่อสารแบบดิจิตอลจะขยายตัวเร็วมาก แต่จะติดต่อต่างชุดเพิ่มขึ้นไม่ค่อยมากนัก และจะมีการผสมเข้ากับเทคโนโลยีใหม่ ๆ เพื่อรักษาของเก่าไว้ FCC ได้ให้บริการกับนักวิทยุสมัครเล่น ในความอิสระในการใช้การสื่อสารแบบดิจิตอล การนำเสนอกฎ 97.69 จะให้การแต่งเติม :"การสื่อสารของนักวิทยุสมัครเล่นอาจจะรวมรหัสดิจิตอลซึ่งแทนด้วยตัวอักขระ, การวัดเชิงเส้น หรือ ข้อมูลอื่น ๆ รหัสดิจิตอลเหล่านี้ อาจถูกใช้สำหรับการสื่อสาร (โดยไม่จำกัด) วิทยุโทรพิมพ์, เสียง, โทรสาร, โทรทัศน์, การสื่อสารเพื่อควบคุมสถานีวิทยุสมัครเล่น, แบบจำลอง และสิ่งอื่น ๆ การเคลื่อนย้ายโปรแกรม หรือ การสื่อสารระหว่างคอมพิวเตอร์โดยตรง และการสื่อสารแบบเครือข่ายข้อมูลชนิดต่าง ๆ (รวมทั้งที่เรียกว่า ระบบ "packet switching"); โดยรหัสดิจิตอลเหล่านี้ ไม่ได้มุ่งที่จะปิดบังความหมายแต่เพื่อทำให้การสื่อสารง่ายเข้า

เทอมที่ว่าวิทยุโทรพิมพ์ใช้ใน ความหมายการสื่อสารระหว่างสิ่งที่คล้ายเครื่องพิมพ์ โดยใช้รหัสโทรเลข ที่เรียกว่า Baudot (หรือ Murray) ไม่สนใจว่าจะใช้รหัสนี้หรือไม่ กฎของ FCC อนุญาตเพียงรหัส Baudot สำหรับการสื่อสาร RTTY ก่อน 1975 นักวิทยุสมัครเล่นสองสามคนได้คิดเกี่ยวกับสิ่งอื่นที่ไม่ใช่เครื่องพิมพ์

ในปี 1975 ระบุถึงคอมพิวเตอร์ส่วนตัวที่พอจะหาซื้อได้ นักวิทยุสมัครเล่นจำนวนหนึ่งขยายงาน (เป็นแบบ 8 บิต) และ ทันทีทันใดก็คิดถึงการเชื่อมต่อมันกับโทรพิมพ์แบบ Baudot(ซึ่งใช้รหัส 5 หลัก) การประยุกต์ใช้อย่างหนึ่งก็คือ ใช้เป็นโทรพิมพ์เพื่อพิมพ์ (ออกกระดาษ) สำหรับคอมพิวเตอร์ของเขา อีกอย่างคือใช้คอมพิวเตอร์เพื่อส่งและรับ Baudot RTTY และ cw ในช่วงความถี่ของนักสมัครเล่น

2.3 มาตรฐานสากลในการอินเตอร์เฟส

องค์กรต่าง ๆ ได้สร้างมาตรฐานของตนเองขึ้นมาใช้ซึ่งมีอยู่มากมาย ดังนั้นจึงจำเป็นต้องทราบถึงมาตรฐานต่าง ๆ ที่มีใช้กันอยู่ให้ครบถ้วน

EIA : The Electronics Industries Association

เป็นมาตรฐานที่กำหนดขึ้น โดยสมาคมของโรงงานอุตสาหกรรมผู้ผลิตอุปกรณ์อิเลกทรอนิคส์แห่งสหรัฐอเมริกา มาตรฐานที่ตั้งขึ้นมาใช้กำหนดมาตรฐานของเครื่องมือ อุปกรณ์อิเลกทรอนิคส์ต่าง ๆ ในการกำหนดมาตรฐานจะใช้รหัส RS เป็นหลัก เช่นมาตรฐาน RS-232C ซึ่งใช้กันแพร่หลายในระบบการสื่อสารข้อมูลด้วยคอมพิวเตอร์ โดยจะกล่าวถึงมาตรฐานของลักษณะสัญญาณไฟฟ้าในการอินเตอร์เฟสเทอร์มินัล เข้ากับโมเด็ม หรืออินเตอร์เฟสเทอร์มินัลเข้ากับไมโครคอมพิวเตอร์ หรือ อินเตอร์เฟสเครื่องพิมพ์เข้ากับคอมพิวเตอร์เป็นต้น นอกจากนี้มาตรฐานอื่นที่มีการตั้งขึ้นมาใช้ก็ประกอบด้วย EIA RS-449, RS-422a และ RS-423A เป็นต้น

CCITT : The Consultative Committee In Interantional Telegraphy and Telephony.

เป็นองค์กรสากลที่กำหนดมาตรฐานเกี่ยวกับ การควบคุมระบบภาษี, แนะนำมาตรฐานหรือกำหนดมาตรฐานของระบบสื่อสารระหว่างประเทศทั้งโทรเลข และโทรศัพท์, CCITT เป็นหน่วยงานหนึ่งของ ITU หรือ International Telecommunication Union โดยมีหน้าที่เกี่ยวข้องกับการกำหนดมาตรฐานทางด้านการสื่อสาร

-CCITT study group VII : ทำหน้าที่พัฒนาและค้นคว้าเกี่ยวกับมาตรฐานการสื่อสารข้อมูลที่ติดต่อกันเป็นเครือข่าย โดยผลงานการวิจัยต่าง ๆ จะตีพิมพ์เผยแพร่ลงในวารสาร ISO X-Series Recommendation.

-CCITT study group XVII : ทำหน้าที่พัฒนามาตรฐานที่เกี่ยวกับการสื่อสารทางโทรศัพท์ โดยผลงานต่าง ๆ จะมีการตีพิมพ์ลงในวารสาร ISO V-Series documents

สำหรับมาตรฐานของ CCITT ที่ใช้กันแพร่หลายทางด้านการสื่อสารข้อมูล ดังเช่น V.28 (ใช้แทน RS-232C ได้), V.10 (ใช้แทน EIA RS-423A ได้), V.11 (ใช้แทน EIA RS-422A ได้) และ X.21 (ใช้แทน EIA RS-449 ได้) เป็นต้น

ในการกำหนดมาตรฐานของ CCITT จะกำหนดออกมาเป็นฉบับต่าง ๆ คือ Series A,B, C,D,E,F,...,Z ทุก ๆ ปีจะมีการร่วมประชุมปรึกษาหารือกันระหว่างประเทศสมาชิกในข้อกำหนดใหม่ ๆ หรือปรับปรุงมาตรฐานเก่าให้ทันสมัยและทุก ๆ 2-4 ปี จะออกหนังสือ Recommendation Series ต่าง ๆ ให้ทันสมัยขึ้น การอ้างถึง Recommendation ของ CCITT จึงต้องระบุว่าเป็น CCITT ปีใด, ลือะไร ตัวอย่างเช่น CCITT Recommendation Yellow book Series.... Year... หรือ Orange book Series ... Year ... หรือ Red Book Series...Year เป็นต้น ในตัวอย่างที่จะกล่าวถึงต่อไปนี้จะยกตัวอย่างเฉพาะ Orange book ดังมีรายละเอียดดังนี้

CCITT Series	Orange Book
A	กล่าวถึงโครงสร้างขององค์กร (CCITT) และการปฏิบัติงาน
B	เกี่ยวกับความหมายของคำที่ใช้หรือชื่อ
C	เกี่ยวกับการสื่อสารทั่ว ๆ ไป
D	คำแนะนำเกี่ยวกับภาษีทั่วไป เช่น การเช่าวงจรไพร์เวทไลน์ (Private line) หรือการให้การบริการ
E	คุณภาพของบริการโทรศัพท์ และภาษี รวมทั้งค่าบริการระหว่างประเทศ
F	การบริการโทรเลข และภาษีรวมทั้งค่าบริการระหว่างประเทศ
G,H,J	เกี่ยวกับการสายสื่อสาร
M,N	เกี่ยวข้องกับการวัด, ตรวจสอบและการบำรุงรักษาสายสื่อสาร
O	คุณลักษณะเฉพาะของเครื่องทดสอบและเครื่องมือตรวจวัด
P	คุณภาพของสายโทรศัพท์และเครื่องโทรศัพท์
Q	เรื่องทั่ว ๆ ไปที่เกี่ยวกับชุมสายโทรศัพท์และระบบสัญญาณที่ใช้
R1,R2	เกี่ยวกับระบบสัญญาณต่าง ๆ

CCITT Series	Orange Book
R,S,T และ U	เกี่ยวกับเทคนิคทางด้านโทรเลข
V	การสื่อสารข้อมูลผ่านสายโทรศัพท์
K/L	การป้องกันต่าง ๆ
X	โครงข่ายของการสื่อสารข้อมูลสาธารณะ
Z	การใช้ภาษาสำหรับโปรแกรมที่ใช้สำหรับการสื่อความหมาย

ISO : The International Standard Organization

เป็นองค์กรที่ตั้งอยู่ในเมืองเจนีวา ประเทศสวิสเซอร์แลนด์ ซึ่งทำหน้าที่กำหนดมาตรฐานทางกายภาพของอุปกรณ์ต่าง ๆ ที่ใช้ในการสื่อสารโทรคมนาคม โดยเฉพาะที่เกี่ยวกับคอมพิเตอร์ และ อินฟอร์เมชั่นโพรเซสซิ่ง องค์กรนี้จะประสานงานกับ CCITT อย่างใกล้ชิด

ANSI : The American National Standard Institute

มาตรฐานของ ANSI ส่วนใหญ่จะเกี่ยวกับ

1.ให้ความหมายหรือข้อกำหนดของลักษณะของระบบผลิตสัญญาณ และรับสัญญาณที่ใช้ในระบบสื่อสาร

2.กำหนดคุณภาพและคุณลักษณะของข้อมูลขณะที่กำลังส่งออกไป

3.ให้บริการเกี่ยวกับมาตรฐานสากล และมาตรฐานภายในประเทศสหรัฐอเมริกา

สำหรับหน่วยงานของ ISO จะแบ่งออกเป็น 2 ชุด คือ

Technical comittee 97, จะศึกษาเกี่ยวกับคอมพิวเตอร์และอินฟอร์เมชั่นโพรเซสซิ่ง โดยคณะกรรมการชุดแรกคือ ISO/TC97/SC6 จะรับผิดชอบเกี่ยวกับการพัฒนามาตรฐานทางด้านการสื่อสารข้อมูล และคณะกรรมการชุดที่สองคือ ISO/TC97/SC16 จะพัฒนาทางด้าน Open System Interconnection (OSI)

มาตรฐานของ ISO จะสามารถใช้แทนมาตรฐานประเภทเดียวกันของ CCITT และ EIA ได้เช่น ISO 2110 สามารถใช้แทน EIA RS-232C และ RS-366A ได้ นอกจากนี้ ISO 4902 ยังใช้แทน EIA RS-499 ได้เป็นต้น

สำหรับคณะกรรมการของ ANSI ที่ค้นคว้าทางด้านการสื่อสารข้อมูลคือ Commitee on Computers and Information Processing. X3 มาตรฐานของ ANSI ที่รู้จักกันแพร่หลายคือ รหัสแอสกี้ (ASCII:American Standard Code for Information Interchange) ซึ่งเป็นรหัสมาตรฐานที่ใช้กันทั่วไปในระบบคอมพิวเตอร์

Federal Goverment Standard

เป็นหน่วยงานของรัฐบาลสหรัฐอเมริกา ที่ทำหน้าที่กำหนดมาตรฐานทางด้านการสื่อสารในส่วนที่เกี่ยวกับการสื่อสารข้อมูล เมื่อถูกกำหนดใช้เป็นมาตรฐานโดย National Bureau of Standards (NBS) แล้วเราจะรู้จักกันในชื่อของ Federal Information Processing Standards (FIPS) ซึ่งมาตรฐานส่วนใหญ่จะเหมือนกับมาตรฐานของ EIA

Millitary Standard_188

เป็นมาตรฐานของทหาร ซึ่งถูกกำหนดเป็นมาตรฐานสำหรับเทคนิคการสื่อสารโทรคมนาคมของทหาร ตัวอย่างหนึ่งของมาตรฐานคือ MIL-STD-185, MIL-STD-188A และ MIL-STD-1888 และที่ใช้กันแพร่หลายคือ MIL-STD-188C มาตรฐานเหล่านี้เรียกว่า MIL-STD-188 Series ซึ่งใน Series นี้อาจจะแยกออกเป็นหลายหมวด เช่น มาตรฐาน Common Long Haul/Tractical ใช้ MIL-STD-188-100 เป็นต้น

Bell System

Bell เป็นมาตรฐานที่กำหนด โดยองค์กร (ซึ่งมีลักษณะคล้ายกับองค์การโทรศัพท์แห่งประเทศไทย) ทางโทรศัพท์ของบริษัท Bell Laboratory... มาตรฐานของ Bell ถูกกำหนดขึ้นเพื่อใช้ควบคุมโรงงานผู้ผลิตสินค้าที่ต้องการใช้งานร่วมกับระบบของ Bell เนื่องจาก Bellเป็นหน่วยงานที่ใหญ่มากในสหรัฐอเมริกา และมีส่วนแบ่งการขายในตลาดสูง ทำให้อุปกรณ์ชนิดที่จะใช้ร่วมกับระบบโทรศัพท์ของ Bell จะต้องเป็นไปตามมาตรฐานที่ Bell กำหนด แต่ระยะหลัง ๆ Bell เอง ก็เริ่มผ่อนปรน หรือแก้ข้อกำหนดของตนให้เข้ากับ CCITT ได้ (เฉพาะบางส่วนเท่านั้น)

ข้อกำหนดหรือมาตรฐานของแต่ละองค์กรต่าง ๆ มีรายละเอียดมากกว่านี้ ในที่นี้จะยกมาพอสังเขปดังนี้

X3.1 เป็นรายละเอียดคุณลักษณะของ Singaling rate สำหรับการสื่อสารข้อมูลระบบ Synchronous ทั้งแบบ Serial or Parallel อัตราความเร็วที่ว่านี้ apply ที่ Interface ระหว่าง DTE กับ DCE ซึ่งใช้งานบนช่องสัญญาณเสียงปกติคือ 4 KHz. (FIPS 22)

X3.4 Code ของ Character ทั่วไปที่ใช้งานสื่อสารกันระหว่างเครื่องมือชนิดต่าง ๆ (FIPS 1)

X3.24 มาตรฐานนี้ใช้สำหรับบอกให้ทราบถึงสัญญาณข้อมูลและสัญญาณเวลาระหว่าง Interface DTE กับ DCE

X3.15 กำหนดมาตรฐานของ Bit ใน ASCII Code (American National Standard Code for Information interchange) แบบ Serial bit สำหรับการสื่อสารข้อมูล (FIPS 16)

X3.16 กำหนดโครงสร้างของ Character และการตรวจรับ Character parity แบบ Serial bit ที่ใช้ ASCII ในการสื่อสารข้อมูล (FIPS 17) ทั้งแบบ Synchronous and Asynchronous

X3.25 กำหนดโครงสร้างของ Character และการตรวจรับ Character parity แบบ Parallel bit ใน ASCII Code ที่การสื่อสารแบบเรียงตัวอักษร (FIPS 18)

X3.28 ขั้นตอนในการใช้ Control Character ในการติดต่อควบคุมการสื่อสารข้อมูล

X3.36 กำหนดคุณลักษณะของการสื่อสารข้อมูลความเร็วสูงระบบ Synchronous และกล่าวถึงอัตราการสื่อสารความเร็วสูงระหว่าง DTE กับ DCE ซึ่งสื่อสารบนช่องสัญญาณความเร็วสูง

X3.44 กำหนดมาตรฐาน และ Performance ของอุปกรณ์ที่การสื่อสารข้อมูลใช้เป็นตัวกลาง

การอินเตอร์เฟสตามมาตรฐานของ RS-232C มีข้อจำกัดไว้ คือ ความยาวสายเคเบิลไม่เกิน 50 ฟุต และความเร็วในการส่งไม่เกิน 20 กิโลบิทต่อวินาที และเนื่องจากข้อจำกัดของ RS-232C นี้ทำให้มาตรฐาน RS-422, RS-423 และ RS-449 ถูกพัฒนาขึ้นมาสำหรับการส่งที่ความเร็วสูงขึ้น อย่างไรก็ตามอินเตอร์เฟส RS-232C ยังเป็นที่นิยมใช้กันมากเพราะสามารถใช้งานร่วมกับ RS-449 ได้ รายละเอียดของ RS-Series อื่น ๆ สรุปใจความสำคัญได้ดังนี้

RS-269A กำหนด Signaling rate สำหรับการสื่อสารระบบ Synchronous แบบ Serial or Parallel ผ่านช่องสัญญาณเสียง 4 KHz. RS-269 กำหนดมาตรฐานความเร็วในการสื่อสารข้อมูลไว้ดังนี้ 600,1200,1800,2000, 2400 และ 3000 baud มาตรฐานนี้ยังครอบคลุมถึง signal element duration และการวัด tolerances ของสัญญาณ RS-269A (ปรับปรุงจาก RS-269) ได้กำหนดให้ความเร็วในการสื่อสารใหม่คือ 600,1200, 2400,4800,7200 และ 9600 BPS

RS-328 กำหนด Interchange ระหว่างชนิดของข่าวสารของเครื่อง Fascimile ซึ่งทำงานผ่าน Modem (over switched Voice grade telephone facilities)

RS-334 กำหนดให้ Interface ระหว่าง DTE กับ Synchronous DCE ซึ่งใช้ RS-232C มีคุณลักษณะทางไฟฟ้ามาตรฐานเดียวกัน

RS-366 มาตรฐาน Interface ระหว่างอุปกรณ์ Automatic Calling Unit (ACU) กับ DTE

RS-422 กำหนดคุณลักษณะทางไฟฟ้าของ balanced Voltage digital interface circuit

RS-423 กำหนดคุณลักษณะทางไฟฟ้าของ Unbalanced Voltage digital interface circuit

RS-449 EIA RS-449 เป็นมาตรฐาน ซึ่งกำหนดคุณลักษณะทาง mechanical ซึ่ง Interface ระหว่าง DTE กับ DCE โดยให้มีมาตรฐานตามข้อกำหนดทาง Electrical เป็นไปตาม RS-422 and RS-423

RS-429 แบบ 37 position connector สำหรับทุก ๆ Interchange circuit ยกเว้น secondary channel ซึ่งจะมี connector แบบ 9 position แยกต่างหากสำหรับ 9 position connector นี้จะต้องใช้เมื่อ secondary channel อยู่ด้วยความเร็วในการสื่อสารสำหรับ Interface นี้อนุญาตให้ส่งได้ด้วยความเร็ว 2 Mbit/sec และความยาวของสายเคเบิล ระหว่าง DTE กับ DCE อนุญาตให้ยาวถึง 200 ฟุต

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

2.4 โปรโตคอลและเน็ทเวิร์คฟังก์ชัน

โปรโตคอล (Protocol) หมายถึงกฎเกณฑ์, ขบวนการ และรูปแบบของภาษา ซึ่งได้ถูกกำหนดขึ้นใช้ในการสื่อสารข้อมูล โดยจะอยู่ในรูปของซอฟท์แวร์และฮาร์ดแวร์ประกอบกัน ลักษณะของโปรโตคอลก็ได้แก่ รูปแบบของการส่งข้อมูลที่เป็นแบบ HDX (Half Duplex), แบบ FDX (Full Duplex) และ FFDX (Full-Full Duplex) เป็นต้น

โปรโตคอลที่พบเห็นการใช้งานบ่อย ๆ ก็เช่น SDLC (Synchronous Data Link Control) ซึ่งกำหนดและสร้างขึ้นโดยบริษัท IBM, HDLC (High level Data Lind Control) ของบริษัท Honeywell นอกจากนี้ก็ยังมีโปรโตคอลที่สร้างขึ้นโดยบริษัทอื่น ๆ อีกมากมาย เช่น DDCMP (Digital Data Communication Message Protocol), BDLC, DLC, CDC และ NAPLP เป็นต้น

โปรโตคอลเหล่านี้จะอยู่ในลักษณะของซอฟท์แวร์ ซึ่งโครงสร้างของระบบซอฟท์แวร์เหล่านี้จะต้องเขียนให้สอดคล้องกับมาตรฐานของ OSI ด้วย ซึ่งมาตรฐานของ OSI จะประกอบไปด้วยมาตรฐานย่อย ๆ ถึง 7 ระดับ (level) ด้วยกัน ดังรายละเอียดต่อไปนี้

1. Physical Level

เป็นมาตรฐานที่กำหนดเกี่ยวกับการอินเตอร์เฟสระหว่างอุปกรณ์ปลายทาง (DTE : Data Termianl Equipment) กับอุปกรณ์สื่อสาร (DCE : Data Communication Equipment) โดยจะรวมถึงทางด้านกายภาพ, ทางด้านสัญญาณไฟฟ้า และกระบวนการในการติดตั้ง, การบำรุงรักษาตลอดจนรวมถึงรายละเอียดของช่องสัญญาณที่ใช้ในการติดต่อระหว่าง DTE กับ DCE เพราะฉะนั้นอาจสรุปได้ว่า Physical level นี้ก็หมายถึง มาตรฐานต่าง ๆ เช่น EIA RS-232C, RS-449, RS-422A และ RS-423A เป็นต้น

2. Data Link Level

เป็นกฎเกณฑ์สำหรับการส่งผ่านข้อมูล จากจุดหนึ่งไปยังอีกจุดหนึ่ง โดยผ่านตัวเชื่อมต่อ สำหรับโปรโตคอลของ Data Link Level นี้สำคัญต่อระบบไมโครคอมพิวเตอร์ก็คือโปรโตคอลในแบบการส่งข้อมูลแบบอนุกรมและเป็นแบบอะซิงโครนัสด้วย

3. Network Level

ใช้สำหรับควบคุมระบบสวิชชิ่ง และทิศทางการไหลของข้อมูล ในเครื่อข่ายต่าง ๆ โดยที่ Network Level ประกอบไปด้วยการเชื่อมต่อทางกายภาพ และทางลอจิกที่จำเป็นต้องใช้ในการส่งผ่านข้อมูลจากจุดส่งไปยังจุดรับ

4. Transfer Level

รับผิดชอบเกี่ยวกับการบริการการติดต่อข่าวสารอย่างมีคุณภาพ สิ่งสำคัญ จุดสำคัญของมาตรฐานชั้นนี้คือ การควบคุมความแน่นอน และความเชื่อถือได้ของระบบในการส่งข้อมูลจากจุดหนึ่งไปยังอีกจุดหนึ่ง ซึ่งจุดประสงค์หลักในชั้นนี้ก็เพื่อจะแสดงถึง ลักษณะการอินเตอร์เฟสระหว่างเครือข่ายอย่างมีประสิทธิภาพ

5. Session Level

กำหนดขึ้นเพื่อใช้ในการติดต่อ และเลิกติดต่อการรับส่งข้อมูล

6. Presentation Level

สำหรับกำหนดโครงสร้าง และรูปแบบ (Syntax and Format) และทรานสฟอร์มเมชั่นของข้อมูล

7. Application Level

กำหนดโครงสร้างของวิธีการ และขบวนการที่ใช้ในการประยุกต์ระบบเข้ากับมาตรฐานของ OSI รวมถึงการกำหนดการใช้รหัสผ่าน (Pass word), Log on Utility และการทรานสเฟอร์ไฟล์ เป็นต้น

2.5 มาตรฐาน RS-232C

มาตรฐานตระกูล RS ทั้ง 4 แบบนั้นถูกออกแบบมาจาก EIA (ELECTRONIC INDUSTRY ASSOCIATION) โดยคำว่า RS ย่อมาจาก RECCOMEND STANDARD ซึ่งแปลเป็นไทยว่ามาตรฐานที่ได้รับการยอมรับว่าใช้งานได้ดี ส่วนตัวเลข 232,422,423 และ 449เป็นรหัสประจำของมาตรฐานหนึ่ง ๆ ส่วนตัวอักษร A และ C ที่ตามหลังนี้บ่งบอกให้รู้ว่ามาตรฐานนี้ ได้รับการพัฒนามาแล้วกี่ครั้ง เช่น A ก็หมายถึงครั้งที่ 1 C หมายถึงครั้งที่ 3 เป็นต้น

มาตรฐาน RS-232C มีใช้กันครั้งแรกเมื่อปี พ.ศ. 2505 ซึ่งใช้ในการอินเตอร์เฟสระหว่างอุปกรณ์ตัวส่งหรือที่เรียกว่า อุปกรณ์ต้นทาง (SENDER DRIVER) และอุปกรณ์ปลายทาง (RECEIVER TERMINATOR)โดยตามศัพท์ที่ใช้งานคอมพิวเตอร์ จะหมายถึงเทอร์มินอลและโมเด็ม ตามลำดับ โดยเทอร์มินอล มักจะถูกเรียกว่าอุปกรณ์ DTE (DATA TERMINAL EQUIPMENT) และ โมเด็มก็จะถูกเรียกว่าอุปกรณ์ DCE (DATA COMUNICATION EQUIPMENT) ในทางทฤษฎีข้อกำหนดหลักที่บ่งบอกถึงข้อจำกัดการใช้งานของ RS-232C นี้ก็คือ อัตราความเร็วในการส่งข้อมูล (บอดเรต) สูงสุด สำหรับสายเคเบิ้ลความยาว 50 ฟุต จะมีได้ไม่เกิน 19,200 บอด แต่ในทางปฏิบัติสายเคเบิ้ลสามารถที่จะให้ยาวกว่านี้ได้ แต่จะต้องส่งข้อมูลในอัตราความเร็วที่ต่ำกว่าที่กำหนดนี้ (ความยาวของสายเคเบิ้ล เป็นปฏิภาคส่วนกลับอัตราความเร็วในการส่งข้อมูล) แรงดันมาตรฐานของ RS-232C จะใช้ระดับลอจิก 1 ด้วยค่าแรงดัน -12 โวลต์ และลอจิก 0 ด้วยแรงดัน +12 โวลต์

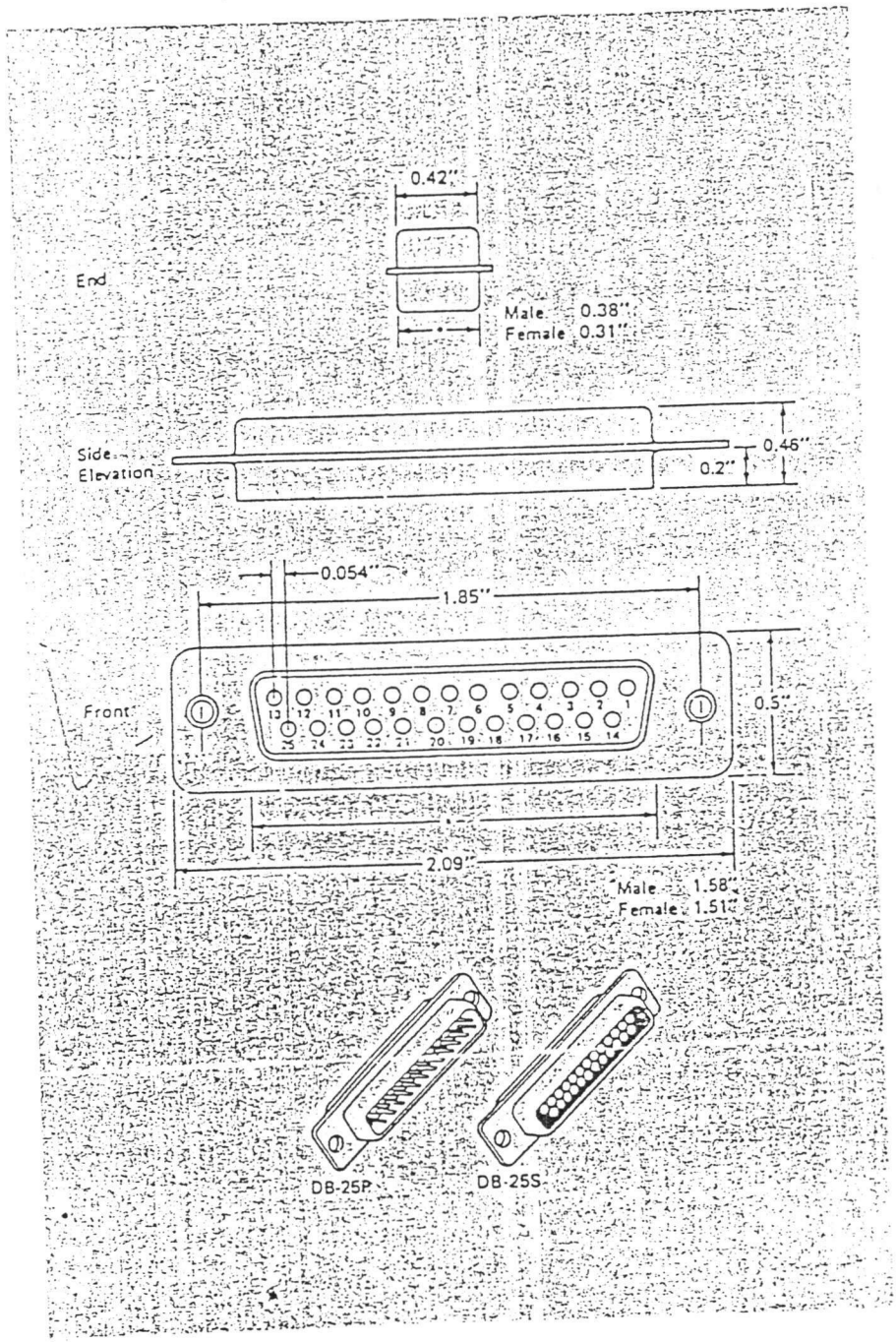
2.5.1 คุณสมบัติทางไฟฟ้าของ RS-232C

โดยแท้จริงแล้วรูปแบบของคอนเนคเตอร์ที่เป็นมาตรฐานของ RS-232C นั้น ไม่ได้มีการระบุไว้ตั้งแต่ต้น แต่มาภายหลังมักนิยมใช้คอนเนคเตอร์ที่มีลักษณะคล้ายตัว D จำนวน 25 ขา เป็นหลักดังแสดงให้เห็นทั้งแบบตัวผู้ (DB-25P*) และแบบตัวเมีย (DB-25S**) ซึ่งเรามักนิยมเรียกกันติดปากว่าคอนเนคเตอร์แบบ D-PIN 25 ขา

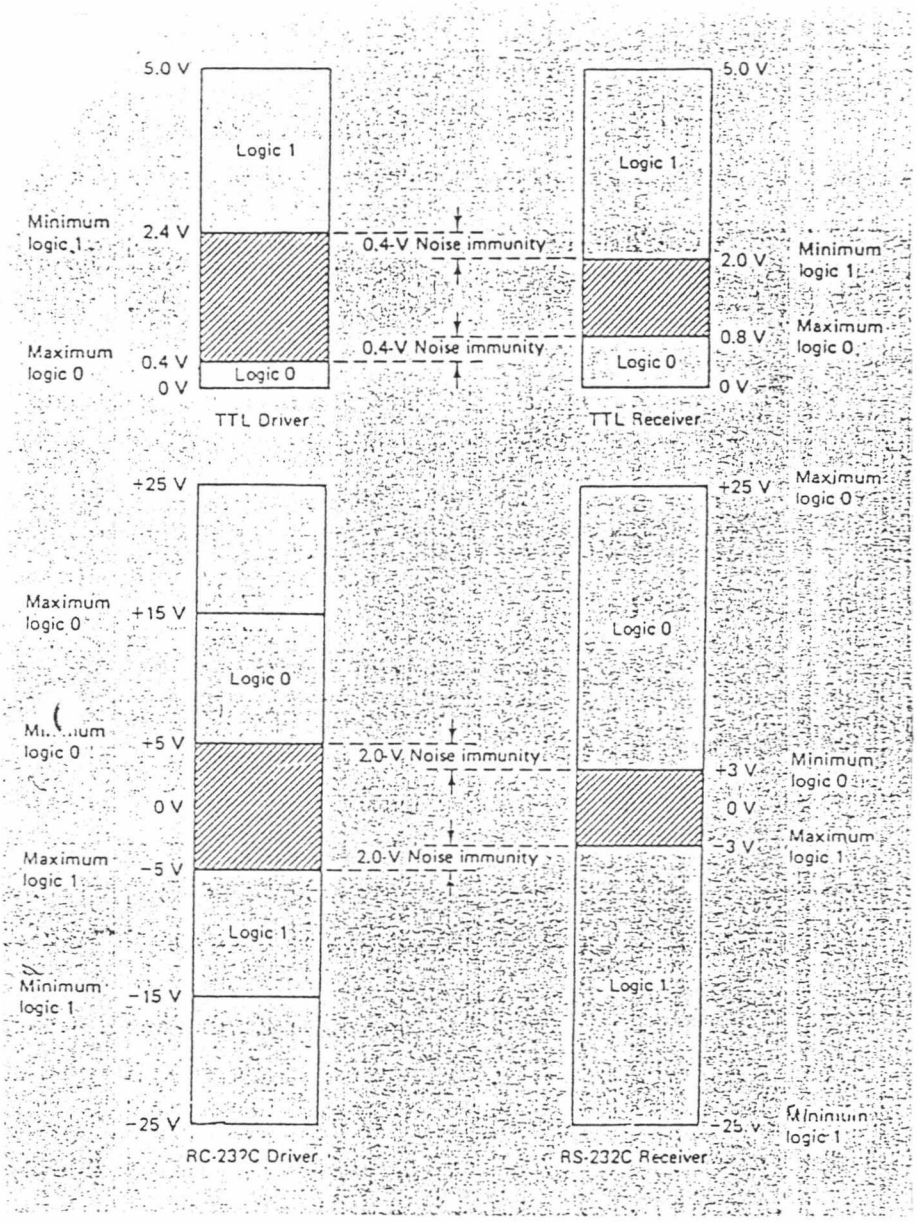
หมายเหตุ

* P ย่อมาจาก PLUG (ตัวผู้)

** S ย่อมาจาก SOCKET (ตัวเมีย)

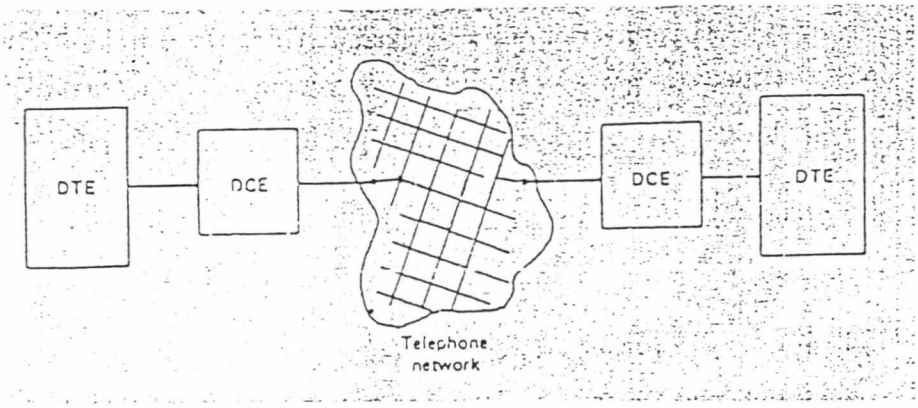


รูปที่ 2.4 แสดงรูปร่างของคอนเนคเตอร์แบบ DB-25P และ DB-25S ที่ใช้เป็นมาตรฐาน RS-232C



รูปที่ 2.5 แสดงการเปรียบเทียบการกำหนดลอจิก ระหว่างระดับลอจิกแบบ TTL กับระดับลอจิกที่เป็นมาตรฐาน RS-232C

จากรูปที่ 2.5 จะสังเกตเห็นได้ว่า ผลจากการเปรียบเทียบระดับลอจิกของมาตรฐานทั้ง 2 มาตรฐานของ TTL จะใช้ระดับแรงดันค่าบวกแทนลอจิก 1 และมี NOISE IMMUNITY เท่ากับ 0.4 โวลต์ แต่มาตรฐาน RS-232C จะแตกต่างไปอย่างมาก โดยจะใช้ระดับแรงดันที่สูงขับออกทางเอาท์พุท ($\pm$12 โวลต์) ซึ่งจำเป็นต่อการใช้งานทั้งนี้ก็เพราะเพื่อให้แน่ใจว่า การทำงานกับสายเคเบิ้ลซึ่งยาวจะมีความถูกต้องและเป็นไปได้ สำหรับ NOISE IMMUNITY จะกำหนดให้มีค่าเป็น 2 โวลต์ ทั้งนี้ก็เพื่อที่จะหลีกเลี่ยงสัญญาณรบกวนจากแหล่งอื่น ๆ ภายนอกที่สร้างปัญหาให้แก่ระบบมาตรฐาน TTL



รูปที่ 2.6 แสดงให้เห็นการติดต่อสื่อสารระหว่าง อุปกรณ์ DTE กับ DCE โดยผ่านข่ายสายโทรศัพท์

ความหมาย DCE และ DTE

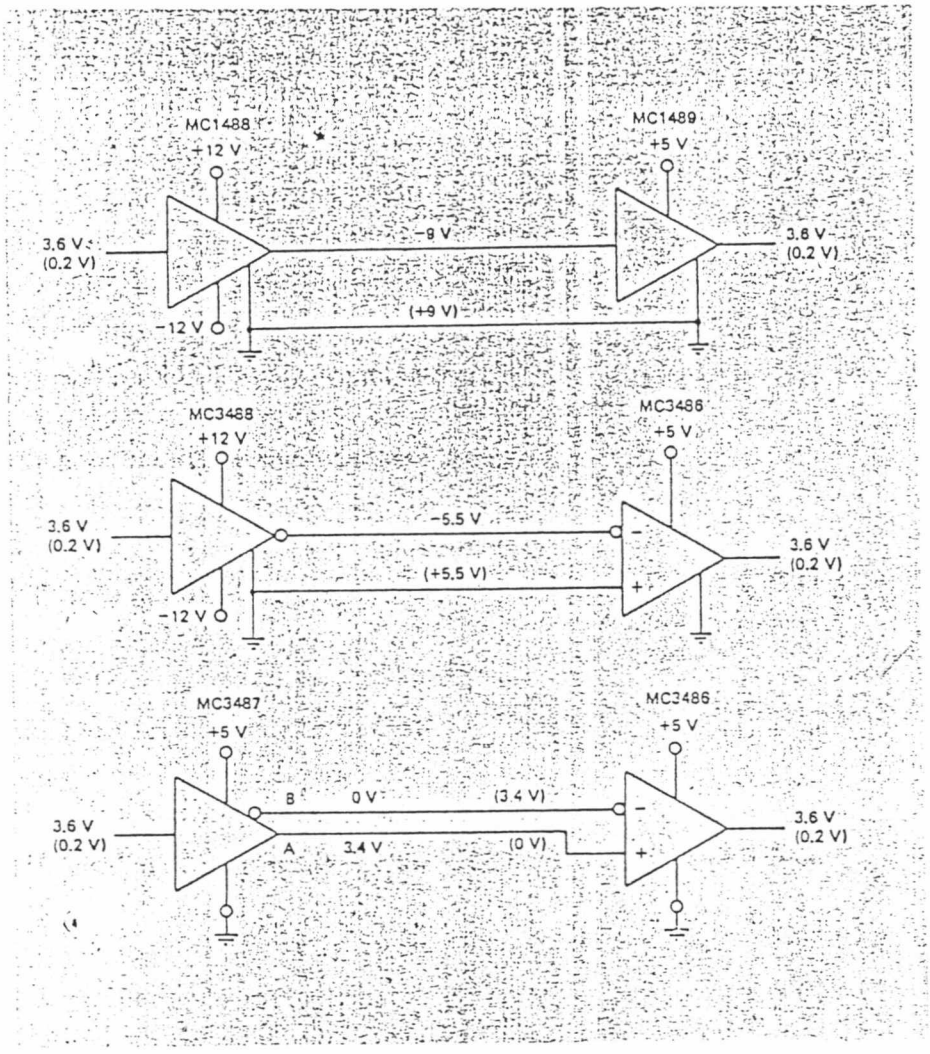
DCE ก็คืออุปกรณ์ที่มีระบบการทำงานทำให้เกิดการเชื่อมต่อแหล่งที่เป็น DTE ตั้งแต่ 2 ชุด นอกจากนี้ยังใช้เปลี่ยนลักษณะของสัญญาณและสร้างรหัสสัญญาณต่าง ๆ ที่จำเป็นต่อการสื่อสารข้อมูลระหว่าง DTE ดังได้ยกตัวอย่างไปแล้วว่า ตัวอย่างที่เห็นง่าย ๆ ของ DCE ก็คือ โมเด็ม ส่วน DTE จะมีความหมายว่า เป็นอุปกรณ์ที่ประกอบไปด้วยตัวส่งข้อมูล (DATA SOURCE) หรือตัวรับข้อมูล (DATA SINK) หรืออาจจะเป็นทั้งตัวส่งและตัวรับก็ได้ โดยจะมีฟังก์ชั่น การทำงานต่าง ๆ อาทิเช่น หน่วยควบคุมลอจิก, บัฟเฟอร์ และ อุปกรณ์อินพุท, เอาท์พุท หรืออาจพูดง่าย ๆ ก็คือ คอมพิวเตอร์นั่นเอง และโดยสภาพที่แท้จริงแล้ว DTE มักจะแทนด้วยแหล่งกำเนิดข้อมูลแหล่งแรกและแหล่งรับข้อมูลสุดท้าย อาทิเช่น เครื่องพิมพ์ จอมอนิเตอร์ เทอร์มินอล เป็นต้น

การอินเตอร์เฟสระหว่าง RS-232C กับ TTL จะกำหนดให้ทางด้านส่ง (DRIVER) และทางด้านรับ (RECEIVER) เป็นไปตามระบบที่แสดงได้ดังรูปที่ 2.6 จากรูปที่ 2.7.1 เราจะใช้ชิฟ MC 1488 ทำการรับอินพุทที่เป็นระดับลอจิก TTL (+5 โวลต์) และให้ระดับลอจิกเอาท์พุทเป็นระดับลอจิกของ RS-232C ($\pm$12 โวลต์) เป็นชิฟ MC 1489 ก็จะทำงานตรงกันข้ามกับที่กล่าวมา คือเปลี่ยนระดับลอจิกที่เป็นมาตรฐาน RS-232C ไปเป็นระดับลอจิก TTL

ผ่านทางสายส่งสัญญาณไปยังชิฟ USART ทั้งนี้ก็เพราะว่า ตัวส่งและตัวรับเหล่านี้ จะทำการส่งระดับลอจิกลบของ RS-232C ไปยังผู้ใช้ปลายทาง

คุณลักษณะทางไฟฟ้าประการหนึ่งที่ถือได้ว่าเป็นมาตรฐานของ RS-232C ก็คือ ค่าของเวลาทรานซิชั่น (TRANSITION TIME) ก็หมายถึง ช่วงเวลาที่ใช้สำหรับการเปลี่ยนจากระดับลอจิกหนึ่งไปสู่อีกระดับลอจิกหนึ่ง จะต้องมีได้ไม่เกิน 4% ของเวลา 1 บิท (BIT TIME) ดังนั้นถ้าอัตราความเร็วในการส่งเป็น 19,200 บอด (ซึ่งถือว่าเป็นบอดเรตที่สูงสุดที่มักนิยมใช้กัน) จะต้องมีค่าทรานซิชั่นเกิดขึ้นได้ไม่เกิน 0.04 * 1 / 19,200 = 2.1 uSEC ($2.1 * 10^{-6}$ วินาที) หรือเราอาจจะพูดได้ว่า ความยาวสายเคเบิ้ลนี้เป็นแฟคเตอร์อย่างหนึ่งที่จะมาจำกัดค่าเวลาทรานซิชั่น เพราะสายเคเบิ้ลยิ่งยาวเท่าไร ค่าความจุ (CAPACITIVE LOAD) ของสายเคเบิ้ล ก็ยิ่งมากตาม อันเป็นผลให้เวลาทรานซิชั่นถูกทำให้ช้าลง ซึ่งในเรื่องนี้ก็ได้มีการทดสอบกันมาแล้วว่า ที่ความเร็วบอดเรต 19,200 บอดนั้นค่าความยาวของสายเคเบิ้ลจะมีได้ไม่เกิน 50 ฟุต

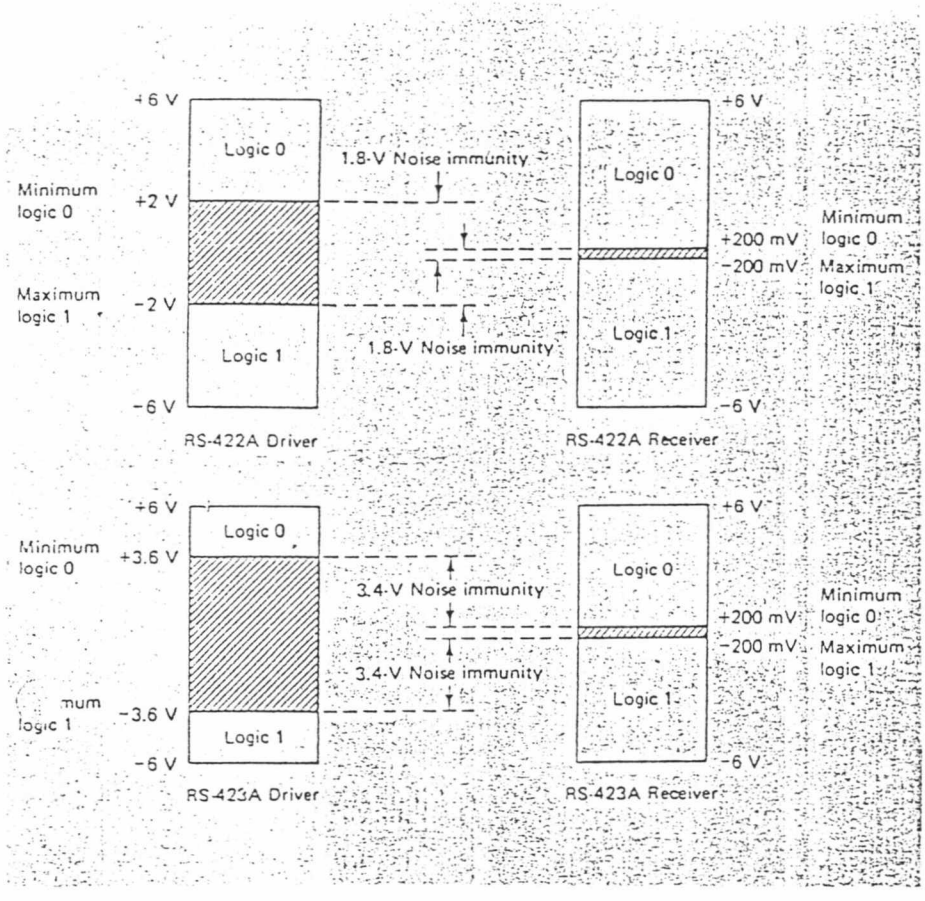
ปัญหาหลักของการใช้ RS-232C ก็คือ ด้านตัวส่ง (DRIVER) กับด้านตัวรับ (RECEIVER) จะถูกต่อด้วยกันในลักษณะที่ไม่สมดุลย์ (UNVALANCED) หรือ (SINGLE-ENDED) กล่าวคือ ทั้งสัญญาณอินพุทและสัญญาณเอาท์พุทจะถูกอ้างอิงจากกราวด์ร่วมเดียวกัน (COMMON GROUND) ดูรูปที่ 2.7.1 ประกอบ ก็จะเห็นว่าด้านส่งและด้านรับ แม้ว่าจะมีจุดที่เป็นกราวด์ร่วมกันก็จริง แต่ศักยภาพทางศักดาไฟฟ้าจะแตกต่างกัน ดังนั้นจึงเป็นสาเหตุทำให้เกิดกระแสไหลวนผ่านสายที่เป็นกราวด์ร่วมนี้ด้วย ผลที่ติดตามมานั่นก็คือ จะเกิดศักดาไฟฟ้าตกคร่อม เกิดจาก I (กระแสที่ไหล) * R (ความต้านทานของสายกราวด์) เกิดขึ้น ยังผลทำให้ค่าของการแยกสัญญาณรบกวน NOISE IMMUNITY) จากเดิมที่มีค่าเท่ากับ 2 โวลต์ ถูกลดลงไป และด้วยปรากฏการณ์ความไม่สมดุลย์ที่เกิดขึ้นระหว่างด้านส่งและด้านรับ อันนี้กลายมาเป็นแฟคเตอร์ข้อจำกัดความยาวของสายเคเบิ้ลที่ใช้อินเตอร์เฟสอีกประการหนึ่งด้วย



รูปที่ 2.7 แสดงการเปลี่ยนระดับลอจิกของ มาตรฐาน RS ทั้ง 3 ตัว (RS-232C, RS-423A,RS-422A) ไปเป็นระดับลอจิกของ TTL

สำหรับในรูปที่ 2.7.2 และ 2.7.3 นั้น จะแสดงมาตรฐานการอินเตอร์เฟสทางไฟฟ้าแบบใหม่ ที่มีชื่อมาตรฐานว่า RS-423A และ RS-422A ตามลำดับ โดย RS-422A จะใช้การส่งแบบดิฟเฟอเรนเชียล (DIFFERENTIAL) ระหว่างตัวส่งและตัวรับ ทั้งนี้ก็เพื่อที่ตัดปัญหาเกี่ยวกับสายกราวด์ที่เคยเกิดขึ้นแก่ RS-232C ทางด้านรับจะเป็นฝ่ายตรวจรับความแตกต่างของสัญญาณอินพุททั้งสองว่าเป็น บวกหรือลบ ส่วน RS-423A ก็จะมีลักษณะที่คล้ายคลึงกัน แต่จะใช้สายส่งสัญญาณเพียงเส้นเดียว โดยให้ทางด้านรับได้รับสัญญาณแบบดิฟเฟอเรนเชียลแทน (การทำเช่นนี้ไม่ถือว่าเกิดจุดกราวด์ร่วมเกิดขึ้น) และในรูปที่ 2.8 จะเป็นรูปแสดงระดับลอจิกที่ถูกกำหนดไว้สำหรับมาตรฐาน 422A และ 423A

การออกแบบการรับให้เป็นแบบดิฟเฟอเรนเชียล เหตุผลก็คือการออกแบบในลักษณะเช่นนี้จะเป็นวิธีที่สามารถลดค่าเวลาทรานซิชั่นลงได้มาก และผลที่ตามมาก็คือ สามารถที่จะส่งข้อมูลในอัตราบอดเรดได้สูงยิ่งขึ้นอีกทั้งสามารถส่งได้ในระยะไกลขึ้นกว่าเดิมมาก อาทิเช่น มาตรฐาน RS-422A สามารถส่งในอัตรบอดได้สูงถือ 100,000 บอด และใช้กับสายเคเบิ้ลได้ถึงความยาว 4,000 ฟุตทีเดียว



รูปที่ 2.8 แสดงการเปรียบเทียบระดับลอจิกที่กำหนด เป็นมาตรฐานสำหรับ RS-422A และ RS-423A

ในตารางที่ 2.2 จะเป็นทางเปรียบเทียบคุณลักษณะทางไฟฟ้าที่สำคัญ ๆ ของมาตรฐาน RS ทั้ง 3 ที่ได้อธิบายไปแล้ว และควรจะได้รู้ไว้ด้วยว่ามาตรฐาน RS-422A และ RS-423A นั้นจริง ๆ แล้วมาตรฐานคุณลักษณะทางไฟฟ้าของมันจะถูกนำมาใช้ในการสร้างมาตรฐานใหม่ของ EIA ในเวลาต่อมา นั่นก็คือมาตรฐาน RS-499 ซึ่งถือว่าเป็นมาตรฐานการอินเฟสแบบอนุกรมมาตรฐานใหม่ที่สุดที่อาจเปรียบเทียบได้กับ RS-232C อย่างไรก็ตามมันจะมีอะไรบางอย่างที่ต่างจาก RS-232C คือ มันจะประกอบไปด้วยคอนเนคเตอร์ทั้งหมด 2 ชุด ชุดแรกก็จะเป็นคอนเนคเตอร์แบบ 37 ขาไว้สำหรับการอินเตอร์เฟส สัญญาณหลักที่สำคัญ ๆ อีกชุดจะเป็นคอนเนคเตอร์ชนิด 9 ขามีไว้เพื่อให้ผู้ใช้ได้เลือกใช้ในวิธีการสื่อสารข้อมูลแชนแนลที่ 2 สำหรับอัตราบอดเรตนั้น จะมีได้สูงสุดไม่เกิน 20,000 บอด ซึ่งก็จะทำให้สามารถนำเอา RS-422A และ RS-423A มาใช้ด้วยได้

Parameter	RS-232C	RS-422A	RS-423A
Line length (max.)	50 ft	4000 ft	4000 ft
Frequency (max.)	20 kbaud/50 ft	10 Mbaud/40 ft 1 Mbaud/400 ft 100 kbaud/4000 ft	100 kbaud/30 ft 10 kbaud/300 ft 1 kbaud/4000 ft
Mode of operation	Single-ended input and output	Differential input and output	Single-ended output, differential input
Driver logic levels			
"0"	> +5 to +15 V	> +2 to +5 V	> +3.6 to +6 V
"1"	< −5 to −15 V	< −2 to −5 V	< −3.6 to −6 V
Noise immunity	2.0 V	1.8 V	3.4 V
Number of receivers allowed on one line	1	10	10
Input impedance	3–7 kΩ and 2500 pF	> 4 kΩ	> 4 kΩ
Output impedance	—	< 100 Ω balanced	< 50 Ω
Short circuit current	500 mA	150 mA	150 mA
Output slew rate	30 V/μs maximum	—	Controls provided
Receiver input voltage range	±15 V	±7 V	±12 V
Maximum voltage applied to driver output	±25 V	−0.25 to +6 V	±6 V

ตารางที่ 2.2 แสดงคุณลักษณะทางไฟฟ้าของมาตรฐาน RS-232C, RS-422A และ RS-423A

2.5.2 สัญญาณและการอินเตอร์เฟสทางแมคคานิคของ RS-232C

ในตารางที่ 2.3 จะแสดงรายชื่อของสัญญาณทั้ง 25 ขา ซึ่งถือว่าเป็นมาตรฐานการอินเตอร์เฟสของ RS-232C ขาที่เป็นข้อมูล (DATA) ก็คือกลุ่มที่เป็นแชนแนลปฐมภูมิ (PRIMARY CHANNEL) และแชนแนลทุติยภูมิ (SECONDARY CANNEL) ส่วนขาสัญญาณควบคุมซึ่งนับได้ว่ามีความสำคัญที่สร้างขึ้นมา เพื่อเป็นโปรโตคอลระหว่างโมเด็ม (DCE) และเทอร์มินัล (DTE) ก็คือขาสัญญาณ DTE, DSR, CTS, RTS และ DCD สำหรับขาสัญญาณ RING INDICATOR (ขา 22) จะใช้สำหรับโมเด็มที่เป็นแบบออโต้ไดอัลและ ขา BAUD-RATE SELECT (ขา 23) จะใช้สำหรับการเลือกบอดเรตให้แก่โมเด็มทั้งภาคตัวส่งและตัวรับ

		Data		Control	
Pin	Signal name	From DTE to DCE	To DTE from DCE	From DTE to DCE	To DTE from DCE
1	Protective ground				
2	Transmitted data	×			
3	Received data		×		
4	Request to send (RTS)			×	
5	Clear to send (CTS)				×
6	Data set ready (DSR)				×
7	Signal ground				
8	Data carrier detect (DCD)				×
9/10	Reserved for data set testing				
11	Unassigned				
12	Secondary data carrier detect				×
13	Secondary clear to send				×
14	Secondary transmitted data	×			
15	Transmit signal element timing				×
16	Secondary received data		×		
17	Receive signal element timing				×
18	Unassigned				
19	Secondary request to send			×	
20	Data terminal ready (DTR)			×	
21	Signal-quality detector * (indicates probability of error)				×
22	Ring indicator				×
23	Data signal rate select (allows selection of two different baud rates)				×
24	Transmit signal element timing			×	
25	Unassigned				

ตารางที่ 2.3 ตารางแสดงสัญญาณทั้ง 25 ของมาตรฐานการอินเตอร์เฟสแบบอนุกรมของ RS-232C

2.5.3 โปรโตคอลสำหรับการแลกเปลี่ยนข้อมูลของ RS-232C

เราก็คงพอจะทราบกันมาบ้างแล้ว ถึงความจำเป็นของสัญญาณโต้ตอบซึ่งกันและกัน (HANDSHAKING SIGNAL) ในการสื่อสารข้อมูลแบบอนุกรม เครื่องคอมพิวเตอร์จะสามารถทำการรับส่งข้อมูลได้ตรงกัน หรือ ที่เรียกว่า ซิงโคไนซ์ (SYNCHRONIZED) กัน กับอัตราความเร็วของตัวชิฟ USART โดยการตรวจสอบสถานะของแฟล็กที่มีชื่อว่า TxRDY

ยกตัวอย่างเช่น เมื่อทำการส่งรหัสแอสกี้ 1BH 7AH(ESC "C") เพื่อจุดประสงค์ในการล้างจอภาพ ซึ่งโดยทั่วไปแล้ว จะต้องใช้เวลาหลายมิลลิวินาที (MILLISECOND) แต่อย่างไรก็ดี ในอัตราบอดเรตที่ 9,600 บอด การรับส่งโดยไม่มีการโต้ตอบสัญญาณกันแล้ว การรับส่งข้อมูลก็จะเร็วขึ้นประมาณว่า ตัวอักษร 1 ตัวจะใช้เวลาประมาณ 1 มิลลิวินาที ทั้งนี้ก็เพราะว่าด้านตัวส่งไม่อาจจะรู้ได้เลยว่า ในขณะนั้นด้านรับได้ทำกระบวนการของตัวเองเสร็จแล้วหรือยัง ก็จะพยายามส่งไปเรื่อย ๆ อันเป็นผลให้ข้อมูลบางอย่างเกิดหายไปได้ เพราะเหตุนี้จึงเกิดความจำเป็นที่จะทำการโต้ตอบสัญญาณด้วยกัน

สัญญาณควบคุม 5 สัญญาณที่ใช้ร่วมกับโมเด็ม เพื่อใช้เป็นโปรโตคอลแบบโต้ตอบสัญญาณตามแบบมาตรฐาน RS-232C มีดังต่อไปนี้คือ

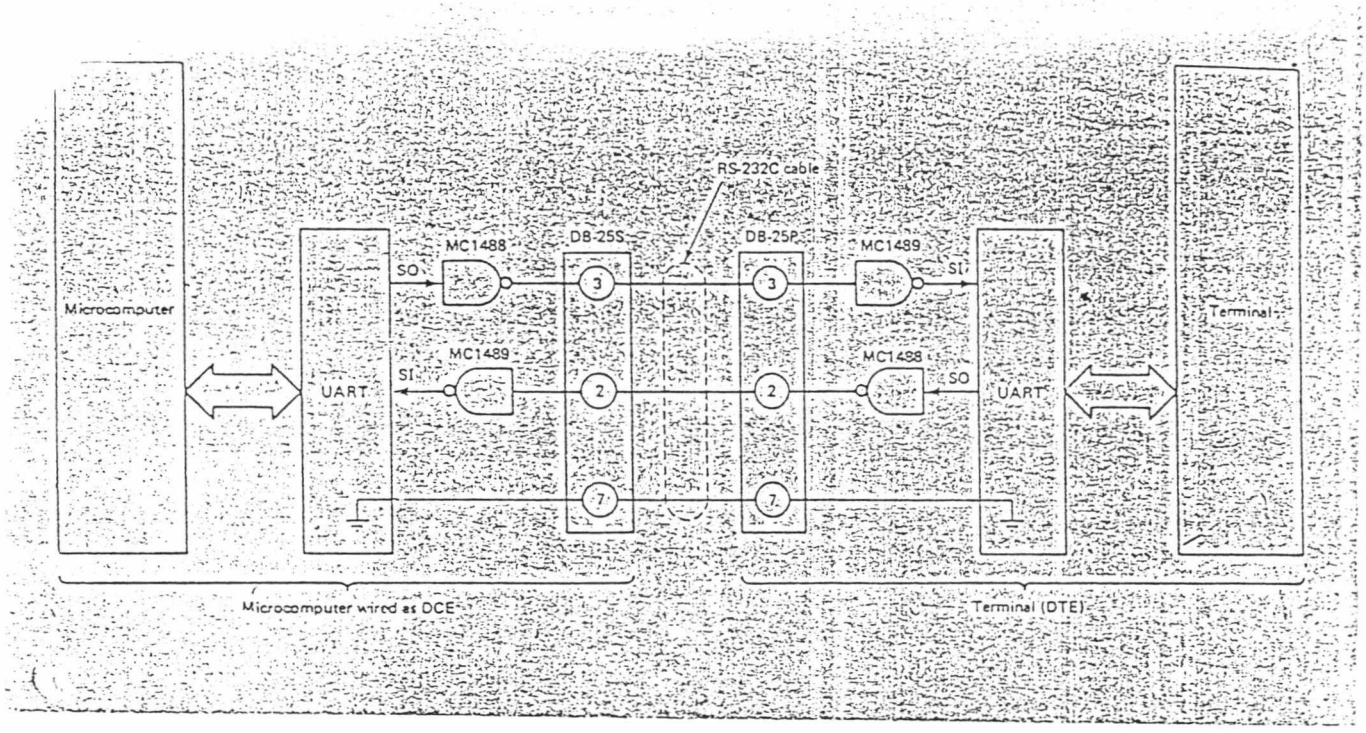
1. DATA CARRIER DETECT (DCD) สัญญาณนี้เป็นเอาท์พุทที่เกิดขึ้นมาจาก DCE เพื่อแสดงให้โมเด็มทราบว่าในขณะนั้นได้ตรวจเจอว่า สัญญาณพาหะเกิดการเปลี่ยนแปลงขึ้น (VALID CARRIER)

2. DATA TERMINAL READY (DTR) สัญญาณนี้จะเป็นเอาท์พุทที่เกิดขึ้นมาจาก DTE เพื่อแสดงให้ทราบว่าการสื่อสารข้อมูลในขณะนั้นอยู่ในสภาพที่พร้อม (ซึ่งเป็นสัญญาณที่สามารถใช้เป็นสวิทช์เพื่อให้โมเด็มทำงาน)

3. DATA SET READY (DSR) สัญญาณนี้ จะเป็นเอาท์พุทที่เกิดจาก DCE เพื่อตอบสนองต่อสัญญาณ DTR และเป็นการแสดงตัวว่า DCE ถูก ON โดยต่ออยู่กับแชนแนลที่ทำการสื่อสารอยู่

4. REQUEST TO SENT (RTS) เป็นสัญญาณเอาท์พุทที่เกิดจาก DTE เพื่อใช้แสดงตัวว่าขณะนี้ DTE มีความพร้อมสำหรับที่จะส่งข้อมูลแล้ว

5. CLEAR TO SEND (CTS) เป็นสัญญาณเอาท์พุทที่เกิดจาก DCE และตอบรับสัญญาณ RTS เพื่อใช้แสดงตัวว่า มันมีความพร้อมเกิดขึ้นแล้ว



รูปที่ 2.9 แสดงการอินเตอร์เฟสไมโครคอมพิวเตอร์ และเทอร์มินัลผ่าน RS-232C อย่างง่าย ๆ

2.6 PACKET RADIO -INTEGRATED DIGITAL COMMUNICATION

คำว่า data communications จะหมายถึง การส่งผ่านข้อมูลระหว่างคอมพิวเตอร์ ซึ่งในความเป็นจริงแล้ว ยังรวมถึงรูปแบบของสัญญาณดิจิตอลทุกรูปแบบซึ่งในระบบการสื่อสารปัจจุบันก็ได้เปลี่ยนมาเป็นการใช้การส่งแบบดิจิตอล ในการส่งแบบดิจิตอล จะมีการแยกส่วนหลัก ๆ ออกเป็นส่วน ๆ โดยจะสร้างข้อมูลดิจิตอล เป็นส่วน ๆ แล้วส่งไปเช่น error control techniques ใช้ในการแยกข้อมูลที่ผิดพลาดโดย สัญญาณรบกวนชนิดต่าง ๆ ซึ่งเกิดจาก ขบวนการทางดิจิตอลไม่ได้เกิดจากการส่งข้อมูลดิจิตอล ในการติดต่อสื่อสารของระบบใหญ่ ๆ ได้ สร้าง integrated digital networks และนักวิทยุสมัครเล่นได้นำมาใช้ในนามของ packet radio ซึ่งในที่สุดก็ได้พัฒนามาเป็น amateur packet radio networks ซึ่งสามารถรับข้อมูลจาก keyboard และสามารถส่งได้อย่างอัตโนมัติ และสามารถที่จะติดต่อกับนักวิทยุสมัครเล่นได้ทั่วโลก เนื่องจากเทคโนโลยีมีความก้าวหน้ามากยิ่งขึ้นจึงมี ผู้คิดค้นหลายอย่าง ซึ่งแตกต่างกันจึงเป็นเหตุให้เห็นสมควรตั้งมาตรฐานขึ้นมาโดย ARRL ได้จัดการประชุมสำหรับนักทดลอง packet radio ขึ้นเพื่อแลกเปลี่ยนความคิดเห็นโดยใช้ชื่อว่า Amatuer radio computer networks conference ในปี 1981 ARRL Ad Hoc Committee of Amateur Radio Digital Communicationsได้จัดตั้งผู้ทรงคุณวุฒิในการพัฒนามาตรฐานโดยคณะกรรมการนี้ได้มาจากสมาคม packet radio ต่าง ๆ และจากผู้ที่ประสบความสำเร็จในด้าน packet radio

พื้นฐานของ amateur packet radio networks ได้ถูกกำหนดไว้โดยใช้สัญญาณดิจิตอล 8 บิทในการส่งข้อมูลโดยต่อกันแบบ LAN โดยการใช้งานในย่านความถี่ VHF และในปี 1984 ก็ได้ขยายไปใช้งานในย่าน HF และระบบดาวเทียม ซึ่งทำให้ผู้ใช้ทางฝั่งตะวันตกและฝั่งตะวันออกของสหรัฐอเมริกาสามารถที่จะติดต่อกันได้

Handbook of 1984 ได้ประมาณว่ามีนักวิทยุสมัครเล่นที่ใช้ packet radio ในอเมริกา 300 คนต่อมา Handbook of 1986 ได้พบว่ามีผู้ใช้ packet radio ถึง 3900 คนข้อกำหนดในการใช้งานมีดังนี้คือ

-เครื่องที่ใช้จะต้องสามารถรับและส่ง ในการติดต่อสื่อสารระหว่างเครื่องของเรากับของผู้อื่นภายใน networks เดียวกัน

-สามารถส่งข้อมูลเป็นไฟล์ (file) ได้ในการติดต่อระหว่างสถานี

-จะต้องมีคอมพิวเตอร์แม่ข่ายเพื่อกระจายข้อมูลไปยังเครื่องอื่น ๆ ภายใน networks

ในการติดต่อระหว่าง network ควรมีข้อปฏิบัติดังนี้

-ควรจะมีมาตรฐานในการติดต่อระหว่าง networks

-ควรที่จะส่งข้อมูลด้วยความเร็วสูง(9.6-56Kbit)

2.7 MODEMS

Modems ประกอบด้วย modulator และ demodulator ใน telephone terminalogy จะเรียกว่า data set modem ใช้ในการต่อ digital baseband signals form กับ analog form โดยส่งผ่านสื่อกลาง analog โดยการสื่อสารแบบ analog มี bandwidth 3 kHz ซึ่ง bandwidth ของสัญญาณ analog จะมีความสัมพันธ์กับอัตราการส่งข้อมูลซึ่งส่วนใหญ่จะใช้ 1200 bit/s ใน 3 kHz analog channel ซึ่งจะมีราคาค่อนข้างถูกและในปัจจุบันนี้ก็สามารถทำได้ถึง 9600 bit/s ใน 3 kHz analog chanel โดยใช้เทคนิคการ modulation ที่ซับซ้อน

โมดูเลท (Modulate)

คือขบวนการอย่างหนึ่งที่นำเอาคลื่นความถี่สูงค่าหนึ่งที่เรียกว่า สัญญาณพาหะ (Carrier Signal) เข้าไปรวมกับคลื่นความถี่เสียงซึ่งจะทำให้เกิดความถี่ใหม่ขึ้นอีกสองค่า เช่น สมมติให้สัญญาณพาหะมีความถี่เป็น F_1 และความถี่เสียงเป็น F_2 เมื่อผ่านขบวนการโมดูเลเตอร์แล้ว ความถี่ที่เกิดขึ้นจะเป็น $F_1 + F_2$, $F_1 - F_2$ และ F_1 อีกค่าหนึ่ง ซึ่งจะทำให้เกิดแถบความถี่ที่เรียกว่า Upper Side Band และ Lower side band ขึ้นมา

โดยทั่วไปขบวนการโมดูเลทของสัญญาณอะนาล็อกจะมีเทคนิคอยู่ 3 แบบคือ

1. แอมปลิจูดโมดูเลชั่น (Amplitude Modulation) หรือแบบ AM.
2. ฟรีเควนซี่โมดูเลชั่น (Frequnecy Modulation) หรือแบบ FM.
3. เฟสโมดูเลชั่น (Phase Modulation) หรือแบบ PM.

ดีโมดูเลท (Demodulate)

เป็นขบวนการที่ทำงานตรงข้ามกับขบวนการโมดูเลท คือ ทำการแยกเอาสัญญาณพาหะออก ซึ่งจะทำให้ได้สัญญาณของข้อมูลอย่างแท้จริงเพียงอย่างเดียว

จุดประสงค์สำคัญที่จำเป็นต้องมีการโมดูเลทสัญญาณความถี่เสียงนี้ก็เพื่อทำให้สามารถส่งไปได้ในระยะไกล ๆ โดยเกิดการผิดเพี้ยนของสัญญาณน้อยที่สุด

การโมดูเลทสัญญาณดิจิตอลซึ่งมีเพียง 2 ระดับคือ 0 กับ 1โดยทั่วไปเทคนิคที่ใช้มีอยู่หลายแบบด้วยกันคือ

1. ฟรีเควนซี่ชิฟคีย์อิ้ง (Frequency Shift Keying : FSK)
2. เฟสชิฟคีย์อิ้ง (Phase Shift Keying : PSK)
3. แอมปลิจูดชิฟคีย์อิ้ง (Amplitue Shift Keying)

โมเด็ม เป็นอุปกรณ์ที่ทำหน้าที่ร่วมระหว่างขบวนการโมดูเลท และดีโมดูเลท (Modulate/DEModulate) นั่นเอง ซึ่งบางทีก็อาจจะเรียกว่าเป็น "ตัวแปลงสัญญาณ" (Signal Converter) ก็ได้ โดยเป็นอุปกรณ์ที่ทำการแปลงสัญญาณดิจิตอลที่ส่งออกมาจากเครื่องคอมพิวเตอร์ให้เป็นสัญญาณเสียง (Voice Signal)เพื่อส่งผ่านไปในสายโทรศัพท์ได้ โดยต้องผ่านขบวนการโมดูเลทก่อน

ส่วนทางคอมพิวเตอร์ด้านรับจะมีการต่ออุปกรณ์โมเด็มคั่นไว้ เมื่อโมเด็มด้านรับ ๆ สัญญาณเสียงเข้ามา ก็จะทำการแปลงสัญญาณเสียงให้กลับไปเป็นสัญญาณดิจิตอล (โดยใช้ขบวนการดีโมดูเลท)ก่อนที่จะส่งผ่านไปยังเครื่องคอมพิวเตอร์ต่อไป ในการส่งผ่านสัญญาณระหว่างเครื่องคอมพิวเตอร์กับอุปกรณ์รอบข้าง เช่นเครื่องพิมพ์ที่ตั้งอยู่ในระยะไกลออกไปก็จำเป็นอย่างยิ่งที่จะต้องส่งสัญญาณผ่านเข้าไปในสายโทรศัพท์ในรูปของสัญญาณอะนาล็อก และเมื่อส่งถึงด้านรับก็จะแปลงกลับมาเป็นสัญญาณดิจิตอลเหมือนเดิม

การแบ่งชนิดของโมเด็มตามอัตราส่งข้อมูล

1. อัตราการส่งข้อมูลต่ำ (Low-speed) มีอัตราการส่งข้อมูลไม่เกิน 600 bps
2. อัตราการส่งข้อมูลปานกลาง (Medium-speed) มีอัตราการส่งข้อมูลระหว่าง 1200 ถึง 9600 bps
3. อัตราการส่งข้อมูลสูง (High-speed) มีอัตราการส่งข้อมูลมากกว่า 9600 bps

โมเด็มชนิดอัตราการส่งข้อมูลต่ำ

The Bell 103 Modem:

เป็นโมเด็มชนิดที่มีอัตราการส่งข้อมูลต่ำที่นิยมใช้กันอย่างแพร่หลาย เป็นผลผลิตของบริษัท Bell-Laboratory สหรัฐอเมริกา นอกจากนี้ก็มีผลิตภัณฑ์ของบริษัทที่ทำเลียนแบบคือมีความสามารถเหมือนกับโมเด็ม Bell 103 หรือที่เรียกว่า Bell 103 compatible ซึ่งโมเด็มชนิดนี้ จะพบว่า มีการติดตั้งใช้งานในเครื่องคอมพิวเตอร์เมนเฟรมทั่ว ๆ ไป และอาจจะมีใช้บ้างในเครื่องไมโครคอมพิวเตอร์บางยี่ห้อ สำหรับใน Bell 103 compatible Modem นั้นคุณสมบัติบางอย่างอาจจะขาดหายไป คือไม่คอมแพทกับ Bell 103 ทุกประการ สำหรับความสามารถของ Bell 103 อาจจะสรุปได้ว่าประกอบด้วย

1. สามารถทำงานได้ทั้งเป็นออริจิเนติ้ง (Originating) หรือแอนเซอร์ริ่ง (Answering) หรือ CY lead-Controlled ก็ได้ สำหรับ

Originating หรือ ในระบบโทรศัพท์เรียกว่า เป็นผู้เรียก นั่นเอง

Answering หรือ ในระบบโทรศัพท์เรียกว่า เป็นผู้ตอบรับ

CY lead-Controlled หมายถึง เป็นการทำงานของโมเด็มที่สามารถเปลี่ยนแปลงโหมดของการทำงานเป็นผู้เรียก หรือผู้ตอบรับก็ได้ โดยใช้ซอฟท์แวร์ควบคุมการทำงานผ่าน RS-232 ขา 11

2. จะยกเลิกการเชื่อมต่อ (ตัดสายโทรศัพท์) ทันทีถ้าสัญญาณ incoming carrier หายไป

3. สามารถที่จะสร้างสัญญาณเอาท์พุทให้มีระดับตามที่ต้องการได้ และสามารถปรับให้มีอัตราส่วนของสัญญาณข้อมูลต่อสัญญาณรบกวน (Signal-to-noise ratio) และคุณลักษณะของสัญญาณไฟฟ้าตามต้องการได้

4. สามารถสร้างระดับเสียง (tone) หรือสัญญาณพัลส์ที่ถูกต้อง เพื่อใช้ในการเรียก /ต่อ โทรศัพท์ไปยังสถานที่ไกล ๆ ได้ซึ่งสัญญาณดังกล่าวนี้จะต้องสอดคล้องกับสัญญาณที่ใช้ในเครือข่ายโทรศัพท์ หรือ PSTN (Package Switch Telephone Network) การทำงานในลักษณะนี้ เรียกว่าเป็นการทำงานแบบเรียกโทรศัพท์อัตโนมัติ (Auto dial)

5. มีความสามารถที่จะใช้ซอฟท์แวร์ควบคุมการบ่งบอกสถานะขณะนั้น ๆ เพื่อตอบรับต่อการเรียกเข้ามาว่า ขณะนี้โทรศัพท์ไม่ว่าง

นอกจากคุณลักษณะต่าง ๆ ดังกล่าวแล้ว Bell-103 โมเด็มยังสามารถใช้งานร่วมกับการทำงานที่เป็นข้อยกเว้นพิเศษของสัญญาณควบคุมตามมาตรฐาน RS-232C ได้อีกด้วย

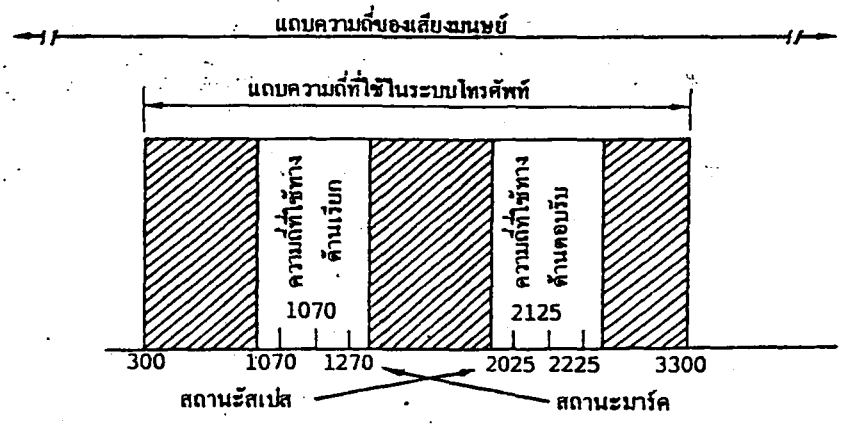
เทคนิคการโมดูเลชั่น

การกำหนดความถี่ของด้านผู้เรียกกับผู้ตอบรับ

สัญญาณดิจิตอลประกอบด้วยสัญญาณ 2 สถานะคือ 1 และ 0 ดังนั้นระดับโวลเตจหนึ่งแทน 0 อีกระดับโวลเตจหนึ่งก็จะแทนด้วย 1 เช่นกัน จึงได้มีการกำหนดความถี่ของสัญญาณเพื่อใช้แทนระดับโวลเตจของทางด้านผู้เรียกกับผู้ตอบรับ ซึ่งต้องใช้ต่าง ๆ กัน 4 ความถี่เพื่อไม่ให้เกิดการรบกวนของสัญญาณ (ในกรณีที่มีการสื่อสารแบบฟูลดูเพล็กซ์) โดยทางด้านส่งจะต้องใช้ความถี่ของสัญญาณ 2 ค่า เพื่อแทนโวลเตจระดับ 1 กับระดับ 0 และทางด้านรับก็เช่นเดียวกัน แต่ความถี่ที่ใช้จะต่างกัน ในกรณีของ Bell-103 ได้มีการกำหนดความถี่ของสัญญาณที่ใช้ไว้ดังนี้คือ

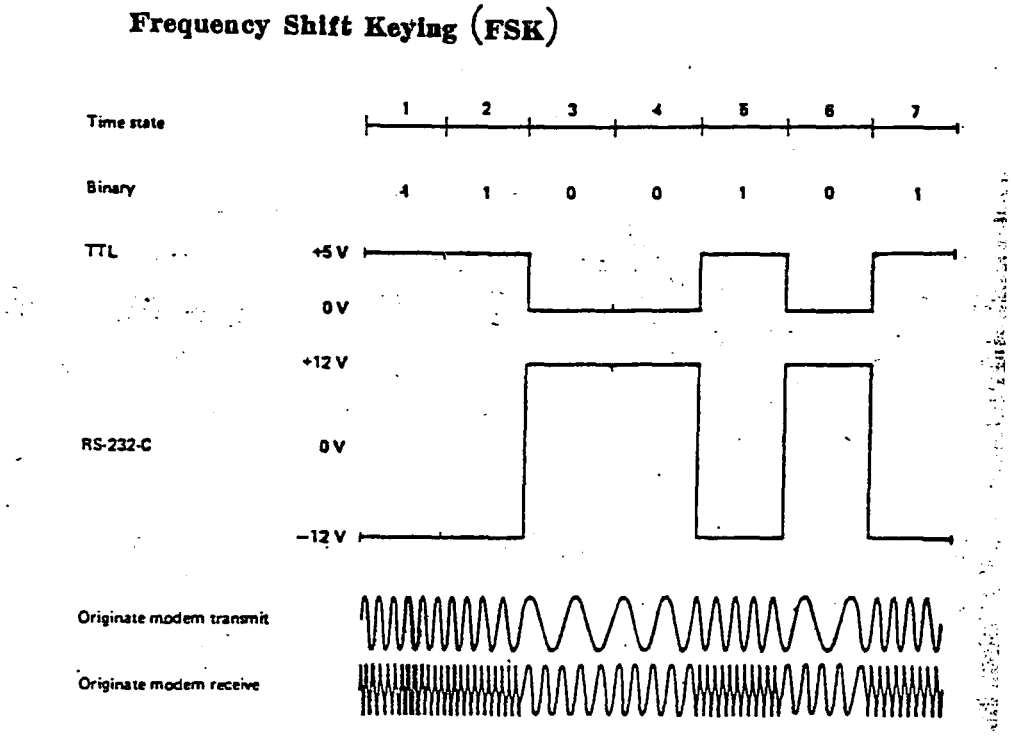
ความถี่ 1070 Hz สำหรับทางด้านส่ง แทนลอจิก 0
ความถี่ 1270 Hz สำหรับทางด้านส่ง แทนลอจิก 1
ความถี่ 2025 Hz สำหรับทางด้านรับ แทนลอจิก 0
ความถี่ 2225 Hz สำหรับทางด้านรับ แทนลอจิก 0

โดยแสดงเป็นแถบความถี่ได้ดังรูป 2.10



รูปที่ 2.10 แสดงแบนด์วิดท์ของความถี่ที่กำหนดให้ใช้ในโมเด็มชนิด Bell-103 โดยใช้งานร่วมกับ PSTN ได้

จากรูปจะเห็นว่า ความถี่ที่ระบุมานั้นจะต้องมีความสัมพันธ์กันระหว่างโมเด็มทางด้านส่ง (Originate modem)ที่ทำการส่งสัญญาณด้วยความถี่ ๆ หนึ่ง ส่วนโมเด็มทางด้านรับ (Answer Modem) ก็จะรับสัญญาณความถี่ค่านั้นเข้ามา แล้วตอบกลับไปด้วยสัญญาณที่ความถี่อีกค่าหนึ่ง ซึ่งเทคนิคในการนำสัญญาณดิจิตอลมาโมดูเลทเข้ากับสัญญาณอะนาล็อกแล้วส่งผ่านตัวนำออกไป เราเรียกว่า เทคนิคของ FSK หรือ Frequency Shift Keying



รูปที่ 2.11 แสดงสัญญาณที่เกิดจากการส่งอักษร S ด้วยเทคนิคการโมดูเลท (รวมถึงวิธี FSK ด้วย)

จากรูปแสดงให้เห็นว่าเทคนิค FSK นำมาใช้ในการส่งสัญญาณดิจิตอลได้อย่างไร ในกรณีนี้สมมติว่าเราต้องการจะส่งอักษร S ออกไป ในรหัสแอสกี้ (ASCII) แทนตัวอักษร S ด้วยเลข 53H หรือ 53 ในฐานสิบหก หรือในฐานสองแทนด้วย 1010011 นั่นเอง ดังนั้นในบรรทัดแรกแสดงถึงสถานะเวลา (Time State) ของสัญญาณดิจิตอลที่แทนเลข 101011 ขนาด 7 บิทตามมาตรฐานของรหัสแอสกี้ และส่งออกไปแบบอนุกรม

ในบรรทัดต่อมาแสดงถึงสัญญาณดิจิตอลที่แทนรหัสฐานของ 1010011 โดยมีบิทต่ำสุด (Least Significant Bit:LSB) อยู่ทางซ้ายมือ

ในบรรทัดที่ - 3 แสดงถึงสัญญาณเอาท์พุทที่ได้จาก TTL (Transistor Transistor Logic) เพราะอุปกรณ์จำพวก TTL ใช้กับระดับไฟที่ 5 โวลต์ กับ 0 โวลต์ ดังนั้นจึงแทนลอจิก 1 ด้วยระดับไฟ 5 โวลต์ และลอจิก 0 ด้วยระดับไฟ 0 โวลต์

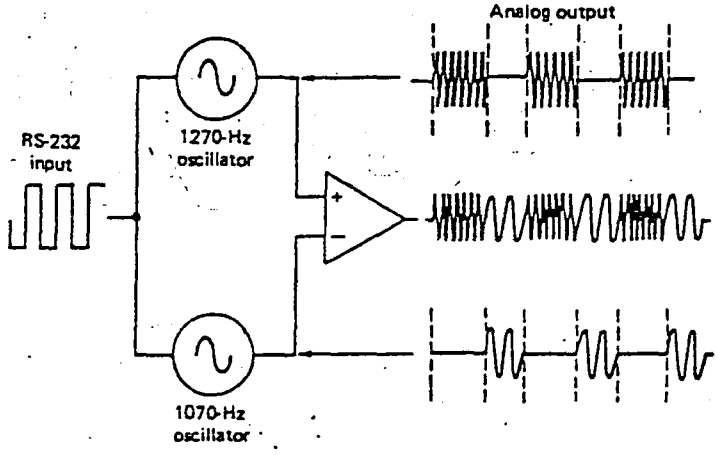
ในบรรทัดที่ 4 แสดงถึงสัญญาณที่ต่อผ่านพอร์ท RS-232C ออกมาโดยใช้กับไฟระดับ +12 V และ -12 V ดังนั้นจึงแทนลอจิก 0 ด้วยระดับไฟ +12 V และลอจิก 1 ด้วยระดับไฟ -12 V

สำหรับในสองบรรทัดสุดท้าย แสดงถึงสัญญาณที่ผ่านการโมดูเลทด้วยเทคนิค FSK ซึ่งส่งผ่านระหว่างโมเด็ม จะสังเกตเห็นว่าในกรณีของเทคนิค FSK นี้ ถ้าระดับบอจิกของสัญญาณมีค่าเป็น 1 ความถี่ของสัญญาณที่ผ่านการโมดูเลทจะสูงขึ้น และถ้าระดับลอจิกของสัญญาณมีค่าเป็น 0 ความถี่ก็จะลดลง ในรูปนี้ความถี่กำหนดเพียงคร่าว ๆ เท่านั้น จากที่กล่าวมาแล้วว่าสัญญาณที่ส่ง/รับใน Originate Modem และ Answer Modem นั้นจะต่างกัน ในกรณี Originate Modem จะส่งสัญญาณ 2 บิทแรก ที่มีระดับลอจิกเป็น 1 ออกไปด้วยความถี่ของสัญญาณที่ผ่านการโมดูเลทแล้วเท่ากับ 1270 Hz ส่วนในบิทที่ 3 และ 4 แทนลอจิก 0 จะส่งออกไปด้วยความถี่ 1070 Hz และในบิทที่ 5 และ 7 ก็จะถูกส่งออกไปด้วยความถี่ 1270 Hz ส่วนในบิทที่ 6 ก็จะถูกส่งด้วยความถี่ 1070 Hz ตามลำดับ ดังนั้นจึงสรุปได้ว่าเทคนิคของ FSK ก็คือเทคนิคในการโมดูเลทสัญญาณดิจิตอลด้วยวิธีฟรีเควนซี่โมดูเลทชั่น (FM) นั่นเอง

แต่ในสถานะที่ Originating Modem ทำหน้าที่รับสัญญาณเข้ามา ความถี่ของสัญญาณที่แทนลอจิก 4 คือในบิทที่ 1,2,5 และบิทที่ 7 จะถูกโมดูเลทส่งออกม่ด้วยความถี่ 2225 Hz ส่วนในบิทที่ 3,4 และบิทที่ 6 จะถูกโมดูเลทส่งออกมาด้วยความถี่ 2025 Hz ซึ่งจุดสำคัญก็คือป้องกันการรบกวนกันระหว่างสัญญาณจาก Originating และ Answering ในสายส่งนั่นเอง

2.7.1 **โมเด็มทรานสมิทเตอร์**

หลักการในการสร้างสัญญาณต่าง ๆ ภายในโมเด็ม



รูปที่ 2.12 โครงสร้างในการกำเนิดสัญญาณ FSK

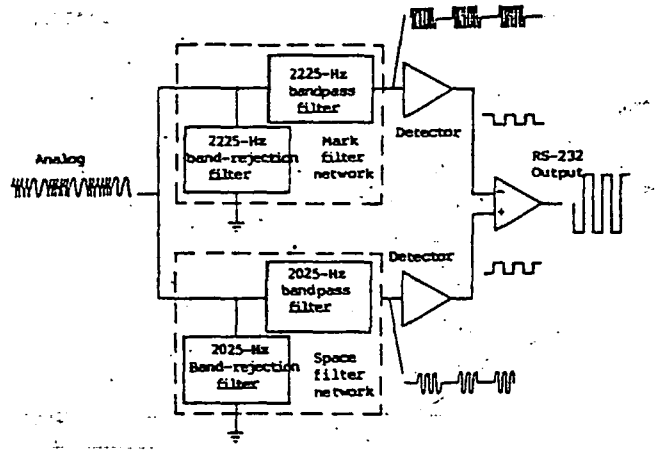
จากรูป จะเห็นว่าในโมเด็มจะประกอบไปด้วย ออสซิสเลเตอร์ที่กำเนิดสัญญาณพาหะ (carrier) ด้วยความถี่ 1270 Hz และ 1070 Hz โดยที่ออสซิลเลเตอร์ตัวบนจะทำงานเมื่อสัญญาณที่เข้ามามีระดับโวลเตจเกิน 5 โวลต์ ส่วนออสซิสเลเตอร์ตัวล่างจะทำงาน และตัวบนจะหยุดทำงาน เมื่อระดับโวลเตจของสัญญาณที่เข้ามาเกินกว่า + 5 โวลต์ เมื่อรับเอาพุที่ออกจากพอร์ท RS-232C เข้ามายังอินพุทของออสซิลเลเตอร์ ก็จะสามารถทำให้ออสซิลเตอร์ทั้งสองตัวทำงานได้ดังนี้

เมื่อสัญญาณที่ผ่านพอร์ท RS-232C เข้ามามีระดับโวลเตจเท่ากับ -12 โวลต์ ออสซิสเลเตอร์ที่ผลิตสัญญาณพาหะความถี่ 1270 Hz ก็จะทำงานแต่ออสซิสเลเตอร์ที่ผลิตสัญญาณพาหะความถี่ 1070 Hz หยุดทำงาน เราทราบแล้วว่าระดับโวลตเตจ -12 V จะแทนลอจิก 1 และเมื่อผ่านการโมดูเลทแล้วความถี่ 1270 Hz ทางด้านโมเด็มทรานมิสเตอร์ก็จะแทนลอจิก 1 นั่นเอง ดังนั้น จากรูป จะแสดงถึงการแปลงสัญญาณจากพอร์ท RS-232C ที่แทนลอจิก 1 ไปเป็นสัญญาณข้อมูลที่แทนลอจิก 1 เช่นกัน

สำหรับในกรณีที่สัญญาณข้อมูลที่จะส่งมีระดับโวลเตจเป็น +12 V ออสซิลเลเตอร์ที่ผลิตความถี่ 1070 Hz ก็จะทำงานแทน และทำการผลิตสัญญาณที่แทนลอจิก 0 ออกมา หลังจากนั้นจึงทำการรวมสัญญาณเอาท์พุทจากออสซิลเลเตอร์ทั้งสองตัวเข้าด้วยกัน โดยท่านเข้าไปยังอินพุทของอ็อปแอม (Operational Amplifier) ก็จะทำให้ได้สัญญาณที่เอาท์พุทเป็นสัญญาณโมดูเลททันที จากรูปจะเห็นว่าสัญญาณข้อมูลก็ยังมีลอจิกเป็น 1 หรือ 0 เช่นเดิม แต่เปลี่ยนรูปแบบจากระดับโวลเตจไปอยู่ในรูปของความถี่แทน คือถ้าเป็นลอจิก 1 ก็จะมีความถี่ของสัญญาณเป็น 1270 Hz และถ้าเป็นลอจิก 0 จะมีความถี่เป็น 1070 Hz

2.7.2 โมเด็มรีซีฟเวอร์ (Modem Receiver)

เมื่อสัญญาณที่ส่งออกมาจากโมเด็มทางด้านส่งผ่านขบวนการโมดูเลทแล้ว เมื่อมาถึงโมเด็มทางด้านรับ สัญญาณเหล่านี้ก็จะผ่านขบวนการดีโมดูเลท แยกเอาสัญญาณพาหะออกจากสัญญาณข้อมูลแล้วผ่านสัญญาณข้อมูลไปใช้งานต่อไป



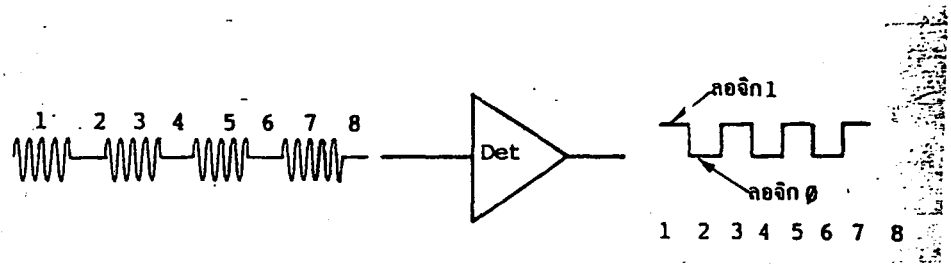
รูปที่ 2.13 โครงสร้างการทำงานของโมเด็มรีซีฟเวอร์

จากรูปจะแสดงถึงวิธีการในการแปลงสัญญาณอะนาล็อก (ที่ส่งผ่านสายโทรศัพท์เข้ามา)กลับไปเป็นสัญญาณที่จะส่งต่อผ่านเข้าไปยังพอร์ท RS-232C ต่อไป ซึ่งแสดงถึงบล็อกไดอะแกรมของวงจรในโมเด็มทางด้านรับ จะเห็นว่า สัญญาณที่ส่งผ่านสายโทรศัพท์แล้วต่อเข้าที่อินพุทของโมเด็มนั้น เป็นสัญญาณที่ประกอบด้วยความถี่สองความถี่ที่ต่างกัน ดังนั้นจึงต้องทำการแยกความถี่ของสัญญาณนี้ออกจากกันก่อน โดยผ่านวงจรฟิลเตอร์ (Filter) สองวงจร ซึ่งหน้าที่ของวงจรฟิลเตอร์หรือวงจรกรองความถี่นี้ก็คือ ทำการแยกสัญญาณที่ประกอบด้วยความถี่หลัก ๆ ให้ออกจากกัน แล้วส่งต่อไปยังส่วนของวงจรที่ทำงานตอบสนองต่อสัญญาณความถี่นั้นเนียงค่าเดียวต่อไป

จากรูปวงจรส่วนบนประกอบด้วยวงจรฟิลเตอร์สองวงจรแยกกัน โดยทำการกรองเอาความถี่ค่าอื่น ๆ ออกไป ยกเว้นสัญญาณความถี่ 2225 Hz และแถบความถี่แคบ ๆ ที่ใกล้เคียงกับ 2225 Hz ไว้ ซึ่งหน้าที่ของแบนด์รีเจคชั่นฟิลเตอร์ ก็คือมีลักษณะเป็นตัวต้านทานต่อสัญญาณความถี่ 2225 Hz ซึ่งหมายความว่าสัญญาณที่รับเข้ามานั้นถ้ามีความถี่เป็นค่าอื่น ๆ นอกจาก 2225 Hz แล้ว ก็จะผ่านฟิลเตอร์ส่วนนี้แล้วไหลลงจุดกราวด์ไป (เอาท์พุทของแบนด์รีเจคชั่นฟิลเตอร์ จะต่อลงจุดกราวด์บางทีจึงเรียกว่าNotchFilter)ส่วนสัญญาณความถี่ 2225 Hz ซึ่งผ่านฟิลเตอร์ส่วนนี้ไม่ได้ก็จะผ่านเข้าไปยังแบนด์พาสฟิลเตอร์ (Bandpass Filter) ต่อไป ซึ่งการทำงานของแบนด์พาสฟิลเตอร์นี้ก็คือ จะยอมให้สัญญาณที่มีความถี่ 2225 Hz เท่านั้นที่ผ่านไปได้ ส่วนสัญญาณความถี่อื่นก็จะถูกเอาไว้ ดังนั้น วงจรฟิลเตอร์ส่วนบนก็มีการทำงานเพื่อต้องการกรองเอาสัญญาณความถี่ 2225 Hz ไปผ่านออกที่เอาท์พุทของแบนด์พาสฟิลเตอร์นั่นเอง

ส่วนในกลุ่มวงจรฟิลเตอร์ส่วนล่างนั้นก็มีการทำงานในลักษณะที่คล้ายคลึงกัน แต่ต่างกันที่มีการทำงานตอบสนองต่อสัญญาณความถี่เป็น 2025 Hz คือการทำงานผลสุดท้ายก็เพื่อต้องการกรองเอาความถี่อื่น ๆ ที่ไม่ใช่ 2025 Hz ไว้แล้วปล่อยให้สัญญาณความถี่ 2025 Hz ไปผ่านออกที่เอาท์พุทของแบนด์พาสฟิลเตอร์

เมื่อได้กรองเอาความถี่ที่ต้องการคือ ความถี่ 2225 Hz (ซึ่งแทนลอจิก 1) และความถี่ 2025 Hz (แทนลอจิก 0) ได้แล้ว ก็จะต่อเอาท์พุทของฟิลเตอร์แบนด์พาสแต่ละตัวเข้ากับวงจรอีกส่วนหนึ่งซึ่งเรียกว่า "ดีเทคเตอร์" (Detector) หน้าที่ของวงจรดีเทคเตอร์ก็คือ การทำดีโมดูเลทสัญญาณ เพื่อแยกเอาสัญญาณพาหะและสัญญาณข้อมูลออกจากกัน แล้วจึงผ่านเอาสัญญาณข้อมูลไปใช้จากในวงจรดังรูป เมื่อมีสัญญาณรูปไซน์ (Sine wave) ผ่านเข้าไปยังวงจรดีเทคเตอร์ก็จะสร้างระดับแรงดันเอาท์พุทของดีเทคเตอร์ก็จะมีค่าเป็นศูนย์ นั่นคือการทำงานดังรูป 2.14



รูปที่ 2.14 การทำงานของดีเทคเตอร์

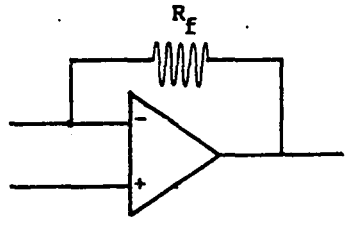
ซึ่งจะหมายความว่าที่ช่วงสัญญาณ 1,3,5 และ 7 ซึ่งมีความถี่ของสัญญาณไซน์เป็น 2225 Hz เมื่อผ่านวงจรดีเทคเตอร์ออกมา ก็จะมีระดับลอจิก 1 ส่วนในช่วงสัญญาณ2, 4, 6 และ 8 เมื่อผ่านดีเทคเตอร์ออกมาจะมีระดับลอจิกเป็น 0 จากรูป จะสังเกตเห็นว่าดีเทคเตอร์แต่ละตัวมีการทำงานเพื่อสร้างสัญญาณดิจิตอล โดยมีเฟสของสัญญาณต่างกันคือ วงจรดีเทคเตอร์ตัวบนจะสร้างสัญญาณลอจิก 1 ออกมา ก็เมื่อสัญญาณไซน์ที่อินพุทแทนสถานะ Mark หรือลอจิก 1 หรือ มีความถี่ 2225 Hz แต่วงจรดีเทคเตอร์ตัวล่าง จะสร้างสัญญาณลอจิก 1 ออกมา ก็ต่อเมื่อสัญญาณไซน์ที่อินพุทแทนสถานะ Space หรือลอจิก 0 หรือความถี่ 2025 Hz เป็นต้น

ในส่วนของวงจรภาคต่อไปก็เป็นการรวมเอาสัญญาณดิจิตอลจากเอาท์พุทของดีเทคเตอร์เข้าด้วยกันโดยใช้ออฟแอมป์ (Operational Amplifier)ซึ่งหน้าที่ของออฟแอมป์หรือ op-Amp นี้ก็คือทำการรวมสัญญาณดิจิตอลจากสองแหล่งเข้าด้วยกัน แล้วเปลี่ยนสัญญาณให้อยู่ในรูปของสัญญาณมาตรฐานที่สามารถส่งผ่านพอร์ท RS-232C (+12 V)ซึ่งวิธีการในการเปลี่ยนสัญญาณดังกล่าวนี้ก็โดยการต่อเอาท์พุทของสเปส (Space) ดีเทคเตอร์ (หรือดีเทคเตอร์ตัวล่าง) เข้ากับขา + หรือขานอนอินเวิทติ้ง (Non inverting) ของ op-Amp และต่อเอาท์พุทของมาร์ค (mark) ดีเทคเตอร์ (ดีเทคเตอร์ตัวบน) เข้ากับขาอินเวอร์ทติ้ง หรือขาลบของ op-Amp ซึ่ง op-Amp สร้างสัญญาณเอาท์พุทขึ้นมาโดย

-เมื่อเอาท์พุทของสเปสดีเทคเตอร์ แอคทีฟ, op-Amp จะสร้างระดับแรงดันค่า + ขึ้นที่เอาท์พุท

-เมื่อเอาท์พุทของมาร์กดีเทคเตอร์ แอคทีฟ, op-Amp จะสร้างระดับแรงดันค่า - ขึ้นที่เอาท์พุท

นอกจากนี้อาจจะมีการใช้เทคนิคการฟีดแบ็ค (Feedback) โดยต่อตัวต้านทานฟีดแบ็คเข้าขานอนอินเวอร์ทติ้งของ op-Amp เพื่อทำให้การปรับเกน (Gain) ของระบบดีขึ้น สามารถสร้างสัญญาณระดับแรงดันจากยอดถึงยอด (peak to peak) เท่ากับ 24 V (จาก -12 ถึง +12 V)



รูปที่ 2.15 การต่อตัวต้านทานฟีดแบ็ค

ซึ่งมีผลถึงการต่อสัญญาณข้อมูลไปใช้ส่งผ่านพอร์ท RS-232C ได้ดีขึ้น

โมเด็มชนิดความเร็วปานกลางและความเร็วสูง

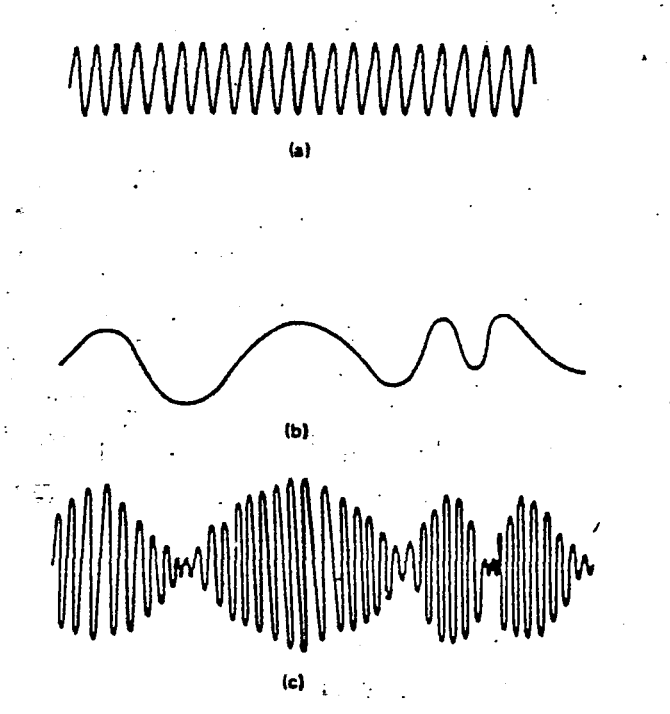
เทคนิคการโมดูเลท

เทคนิคต่าง ๆ ที่มีการนำมาใช้งานในอุปกรณ์โมเด็มก็เพื่อทำให้โมเด็มสามารถส่งข้อมูลด้วยอัตราการส่ง (บิทต่อวินาที) ที่สูงขึ้น สำหรับโมเด็มที่ใช้เทคนิคการโมดูเลทแบบ FSK นั้นจะมีอัตราการส่งข้อมูลสูงสุดเพียง 600 บิทต่อวินาทีเท่านั้น

เทคนิคการโมดูเลทที่ใช้ในโมเด็มชนิดอัตราการส่งข้อมูลปานกลาง และสูงนั้นจะมีข้อยุ่งยากขึ้นมาอีกขั้นหนึ่งคือการโมดูเลทแบบ AM หรือแอมปลิจูดโมดูเลชั่น (Amplitude Modulation) และแบบ FM หรือหรีเควนซี่ โมดูเลชั่น (Frequency Modulation) ซึ่งเป็นเทคนิคการโมดูเลท ที่นิยมใช้กันทั่ว ๆ ไป ในการส่งกระจายคลื่นของสถานีวิทยุกระจายเสียงต่าง ๆ แต่การนำมาประยุกต์ใช้ในอุปกรณ์โมเด็มนั้น จะต้องใช้ร่วมกับเทคนิคการโมดูเลทแบบอื่น ๆ เช่น เฟสโมดูเลชั่น (Phase Modulation) หรือ PM แต่ก่อนอื่นเราจะมาทำความเข้าใจกับวิธีการของ AM และ FM เสียก่อน ก่อนที่จะกล่าวละเอียดไปถึง PM

แอมปลิจูดโมดูเลชั่น (AM)

เป็นเทคนิคการโมดูเลทสัญญาณเสียงเข้ากับสัญญาณพาหะ (Carrier Signal) ซึ่งเป็นสัญญาณรูปไซน์ (Sine Wave) ที่มีความถี่สูง ทำให้สัญญาณที่ผ่านการโมดูเลทแล้วมีลักษณะการเปลี่ยนแปลงของสัญญาณพาหะตามแอมปลิจูด (ความสูงของสัญญาณ) ของสัญญาณเสียงแต่ความถี่ของสัญญาณพาหะยังคงที่ ดังรูป



รูปที่ 2.16 แสดงลักษณะของสัญญาณตามวิธีแอมปลิจูดโมดูเลชั่น

a) สัญญาณพาหะ (Carrier)

b) สัญญาณเสียงที่จะนำเข้าไปโมดูเลท

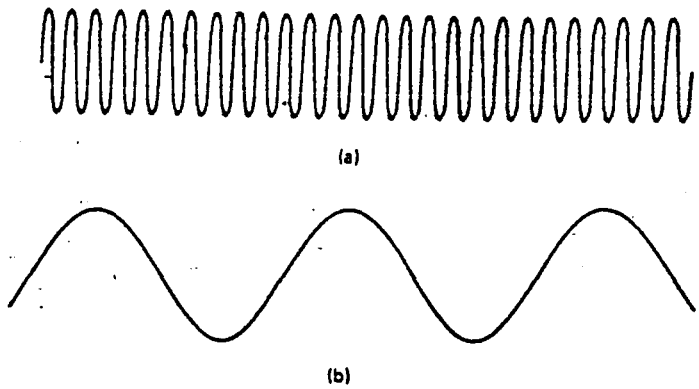
c) สัญญาณที่ผ่านการโมดูเลทแล้ว

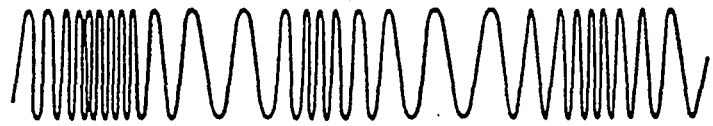
จุดสำคัญของการโมดูเลทชนิดนี้ก็อยู่ที่ "แอมปลิจูดเปลี่ยนแปลง แต่ความถี่คงที่"

สำหรับหลักการทำงานของโมเด็มที่ใช้เทคนิคโมดูเลทแบบ AM นั้น ในส่วนของวงจรทางด้านส่งจะต้องทำการแปลงสัญญาณดิจิตอลไปเป็นสัญญาณอะนาล็อกก่อน โดยใช้ D/A converter (Digital convert to Analog) แล้วผ่านขบวนการโมดูเลทส่งออกไป ส่วนในโมเด็มทางด้านรับเมื่อรับสัญญาณเข้ามา ก็จะผ่านขบวนการดีโมดูเลท แยกเอาสัญญาณพาหะออกแล้วจึงส่งผ่านวงจร A/D converter (Analog to Digital Converter) เพื่อแปลงสัญญาณอะนาล็อกไปเป็นสัญญาณดิจิตอล แล้วส่งเข้าเทอร์มินัลใช้งานต่อไป

ฟรีเควนซีโมดูเลชั่น (Frequency Modulation : FM)

หลักการของการโมดูเลททางความถี่นี้จุดสำคัญอยู่ที่ "แอมปลิจูดของสัญญาณจะคงที่แต่ความถี่ของสัญญาณพาหะจะเปลี่ยนแปลงไปตามแอมปลิจูดของสัญญาณที่นำเข้าโมดูเลท (Modulation Signal)" ดังรูป





(c)

รูปที่ 2.17 ฟรีเควนซี่โมดูเลชั่น

a) สัญญาณพาหะ

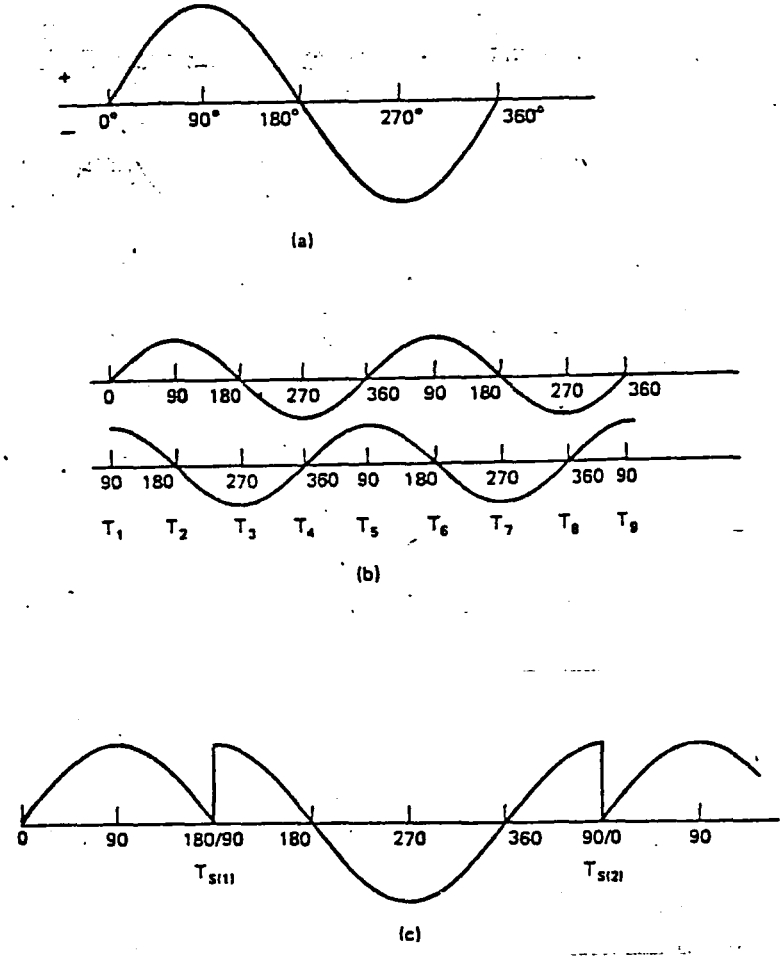
b) สัญญาณอะนาล็อกที่นำเข้ามาโมดูเลท

c) สัญญาณที่ผ่านการโมดูเลทแล้ว

เมื่อพิจารณารูป (a) และรูป (b) ประกอบกัน ในขณะที่แอมปลิจูดของสัญญาณอะนาล็อกมีค่าเป็นบวก ความถี่ของสัญญาณที่ผ่านการโมดูเลทจะสูงขึ้น แต่ในขณะที่แอมปลิจูดของสัญญาณอะนาล็อกมีค่าเป็นลบ ความถี่ของสัญญาณที่ผ่านการโมดูเลทจะลดลง

เฟสโมดูเลชั่น (Phase Modulation : PM)

คำว่าเฟส (Phase) ในที่นี้หมายถึงว่า ถ้ามีสัญญาณอะนาล็อก 2 สัญญาณเดินทางไปแต่แตกต่างกันด้วยค่าเวลาค่าหนึ่ง จุด ๆ เดียวกันของสัญญาณคู่นี้ตลอดเวลาจะพบว่าจุดที่สังเกตุบนคลื่นสัญญาณทั้งสองนี้มีมุมที่แตกต่างกันเสมอ ซึ่งมุมผลต่างนี้เราเรียกว่า "มุมเฟส" (Phase Angle)



รูปที่ 2.18 หลักการของมุมเฟสและ PM

a) สัญญาณไซน์ที่เดินทางไป 1 คาบเวลา (Period)

b) สัญญาณไซน์สองสัญญาณที่มีความต่างเฟสกัน 90 องศา

c) สัญญาณที่เฟสเลื่อนไป 90 องศา

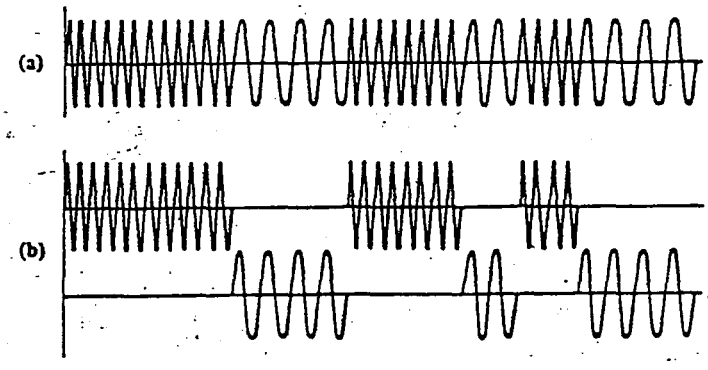
จากรูปที่ 2.18 (a) แสดงสัญญาณอะนาล็อกให้ 1 คาบเวลา (Period) โดยจะแบ่งช่วงเวลาของ สัญญาณออกเป็นมุมได้ตั้งแต่ 0 ถึง 360 องศา จุดเริ่มต้นของสัญญาณไซน์อยู่ที่ 90 องศา และมีแอมปลิจูดเป็น 0 และในเวลา 1/4 ของคาบเวลา แอมปลิจูดจะมีค่าสูงสุดทางบวก ซึ่งตรงกับมุม 90 องศา แล้วแอมปลิจูดจะลดลงเป็น 0 อีกที่มุม 180 องศา หรือ 1/2 ของคาบเวลา ต่อมาแอมปลิจูดจะมีค่าสูงสุดทางลบที่มุม 270 องศา แล้วกลับมามีแอมปลิจูดเป็น 0 อีกครั้งหนึ่งเมื่อครบ 1 คาบเวลา วนเวียนเป็นวัฏจักรอย่างนี้เรื่อยไป

สำหรับในกรณีมีการเปรียบเทียบสัญญาณไซน์ 2 สัญญาณ เราก็ใช้หลักการพิจารณาอย่างเดียวกันคือ ยึดเอาจุดใดจุดหนึ่งบนสัญญาณนั้น ๆ เป็นหลัก เช่นในรูปที่ 2.18 (b) จะเห็นว่าสัญญาณคู่นี้มีความต่างเฟสกันอยู่ด้วยมุมเฟสค่าหนึ่ง ที่เวลา T_1 สัญญาณอันล่างจะมีความต่างเฟสกับสัญญาณอันบนเท่ากับ 90 องศา ที่เวลา T_3 สัญญาณอันล่างมีเฟสอยู่ที่ 270 องศา ในขณะที่สัญญาณอันบนมีเฟสอยู่ที่ 180 องศา นั่นคือมีความต่างเฟสอยู่ 90 องศสรุปได้ว่าสัญญาณคู่นี้มีความต่างเฟสของสัญญาณอยู่ค่า ๆ หนึ่งที่คงที่คือเท่ากับ 90 องศานั่นเอง

เทคนิคปลีกย่อยของเฟสโมดูเลชั่นที่จะกล่าวถึงต่อไปนี้ก็คือ Phase Shift Keying (PSK) และ Phase Amplitude Modulation(PAM)รวมถึง Frequency Shift Keying (FSK) ซึ่งมีรายละเอียดดังต่อไปนี้

FREQUENCY-SHIFT KEYING (FSK)

หลักการคือ ความถี่ของสัญญาณพาหะจะแปรเปลี่ยนตามสัญญาณดิจิตอลที่เข้ามาคือ ค่าสัญญาณมีระดับลอจิกเป็น 1 ความถี่ของสัญญาณพาหะจะสูงขึ้น แต่ค่าสัญญาณมีระดับลอจิกเป็น 0 ความถี่ของสัญญาณพาหะจะลดลง ดังรูป โดยเราสามารถแยกข้อแตกต่างของ FSK จาก FM ได้ว่า ในขบวน FSK คลื่นพาหะอาจจะมีความถี่ของคลื่นได้มากกว่า 2 ความถี่ แต่ใน FM จะมีความถี่ของคลื่นพาหะได้เพียง 1 ความถี่เท่านั้น



รูปที่ 2.19 แสดงสัญญาณที่ได้จากเทคนิค FSK

จากรูปจะเห็นว่าเราสามารถจะแยกสัญญาณ FSK ออกเป็นสัญญาณ ออกเป็นสัญญาณ Amplitude Shift Keying (ASK) ได้สองสัญญาณ

PHASE SHIFT KEYING (PSK)

หลักการคือ เฟสของสัญญาณพาหะจะเปลี่ยนไปตามสัญญาณดิจิตอลที่เข้ามาคือ ถ้าระดับลอจิกของสัญญาณเปลี่ยนจาก 0 เป็น 1 เฟสของสัญญาณก็จะเปลี่ยนไป 90 องศา

จากรูปที่ 2.18 (c) จะเห็นว่าเฟสของสัญญาณหลังจาก 180 องศา แล้วควรจะเป็น 270 องศา แต่สัญญาณจะกลับเฟสไปเป็น 90 องศาแทน เช่นที่เวลา Ts(1) และที่เวลา Ts (2) เฟสของสัญญาณในช่วงต่อไปควรจะเป็น 180 องศา แต่ปรากฏว่าเฟสของสัญญาณกลับมีค่าเป็น 0 องศา เพราะฉะนั้นสรุปได้ว่าการเลื่อนเฟสจะมีค่าเป็น 90 องศานั่นเอง

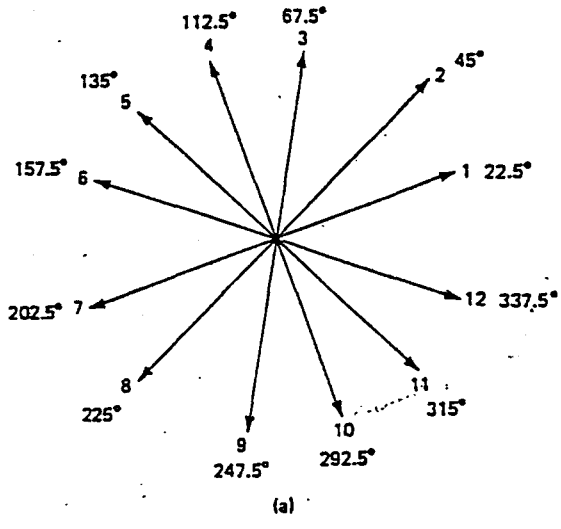
ในลักษณะเช่นนี้ ถ้าเราสามารถเลื่อนเฟสของสัญญาณไซน์ออกเป็นค่าต่าง ๆ เช่น อาจจะต่าง ๆ กันสี่ค่า ประสิทธิภาพของโมเด็มย่อมจะสูงขึ้น ซึ่งในโมเด็มชนิดที่ใช้เทคนิค PSK ก็ใช้หลักการนี้เช่นกัน เช่นในโมเด็ม Bell 201 B มีการเลื่อนสัญญาณไปเป็น 45°, 135°, 225° และ 315° ซึ่งค่าของเฟสที่เลื่อนไปจะสัมพันธ์กับการวนรอบของเฟสอื่น ๆ ด้วยมุมเฟสต่าง ๆ กันทั้ง 4 ค่านี้จะใช้แทนบิทข้อมูลได้ 4 บิท (แทนที่จะเป็น 1 บิท เช่นใน FSK) คือ เป็น ค่า 11, 01, 10 และ 11 ตามลำดับ ซึ่งการเรียงบิทในลักษณะนี้เรียกว่า dibits ซึ่งมีผลดีคือเป็นเทคนิคที่ทำให้สามารถเพิ่มอัตราโบดได้สูงยิ่งขึ้น เช่น ถ้าโมเด็มสามารถส่งข้อมูลในอัตรา 600 dibits ต่อ วินาที (600 โบด) ก็จะมีค่าเท่ากับโมเด็มส่งข้อมูลด้วยอัตรา 1200 บิท/วินาที ในปัจจุบันขีดความสามารถของโมเด็มได้ถูกพัฒนาขึ้นอย่างมาก เช่น โมเด็มรุ่น Bell 201C มีความสามารถในการเลื่อนมุมเฟสได้ต่าง ๆ กันถึง 8 ค่า ซึ่งในมุมเฟสค่าหนึ่งใช้แทนบิทข้อมูลได้ 3 บิท ในกรณีนี้อัตราการส่งข้อมูลยิ่งสูงขึ้นอย่างมาก สามารถส่งข้อมูลได้ถึง 4800 บิท/วินาที เช่นในกรณีที่มีอัตราของสัญญาณในสายส่ง (Line Signaling Rate) มีค่าเป็น 1600 โบด โมเด็มจะมีอัตราการส่งข้อมูลได้ถึง 4800 บิท/วินาที ซึ่งผู้อ่านจะสับสนระหว่างอัตราบิทต่อวินาที อัตราโบด และอัตราสัญญาณในสายส่ง ดังนั้นจึงได้แสดงรายละเอียดได้ในตารางต่อไปนี้

Modulation technique	Line Signaling rate, bauds	Bit rate, bps
FSK	n(300)	n(300)
Dibits	n(1200)	2n(2400)
Tribits	n(1200)	8n(9600)
Quadbits	n(1200)	16n(19,200)

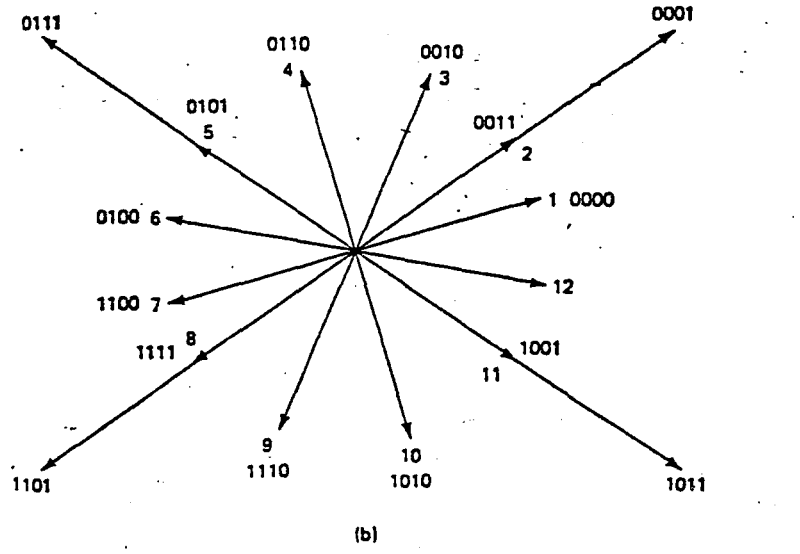
ตารางที่ 2.4

PHASE AMPLITUDE MODULATION (PAM)

เป็นเทคนิคที่เกิดจากการรวมเอาสัญญาณโมดูเลทที่เกิดจากเทคนิค PSK และ AM เข้าด้วยกัน ทำให้ได้อัตราการส่งสูงขึ้นคือสามารถส่งบิทข้อมูลได้ถึง 16 แบบต่าง ๆ กันดังรูป



(a)



รูปที่ 2.20 PAM

a) มุมเฟสต่าง ๆ กัน 12 มุม

b) การเข้ารหัสขนาด 4 บิท

ในรูปที่ 2.20 จะเห็นว่ามีการแบ่งมุมเฟสออกเป็น 12 มุม สำหรับใช้ในการโมดูเลท แบบ PAM และถ้าเราเพิ่มขนาดของแอมปลิจูดให้แก่มุมเฟส 4 มุมจากทั้งหมดทำให้มุมเฟส 4 มุมที่เราเพิ่มแอมปลิจูดเข้าไปนั้น มีแอมปลิจูดถึง 2 ค่า ดังนั้นเกิดมุมเฟสขึ้นอีก 4 ค่ารวมเป็น 16 ค่า ดังรูป 2.20 (b) ซึ่งมุมเฟสแต่ละค่าก็ถูกใช้ในการแทนข้อมูลกลุ่มหนึ่ง เพราะฉะนั้น จึงสามารถแทนกลุ่มข้อมูลได้ทั้งหมด 16 แบบต่าง ๆ กัน ดังนั้นเทคนิควิธีนี้จึงทำให้อัตราการส่งข้อมูลขึ้นถึง 9600 บิทต่อวินาที ส่วนใหญ่โมเด็มประเภทนี้ จะใช้กับการส่งข้อมูลแบบซิงโครนัส

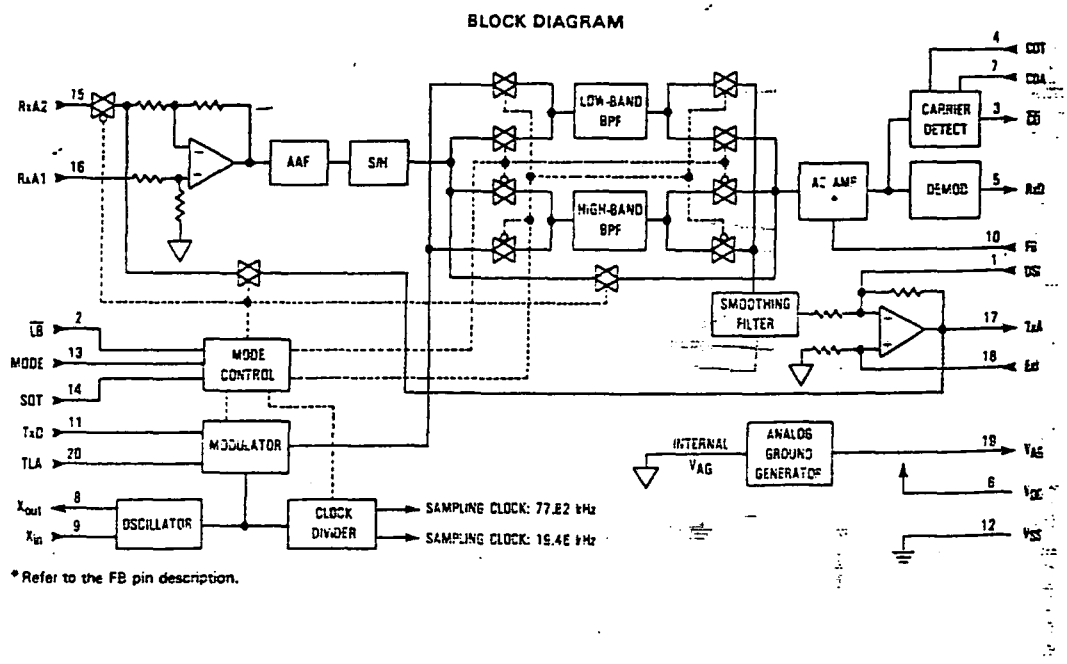
บทที่ 3

การออกแบบและการสร้าง

เนื่องจากโครงงานนี้ประกอบไปด้วยอุปกรณ์ 2 ส่วนด้วยกันคือ ส่วน packet radio (TNC) เพื่อใช้ในการติดต่อระหว่างคอมพิวเตอร์กับคอมพิวเตอร์หรือระหว่างคอมพิวเตอร์กับส่วนควบคุมย่อยโดยผ่านทางเครื่องวิทยุสมัครเล่น และส่วนควบคุมย่อยที่มีหน้าที่ในการเก็บข้อมูลจากบริเวณต่าง ๆ และส่งข้อมูลนั้นกลับมาให้กับผู้สั่งงานได้รู้เมื่อต้องการ โดยควบคุมผ่านทางเครื่องวิทยุสมัครเล่น โดยใช้ packet radio ช่วย

3.1 ส่วน packet modem

การออกแบบส่วนนี้ได้เลือกใช้ IC สำเร็จรูปเบอร์ MC 145442 ของบริษัท MOTOROLA ซึ่งเป็น IC CMOS single-chip low-speed modem โดยมี baud rate เท่ากับ 300 มาทำหน้าที่ในการแปลงสัญญาณดิจิตอลเป็นสัญญาณเสียง ซึ่งภายในประกอบด้วยส่วน frequency shift keying(FSK) modulator,demodulator และ filter โดย IC นี้เป็น single-chip modem ที่เป็นไปตามมาตรฐาน CCITT V.21 โดย IC นี้สามารถใช้ได้ทั้งแบบ full-duplex หรือ half-duplex ขนาด 300 baud เพื่อใช้ในการรับส่งข้อมูลผ่านสายโทรศัพท์ แต่ในที่นี้ได้ปรับปรุงพัฒนาให้ใช้ผ่านเครื่องวิทยุสมัครเล่นและ IC เบอร์นี้ยังมีส่วน carrier detect circuit ในส่วนของ demodulator



รูปที่ 3.1 แสดง block diagram ของ MC 145442

ขาสัญญาณต่าง ๆ ของ IC เบอร์ MC 145442 มีหน้าที่ดังนี้

ขา 6 - V_{DD} (positive power supply) ต่อกับไฟแรงดันขนาด 5 V

ขา 12 - V_{SS} (negative power supply) ต่อลง ground

ขา 19 - V_{AG} (analog ground) analog ground ถูก bias จากภายในให้มีค่าเท่ากับ $(V_{DD}-V_{SS})/2$ โดยขานี้จะถูก decoupling โดยตัวเก็บประจุระหว่าง V_{AG} กับ V_{SS} และระหว่าง V_{AG} กับ V_{DD} โดยใช้ค่าตัวเก็บประจุเท่ากับ 0.1 μF

ขา 20 - TLA (transmit level adjust) ขานี้ใช้เพื่อปรับระดับการของส่ง โดยต่อ R_{TLA} คร่อมระหว่าง ขา TLA กับ V_{DD} จะได้ค่า transmit level ตามตารางที่ 3.1

Output Transmit Level (Typical into 600 Ω)	R_{TLA}
−12 dBm	∞
−11 dBm	19.8 kΩ
−10 dBm	9.2 kΩ
−9 dBm	5.5 kΩ

ตารางที่ 3.1 แสดงค่า transmit level

ขา 11 -TxD (transmit data) เป็นขา input เพื่อรับข้อมูลเลขฐาน 2 ที่ต้องการจะส่ง โดย level high จะใช้แทนว่า mark และ level low ใช้แทนว่า space โดยความถี่ที่แปลงได้จะเป็นไปตามตารางที่ 3.2

Frequency Characteristics

Bell 103 (MC145443)				
Data	Originate Mode		Answer Mode	
	Transmit	Receive	Transmit	Receive
Space	1070 Hz	2025 Hz	2025 Hz	1070 Hz
Mark	1270 Hz	2225 Hz	2225 Hz	1270 Hz
CCITT V.21 (MC145442)				
Data	Originate Mode		Answer Mode	
	Transmit	Receive	Transmit	Receive
Space	1180 Hz	1850 Hz	1850 Hz	1180 Hz
Mark	980 Hz	1650 Hz	1650 Hz	980 Hz

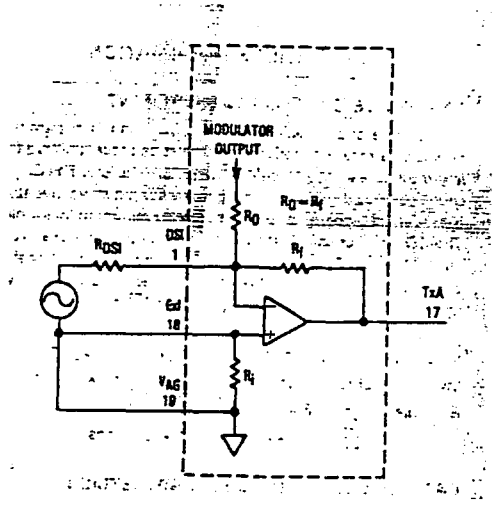
NOTE: Actual frequencies may be ± 5 Hz assuming a 3.579545 MHz crystal is used.

ตารางที่ 3.2 แสดงค่าความถี่ของ mode ต่าง ๆ ของ MC 145442

ขา 17 -TxA (transmit carrier) เป็นขา output ของ line driver amp. โดยจะให้ output ออกมาเป็น sine wave โดยใช้ crystal oscillator เป็น reference เมื่อใช้ crystal ความถี่ 3.579 MHz จะได้ค่าความถี่ตามตารางที่ 3.2

ขา 18 -ExI (external input) เป็นขา noninverting input ของ line driver เพื่อใช้ในการรวมสัญญาณ auxiliary หรือ speech signal ผ่านเข้าไปในสายโทรศัพท์ โดยใช้ line driver ถ้าไม่ได้ใช้ขานี้ควรจะต่อไปที่ V_{AG}

ขา 1 -DSI (driver summing) ใช้ต่อกับสัญญาณภายนอกเช่น DTMF โดยต่อ R_{DSI} อนุกรมเพื่อกำหนด gain (A_v) โดยเมื่อให้สัญญาณเข้าที่ขา DSI นี้ MC 145442 จะต้องทำงานอยู่ในโหมด squelch โดยให้ SQT (ขา 14) เป็น level high โดยสามารถหา gain ได้จากสูตร $A_v = -R_f/R_{DSI}$ (โดย R_f = 20 k) ถ้าไม่ใช้ให้ปล่อยขานี้ลอยไว้ โดยวงจรภายในของ driver summing แสดงดังรูป 3.2



รูป 3.2 แสดงวงจรภายในของ driver summing

ขา 5 -RxD (receive data) เป็นขา output โดยให้ข้อมูลออกมาเป็นสัญญาณดิจิตอล ซึ่งข้อมูลที่ได้ออกมาจากส่วน demodulation ของวงจร receive carrier ถ้าไม่มี carrier เข้ามา ขา CD จะเป็น level high และขาRxD จะเป็น level high ด้วย

ขา 15,16 - RxA2,RxA1 (receive carrier) เป็นขา input เพื่อรับ FSK เข้ามาเพื่อทำการ demodulation ผ่านส่วน band-pass filter โดย RxA1 เป็นขา noninverting และ RxA2 เป็นขา inverting input ของ receive hybrid (duplexer) operational amplifier

ขา 2 -LB (analog loopback) เมื่อให้ level high กับขานี้ (SQT ต้องเป็น low) analog loopback test จะทำงานโดย TxA จะต่อกับ RxA2 และ RxA1 ต่อกับ analog ground โดย demodulator frequency ถูก switch เป็น modulation frequency ตามแต่การเลือกโหมด ดังตาราง 3.2,3.3 และตามรูป 3.5c และ 3.5d

เมื่อ LB ต่อกับ analog ground (V_{AG}) ตัว modulator จะ generate สัญญาณ echo concellation tone ความถี่ 2100 Hz ในการปฏิบัติงานปกติ ขานี้จะต่อกับ V_{ss}

ส่วน power-down mode จะทำงานเมื่อ ขา LB และขา SQT เป็น level high ตามตารางที่ 3.3

MODE Pin 13	SQT Pin 14	LB Pin 2	Operating Mode
1	0	0	Originate Mode
0	0	0	Answer Mode
X	0	V_{AG} (V_{DD}/2)	Echo Tone
X	0	1	Analog Loopback
X	1	0	Squelch Mode
X	1	V_{AG} (V_{DD}/2)	Squelch Mode
X	1	1	Power Down

ตารางที่ 3.3 แสดงโหมดการทำงานต่าง ๆ ของ MC 145442

ขา 13 -MODE เป็นขา input ใช้เลือกความถี่ที่จะใช้ในการส่ง และ รับข้อมูลของภาค modulation และ demodulation ตามตารางที่ 3.3

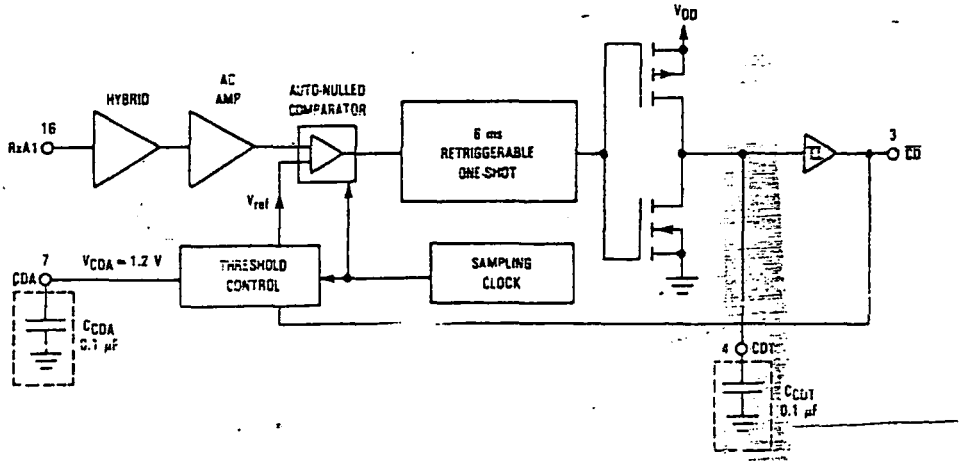
ขา 4 -CDT (carrier detect timing) โดยค่าตัวเก็บประจุที่ต่อระหว่างขานี้ กับ V_{ss} ใช้เพื่อกำหนดว่ามี carrier มานานเท่าใดก่อนที่ขา CD จะเป็น low โดย C_{CDT} มีหน่วยเป็น μF และ t_{CDX} มีหน่วยเป็นวินาที จะได้ว่า

Valid signal to CD response time : $t_{CDL} = 6.4*C_{CDT}$

Invalid signal to CD off time : $t_{CDH} = 0.54*C_{CDT}$

ขา 3 -CD (carrier detect) เป็นขา output เพื่อใช้บอกว่ามี carrier ที่มีความแรงของสัญญาณตั้งแต่ระดับ threshold level ขึ้นไป (หาได้จาก CDA ขา 7) เข้ามาที่

วงจร hybrid นานตามช่วง response time โดยจะได้ output ออกมาเป็น low level ตามรูปที่ 3.3 เป็นวงจร carrier detect



รูปที่ 3.3 เป็นวงจร carrier detect

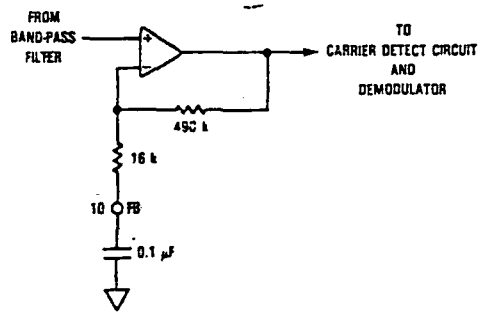
ขา 7 -CDA (carrier detect adjust) สามารถที่จะให้ voltage จากภายนอกเข้าไปที่ขานี้เพื่อปรับค่า carrier detect threshold โดยค่า threshold hysteresis จากภายในจะกำหนดค่าที่ 3 dB โดยค่า

$$V_{CDA} = 244*V_{on}$$

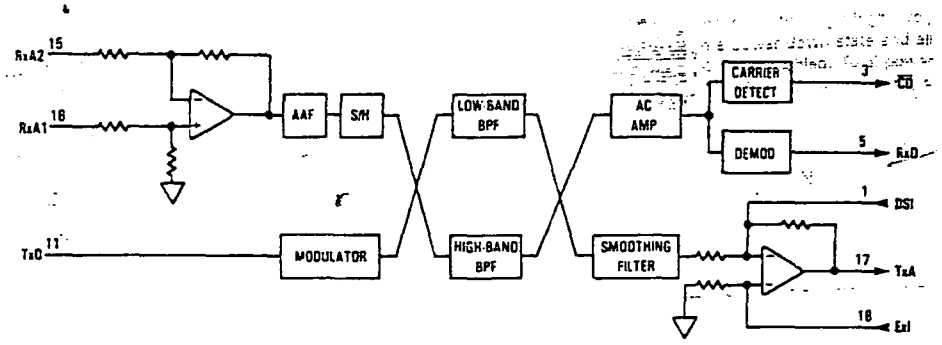
$$V_{CDA} = 345*V_{off}$$

ขา 10 -FB (filter bias) เป็นขา negative input ของ ac amplifier ในการทำงานปกติจะต่อค่าตัวเก็บประจุ bypass ขนาด 0.1 µF คร่อมระหว่างขานี้กับ V_{AB} ซึ่งแสดงไว้ดังรูป 3.4

ขา 14 -SQT (transmit squelch) เมื่อขานี้เป็น high ภาค modulation จะไม่ทำงานโดย line driver จะทำงานถ้า LB เป็น low ตามตาราง 3.3 เมื่อ LB และ SQT เป็น high ตามตาราง 3.3 จะเปลี่ยนเป็น power down state ทำให้กระแสลดลงจาก 10 mA เหลือเพียง 300 µA โดยวงจร oscillator จะถูกตัดการทำงานไป

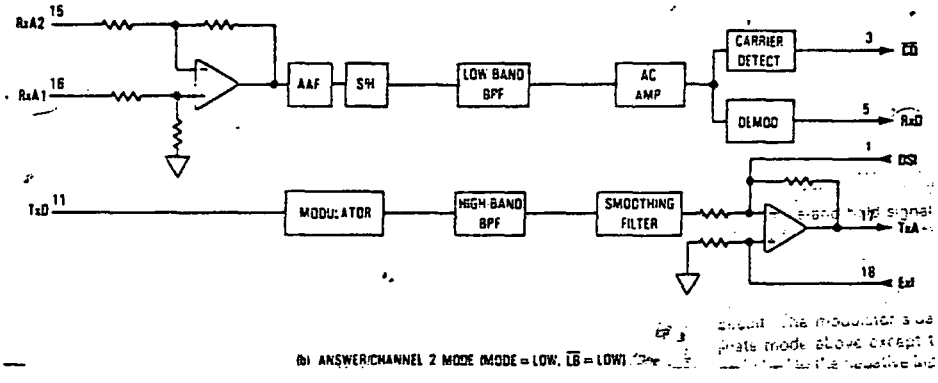


รูปที่ 3.4 แสดงวงจรภายในของส่วน filter bias



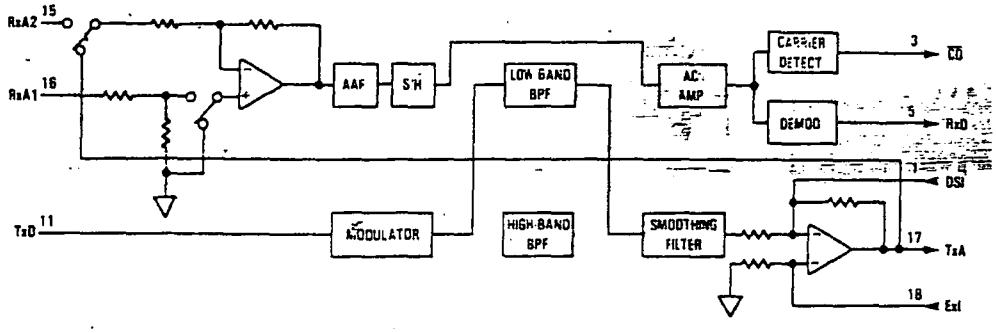
(a) ORIGINATE/CHANNEL 1 MODE (MODE = HIGH, $\overline{LB}$ = LOW)

รูปที่ 3.5a โหมด originate



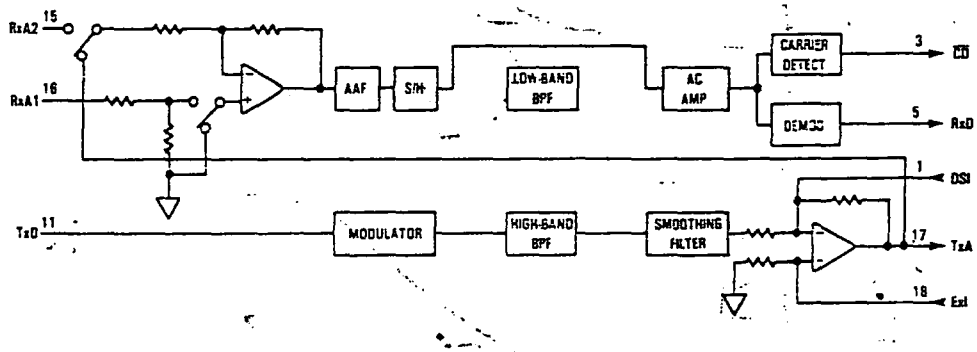
(b) ANSWER/CHANNEL 2 MODE (MODE = LOW, $\overline{LB}$ = LOW)

รูปที่ 3.5b โหมด answer



(c) ORIGINATE/CHANNEL 1 MODE AND ANALOG LOOPBACK STATE (MODE = HIGH, $\overline{LB}$ = HIGH)

รูปที่ 3.5c โหมด originate และ analog loopback



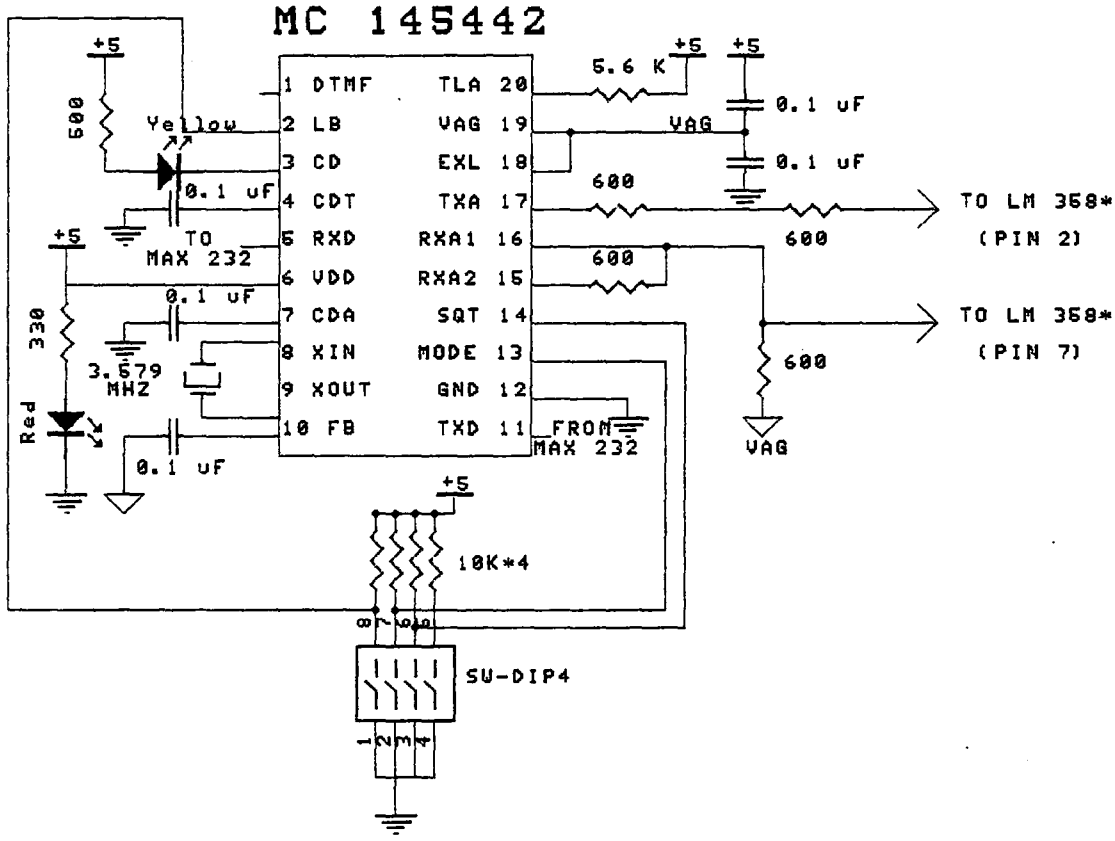
(d) ANSWER/CHANNEL 2 MODE AND ANALOG LOOPBACK STATE (MODE = LOW, $\overline{LB}$ = HIGH)

รูปที่ 3.5d โหมด answer และ analog loopback

รูปที่ 3.5 แสดง block diagram ของโหมดการทำงานทั้งหมด

<u>การออกแบบวงจรเพื่อทำ packet modem</u> แยกเป็นวงจรส่วนย่อยได้ดังนี้

1.ในส่วนของตัว modem ใช้ IC เบอร์ MC 145442 ทำการต่อวงจรดังรูป 3.6



รูปที่ 3.6 วงจรของ MC 145442

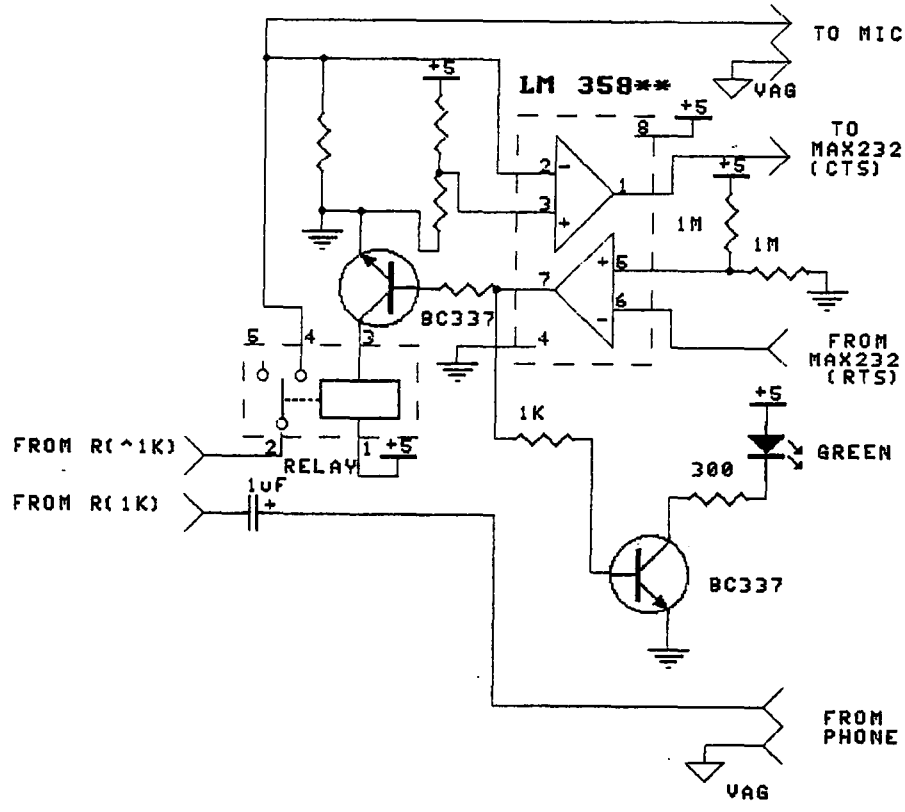
โดย dipswitch หมายเลข 1 ไม่ได้ใช้งาน

หมายเลข 2 ต่ออยู่กับขา LB โดยเมื่อปรับไปที่ on จะให้ level low

หมายเลข 3 ต่ออยู่กับขา MODE โดยเมื่อปรับไปที่ on จะให้ level low

หมายเลข 4 ต่ออยู่กับขา SQT โดยเมื่อปรับไปที่ on จะให้ level low

4.วงจรควบคุม รีเลย์ และ ส่งสัญญาณ CTS กลับไปให้คอมพิวเตอร์ ในการติดต่อระหว่างคอมพิวเตอร์กับ packet modem นั้น เมื่อคอมพิวเตอร์ต้องการจะส่งข้อความคอมพิวเตอร์จะส่งสัญญาณ RTS ออกมาก่อน โดยให้ระดับแรงดันเป็น low เมื่อ packet modem รับสัญญาณได้จะไปสั่งให้ รีเลย์ทำงาน และเมื่อหน้าสัมผัสของรีเลย์แตะแล้วก็จะมีวงจรส่วนตรวจสอบและจะให้สัญญาณเป็น low ให้แก่ MAX 232 เพื่อตอบกลับไปยังคอมพิวเตอร์เป็นสัญญาณ CTS จากนั้นคอมพิวเตอร์ก็จะส่งข้อความที่ต้องการส่งออกมาทีละตัวทางขา Tx ส่วนขา Rx นั้นเมื่อ modem รับสัญญาณได้ ก็จะส่งข้อความไปให้คอมพิวเตอร์ทันที โดยผ่านขา Rx นี้ ซึ่งวงจรในส่วนควบคุมทั้งหมดนี้แสดงดังรูป 3.9



รูปที่ 3.9 วงจรควบคุมรีเลย์และส่งสัญญาณ CTS กลับ

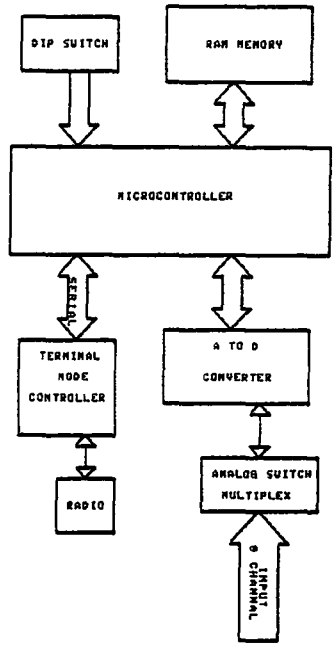
ในส่วนของ LED นั้น สีแดงแสดงว่ามีการจ่ายไฟให้กับ modem แล้ว

สีเขียวแสดงว่ากำลังส่งข้อมูลออกอากาศ

สีเหลืองแสดงว่ารับสัญญาณ carrier ได้

3.2 **ส่วนควบคุมย่อย**

การออกแบบส่วนควบคุมย่อยที่ได้ออกแบบไว้มีหน้าที่ในการอ่านข้อมูลจาก senser ทั้งหมด 8 ชุด และนำสัญญาณที่ได้ซึ่งเป็นสัญญาณ อะนาล็อกมาทำการแปลงให้เป็นสัญญาณ ดิจิตอลโดยใช้ตัว A TO D CONVERTER(ADC) และควบคุมการอ่านข้อมูลจาก ADC โดยใช้ ไมโครคอนโทรลเลอร์ และส่วนควบคุมย่อยนี้จะติดต่อกับ คอมพิวเตอร์หลักได้โดยผ่านทางวิทยุสมัครเล่น จึงต้องสร้างส่วน terminal node controller ขึ้นบนส่วนควบคุมย่อยนี้ด้วยโดยให้ ไมโครคอนโทรลเลอร์ควบคุมการรับส่งและตรวจสอบข้อมูลและคำสั่งที่ได้รับเพื่อสามารถติดต่อกับ คอมพิวเตอร์หลักได้ตรงตามความต้องการ ดังรูปที่ 3.10



รูปที่ 3.10 แสดง block diagram ของส่วนควบคุมย่อย

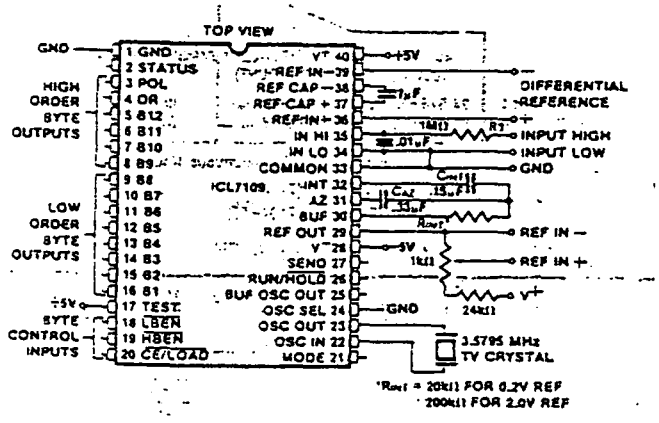
ในส่วนของไมโครคอนโทรลเลอร์นั้น ใช้ IC เบอร์ 8751 ซึ่งเป็นไมโครคอนโทรลเลอร์ของบริษัท Intel โดยชิปนี้จะมี EPROM ซึ่งเก็บโปรแกรมการทำงานของไมโครคอนโทรลเลอร์ไว้อยู่ภายในตัวแล้วทำให้สามารถนำไปใช้งานได้สะดวกมากยิ่งขึ้น

ในส่วนของ multiplex นั้นเนื่องจากมีสัญญาณจาก senserทั้งหมด 8 สัญญาณแต่ใช้ส่วนแปลงข้อมูล (ADC)เพียงตัวเดียวจึงจำเป็นที่จะต้องให้สัญญาณจาก senser เพียงตัวเดียวเข้ามาเมื่อทำการแปลงสัญญาณช่องนี้เสร็จแล้วจึงไปรับสัญญาณจาก senser ตัวถัดไปเข้ามาแปลงสัญญาณ โดยในที่นี้ใช้ IC CMOS เบอร์ 4051 ซึ่งเป็น multiplex analog switch ขนาด 8 ช่องมาใช้ และควบคุมการเลือกช่องสัญญาณ โดยใช้ไมโครคอนโทรลเลอร์มาทำการเลือก และควบคุม

ในส่วนของ dip switch มีไว้ใช้เพื่อกำหนดแอดเดรสประจำบอร์ดเพื่อสามารถที่จะเลือกว่าจะติดต่อกับบอร์ดใดได้อย่างถูกต้อง

ในส่วนของ TNC นั้นใช้ single chip modem เบอร์ MC145442 เหมือนกับวงจรของส่วน packet modem แต่ TNC นี้จะถูกควบคุมการรับส่งโดยไมโครคอนโทรลเลอร์แทน โดยให้ไมโครคอนโทรลเลอร์ควบคุม รีเลย์แทนเพื่อสามารถส่งออกอากาศได้

ในส่วนของ ADC ใช้ ADCเบอร์ ICL7109 ซึ่งเป็น ADC ขนาด 12 bit โดยICL7109 เป็น IC CMOS ซึ่งกินกำลังไฟต่ำ และสามารถติดต่อกับไมโครโปรเซสเซอร์ได้ง่าย โดยข้อมูลเอาท์พุท (12 bit,bit polarity และ bit overange) สามารถควบคุมได้โดยขาอินพุท 2 ขาและ chip select inputสำหรับการติดต่อแบบง่าย ๆ และยังมีการ handshake แบบ UART อีกแบบหนึ่ง ดังรูปที่ 3.11 แสดงการต่อวงจรทดสอบ ของ ICL7109



รูปที่ 3.11 แสดงการต่อวงจร test circuit ของ ICL7109

หน้าที่การทำงานของขาต่าง ๆ ของ ICL7109 แสดงดังตาราง 3.4

Pin	Symbol	Description		
1	GND	Digital Ground, 0V. Ground return for all digital logic.		
2	STATUS	Output High during integrate and deintegrate until data is latched. Output Low when analog section is in Auto-Zero configuration.		
3	POL	Polarity — HI for Positive input.		
4	OR	Overrange — HI if Overranged.		
5	B12	Bit 12	(Most Significant Bit)	All three state output data bits
6	B11	Bit 11		
7	B10	Bit 10		
8	B9	Bit 9		
9	B8	Bit 8		
10	B7	Bit 7	Hi = true	
11	B6	Bit 6		
12	B5	Bit 5		
13	B4	Bit 4		
14	B3	Bit 3		
15	B2	Bit 2		
16	B1	Bit 1	(Least Significant Bit)	
17	TEST	Input High — Normal Operation. Input Low — Forces all bit outputs high. Note: This input is used for test purposes only. Tie high if not used.		
18	$\overline{LBEN}$	Low Byte Enable — With Mode (Pin 21) low and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates low order byte outputs B1 — B8. — With Mode (Pin 21) high, this pin serves as a low byte flag output used in handshake mode. See Figures 8, 9, 10.		
19	$\overline{HBEN}$	High Byte Enable — With Mode (Pin 21) low and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates high order byte outputs B9 — B12, POL, OR. — With Mode (Pin 21) high, this pin serves as a high byte flag output used in handshake mode. See Figures 8, 9, 10.		
20	$\overline{CE/LOAD}$	Chip Enable Load — With Mode (Pin 21) low, $\overline{CE/LOAD}$ serves as a master output enable. When high, B1 — B12, POL, OR outputs are disabled. — With Mode (Pin 21) high, this pin serves as a load strobe used in handshake mode. See Figures 8, 9, 10.		

Pin	Symbol	Description
21	MODE	Input Low — Direct output mode where $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19) and $\overline{LBEN}$ (Pin 18) act as inputs directly controlling byte outputs. Input Pulsed High — Causes immediate entry into handshake mode and output of data as in Figure 10. Input High — Enables $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19), and $\overline{LBEN}$ (Pin 18) as outputs, handshake mode will be entered and data output as in Figures 8 and 9 at conversion completion.
22	OSC IN	Oscillator Input
23	OSC OUT	Oscillator Output
24	OSC SEL	Oscillator Select — Input high configures OSC IN, OSC OUT, BUF OSC OUT as RC oscillator — clock will be same phase and duty cycle as BUF OSC OUT. — Input low configures OSC IN, OSC OUT for crystal oscillator — clock frequency will be 1/58 of frequency at BUF OSC OUT.
25	BUF OSC OUT	Buffered Oscillator Output
26	RUN/$\overline{HOLD}$	Input High — Conversions continuously performed every 8192 clock pulses. Input Low — Conversion in progress completed, converter will stop in Auto-Zero 7 counts before integrate.
27	SEND	Input — Used in handshake mode to indicate ability of an external device to accept data. Connect to +5V if not used.
28	V^-	Analog Negative Supply — Nominally −5V with respect to GND (Pin 1).
29	REF OUT	Reference Voltage Output — Nominally 2.8V down from V^+ (Pin 40).
30	BUFFER	Buffer Amplifier Output
31	AUTO-ZERO	Auto-Zero Node — Inside foil of C_{AZ}
32	INTEGRATOR	Integrator Output — Outside foil of C_{INT}
33	COMMON	Analog Common — System is Auto-Zeroed to COMMON
34	INPUT LO	Differential Input Low Side
35	INPUT HI	Differential Input High Side
36	REF IN +	Differential Reference Input Positive
37	REF CAP +	Reference Capacitor Positive
38	REF CAP −	Reference Capacitor Negative
39	REF IN −	Differential Reference Input Negative
40	V^+	Positive Supply Voltage — Nominally +5V with respect to GND (Pin 1).

Note: All digital levels are positive true.

ตารางที่ 3.4 แสดงหน้าที่การทำงานของขาต่าง ๆของ ICL7109

ในส่วนอะนาล็อก

ICL7109 จะทำงานเมื่อป้อนไฟ +5 V และ -5 V โดยจะสามารถแปลงค่าได้สูงสุด 4V full scale ทั้งทางด้านบวกและลบ

ในการเลือกค่าอุปกรณ์เพิ่มเติมที่ต้องต่อกับ ICL7109 มีดังนี้

-Integrating Resistor สำหรับ 4.096 volt full scale ใช้ค่า 200k สำหรับ 409.6 mV full scale ใช้ค่า 20k แต่สำหรับค่า full scale อื่น ๆ หาค่า R_{INT} ได้จาก

$$R_{INT} = \frac{\text{full scale voltage}}{20\mu A}$$

-Integrating Capacitor สำหรับ ICL7109 ที่จ่ายไฟ $\pm$5 V และ analog common ต่อลง GND สำหรับการแปลงข้อมูลด้วยอัตรา 7.5 ครั้งต่อวินาที (ความถี่นาฬิกา 61.72 kHz) จะใช้ C_{INT} และ C_{AZ} เท่ากับ 0.15 μF และ 0.33 μF ตามลำดับ สำหรับความถี่นาฬิกาอื่น ๆ สามารถหาค่า C_{INT} ได้จาก

$$C_{INT} = \frac{(2048 * \text{clock period})(20\mu A)}{\text{integrator output voltage swing}} \quad \mu F$$

-Auto-Zero สามารถหาค่าได้จาก

$$C_{AZ} = 2C_{INT}$$

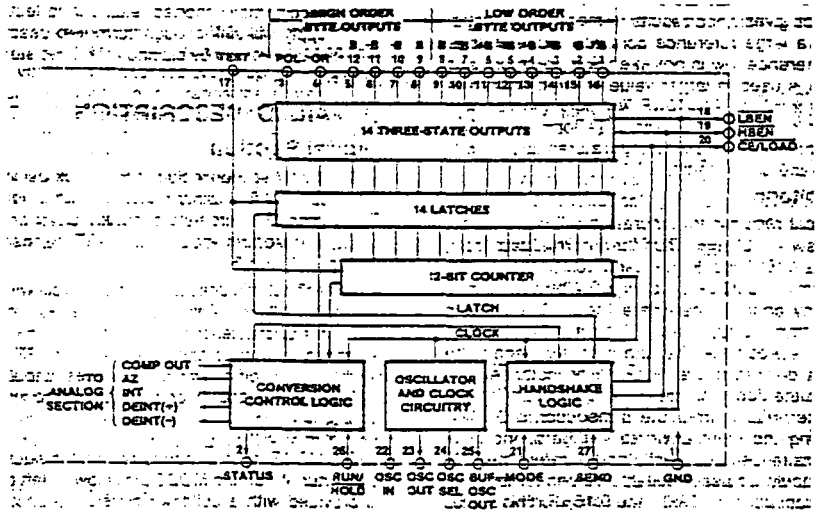
-Reference Capacitor ให้ใช้ค่าตัวเก็บประจุเท่ากับ 1 μF

-Reference Voltage ในการกำหนดค่า reference voltage นั้นจะขึ้นอยู่กับค่า input voltage full scale โดย $V_{in} = 2V_{ref}$ เช่นถ้าใช้ 4.096 V full scale จะให้ $V_{ref} = 2.048$ V

-Reference Source reference voltage นั้นจะต้องมีค่าเสถียรเพื่อที่จะให้การแปลงสัญญาณเป็นไปอย่างถูกต้องที่สุด ซึ่ง ICL7109มีขา reference output (ขา 29) เพื่อ

สร้าง reference voltage ขึ้นโดยขานี้จะรับกระแสเข้าได้สูงสุดประมาณ 20 mA โดยปกติขานี้จะให้ voltage เท่ากับ 2.8 V ซึ่งจะต้องต่อขา REF OUT เข้ากับขา REF- และ REF+ ต่อกับขากลางของ ตัวต้านทานปรับบค่าได้ โดยวงจรสำหรับ reference 204.8 mV ใช้ตามวงจรtest circuit สำหรับ reference 2.048 V นำ ตัวต้านทานคงที่ออกและใช้ ตัวต้านทานปรับค่าได้ขนาด 25k แทน

ในส่วน ดิจิตอล

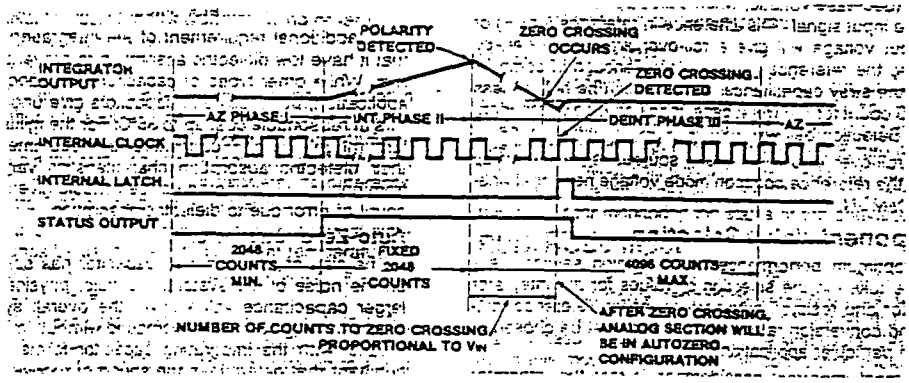


รูปที่ 3.12 แสดงส่วน digital section

ในส่วนนี้จะมีส่วน ภาคกำเนิดสัญญาณนาฬิกา (clock oscillator), สัญญาณข้อมูลต่าง ๆ และสัญญาณควบคุมต่าง ๆ ด้วย ดังนี้

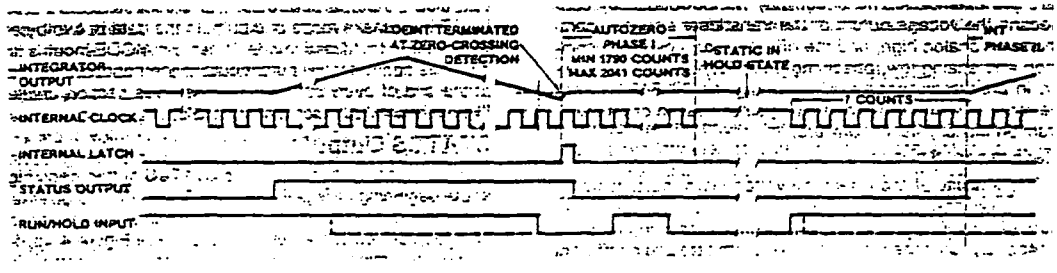
-Mode input ใช้เลือกโหมดในการแปลง เมื่อขา mode เป็น low หรือ open ตัวแปลงสัญญาณ จะเป็นโหมด direct เมื่อให้เป็น pulse high ตัวแปลงสัญญาณจะเป็นโหมด UART handshake และข้อมูลเอาท์พุทเป็นแบบ 2 ไบท์ จากนั้นจะกลับไปเป็นแบบโหมด direct เมื่อให้โหมดเป็น high ตัวแปลงสัญญาณจะให้ข้อมูลเอาท์พุทในแบบโหมด handshake ในทุก ๆ วงรอบของการแปลงเสร็จ

-STATUS output ระหว่างวงรอบของการแปลงสัญญาณ status จะเป็น high ตอนเริ่มแรกและจะตกเป็น low หลังจากข้อมูลใหม่ที่ได้จากการแปลงสัญญาณ ตามรูปที่ 3.13 เป็น timing ของ status



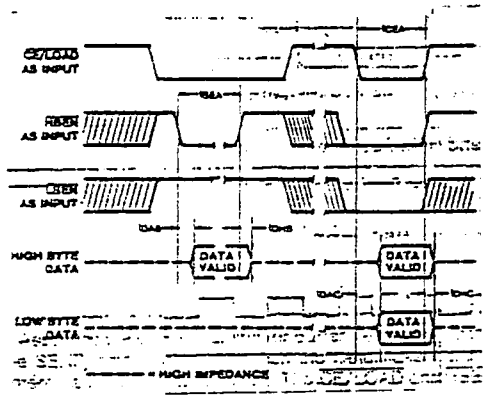
รูปที่ 3.13 เป็น timing ของ status

-RUN/HOLD input เมื่ออินพุทเป็น high หรือ open วงจรจะทำการแปลงข้อมูลอย่างต่อเนื่องถ้า RUN/HOLD เป็น low ที่เวลาใดระหว่างการ ดีอินทิเกรต วงจรจะไปทำงานแบบ auto-zero แทนคือจะไม่แปลงข้อมูลใหม่ ตามรูปที่ 3.14



รูปที่ 3.14 แสดงการทำงานของ RUN/HOLD

-Direct Mode เมื่อ MODE เป็น low จะเป็นการติดต่อกับไมโครโปรเซสเซอร์แบบโดยตรง ดังแสดงดังรูป 3.15 และตารางที่ 3.5

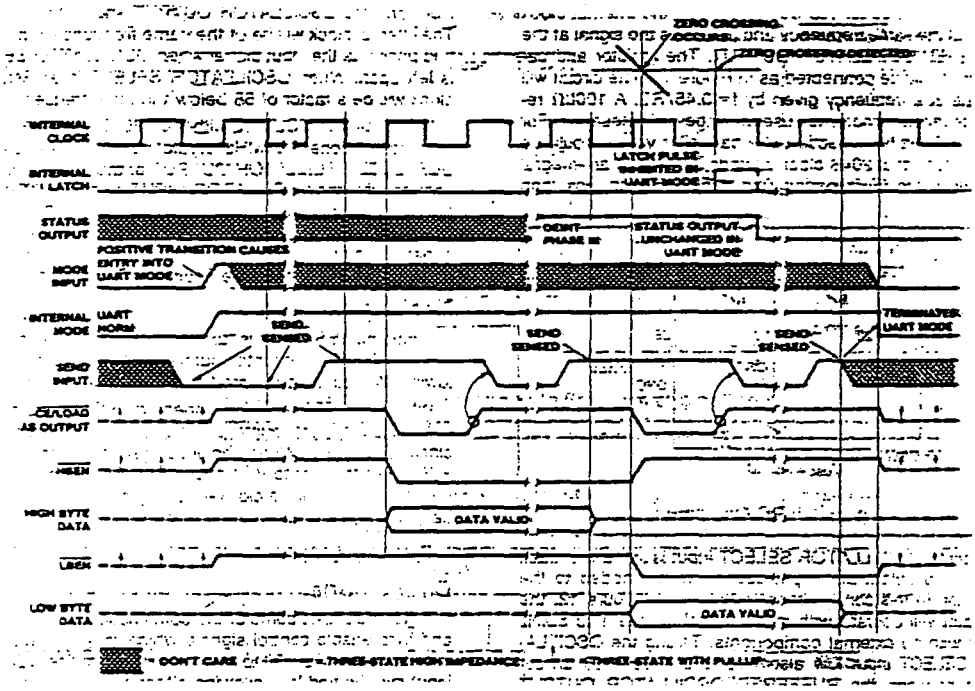


รูปที่ 3.15 แสดง timing diagram ของการติดต่อโดยตรง

SYMBOL	DESCRIPTION	MIN	TYP	MAX	UNIT
t_{BEA}	Byte Enable Width	350	220		ns
t_{DAB}	Data Access Time from Byte Enable		210	350	ns
t_{DHB}	Data Hold Time from Byte Enable		150	300	ns
t_{CEA}	Chip Enable Width	400	260		ns
t_{DAC}	Data Access Time from Chip Enable		260	400	ns
t_{DHC}	Data Hold Time from Chip Enable		240	400	ns

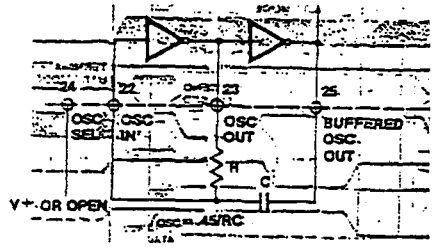
ตารางที่ 3.5 แสดงเวลาของขนาดของสัญญาณต่าง ๆ ของโหมดโดยตรง

-Handshake Mode เป็นการติดต่อกับไมโครโปรเซสเซอร์ที่จะต้องมีการทำ handshake ซึ่งมีอยู่หลายแบบด้วยกันดังแสดงดังรูป 3.16



รูปที่ 3.16 แสดง timing diagram ของการติดต่อแบบ handshake

-Oscillator ICL7109 สามารถที่จะทำวงจร oscillator ที่มีอยู่ภายในโดยต่ออุปกรณ์ภายนอกโดยเลือกได้ 2 แบบคือ ใช้ RC หรือใช้ crystal โดยเลือกใช้ oscillator select input เป็นตัวเลือก โดยเมื่อเป็น high หรือ open จะเป็นแบบ RC โดยต่อวงจรตามรูปที่ 3.17



รูปที่ 3.17 แสดงการต่อวงจร oscillator แบบใช้ RC

โดยสามารถหาค่า C ได้จาก

f = 0.45/(RC)

f = 8192/เวลาของการแปลง 1 ครั้ง

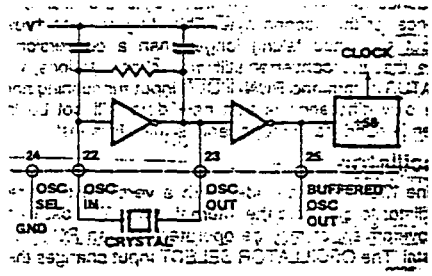
ให้ใช้ R = 100 k

เมื่อ oscillator select เป็น low จะใช้ crystal ตั้งแต่ 1-5 MHz ซึ่งความถี่ที่ crystal จะถูกหารด้วย 58 ซึ่งถ้าใช้การแปลงในอัตรา 7.5 ครั้งต่อวินาที จะใช้ crystal ความถี่ 3.579 MHz ซึ่งสามารถหาเวลาในการ integration ได้จาก

$$T_{INT} = (2048 \text{ clock periods})*(T_{clock})$$

โดย $T_{clock} = 58/3.579\ MH_z$

โดยต่อวงจร oscillator ที่ใช้ crystal ตามรูป 3.18

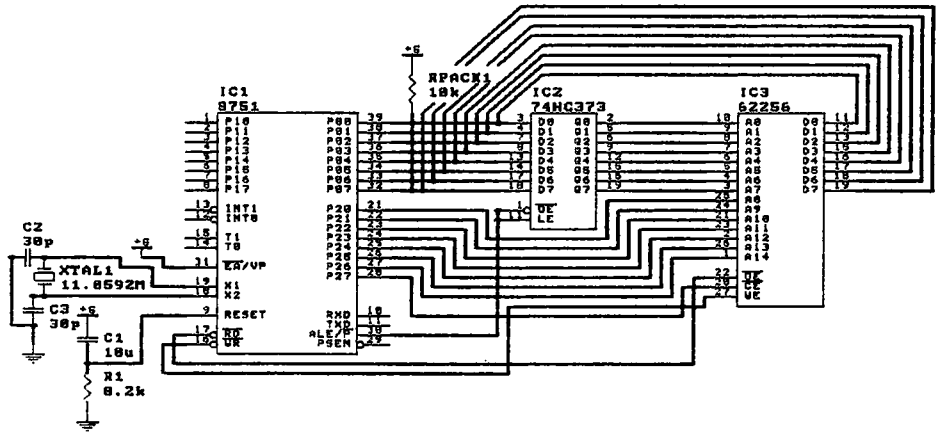


รูปที่ 3.18 แสดงการต่อวงจร oscillator ที่ใช้ crystal

-Test Input เมื่อ test input มีระดับแรงดันเป็นครึ่งหนึ่งของ V^+ เทียบ GND ทำให้ counter output latch จะทำงานซึ่งสามารถทดสอบได้ทุกเวลาเมื่อ test ต่อกับ GND จะทำให้ counter output เป็น high state และ internal clock จะไม่ทำงาน และเมื่ออินพุทกลับไปเป็นครึ่งหนึ่งของ V^+ เทียบ GND และให้ clock เข้าไป 1 ลูก counter output จะกลับเป็น low ทำให้ตรวจสอบได้ง่าย

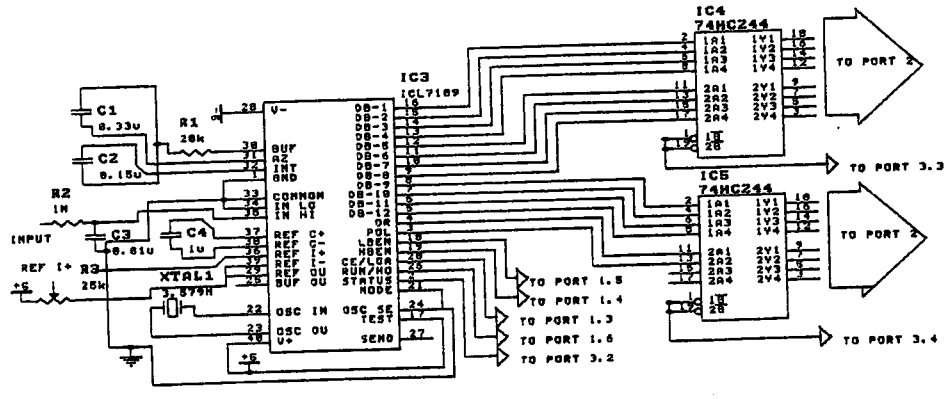
การออกแบบและการสร้างส่วนควบคุมย่อย

-การต่อวงจรไมโครคอนโทรลเลอร์ กับหน่วยความจำแรม โดยในส่วนของหน่วยความจำนั้นมีหน้าที่ในการเก็บข้อมูลที่อ่านได้จาก senser โดยผ่านทางไมโครคอนโทรลเลอร์และในส่วนของไมโครคอนโทรลเลอร์นั้น จะต้องผ่านการโปรแกรมที่ได้ออกแบบไว้เรียบร้อยแล้ว โดยต่อวงจรตามรูปที่ 3.19



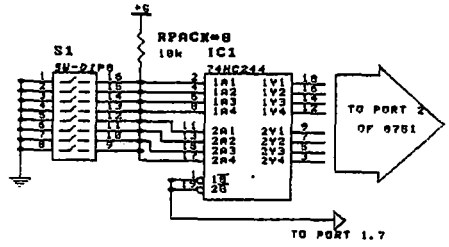
รูปที่ 3.19 แสดงการต่อไมโครคอนโทรลเลอร์เบอร์ 8751 กับ แรม

-การต่อวงจรแปลงสัญญาณอะนาลอกเป็นสัญญาณดิจิตอล ที่ใช้เบอร์ ICL7109 โดยต่อวงจรตามรูปที่ 3.20



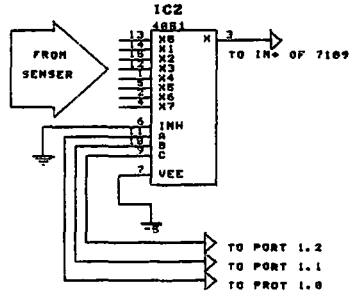
รูปที่ 3.20 แสดงการต่อ ICL7109

-การต่อวงจรในส่วนของ dip switch นั้นจะกั้นด้วย buffer เบอร์ 74HCT244 เพื่อสามารถจะใช้ พอร์ทเดียวกันนี้ได้กับอุปกรณ์อื่นอีก โดยต่อวงจรตามรูป 3.21



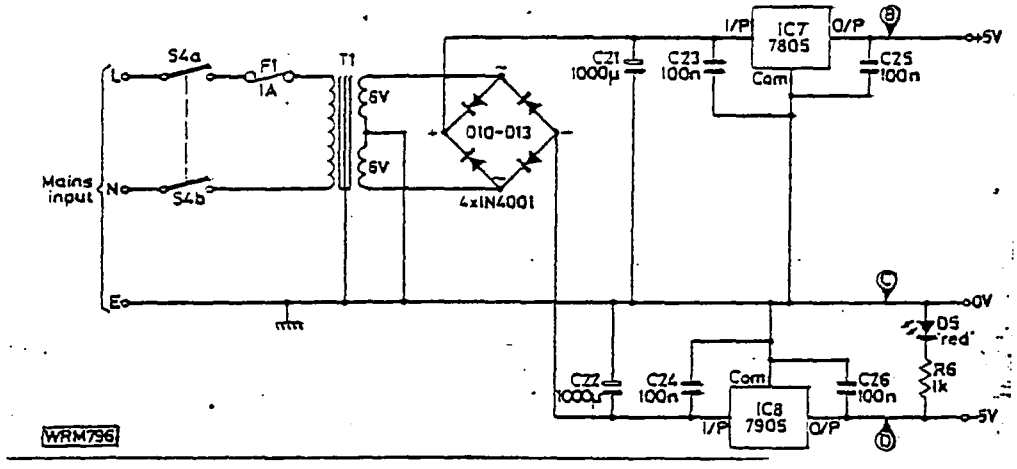
รูปที่ 3.21 แสดงการต่อกับ dip switch

-การต่อวงจรในส่วนของ multiplex โดยใช้ multiplex analog switch เบอร์ 4051 เพื่อสามารถต่อกับ senser ได้ทั้ง 8 ชุด เพื่อใช้ร่วมกับ ADC เพียงตัวเดียว โดยต่อวงจรตามรูปที่ 3.22



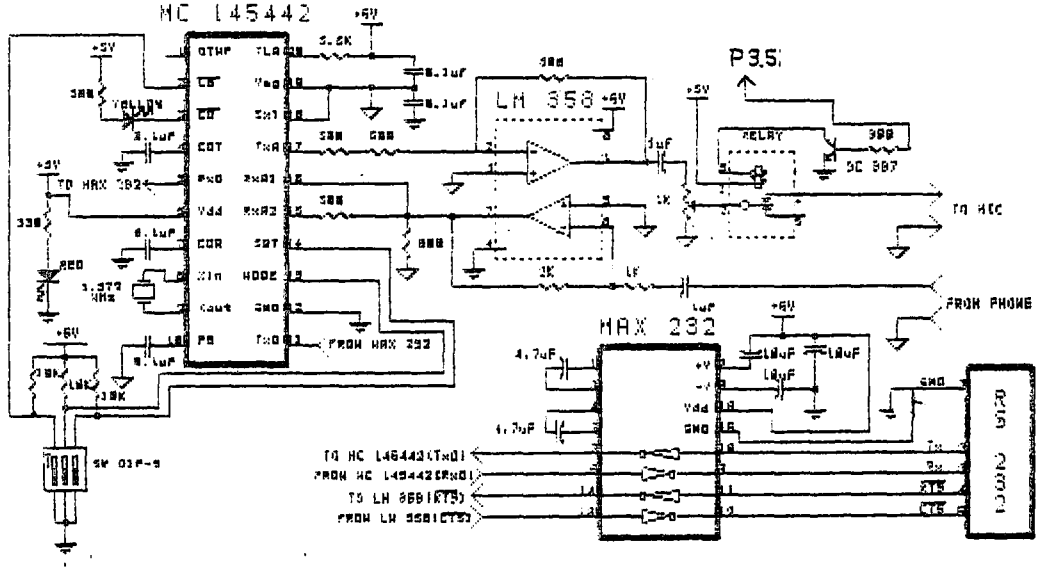
รูปที่ 3.22 วงจรส่วน multiplex analog switch เบอร์ 4051

-ในส่วนของ supply ทำ supply ±5 V โดยใช้ IC regulate เบอร์ 7805,7905 ดังแสดงดังรูปที่ 3.23



รูปที่ 3.23 แสดงส่วน supply

-ในส่วนของวงจร terminal node controller โดยใช้ single chip modem เบอร์ MC145442 ซึ่งคล้ายกับในส่วนของ packet modem เพียงแต่ใช้ไมโครคอนโทรลเลอร์ควบคุมรีเลย์แทนและไม่มีส่วนส่งสัญญาณ CTS เท่านั้น โดยต่อวงจรตามรูปที่ 3.24



รูปที่ 3.24 แสดงวงจรส่วน TNC

หลักการทำงานของโปรแกรม ในส่วนของส่วนควบคุมย่อย

ในการติดต่อสื่อสาร โดยทางคลื่นวิทยุที่ได้ออกแบบนี้มีอัตราในการส่งเท่ากับ 300 baud จึงจำเป็นที่จะให้ตัวไมโครคอนโทรลเลอร์ รับส่งด้วยอัตราเดียวกันโดยเมื่อผู้ใช้ทำการติดต่อจะต้องมีการส่งข้อมูลตามลำดับดังนี้คือ

-การตรวจสอบ password จะต้องมีการส่ง password " ! " ออกมาก่อนเพื่อกันสัญญาณที่ไม่ต้องการมารบกวนระบบได้

-การตรวจสอบ แอดเดรส ของตัวรับซึ่งจะต้องมีการส่งแอดเดรสมาและไมโครคอนโทรลเลอร์จะต้องตรวจสอบว่าใช่ แอดเดรสประจำตัวหรือไม่ โดยสามารถอ้างได้ตั้งแต่ 0-255 ถ้าไม่ใช่ก็จะไม่รับข้อมูลถัดไป โดยจะตรวจสอบจาก dip switch แต่ถ้าใช่ก็จะคอยรับข้อมูลถัดจากแอดเดรส

-การตรวจสอบคำสั่ง หลังจากผู้ใช้ส่งแอดเดรสมาแล้วจะส่งคำสั่งมาดังนี้

R = รีเซทระบบ

S = ส่งข้อมูลทั้งหมดที่เก็บอยู่ในแรม

D = รับเวลาที่จะให้ A TO D แปลงสัญญาณห่างกันต่อครั้งเป็นเวลาเท่าใดโดยข้อมูลที่ตามมาจะเป็นเวลาซึ่งสามารถอ้างได้ตั้งแต่ 0-99,999 วินาที

O = ส่งข้อมูลเพียงช่องเดียวในแต่ละชุดของข้อมูลที่เก็บไว้ในแรม

B = ส่งข้อมูลในชุดใดถึงชุดใด โดยข้อมูลที่ตามมาจะเป็น ชุดเริ่มต้น แล้วตามด้วย "-" และชุดสุดท้าย โดยมีจำนวนชุดทั้งหมดตั้งแต่ 0 ถึง 1000 ชุด

C = ส่งข้อมูลเพียงช่องเดียวในชุดใดถึงชุดใด โดยข้อมูลที่ตามมาจะเป็น ช่องที่ต้องการจะให้ส่ง ตามด้วย "," ตามด้วย ชุดเริ่มต้น ตามด้วย "-" และชุดสุดท้าย

T = ส่งข้อมูลเพียงชุดเดียว(ทั้ง 8 ช่อง)ในเวลาขณะนั้น

-การตรวจสอบโค๊ด enter จะเป็นตัวสุดท้ายในทุก ๆ คำสั่งเมื่อตรวจสอบพบโค๊ดนี้ แล้วจะทำตามคำสั่งนั้นทันที ถ้าเป็นคำสั่ง R , D จะส่ง แอดเดรสและตัว ACK กลับมา

-ถ้าข้อมูลที่เก็บไว้ใน แรมเก็บครบ 1000 ชุดแล้วตัวควบคุมย่อยจะส่งแอดเดรสของตัวควบคุมย่อยนั้นและตัว F

-ถ้ามีการติดต่อโดยให้ส่วนควบคุมย่อยส่งข้อมูลในขณะที่ยังไม่มีข้อมูลใน แรม ตัวควบคุมย่อยจะส่งแอดเดรสและตัว E เพื่อบอกว่า ไม่มีข้อมูลในหน่วยความจำ

ในการติดต่อกันทางคลื่นวิทยุ อาจมีคลื่นรบกวนหรือมีคนใช้ความถี่นั้นอยู่ทำให้คำสั่งสูญหายไป บางส่วนทำให้ส่วนควบคุมย่อยไม่เข้าใจคำสั่งที่ส่ง ส่วนควบคุมย่อยก็จะส่ง "?" กลับมาเพื่อบอกว่าข้อมูลสูญหาย

บทที่ 4

การทดลอง

ในการทดลอง จะแบ่งการทดลองออกเป็น 2 ตอนด้วยกัน คือ

4.1 การทดลองการติดต่อสื่อสารระหว่างไมโครคอมพิวเตอร์ 2 ชุด ซึ่งใช้ packet modem ที่สร้างขึ้น มาเป็นอุปกรณ์ช่วยในการติดต่อสื่อสาร โดยส่งผ่านทางคลื่นวิทยุสมัครเล่น

4.2 การทดลองการติดต่อสื่อสารระหว่างส่วนควบคุมย่อย (ที่สร้างขึ้น) กับส่วนของไมโครคอมพิวเตอร์ โดยส่งผ่านทางคลื่นวิทยุสมัครเล่น

ซึ่งแต่ละการทดลองมีรายละเอียดดังต่อไปนี้

4.1 การทดลองการติดต่อสื่อสารระหว่างไมโครคอมพิวเตอร์ 2 ชุด

อุปกรณ์ที่ใช้ในการทดลองมีดังต่อไปนี้

1. ไมโครคอมพิวเตอร์	2 ชุด
2. packet modem	2 ชุด
3. วิทยุรับ/ส่ง	2 เครื่อง
4. โปรแกรมการสื่อสารข้อมูล	

เนื่องจากในการเชื่อมต่อระหว่างไมโครคอมพิวเตอร์ และ packet modem ที่สร้างขึ้น จำเป็นต้องใช้ขาสัญญาณ RTS (Request to Send) และขาสัญญาณ CTS (Clear to Send) ของ RS-232C ในการควบคุมตัวรีเลย์ภายใน packet modem ซึ่งทำหน้าที่ปิด/เปิดสัญญาณในการออกอากาศผ่านทางวิทยุรับ/ส่ง จากการทดลองใช้โปรแกมการสื่อสารข้อมูลสำเร็จรูปทั่วไป เช่น PROCOMM นั้น จะพบว่า ไม่สามารถใช้ขาสัญญาณทั้งสองในการควบคุมได้ ดังนั้นจึงจำเป็นต้องเขียนโปรแกรมการสื่อสารข้อมูลขึ้นมาใหม่ ซึ่งสามารถควบคุมลักษณะสัญญาณที่ขาสัญญาณทั้งสองได้ โดยภาษาที่ใช้ในการเขียนโปรแกรมคือ ภาษาปาสคาล

วิธีการทดลอง

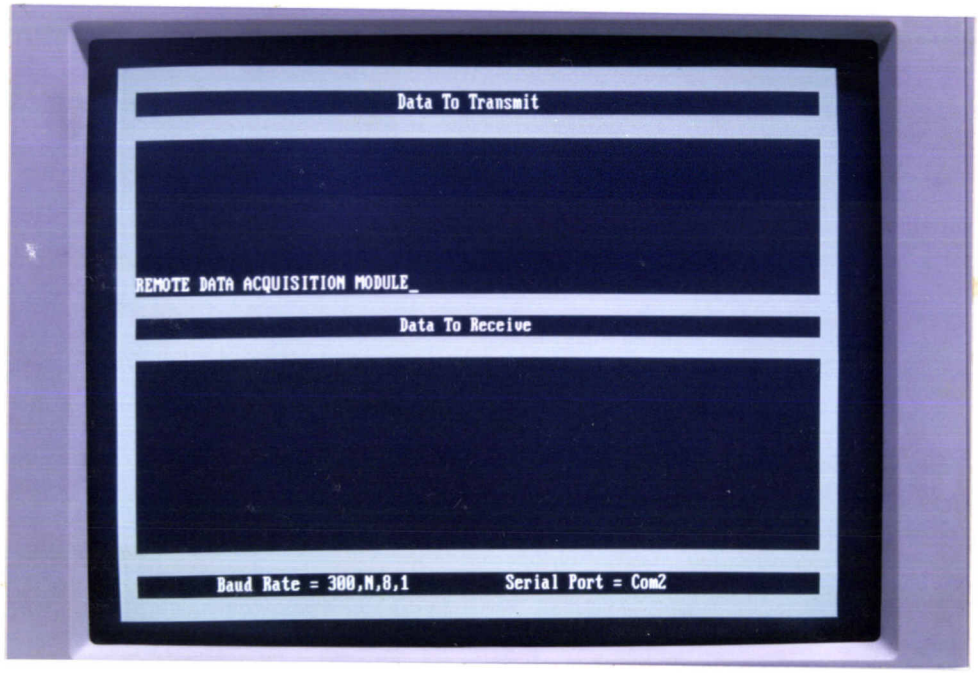
ให้ต่ออุปกรณ์ทั้งสองฝ่าย(รับ/ส่งข้อมูลกัน) ดังรูปข้างล่างนี้

รูปที่ 4.1 แสดงการต่ออุปกรณ์ในการรับส่งข้อมูล

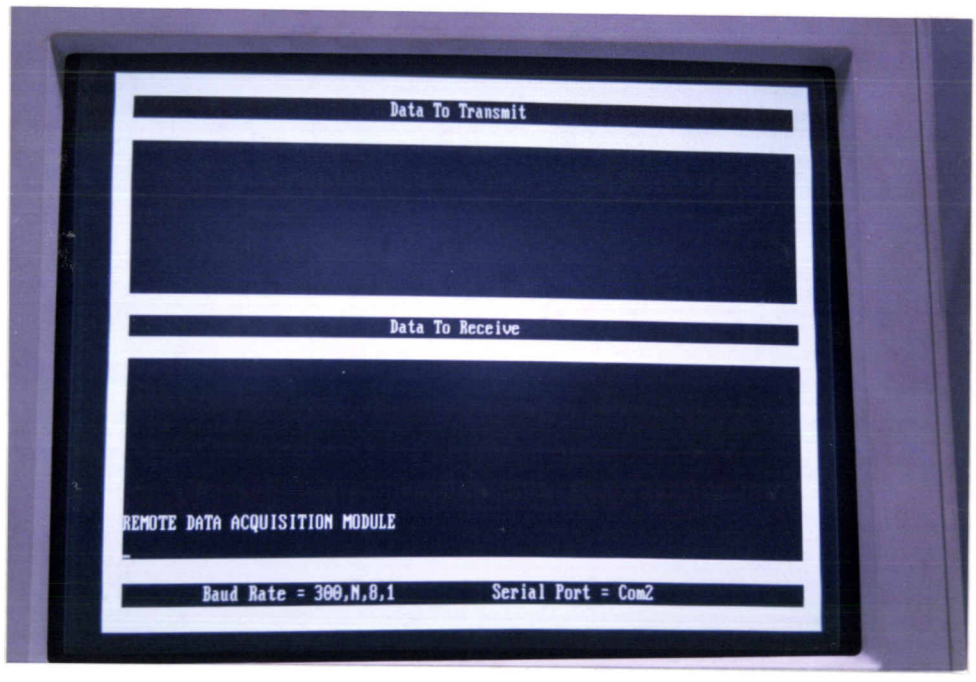
จากนั้นใช้โปรแกรมการสื่อสารข้อมูลที่เขียนขึ้นมาใหม่ โดยโหลดโปรแกรมให้กับไมโครคอมพิวเตอร์ทั้งสองชุด ซึ่งจะต้องตั้งค่าพารามิเตอร์ในการรับส่งข้อมูลให้ตรงกัน ในที่นี้ใช้อัตราการรับส่งข้อมูลที่ 300 บิทต่อวินาที เนื่องจากชิฟโมเด็ม(MC 145442) ที่ใช้ใน packet modem มีอัตราเร็วในการรับส่งข้อมูล 300 บิทต่อวินาที การรับส่งเป็นแบบ Half Duplex

4.1.1 การรับส่งข้อความ

เลือกโหมดของโปรแกรมสื่อสาร ทั้ง 2 ฝ่าย ให้อยู่ในโหมดของการรับส่งข้อความ (Trans/Recei) จากนั้น ผลัดกันรับส่งข้อความ ซึ่งลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่ส่งและทางฝ่ายที่รับ จะเป็นดังรูปที่ 4.2 และ รูปที่ 4.3 ตามลำดับ



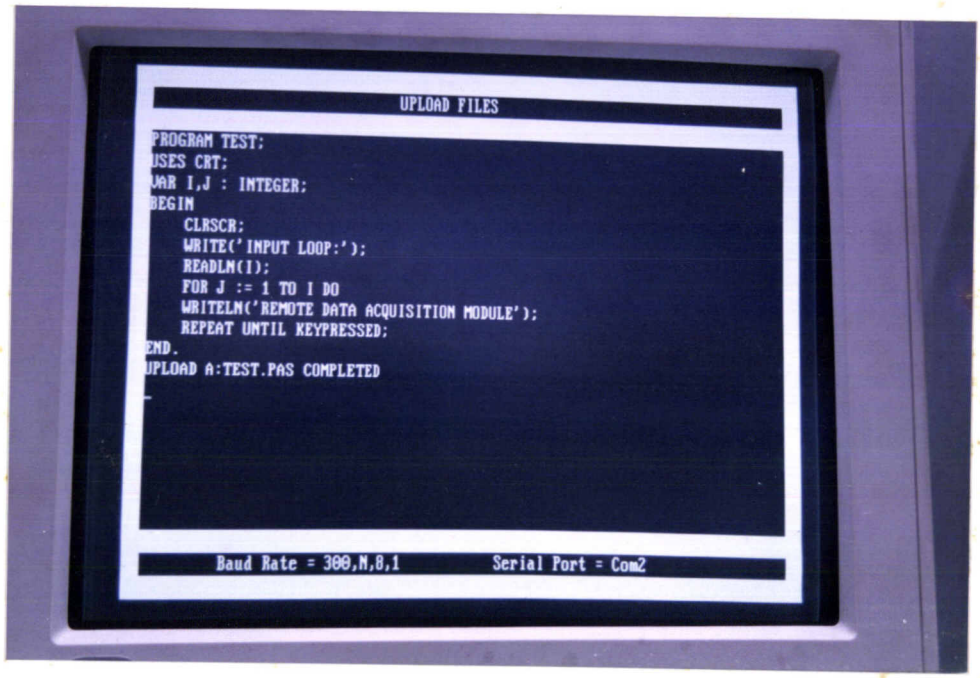
รูปที่ 4.2 แสดงลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่ส่งข้อความ



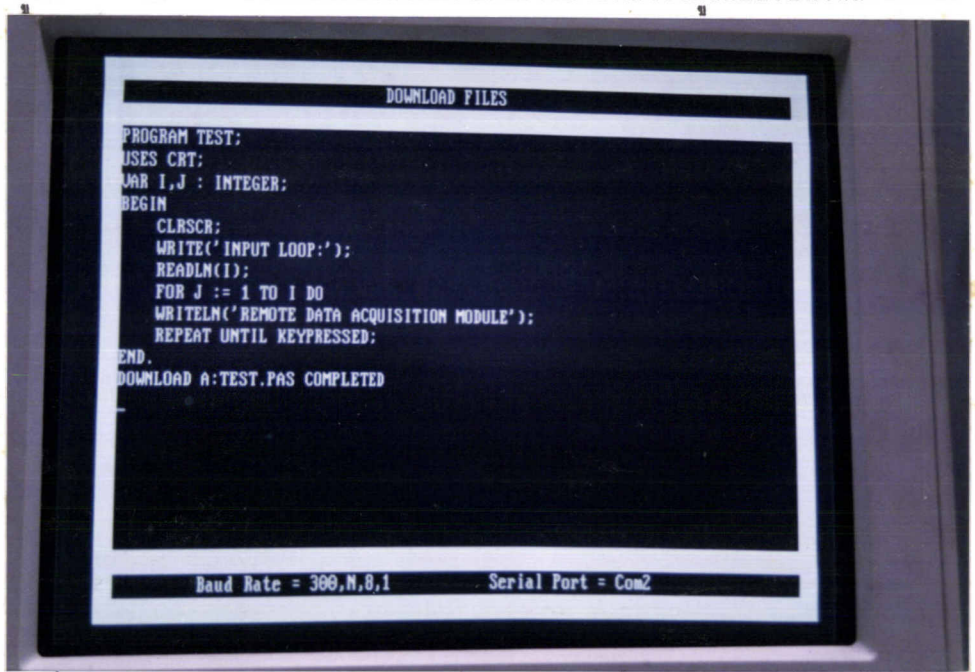
รูปที่ 4.3 แสดงลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่รับข้อความ

4.1.2 การรับส่งแบบเป็นไฟล์ (Text)

เลือกโหมดของโปรแกรมสื่อสารของฝ่ายที่จะส่งไฟล์ ให้อยู่ในโหมดการส่งไฟล์(Upload File) ส่วนทางฝ่ายที่จะรับ ให้อยู่ในโหมดการรับไฟล์(Download File) จากนั้นก็ทำการรับส่งไฟล์กัน ซึ่งลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่ส่งไฟล์ และทางฝ่ายที่รับไฟล์ จะเป็นดังรูปที่ 4.4 และ รูปที่ 4.5 ตามลำดับ



รูปที่ 4.4 แสดงลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่ส่งข้อมูลแบบเป็นไฟล์(Text)



รูปที่ 4.5 แสดงลักษณะบนจอมอนิเตอร์ของทางฝ่ายที่รับข้อมูลแบบเป็นไฟล์(Text)

4.2 การทดลองการติดต่อสื่อสารระหว่างส่วนควบคุมย่อยกับไมโครคอมพิวเตอร์

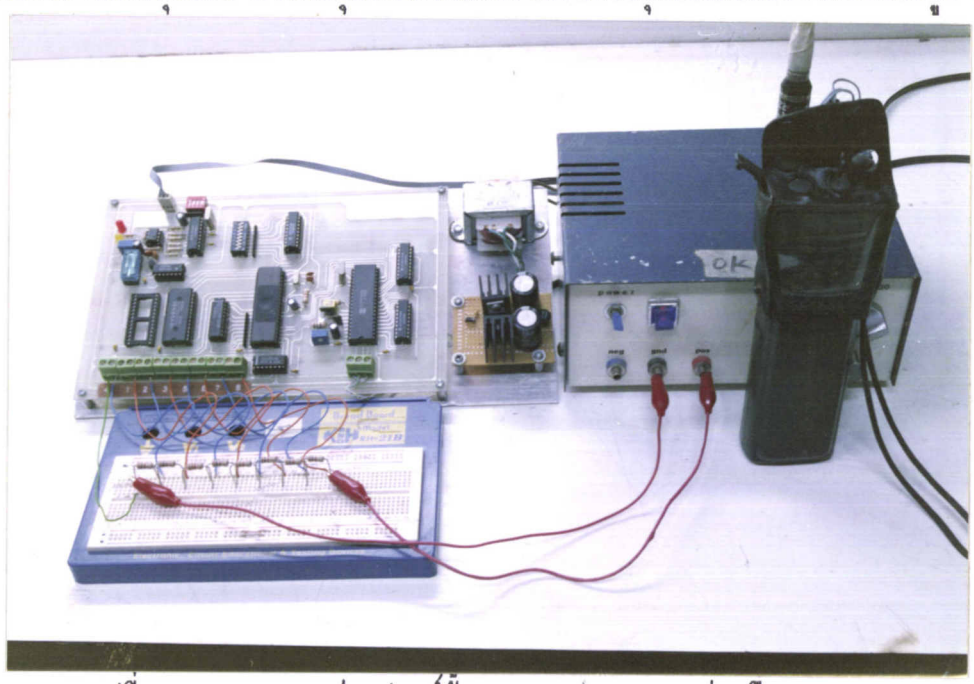
อุปกรณ์ที่ใช้ในการทดลองมีดังต่อไปนี้

1. ส่วนควบคุมย่อย 1 ชุด
2. ไมโครคอมพิวเตอร์ 1 ชุด
3. วิทยุรับ/ส่ง 2 เครื่อง
4. โปรแกรมการสื่อสารข้อมูล

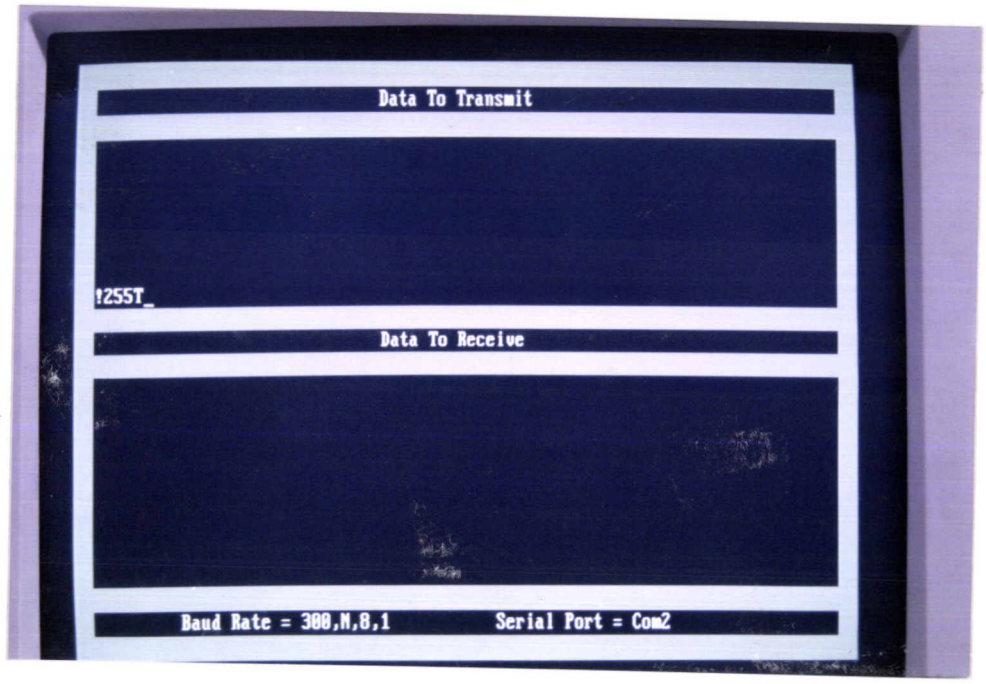
วิธีการทดลอง

ในส่วนของไมโครคอมพิวเตอร์ยังคงต่อตามรูปที่ 4.1 โดยเลือกอัตราเร็วในการรับส่งเป็น 300 บิทต่อวินาทีเช่นเดิม และเลือกโหมดของโปรแกรมให้อยู่ในโหมดรับส่งข้อความ (Trans/Recei)

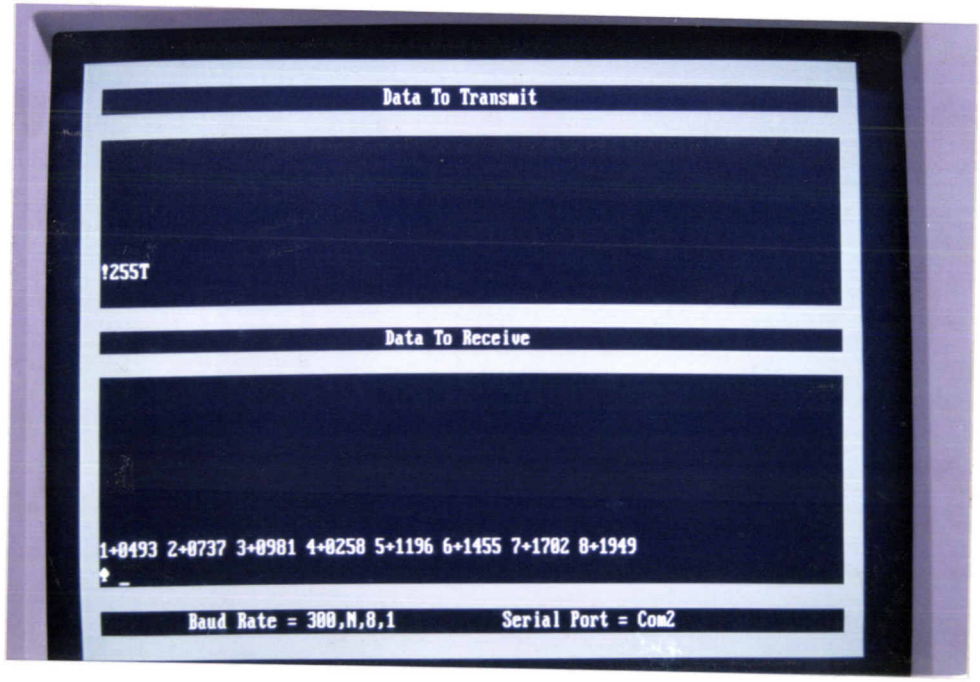
ส่วนควบคุมย่อยนั้น จะมีแชนแนลอยู่ 8 แชนแนล เอาไว้ต่อกับตัวเซ็นเซอร์ภายนอก ในที่นี้เราจะใช้ส่วนควบคุมย่อยเป็นตัวแทนของโวลต์มิเตอร์ ในการบอกค่าความต่างศักย์ไฟฟ้าของวงจรแบ่งแรงดัน (โดยแรงดันสูงสุดไม่เกิน 4 โวลต์) ซึ่งในวงจรที่ทดลองนี้ จะใช้แรงดันสูงสุดมีค่าเท่ากับ 2 โวลต์ โดยจะแบ่งแรงดันออกเป็น 8 ช่วง ซึ่งแต่ละช่วงจะต่อเข้ากับแชนแนลที่อยู่บนบอร์ดของส่วนควบคุมย่อย การต่ออุปกรณ์ทั้งหมดของส่วนควบคุมย่อยที่ทดลองจะเป็นดังรูป 4.6



รูปที่ 4.6 แสดงการต่ออุปกรณ์ทั้งหมดของส่วนควบคุมย่อยที่ทดลอง



รูปที่ 4.7 แสดงลักษณะบนจอมอนิเตอร์ที่เกิดขึ้น ในการส่งคำสั่งของไมโครคอมพิวเตอร์ให้กับส่วนควบคุมย่อย



รูปที่ 4.8 แสดงลักษณะบนจอมอนิเตอร์ที่เกิดขึ้น หลังจากส่วนควบคุมได้ส่งข้อมูลกลับมายังส่วนของไมโครคอมพิวเตอร์

บทที่ 5

สรุปและข้อเสนอแนะ

อุปกรณ์ packet modem (terminal node controller) คืออุปกรณ์ที่ใช้สำหรับการติดต่อสื่อสารข้อมูลดิจิตอลผ่านทางสัญญาณวิทยุ ซึ่งให้ข้อดีกว่าการติดต่อสื่อสารในระบบดิจิตอลผ่านสายโทรศัพท์ ตรงที่สามารถติดต่อกับอุปกรณ์อีกตัวหนึ่งที่บริเวณใดก็ได้ แต่อาจมีปัญหาจากสัญญาณรบกวนจากภายนอก ทำให้ข้อมูลเกิดการสูญหาย ซึ่งเป็นข้อเสียส่วนหนึ่งของระบบนี้

สำหรับอุปกรณ์ packet modem ที่ได้สร้างขึ้น เพื่อใช้ในการติดต่อระหว่างคอมพิวเตอร์ 2 ตัว นั้นได้ผลเป็นที่น่าพอใจ คือ สามารถส่งข้อความในการติดต่อได้ค่อนข้างถูกต้อง แต่อาจจะมีปัญหาเกี่ยวกับข้อมูลในส่วนแรก และส่วนหลัง ซึ่งเป็นเวลาที่หน้าสัมผัสของรีเลย์ทำงานนั้น จะมีการเกิด bounce ขึ้น ทำให้ส่วนตัวรับแปลงข้อมูล ทั้ง ๆ ที่ยังไม่ได้ส่งข้อมูล จึงแก้โดยการตัดข้อมูลในส่วนแรก และส่วนสุดท้าย ของตัวรับที่รับได้ออก

ในส่วนของตัวควบคุมย่อยนั้น ค่อนข้างที่จะทำงานได้ดี เพราะใช้ไมโครคอนโทรลเลอร์ในการงาน และจะรับข้อมูลเพียงไม่กี่ตัวเท่านั้นในการตีความ (decode) ทำให้ไม่ค่อยมีปัญหาเหมือนกับในส่วนของ packet modem แต่ในการติดต่อกันนี้จะใช้ตัวอักษรเพียงไม่กี่ตัว จึงทำการเลือกตัวอักษรที่ต้องการมาแสดงบนจอมอนิเตอร์ของคอมพิวเตอร์เท่านั้น

ปัญหา

ปัญหาที่เกิดขึ้นในการทดลองนี้ คือ เมื่อมีสัญญาณจากภายนอกที่ไม่พึงประสงค์เข้ามา จะทำให้ข้อมูลบางส่วนสูญหาย หรือรับข้อมูลใด ๆ ที่ไม่ได้เกิดจากคู่กรณีที่ทำการติดต่ออยู่ ทำให้เกิดตัวอักษรที่ไม่ต้องการ และไม่มีความหมายขึ้นมา และในการวางอุปกรณ์ packet modem นั้น ไม่สามารถวางไว้ใกล้กับคอมพิวเตอร์ได้ เพราะคอมพิวเตอร์จะให้กำเนิดสัญญาณรบกวนออกมา ทำให้ไม่สามารถรับสัญญาณข้อมูลขนาดอ่อน ๆ ได้

ข้อเสนอแนะ

การใช้อุปกรณ์ในการสื่อสารข้อมูลผ่านทางสัญญาณวิทยุนั้น ควรเลือกช่องสัญญาณที่ไม่มีการติดต่อกัน (เป็นช่องสัญญาณที่ว่าง) เพื่อหลีกเลี่ยงกับการที่ข้อมูลจะเสียหาย และสัญญาณรบกวนจากภายนอก และไม่ควรนำอุปกรณ์ในการสื่อสารข้อมูล ไปไว้ใกล้กับอุปกรณ์ที่กำเนิดสัญญาณคลื่นวิทยุ ซึ่งจะมารบกวนระบบได้ โดยทางคณะผู้จัดทำโครงงานพิเศษนี้ ก็หวังว่าคงจะมีประโยชน์และสามารถนำไปปรับปรุง พัฒนา ประยุกต์ใช้กับอุปกรณ์สื่อสารอื่น ๆ ต่อไป

บรรณานุกรม

1. ชูชัย ธนสารตั้งเจริญ และ ทินกร ดุ๊ก, " การสื่อสารข้อมูล " , หจก. สำนักพิมพ์ ฟิสิกส์เซ็นเตอร์ , กรุงเทพ
2. วิสันติ์ อาชาเดโชพล, " ระบบโทรศัพท์ดิจิตอล " , หจก. สำนักพิมพ์ ฟิสิกส์เซ็นเตอร์, กรุงเทพ
3. เอกสิทธิ์ ฐิติวรการ, " เทคโนโลยีการสื่อสารข้อมูล ตอน มาตรฐานตระกูล RS " , กรุงเทพ
4. ชาญยุทธ พฤกษ์คุ้มวงศ์, " เรียนรู้..การสื่อสารข้อมูล " ,เซมิคอนดัคเตอร์ อิเลคทรอนิคส์ 130(2536) : 95-97

ภาคผนวก

```
            ;PROGRAM SUBMICROCONTROLLER
            ORG 0000H
            LJMP MAIN              ;GO TO MAIN
    ;---------------------------------------;
            ;SUBROUTINE EXTERNAL 0 INTERUPT
            ORG 0003H
            LJMP R_7109            ;GO TO ROUTINE READ DATA FROM 7109
    ;---------------------------------------;
            ;SUBROUTINE TIMER 0 INTERUPT
            ORG 000BH
            LJMP TIM0_0            ;GO TO ROUTINE TIMER 0
    ;---------------------------------------;
            ;SUBROUTINE SERIAL INTERUPT
            ;IN SBUF,(00,01,02,03,04,05,06,0BH,TI,RI (BIT))
            ;OUT 01H (BIT)
            ORG 0023H
SERIAL:     JB TI,TRANS            ;CHECK TI
            JNB 01H,NORMAL         ;CHECK BIT 01H FOR CONDITION OK
            PUSH 0E0H
            MOV A,SBUF
            CJNE A,#0DH,NOT_STOP ;CHECK KEY ENTER
            CLR 01H                ;STOP TO RECEIVE
NOT_STOP:   POP 0E0H
            CLR SCON.0
            RETI
NORMAL:     JB 1FH,DAT_F
            JB 00H,NORMAL1
            AJMP PASSWORD
NORMAL1:    JB 02H,INST_1          ;INSTRUCTION 1
            JB 03H,INST_2          ;INSTRUCTION 2
```

```
          JB 04H,INST_3        ;INSTRUCTION 3
          JB 05H,INST_4        ;INSTRUCTION 4
          JB 06H,INST_5        ;INSTRUCTION 5
          JB 0BH,INST_6        ;INSTRUCTION 6
          JB 22H,INST_7        ;INSTRUCTION 7
          LJMP DATA
    ;-------------------------------------;
             ;SUBROUTINE INSTRUCTION 1 (SEND ALL DATA IN RAM BACK)
             ;IN 02H (BIT)
             ;OUT P3.5,02H,TR0 (BIT),T0
INST_1:   LCALL CL_DATA
               PUSH 0E0H
               MOV A,SBUF
               CJNE A,#0DH,INST_1_1
               CLR SCON.4              ;DISABLE REN
INST_1_2: MOV TH0,#3CH
          MOV TL0,#0AFH
          SETB P3.5            ;CONTROL RELAY
          SETB TR0             ;RUN TIMER 0
          CLR 00H
               POP 0E0H
          CLR SCON.0
          CLR 02H
          RETI
INST_1_1:      SETB 13H
               SJMP INST_1_2
    ;-------------------------------------;
INST_2:   AJMP INST_2_1
INST_3:   AJMP INST_3_1
INST_4:   AJMP INST_4_1
```

```
INST_5:   AJMP INST_5_1
INST_6:   AJMP INST_6_1
INST_7:   AJMP INST_7_1
DAT_F:    AJMP DATA_FULL
    ;-------------------------------------;
           ;SUBROUTINE TRANSMIT DATA BACK
           ;IN (05,06,0B,0E,10,17,18)
           ;OUT (05,06,0B,0E,10,17,18)
TRANS:    JB 0EH,TRANS_1          ;SEND ACK
             jb 25h,trans_1
          JB 1FH,TRANS_1          ;FOR TELL DATA FULL
          JB 17H,TRANS_2          ;FOR SERVICE INTERUUPT LAST DATA
          JB 05H,TRANS_3
          JB 06H,TRANS_4
          JB 0BH,TRANS_5
          AJMP TRAN_ALL
TRANS_1:  JB 1BH,TRAN_1_1
          SETB 1BH
          CLR SCON.1
          MOV SBUF,32H            ;SEND MID ORDER OF CPU
          RETI
TRAN_1_1: JB 1CH,TRAN_1_2         ;SEND LOW ORDER OF CPU
          SETB 1CH
          CLR SCON.1
          MOV SBUF,31H
          RETI
TRAN_1_2: CLR SCON.1
             jb 25h,tran_1_5
          JB 1FH,TRAN_1_3
          MOV SBUF,#06H           ;ACK
```

```
TRAN_1_4: CLR 1BH
          CLR 1CH
          CLR 0EH
          SETB 17H
          RETI
TRANS_2:  CLR SCON.1
               SETB SCON.4              ;ENABLE RECEPTOR
          CLR P3.5
          CLR 17H
          RETI
TRAN_1_3: MOV SBUF,#'F'
          CLR 1FH
          SJMP TRAN_1_4
TRAN_1_5:      mov sbuf,#'E'
               CLR 25H
               sjmp tran_1_4
TRANS_3:  AJMP TRAN_ONE          ;SEND ONE CHANNAL
TRANS_4:  AJMP TRAN_AMT          ;SEND AMOUNT
TRANS_5:  AJMP TRAN_OMT          ;SEND ONE AND AMOUNT
    ;-------------------------------------------------------;
          ;SUBROUTINE SEND DATA FULL
          ;IN (1F),SBUF
          ;OUT (1F)
DATA_FULL:PUSH 0E0H
          MOV A,SBUF
          CJNE A,#06H,DAT_F_1
          MOV TH0,#3CH
          MOV TL0,#0AFH
          SETB P3.5
          MOV R2,#25D
```

```
          SETB TR0
DAT_F_1:  CLR SCON.0
          POP 0E0H
          RETI
    ;-------------------------------------------------------;
             ;SUBROUTINE SEND FORM REAL TIME
             ;IN (22)
             ;OUT
INST_7_1: LCALL CL_DATA
               PUSH 0E0H
               MOV A,SBUF
               CJNE A,#0DH,INST_7_2
               CLR SCON.4
          CLR 22H
          SETB 30H
          CLR 00H
          MOV C,IE.0
          MOV 2BH,C              ;SAVE IE.0
          MOV 4AH,P1
          MOV P1,#0F0H
          SETB IE.0
INST_7_3:      POP 0E0H
          CLR SCON.0
          RETI
INST_7_2:      LCALL CR_DATA
               CLR 00H
               CLR 22H
               SJMP INST_7_3
    ;-------------------------------------------------------;
             ;SUBROUTINE SEND DATA ALL IN THE MEMORY
```

```
            ;IN (20,TI,OD),34,35,3B,3C
            ;OUT SBUF,(20,18),3B,3C
TRAN_ALL: JB 18H,TRAN_ALL2        ;SEND DATA FOR READY 8 TIMES
TRAN_ALL3:JB 28H,TRAN_ALL5
          PUSH OEOH
          MOV DPH,3BH             ;CHECK LAST ADDRESS OF DATA
          MOV DPL,3CH
          MOV A,DPH
          CJNE A,34H,TRAN_ALL1
          MOV A,DPL
          CJNE A,35H,TRAN_ALL1
TRAN_ALL8:  ;  CLR P3.5                ;RELAY OFF
          CLR SCON.1
                mov 1fH,#20h
                mov sbuf,#0Ah
                setb 0eh
                setb 1bh
                setb 1ch
          CLR 20H
;                SETB SCON.4
          CLR ODH                 ;SEND DATA READY AND RUN SYSTEM
          CLR 1FH
                MOV 3EH,#31H
          CLR 19h
          POP OEOH
          RETI
TRAN_ALL1:JC TRAN_ALL7
          SJMP TRAN_ALL8
TRAN_ALL7:LCALL CHANNAL
                CLR 29H
```

```
TRAN_ALL6:MOVX A,@DPTR
          MOV 18H,A              ;SAVE LOW ORDER
          INC DPTR
          MOVX A,@DPTR           ;READ HIGH BYTE
          LCALL A_TO_D
          SETB 18H               ;FOR SEND DATA READY 8 TIMES
          INC DPTR
          MOV 3BH,DPH
          MOV 3CH,DPL
          POP 0E0H
          JB 20H,TRAN_ALL4
TRAN_ALL2:ACALL NEXT_DATA
          JB 20H,TRAN_ALL3
          RETI
TRAN_ALL4:CLR 20H
          RETI
TRAN_ALL5:      DJNZ 4CH,ALL_5_1
                MOV 4CH,#8D
                MOV 4BH,#50H
                mov 1fh,#20h
              ;  CLR P3.5
                CLR SCON.1
              ;  SETB SCON.4
                        mov sbuf,#0Ah
                        setb 0eh
                        setb 1bh
                        setb 1ch
                CLR 20H
                CLR 28H
                CLR 19H
```

```
                MOV 3EH,#31H
                MOV 3FH,#1AH
                RETI
ALL_5_1:        PUSH OEOH
                PUSH OOH
                CLR SCON.1
                ACALL CHANNAL
                MOV RO,4BH
                MOV 18H,@RO
                INC RO
                MOV A,@RO
                LCALL A_TO_D
                CLR 20H
                CLR 29H
                SETB 18H
                INC RO
                MOV 4BH,RO
                POP OOH
                POP OEOH
                RETI
    ;-----------------------------------------------------------;
          ;SUBROUTINE SEND DATA ALL
          ;IN (19),1A,1B,1C,1D,1E,1F,3E
          ;OUT SBUF,(19)
NEXT_DATA:JB 19H,NEXT_1
          JB O5H,NEXT_5
          JB OBH,NEXT_5
NEXT_2:   CLR SCON.1
          MOV SBUF,3EH          ;SEND NUMBER OF CHANNAL
          INC 3EH
```

```
            SETB 19H
NEXT_3:     MOV 3FH,#1AH
            RET
NEXT_1:     PUSH OEOH
            PUSH OOH
            MOV RO,3FH
            MOV A,RO
            CJNE A,#2OH,TRANS_6_2
            MOV 3FH,#1AH
            CLR 18H
            SETB 2OH
            SJMP TRANS_6_3
NEXT_5:     CLR SCON.1
            MOV SBUF,OFH
            MOV 3FH,#1AH
            SETB 19H
            RET
TRANS_6_2:CLR SCON.1
            MOV SBUF,@RO
            INC 3FH
TRANS_6_3:POP OOH
            POP OEOH
            RET
    ;-------------------------------------------------------------;
            ;SUBROUTINE SEND DATA NEXT OF ONE CHANNAL
            ;IN (05,18),34,35,3B,3C
            ;OUT 3B,3C(05,18)
TRAN_ONE: JB 18H,TR_ONE_4
TR_ONE_5: MOV DPH,3BH
            MOV DPL,3CH
```

```
          LCALL RAM1              ;INCREMENT RAM =14 TIMES
          PUSH OEOH
          MOV A,DPH
          CJNE A,34H,TR_ONE_1
          MOV A,DPL
          CJNE A,35H,TR_ONE_1
TR_ONE_7:   ;    CLR P3.5                ;RELAY OFF
          CLR SCON.1
               mov sbuf,#06h
               setb 17h
        ;       SETB SCON.4
;          CLR ODH                 ;SEND DATA READY AND RUN SYSTEM
          CLR 1FH
          POP OEOH
               CLR 19H
               CLR 20H
          CLR 05H
          RETI
TR_ONE_4: ACALL NEXT_DATA
          JB 20H,TR_ONE_5
          RETI
TR_ONE_1: JC TR_ONE_6
          SJMP TR_ONE_7
TR_ONE_6: LCALL CHANNAL
TR_ONE_3: MOVX A,@DPTR
          MOV 18H,A               ;SAVE LOW ORDER
          INC DPTR
          MOVX A,@DPTR            ;READ HIGH BYTE
          LCALL A_TO_D
          SETB 18H                ;FOR SEND DATA READY 8 TIMES
```

```
            INC DPTR
            MOV 3BH,DPH
            MOV 3CH,DPL
            CLR 20H
            POP 0E0H
            RETI
    ;-------------------------------------------------------------;
            ;SUBROUTINE SEND NEXT DATA OF AMOUNT FUNCTION
            ;IN (06,18),30,34,35,3B,3C,40,41
            ;OUT 30,3B,3C,40,41(06,18,0D)
TRAN_AMT: JB 18H,TRAN_AMT2
TR_AMT_5: MOV DPH,3BH
            MOV DPL,3CH
            PUSH 0E0H
            MOV A,DPH
            CJNE A,34H,TR_AMT_1
            MOV A,DPL
            CJNE A,35H,TR_AMT_1
TR_AMT_8:  ; CLR P3.5                ;RELAY OFF
            CLR SCON.1
            mov 1fh,#20h
            mov sbuf,#0AH
            setb 0eh
            setb 1bh
            setb 1ch
        clr 18h
                clr 29h
                mov 3ah,#00h
                mov 3bh,#00h
        ;         SETB SCON.4
```

```
                MOV 3EH,#31H
          CLR 20H
;          CLR 0DH                ;SEND DATA READY AND RUN SYSTEM
          CLR 1FH
          POP 0E0H
                CLR 19H
          CLR 06H
          RETI
TR_AMT_1: JC TR_AMT_9
          SJMP TR_AMT_8
TR_AMT_9:       LCALL CHANNAL
                JNB 29H,TR_AMT_3
TR_AMT_A:       CLR 29H
          CLR C
          MOV A,40H
          SUBB A,#1H
          MOV 40H,A
          MOV A,41H
          SUBB A,#0H
          JC TR_AMT_8
          MOV 41H,A
TR_AMT_4:       CLR SCON.1
                MOV SBUF,#0AH
TR_AMT_3: MOVX A,@DPTR
          MOV 18H,A              ;SAVE LOW ORDER
          INC DPTR
          MOVX A,@DPTR           ;READ HIGH BYTE
          LCALL A_TO_D
          SETB 18H               ;FOR SEND DATA READY 8 TIMES
          INC DPTR
```

```
          MOV 3BH,DPH
          MOV 3CH,DPL
          POP 0E0H
          sjmp tr_amt_6
TRAN_AMT2:ACALL NEXT_DATA
          JB 20H,TR_AMT_5
          RETI
TR_AMT_6: CLR 20H
          RETI
    ;-------------------------------------------------------------;
          ;SUBROUTINE SEND NEXT DATA OF ONE AND AMOUNT
          ;IN (0B,18,0D)3B,3C,34,35,40,41
          ;OUT (0B,18,0D)3B,3C,34,35,40,41
TRAN_OMT: JB 18H,TR_OMT_2
TR_OMT_5: MOV DPH,3BH
          MOV DPL,3CH
          LCALL RAM1              ;INCREMENT RAM =14 TIMES
          PUSH 0E0H
          MOV A,DPH
          CJNE A,34H,TR_OMT_1
          MOV A,DPL
          CJNE A,35H,TR_OMT_1
TR_OMT_8: ;   CLR P3.5                ;RELAY OFF
          CLR SCON.1
          mov sbuf,#06h
          setb 17h
          ;       SETB SCON.4
;          CLR 0DH                 ;SEND DATA READY AND RUN SYSTEM
          CLR 1FH
          CLR 20H
```

```
          POP 0E0H
          CLR 19H
          CLR 0BH
          RETI
TR_OMT_1: JC TR_OMT_3
          SJMP TR_OMT_8
TR_OMT_3: MOV A,40H
          CLR C
          SUBB A,#1H
          MOV 40H,A
          MOV A,41H
          SUBB A,#0H
          JC TR_OMT_8
          MOV 41H,A
TR_OMT_4: LCALL CHANNAL
          MOVX A,@DPTR
          MOV 18H,A              ;SAVE LOW ORDER
          INC DPTR
          MOVX A,@DPTR           ;READ HIGH BYTE
          LCALL A_TO_D
          SETB 18H               ;FOR SEND DATA READY 8 TIMES
          INC DPTR
          MOV 3BH,DPH
          MOV 3CH,DPL
          POP 0E0H
          JB 20H,TR_OMT_6
TR_OMT_2: LCALL NEXT_DATA
          JB 20H,TR_OMT_5
          RETI
TR_OMT_6: CLR 20H
```

```
          RETI
    ;-------------------------------------------------------------;
          ;SUBROUTINE SEND CHANNAL
          ;IN (05,0B),3E,3F
          ;OUT SBUF,(29)
CHANNAL:  JB 05H,CHANN_1
          JB 0BH,CHANN_1
          PUSH 0E0H
          MOV A,3EH
                cjne a,#38h,chann_4
                mov 1fh,#0dh
CHANN_4:  CJNE A,#39H,CHANN_2
          SETB 29H
                mov 1fh,#20h
          LCALL SAFETY
          MOV 3EH,#31H
                clr 19h
                JB 06H,CHANN_3
                clr scon.1
                mov sbuf,#0ah
                sjmp chann_3
CHANN_2:  CLR SCON.1
          MOV SBUF,3EH
          INC 3EH
CHANN_3:  MOV 3FH,#1AH
          POP 0E0H
          RET
CHANN_1:  CLR SCON.1
          MOV SBUF,0FH
          RET
```

```
;-------------------------------------------------;
        ;SUBROUTINE RECEIVE INSTRUCTION
        ;IN SBUF,(00H)
        ;OUT SCON.0
DATA:   LCALL CL_DATA
        PUSH 0E0H
        PUSH 01H
        MOV A,SBUF
        MOV R1,A
        CLR C
        SUBB A,#3AH          ;CHECK DATA MORE THAN 3AH ?
        JNC NUMBER_NO
        MOV A,R1
        CLR C
        SUBB A,#30H          ;CHECK DATA MORE THAN 30H ?
        JC DATA_NO
            SETB 21H
        MOV A,R1
        LCALL CUT_ZERO       ;GO TO RECEIVE NUMBER THAT INTEREST
        SJMP G_BACK_1
NUMBER_NO:    JNB 21H,NUMB_NO
        MOV A,R1
            CLR 21H
        ACALL CH_INST        ;GO TO CHECK INSTRUCTION
        SJMP G_BACK_1
DATA_NO: CLR C
NUMB_NO: CLR 00H
            CLR SCON.4
        CLR 21H
        LCALL CR_DATA
```

```
G_BACK_1: POP 01H
          POP 0E0H
          CLR SCON.0
          RETI
    ;-------------------------------------------------------------;
             ;SUBROUTINE RECEIVE INSTRUCTION TO SELECT
             ;IN A
             ;OUT (00,02,03,04,05,06,0B,0C)
CH_INST:  CJNE A,#'S',INST_NO0
          SETB 02H
          SJMP INST_OK
INST_NO0: CJNE A,#'s',INST_NO  ;INSTRUCTION SEND DATA
          SETB 02H
          SJMP INST_OK
INST_NO:  CJNE A,#'R',INST_NO1
          SETB 03H
          SJMP INST_OK
INST_NO1: CJNE A,#'r',INST_NO2  ;INSTRUCTION RECEIVE DELAY TIME
          SETB 03H
          SJMP INST_OK
INST_NO2: CJNE A,#'D',INST_NO3
          SETB 04H
          SJMP INST_OK
INST_NO3: CJNE A,#'d',INST_NO4
          SETB 04H
          SJMP INST_OK
INST_NO4: CJNE A,#'O',INST_NO5
          SETB 05H
          SJMP INST_OK
INST_NO5: CJNE A,#'o',INST_NO6
```

```
          SETB 05H
          SJMP INST_OK
INST_NO6: CJNE A,#'B',INST_NO7
          SETB 06H
          SJMP INST_OK
INST_NO7: CJNE A,#'b',INST_NO8
          SETB 06H
          SJMP INST_OK
INST_NO8: CJNE A,#'C',INST_NO9
          SETB 14H                    ;FOR RECEIVE CHANNAL
          SETB 0BH
          SJMP INST_OK
INST_NO9: CJNE A,#'c',INST_NOA
          SETB 0BH
          SETB 14H                    ;FOR RECEIVE CHANNAL
          SJMP INST_OK
INST_NOA: CJNE A,#'T',INST_NOB
          SETB 22H
          SJMP INST_OK
INST_NOB: CJNE A,#'t',NO_INST
          SETB 22H
INST_OK:  ACALL R_CODE                ;CHECK CODE OK ?
          JB 0CH,CODE_OK
          CLR 00H                     ;CODE NO_CORRECT
          CLR 02H
          CLR 03H
          CLR 04H
          CLR 05H
          CLR 06H
          CLR 0BH
```

```
            CLR 14H
            CLR 22H
                  CLR 07H
                  CLR 08H
                  CLR 09H
                  CLR 0AH
            SETB 01H
            RET
NO_INST:    CLR 00H
CODE_OK:    LCALL CR_DATA
            CLR 0CH
            RET
    ;---------------------------------------;
               ;SUBROUTINE RESET SYSTEM
               ;IN (03)
               ;OUT (03,0E,0F)
INST_2_1:   LCALL CL_DATA
                  PUSH 0E0H
                  MOV A,SBUF
                  CJNE A,#0DH,INST_2_2
                  CLR SCON.4
            SETB 0FH                ;FOR TELL RESET SYSTEM
            CLR 03H
            CLR 00H
            LCALL RESET_SYS
            SETB 0EH                ;FOR SEND ACK
INST_2_3:   MOV TH0,#3CH
            MOV TL0,#0AFH
            SETB P3.5
            SETB TR0
```

```
            POP 0E0H
        CLR SCON.0
        RETI
INST_2_2:   SETB 13H
            CLR 00H
            CLR 03H
            SJMP INST_2_3
;-------------------------------------;
        ;SUBROUTINE RECEIVE DELAY TIME FROM MAIN COM
        ;IN SBUF,(04)
        ;OUT (04)
INST_3_1:   JNB 0DH,INST_3_2
            SETB 1FH
            SJMP G_BACK_21
INST_3_2: LCALL CL_DATA
        PUSH 0E0H
        PUSH 01H
        MOV A,SBUF
        MOV R1,A
        CLR C
        SUBB A,#3AH         ;CHECK DATA MORE THAN 3AH ?
        JNC NO_INST_3
        MOV A,R1
        CLR C
        SUBB A,#30H         ;CHECK DATA MORE THAN 30H ?
        JC NO_3_1
            SETB 23H
        MOV A,R1
        LCALL CUT_ZERO      ;GO TO RECEIVE NUMBER THAT INTEREST
        SJMP G_BACK_2
```

```
NO_INST_3:CLR C
          CLR 32H
          CLR 23H
          CLR SCON.4
          LCALL CR_DATA
          CLR 00H
          CLR 04H
          SJMP G_BACK_2
NO_3_1:   JNB 23H,NO_INST_3
          CLR 23H
          MOV A,R1
          CJNE A,#0DH,NO_INST_3
          CLR 0FH              ;RUN SYSTEM
          LCALL R_DELAY
          JB 32H,NO_INST_3
          MOV TH0,#3CH
          MOV TL0,#0AFH
          CLR SCON.4
          SETB P3.5
          SETB TR0
          CLR C
          CLR 00H
          CLR 04H
          SETB 0EH             ;FOR SEND ACK BACK
G_BACK_2: POP 01H
          POP 0E0H
G_BACK_21:CLR SCON.0
          RETI
    ;-------------------------------------------------------;
          ;SUBROUTINE RECEIVE CHANNAL FOR SEND BACK
```

```
            ;IN SBUF,(05,11)
            ;OUT 0FH,(TR0,11)
INST_4_1: LCALL CL_DATA
          JB 11H,INST_4_2
          PUSH 0E0H
          PUSH 01H
          MOV A,SBUF
          MOV R1,A
          CLR C
          SUBB A,#39H          ;CHECK DATA MORE THAN 39H ?
          JNC NO_4_2
          MOV A,R1
          CLR C
          SUBB A,#31H          ;CHECK DATA MORE THAN 31H ?
          JC NO_4_2
          SETB 11H
          MOV 0FH,A            ;SAVE CHANNAL
NO_4_1:   POP 01H
          POP 0E0H
OK_4_2:   LCALL CR_DATA
OK_4_1:   CLR SCON.0
          RETI
INST_4_2:       PUSH 0E0H
                MOV A,SBUF
                CJNE A,#0DH,INST_4_3
                CLR SCON.4
                POP 0E0H
          MOV TH0,#3CH
          MOV TL0,#0AFH
          SETB P3.5
```

```
          SETB TR0
          CLR 11H
          CLR 00H
          SJMP OK_4_1
NO_4_2:   CLR 00H
               CLR SCON.4
          CLR 05H
          SJMP NO_4_1
INST_4_3:      CLR 00H
               CLR 05H
               POP 0E0H
               SJMP OK_4_2
    ;-------------------------------------------------------------;
          ;SUBROUTINE SEND DATA ALL CHANNAL BUT LESS THAN 1000 VALUES
          ;IN (06)
          ;OUT (06)
INST_5_1: LCALL CL_DATA
          PUSH 0E0H
          PUSH 01H
          MOV A,SBUF
               JNB 23H,INST_5_2
               JNB 24H,INST_5_3
               CLR 24H
          CJNE A,#0DH,INST_5_3
               CLR SCON.4
              CLR 23H
          ACALL CH_NUM
               JB 31H,NO_5_1
          MOV TH0,#3CH
          MOV TL0,#0AFH
```

```
          SETB P3.5
          SETB TRO
          CLR 00H
          SJMP OK_5_2
INST_5_2:       JNB 24H,INST_5_3
                CLR 24H
          CJNE A,#'-',INST_5_3
                SETB 23H
          ACALL CH_NUM
                JB 31H,NO_5_1
          SJMP OK_5_1
INST_5_3: MOV R1,A
          CLR C
          SUBB A,#3AH           ;CHECK DATA MORE THAN 40H ?
          JNC NO_5_1
          MOV A,R1
          CLR C
          SUBB A,#30H           ;CHECK DATA MORE THAN 30H ?
          JC NO_5_1
                SETB 24H
          CLR C
          MOV A,R1
          LCALL CUT_ZERO
          SJMP OK_5_1
NO_5_1:   CLR 06H
          CLR 00H
                CLR C
                CLR SCON.4
                CLR 31H
                CLR 23H
```

```
                CLR 24H
OK_5_1:    LCALL CR_DATA
OK_5_2:    POP 01H
           POP 0E0H
           CLR SCON.0
           RETI
    ;-------------------------------------------------------------;
              ;SUBROUTINE CHECK CHANNAL AND AMOUNT OF GROUP DATA
              ;IN SBUF,(0B)
              ;OUT 0FH,(0BH)
INST_6_1: LCALL CL_DATA
           PUSH 0E0H
           PUSH 01H
           MOV A,SBUF
           MOV R1,A
           CLR C
           JB 14H,INST_6_5       ;FOR RECEIVE CHANNAL
           SUBB A,#3AH           ;CHECK DATA MORE THAN 3AH ?
           JNC NO_6_1
           MOV A,R1
           CLR C
           SUBB A,#30H           ;CHECK DATA MORE THAN 30H ?
           JC INST_6_2
           CLR C
                 SETB 23H
           MOV A,R1
           LCALL CUT_ZERO
           SJMP OK_6_1
INST_6_2: MOV A,R1
           JB 15H,INST_6_3
```

```
            JB 16H,INST_6_4
                  JNB 24H,NO_6_1
                  JNB 23H,NO_6_1
                  CLR 23H
            CJNE A,#0DH,NO_6_1
                  CLR SCON.4
            CLR 24H
            ACALL CH_NUM
                  JB 31H,NO_6_1
            MOV TH0,#3CH
            MOV TL0,#0AFH
            SETB P3.5
            SETB TR0
            CLR 00H
            SJMP OK_6_2
INST_6_3:   CLR 15H
            JB 23H,NO_6_1
            CJNE A,#',',NO_6_1
            SETB 16H
            SJMP OK_6_1
INST_6_4:   CLR 16H
                  JNB 23H,NO_6_1
                  CLR 23H
            CJNE A,#'-',NO_6_1
            SETB 24H
            ACALL CH_NUM
                  JB 31H,NO_6_1
            SJMP OK_6_1
INST_6_5:   SUBB A,#39H
            CLR 14H
```

```
            JNC NO_6_1
            CLR C
            MOV A,R1
            SUBB A,#31H
            JC NO_6_1
            MOV 0FH,A
            SETB 15H
            SJMP OK_6_1
NO_6_1:     CLR 0BH
            CLR 00H
                  CLR SCON.4
                  CLR 31H
            CLR C
            CLR 23H
            CLR 24H
            CLR 16H
            CLR 15H
OK_6_1:     LCALL CR_DATA
OK_6_2:     POP 01H
            POP 0E0H
            CLR SCON.0
            RETI
    ;-------------------------------------------------------------;
               ;SUBROUTINE CHECK AMOUNT OF GROUP DATA
               ;IN A,08,09,0A,0B,0D,(12)
               ;OUT 11,12,13,14,15,16,17,(12)
CH_NUM:     JB 12H,CH_NUM_6
            MOV A,0DH
            CJNE A,#1H,CH_NUM_1
            ACALL CHEC1
```

```
          MOV 10H,08H           ;1
          MOV 11H,#00H          ;2
CH_N_1_1: MOV 12H,#00H          ;3
CH_N_1_2: MOV 13H,#00H          ;4
          SJMP CH_NUM_4
CH_NUM_1: CJNE A,#2H,CH_NUM_2
          ACALL CHEC1
          MOV 11H,08H           ;1
          ACALL CHEC2
          MOV 10H,09H           ;2
          SJMP CH_N_1_1
CH_NUM_2: CJNE A,#3H,CH_NUM_3
          ACALL CHEC1
          MOV 12H,08H
          ACALL CHEC2
          MOV 11H,09H
          ACALL CHEC3
          MOV 10H,0AH
          SJMP CH_N_1_2
CH_NUM_3:       CJNE A,#4H,CH_NUM_NO
          ACALL CHEC1
          MOV 13H,08H
          ACALL CHEC2
          MOV 12H,09H
          ACALL CHEC3
          MOV 11H,0AH
          ACALL CHEC4
          MOV 10H,0BH
CH_NUM_4: SETB 12H
CH_NUM_5: MOV 08H,#00H
```

```
            MOV 09H,#00H
            MOV 0AH,#00H
            MOV 0BH,#00H
            MOV 0CH,#00H
            MOV 0DH,#00H
                  CLR 0AH
                  CLR 09H
                  CLR 08H
                  CLR 07H
            RET
CH_NUM_6:   MOV A,0DH
            CJNE A,#1H,CH_NUM_7
            ACALL CHEC1
            MOV 14H,08H           ;1
            MOV 15H,#00H          ;2
CH_N_2_1:   MOV 16H,#00H          ;3
CH_N_2_2:   MOV 17H,#00H          ;4
            SJMP CH_NUM_A
CH_NUM_7:   CJNE A,#2H,CH_NUM_8
            ACALL CHEC1
            MOV 15H,08H           ;1
            ACALL CHEC2
            MOV 14H,09H           ;2
            SJMP CH_N_2_1
CH_NUM_NO:        SETB 31H
                  SJMP CH_NUM_A
CH_NUM_8:   CJNE A,#3H,CH_NUM_9
            ACALL CHEC1
            MOV 16H,08H
            ACALL CHEC2
```

```
            MOV 15H,09H
            ACALL CHEC3
            MOV 14H,0AH
            SJMP CH_N_2_2
CH_NUM_9:       CJNE A,#4H,CH_NUM_NO
            ACALL CHEC1
            MOV 17H,08H
            ACALL CHEC2
            MOV 16H,09H
            ACALL CHEC3
            MOV 15H,0AH
            ACALL CHEC4
            MOV 14H,0BH
CH_NUM_A:   CLR 12H
            AJMP CH_NUM_5
CHEC1:      CLR C
            MOV A,08H
            SUBB A,#30H
            MOV 08H,A
            RET
CHEC2:      CLR C
            MOV A,09H
            SUBB A,#30H
            MOV 09H,A
            RET
CHEC3:      CLR C
            MOV A,0AH
            SUBB A,#30H
            MOV 0AH,A
            RET
```

```
CHEC4:    CLR C
          MOV A,0BH
          SUBB A,#30H
          MOV 0BH,A
          RET
    ;-------------------------------------------------------;
             ;SUBROUTINE CHECK CODE OF SUBMICRO
             ;IN 0DH,08H,09H,0AH,31H,32H,33H
             ;OUT 0CH
R_CODE:   MOV A,0DH
          CJNE A,#1H,RC_1
          MOV A,33H              ;ORDER 3
          CJNE A,#30H,NO_CO
          MOV A,32H              ;ORDER 2
          CJNE A,#30H,NO_CO
          MOV A,31H              ;ORDER 1
          CJNE A,08H,NO_CO
          SETB 0CH
          CLR 0AH
          SJMP NO_CO_1
NO_CO:         JNB 0DH,NO_CO_1
               SETB 1FH
NO_CO_1:  MOV 0DH,#00H
          MOV 08H,#00H
          MOV 09H,#00H
          MOV 0AH,#00H
          MOV 0BH,#00H
          MOV 0CH,#00H
          RET
RC_1:     CJNE A,#2H,RC_3          ;1 ZERO
```

```
            MOV A,33H
            CJNE A,#30H,NO_CO
            MOV A,32H
            CJNE A,08H,NO_CO
            MOV A,31H
            CJNE A,09H,NO_CO
            SETB 0CH
            CLR 09H
            SJMP NO_CO_1
RC_3:           CJNE A,#3H,RC_4
            MOV A,33H                   ;NO ZERO
            CJNE A,08H,NO_CO
            MOV A,32H
            CJNE A,09H,NO_CO
            MOV A,31H
            CJNE A,0AH,NO_CO
            SETB 0CH
            CLR 08H
            SJMP NO_CO_1
RC_4:   ;  LCALL CR_DATA
                CLR 07H
                CLR 09H
                CLR 0AH
                CLR 08H
                CLR SCON.4
                SETB 0CH
            SJMP NO_CO_1
    ;-----------------------------------------------;
            ;SUBROUTINE RECEIVE PASSWORD
            ;IN SBUF
```

```
            ;OUT 00H(BIT)
PASSWORD: PUSH 0E0H
          MOV A,SBUF
          CJNE A,#'!',NO_MATCH
          SETB 00H              ;PASSWORD OK
          LCALL CR_DATA
NO_MATCH: POP 0E0H
          CLR SCON.0
          RETI
    ;-------------------------------------;
             ;SUBROUTINE TIMER 0 INTERUPT
             ;IN R2,0EH,30H,(10,13,0E)
             ;OUT R2,0EH,30H,(10,13,0E)
TIM0_0:   JB 10H,TIM0_3         ;DELAY FOR RECEIVE DATA NEXT
          DJNZ R2,TIM0_1        ;DELAY = 700 ms
          CLR TR0
          JB 28H,TIM0_B         ;SEND FORM REAL TIME
          JB 13H,TIM0_5
          JB 1FH,TIM0_2         ;FOR SEND DATA FULL
          JB 0EH,TIM0_2         ;FOR SEND ACK
          PUSH 0E0H
                MOV A,34H
                CJNE A,#00H,TIM0_A
                MOV A,35H
                CJNE A,#00H,TIM0_A
                CLR 05H
                CLR 06H
                CLR 0BH
                SETB 25H
                SJMP TIM0_2_1
```

```
TIM0_A:   MOV DPTR,#0000H       ;SAVE R1 FOR ADDRESS OF DATA FOR SEND
          JB 05H,TIM0_6         ;SEND ONE CHANNAL
          JB 06H,TIM0_7         ;SEND SELECT AMOUNT
          JB 0BH,TIM0_8         ;SEND ONE CHANNAL AND SELECT AMOUNT
TIM0_9:   MOVX A,@DPTR
          MOV 18H,A             ;SAVE LOW ORDER
          INC DPTR
          MOVX A,@DPTR          ;READ HIGH BYTE
          LCALL A_TO_D
          MOV SBUF,3DH          ;SEND PASSWORD " ~ "
          INC DPTR
          MOV 3BH,DPH           ;TO POINT DATA-
          MOV 3CH,DPL           ;FOR SEND
          SETB 18H              ;FOR SEND DATA ALL 8
TIM1:     POP 0E0H
TIM:      MOV R2,#25D           ;**********************
          RETI
TIM0_B:   LJMP TIM0_B_1
TIM0_2_1: POP 0E0H
TIM0_2:   MOV SBUF,33H           ;SEND HIGH ORDER ADDRESS OF CPU
          SJMP TIM
TIM0_1:   CLR TR0
          MOV TL0,#0AFH
          MOV TH0,#3CH
          SETB TR0
          RETI
TIM0_3:   CLR TR0
          DJNZ 0EH,TIM0_4
          LCALL RESET_1
          CLR 10H
```

```
          SETB P3.5
          MOV R2,#25D
          SETB 13H
          SJMP TIMO_1
TIMO_5:   MOV SBUF,#'?'
          SETB 17H              ;USE FOR TRANSMITE INT.
          CLR 13H
          SJMP TIM
TIMO_4:   MOV THO,#OOH
          MOV TLO,#OOH
          SETB TRO
          RETI
TIMO_6:   LCALL SE_ONE
          LJMP TIMO_9
TIMO_7:   LCALL SE_AMT
          JB 16H,TIMO_7_1
          LCALL CH_AMT
               JB 16H,TIMO_7_1
          LJMP TIMO_9
TIMO_7_1: CLR 16H
               CLR O6H
               CLR OBH
          LJMP TIM1
TIMO_8:   LCALL SE_ON_AM
          JB 16H,TIMO_7_1
          LCALL CH_AMT
               JB 16H,TIMO_7_1
          LJMP TIMO_9
TIMO_B_1: PUSH OEOH
          MOV A,5OH
```

```
            MOV 18H,A
            MOV A,51H
            LCALL A_TO_D
            MOV SBUF,3DH
            SETB 18H
            POP 0E0H
            SJMP TIM
        ;---------------------------------------;
            ;SUBROUTINE POINT DATA IN RAM FOR SEND ONE CHANNAL
            ;IN DPTR,0FH
            ;OUT DPTR,0FH
SE_ONE:     MOV A,0FH
SE_ONE_1:   CJNE A,#00H,SE_ONE_2
            MOV A,0FH
            ADD A,#31H
            MOV 0FH,A
            RET
SE_ONE_2:   INC DPTR
            INC DPTR
            DEC A
            SJMP SE_ONE_1
        ;-----------------------------------------------------;
            ;SUBROUTINE POINT DATA IN RAM FOR SEND SELECT AMOUNT
            ;IN  10H,11H,12H,13H,DPTR
            ;OUT DPTR,10H,11H,12H,13H,(16H)
SE_AMT:     CLR C
            PUSH 10H
            PUSH 11H
            PUSH 12H
            PUSH 13H
```

```
                MOV A,13H
                CJNE A,17H,SE_AMT_7
                MOV A,12H
                CJNE A,16H,SE_AMT_7
                MOV A,11H
                CJNE A,15H,SE_AMT_7
                MOV A,10H
                CJNE A,14H,SE_AMT_7
SE_AMT_6:  MOV A,10H                    ;RECEIVE GROUP FIRST
           SUBB A,#1H
           MOV 10H,A
           MOV A,11H
           SUBB A,#0H
           MOV 11H,A
           MOV A,12H
           SUBB A,#0H
           MOV 12H,A
           MOV A,13H
           SUBB A,#0H
           MOV 13H,A
           JC Ma_Error
SE_AMT_1:  MOV A,10H
           CJNE A,#00H,SE_AMT_2
           MOV A,11H
           CJNE A,#00H,SE_AMT_3
           MOV A,12H
           CJNE A,#00H,SE_AMT_4
           MOV A,13H
           CJNE A,#00H,SE_AMT_5
           POP 13H
```

```
            POP 12H
            POP 11H
            POP 10H
            RET
SE_AMT_7:       JNC Ma_Error
                CLR C
                SJMP SE_AMT_6
SE_AMT_2: DEC 10H
            ACALL RAM
            SJMP SE_AMT_1
SE_AMT_3: DEC 11H
            MOV 10H,#9D
            ACALL RAM
            SJMP SE_AMT_1
SE_AMT_4: DEC 12H
            MOV 11H,#9D
            MOV 10H,#9D
            ACALL RAM
            SJMP SE_AMT_1
SE_AMT_5: DEC 13H
            MOV 12H,#9D
            MOV 11H,#9D
            MOV 10H,#9D
            ACALL RAM
            SJMP SE_AMT_1
Ma_Error: LCALL CR_DATA
                CLR SCON.4
            SETB 16H
                CLR C
            POP 13H
```

```
          POP 12H
          POP 11H
          POP 10H
          RET
    ;-------------------------------------------------------;
          ;SUBROUTINE CHECK AMOUNT OF DATA
          ;IN 10,11,12,13,14,15,16,17
          ;OUT 40,41
CH_AMT:   MOV 40H,#00H                  ;LOW ORDER
          MOV 41H,#00H                  ;HIGH ORDER
CH_AMT0:  MOV A,10H
          CJNE A,14H,CH_AMT1
CH_AMT6:  MOV A,11H
          CJNE A,15H,CH_AMT2
CH_AMT7:  MOV A,12H
          CJNE A,16H,CH_AMT3
CH_AMT8:  MOV A,13H
          CJNE A,17H,CH_AMT4
CH_AMT9:  RET
CH_AMT1:  INC 10H
                ACALL CH_AMTA
          ACALL CH_AMT5
          SJMP CH_AMT0
CH_AMT2:  INC 11H
                ACALL CH_AMTA_1
                PUSH 10H
                MOV 10H,#0AH
CH_AMT2_1:ACALL CH_AMT5
                DJNZ 10H,CH_AMT2_1
                POP 10H
```

```
          SJMP CH_AMT6
CH_AMT3:  INC 12H
                ACALL CH_AMTA_2
                PUSH 10H
                MOV 10H,#100D
CH_AMT3_1:ACALL CH_AMT5
                DJNZ 10H,CH_AMT3_1
                POP 10H
          SJMP CH_AMT7
CH_AMT4:  ACALL CH_AMTA_4
          SJMP CH_AMT9
CH_AMT5:  PUSH 0E0H
          MOV A,40H
          CLR C
          ADD A,#1H
          MOV 40H,A
          MOV A,41H
          ADDC A,#0H
          MOV 41H,A
          POP 0E0H
          RET
CH_AMTA:        MOV A,10H
                CJNE A,#0AH,CH_AMTA_3
                MOV 10H,#00H
                INC 11H
CH_AMTA_1:      MOV A,11H
                CJNE A,#0AH,CH_AMTA_3
                MOV 11H,#00H
                INC 12H
CH_AMTA_2:      MOV A,12H
```

```
                CJNE A,#0AH,CH_AMTA_3
                MOV 12H,#00H
CH_AMTA_4:      INC 13H
                MOV A,13H
                CJNE A,#1H,Ma_Error1
CH_AMTA_3:      RET
Ma_Error1:      LCALL CR_DATA
                SETB SCON.4
                SETB 16H
                MOV 40H,#00H
                MOV 41H,#00H
                RET
    ;-------------------------------------------------------;
          ;SUBROUTINE INCREMENT DPTR =16 VALUES
          ;IN DPTR
          ;OUT DPTR
RAM:      PUSH 0E0H
          CLR C
          MOV A,DPL
          ADD A,#16D
          MOV DPL,A
          MOV A,DPH
          ADDC A,#0H
          MOV DPH,A
          POP 0E0H
          RET
      ;-------------------------------------------------------;
          ;SUBROUTINE INCREMENT DPTR = 14 VALUES
          ;IN DPTR
          ;OUT DPTR
```

```
RAM1:     PUSH OEOH
          CLR C
          MOV A,DPL
          ADD A,#14D
          MOV DPL,A
          MOV A,DPH
          ADDC A,#OH
          MOV DPH,A
          POP OEOH
          RET
    ;-------------------------------------------------------;
          ;SUBROUTINE POINT DATA IN RAM FOR SEND ONE CH. AND AMOUNT
          ;IN OFH,10H,11H,12H,13H,DPTR
          ;OUT  DPTR,(16)
SE_ON_AM: LCALL SE_ONE
          LCALL SE_AMT
          RET
      ;-------------------------------------------------------;
          ;SUBROUTINE READ CODE OF CPU FROM DIP SWITCH
          ;IN P2
          ;OUT 31H,32H,33H
READ_CODE:CLR P1.7                ;CLOSE GATE 244
          MOV A,P2                ;READ CODE
          SETB P1.7               ;OPEN GATE
          MOV R1,#OOH             ;CLEAR RO
          CLR C                   ;CLEAR CARRY
R_C_1:    MOV RO,A                ;SAVE ACC
          SUBB A,#10D
          JNC R_C_2               ;FIND N TIMES OF 10D
          CLR C                   ;VALUE < 10
```

```
        MOV A,R0                ;LOW
        ADD A,#30H              ;CONVERT TO ASCII
        MOV 31H,A               ;SAVE DATA AT 31H
        MOV A,R1
        MOV R1,#00H             ;FIND MID AND HIGH
R_C_3:  MOV R0,A
        SUBB A,#10D             ;LEAVE MID AND HIGH
        JNC R_C_4               ;FIND HIGH
        CLR C
        MOV A,R0
        ADD A,#30H
        MOV 32H,A               ;MID
        MOV A,R1
        ADD A,#30H
        MOV 33H,A               ;HIGH
        RET
R_C_4:  INC R1
        SJMP R_C_3
R_C_2:  INC R1
        SJMP R_C_1
    ;-----------------------------------------;
        ;SUBROUTINE READ DATA FROM 7109
        ;IN 30H,34H,35H
        ;OUT 30H,34H,35H,RAM 2 BYTE,(0D)

R_7109:   JB 30H,R_7109_3_1
          PUSH 0E0H
          MOV A,34H             ;FROM DPH OF RAM
          CJNE A,#3EH,R_7109_2_1
          MOV A,35H             ;FROM DPL OF RAM
```

```
                CJNE A,#80H,R_7109_2_1
                CLR IE.0
                SETB 0DH                ;DATA NOW FULL
                MOV 30H,#8D
                MOV P1,#0F0H
                POP 0E0H
                JB 30H,R_7109_2_3
                RETI
R_7109_2_1:     CLR P1.6                ;FOR HOLD DATA
                MOV DPH,34H
                MOV DPL,35H
                CLR P1.3                ;CE\LOAD LOW
                CLR P1.5                ;LBEN LOW
                CLR P3.3                ;CLOSE GATE 244
                MOV A,P2
                SETB P3.3               ;OPEN GATE 244
                SETB P1.5               ;LBEN HIGH
                MOVX @DPTR,A
                INC DPTR
                CLR P1.4                ;HBEN LOW
                CLR P3.4                ;CLOSE GATE 244
                MOV A,P2
                SETB P3.4               ;OPEN GATE 244
                SETB P1.4               ;HBEN HIGH
                SETB P1.3               ;CE\LOAD HIGH
                MOVX @DPTR,A
                INC DPTR
                MOV 34H,DPH
                MOV 35H,DPL
                SETB P1.6
```

```
                POP OEOH
                DJNZ 30H,R_7109_1
                MOV 30H,#8D
                MOV P1,#0F0H            ;GO TO CHANNAL 1
                mov c,27h
                mov ie.0,c
                SETB 1EH                ;DATA READY 8 CHANNALS
                RETI
R_7109_1:       INC P1
                RETI
R_7109_2_3:     MOV C,IE.0
                MOV 27H,C
                SETB IE.0
                RETI
R_7109_3_1:     PUSH OEOH
                PUSH 00H
                CLR P1.6                ;FOR HOLD DATA
                CLR P1.3                ;CE\LOAD LOW
                CLR P1.5                ;LBEN LOW
                CLR P3.3                ;CLOSE GATE 244
                MOV A,P2
                SETB P3.3               ;OPEN GATE 244
                SETB P1.5               ;LBEN HIGH
                MOV R0,4BH
                MOV @R0,A
                INC R0
                CLR P1.4                ;HBEN LOW
                CLR P3.4                ;CLOSE GATE 244
                MOV A,P2
                SETB P3.4               ;OPEN GATE 244
```

```
                SETB P1.4               ;HBEN HIGH
                SETB P1.3               ;CE\LOAD HIGH
                MOV @R0,A
                INC R0
                MOV 4BH,R0
                SETB P1.6
                POP 00H
                POP 0E0H
                DJNZ 4CH,R_7109_3_2
                MOV 4CH,#8D
                MOV P1,4AH
                MOV 4BH,#52H
                MOV C,2BH
                MOV IE.0,C
                CLR 30H
                SETB 28H
                MOV TH0,#3CH
                MOV TL0,#0AFH
                MOV R2,#25D
                SETB TR0
                RETI
R_7109_3_2:     INC P1
                RETI
    ;-----------------------------------------------------;
                ;SUBROUTINE CONVERT HEX TO BCD
                ;IN A
                ;OUT 1A,1B,1C,1D,1E,1F
A_TO_D:     PUSH 04H
            PUSH 05H
            PUSH 06H
```

```
        MOV R5,#00H
        JB ACC.4,OVER_LOAD      ;OVER LOAD CANNOT CONVERT
A_D:    JNB ACC.5,POLARITY      ;CHECK POLARITY
        MOV 1AH,#'+'
A_DD:   ANL A,#0FH              ;FOR 4 BIT LOW
        MOV 19H,A
        CLR C
        MOV A,18H
A_D_1:  MOV R4,A
        SUBB A,#100D
        JNC A_D_2
        MOV 18H,A
        MOV A,19H
        SUBB A,#00H
        MOV 19H,A
        JNC A_D_3
        CLR C
        MOV A,R5
        MOV R5,#00H
A_D_5:  MOV R6,A
        SUBB A,#10D
        JNC A_D_4
        CLR C
        MOV A,R6
        ADD A,#30H
        MOV 1CH,A
        MOV A,R5
        ADD A,#30H
        MOV 1BH,A
        MOV R5,#00H
```

```
            MOV A,R4
A_D_6:      MOV R4,A
            SUBB A,#10D
            JNC A_D_7
            CLR C
            MOV A,R4
            ADD A,#30H
            MOV 1EH,A
            MOV A,R5
            ADD A,#30H
            MOV 1DH,A
              POP 06H
              POP 05H
              POP 04H
              RET
OVER_LOAD:MOV 1BH,#'x'
            MOV 1CH,#'x'
            MOV 1DH,#'x'
            MOV 1EH,#'x'
            JNB ACC.5,OV_1
            MOV 1AH,#'+'
OV_2:         POP 06H
              POP 05H
              POP 04H
              RET
OV_1:       MOV 1AH,#'-'
            SJMP OV_2
POLARITY: MOV 1AH,#'-'
            SJMP A_DD
A_D_2:      INC R5
```

```
            SJMP A_D_1
A_D_3:      MOV A,18H
            INC R5
            SJMP A_D_1
A_D_4:      INC R5
            SJMP A_D_5
A_D_7:      INC R5
            SJMP A_D_6
      ;---------------------------------------;
            ;SUBROUTINE DELAY TIME = 0.99s  (BASE TIME)
            ;IN (1DH)
            ;OUT (1DH)
DELAY:      MOV 42H,#0FFH
DELAY_1:    DJNZ 42H,DELAY_2
            SETB 1DH
DELAY_2:    MOV 43H,#0FFH
DELAY_3:    NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
            NOP
```

```
            DJNZ 43H,DELAY_3
            JNB 1DH,DELAY_1
            CLR 1DH
            RET
        ;---------------------------------------;
            ;SUBROUTINE TIMER DELAY FOR READ DATA FROM 7109
            ;IN 36H,37H,38H,39H,3AH,(08H)
            ;OUT 44H,45H,46H,47H,48H,(IE.0)
D_R:        MOV 45H,37H                 ;SAVE ORDER 2
            MOV 44H,36H                 ;SAVE ORDER 1
            MOV 46H,38H                 ;ORDER 3
            MOV 47H,39H                 ;ORDER 4
            MOV 48H,3AH                 ;ORDER 5
D_R_1:      JB OFH,D_R_9                ;TO CLEAR SYSTEM
            MOV A,44H
            CJNE A,#00H,D_R_2
            MOV A,45H
            CJNE A,#00H,D_R_3
            MOV A,46H
            CJNE A,#00H,D_R_4
            MOV A,47H
            CJNE A,#00H,D_R_5
            MOV A,48H
            CJNE A,#00H,D_R_6
            SETB IE.0                   ;ENABLE EXT. INTERUPT
D_R_8:      JB OFH,D_R_A                ;RESET SYSTEM
D_R_7:      JB 1EH,D_R_9_1              ;CHECK READ DATA = 8 CHANEL
            SJMP D_R_8
D_R_9_1:    CLR 1EH
D_R_9:      RET
```

```
D_R_2:    DEC 44H
          LCALL DELAY
          SJMP D_R_1
D_R_3:    DEC 45H
          MOV 44H,#9D
          LCALL DELAY
          SJMP D_R_1
D_R_4:    DEC 46H
          MOV 44H,#9D
          MOV 45H,#9D
          LCALL DELAY
          SJMP D_R_1
D_R_5:    DEC 47H
          MOV 46H,#9D
          MOV 45H,#9D
          MOV 44H,#9D
          LCALL DELAY
          SJMP D_R_1
D_R_6:    DEC 48H
          MOV 47H,#9D
          MOV 46H,#9D
          MOV 45H,#9D
          MOV 44H,#9D
          LCALL DELAY
          SJMP D_R_1
D_R_A:    CLR IE.0
          SJMP D_R_9
        ;--------------------------------------;
         ;SUBROUTINE RECEIVE ZERO THAT INTEREST ONLY
         ;IN A,(07H,08H,09H,0AH),0DH
```

```
        ;OUT 08H,09H,0AH,0BH,0CH,0DH
CUT_ZERO: INC 0DH
          JB 07H,CUT_1
          JB 08H,CUT_2
          JB 09H,CUT_3
          JB 0AH,CUT_4
          MOV 08H,A
          SETB 0AH
          MOV 09H,#00H
          MOV 0AH,#00H
          MOV 0BH,#00H
          MOV 0CH,#00H
          SJMP G_BACK
CUT_4:    MOV 09H,A
          SETB 09H
          CLR 0AH
          SJMP G_BACK
CUT_3:    MOV 0AH,A
          SETB 08H
          CLR 09H
          SJMP G_BACK
CUT_2:    MOV 0BH,A
          SETB 07H
          CLR 08H
          SJMP G_BACK
CUT_1:    MOV 0CH,A
          CLR 07H
G_BACK:   ACALL CR_DATA
          RET
    ;-------------------------------------;
```

```
        ;SUBROUTINE RECEIVE DELAY TIME
        ;IN 08H,09H,0AH,0BH,0CH,0DH
        ;OUT 36H,37H,38H,39H,3AH,08H,09H,0AH,0BH,0CH,0DH
R_DELAY:  MOV A,0DH
          CJNE A,#1H,R_DE_1
          ACALL CHECK1
          MOV 36H,08H          ;1
          MOV 37H,#00H         ;2
R_DE_1_1: MOV 38H,#00H         ;3
R_DE_1_2: MOV 39H,#00H         ;4
R_DE_1_3: MOV 3AH,#00H         ;5
          SJMP R_DE_5
R_DE_1:   CJNE A,#2H,R_DE_2
          ACALL CHECK1
          MOV 37H,08H          ;2
          ACALL CHECK2
          MOV 36H,09H          ;3
          SJMP R_DE_1_1
R_DE_2:   CJNE A,#3H,R_DE_3
          ACALL CHECK1
          MOV 38H,08H
          ACALL CHECK2
          MOV 37H,09H
          ACALL CHECK3
          MOV 36H,0AH
          SJMP R_DE_1_2
R_DE_3:   CJNE A,#4H,R_DE_4
          ACALL CHECK1
          MOV 39H,08H
          ACALL CHECK2
```

```
            MOV 38H,09H
            ACALL CHECK3
            MOV 37H,0AH
            ACALL CHECK4
            MOV 36H,0BH
            SJMP R_DE_1_3
R_DE_4:         CJNE A,#5H,R_DE_NO
            ACALL CHECK1
            MOV 3AH,08H
            ACALL CHECK2
            MOV 39H,09H
            ACALL CHECK3
            MOV 38H,0AH
            ACALL CHECK4
            MOV 37H,0BH
            ACALL CHECK5
            MOV 36H,0CH
R_DE_5:     MOV 08H,#00H
            MOV 09H,#00H
            MOV 0AH,#00H
            MOV 0BH,#00H
            MOV 0CH,#00H
            MOV 0DH,#00H
                CLR 0AH
                CLR 09H
                CLR 08H
                CLR 07H
            RET
R_DE_NO:        SETB 32H
                SJMP R_DE_5
```

```
CHECK1:   CLR C
          MOV A,08H
          SUBB A,#30H
          MOV 08H,A
          RET
CHECK2:   CLR C
          MOV A,09H
          SUBB A,#30H
          MOV 09H,A
          RET
CHECK3:   CLR C
          MOV A,0AH
          SUBB A,#30H
          MOV 0AH,A
          RET
CHECK4:   CLR C
          MOV A,0BH
          SUBB A,#30H
          MOV 0BH,A
          RET
CHECK5:   CLR C
          MOV A,0CH
          SUBB A,#30H
          MOV 0CH,A
          RET
    ;-----------------------------------------------------;
CR_DATA:  MOV TH0,#00H
          MOV TL0,#00H
          MOV 0EH,#34D
          SETB TR0
```

```
          SETB 10H
          RET
     ;-------------------------------------------------;
CL_DATA:  CLR TR0
          MOV 0EH,#34D
          CLR 10H
          RET
     ;-------------------------------------------------;
             ;MAIN PROGRAM
MAIN:     MOV SP,#60H
          LCALL READ_CODE
          ACALL SET_Fn
                ACALL RESET_SYS
MAIN_1:   JB 0DH,MAIN_1                    ;DATA FULL
          JB 0FH,MAIN_1                    ;RESET SYSTEM
          LCALL D_R
          SJMP MAIN_1
    ;--------------------------------------;
             ;SUBROUTINE SET SFRs
SET_Fn:   MOV PCON,#80H
          MOV TMOD,#21H
          MOV TH1,#64D
          MOV TL1,#64D
          MOV 30H,#8H
          MOV 3DH,#'~'
          MOV P1,#0F0H
          MOV SCON,#50H
          MOV IP,#10H
          MOV IE,#92H
          MOV TCON,#01000001B
```

```
            MOV R2,#25D                 ;**********************
            CLR P3.5                    ;*** CONTROL RELAY ***
            MOV 1Fh,#20H                ;code enter key
            MOV 3EH,#31H
            MOV 4BH,#50H
            MOV 4CH,#8D
                  MOV ODH,#00H
            RET
     ;-------------------------------------;
          ;SUBROUTINE RESTE SYSTEM
RESET_SYS:MOV 3BH,#00H
            MOV 3CH,#00H
            MOV 34H,#00H
            MOV 35H,#00H
            MOV IE,#92H
            MOV 30H,#8D
            MOV 4BH,#50H
            MOV 4CH,#8D
            MOV P1,#0F0H
            CLR 0DH
            CLR 0EH
RESET_1:    MOV 0DH,#00H
            CLR 00H
            CLR 01H
            CLR 02H
            CLR 03H
            CLR 04H
            CLR 05H
            CLR 06H
            CLR 07H
```

```
CLR 08H
CLR 09H
CLR 0AH
CLR 0BH
CLR 0CH
CLR 10H
CLR 11H
CLR 12H
CLR 13H
CLR 14H
CLR 15H
CLR 16H
CLR 18H
CLR 19H
CLR 1AH
CLR 1BH
CLR 1CH
CLR 1DH
CLR 1EH
CLR 1FH
CLR 20H
CLR 21H
CLR 22H
CLR 23H
CLR 24H
CLR 25H
CLR 26H
CLR 27H
    CLR 28H
    CLR 29H
```

```
                CLR 30H
                CLR 31H
                CLR 32H
          RET
    ;-------------------------------------------------------;
          ;SUBROUTINE SAFETY P.A. OF RADIO
          ;IN R7
          ;OUT R7
SAFETY:   INC R7
          MOV 49H,#60D                    ;DELAY 1 MIN.
          CJNE R7,#200D,SAFE_1
          CLR P3.5
SAFE_2:   LCALL DELAY
          DJNZ 49H,SAFE_2
          MOV R7,#00H
          SETB P3.5
          LCALL DELAY
SAFE_1:   RET
    ;-------------------------------------------------------;
END
```

```
Program Serial1;

Uses Crt,Dos,Graph,Frame;

Const  Com1  = 0;

       Com2  = 1;

       Com3  = 2;

       Com4  = 3;


Var Regs                    : Registers;

    lead,ch,J,C,S,k         : Char;

    out,Initbyte,Com        : Byte;

    Portbase                : Word;

    i,Show,l                : Integer;

    Row1                    : Array[1..76] of Char;

    FILENAME                : String;

    F                       : Text;

    Home,KEYS,Check         : Char;

    Msg,Msg01               : String;

    ForG,BacG               : Integer;

    Mode,T                  : Byte;

    Color,Size              : Word;

    Mono,VgaMono,TisMode    : Boolean;

    D                       : LongInt;


Procedure InitSerial(Com,Initbyte:Byte);

   Begin

     Regs.ah := $00;

     Regs.dx := Com;

     Regs.al := Initbyte;

     Intr($14,Regs);

   End;
```

```
Function SendSerial(Data:Char):Byte;
   Begin
     Regs.ah := $01;
     Regs.dx := Com;
     Regs.al := Ord(Data);
     Intr($14,Regs);
     If (Regs.al And $80) = $80 Then SendSerial := $FF Else SendSerial := 0;
   End;


Procedure Setrts(Com:Byte);
Var OldVal : Byte;
Begin
   Case Com of
     Com1 : Begin
                 OldVal := Port[$3FC];
                 OldVal := OldVal Or $02;            { set pin RTS }
                 Port[$3FC] := OldVal ;
            End;
     Com2 :  Begin
                 OldVal := Port[$2FC];
                 OldVal := OldVal Or $02;            { set pin RTS }
                 Port[$2FC] := OldVal ;
            End;
     Com3 :  Begin
                 OldVal := Port[$3EC];
                 OldVal := OldVal Or $02;            { set pin RTS }
                 Port[$3EC] := OldVal ;
            End;
     Com4 :  Begin
```

```
                OldVal := Port[$2EC];
                OldVal := OldVal Or $02;            { set pin RTS }
                Port[$2EC] := OldVal ;
            End;
   End;
End;

Procedure  Clrts(Com:Byte);
Var OldVal : Byte;
Begin
   Case Com Of
     Com1 : Begin
                OldVal := Port[$3FC];
                OldVal := OldVal And $0FD;          { clear pin RTS }
                Port[$3FC] := OldVal ;
            End;
     Com2 :  Begin
                OldVal := Port[$2FC];
                OldVal := OldVal And $0FD;          { clear pin RTS }
                Port[$2FC] := OldVal ;
            End;
     Com3 :  Begin
                OldVal := Port[$3EC];
                OldVal := OldVal And $0FD;          { clear pin RTS }
                Port[$3EC] := OldVal ;
            End;
     Com4 :  Begin
                OldVal := Port[$2EC];
                OldVal := Oldval And $0FD;          { clear pin RTS }
                Port[$2EC] := OldVal ;
```

```
            End;
   End;
End;

Function ReadSerial : Integer;
Var Data  : Byte;
    Check : Byte;
Begin
    Regs.ah := $02;
    Regs.dx := Com;
    intr($14,Regs);
    Check := Regs.AH;
    Data  := Regs.AL;
    If (Check AND $80) = $80 Then ReadSerial := -1
    Else ReadSerial := Data;
End;

Procedure Receive1;
Begin
        Repeat
         Show := ReadSerial;
        Until (Chr(Show) = '~') Or KeyPressed;
        Check := '~'
End;

Procedure Receive;
Begin
       Show := ReadSerial;
       If Show = $0d Then Writeln;
       If Show = $06 Then Write(Chr(Show));
```

```
        If Chr(Show) In ['1','2','3','4','5','6','7','8','9','0','+','-',' ',
        'x','?','E','F']
        Then Write(Chr(Show));
    {   If (Show <> -1)  Then Write(Chr(Show));
        If (Show = $0d) Then
                            Begin
                              WriteLn;
                              Repeat
                                Show := ReadSerial;
                              Until Show = -1;
                            End;                    }
End;



Procedure Tranrece;
Begin
      ClrScr;
      TextBackground(15);
      ClrScr;
      Window(3,2,78,2);
      TextBackground(0);
      ClrScr;
      GotoXY(30,1);Write('Data To Transmit');
      Window(3,4,78,10);
      TextBackground(0);
      ClrScr;
      Window(3,12,78,12);
      TextBackground(0);
      ClrScr;
```

```
    GotoXY(30,1);Write('Data To Receive');
    Window(3,14,78,22);
    TextBackground(0);
    ClrScr;
    Check := '~';
Repeat
  Window(3,14,78,22);
  GotoXY(1,9);
  WriteLn;
  While Check <> '~' Do Receive1;
  Repeat
    Receive;
  Until KeyPressed;
{  GoToXY(1,9);ClrEol;  }
  Lead := Readkey;
  If Lead <> #27 Then
                  Begin
                    Window(3,4,78,10);
                    GotoXY(1,7);
                   Begin
                      Write(Lead);
                      Row1[1] := Lead;
                      i := 2;
                      l := 1;
                      D := 1;
                      Repeat
                          Lead := Readkey;
                          Write(Lead);
                          Row1[i] := Lead;
                          If Lead = #27 Then Begin
```

```
                                        window(1,1,80,25);
                                        Exit;
                                    End;
                    If Lead = #13 Then i := 76;
                    If Lead = #08 Then
                         Begin
                          i := i-1;
                          If i <= 0 Then i := 1;
                          ClrEol;
                          Row1[i] := ' ';
                          If Row1[1] = ' ' Then
                                 Begin
                                   ClrEol;
                                   D := 1;
                                   i := 0;
                                   Repeat
                                    D := D+1;
                                    If KeyPressed Then D:=9000;
                                   Until (D = 8000) Or (D = 9000);
                                 End;
                          l := l+1;
                         End;
                    i :=i+1;
               Until (i > 76) Or (D = 8000);
          While D <> 8000 Do
            Begin
               WriteLn;
               Setrts(Com); Delay(1000);
               For i:=1 To 76 Do
                 Begin
```

```
                                Out := SendSerial(Row1[i]);
                                If out = -1 Then  Begin
                                                    Lead := #27;
                                                    Write('Error');
                                                  End;
                                If Row1[i] = #13 Then i := 76;
                              End;
                                Delay(70);
                                Clrts(Com);
                                D := 8000;
                            End;
                        End;
                    End;
        window(1,1,80,25);
  Until Lead = #27;
End;

Procedure Upload;
Begin
      ClrScr;
      TextBackground(15);
      ClrScr;
      Window(3,2,78,2);
      TextBackground(0);
      ClrScr;
      GotoXY(30,1);Write('UPLOAD FILES');
      Window(3,4,78,22);
      TextBackground(0);
      ClrScr;
      Write('ENTER FILENAME TO UPLOAD: ');
```

```
ReadLn(Filename);
If Filename = '' then exit;
{$I-}
Assign(f, Filename);
Reset(f);
{$I+}
If IOResult = 0 Then
                    Begin
                       ClrScr;
                       Setrts(Com); Delay(1000);
                       Out := SendSerial('"');
                       Delay(2000);
                       While Not Eof(f) Do
                                    Begin
                                      Read(f,Home);
                                      Write(Home);
                                      Out := SendSerial(Home);
                                    End;
                       Out := SendSerial('~');
                       Delay(100);
                       Clrts(Com);
                       Close(f);
                       Writeln;
                       Writeln('UPLOAD ',FILENAME,' COMPLETED');
                    End
Else   Begin
          WriteLn('File Not Found !');
          Write('Press Any Key To Main Menu.');
       End;
Repeat Until KeyPressed;
```

```
End;


Procedure Download;
Begin
      ClrScr;
      TextBackground(15);
      ClrScr;
      Window(3,2,78,2);
      TextBackground(0);
      ClrScr;
      GotoXY(30,1);Write('DOWNLOAD FILES');
      Window(3,4,78,22);
      TextBackground(0);
      Clrscr;
      Write('ENTER FILENAME TO DOWNLOAD: ');
      ReadLn(Filename);
      If Filename = '' Then Exit;
      {$I-}
      Assign(f, Filename);
      Reset(f);
      {$I+}
      If IOResult = 0 Then
        Begin
          WriteLn('OLD FILE HAVE NAME ',FILENAME,' YOU WANT OVERWRITE:(Y/N)'
          Repeat
              Keys := ReadKey;
          Until Keys In ['y','Y','n','N'];
          Case Keys Of
            'Y','y' : Begin
                            ClrScr;
```

```
                        Repeat
                        Show := ReadSerial;
                        Until Chr(Show) ='"';
                        ReWrite(F);
                         Repeat
                           Show := ReadSerial;
                           Write(Chr(Show));
                           Write(F,Chr(Show));
                         Until  Chr(Show) ='~';
                         Clrts(Com);
                         Close(F);
                        WriteLn;
                        Write('DOWNLOAD',FILENAME,'COMPLETED');
                    End;
        'n','N' : Write('Press Any Key To Main Menu');
     End;
  End
Else   Begin
          ClrScr;
          Repeat
           Show := ReadSerial;
          Until Chr(Show) ='"';
          ReWrite(F);
           Repeat
             Show := ReadSerial;
             Write(F,Chr(Show));
             Write(Chr(Show));
           Until Chr(Show) = '~';
          Clrts(Com);
          Close(f);
```

```
                Write('DOWNLOAD',FILENAME,'COMPLETED');
              End;
      Repeat Until KeyPressed;
End;

Procedure Menu01;
Begin
  ClrScr;
  WriteLn;
  WriteLn('      1) 300,N,8,1');
  WriteLn('      2) 1200,N,8,1');
  WriteLn('      3) 2400,N,8,1');
  WriteLn('      4) 4800,N,8,1');
  WriteLn('      5) 9600,N,8,1');
  WriteLn;
  Write('    Select Baud Rate(1-5):');
  Repeat
    J := Readkey;
    If j = #27 then exit;
  Until J In ['1'..'5'];
  WriteLn(J);
    Case J Of
      '1' : Begin
              Initbyte:= $43;
              Msg := 'Baud Rate = 300,N,8,1';
              Window(3,24,40,24); TextBackground(0);
              ClrScr;  GotoXY( 10,1);Write(Msg);
            End;
      '2' : Begin
              Initbyte:= $83;
```

```
                Msg := 'Baud Rate = 1200,N,8,1';
                Window(3,24,40,24); TextBackground(0);
                ClrScr;  GotoXY( 10,1);Write(Msg);
              End;
        '3' : Begin
                Initbyte:= $A3;
                Msg := 'Baud Rate = 2400,N,8,1';
                Window(3,24,40,24); TextBackground(0);
                ClrScr;  GotoXY( 10,1);Write(Msg);
              End;
        '4' : Begin
                Initbyte:= $C3;
                Msg := 'Baud Rate = 4800,N,8,1';
                Window(3,24,40,24); TextBackground(0);
                ClrScr;  GotoXY( 10,1);Write(Msg);
              End;
        '5' : Begin
                Initbyte:= $E3;
                Msg := 'Baud Rate = 9600,N,8,1';
                Window(3,24,40,24); TextBackground(0);
                ClrScr;  GotoXY( 10,1);Write(Msg);
              End;
      End;
  InitSerial(Com,Initbyte);
End;

Procedure Menu02;
Begin
  ClrScr;
  WriteLn;
```

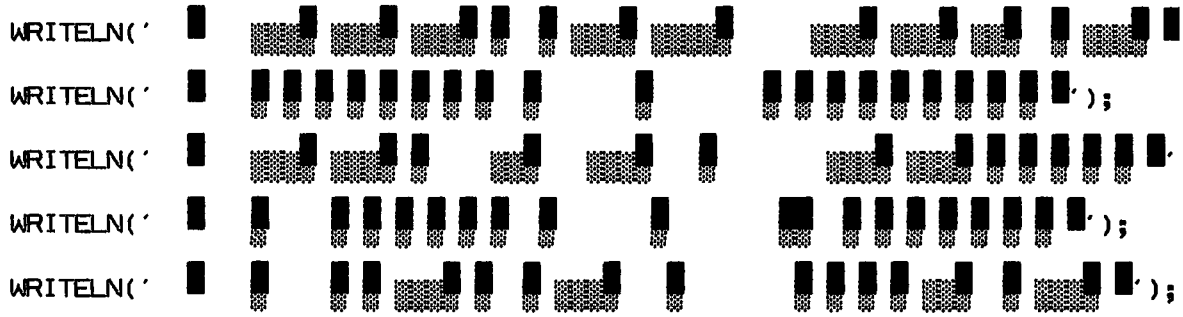
```
WriteLn('        1)Com1');
WriteLn('        2)Com2');
WriteLn('        3)Com3');
WriteLn('        4)Com4');
WriteLn;
Write('     Select Port(1-4) :');
Repeat
  C := Readkey;
  If C = #27 Then Exit;
Until C In ['1'..'4'];
WriteLn(C);
  Case C Of
     '1' :  Begin
              Com:= Com1;
              Msg01:='Serial Port = Com1';
              Window(40,24,78,24); TextBackground(0);
              ClrScr; GotoXY(5,1); Write(Msg01);
            End;
     '2' :  Begin
              Com:= Com2;
              Msg01:='Serial Port = Com2';
              Window(40,24,78,24); TextBackground(0);
              ClrScr; GotoXY(5,1); Write(Msg01);
            End;
     '3' :  Begin
              Com:= Com3;
              Msg01:='Serial Port = Com3';
              Window(40,24,78,24); TextBackground(0);
              ClrScr; GotoXY(5,1); Write(Msg01);
            End;
```

```
        '4' :  Begin
                 Com:= Com4;
                 Msg01:='Serial Port = Com4';
                 Window(40,24,78,24); TextBackground(0);
                 ClrScr; GotoXY(5,1); Write(Msg01);
               End;
    End;
  InitSerial(Com,Initbyte);
End;


Procedure Menu03;
Begin
  ClrScr;
  WriteLn;
  WriteLn;
  WriteLn('       1)Trans/Recei');
  WriteLn('       2)Upload File(Text)');
  WriteLn('       3)Download File(Text)');
  WriteLn;
  Write('     Select Choice :');
  Repeat
   s := Readkey;
   If S = #27 Then Exit;
  Until S In ['1'..'3'];
  WriteLn(s);
    Case S Of
       '1': TranRece;
       '2': Upload;
       '3': Download;
    End;
```

```
End;


Procedure Menu00;
Begin
  ClrScr;
  TextBackground(15);
  ClrScr;
  Window(3,2,78,2);
  TextBackground(0);
  ClrScr;
  GotoXY(33,1);Write('MAIN MENU');
  Window(3,24,39,24);
  TextBackground(0);
  ClrScr;
  GotoXY(10,1);Write(Msg);
  Window(40,24,78,24);
  TextBackground(0);
  ClrScr;
  GotoXY(5,1);Write(Msg01);
  Window(23,6,55,14);
  TextBackground(0);
  ClrScr;
  WriteLn;
  WriteLn('     1)Select Baud Rate');
  WriteLn('     2)Select Port');
  WriteLn('     3)Select Mode');
  WriteLn('     4)Exit');
  WriteLn;
  Write('     Select Number(1-4) :');
  Repeat
```

```
    k := Readkey;
    If k = #27 Then k := '4';
  Until k In ['1'..'4'];
  WriteLn(k);
    Case k Of
       '1' : Begin Menu01;window(1,1,80,25);clrscr;end;
       '2' : Menu02;
       '3' : Menu03 ;
       '4' : Exit;
    End;
End;

                {********** Title ***********}

Procedure Tit;

Procedure CursorOff;
Var regs  : registers;
Begin
  regs.ah := 1;
  regs.cx := $1600;
  intr($10,regs);
End;

Procedure CursorOn;
Var regs  : registers;
Begin
  regs.ah := 1;
  regs.cx := $0607;
  intr($10,regs);
```

```
End;


BEGIN
   CURSOROFF;
   CLRSCR;
   WRITELN;
   WRITELN('  ');

                 

   WRITELN('
```



```
   WRITELN('
```

```
    WINDOW(20,24,60,24);
    REPEAT
       WRITE(' * *  PRESS ANY KEYS TO MAIN MENU  * *');
       DELAY(400);
       CLRSCR;
       DELAY(300);
    UNTIL  KEYPRESSED;
    WINDOW(1,1,80,25);
    CURSORON;
END;

Procedure Opengraph;
Var GrDriver,GrMode : Integer;
Begin
  Mono := False; VgaMono := False;
  Color := 3;
  DetectGraph(GrDriver,GrMode);
   Case GrDriver Of
     5 : VgaMono := True;
     7 : Begin Mono := True; Color := 1; End;
     9 : GrMode := 2;
   End;
     InitGraph(GrDriver,GrMode,'');
End;

Procedure LoadFont(FontName :String; Size:Byte);
Begin
     T := InstallUserFont(Fontname);
     SetTextStyle(T,0,Size);
End;
```

```
Procedure OutTextsha(x,y : Integer; St : String);
Begin
     SetColor(0);
     OutTextxy(x+3,y+4,St);
     SetColor(15);
     OutTextxy(x,y,St);
End;

Procedure OutTxt(x,y : Integer;Size : Byte;Color : Word; Msg : String);
Begin
    SetColor(Color);
    SetTextStyle(0,0,Size);
    OutTextXY(x,y,Msg);
End;

Procedure LoadPic(x,y:integer);
Var Size : Word;
     P,T : Pointer;
      FP : File;
Begin
     Assign(FP,'KMITL.VGA');
     {$I-}
     Reset(FP,1);
     {$I+}
     If IoResult <> 0 Then
     Begin
          Closegraph;
          Writeln('File KMITL.VGA not found');
          Halt(0);
```

```
      End;
      Bar(x-1,y-1,x+177,y+171);
      Size := FileSize(FP);
      GetMem(P,Size);
      BlockRead(FP,P^,Size);
      PutImage(x,y,P^,Normalput);
      Close(FP);
      FreeMem(P,Size);
End;

Procedure Frame0(x1,y1,x2,y2 : Integer; BackGr : Word);
 Begin
  SetFillStyle(1,BackGr);
  Bar(x1,y1,x2,y2);
  Setcolor(15);
  Rectangle(x1,y1,x2,y2);
End;

Procedure Frame1(x1,y1,x2,y2 : Integer; BackGr : Word);
Begin
  SetFillStyle(1,BackGr);
  Bar(x1,y1,x2-10,y2-10);
  Setcolor(0);
  Rectangle(x1+5,y1+5,x2-15,y2-15);
  SetFillStyle(1,0);
  Bar(x2-10,y1+10,x2,y2);
  Bar(x1+10,y2-10,x2,y2);
  Setcolor(15);
  Rectangle(x1,y1,x2-10,y2-10);
End;
```

```
Procedure Frame2(x1,y1,x2,y2 : Integer; BackGr : Word);
Begin
  SetFillStyle(1,BackGr);
  Bar(x1,y1,x2-10,y2-10);
  SetFillStyle(1,0);
  Bar(x2-10,y1+10,x2,y2);
  Bar(x1+10,y2-10,x2,y2);
  Setcolor(15);
  Rectangle(x1,y1,x2-10,y2-10);
End;

Procedure Frame3(x1,y1,x2,y2 : Integer; BackGr : Word);
Begin
  SetFillStyle(1,BackGr);
  Bar(x1+1,y1+1,x2-1,y2-1);
End;

Procedure Title;
Begin
 Frame0(0,0,GetMaxX,GetMaxY,7);
 LoadPic(30,40);
 LoadFont('Trip',6);
 OutTextsha(250,65,'PACKET RADIO');
 LoadFont('Trip',4);
 OutTextsha(330,130,'Version #1.0');
 LoadFont('Trip',1);
 OutTextsha(140,220,'By..');
 OutTextsha(190,250,'Mr.Boonchai Photchanasomsaman 33504017');
 OutTextsha(190,280,'Mr.Somnuk  Wisadpanichkij       33504037');
```

```
 OutTextsha(200,310,'Department Of Applied Physics ');
 OutTextsha(240,340,'Faculty Of Science');
 OutTextsha(100,370,'King Mongkut'#39's Institute Of Technology Ladkrabang');
 Repeat Until KeyPressed;
End;

{main}

Begin
     ClrScr;
{     OpenGraph;
     If (Mono = True) And (Color = 1)
     Then Begin
           CloseGraph;
           Tit;
          End
     Else Begin
           Title;
           CloseGraph;
          End;}
     Msg  := 'Baud Rate = 300,N,8,1';
     Initbyte:= $43;
     Msg01:= 'Serial Port = Com1';
     Com:= Com1;
     InitSerial(Com,Initbyte);
     Repeat
       Menu00;
     Until k = '4';
     Window(1,1,80,25);
     ClrScr;
```

```
    WriteLn('Thanks For Used Program (TTR15)');
    WriteLn('From KMIT''L Applied Physics Solidy #9 (1993-1994)');
    WriteLn('');
End.
```

MAXIM

+5V Powered RS-232 Drivers/Receivers

General Description

Maxim's family of line drivers/receivers are intended for all RS-232 and V.28/V.24 communications interfaces, and in particular, for those applications where ±12V is not available. The MAX230, MAX236, MAX240 and MAX241 are particularly useful in battery powered systems since their low power shutdown mode reduces power dissipation to less than 5μW. The MAX233 and MAX235 use no external components and are recommended for applications where printed circuit board space is critical.

All members of the family except the MAX231 and MAX239 need only a single +5V supply for operation. The RS-232 drivers/receivers have on-board charge pump voltage converters which convert the +5V input power to the ±10V needed to generate the RS-232 output levels. The MAX231 and MAX239, designed to operate from +5V and +12V, contain a +12V to -12V charge pump voltage converter.

Since nearly all RS-232 applications need both line drivers and receivers, the family includes both receivers and drivers in one package. The wide variety of RS-232 applications require differing numbers of drivers and receivers. Maxim offers a wide selection of RS-232 driver/receiver combinations in order to minimize the package count (see table below).

Both the receivers and the line drivers (transmitters) meet all EIA RS-232C and CCITT V.28 specifications.

Features

- ♦ **Operates from Single 5V Power Supply (+5V and +12V — MAX231 and MAX239)**
- ♦ **Meets All RS-232C and V.28 Specifications**
- ♦ **Multiple Drivers and Receivers**
- ♦ **Onboard DC-DC Converters**
- ♦ **±9V Output Swing with +5V Supply**
- ♦ **Low Power Shutdown — <1μA (typ)**
- ♦ **3-State TTL/CMOS Receiver Outputs**
- ♦ **±30V Receiver Input Levels**

Applications

Computers
Peripherals
Modems
Printers
Instruments

Selection Table

Part Number	Power Supply Voltage	No. of RS-232 Drivers	No. of RS-232 Receivers	External Components	Low Power Shutdown /TTL 3-State	No. of Pins
MAX230	+5V	5	0	4 capacitors	Yes/No	20
MAX231	+5V and +7.5V to 13.2V	2	2	2 capacitors	No/No	14
MAX232	+5V	2	2	4 capacitors	No/No	16
MAX233	+5V	2	2	None	No/No	20
MAX234	+5V	4	0	4 capacitors	No/No	16
MAX235	+5V	5	5	None	Yes/Yes	24
MAX236	+5V	4	3	4 capacitors	Yes/Yes	24
MAX237	+5V	5	3	4 capacitors	No/No	24
MAX238	+5V	4	4	4 capacitors	No/No	24
MAX239	+5V and +7.5V to 13.2V	3	5	2 capacitors	No/Yes	24
MAX240	+5V	5	5	4 capacitors	Yes/Yes	44 (Flatpak)
MAX241	+5V	4	5	4 capacitors	Yes/Yes	28 (Small Outline)

*Patent Pending

+5V Powered RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS

V_{CC} ..... -0.3V to +6V
V^+ ..... (V_{CC} - 0.3V) to +14V
V^- ..... +0.3V to -14V
Input Voltages
T_{IN} ..... -0.3 to (V_{CC} + 0.3V)
R_{IN} ..... ±30V
Output Voltages
T_{OUT} ..... (V^+ + 0.3V) to (V^- - 0.3V)
R_{OUT} ..... -0.3V to (V_{CC} + 0.3V)

Short Circuit Duration
T_{OUT} ..... continuous
Power Dissipation
CERDIP 675mW
(derate 9.5mW/°C above +70°C)
Plastic DIP 375mW
(derate 7mW/°C above +70°C)
Small Outline (SO) 375mW
(derate 7mW/°C above +70°C)
Lead Temperature (soldering 10 seconds) ..... +300°C
Storage Temperature ..... -65°C to +160°C

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

ELECTRICAL CHARACTERISTICS

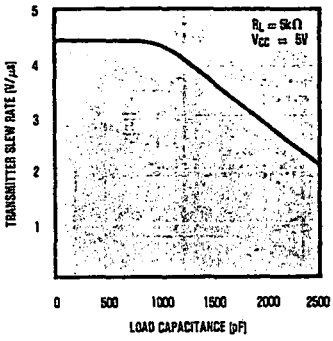
(MAX232, 234, 236, 237, 238, 240, 241 V_{CC} = 5V ± 10%; MAX233, 235 V_{CC} = 5V ± 5%; MAX231, 239 V_{CC} = 5V ± 10%, V^+ = 7.5V to 13.2V; T_A = Operating Temperature Range, Figures 3-14, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN.	TYP.	MAX.	UNITS
Output Voltage Swing	All Transmitter Outputs loaded with 3kΩ to Ground		±5	±9		V
V_{CC} Power Supply Current	No load, T_A = +25°C			5	10	mA
	MAX231, MAX239			0.4	1	
V^+ Power Supply Current	No load, MAX231 and MAX239 only	MAX231		1.8	5	mA
		MAX239		5	15	
Shutdown Supply Current	Figure 1, T_A = +25°C			1	10	µA
Input Logic Threshold Low	T_{IN}, $\overline{EN}$, Shutdown				0.8	V
Input Logic Threshold High	T_{IN}		2.0			V
	$\overline{EN}$, Shutdown		2.4			
Logic Pullup Current	T_{IN} = 0V			15	200	µA
RS-232 Input Voltage Operating Range			-30		+30	V
RS-232 Input Threshold Low	V_{CC} = 5V, T_A = +25°C (MAX231, 239 V^+ = 0V)		0.8	1.2		V
RS-232 Input Threshold High	V_{CC} = 5V, T_A = +25°C (MAX231, 239 V^+ = 12V)			1.7	2.4	V
RS-232 Input Hysteresis	V_{CC} = 5V		0.2	0.5	1.0	V
RS-232 Input Resistance	T_A = +25°C, V_{CC} = 5V		3	5	7	kΩ
TTL/CMOS Output Voltage Low	I_{OUT} = 1.6mA (MAX231-233, I_{OUT} = 3.2mA)				0.4	V
TTL/CMOS Output Voltage High	I_{OUT} = -1.0mA		3.5			V
TTL/CMOS Output Leakage Current	$\overline{EN}$ = V_{CC}, 0V ≤ R_{OUT} ≤ V_{CC}			0.05	±10	µA
Output Enable Time (Figure 2)	MAX235, MAX236, MAX239, MAX240, 241			400		ns
Output Disable Time (Figure 2)	MAX235, MAX236, MAX239, MAX240, 241			250		ns
Propagation Delay	RS-232 to TTL			0.5		µs
Instantaneous Slew Rate	C_L = 10pF, R_L = 3-7kΩ T_A = +25°C (Note 1)				30	V/µs
Transition Region Slew Rate	R_L = 3kΩ, C_L = 2500pF Measured from +3V to -3V or -3V to +3V			3		V/µs
Output Resistance	V_{CC} = V^+ = V^- = 0V, V_{OUT} = ±2V		300			Ω
RS-232 Output Short Circuit Current				±10		mA

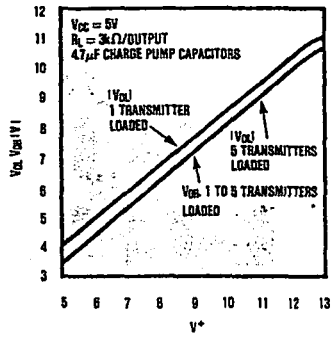
Note 1: Sample tested.

Typical Operating Characteristics

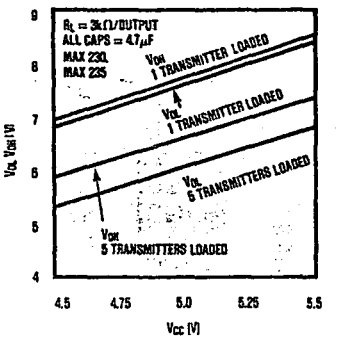
TRANSMITTER SLEW RATE vs. LOAD CAPACITANCE



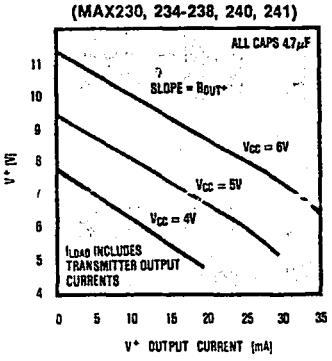
MAX239 TRANSMITTER OUTPUT VOLTAGE vs. V+ VOLTAGE



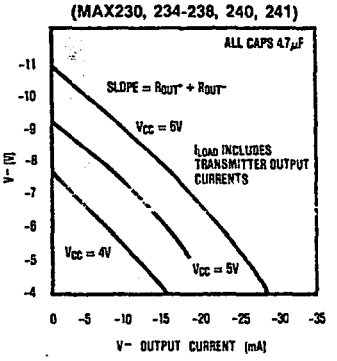
TRANSMITTER OUTPUT VOLTAGE vs. V_{CC} VOLTAGE



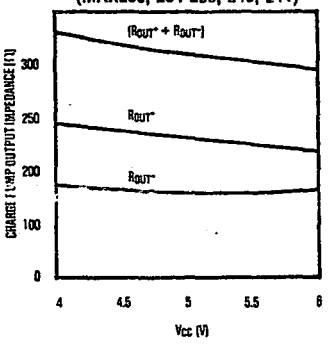
V+ SUPPLY VOLTAGE vs. LOAD CURRENT (MAX230, 234-238, 240, 241)



V- SUPPLY VOLTAGE vs. LOAD CURRENT (MAX230, 234-238, 240, 241)



CHARGE PUMP OUTPUT IMPEDANCE vs. V_{CC} (MAX230, 234-238, 240, 241)



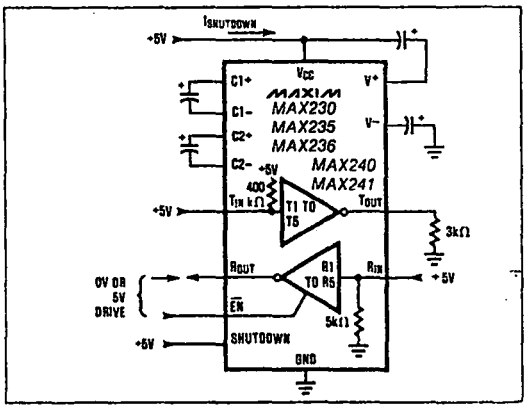


Figure 1. Shutdown Current Test Circuit

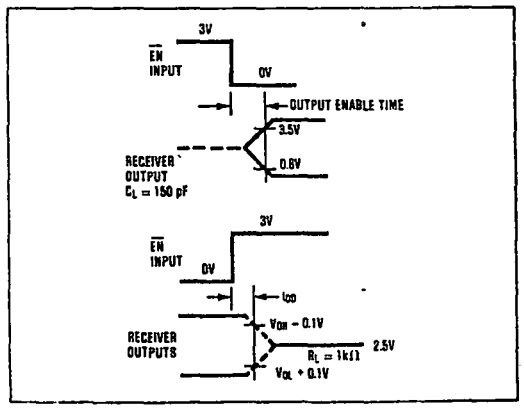
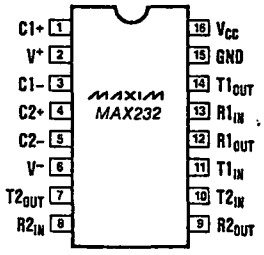


Figure 2. Receiver Output Enable and Disable Timing



16 Lead Small Outline
also available.

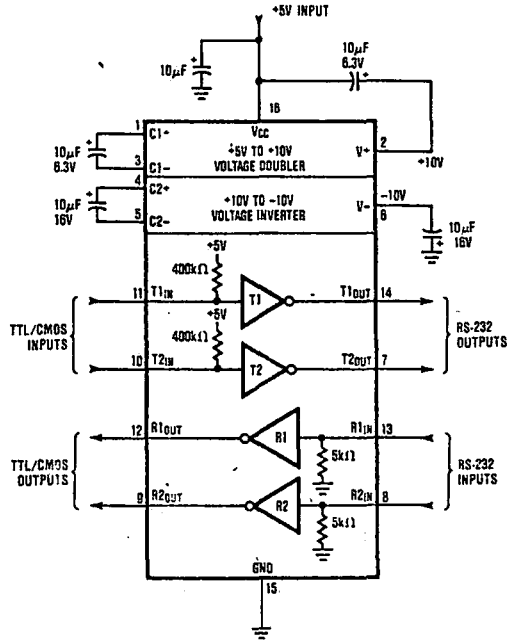
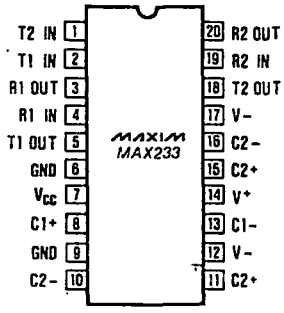


Figure 5. MAX232 Typical Operating Circuit



Small Outline Not Available

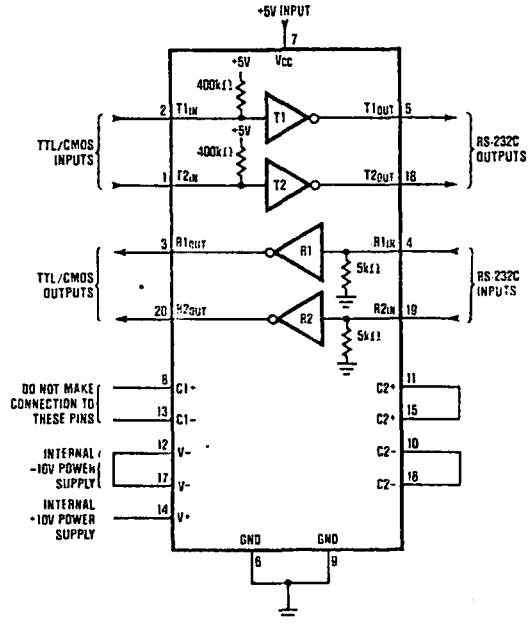


Figure 6. MAX233 Typical Operating Circuit

+5V Powered RS-232 Drivers/Receivers

Typical Applications

Figures 3 through 14 show typical applications. The capacitor values are non-critical. Reducing the capacitors C1 and C2 to 1μF will slightly increase the impedance of the charge pump, lowering the RS-232 driver output voltages by about 100mV. Lower values of C3 and C4 increase the ripple on the V^+ and V^- outputs.

If the power supply input to the device has a very fast rate-of-rise (as would occur if a PCB were to be plugged into a card cage with power already on), use the simple RC filter shown in Figure 15. This bypass network is not needed if the V_{CC} rate-of-rise is below 1V/μs.

All receivers and drivers are inverting. The $\overline{EN}$able control of the MAX235, MAX236, MAX239, MAX240 and MAX241 enables the receiver TTL/CMOS outputs when it is at a low level, and places the TTL/CMOS outputs of the receivers into a high impedance state when it is a high level.

When the Shutdown control of the MAX230, MAX235, MAX236, MAX240 and MAX241 is at a logic 1 the charge pump is turned off, the receiver outputs are put into the high impedance state, V^+ is pulled down to V_{CC}, V^- is pulled up to ground, and the transmitter outputs are disabled. The supply current drops to less than 10μA.

Detailed Description

The following sections provide supplementary information for those designers with non-standard applications and for those with interest in the internal operation of the devices.

The devices consist of 3 sections: the transmitters, the receivers, and the charge pump DC-DC voltage converter.

+5V to ±10V Dual Charge Pump Voltage Converter

All but the MAX231 and MAX239 convert +5V to ±10V. This conversion is performed by two charge pump voltage converters. The first uses capacitor C1 to double the +5V to +10V, storing the +10V on the V^+ output filter capacitor, C3. The second charge pump voltage converter uses capacitor C2 to invert the +10V to -10V, storing the -10V on the V^- output filter capacitor, C4. The equivalent circuit of the charge pump section is shown in Figure 16.

A small amount of power may be drawn from the +10V (V^+) and -10V (V^-) outputs to power external circuitry. The typical characteristics graphs show the typical output voltage vs. load current characteristics.

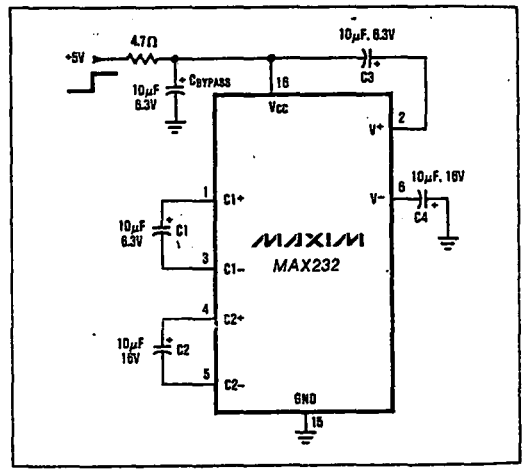


Figure 15. Protection from High $\frac{dV}{dT}$

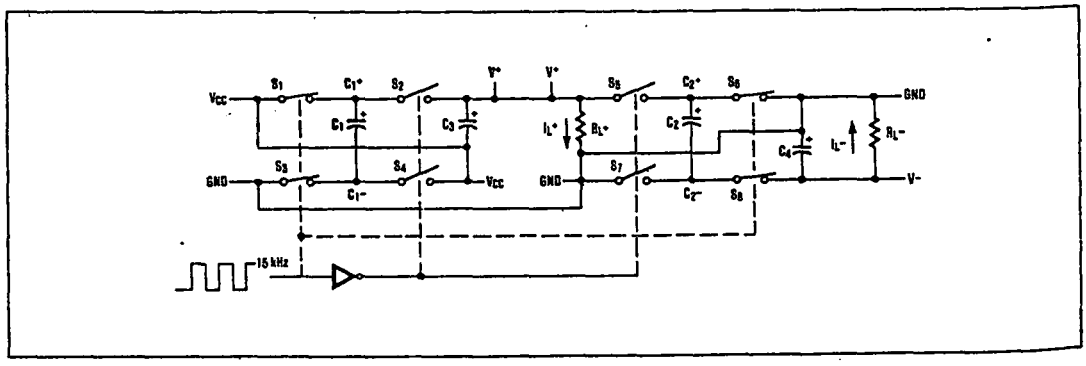


Figure 16. Charge Pump Diagram.

For applications needing only the +5V to ±10V charge pump voltage converter, the MAX680 is available.

The capacitor values for C1 through C4 are non-critical. At the 15kHz typical switching frequency of the voltage converter, a 1μF capacitor has approximately 10Ω impedance, and replacing the 4.7μF and 10μF capacitors shown in the typical applications with 1μF for C1 and C2 will increase the output impedance of the V^+ output by about 10Ω and the output impedance of V^- by about 20Ω. Lowering the value of C3 and C4 increases the ripple on the V^+ and V^- outputs. Where operation to the upper temperature limit is not required, or V_{CC} will not go below 4.75V, C1 and C2 can be 1μF, and C3 and C4 can be 1μF per output channel (1μF if one transmitter is used, 5μF if five transmitters are used).

There are parasitic diodes which become forward biased if V^+ goes below V_{CC} or V^- goes above ground. When in the shutdown mode (MAX230, MAX235, MAX236, MAX240 and MAX241 only), V^+ is internally connected to V_{CC} by a 1kΩ pulldown, and V^- is internally connected to ground via a 1kΩ pullup.

The MAX233 and MAX235 contain all charge pump components, including the capacitors, and operate with NO external components.

The MAX231 and MAX239 include only the V^+ to V^- charge pump, and are intended for applications which have a +5V supply and either a +12V ± 10% supply or a 7.5V to 13.2V battery voltage. When operating with V^+ greater than 8.0V, both capacitors can be 1μF.

Driver (Transmitter) Section

The transmitters or line drivers are inverting level translators which convert the CMOS or TTL input levels to RS-232 or V.28 voltage levels. With +5V V_{CC}, the typical output voltage swing is ±9V when loaded with the nominal 5kΩ input resistance of an RS-232 receiver. The output swing is guaranteed to meet the RS-232/V.28 specification of ±5V minimum output swing under the worst case conditions of all transmitters driving the 3kΩ minimum allowable load impedance, V_{CC} = 4.5V, and maximum operating ambient temperature. The open circuit output voltage swing is from (V^+ - 0.6V) to V^-.

The input thresholds are both CMOS and TTL compatible, with a logic threshold of about 25% of V_{CC}. The inputs of unused drivers sections can be left unconnected; an internal 400kΩ input pullup resistor to V_{CC} will pull the inputs high, forcing the unused transmitter outputs low. The input pullup resistors source about 12μA, and the driver inputs should be driven high or open circuited to minimize power supply current in the shutdown mode.

When in the low power shutdown mode, the driver outputs are turned off and their leakage current is less than 1μA with the driver output pulled to ground. The driver output leakage remains less than 1μA, even if the transmitter output is backdriven between 0V and (V_{CC} + 6V). Below -0.5V the transmitter is diode clamped to ground with 1kΩ series impedance. The transmitter is also zener clamped to approximately V_{CC} + 6V, with a series impedance of 1kΩ. As required by the RS232 and V.28, the slew rate is limited to less than 30V/μs. This limits the maximum usable baud rate to 19,200 baud.

Receiver Section

All but the MAX230 and MAX234 contain RS-232/V.28 receivers. These receivers convert the ±5V to ±15V RS-232 signals to 5V TTL/CMOS outputs. Since the RS-232C/V.28 specifications define a voltage level greater than +3V as a 0, the receivers are inverting. Maxim has set the guaranteed input thresholds of the receivers to 0.8V minimum and 2.4V maximum, which are significantly tighter than the -3.0V minimum and +3.0V maximum required by the RS-232 and V.28 specifications. This allows the receivers to respond both to RS-232/V.28 levels and TTL level inputs. The receivers are protected against input overvoltage up to ±30V.

The 0.8V guaranteed lower threshold is important to ensure that the receivers will have a logic 1 output if the receiver is not being driven because the equipment containing the line driver is turned off or disconnected, or if the connecting cable has an open circuit or short circuit. In other words, the receiver implements Type 1 interpretation of fault conditions (§7 of V.28, §2.5 of RS-232C). While a 0V or even a -3V receiver threshold would be acceptable for the data lines, these lower thresholds would not give proper indication on the control lines such as DTR and DSR. The receivers, on the other hand, have a full 0.8V noise margin for detecting the power-down or cable-disconnected states.

The receivers have a hysteresis of approximately 0.5V, with a minimum guaranteed hysteresis of 200mV. This aids in obtaining clean output transitions, even with slow rise and fall time input signals with moderate amounts of noise and ringing. The propagation delays of the receivers are 350ns for negative-going input signals, and 650ns for positive-going input signals (see Typical Characteristics graphs).

The MAX239 has a receiver 3-state control line, and the MAX235, MAX236, MAX240 and MAX241 have both a receiver 3-state control line and a low power shutdown control. The receiver TTL/CMOS outputs are in a high impedance 3-state mode whenever the 3-state $\overline{\text{EN}}$able line is high, and are also high impedance whenever the Shutdown control line is high.

+5V Powered RS-232 Drivers/Receivers

Review of EIA Standard RS-232-C and CCITT Recommendations V.28 and V.24

The most common serial interface between electronic equipment is the "RS232" interface. This serial interface has been found to be particularly useful for the interface between units made by different manufacturers since the voltage levels are defined by the EIA Standard RS-232-C and CCITT Recommendation V.28. The RS-232 specification also contains signal circuit definitions and connector pin assignments, while CCITT circuit definitions are contained in a separate document, Recommendation V.24. Originally intended to interface modems to computers and terminals, these standards have many signals which are not used for computer-to-computer or computer-to-peripheral communication.

Serial interfaces can be used with a variety of transmission formats. The most popular by far is the asynchronous format, generally at one of the standard baud rates of 300, 600, 1200, etc. The maximum recommended baud rate for RS-232 and V.28 is 20,000 baud, and the fastest commonly used baud rate is 19,200 baud. Asynchronous serial links use a variety of combinations of the number of data bits, what type (if any) of parity bit, and the number of stop bits. A typical combination is 7 data bits, even parity, and 1 stop bit

RS232/V.28 physical links are also suitable for synchronous transmission protocols. These higher level protocols often use the standard RS-232C/V.28 voltage levels. Note that one type of physical link (such as RS-232/V.28 voltage levels) can be used for a variety of higher level protocols. Table 2 summarizes the voltage levels and other requirements of V.28 and RS-232.

Comparison of RS-232C/V.28 with other Standards

The other two most common serial interface specifications are the EIA RS423 and RS422/RS485 (CCITT recommendations V.10 and V.11). While the RS-232 or V.28/V.24 interface is the most common interface for communication between equipment made by different manufacturers, the RS423/V.10 interface and RS422/V.11 interfaces can operate at higher baud rates. In addition, the RS485 interface can be used for low cost local area networks.

The RS423 and V.10 interfaces are unbalanced or "single-ended" interfaces which use a differential receiver. This standard is intended for data signaling rates up to 100 kbit/s (100 kilobaud). It achieves this higher baud rate through more precise requirements on the waveshape of the transmitters and through the use of differential receivers to compensate for ground potential variations beween the transmitting and receiving equipment. With certain limitations, this interface is compatible with RS-232 and V.28. The limitations are:

1) less than 20,000 baud rate,
2) maximum cable lengths determined by RS-232 performance,
3) RS423/V.10 DTE and DCE signal return paths must be connected to the the RS232/V.28 signal ground,
4) the RS-232 transmitter output voltages must be limited to ±12V, or additional protection must be provided for the RS423/V.10 receivers, and
5) not all RS232/V.28 receivers will show proper power-off detection of V.10 transmitter outputs.

Maxim's MAX230 and MAX232-MAX238, MAX240 and MAX241 meet restrictions 4 and 5 over the entire range of recommended operating conditions. The MAX231 and MAX239 meet restrictions 4 and 5 provided that the V^+ voltage is 12.5V or less.

The RS422, RS485, and V.11 interfaces are balanced double-current interchanges suitable for baud rates up to 10 Mbit/s. These interfaces are not compatible with RS-232 or V.28 voltage levels.

Application Hints

Operation at High Baud Rates

V.28 states that "the time required for the signal to pass through the transition region during a change in state shall not exceed 1 millisecond or 3 percent of the nominal element period on the interchange circuit, whichever is less." RS-232C allows the transition time to be 4 percent of the duration of a signal element. At 19,200 baud, the "nominal element period" is approximately 50μs, of which 3 percent is 1.5μs. Since the transition region is from -3V to +3V, this means the V.28 slew rate would ideally be faster than 6V/1.5μs = 4V/μs at 19.2 kbaud and 2V/μs at 9600 baud. The RS-232 requirement is equivalent to 3V/μs at 19.2 kbaud, 1.5V/μs at 9600 baud, etc. The slew rate of the MAX230 series devices is about 3V/μs with the maximum recommended load of 2500pF. In practice, the effect of less than optimum slew rate is a distortion of the recovered data, where the 1's and 0's no longer have equal width. This distortion generally has negligible effect and the devices can be reliably used for 19.2 kbaud serial links when the cable capacitance is kept below 2500pF. With very low capacitance loading, the MAX230 and MAX234-239, MAX240 and MAX241 may even be used at 38.4 kbaud, since the typical slew rate is 5V/μs when loaded with 500pF in parallel with 5kΩ. Under no circumstance will the

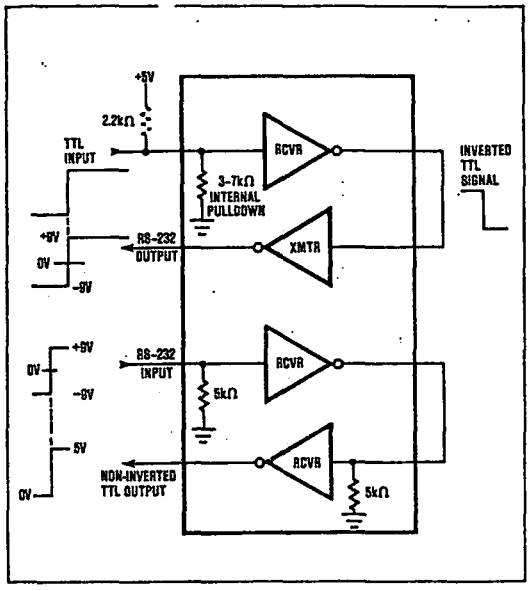


Figure 17. Non-inverting RS-232 Transmitters and Receivers.

slew rate exceed the RS-232/V.28 maximum spec of 30V/μs and, unlike the 1488 driver, no external compensation capacitors are needed under any load condition.

Driving Long Cables

The RS-232 standard states that "The use of short cables (each less than approximately 50 feet or 15 meters) is recommended; however, longer cables are permissible, provided that the load capacitance . . . does not exceed 2500pF."

Baud rate and cable length can be traded off: use lower baud rates for long cables, use short cables if high baud rates are desired. For both long cables and high baud rates, use RS422/V.11. The maximum cable length for a given baud rate is determined by several factors, including the capacitance per meter of cable, the slew rate of the driver under high capacitive loading, the receiver threshold and hysteresis, and the acceptable bit error rate. The receivers have 0.5V of hysteresis, and the drivers are designed such that the slew rate reduction caused by capacitive loading is minimized (see Typical Characteristics).

Non-Inverting Drivers and Receivers

Occasionally, a non-inverting driver or receiver is needed instead of the inverting drivers and receivers of the family. Simply use one of the receivers as a TTL/CMOS inverter to get the desired operation (Figure 17). If the logic output driving the receiver input has less than 1mA of output source capability, then add the 2.2kΩ pullup resistor.

The receiver TTL outputs can directly drive the input of another receiver to form a non-inverting RS-232 receiver.

Protection for Shorts to ±15V Supplies

All driver outputs except on the MAX231, MAX232 and MAX233 are protected against short circuits to ±15V, which is the maximum allowable loaded output voltage of an RS-232/V.28 transmitter. The MAX231, MAX232, and MAX233 can be protected against short circuits to ±15V power supplies by the addition of a series 220Ω resistor in each output. This protection is not needed to protect against short circuits to most RS-232 transmitters such as the 1488, since they have an internal short circuit current limit of 12mA.

The power dissipation of the MAX230 and MAX234-MAX239, MAX240 and MAX241 is about 200mW with all transmitters shorted to ±15V.

Isolated RS-232 Interfaces

RS-232 and V.28 specifications require a common ground connection between the two units communicating via the RS-232/V.28 interface. In some cases, there may be large differences in ground potential between the two units, and in other cases it may be desired to avoid ground loop currents by isolating the two grounds. In other cases, a computer or control system must be protected against accidental connection of the RS-232/V.28 signal lines to 110/220VAC power lines. Figure 18 shows a circuit with this isolation. The power for the MAX233 is generated by a MAX635 DC-DC converter. When the MAX635 regulates point "A" to -5V, the isolated output at point "B" will be semi-regulated to +5V. The two optocouplers maintain isolation between the system ground and the RS-232 ground while transferring the data across the isolation barrier. While this circuit will not withstand 110VAC between the RS-232 ground and either the receiver or transmitter lines, the voltage difference between the two grounds is only limited by the optocoupler and DC-DC converter transformer breakdown ratings.

+5V Powered RS-232 Drivers/Receivers

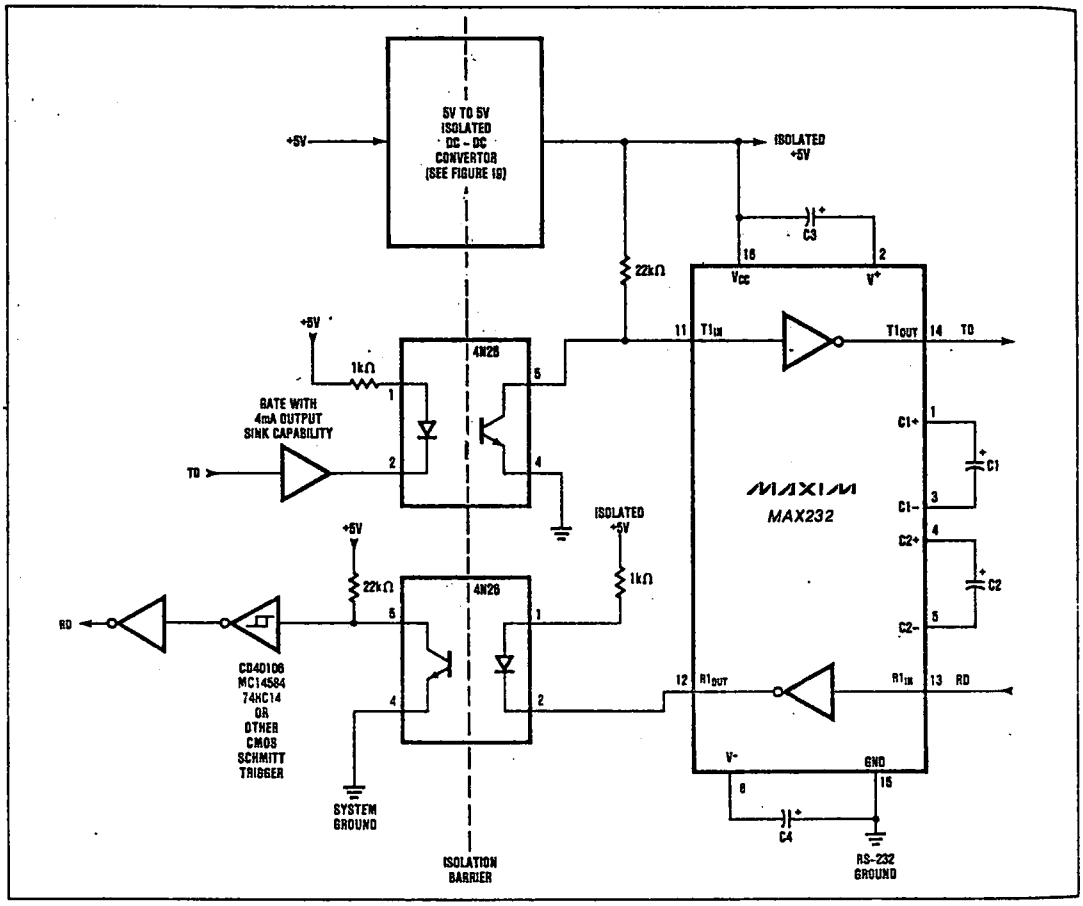


Figure 18. Optically isolated RS-232 Interface.

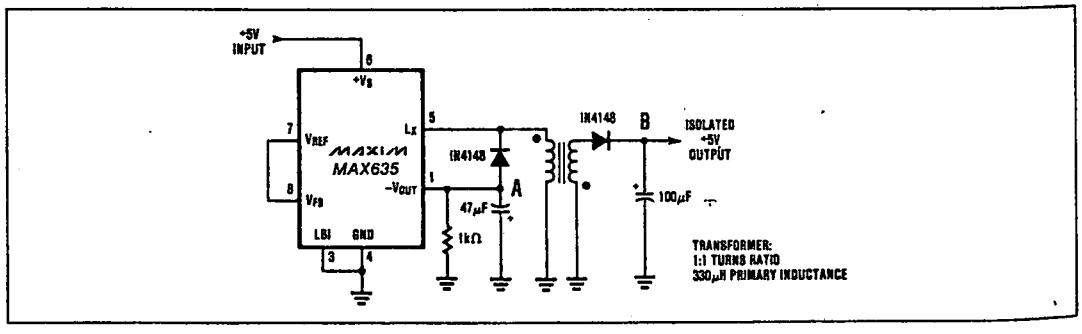


Figure 19. +5V Isolated Power Supply For Optically Isolated RS-232 Interface.

Ordering Information

PART	TEMP. RANGE	PACKAGE
MAX230		**0.3" Wide**
MAX230CPP	0°C to +70°C	20 Lead Plastic DIP
MAX230CWP	0°C to +70°C	20 Lead Wide S.O.
MAX230C/D	0°C to +70°C	Dice
MAX230EPP	-40°C to +85°C	20 Lead Plastic DIP
MAX230EWP	-40°C to +85°C	20 Lead Wide S.O.
MAX230EJP	-40°C to +85°C	20 Lead CERDIP
MAX230MJP	-55°C to +125°C	20 Lead CERDIP
MAX231		**0.3" Wide**
MAX231CPD	0°C to +70°C	14 Lead Plastic DIP
MAX231CWE	0°C to +70°C	16 Lead Wide S.O.
MAX231C/D	0°C to +70°C	Dice
MAX231EPD	-40°C to +85°C	14 Lead Plastic DIP
MAX231EWE	-40°C to +85°C	16 Lead Wide S.O.
MAX231EJD	-40°C to +85°C	14 Lead CERDIP
MAX231MJD	-55°C to +125°C	14 Lead CERDIP
MAX232		**0.3" Wide**
MAX232CPE	0°C to +70°C	16 Lead Plastic DIP
MAX232CWE	0°C to +70°C	16 Lead Wide S.O.
MAX232C/D	0°C to +70°C	Dice
MAX232EPE	-40°C to +85°C	16 Lead Plastic DIP
MAX232EJE	-40°C to +85°C	16 Lead CERDIP
MAX232EWE	-40°C to +85°C	16 Lead Wide S.O.
MAX232MJE	-55°C to +125°C	16 Lead CERDIP
MAX233		**0.3" Wide**
MAX233CPP	0°C to +70°C	20 Lead Plastic DIP
MAX233EPP	-40°C to +85°C	20 Lead Plastic DIP
MAX234		**0.3" Wide**
MAX234CPE	0°C to +70°C	16 Lead Plastic DIP
MAX234CWE	0°C to +70°C	16 Lead Wide S.O.
MAX234C/D	0°C to +70°C	Dice
MAX234EPE	-40°C to +85°C	16 Lead Plastic DIP
MAX234EWE	-40°C to +85°C	16 Lead Wide S.O.
MAX234EJE	-40°C to +85°C	16 Lead CERDIP
MAX234MJE	-55°C to +125°C	16 Lead CERDIP
MAX235		**0.6" Wide**
MAX235CPG	0°C to +70°C	24 Lead Plastic DIP*
MAX235EPG	-40°C to +85°C	24 Lead Plastic DIP*
MAX235EDG	-40°C to +85°C	24 Lead Ceramic*
MAX235MDG	-55°C to +125°C	24 Lead Ceramic*

* = 0.600" package

PART	TEMP. RANGE	PACKAGE
MAX236		**0.3" Wide**
MAX236CNG	0°C to +70°C	24 Lead Plastic DIP
MAX236CWG	0°C to +70°C	24 Lead Wide S.O.
MAX236C/D	0°C to +70°C	Dice
MAX236ENG	-40°C to +85°C	24 Lead Plastic DIP
MAX236EWG	-40°C to +85°C	24 Lead Wide S.O.
MAX236ERG	-40°C to +85°C	24 Lead CERDIP
MAX236MRG	-55°C to +125°C	24 Lead CERDIP
MAX237		**0.3" Wide**
MAX237CNG	0°C to +70°C	24 Lead Plastic DIP
MAX237CWG	0°C to +70°C	24 Lead Wide S.O.
MAX237C/D	0°C to +70°C	Dice
MAX237ENG	-40°C to +85°C	24 Lead Plastic DIP
MAX237EWG	-40°C to +85°C	24 Lead Wide S.O.
MAX237ERG	-40°C to +85°C	24 Lead CERDIP
MAX237MRG	-55°C to +125°C	24 Lead CERDIP
MAX238		**0.3" Wide**
MAX238CNG	0°C to +70°C	24 Lead Plastic DIP
MAX238CWG	0°C to +70°C	24 Lead Wide S.O.
MAX238C/D	0°C to +70°C	Dice
MAX238ENG	-40°C to +85°C	24 Lead Plastic DIP
MAX238EWG	-40°C to +85°C	24 Lead Wide S.O
MAX238ERG	-40°C to +85°C	24 Lead CERDIP
MAX238MRG	-55°C to +125°C	24 Lead CERDIP
MAX239		**0.3" Wide**
MAX239CNG	0°C to +70°C	24 Lead Plastic DIP
MAX239CWG	0°C to +70°C	24 Lead Wide S.O.
MAX239C/D	0°C to +70°C	Dice
MAX239ENG	-40°C to +85°C	24 Lead Plastic DIP
MAX239EWG	-40°C to +85°C	24 Lead Wide S.O.
MAX239ERG	-40°C to +85°C	24 Lead CERDIP
MAX239MRG	-55°C to +125°C	24 Lead CERDIP
MAX240		**Flatpak**
MAX240CMH	0°C to +70°C	44 Lead Flatpak
MAX240EMH	-40°C to +85°C	44 Lead Flatpak
MAX241		**0.3" Wide**
MAX241CWI	0°C to +70°C	28 Lead Wide S.O.
MAX241EWI	-40°C to +85°C	28 Lead Wide S.O.

+5V Powered RS-232 Drivers/Receivers

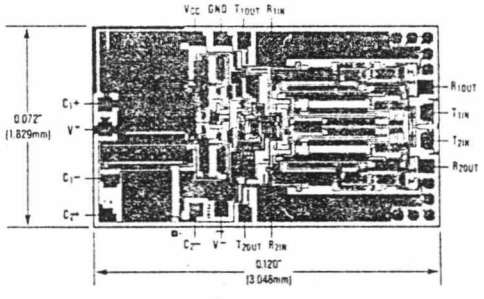
Table 1. Circuits Commonly Used for RS-232C and V.24 Asynchronous Interfaces

PIN	CIRCUIT	
1	Protective Ground	Connect to Earth Ground
2	Transmit Data (TD)	Data from DTE
3	Receive Data (RD)	Data from DCE
4	Request To Send (RTS)	Handshake from DTE
5	Clear to Send (CTS)	Handshake from DCE
6	Data Set ready (DSR)	Handshake from DCE
7	Signal Ground	Reference Point for Signals
8	Received Line Signal Detector (sometimes called Carrier Detect, DCD)	Handshake from DCE
11	Printer Busy Signal	Handshake from Printer
20	Data Terminal Ready	Handshake from DTE
22	Ring Indicator	Handshake from DCE

Table 2. Summary of RS-232C and V.28 Electrical Specifications

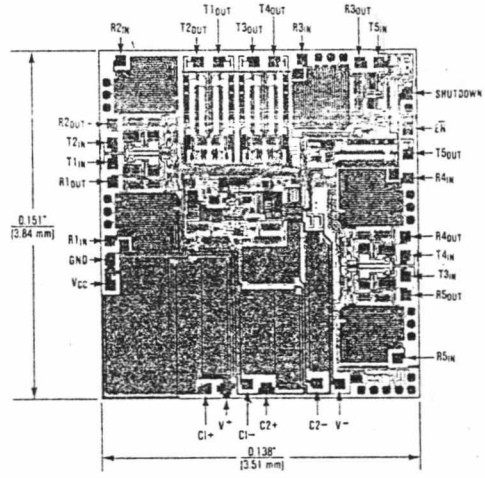
PARAMETER	SPECIFICATION	COMMENTS
Driver Output Voltage		
0 level	+5V to +15V	With 3-7kΩ load
1 level	-5V to -15V	With 3-7kΩ load
Max. output	±25V Max.	No Load
Receiver Input Thresholds (data and clock signals)		
0 level	+3V to +25V	
1 level	-3V to -25V	
Receiver Thresholds RTS, DSR, DTR		
On level	+3V to +25V	
Off level	Open Circuit or -3V to -25V	Detects Power Off Condition at Driver
Receiver Input Resistance	3kΩ to 7kΩ	
Driver Output Resistance, power off condition	300Ω Min.	V_{OUT} < ±2V
Driver Slew Rate	30V/μs Max.	3kΩ < R_L < 7kΩ; 0pF < C_L < 2500pF
Signalling Rate	Up to 20kbits/sec.	
Cable Length	50'/15 m. Recommended Max. Length	Longer cables permissible, if C_{LOAD} ≤ 2500pF

Chip Topography



MAX231, MAX232 and MAX233

Note: Connect substrate to V+.



MAX230 and MAX234-239, MAX240, MAX241

Notes:
1. Shutdown pin of MAX234, MAX237, MAX238, MAX239, MAX240 and MAX241 are internally connected to ground.
2. Connect substrate to V+.

Maxim cannot assume responsibility for use of any circuitry other than circuitry entirely embodied in a Maxim product. No circuit patent licenses are implied. Maxim reserves the right to change the circuitry and specifications without notice at any time.

HARRIS SEMICONDUCTOR

ICL7109
12-Bit µP-Compatible A/D Converter

GENERAL DESCRIPTION

The ICL7109 is a high performance, CMOS, low-power integrating A/D converter designed to easily interface with microprocessors.

The output data (12 bits, polarity and overrange) may be directly accessed under control of two byte enable inputs and a chip select input for a simple parallel bus interface. A UART handshake mode is provided to allow the ICL7109 to work with industry-standard UARTs in providing serial data transmission, ideal for remote data logging applications. The RUN/HOLD input and STATUS output allow monitoring and control of conversion timing.

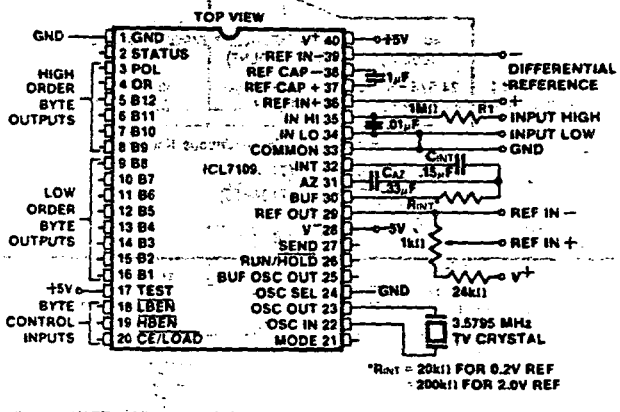
The ICL7109 provides the user with the high accuracy, low noise, low drift, versatility and economy of the dual-slope integrating A/D converter. Features like true differential input and reference, drift of less than 1µV/°C, maximum input bias current of 10pA, and typical power consumption of 20mW make the ICL7109 an attractive per-channel alternative to analog multiplexing for many data acquisition applications.

FEATURES

- **12 Bit Binary (Plus Polarity and Overrange) Dual Slope Integrating Analog-to-Digital Converter**
- **Byte-Organized TTL-Compatible Three-State Outputs and UART Handshake Mode for Simple Parallel or Serial Interfacing to Microprocessor Systems**
- **RUN/HOLD Input and STATUS Output Can Be Used to Monitor and Control Conversion Timing**
- **True Differential Input and Differential Reference**
- **Low Noise — Typically 15µV p-p**
- **1pA Typical Input Current**
- **Operates At Up to 30 Conversions Per Second**
- **On-Chip Oscillator Operates With Inexpensive 3.58MHz TV Crystal Giving 7.5 Conversions Per Second for 60Hz Rejection. May Also Be Used With An RC Network Oscillator for Other Clock Frequencies**

ORDERING INFORMATION

Part Number	Temp. Range	Package
ICL7109MDL	−55°C to +125°C	40-Pin Ceramic DIP
ICL7109IDL	−25°C to +85°C	40-Pin Ceramic DIP
ICL7109IJL	−25°C to +85°C	40-Pin CERDIP
ICL7109CPL	0°C to 70°C	40-Pin Plastic DIP



(See Figure 2 for typical connection to a UART or Microcomputer)

Figure 1. Pin Configuration and Test Circuit

HARRIS SEMICONDUCTOR'S SOLE AND EXCLUSIVE WARRANTY OBLIGATION WITH RESPECT TO THIS PRODUCT SHALL BE THAT STATED IN THE WARRANTY ARTICLE OF THE CONDITION OF SALE. THE WARRANTY SHALL BE EXCLUSIVE AND SHALL BE IN LIEU OF ALL OTHER WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, INCLUDING THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR USE.

301655-003

NOTE: All typical values have been characterized but are not tested.

ABSOLUTE MAXIMUM RATINGS

Positive Supply Voltage (GND to V+) ..... +6.2V
Negative Supply Voltage (GND to V−) ..... −9V
Analog Input Voltage (Lo or Hi) (Note 1) ..... V+ to V−
Reference Input Voltage (Lo or Hi) (Note 1) ..... V+ to V−
Digital Input Voltage (Pins 2-27) (Note 2) ..... V+ +0.3V to GND −0.3V

Power Dissipation (Note 3)
Ceramic Package ..... 833mW @ +85°C
Plastic Package ..... 500mW @ +70°C
Operating Temperature
Ceramic Package (MDL) ..... −55°C to +125°C
Ceramic Package (IDL) ..... −25°C to +85°C
Plastic Package (CPL) ..... 0°C to +70°C
Storage Temperature ..... −65°C to +150°C
Lead Temperature (Soldering, 10sec) ..... +300°C

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

ELECTRICAL CHARACTERISTICS

($V^+ = +5V$, $V^- = -5V$, GND = 0V, $T_A = 25°C$, $f_{CLK} = 3.58$ MHz, unless otherwise indicated.) Test circuit as shown on first page of this data sheet.

ANALOG SECTION

Symbol	Parameter	Test Conditions	Min	Typ	Max	Unit
	Zero Input Reading	V_{IN} = 0.0000V V_{REF} = 204.8 mV	−0000	±0000	+0000	Counts
	Ratiometric Error(4)	$V_{IN} = V_{REF}$ = 204.8 mV	−3		0	Counts
	Non-Linearity (Max deviation from best straight line fit)	Full Scale = 409.6mV to 2.048V Over full operating temperature range. (Note 4), (Note 6)	−1	±.2	+1	Counts
	Roll-over Error (difference in reading for equal pos. and neg. inputs near full scale)	Full Scale = 409.6mV to 2.048V (Note 5), (Note 6)	−1	±.2	+1	Counts
CMRR	Common Mode Rejection Ratio	V_{CM} ±1V, V_{IN} = 0V Full Scale = 409.6mV		50		µV/V
VCMR	Input Common Mode Range	Input Hi, Input Lo, Common (Note 4)	V− +1.5		V+ −1.0	V
e_n	Noise (p-p value not exceeded 95% of time)	V_{IN} = 0V Full Scale = 409.6mV		15		µV
I_{ILK}	Leakage current at Input	V_{IN} = 0 All devices at 25°C ICL7109CPL 0°C ≤ T_A ≤ +70°C (Note 4) ICL7109IDL −25°C ≤ T_A ≤ +85°C (Note 4) ICL7109MDL −55°C ≤ T_A ≤ +125°C		1 20 100 2	10 100 250 5	pA pA pA nA
	Zero Reading Drift	V_{IN} = 0V R_1 = 0Ω (Note 4)		0.2	1	µV/°C
	Scale Factor Temperature Coefficient	V_{IN} = 408.9mV = >7770$_8$ reading Ext. Ref. 0 ppm/°C (Note 4)		1	5	ppm/°C
I^+	Supply Current V+ to GND	V_{IN} = 0, Crystal Osc 3.58MHz test circuit		700	1500	µA
I_{SUPP}	Supply Current V+ to V−	Pins 2-21, 25, 26, 27, 29; open		700	1500	µA
V_{REF}	Ref Out Voltage	Referred to V+, 25kΩ between V+ and REF OUT	−2.4	−2.8	−3.2	V
	Ref Out Temp. Coefficient	25kΩ between V+ and REF OUT		80		ppm/°C

NOTE: All typical values have been characterized but are not tested.

ELECTRICAL CHARACTERISTICS ($V^+ = +5V$, $V^- = -5V$, GND = 0V, $T_A = 25°C$ unless otherwise indicated.) Test circuit as shown on first page of this data sheet. (Continued)

DIGITAL SECTION

Symbol	Parameter		Test Conditions	Min	Typ	Max	Unit
V_{OH}	Output High Voltage		$I_{OUT} = 100\mu A$ Pins 2-16, 18, 19, 20	3.5	4.3		V
V_{OL}	Output Low Voltage		$I_{OUT} = 1.6mA$		0.2	0.4	V
	Output Leakage Current		Pins 3-16 high impedance		±.01	±1	µA
	Control I/O Pullup Current		Pins 18, 19, 20 $V_{OUT} = V^+ - 3V$ MODE input at GND		5		µA
	Control I/O Loading		HBEN Pin 19 LBEN Pin 18 (Note 4)			50	pF
V_{IH}	Input High Voltage		Pins 18-21, 26, 27 referred to GND	3.0			V
V_{IL}	Input Low Voltage		Pins 18-21, 26, 27 referred to GND			1	V
	Input Pull-up Current		Pins 26, 27 $V_{OUT} = V^+ - 3V$		5		µA
	Input Pull-up Current		Pins 17, 24 $V_{OUT} = V^+ - 3V$		25		µA
	Input Pull-down Current		Pin 21 V_{OUT} = GND + 3V		5		µA
O_{OH}	Oscillator Output	High	$V_{OUT} = 2.5V$		1		mA
O_{OL}	Current	Low	$V_{OUT} = 2.5V$		1.5		mA
BO_{OH}	Buffered Oscillator	High	$V_{OUT} = 2.5V$		2		mA
BO_{OL}	Output Current	Low	$V_{OUT} = 2.5V$		5		mA
t_W	MODE Input Pulse Width		(Note 4)	50			ns

NOTES: 1. Input voltages may exceed the supply voltages provided the input current is limited to ±100µA.
2. Due to the SCR structure inherent in the process used to fabricate these devices, connecting any digital inputs or outputs to voltages greater than V^+ or less than GND may cause destructive device latchup. For this reason it is recommended that no inputs from sources other than the same power supply be applied to the ICL7109 before its power supply is established, and that in multiple supply systems the supply to the ICL7109 be activated first.
3. This limit refers to that of the package and will not be obtained during normal operation.
4. This parameter is not production tested, but is guaranteed by design.
5. Roll-over error for $T_A = -55°C$ to $+125°C$ is ±3 counts maximum.
6. A full scale voltage of 2.048V is used because a full scale voltage of 4.096V exceeds the devices Common Mode Voltage Range.
7. For Cerdip package the Ratiometric error can be −4 (Min.).

NOTE: All typical values have been characterized but are not tested.

TABLE 1: Pin Assignment and Function Description

Pin	Symbol	Description		
1	GND	Digital Ground, 0V. Ground return for all digital logic.		
2	STATUS	Output High during integrate and deintegrate until data is latched. Output Low when analog section is in Auto-Zero configuration.		
3	POL	Polarity — HI for Positive input.		
4	OR	Overrange — HI if Overranged.		
5	B12	Bit 12	(Most Significant Bit)	All three state output data bits
6	B11	Bit 11		
7	B10	Bit 10		
8	B9	Bit 9		
9	B8	Bit 8		
10	B7	Bit 7	Hi = true	
11	B6	Bit 6		
12	B5	Bit 5		
13	B4	Bit 4		
14	B3	Bit 3		
15	B2	Bit 2		
16	B1	Bit 1	(Least Significant Bit)	
17	TEST	Input High — Normal Operation. Input Low — Forces all bit outputs high. Note: This input is used for test purposes only. Tie high if not used.		
18	$\overline{LBEN}$	Low Byte Enable — With Mode (Pin 21) low, and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates low order byte outputs B1 — B8. — With Mode (Pin 21) high, this pin serves as a low byte flag output used in handshake mode. See Figures 8, 9, 10.		
19	$\overline{HBEN}$	High Byte Enable — With Mode (Pin 21) low, and $\overline{CE/LOAD}$ (Pin 20) low, taking this pin low activates high order byte outputs B9 — B12, POL, OR. — With Mode (Pin 21) high, this pin serves as a high byte flag output used in handshake mode. See Figures 8, 9, 10.		
20	$\overline{CE/LOAD}$	Chip Enable Load — With Mode (Pin 21) low, $\overline{CE/LOAD}$ serves as a master output enable. When high, B1 — B12, POL, OR outputs are disabled. — With Mode (Pin 21) high, this pin serves as a load strobe used in handshake mode. See Figures 8, 9, 10.		

Pin	Symbol	Description
21	MODE	Input Low — Direct output mode where $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19) and $\overline{LBEN}$ (Pin 18) act as inputs directly controlling byte outputs. Input Pulsed High — Causes immediate entry into handshake mode and output of data as in Figure 10. Input High — Enables $\overline{CE/LOAD}$ (Pin 20), $\overline{HBEN}$ (Pin 19), and $\overline{LBEN}$ (Pin 18) as outputs, handshake mode will be entered and data output as in Figures 8 and 9 at conversion completion.
22	OSC IN	Oscillator Input
23	OSC OUT	Oscillator Output
24	OSC SEL	Oscillator Select — Input high configures OSC IN, OSC OUT, BUF OSC OUT as RC oscillator — clock will be same phase and duty cycle as BUF OSC OUT. — Input low configures OSC IN, OSC OUT for crystal oscillator — clock frequency will be 1/58 of frequency at BUF OSC OUT.
25	BUF OSC OUT	Buffered Oscillator Output
26	RUN/$\overline{HOLD}$	Input High — Conversions continuously performed every 8192 clock pulses. Input Low — Conversion in progress completed, converter will stop in Auto-Zero 7 counts before integrate.
27	SEND	Input — Used in handshake mode to indicate ability of an external device to accept data. Connect to + 5V if not used.
28	V^-	Analog Negative Supply — Nominally −5V with respect to GND (Pin 1).
29	REF OUT	Reference Voltage Output — Nominally 2.8V down from V^+ (Pin 40).
30	BUFFER	Buffer Amplifier Output
31	AUTO-ZERO	Auto-Zero Node — Inside foil of C_{AZ}
32	INTEGRATOR	Integrator Output — Outside foil of C_{INT}
33	COMMON	Analog Common — System is Auto-Zeroed to COMMON
34	INPUT LO	Differential Input Low Side
35	INPUT HI	Differential Input High Side
36	REF IN +	Differential Reference Input Positive
37	REF CAP +	Reference Capacitor Positive
38	REF CAP −	Reference Capacitor Negative
39	REF IN −	Differential Reference Input Negative
40	V^+	Positive Supply Voltage — Nominally + 5V with respect to GND (Pin 1).

Note: All digital levels are positive true.

NOTE: All typical values have been characterized but are not tested.

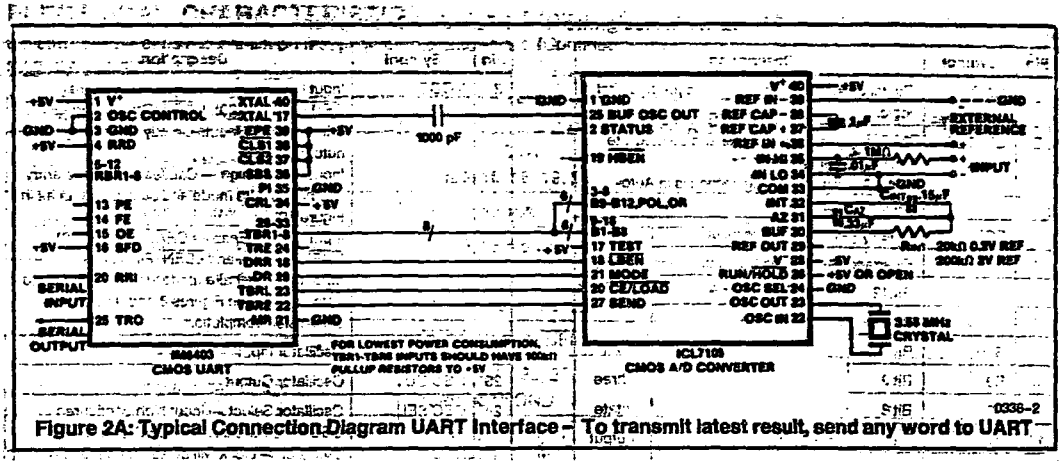


Figure 2A: Typical Connection Diagram UART Interface – To transmit latest result, send any word to UART

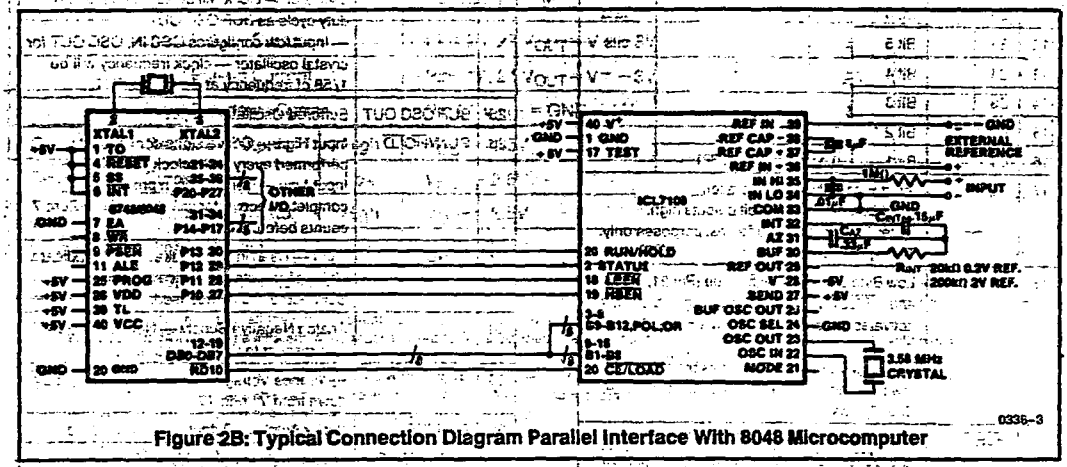


Figure 2B: Typical Connection Diagram Parallel Interface With 8048 Microcomputer

DETAILED DESCRIPTION

Analog Section

Figure 3 shows the equivalent circuit of the Analog Section of the ICL7109. When the RUN/HOLD input is left open or connected to V+, the circuit will perform conversions at a rate determined by the clock frequency (8192 clock periods per cycle). Each measurement cycle is divided into three phases as shown in Figure 4. They are (1) Auto-Zero (AZ), (2) Signal Integrate (INT) and (3) Deintegrate (DE).

Auto-Zero Phase

During auto-zero three things happen. First, input high and low are disconnected from their pins and internally shorted to analog COMMON. Second, the reference capacitor is charged to the reference voltage. Third, a feedback loop is closed around the system to charge the auto-zero capacitor C_{AZ} to compensate for offset voltages in the buffer amplifier, integrator, and comparator. Since the comparator is included in the loop, the AZ accuracy is limited only by the noise of the system. In any case, the offset referred to the input is less than 10μV.

NOTE: All typical values have been characterized but are not tested.

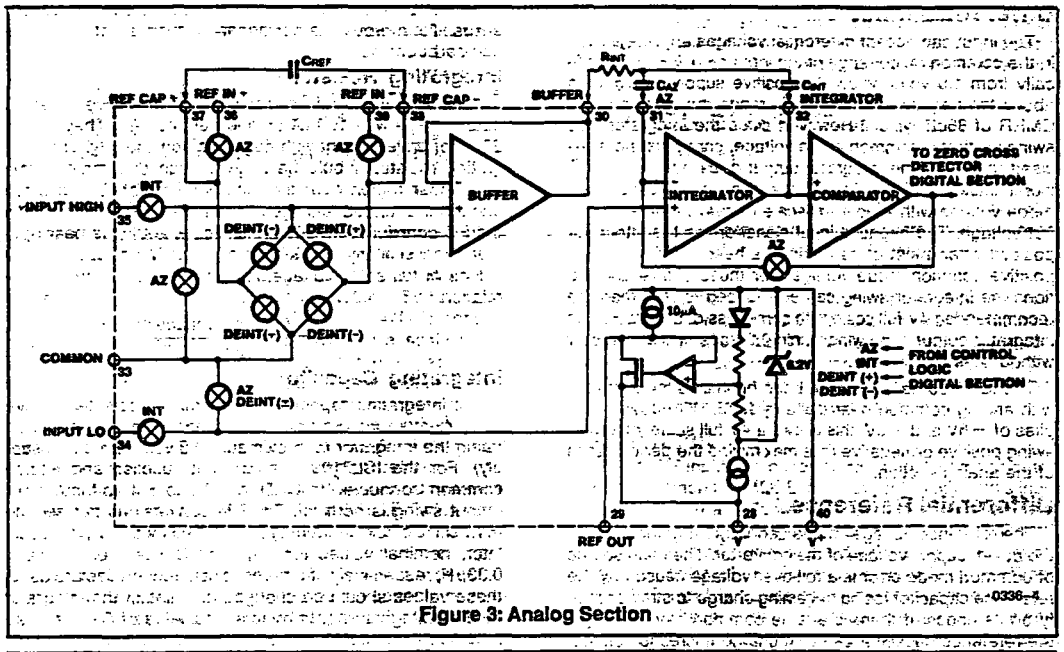


Figure 3: Analog Section

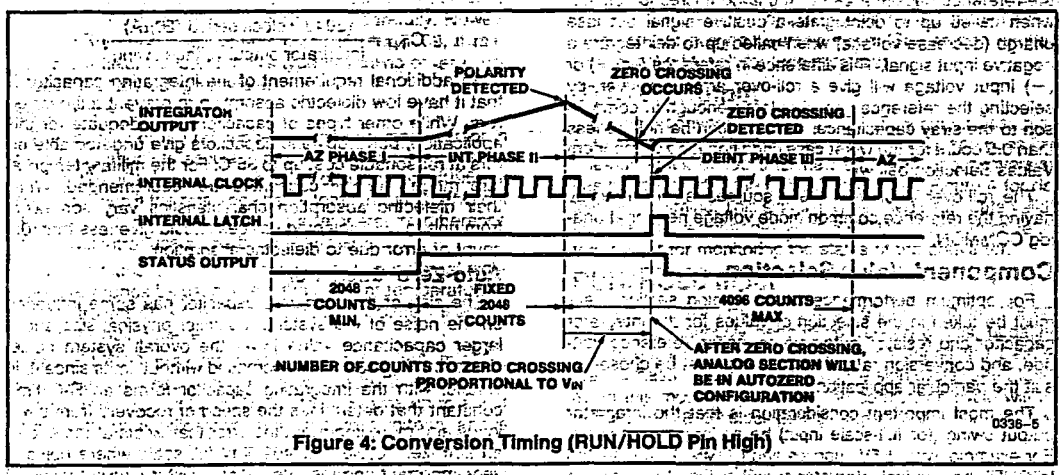


Figure 4: Conversion Timing (RUN/HOLD Pin High)

Signal Integrate Phase

During signal integrate the auto-zero loop is opened, the internal short is removed and the internal high and low inputs are connected to the external pins. The converter then integrates the differential voltage between IN HI and IN LO for a fixed time of 2048 clock periods. Note that this differential voltage must be within the common mode range of the inputs. At the end of this phase, the polarity of the integrated signal is determined.

De-Integrate Phase

The final phase is de-integrate, or reference integrate. Input low is internally connected to analog COMMON and input high is connected across the previously charged (during auto-zero) reference capacitor. Circuitry within the chip ensures that the capacitor will be connected with the correct polarity to cause the integrator output to return to zero crossing (established in Auto Zero) with a fixed slope. Thus the time for the output to return to zero (represented by the number of clock periods counted) is proportional to the input signal.

NOTE: All typical values have been characterized but are not tested

Differential Input

The input can accept differential voltages anywhere within the common mode range of the input amplifier, or specifically from 1.0 volts below the positive supply, to 1.5 volts above the negative supply. In this range the system has a CMRR of 86dB typical. However, since the integrator also swings with the common mode voltage, care must be exercised to assure the integrator output does not saturate. A worst case condition would be a large positive common mode voltage with a near full-scale negative differential input voltage. The negative input signal drives the integrator positive when most of its swing has been used up by the positive common mode voltage. For these critical applications the integrator swing can be reduced to less than the recommended 4V full scale with some loss of accuracy. The integrator output can swing within 0.3 volts of either supply without loss of linearity.

The ICL7109 has, however, been optimized for operation with analog common near digital ground. With power supplies of +5V and −5V, this allows a 4V full scale integrator swing positive or negative thus maximizing the performance of the analog section.

Differential Reference

The reference voltage can be generated anywhere within the power supply voltage of the converter. The main source of common mode error is a roll-over voltage caused by the reference capacitor losing or gaining charge to stray capacity on its nodes. If there is a large common mode voltage, the reference capacitor can gain charge (increase voltage) when called up to deintegrate a positive signal but lose charge (decrease voltage) when called up to deintegrate a negative input signal. This difference in reference for (+) or (−) input voltage will give a roll-over error. However, by selecting the reference capacitor large enough in comparison to the stray capacitance, this error can be held to less than 0.5 count for the worst case condition (see Component Values Selection below).

The roll-over error from these sources is minimized by having the reference common mode voltage near or at analog COMMON.

Component Value Selection

For optimum performance of the analog section, care must be taken in the selection of values for the integrator capacitor and resistor, auto-zero capacitor, reference voltage, and conversion rate. These values must be chosen to suit the particular application.

The most important consideration is that the integrator output swing (for full-scale input) be as large as possible. For example, with ±5V supplies and COMMON connected to GND, the nominal integrator output swing at full scale is ±4V. Since the integrator output can go to 0.3V from either supply without significantly affecting linearity, a 4V integrator output swing allows 0.7V for variations in output swing due to component value and oscillator tolerances. With ±5V supplies and a common mode range of ±1V required, the component values should be selected to provide ±3V integrator output swing. Noise and rollover errors will be slightly worse than in the ±4V case. For larger common mode voltage ranges, the integrator output swing must be reduced further. This will increase both noise and rollover errors. To improve the performance, supplies of ±6V may be used.

Integrating Resistor

Both the buffer amplifier and the integrator have a class A output stage with 100μA of quiescent current. They supply 20μA of drive current with negligible non-linearity. The integrating resistor should be large enough to remain in this very linear region over the input voltage range, but small enough that undue leakage requirements are not placed on the PC board. For 4.096 volt full scale, 200kΩ is near optimum and similarly a 20kΩ for a 409.6mV scale. For other values of full scale voltage, R_{INT} should be chosen by the relation

$$R_{INT} = \frac{\text{full scale voltage}}{20\mu A}$$

Integrating Capacitor

The integrating capacitor C_{INT} should be selected to give the maximum integrator output voltage swing without saturating the integrator (approximately 0.3 volt from either supply). For the ICL7109 with ±5 volt supplies and analog common connected to GND, a ±3.5 to ±4 volt integrator output swing is nominal. For 7-½ conversions per second (61.72kHz clock frequency) as provided by the crystal oscillator, nominal values for C_{INT} and C_{AZ} are 0.15μF and 0.33μF, respectively. If different clock frequencies are used, these values should be changed to maintain the integrator output voltage swing. In general, the value of C_{INT} is given by

$$C_{INT} = \frac{(2048 \times \text{clock period})(20\mu A)}{\text{integrator output voltage swing}}\ \mu F$$

An additional requirement of the integrating capacitor is that it have low dielectric absorption to prevent roll-over errors. While other types of capacitors are adequate for this application, polypropylene capacitors give undetectable errors at reasonable cost up to 85°C. For the military temperature range, Teflon® capacitors are recommended. While their dielectric absorption characteristics vary somewhat from unit to unit, selected devices should give less than 0.5 count of error due to dielectric absorption.

Auto-Zero Capacitor

The size of the auto-zero capacitor has some influence on the noise of the system: a smaller physical size and a larger capacitance value lower the overall system noise. However, C_{AZ} cannot be increased without limits since it, in parallel with the integrating capacitor forms an R-C time constant that determines the speed of recovery from overloads and more important the error that exists at the end of an auto-zero cycle. For 409.6mV full scale where noise is very important and the integrating resistor small, a value of C_{AZ} twice C_{INT} is optimum. Similarly for 4.096V full scale where recovery is more important than noise, a value of C_{AZ} equal to half of C_{INT} is recommended.

For optimal rejection of stray pickup, the outer foil of C_{AZ} should be connected to the R-C summing junction and the inner foil to pin 31. Similarly the outer foil of C_{INT} should be connected to pin 32 and the inner foil to the R-C summing junction. Teflon®, or equivalent, capacitors are recommended above 85°C for their low leakage characteristics.

NOTE: All typical values have been characterized but are not tested.

Reference Capacitor

A 1µF capacitor gives good results in most applications. However, where a large reference common mode voltage exists (i.e. the reference low is not at analog common) and a 409.6mV scale is used, a larger value is required to prevent roll-over error. Generally 10µF will hold the roll-over error to 0.5 count in this instance. Again, Teflon®, or equivalent capacitors should be used for temperatures above 85°C for their low leakage characteristics.

Reference Voltage

The analog input required to generate a full scale output of 4096 counts is $V_{IN} = 2V_{REF}$. Thus for a normalized scale, a reference of 2.048V should be used for a 4.096V full scale, and 204.8mV should be used for a 0.4096V full scale. However, in many applications where the A/D is sensing the output of a transducer, there will exist a scale factor other than unity between the absolute output voltage to be measured and a desired digital output. For instance, in a weighing system, the designer might like to have a full scale reading when the voltage from the transducer is 0.682V. Instead of dividing the input down to 409.6mV, the input voltage should be measured directly and a reference voltage of 0.341V should be used. Suitable values for integrating resistor and capacitor are 33kΩ and 0.15µF. This avoids a divider on the input. Another advantage of this system occurs when a zero reading is desired for non-zero input. Temperature and weight measurements with an offset or tare are examples. The offset may be introduced by connecting the voltage output of the transducer between common and analog high, and the offset voltage between common and analog low, observing polarities carefully. However, in processor-based systems using the ICL7109, it may be more efficient to perform this type of scaling or tare subtraction digitally using software.

Reference Sources

The stability of the reference voltage is a major factor in the overall absolute accuracy of the converter. The resolution of the ICL7109 at 12 bits is one part in 4096, or 244ppm. Thus if the reference has a temperature coefficient of 80ppm/°C (onboard reference) a temperature difference of 3°C will introduce a one-bit absolute error.

For this reason, it is recommended that an external high-quality reference be used where the ambient temperature is not controlled or where high-accuracy absolute measurements are being made.

The ICL7109 provides a REFerence OUTput (pin 29) which may be used with a resistive divider to generate a suitable reference voltage. This output will sink up to about 20mA without significant variation in output voltage, and is provided with a pullup bias device which sources about 10µA. The output voltage is nominally 2.8V below V+, and has a temperature coefficient of ±80ppm/°C typ. When using the onboard reference, REF OUT (Pin 29) should be connected to REF− (pin 39), and REF+ should be connected to the wiper of a precision potentiometer between REF OUT and V+. The circuit for a 204.8mV reference is shown in the test circuit. For a 2.048mV reference, the fixed resistor should be removed, and a 25kΩ precision potentiometer between REF OUT and V+ should be used.

Note that if pins 29 and 39 are tied together and pins 39 and 40 accidentally shorted (e.g., during testing), the reference supply will sink enough current to destroy the device. This can be avoided by placing a 1kΩ resistor in series with pin 39.

DETAILED DESCRIPTION

Digital Section

The digital section includes the clock oscillator and scaling circuit, a 12-bit binary counter with output latches and TTL-compatible three-state output drivers, polarity, over-range and control logic, and UART handshake logic, as shown in Figure 5.

Throughout this description, logic levels will be referred to as "low" or "high". The actual logic levels are defined in the Electrical Characteristics Table. For minimum power consumption, all inputs should swing from GND (low) to V+ (high). Inputs driven from TTL gates should have 3-5kΩ pullup resistors added for maximum noise immunity.

MODE Input

The MODE input is used to control the output mode of the converter. When the MODE pin is low or left open (this input is provided with a pulldown resistor to ensure a low level when the pin is left open), the converter is in its "Direct" output mode, where the output data is directly accessible under the control of the chip and byte enable inputs. When the MODE input is pulsed high, the converter enters the UART handshake mode and outputs the data in two bytes, then returns to "direct" mode. When the MODE input is left high, the converter will output data in the handshake mode at the end of every conversion cycle. (See section entitled "Handshake Mode" for further details).

STATUS Output

During a conversion cycle, the STATUS output goes high at the beginning of Signal Integrate (Phase II), and goes low one-half clock period after new data from the conversion has been stored in the output latches. See Figure 4 for details of this timing. This signal may be used as a "data valid" flag (data never changes while STATUS is low) to drive interrupts, or for monitoring the status of the converter.

RUN/HOLD Input

When the RUN/HOLD input is high, or left open, the circuit will continuously perform conversion cycles, updating the output latches after zero crossing during the Deintegrate (Phase III) portion of the conversion cycle (See Figure 4). In this mode of operation, the conversion cycle will be performed in 8192 clock periods, regardless of the resulting value.

If RUN/HOLD goes low at any time during Deintegrate (Phase III) after the zero crossing has occurred, the circuit will immediately terminate Deintegrate and jump to Auto-Zero. This feature can be used to eliminate the time spent in Deintegrate after the zero-crossing. If RUN/HOLD stays or goes low, the converter will ensure minimum Auto-Zero time, and then wait in Auto-Zero until the RUN/HOLD input goes high. The converter will begin the Integrate (Phase II) portion of the next conversion (and the STATUS output will go high) seven clock periods after the high level is detected at RUN/HOLD. See Figure 6 for details.

NOTE: All typical values have been characterized but are not tested.

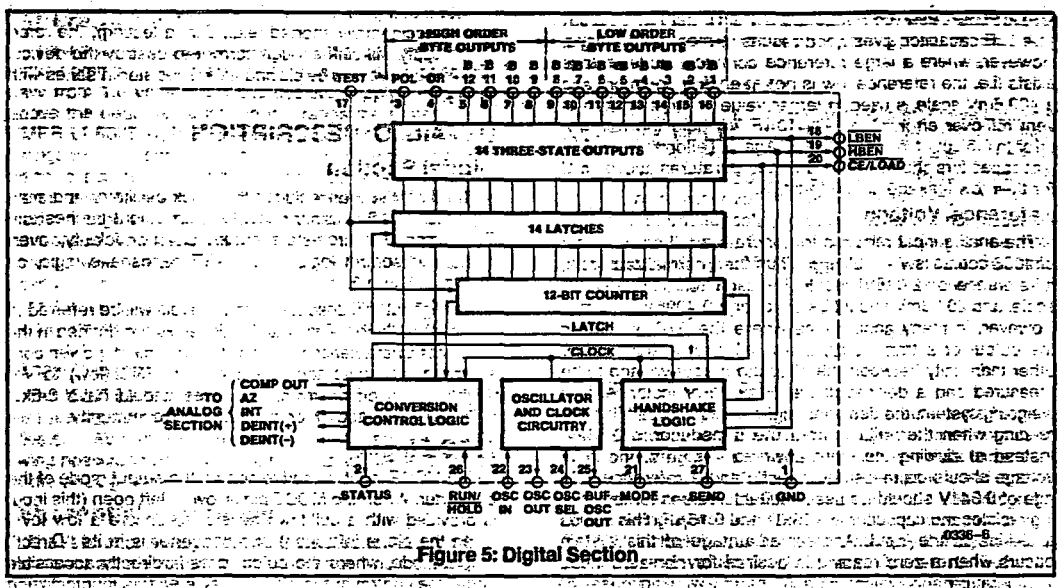


Figure 5: Digital Section

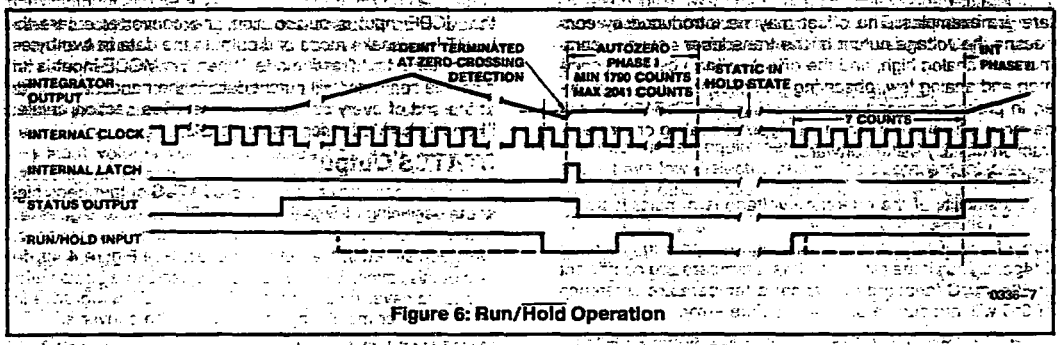


Figure 6: Run/Hold Operation

Using the RUN/HOLD input in this manner allows an easy "convert on demand" interface to be used. The converter may be held at idle in auto-zero with RUN/HOLD low. When RUN/HOLD goes high the conversion is started, and when the STATUS output goes low the new data is valid (or transferred to the UART — see Handshake Mode). RUN/HOLD may now be taken low which terminates deintegrate and ensures a minimum Auto-Zero time before the next conversion.

Alternately, RUN/HOLD can be used to minimize conversion time by ensuring that it goes low during Deintegrate, after zero crossing, and goes high after the hold point is reached. The required activity on the RUN/HOLD input can be provided by connecting it to the Buffered Oscillator Output. In this mode the conversion time is dependent on the input value measured. Also refer to Harris Application Bulletin A032 for a discussion of the effects this will have on Auto-Zero performance.

If the RUN/HOLD input goes low and stays low during Auto-Zero (Phase I), the converter will simply stop at the end of Auto-Zero and wait for RUN/HOLD to go high. As above, Integrate (Phase II) begins seven clock periods after the high level is detected.

Direct Mode

When the MODE pin is left at a low level, the data outputs (bits 1 through 8 low order byte, bits 9 through 12, polarity and over-range high order byte) are accessible under control of the byte and chip enable terminals as inputs. These three inputs are all active low, and are provided with pullup resistors to ensure an inactive high level when left open. When the chip enable input is low, taking a byte enable input low will allow the outputs of that byte to become active (three-stated on). This allows a variety of parallel data accessing techniques to be used, as shown in the section entitled "Interfacing." The timing requirements for these outputs are shown in Figure 7 and Table 2.

NOTE: All typical values have been characterized but are not tested.

Table 2—Direct Mode Timing Requirements
(See Note 4 of Electrical Characteristics)

SYMBOL	DESCRIPTION	MIN	TYP	MAX	UNIT
t_{BEA}	Byte Enable Width	350	220		ns
t_{DAB}	Data Access Time from Byte Enable		210	350	ns
t_{DHB}	Data Hold Time from Byte Enable	150	300		ns
t_{CEA}	Chip Enable Width	400	260		ns
t_{DAC}	Data Access Time from Chip Enable		260	400	ns
t_{DHC}	Data Hold Time from Chip Enable		240	400	ns

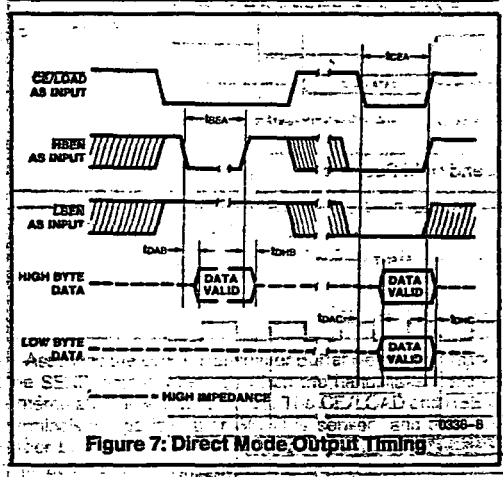


Figure 7: Direct Mode Output Timing

It should be noted that these control inputs are asynchronous with respect to the converter clock — the data may be accessed at any time. Thus it is possible to access the latches while they are being updated, which could lead to erroneous data. Synchronizing the access of the latches with the conversion cycle by monitoring the STATUS output will prevent this. Data is never updated while STATUS is low.

Handshake Mode

The handshake output mode is provided as an alternative means of interfacing the ICL7109 to digital systems, where the A/D converter becomes active in controlling the flow of data instead of passively responding to chip and byte enable inputs. This mode is specifically designed to allow a direct interface between the ICL7109 and industry-standard UARTs (such as the Harris IM6402/3) with no external logic required. When triggered into the handshake mode, the ICL7109 provides all the control and flag signals necessary to sequentially transfer two bytes of data into the UART and initiate their transmission in serial form. This greatly eases the task and reduces the cost of designing remote data acquisition stations using serial data transmission.

Entry into the handshake mode is controlled by the MODE pin. When the MODE terminal is held high, the ICL7109 will enter the handshake mode after new data has been stored in the output latches at the end of a conversion (See Figures 8 and 9). The MODE terminal may also be used to trigger entry into the handshake mode on demand. At any time during the conversion cycle, the low to high transition of a short pulse at the MODE input will cause immediate entry into the handshake mode. If this pulse occurs while new data is being stored, the entry into handshake mode is delayed until the data is stable. While the converter is in the handshake mode, the MODE input is ignored, and although conversions will still be performed, data updating will be inhibited (See Figure 10) until the converter completes the output cycle and clears the handshake mode.

When the converter enters the handshake mode, or when the MODE input is high, the chip and byte enable terminals become TTL-compatible outputs which provide the control signals for the output cycle (See Figures 8, 9, and 10).

In handshake mode, the SEND input is used by the converter as an indication of the ability of the receiving device (such as a UART) to accept data.

Figure 8 shows the sequence of the output cycle with SEND held high. The handshake mode (internal MODE high) is entered after the data latch pulse, and since MODE remains high the $\overline{CE}$/LOAD, $\overline{LBEN}$ and $\overline{HBEN}$ terminals are active as outputs. The high level at the SEND input is sensed on the same high to low internal clock edge that terminates the data latch pulse. On the next low to high internal clock edge the $\overline{CE}$/LOAD and the $\overline{HBEN}$ outputs assume a low level, and the high-order byte (bits 9 through 12, POL, and OR) outputs are enabled. The $\overline{CE}$/LOAD output remains low for one full internal clock period only, the data outputs remain active for 1-½ internal clock periods, and the high byte enable remains low for two clock periods. Thus the $\overline{CE}$/LOAD output low level or low to high edge may be used as a synchronizing signal to ensure valid data, and the byte enable as an output may be used as a byte identification flag. With SEND remaining high the converter completes the output cycle using $\overline{CE}$/LOAD and $\overline{LBEN}$ while the low order byte outputs (bits 1 through 8) are activated. The handshake mode is terminated when both bytes are sent.

Figure 9 shows an output sequence where the SEND input is used to delay portions of the sequence, or handshake to ensure correct data transfer. This timing diagram shows the relationships that occur using an industry-standard IM6402/3 CMOS UART to interface to serial data channels. In this interface, the SEND input to the ICL7109 is driven by the TBRE (Transmitter Buffer Register Empty) output of the UART, and the $\overline{CE}$/LOAD terminal of the ICL7109 drives the TBRL (Transmitter Buffer Register Load) input to the UART. The data outputs are paralleled into the eight Transmitter Buffer Register inputs.

3

NOTE: All typical values have been characterized but are not tested.

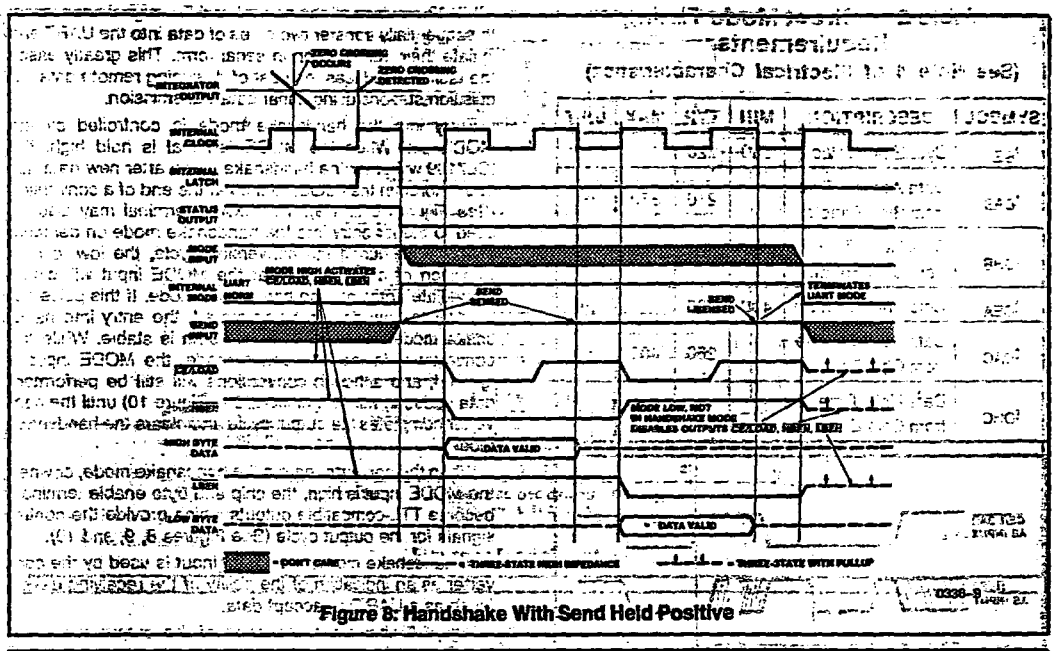


Figure 8: Handshake With Send Held Positive

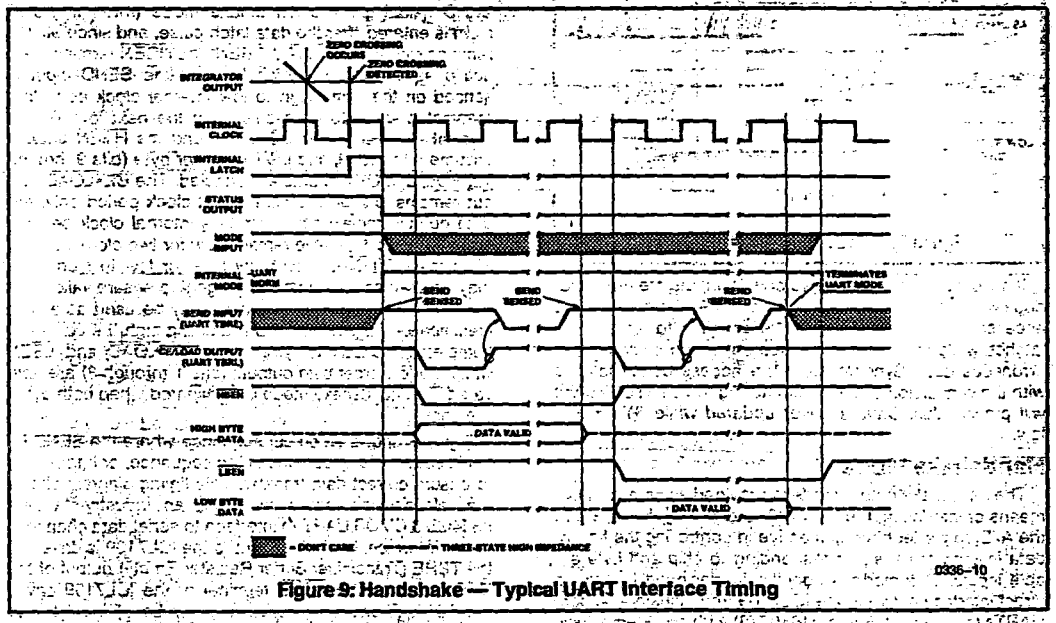


Figure 9: Handshake — Typical UART Interface Timing

NOTE: All typical values have been characterized but are not tested.

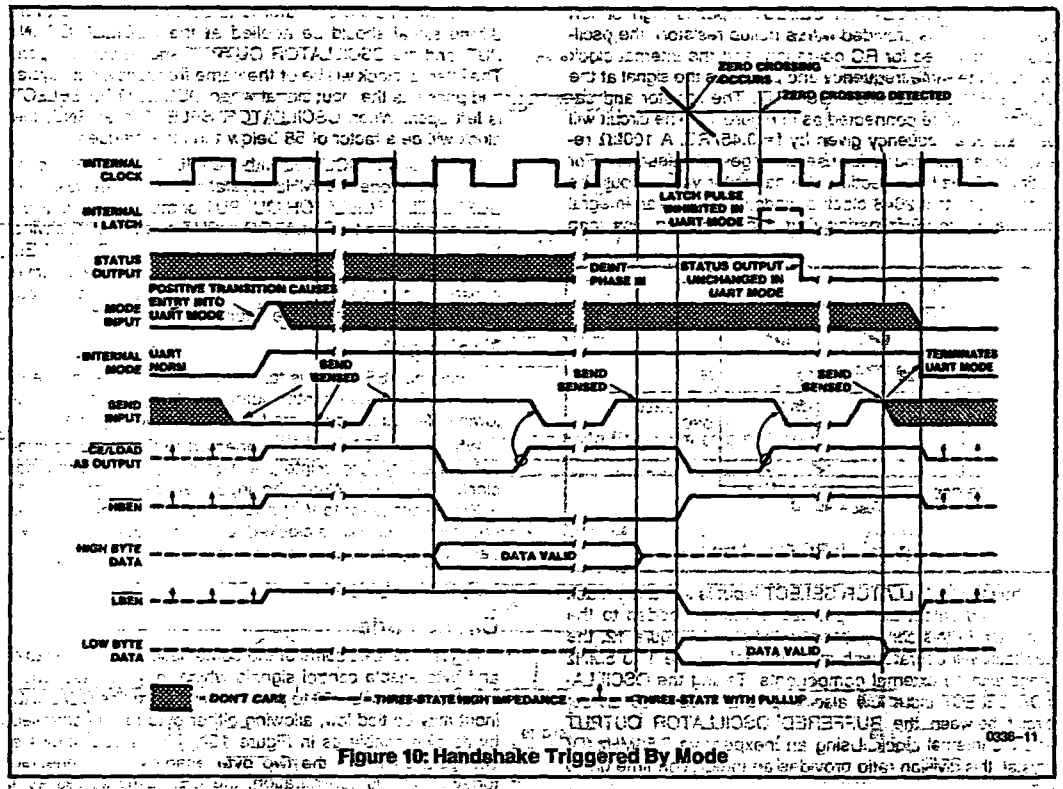


Figure 10: Handshake Triggered By Mode

Assuming the UART Transmitter Buffer Register is empty, the SEND input will be high when the handshake mode is entered after new data is stored. The $\overline{CE/LOAD}$ and $\overline{HBEN}$ terminals will go low after SEND is sensed, and the high order byte outputs become active. When $\overline{CE/LOAD}$ goes high at the end of one clock period, the high order byte data is clocked into the UART Transmitter Buffer Register. The UART TBRE output will now go low, which halts the output cycle with the $\overline{HBEN}$ output low, and the high order byte outputs active. When the UART has transferred the data to the Transmitter Register and cleared the Transmitter Buffer Register, the TBRE returns high. On the next ICL7109 internal clock high to low edge, the high order byte outputs are disabled, and one-half internal clock later, the $\overline{HBEN}$ output returns high. At the same time, the $\overline{CE/LOAD}$ and $\overline{LBEN}$ outputs go low, and the low order byte outputs become active. Similarly, when the $\overline{CE/LOAD}$ returns high at the end of one clock period, the low order data is clocked into the UART Transmitter Buffer Register, and TBRE again goes low. When TBRE returns to a high it will be sensed on the next ICL7109 internal clock high to low edge, disabling the data outputs. One-half internal clock later, the handshake mode will be cleared, and the $\overline{CE/LOAD}$, $\overline{HBEN}$, and $\overline{LBEN}$ terminals return high and stay active (as long as MODE stays high).

With the MODE input remaining high as in these examples, the converter will output the results of every conversion except those completed during a handshake operation. By triggering the converter into handshake mode with a low to high edge on the MODE input, handshake output sequences may be performed on demand. Figure 10 shows a handshake output sequence triggered by such an edge. In addition, the SEND input is shown as being low when the converter enters handshake mode. In this case, the whole output sequence is controlled by the SEND input, and the sequence for the first (high order) byte is similar to the sequence for the second byte. This diagram also shows the output sequence taking longer than a conversion cycle. Note that the converter still makes conversions, with the STATUS output and RUN/$\overline{HOLD}$ input functioning normally. The only difference is that new data will not be latched when in handshake mode, and is therefore lost.

Oscillator

The ICL7109 is provided with a versatile three terminal oscillator to generate the internal clock. The oscillator may be overdriven, or may be operated with an RC network or crystal. The OSCILLATOR SELECT input changes the internal configuration of the oscillator to optimize it for RC or crystal operation.

NOTE: All typical values have been characterized but are not tested.

When the OSCILLATOR SELECT input is high or left open (the input is provided with a pullup resistor), the oscillator is configured for RC operation, and the internal clock will be of the same frequency and phase as the signal at the BUFFERED OSCILLATOR OUTPUT. The resistor and capacitor should be connected as in Figure 11. The circuit will oscillate at a frequency given by $f = 0.45/RC$. A 100kΩ resistor is recommended for useful ranges of frequency. For optimum 60Hz line rejection, the capacitor value should be chosen such that 2048 clock periods is close to an integral multiple of the 60Hz period (but should not be less than 50pF).

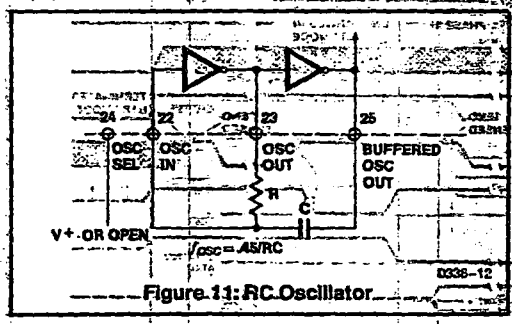


Figure 11: RC Oscillator

When the OSCILLATOR SELECT input is low a feedback device and output and input capacitors are added to the oscillator. In this configuration, as shown in Figure 12, the oscillator will operate with most crystals in the 1 to 5MHz range with no external components. Taking the OSCILLATOR SELECT input low also inserts a fixed ÷58 divider circuit between the BUFFERED OSCILLATOR OUTPUT and the internal clock. Using an inexpensive 3.58MHz TV crystal, this division ratio provides an integration time given by:

$$T_{INT.} = (2048 \text{ clock periods}) \times (T_{CLOCK}) = 33.16 \text{ ms}$$

where $T_{CLOCK} = \frac{58}{3.58 \text{ MHz}}$

This time is very close to two 60Hz periods or 33.33ms. The error is less than one percent, which will give better than 40dB 60Hz rejection. The converter will operate reliably at conversion rates of up to 30 per second, which corresponds to a clock frequency of 245.8kHz.

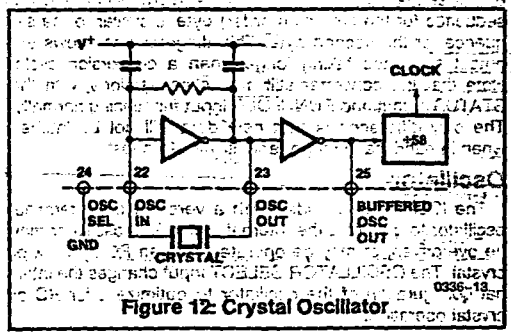


Figure 12: Crystal Oscillator

If at any time the oscillator is to be overdriven, the overdriving signal should be applied at the OSCILLATOR INPUT, and the OSCILLATOR OUTPUT should be left open. The internal clock will be of the same frequency, duty cycle, and phase as the input signal when OSCILLATOR SELECT is left open. When OSCILLATOR SELECT is at GND, the clock will be a factor of 58 below the input frequency.

When using the ICL7109 with the IM6403 UART, it is possible to use one 3.58MHz crystal for both devices. The BUFFERED OSCILLATOR OUTPUT of the ICL7109 may be used to drive the OSCILLATOR INPUT of the UART, saving the need for a second crystal. However, the BUFFERED OSCILLATOR OUTPUT does not have a great deal of drive capability, and when driving more than one slave device, external buffering should be used.

Test Input

When the TEST input is taken to a level halfway between V+ and GND, the counter output latches are enabled, allowing the counter contents to be examined anytime.

When the TEST input is connected to GND, the counter outputs are all forced into the high state, and the internal clock is disabled. When the input returns to the ½ (V+ – GND) voltage (or to V+) and one clock is applied, all the counter outputs will be clocked to the low state. This allows easy testing of the counter and its outputs.

INTERFACING

Direct Mode

Figure 13 shows some of the combinations of chip enable and byte enable control signals which may be used when interfacing the ICL7109 to parallel data lines. The CE/LOAD input may be tied low, allowing either byte to be controlled by its own enable as in Figure 13A. Figure 13B shows a configuration where the two byte enables are connected together. In this configuration, the CE/LOAD serves as a chip enable, and the HBEN and LBEN may be connected to GND or serve as a second chip enable. The 14 data outputs will all be enabled simultaneously. Figure 13C shows the HBEN and LBEN as flag inputs, and CE/LOAD as a master enable, which could be the READ strobe available from most microprocessors.

Figure 14 shows an approach to interfacing several ICL7109s to a bus, ganging the HBEN and LBEN signals to several converters together, and using the CE/LOAD inputs (perhaps decoded from an address) to select the desired converter.

NOTE: All typical values have been characterized but are not tested.

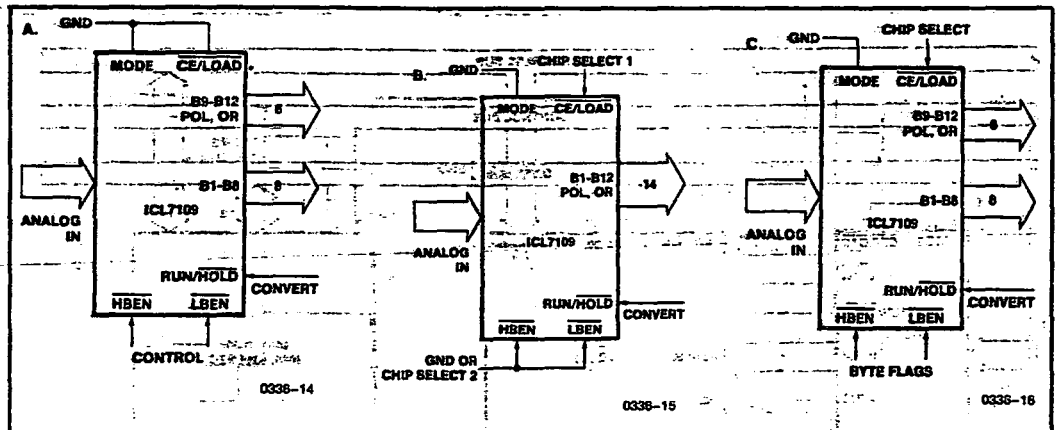


Figure 13: Direct Mode Chip and Byte Enable Combinations

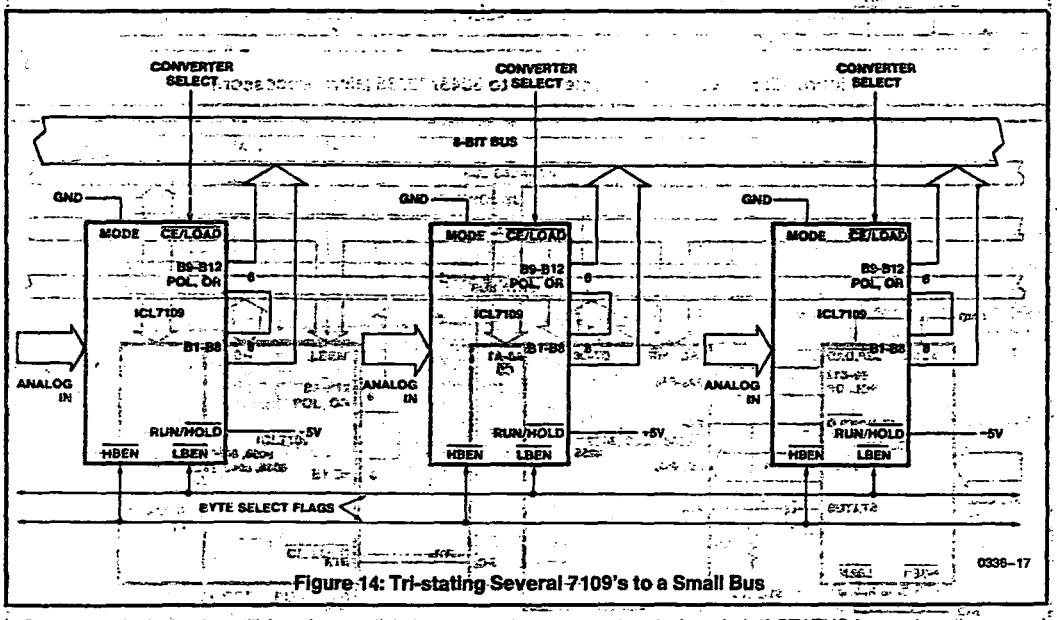


Figure 14: Tri-stating Several 7109's to a Small Bus

Some practical circuits utilizing the parallel three-state output capabilities of the ICL7109 are shown in Figures 15 through 20. Figure 15 shows a straightforward application to the Intel 8048/80/85 microprocessors via an 8255PPI, where the ICL7109 data outputs are active at all times. The I/O ports of an 8155 may be used in the same way. This interface can be used in a read-anytime mode, although a read performed while the data latches are being updated will lead to scrambled data. This will occur very rarely, in the proportion of setup-skew times to conversion time. One way to overcome this is to read the STATUS output as well, and if it is high, read the data again after a delay of more than ½ converter clock period. If STATUS is now low, the second reading is correct, and if it is still high, the first reading is correct. Alternatively, this timing problem is completely avoided by using a read-after-update sequence, as shown in Figure 16. Here the high to low transition of the STATUS output drives an interrupt to the microprocessor causing it to access the data latches. This application also shows the RUN/HOLD input being used to initiate conversions under software control.

A similar interface to Motorola MC6800 or Rockwell R650X systems is shown in Figure 17. The high to low transition of the STATUS output generates an interrupt via the

NOTE: All typical values have been characterized but are not tested.

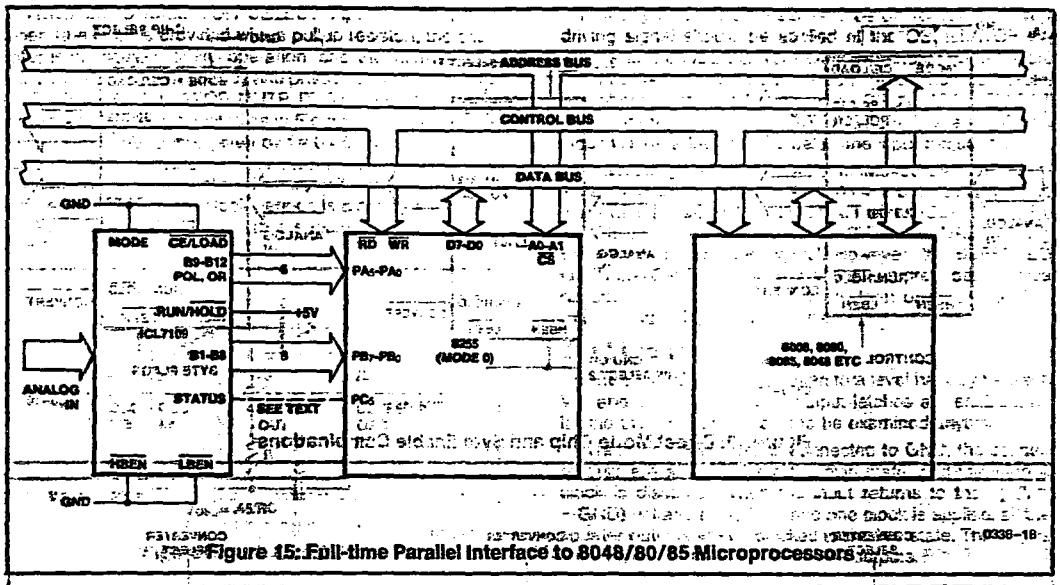


Figure 15: Full-time Parallel Interface to 8048/80/85 Microprocessors

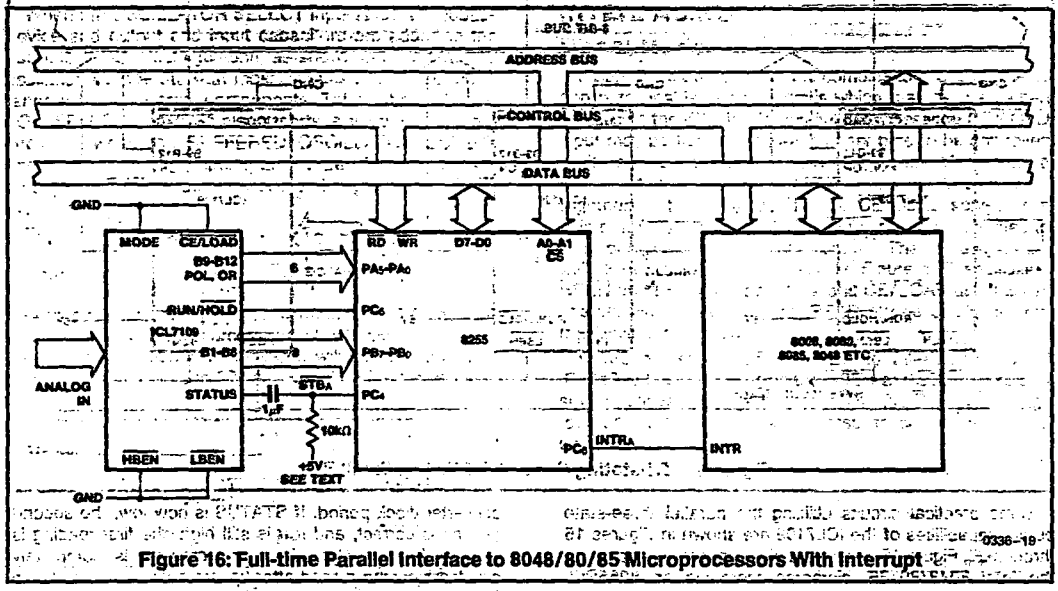


Figure 16: Full-time Parallel Interface to 8048/80/85 Microprocessors With Interrupt

Control Register B CB1 line. Note that CB2 controls the RUN/HOLD pin through Control Register B, allowing software-controlled initiation of conversions in this system as well.

The three-state output capability of the ICL7109 allows direct interfacing to most microprocessor busses. Examples of this are shown in Figures 18 and 19. It is necessary to carefully consider the system timing in this type of interface, to be sure that requirements for setup and hold times, and minimum pulse widths are met. Note also the drive limitations on long buses. Generally this type of interface is only favored if the memory peripheral address density is low so that simple address decoding can be used. Interrupt handling can also require many additional components, and using an interface device will usually simplify the system in this case.

NOTE: All typical values have been characterized but are not tested.

ICL7109

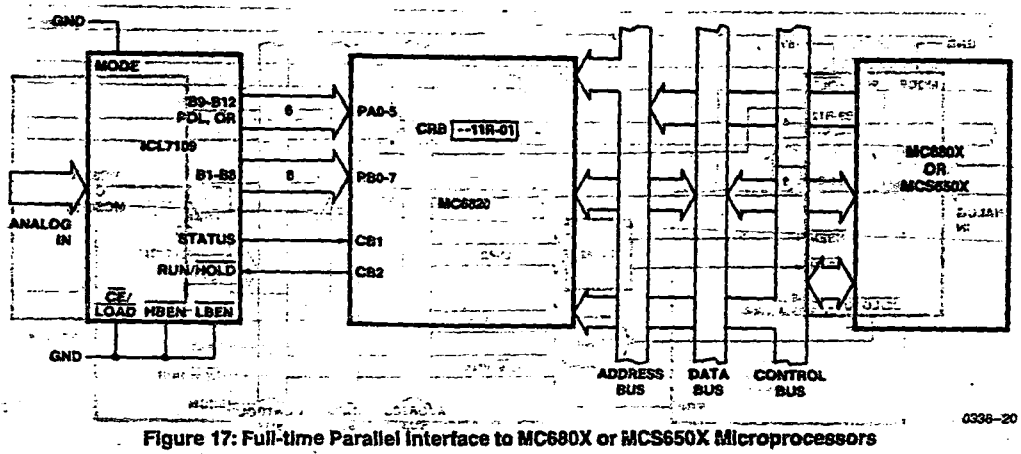


Figure 17: Full-time Parallel Interface to MC680X or MCS650X Microprocessors

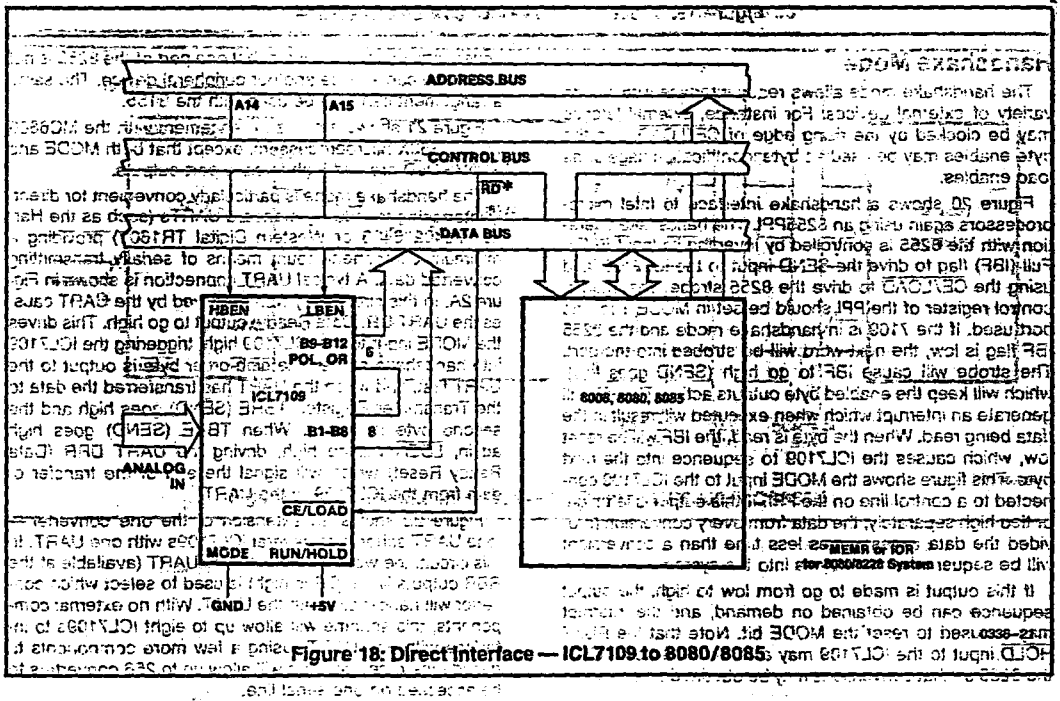


Figure 18: Direct Interface — ICL7109 to 8080/8085

3

NOTE: *All typical values have been characterized but are not tested.*

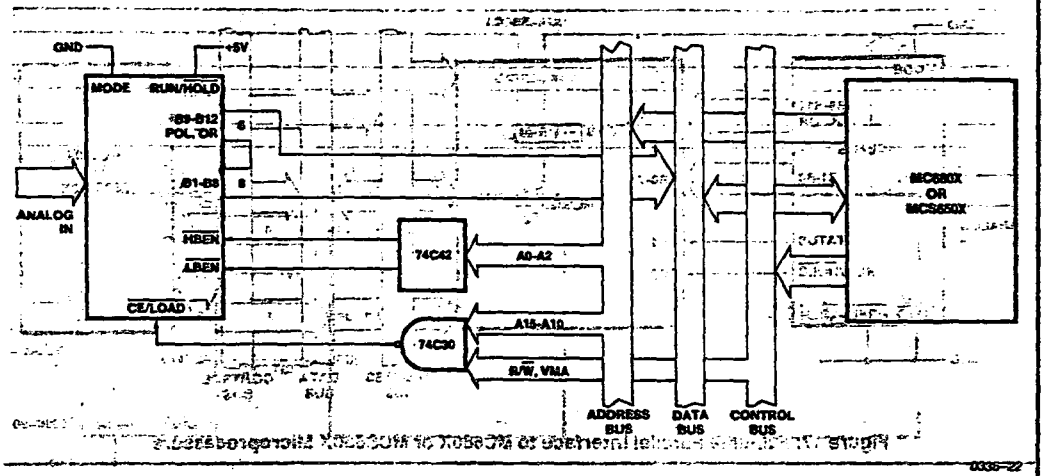


Figure 19: Direct ICL7109 — MC680X Bus Interface

Handshake Mode

The handshake mode allows ready interface with a wide variety of external devices. For instance, external latches may be clocked by the rising edge of CE/LOAD, and the byte enables may be used as byte identification flags or as load enables.

Figure 20 shows a handshake interface to Intel microprocessors again using an 8255PPI. The handshake operation with the 8255 is controlled by inverting its Input Buffer Full (IBF) flag to drive the SEND input to the ICL7109, and using the $\overline{CE/LOAD}$ to drive the 8255 strobe. The internal control register of the PPI should be set in MODE 1 for the port used. If the 7109 is in handshake mode and the 8255 IBF flag is low, the next word will be strobed into the port. The strobe will cause IBF to go high (SEND goes low), which will keep the enabled byte outputs active. The PPI will generate an interrupt which when executed will result in the data being read. When the byte is read, the IBF will be reset low, which causes the ICL7109 to sequence into the next byte. This figure shows the MODE input to the ICL7109 connected to a control line on the PPI. If this output is left high, or tied high separately, the data from every conversion (provided the data access takes less time than a conversion) will be sequenced in two bytes into the system.

If this output is made to go from low to high, the output sequence can be obtained on demand, and the interrupt may be used to reset the MODE bit. Note that the RUN/$\overline{HOLD}$ input to the ICL7109 may also be driven by a bit of the 8255 so that conversions may be obtained on command under software control. Note that one port of the 8255 is not caused, and can service another peripheral device. The same arrangement can also be used with the 8155.

Figure 21 shows a similar arrangement with the MC6800 or MCS650X microprocessors, except that both MODE and RUN/$\overline{HOLD}$ are tied high to save port outputs.

The handshake mode is particularly convenient for directly interfacing to industry standard UARTs (such as the Harris IM6402/6403 or Western Digital TR1602) providing a minimum component count means of serially transmitting converted data. A typical UART connection is shown in Figure 2A. In this circuit, any word received by the UART causes the UART DR (Data Ready) output to go high. This drives the MODE input to the ICL7109 high, triggering the ICL7109 into handshake mode. The high order byte is output to the UART first, and when the UART has transferred the data to the Transmitter Register, TBRE (SEND) goes high and the second byte is output. When TBRE (SEND) goes high again, $\overline{LBEN}$ will go high, driving the UART DRR (Data Ready Reset) which will signal the end of the transfer of data from the ICL7109 to the UART.

Figure 22 shows an extension of the one converter — one UART scheme to several ICL7109s with one UART. In this circuit, the word received by the UART (available at the RBR outputs when DR is high) is used to select which converter will handshake with the UART. With no external components, this scheme will allow up to eight ICL7109s to interface with one UART. Using a few more components to decode the received word will allow up to 256 converters to be accessed on one serial line.

NOTE: All typical values have been characterized but are not tested.

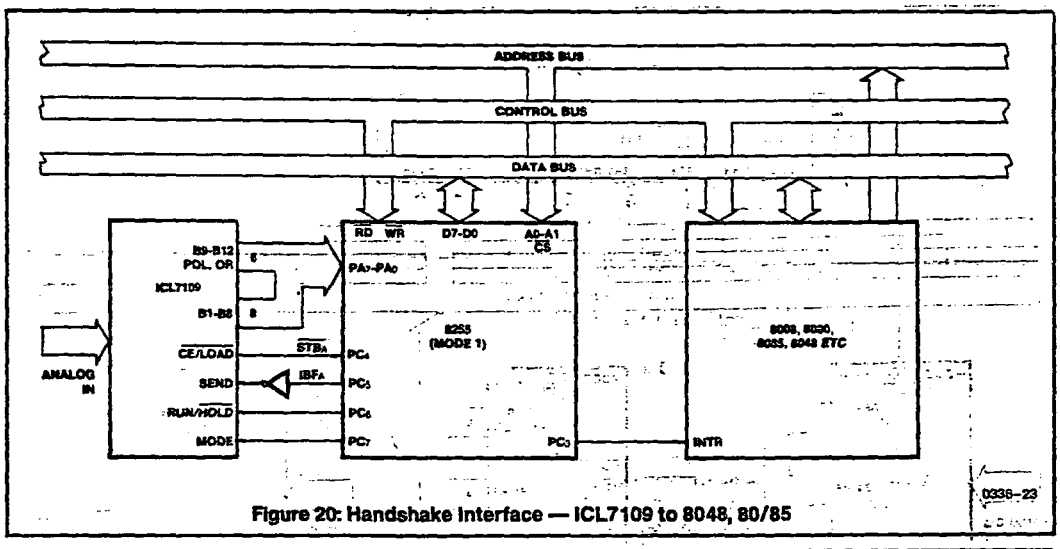


Figure 20: Handshake Interface — ICL7109 to 8048, 80/85

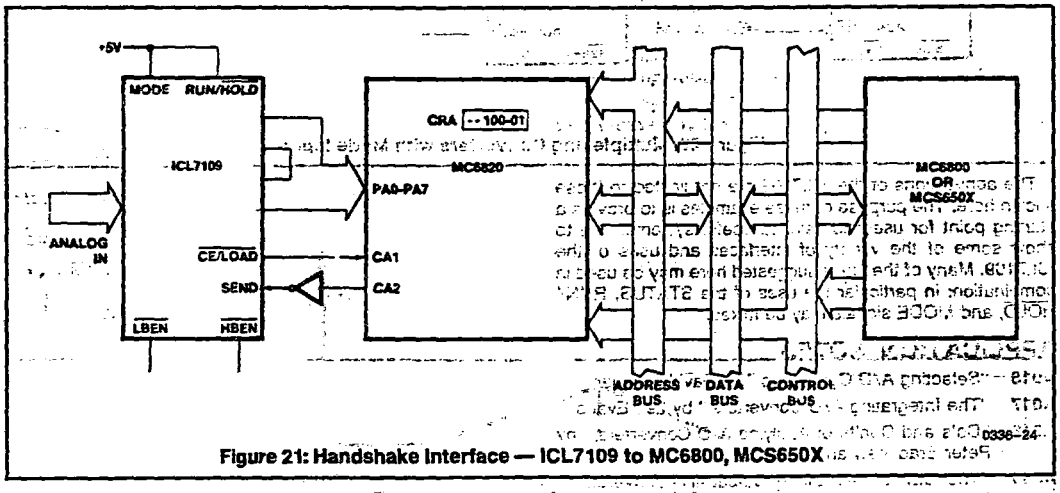


Figure 21: Handshake Interface — ICL7109 to MC6800, MCS650X

3

NOTE: All typical values have been characterized but are not tested.

ICL7109
INTEGRATING A/D CONVERTER
EQUATIONS

Oscillator Frequency
$f_{OSC} = 0.45/RC$,
$C_{OSC} > 50$ pF; $R_{OSC} > 50$ kΩ
f_{OSC} typ. = 60 kHz
or
f_{OSC} = 3.58 MHz Crystal

Oscillator Period
$t_{OSC} = RC/0.45$
$t_{OSC} = 1/3.58$ MHz (Crystal)

Integration Clock Frequency
$f_{CLOCK} = f_{OSC}$ (RC Mode)
$f_{CLOCK} = f_{OSC}/58$ (Crystal)
$t_{CLOCK} = 1/f_{CLOCK}$

Integration Period
$t_{INT} = 2048 \times t_{CLOCK}$

60/50 Hz Rejection Criterion
$t_{INT}/t_{60\ Hz}$ or $t_{INT}/t_{50\ Hz}$ = Integer

Optimum Integration Current
$I_{INT} = 20.0\ \mu A$

Full Scale Analog Input Voltage
V_{INFS} Typically = 200 mV or 2.0V

Integrate Resistor

$$R_{INT} = \frac{V_{INFS}}{I_{INT}}$$

Integrate Capacitor

$$C_{INT} = \frac{(t_{INT})(I_{INT})}{V_{INT}}$$

Integrator Output Voltage Swing

$$V_{INT} = \frac{(t_{INT})(I_{INT})}{C_{INT}}$$

V_{INT} Maximum Swing:
$(V^- + 0.5V) < V_{INT} < (V^+ - 0.5V)$
V_{INT} Typically = 2.0V

Display Count

$$COUNT = 2048 \times \frac{V_{IN}}{V_{REF}}$$

Conversion Cycle
$t_{CYC} = t_{CLOCK} \times 8192$
(In Free-Run Mode, Run/HOLD = 1)
when f_{CLOCK} = 60 kHz, t_{CYC} = 133 ms

Common Mode Input Voltage
$(V^- + 1.0V) < V_{IN} < (V^+ - 0.5V)$

Auto Zero Capacitor
$0.01\ \mu F < C_{AZ} < 1.0\ \mu F$

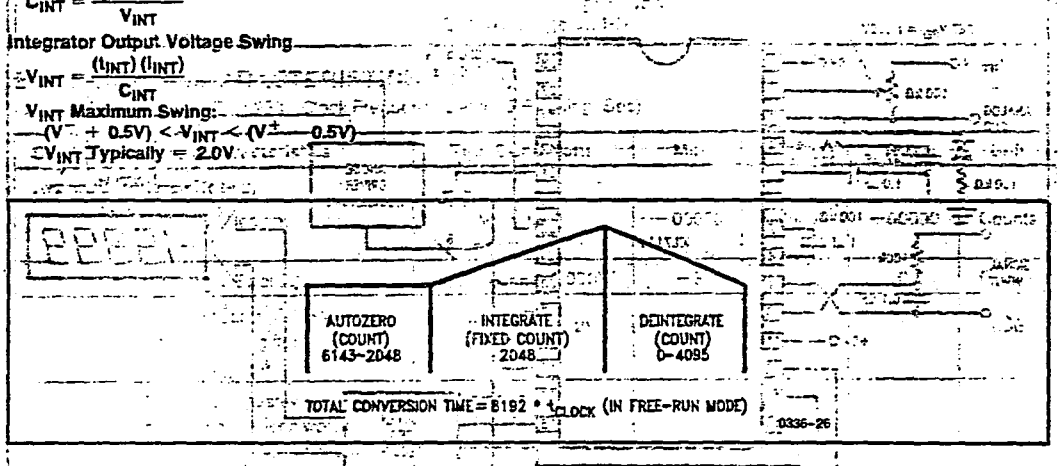
Reference Capacitor
$0.1\ \mu F < C_{REF} < 1.0\ \mu F$

V_{REF}
Biased between V^+ and V^-
$V_{REF} = V^+ - 2.8V$
Regulation lost when V^+ to $V^- < 6.4V$
If V_{REF} is not used, float output pin.

Power Supply: Dual ±5.0V
V^+ = +5.0 to GND
V^- = −5.0 to GND

Output Type:
Binary Amplitude with Polarity and Overrange Bits

Tips: Always tie TEST pin HIGH.
Don't leave any inputs floating.



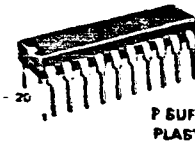
NOTE: All typical values have been characterized but are not tested.

MC145442
MC145443

2

Advance Information

Single Chip 300 Baud Modem



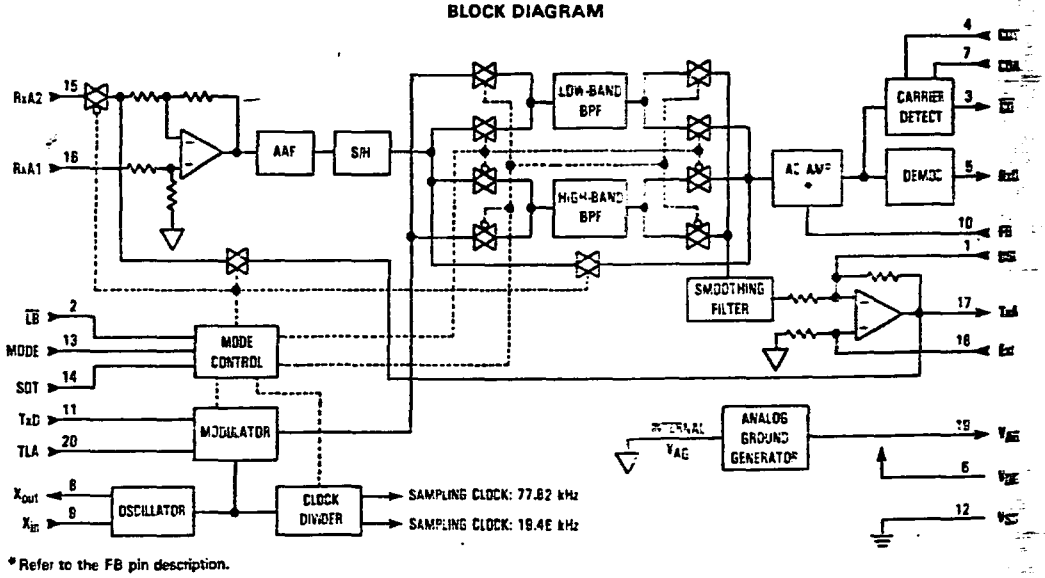
P SUFFIX
PLASTIC
CASE 738

DW SUFFIX
SOIC
CASE 751D

The MC145442 and MC145443 silicon-gate CMOS single-chip low-speed modems contain a complete frequency shift keying (FSK) modulator, demodulator, and filter. These devices are compatible with CCITT V.21 (MC145442) and Bell 103 (MC145443) specifications. Both devices provide full-duplex or half-duplex 300 baud data communicaton over a pair of telephone lines. They also include a carrier detect circuit for the demodulator section and a duplexer circuit for direct operation on a telephone line through a simple transformer.

- MC145442 Compatible with CCITT V.21
- MC145443 Compatible with Bell 103
- Low-Band and High-Band Bandpass Filters On-Chip
- Simplex, Half-Duplex, and Full-Duplex Operation
- Originate and Answer Mode
- Analog Loopback Configuration for Self Test
- Hybrid Network Function On-Chip
- Carrier Detect Circuit On-Chip
- Adjustable Transmit Level and $\overline{CD}$ Delay Timing
- On-Chip Crystal Oscillator (3.579 MHz)
- Single +5 Volt Power Supply Operation
- Internal Mid-Supply Generator
- Power-Down Mode
- Pin Compatible with MM74HC943
- Capable of Driving −9 dBm into a 600-Ohm Load

BLOCK DIAGRAM



*Refer to the FB pin description.

This document contains information on a new product. Specifications and information herein are subject to change without notice.

442
443

P SUFFIX
PLASTIC
CASE 738

/ SUFFIX
SOIC
.SE 751D

2

ABSOLUTE MAXIMUM RATINGS (Voltages Referenced to V_{SS})

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.5 to 7.0	V
DC Input Voltage	V_{in}	−0.5 to V_{DD}+0.5	V
DC Output Voltage	V_{out}	−0.5 to V_{DD}+0.5	V
Clamp Diode Current, per Pin	I_{IK}, I_{OK}	±20	mA
DC Output Current, per Pin	I_{out}	±28	mA
Power Dissipation	PD	500	mW
Operating Temperature Range	T_A	−40 to 85	°C
Storage Temperature Range	T_{stg}	−65 to 150	°C

This device contains circuitry to protect against damage due to high static voltages or electric fields, however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$.

Unused inputs must always be tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

RECOMMENDED OPERATING CONDITIONS

Parameter	Symbol	Min	Max	Unit
Supply Voltage	V_{DD}	4.5	5.5	V
DC Input or Output Voltage	V_{in}, V_{out}	0	V_{DD}	V
Input Rise or Fall Time	t_r, t_f	—	500	ns
Crystal Frequency*	$f_{crystal}$	3.2	5.0	MHz

*Changing the crystal frequency from 3.579 MHz will change the output frequencies. The change in output frequency will be proportional to the change in crystal frequency.

DC ELECTRICAL CHARACTERISTICS (V_{DD} = 5.0 V ± 10%, T_A = −40 to 85°C)

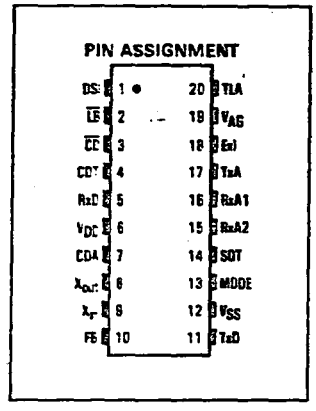
Characteristic		Symbol	Min	Typ	Max	Unit
High-Level Input Voltage	$\overline{LB}$	V_{IH}	V_{DD} − 0.8	—	—	V
	X_{in}, TxD, Mode, SQT		3.15	—	—	
Low-Level Input Voltage	$\overline{LB}$	V_{IL}	—	—	0.8	V
	X_{in}, TxD, Mode, SQT		—	—	1.1	
High-Level Output Voltage		V_{OH}				V
I_{OH} = 20 μA	$\overline{CD}$, RxD		V_{DD} − 0.1	—	—	
I_{OH} = 2 mA	$\overline{CD}$, RxD		3.7	—	—	
I_{OH} = 20 μA	X_{out}		—	V_{DD} − 0.05	—	
Low-Level Output Voltage		V_{OL}				V
I_{OL} = 20 μA	$\overline{CD}$, RxD		—	—	0.1	
I_{OL} = 2 mA	$\overline{CD}$, RxD		—	—	0.4	
I_{OL} = 20 μA	X_{out}		—	0.05	—	
Input Current	$\overline{LB}$, TxD, Mode, SQT	I_{in}	—	—	±1.0	μA
	RxA1, RxA2		—	10	±12	
	X_{in}		—	—	±10	
Quiescent Supply Current		I_{DD}	—	7	10	mA
X_{in} or $f_{crystal}$ = 3.579 MHz						
Power-Down Supply Current			—	200	300	μA
Input Capacitance	X_{in}	C_{in}	—	10	—	pF
	All Other Inputs		—	—	10	
V_{AG} Output Voltage (I_O = ± 10 μA)		V_{AG}	2.4	2.5	2.6	V
CDA Output Voltage (I_O = ± 10 μA)		V_{CDA}	1.1	1.2	1.3	V
Line Driver Feedback Resistor		R_f	10	20	30	kΩ

4 CDT
7 CDA
3 $\overline{CD}$
5 RxD
10 FB
1 DSI
17 TxA
18 ExI
16 V_{AG}
6 V_{DD}
12 V_{SS}

2

AC ELECTRICAL CHARACTERISTICS (V_{DD} = 5.0 V ± 10%, T_A = – 40 to 85°C, Crystal Frequency = 3.579 MHz ± 0.1%; See Figure 1)

Characteristic		Min	Typ	Max	Unit
TRANSMITTER					
Power Output on TxA RL = 1.2 kΩ, R_{TLA} = ∞ RL = 1.2 kΩ, R_{TLA} = 5.5 kΩ		 – 13 – 10	 – 12 – 9	 – 11 – 8	dBm
Second Harmonic Power RL = 1.2 kΩ		–	– 56	–	dBm
RECEIVE FILTER AND HYBRID					
Hybrid Input Impedance RxA1, RxA2		40	50	–	kΩ
FB Output Impedance		–	16	–	kΩ
Adjacent Channel Rejection		– 48	–	–	dBm
DEMODULATOR					
Receive Carrier Amplitude		– 48	–	– 12	dBm
Dynamic Range		–	36	–	dB
Bit Jitter (S/N = 30 dB, Input = – 38 dBm, Bit Rate = 300 baud)		–	100	–	μs
Bit Bias		–	5	–	%
Carrier Detect Threshold (CDA = 1.2 V or CDA grounded through a 0.1 μF capacitor)	On to Off Off to On	– –	– 44 – 47	– –	dBm



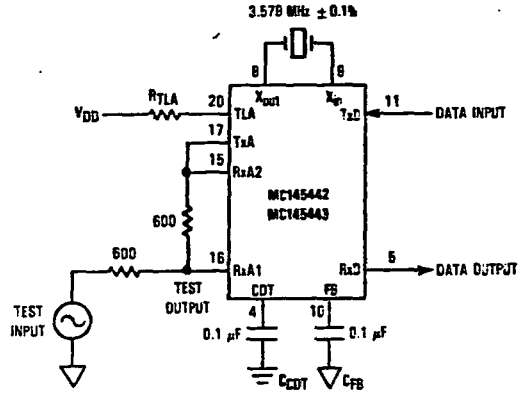


Figure 1. AC Characteristics Evaluation Circuit

PIN DESCRIPTIONS

V_{DD} – POSITIVE POWER SUPPLY (PIN 6)

This pin is normally tied to 5.0 V.

V_{SS} – NEGATIVE POWER SUPPLY (PIN 12)

This pin is normally tied to 0 V.

V_{AG} – ANALOG GROUND (PIN 19)

Analog ground is internally biased to $(V_{DD} - V_{SS})/2$. This pin must be decoupled by a capacitor from V_{AG} to V_{SS} and a capacitor from V_{AG} to V_{DD}. Analog ground is the common bias line used in the switched capacitor filters, limiter, and slicer in the demodulation circuitry.

TLA – TRANSMIT LEVEL ADJUST (PIN 20)

This pin is used to adjust the transmit level. Transmit level adjustment range is typically from – 12 dBm to – 9 dBm. (See Applications Information.)

TxD – TRANSMIT DATA (PIN 11)

Binary information is input to the transmit data pin. Data entered for transmission is modulated using FSK techniques. A logic high input level represents a mark and a logic low represents a space. (See Table 1.)

Table 1. Bell 103 and CCITT V.21 Frequency Characteristics

Bell 103 (MC145443)				
Data	Originate Mode		Answer Mode	
	Transmit	Receive	Transmit	Receive
Space	1070 Hz	2025 Hz	2025 Hz	1070 Hz
Mark	1270 Hz	2225 Hz	2225 Hz	1270 Hz
CCITT V.21 (MC145442)				
Data	Originate Mode		Answer Mode	
	Transmit	Receive	Transmit	Receive
Space	1180 Hz	1850 Hz	1850 Hz	1180 Hz
Mark	980 Hz	1650 Hz	1650 Hz	980 Hz

NOTE: Actual frequencies may be ±5 Hz assuming a 3.579545 MHz crystal is used.

TxA—TRANSMIT CARRIER (PIN 17)

This is the output of the line driver amplifier. The transmit carrier is the digitally synthesized sine wave output of the modulator derived from a crystal oscillator reference. When a 3.579 MHz crystal is used the frequency outputs shown in Table 1 apply. (See Applications Information.)

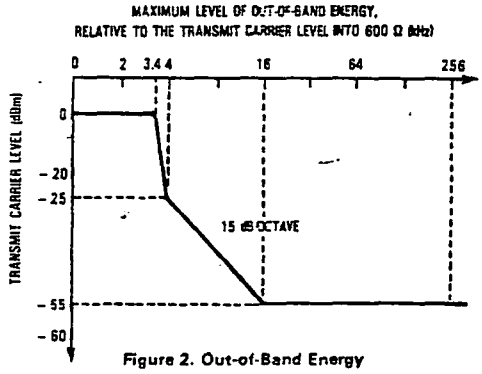


Figure 2. Out-of-Band Energy

EXI—EXTERNAL INPUT (PIN 18)

The external input is the noninverting input to the line driver. It is provided to combine an auxiliary audio signal or speech signal to the phone line using the line driver. This pin should be connected to V_{AG} if not used. The average level must be the same as V_{AG} to maintain proper operation. (See Applications Information.)

DSI—DRIVER SUMMING INPUT (PIN 1)

The driver summing input may be used to connect an external signal, such as a DTMF dialer, to the phone line. A series resistor, R_{DSI}, is needed to define the voltage gain A_V (see Applications Information and Figure 6). When applying a signal to the DSI pin, the modulator should be squelched by bringing SQT (pin 14) to a logic high level. The voltage gain, A_V, is calculated by the formula $A_V = -R_f/R_{DSI}$ (where $R_f = 20$ kΩ). For example, a 20 kΩ resistor for R_{DSI} will provide unity gain ($A_V = -20$ kΩ/20 kΩ = −1). This pin MUST be left OPEN if not used.

RxD—RECEIVE DATA (PIN 5)

The receive data output pin presents the digital binary data resulting from the demodulation of the receive carrier. If no carrier is present, CD high, the receive data output (RxD) is clamped high.

RxA2, RxA1—RECEIVE CARRIER (PINS 15, 16)

The receive carrier is the FSK input to the demodulator through the receive band-pass filter. RxA1 is the noninverting input and RxA2 is the inverting input of the receive hybrid (duplexer) operational amplifier.

$\overline{LB}$—ANALOG LOOPBACK (PIN 2)

When a high level is applied to this pin (SQT must be low), the analog loopback test is enabled. The analog loopback test connects the TxA pin to the RxA2 pin and the RxA1 to analog ground. In loopback, the demodulator frequencies are switched to the modulation frequencies for the selected mode. (See Tables 1 and 2 and Figures 4c and 4d.)

When $\overline{LB}$ is connected to analog ground (V_{AG}), the modulator generates an echo cancellation tone of 2100 Hz for MC145442 CCITT V.21 and 2225 Hz for MC145443 Bell 103 systems. For normal operation, this pin should be at a logic low level (V_{SS}).

The power-down mode is enabled when both $\overline{LB}$ and SQT are connected to a logic high level. (See Table 2.)

Table 2. Functional Table

MODE Pin 13	SQT Pin 14	$\overline{LB}$ Pin 2	Operating Mode
1	0	0	Originate Mode
0	0	0	Answer Mode
X	0	V_{AG} (V_{DD}/2)	Echo Tone
X	0	1	Analog Loopback
X	1	0	Squelch Mode
X	1	V_{AG} (V_{DD}/2)	Squelch Mode
X	1	1	Power Down

MODE—MODE (PIN 13)

This input selects the pair of transmit and receive frequencies used during modulation and demodulation. When a logic high level is placed on this input, originate (Bell) or channel 1 (CCITT) is selected. When a low level is placed on this input, answer (Bell) or channel 2 (CCITT) is selected. (See Tables 1 and 2 and Figure 4.)

CDT—CARRIER DETECT TIMING (PIN 4)

A capacitor on this pin to V_{SS} sets the amount of time the carrier must be present before $\overline{CD}$ goes low. (See Applications Information for the capacitor values.)

2

$\overline{CD}$—CARRIER DETECT OUTPUT (PIN 3)

This output is used to indicate when a carrier has been sensed by the carrier detect circuit. This output goes to a logic low level when a valid signal above the minimum threshold level (defined by CDA pin 7) is maintained on the input to the hybrid circuit longer than the response time (defined by CDT pin 4). This pin is held at the logic low level until the signal falls below the maximum threshold level for longer than the turn off time. (See Applications Information and Figure 5.)

CDA—CARRIER DETECT ADJUST (PIN 7)

An external voltage may be applied to this pin to adjust the carrier detect threshold. The threshold hysteresis is internally fixed at 3 dB. (See Applications Information.)

X_{out}, X_{in}—CRYSTAL OSCILLATOR (PINS 8, 9)

A crystal reference oscillator is formed when a 3.579 MHz crystal is connected between these two pins. X_{out} (pin 8) is the output of the oscillator circuit, and X_{in} (pin 9) is the input to the oscillator circuit. When using an external clock, apply the clock to the X_{in} (pin 9) pin and leave X_{out} (pin 8) open. An internal 10 mΩ resistor and internal capacitors, typically 10 pF on X_{in} and 16 pF on X_{out}, allow the crystal to be connected without any other external components. Printed circuit board layout should keep external stray capacitance to a minimum.

FB—FILTER BIAS (PIN 10)

This is the negative input to the ac amplifier. In normal operation, this pin is connected to analog ground through a 0.1 μF bypass capacitor in order to cancel the input offset voltage of the limiter. It has a nominal input impedance of 16 kΩ. (See Figure 3.)

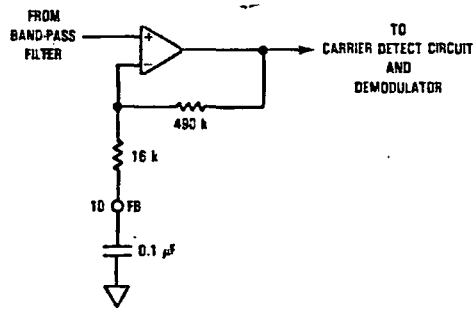


Figure 3. AC Amplifier Circuit

SQT—TRANSMIT SQUELCH (PIN 14)

When this input pin is at a logic high level, the modulator is disabled. The line driver remains active if $\overline{LB}$ is at a logic low level. (See Table 2.)

When both $\overline{LB}$ and SQT are connected to a logic high level, see Table 2, the entire chip is in a power down state and all circuitry except the crystal oscillator is disabled. Total power supply current decreases from 10 mA (maximum) to 300 μA (maximum).

GENERAL DESCRIPTION

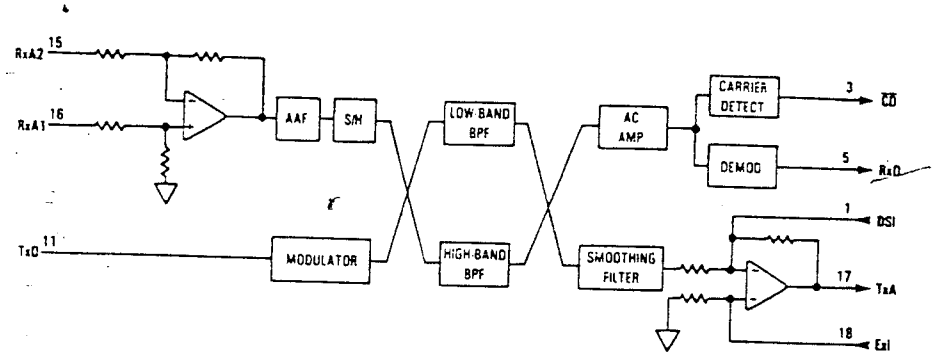
The MC145442 and MC145443 are full-duplex low-speed modems. They provide a 300 baud FSK signal for bidirectional data transmission over the telephone network. They can be operated in one of four basic configurations as determined by the state of MODE (pin 13) and $\overline{LB}$ (pin 2). The normal (non-loopback) and self test (loopback) modes in both answer and originate modes will be discussed.

For an originate or channel 1 mode, a logic high level is placed on MODE (pin 13) and a logic low level is placed on $\overline{LB}$ (pin 2). In this mode, transmit data is input on TxD, where it is converted to a FSK signal and routed through a low-band band-pass filter. The filtered output signal is then buffered by the Tx op-amp line driver, which is capable of driving −9 dBm onto a 600 Ω line. The receive signal is connected through a hybrid duplexer circuit on pins 15 and 16, RxA2 and RxA1. The signal then passes through the anti-aliasing filter, the sample-and-hold circuit, is switched into the high-band band-pass filter, and then switched into the ac amplifier circuit. The output of the ac amplifier circuit is routed to the demodulator circuit and demodulated. The resulting digital data is then output through RxD (pin 5). The carrier detect circuit receives its signal from the output of the ac amplifier circuit and goes low when the incoming signal is detected. (See Figure 4a.)

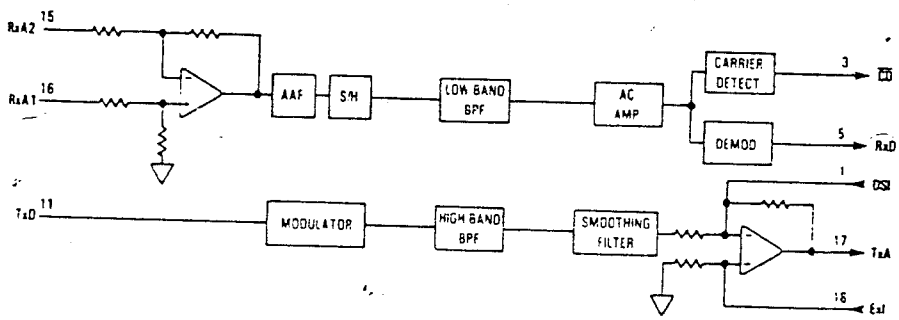
In the answer or channel 2 mode, a logic low level is placed on MODE (pin 13) and on $\overline{LB}$ (pin 2). In this mode, the data follows the same path except the FSK signal is routed to the high-band band-pass filter and the sample-and-hold signal is routed through the low-band band-pass filter. (See Figure 4b.)

In the analog loopback originate or channel 1 mode, a logic high level is placed on MODE (pin 13) and on $\overline{LB}$ (pin 2). This mode is used for a self check of the modulator, demodulator, and low-band pass-band filter circuit. The modulator side is configured exactly like the originate mode above except the line driver output (TxA pin 17) is switched to the negative input of the hybrid op-amp. The RxA2 input pin is open in this mode and the noninverting input of the hybrid circuit is connected to V_{AG}. The sample-and-hold output bypasses the filter so that the demodulator receives the modulated Tx data (see Figure 4c). This test checks all internal device components except the high-band band-pass filter which can be checked in the answer or channel 2 mode loopback test.

In the analog loopback answer or channel 2 mode, a logic low level is placed on MODE (pin 13) and a logic high level on $\overline{LB}$ (pin 2). This mode is used for a self check of the modulator, demodulator, and high-band pass-band filter circuit. This configuration is exactly like the originate loopback mode above, except the signal is routed through the high-band pass-band filter. (See Figure 4d.)

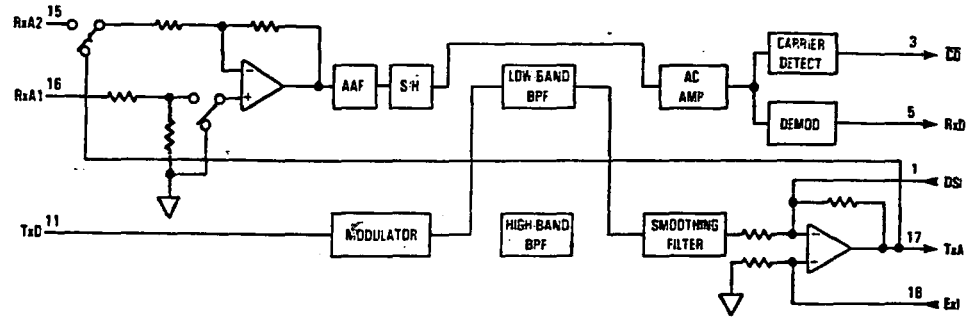


(a) ORIGINATE/CHANNEL 1 MODE (MODE = HIGH, $\overline{LB}$ = LOW)

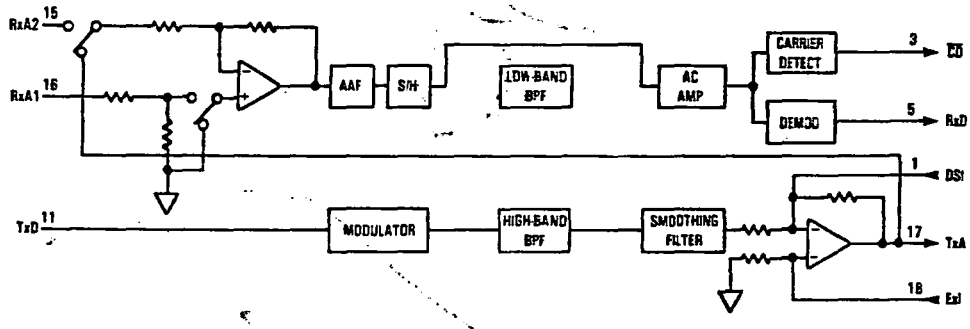


(b) ANSWER/CHANNEL 2 MODE (MODE = LOW, $\overline{LB}$ = LOW)

Figure 4. Basic Operating Modes



(c) ORIGINATE/CHANNEL 1 MODE AND ANALOG LOOPBACK STATE (MODE = HIGH, $\overline{LB}$ = HIGH)



(d) ANSWER/CHANNEL 2 MODE AND ANALOG LOOPBACK STATE (MODE = LOW, $\overline{LB}$ = HIGH)

Figure 4. Basic Operating Modes

APPLICATIONS INFORMATION

CARRIER DETECT TIMING ADJUSTMENT

The value of a capacitor, C_{CDT} at CDT (pin 4) determines how long a received modem signal must be present above the minimum threshold level before $\overline{CD}$ (pin 3) goes low. The C_{CDT} capacitor also determines how long the $\overline{CD}$ pin stays low after the received modem signal goes below the minimum threshold. The $\overline{CD}$ pin is used to distinguish a strong modem signal from random noise. The following equations show the relationship between t_{CDL}, the time in seconds required for $\overline{CD}$ to go low; t_{CDH}, the time in seconds required for $\overline{CD}$ to go high; and C_{CDT}, the capacitor value in μF.

Valid signal to $\overline{CD}$ response time: $t_{CDL} \approx 6.4 \times C_{CDT}$

Invalid signal to $\overline{CD}$ off time: $t_{CDH} \approx 0.54 \times C_{CDT}$

Example: $t_{CDL} \approx 6.4 \times 0.1\ \mu F \approx 0.64$ seconds

$t_{CDH} \approx 0.54 \times 0.1\ \mu F \approx 0.054$ seconds

CARRIER DETECT THRESHOLD ADJUSTMENT

The carrier detect threshold is set by internal resistors to activate $\overline{CD}$ with a typical −44 dBm (into 600 Ω) signal and deactivate $\overline{CD}$ with a typical −47 dBm signal applied to the input of the hybrid circuit. The carrier detect threshold level can be adjusted by applying an external voltage on CDA (pin 7). The following equations may be used to find the CDA voltage required for a given threshold voltage. (V_{on} and V_{off} are in volts RMS.)

$$V_{CDA} = 244 \times V_{on}$$

$$V_{CDA} = 345 \times V_{off}$$

Example (internally set)

V_{on} = 4.9 mV ≈ −44 dBm: $V_{CDA} = 244 \times 4.9$ mV = 1.2 V

V_{off} = 3.5 mV ≈ −47 dBm: $V_{CDA} = 345 \times 3.5$ mV = 1.2 V

Example (externally set):

V_{on} = 7.7 mV ≈ −40 dBm: $V_{CDA} = 244 \times 7.7$ mV = 1.9 V

V_{off} = 5.4 mV ≈ −43 dBm: $V_{CDA} = 345 \times 5.4$ mV = 1.9 V

The CDA pin has an approximate Thevenin equivalent voltage of 1.2 V and an output impedance of 100 kΩ. When using the internal 1.2 volt reference a 0.1 μF capacitor should be connected between this pin and V_{SS}. (See Figure 5.)

TRANSMIT LEVEL ADJUSTMENT

The power output at TxA (pin 17) is determined by the value of resistor R_{TLA} that is connected between TLA (pin 20) to V_{DD} (pin 6). Table 3 shows the R_{TLA} values and the corresponding power output for a 600 Ω load. The voltage at TxA is twice the value of that at ring and tip because TxA feeds the signal through a 600 Ω resistor R_{Tx} to a 600 Ω line transformer. (See Figure 7.) When choosing resistor R_{TLA}, keep in mind that −9 dBm is the maximum output level allowed from a modem onto the telephone line (in the U.S.). In addition, keep in mind that maximizing the power output from the modem optimizes the signal-to-noise ratio, improving accurate data transmission.

Table 3. Transmit Level Adjust

Output Transmit Level (Typical into 600 Ω)	R_{TLA}
−12 dBm	∞
−11 dBm	19.8 kΩ
−10 dBm	9.2 kΩ
−9 dBm	5.5 kΩ

THE LINE DRIVER

The line driver is a power amplifier used for driving the telephone line. Both the inverting and noninverting input to the line driver are available for transmitting externally generated tones.

Exl (pin 18) is the noninverting input to the line driver and gives a fixed gain of 2 (R_i = 50 kΩ). The average signal level must be the same as V_{AG} to maintain proper operation. This pin should be connected to V_{AG} if not used.

The driver summing input (DSI, pin 1) may be used to connect an external signal, such as a DTMF dialer, to the phone line. When applying a signal to the DSI pin, the modulator should be squelched by bringing SQT (pin 14) to a logic high level. DSI MUST be left OPEN if not used.

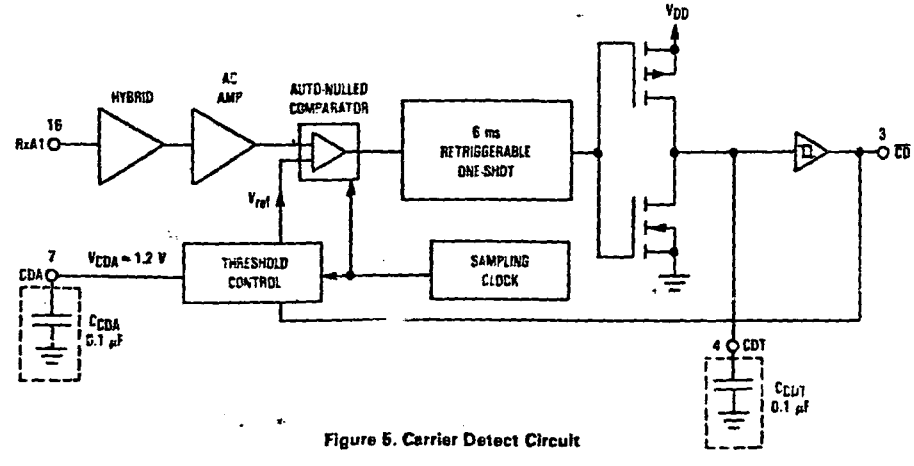


Figure 5. Carrier Detect Circuit

2

In addition, the DSI pin is the inverting side of the line driver and allows adjustable gain with a series resistor R_{DSI}. (See Figure 6.) The voltage gain, A_V, is determined by the equation:

$$A_V = -\frac{R_f}{R_{DSI}}$$

where $R_f = 20\ k\Omega$.

Example: A resistor value of 20 kΩ for R_{DSI} will provide unity gain.

$$A_V = -(20\ k\Omega/20\ k\Omega) = -1$$

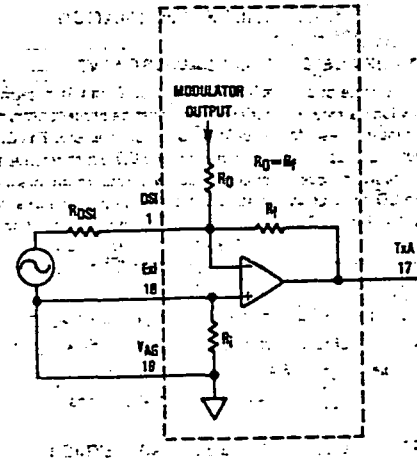
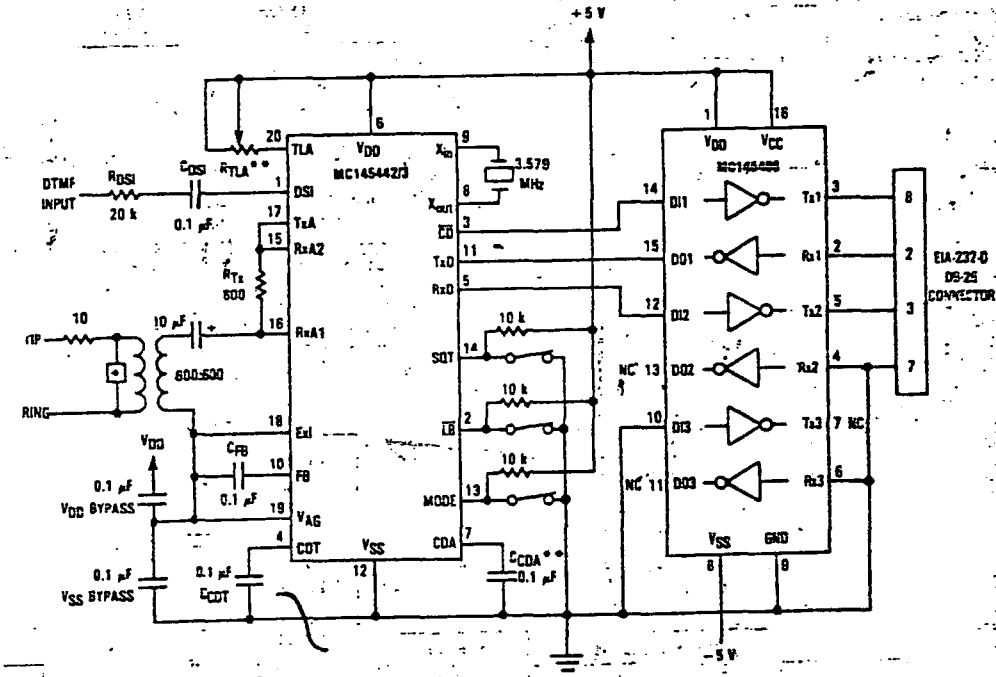


Figure 6. Line Driver Using the DSI Input



* Line protection circuit
** Refer to the applications information for values of C_{CDA} and R_{TLA}

Figure 7. Typical MC145442/MC145443 Applications Circuit