เอกสารอ้างอิง

1. P.E. Green, Jr., "Communication Milestones and Prophecies," IEEE Comm. Mag., Vol-22, No.5, pp. 49-63, May 1984.
2. Edward M. Dunlap, Jr., and David C. Rife,"Personal Computer Communication Via Telephone Facillities," IEEE Journal Selected Areas in Comm., Vol SAC-3, No.3, pp.449-456, May 1985.
3. Robert Cole, Computer Communications, Macmillan Press Ltd., 1982.
4. Alan J. Weissberger, Data Communications Handbook, Signetics Corp., Oct. 1977.
5. J.R. DAVEY, "Modem," Prod. of IEEE , Vol.60, No.11, Nov.1972.
6. Edwin B. Parker, "Cost-Effective Data Communications for Personal Application Using Micro Earth Stations," IEEE Jour.Selected Areas in Comm., Vol SAC-3, No. 3, pp. 449-456, May 1985.
7. George R. Cooper,Clare D. Mc Gillem, Modern Communication and Spread Spectrum, McGraw-Hill 1986.
8. John Bellamy, Digital Telephony, John Wiley & Sons, Inc. 1982.
9. K. Sam Shanmugan, Digital and Analog Communications System, John Wiley & Son, USA, 1979.
10. EIA Standard RS-232-C, "Interface Between Data Terminal Equipment and Data Communication Equipment Employing Binary Data Interfange," Oct. 1969, June 1981.
11. EIA Standard RS-363, " Standard for Specifying Signal Quality for Transmitting and Receiving Data processing Terminal Equipment Using Serial Data Transmission at the Interface with non-synchronous Data Communication Equipment,"May 1969.

12. EIA Standard RS-334-A, "Signal Quality at Interface between Data Terminal Equipment and Synchronous Data Communication Equipment for Serial Data Transmission, " Aug. 1981.

13. CCITT V. Series Recommendations(Yellow Book 1981).

14. CCITT V.25 bis Recommendations,"Automatic Calling and/or Answering Equipment on the General Switched Telephone Network (GSTN) using The 100 Series," (Red Book 1985).

15. Ralgh Glasgal, Techniques in Data Communications, Artech House Inc., 1983.

16. Paul E. Green,Jr., Computer Network Architectures and Protocols, Plenum New Yerk, May 1983.

17. F. Da. Cruz and Bk. Catchings, "Kermit: A File Transfer Protocol for Universities Part 1: Design Considerations and Specifications," Byte, pp. 255,256,259,260,262,264, 268,270,272,274,276,278, June 1984.

18. F. Da. Cruz and Bk. Catchings, "Kermit: A File Transfer Protocol for Universities Part 2: States and Transitions, Heuristic Rules, and Examples, " Byte, pp. 143-145, 400-402, July 1984.

19. Crosstalk XVI Program, Microstaf, Inc.,Norcross, GA, 1983.

20. Intellimodem XL User Guide, Appendix C, Bussiness Computer Corp. (Bizcomp)

21. Telecommunications Device Data, pp. 2.77-2.99, pp. 2.136-2.139, Motorola Inc.

22. Peter Single, "Optimize the Hybrid Interface to Increase Modem Dynamic Range," EDN Mag., Oct. 1984.

23. Raj Chirayil and P. Ehlig, "IC-Modem/Phone-Network Interfaces Meet FCC Isolation, Protection Rules," EDN Mag., Mar.1984.

24. EXAR Modem Design Handbook, 3 rd printing, EXAR Corp., Sunnyvale, C.A., U.S.A., Dec. 1984.

25. Larry E. Jordan, Bruce Churchill, Communications and Networking for the IBM PC, pp, 163-169, Prentice-Hall Publishing and Communication Company, Bowie Maryland 20175, 1983.

26. Turbo Pascal Version 3.0 Reference Manaul, 3 rd ed., Borland International Inc., 1985.

27. Dorothy Elizabeth Robling Denning, Cryptography and Data Security, pp. 1-10,pp. 90-97, Addison-Wesley Publishing Company, 1982.

28. ANSI X3.92-1981, "Data Encryption Algorithm," 1981.

29. Technical Reference IBM Personal Computer XT Hardware Reference Library, pp. 1:215-1:245, D78, A21-A24.

30. ANSI X3.4-1977, "Code for Information Interchange," 1977.

31. Andrew S.Tanenbaum, Computer Networks, pp. 396-400, Prentice/Hall, 1981.

32. ดร. เอกชัย ลีลารัศมี, คู่มือการใช้ "เล็ก 5.0", ภาควิชาวิศวกรรมไฟฟ้า จุฬาลงกรณ์มหาวิทยาลัย, 2530

33. Kamilo Feher, Digital Communications Satellite/Earth Station Engineering, Prentice-Hall Inc., Egnlewood Cliffs, 1983.

34. Maurice Bellanger, Digital Processing of Signals Theory and Practice, John Wiley & Sons, 2 nd ed., 1985.

35. Walter C. Johnson, Transmission lines and Networks, McGraw. Hill, 15 th Printing 1981.

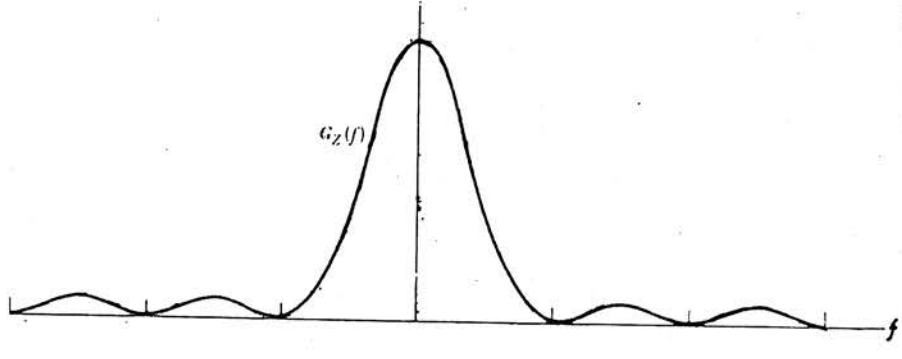
ภาคผนวก

ภาคผนวก ก.

องค์ประกอบทางความถี่ของข้อมูล NRZ และข้อมูล Scrambling

ก.1 องค์ประกอบทางความถี่ของข้อมูล NRZ

สัญญาณพัลส์สี่เหลี่ยมจะมีความหนาแน่นกำลังงานสเปคตรัม แสดงดังในรูป ก.1 องค์ประกอบทางความถี่ของสัญญาณพัลส์สี่เหลี่ยมมีขนาดกว้างมากหรือมีแบนด์วิดท์ไม่จำกัด และความหนาแน่นกำลังงานสเปคตรัมมีความหนาแน่นมากที่ความถี่ศูนย์ ในกรณีสมมติที่การรับส่งข้อมูลมีอัตราการส่ง "0" และ "1" เท่ากัน ความหนาแน่นกำลังงงานสเปคตรัมของสัญญาณข้อมูล NRZ จะเป็นไปตามสมการ (ก.1)



รูป ก.1 ความหนาแน่นกำลังงานสเปคตรัมของสัญญาณพัลส์สี่เหลี่ยม

$$w_s(f) = 2A^2T_s\left(\frac{\sin \pi f T_s}{\pi f T_s}\right)^2 \qquad (ก.1)$$

โดย T_s= duration ของสัญญาณข้อมูล

ในกรณีของการรับส่งข้อมูลของอุปกรณ์รับส่งข้อมูลปลายทาง อัตราการรับส่ง "0" และ "1" จะไม่เท่ากัน จึงทำให้เกิดสัญญาณดีซี และความถี่ขาดช่วงขึ้น [8] ในกรณีที่ทำการรับส่งข้อมูลที่อัตราเร็วสูงขึ้น ค่า duration T_s ลดลงทำให้เกิดองค์ประกอบทางความถี่สูงเพิ่มมาก

ขึ้น เมื่อทำการรับส่งข้อมูลโดยใช้สายส่งที่ไม่มีวงจร Repeater เพื่อทำการชดเชยค่าความลดทอนแล้ว ระยะทางในการรับส่งข้อมูลจึงได้ไม่ไกลนัก เช่นที่กำหนดในมาตรฐาน RS-232-C การทำ Line Coding เป็นวิธีการหนึ่งที่ทำเพื่อให้องค์ประกอบความถี่สูงลดลง ทำให้สามารถที่ทำการส่งข้อมูลได้ที่อัตราสูงขึ้น

ก.2 องค์ประกอบทางความถี่ของข้อมูล Scrambling

เป็นวิธีการเพื่อทำให้ปริมาณการรับส่งข้อมูล "0" และ "1" เกิดขึ้นเท่ากัน เพื่อให้ภาครับสามารถทำการกำเนิดสัญญาณนาฬิกาได้ถูกต้อง เนื่องจากไม่พบปัญหา dc Wander [8] อันเกิดจากการส่งสัญญาณ "0" หรือ "1" นานเกินไป นอกจากนั้นยังเป็นการแก้ปัญหาการเกิด Discrete DC Component ได้ ลักษณะของ psuedo data ทำให้สัญญาณข้อมูลที่ทำการรับส่ง มีความหนาแน่นของกำลังงานสเปคตรัม ดังสมการ (ก.2)

$$w_{PR}(f)=\frac{L+1}{L^2}\left(\frac{\sin \pi f T_b}{\pi f T_b}\right)^2 \sum_{\substack{n=-\infty \\ n\neq 0}}^{\infty} \delta\left(f-\frac{n}{LT_b}\right)+\frac{1}{L^2}\delta(f) \qquad \text{(ก.2)}$$

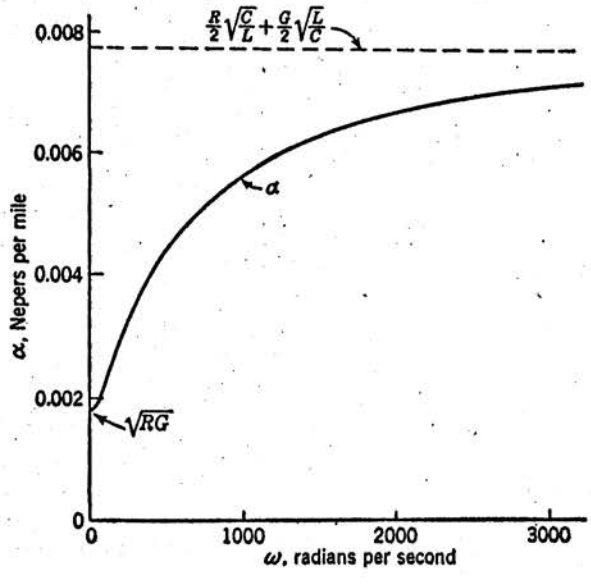
L เป็นความยาวของอนุกรมกำลังที่ใช้ใน Psuedo Data

ลักษณะของสัญญาณข้อมูลที่ถูกสแครมเบอร์ดังกล่าว เมื่อผ่านช่องสัญญาณกรองผ่านแถบที่เหมาะสม จะได้ความน่าจะเป็นในการกระจายเป็นแบบปกติ (Normal Distribution) [33][34] ด้วยเหตุนี้จึงจำเป็นที่ต้องเลือกอนุกรมกำลังเพื่อให้การกระจายเป็นแบบปกติ เช่น สแครมเบอร์ ตามมาตรฐาน V.52 ที่ใช้กับโมเดม V.26 bis ที่คลื่นพาห์ 1800 HZ และมีแบนด์วิดท์ 2400 HZ (2 เท่าของอัตราการรับส่งข้อมูล)

ภาคผนวก ข.

1. คุณสมบัติของสายส่งแบบ Twist pair

คุณสมบัติของสายส่งแบบ Twist pair [35] ทำให้เกิดการลดทอนขนาดเป็นสัดส่วนกับความถี่ของสัญญาณที่ใช้ในอัตรา ค่าสมบัติการลดทอนทางขนาดที่เกิดจากคุณสมบัติของสายส่งแบบ Twist pair แสดงดังในรูป ข.1 จะเห็นว่าเราสามารถรับส่งข้อมูลได้ที่อัตราเร็วสูงมาก แต่จะมีปัญหาว่าองค์ประกอบทางด้านความถี่สูงของสัญญาณข้อมูลจะผ่านไปได้น้อย ดังนั้นในการรับส่งข้อมูลที่ความถี่สูงของสายส่ง ลักษณะสมบัติของสายส่งจึงมีแบนด์วิดท์ขนาดไม่จำกัด สำหรับองค์ประกอบทางความถี่ของสัญญาณข้อมูลได้กล่าวถึงภาคผนวก ก



รูป ข.1 แสดงการลดทอนทางขนาดของสายส่งแบบ Twist pair

สำหรับค่าความลดทอนของขนาดกับความถี่ของสายส่งแบบ Twist pair แสดงดังในสมการ (ข.1)

$$\alpha^2 = \tfrac{1}{2}[\sqrt{(R^2 + \omega^2L^2)(G^2 + \omega^2C^2)} + (RG - \omega^2LC)] \qquad (ข.1)$$

α = Attenuation Constant per Unit length

R,L,G,C = Resistance, Inductance, Conductance, Capacitance per Unit Length

นอกจากความลดทอนทางขนาดแล้ว ยังมีความบั่นทอนทางเฟสของสายส่ง และความบั่นทอนทางเฟสของอุปกรณ์และฟิลเตอร์ของช่องสัญญาณ ค่าความบั่นทอนทางเฟสกำหนดเป็นค่า Group Delay หรือ Envelope Delay ซึ่งเป็นค่าที่กำหนดคุณภาพของการดีมอดูเลชั่น ตารางกำหนดความลดทอนทางขนาดและความบั่นทอนทางเฟสในช่องสัญญาณโทรศัพท์ แสดงดังตาราง ข.1

ค่าความบั่นทอนทางเฟสกับความถี่ของสายส่งแบบ twist pair แสดงดังในสมการ (ข.2)

$$\beta^2 = \tfrac{1}{2}[\sqrt{(R^2 + \omega^2L^2)(G^2 + \omega^2C^2)} - (RG - \omega^2LC)] \qquad (ข.2)$$

β = Phase Constant per Unit Length

ตาราง ข.1 การลดทอนทางขนาดและเฟสในช่องสัญญาณโทรศัพท์แบบต่าง ๆ

BELL SCHEDULE	3002	C1	C2	C4	DCS-S #
Attenuation Characteristic (referenced to 1000 Hz)	300 to 3000 Hz -3 to +12 dB	300 to 2700 Hz -2 to +6 dB	300 to 3000 Hz -2 to +6 dB	300 to 3200 Hz -2 to +6 dB	300 to 3000 Hz -1 to +3 dB
Envelope Delay Distortion ((max. μsec)	800 to 2600 Hz 1750 μsec	1000 to 2400 Hz 1000 μsec	1000 to 2600 Hz 500 μsec	1000 to 2600 Hz 300 μsec	1000 to 2600 Hz 100 μsec
		800 to 2600 Hz 1750 μsec	600 to 2600 Hz 1500 μsec	800 to 2800 Hz 500 μsec	600 to 2600 Hz 300 μsec
			500 to 2800 Hz 3000 μsec	600 to 3000 Hz 1500 μsec	500 to 2800 Hz 600 μsec
				500 to 3000 Hz 3000 μsec	

ภาคผนวก ค.

ข้อแนะนำ V.26 bis ของ CCITT

2400/1200 BITS PER SECOND MODEM STANDARDIZED FOR USE IN THE GENERAL SWITCHED TELEPHONE NETWORK

(Geneva, 1972; amended at Geneva, 1976 and 1980)

The CCITT,

considering

(a) that there is a demand for data transmission at 2400 bit/s over the general switched telephone network;

(b) that a majority of connections over the general switched telephone network within some countries are capable of carrying data at 2400 bit/s;

(c) that a much lower proportion of international connections in the general switched telephone service are capable of carrying data at 2400 bit/s;

unanimously declares the view

(1) that transmission at 2400 bit/s should be allowed on the general switched telephone network. Reliable transmission cannot be guaranteed on every connection or routing and tests should be made between the most probable terminal points before a service is provided.

The CCITT expects that developments during the next few years in modern technology will bring about modems of more advanced design enabling reliable transmission to be given on a much higher proportion of connections.

Note – The provisions of this Recommendation are to be regarded as provisional in order to provide service where it is urgently required and between locations where it is expected that a reasonably satisfactory service can be given. The study of improved methods of transmission at 2400 bits/s or above over the general switched telephone network will be urgently continued with the aim of recommending a method of transmission which will enable a more reliable service to be given over a high proportion of the connections encountered in normal service.

(2) that the characteristics of the modems for this service shall provisionally be the following:

1 Principal characteristics

a) Use of a data signalling rate of 2400 bit/s with carrier frequency, modulation and coding according to Recommendation V.26, Alternative B (see Note below) on the communication channel. Administrations and users should note that the performance of this modem on international connections may not always be suitable for this service without prior testing and conditioning if required.

b) Reduced rate capability at 1200 bit/s.

c) Inclusion of a backward channel at modulation rates up to 75 bauds, use of this channel being optional.

Note – Attention is drawn to the fact that there are some old-type modems currently in operation for which the coding method in accordance with Recommendation V.26, Alternative A, is used.

2 Line signals at 2400 and 1200 bit/s

2.1 The carrier frequency is to be 1800 ± 1 Hz. No separate pilot frequencies are provided. The power levels used will conform to Recommendation V.2.

2.2 *Phase distortion limits*

The transmitted line signal spectrum should have linear phase characteristics (to be obtained by means of filters or equalizers or digital means). The deviation of the phase distortion characteristic should not exceed the limits specified in Figure 1/V.26 *bis*.

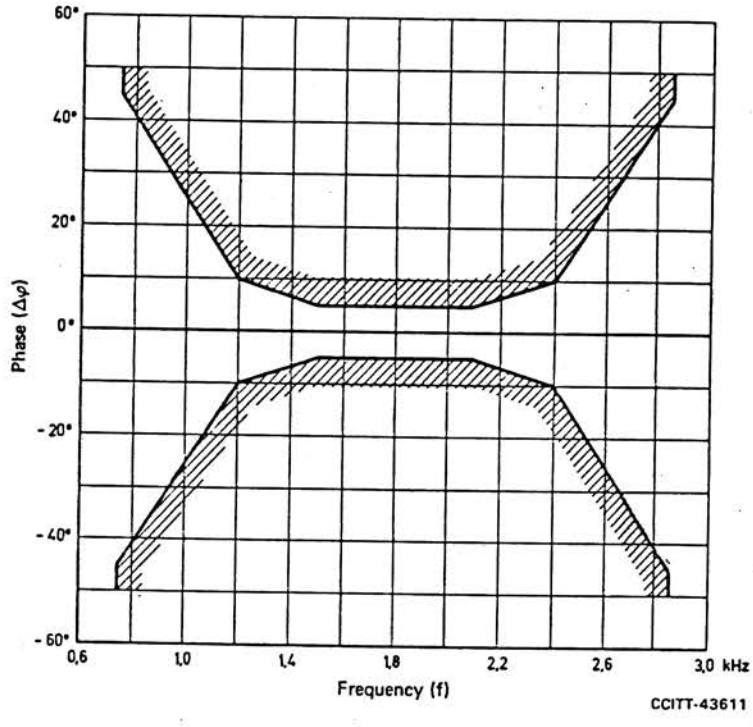


FIGURE 1/V.26 *bis*

Tolerance limit for phase distortion of the signal transmitted to the line

2.3 *Division of power between forward and backward channels*

Equal division of power between the forward and backward channels is recommended provisionally.

2.4 *Operation at 2400 bit/s*

2.4.1 The data stream to be transmitted is divided into pairs of consecutive bits (dibits). Each dibit is encoded as a phase change relative to the phase of the immediately preceding signal element (see Table 1/V.26 *bis*). At the receiver the dibits are decoded and the bits are reassembled in correct order. The left-hand digit of the dibit is the one occurring first in the data stream.

The meaning of phase change is illustrated by the line signal diagram given in Figure 2/V.26 *bis*.

2.4.2 *Synchronizing signal*

For the whole duration of the interval between the OFF to ON transitions of circuits 105 or 107 and 106, operator signal shall be that corresponding to the continuous transmission of dibit 11. This shall be known as the ...ing signal (see § 5.2.2 below).

TABLE 1/V.26 *bis*

Dibit	Phase change (see Note)
00	+45°
01	+135°
11	+225°
10	+315°

Note – The phase change is the actual on-line phase shift in the transition region from the centre of one signalling element to the centre of the following signalling element.

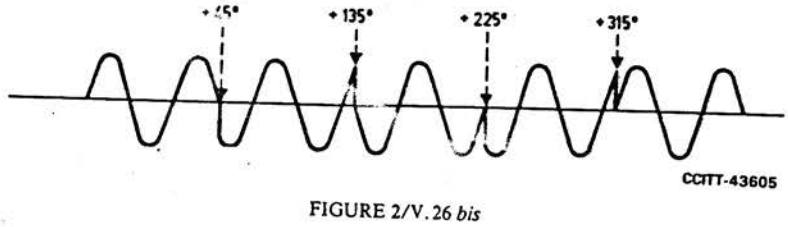


FIGURE 2/V.26 *bis*

Note – Owing to several causes the stability of timing recovery at the receiver is liable to be data-pattern sensitive. The presence of dibit 11 provides a stabilizing influence irrespective of the cause of lack of stability. Users are advised to include sufficient binary 1s in the data which will ensure that the dibit 11 will occur frequently.

2.4.3 *Data signalling and modulation rates*

The data signalling rate shall be 2400 bit/s ± 0.01%, i.e. the modulation rate is 1200 bauds ± 0.01%.

2.5 *Operation at 1200 bit/s*

2.5.1 Coding and modulation used are 2-phase differential modulation with binary 0 for +90° and binary 1 for +270°

2.5.2 The data signalling rate shall be 1200 bit/s ± 0.01%, the modulation rate remains at 1200 bauds ± 0.01%.

3 Received signal frequency tolerance

Noting that the carrier frequency tolerance allowance at the transmitter is ± 1 Hz and assuming a maximum frequency drift of ± 6 Hz in the connection between the modems, then the receiver must be able to accept errors of at least ± 7 Hz in the received frequencies.

4 Backward channel

4.1 *Modulation rate and characteristic frequencies for the backward channel*

The modulation rate and characteristic frequencies for the backward channel are as follows:

	F_Z (symbol 1, mark)	F_A (symbol 0, space)
Modulation rate up to 75 bauds	390 Hz	450 Hz

In the absence of any signal on the backward channel interface, the condition Z signal is to be transmitted.

4.2 *Tolerances on the characteristic frequencies of the backward channel*

As the backward channel is a VF telegraph-type channel, the frequency tolerances should be as recommended in Recommendation R.35 [1] for frequency-shift voice-frequency telegraphy.

The ± 6 Hz frequency drift in the connection between the modems postulated in § 3 above would produce additional distortion in the backward channel. This should be taken into account in the design.

5 Interchange circuits

5.1 *List of essential interchange circuits*

The list of interchange circuits essential for the modems when used on the general switched telephone network, including terminals equipped for manual calling or answering or automatic calling or answering is given in Table 2/V.26 *bis*.

5.2 *Response times of circuits 106, 109, 121 and 122* (see Table 3/V.26 *bis*).

5.2.1 Circuit 109 response times are the times that elapse between the connection or removal of the test synchronizing signal to or from the modem receive line terminals and the appearance of the corresponding ON and OFF condition on circuit 109.

The level of the test synchronizing signal should fall within the level range between 3 dB above the actual OFF to ON threshold of the received line signal detector and the maximum admissible level of the received signal. At all levels within this range, the measured response times shall be within the specified limits.

5.2.2 Circuit 106 response times are from the connection to an ON or OFF condition on:

- circuit 105 to the appearance of the corresponding ON or OFF condition on circuit 106; or
- circuit 107 (where circuit 105 is not required to initiate the synchronizing signal) to the appearance of the corresponding ON or OFF condition on circuit 106.

5.3 *Threshold of data channel and backward channel received line signal detectors*

Level of received line signal at receive line terminals of modem for all types of connections, i.e. general switched telephone network or non-switched leased telephone circuits:

- greater than −43 dBm: circuits 109/122 ON
- less than −48 dBm: circuits 109/122 OFF

The condition of circuits 109 and 122 for levels between −43 dBm and −48 dBm is not specified except that the signal detectors shall exhibit a hysteresis action such that the level at which the OFF to ON transition occurs is at least 2 dB greater than that for the ON to OFF transition.

Where transmission conditions are known and allowed, it may be desirable at the time of modem installation to change these response levels of the received line signal detector to less sensitive values (e.g. −33 dBm and −38 dBm respectively).

5.4 *Clamping in half-duplex mode*

The DCE, when operating in half-duplex mode on a 2-wire line, shall hold, where implemented:

a) circuit 104 in the binary 1 condition and circuit 109 in the OFF condition when circuit 105 is in the ON condition and, where required to protect circuit 104 from false signals, for a period of 150 ± 25 ms following the ON to OFF transition on circuit 105; the use of this additional delay is optional, based on system considerations;

TABLE 2/V.26 *bis*

Interchange circuit		Forward (data) channel one-way system (see Note 1)				Forward (data) channel either-way system (see Note 1)	
		Without backward channel		With backward channel			
No.	Designation	Transmit end	Receive end	Transmit end	Receive end	Without backward channel	With backward channel
102	Signal ground or common return	X	X	X	X	X	X
102a (see Note 2)	DTE common return	X	X	X	X	X	X
102b (see Note 2)	DCE common return	X	X	X	X	X	X
103	Transmitted data	X		X		X	X
104	Received data		X		X	X	X
105	Request to send	X		X		X	X
106	Ready for sending	X		X		X	X
107	Data set ready	X	X	X	X	X	X
108/1 or 108/2 (see Note 3)	Connect data set to line Data terminal ready	X	X	X	X	X	X
109	Data channel received line signal detector		X		X	X	X
111	Data signalling rate selector (DTE source)	X	X	X	X	X	X
113	Transmitter signal element timing (DTE source)	X		X		X	X
114	Transmitter signal element timing (DCE source)	X		X		X	X
115	Receiver signal element timing (DCE source)		X		X	X	X
118	Transmitted backward channel data				X		X
119	Received backward channel data			X			X
120	Transmit backward channel line signal						X
121	Backward channel ready				X		X
122	Backward channel received line signal detector			X			X
125	Calling indicator	X	X	X	X	X	X

Note 1 – All essential interchange circuits and any others which are provided shall comply with the functional and operational requirements of Recommendation V.24. All interchange circuits indicated by X shall be properly terminated in the data terminal equipment and in the data circuit-terminating equipment in accordance with the appropriate recommendation for electrical characteristics (see § 7).

Note 2 – Interchange circuits 102a and 102b are required where the electrical characteristics defined in Recommendation V.10 are used.

Note 3 – This circuit shall be capable of operation as circuit 108/1 or circuit 108/2 depending on its use. For automatic calling it shall be used as 108/2 only.

TABLE 3/V.26 *bis*

Response times

Circuit 106		
OFF to ON	750 ms to 1400 ms (see Note 1)	a) 65 ms to 100 ms (see Note 2) b) 200 ms to 275 ms (see Note 2)
ON to OFF	⩽ 2 ms	
Circuit 109		
OFF to ON	300 ms to 700 ms (see Note 1)	5 ms to 15 ms (see Note 1)
ON to OFF	5 ms to 15 ms	
Circuit 121		
OFF to ON	80 ms to 160 ms	
ON to OFF	⩽ 2 ms	
Circuit 122		
OFF to ON	< 80 ms	
ON to OFF	15 ms to 80 ms	

Note 1 – For automatic calling and answering, the longer response times of circuits 106 and 109 are to be used during call establishment only.

Note 2 – The choice of response times depends upon the system application: a) limited protection given against line echoes; b) protection given against line echoes.

Note 3 – The above parameters and procedures, particularly in the case of automatic calling and answering are provisional and are the subject of further study. Especially the shorter response times for circuit 109 may need revision to prevent remnants of the synchronizing signal from appearing on circuit 104.

b) circuit 119 in the binary 1 condition and circuit 122 in the OFF condition when circuit 120 is in the ON condition and, where required to protect circuit 119 from false signals, for a time interval following the ON to OFF transition on circuit 120. The specific duration of this time interval is left for further study. The additional delay is optional, based on system considerations.

5.5 *Fault condition of interchange circuits*

(See Recommendations V.10, § 11 and V.11, § 9 and V.28, § 7 for association of the receiver failure detection types.)

5.5.1 The DTE should interpret a fault condition on circuit 107 as an OFF condition using failure detection type 1.

5.5.2 The DCE should intepret a fault condition on circuits 105 and 108 as an OFF condition using failure detection type 1.

5.5.3 All other circuits not referred to above may use failure detection type 0 or 1.

6 Timing arrangements

Clocks should be included in the modem to provide the data terminal equipment with transmitter signal element timing, circuit 114 and receiver signal element timing, circuit 115. Alternatively, the transmitter signal

element timing may be originated in the data terminal equipment instead of in the data circuit-terminating equipment and be transferred to the modem via circuit 113.

7 Electrical characteristics of interchange circuits

7.1 Use of electrical characteristics conforming to Recommendations V.28 is recommended together with the connector pin assignment plan specified by ISO 2110 [2].

7.2 Application of electrical characteristics conforming to Recommendations V.10 and V.11 is recognized as an alternative together with the use of the connectors and pin assignment plan specified by ISO 4902 [3].

i) Concerning circuits 103, 104, 105, 106, 107, 108, 109, 113, 114 and 115, the receivers shall be in accordance with Recommendation V.11 or alternatively Recommendation V.10, category 1. Either V.10 or V.11 generators may be utilized.

ii) In the case of circuits 111, 118, 119, 120, 121, 122 and 125, Recommendation V.10 applies with receivers configured as specified by Recommendation V.10 for category 2.

iii) It is preferred that backward channel circuits appear on a separate connector and comprise circuits 118, 119, 120, 121, 122 (category 2) and 102, 102a and 102b.

iv) Interworking between equipment applying Recommendation V.10 and/or V.11 and equipment applying Recommendation V.28 is allowed on a non-interference basis. The onus for adaptation to V.28 equipment rests solely with the alternative V.10/V.11 equipment.

Note – Manufacturers may wish to note that the long-term objective is to replace electrical characteristics specified in Recommendation V.28, and Study Group XVII has agreed that the work shall proceed to develop a more efficient all balanced interface for the V-Series application which minimizes the number of interchange circuits. It is expected that this work would be based upon the alternative application given in § 7.2 above utilizing the V.11 electrical characteristics.

8 The following information is provided to assist equipment manufacturers:

The data modem should have no adjustment for send level or receive sensitivity under the control of the operator.

115

9 It will be for the user to decide whether, in view of the connections he makes with this system, he will have to request that the data circuit-terminating equipment be equipped with facilities for disabling echo suppressors. The international characteristics of the echo suppressor tone disabler have been standardized by the CCITT (Recommendation G.164 [4]) and the disabling tone should have the following characteristics:

- disabling tone transmitted: 2100 ± 15 Hz at a level of −12 ± 6 dBm0,
- the disabling tone to last at least 400 ms, the tone disabler should hold in the disabled mode for any single-frequency sinusoid in the band from 390-700 Hz having a level of −27 dBm0 or greater, and from 700-3000 Hz having a level of −31 dBm0 or greater. The tone disabler should release for any signal in the band from 200-3400 Hz having a level of −36 dBm0 or less,
- the tolerable interruptions by the data signal to last not more than 100 ms.

10 Fixed compromise equalizer

A fixed compromise equalizer shall be incorporated into the receiver. The characteristics of this equalizer may be selected by Administrations but this should be the matter for further study.

References

[1] CCITT Recommendation *Standardization of FMVFT systems for a modulation rate of 50 bauds*, Vol. VII, Fascicle VII.1, Rec. R.35.

[2] *Data communication – 25-pin DTE/DCE interface connector and pin assignments*, ISO Standard 2110-1980.

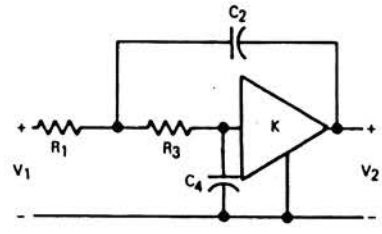
[3] *Data communication – 37-pin and 9-pin DTE/DCE interface connectors and pin assignments*, ISO Standard 4902-1980.

[4] CCITT Recommendation *Echo suppressors*, Vol. III, Fascicle III.1, Rec. G.164.

การออกแบบฟิวเตอร์อุปกรณ์มอดูเลชั่น-ดีมอดูเลชั่น

4th-Order Chebyshev Low-Pass Filter Design

This filter was realized by cascading two second-order Salen and Key filter sections. The design equations were taken from *INTRODUCTION TO THE THEORY AND DESIGN OF ACTIVE FILTERS* by Huelsman and Allen, pages 157-158. The general circuit realization for a Salen and Key is shown below.



Specifications:

Pass-band ripple = 0.01 dB f_c = 3600 Hz
Op-amp gain = K = 1

Pole Locations (normalized):

$\frac{a}{-0.6762} \pm \frac{b}{j.3828}$ first stage
$-0.2801 \pm j.9241$ second stage

Q = Quality Factor = $\frac{(a^2 + b^2)^{1/2}}{2a}$

where a = real pole coordinate
b = imaginary pole coordinate

Q_1 = first stage Q = 0.5746

Q_2 = second stage Q = 1.7237

Let $n = \frac{R3}{R1}$ and $m = \frac{C4}{C2}$

For this design procedure it is necessary that $m \leq \frac{1}{4Q^2}$

Therefore $m_1 \leq \frac{1}{4(0.5746)^2} = 0.7573$ Choose m = 0.1

$n = \frac{1}{2mQ^2} - 1 \pm \frac{(1-4mQ^2)^{1/2}}{2mQ^2}$

either value obtained is valid

n = 28.25675 for first stage

$R1C2 = \frac{1}{2(pi)(f_c)(mn)^{1/2}} = 2.63 \times 10^{-5}$

Let C2 = 0.01 μF

Then R1 = 2.63 kΩ----------> 2.61 kΩ
R3 = nR1 = 74.315 kΩ----> 75 kΩ
C4 = mC2 = 0.001 μF

The second stage was designed using the same equations and resulted in the following values for Q_2 = 1.7237 and f_c = 3600:

C2 = 0.01 μF
C4 = 100 pF

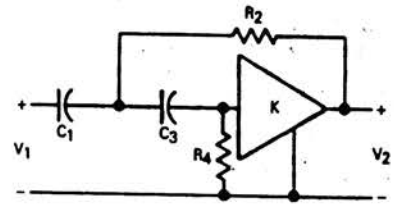
R1 = 7.87 kΩ
R3 = 249 kΩ

2nd-Order Chebyshev High-Pass Filter

Specifications:

Pass-band ripple = 0.01 dB f_c = 400 Hz
Op-amp gain = K = 1

The general Salen and Key high-pass realization is shown below:



Pole Locations (normalized low-pass values) =

$\frac{a}{-0.6743} \pm \frac{b}{j.7075}$

To transform the low-pass poles into high-pass poles,

$a_{HP} = \frac{a}{a^2 + b^2} = -0.7059$ $b_{HP} = \frac{b}{a^2 + b^2} = 0.740654$

$Q = 0.7247$ $\frac{1}{Q} = (mn)^{1/2}$ $\frac{m+1}{}$ from page 165.

For minimum Q, m should equal 1 which reduces the above equation to:

$Q = \frac{(n)^{1/2}}{2}$ or $n = 4Q^2 = 2.101$

$R2C1 = \frac{1}{2(pi)(f_c)(n)^{1/2}} = 2.745 \times 10^{-4}$

Letting C1 = C2 = 0.1 μF, we obtain

R2 = 2.745 kΩ-----> 2.74 kΩ
R4 = 5.767 kΩ-----> 5.76 kΩ

รายละเอียด IC MC 6172 และ 6173

MOTOROLA

MC6172
(Formerly MC6862)

2400 bps DIGITAL MODULATOR

The MC6172 is a MOS subsystem designed to be integrated into a wide range of equipment utilizing serial data communication.

The modulator provides the necessary modulation and control functions to implement a serial data communication link over a voice grade channel, utilizing differential phase shift keying (DKSP) at bit rates of 1200 or 2400 bps. Phase options are provided for both the U.S. and international markets. The MC6172 can be implemented into a wide range of data handling systems, including stand-alone modems, data storage devices, remote data commuication terminals, and I/O interfaces for counters.

N-channel silicon-gate technology permits the MC6172 to operate using a single voltage supply and be fully TTL compatible.

The modulator is compatible with the MC6173 demodulator to provide medium-speed data communications capability.

- Clear-to-Send Delay Options
- 511-Bit CCITT Test Pattern
- Terminal Interfaces are TTL Compatible
- Compatible Functions for 201B/C Data Sets
- CCITT and U.S. Phase Options
- 1200/2400 bps Operation
- Answer-Back Tone
- The MC6173 Is the Companion Demodulator
- Application Note Available – AN-870

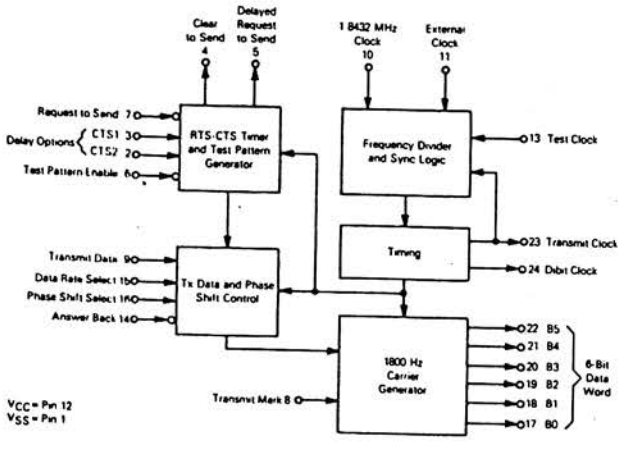
MOS
(N-CHANNEL, SILICON-GATE)

2400 bps MODULATOR

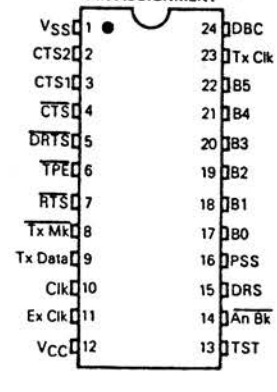
L SUFFIX
CERDIP PACKAGE
CASE 623

P SUFFIX
PLASTIC PACKAGE
CASE 709

BLOCK DIAGRAM



PIN ASSIGNMENT



MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	T_L to T_H 0 to 70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C

THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance Plastic Package Cerdip Package	θ_{JA}	 120 65	°C/W

This device contains circuitry to protect the inputs 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. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{CC}).

POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \bullet \theta_{JA}) \quad (1)$$

Where:

T_A = Ambient Temperature, °C

θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W

$P_D = P_{INT} + P_{PORT}$

$P_{INT} = I_{CC} \times V_{CC}$, Watts — Chip Internal Power

P_{PORT} = Port Power Dissipation, Watts — User Determined

For most applications $P_{PORT} < P_{INT}$ and can be neglected. P_{PORT} may become significant if the device is configured to drive Darlington bases or sink LED loads.

An approximate relationship between P_D and T_J (if P_{PORT} is neglected) is:

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations 1 and 2 for K gives:

$$K = P_D \bullet (T_A + 273°C) + \theta_{JA} \bullet P_D^2 \quad (3)$$

Where K is a constant pertaining to the particular part. K can be determined from equation 3 by measuring P_D (at equilibrium) for a known T_A. Using this value of K the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

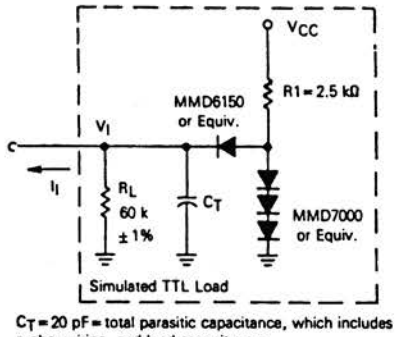
DC ELECTRICAL CHARACTERISTICS

(V_{CC} = 5.0 ±0.25 Vdc, V_{SS} = 0, $T_A = T_L$ to T_H, all outputs loaded as shown in Figure 1 unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}	V_{SS} + 2.0	–	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	–	V_{SS} + 0.8	V
Input Current ($V_{in} = V_{SS}$) CTS1, CTS2, PSS, DRS, $\overline{An\ Bk}$, and $\overline{Tx\ MK}$ $\overline{RTS}$ and $\overline{TPE}$	I_{in}	– –	– –	−0.2 −1.6	mA
Input Leakage Current (V_{in} = 5.25 V, $V_{CC} = V_{SS}$)	I_{IL}	–	–	2.5	µA
Output High Voltage (I_{OH} = −0.04 mA, Load A) (I_{OH} = 0.0 mA, Load B)	 V_{OH1} V_{OH2}	 V_{SS} + 2.4 V_{CC} − 0.5 V	 – –	 V_{CC} V_{CC}	V
Output Low Voltage (I_{OL} = 1.6 mA, Load A)	V_{OL}	V_{SS}	–	V_{SS} + 0.4	V
Input Capacitance (f = 0.1 MHz, T_A = 25°C)	C_{in}	–	5.0	–	pF
Internal Power Dissipation (Measured at $T_A = T_L$) (All inputs at V_{SS} except Pin 13 = 57.6 kHz and ALL outputs open)	P_{int}	–	210	315	mW
Input Transition Times, All Inputs Except 1.8432 MHz Input (From 10% to 90% points)	t_r, t_f	–	–	1.0*	µs
Input Transition Times, 1.8432 MHz Input (From 0.8 V to 2.0 V)	t_r, t_f	–	–	40	ns
Input Clock Duty Cycle, 1.8432 MHz Input (Measured at 1.5 V level)	D.C.	30	–	70	%
Tx Data Setup Time (Figure 2)	t_{su}	35	–	–	µs
Tx Data Hold Time (Figure 2)	t_h	35	–	–	µs
Output Transition Times	t_r, t_f	–	–	5.0	µs

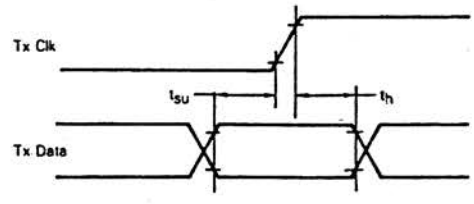
*Maximum Input Transition Times are ≤0.1 x Pulse Width or the specified maximum of 1.0 µs, whichever is smaller.

FIGURE 1 — OUTPUT TEST LOAD



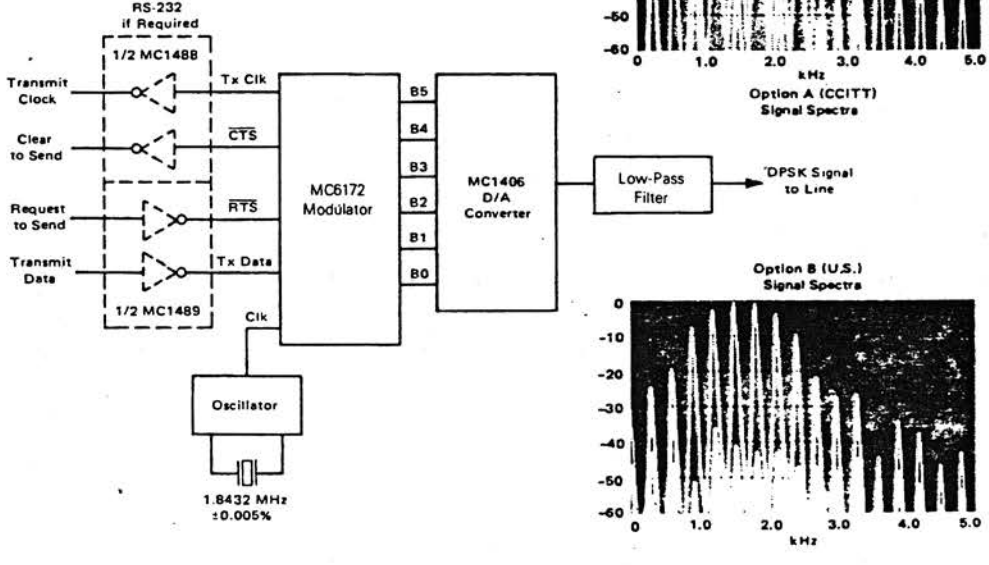
C_T = 20 pF = total parasitic capacitance, which includes probe, wiring, and load capacitances.

FIGURE 2 — TRANSMIT DATA SETUP AND HOLD TIME



Note: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

FIGURE 3 — 2400 bps MODULATOR INTERFACE

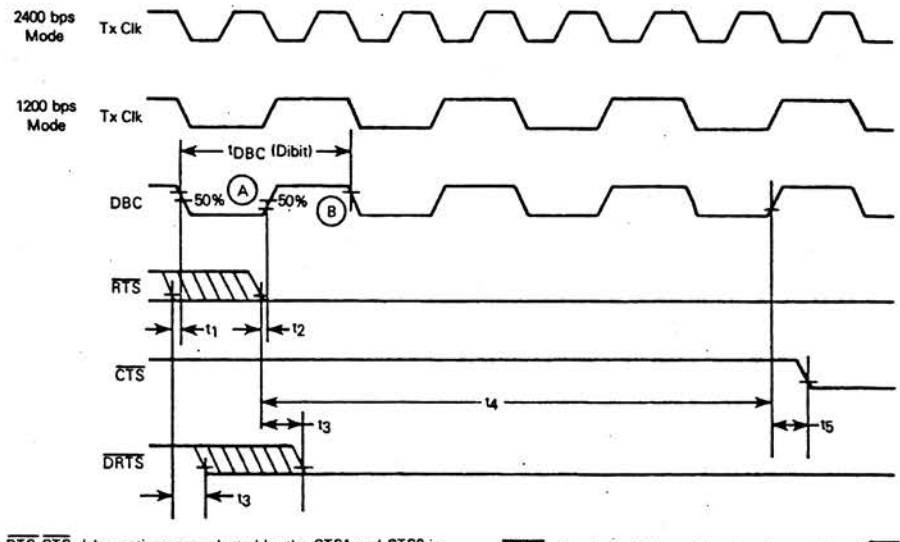


DELAY TIMINGS (See Figures 4 and 5)

Characteristic	Symbol	Min	Typ	Max	Unit
$\overline{RTS}$ to DBC Delay	t_1	–	–	8	µs
DBC to $\overline{RTS}$ Delay	t_2	45	–	–	µs
$\overline{RTS}$-$\overline{DRTS}$ Delay	t_3	–	–	35	µs
$\overline{RTS}$-$\overline{CTS}$ Delay					
CTS1 = 0, CTS2 = 1	t_4*	0	–	35	µs
CTS1 = 1, CTS2 = 0		8.55	–	9.35	ms
CTS1 = 1, CTS2 = 1		24.9	–	26.4	ms
CTS1 = 0, CTS2 = 0		147.0	–	154.0	ms
$\overline{CTS}$-DBC Delay					
CTS1 = 1, CTS2 = 0	t_5	–	–	35	µs
CTS1 = 1, CTS2 = 1		–	–	35	
CTS1 = 0, CTS2 = 0		–	–	35	
$\overline{RTS}$ to $\overline{CTS}$ Low	t_6	–	–	1.60	ms
$\overline{RTS}$ Min Delay	t_7	–	–	1.67	ms
DBC to $\overline{DRTS}$ Delay	t_8	–	–	35	µs
DBC Cycle Time	t_{DBC}	833.28	833.33	833.37	µs

*The reference frequency tolerance is not included.

FIGURE 4 — $\overline{RTS}$-$\overline{CTS}$ AND $\overline{RTS}$-$\overline{DRTS}$ DELAYS



$\overline{RTS}$-$\overline{CTS}$ delay options are selected by the CTS1 and CTS2 inputs, and are stated as time delay interval t_4. An $\overline{RTS}$ input signal synchronized about point A will synchronize $\overline{CTS}$ with the positive transition of DBC (Dibit Clock). Delay t_4 is measured with respect to the negative transition of $\overline{RTS}$.

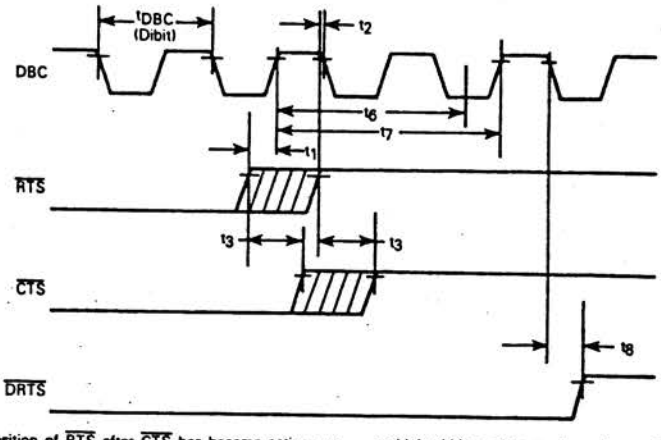
$\overline{RTS}$ signals synchronized with the positive transition of DBC (point B), will result in the same $\overline{CTS}$ delay (t_4). For this case the negative transition of $\overline{CTS}$ is synchronized with the negative transition of DBC with delay t_4 measured with respect to the negative transition of $\overline{RTS}$.

$\overline{DRTS}$ will go low within t_3 of the negative transition of $\overline{RTS}$. With the exception of the no-delay option, $\overline{CTS}$ will go low within t_5 of the positive transition of DBC, following the t_4 delay selected. This applies when $\overline{RTS}$ is synchronized to Point A as shown.

If $\overline{RTS}$ goes high and remains high ≥ 20 µs within time interval t_4, a reset of the internal $\overline{RTS}$-$\overline{CTS}$ timer function will occur. If $\overline{RTS}$ goes high for less than 20 µs, the circuit may or may not respond to this momentary loss of the $\overline{RTS}$ signal.

Note: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

FIGURE 5 — LOSS OF $\overline{RTS}$ TO $\overline{DRTS}$ DELAY



A positive transition of $\overline{RTS}$ after $\overline{CTS}$ has become active can result in different functional characteristics of the $\overline{CTS}$ and $\overline{DRTS}$ output signals, depending on the time duration that $\overline{RTS}$ remains inactive.

Under all conditions, $\overline{CTS}$ will go high within t_3 following a positive transition of $\overline{RTS}$. If $\overline{RTS}$ goes high in the shaded region shown (i.e., synchronized to the positive transition of DBC) and remains high beyond the time interval defined as t_7, then $\overline{DRTS}$ will go high within t_8 of the next negative transition of DBC. If $\overline{RTS}$ were to go low after t_7, the $\overline{RTS}$-$\overline{CTS}$ delay times given in Figure 4 will result.

If $\overline{RTS}$ goes high in the shaded region shown, and then returns low within time interval t_6, the negative transition of $\overline{CTS}$ will follow within 35 μs, and $\overline{DRTS}$ will remain in the active or low state. Under these conditions, the normal $\overline{RTS}$-$\overline{CTS}$ delay times are not encountered when $\overline{RTS}$ is reactivated. If $\overline{RTS}$ goes low for less than 20 μs, the circuit may or may not respond

NOTE: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

DEVICE OPERATION

GENERAL

Figure 3 shows the modulator and its intra-connections. The data to be transmitted is presented in synchronous serial format to the modulator for conversion to DPSK signals used in transmission. The modulator output is digital; therefore, a D/A converter and a filter transform the signal to an analog form.

The control functions provide four different Clear-to-Send delay options. An Answer-Back tone is available for automatic answering applications. The modulator has a built-in 511-bit pseudorandom pattern generator for use in system diagnostic tests.

INPUT/OUTPUT FUNCTIONS

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

The $\overline{RTS}$ signal from the data terminal controls transmission from the modulator. A low level on $\overline{RTS}$ activates the modulator data output. A constant mark, for synchronization, is sent during the $\overline{RTS}$ to $\overline{CTS}$ delay interval. Termination of the transmission is accomplished by taking $\overline{RTS}$ high (see Figures 4 and 5).

Delayed Request to Send ($\overline{DRTS}$)

This output can be used to control transmission as specified by the Transmit Mark control input. $\overline{DRTS}$ follows the negative transition of $\overline{RTS}$, and goes negative within t_3 of the negative transition of $\overline{RTS}$ (Figure 4). The delay from a positive transition of $\overline{RTS}$ to a positive transition of $\overline{DRTS}$ is shown in Figure 5. The $\overline{DRTS}$ delay allows data within the modulator to be transmitted before transmission is inhibited.

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

$\overline{CTS}$ follows $\overline{RTS}$ to both the logic 0 and logic 1 levels. The delay from a negative transition of $\overline{RTS}$ to a negative $\overline{CTS}$ transition is selectable by external strapping of CTS1 and CTS2. The delay from a positive transition of $\overline{RTS}$ to a positive $\overline{CTS}$ transition is less than t_4.

$\overline{CTS}$ will go low within t_5 after the positive transition of the Dibit Clock (see Figure 4) except when the non-delay option is selected. For the no-delay option, $\overline{CTS}$ follows $\overline{RTS}$ within t_5.

$\overline{RTS}$-$\overline{CTS}$ Delay Options (CTS1, CTS2)

The $\overline{RTS}$-$\overline{CTS}$ delays are selectable according to the following strapping options

$\overline{RTS}$-$\overline{CTS}$ Delay	CTS1	CTS2
0.0 + 0.035 ms, −0.0 ms	0	1
8.55 to 9.35 ms	1	0
24.90 to 26.4 ms	1	1
147.0 to 154.0 ms	0	0

Transmit Mark (Tx Mk)

The Transmit Mark control allows the system designer to select whether the Delayed Request to Send activitates and deactivates the transmission on the modulator chip or off the chip in the output amplifier.

When Tx Mk is high, transmission is controlled on the modulator chip, and occurs from the chip only when $\overline{DRTS}$ or Answer Back is in the logic 0 state (see Figure 6).

When Tx Mk is low, transmission is controlled off the modulator chip. In this mode, the modulator chip transmits marks at all times except when data or an Answer-Back tone is being transmitted (see Figure 6).

Test Pattern Enable ($\overline{TPE}$)

A 511-bit test pattern generator is contained on the modulator chip. This pattern is in accord with CCITT specification V52.

The 511-bit test pattern is activated by applying a logic 0 to $\overline{TPE}$. A mark (logic 1) condition on the Transmit Data input with $\overline{TPE}$ activated (logic 0) causes the test pattern to appear at the data output. A space (logic 0) condition on Tx Data with $\overline{TPE}$ activated causes the test pattern data to appear inverted at the data output.

Although the Motorola 2400 bps modulator contains a CCITT 511 test pattern generator it does not incorporate the 511 data randomizer or scrambler.

Random data applied to Tx Data with TPE activated causes the test pattern data to be scrambled (exclusive NORed) with the data, and the result appears at the data output.

The MC6173 demodulator does contain a built-in data descrambler, which is enabled by $\overline{TPE}$ input going active. To scramble data using the modulator, the circuit in Figure 7 must precede the Tx Data input of the modulator. Tx Data is added to the scrambler output pattern. Then the data is delayed by a full data bit before being transmitted by the modem. This assures a proper Transmit Data/Transmit Clock phase relationship.

If the data scrambler is to be an optional feature, then the transmit data multiplexer would also have to be built. This is selected by the Test Pattern Enable signal or any other signal that is found suitable.

The scrambling of data in the data comm environment is not done in an attempt to encrypt information in the normal sense of the word. Rather, the purpose of the scrambling of data is to guarantee that with respect to the modem carrier, there is always random data on the line with little chance for a long string of ones or zeros to exist. This is particularly important if an adaptive equalizer is being incorporated at the demodulator. The adaptive equalizer will require reasonably evenly distributed data to optimize its statistical response to the incoming signal. The normally used code is the CCITT 511 sequence which is exclusive ORed with data.

The test pattern generator can be enabled only when $\overline{CTS}$ and $\overline{RTS}$ are logic 0. If $\overline{TPE}$ is activated outside this time interval, the previously stated $\overline{RTS}$-$\overline{CTS}$ and $\overline{RTS}$-$\overline{DRTS}$ delays, shown in Figures 4 and 5, are not valid.

Data-Rate Select (DRS)

The modulator can transmit at either 2400 bps or 1200 bps. Both data rates utilize an 1800 Hz carrier signal and employ phase shifting at 1200 Hz. The 2400 bps rate is obtained by encoding two bits of data into each phase shift. The 2400 Hz rate is selected by applying a logic 1 to the Data-Rate Select lead. The 1200 Hz rate is selected by applying a logic 0 to DRS.

Phase-Shift Select (PSS)

Option A (CCITT) or Option B (U.S.) phase shift can be selected for 2400 bps operation. The input data format and phase shift relationship for these two options are as follows:

Data	PSS = 0 Option A*	PSS = 1 Option B
00	0°	+45°
01	+90°	+135°
11	+180°	+225°
10	+270°	+315°

*See example Figure 8.

FIGURE 6 — TRANSMIT MARK CONTROL

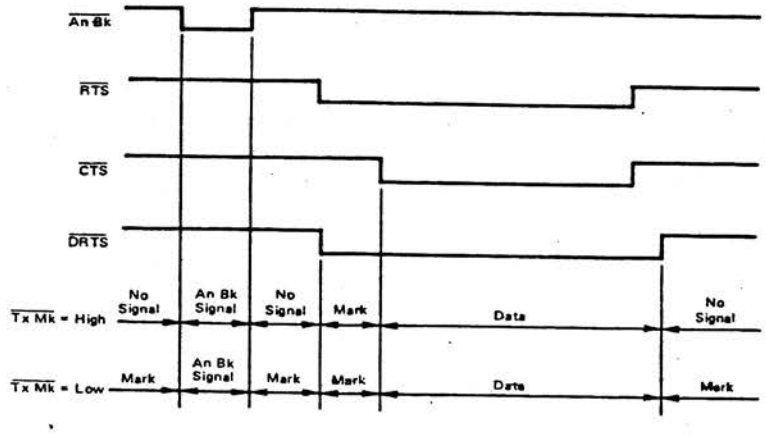


FIGURE 7 — MODULATOR CCITT 511 DATA SCRAMBLER

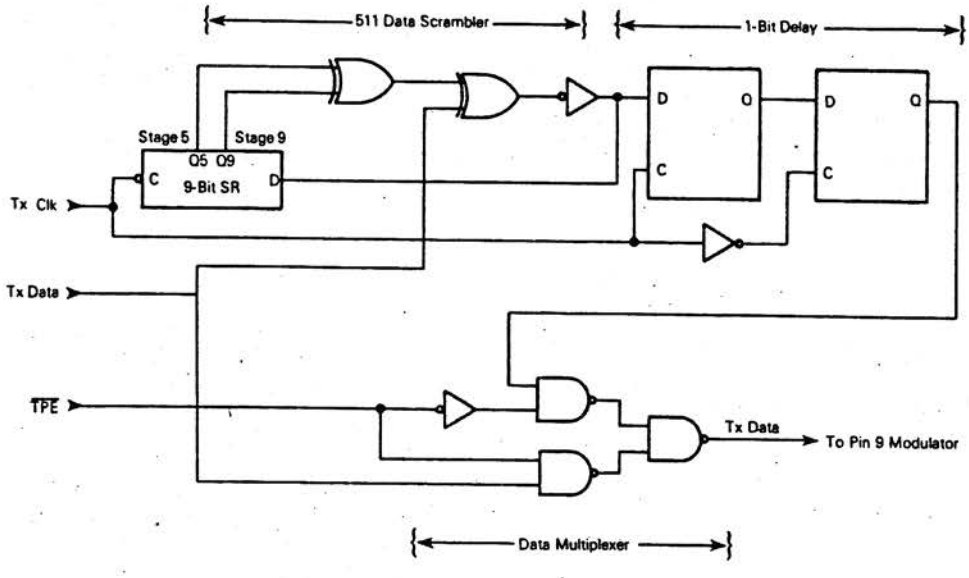
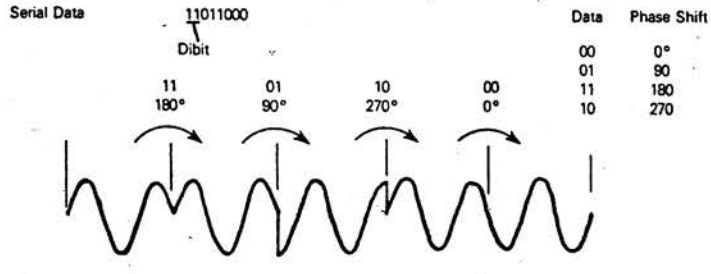


FIGURE 8 — EXAMPLE-CARRIER PHASE SHIFTS FOR OPTION A



For 1200 bps operation, Option C (CCITT) or Option D (U.S.) phase shift can be selected:

Data	PSS = 0 Option C	PSS = 1 Option D
0	+90°	+45°
1	+270°	+225°

Option C is selected by applying a logic 0 to the Phase Shift Select lead when the Data Rate Select lead is strapped for 1200 bps operation (logic 0). Option D is selected by applying a logic 1 to PSS with DRS at logic 0. The phase shifts shown are the difference in phase between the signal at the end of one dibit period and the new signal at the beginning of the next dibit.

Transmit Data (Tx Data)

Transmit Data is the serial binary information presented for DPSK modulation. A high level represents a mark. For timing, see Transmit Clock (Figure 4).

Transmit Clock (Tx Clk)

A 2400/1200 Hz Transmit Clock output is provided for the communication terminal. The Transmit Data signal is sampled on the positive transition of Transmit Clock. The Transmit Data to Transmit Clock setup and hold time requirements are shown in the Electrical Characteristics Table and in Figure 2.

Dibit Clock (DBC)

A 1200 Hz Dibit Clock identifies the modulation timing. This signal goes negative less than 100 μs prior to the start of dibit modulation.

External Clock (Ex Clk)

A 2400/1200 Hz clock signal applied to the External Clock input causes Transmit Clock to be synchronized with Ex Clk. This input must have an accuracy within ±0.005%.

When no transitions occur on this input, the internal clock provides the 2400/1200 Hz transmit timing signal. Fast synchronization of Tx Clk to Ex Clk is not provided on the chip. *When Ex Clk is not used, it should be tied to either the logic 0 or logic 1 state.*

9.8432 MHz (Clk)

This input must be a square wave with rise and fall times of less than 40 ns and a 50 ±20% duty cycle. The clock accuracy must be written ±0.005%.

Answer Back ($\overline{\text{An Bk}}$)

A logic 0 level applied to Answer Back causes a 2025 Hz carrier to be generated on the modulator chip instead of a phase shifted 1800 Hz carrier. A logic 1 level applied to $\overline{\text{An Bk}}$ enables the modulator to generate the normal phase shifted 1800 Hz carrier signal, as shown in Figure 6. The time delay from a transition on $\overline{\text{An Bk}}$ to the appropriate signal at the modulator chip output is less than 2 ms.

Activation of $\overline{\text{An Bk}}$ (a logic 0) will disable all other operation modes including the $\overline{\text{Tx Mk}}$ function, and will reset $\overline{\text{CTS}}$ to an inactive state along with the $\overline{\text{RTX-CTS}}$ internal timer. $\overline{\text{An Bk}}$ should therefore be activated only before initiating $\overline{\text{RTS}}$ or after loss of the $\overline{\text{DRTS}}$ output signal. The combination of a logic 0 on $\overline{\text{An Bk}}$ with a logic 0 on $\overline{\text{TPE}}$ is not used in normal system operation, and hence is used as a reset input during device test.

Digital Output (B0-B5)

These outputs are designed to interface with a 6-bit digital-to-analog converter. The resultant signal out of the D/A is the differential phase shift keyed signal quantized at a 14.4 kHz rate. A low-pass filter can then be used to smooth the data transitions. B0 is the least-significant bit, and the positive level the active state.

Test Clock (TST)

A test signal input is provided to decrease test time of the chip. *In normal operation this input must be strapped low.*

MOTOROLA

MC6173

2400 bps DIGITAL DEMODULATOR

The MC6173 is a MOS subsystem designed to be integrated into a wide range of equipment utilizing serial data communication.

The demodulator provides the necessary demodulation and control functions to implement a serial data communication link over a voice grade channel, utilizing differential phase shift keying (DPSK) at bit rates of 1200 or 2400 bps. Phase options are provided for both the U.S. and international markets. The MC6173 can be implemented into a wide range of data handling systems, including stand-alone modems, data storage devices, remote data communication terminals, and I/O interfaces for counters.

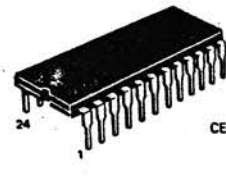
N-channel silicon gate technology permits the MC6173 to operate using a single voltage supply and be fully TTL compatible.

The demodulator is compatible with the M6800 microcomputer family, and provides medium-speed data communications capability.

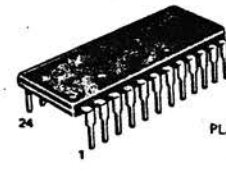
- Compatible with MC6172 Modulator
- 511-Bit CCITT V.52 Test Pattern
- Terminal Interfaces Are TTL Compatible
- Compatible Functions for 201B/C and V.26 Data Sets
- CCITT and U.S. Phase Options
- 1200/2400 bps Operation

MOS
(N-CHANNEL, SILICON-GATE)

2400 bps DEMODULATOR

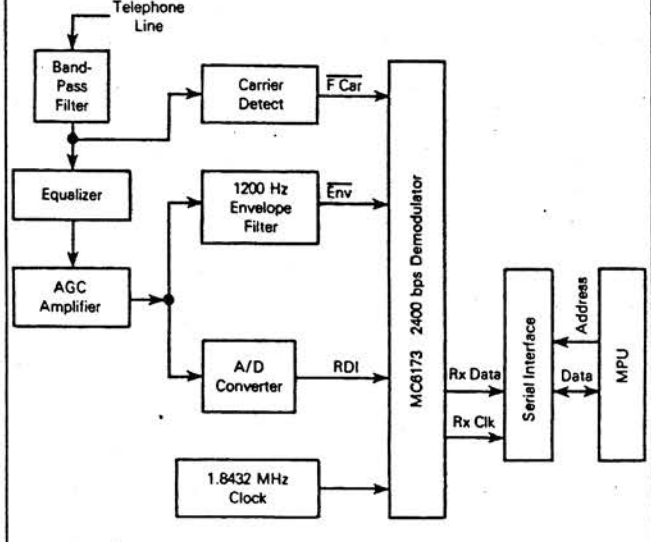


L SUFFIX
CERDIP PACKAGE
CASE 623

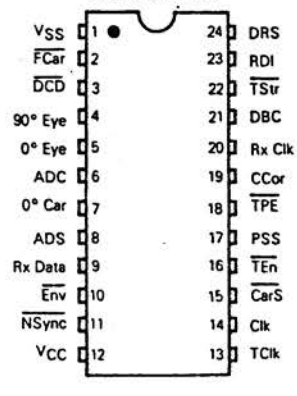


P SUFFIX
PLASTIC PACKAGE
CASE 709

FIGURE 1 — TYPICAL APPLICATIONS



PIN ASSIGNMENT



MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature Range	T_A	0 to 70	°C
Storage Temperature Range	T_{stg}	−55 to +150	°C
Thermal Resistance	θ_{JA}	82.5	°C/W

This device contains circuitry to protect the inputs 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. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{SS}).

DC ELECTRICAL CHARACTERISTICS (V_{CC} = 5.0 ±0.25 Vdc, V_{SS} = 0, $T_A = T_L$ to T_H all outputs loaded as shown in Figure 3 unless otherwise noted.)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage	V_{IH}	V_{SS}+2.0	–	V_{CC}	V
Input Low Voltage	V_{IL}	V_{SS}	–	V_{SS}+0.8	V
Input Current ($V_{in} = V_{IL}$) Pins 3, 11, 13, 15, 16, 17, 18, 22, 24	I_{IL}	–	–	−0.2	mA
Input Leakage Current (V_{in} = 5.25 Vdc, $V_{CC} = V_{SS}$) Pins 2, 10, 14, 23	I_{in}	–	–	2.5	µA
Output High Voltage (I_{OH} = −0.04 mA, Load A) (I_{OH} = 0.0 mA, Load B)	V_{OH1} V_{OH2}	V_{SS}+2.4 V_{CC}−0.5 V	– –	V_{CC} V_{CC}	V
Output Low Voltage (I_{OL} = 1.6 mA, Load A)	V_{OL}	V_{SS}	–	V_{SS}+0.4	V
Input Capacitance (f = 0.1 MHz, T_A = 25°C)	C_{in}	–	5.0	–	pF
Internal Power Dissipation (measured at $T_A = T_L$) (All Inputs at V_{SS} except Pin 13 = 57.6 kHz and ALL Outputs Open)	PInt	–		630	mW
Input Transition Times, 1.8432 MHz Input (From 0.8 V to 2.0 V)	t_r t_f	– –	– –	40 40	ns
Input Transition Times, All Inputs Except 1.8432 MHz Input (From 10% to 90% Points)	t_r, t_f	–	–	1.0*	µs
Output Transition Times (From 10% to 90% Points)	t_r, t_f	–	–	5.0	µs
Input Clock Duty Cycle, 1.8432 MHz Input (Measured at 1.5 V level)	D.C.	30	–	70	%
Data Setup Time	t_{DS}	770	–	–	ns
Rx Data Setup Time	t_{su}	35	–	–	µs
Data Hold Time	$t_{h(D)}$	0	–	–	ns
Rx Data Hold Time	t_h	35	–	–	µs
Data-Clamp Delay Time Option 1 Option 2 Option 3 Option 4	 t_{DCD1} t_{DCD2} t_{DCD3} t_{DCD4}	 5.7 4.135 20.795 104.135	 6 4.17 20.83 104.17	 6.3 4.205 20.865 104.205	 ns ms ms ms
A/D Clock to A/D Strobe Delay Time	t_{ADCD}	1.06	–	1.11	µs
Envelope-to-Dibit Clock Delay Time	t_{ED}	140	–	220	µs
Clock Frequency, ±0.005%	f_{Clk}	–	1.8432	–	MHz
A/D Clock Cycle Time (f_{Clk}/4)	t_{cyc}	–	2.17	–	µs
A/D Clock Pulse Width	$t_{w(ADC)}$	940	1000	1040	ns
A/D Strobe Pulse Width	$t_{w(ADS)}$	–	10.85	–	ns
New Sync Input Pulse Width	$t_{w(NSync)}$	0.84	–	–	ms

*Maximum input transition times are 2 µs pulse width or the specified maximum of 1.0 µs, whichever is smaller.

FIGURE 2 — DEMODULATOR BLOCK DIAGRAM

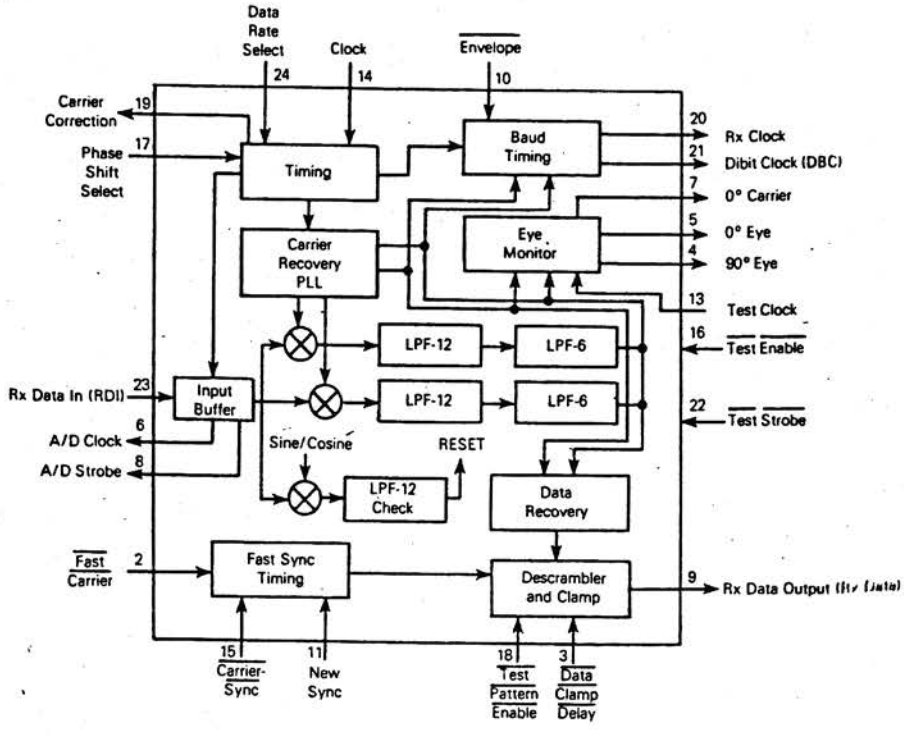
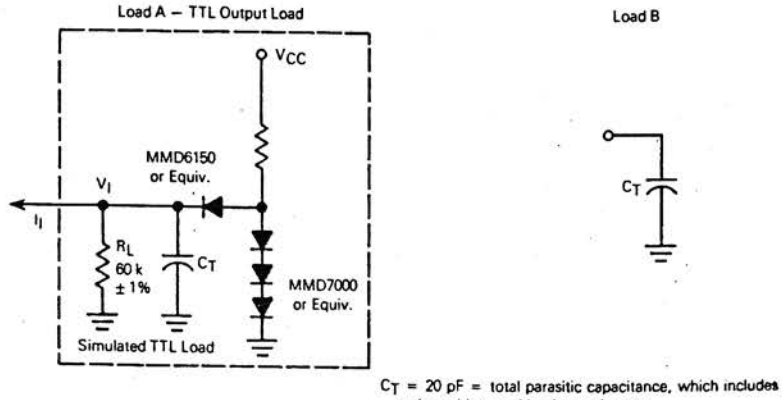


FIGURE 3 — OUTPUT TEST LOADS



C_T = 20 pF = total parasitic capacitance, which includes probe, wiring, and load capacitances

GENERAL DESCRIPTION

The MC6173 Phase-Shift Key (PSK) Demodulator serves as an integral part of a system to recover synchronous data from an 1800 Hz PSK modulated carrier. Data rates of 1200 and 2400 bits-per-second are available. In the case of 1200 bps operation, the MC6173 detects phase shifts of 0 to 180 degrees to represent digital "0s" and "1s". When 2400 bps operations is desired, the MC6173 detects phase shifts of 0, 90, 180, and 270 (option A) or 45, 135, 225, and 315 (option B) degrees to represent two bits of data called dibits. These phase shifts decode to 00, 01, 10, and 11, respectively. In either data rate, the 1800 Hz carrier is modulated at a 1200 rate.

Figure 1 shows the MC6173 demodulator in a typical application. The band-pass filter, equalizer, analog-to-digital (A/D) converter, 1200 Hz envelope filter, AGC amplifier, and 1800 Hz carrier detector are external to the MC6173. The band-pass filter passes roughly 300 Hz to 3000 Hz eliminating noise, 60 Hz and 120 Hz pickup, and harmonics of 1800 Hz. The output of this filter is fed to the equalizer which adjusts phase versus amplitude such that a constant amplitude is maintained regardless of phase and is fed into the carrier detect circuit. The AGC amplifier provides a constant level signal regardless of the input level from the equalizer. The output of the AGC amplifier drives two basic sections of external circuitry, i.e., the A/D converter, and 1200 Hz envelope filter.

The A/D converter samples each 1200 Hz cycle or dibit 12 times. After each sample, digital data is clocked serially to the MC6173 receiver data input (RDI). The MC6173 generates the sampling clock for AD Strobe (ADS) and the serial clock (ADC) from the 1.8432 MHz internal oscillator.

The 1200 Hz envelope filter recovers the 1200 Hz component of the equalizer output during fast training and generates a 1200 Hz square wave. This square wave is connected to the envelope ($\overline{\text{Env}}$) input and is used for internal timing.

The carrier detect circuit is used to signal the fast carrier ($\overline{\text{FCar}}$) input that a carrier is present. Immediately after $\overline{\text{FCar}}$ has received a negative transition, the internal phase-lock loop temporarily widens its band width so that it can quickly adjust the internal timing of the MC6173 with respect to the 1200 Hz $\overline{\text{Env}}$ input (this is called fast Sync or fast training). The timing adjustments are made so that each dibit can be sampled at the most advantageous places.

The internal circuitry digests the dibit samples and produces the digital data (Rx Data) along with the receive data clock (Rx Clk). These two signals are used to drive a serial-to-parallel interface such as an MC6852 Synchronous Serial Interface Adapter.

PIN DESCRIPTION

$\overline{\text{FAST}}$ $\overline{\text{CARRIER}}$ ($\overline{\text{FCar}}$), Pin 2 — A negative transition on this input will force a period of approximately 8.3 ms of fast training for both baud and carrier timing.* Fast Sync or fast training allows for large corrections to be made in the internal timing of the demodulator. After the fast training period, the timing should be reasonably well adjusted. Small adjustments are made automatically to maintain proper phase relationships internally after the fast-train period.

The $\overline{\text{FCar}}$ input, which normally comes from the carrier threshold detect circuits, must remain at a low level during the entire period of baud and carrier synchronization.

A positive level on the $\overline{\text{FCar}}$ input will disable the baud and carrier correction circuitry. Baud and carrier timing are then direct derivatives of the 1.8432 MHz clock as illustrated in Figure 4.

The first positive edge of the envelope ($\overline{\text{Env}}$) input will be totally asynchronous to the demodulator. This will be ± ½ cycle of the 2400 clock (± 208 μs). The nine following positive edges will introduce added tolerance equal to nine times the offset of $\overline{\text{Env}}$ from the absolute 1200 Hz (as defined by the 1.8432 MHz ± 0.005% clock). Thus . . .

$$\text{Max Fast Train Time} = 4.17\ \text{ms} + 9\ f_{\overline{\text{Env}}} + 0.21\ \text{ms}$$
$$= 4.38\ \text{ms} + 9/f_{\overline{\text{Env}}}$$
$$\text{Min Fast Train Time} = 4.17\ \text{ms} - 0.21\ \text{ms} + 9/f_{\overline{\text{Env}}}$$
$$= 3.96\ \text{ms} + 9/f_{\overline{\text{Env}}}$$

$\overline{\text{DATA-CLAMP}}$ $\overline{\text{DELAY}}$ ($\overline{\text{DCD}}$), Pin 3 — Data-clamp delay enables the selection of one of four delays during which Rx Data is held to a logic-high condition. This delay is measured from the negative edge of $\overline{\text{FCar}}$. The four options are available at one pin through the use of the internal multiplexing in the demodulator. Options 3 and 4 are available by demultiplexing the dibit clock as demonstrated in Figure 5. The available delay options are listed in Table 1, these times will be approximate due to their direct relationship to the $\overline{\text{Env}}$ input during the first 8.3 ms. Also, these times are further dependent upon carrier offset. The delays given in Table 1 assume no carrier offset and that $\overline{\text{Env}}$ is synchronous with the Tx Clk. Figure 4 is illustrative of the timing and sequencing of this circuit.

A scheme for programming the data-clamp delay is illustrated in Figure 5. The $\overline{\text{DCD}}$ input may either be a constant high or low level which will produce options 1 and 2. If the input "A" is exclusive ORed with the dibit clock options 3 and 4 are produced at the same input pin.

$\overline{\text{ENVELOPE}}$ ($\overline{\text{Env}}$), Pin 10 — The envelope input comes from the 1200 Hz envelope detection circuitry. Envelope detection will normally consist of a 1200 Hz filter and a voltage comparator to generate an approximate limited square wave. This is normally derived from a constant mark signal sent by the modulator for Sync acquisition purposes.

Each positive edge that is input to $\overline{\text{Env}}$ will reset both baud timing and the dibit clock to a logic "0". The optimum timing of the positive transition at the $\overline{\text{Env}}$ input will be t_{ED} prior to the falling edge of the dibit clock. Timing is illustrated in Figure 6.

$\overline{\text{Env}}$ will be effective in the training of baud timing and dibit clock only if $\overline{\text{FCar}}$ is in the active low state.

Minimum positive pulse width at the $\overline{\text{Env}}$ is ≥ 2.17 μs.

$\overline{\text{NEW-SYNC}}$ ($\overline{\text{NSync}}$), Pin 11 — This input port is normally controlled by the business machine. If $\overline{\text{FCar}}$ is at an active low, then an active low pulse in excess of 0.84 ms on the $\overline{\text{NSync}}$ lead will put the demodulator into the fast-Sync

*The postive transition of the 1200 Hz signal, present at the $\overline{\text{Env}}$ input, provides a divide-by-20 counter with every other clock. This will cause approximately 8.3 ms of fast training to the incoming signal at the demodulator.

FIGURE 4 — DEMODULATOR SYNC TIMING DIAGRAM

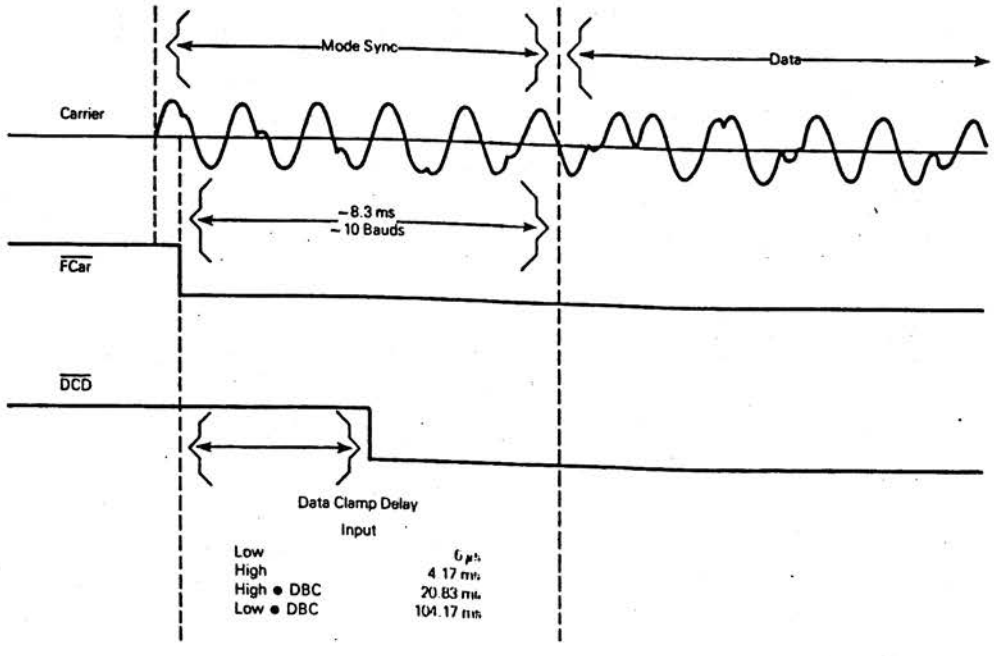


TABLE 1 — DATA-CLAMP DELAY OPTIONS

Option	A	C	$\overline{DCD}$	Data-Clamp Delay
1	1	0	0	6 μs
2	0	0	1	4.17 ms ± 35 μs
3	1	DBC	DBC	20.83 ms ± 35 μs
4	0	DBC	$\overline{DBC}$	104.17 ms ± 35 μs

FIGURE 5 — DATA-CLAMP DELAY DEMULTIPLEXER

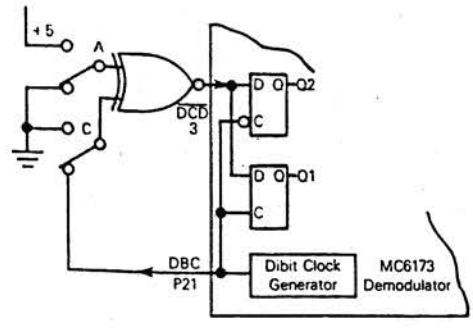
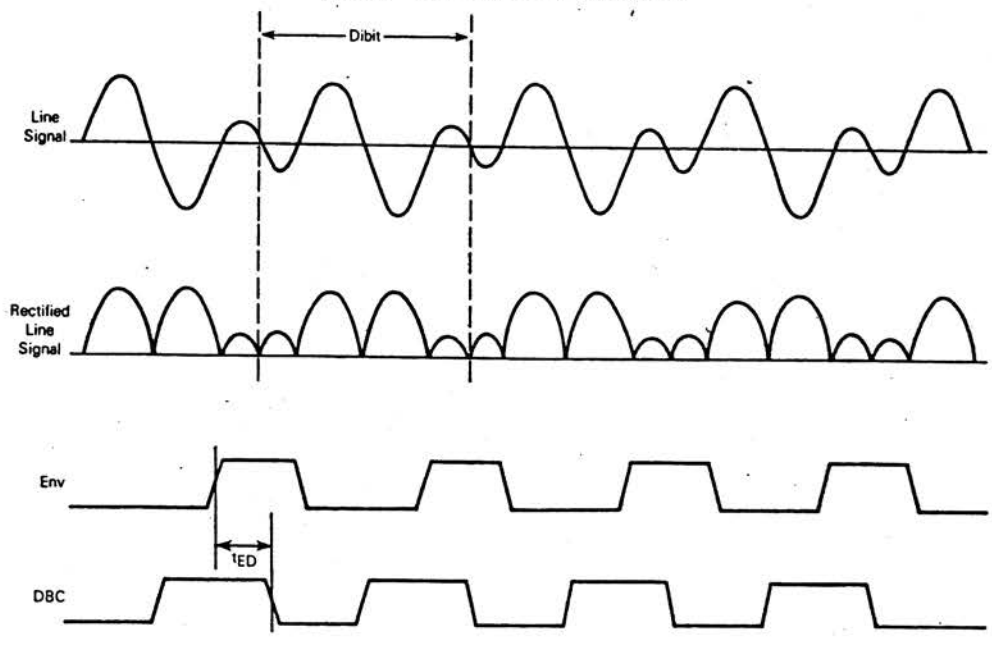


FIGURE 6 — ENVELOPE CLOCK TIMING DIAGRAM



or fast-train mode (these terms are synonymous).

Activation of $\overline{\text{NSync}}$ allows large corrections to be made to both baud and carrier timing similar to initial activation of the $\overline{\text{FCar}}$ lead. These corrections will be applied for approximately 8.3 ms. The receiver must complete the 8.3 ms period of fast Sync before another $\overline{\text{NSync}}$ is recognized.

$\overline{\text{CARRIER-SYNC}}$ ($\overline{\text{CarS}}$), Pin 15 — When $\overline{\text{CarS}}$ is taken to an active low, baud timing will be taken from the $\overline{\text{Env}}$ input. In addition, the slow carrier correction will be doubled in the 2400 baud mode as defined by the data-rate select (DRS) and phase-shift select (PSS) inputs. (This is not the same as the fast training that is incorporated when $\overline{\text{FCar}}$ or $\overline{\text{NSync}}$ are active, which is a changing of the bandwidth of the internal phase-lock loop (PLL)). This widening of the PLL band width will allow a faster search and lock on the 1800 Hz carrier. This Carrier-Sync mode will remain active as long as $\overline{\text{CarS}}$ is held in the active state. The normal application of this option would be to extend the training or Sync time under the mark input data condition that exceeds 8.3 ms.

If $\overline{\text{FCar}}$ is at a logic "1" inactive state, this input is ignored by the demodulator.

A/D CLOCK (ADC), Pin 6 — This output will allow, in a serial format, the six A/D data bits plus sign information to be synchronously clocked into the demodulator. (See Figure 8.)

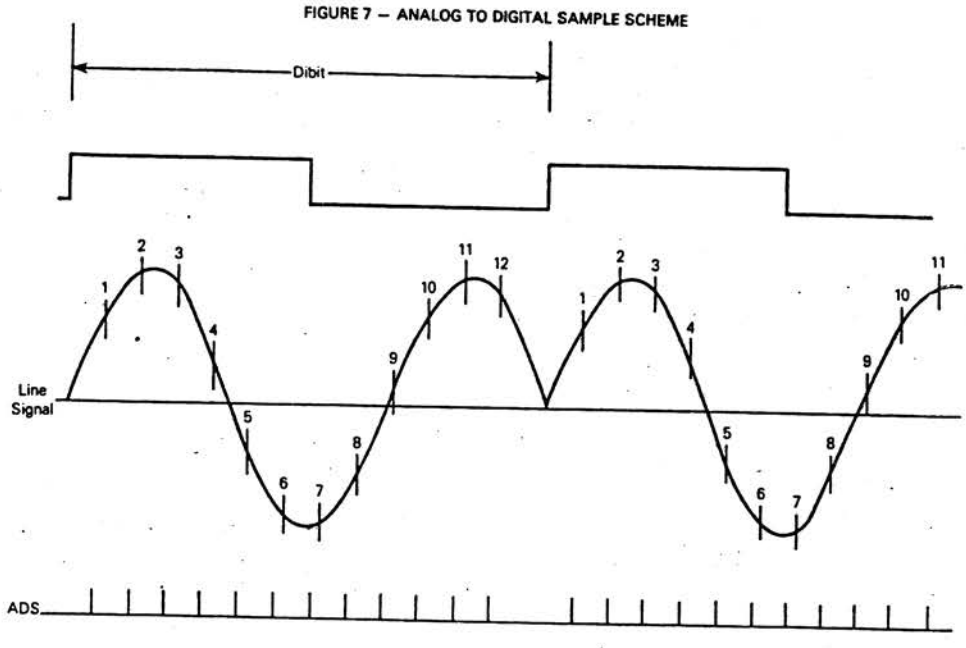
There are nine 1 μs positive pulses occuring at a 460 kHz rate. The first pulse, along with ADS, is used to begin the A/D conversion sequence. The next seven positive edges strobe data serially from the A/D converter to the demodulator input (RDI) enabling the demodulator to properly decode the A/D data.

This signal is also used to clock 0 and 90 degree eye data out of the demodulator. This is described in the Eye Pattern section. When $\overline{\text{TEn}}$ is low, ADC monitors check accumulator output (see $\overline{\text{TEn}}$).

A/D STROBE (ADS), Pin 8 — A positive going, approximately 11 μs, pulse is used as an enable signal for a sample and hold circuit prior to the A/D converter. The negative edge of this pulse is used to start the conversion process. Pulse rate of this signal is 14.4 kHz which allows each dibit to be sampled 12 times. (See Figure 7.) When $\overline{\text{TEn}}$ is low, ADS monitors zero crossings (see $\overline{\text{TEn}}$).

RECEIVER DATA INPUT (RDI), Pin 23 — The digital decode of the line signal magnitude, as sampled by the A/D, is input to the demodulator at this port. The data format is scaled binary. This sign bit occurs on the second A/D clock, followed by six magnitude bits which begin with the most-significant bit as shown in Figure 8. The data is strobed syn-

FIGURE 7 – ANALOG TO DIGITAL SAMPLE SCHEME



chronously with the positive edges of the ADC.

A logic one in the sign bit slot will represent a positive value. The magnitude of the six data bits increases from 000000 to 111111 with all ones always representing the most-positive value as illustrated below:

Sign	MSB					LSB	Value
1	1	1	1	1	1	1	+63
1	0	0	0	0	0	0	0
0	1	1	1	1	1	1	−0
0	0	0	0	0	0	0	−63

RECEIVE DATA OUTPUT (Rx Data), Pin 9 – This pin is the demodulator output for mark and space serial data. Data is synchronous with the receiver clock output with the positive going edge of the receiver clock occurring in the center of the data bit. A mark is represented by a logic high ("1") level except for the conditions described under PSS and $\overline{TPE}$.

The Rx Data output is inhibited in a logic-high level when $\overline{FCar}$ is in the inactive high state. The delay from the positive edge of $\overline{FCar}$ to the inhibiting of data is 2 μs.

RECEIVE CLOCK (Rx Clk), Pin 20 – The receive clock output provides the 2400 Hz ±0.005% timing signal to the business machine for sampling the demodulated received data marks and spaces (Rx Data). Receive clock is present at the demodulator chip output at all times; is not clamped to an inactive state when the carrier detected is not presented on $\overline{FCar}$; nor is Rx Clk clamped by any other combination of inputs to the demodulator.

Timing corrections to the receive clock, that are generated internally, are made following $\overline{FCar}$ going active. As described in $\overline{FCar}$, if $\overline{CarS}$ is held active the receive clock is continuously updated from dibit Sync.

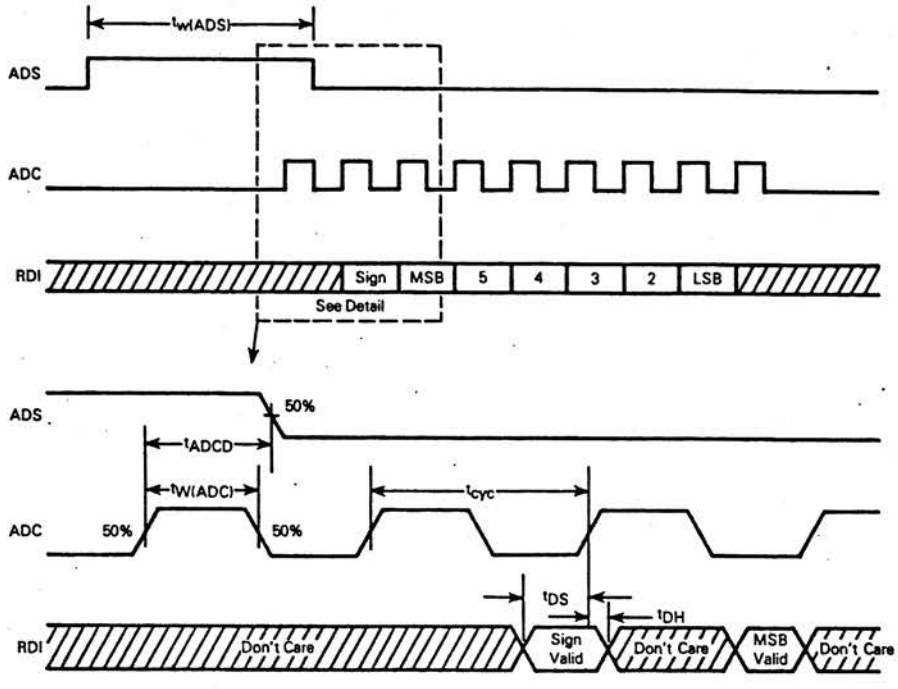
The positive transition of the Receive Clock, which occurs in the middle of the data bit, should be used to strobe data from the demodulator, under normal operating conditions. When $\overline{TPE}$ scrambler/descrambler is being incorporated, then the negative edge of the Rx Clk will occur in the center of the data bit.

Receive Clock will be 2400 bps or 1200 bps depending on the logic input at the DRS input. The Rx Clk edges described above apply to either 2400 bps or 1200 bps data rates.

Under $\overline{TPE}$ active, the Dibit relation to Rx Clk does not change. See Figure 9 for relative timing of Rx Clk, DBC and Rx Data.

Figure 10 depicts the requirements at the demodulator if the data scrambler is being incorporated. The exclusive Nor gating of $\overline{TPE}$ and Rx Clk would then maintain proper phasing of Rx Clk as it goes to the RS-232 driver. This circuit would be required since the positive edge of Receive Clock is a Data Communications Standard.

FIGURE 8 — ANALOG-TO-DIGITAL TIMING DIAGRAM



DATA RATE SELECT (DRS), Pin 24 — The following levels are valid for either phase-shift select:

Logic high equals 2400 bps,
Logic low equals 1200 bps.

PHASE-SHIFT SELECT (PSS), Pin 17 — Option A (CCITT) or option B (U.S.) phase shift can be selected for 2400 bps operation. The input data format and phase shift relationship for these two options are as follows:

Data	PSS = 0 Option A (Degrees)	PSS = 1 Option B (Degrees)
00	0	+45
01	+90	+135
11	+180	+225
10	+270	+315

For 1200 bps operation, option A (CCITT) or option B (U.S.) phase shift can be selected as follows:

Data	PSS = 0 Option A (Degrees)	PSS = 1 Option B (Degrees)
0	+90	+45
1	+270	+225

The phase shifts shown are the difference in phase between the signal at the end of one dibit period and the new signal at the beginning of the next dibit.

If the logic level inputs to PSS are EXORed with DBC (dibit clock) or $\overline{DBC}$, then the test-pattern enable option may be selected and produce the compliment of normal data at Rx Data as explained in the $\overline{TPE}$ description. (See Figure 11.)

$\overline{\text{TEST-PATTERN ENABLE}}$ ($\overline{\text{TPE}}$), Pin 18 — Incorporated in the demodulator is the 511-bit test pattern shift register that is in accord with CCITT specification V52. This is the pattern that is generated by feedback from the 5th and 9th stages of a 9-bit shift register.

When the $\overline{TPE}$ input is allowed to be pulled up internally, there is normal data flow through the receiver. When the $\overline{TPE}$ input is pulled low, the incoming data is passed through this self-synchronous decoder which will produce the inverse of the 511-bit CCITT V52 pattern.

$\overline{TPE}$ works in coordination with PSS. If PSS is directly pulled high or low to represent option A or option B, then the presence of the 511-test pattern at the (RDI) input and $\overline{TPE}$ active will result in logic "1" condition at Rx Data output. If the DBC option is being utilized at the PSS input and $\overline{TPE}$ is active while the 511-bit test pattern is being received, the receiver data output will equal a logic "0". These options (Figure 11) are summarized in Table 2.

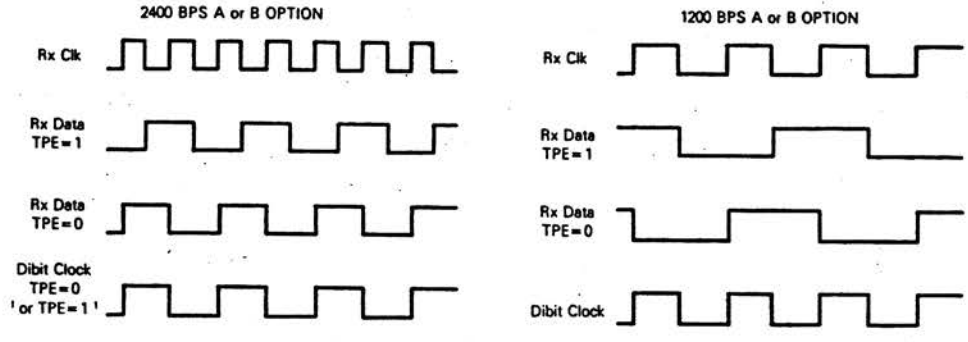
This assumes the modulator is sending the 511-bit test pattern with Rx Data being either a constant mark (logic "1") or space (logic "0"). If a logic "0" is received in options 1 or 2 or a logic "1" is received in options 3 or 4, then a transmission error has occurred. The number of errors-per-unit time is a measure of the transmission line quality.

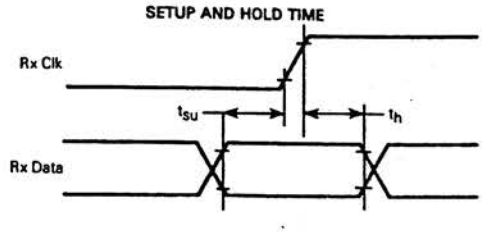
A feature of the above type of pattern detector is that it will be self-synchronizing. It should be pointed out that there will be at least two error counts each time an error is detected.

If the $\overline{TPE}$ input is in the active state, it is important to note that the Rx Clk phase changes. The necessary circuit to regain proper phase is shown in Figure 10.

A scheme for programming the phase-shift select is illustrated in Figure 11. The PSS input may either be a constant high or low level which will produce options 1 and 2. If the input "A" is exclusive ORed with the dibit clock, options 3 and 4 are produced at the same input pin.

FIGURE 9 — CLOCK TIMING DIAGRAM





Note: Timing measurements are referenced to and from a low voltage of 0.8 volts and a high voltage of 2.0 volts, unless otherwise noted.

FIGURE 10 — DEMODULATOR DATA SCRAMBLER RECEIVE CLOCK PHASE CORRECTION REQUIREMENTS

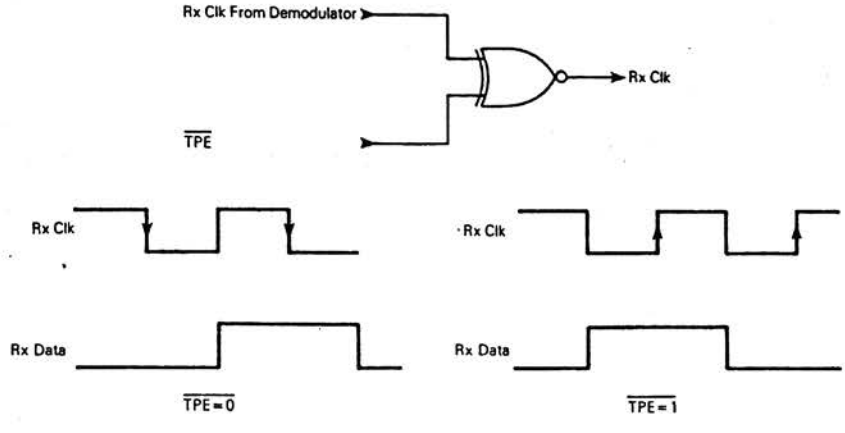


FIGURE 11 — PHASE-SHIFT SELECT DEMULTIPLEXER FOR TEST PATTERN ENABLE

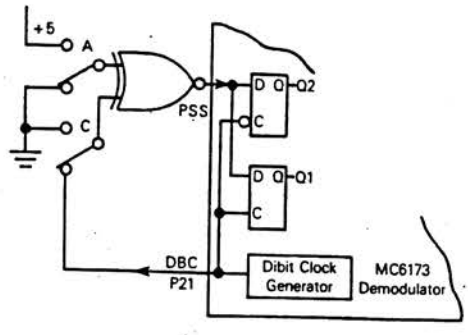


TABLE 2 — TEST PATTERN ENABLE OPTIONS

Option	$\overline{TPE}$	A	C	Phase Option	PSS	Output State
1	0	1	0	A	0	Rx Data Output = 1
2	0	0	0	B	1	Rx Data Output = 1
3	0	1	DBC	A	DBC	Rx Data Output = 0
4	0	0	DBC	B	$\overline{DBC}$	Rx Data Output = 0

CLOCK (Clk), Pin 14 — A 1.8432 MHz signal input ±0.005% is required at this port. The clock requirements are the same as the modulator clock specifications. See Figure 12 for a suggested clock circuit.

The receive clock is generated by dividing down the 1.8432 MHz. Since receive clock accuracy must be at least ±0.005%, the clock source must be of the same accuracy.

TEST-CLOCK (TClk), Pin 13 — This input is used for production testing of the demodulator device. In normal operation this pin should be left open which will enable the internal pullup resistor.

Pin 5	0 Degree Eye
Pin 4	90 Degree Eye
Pin 7	0 Degree Carrier
Pin 19	Carrier Correction

These test outputs are explained in the test enable ($\overline{TEN}$) description below.

$\overline{TEST}$ $\overline{ENABLE}$ ($\overline{TEn}$), Pin 16; 0° Eye, Pin 5; 90° Eye, Pin 4; 0° Car, Pin 7; CCor, Pin 19 — These pins allow the monitoring of ten internal points within the demodulator. A low level on $\overline{TEn}$ is normally associated with testing of the demodulator such as in a production test environment or incoming testing. Activation of $\overline{TEn}$ affects internal timing.

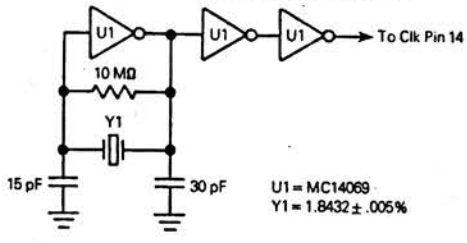
TABLE 3 — INTERNAL MONITORS

Output	$\overline{TEn}$	Function
ADS (Pin 8)	H	See Description Under ADS (Pin 8)
	L	Monitors Zero Crossings
ADC (Pin 6)	H	See Description Under ADC (Pin 6)
	L	Monitors Check Accumulator Output
0 Degree Eye (Pin 5)	H	Monitors 0 Degree Eye 2s Complement Information from 6 Tap Filter
	L	Monitors 0 Degree Eye 2s Complement Information from 12 Tap Filter
90 Degree Eye (Pin 4)	H	Monitors 90 Degree Eye 2s Complement Information from 12 Tap Filter
0°Car (Pin 7)	H	Monitors 0 Degree Carrier
	L	Monitors Check Accumulator Compare Errors
CCor (Pin 19)	H	Monitors Carrier Correction Enable
	L	Monitors Carrier Correction Direction

DIBIT CLOCK (DBC), Pin 21 — This output is a 1200 Hz clock which is derived from incoming data envelope and provides a dibit reference. This signal is representative of "data derived timing." When studying the quality of the demodulated signal, through the use of eye patterns, this output is necessary for proper synchronization of the oscilloscope.

$\overline{TEST}$ $\overline{STROBE}$ ($\overline{TStr}$), Pin 22 — This input is used to facilitate testing of the demodulator during the manufacturing process. It should be left unconnected which will result in

FIGURE 12 — OSCILLATOR CONFIGURATION



the internal pullup resistor causing the high level on this pin.

V_{SS} Pin 1 = The most negative supply, typically ground.

V_{CC} Pin 12 = The most positive supply, typically 5 volts.

DATA SCRAMBLER

The scrambling of data in the data communication environment is not done in an attempt to encrypt information in the normal sense of the word. Rather, the purpose of the scrambling of data is to guarantee that, with respect to the modem carrier, there is always random data on the line with little chance for a long string of "1s" or "0s" to exist. This is particularly important if an adaptive equalizer is being incorporated in the modem as the adaptive equalizer will require reasonably evenly distributed data to optimize its statistical response to the incoming signal. The normally used code is the CCITT 511 sequence which is EXORed with data.

The Motorola 2400 bps modulator contains a CCITT 511-bit test-pattern generator. It does not, however, incorporate the 511 data randomizer or scrambler. To scramble data using the MC6172 modulator, the circuit in Figure 10 must precede the Tx Data input of the modulator. Tx Data is added to the scrambler output pattern; then, the data is delayed by a full data bit before being transmitted by the modem. This assures a proper transmit-data/transmit-clock phase relationship. If the data scrambler is to be an optional feature, then the transmit-data multiplexer would also have to be built. This is selected by the test-pattern enable signal or any other signal that is found suitable.

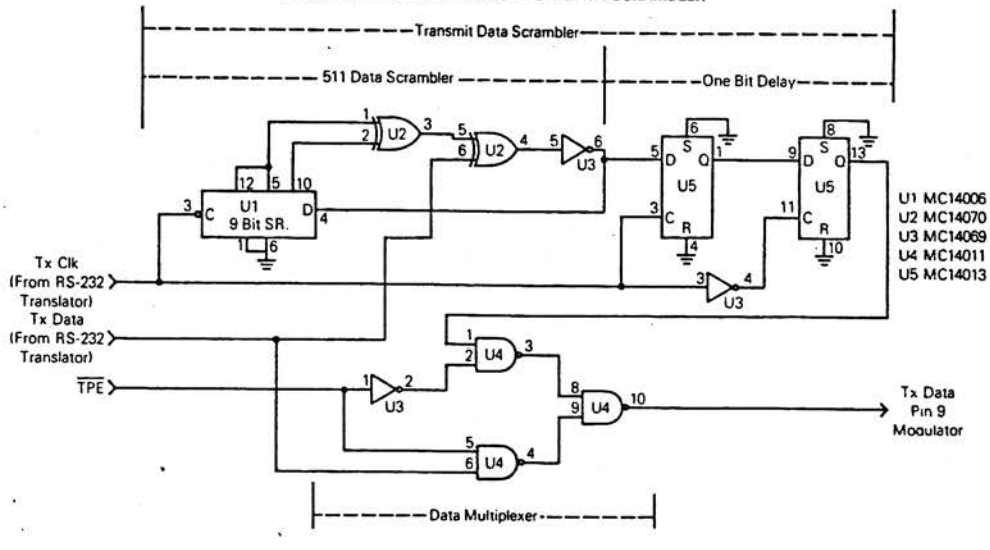
The demodulator does contain a built-in data descrambler which is enabled by the $\overline{TPE}$ input going active. The receive phasing, with respect to data, changes when $\overline{TPE}$ goes active. The exclusive NOR gating of $\overline{TPE}$ and Rx Clk, as shown in Figure 10 will maintain proper phase of Rx Clk. This circuit is required since the clocking of data on a positive edge is a data communications standard.

EYE PATTERN

When performing an evaluation of an 2400 bps modem, one common point of comparison is the quality of the eye patterns produced by the demodulator. The eye pattern may also be used as an indicator of the incoming signal with respect to level and line perturbations. Eye patterns are for test and evaluation only and are not used in the demodulation of the incoming signal.

Timing information in the Motorola 2400 bps demodulator is derived directly from the demodulated data signal. This is referred to as data derived timing. The advantage of data

FIGURE 13 — MODULATOR CCITT 511 DATA SCRAMBLER

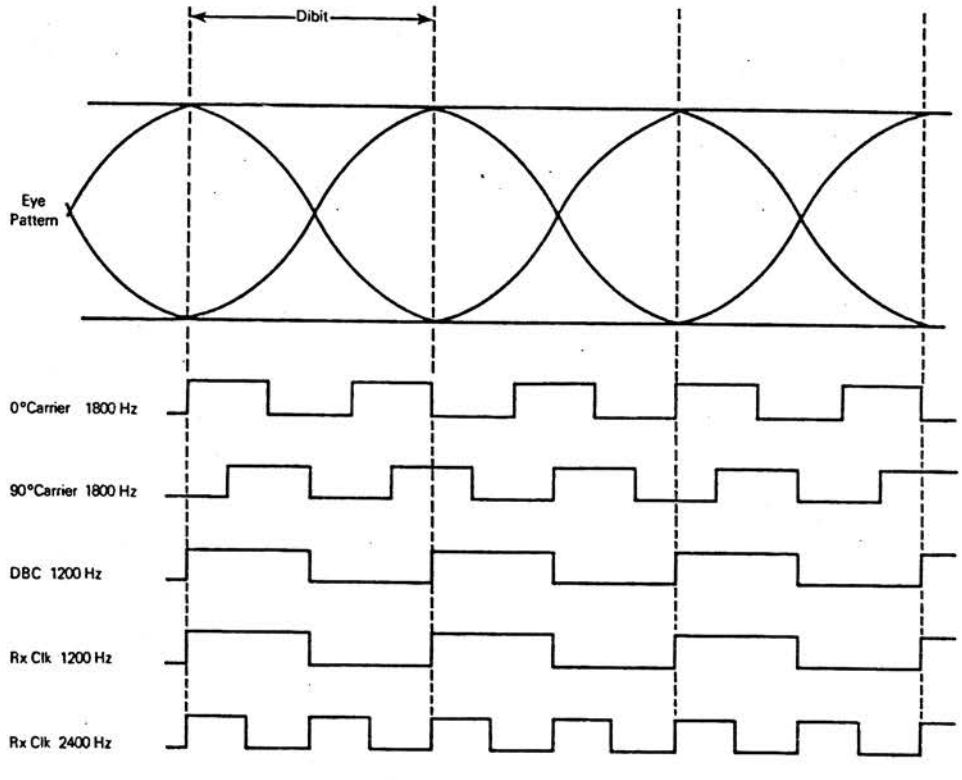


derived time is that it allows data to be sampled at optimum times. The demodulated signals, in differential phase-shift keying, take the form of "eye patterns" as shown in Figure 14. The demodulator, in optimizing its performance for minimum error rates, strobes data at the point of maximum eye opening. The demodulator constantly examines the eye opening to assure that the data sample is being taken at exactly the optimum point. As a result of constantly adjusting timing control, correct sampling is maintained. This technique provides improvements in reception that are significant, especially in a poor communications media environment.

The circuit in Figure 16 is required to observe the eye patterns. This circuit was built using Motorola CMOS devices. The 0 and 90 degree eye data is strobed from pins 4 and 5, respectively, into the shift register by the A/D clock. The A/D strobe then latches the data sample into the "D" type storage devices. The output of the storage devices taken across the scaled resistors will then represent the appropriate value of the sample taken. To properly observe the actual eye patterns, it is necessary to Sync on dibit clock while observing the 0 to 90 degree eye data. Overlaying the two patterns produces a two-level digital-eye pattern from which the quality of the incoming signal may be judged.

Figures 15 thru 17 show a typical receive/demodulator and transmit/modulator circuit, respectively. The transmit filter illustrated in Figure 17 limits the bandwidth of the signal to those frequencies allowed on a telephone line. The receive filter and equalizer in Figure 15 clean up and normalize the incoming signal for the A/D network, 1200 Hz envelope detector, and 1800 Hz carrier detector.

FIGURE 14 — EYE PATTERN



MOTOROLA

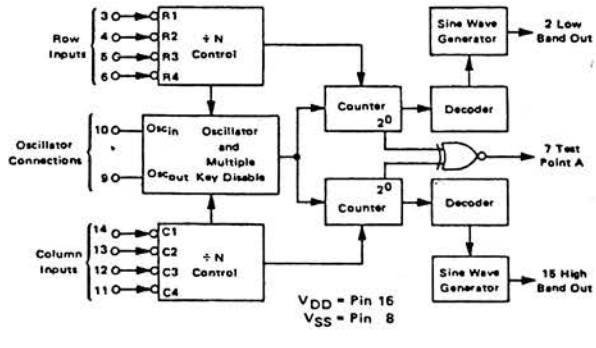
MC14410

2-OF-8 TONE ENCODER

The MC14410 2-of-8 tone encoder is constructed with complementary MOS enhancement mode devices. It is designed to accept digital inputs in a 2-of-8 code format and to digitally synthesize the high and low band sine waves specified by telephone tone dialing systems. The inputs are normally originated from a 4 x 4 matrix keypad, which generates 4 row and 4 column input signals in a 2-of-8 code format (1 row and 1 column are simultaneously connected to V_{SS}). The master clocking for the MC14410 is achieved from a crystal controlled oscillator which is included on the chip. Internal clocks, which operate the logic, are enabled only by one or more row and column signals being activated simultaneously. The two sine wave outputs have NPN bipolar structures on the same substrate which allows for low output impedance and large source currents. Applications of this device include telephone tone dialing, radio and mobile telephones, process control, point-of-sale terminals, and credit card verification terminals.

- Diode Protection on All Inputs
- Noise Immunity = 45% of V_{DD} Typical
- Supply Voltage Range = 4.4 Vdc to 6.0 Vdc
- On-Chip Oscillator (Crystal or External Clock Source may be applied to Pin 10)
- On-Chip Pull-Up Resistors on Row and Column Inputs
- Designed with Multiple Key Lockout (Eliminates Need for Mechanical Lockout in Keypad)
- Two Sine Wave Generators On-Chip
- Frequency Accuracy ±0.2%
- Low Harmonic Distortion
- Single Tone Capability
- Fast Oscillator Turn-On and Turn-Off Times

BLOCK DIAGRAM



CMOS LSI

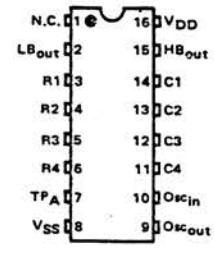
(LOW-POWER COMPLEMENTARY MOS)

2-OF-8 TONE ENCODER

L SUFFIX
CERAMIC PACKAGE
CASE 620

P SUFFIX
PLASTIC PACKAGE
CASE 648

PIN ASSIGNMENT



This device contains circuitry to protect the inputs 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. A destructive high-current mode may occur if V_{in} and V_{out} are not constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$.
Due to the sourcing capability of this circuit, damage can occur to the device if V_{DD} is applied, and the outputs are shorted to V_{SS} and are at a peak sinewave voltage.

MAXIMUM RATINGS (Voltages referenced to V_{SS}, Pin 8.)

Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	−0.5 to +6.0	Vdc
Input Voltage, All Inputs	V_{in}	V_{SS} −0.5 to V_{DD} + 0.5	Vdc
DC Current Drain per Pin	I	10	mAdc
Operating Temperature Range	T_A	−40 to +85	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

ELECTRICAL CHARACTERISTICS

Characteristic	Symbol	V_{DD} Vdc	−40°C Min	−40°C Max	25°C Min	25°C Typ	25°C Max	+85°C Min	+85°C Max	Unit
Supply Voltage	V_{DD}	–	4.4	6.0	4.4	5.0	6.0	4.4	6.0	Vdc
Output Voltage "0" Level Pins 7 and 9	V_{out}	5.0	–	0.05	–	0	0.05	–	0.05	Vdc
"1" Level		5.0	4.95	–	4.95	5.0	–	4.95	–	Vdc
Input Voltage (V_O = 4.5 or 0.5 Vdc) "0" Level	V_{IL}	5.0	–	1.5	–	2.25	1.5	–	1.5	Vdc
(V_O = 0.5 or 4.5 Vdc) "1" Level	V_{IH}	5.0	3.5	–	3.5	2.75	–	3.5	–	Vdc
Output Drive Current (V_{OH} = 2.5 Vdc) Source Pin 7	I_{OH}	5.0	-0.05	–	-0.05	-0.4	–	-0.04	–	mAdc
Pin 9			-0.23	–	-0.20	-1.7	–	-0.16	–	
(V_{OL} = 0.4 Vdc) Sink Pin 7	I_{OL}	5.0	0.05	–	0.05	0.20	–	0.04	–	mAdc
Pin 9			0.23	–	0.20	0.78	–	0.16	–	
Input Pull-Up Resistor Source Current (V_{in} = 0 Vdc) Pins 3-6, 11-14	I_{IL}	6.0	–	140	–	30	100	–	80	μAdc
Input Capacitance (V_{in} = 0 Vdc)	C_{in}	–	–	–	–	5.0	–	–	–	pF
Quiescent Current	I_Q	4.4	–	0.48	–	0.2	0.4	–	0.33	mAdc
		6.0	–	1.3	–	0.55	1.1	–	0.9	
Total Supply Current (Dynamic plus Quiescent) (R_L = 15 kΩ, f = 1 MHz)	I_T	4.4	–	1.7	–	0.7	1.4	–	1.15	mAdc
		6.0	–	3.5	–	1.45	2.9	–	2.4	
Low Band Output Voltage Swing Pin 2 Only (R_L = 100 k)	V_{Lpp}	4.4	400	600	500	600	700	550	750	mVpp
		6.0	800	1000	900	1000	1100	950	1150	
High Band Output Voltage Swing Pin 15 Only (R_L = 100 k)	V_{Hpp}	4.4	600	900	700	850	1000	800	1100	mVpp
		6.0	1000	1400	1100	1350	1500	1200	1600	
Low Band–High Band Voltage Differential	ΔV	5.0	–	–	–	2.5	–	–	–	dB
Low Band–High Band Pin 2,15 Output Impedance AC only	z_o	–	–	–	–	80	–	–	–	Ω
Low Band–High Band Pin 2,15 2nd thru 14th Harmonics (R_L = 15 kΩ)	V_{2H}-V_{14H}	4.4 to 6.0	–	-20	–	-30	-25	–	-25	dB
Maximum Clock Pulse Frequency	f_{cl}	4.4	–	–	–	1.0	–	1.1	–	MHz
Turn-on Time (Power on to oscillation)	t_{on}	5.0	–	–	–	8.0	–	–	–	ms

TABLE 1 – FUNCTIONAL TRUTH TABLE

ACTIVE LOW INPUTS		OUTPUTS	
Activated Row Lines	Activated Column Lines	Low Band Pin 2	High Band Pin 15
None	X**	dc level	dc level
X**	None	dc level	dc level
One	One	f_L*	f_H*
Two or more	One	dc level	f_H*
One	Two or more	f_L*	dc level
Two or more	Two or more	dc level	dc level

*See Table 2
**X = Don't care

TABLE 2 – OUTPUT FREQUENCY TABLE

Input Line Activated (low)	Frequency Generated**	
	f_L (Hz)	f_H (Hz)
R1	697	–
R2	770	–
R3	852	–
R4	941	–
C1	–	1209
C2	–	1336
C3	–	1477
C4	–	1633

**All frequencies are accurate to ±0.2% (crystal tolerance not included).

FIGURE 1 – TYPICAL SINE WAVE OUTPUT
(Pins 2 or 15, No External Filtering)

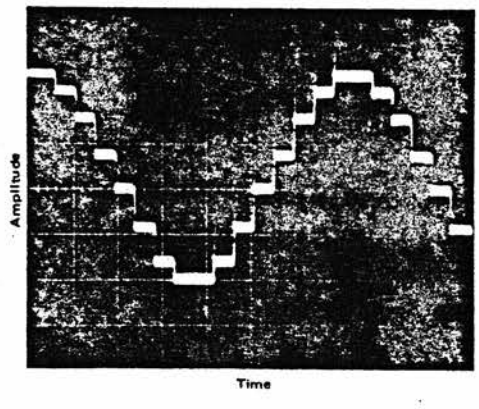


FIGURE 2 – TYPICAL FREQUENCY SPECTRUM
(Pins 2 or 15, No External Filtering)

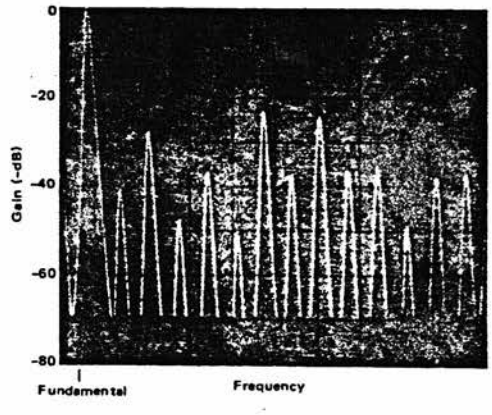


FIGURE 3 – TYPICAL CRYSTAL CIRCUIT

R_f = 15 MΩ ± 10%

CRYSTAL SPECIFICATION

Crystal Mode	Parallel
Frequency	1 MHz ± 0.1%
R_S	540Ω typ
C_0	7.0 pF typ
Temperature Range	-40°C to +85°C
Test Level	1 mW
Test Set	TS-330/TSM or Equivalent

*Recommended Crystals: CTS KNIGHT

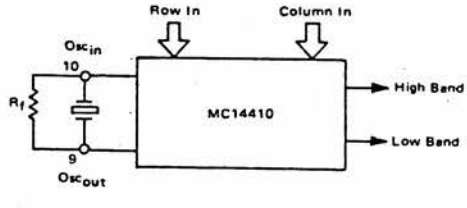


FIGURE 4 – TYPICAL TELEPHONE INTERFACE APPLICATION

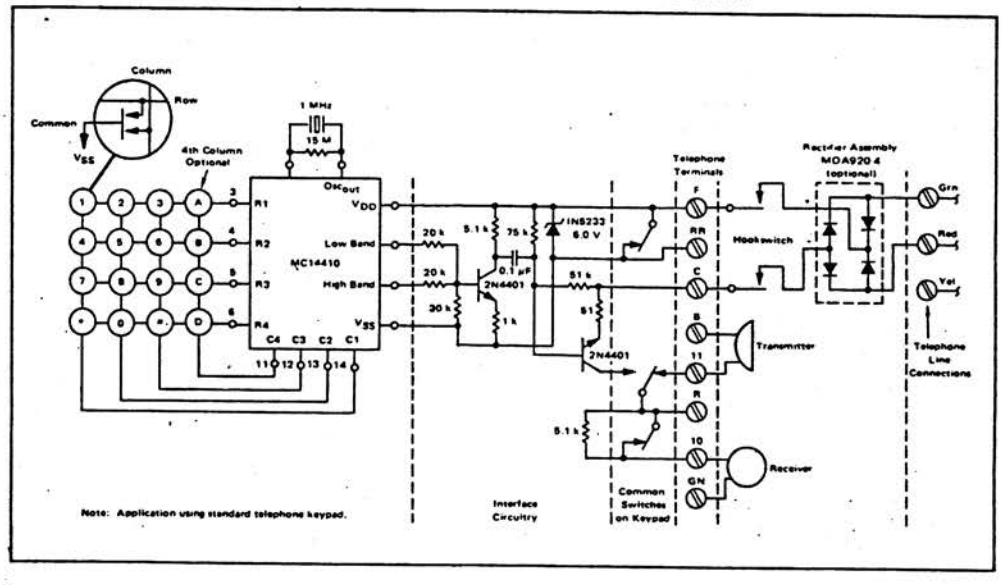


FIGURE 5 – LOW LEVEL OUTPUT TONE GENERATOR APPLICATION

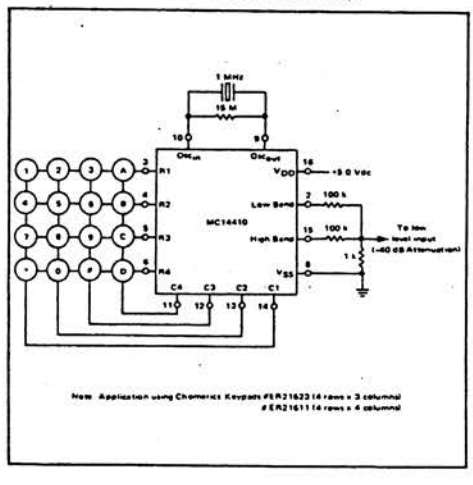
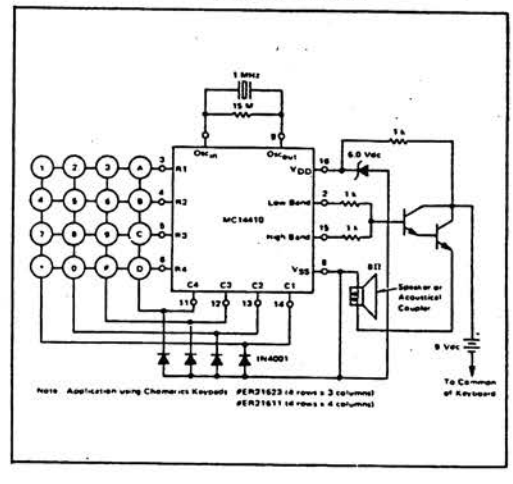


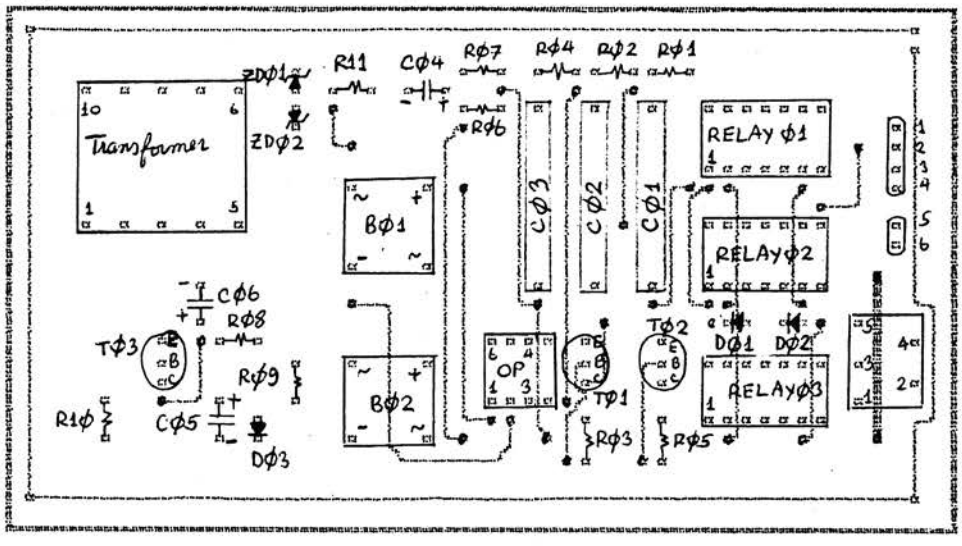
FIGURE 6 – BATTERY POWERED OPERATION (Driving Audio Speaker)



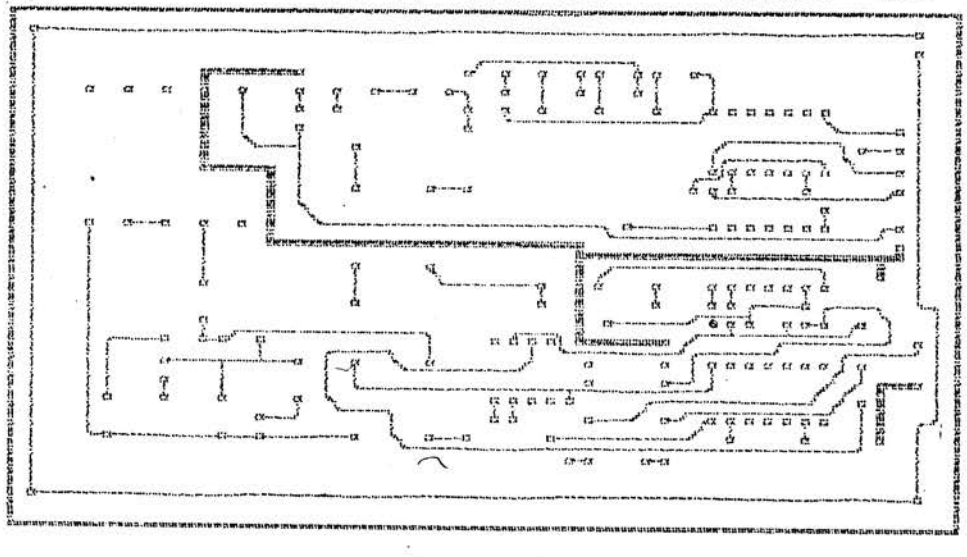
ภาคผนวก ฉ.

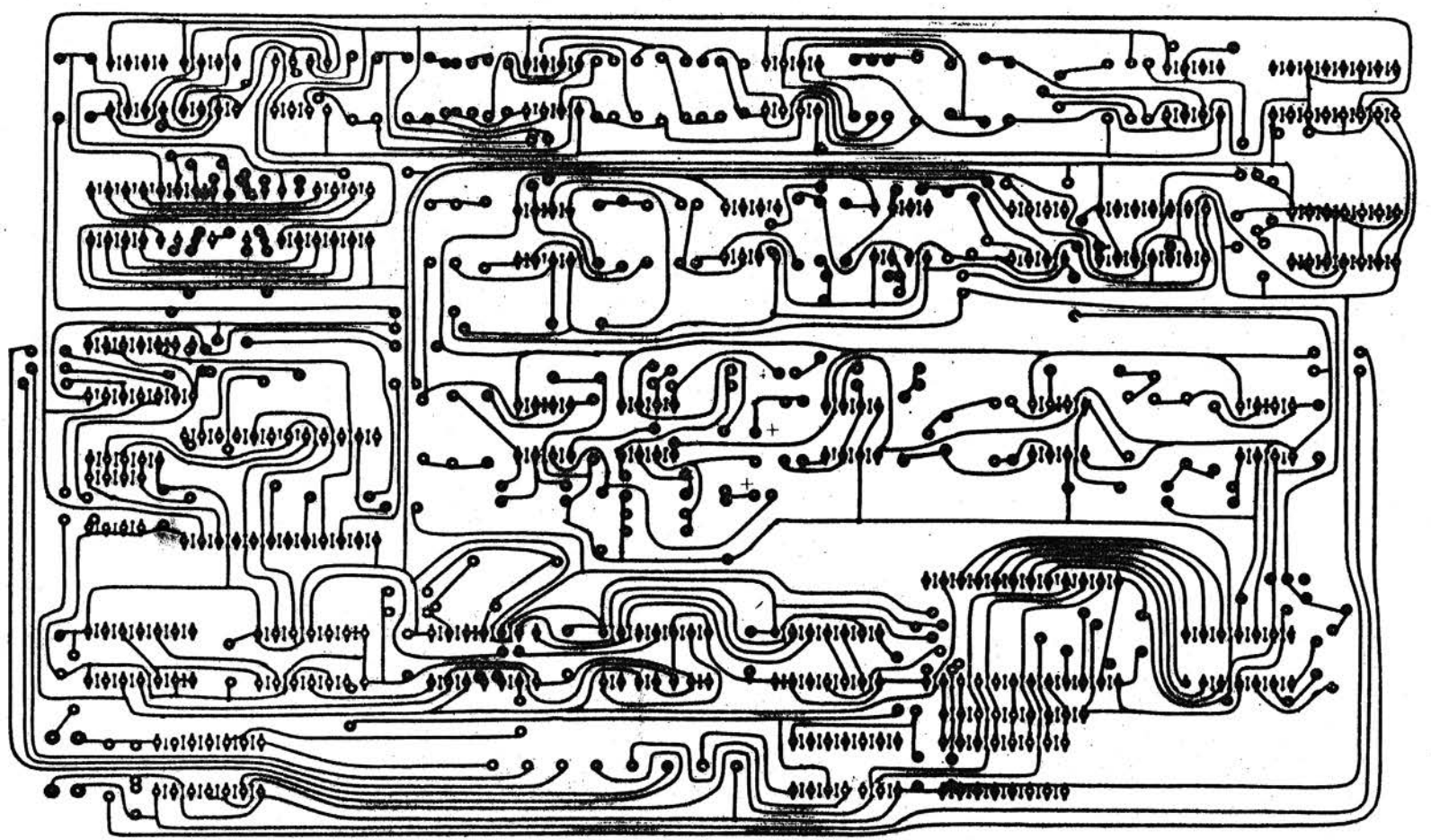
แผ่นวงจรพิมพ์

checkplot v1.0 r4 1 Jan 1980 00:02:17
file: b:joe.pcb upper layer
approx. size: 4.85 by 2.45 in. holes: 172



checkplot v1.0 r4 1 Jan 1980 00:04:21
file: b:joe.pcb lower layer
approx. size: 4.85 by 2.45 in. holes: 172





ภาคผนวก ช.

โปรแกรมควบคุมการทำงาน

```
{ this version has mark comment on procedure concerned modem for development
  if you want to use, connect the modem and mark out all comment}
program testmodem;


const
  Directory=              'PHONE.DAT';
  Protocol=               'DEFAULT.CFG';
  InteractFile=           'INTERACT.IVE';
  ActFile=                'ACTIVITY.TXT';
  DialMode=               'T';
  P=                      1016;
  TimeOut=                  5;
  Line= '------------------------------------------------------------------------------';
  NUL=                      $0;
  SOH=                      $1;
  ETX=                      $3;
  EOT=                      $4;
  ENQ=                      $5;
  ACK=                      $6;
  DLE=                      $10;
  CR=                       $13;
  NAK=                      $15;
  ESC=                      $1B;
  RetryMax=                 10;
  btod:                     array[1..8] of integer= (1,2,4,8,16,32,64,128);

type
  string8=                string[8];
  AnyString=                string[80];
  RegPack=                  record
                              ax,bx,cx,dx,bp,di,si,ds,es,flags: integer;
                            end;
  Data_buff=                array[1..128] of byte;
  block=                    array[1..64] of 0..1;
  PhoneType=                 record
                              Name:        AnyString;
                              Num:         AnyString;
                            end;
  ProtoType=                record
                              BaudRate:    0..7;
                              Parity:      0..3;
                              StopBit:     0..1;
                              WordLength: 0..3;
                            end;
  Passtype=                 record
                              password: string[12];
                              greeting: string[30];
                            end;
var
  kkbuff,
  Diag, Status, Buff:       byte;
  Ch, Choice,PageUD,
  mask,Command,Swap:          char;
  phlength,pk,
  N,i,j,k,Temp,Count,Data,
  valueP,valueK:            integer;
  Flag, Flag1, Flag2:       boolean;
  Regs:                     RegPack;
  CipherText, Plaintext,
  Key, DoorKey:             Block;
  St, Buff1, Buff2,password,greeting,
  FileName, SourName, DestName:AnyString;
  PlainChar, PlainChar1, PlainChar2,
  CipherChar1, CipherChar2,
  KeyChar:                  String8;
  Passrec:                  Passtype;
  PhoneRec:                 PhoneType;
  ProtoRec:                 ProtoType;
  Buffer:                   array[0..255] of record
                              Flag:boolean;
                              Key: array[0..6] of byte;
                            end;
  puzz1,puzz2:              array[1..8] of byte;
  Passfile:                 file of Passtype;
  PhoneFile:                file of PhoneType;
```

```
  ProtoFile:                    file of ProtoType;
  UseFile:                      file of byte;
  Source, Dest:                 file of byte;

{ ********************************
  core procedure:
  use to communicate with hardware
  ******************************** }

  procedure Setup(ProtoRec:ProtoType);
  var
    Temp:       byte;
  begin
    with ProtoRec do Temp:=
      32*BaudRate +8*Parity +4*StopBit +WordLength;
    Regs.ax:= $0000+Temp;Regs.dx:=$0000;
    intr($14,Regs);
  end;

  procedure ModemSetup;
  begin
    with ProtoRec do begin
      BaudRate:=   4;
      Parity:=     3;
      StopBit:=    0;
      WordLength:= 3;
    end;
    Setup(ProtoRec);
  end;

  procedure Check_status(var Regs: Regpack);
  begin
        Regs.ax:=$0300; Regs.dx:=$0000;
        Intr($14,Regs);
  end;

  procedure SetDTR;
  var Buff: byte;
  begin
    Buff:= Port[P+4];
    Port[P+4]:= Buff or 1;
  end;

  procedure ClrDTR;
  var Buff: byte;
  begin
    Buff:= Port[P+4];
    Port[P+4]:= Buff and $fe;
  end;

  procedure SetRTS;
  var Buff: byte;
  begin
    Buff:= Port[P+4];
    Port[P+4]:= Buff or 2;
  end;

  procedure ClrRTS;
  var Buff: byte;
  begin
    Buff:= Port[P+4];
    Port[P+4]:= Buff and $fd;
  end;

  function CTS:boolean;
  var Buff: byte;
  begin
    Buff:= Port[P+6];
    CTS:= ((Buff and 64)=64);
  end;

  function DSR:boolean;
  var Buff: byte;
  begin
    Buff:= Port[P+6];
    DSR:= ((Buff and 32)=32);
  end;
```

```
procedure Outmodem(Temp:   byte);
begin
  repeat
        Check_status(Regs);
  until Regs.ax and $2000 = $2000;
  Regs.ax:= $0100 + ord(Temp);Regs.dx:=$0000;
  Intr($14,Regs);delay(10);
  if Regs.ax shr 8> 127 then begin writeln(^g,' Modem not connect properly, Terminate..'); Halt; End;
end;

procedure Outstmodem(Msg:AnyString);
var
  i: integer;
begin
    for i:=1 to Length(Msg) do begin
    repeat
        Check_status(Regs);
    until Regs.ax and $2000 = $2000;
    Regs.ax:= $0100 + Ord(Msg[i]);Regs.dx:=$0000;
    intr($14,Regs);delay(10);
    if Regs.ax shr 8> 127 then begin writeln(^g,' Modem not connect properly, Terminate..'); Halt; End;
  end;
end;

procedure InModem(var Temp: byte);
begin
     repeat
         Check_status(Regs);
     until Regs.ax and $0100 = $0100;
     Regs.ax:= $0200;Regs.dx:=$0000;
     Intr($14,Regs);
     Temp:= Regs.ax and $00ff;
end;


procedure Attention;
begin
  ModemSetup;
  OutStModem('   +++   ');
  OutStModem('AT');
end;

procedure Cancel;
begin
  Attention;
  OutStModem('K');
end;


function Exist(Filename:anystring):boolean;
var UseFile:  file of byte;
begin
   assign(Usefile,Filename);
   {$i-} reset(UseFile); {$i+}
   Exist:= (IOResult = 0);
   close(UseFIle);
end;

procedure txhdx;
begin
   setrts;delay(200);
end;

procedure rxhdx;
begin
   delay(150);clrrts;
end;

function VerifyTarget:boolean;
begin
   outmodem(ENQ);
   rxhdx;
   inmodem(status);
   txhdx;
   if status = ACK then VerifyTarget:= true else begin
      writeln(^g,'error: Verify Target error');
      Cancel;
  end;
end;
```

```
Function VerifySource(status: byte):boolean;
begin
    rxhdx;
    if status = ENQ then
    txhdx;
    outmodem(ACK);
end;

procedure printdata;
begin
   write(chr(status));
end;

procedure message_transfer(var j,k,VertRecCheck: byte);
begin
    txhdx;
    outmodem(SOH);write(chr(SOH));VertRecCheck:= 0;
    repeat
        read(UseFile,buff);
        VertRecCheck:= VertRecCheck xor Buff;
        outmodem(buff);write(chr(buff));
        k:=Pred(k);
    until ( k=0 ) or (buff = 26);
end;

procedure Recv_message(Var i,u: integer;Var Recv_buff:  Data_buff;
Var Flag: boolean);
var   z:          integer;
     VertRecCheck:   byte;
begin
    rxhdx;
    Flag:=true;VertRecCheck:=0;i:=1;
    repeat
         status:=0;
         check_status(regs);
         if regs.ax and $0100 = $0100 then begin
            inmodem(status);
            if status = enq then
               begin outmodem (u);rxhdx;end;
            end;
    until (status=soh);
    repeat
         inmodem(status);
         if Regs.ax and $0e00 > 0 then flag:=false else flag:=true;
         if ( status <> EOT ) and ( status <> ETX ) then begin
            Recv_buff[i]:=status;i:=succ(i);write(chr(status));
            VertRecCheck:=VertRecCheck xor status;
         end;
    until  ( status = EOT ) or ( status = ETX );
    Recv_buff[i]:=status;
    inmodem(status);
    if status <> VertRecCheck then flag:=false;
end;

procedure Filetranf(c: integer);
const
    Maxbuff = 128;
Var
    j,k,VertRecCheck:                    byte;

procedure reacknowledge;
var z:   integer;
begin
    z:=0;status:=0;
    rxhdx;
    repeat
       check_status(regs);
       if regs.ax and $0100 =$0100 then inmodem(status)
       else begin delay(1000);z:=succ(z);
                  txhdx;outmodem(enq);rxhdx; end;
   until (status= ack) or (status=nak) or (z=timeout);
end;

procedure Pri_Calling_station;
begin
    Clrscr;status:=0;
    writeln('enter filename which you want to transfer');
    Readln(FileName);
```

```
      assign(Usefile,Filename);
      if Exist(Filename) then begin
         txhdx;outmodem(enq);delay(1000);outmodem(enq);
         outstmodem(filename);
         outmodem(NUL);
         rxhdx;inmodem(status);
       if status = ack then begin
           reset(UseFile);
           n:=Filesize(Usefile);
           i:=n;j:=1;k:=Maxbuff;
           delay(3000);
           repeat
             seek(Usefile,n-i);
             message_transfer(j,k,VertRecCheck);
             outmodem(ETX);write(chr(etx));
             outmodem(VertRecCheck);write(chr(vertreccheck));
             reacknowledge;writeln(status);
             if status = ack then begin
                i:=i-Maxbuff;j:=succ(j);
                  if i<=Maxbuff then begin
                     repeat
                       k:=i;
                       seek(UseFile,n-i);message_transfer(j,k,VertRecCheck);
                       outmodem(EOT);write(chr(EOT));
                       outmodem(VertRecCheck);write(chr(VertRecCheck));
                       i:=0;
                      until i=0;
                    end
                 end;
              k:=Maxbuff;
           until i<=0;
        end;
        close(Usefile);{ Called_station}
end;
end;
procedure Pri_Called_station;
var  Recv_buff:  Data_buff;
     k,i,u:        integer;
     Usefile:    file;
begin
     Clrscr;filename:=' ';
     rxhdx;
     repeat
        inmodem(status);
     until status=enq;
     writeln('verify o.k.');
     repeat
        inmodem(status);Filename:=Filename+chr(status);
     until status = NUL ;
     txhdx;u:=ack;
     outmodem(u);
     writeln('Now you have Transfered File');
     write('Filename is  ');
     filename:=copy(filename,2,length(filename)-1);
     writeln(filename);
     assign(Usefile,filename);rewrite(Usefile);
     repeat
         Recv_message(i,u,Recv_buff,Flag);
         if flag = true then begin
            {k:=1;
            repeat
               write(chr(Recv_buff[k]));k:=succ(k);
               blockwrite(Usefile,Recv_buff,1);
            until k=i;}
            u:=ack;
            end
             else u:=nak;
             txhdx;
             outmodem(u);
      until Recv_buff[i] = EOT;
      close(Usefile);
end;

procedure Sec_Calling_station;
begin
    Clrscr;
    writeln(' Remote service request file ');
    filename:='';
    repeat
```

```
      inmodem(status);
    until status=enq;
    repeat
       inmodem(status);
       filename:=filename+chr(status);
    until status=nul;
    txhdx;
    outmodem(ack);delay(3000);
    filename:=copy(filename,1,length(filename)-1);
    assign(Usefile,Filename);
    if Exist(Filename) then begin
         reset(UseFile);
         n:=Filesize(Usefile);
         i:=n;j:=1;k:=Maxbuff;
         repeat
           seek(Usefile,n-i);
           message_transfer(j,k,VertRecCheck);
           outmodem(ETX);write(chr(etx));
           outmodem(VertRecCheck);write(chr(VertrecCheck));
           reacknowledge;writeln(status);
           if status = ACK then begin
              i:=i-Maxbuff;j:=succ(j);
                if i<=Maxbuff then begin
                   repeat
                     k:=i;
                     seek(UseFile,n-i);message_transfer(j,k,VertRecCheck);
                     outmodem(EOT);write(chr(EOT));
                     outmodem(VertRecCheck);write(chr(VertRecCheck));
                     i:=0;
                   until i=0;
                 end;
                end;
                k:=Maxbuff;
         until i<=0;
       end;
       close(Usefile);{ Called_station}
end;

procedure Sec_Called_station;
var  Recv_buff:  Data_buff;
     k,i,u:        integer;
     Usefile:     file;
begin
     Clrscr;
     write(' Enter filename which you want to request ');
     write(' ');
     readln(filename);
     txhdx;
     outmodem(enq);
     outstmodem(filename);
     outmodem(nul);
     reacknowledge;
     if status=ack then begin
     assign(Usefile,filename);rewrite(Usefile);
     repeat
         Recv_message(i,u,Recv_buff,Flag);
         if flag = true then begin
            {k:=1;
            repeat
               write(chr(Recv_buff[k]));k:=succ(k);
               blockwrite(Usefile,Recv_buff,1);
            until k=i;}
              u:=ACK;
            end else
              u:=NAK;
            txhdx;
            outmodem(u);
      until Recv_buff[i] = EOT;
      close(Usefile);
 end;
 end;

 begin
 case c of
     1:  pri_calling_station;
     2:  pri_called_station;
     3:  sec_called_station;
     4:  sec_calling_station;
     else writeln(' Transfer Status Error ');
```

```
  end; {end case}
  end;

{ ********************************
  miscellanous procedure:
  general usage troughtout part of supporting procedure
  ******************************** }

  procedure Center(text:AnyString);
  begin
    writeln(text:40+length(text) div 2);
  end;

  procedure CenterF(text:AnyString;var r:integer);
  var
    a: AnyString;
  begin
    lowvideo;
    a:= Copy(text,1,r-1);
    write('':30,a);
    a:= Copy(text,r,1);
    normvideo;
    write(a);
    lowvideo;
    a:= Copy(text,r+1,20);
    writeln(a);
  end;

  procedure Wait;
  begin
    GotoXY(1,23); ClrEol; writeln; ClrEol;
    write('':5,'Press any key to continue.....');
    repeat until KeyPressed;
  end;

  procedure Beep;
  begin
    sound(2000);
    delay(50);
    nosound;
  end;

  procedure Alert;
  var i,j:integer;
  begin
    for j:=1 to 3 do begin
      for i:=1000 to 2000 do begin
        sound(i);
        sound(3000-i);
      end;
      nosound;
      delay(200);
    end;
  end;

  function Time:AnyString;
  var
    St1,St2,St3: string[2];
    Hour,Min,Sec,Frac: integer;
  begin
    with Regs do
    begin
      AX:=$2c00;
      MsDos(Regs);
      Str(hi(CX):2,St1);
      Str(lo(CX):2,St2);
      Str(hi(DX):2,St3);
    end;
    Time:=St1+St2+St3;
  end;

  function Date:AnyString;
  const
    Day: array[0..6] of string[3]=
      ('Sun','Mon','Tue','Wed','Thu','Fri','Sat');
    Month: array[1..12] of string[3]=
      ('Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec');
  type
    RegPack= record
```

```
                AX,BX,CX,DX,BP,SI,DI,DS,ES,Flags: integer;
              end;
  var
    St1: string[2];
    St2: string[4];
    Regs: RegPack;
  begin
    with Regs do
    begin
      AX:=$2a00;
      MsDos(Regs);
      Str(lo(DX):2,St1);
      Str(hi(DX):2,St2);
      Date:=St2+St1;
    end
  end;

  procedure Border(x1,y1,x2,y2:integer);
  var i:integer;
  begin
    window(1,1,80,25);
    gotoxy(x1,y1);
    write(chr(201));
    for i:=x1 to x2-2 do write(chr(205));
    write(chr(187));
    for i:=y1+1 to y2-1 do begin
      gotoxy(x1,i); write(chr(186));
      gotoxy(x2,i); write(chr(186));
    end;
    gotoxy(x1,y2);
    write(chr(200));
    for i:=x1 to x2-2 do write(chr(205));
    write(chr(188));
    window(x1+1,y1+1,x2-1,y2-1);
    clrscr;
  end;

  procedure Activity(St:AnyString);
  var UseFile: file of byte;
  begin
    assign(UseFile,ActFile);
    if Exist(ActFile) then reset(UseFile) else rewrite(UseFile);
    Seek(UseFile,FileSize(UseFile));
    while (Length(St)>0) begin
      Buff:= ord(St[1]);
      write(UseFile,Buff);
      Delete(St,1,1);
    end;
    Buff:=13;       { return/line feed to mark EoLn }
    write(UseFile,Buff);
    Buff:=10;
    write(UseFile,Buff);
    close(UseFile);
  end;

  procedure ActNote;
  begin
{
    Border(1,10,80,24);
    write('Do you want to take some note ? (y/n) ');
    read(Kbd,Ch);
    if upcase(Ch)='Y' then begin
      writeln; writeln('  take note: enter blank line to end note');
      repeat
        Buff1:='';
        write('->'); readln(Buff1);
        if Buff1<>'' then Activity(Buff1);
      until Buff1='';
    end;
    window(1,1,80,25);
    Activity('');
}
  end;


{ ***********************************
  supportting procedure:
  use to support main part of subroutine
  *********************************** }
```

```
procedure StampDatim;
begin
  Activity(Date+' '+Time+'>');
end;

procedure GetConfig;
var  Confile: anystring;
begin
  Confile:='';
  writeln(' Default Config. is Protocol');
  writeln(' Enter Protocol ');
  write(' Configuration ( .CFG);');
  readln(ConFile);
  if Exist(ConFile) then
    assign(ProtoFile,ConFile)
  else
    assign(ProtoFile,Protocol);
  reset(ProtoFile);
  read(ProtoFile,ProtoRec);
end;

procedure GetKeyWord;
begin
  KeyChar:='';
  repeat
    read(Kbd,Ch);
    if Ch<>chr(13) then KeyChar:=KeyChar+Ch;
    write('*');
    sound(50);delay(50);nosound;
  until Ch=chr(13);
  delete(KeyChar,8,10);
  writeln;
  if Exist(KeyChar+'.key') then begin
    assign(UseFile,KeyChar+'.key');
    reset(UseFile);
    KeyChar:='';
    repeat
      read(UseFile,Buff);
      KeyChar:= KeyChar+ chr(Buff);
    until Eof(UseFile);
    close(UseFile);
  end;
end;

procedure GotNumber(Num:AnyString);
var Stop,i: integer;
begin
  St:= '';
  if Pos(',',PhoneRec.Num)= 0
    then Stop:= Length(PhoneRec.Num)
    else Stop:= Pos(',',PhoneRec.Num)-1;
  for i:=1 to Stop do
    if PhoneRec.Num[i] in ['0'..'9'] then St:= St+PhoneRec.Num[i];
  Delete(PhoneRec.Num,1,Stop+1);
end;

procedure GetPhone;
const
  MaxLine=          12;
type
  anystring=        string[20];
  PhonePtr=         ^PhoneTypePtr;
  PhoneTypePtr=     record
                      Name: AnyString;
                      Num:  AnyString;
                      Prev: PhonePtr;
                      Next: PhonePtr;
                    end;
var
  Start, Stop,
  Current,
  First,
  Last,
  PhoneRecPtr:      PhonePtr;
  i,Line,High,Pad: integer;
  HeapTop:          ^integer;

  procedure Heading;
```

```
begin
  lowvideo;
  for i:=1 to 110 do
    write(chr(219));
  normvideo;
  write('TELEPHONE DIRECTORY');
  lowvideo;
  for i:=1 to 111 do
    write(chr(219));
  normvideo;
end;

procedure Footing;
begin
  Border(1,21,80,24);
  write  (':10);
  write(chr(24));lowvideo;write('   Up',':5);normvideo;
  write(chr(25));lowvideo;write('   Dn',':5);normvideo;
  writeln('Home ',':5,'End  ',':5,'PgUp ',':5,'PgDn');
  write  (':10);
  write('Ins');lowvideo;write(' Insert',':1);normvideo;
  write('Del');lowvideo;write(' Delete',':1);normvideo;
  write('F1');lowvideo;write(' Select',':1);normvideo;
  write('F2');lowvideo;write(' Quit',');
end;

procedure Show(First:PhonePtr; High:integer);
begin
  normvideo;
  PhoneRecPtr:=First;
  Line:= 0;
  gotoXY(1,1);
  repeat
    Line:= succ(Line);
    if Line=High then lowvideo;
    clreol;
    Pad:= 5;
    for i:= Length(PhoneRecPtr^.Name) to 40 do Pad:= succ(Pad);
    writeln(Line:2,':',':5,PhoneRecPtr^.Name,':Pad,Copy(PhoneRecPtr^.Num,1,15));
    Last:= PhoneRecPtr;
    PhoneRecPtr:= PhoneRecPtr^.Next;
    if Line=High then normvideo;
  until (PhoneRecPtr=nil) or (Line=MaxLine);
  normvideo;
  repeat
    clreol;
    Line:=succ(Line);
  until Line>MaxLine;
end;

begin
  window(1,1,80,25);
  clrscr;
  mark(HeapTop);
  Heading;
  Footing;
  window(1,4,80,25);
  new(PhoneRecPtr);
  First:= PhoneRecPtr;
  Start:= First;
  assign(PhoneFile,Directory);
  reset(PhoneFile);
  read(PhoneFile,PhoneRec);
  PhoneRecPtr^.Name:= PhoneRec.Name;
  PhoneRecPtr^.Num:=  PhoneRec.Num;
  PhoneRecPtr^.Prev:= PhoneRecPtr;
  High:=1; {use high as flag to get stop}
  repeat
    new(PhoneRecPtr^.Next);
    PhoneRecPtr^.Next^.Prev:= PhoneRecPtr;
    PhoneRecPtr:= PhoneRecPtr^.Next;
    if High =1 then begin
      Line:= FileSize(PhoneFile)- FilePos(PhoneFile);
      if Line<=12 then begin
        High:=2;
        Stop:= PhoneRecPtr;
      end;
    end;
    read(PhoneFile,PhoneRec);
```

```
  PhoneRecPtr^.Name:= PhoneRec.Name;
  PhoneRecPtr^.Num:=  PhoneRec.Num;
  Line:= succ(Line);
until Eof(PhoneFile);
PhoneRecPtr^.Next:= nil;
Current:= First; High:= 1;
Show(First,High);
repeat
  read(Kbd,Ch); read(Kbd,Ch);
  case Ch of
    #72:begin {up}
          if Current^.Prev<>Current then begin
            if (High>1) then begin
              normvideo;
              gotoxy(1,High);
              clreol;
              Pad:= 5;
              for i:= Length(Current^.Name) to 40 do Pad:= succ(Pad);
              writeln(High:2,':',':5,Current^.Name,':Pad,Copy(Current^.Num,1,15));
              High:= pred(High);
              Current:= Current^.Prev;
              lowvideo;
              gotoxy(1,High);
              clreol;
              Pad:= 5;
              for i:= Length(Current^.Name) to 40 do Pad:= succ(Pad);
              writeln(High:2,':',':5,Current^.Name,':Pad,Copy(Current^.Num,1,15));
              normvideo;
            end else begin
              Current:= Current^.Prev;
              High:=0;
              repeat
                First:= First^.Prev;
                High:=succ(High);
              until (High=4) or (First^.Prev=First);
              Show(First,High);
            end;
          end else Beep;
        end;
    #80:begin {down}
          if Current^.Next<>nil then begin
            if (High <MaxLine) then begin
              normvideo;
              gotoxy(1,high);
              clreol;
              Pad:= 5;
              for i:= Length(Current^.Name) to 40 do Pad:= succ(Pad);
              writeln(High:2,':',':5,Current^.Name,':Pad,Copy(Current^.Num,1,15));
              High:= succ(High);
              Current:= Current^.Next;
              lowvideo;
              gotoxy(1,high);
              clreol;
              Pad:= 5;
              for i:= Length(Current^.Name) to 40 do Pad:= succ(Pad);
              writeln(High:2,':',':5,Current^.Name,':Pad,Copy(Current^.Num,1,15));
              normvideo;
            end else begin
              Current:= Current^.Next;
              High:= MaxLine+1;
              repeat
                First:= First^.Next;
                Last:= Last^.Next;
                High:= pred(High);
              until (High=MaxLine-3) or (Last^.Next=nil);
              Show(First,High);
            end;
          end else Beep;
        end;
    #82:begin { enter new name }
          Border(29,10,80,13);
          write('Name ---'); readln(PhoneRec.Name);
          write('Number -'); read  (PhoneRec.Num);
          Seek(PhoneFile,FileSize(PhoneFile));
          write(PhoneFile,PhoneRec);
        end;
    #83:begin    { Delete }
          PhoneRecPtr:=Current;
          PhoneRecPtr^.Prev^.Next:=Current^.Next;
```

```
            PhoneRecPtr^.Next^.Prev:=Current^.Prev;
            Dispose(PhoneRecPtr);
            write(PhoneFile,Phonerec);
            repeat
              Current:=Current^.Prev;
              High:=Pred(High);
            until high=1;
            First:=Current;
            Show(First,High);
          end;
        #59:begin   { Dial }
            PhoneRec.Name:= Current^.Name;
            PhoneRec.Num:=  Current^.Num;
          end;
        #81:if Last^.Next<>nil then begin   { page down }
            High:= MaxLine+1;
            repeat
              First:= First^.Next;
              Last:= Last^.Next;
              High:= pred(High);
            until (High=1) or (Last^.Next=nil);
            High:=1;
            Current:=First;
            Show(First,High);
          end else begin
            write(^g);
            Current:=Start;
            First:=Start;
            High:=1;
            Show(First,High);
          end;
        #73:if First^.Prev<>First then begin    { Page up }
            High:= MaxLine+1;
            repeat
              First:= First^.Prev;
              High:= pred(High);
            until (High=1) or (First^.Prev=First);
            High:=1;
            Current:=First;
            Show(First,High);
          end else begin
            write(^g);
            Current:=Stop;
            First:=Stop;
            High:=1;
            Show(First,High);
          end;
        #71:begin  {home}
            write(^g);
            Current:=Start;
            First:=Start;
            High:=1;
            Show(First,High);
          end;
        #79:begin  {end}
            write(^g);
            Current:=Stop;
            First:=Stop;
            High:=1;
            Show(First,High);
          end;
      end; {case}
    until (Ch=#59) or (Ch=#82) or (ch=#60);
    close(PhoneFile);
    Border(20,10,70,15);
  end; {getPhone}

  function Dial(TimeOut:integer): boolean;
  begin
    writeln(' >>> Try number: ',Count);
    GotNumber(PhoneRec.Num);
    repeat
      Attention;
      OutStModem('D'+DialMode+St);
      TimeOut:=pred(TimeOut);
      InModem(Status);
    until (Status = $43) or (status = $4e) or (TimeOut=0);
    if  Status  = $4e then begin
      writeln(^g,'ERROR: cannot dial to target');
```

```
    Cancel;
  end;
  if  Status  = $43 then Dial:=true else Dial:=false;
  Count:= succ(Count);
end;

procedure Display(St:AnyString);
var
  i:integer;
begin
  if(Diag and $40) >1 then write(Lst,St);
  if(Diag and $20) >1 then write(St);
  if(Diag and $10) >1 then
    for i:=1 to Length(St) do begin
      Status:= ord(St[i]);
      write(UseFile,Status);
    end;
end;

procedure SetupEquipment;
begin
  if (Diag and $10)>1 then begin
    assign(UseFile,InteractFile);
    if Exist(InteractFile) then reset(UseFile) else rewrite(UseFile);
    seek(UseFile,filesize(UseFile));
    Diag:= Diag and $df;
    Display('A:'+Date+' '+Time+')'+^m^j);
    Diag:= Diag or $20;
  end else begin
    {$i-} close(UseFile); {$i+}
    if IoResult<>0 then Beep;
  end;
end;

procedure InteractiveScreen;
var func:boolean;
    oldx, oldy: byte;
    inactchar:  anystring;
  procedure ShowStat;
  begin
    window(1,1,80,25);
    gotoxy(1,24);lowvideo;
                write('F2 Voice/Data   ');
                write('F7 Toggle Printer   ');
                write('F8 Toggle File    ');
                write('F10 Exit   ');
    gotoxy(1,25);normvideo;
    write('Terminal: ');
    if (Diag and $20)>1 then write('On / ')
    else                     write('Off/ ');
    write('Printer : ');
    if (Diag and $40)>1 then write('On / ')
    else                     write('Off/ ');
    write('DiskFile: ');
    if (Diag and $10)>1 then write('On /')
    else                     write('Off/');
  end;
begin
  flag:= true; {true= data mode, false= voice mode}
  Buff1:= ''; Buff2:='';
  ClrScr; Center('Interactive Screen');
  ShowStat;
  Border(1,2,80,23);
  repeat;
    func:=false;
    if KeyPressed then begin
      read(Kbd,Ch);
      if ord(Ch)=ESC then begin
        func:=true;
        read(Kbd,Ch);
        case Ch of
          #60:begin {switch voice/data}
                flag:= not(flag);
                clrscr;
                gotoxy(1,20);
                OutStModem('ATH');outmodem(0);outmodem($13);
                if flag= false then Center('-----Voice-----')
```

```
                else Center ('-----Data-----');
              end;
          #65:begin {Printer}
                Diag:= Diag xor $40;
                SetupEquipment;
                oldx:=wherex; oldy:= wherey;
                ShowStat;
                window(2,3,79,23);
                gotoxy(oldx,oldy);
              end;
          #66:begin {File}
                Diag:= Diag xor $10;
                SetupEquipment;
                oldx:=wherex; oldy:= wherey;
                ShowStat;
                window(2,3,79,23);
                gotoxy(oldx,oldy);
              end;
          #68:begin {bye}
                Display(' sign off from here ');
                OutStModem('BYE');
              end;
        end; {case}
      end {if esc}
      else
      begin
        Display(' A: ');
        txhdx;
        readln(inactchar);outmodem(soh);txhdx;
        outstmodem(inactchar);outmodem($13);
        rxhdx;
      end;
      end; {end keypressed}
      check_status(regs);
      if (regs.ax and $0100) = $0100 then
      begin
        inmodem(status);
        if status=soh then
           begin
             write(' B: ');
             repeat
               inmodem(status);
               if not (status= $13) then write(chr(status))
             until status = $13;
             writeln;
           end;
      end;
   until  (ch=#68)
end;

procedure des(option:integer; plaintext,key:block; var ciphertext:block);
type
  Ordering=            array[1..64] of 1..64;
  Rectangular=         array[1..8,1..64] of 0..15;
  Line=                array[1..16] of 1..2;
var
  x:                   block;
  k:                   integer;
const
  InitialTr:Ordering= (58,50,42,34,26,18,10,02,60,52,44,36,
                       28,20,12,04,62,54,46,38,30,22,14,06,
                       64,56,48,40,32,24,16,08,57,49,41,33,
                       25,17,09,01,59,51,43,35,27,19,11,03,
                       61,53,45,37,29,21,13,05,63,55,47,39,
                       31,23,15,07);
  FinalTr:Ordering=   (40,08,48,16,56,24,64,32,39,07,47,15,
                       55,23,63,31,38,06,46,14,54,22,62,30,
                       37,05,45,13,53,21,61,29,36,04,44,12,
                       52,20,60,28,35,03,43,11,51,19,59,27,
                       34,02,42,10,50,18,58,26,33,01,41,09,
                       49,17,57,25);
  Swap:Ordering=      (33,34,35,36,37,38,39,40,41,42,43,44,
                       45,46,47,48,49,50,51,52,53,54,55,56,
                       57,58,59,60,61,62,63,64,01,02,03,04,
                       05,06,07,08,09,10,11,12,13,14,15,16,
                       17,18,19,20,21,22,23,24,25,26,27,28,
                       29,30,31,32);
  KeyTr1:Ordering=    (57,49,41,33,25,17,09,01,58,50,42,34,
                       26,18,10,02,59,51,43,35,27,19,11,03,
```

```
                    60,52,44,36,63,55,47,39,31,23,15,07,
                    62,54,46,38,30,22,14,06,61,53,45,37,
                    29,21,13,05,28,20,12,04,01,02,03,04,
                    05,06,07,08);
KeyTr2:Ordering=   (14,17,11,24,01,05,03,28,15,06,21,10,
                    23,19,12,04,26,08,16,07,27,20,13,02,
                    41,52,31,37,47,55,30,40,51,45,33,48,
                    44,49,39,56,34,53,46,42,50,36,29,32,
                    01,02,03,04,05,06,07,08,09,10,11,12,
                    13,14,15,16);
Etr:Ordering=      (32,01,02,03,04,05,04,05,06,07,08,09,
                    08,09,10,11,12,13,12,13,14,15,16,17,
                    16,17,18,19,20,21,20,21,22,23,24,25,
                    24,25,26,27,28,29,28,29,30,31,32,01,
                    01,02,03,04,05,06,07,08,09,10,11,12,
                    13,14,15,16);
Ptr:Ordering=      (16,07,20,21,29,12,28,17,01,15,23,26,
                    05,18,31,10,02,08,24,14,32,27,03,09,
                    19,13,30,06,22,11,04,25,01,02,03,04,
                    05,06,07,08,09,10,11,12,13,14,15,16,
                    17,18,19,20,21,22,23,24,25,26,27,28,
                    29,30,31,32);
S:Rectangular=     ((14,04,13,01,02,15,11,08,03,10,06,12,
                    05,09,00,07,00,15,07,04,14,02,13,01,
                    10,06,12,11,09,05,03,08,04,01,14,08,
                    13,06,02,11,15,12,09,07,03,10,05,00,
                    15,12,08,02,04,09,01,07,05,11,03,14,
                    10,00,06,13),
                   (15,01,08,14,06,11,03,04,09,07,02,13,
                    12,00,05,10,03,13,04,07,15,02,08,14,
                    12,00,01,10,06,09,11,05,00,14,07,11,
                    10,04,13,01,05,08,12,06,09,03,02,15,
                    13,08,10,01,03,15,04,02,11,06,07,12,
                    00,05,14,09),
                   (10,00,09,14,06,03,15,05,01,13,12,07,
                    11,04,02,08,13,07,00,09,03,04,06,10,
                    02,08,05,14,12,11,15,01,13,06,04,09,
                    08,15,03,00,11,01,02,12,05,10,14,07,
                    01,10,13,00,06,09,08,07,04,15,14,03,
                    11,05,02,12),
                   (07,13,14,03,00,06,09,10,01,02,08,05,
                    11,12,04,05,13,08,11,05,06,15,00,03,
                    04,07,02,12,01,10,14,09,10,06,09,00,
                    12,11,07,13,15,01,03,14,05,02,08,04,
                    03,15,00,06,10,01,13,08,09,04,05,11,
                    12,07,02,14),
                   (02,12,04,01,07,10,11,06,08,05,03,15,
                    13,00,14,09,14,11,02,12,04,07,13,01,
                    05,00,15,10,03,09,08,06,04,02,01,11,
                    10,13,07,08,15,09,12,05,06,03,00,14,
                    11,08,12,07,01,14,02,13,06,15,00,09,
                    10,04,05,03),
                   (12,01,10,15,09,02,06,08,00,13,03,04,
                    14,07,05,11,10,15,04,02,07,12,09,05,
                    06,01,13,14,00,11,03,08,09,14,15,05,
                    02,08,12,03,07,00,04,10,01,13,11,06,
                    04,03,02,12,09,05,15,10,11,14,01,07,
                    06,00,08,13),
                   (04,11,02,14,15,00,08,13,03,12,09,07,
                    05,10,06,01,13,00,11,07,04,09,01,10,
                    14,03,05,12,02,15,08,06,01,04,11,03,
                    12,03,07,14,10,15,06,08,00,05,09,02,
                    06,11,13,08,01,04,10,07,09,05,00,15,
                    14,03,02,12),
                   (13,02,08,04,06,15,11,01,10,09,03,14,
                    05,00,12,07,01,15,13,08,10,03,07,04,
                    12,05,06,11,00,14,09,02,07,11,04,01,
                    09,12,14,02,00,06,10,12,15,03,05,08,
                    02,01,14,07,04,10,08,13,15,12,09,00,
                    03,05,06,11));
rotls:Line=         (1,1,2,2,2,2,2,2,1,2,2,2,2,2,2,1);
rotrs:Line=         (1,1,2,2,2,2,2,2,1,2,2,2,2,2,2,1);

procedure transpose( var data:block; t:ordering; n:integer);
var
  x:block;
  i:1..64;
begin
  x:=data;
```

```
  for i:=1 to n do
    data[i]:=x[t[i]];
end;

procedure rotatel(var key:block);
var
  i:1..55;
  x:block;
begin
  x:=key;
  for i:=1 to 55 do
    x[i]:=x[i+1];
  x[28]:=key[1]; x[56]:=key[29];
  key:=x;
end;

procedure rotater(var key:block);
var
  i,r:1..55;
  x:block;
begin
  x:=key;
  for i:=1 to 55 do
    x[57-i]:=x[56-i];
  x[1]:=key[28]; x[29]:=key[56];
  key:=x;
end;

procedure f(i:integer; var key,a,x:block);
var
  e,ikey,y: block;
  r: 0..64;
  k: 1..8;
  j: 1..48;
begin
  e:=a;
  transpose(e,etr,48);
  for j:=1 to rotls[i] do rotatel(key);
  ikey:=key;
  transpose(ikey,keyTr2,48);
  for j:=1 to 48 do if e[j]+ikey[j]=1 then y[j]:=1 else y[j]:=0;
  for k:=1 to 8 do begin
    r:=32*y[6*k-5]+16*y[6*k]+8*y[6*k-4]+4*y[6*k-3]+2*y[6*k-2]+y[6*k-1]+1;
    if odd(s[k,r] div 8) then x[4*k-3]:=1 else x[4*k-3]:=0;
    if odd(s[k,r] div 4) then x[4*k-2]:=1 else x[4*k-2]:=0;
    if odd(s[k,r] div 2) then x[4*k-1]:=1 else x[4*k-1]:=0;
    if odd(s[k,r])       then x[4*k]  :=1 else x[4*k]  :=0;
  end;
  transpose(x,ptr,32);
end;

procedure f2(i:integer; var key,a,x: block);
var
  e,ke,ikey,y: block;
  r: 0..64;
  k: 1..8;
  j,l: 1..48;
begin
  e:=a;
  transpose(e,etr,48);if i=1 then key:=key else
  for j:=1 to rotrs[i] do rotater(key);
  ikey:=key;
  transpose(ikey,keyTr2,48);
  for j:=1 to 48 do if e[j]+ikey[j]=1 then y[j]:=1 else y[j]:=0;
  for k:=1 to 8 do begin
    r:=32*y[6*k-5]+16*y[6*k]+8*y[6*k-4]+4*y[6*k-3]+2*y[6*k-2]+y[6*k-1]+1;
    if odd(s[k,r] div 8) then x[4*k-3]:=1 else x[4*k-3]:=0;
    if odd(s[k,r] div 4) then x[4*k-2]:=1 else x[4*k-2]:=0;
    if odd(s[k,r] div 2) then x[4*k-1]:=1 else x[4*k-1]:=0;
    if odd(s[k,r])       then x[4*k]  :=1 else x[4*k]  :=0;
  end;
  transpose(x,ptr,32);
end;

procedure des1(plaintext,key: block; var ciphertext: block);
var
  i: 1..16;
  j: 1..32;
  k: 1..64;
```

```
    a,b,x: block;
  begin
    a:=plaintext;
    transpose(a,InitialTr,64);
    transpose(key,keyTr1,56);
    for i:=1 to 16 do begin
      b:=a;
      for j:=1 to 32 do a[j]:= b[j+32];
      f(i,key,a,x);
      for j:=1 to 32 do if b[j]+x[j]=1 then
        a[j+32]:=1 else a[j+32]:=0;
    end;
    transpose(a,swap,64);
    transpose(a,FinalTr,64);
    ciphertext:=a;
    ciphertext:=ciphertext;
  end;

  procedure des2(plaintext,key: block; var ciphertext: block);
  var
    i,m: 1..16;
    j: 1..32;
    k: 1..64;
    a,b,x: block;
  begin
    a:=plaintext;
    transpose(a,InitialTr,64);
    transpose(key,keyTr1,56);
    for i:=1 to 16 do begin
      b:=a;
      for j:=1 to 32 do a[j]:= b[j+32];
      f2(i,key,a,x);
      for j:=1 to 32 do if b[j]+x[j]=1 then
        a[j+32]:=1 else a[j+32]:=0;
    end;
    transpose(a,swap,64);
    transpose(a,FinalTr,64);
    ciphertext:=a;
    ciphertext:=ciphertext;
  end;

begin {des}
  case option of
    1:des1(plaintext,key,ciphertext);
    2:des2(plaintext,key,ciphertext);
  end;
end;

procedure Convert(option:integer; PlainChar,KeyChar: string8; var CipherChar: string8);
  procedure Conv(PlainChar,KeyChar: string8; var CipherChar: string8);
  begin
    for i:=1 to 8 do begin
      valueP:= ord(PlainChar[i]);
      valueK:= ord(KeyChar[i]);
      if i>Length(PlainChar) then valueP:=0;
      if i>Length(KeyChar)   then valueK:=0;
      for j:=1 to 8 do begin
        PlainText[8*(i-1)+9-j]:= valueP mod 2;
        DoorKey[8*(i-1)+9-j]:= valueK mod 2;
        valueP:= valueP div 2;
        valueK:= valueK div 2;
      end;
    end;
    key:= doorkey;
    des(1,plaintext,key,ciphertext);
    for i:=1 to 8 do begin
      valueP:= 0;
      for j:=1 to 8 do
        valueP:=valueP+ciphertext[8*(i-1)+9-j]*btod[j];
      PlainChar[i]:= chr(valueP);
    end;
    for i:=1 to 8 do
      CipherChar[i]:= PlainChar[i];
    CipherChar[0]:=chr(8);
  end;

  procedure DConv(PlainChar,KeyChar: string8; var CipherChar:string8);
  begin
    for i:=1 to 8 do begin
```

```
        valueP:= ord(PlainChar[i]);
        valueK:= ord(KeyChar[i]);
        if i>Length(PlainChar) then valueP:= 0;
        if i>Length(KeyChar) then valueK:= 0;
        for j:=1 to 8 do begin
          PlainText[8*(i-1)+9-j]:= valueP mod 2;
          DoorKey[8*(i-1)+9-j]:= valueK mod 2;
          valueP:= valueP div 2;
          valueK:= valueK div 2;
        end;
      end;
      key:= doorkey;
      des(2,plaintext,key,ciphertext);
      for i:=1 to 8 do begin
        valueP:= 0;
        for j:=1 to 8 do
          valueP:=valueP+ciphertext[8*(i-1)+9-j]*btod[j];
        PlainChar[i]:= chr(valueP);
      end;
      for i:=1 to 8 do begin
        CipherChar[i]:=PlainChar[i];
      end;
    end;

  begin {convert}
    case option of
      1:Conv(PlainChar,KeyChar,CipherChar);
      2:DConv(PlainChar,KeyChar,CipherChar);
    end;
  end;  {convert}

{$I f11.pas}
{ ********************************
  main subroutine:
  subcroutine called directly from main program
  ******************************** }

  var  kc,kkk:       byte;

  procedure GetMsg;
  var  RecvFile:     anystring;
       Recvbuf:      array[1..1280] of byte;
       Usefile:      file;
  begin
    clrscr;
    RecvFile:=' ';
    border(1,1,80,8);
       writeln;
       center(' There are some messages receipted while you are not around ');
       center(' Please waiting for Unattended Message ...');
       writeln;
       write(' Unattendded File  :    ');
       for i:=1 to 11 do begin inmodem(status);write(chr(status));end;
       writeln;
       repeat
           inmodem(status);write(chr(status));
       until status =$04;
       wait;
       {repeat
        inmodem(status);
        if Status = DLE then
        begin
        repeat
         inmodem(status);Recvfile:=Recvfile+chr(status);
        until status = nul;
        RecvFile:=copy(RecvFile,1,length(RecvFile)-1);
        write('     ',RecvFile,'     ');
        assign(Usefile,RecvFile);rewrite(Usefile);
        repeat
           for i:= 1 to 128 do
           begin
               inmodem(status);recvbuf[i]:=status;
               if (status = EOT) or (status=ETX) then i:=128;
           end;
           blockwrite(Usefile,recvbuf,1);
        until (status = EOT) or (status = ETX);
        close(Usefile);
        end;
       until status = DLE;
```

```
  end }
end; {GetMsg}

procedure ReceiverMode;
var c: integer;
begin
   border(1,1,80,24);
   modemsetup;
   c:= 2;
   filetranf(c);
   wait;
end; {receive}

procedure HelpFile;
procedure help;
const max_line=10;
    max_page=10;
    maxC=70;
    comcol   =23;
    Fil_help ='helpf.pas';
type
   page = array[1..max_line] of string[maxC];
   block= array[1..max_page] of page;
Var  helptxt : block;
   (* helptxt[page_no,line_no] *)
   help_file :text;

procedure Init;
var
   filvar:text;
   i,j:integer;
begin
   assign(filvar,fil_help);
   reset(filvar);
   for i:=1 to max_page do
   begin
        for j:=1 to max_line do
        begin
             readln(filvar,helptxt[i,j]);
        end;
   end;
   close(filvar);
end;

Procedure show(page_no:integer);
var i,j:integer;
  aa:string[maxC];
begin
   lowvideo;
   write('   COmmand                              DEscription        ');
   normvideo;
   writeln('Page ',page_no:2);
   for i:=1 to max_line do
   begin
        aa:=helptxt[page_no,i];
        if aa<>'' then
        begin
             for j:=1 to comcol-1 do
             begin
                  if ord(aa[j]) in [65..91]  then
                  begin
                     lowvideo;
                     write(aa[j]);
                     normvideo;
                  end
                  else write(aa[j]);
             end;
             for j:=comcol to length(aa) do write(aa[j]);
        writeln;
        end
        else writeln;
   end;
end;
var
 i,j:integer;
 exit :boolean;
 c:char;

BEGIN
```

```
    init;
    exit:=false;
    i:=1;
    repeat
          clrscr;
          show(i);
          read(kbd,c);
          if c=#27 then
          begin
               read(kbd,c);
               j:=ord(c);
               case j of
                    80,81,75 : i:=i+1;
                    72,73,77 : i:=i-1;
                    else       exit:=true;
               end;
               if i>max_page then i:=max_page;
               if i<1 then i:=1;
          end
          else exit:=true;
    until  exit;
END;

begin
     border(2,10,79,23);clrscr;
     help;
end;

procedure SendFile;
var c: integer;
begin
     border(1,1,80,24);
     modemsetup;
     clrscr;
     c:= 1;
     filetranf(c);
end;

procedure ConInteractive;
begin
  StampDatim;
  Activity('Continue Interactive');
  ActNote;
  Ch:=^m;
  Diag:= $A0;
  SetupEquipment;
  InteractiveScreen;
end;

procedure Remote;
var     c:           integer;
        comm:        anystring;
begin
  clrscr;
  border(1,1,80,24);
  center('Remote Service');
  center('-----------------');
  writeln;writeln;write(' Enter Command :');readln(comm);
  txhdx;
  outstmodem('R');
  if upcase(copy(comm,1,1))='D' then
  begin
     txhdx;
     outstmodem('D');rxhdx;
     repeat
       inmodem(status);write(chr(status));
     until status=18;
  end else
  if upcase(copy(comm,1,1))='R' then begin
        txhdx;
        outstmodem('R');
        border(1,1,80,24);
        clrscr;
        c:=3;
        filetranf(c);
      end;
end;

procedure Recvremote;
```

```
var       c:    integer;
procedure Dir(var mask: char);
type
  Anystring= string[10];
  RegPack= record
             AX,BX,CX,DX,BP,SI,DI,DS,ES,Flags: integer;
           end;
var
  Regs: RegPack;
  St: Anystring;
  p: ^char;
  i,j,h,l: integer;
  Buff: array[0..50] of byte;

begin
  for i:= 1 to 50 do buff[i]:=0;
  with Regs do begin
    Ds:= Seg(Buff[0]); h:= Ds;
    Dx:= Ofs(Buff[0]); l:= Dx;
    Ax:= $1a00;
    MsDos(Regs);
    outstmodem(' Disk directory: ');outstmodem(mask);
    outmodem(10);outmodem(13);outmodem(32);
    St:= mask + ':\*.*' + chr(0);
    Ds:= Seg(St[1]);
    Dx:= Ofs(St[1]);
    Cx:= 32;
    Ax:= $4e00;
    MsDos(Regs);
    repeat
      for j:=1 to 5 do begin
          for i:=30 to 43 do begin
          p:= Ptr(h,l+i);
          outstmodem(p^);
          end;
          fillchar(Mem[h:l+30],14,32);
          Ax:= $4f00;
          MsDos(Regs);
          end;
          outmodem(10);outmodem(13);outmodem(32);
     until Ax = 18;
     end;
     outmodem(18);
end;

begin
  clrscr;
  Center(' Receive remote data service ');
  Center('------------------------------');
  inmodem(status);
  if status=ord('D') then begin
      txhdx;
      mask:='c';
      dir(mask);rxhdx;
      wait;
     end else
     begin
       border(1,1,80,24);
       clrscr;
       c:=4;
       filetranf(c);
       wait;
     end;
end;

procedure Convertstr(var j,kk:   byte);
var       i:      integer;
          k:          char;
begin
    kk:=0;
    for i:=1 to 2 do
    begin
      k:=copy(st,(j*2)+i,1);
      if k in ['0'..'w'] then
      kk:=kk + ((2-i)*16*(ord(k)-48)+(ord(k)-48)*(i-1));
    end;
end;

procedure Email;
```

```
var     k:                              char;
        phlength,pk,
        exe,exes:                       integer;
        j,kk,n1:                        byte;
        fsize,fstr,buff1,buff2,
        EmailFile,EmailFiles:           anystring;


begin
  Clrscr;
  border(1,1,80,24);
  Center('ELECTRONIC MAIL MODE');
  center('-----------------------');
  EmailFile:=' ';
  write(' Enter filename (Based Electronic Mail) : ');
  readln(EmailFile);
  if exist(Emailfile) then
  begin
   assign(Usefile,EmailFile);
   reset(Usefile);
   n:=filesize(Usefile);
   exe:=pos('.',EmailFile);
   if exe > 9 then exes:= 9 else exes:= exe;
   EmailFiles:=copy(EmailFile,1,exes-1)+'.'+copy(EmailFile,exe+1,length(EmailFile));
   GetPhone;
   GotNumber(PhoneRec.Num);
   clrscr;
   phlength:=length(st);
   if odd(phlength) then st:=st+'?' else st:=st + 'WW';
   pk:=length(st) div 2;
   getconfig;
   with ProtoRec do
        Temp:=32*BaudRate +8*Parity +4*StopBit +WordLength;
   clrscr;
   writeln(' Enter send time(hh:mm:ss):');
   readln(Buff1);
   writeln(' Enter send date(mm-dd-yy):');
   readln(Buff2);
   StampDatim;
   Activity('Assign destination for Electronic Mail');
   Activity('  Send message to '+PhoneRec.Name+' at '+Buff1);
   ActNote;
   Delete(Buff1,3,1);
   Delete(Buff2,3,1);
   Delete(Buff2,5,1);
   Attention;
   OutStModem('R');outmodem($13);
   for j:= 0 to pk-1 do begin
     ConvertStr(j,kk);outmodem(kk);
   end;
   for j:=pk-1 to 6 do outmodem($0);
   outmodem(Temp);
   outstmodem(EmailFiles);
   for i:= length(EmailFiles)+1 to 11 do outmodem($20);
   { protocol }
   for i:=1 to 4 do begin k:=copy(buff1,5-i,1); outmodem(ord(k)-48); end;
   for i:=1 to 4 do begin k:=copy(buff2,5-i,1); outmodem(ord(k)-48); end;
   { buff1,buff2 = time (8 byte) ,st = number }
   n1:=n mod 256;outmodem(n1-1);
   n1:=n div 256;outmodem(n1);
   repeat
     read(Usefile,n1);outmodem(n1);
   until n1 = 26;
  end
  else begin clrscr; writeln(' File not exit ');end;
  end; {Email}


procedure SetupProtocol;
begin
  clrscr;
  Center('SET UP PROTOCOL');
  write(Line);
  writeln; write('Do you want to recall old configuration ?(y/n) -');
  read(Kbd,Ch); writeln;
  if Upcase(Ch)='Y' then begin
    FileName:='';
    write('Enter config. filename -');
    readln(FileName);
```

```
    FileName:=Copy(FileName,1,8)+'.CFG';
    if FileName='.CFG' then FileName:= Protocol;
    assign(ProtoFile,FileName);
    reset(ProtoFile); read(ProtoFile,ProtoRec); Close(ProtoFile);
  end else begin
    writeln; writeln;
    Center('Modify Configurations');
    Center('-----------------------');
    write('Name  :'); readln(FileName);
      FileName:=Copy(FileName,1,8)+'.CFG';
    border(1,14,60,21);
    writeln(' Baud rate   :');
    writeln('  0= 110 bps          4= 1200 bps');
    writeln('  1= 150 bps          5= 2400 bps');
    writeln('  2= 300 bps          6= 4800 bps');
    writeln('  3= 600 bps          7= 9600 bps');
    write(' Enter  :'); readln(Temp);
      ProtoRec.BaudRate:= Temp;
    gotoxy(1,14);
    writeln(' Parity  :');
    writeln('  0= No Parity                    ');ClrEol;ClrEol;
    writeln('  1= Odd Parity');ClrEol;
    writeln('  3= Even Parity');ClrEol;writeln;
    write(' Enter  :'); readln(Temp);
      ProtoRec.Parity:= Temp;
    gotoxy(1,14);
    writeln(' StopBit  :');
    writeln('  0= 1 Stop Bit                  ');ClrEol;ClrEol;
    writeln('  1= 2 Stop Bit');ClrEol;writeln;writeln;
    write(' Enter  :'); readln(Temp);
      ProtoRec.StopBit:= Temp;
    gotoxy(1,14);
    writeln(' Word Length :');
    writeln('  2= 7 Bit Word Length           ');ClrEol;ClrEol;
    writeln('  3= 8 Bit Word Length');ClrEol;writeln;writeln;
    write(' Enter   :'); readln(Temp);
      ProtoRec.WordLength:= Temp;
    writeln;writeln;writeln;writeln;
    assign(ProtoFile,FileName);
    rewrite(ProtoFile);
    write(ProtoFile,ProtoRec);
    close(ProtoFile);
  end;
  StampDatim;
  Activity('Setup protocol for '+FileName);
  ActNote;
  Setup(ProtoRec);
end;

procedure Encrypt;
begin {encrypt}
  Clrscr;
  Center('Data Encryption');
  writeln(Line);
  write('Source file name:       ');readln(SourName);
  write('Destination file name: ');readln(DestName);
  Assign(Source,SourName); Assign(Dest,DestName);
  if not(Exist(SourName)) then begin
    writeln(^g,'ERROR: unable to open file');
    halt;
  end;
  writeln;write('(E) ncryption or (D) ecryption: ');
  read(Kbd,Command); Command:= Upcase(Command);
  reset(Source);
  rewrite(Dest);
  ClrScr;
  if Command='E' then begin
    writeln('                                Encrypt ',SourName,' --> ',DestName);
    StampDatim;
    Activity('Encrypt '+SourName+' --> '+DestName);
  end else begin
    writeln('                                Decrypt ',SourName,' --> ',DestName);
    StampDatim;
    Activity('Decrypt '+SourName+' --> '+DestName);
  end;
  writeln(Line);
  GotoXY(1,1); write('Keyword: ');
  GetKeyWord;
  ValueP:= FileSize(Source);
```

```
  gotoXY(70,1);
  writeln('Time: ',0.15*ValueP:5:0);
  window(1,3,80,25);
  while not Eof(Source) do begin
    for k:=1 to 8 do begin
      if not(Eof(Source)) then begin
        read(Source,Buff);
        PlainChar[k]:=Chr(Buff);
      end else
        PlainChar[k]:= Chr(0);
    end;
    for k:=1 to 8 do begin
      valueP:= ord(PlainChar[k]);
      valueK:= ord(KeyChar[k]);
      for j:=1 to 8 do begin
        PlainText[8*(k-1)+9-j]:= valueP mod 2;
        DoorKey[8*(k-1)+9-j]:= valueK mod 2;
        valueP:= valueP div 2;
        valueK:= valueK div 2;
      end;
    end;
    key:= doorkey;
    for k:=1 to 64 do ciphertext[k]:=0;
    if Command= 'E' then
      des(1,plaintext,key,ciphertext)
    else
      des(2,plaintext,key,ciphertext);
    for k:=1 to 8 do begin
      Buff:= 0;
      for j:=1 to 8 do
        Buff:=Buff+ciphertext[8*(k-1)+9-j]*btod[j];
      write(Dest,Buff);
      write(chr(Buff));
    end;
  end;
  Close(Source); Close(Dest);
  ActNote;
end; {encrypt}


procedure Dir(var mask: char);
type
  Anystring= string[10];
  RegPack= record
             AX,BX,CX,DX,BP,SI,DI,DS,ES,Flags: integer;
           end;
var
  Regs: RegPack;
  St: Anystring;
  p: ^char;
  i,j,h,l: integer;
  Buff: array[0..50] of byte;
begin
  ClrScr;
  for i:= 1 to 50 do buff[i]:=0;
  with Regs do begin
    Ds:= Seg(Buff[0]); h:= Ds;
    Dx:= Ofs(Buff[0]); l:= Dx;
    Ax:= $1a00;
    MsDos(Regs);
    writeln('Disk directory: ',mask);
    writeln;writeln;St:= mask + ':\*.*' + chr(0);
    Ds:= Seg(St[1]);
    Dx:= Ofs(St[1]);
    Cx:= 32;
    Ax:= $4e00;
    MsDos(Regs);
    repeat
      for j:=1 to 5 do begin
          for i:=30 to 43 do begin
          p:= Ptr(h,l+i);
          write(p^);
          end;
      fillchar(Mem[h:l+30],14,32);
      Ax:= $4f00;
      MsDos(Regs);
      end;writeln;
    until Ax = 18;
  end;
```

```
end;

procedure FileTypeF;
type Text=string[12];
     Page=array[1..10000] of integer;
     numa=array[1..15] of integer;
var  FileName:  Text;
     UseFile: file of byte;
     b: byte;
     i,j,k:integer;
     PageUp:Page;
     c:numa;
begin
  ClrScr;
  PageUD:=' ';
  writeln; write('enter filename you want to type -');
  readln(FileName);
  assign(UseFile,FileName);
  reset(UseFile);
  i:=1;k:=1;
repeat
       read(UseFile,b);
       PageUp[i]:=b;
       i:=i+1;
until Eof(UseFile);PageUp[i]:=26;
i:=1;ClrScr;
repeat
      write(chr(PageUp[i]));i:=i+1;
      if PageUp[i]=13 then k:=k+1;
until ((k=25) or (PageUp[i]=26));
j:=1;c[j]:=1;
repeat
        if KeyPressed then begin
           read(Kbd,PageUD);
           if ord(PageUD)=ESC then begin
                k:=1;
                read(Kbd,PageUD);
                case PageUD of
                 #73:begin
                      ClrScr;j:=j+1;
                      if PageUp[i]=26 then begin j:=j-1;i:=c[j];end
                        else c[j]:=i+1;
                        repeat
                          write(chr(PageUp[i]));i:=i+1;
                          if PageUp[i]=13 then k:=k+1;
                        until ((k=25) or (PageUp[i]=26));
                     end;
                 #81:begin
                      ClrScr;j:=j-1;
                      if j=0 then j:=1;
                      i:=c[j];
                      repeat
                        write(chr(PageUp[i]));i:=i+1;
                        if PageUp[i]=13 then k:=k+1;
                      until ((k=25) or (PageUp[i]=26));
                     end;
                end;
           end;
        end;{esc}
 until (PageUD=#79);
  close(UseFile);
 end;

{ ********************************
  main program:
  program start here...
  ******************************** }
  procedure getpassword;
  begin
     password:='';
     assign(passfile,'pass.pas');
     reset(passfile);
     read(passfile,passrec);
     if not(passrec.password='') then
        begin
          txhdx;
          outstmodem('p');
          rxhdx;
          for i:=1 to length(passrec.password) do
```

```
          begin
            inmodem(status);password:=password+chr(status);
          end;
          txhdx;
          if not(password = passrec.password ) then begin
               outmodem($1); halt;
             end else begin
               outmodem($13);outstmodem(passrec.greeting);
               outmodem($13);
          end;
  end;
  end;

  procedure verifypassword;
  begin
   inmodem(status);
   if status=ord('p') then begin
      border(5,5,40,22);
      clrscr;
      write('Enter your password (up to 12 chararcter):');
      readln(password);outstmodem(password);
      inmodem(status);
      if not(status=$13) then begin
           clrscr;
           write('Your password is wrong, Bye!');
         end else begin
           clrscr;
           writeln('Your password , O.K.');
           writeln('Wait for access');
           repeat
             inmodem(status);write(chr(status));
           until status=$13;
         end;
      end;
  end;

procedure  setbr;
begin
  port[1019]:=$40;
end;

procedure relbr;
begin
  port[1019]:=$00;
end;

var   r,brreg: integer;
begin {main}
  StampDatim;
  Activity('Start SuperMODEM system');
  Activity('');
  ClrScr;
  lowvideo;
    writeln('Software Communications Programs');
    writeln('version 1.0, (C) Copyright 1986.');
    writeln('Communications Laboratories, Electrical Engineering Department');
    write('Faculty of Engineering, Chulalongkorn University.');
    lowvideo;
    gotoxy(1,7);
    writeln('These Programs Conform to Standardization, Recommendations on CCITT & ANSI ');
    writeln('Standards, which are as follows.');
    writeln(':10,'1. Interface Between Data Terminal Equipment and Automatic ');
    writeln(':10,'   Calling /Answering Equipment for Data Communications,');
    normvideo;writeln(':10,'   Use ANSI Stds RS-366-A  Type VII.');
    lowvideo;
    writeln(':10,'2. Interfacing Automatic Calling And/Or Answering Equipment ');
    writeln(':10,'   On The General Switched Telephone Network,');
    normvideo;writeln(':10,'   Use Recommendations CCITT V.25.');
    lowvideo;
    writeln(':10,'3. Data Communications, 2400/1200 bits per sec Modem Standadized ');
    writeln(':10,'   for Use in the General Switched Telephone Network,');
    normvideo;writeln(':10,'   Use Recommendations CCITT V.26 bis.');
    lowvideo;
    writeln(':10,'4. Data Security for Electronic Mails or Computer-Base Mail ');
    writeln(':10,'   System:CBMS, Data Encryption Algorithm,');
    normvideo;writeln(':10,'   Use Data Encryption Standard  US Goverment:NBS77');
    writeln(':10,'   ( ANSI Stds X3.92 ).');
    modemsetup;
    setdtr;setrts;
```

```
    delay(300);
    st:=time;
    for i:=1 to 6 do begin
       mask:=copy(st,7-i,1);
       if mask=chr(32) then begin
          outmodem($0);write('0');
          end  else  begin
          outmodem(ord(mask)-48);write(mask);
          end;
    end;
    outmodem($0);writeln;
    st:=date;
    for i:=1 to 4 do begin
       mask:=copy(st,5-i,1);
       if mask=chr(32) then begin
          outmodem($0);write('0');
          end else begin
          outmodem(ord(mask)-48);write(mask);
          end;
    end;outmodem($0);outmodem($0);
    {chk for time record}
    status:=0;
    outmodem($53);
    repeat
      inmodem(status);
    until status = $CC;
    inmodem(status);
    if status = $59 then getmsg
    else
    begin
       clrscr;
       center('No Unattended Message');
       wait;
    end;
    repeat
{    Attention;
    OutStModem('S1'); }
    ClrScr;
    border(1,1,80,24);
    lowvideo;
    writeln;
    center('Main Menu');center('---------');writeln;
    writeln('':5,'Do you want to--');writeln;r:=1;
    CenterF('Help                              ',r);
    CenterF('Send file                         ',r);
    CenterF('Autodial Mode                     ',r);r:=1;
    CenterF('Unattended Mail                   ',r);
    CenterF('Continue Interactive Mode         ',r);
    CenterF('Remote Service                    ',r);
    CenterF('Electronic Mail                   ',r);r:=8;
    CenterF('Set up Protocol                   ',r);r:=7;
    CenterF('Data ENcryption                   ',r);r:=1;
    CenterF('Type File                         ',r);
    CenterF('Disk Directory                    ',r);r:=5;
    CenterF('PassWord and Greeting              ',r);r:=1;
    CenterF('Reset                             ',r);r:=1;
    CenterF('Quit                              ',r);
    writeln;write('':5,'Enter Your Choice-- ');
    normvideo;window(1,1,80,25);
    rxhdx;
    repeat
        check_status(regs);
        brreg:= regs.ax and $1100;
        if brreg and $0100 = $0100 then
           begin
              inmodem(status);
              case status of
                 enq:     Receivermode;
                 $0d:     Coninteractive;
                 $52:     Recvremote;
                 $46:     begin
                          clrscr;
                          border(1,1,80,5);
                          writeln;
                          writeln(' Electronic Mail File Transfer System ');
                          write(' File Name is  :  ');
                          for i:=1 to 11 do begin
                             inmodem(status);
                             write(chr(status));
```

```
                    end;
                    writeln;
                  repeat
                    inmodem(status);
                    write(chr(status));
                  until status = $04;
                  wait;
                  outstmodem('ATH');outmodem(1);outmodem($13);
                  end;
          $55:    begin
                   clrscr;
                   border(5,5,80,10);
                   writeln(' Now, The Destination is Unattended Terminal.');
                   writeln(' Pls, Use Unattended Key for transfer file only');
                  end;
        end;
      end;
    if brreg and $1000 = $1000 then
      begin
        outstmodem('ATH');outmodem(1);outmodem($13);
        { set answer reg again}
      end;
  until KeyPressed;
  read(Kbd,Choice);
{  Attention;
  OutStModem('S3');
 } case Choice of
      'h','H':helpfile;
      's','S':begin
                Sendfile;
              end;
      'a','A':begin
                attention;outstmodem('DT');
                GetPhone;
                GotNumber(PhoneRec.Num);
                phlength:=length(st);
                if odd(phlength) then st:=st+'?' else st:=st + 'WW';
                pk:=length(st) div 2;
                for buff:= 0 to pk-1 do begin
                    ConvertStr(buff,kkbuff);outmodem(kkbuff);
                end;
                outmodem($13);clrrts;
                {writeln('':14,'wait a minute');
                Count:=1;
                repeat
                Flag:= Dial(TimeOut)
                until Flag or (Length(PhoneRec.Num)=0);}
              end;
      'c','C':begin
                txhdx;
                outmodem($0d);
                rxhdx;
                Coninteractive;
              end;
      'r','R':Remote;
      'e','E':Email;
      'p','P':SetupProtocol;
      'n','N':Encrypt;
      'u','U':begin
                clrscr;
                Center('Unattended File Transfer');
                writeln;writeln;
                writeln(' Pls Enter Filename which you Want to Transfer :  ');
                write(' ');readln(filename);
                assign(Usefile,filename);
                reset(Usefile);
                setrts;delay(300);
                outmodem($48);delay(10);
                for i:=1 to 9 do outmodem($0);
                outstmodem(filename);for i:=1 to (19-length(filename)) do outmodem($0);
                i:=filesize(Usefile);
                kc:=i mod 256;outmodem(kc-1);
                kc:=i div 256;outmodem(kc);
                repeat
                   read(usefile,kkk);outmodem(kkk);
                until kkk=26;
                outstmodem('ATH');outmodem(1);outmodem($13);
              end;
      'd','D':begin
```

```
                writeln;lowvideo;
                gotoXY(7,22);
                write('Disk Mask :');normvideo;
                read(kbd,mask);
                Dir(mask);
               end;
        't','T': FileTypeF;
        'z','Z': begin
                  outstmodem('ATZ');outmodem($13);
                 end;
        'w','W':begin
                 border(1,1,80,24);
                 clrscr;
                 center('Password and greeting');
                 center('-----------------------');
                 assign(Passfile,'Pass.pas');
                 reset(passfile);
                 writeln;writeln;
                 write(' Enter your password (up to 12 chars):');
                 readln(password);
                 write(' Enter your greeting word (up to 30 chars):');
                 readln(greeting);
                 passrec.password:=password;
                 passrec.greeting:=greeting;
                 write(Passfile,passrec);
                 close(passfile);
                end;
        else  r:=10;
    end; { case }
    if r=10 then r:=3 else Wait;
  until (Choice in ['q','Q']);
  delay(208);
  setbr;
  delay(208);
  relbr;
  modemsetup;
  outstmodem('ATH');outmodem(1);outmodem($13);
  clrdtr;clrrts;
  StampDatim;
  Activity('Stop SuperMODEM system');
  Activity('');
end.
```

```
ERR  LINE  ADDR  OBJ

       1                                    ASEG
       2                                    ORG     0000H
       3                                    ; MAIN PROGRAM INITIALIZE Z80-CTC1, Z80-CTC2,8250 AND 8255
       4                                    ; STATEDIAGRAM 1 SET MODE1,OFFHOOK,STATEDIAG 2 CHK DTR OF DTE
       5                                    ; STATEDIAGRAM 1
       6  0000  06 FF                       LD      B,PWDLY         ;DELAY POWER UP
       7  0002  10 FE                       DJNZ    $
       8  0004  31 5C 17                    LD      SP,STKPT
       9  0007  18 1D                       JR      INIT
      10                                    ORG     10H
      11  0010  2D 07          INTRTN:     DEFW    INTRTNS
      12  0012  2E 07                       DEFW    INTRTN1
      13                                    ORG     20H
      14  0020                  INTVEC:     DEFS    4
      15  0024  3F 07                       DEFW    TIMEINT
      16  0026  21 FF 17        INIT:       LD      HL,17FFH
      17  0029  36 00           CLRMEM:     LD      (HL),00H        ;CLEAR WORKING MEMORY
      18  002B  2B                          DEC     HL
      19  002C  7C                          LD      A,H
      20  002D  FE 0F                       CP      00FH            ;CHK CLR WORKING MEMORY MAY NOT BE
      21  002F  20 F8                       JR      NZ,CLRMEM       ;CLR
      22  0031  21 2E 16                    LD      HL,LASTADD      ;BUFF ADD. START
      23  0034  36 00                       LD      (HL),00H
      24  0036  23                          INC     HL
      25  0037  36 30                       LD      (HL),30H
      26  0039  21 2C 16                    LD      HL,STRADD
      27  003C  36 00                       LD      (HL),00H
      28  003E  23                          INC     HL
      29  003F  36 90                       LD      (HL),90H
      30  0041  3E 91                       LD      A,091H          ;INITIAL 8255 FOR TEST PULSE OR TONE
      31  0043  D3 D3                       OUT     (CTL1),A        ;PORT A,C(LOWER)=INP PORT B,UPPER= OUT
      32  0045  DB D0                       IN      A,(PORTA)       ;TEST PULSE OR DTMF DIAL
      33  0047  E6 40                       AND     01000000B
      34  0049  32 30 16                    LD      (DIALTYPE),A
      35  004C  3E B1                       LD      A,B1H           ;INITIAL 8255 PORT A,B=OUTPUT
      36  004E  D3 D3                       OUT     (CTL1),A        ;PORT C(UPPER) = OUTPUT LOWER= INP
      37  0050  3E B0                       LD      A,0B0H          ;SET ANS TONE DISABLE
      38  0052  32 28 16                    LD      (SPORTA),A      ;LD STATUS PORT A
      39  0055  D3 D0                       OUT     (PORTA),A
      40  0057  3E C0                       LD      A,0C0H          ;SET MODEM,TERMINAL = ACTIVE
      41  0059  32 22 16                    LD      (SPORTC),A      ;LD STATUS PORT C
      42  005C  D3 D2                       OUT     (PORTC),A       ;MODE 1,MODE 01=0E0H
      43  005E  CD 35 03                    CALL    NORHOOK         ;HOOK FOR TEST RI ONLY NOTOFFHOOK
      44  0061  21 10 00                    LD      HL,INTRTN       ;SET CTC0 INTERRUPT VECTOR
      45  0064  7C                          LD      A,H
      46  0065  ED 47                       LD      I,A
      47  0067  ED 5E                       IM      2
      48  0069  7D                          LD      A,L
      49  006A  D3 40                       OUT     (CTC1CH0),A
      50  006C  3E 0F                       LD      A,0FH           ;SET CTC CH0 TIMER, EDGE TRIG
```

```
ERR  LINE  ADDR  OBJ

      51  006E  D3 40                    OUT     (CTC1CH0),A     ;INTERRUPT DISABLE
      52  0070  3E 18                    LD      A,018H          ;TIMER DIVIDED BY 256+12
      53  0072  D3 40                    OUT     (CTC1CH0),A
      54  0074  3E DF                    LD      A,0DFH          ;SET CTC CH1 COUNTER, EDGE TRIG
      55  0076  D3 41                    OUT     (CTC1CH1),A     ;FROM IC CH0 INTR ENABLE
      56  0078  3E 09                    LD      A,09H           ;COUNT FOR 13 PULSE
      57  007A  D3 41                    OUT     (CTC1CH1),A
      58  007C  21 2B 00                 LD      HL,INTVEC       ;SET CTC CH2
      59  007F  7C                       LD      A,H
      60  0080  ED 47                    LD      I,A
      61  0082  ED 5E                    IM      2
      62  0084  7D                       LD      A,L
      63  0085  D3 80                    OUT     (CTC2CH0),A
      64  0087  3E 27                    LD      A,00100111B     ;SET CTC TIMER
      65  0089  D3 80                    OUT     (CTC2CH0),A
      66  008B  3E 2F                    LD      A,02FH
      67  008D  D3 80                    OUT     (CTC2CH0),A
      68  008F  3E 5F                    LD      A,01011111B     ;SET COUNTER
      69  0091  D3 81                    OUT     (CTC2CH1),A
      70  0093  3E FF                    LD      A,0FFH
      71  0095  D3 81                    OUT     (CTC2CH1),A
      72  0097  3E DF                    LD      A,11011111B     ;SET COUNTER
      73  0099  D3 82                    OUT     (CTC2CH2),A
      74  009B  3E 47                    LD      A,047H
      75  009D  D3 82                    OUT     (CTC2CH2),A
      76  009F  FB                       EI
      77  00A0  3E 9B                    LD      A,09BH          ;SET 8250 FOR 1200BPS, 8 BIT
      78  00A2  CD 25 04                 CALL    PROTOCOL        ;1 STOP BIT EVEN PARITY
      79                                 FILE    STATEDIA
      79                                 ; STATEDIAGRAM 2
      79  00A5  79                       LD      A,C             ;CHK TERMINAL TURN OFF
      79  00A6  E6 20                    AND     20H
      79  00A8  FE 20                    CP      20H
      79  00AA  20 2E                    JR      NZ,DTENOTACTIVE
      79                                 ; STATEDIAGRAM 2A : DTE ACTIVE ,2B : DTE NOT ACTIVE
      79                                 ; STATEDIAG 2A
      79  00AC  06 0D           DTEACTIVE: LD      B,00DH          ;SET NUMBER OF SETMSM
      79  00AE  21 00 16                 LD      HL,TIMEADD      ;SET TIMEBUFF
      79  00B1  C5                       PUSH    BC
      79  00B2  CD 7D 04                 CALL    RECVCHAR        ;NOTE: MAY NOT BE USED WITH NiCd BATT
      79  00B5  C1                       POP     BC
      79  00B6  77                       LD      (HL),A
      79  00B7  23                       INC     HL
      79  00B8  10 F7                    DJNZ    DTEACTIVE+5
      79  00BA  CD 5F 02                 CALL    WRMSM           ;SET TIME MSM5832
      79  00BD  CD FC 00                 CALL    CHKEHMRECV      ;STATEDIAG 3A,4A,5A,6A,7A,8A,9A
      79  00C0  CD 3B 01         DTERNTS: CALL    SCANSEQ         ;CHK 10A,11A,12A,13A,14A,15A,16A,17A
      79  00C3  CD 41 04                 CALL    RDSTATUS        ;CHK STATEDIAG 22A,23A
      79  00C6  79                       LD      A,C
      79  00C7  E6 40                    AND     40H
```

```
ERR  LINE  ADDR  OBJ

      79  00C9  FE 40                    CP      40H             ;CHK RI ACTIVE
      79  00CB  CC AE 02                 CALL    Z,AUTOANSWER
      79  00CE  CD 41 04                 CALL    RDSTATUS
      79  00D1  79                       LD      A,C             ;CHK STATEDIAG 2A OR 2B
      79  00D2  E6 20                    AND     20H
      79  00D4  FE 20                    CP      20H             ;CHK DTE ACTIVE AGAIN
      79  00D6  20 02                    JR      NZ,DTENOTACTIVE
      79  00D8  18 E6                    JR      DTERNTS
      79                                 ; STATEDIAG 2B
      79  00DA  CD 17 07        DTENOTACTIVE: CALL    SETMODE3        ;STATEDIAG 3B MODE 3 ACTIVE
      79  00DD  CD 41 04                 CALL    RDSTATUS        ;CHK STATEDIAG 7B RING INDICATION
      79  00E0  79                       LD      A,C
      79  00E1  E6 40                    AND     40H
      79  00E3  FE 40                    CP      40H
      79  00E5  CC 4B 06                 CALL    Z,EMMREAD2
      79  00E8  CD EF 06                 CALL    SETMODE1
      79  00EB  06 FF                    LD      B,0FFH
      79  00ED  CD 1D 04                 CALL    LOOP
      79  00F0  CD 41 04                 CALL    RDSTATUS        ;CHK STATEDIAG 2A OR 2B
      79  00F3  79                       LD      A,C
      79  00F4  E6 20                    AND     20H
      79  00F6  FE 20                    CP      20H
      79  00F8  28 E0                    JR      NZ,DTENOTACTIVE
      79  00FA  18 B0                    JR      DTEACTIVE
      79                                 ; STATEDIAG 3A
      79  00FC  DD 21 23 16     CHKEMMRECV: LD      IX,RER          ;CHK STATEDIAG 3A
      79  0100  CD 7D 04        DTECHKEMM:  CALL    RECVCHAR        ;CHK SEND CMD
      79  0103  FE 53                    CP      053H            ;S CMD
      79  0105  28 F9                    JR      NZ,DTECHKEMM    ;WAIT FOR S CHAR
      79                                 ; STATEDIAG 5A
      79  0107  CD 03 07                 CALL    SETMODE2        ;SET MODE 2 FOR REPLY
      79  010A  3E 02                    LD      A,02H
      79  010C  CD A5 02                 CALL    LOOPD
      79  010F  3E CC                    LD      A,0CCH
      79  0111  CD 48 04                 CALL    SENDCHAR
      79  0114  DD 7E 00                 LD      A,(IX+0)
      79  0117  FE 00                    CP      00H
      79  0119  20 17                    JR      NZ,RTNMG
      79  011B  DD 7E 01                 LD      A,(IX+1)
      79  011E  FE 00                    CP      00H
      79  0120  20 10                    JR      NZ,RTNMG
      79  0122  DD 7E 02                 LD      A,(IX+2)
      79  0125  FE 00                    CP      00H
      79  0127  28 09                    JR      NZ,RTNMG        ;STATEDIAG 4A
      79                                 ; STATEDIAG 7A
      79  0129  3E 4E           NORTNMG:  LD      A,04EH          ;SEND 'NO' REPLY
      79  012B  CD 48 04                 CALL    SENDCHAR
      79                                 ; STATEDIAG 9A
      79  012E  CD EF 06                 CALL    SETMODE1
      79  0131  C9                       RET
```

```
ERR  LINE  ADDR  OBJ

      79                                    ; STATEDIAG 6A
      79  0132  3E 59          RTNMG:      LD      A,'Y'           ;LD 'Y' CHAR REPLY
      79  0134  CD 48 04                    CALL    SENDCHAR
      79                                    ;STATEDIAG 9A
      79  0137  CD E6 04                    CALL    EMMWRITE1
      79  013A  C9                         RET
      79  013B
      79  013B
      80                                    FILE    SCANSEQ
      80                                    ; ATTENTION ROUTINE
      80                                    ; STATEDIAG 10A
      80                                    ; CHK STATEDIAG 11A,12A,13A
      80  013B  DB C5           SCANSEQ:    IN      A,(LSR)         ;CHK RECEIVE STATUS
      80  013D  E6 01                       AND     01H
      80  013F  FE 01                       CP      01H
      80  0141  C0                         RET     NZ
      80  0142  3A 5B 16                    LD      A,(CMDSTATE)
      80  0145  FE 00                       CP      00H
      80  0147  20 46                       JR      NZ,ESCSEQ
      80  0149  CD 7D 04        ATSEQ:      CALL    RECVCHAR
      80  014C  FE 41                       CP      41H             ;'A'
      80  014E  C0                         RET NZ
      80  014F  CD 7D 04                    CALL    RECVCHAR
      80  0152  FE 54                       CP      54H             ;'T'
      80  0154  C0                         RET NZ
      80  0155  21 33 16        CMDSEQ:     LD      HL,CMDROOM      ;LOAD COMMAND SEQ. AND DELIMITED
      80  0158  CD 7D 04                    CALL    RECVCHAR
      80  015B  77                         LD      (HL),A
      80  015C  23                         INC     HL
      80  015D  FE 13                       CP      13H
      80  015F  20 F7                       JR      NZ,CMDSEQ+3
      80  0161  21 33 16                    LD      HL,CMDROOM
      80  0164  7E                         LD      A,(HL)
      80  0165  23                         INC     HL              ;FOR NEXT CHAR
      80  0166  FE 44                       CP      44H             ;'D'
      80  0168  CC 96 03                    CALL    Z,AUTODIAL
      80  016B  FE 6F                       CP      86FH            ;'o'
      80  016D  CC B7 01                    CALL    Z,MODE
      80  0170  FE 61                       CP      861H            ;'a'
      80  0172  CC B7 01                    CALL    Z,MODE
      80  0175  FE 48                       CP      48H             ;'H'
      80  0177  CC 88 03                    CALL    Z,HOOK
      80  017A  FE 53                       CP      53H             ;'S'
      80  017C  CC BB 01                    CALL    Z,STATUS
      80  017F  FE 5A                       CP      5AH             ;'Z'
      80  0181  CC C3 01                    CALL    Z,RESET
      80  0184  FE 4B                       CP      4BH             ;'K'
      80  0186  CC C8 01                    CALL    Z,CANCEL
      80                                    ; STATEDIAG 13A,16A,17A
      80  0189  FE 52                       CP      52H             ;'R'
```

```
ERR  LINE  ADDR  OBJ

      80  018B  CC CB 05                    CALL    I,EMMREAD
      80  018E  C9                          RET
      80  018F  CD AB 01        ESCSEQ:     CALL    SPLOOP          ;'  '
      80  0192  CD 9F 01                    CALL    PLLOOP
      80  0195  CD AB 01                    CALL    SPLOOP
      80  0198  3E FF                       LD      A,0FFH
      80  019A  32 5B 16                    LD      (CMDSTATE),A    ;TOGGLE CMD SEQ
      80  019D  18 B6                       JR      CMDSEQ
      80  019F  16 03           PLLOOP:     LD      D,03H
      80  01A1  CD 7D 04                    CALL    RECVCHAR        ;CHK 3 +++
      80  01A4  FE 2B                       CP      2BH
      80  01A6  C0                          RET     NZ              ;'+++'
      80  01A7  15                          DEC     D
      80  01A8  20 F5                       JR      NZ,PLLOOP
      80  01AA  C9                          RET
      80  01AB  16 03           SPLOOP:     LD      D,03H
      80  01AD  CD 7D 04                    CALL    RECVCHAR        ;CHK 3 SPACE AGAIN
      80  01B0  FE 20                       CP      20H
      80  01B2  C0                          RET     NZ              ;'  '
      80  01B3  15                          DEC     D
      80  01B4  20 F5                       JR      NZ,SPLOOP
      80  01B6  C9                          RET
      80  01B7  32 32 16        MODE:       LD      (SMODE),A
      80  01BA  C9                          RET
      80  01BB  21 31 16        STATUS:     LD      HL,SLINE        ;LINE MONITOR STATUS
      80  01BE  7E                          LD      A,(HL)
      80  01BF  CD 4B 04                    CALL    SENDCHAR
      80  01C2  C9                          RET
      80  01C3  FD E1           RESET:      POP     IY
      80  01C5  C3 AC 00                    JP      DTEACTIVE
      80  01C8  CD 41 04        CANCEL:     CALL    RDSTATUS        ;CANCEL CMD JMP RD DSR
      80  01CB  79                          LD      A,C
      80  01CC  FE 2B                       CP      2BH
      80  01CE  C2 DA 00                    JP      NZ,DTENOTACTIVE
      80  01D1  C3 AC 00                    JP      DTEACTIVE
      81                                    FILE    RDMSM
      81                                    ; SUBROUTINE FOR READ SEC,MINUTE,HOUR,DATE,MONTH,YEAR
      81                                    ; FOR WRITE SEC,MINUTE,HOUR,DATE,MONTH,YEAR
      81                                    ; OUTPUT REG.:
      81  01D4  CD 35 02        CHKMSM:     CALL    RDMSM
      81  01D7  DD 21 02 16                 LD      IX,TIMEADD+2    ;BUFF ADD FOR RDMSM
      81  01DB  21 17 10                    LD      HL,BUFF+17H     ;BUFF ON EMM WRITE MEMO
      81  01DE  11 2B 00                    LD      DE,2BH
      81  01E1  FD 21 27 16     CHKEMM:     LD      IY,SER          ;CHK SEND REG.
      81  01E5  FD 7E 00                    LD      A,(IY+0)
      81  01E8  FE 00                       CP      00H
      81  01EA  20 19                       JR      NZ,CHKTIME
      81  01EC  21 17 11                    LD      HL,BUFF+117H
      81  01EF  FD 23                       INC     IY
      81  01F1  FD 7E 00                    LD      A,(IY+0)
```

```
ERR  LINE  ADDR  OBJ

      81  01F4  FE 00                    CP      00H
      81  01F6  20 00                    JR      NZ,CHKTIME
      81  01F8  21 17 12                 LD      HL,BUFF+217H
      81  01FB  FD 23                    INC     IY
      81  01FD  FD 7E 00                 LD      A,(IY+0)
      81  0200  FE 00                    CP      00H
      81  0202  20 01                    JR      NZ,CHKTIME
      81  0204  C9                       RET
      81  0205  E5              CHKTIME: PUSH    HL
      81  0206  CB 47                    BIT     0,A
      81  0208  28 05                    JR      NZ,TIMECHK
      81  020A  19                       ADD     HL,DE
      81  020B  CB 3F                    SRL     A
      81  020D  18 F6                    JR      CHKTIME
      81  020F  DD 7E 03        TIMECHK: LD      A,(IX+3)       ;CHK ONLY HOUR BYTE
      81  0212  46                       LD      B,(HL)
      81  0213  90                       SUB     B
      81  0214  D8                       RET     C
      81  0215  C2 85 05                 JP      NZ,EMMWRITE2
      81  0218  2B                       DEC     HL
      81  0219  DD 7E 02                 LD      A,(IX+2)       ;MIN
      81  021C  46                       LD      B,(HL)
      81  021D  90                       SUB     B
      81  021E  D8                       RET     C
      81  021F  C2 85 05                 JP      NZ,EMMWRITE2
      81  0222  2B                       DEC     HL
      81  0223  DD 7E 01                 LD      A,(IX+1)       ;HOUR
      81  0226  46                       LD      B,(HL)
      81  0227  90                       SUB     B
      81  0228  D8                       RET     C
      81  0229  C2 85 05                 JP      NZ,EMMWRITE2
      81  022C  2B                       DEC     HL
      81  022D  DD 7E 00                 LD      A,(IX+0)       ;HOUR
      81  0230  46                       LD      B,(HL)
      81  0231  90                       SUB     B
      81  0232  D8                       RET     C
      81  0233  C3 85 05                 JP      EMMWRITE2
      81  0236  3E 81            RDMSM:  LD      A,81H          ;INIT 8255-1 PORT A,B=OUTPUT
      81  0238  D3 D3                    OUT     (CTL1),A       ;PORT C(UPPER)=OUTPUT LOWER=INP
      81  023A  CD 95 02                 CALL    INITPORT
      81  023D  21 00 16                 LD      HL,TIMEADD     ;SET INDEX FOR TIME DATA
      81  0240  0E 20                    LD      C,020H         ;SET CTLCODE FOR READ TIME
      81  0242  3A 21 16         NTRD:   LD      A,(SPORTB)
      81  0245  81                       OR      C
      81  0246  D3 D1                    OUT     (PORTB),A      ;READ-HIGH,HOLD-LOW
      81  0248  06 17                    LD      B,17H          ;DELAY ABOUT 150 us
      81  024A  10 FE                    DJNZ    $
      81  024C  D3 D1                    OUT     (PORTB),A      ;ADD. OUT
      81  024E  06 01                    LD      B,01H          ;DELAY REQUIRED 6 us
      81  0250  10 FE                    DJNZ    $
```

```
ERR  LINE  ADDR  OBJ

      81  0252  DB D2
      81  0254  E6 0F
      81  0256  77              SAVETM:   LD      (HL),A          ;SAVE DATA 13 BYTES IN HL
      81  0257  0C                        INC     C               ;READ NEXT DATA
      81  0258  23                        INC     HL
      81  0259  79                        LD      A,C
      81  025A  FE 2D                     CP      02DH            ;CHECK END TIME DATA
      81  025C  38 E4                     JR      C,NTRD
      81  025E  C9                        RET
      81  025F  3E 80            WRMSM:    LD      A,80H           ;INIT 8255 PORT A,B,C=OUTPUT
      81  0261  D3 D3                     OUT     (CTL1),A
      81  0263  CD 95 02                  CALL    INITPORT
      81  0266  21 00 16                  LD      HL,TIMEADD
      81  0269  0E 10                     LD      C,010H
      81  026B  3A 21 16          NTRD0:    LD      A,(SPORTB)
      81  026E  B1                        OR      C
      81  026F  D3 D1                     OUT     (PORTB),A       ;SET WR TIME INIT
      81  0271  06 17             WRTIME:   LD      B,17H           ;DELAY REQ 150 uS
      81  0273  10 FE                     DJNZ    $
      81  0275  7E                        LD      A,(HL)          ;LOAD DATA TO SET MSM
      81  0276  ED 5B 22 16               LD      DE,(SPORTC)     ;USE 4 BITS FOR MSM ONLY
      81  027A  B2                        OR      D
      81  027B  D3 D2                     OUT     (PORTC),A       ;OUT DATA
      81  027D  79                        LD      A,C
      81  027E  D3 D1                     OUT     (PORTB),A       ;ADD
      81  0280  C6 40                     ADD     A,40H
      81  0282  D3 D1                     OUT     (PORTB),A       ;WR ENT DELAY REQ 0.5 uS
      81  0284  0C                        INC     C
      81  0285  23                        INC     HL
      81  0286  79                        LD      A,C
      81  0287  D3 D1                     OUT     (PORTB),A       ;DISABLE WR
      81  0289  FE 1D                     CP      01DH            ;CHECK END OF TIME DATA
      81  028B  38 DE                     JR      C,NTRD0
      81  028D  3E 81                     LD      A,81H           ;RTN TO NOR OP.
      81  028F  D3 D3                     OUT     (CTL1),A
      81  0291  CD 95 02                  CALL    INITPORT
      81  0294  C9                        RET
      81  0295  3A 20 16          INITPORT: LD      A,(SPORTA)
      81  0298  D3 D0                     OUT     (PORTA),A
      81  029A  3A 21 16                  LD      A,(SPORTB)
      81  029D  D3 D1                     OUT     (PORTB),A
      81  029F  3A 22 16                  LD      A,(SPORTC)
      81  02A2  D3 D2                     OUT     (PORTC),A
      81  02A4  C9                        RET
      81  02A5
      81  02A5
      81  02A5
      82                                  FILE    ANSWER
      82  02A5  06 FF             LOOPD:    LD      B,0FFH
      82  02A7  CD 1D 04                    CALL    LOOP
```

```
ERR  LINE  ADDR  OBJ

      82  02AA  3D                        DEC     A
      82  02AB  20 F8                     JR      NZ,LOOPD
      82  02AD  C9                        RET
      82  02AE  3E 86         AUTOANSWER: LD      A,86H       ;CALL BILLING DELAY
      82  02B0  CD A5 02                  CALL    LOOPD
      82  02B3  CD 5A 03                  CALL    OFFHOOK     ;OFF HOOK REF. HANDSHAKE SEQ.
      82  02B6  3E 02                     LD      A,02H       ;CALL BILLING DELAY
      82  02B8  CD A5 02                  CALL    LOOPD
      82  02BB  CD CC 02                  CALL    CONNECTED
      82  02BE  C9                        RET
      82  02BF  C1           NOTCONNECT:  POP     BC
      82  02C0  CD 35 03                  CALL    NORHOOK
      82  02C3  CD EF 06                  CALL    SETMODE1
      82  02C6  3E 4E                     LD      A,04EH      ;SET NO CARRIER
      82  02C8  32 31 16                  LD      (SLINE),A
      82  02CB  C9                        RET
      82  02CC  3E 43         CONNECTED:  LD      A,043H      ;SET HV CARRIER
      82  02CE  32 31 16                  LD      (SLINE),A
      82  02D1  C9                        RET
      82  02D2  3A 20 16      ANSTONE:    LD      A,(SPORTA)  ;SET ANS TONE
      82  02D5  E6 7F                     AND     ANS
      82  02D7  32 20 16                  LD      (SPORTA),A
      82  02DA  D3 D0                     OUT     (PORTA),A
      82  02DC  C9                        RET
      82  02DD  3A 20 16      RELANSTONE: LD      A,(SPORTA)  ;SET ANS TONE DISA.
      82  02E0  F6 80                     OR      RELANS
      82  02E2  32 20 16                  LD      (SPORTA),A
      82  02E5  D3 D0                     OUT     (PORTA),A
      82  02E7  C9                        RET
      82  02E8  33            ORGDATA:   INC     SP
      82  02E9  33                        INC     SP
      82  02EA  3E 86                     LD      A,86H       ;WAIT FOR ANSWER TONE CARRIER
      82  02EC  CD A5 02                  CALL    LOOPD
      82  02EF  CD CC 02                  CALL    CONNECTED
      82  02F2  C9                        RET
      82  02F3  CD 41 04      BRDETC:     CALL    RDSTATUS    ;CHK BR DETECT IN 9250 REGISTER B
      82  02F6  78                        LD      A,B
      82  02F7  E6 10                     AND     BRDETCHR    ;BREAK DETECT CHAR
      82  02F9  C9                        RET
      82  02FA  3E 96         BRTR:       LD      A,SLCR      ;LD BR CHAR
      82  02FC  F6 40                     OR      BRTRCHR     ;BREAK TRANSMIT CHAR
      82  02FE  D3 C3                     OUT     (LCR),A
      82  0300  C9                        RET
      82  0301  3E 96         BRTRREL:    LD      A,SLCR      ;LD SLCR STATUS
      82  0303  F6 BF                     OR      BRRELCHR
      82  0305  D3 C3                     OUT     (LCR),A
      82  0307  C9                        RET
      82  0308
      82  0308
      83                                  FILE    HOOK
```

```
ERR  LINE  ADDR  OBJ

      83                                    ; STATEDIAG 11A
      83                                    ; THIS ROUTINE FOR CTL ON&OFF HOOK
      83  0308  21 34 16       HOOK:       LD      HL,CMDRGOH+1
      83  030B  7E                         LD      A,(HL)
      83  030C  FE 00                      CP      00H
      83  030E  28 05                      JR      Z,ONHOOK
      83  0310  FE 01                      CP      01H
      83  0312  28 18                      JR      Z,DISCHOOK
      83  0314  C9                         RET
      83  0315  3A 20 16       ONHOOK:     LD      A,(SPORTA)
      83  0318  F6 01                      OR      OFFHOOKA
      83  031A  D3 00                      OUT     (PORTA),A
      83  031C  06 20                      LD      B,020H
      83  031E  CD 1D 04                   CALL    LOOP
      83  0321  3A 21 16                   LD      A,(SPORTB)
      83  0324  EE 00                      XOR     00H
      83  0326  32 21 16                   LD      (SPORTB),A
      83  0329  D3 01                      OUT     (PORTB),A
      83  032B  C9                         RET
      83  032C  3E 4E           DISCHOOK:   LD      A,04EH
      83  032E  32 31 16                   LD      (SLIME),A
      83  0331  CD 35 03                   CALL    NORHOOK
      83  0334  C9                         RET
      83  0335  3A 20 16       NORHOOK:    LD      A,(SPORTA)      ;SET PORT A ON HOOK
      83  0338  E6 FE                      AND     ONHOOKA
      83  033A  32 20 16                   LD      (SPORTA),A
      83  033D  D3 00                      OUT     (PORTA),A
      83  033F  3A 21 16                   LD      A,(SPORTB)      ;SET PORT B OFF HOOK WAIT FOR
      83  0342  F6 00                      OR      OFFHOOKB        ;RING INDICATOR
      83  0344  32 21 16                   LD      (SPORTB),A
      83  0347  D3 01                      OUT     (PORTB),A
      83  0349  06 FF                      LD      B,0FFH
      83  034B  CD 1D 04                   CALL    LOOP
      83  034E  C9                         RET
      83  034F  3A 20 16       PULSEONHOOK:LD      A,(SPORTA)      ;ON PULSE FOR LOOP RESISTANCE
      83  0352  E6 FE                      AND     ONHOOKA
      83  0354  32 20 16                   LD      (SPORTA),A
      83  0357  D3 00                      OUT     (PORTA),A
      83  0359  C9                         RET
      83  035A  3A 20 16       OFFHOOK:    LD      A,(SPORTA)      ;ON PULSE FOR LOOP RESISTANCE
      83  035D  F6 01                      OR      OFFHOOKA
      83  035F  32 20 16                   LD      (SPORTA),A
      83  0362  D3 00                      OUT     (PORTA),A
      83  0364  3A 21 16                   LD      A,(SPORTB)      ;ON TRANSFER PHONE
      83  0367  F6 00                      OR      OFFHOOKB
      83  0369  32 21 16                   LD      (SPORTB),A
      83  036C  D3 01                      OUT     (PORTB),A
      83  036E  C9                         RET
      83  036F  3A 20 16       PULSEOFFHOOK:LD     A,(SPORTA)      ;ON PULSE FOR LOOP RESISTANCE
      83  0372  F6 01                      OR      OFFHOOKA
```

```
ERR  LINE  ADDR  OBJ

      83  0374  32 28 16                LD      (SPORTA),A
      83  0377  D3 D0                   OUT     (PORTA),A
      83  0379  C9                      RET
      83  037A  3E 96          DTRRELEASE: LD    A,SLCR
      83  037C  E6 FE                   AND     11111110B
      83  037E  D3 C4                   OUT     (MCR),A
      83  0380  C9                      RET
      83  0381  3E 96          RTSRELEASE: LD    A,SLCR
      83  0383  E6 FD                   AND     11111101B
      83  0385  D3 C4                   OUT     (MCR),A
      83  0387  C9                      RET
      83  0388  3E 96          SETDTR:  LD       A,SLCR
      83  038A  F6 01                   OR      00000001B
      83  038C  D3 C4                   OUT     (MCR),A
      83  038E  C9                      RET
      83  038F  3E 96          SETRTS:  LD       A,SLCR
      83  0391  F6 02                   OR      00000010B
      83  0393  D3 C4                   OUT     (MCR),A
      83  0395  C9                      RET
      83  0396
      84                                FILE    AUTODIAL
      84                                ;STATEDIAG 9A
      84  0396  21 34 16       AUTODIAL: LD      HL,CMDROOM+1    ;READ FIRST T/P
      84  0399  7E                      LD      A,(HL)
      84  039A  23                      INC     HL
      84  039B  E5                      PUSH    HL
      84  039C  21 3B 16                LD      HL,DIALTYPE     ;LD PARAMETER PULSE OR DTMF
      84  039F  7E                      LD      A,(HL)
      84  03A0  E1                      POP     HL
      84  03A1  FE 40                   CP      40H
      84  03A3  20 3F                   JR      NZ,PULSE
      84  03A5  CD 5A 03       DTMFDIAL: CALL    OFFHOOK
      84  03A8  06 FF                   LD      B,0FFH
      84  03AA  CD 1D 04                CALL    LOOP
      84  03AD  06 02                   LD      B,02H
      84  03AF  C5              BCDLOOP: PUSH    BC
      84  03B0  ED 6F                   RLD
      84  03B2  E6 0F                   AND     0FH             ;COMPARE 'F'
      84  03B4  FE 0F                   CP      0FH
      84  03B6  CA E8 02                JP      Z,ORGDATA
      84  03B9  CD D3 03                CALL    DTMFCALL
      84  03BC  06 50                   LD      B,50H           ;DELAY TIME
      84  03BE  CD 1D 04                CALL    LOOP
      84  03C1  3A 28 16                LD      A,(SPORTA)      ;SILENT PERIOD
      84  03C4  F6 20                   OR      DTMFREL
      84  03C6  D3 D0                   OUT     (PORTA),A
      84  03C8  06 FF                   LD      B,0FFH
      84  03CA  CD 1D 04                CALL    LOOP
      84  03CD  C1                      POP     BC
      84  03CE  10 DF                   DJNZ    BCDLOOP
```

```
ERR  LINE  ADDR  OBJ

      84  03D0  23                        INC     HL
      84  03D1  18 DA                     JR      BCDLOOP-2
      84  03D3  06 00           DTMFCALL: LD      B,00H           ;CONVERT BCD FOR DTMF CHIP
      84  03D5  4F                        LD      C,A
      84  03D6  DD 21 60 07               LD      IX,DTMF
      84  03DA  DD 09                     ADD     IX,BC
      84  03DC  DD 7E 0B                  LD      A,(IX)
      84  03DF  D3 D0                     OUT     (PORTA),A
      84  03E1  3E 0F                     LD      A,0FH           ;MARK REG A. PREVENT EFFECT SCANSEQ
      84  03E3  C9                        RET
      84  03E4  06 02           PULSE:    LD      B,02H
      84  03E6  C5              PULSELOOP: PUSH   BC
      84  03E7  CD 6F 03                  CALL    PULSEOFFHOOK
      84  03EA  06 FF                     LD      B,0FFH          ;((21*127)+13)*255-5
      84  03EC  CD 1D 04                  CALL    LOOP            ;68395=370 ms
      84  03EF  ED 6F                     RLD
      84  03F1  E6 0F                     AND     0FH
      84  03F3  FE 0F                     CP      0FH
      84  03F5  CA E8 02                  JP      Z,ORGDATA
      84  03F8  FE 00                     CP      00H
      84  03FA  20 02                     JR      NZ,NZPULSE
      84  03FC  3E 0A                     LD      A,0AH
      84  03FE  CD 07 04        NZPULSE:  CALL    PULSECALL
      84  0401  C1                        POP     BC
      84  0402  10 E2                     DJNZ    PULSELOOP
      84  0404  23                        INC     HL
      84  0405  18 DD                     JR      PULSE
      84  0407  F5              PULSECALL: PUSH   AF
      84  0408  CD 4F 03                  CALL    PULSEONHOOK
      84  040B  06 2E                     LD      B,2EH           ;((21*127)+13)*46-5
      84  040D  CD 1D 04                  CALL    LOOP            ;123273=66.08 ms
      84  0410  CD 6F 03                  CALL    PULSEOFFHOOK
      84  0413  06 17                     LD      B,17H           ;((21*127)+13)*23-5
      84  0415  CD 1D 04                  CALL    LOOP            ;61635=33.4 ms
      84  0418  F1                        POP     AF
      84  0419  3D                        DEC     A
      84  041A  C8                        RET     Z
      84  041B  18 EA                     JR      PULSECALL
      84  041D  0E A0           LOOP:     LD      C,0A0H
      84  041F  0D              LOOP1:    DEC     C
      84  0420  20 FD                     JR      NZ,LOOP1
      84  0422  10 F9                     DJNZ    LOOP
      84  0424  C9                        RET
      84  0425
      84  0425
      85                                  FILE    PROTOCOL
      85                                  ; 8250 SERIAL COMMUNICATION SUBROUTINE
      85                                  ; SET UP PROTOCOL
      85                                  ; INPUT REGISTER : A
      85                                  ; OUTPUT REGISTER: ON EXIT HAVE STATUS ON BC REG.
```

```
ERR  LINE  ADDR  OBJ

      85                                ; (LINE STATUS REGISTER AND MODEM STATUS REGISTER RESPECTIVELY)
      85                                ; DESTROYED REG. :B,BC
      85                                ; REGISTER MAPPING
      85                                ; REGISTER B:   7= TIMEOUT
      85                                ;               6= TRANS SHIFT REGISTER EMPTY
      85                                ;               5= TRAN HOLDING REGISTER EMPTY
      85                                ;               4= BREAK DETECT
      85                                ;               3= FRAMING ERROR
      85                                ;               2= PARITY ERROR
      85                                ;               1= OVERRUN ERROR
      85                                ;               0= DATA READY
      85                                ; REGISTER C:   7= RECEIVED LINE SIGNAL DETECT
      85                                ;               6= RING INDICATOR
      85                                ;               5= DATA SET READY
      85                                ;               4= CLEAR TO SEND
      85                                ;               3= DELTA RECEIVED LINE SIGNAL DETECT
      85                                ;               2= TRAILING EDGE RING INDICATOR
      85                                ;               1= DELTA DATA SET READY
      85                                ;               0= DELTA CLEAR TO SEND
      85  0425  57          PROTOCOL:   LD      D,A             ;SAVE LCR
      85  0426  07                      RLCA
      85  0427  07                      RLCA
      85  0428  07                      RLCA
      85  0429  E6 07                   AND     07H             ;SELECT PROPER SPEED
      85  042B  CD C3 04                CALL    COMPARE         ;REG. BC SAVE DLM,DLL
      85  042E  3E 80                   LD      A,80H
      85  0430  D3 C3                   OUT     (LCR),A
      85  0432  78                      LD      A,B             ;PROGRAM SPEED
      85  0433  D3 C1                   OUT     (DLM),A
      85  0435  79                      LD      A,C
      85  0436  D3 C0                   OUT     (DLL),A
      85  0438  7A                      LD      A,D
      85  0439  E6 1F                   AND     1FH             ;STRIP 3 BITS OFF
      85  043B  D3 C3                   OUT     (LCR),A         ;SET LCR ALREADY
      85  043D  3E 00                   LD      A,00H           ;INT. ENABLE
      85  043F  D3 C1                   OUT     (DLM),A
      85                                ; READ THE LINE STATUS REGISTER AND MODEM STATUS REGISTER
      85                                ; DESTROYED REG.: BC
      85  0441  DB C5          RDSTATUS: IN      A,(LSR)
      85  0443  47                      LD      B,A
      85  0444  DB C6                   IN      A,(MSR)
      85  0446  4F                      LD      C,A
      85  0447  C9                      RET
      85                                ; SEND CHAR. ON A REG.
      85                                ; DESTROYED REG.:DE,BC
      85                                ; ON EXIT HAVE TIME OUT ERROR ON B TX DATA ON A
      85  0448  F5          SENDCHAR:   PUSH    AF
      85  0449  DB C5                   IN      A,(LSR)         ;CHK TX DATA REG. EMPTY
      85  044B  E6 60                   AND     60H
      85  044D  FE 60                   CP      60H
```

```
ERR  LINE  ADDR  OBJ

      85  044F  20 F8                        JR      NZ,SENDCHAR+1
      85  0451  F1                          POP     AF
      85  0452  5F              ISENDCHAR:  LD      E,A             ;SAVE CHAR TO SEND
      85  0453  3E 03                       LD      A,03H           ;SET DTR,RTS ACTIVE
      85  0455  D3 C4                       OUT     (MCR),A
      85  0457  0E 20                       LD      C,TIMEOUT       ;TEST DSR,RTS
      85  0459  16 00                       LD      D,00H
      85  045B  CD B3 04                    CALL    WAITSTATE
      85  045E  E6 30                       AND     30H
      85  0460  FE 30                       CP      30H
      85  0462  C4 BB 04                    CALL    NZ,CHECKSTATE
      85  0465  FE 01                       CP      01H
      85  0467  28 6F                       JR      Z,RELOAD        ;DECLARE TIME OUT ERR.
      85  0469  16 01                       LD      D,01H           ;TEST TX HOLDING R EMPTY
      85  046B  CD B3 04                    CALL    WAITSTATE
      85  046E  E6 20                       AND     20H
      85  0470  FE 20                       CP      20H
      85  0472  C4 BB 04                    CALL    NZ,CHECKSTATE
      85  0475  FE 01                       CP      01H
      85  0477  28 5F                       JR      Z,RELOAD
      85  0479  7B                          LD      A,E             ;SEND DATA
      85  047A  D3 C0                       OUT     (DLL),A
      85  047C  C9                          RET
      85                                    ; RECV  CHAR ROUTINE
      85                                    ; ON EXIT TIME OUT ERR OR RECV. ERR ON REG. B
      85                                    ; RECV. DATA ON REG. A
      85  047D  DB C5             RECVCHAR:  IN      A,(LSR)
      85  047F  E6 01                       AND     01H
      85  0481  FE 01                       CP      01H
      85  0483  20 F8                       JR      NZ,RECVCHAR
      85  0485  3E 01             IRECVCHAR: LD      A,01H           ;SET DTR
      85  0487  D3 C4                       OUT     (MCR),A
      85  0489  0E 20                       LD      C,TIMEOUT       ;TEST DSR
      85  048B  16 00                       LD      D,00H
      85  048D  CD B3 04                    CALL    WAITSTATE
      85  0490  E6 20                       AND     20H
      85  0492  FE 20                       CP      20H
      85  0494  C4 BB 04                    CALL    NZ,CHECKSTATE
      85  0497  FE 01                       CP      01H
      85  0499  28 3E                       JR      Z,TIMEDOWN
      85  049B  16 01                       LD      D,01H           ;TEST RECV. BUFFER FULL
      85  049D  CD B3 04                    CALL    WAITSTATE
      85  04A0  F5                          PUSH    AF
      85  04A1  E6 01                       AND     01H
      85  04A3  FE 01                       CP      01H
      85  04A5  C4 BB 04                    CALL    NZ,CHECKSTATE
      85  04A8  F1                          POP     AF
      85  04A9  FE 01                       CP      01H
      85  04AB  28 2C                       JR      Z,TIMEDOWN
      85  04AD  E6 1E                       AND     00011110B       ;TEST FOR ERR CONDITION
```

```
ERR  LINE  ADDR  OBJ

      85  04AF  47                        LD      B,A             ;RECV.CHAR
      85  04B0  DB C0                     IN      A,(DLL)         ;GET CHAR FROM LINE
      85  04B2  C9                        RET
      85  04B3  7A              WAITSTATE: LD      A,D
      85  04B4  FE 00                     CP      00H
      85  04B6  28 1D                     JR      Z,INMSR
      85  04B8  DB C5                     IN      A,(LSR)
      85  04BA  C9                        RET
      85  04BB  0D              CHECKSTATE: DEC     C               ;CHECK LOOP
      85  04BC  28 02                     JR      Z,WAITBUZZ
      85  04BE  18 F3                     JR      WAITSTATE
      85  04C0  C6 01            WAITBUZZ: ADD     A,01H           ;SET NON ZERO FLAG
      85  04C2  C9                        RET
      85  04C3  CB 27            COMPARE:  SLA     A
      85  04C5  06 00                     LD      B,00H
      85  04C7  4F                        LD      C,A
      85  04C8  DD 21 5D 07               LD      IX,SPEED
      85  04CC  DD 09                     ADD     IX,BC
      85  04CE  DD 4E 00                  LD      C,(IX+0)
      85  04D1  DD 46 01                  LD      B,(IX+1)
      85  04D4  C9                        RET
      85  04D5  DB C6            INMSR:    IN      A,(MSR)
      85  04D7  C9                        RET
      85  04D8  7B              RELOAD:   LD      A,E
      85  04D9  06 00            TIMEDOWN: LD      B,00H
      85  04DB  C9                        RET
      85  04DC
      95  04DC

      86                                  FILE    EMMWRITE
      86                                  ; THIS SUBROUTINE IS FOR STATEDIAG 6A,14A,4B,6B
      86                                  ; THESE ROUTINES HV 4 SUBROUTINES
      86                                  ; EMMWRITE1 FOR ACU SEND DATA TO DTE
      86                                  ; EMMWRITE2 FOR ACU WRITE DATA TO REMOTE DTE (ELECTRONIC MAIL)
      86                                  ; EMMREAD1  FOR DTE SEND DATA TO ACU(BASED ELECTRONIC MIAL)
      86                                  ; EMMREAD2 FOR REMOTE DTE SEND DATA TO ACU(TERMINAL NOT ACTIVE)
      86                                  ; STATEDIAG 8A
      86  04DC  C5              LOOPC:    PUSH    BC
      86  04DD  0E AF                     LD      C,0AFH
      86  04DF  0D              LOOPC1:   DEC     C
      86  04E0  20 FD                     JR      NZ,LOOPC1
      86  04E2  10 F9                     DJNZ    LOOPC+1
      86  04E4  C1                        POP     BC
      86  04E5  C9                        RET
      86  04E6  DD 21 23 16      EMMWRITE1: LD     IX,RER          ;SET RECV EMM REGISTER
      86  04EA  21 00 13                  LD      HL,RECVBUFF     ;LD FIRST USER AREA
      86  04ED  DD 7E 00                  LD      A,(IX+0)        ;CHK HV MSG
      86  04F0  FE 00                     CP      00H
      86  04F2  C4 1D 05                  CALL    NZ,WRMSG
      86  04F5  DD 23                     INC     IX
      86  04F7  21 00 14                  LD      HL,RECVBUFF+100H;LD NEXT 8 USER AREA
```

```
ERR  LINE  ADDR  OBJ

      86  04FA  DD 7E 00                    LD      A,(IX+0)
      86  04FD  FE 00                       CP      00H
      86  04FF  C4 1D 05                    CALL    NZ,WRMSG
      86  0502  DD 23                       INC     IX
      86  0504  21 00 15                    LD      HL,RECVBUFF+200H
      86  0507  DD 7E 00                    LD      A,(IX+0)
      86  050A  FE 00                       CP      00H
      86  050C  C4 1D 05                    CALL    NZ,WRMSG
      86  050F  3E 00                       LD      A,00H
      86  0511  32 2C 16                    LD      (STRADD),A
      86  0514  3E 90                       LD      A,90H
      86  0516  32 2D 16                    LD      (STRADD+1),A
      86  0519  CD EF 06                    CALL    SETMODE1
      86  051C  C9                          RET
      86  051D  CB 47           WRMSG:      BIT     0,A             ;THIS RTN TEST WHICH USER HV MSG
      86  051F  F5                          PUSH    AF
      86  0520  C4 31 05                    CALL    NZ,SENDHEADER
      86  0523  F1                          POP     AF
      86  0524  CB 3F                       SRL     A
      86  0526  FE 00                       CP      00H
      86  0528  CC 01 05                    CALL    Z,SAVERG        ;SAVE REGISTER
      86  052B  11 20 00                    LD      DE,020H
      86  052E  19                          ADD     HL,DE
      86  052F  18 EC                       JR      WRMSG
      86  0531  CD B3 07        SENDHEADER: CALL    SETMODE2
      86  0534  3E 04                       LD      A,04H
      86  0536  CD A5 02                    CALL    LOOPD
      86  0539  01 09 00                    LD      BC,09H
      86  053C  09                          ADD     HL,BC
      86  053D  06 0B                       LD      B,0BH           ;SEND PROTOCOL & FILENAME
      86  053F  7E              SENDHEAD:   LD      A,(HL)          ;LD CHAR TO SEND
      86  0540  C5                          PUSH    BC
      86  0541  CD 48 04                    CALL    SENDCHAR
      86  0544  06 07                       LD      B,07H           ;DELAY FOR 10 mS
      86  0546  CD BC 04                    CALL    LOOPC
      86  0549  C1                          POP     BC
      86  054A  23                          INC     HL
      86  054B  10 F2                       DJNZ    SENDHEAD        ;CHK UNTIL SEND 11 CHAR
      86  054D  01 08 00                    LD      BC,08H
      86  0550  09                          ADD     HL,BC
      86  0551  4E                          LD      C,(HL)
      86  0552  23                          INC     HL
      86  0553  46                          LD      B,(HL)
      86  0554  23                          INC     HL
      86  0555  5E                          LD      E,(HL)
      86  0556  23                          INC     HL
      86  0557  56                          LD      D,(HL)          ;DE=FILE LENGTH
      86  0558  ED 43 2B 16                 LD      (BUFFADD),BC
      86  055C  DD 2A 2B 16                 LD      IX,(BUFFADD)    ;IX=STR ADD
      86  0560  EB                          EX      DE,HL           ;HL=FILE LENGTH
```

```
ERR  LINE  ADDR  OBJ

      86  0561  11 FF FF              LD      DE,-1
      86  0564  3E 04                 LD      A,04H
      86  0566  CD A5 02              CALL    LOOPD
      86  0569  DD 7E 00    SENDMESS:  LD      A,(IX+0)
      86  056C  D5                    PUSH    DE
      86  056D  CD 48 04              CALL    SENDCHAR
      86  0570  06 07                 LD      B,07H
      86  0572  CD DC 04              CALL    LOOPC
      86  0575  D1                    POP     DE
      86  0576  DD 23                 INC     IX
      86  0578  19                    ADD     HL,DE
      86  0579  38 EE                 JR      C,SENDMESS
      86  057B  3E 04                 LD      A,04H
      86  057D  CD 48 04              CALL    SENDCHAR
      86  0580  C9                    RET
      86  0581  DD 77 00    SAVEREG:   LD      (IX+0),A        ;SAVE UPDATED REGISTER
      86  0584  C9                    RET
      86                             ; STATEDIAG 19A
      86  0585  E1          EMMWRITE2: POP     HL
      86  0586  D5                    PUSH    DE
      86  0587  C5                    PUSH    BC
      86  0588  F5                    PUSH    AF
      86  0589  CD 17 07              CALL    SETMODE3
      86  058C  01 E9 FF              LD      BC,-17H
      86  058F  09                    ADD     HL,BC
      86  0590  E5                    PUSH    HL
      86  0591  CD 9B 03              CALL    AUTODIAL+5
      86  0594  E1                    POP     HL
      86  0595  3A 31 16              LD      A,(SLINE)
      86  0598  FE 43                 CP      43H
      86  059A  CC A7 05              CALL    Z,SENDFAX
      86  059D  CD EF 06              CALL    SETMODE1
      86  05A0  CD 35 03              CALL    NORHOOK
      86  05A3  F1                    POP     AF
      86  05A4  C1                    POP     BC
      86  05A5  D1                    POP     DE
      86  05A6  C9                    RET
      86  05A7  CD 8F 03    SENDFAX:   CALL    SETRTS
      86  05AA  CD 88 03              CALL    SETDTR
      86  05AD  3E 02                 LD      A,02H
      86  05AF  CD A5 02              CALL    LOOPD
      86  05B2  3E 46                 LD      A,46H
      86  05B4  CD 48 04              CALL    SENDCHAR
      86  05B7  06 07                 LD      B,07H
      86  05B9  CD DC 04              CALL    LOOPC
      86  05BC  06 03                 LD      B,03H
      86  05BE  CD A5 02              CALL    LOOPD
      86  05C1  01 09 00              LD      BC,09H
      86  05C4  09                    ADD     HL,BC
      86  05C5  06 00                 LD      B,00H
```

```
ERR  LINE  ADDR  OBJ

      86  05C7  CD 3F 05                    CALL    SENDHEAD
      86  05CA  C9                         RET
      86                                    ; STATEDIAG 20A
      86                                    ; STATEDIAG 13A,16A
      86  05CB  F3              EMMREAD:    DI
      86  05CC  CD 03 07                    CALL    SETMODE2
      86  05CF  DD 21 27 16     EMMREAD1:   LD      IX,SER          ;LD IX BY SEND EMM REGISTER
      86  05D3  21 00 10                    LD      HL,BUFF         ;LD HL WITH FIRST 8 USERS AREA
      86  05D6  DD 7E 00                    LD      A,(IX+0)        ;CHK NUMBER OF MSG
      86  05D9  FE FF                       CP      0FFH
      86  05DB  20 19                       JR      NZ,RECVMSG
      86  05DD  21 00 11                    LD      HL,BUFF+100H    ;LD HL FOR CHK NEXT 8 USER AREA
      86  05E0  DD 23                       INC     IX
      86  05E2  DD 7E 00                    LD      A,(IX+0)
      86  05E5  FE FF                       CP      0FFH
      86  05E7  20 0D                       JR      NZ,RECVMSG
      86  05E9  21 00 12                    LD      HL,BUFF+200H
      86  05EC  DD 23                       INC     IX
      86  05EE  DD 7E 00                    LD      A,(IX+0)
      86  05F1  FE FF                       CP      0FFH
      86  05F3  20 01                       JR      NZ,RECVMSG
      86  05F5  C9                         RET
      86  05F6  CB 47           RECVMSG:    BIT     0,A             ;TEST UNTIL FIND SP ROOM
      86  05F8  28 08                       JR      Z,RECVHEADDER
      86  05FA  CB 3F                       SRL     A
      86  05FC  11 20 00                    LD      DE,020H
      86  05FF  19                         ADD     HL,DE
      86  0600  18 F4                       JR      RECVMSG
      86  0602  DD 7E 00        RECVHEADDER: LD      A,(IX+0)
      86  0605  CB 27                       SLA     A
      86  0607  F6 01                       OR      01H
      86  0609  DD 77 00                    LD      (IX+0),A
      86  060C  06 1C                       LD      B,01CH          ;SET 28 RECV CHAR
      86  060E  C5              RECVHEAD:   PUSH    BC
      86  060F  CD 7D 04                    CALL    RECVCHAR
      86  0612  77                         LD      (HL),A
      86  0613  23                         INC     HL
      86  0614  C1                         POP     BC
      86  0615  10 F7                       DJNZ    RECVHEAD
      86  0617  ED 4B 2E 16                 LD      BC,(LASTADD)    ;LAST ADD WHICH LOCATE ON 6116
      86  061B  71                         LD      (HL),C
      86  061C  23                         INC     HL
      86  061D  70                         LD      (HL),B          ;SET STR. ADD
      86  061E  23                         INC     HL
      86  061F  CD 7D 04                    CALL    RECVCHAR        ;SET REGISTER BC FOR UPDATED LAST ADD
      86  0622  77                         LD      (HL),A
      86  0623  5F                         LD      E,A
      86  0624  23                         INC     HL
      86  0625  D5                         PUSH    DE
      86  0626  CD 7D 04                    CALL    RECVCHAR
```

```
ERR  LINE  ADDR  OBJ

      86  0629  D1                        POP     DE
      86  062A  77                        LD      (HL),A
      86  062B  57                        LD      D,A             ;DE=FILE LENGTH
      86  062C  DD 2A 2E 16                LD      IX,(LASTADD)
      86  0630  2A 2E 16                  LD      HL,(LASTADD)
      86  0633  19                        ADD     HL,DE           ;SET LAST ADD. AGAIN
      86  0634  23                        INC     HL
      86  0635  22 2E 16                  LD      (LASTADD),HL
      86  0638  EB                        EX      DE,HL           ;HL=FILE LENGTH
      86  0639  11 FF FF                  LD      DE,-1
      86  063C  D5            READDATA:   PUSH    DE
      86  063D  CD 7D 04                  CALL    RECVCHAR
      86  0640  D1                        POP     DE
      86  0641  DD 77 00                  LD      (IX+0),A
      86  0644  DD 23                     INC     IX
      86  0646  19                        ADD     HL,DE
      86  0647  38 F3                     JR      C,READDATA
      86  0649  FB                        EI
      86  064A  C9                        RET
      86                                 ; STATEDIAG 9B
      86  064B  CD AE 02       EMNREAD2:   CALL    AUTOANSWER
      86  064E  3A 31 16                  LD      A,(SLINE)
      86  0651  FE 43                     CP      43H
      86  0653  C0                        RET     NZ
      86  0654  CD 17 07                  CALL    SETMODE3
      86  0657  CD 9F 03                  CALL    SETRTS
      86  065A  CD 88 03                  CALL    SETDTR
      86  065D  06 FF                     LD      B,0FFH
      86  065F  CD 1D 04                  CALL    LOOP
      86  0662  3E 55                     LD      A,055H
      86  0664  CD 4B 04                  CALL    SENDCHAR
      86  0667  CD 91 03                  CALL    RTSRELEASE
      86  066A  DD 21 23 16               LD      IX,RER          ;LD IX BY SEND EMM REGISTER
      86  066E  21 00 13                  LD      HL,RECVBUFF     ;LD HL WITH FIRST 8 USERS AREA
      86  0671  DD 7E 00                  LD      A,(IX+0)        ;CHK NUMBER OF MSG
      86  0674  FE FF                     CP      0FFH
      86  0676  20 19                     JR      NZ,RECVMSGIN
      86  0678  21 00 14                  LD      HL,RECVBUFF+100H;LD HL FOR CHK NEXT 8 USER AREA
      86  067B  DD 23                     INC     IX
      86  067D  DD 7E 00                  LD      A,(IX+0)
      86  0680  FE FF                     CP      0FFH
      86  0682  20 0D                     JR      NZ,RECVMSGIN
      86  0684  21 00 15                  LD      HL,RECVBUFF+200H
      86  0687  DD 23                     INC     IX
      86  0689  DD 7E 00                  LD      A,(IX+0)
      86  068C  FE FF                     CP      0FFH
      86  068E  20 01                     JR      NZ,RECVMSGIN
      86  0690  C9                        RET
      86  0691  CB 47          RECVMSGIN:  BIT     0,A             ;TEST UNTIL FIND SP ROOM
      86  0693  28 08                     JR      Z,RECVTAG
```

```
ERR  LINE  ADDR  OBJ

      86  0695  CB 3F                        SRL     A
      86  0697  11 20 00                     LD      DE,0020H
      86  069A  19                           ADD     HL,DE
      86  069B  18 F4                        JR      RECVHSGIN
      86  069D  DD 7E 00        RECVTAG:     LD      A,(IX+0)
      86  06A0  CB 27                        SLA     A
      86  06A2  F6 01                        OR      01H
      86  06A4  DD 77 00                     LD      (IX+0),A
      86  06A7  CD 7D 04                     CALL    RECVCHAR
      86  06AA  FE 48                        CP      48H
      86  06AC  20 F9                        JR      NZ,$-5
      86  06AE  06 1C                        LD      B,01CH          ;SET 28 RECV CHAR
      86  06B0  C5              RECVHEADTAG: PUSH    BC
      86  06B1  CD 7D 04                     CALL    RECVCHAR
      86  06B4  77                           LD      (HL),A
      86  06B5  23                           INC     HL
      86  06B6  C1                           POP     BC
      86  06B7  10 F7                        DJNZ    RECVHEADTAG
      86  06B9  ED 4B 2C 16                  LD      BC,(STRADD)     ;LAST ADD WHICH LOCATE ON 6116
      86  06BD  71                           LD      (HL),C
      86  06BE  23                           INC     HL
      86  06BF  70                           LD      (HL),B          ;SET STR. ADD
      86  06C0  23                           INC     HL
      86  06C1  CD 7D 04                     CALL    RECVCHAR        ;SET REGISTER BC FOR UPDATED LAST ADD
      86  06C4  77                           LD      (HL),A
      86  06C5  5F                           LD      E,A
      86  06C6  23                           INC     HL
      86  06C7  D5                           PUSH    DE
      86  06C8  CD 7D 04                     CALL    RECVCHAR
      86  06CB  D1                           POP     DE
      86  06CC  77                           LD      (HL),A
      86  06CD  57                           LD      D,A             ;DE=FILE LENGTH
      86  06CE  DD 2A 2C 16                  LD      IX,(STRADD)
      86  06D2  2A 2C 16                     LD      HL,(STRADD)
      86  06D5  19                           ADD     HL,DE           ;SET LAST ADD. AGAIN
      86  06D6  23                           INC     HL
      86  06D7  22 2C 16                     LD      (STRADD),HL
      86  06DA  EB                           EX      DE,HL           ;HL=FILE LENGTH
      86  06DB  11 FF FF                     LD      DE,-1
      86  06DE  D5              READDATAIN:  PUSH    DE
      86  06DF  CD 7D 04                     CALL    RECVCHAR
      86  06E2  D1                           POP     DE
      86  06E3  DD 77 00                     LD      (IX+0),A
      86  06E6  DD 23                        INC     IX
      86  06E8  19                           ADD     HL,DE
      86  06E9  38 F3                        JR      C,READDATAIN
      86  06EB  CD 35 03                     CALL    NORMOSK
      86  06EE  C9                           RET
      86  06EF  06 0F            SETMODE1:   LD      B,0FH
      86  06F1  CD 1D 04                     CALL    LOOP
```

```
ERR  LINE  ADDR  OBJ

      86  06F4  3A 22 16                LD      A,(SPORTC)      ;LD CURRENT STATUS OF PORT C
      86  06F7  CB F7                   SET     6,A
      86  06F9  CB FF                   SET     7,A
      86  06FB  E6 CF                   AND     MODE1           ;PRG FOR MODE1 COMM
      86  06FD  32 22 16                LD      (SPORTC),A
      86  0700  D3 D2                   OUT     (PORTC),A
      86  0702  C9                      RET
      86  0703  06 0F          SETMODE2: LD      B,0FH
      86  0705  CD 1D 04                CALL    LOOP
      86  0708  3A 22 16                LD      A,(SPORTC)      ;LD CURRENT STATUS OF PORT C
      86  070B  CB E7                   SET     4,A
      86  070D  CB FF                   SET     7,A
      86  070F  E6 9F                   AND     MODE2           ;PRG FOR MODE2 COMM
      86  0711  32 22 16                LD      (SPORTC),A
      86  0714  D3 D2                   OUT     (PORTC),A
      86  0716  C9                      RET
      86  0717  06 0F          SETMODE3: LD      B,0FH
      86  0719  CD 1D 04                CALL    LOOP
      86  071C  3A 22 16                LD      A,(SPORTC)      ;LD CURRENT STATUS OF PORT C
      86  071F  CB E7                   SET     4,A
      86  0721  CB EF                   SET     5,A
      86  0723  CB F7                   SET     6,A
      86  0725  E6 7F                   AND     MODE3           ;PRG FOR MODE3 COMM
      86  0727  32 22 16                LD      (SPORTC),A
      86  072A  D3 D2                   OUT     (PORTC),A
      86  072C  C9                      RET
      86  072D
      87  072D  00              INTRTN0: NOP
      88  072E  F5              INTRTN1: PUSH    AF
      89  072F  3E 03                   LD      A,03H           ;STOP TIMER CH0
      90  0731  D3 40                   OUT     (CTC1CH0),A
      91  0733  3E 0F                   LD      A,0FH           ;SET CTC CH0 AGAIN
      92  0735  D3 40                   OUT     (CTC1CH0),A
      93  0737  3E 18                   LD      A,18H
      94  0739  D3 40                   OUT     (CTC1CH0),A
      95  073B  F1                      POP     AF
      96  073C  FB                      EI
      97  073D  ED 4D                   RETI
      98  073F  F5              TIMEINT: PUSH    AF
      99  0740  C5                      PUSH    BC
     100  0741  D5                      PUSH    DE
     101  0742  E5                      PUSH    HL
     102  0743  DD E5                   PUSH    IX
     103  0745  FD E5                   PUSH    IY
     104  0747  21 31 16                LD      HL,SLINE
     105  074A  7E                      LD      A,(HL)
     106  074B  FE 43                   CP      043H
     107  074D  28 03                   JR      Z,$+5
     108  074F  CD D4 01                CALL    CHKMSM
     109  0752  FD E1                   POP     IY
```

```
ERR  LINE  ADDR  OBJ

     110  0754  DD E1                        POP     IX
     111  0756  E1                          POP     HL
     112  0757  D1                          POP     DE
     113  0758  C1                          POP     BC
     114  0759  F1                          POP     AF
     115  075A  FB                          EI
     116  075B  ED 4D                       RETI
     117                                    FILE    EQUATE
     117  00FF                PWDLY         EQU     0FFH
     117  0020                TIMEOUT       EQU     20H
     117                                    ;FIRST Z80 CTC REGISTER
     117  0040                CTC1CH0       EQU     040H
     117  0041                CTC1CH1       EQU     041H
     117  0042                CTC1CH2       EQU     042H
     117  0043                CTC1CH3       EQU     043H
     117                                    ;SECOND Z80 CTC REGISTER
     117  0090                CTC2CH0       EQU     090H
     117  0091                CTC2CH1       EQU     091H
     117  0092                CTC2CH2       EQU     092H
     117  0093                CTC2CH3       EQU     093H
     117                                    ;REGISTER 8250 SUCH AS LINE CONTROL ,MODEM CONTROL,
     117                                    ;LINE STATUS,MODEM STATUS REGISTER
     117  00C0                DLL           EQU     0C0H
     117  00C1                DLM           EQU     0C1H
     117  00C3                LCR           EQU     0C3H
     117  00C4                MCR           EQU     0C4H
     117  00C5                LSR           EQU     0C5H
     117  00C6                MSR           EQU     0C6H
     117                                    ;PORT A,B,C IS 8255 PORT
     117  00D0                PORTA         EQU     0D0H
     117  00D1                PORTB         EQU     0D1H
     117  00D2                PORTC         EQU     0D2H
     117  00D3                CTL1          EQU     0D3H
     117                                    ;MODE1,2,3 IS CONDITION COMMUNICATION
     117  00CF                MODE1         EQU     0CFH          ;IBMPC + MODEM ACTIVE
     117  009F                MODE2         EQU     09FH          ;IBMPC + ACU ACTIVE
     117  007F                MODE3         EQU     07FH          ;ACU + MODEM ACTIVE
     117                                    ;ONHOOK OR OFFHOOK
     117  00FE                ONHOOKA       EQU     0FEH
     117  007F                ONHOOKB       EQU     07FH
     117  0001                OFFHOOKA      EQU     001H
     117  0080                OFFHOOKB      EQU     080H
     117  00DF                DTMFEN        EQU     0DFH
     117  0020                DTMFREL       EQU     020H
     117                                    ;ANSWER TONE ACTIVATE OR RELEASE
     117  007F                ANS           EQU     07FH
     117  0080                RELANS        EQU     080H
     117                                    ;SLCR IS STATUS LINE CONTROL REGISTER
     117  0096                SLCR          EQU     096H
     117  0018                BRDETCHR      EQU     018H
```

```
ERR  LINE  ADDR  OBJ

     117  0040            BRTRCHR     EQU    040H
     117  008F            BRRELCHR    EQU    08FH
     117  075D
     117  075D
     118                              FILE   DEFB
     118  075D  17        SPEED:      DEFB   17H     ; 110 BPS
     118  075E  04                    DEFB   04H
     118  075F  00                    DEFB   00H     ; 150 BPS
     118  0760  03                    DEFB   03H
     118  0761  80                    DEFB   80H     ; 300 BPS
     118  0762  01                    DEFB   01H
     118  0763  C0                    DEFB   0C0H    ; 600 BPS
     118  0764  00                    DEFB   00H
     118  0765  60                    DEFB   60H     ;1200 BPS
     118  0766  00                    DEFB   00H
     118  0767  30                    DEFB   30H     ;2400 BPS
     118  0768  00                    DEFB   00H
     118  0769  18                    DEFB   18H     ;4800 BPS
     118  076A  00                    DEFB   00H
     118  076B  0C                    DEFB   0CH     ;9600 BPS
     118  076C  00                    DEFB   00H
     118  076D  9B        DTMF:       DEFB   9BH             ;'0'
     118  076E  81                    DEFB   81H             ;'1'
     118  076F  83                    DEFB   83H             ;'2'
     118  0770  85                    DEFB   85H             ;'3'
     118  0771  89                    DEFB   89H             ;'4'
     118  0772  8B                    DEFB   8BH             ;'5'
     118  0773  8D                    DEFB   8DH             ;'6'
     118  0774  91                    DEFB   91H             ;'7'
     118  0775  93                    DEFB   93H             ;'8'
     118  0776  95                    DEFB   95H             ;'9'
     118  0777  50        TMRCD:      DEFB   50H             ;WR TIMER RCD 1
     118  0778  70                    DEFB   70H
     118  0779  90                    DEFB   90H
     118  077A  B0                    DEFB   0B0H
     118  077B  D0                    DEFB   0D0H
     118  077C
     118  077C
     119                              FILE   DEFS
     119                              org    1000H
     119  1000            BUFF:       DEFS   768             ;24 CUSTOMER USER IDEN.
     119  1300            RECVBUFF:   DEFS   768             ;24 RECV USER IDEN.
     119  1600            TIMEADD:    DEFS   16              ;TIME BUFF ADDRESS FOR RD AND WR TIMER
     119  1610            TELBUFF:    DEFS   16              ;TELEPHONE WORKING AREA
     119  1620            SPORTA:     DEFS   1               ;STATUS OF PORT A
     119  1621            SPORTB:     DEFS   1               ;STATUS OF PORT B
     119  1622            SPORTC:     DEFS   1               ;STATUS OF PORT C
     119  1623            RER:        DEFS   4               ;RECV ENM REG. 4 BYTE 24 USERS
     119  1627            SER:        DEFS   4               ;SEND ENM REG.
     119  162B            BUFFADD:    DEFS   1               ;STORAGE FOR ENMW1 & ENMW2
```

```
ERR  LINE  ADDR  OBJ

     119  162C              STRADD:     DEFS   2          ;FIRST ADDRESS
     119  162E              LASTADD:    DEFS   2          ;LAST ADDRESS
     119  1630              DIALTYPE:   DEFS   1          ;CHK BUFFER PULSE OR DTMF
     119  1631              SLINE:      DEFS   1          ;LINE MONITORING STATUS FOR CHK TEL.
     119  1632              SMODE:      DEFS   1          ;MODE TRANSMIT
     119  1633              CMDROOM:    DEFS   40         ;SPACE FOR COMMAND SEQUENCE
     119  165B              CMDSTATE:   DEFS   1          ;COMMAND STATUS FOR ESC SEQ
     119  165C              SPACE1:     DEFS   256        ;STACK PUSH SPACE
     119  175C              STKPT:      DEFS   1          ;STACK POINTER
     119  175D              SPACE2:     DEFS   2          ;STACK POP SPACE
     119  175F
     120  175F                          END

  ASSEMBLER ERRORS =    0
```

SYMBOL TABLE

ANS	007F	ANSTONE	02D2	ATSEQ	0149
AUTOANSWER	02AE	AUTODIAL	0396	BCDLOOP	03AF
BRDETC	02F3	BRDETCHR	0010	BRRELCHR	00BF
BRTR	02FA	BRTRCHR	0040	BRTREL	0301
BUFF	1000	BUFFADD	162B	CANCEL	01C8
CHECKSTATE	04BB	CHKEMM	01E1	CHKEMMRECV	00FC
CHKMSM	01D4	CHKTIME	0205	CLRMEM	0029
CMDROOM	1633	CMDSEQ	0155	CMDSTATE	1658
COMPARE	04C3	CONNECTED	02CC	CTC1CH0	0040
CTC1CH1	0041	CTC1CH2	0042	CTC1CH3	0043
CTC2CH0	0080	CTC2CH1	0081	CTC2CH2	0082
CTC2CH3	0083	CTL1	00D3	DIALTYPE	1630
DISCHOOK	032C	DLL	00C0	DLM	00C1
DTEACTIVE	00AC	DTECHKEMM	0100	DTENOTACTIVE	00DA
DTERNTS	00C0	DTMF	076D	DTMFCALL	03D3
DTMFDIAL	03A5	DTMFEN	00DF	DTMFREL	0020
DTRRELEASE	037A	EMMREAD	05CB	EMMREAD1	05CF
EMMREAD2	064B	EMMWRITE1	04E6	EMMWRITE2	0585
ESCSEQ	018F	HOOK	0388	INIT	0026
INITPORT	0295	INMSR	04D5	INTRTN	0010
INTRTN1	072E	INTRTNS	072D	INTVEC	0020
IRECVCHAR	0485	ISENDCHAR	0452	LASTADD	162E
LCR	00C3	LOOP	041D	LOOP1	041F
LOOPC	04DC	LOOPC1	04DF	LOOPD	02A5
LSR	00C5	MCR	00C4	MEMORY	M 0000
MODE	01B7	MODE1	00CF	MODE2	009F
MODE3	007F	MSR	00C6	NARG	0000
NORHOOK	0335	NORTNMG	0129	NOTCONNECT	02BF
NTRD	0242	NTRD0	026B	NZPULSE	03FE
OFFHOOK	035A	OFFHOOKA	0001	OFFHOOKR	0090
ONHOOK	0315	ONHOOKA	00FE	ONHOOKR	007F
ORGDATA	02E9	PLLOOP	019F	PORTA	00D0
PORTB	00D1	PORTC	00D2	PROTOCOL	0425
PULSE	03E4	PULSECALL	0407	PULSELOOP	03E6
PULSEOFFHOOK	036F	PULSEONHOOK	034F	PWDLY	00FF
RDMSM	0236	RDSTATUS	0441	READDATA	063C
READDATAIN	04DE	RECVBUFF	1300	RECVCHAR	047D
RECVHEAD	060E	RECVHEADDER	0602	RECVHEADTAG	06B0
RECVMSG	05F6	RECVMSGIN	0691	RECVTAG	069D
RELANS	0000	RELANSTONE	02DD	RELOAD	04D8
RER	1623	RESET	01C3	RINMG	0132
RTSRELEASE	0381	SAVERG	0581	SAVETM	0256
SCANSEQ	013B	SENDCHAR	044B	SENDFAX	05A7
SENDHEAD	053F	SENDHEADER	0531	SENDMESS	0569
SER	1627	SETDTR	0388	SETMODE1	06EF
SETMODE2	0703	SETMODE3	0717	SETRTS	038F
SLCR	0096	SLINE	1631	SMODE	1632
SPACE1	165C	SPACE2	175D	SPEED	075D
SPLOOP	01A8	SPORTA	1620	SPORTB	1621

SPORTC	1622	STACK	S 0000	STATUS	0188
STKPT	175C	STRADD	162C	TELBUFF	1610
TIMEADD	1600	TIMECHK	020F	TIMEDOWN	04D9
TIMEINT	073F	TIMEOUT	0028	TMRCD	0777
WAITBUZZ	04C0	WAITSTATE	04B3	WRMSG	051D
WRMSM	025F	WRTIME	0271		

ประวัติผู้เขียน

นายวิทยากร อัศดรวิเศษ สำเร็จปริญญาวิศวกรรมศาสตร์บัณฑิต สาขาวิศวกรรมไฟฟ้า จากมหาวิทยาลัยเกษตรศาสตร์ เมื่อ พ.ศ. 2528 เคยทำงานเป็นวิศวกรในบริษัทซีเกทเทคโนโลยี่ จำกัด ปี พ.ศ. 2529-2530 ขณะที่ศึกษาอยู่ในระดับบัณฑิตวิทยาลัย ได้มีโอกาสเสนอบทความผลงานวิจัยในการประชุมวิชาการวิศวกรรมไฟฟ้า 9 สถาบัน ครั้งที่ 9 ณ มหาวิทยาลัยขอนแก่น และ ครั้งที่ 10 ณ จุฬาลงกรณ์มหาวิทยาลัย ในปี 2529 และ 2530 ตามลำดับ

เป็นสมาชิก สภาพนิสิต ของสมาคม IEEE