```
                        .REM @
```

PRODUCT CODE: AC-8496D-MC

PRODUCT NAME: CZDHND0 DH11 DATA RELIAB TST

DATE: JAN 1979

MAINTAINER: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 GENERAL PROGRAM DESCRIPTION

1.1 PROGRAM PURPOSE

"CZDHND" IS A GENERAL PURPOSE TEST AND EXERCISER PROGRAM FOR THE DH11, 16. LINE ASYNCHRONOUS LINE MULTIPLEXOR. IT CONSISTS OF THREE INDEPENDENT SUB-PROGRAMS THAT MAY BE USED FOR ACCEPTANCE TESTING, INSTALLATION CHECKOUT, AND CORRECTIVE MAINTENANCE OF THE DH11 SUB-SYSTEM.

1.1.1 SUBPROGRAM 1 DH11 DATA RELIABILITY TESTS

ONCE CONFIGURED BY THE AUTOSIZER OR BY INITIAL CONSOLE DIALOGUE THIS PROGRAM CAN TEST UP TO 16. DH11'S. ALL LINES ON EACH DH11 ARE TESTED (ONE AT A TIME) WITH ALL COMBINATIONS OF LINE PARAMETERS (BAUD RATE,CHAR LENGTH,PARITY ETC.) BY TRANSMITTING AND RECEIVING A BINARY COUNT PATTERN. ALL ERRORS DETECTED ARE REPORTED ON THE CONSOLE DEVICE AS THEY OCCUR AND ALSO LOGGED IN ERROR STATISTICS TABLES. AT THE COMPLETION OF TESTING FOR EACH DH11 THESE ERROR STATISTICS TABLES ARE DUMPED ON THE CONSOLE DEVICE TO PROVIDE HISTORICAL EVIDENCE OF THE DATA RELIABILITY OF EACH DH11. REFER TO SECTION 4.0 FOR A DETAILED DESCRIPTION OF THE ERROR STATISTICS PROVIDED. THIS SUB-PROGRAM IS NORMALLY SELECTED FOR OVERALL DH CHECKOUT.

1.1.2 SUB-PROGRAM 2 DH11 'NGLE LINE ECHO TEST

THIS PROGRAM PROVIDES THE MEANS OF TESTING ANY LINE ON ANY DH11 BY USING AN ASYNCHRONOUS TERMINAL DEVICE (VT50,LA36 ETC) CONNECTED TO THE LINE UNDER TEST. THIS SUB-PROGRAM WOULD NORMALLY BE SELECTED WHEN A PROBLEM IS ISOLATED TO A SPECIFIC LINE. IT HAS TWO MODES OF OPERATION, SEND MODE OR ECHO MODE:

SEND MODE: THE USER TYPES AN ASCIZ BUFFER IN ON THE CONSOLE DEVICE AND THEN TYPES A UNIQUE CONTROL CHARACTER TO SEND THIS BUFFER TO THE DH11 TEST TERMINAL.

THE USER CAN THEN COMPARE THE TWO IMAGES FOR ACCURACY OF TRANSMISSION.

ECHO MODE: THE USER TYPES IN ON THE DH11 TEST TERMINAL AND CAN OBSERVE EACH CHAR TYPED BEING ECHOED ON THE TERMINAL. BY TYPING A UNIQUE CONTROL CHARACTER THE PROGRAM WILL ECHO THE ENTIRE BUFFER TYPED IN UP TO THAT POINT.

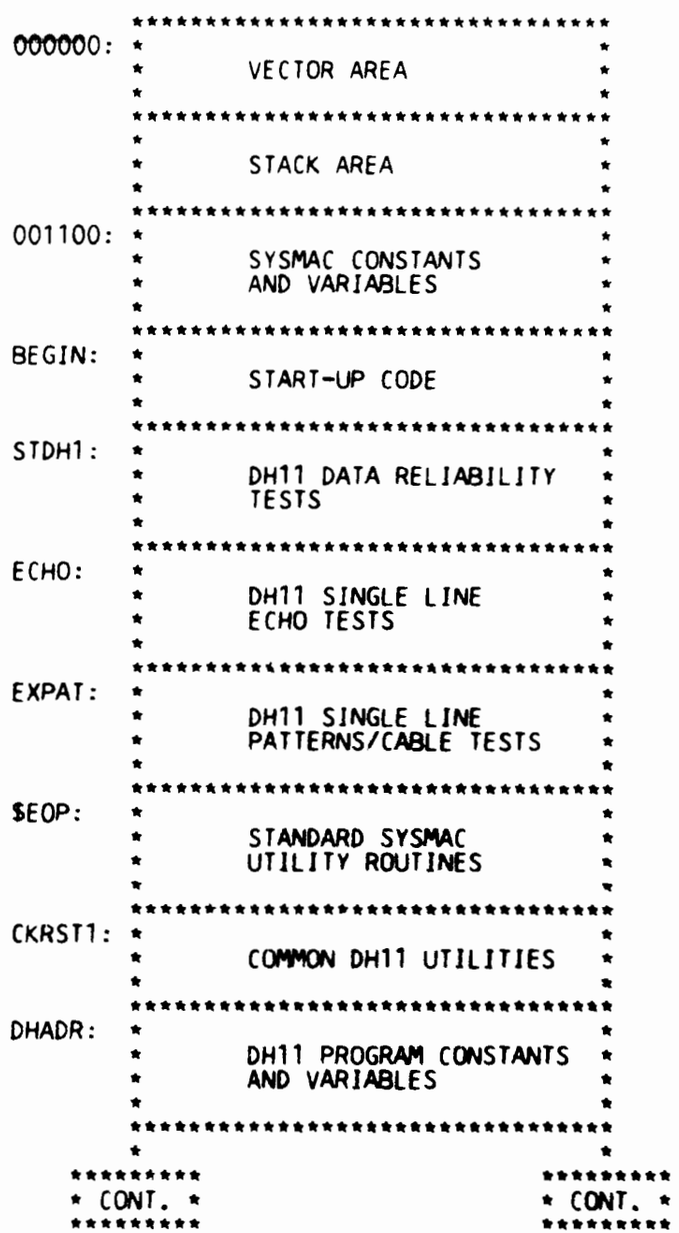
1.1.3 SUBPROGRAM 3 DH11 DATA PATTERNS/CABLE TESTS

THIS PROGRAM PROVIDES THE MEANS OF TESTING ANY LINE ON ANY DH11 USING AN H315 TEST CONNECTOR TO TERMINATE THE LINE UNDER TEST. THE USER CAN SPECIFY BUFFER SIZE AND LINE PARAMETERS PRIOR TO SELECTING ONE OF THE FOLLOWING DATA PATTERNS FOR TRANSMISSION, RECEPTION, AND ERROR CHECKING:

A. ALTERNATING 1/0 PATTERN
B. BINARY UP COUNT PATTERN
C. BINARY DOWN COUNT PATTERN
D. RANDOM DATA PATTERN
E. CUMULATIVE SEQUENCE OF (A) THRU (D)
F. SINGLE CHARACTER PATTERN
G. TYPED IN BUFFER PATTERN

ALL ERRORS DETECTED ARE REPORTED AS THEY OCCUR AND A SWITCH REGISTER OPTION ALLOWS LOCKING ON A PARTICULAR PATTERN. THIS SUB-PROGRAM WOULD NORMALLY BE SELECTED FOR TROUBLESHOOTING A SPECIFIC PROBLEM.

1.1.4 CZDHND CORE MEMORY MAP



```
                  *********                   *********
                  * CONT. *                   * CONT. *
                  *********                   *********
                      *                           *
                      *****************************
        EM1:          *                           *
                      *    SYSMAC ERROR MESSAGE   *
                      *    BUFFERS                *
                      *                           *
                      *****************************
        TITLE:        *                           *
                      *    DH11 MISCELLANEOUS     *
                      *    MESSAGE BUFFERS        *
                      *                           *
                      *****************************
        RBUF:         *                           *
                      *    TRANSMIT AND RECEIVE   *
                      *    DATA BUFFERS           *
                      *                           *
                      *****************************
        ENBUFS:
```

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE REQUIREMENTS

A. ANY PDP11 COMPUTER SYSTEM WITH 12K OF CORE MEMORY
AND A CONSOLE TERMINAL DEVICE (VT50,LA36 ETC)

NOTE: FOR PAPER TAPE SYSTEMS USING THE PDP11 ABSOLUTE LOADER THE PROGRAM WILL LOAD AND RUN IN 8K OF CORE

B. A DH11 16. LINE ASYNCHRONOUS SERIAL LINE MULTIPLEXOR

C. A DH11 TERMINAL DEVICE (LA36,VT50 ETC.) [ECHO TESTS ONLY]

D. TEST CONNECTORS AND MODULE (THE NO. OF EACH REQUIRED IS DETERMINED BY THE PARTICULAR TEST APPLICATION. REFER TO SECTION 6.1 FOR A COMPLETE DISCUSSION OF THE MAINTENANCE CONNECTORS.)

1. H315 TEST CONNECTOR
2. H8611 TEST CONNECTOR
3. M974 TEST MODULE

1.2.2 SOFTWARE REQUIREMENTS

A. ACT11/APT11 — THE PROGRAM CONTAINS THE NECESSARY "SOFTWARE HOOKS" FOR INTERFACING TO THE ACT11/APT11 MANUFACTURING SYSTEMS. THE PROGRAM CAN BE RUN AS PART OF A QUICK VERIFY "CHAIN" SINCE IT CONTAINS AN AUTOSIZER.

B. XXDP — THE PROGRAM MAY BE LOADED FROM ANY "XXDP" MEDIA. IF AUTO-STARTED BY THE "XXDP" MONITOR CONTROL WILL BE TRANSFERRED TO THE DATA RELIABILITY PROGRAM.

1.3 RELATED DOCUMENTS AND STANDARDS

A. DH11-0 ENGINEERING DRAWINGS
B. DH11 MANUAL EK-DH11-MM-002
C. PDP11 PERIPHERALS HANDBOOK
D. PDP11 PROCESSOR HANDBOOK
E. MD-11-DZQAC-C1 SYSMAC.SML
F. MD-11-DZQXA "XXDP" USER'S GUIDE
G. DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS PROGRAMMING PRACTICES DOC NO. 175-003-009-00

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

CZDHND ASSUMES THAT THE FOLLOWING DIAGNOSTICS HAVE BEEN RUN PRIOR TO ITS EXECUTION AND THAT NO ERRORS WERE DETECTED:

A. CPU/CORE MEMORY DIAGNOSTICS
B. MD-11-DZDHM DH11 BASIC DIAGNOSTIC

1.5 FAILURE ASSUMPTIONS

CZDHND ASSUMES THAT THE DH11 HARDWARE VERIFIED BY MD-11-DZDHM, THE BASIC DH11 DIAGNOSTIC, IS FUNCTIONING ERROR FREE.

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURES

2.1.1 LOADING PROCEDURES

A. PAPER TAPE SYSTEMS

USE THE STANDARD PDP11 ABSOLUTE LOADER PROCEDURE FOR LOADING PAPER TAPES. AFTER LOADING THE PROGRAM MUST BE MANUALLY STARTED. (REFER TO SECTION 2.1.2)

B. "XXDP" SYSTEMS (REFER TO "XXDP" USER'S GUIDE MD-11-DZQXA)

1. MOUNT THE APPROPRIATE MEDIUM (DECTAPE,DISK ETC) CONTAINING THE "XXDP" MONITOR AND CZDHND.
2. BOOT THE SYSTEM TO LOAD THE MONITOR
3. ONCE LOADED THE "XXDP" MONITOR PRINTS AN INTRODUCTORY MESSAGE AND RESPONDS WITH A ".".
4. TYPE: "CZDHND" FOLLOWED BY EITHER A <CR> CARRIAGE RETURN OR AN "ALTMODE" TO LOAD THE PROGRAM.

IF A <CR> WAS TYPED THE USER MUST MANUALLY START THE PROGRAM AFTER LOADING.

IF AN "ALTMODE" WAS TYPED THE MONITOR WILL AUTO START THE PROGRAM AT LOCATION 000200(8) WHICH WILL BEGIN EXECUTION OF THE DATA RELIABILITY PROGRAM.

NOTE: WHENEVER THE DH11 CONFIGURATION IS CHANGED THE DIAGNOSTIC SHOULD BE RELOADED.

2.1.2 STARTING PROCEDURES

THERE ARE FIVE DIFFERENT STARTING ADDRESSES FOR THIS PROGRAM DEPENDING UPON WHICH SUB-PROGRAM IS TO BE STARTED. THERE ARE THREE FOR THE DATA RELIABILITY PROGRAM AND ONE EACH FOR THE ECHO AND DATA PATTERNS TESTS AS DESCRIBED BELOW:

2.1.2.1 SUB-PROGRAM 1 DH11 DATA RELIABILITY TESTS

A. TO AUTOMATICALLY START THE PROGRAM USING THE AUTOSIZER (START AT LOC 000200(8))

1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE PARTICULAR TEST APPLICATION (REFER TO SECTION 6.1)
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000200(8)
4. DEPRESS LOAD ADDRESS
5. SET THE SR=000200 (WORST CASE TESTING)

   SET THE SR=000002 (TO TYPE THE DEVICE MAP)

   SET THE SR=000000 (QUICK PASS)

   SET THE SR=000400 (HALT AFTER PARAMETER SET-UP)

6. SET THE HALT/ENABLE SWITCH TO ENABLE
7. DEPRESS START - THE PROGRAM WILL TEST ALL LINES ON ALL DH'S FOUND.

B. TO TYPE IN ALL REQUIRED PARAMETERS (START AT LOCATION 000200(8))

1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE PARTICULAR TEST APPLICATION. (REFER TO SECTION 6.1)
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000200(8)
4. DEPRESS LOAD ADDRESS
   SET THE SR=000001 (FOR INPUT DIALOGUE)
5. SET THE HALT/ENABLE SWITCH TO ENABLE
6. DEPRESS START
7. THE PROGRAM WILL PRINT THE TITLE AND ASK FOR THE NUMBER OF ADDRESSES BETWEEN VECTORS. TYPE EITHER A "10" OR A "20" TO INDICATE TEN OR TWENTY ADDRESS DISPLACEMENT BETWEEN VECTORS FOLLOWED BY A <CR> (CARRIAGE RETURN).

   NOTE: 1. SYSTEMS WHERE THE DM11-BB VECTORS ARE INTERLEAVED WITH THE DH11 VECTORS HAVE 20(8) ADDRESSES BETWEEN VECTORS. (THIS IS THE CASE FOR THE 2040 FRONT END)
   2. STANDARD SYSTEMS HAVE THE DH11 VECTORS CONTIGUOUS WITH A 10(8) ADDRESS DISPLACEMENT.

8. THE PROGRAM WILL ASK FOR THE DEVICE ADDRESS NEXT. TYPE IN THE ADDRESS (OCTAL) OF THE FIRST DH11 IN THE SYSTEM FOLLOWED BY A <CR>.

    IF AN INVALID ADDRESS IS TYPED THE PROGRAM WILL TYPE AN ERROR MESSAGE AND ASK YOU TO TRY AGAIN.

9. THE PROGRAM WILL ASK FOR THE VECTOR ADDRESS. TYPE IN THE VECTOR ADDRESS (OCTAL) OF THE FIRST DH11 FOLLOWED BY A <CR>.

    IF AN INVALID VECTOR ADDRESS IS TYPED THE PROGRAM WILL TYPE AN ERROR MESSAGE AND ASK YOU TO TRY AGAIN.

10. NEXT THE PROGRAM WILL ASK FOR THE DEVICE SELECTION PARAMETER. TYPE IN AN OCTAL NO. ENCODED AS FOLLOWS:

    BIT00=1 TEST DH11 #00
    BIT01=1 TEST DH11 #01
    BIT02-0 DO NOT TEST DH11 #02
    ''
    ''
    BIT15=1 TEST DH11 #15

    EXAMPLES:

    177777<CR> TEST ALL 16. DH11'S
    100000<CR> TEST ONLY DH11 #17(8)
    000005<CR> TEST DH11 #00 AND 02

11. NEXT THE PROGRAM WILL ASK FOR THE LINE SELECTION PARAMETERS. TYPE AN ENCODED OCTAL NO. AS FOLLOWS:

    BIT00=1 TEST LINE #00
    BIT01=1 TEST LINE #01
    BIT02=0 DO NOT TEST LINE #02
    ''
    ''
    BIT15=1 TEST LINE #15

    EXAMPLES:

    177777<CR> TEST ALL 16. LINES
    100000<CR> TEST LINE 17(8) ONLY
    000005<CR> TEST LINES 00 AND 02

    IF A <CR> RETURN ONLY IS TYPED THE PROGRAM WILL DEFAULT TO 16. LINES.

    ****
    NOTE
    ****

```
                        IF MORE THAN ONE DH11 IS TESTED THE SAME COMBINATION
                        OF LINES WILL BE TESTED ON ALL DH11'S SELECTED.

                    12. IF SR8=0 THE PROGRAM WILL BEGIN EXECUTION TESTING
                        THE FIRST SELECTED LINE OF THE FIRST SELECTED DH11.
                        (REFER TO PARA 2.4, 3.0, AND 4.0 FOR ERROR AND STATUS
                        REPORTS)

                    13. IF SR8=1 THE PROGRAM WILL HALT AND TYPE THE FOLLOW-
                        ING MESSAGE:

                            ''DEPRESS CONTINUE TO START TESTING''

                        WHEN CONTINUE IS DEPRESSED, THE PROGRAM WILL BEGIN TEST-
                        ING AS IN STEP 12.

                        THE PURPOSE OF THIS HALT IS TO ALLOW DUMPING
                        UPDATED VERSIONS OF THE PROGRAM AFTER THE PARAMETERS
                        HAVE BEEN SET UP FOR THE PARTICULAR DH11 SYSTEM.


            C. DEFAULT PARAMETERS ** (START AT LOC 000204(8))

                    1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE
                       PARTICULAR TEST APPLICATION.  REFER TO SECION 6.1.
                    2. SET THE HALT/ENABLE SWITCH TO HALT
                    3. SET THE SR=000204(8)
                    4. DEPRESS LOAD ADDRESS
                    5. SET THE SR=000200 (WORST CASE TESTING)

                       SET THE SR=000000 (QUICK PASS)

                    6. SET THE HALT/ENABLE SWITCH TO ENABLE
                    7. DEPRESS START

                    **      IF THIS IS THE INITIAL LOAD,
                            THE DEFAULT PARAMETERS ASSUME ONE DH11
                            WITH THE FOLLOWING ADDRESS ASSIGNMENTS

                            DH11 #0 DEVADR=760020, VECTOR=330, BR5

                            OTHERWISE, THE PROGRAM WILL DEFAULT TO THE
                            PARAMETERS USED IN THE PREVIOUS EXECUTION.

                    8. PROGRAM EXECUTION BEGINS. REFER TO SECTIONS 2.4, 3.0,
                       AND 4.0 FOR EXECUTION TIMES, ERROR REPORTS, AND
                       PROGRESS REPORTS.


            D. CHANGE DEVICE AND LINE SELECT PARAMETERS (START AT LOC 000210(8))

                    1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE
                       SPECIFIC TEST APPLICATION.  (REFER TO SECTION 6.1)
                    2. SET THE HALT/ENABLE SWITCH TO THE HALT POSITION
                    3. SET THE SR=000210
                    4. DEPRESS LOAD ADDRESS
```

```
                                    5. SET THE SR=000200 (WORST CASE TESTING)

                                       SET THE SR=000000 (QUICK PASS)

                                       SET THE SR=000400 (HALT AFTER PARAMETER SETUP)

                                    6. SET THE HALT/ENABLE SWITCH TO ENABLE
                                    7. DEPRESS START
                                    8. PROGRAM WILL ASK FOR DEVICE SELECTION PARAMETER
                                       PROCEED AS IN (B-10) ABOVE.
                                    9. PROGRAM WILL ASK FOR LINE SELECTION PARAMETERS.
                                       PROCEED AS IN (B-11) ABOVE.
                                    10. PROGRAM WILL BEGIN EXECUTION AS DESCRIBED IN
                                        PARA 2.1.2.1 (B,12) ABOVE
```

2.1.2.2 SUBPROGRAM 2 DH11 SINGLE LINE ECHO TESTS

1. CONNECT THE TEST TERMINAL TO THE DH11 LINE TO BE TESTED AND POWER IT UP.
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000214(8)
4. DEPRESS LOAD ADDRESS
5. SET THE SR=000001 (TO INHIBIT THE AUTOSIZER)

   NOTE: WHEN THE AUTOSIZER IS NOT INHIBITED THAT IS, SR=000000, IT WILL LOAD THE FIRST DH CSR ADDRESS AND VECTOR FOUND AND TEST THAT DEVICE ONLY. IF ANY OTHER DH'S ARE TO BE TESTED, THE CSR ADDRESS AND VECTOR MUST BE INPUT MANUALLY, THAT IS, WITH SR=000001.
6. SET THE HALT/ENABLE SW TO ENABLE
7. DEPRESS START
8. THE PROGRAM WILL TYPE THE TITLE MESSAGES AND ASK FOR THE NO. OF ADDRESSES BETWEEN VECTORS. TYPE EITHER A "10" OR "20" AS DESCRIBED IN PARA 2.1.2.1 (B7) ABOVE.

9. TYPE IN THE DEVICE ADDRESS (IN OCTAL) FOLLOWED BY A <CR>

   IF <CR> ONLY IS TYPED THE PROGRAM WILL USE A DEFAULT ADDRESS OF 760020(8)

   IF AN INVALID ADDRESS IS TYPED THE PROGRAM WILL RESPOND WITH AN ERROR MESSAGE AND ASK YOU TO TRY AGAIN.

10. NEXT THE PROGRAM WILL ASK FOR A VECTOR ADDRESS
11. TYPE IN THE VECTOR ADDRESS FOLLOWED BY A <CR>

    IF <CR> ONLY IS TYPED THE PROGRAM WILL USE A DEFAULT VECTOR ADDR OF 330(8)

    IF AN INVALID ADDRESS IS TYPED THE PROGRAM WILL TYPE AN ERROR MESSAGE AND ASK YOU TO TRY AGAIN.

12. NEXT THE PROGRAM WILL ASK FOR THE LINE NO. TO TEST
13. TYPE IN THE LINE NO. (IN OCTAL 00-17) FOLLOWED BY A <CR>

    IF A <CR> ONLY IS TYPED THE PROGRAM WILL DEFAULT TO LINE #00.

14. NEXT THE PROGRAM WILL ASK YOU IF YOU WANT TO CHANGE LINE PARAMETERS.
15. TYPE "Y" FOR YES - "N" OR <CR> FOR NO FOLLOWED BY A <CR>.

    IF "NO" THE PROGRAM WILL DEFAULT TO THE LAST LINE PARAMETERS TYPED IN OR IF THIS IS THE FIRST DIALOGUE IT WILL DEFAULT TO 9600 BAUD, 8 BIT CHARS, 1 STOP BIT, AND ODD PARITY.

16. IF YOU TYPED 'Y' IN (15) DO STEPS (17) THRU (21)
    OTHERWISE GO TO STEP (22)
17. WHEN THE PROGRAM ASKS FOR TRANSMITTER SPEED
    TYPE IN ONE OF THE 13. LEGAL SPEEDS IN DECIMAL
    FOLLOWED BY A <CR>.
18. WHEN THE PROGRAM ASKS FOR RECEIVER SPEED TYPE
    IN ONE OF THE 13. LEGAL SPEEDS IN DECIMAL
    FOLLOWED BY A <CR>.

        NOTE: FOR (17) AND (18) IF THE SPEED DESIRED
              IS 134.5, TYPE IT WITHOUT THE DECIMAL
              POINT.

              REFER TO PARA 5.2.3 FOR DESCRIPTION OF SPEED
              TABLES.

19. WHEN THE PROGRAM ASKS FOR CHAR LENGTH, TYPE
    IN THE NO. DESIRED FOLLOWED BY A <CR>
20. WHEN THE PROGRAM ASKS FOR THE NO. OF STOP BITS
    TYPE IN THE NO. DESIRED FOLLOWED BY A <CR>
21. WHEN THE PROGRAM ASKS FOR PARITY, TYPE IN:

        O   FOR ODD
        E   FOR EVEN
        <CR>   FOR NONE

22. THE PROGRAM WILL NEXT ASK FOR THE FILLER CHARACTER.
    TYPE IN THE FILLER CHAR FOLLOWED BY A <CR>

        IF A <CR> ONLY IS TYPED THE PROGRAM WILL USE
        A "NULL" FILLER WHICH IS THE NORMAL CASE.

23. THE PROGRAM WILL NEXT ASK FOR THE FILLER COUNT.
    TYPE IN THE COUNT IN OCTAL FOLLOWED BY A <CR>.

        IF A <CR> ONLY IS TYPED THE PROGRAM WILL DEFAULT
        TO ONE FILLER. IF A NO. GREATER THAN 4 BITS
        IS TYPED THE PROGRAM WILL TRUNCATE IT TO 4 BITS.
        THE MAXIMUM COUNT ALLOWED IS 15(10).

24. NEXT THE PROGRAM WILL ASK YOU IF YOU WANT SEND
    MODE. TYPE A 'Y' IF YES - 'N' OR "<CR>" IF NO.
25. IF YOU TYPED 'Y' IN RESPONSE TO (24) THE PROGRAM
    WILL ASK YOU TO TYPE IN THE SEND BUFFER ON THE
    CONSOLE TTY. WHEN YOU WANT TO SEND THIS BUFFER TO
    THE TEST TERMINAL ON THE DH11 TYPE A "CONTROL-C".

    NOTE: ALWAYS START THE BUFFER WITH A
          <CR><LF> TO MAKE IT EASIER TO
          INTERPRET THE DISPLAY ON THE DH11
          TERMINAL WHEN THE BUFFER IS SENT.

26. AFTER THE TEST BUFFER IS SENT THE PROGRAM WILL
    ASK FOR LINE # AGAIN AND YOU REPEAT THE SEQUENCE
    STARTING WITH STEP (12) ABOVE.

694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712

27. IF YOU TYPED SOME CHAR OTHER THAN ''Y'' IN RESPONSE TO (24) THE PROGRAM WILL ASSUME ''ECHO'' MODE AND ASK YOU TO TYPE IN ON THE TEST TERMINAL CONNECTED TO THE LINE UNDER TEST.
28. NOW GO TO THE TEST TERMINAL AND BEGIN TYPING. (IF THIS IS A REMOTE TERMINAL, ESTABLISH APPROPRIATE MODEM CONNECTION.) EACH CHAR TYPED SHOULD BE ECHOED ON THE DH11 TEST TERMINAL. IF YOU WANT TO ECHO AN ENTIRE BUFFER TYPE ''CONTROL-E'' AND THE PROGRAM WILL ECHO THE ENTIRE BUFFER TYPED IN ON THE TERMINAL TO THAT POINT.
29. TO CHANGE LINE # AND PARAMETERS - TYPE ''CONTROL-C'' ON THE DH11 TEST TERMINAL AND RETURN TO THE CONSOLE TERMINAL
30. TO TEST A DIFFERENT DH11 UNIT, THE PROGRAM MUST BE RESTARTED AT 000214(8).

2.1.2.3 SUBPROGRAM 3 DH11 SINGLE LINE DATA PATTERNS/CABLE TESTS

1. TERMINATE THE LINE TO BE TESTED WITH AN H315 TEST CONNECTOR.
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000220(8)
4. DEPRESS LOAD ADDRESS
5. SET THE SR=000001 (TO INHIBIT THE AUTOSIZER)

   NOTE: WHEN THE AUTOSIZER IS NOT INHIBITED, THAT IS SR=000000, IT WILL LOAD THE FIRST DH CSR ADDRESS AND VECTOR FOUND AND TEST THAT DEVICE ONLY. IF ANY OTHER DH'S ARE TO BE TESTED, THE CSR ADDRESS AND VECTOR MUST BE INPUT MANUALLY, THAT IS SR=000001.
6. SET THE HALT/ENABLE SW TO ENABLE
7. DEPRESS START
8. THE PROGRAM WILL TYPE THE TITLE MESSAGES AND ASK FOR THE NO. OF ADDRESSES BETWEEN VECTORS. TYPE EITHER A "10" OR "20" AS DESCRIBED IN PARA 2.1.2.1 (B 7).
9. NEXT THE PROGRAM WILL ASK FOR THE DH11 DEVICE ADDRESS

   TYPE IN THE DEVICE ADDRESS (IN OCTAL) FOLLOWED BY A <CR>

   IF <CR> ONLY IS TYPED THE PROGRAM WILL USE A DEFAULT ADDRESS OF 760020(8)

   IF AN INVALID ADDRESS IS TYPED THE PROGRAM WILL RESPOND WITH AN ERROR MESSAGE AND ASK YOU TO TRY AGAIN.
10. NEXT THE PROGRAM WILL ASK FOR A VECTOR ADDRESS
11. TYPE IN THE VECTOR ADDRESS FOLLOWED BY A <CR>

    IF <CR> ONLY IS TYPED THE PROGRAM WILL USE A DEFAULT VECTOR ADDR OF 330(8)

    IF AN INVALID ADDRESS IS TYPED THE PROGRAM WILL TYPE AN ERROR MESSAGE AND ASK YOU TO TRY AGAIN.
12. NEXT THE PROGRAM WILL ASK FOR THE LINE NO. TO TEST
13. TYPE IN THE LINE NO. (IN OCTAL 00-17) FOLLOWED BY A <CR>

    IF A <CR> ONLY IS TYPED THE PROGRAM WILL DEFAULT TO LINE #00.
14. NEXT THE PROGRAM WILL ASK YOU IF YOU WANT TO CHANGE LINE PARAMETERS.
15. TYPE "Y" IF YOU DO - "N" OR <CR> IF YOU DON'T

    IF "NO" THE PROGRAM WILL DEFAULT TO THE LAST LINE PARAMETERS TYPED IN OR IF THIS IS THE FIRST DIALOGUE IT WILL DEFAULT TO 9600 BAUD.

8 BIT CHARS, 1 STOP BIT, AND ODD PARITY.

16. IF YOU TYPED ''Y'' IN (15) DO STEPS (17) THRU (21)
    OTHERWISE GO TO STEP (22)
17. WHEN THE PROGRAM ASKS FOR TRANSMITTER SPEED
    TYPE IN ONE OF THE 13. LEGAL SPEEDS IN DECIMAL
    FOLLOWED BY A <CR>.
18. WHEN THE PROGRAM ASKS FOR RECEIVER SPEED TYPE
    IN ONE OF THE 13. LEGAL SPEEDS IN DECIMAL
    FOLLOWED BY A <CR>.

        NOTE: FOR (17) AND (18) IF THE SPEED DESIRED
               IS 134.5, TYPE IT WITHOUT THE DECIMAL
               POINT.

19. WHEN THE PROGRAM ASKS FOR CHAR LENGTH, TYPE
    IN THE NO. DESIRED FOLLOWED BY A <CR>
20. WHEN THE PROGRAM ASKS FOR THE NO. OF STOP BITS
    TYPE IN THE NO. DESIRED FOLLOWED BY A <CR>
21. WHEN THE PROGRAM ASKS FOR PARITY, TYPE IN:

        O   FOR ODD
        E   FOR EVEN
        <CR>   FOR NONE

    FOLLOWED BY A <CR>
22. THE PROGRAM WILL NEXT ASK FOR THE FILLER CHARACTER.
    TYPE IN THE FILLER CHAR FOLLOWED BY A <CR>

        IF A <CR> ONLY IS TYPED THE PROGRAM WILL USE
        A ''NULL'' FILLER WHICH IS THE NORMAL CASE.

23. THE PROGRAM WILL NEXT ASK FOR THE FILLER COUNT.
    TYPE IN THE COUNT IN OCTAL FOLLOWED BY A <CR>.

        IF A <CR> ONLY IS TYPED THE PROGRAM WILL DEFAULT
        TO ONE FILLER. IF A NO. GREATER THAN 4 BITS
        IS TYPED THE PROGRAM WILL TRUNCATE IT TO 4 BITS.
        THE MAXIMUM COUNT ALLOWED IS 15.

24. NEXT THE PROGRAM WILL ASK YOU THE BUFFER SIZE.
    TYPE IN A DECIMAL NO. BETWEEN 1 TO 512. FOLLOWED
    BY A <CR>.

        IF A <CR> ONLY IS TYPED THE PROGRAM WILL DEFAULT
        TO A BUFFER SIZE OF 256. BYTES.

        IF THE NO. TYPED IS TOO LARGE AN ERROR MESSAGE
        IS TYPED AND YOU ARE ASKED TO TRY AGAIN.

25. NEXT THE PROGRAM WILL ASK FOR THE TYPE OF PATTERN
    AND TELL YOU TO SET SR07=1 IF YOU WANT TO LOCK ON
    THE SELECTED PATTERN.
26. TYPE (A,U,D,R,B,S, OR <CR>) TO SELECT THE PATTERN
    AS DESCRIBED BELOW:

```
                                        A       ALTERNATING 1/0
                                        U       BINARY UP COUNT
                                        D       BINARY DOWN COUNT
                                        R       RANDOM DATA
                                        B       TYPED IN BUFFER
                                        S       SINGLE CHARACTER
                                        <CR>    SEQUENCE OF A,U,D, AND R

                                27. IF YOU TYPED A,U,D,R, OR <CR>, THE PROGRAM WILL
                                    TRANSMIT, RECEIVE, AND DATA CHECK THE SELECTED PATTERN.

                                    SR07=1      IT WILL LOCK ON THIS PATTERN


                                    SR07 0      IT WILL RETURN TO STEP (24)
                                                AFTER COMPLETING THE TEST OF THIS
                                                PATTERN AND ASK FOR A NEW PATTERN.

                                28. IF YOU TYPED A 'B'' IN (26) THE PROGRAM WILL ASK
                                    YOU TO TYPE IN A TEST PATTERN AND TERMINATE IT WITH
                                    A ''CONTROL-C''. WHEN THE PROGRAM SENSES THE TERMIN-
                                    ATOR IT WILL BEGIN EXECUTION AS IN (27) USING THE
                                    THE TYPED IN BUFFER AS THE PATTERN.
                                29. IF YOU TYPED AN ''S'' IN RESPONSE TO (26) THE PROGRAM
                                    WILL ASK FOR A SINGLE CHAR. TYPE A SINGLE CHAR
                                    FOLLOWED BY A <CR>. THE PROGRAM WILL FILL THE BUFFER
                                    WITH THE TYPED IN CHAR AND BEGIN EXECUTION AS IN (27)
                                    USING THE BUFFER FULL OF THE TEST CHAR AS A PATTERN.
                                30. TO CHANGE DH11'S, LINE PARAMETERS ETC. YOU MUST
                                    RESTART THE TESTS AT LOC. 000220(8).

                    2.1.3   RESTART PROCEDURES
                            ------------------

                                    SAME AS THE STARTING PROCEDURES
```

2.2 SPECIAL ENVIRONMENTS

2.2.1 ACT11/APT11

THE PROGRAM MAY BE LOADED BY THE ACT11/APT11 SYSTEMS, AND MAY BE RUN AS PART OF A QUICK VERIFY CHAIN SINCE THE PROGRAM CONTAINS AN AUTOSIZER.

2.2.2 XXDP

THE PROGRAM MAY BE LOADED AND RUN FROM ANY ''XXDP'' MEDIUM PROVIDED THERE IS AT LEAST 12K OF CORE. IT MAY BE RUN AS PART OF AN ''XXDP'' CHAIN.

2.2.3 SWITCHLESS FEATURE

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH REGISTER (LOC. 176) IS USED.

CONTROL:

THIS PROGRAM ALSO SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (LOC. 176) FROM THE TTY. THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

1) TYPE CONTROL G <^G>; THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.

2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)

3) AFTER THE ''NEW='' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:

   A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY OCTAL NUMBERS WILL BE ACCEPTED AND ONLY 6 NUMBERS WILL BE ALLOWED)
   IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.

   B) IF A CONTROL U <^U> IS DEPRESSED THEN THE PROGRAM WILL DO A <CR>. RETYPE THE DESIRED NUMBER.

2.3 PROGRAM OPTIONS

2.3.1 CONSOLE SWITCH REGISTER

A. SUB-PROGRAM 1 — DH11 DATA RELIABILITY TESTS

SR15=1	HALT ON ERROR
SR14=1	LOOP ON CURRENTLY SELECTED DH11
SR13=1	INHIBIT ERROR, PROGRESS, AND PERFORMANCE PRINTOUTS
SR8=1	HALTS AFTER CONFIGURATION TO PERMIT DUMPING PRECONFIGURED COPIES OF THE PROGRAM.
SR7=1	PERFORMS A STANDARD PASS(NOT QUICK VERIFY.)
SR7=0	QUICK VERIFY - DO COMPLETE TESTING ON EACH LINE AT 9600. BAUD ONLY
SR1=1	TYPES DEVICE MAP GENERATED BY THE AUTOSIZER.

```
                        SR0=1           ALLOWS THE USER TO INPUT DH PARAMETERS
                                        MANUALLY.  (INHIBITS THE AUTOSIZER.)

                    B. SUB-PROGRAM 2    DH11 SINGLE LINE ECHO TESTS

                        (NONE)

                    C. SUB-PROGRAM 3    DH11 SINGLE LINE PATTERNS/CABLE TESTS

                        SR15=1          HALT ON ERROR
                        SR13-1          INHIBIT ERROR AND STATUS PRINTOUTS
                        SR07-1          LOOP ON CURRENT TEST PATTERN


2.3.2   CORE MEMORY LOCATIONS
        ---------------------

        A. SUB-PROGRAM 1    DH11 DATA RELIABILITY TESTS

            WHEN THE AUTOSIZER OPTION IS USED, THIS PROGRAM
        CAN RUN NON-STANDARD DH11 CONFIGURATIONS (NON-CONTIGUOUS
        ADDRESSES).  THE USER CAN ALSO PATCH IN HIS OWN ADDRESSES TO MATCH
        HIS CONFIGURATION AND THEN USE THE DEFAULT START TO RUN THE
        UPDATED PROGRAM.  THE TABLES AND LOCATIONS TO MODIFY ARE DES-
        CRIBED BELOW:

            1) DEVICE ADDRESS TABLE

                THERE IS A 16. WORD TABLE STARTING AT THE ADDRESS
            TAGGED 'DHADTB:' THAT IS PROGRAM LOADED TO SPECIFY 16.
            CONTIGUOUS DH11'S STARTING AT THE BUS ADDRESS 160020(8).
            THIS TABLE IS MODIFIED AT CONFIGURATION TIME IF THE
            USER TYPES IN A DIFFERENT STARTING ADDRESS, OR IT MAY
            BE PATCHED TO REFLECT ANY UNIQUE DH11 SYSTEM CONFIGURATION.

            2) VECTOR ADDRESS TABLE

                THERE IS A 16. WORD TABLE STARTING AT THE ADD-
            RESS TAGGED 'DHVCTB:' THAT IS PROGRAM LOADED TO
            SPECIFY 16. CONTIGUOUS VECTORS STARTING WITH 330(8)
            AND EACH ENTRY DISPLACED BY 8. WORDS (330,350,370, ETC.)
            THIS TABLE IS MODIFIED AT CONFIGURATION TIME IF THE USER
            TYPES A DIFFERENT STARTING VECTOR ADDRESS, OR IT
            MAY BE PATCHED TO REFLECT ANY UNIQUE DH11 SYSTEM CON-
            FIGURATION.

            3) BR LEVEL TABLE

                THERE IS A 16. WORD TABLE STARTING AT THE ADDRESS
            TAGGED 'BRLVL:' THAT IS PROGRAM LOADED TO CONTAIN A
            120240(8) IN EACH ENTRY WHICH SPECIFIES BR LEVEL 5
            FOR BOTH XMIT AND RECEIVE FOR ALL 16. DH11'S. IT IS NOT
            CHANGED AT CONFIGURATION TIME BUT MAY BE PATCHED TO
            REFLECT ANY UNIQUE DH11 SYSTEM CONFIGURATION.

            4) DEVICE SELECTION PARAMETER
```

THERE IS A WORD TAGGED 'DHSEL:' THAT IS PROGRAM LOADED TO CONTAIN A 000001(8) WHICH SELECTS ONLY ONE DH11 (DH11 #00). THIS LOCATION CAN BE MODIFIED AT CONFIGURATION TIME TO SPECIFY ANY COMBINATION OF DH11'S UP TO A MAXIMUM OF 16. UNITS. REFER TO PARA 2.1.3.1 (B 10) FOR A DESCRIPTION OF ITS ENCODING.

5) LINE SELECTION PARAMETER (PARA 2.1.2.1 (B11))

THERE IS A WORD TAGGED 'LINSEL:' THAT IS PROGRAM LOADED AS 177777(8) TO SPECIFY ALL 16. LINES ARE TO BE TESTED IN EACH SELECTED DH11. IT MAY BE MODIFIED AT CONFIGURATION TIME TO SPECIFY ANY COMBINATION OF LINES TO TEST. REFER TO SECTION 2.1.2 (B 11) FOR A DESCRIPTION OF ITS ENCODING.

NOTE: THE DATA RELIABILITY PROGRAM IS TABLE DRIVEN IN THAT IT USES 'DHSEL:', 'LINSEL:' AND THE CONTENTS OF THE THREE 16. WORD TABLES TO DEFINE THE DH11 CONFIGURATION TO BE TESTED.

B. SUB-PROGRAM 2 DH11 SINGLE LINE ECHO TESTS

(NONE)

C. SUB-PROGRAM 3 DH11 SINGLE LINE PATTERNS/CABLE TESTS

1)PATLIM: 10.

THERE IS A LOCATION TAGGED 'PATLIM:' THAT SPECIFIES THE NO. OF TEST PATTERN ITERATIONS TO EXECUTE IN THE PATTERNS TESTS. IT IS PROGRAM LOADED TO SPECIFY TEN ITERATIONS BEFORE THE "TEST DONE" REPORT IS TYPED.

2) DATCNT:

THERE IS A LOCATION TAGGED 'DATCNT:' THAT KEEPS A COUNT OF THE NO. OF ITERATIONS COMPLETED DURING THE PATTERNS TESTS. THIS INFORMATION GETS TYPED OUT AS PART OF THE ERROR MESSAGE IF A DATA ERROR OCCURS IN THE PATTERNS TEST UNDER THE HEADING "ICOUNT".

2.4 EXECUTION TIMES

A. SUB-PROGRAM 1 — DH11 DATA RELIABILITY TESTS

1. SR07=0 — QUICK TEST

   APPROXIMATELY 15. SECONDS FOR EACH LINE WITH 1824 CHARS BEING TRANSMITTED AND RECEIVED.

2. SR7=1 — COMPLETE TESTING

   APPROXIMATELY 15. MINUTES FOR EACH LINE WITH 18,720 CHARS BEING TRANSMITTED AND RECEIVED ON EACH LINE SELECTED FOR TEST.

B. SUB-PROGRAM 2 — DH11 SINGLE LINE ECHO TESTS

NOT APPLICABLE SINCE THESE TESTS INVOLVE THE USER MANUALLY TYPING IN ON THE TERMINAL.

C. SUB-PROGRAM 3 — DH11 SINGLE LINE PATTERNS/CABLE TESTS

EXECUTION TIMES VARY FROM LESS THAN 5 SECONDS TO GREATER THAN 15. MINUTES DEPENDING UPON BUFFER SIZE, LINE PARAMETERS, AND PATTERN SELECTED.

```
1065                                3.0     ERROR INFORMATION
1066                                        -----------------
1067
1068                                3.1     ERROR REPORTING PROCEDURES
1069                                        --------------------------
1070
1071                                3.1.1   STANDARD SYSMAC.SML ERROR REPORTING CONVENTIONS
1072                                        ----------------------------------------------
1073
1074                                        THE PROGRAM UTILIZES THE STANDARD PDP11 DIAGNOSTICS ERROR
1075                                        UTILITIES. THE TEST ROUTINE CALLS THESE UTILITIES USING AN
1076                                        'ERROR  N'' INSTRUCTION (CODED EMT) WHERE 'N'' IS THE NUMBER
1077                                        OF THE ERROR MESSAGE. THE UTILITY ROUTINE USES 'N'' TO
1078                                        ACCESS THE PROPER ERROR INFORMATION VIA THE ERROR TABLE
1079                                        DESCRIBED IN SECTION 3.1.2 BELOW. EACH MESSAGE RESULTS
1080                                        IN THREE LINES OF TYPEOUT AS FOLLOWS:
1081
1082                                        LINE 1  A BRIEF DESCRIPTION OF THE FAILING FUNCTION
1083                                        LINE 2  LABELS TO IDENTIFY THE DATA TYPED ON LINE 3
1084                                        LINE 3  THE ACTUAL ERROR DATA (UP TO 8 OCTAL OR DECIMAL NO.S)
1085
1086                                        EXAMPLE:
1087
1088                                        SYSTEM CONTROL REGISTER ERROR
1089                                         (PC)    (PS)    (SP)    TEST   DEVADR  REGADR   WAS     S/B
1090                                        002720  000002  001074  000003  160020  160020  000000  000001
1091
1092                                        THE ERROR TABLE ITEMS SHOWN IN THE NEXT SECTION DESCRIBE ALL
1093                                        THE ERROR MESSAGES WITHIN CZDHND AND ARE INTERPRETED
1094                                        AS FOLLOWS:
1095
1096                                        EM      ADDRESS OF THE MESSAGE FOR LINE 1
1097                                        DH      ADDRESS OF THE DATA HEADER MESSAGE FOR LINE 2
1098                                        DT      ADDRESS OF THE TABLE OF ADDRESSES THAT POINT
1099                                                TO THE DATA WORDS TO BE PRINTED
1100                                        DF      ADDRESS THAT POINTS TO THE DATA DESCRIPTOR
1101                                                TABLE THAT DEFINES WHETHER AN ITEM IS OCTAL OR DECIMAL.
1102                                                IF THIS ENTRY IS '0'' ALL DATA WORDS ARE IN OCTAL.
1103
1104                                        SECTION 3.1.3 DEFINES THE MEANING OF THE MNEUMONICS USED
1105                                        IN THE VARIOUS DATA HEADERS.
```

```
                        3.1.2   ERROR MESSAGE TABLE
                                ------------------



                        ;ERROR TABLE ITEM FOR ERROR 1

                                EM1             ;'NON EX MEMORY ERROR - DROPPED LINE #  ''
                                DH1             ;'' (PC)    CURLPR  DEVADR  REGADR   WAS     S/B''
                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                DF2             ;PRINT ALL OCTAL

                        ;ERROR TABLE ITEM FOR ERROR 2

                                EM2             ;''TRANSMITTER FALSE INTERRUPT - DROPPED LINE#  ''
                                DH1             ;'' (PC)    CURLPR  DEVADR  REGADR   WAS     S/B''
                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                DF2             ;PRINT ALL OCTAL

                        ;ERROR TABLE ITEM FOR ERROR 3

                                EM3             ;'BUFFER ACTIVE REGISTER ERROR - DROPPED LINE #  ''
                                DH1             ;'' (PC)    CURLPR  DEVADR  REGADR   WAS     S/B''
                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                DF2             ;PRINT ALL OCTAL

                        ;ERROR TABLE ITEM FOR ERROR 4

                                EM4             ;'BYTE COUNT REGISTER ERROR - DROPPED LINE #  ''
                                DH1             ;'' (PC)    CURLPR  DEVADR  REGADR   WAS     S/B''
                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                DF2             ;PRINT ALL OCTAL

                        ;ERROR TABLE ITEM FOR ERROR 5

                                EM5             ;''CURRENT ADDRESS REGISTER ERROR - DROPPED LINE #  ''
                                DH1             ;'' (PC)    CURLPR  DEVADR  REGADR   WAS     S/B''
                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                DF2             ;PRINT ALL OCTAL

                        ;ERROR TABLE ITEM FOR ERROR 6

                                EM6             ;''SILO OVERFLOW ERROR - DROPPED LINE #  ''
                                DH1             ;'' (PC)    CURLPR  DEVADR  REGADR   WAS     S/B''
                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                DF2             ;PRINT ALL OCTAL

                        ;ERROR TABLE ITEM FOR ERROR 7

                                EM7             ;''RECEIVER FALSE INTERRUPT - LINE #  ''
                                DH1             ;'' (PC)    CURLPR  DEVADR  REGADR   WAS     S/B''
                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                DF2             ;PRINT ALL OCTAL

                        ;ERROR TABLE ITEM FOR ERROR 10
```

```
1162                                    EM10            ;''INVALID DATA IN SILO - DROPPED LINE #  ''
1163                                    DH2             ;  (PC)    CURLPR   CHAR #  WASADR   SHBADR    WAS      S/B''
1164                                    DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
1165                                    DF2             ;PRINT ALL OCTAL
1166
1167                            ;ERROR TABLE ITEM FOR ERROR 11
1168
1169                                    EM11            ;'DATA ERROR - LINE #  ''
1170                                    DH2             ;  (PC)    CURLPR   CHAR #  WASADR   SHBADR    WAS      S/B''
1171                                    DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
1172                                    DF2             ;PRINT ALL OCTAL
1173
1174                            ;ERROR TABLE ITEM FOR ERROR 12
1175
1176                                    EM12            ;'TEST TIMEOUT - DROPPED LINE #  ''
1177                                    DH3             ;'' (PC)    CURLPR   RTOTAL   XTOTAL   RDONE''
1178                                    DT3             ;'$ERRPC,CURLPR,$TMP0,$TMP1RDONE''
1179                                    DF2             ;PRINT ALL OCTAL
1180
1181                            ;ERROR TABLE ITEM FOR ERROR 13
1182
1183                                    NOTE:   ERROR 13 IS CALLED TO PRINT EACH LINE OF DATA IN THE
1184                                            ERROR STATISTICS TABLE. IT PRINTS ONLY DATA WITHOUT ANY
1185                                            MESSAGE OR DATA HEADERS.
1186
1187
1188                                    0               ;NO MESSAGE
1189                                    0               ;NO DATA HEADER
1190                                    DT4             ;$TMP0,$TMP1,$TMP2,$TMP3,$TMP4,$TMP5,$TMP6
1191                                    DF1             ;PRINT ALL DECIMAL
1192
1193                            ;ERROR TABLE ITEM FOR ERROR 14
1194
1195                                    EM14            ;'BUS ERROR TRAP TO 04''
1196                                    DH4             ;'' (PC)     (PS)     (SP)    TRAPPC   TRAPPS''
1197                                    DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2''
1198                                    DF2             ;PRINT ALL OCTAL
1199
1200                            ;ERROR TABLE ITEM FOR ERROR 15
1201
1202                                    EM15            ;'RSVD INSTR TRAP TO 10''
1203                                    DH4             ;'' (PC)     (PS)     (SP)    TRAPPC   TRAPPS''
1204                                    DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2''
1205                                    DF2             ;PRINT ALL OCTAL
1206
1207                            ;ERROR TABLE ITEM FOR ERROR 16
1208
1209                                    EM16            ;'SINGLE LINE ECHO TEST - INTR WAIT TIMEOUT''
1210                                    DH5             ;'' (PC)    DEVADR   LINE     (SCR)  CURLPR   EXFLAG''
1211                                    DT6             ;$ERRPC,$REG1,LINE,$TMP0,CURLPR,EXFLAG''
1212                                    DF2             ;PRINT ALL OCTAL
1213
```

```
1214
1215                            NOTE:   ERRORS 17 THRU 24 ARE USED TO REPORT PERFORMANCE
1216                                    NOT ERRORS.
1217
1218                    ;ERROR TABLE ITEM FOR ERROR 17
1219
1220                            EM17            ;''ALTERNATING 1/0 PATTERN TEST DONE''
1221                            DH6             ;'' (PC)    DEVADR    LINE    CURLPR   ICOUNT''
1222                            DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
1223                            DF2             ;PRINT ALL OCTAL
1224
1225                    ;ERROR TABLE ITEM FOR ERROR 20
1226
1227                            EM20            ;''BINARY UP COUNT PATTERN TEST DONE''
1228                            DH6             ;'' (PC)    DEVADR    LINE    CURLPR   ICOUNT''
1229                            DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
1230                            DF2             ;PRINT ALL OCTAL
1231
1232                    ;ERROR TABLE ITEM FOR ERROR 21
1233
1234                            EM21            ;''BINARY DOWN COUNT PATTERN TEST DONE''
1235                            DH6             ;'' (PC)    DEVADR    LINE    CURLPR   ICOUNT''
1236                            DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
1237                            DF2             ;PRINT ALL OCTAL
1238
1239                    ;ERROR TABLE ITEM FOR ERROR 22
1240
1241                            EM22            ;''RANDOM DATA PATTERN TEST DONE''
1242                            DH6             ;'' (PC)    DEVADR    LINE    CURLPR   ICOUNT''
1243                            DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
1244                            DF2             ;PRINT ALL OCTAL
1245
1246                    ;ERROR TABLE ITEM FOR ERROR 23
1247
1248                            EM23            ;SINGLE CHAR PATTERN TEST DONE''
1249                            DH6             ;'' (PC)    DEVADR    LINE    CURLPR   ICOUNT''
1250                            DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
1251                            DF2             ;PRINT ALL OCTAL
1252
1253                    ;ERROR TABLE ITEM FOR ERROR 24
1254
1255                            EM24            ;''TYPED BUFFER PATTERN TEST DONE''
1256                            DH6             ;'' (PC)    DEVADR    LINE    CURLPR   ICOUNT''
1257                            DT7             ;$ERRORPC,DHADR,LINE,CURLPR,$REGO
1258                            DF2             ;PRINT ALL OCTAL
1259
1260                    ;ERROR TABLE ITEM FOR ERROR 25
1261
1262                            EM25            ;''DATA PATTERNS TEST TIMEOUT''
1263                            DH7             ;'' (PC) DEVADR   LINE    CURLPR  ICOUNT  PATCDE''
1264                            DT10            ;$ERRPC,DHADR,LINE,CURLPR,$REGO,$REG1
1265                            DF2             ;PRINT ALL OCTAL
1266
1267
```

3.1.3 DATA HEADER MNEUMONIC DEFINITIONS

ALL NUMBERS PRINTED AS ERROR DATA ARE IN OCTAL

(PC) ADDRESS OF THE ERROR CALL (ERROR PC)

(PS) CONTENTS OF THE PSW AT THE TIME OF THE ERROR

(SP) CONTENTS OF THE STACK POINTER AT THE TIME OF THE ERROR

LINE INDICATES THE LINE NUMBER THAT FAILED

DEVADR DEVICE ADDRESS - 1ST ADDRESS IN THE SELECTED DH11

REGADR ADDRESS OF THE DH11 REGISTER BEING TESTED

WAS WHAT THE ACTUAL DATA READ WAS (DH11 REG OR CORE LOC.)

S/B WHAT THE DATA READ SHOULD HAVE BEEN

TRPPC CONTENTS OF THE PC (R7) AT THE TIME OF A BUS ERROR
OR RSVD INSTR TRAP.

TRPPS CONTENTS OF THE PSW AT THE TIME OF A BUS ERROR
OR RSVD INSTR TRAP.

WASADR CORE MEMORY ADDRESS OF THE "WAS" DATA (ACTUAL DATA READ)

SBADR CORE MEMORY ADDRESS OF THE S/B DATA (GOOD DATA)

CHAR # INDICATES THE CHARACTER POSITION IN THE DATA BUFFER

ICOUNT INDICATES ITERATION COUNT OF DATA PATTERNS TESTS -
PROGRAM DEFAULTS TO ITERATING EACH PATTERN 10. TIMES.

PATCDE INDICATES PATTERN BEING TESTED WHEN ERROR OCCURRED
AS SHOWN BELOW:

PATCDE=	101	ALTERNATING 1/0 PATTERN
	125	BINARY UP COUNT PATTERN
	104	BINARY DOWN COUNT
	122	RANDOM DATA PATTERN
	123	SINGLE CHAR PATTERN
	102	TYPED IN BUFFER PATTERN

(SCR) INDICATES CONTENTS OF THE "SCR" REG WHEN ERROR OCCURRED

EXFLAG INDICATES STATE OF THE XMITTER INTR SERVICE ROUTINE
IN THE ECHO TESTS AS SHOWN BELOW:

EXFLAG=	1	CONTROL-C WAS TYPED
	2	CONTROL-E WAS TYPED
	3	BUFFER WAS BEING DUMPED

3.2 POWER FAIL PRINTOUT

IF A POWER FAILURE OCCURS WHILE THE PROGRAM IS RUNNING, THE FOLLOWING PRINTOUT OCCURS:

'?POWER''

AFTER THE PRINTOUT THE PROGRAM WILL BE RESTARTED AUTOMATICALLY FROM THE BEGINNING. NO ATTEMPT IS MADE TO CONTINUE THE PROGRAM FROM THE POINT OF THE POWER FAIL INTERRUPTION.

3.3 ERROR HALTS

A. SYSMAC ERROR SERVICE ROUTINE HALT

WHEN SR15=1 A '?HALT'' IS EXECUTED IN THE SYSMAC ERROR UTILITY AFTER THE ERROR TYPEOUT. TO RESUME TESTING FROM THE POINT OF THE '?HALT'' SIMPLY DEPRESS CONTINUE.

B. POWER FAIL HALT

WHEN A POWER DOWN IS DETECTED, THE PROGRAM HALTS IN THE POWER FAIL UTILITY ROUTINE. IF FOR SOME REASON THE AUTO-START FEATURE FAILS TO RESTART THE PROGRAM, THE PROGRAM WILL ''LOCK'' ON THIS HALT IF CONTINUE IS DEPRESSED. IN THIS CASE THE PROGRAM MUST BE RESTARTED.

C. TRAP CATCHER HALTS

ALL INACTIVE VECTORS ARE SET UP WITH THE STANDARD PDP11 TRAP CATCHER AS DESCRIBED BELOW:

```
VN   / VN+2
VN+2/ HALT
```

IF A TRAP OR INTERRUPT OCCURS TO A VECTOR THAT HAS NOT BEEN SET UP BY THE TEST ROUTINE, A '?HALT'' OCCURS IN THE VECTOR AREA. THE ADDRESS DISPLAY INDICATES WHICH VECTOR THE PROGRAM TRAPPED TO AND THE LAST ENTRY PUSHED ON TO THE STACK INDICATES WHERE THE PROGRAM WAS WHEN THE TRAP OR INTERRUPT OCCURRED.

4.0 PERFORMANCE AND PROGRESS REPORTS

4.1 PERFORMANCE REPORTS

A. SUB-PROGRAM 1 DH11 DATA RELIABILITY TESTS

AT THE COMPLETION OF EACH PASS THROUGH EACH DH11 UNDER TEST, OR WHEN INVOKED BY THE USER TYPING AN ''S'' ON THE CONSOLE TERMINAL, THE STATISTICS TABLE SHOWN BELOW IS TYPED ON THE CONSOLE DEVICE. THE TABLE CONSISTS OF 'NN'' ENTRIES WHERE 'NN'' IS THE NUMBER OF LINES SELECTED FOR TEST.

DH #NN STATISTICS.

LINE #	RTOTAL	XTOTAL	DATERR	PARERR	FRMERR	OVRERR
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000
LLLLLL	RRRRRR	XXXXXX	DDDDDD	PPPPPP	FFFFFF	000000

WHERE:
- N IS THE DH11 # IN OCTAL
- L IS THE LINE # IN OCTAL
- R IS THE TOTAL NO. OF CHARS RECEIVED IN DECIMAL
- X IS THE TOTAL NO. OF CHARS TRANSMITTED IN DECIMAL
- D IS THE TOTAL NO. OF DATA COMPARE ERRORS DETECTED IN DECIMAL
- P IS THE TOTAL NO. OF PARITY ERRORS IN DECIMAL

```
1424
1425
1426
```

F IS THE TOTAL NO. OF FRAMING ERRORS IN DECIMAL
O IS THE TOTAL NO. OF OVERRUN ERRORS IN DECIMAL

```
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
```

NOTES: 1.) IF A LINE WAS DROPPED DURING THE TEST DUE TO AN UNRECOVERABLE READ OR WRITE ERROR THE MESSAGE SHOWN BELOW WILL REPLACE THE NORMAL ERROR STATISTICS ENTRY:

''LINE #NN WAS DROPPED''

WHERE ''NN'' IS THE LINE NO. IN OCTAL.

2.) IF THE PRINTOUT IS INVOKED BY TYPING AN ''S'', THE ''RTOTAL'' AND ''XTOTAL'' ENTRIES MAY OR MAY NOT BE EQUAL DEPENDING UPON WHEN THE PROGRAM ''SAW'' THE ''S''.

3.) AFTER PRINTING THE ERROR STATISTICS TABLE, THE PROGRAM WILL RESTART AND BEGIN TESTING THE NEXT DH11 IN SEQUENCE. IF ONLY ONE DH11 IS SELECTED FOR TEST OR SR14=1, THE SAME DH11 WILL BE TESTED AGAIN.

B. SUB-PROGRAM 2 — DH11 SINGLE LINE ECHO TESTS

1) SEND MODE: THE DISPLAY ON THE DH11 TEST TERMINAL SHOULD MATCH THE BUFFER TYPED IN ON THE CONSOLE TERMINAL.

2) ECHO MODE: THE CHARACTERS ECHOED ON THE DH11 TEST TERMINAL SHOULD MATCH THE CHARACTERS TYPED ON THE TEST TERMINAL KEYBOARD.

C. SUB-PROGRAM 3 — DH11 SINGLE LINE PATTERNS/CABLE TESTS

(NONE PROVIDED)

4.2 PROGRESS REPORTS

A. SUB-PROGRAM 1 DH11 DATA RELIABILITY TESTS

1. EACH TIME A NEW DH11 IS SELECTED FOR TEST THE PROGRAM TYPES:

   ''TESTING DH11 #NN''

   WHERE 'NN'' IS THE NO. IN OCTAL OF THE DH11 CURRENTLY BEING TESTED. (00 - 17)

2. EACH TIME A NEW LINE IS SELECTED FOR TEST THE PROGRAM TYPES.

   ''TESTING LINE #NN''

   WHERE 'NN'' IS THE LINE NO. IN OCTAL (00 - 17)

3. AFTER COMPLETE TESTING OF ALL SELECTED DH11'S THE FOLLOWING MESSAGE IS PRINTED:

   'END PASS #NNNNNN''

   WHERE: N IS THE NO. OF COMPLETE PROGRAM PASSES DURING THE CURRENT 'RUN''

B. SUB-PROGRAM 2 DH11 SINGLE LINE ECHO TESTS

(NONE SUPPLIED)

C. SUB-PROGRAM 3 DH11 SINGLE LINE PATTERNS/CABLE TESTS

EACH TIME A SPECIFIC TEST PATTERN TEST IS COMPLETED (10. ITERATIONS) THE FOLLOWING MESSAGE IS TYPED:

```
'NAME'' PATTERN TEST DONE
 (PC)   DEVADR   LINE   CURLPR  ICOUNT
PPPPPP  DDDDDD  LLLLLL  CCCCCC  IIIIII
```

WHERE:
- NAME — IS THE NAME OF THE PATTERN - IE 'RANDOM'', 'BINARY UP COUNT'', ETC
- P — IS THE PC OF THE MESSAGE CALL
- D — IS THE ADDRESS OF THE DH11 UNDER TEST
- L — IS THE LINE NO. BEING TESTED
- C — IS THE CONTENTS OF THE 'LPR'' DURING THE TEST
- I — IS THE NO. OF TEST PATTERN ITERATIONS COMPLETED

THIS TYPEOUT MAY BE INHIBITED BY SETTING SR13=1.

5.0 DH11 DEVICE INFORMATION

5.1 ADDRESS AND VECTOR ASSIGNMENTS

THE DH11 USES FLOATING ADDRESSES AND IS LOCATED AFTER DJ11'S IN THE FLOATING ADD
BECAUSE THE DH11 HAS EIGHT REGISTERS, IT MUST BE ASSIGNED AN ADDRESS THAT IS A M
SYSTEM SHOULD HAVE CONSECUTIVE ADDRESSES.

EXAMPLE #1: A SYSTEM WITH NO DJ11'S BUT TWO DH11'S.

760 010 CANNOT USE FOR DH11'S BECAUSE NOT MULTIPLE OF 20.
760 020 FIRST DH11
760 040 SECOND DH11
760 060 DH11 GAP (INDICATES THAT THERE ARE NO MORE DH11'S).

EXAMPLE #2: A SYSTEM WITH ONE DJ11, TWO DH11'S:

760 010 FIRST DJ11
760 020 DJ11 GAP (INDICATES THAT THERE ARE NO MORE DJ11'S).
760 030 CANNOT USE FOR DH11'S BECAUSE NOT MULTIPLE OF 20.
760 040 FIRST DH11
760 060 SECOND DH11
760 100 DH11 GAP (INDICATES THAT THERE ARE NO MORE DH11'S).

THE DH11 VECTORS (2) FOLLOW THOSE OF THE DJ11 IN THE FLOATING VECTOR SPACE THAT
AT 300 ARE USED IN THE FOLLOWING ORDER: DC11; KL11/DL11-A, B; DP11; DM11-A; DN11
PA611 PUNCHES; DT11; DX11; DL11-C, D, E; DH11.

THE RECEIVER VECTOR IS THE LOWER NUMBERED VECTOR. THE PRIORITY OF THE RECEIVER A
SELECTABLE BY MEANS OF TWO STANDARD PDP11 PRIORITY JUMPER PLUGS. BR LEVEL 5 IS S

5.2 REGISTER DEFINITION

THE FOLLOWING SECTION DESCRIBES THE BIT ASSIGNMENTS WITHIN EACH REGISTER: BITS M
AS ZERO. ATTEMPTING TO WRITE INTO UNUSED OR READ ONLY BITS HAS NO EFFECT ON THOS
GENERATED BY THE PROCESSOR (E.G. UPON EXECUTION OF A RESET INSTRUCTION). TRANSMI

5.2.1 THE SYSTEM CONTROL REGISTER - ADDRESS X00

THE SYSTEM CONTROL REGISTER IS A BYTE-ADDRESSABLE REGISTER. THE BIT ASSIGNMENT I

BITS	DESCRIPTION
00-03	LINE SELECTION

EACH OF THE 16 LINES SERVED BY THE DH11 HAS ITS OWN STORAGE FOR LINE PAR BYTE COUNT. THESE STORAGE LOCATIONS ARE LOADED BY THE PROGRAM VIA THE LI REGISTER, AND BYTE COUNT REGISTER, BUT THE HARDWARE MUST FIRST BE TOLD W CURRENT ADDRESS, OR BYTE COUNT CHANGED. THIS ROUTING IS ACCOMPLISHED BY THE BINARY ADDRESS (0000-1111) OF THE DESIRED LINE. THESE BITS ARE READ/

04, 05 MEMORY EXTENSION

THE INFORMATION STORED IN THESE BITS BECOMES BITS 16 AND 17 RESPECTIVELY PROGRAM INTO THE CURRENT ADDRESS REGISTER. THESE BITS ARE READ/WRITE BUT OF BITS 4 AND 5 OF THE SYSTEM CONTROL REGISTER, NOT THE STATUS OF ADDRES SEE THE SILO STATUS REGISTER FOR FURTHER INFORMATION. THIS ARRANGEMENT P SAVE THE CONTENTS OF THE SYSTEM CONTROL REGISTER ACCURATELY.

06 RECEIVER INTERRUPT ENABLE

THIS BIT, WHEN SET, ENABLES RECEIVER INTERRUPTS (BIT 7)

07 RECEIVER INTERRUPT

THIS BIT, WHEN SET, INDICATES THAT THE NUMBER OF CHARACTERS STORED IN TH SPECIFIED BY THE LOW BYTE OF THE SILO STATUS REGISTER. THIS BIT IS READ WHERE IT IS READ/WRITE. SETTING OF THIS BIT WILL GENERATE AN INTERRUPT R IS ALSO SET.

08 CLEAR NON-EXISTENT MEMORY INTERRUPT

THIS BIT, WHEN SET, CLEARS THE NON-EXISTENT MEMORY INTERRUPT FLIP-FLOP (IS READ/WRITE.

09 MAINTENANCE

THIS BIT, WHEN SET, PLACES THE DH11 IN MAINTENANCE MODE.

10 NON-EXISTENT MEMORY

THIS BIT IS SET WHENEVER THE NPR HARDWARE PLACES THE ADDRESSES OF A MEMO NO SLAVE SYNC IS RECEIVED IN 20 US. THIS INDICATES THAT THE ADDRESSED LO THIS BIT CAUSES AN INTERRUPT REQUEST IF SET WHILE TRANSMITTER AND NON-EX THIS BIT IS READ ONLY, EXCEPT IN MAINTENANCE MODE, WHERE IT IS READ/WRIT

11 MASTER CLEAR

THIS BIT, WHEN SET, GENERATES "INITIALIZE" WITHIN THE DH11, CLEARING THE EXACT BITS CLEARED ARE DISCUSSED IN THE SECTION ON INITIALIZATION. READ

```
1610                                    12      STORAGE INTERRUPT ENABLE
1611
1612                                            THIS BIT, WHEN SET, PERMITS THE SETTING OF BIT 14 TO GENERATE AN INTERRU
1613
1614                                    13      TRANSMITTER AND NON-EX-MEM INTERRUPT ENABLE
1615
1616                                            THIS BIT, WHEN SET, PERMITS THE SETTING OF BIT 10 OR 15 TO GENERATE AN I
1617
1618                                    14      STORAGE INTERRUPT
1619
1620                                            THIS BIT IS SET WHEN THE RECEIVER SCANNER FINDS A RECEIVER HOLDING BUFFE
1621                                            STORE THAT CHARACTER IN THE SILO, AND CANNOT DO SO BECAUSE OF A LACK OF
1622                                            AN INTERRUPT REQUEST IF BIT 12 IS SET. THIS BIT IS READ ONLY, EXCEPT IN
1623                                            IT IS READ/WRITE.
1624
1625                                    15      TRANSMITTER INTERRUPT
1626
1627                                            THIS BIT IS SET WHEN THE DH11 CONCLUDES AN NPR CYCLE THAT INCREMENTED A
1628                                            CHARACTER IN A MESSAGE BUFFER WAS LOADED INTO A UART TRANSMITTER HOLDING
1629                                            REQUEST IF BIT 13 IS SET.  THIS BIT IS READ/WRITE. (IT IS SET DURING AN
1630
```

5.2.2 NEXT RECEIVED CHARACTER REGISTER ADDRESS X02

BITS	DESCRIPTION
00-07	NEXT RECEIVED CHARACTER
	THESE BITS CONTAIN THE NEXT RECEIVED CHARACTER, RIGHT JUSTIFIED. THE LEA
08-1[illegible]	LINE NUMBER
	THESE BITS INDICATE THE LINE NUMBER ON WHICH THE NEXT RECEIVED CHARACTER LEAST SIGNIFICANT BIT.
12	PARITY ERROR
	THIS BIT IS SET IF THE PARITY OF THE RECEIVED CHARACTER DOES NOT AGREE W
13	FRAMING ERROR
	THIS BIT IS SET IF THE RECEIVER SAMPLES A LINE FOR THE FIRST STOP BIT, A (LOGICAL 0). THIS CONDITION USUALLY INDICATES THE RECEPTION OF A BREAK.
14	DATA OVERRUN
	THIS BIT IS SET WHEN THE RECEIVED CHARACTER WAS PRECEDED BY A CHARACTER RECEIVER SCANNER TO SERVICE THE UART RECEIVER HOLDING BUFFER. REFER TO T FURTHER DETAILS ON DOUBLE-BUFFERED RECEPTION.
15	VALID DATA PRESENT
	THIS BIT INDICATES THAT THE DATA PRESENTED IN BITS 14-00 IS VALID. IT PE CHARACTERS FROM THE SILO UNTIL IT IS EMPTY. THIS IS DONE BY READING THIS IS OBTAINED FOR WHICH BIT 15 IS A ZERO. THE ENTIRE NEXT RECEIVED CHARACT ONLY ON A WORD BASIS.

5.2.3 LINE PARAMETER REGISTER ADDRESS X04

THIS REGISTER SHOULD BE LOADED ONLY AFTER THE LINE SELECTION BITS OF THE SYSTEM LINE TO WHICH THESE PARAMETERS APPLY. THIS REGISTER IS WRITE ONLY.

BITS	DESCRIPTION
00-01	CHARACTER LENGTH

THESE BITS SHOULD BE SET AS SHOWN TO RECEIVE AND TRANSMIT CHARACTERS OF

BIT 01	00	
0	0	5 BIT
0	1	6 BIT
1	0	7 BIT
1	1	8 BIT

02 TWO STOP BITS

THIS BIT, WHEN SET, CONDITIONS A LINE TRANSMITTING WITH 6, 7, OR 8-BIT C MARKS. IF THE LINE IS TRANSMITTING 5-BIT CODE, ASSERTION OF THIS BIT CAU 1.5 STOP MARKS. IF THIS BIT IS NOT ASSERTED, 1 STOP MARK IS SENT.

03 NOT USED

04 PARITY ENABLED

IF THIS BIT IS SET, CHARACTERS TRANSMITTED ON THIS LINE WILL HAVE AN APP RECEIVED ON THIS LINE WILL HAVE THEIR PARITY CHECKED.

05 ODD PARITY

IF THIS BIT AND BIT 4 ARE SET, CHARACTERS OF ODD PARITY WILL BE GENERATE WILL BE EXPECTED TO HAVE ODD PARITY. IF THIS BIT IS NOT SET, BUT BIT 4 I GENERATED ON THIS LINE AND INCOMING CHARACTERS WILL BE EXPECTED TO HAVE OF THIS BIT IS IMMATERIAL.

06-09 RECEIVER SPEED

THE STATE OF THESE BITS DETERMINES THE OPERATING SPEED FOR THIS LINE'S R BELOW IS APPLICABLE.

10-13 TRANSMITTER SPEED

THE STATE OF THESE BITS DETERMINES THE OPERATING SPEED FOR THIS LINE'S T TABLE ON THE NEXT PAGE IS APPLICABLE.

SPEED TABLE FOR RECEIVER AND TRANSMITTER SPEEDS:

	BIT				
TRANSMITTER	13	12	11	10	
RECEIVER	9	8	7	6	
	0	0	0	0	ZERO BAUD
	0	0	0	1	50 BAUDS
	0	0	1	0	75 BAUDS
	0	0	1	1	110 BAUDS
	0	1	0	0	134.5 BAUDS
	0	1	0	1	150 BAUDS
	0	1	1	0	200 BAUDS
	0	1	1	1	300 BAUDS
	1	0	0	0	600 BAUDS
	1	0	0	1	1200 BAUDS
	1	0	1	0	1800 BAUDS
	1	0	1	1	2400 BAUDS
	1	1	0	0	4800 BAUDS
	1	1	0	1	9600 BAUDS
	1	1	1	0	EXTERNAL INPUT A
	1	1	1	1	EXTERNAL INPUT B

14 HALF DUPLEX/FULL DUPLEX

IF THIS BIT IS SET, THIS LINE WILL OPERATE IN HALF-DUPLEX MODE. IF NOT S
IN FULL-DUPLEX MODE.

IN THIS APPLICATION HALF-DUPLEX MEANS THAT THE DH11 RECEIVER IS BLINDED

15 AUTO-ECHO ENABLE

WHEN THIS BIT IS SET, CHARACTERS RECEIVED ON THIS LINE WILL BE HARDWARE
FURTHER DETAILS.

5.2.4 CURRENT ADDRESS REGISTER ADDRESS X06

THIS REGISTER SHOULD BE LOADED ONLY AFTER THE SYSTEM CONTROL REGISTER (SCR) HAS
DESIRED LINE NUMBER. WHEN THIS REGISTER IS LOADED, ADDRESS BITS 00-15 ARE TRANSF
MEMORIES IN THE DH11 FROM BITS 00-15 OF THIS REGISTER. ADDRESS BITS 16-17 ARE TR
MEMORIES IN THE DH11 FROM BITS 4-5 OF THE SYSTEM CONTROL REGISTER.

INTERRUPTS MUST BE INHIBITED OR THE SCR SAVED BETWEEN THE SETTING OF THE SCR BIT
ADDRESS REGISTER.

WHEN THIS REGISTER IS READ, IT WILL INDICATE THE CURRENT ADDRESS OF THE LINE SEL
BITS 16 AND 17 WILL APPEAR IN THE SILO STATUS REGISTER, BITS 6 AND 7.

5.2.5 BYTE COUNT REGISTER ADDRESS X10

IN THE SAME FASHION AS THE LINE PARAMETER AND CURRENT ADDRESS REGISTERS, THIS RE
FIRST SELECTING A LINE NUMBER BY MEANS OF THE LOWER-ORDER FOUR BITS OF THE SYSTE
LOADED WITH THE TWO'S COMPLEMENT OF THE NUMBER OF CHARACTERS (BYTES) TO BE TRANS
IS READ/WRITE.

INTERRUPTS MUST BE INHIBITED OR THE SCR SAVED BETWEEN THE SETTING OF THE SCR BIT
COUNT REGISTER

5.2.6 BUFFER ACTIVE REGISTER (BAR) ADDRESS X12

THIS REGISTER CONTAINS ONE BIT FOR EACH LINE. THE BITS ARE INDIVIDUALLY SET USIN
TRANSMISSION ON THE ASSOCIATED LINE. THE BIT IS CLEARED BY THE HARDWARE WHEN THE
IS LOADED INTO THE TRANSMITTER DATA HOLDING REGISTER OF THE UART FOR THAT LINE.
THE CLEARING OF A BAR DOES INDICATE THAT A MESSAGE MAY BE SENT, IT DOES NOT INDI
FROM THE PRECEDING MESSAGE HAVE BEEN COMPLETELY SENT. SPECIFICALLY, TWO MORE CHA
BIT CLEARS. THESE ARE THE LAST TWO CHARACTERS OF THE MESSAGE, ONE OF THEM WAS JU
AND ONE WAS THAT FINAL CHARACTER THAT WAS LOADED INTO THE HOLDING REGISTER, THUS
IS A NORMAL CONSEQUENCE OF DOUBLE-BUFFERED TRANSMISSION AND IS MENTIONED HERE FO
WANT TO WRITE PROGRAMS THAT CONTROL SUCH MODEM LEADS ARE REQUEST TO SEND. REQUES
DROPPED UNTIL AT LEAST TWO CHARACTER TIMES AFTER THE BAR BIT FOR A GIVEN LINE CL

THIS TIMING MAY BE EFFECTED BY SENDING TWO EXTRA (NULL) CHARACTERS IN A MESSAGE

CLEARING A BAR BIT SHOULD NOT BE USED TO ABORT TRANSMISSION ON A LINE. RATHER, T
TO ZERO. THE BUFFER ACTIVE REGISTER BITS ARE READ/WRITE.

5.2.7 BREAK CONTROL REGISTER ADDRESS X14

THIS REGISTER CONTAINS ONE BIT FOR EACH LINE. SETTING A BIT IN THIS REGISTER WIL
ON THE LINE CORRESPONDING TO THAT BIT NUMBER. CLEARING THE BIT WILL TERMINATE TH
MAY BE TIMED BY SENDING CHARACTERS DURING THE BREAK INTERVAL, SINCE THESE CHARAC
FURTHER COMMENTS CONCERNING THE TRANSMISSION OF BREAK SIGNALS MAY BE FOUND IN TH

5.2.8 SILO STATUS REGISTER ADDRESS X16

THIS REGISTER IS ACTUALLY TWO BYTE-SIZED REGISTERS. THE BIT ASSIGNMENTS ARE:

BIT	DESCRIPTION
00-05	SILO ALARM LEVEL

THE PROGRAM MAY LOAD AN INTEGRAL POWER OF 2 BETWEEN 0 AND 63 INTO THIS L
WHEN THE NUMBER OF CHARACTERS STORED IN THE SILO EXCEEDS THAT NUMBER, AN
REGISTER BIT 7) IS GENERATED, IF SYSTEM CONTROL REGISTER BIT 6 IS SET. T

06-07	READ EXTENDED MEMORY

THESE BITS ARE READ ONLY AND CONTAIN THE A16 AND A17 BITS OF THE CURRENT
SELECTION BITS OF THE SYSTEM CONTROL REGISTER ARE POINTING.

08-13	SILO FILL LEVEL

THESE BITS ARE AN UP-DOWN COUNTER THAT INDICATES THE ACTUAL NUMBER OF CH
BE NOTED THAT THERE ARE SIX BITS, HENCE NUMBERS BETWEEN 0 AND 63 CAN BE
ENTRIES AND THE FILL LEVEL APPEARS AS 00000, BUT ONE MAY EASILY TELL THE
SILO (00000) AND A FULL SILO (00000) BY CHECKING THE STORAGE OVERFLOW BI
THESE BITS ARE READ ONLY.

5.3 DH11 MODULE ALLOCATION CHART

VIEW FROM WIRING SIDE

	SLOT 1	2	3	4	5	6	7
	M920	M7821	M7277	M7287	M7289	M7821	M7360
	CAB E						
ROW A	UNIBUS CONNECTOR (NOTE #3)	NPR CNTL	REG 8 BYTE CNT	CURRENT ADDRS 8 ADDRS	SYSTEM CNTL 8 RCV SCAN	INTR CNTL	PRIORITY SELECTOR (NOTE #9)
		M796				M405	M971
							CABLE
B		UNIBUS MASTER CNTL				EXTERNAL B CLOCK (NOTE #5)	DATA CABLE (NOTES #6 & #9)
	M7247	M7247				M7280	M7280
C	* CONTROL MUX LINES 8-15 (NOTE #7)	* CONTROL MUX LINES 0-7 (NOTE #8)				MULTIPLE UART LINES 0-7	MULTIPLE UART LINES 8-15
D							
	M105	M7246					
E	* ADDRESS SELECTOR (NOTE #7)	* CONTROL SCAN (NOTES #4 & #8)					
	M7821						
F	* INTR CNTL (NOTE #7)						

FIGURE 2-4 DH11 MODULE UTILIZATION DIAGRAM
PAGE 2

NOTES:

1. IF END OF BUS, REPLACE M920 WITH M930.

2. IF LAST UNIT IN BASIC BOX, REPLACE M920 WITH BC11A CABLE WHEN EXPANDING TO PERIPHERAL BOX.

3. IF FIRST UNIT IN EXPANDER BOX, REPLACE M920 WITH BC11A CABLE.

4. E02 MUST BE G727 GRANT CONTINUITY IF MODEM CONTROL MODULE SET IS NOT INSTALLED. * DENOTES DM11-BB MODEM CONTROL OPTION, WITH DH11-AA OR AC.

5. MODULE SLOTS PROVIDE FOR ADDITIONAL CLOCK RATES.

6. FOR DIAGNOSTIC CHECKOUT OF DH11-AA, AB, OR AC, REPLACES M971 WITH M974.

7. THIS SLOT CONTAINS MODEM CONTROL MODULE M7807 WITH DH11-AD.

8. THIS SLOT CONTAINS MODEM CONTROL MODULE M7808 WITH DH11-AD.

9. THIS SLOT CONTAINS EIA CONVERTER AND PRIORITY MODULE M5906 FOR DH11-AD OR AE.

6.0 MAINTENANCE PROCEDURES

THIS SECTION OUTLINES SOME GENERAL TECHNIQUES FOR USING CZDHND FOR MAINTENANCE AND CHECKOUT OF THE DH11 SUBSYSTEM. SINCE THIS PROGRAM DOES NOT TEST ALL POSSIBLE DH11 FEATURES (BREAK,AUTO-ECHO, HALF DUPLEX ETC.) THE USER MUST ALSO RUN THE DIAGNOSTIC, MD-11-DZDHM, PRIOR TO USING THIS PROGRAM TO INSURE COMPLETE CHECKOUT AND VERIFICATION OF THE DH11 HARDWARE.

6.1 MAINTENANC CONNECTORS

BOTH THE DATA RELIABILITY AND PATTERNS/CABLE SUBPROGRAMS REQUIRE THAT THE USER INSTALL THE APPROPRIATE MAINTENANCE JUMPERS OR MODULES BEFORE RUNNING THE PROGRAM. DEPENDENT UPON THE SPECIFIC DH11 CONFIGURATION AND THE TYPE OF TESTING DESIRED, CERTAIN MAINTENANCE AIDS MUST BE INSTALLED AS OUTLINED BELOW:

A. DH11-AA, AB, OR AC CONFIGURATIONS

1) TESTING LOGIC FOR ALL LINES WITHOUT DATA CABLES OR LEVEL CONVERTERS

   A. REMOVE THE DATA CABLE FROM SLOT B7 IN EACH DH11 TO BE TESTED.
   B. INSTALL AN M974 MAINT JUMPER MODULE INTO SLOT B7 OF EACH DH11 TO BE TESTED.

2) TESTING ALL 16. LINES INCLUDING DATA CABLES WHICH CONNECT TO DISTRIBUTION PANEL. DOES NOT TEST LEVEL CONVERTER CIRCUITS LOCATED IN DISTRIBUTION PANEL.

   A. INSTALL THE M974 MAINT JUMPER MODULE INTO SLOT B3 OF THE MULTIPLEXOR DISTRIBUTION PANEL FOR EACH DH11 TO BE TESTED.

3) TESTING ONE OR MORE SINGLE LINES INCLUDING EIA LEVEL CONVERTERS AND DEVICE CABLES WHICH ARE NOT TESTED IN 1 AND 2 ABOVE.

   A. INSTALL AN H315 TEST CONNECTOR AT THE END OF THE DEVICE CABLE FOR EACH LINE TO BE TESTED.

B. DH11-AD CONFIGURATION

1. TESTING ALL 16. LINES WITHOUT DATA CABLES

   A. DISCONNECT THE DATA CABLES (2) FROM THE TWO CONNECTORS ON THE M5906 MODULE (SLOT AB7 OF THE DH11 BACKPLANE.

B. INSTALL TWO H8611 TEST CONNECTORS ON THE M5906
IN PLACE OF THE CABLES.

2. TESTING ONE OR MORE SINGLE LINES INCLUDING DATA CABLES

A. DISCONNECT THE DEVICE CABLE FROM THE DH11-AD
DISTRIBUTION PANEL FOR EACH LINE TO BE TESTED.

B. INSTALL AN H315 TEST CONNECTOR IN ITS PLACE
ON THE DH11-AD DISTRIBUTION PANEL.

NOTE: TO TEST THE DEVICE CABLE AS WELL, INSTALL
THE H315 TEST CONNECTOR AT THE END OF THE
DEVICE CABLE AND LEAVE THE DEVICE CABLE
CONNECTED TO THE DISTRIBUTION PANEL.

6.2 DATA RELIABILITY TESTING

A. COMPLETE RELIABILITY TESTING (OVER NIGHT RUNS)

1) SET UP THE TEST JUMPERS AS REQUIRED FOR THE PARTICULAR
CONFIGURATION TO BE TESTED. (REFER TO PARA 6.1)
2) LOAD CZDHND AND START IT AT LOC 000200(8).
3) TYPE IN THE DESIRED DH11 PARAMETERS - SET THE SR-000200
AND LET THE PROGRAM RUN.

A COMPLETE TEST RUN FOR 16. LINES ON EACH DH11 WILL
TAKE APPROX 4 HOURS (TWO DH11'S WOULD TAKE 8. HOURS) ETC.

AT THE COMPLETION OF TESTING FOR EACH DH11 THE ERROR
STATISTICS TABLE WILL BE TYPED OUT.

4) LET THE PROGRAM RUN AT LEAST ONE PASS (4 HRS/DH11)
PREFERABLY OVERNIGHT, AND THEN ANALYZE ANY ERROR
PRINTOUTS AND THE ERROR STATISTIC TABLE DATA.
5) IF ERRORS OCCUR IT SHOULD BE SIMPLE FOR THE USER TO
DETERMINE WHICH LINE, WHICH DH11, AND THE FAILING MODES.
OF OPERATION TO AID IN FAULT ISOLATION.

B. QUICK DATA RELIABILITY TESTING

1) FOLLOW THE SAME PROCEDURE AS IN PARA 6.2(A) ABOVE EXCEPT
SET THE SR=000000(8) BEFORE STARTING THE RUN.

THE QUICK TESTS VERIFY ALL COMBINATIONS OF LINE
PARAMETERS ON ALL LINES AT 9600. BAUD ONLY. ALL
OTHER BAUD RATES ARE TESTED WITH 5 BIT CHARS,
ONE STOP BIT, AND ODD PARITY ONLY.

2) THE QUICK TEST TAKES APPROX. 15 SECONDS PER LINE SO
2 DH11'S (ALL 16. LINES) COULD BE TESTED IN APPROX
8. MINUTES.

3) THE ERROR INFORMATION PROVIDED IS IDENTICAL TO THAT
FOR THE COMPLETE TEST EXCEPT LESS TOTAL DATA TRANSFERS

OCCUR.

6.3 DATA PATTERNS TESTING

THE DIAGNOSTIC, MD-11-DZDHM, AND THE DATA RELIABILITY TESTS USE ONLY A BINARY UP COUNT PATTERN FOR DATA TESTING WITH A MAXIMUM BUFFER SIZE OF 256. BYTES. TO PROVIDE DIFFERENT DATA PATTERNS, THE USER CAN RUN THE DATA PATTERNS/CABLE TESTS. THESE TESTS ALLOW HIM TO SIT AT THE CONSOLE TERMINAL AND TEST EACH LINE INDIVIDUALLY WITH VARIOUS PARAMETERS, DATA PATTERNS, BUFFER SIZES, ETC.

1) SET UP THE TEST JUMPERS FOR THE LINES TO BE TESTED AS DESCRIBED IN PARA 6.1.

2) LOAD CZDHND AND START IT AT LOC 000220(8) TO RUN THE DATA PATTERNS TESTS.

3) REFER TO PARA 2.1.2.3 FOR THE OPERATING INSTRUCTIONS.

4) ONCE A FAILING PATTERN TEST IS FOUND, THE USER CAN RECONFIGURE THE TEST JUMPERS TO ISOLATE THE FAULT TO EITHER THE DH11 OR A FAULTY CABLE AND/OR CONNECTOR.

6.4 ECHO TESTING

THESE TESTS ALLOW THE USER TO CONNECT AN ASYNCHRONOUS TERMINAL TO THE DH11 DISTRIBUTION PANEL AND VERFIFY THE PARTICULAR LINE AS IT MIGHT BE USED ON-LINE. REFER TO PARA 2.1.2.2 FOR THE OPERATING INSTRUCTIONS FOR THE DH11 ECHO TEST.

```
                                .NLIST  CND,MD,MC
                                .LIST   TOC,ME,SEQ,BIN
          165000                $SWR=165000

                                .ENABLE ABS
                                .TITLE  CZDHN-D
                                ;*COPYRIGHT (C) 1977
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY ED CROWLEY
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*
          000001                $TN 1
                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*         9            LOOP ON ERROR
                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
          000000                        $SVPC=.                 ;SAVE PC
          000046                        .-46
000046    120000                        120000                  ;;1)SET LOC.46 TO ADDRESS OF 120000
          000052                        .-52
000052    000000                        .WORD   0               ;;2)SET LOC.52 TO ZERO
          000000                        .=$SVPC                 ;; RESTORE PC
                                .SBTTL  APT PARAMETER BLOCK

                                ;;*****************************************************************
                                ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                ;;*****************************************************************
          000000                        .$X=.   ;;SAVE CURRENT LOCATION
          000024                        .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024    000200                        200     ;;FOR APT START UP
          000044                        .-44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044    000000                        $APTHDR ;;POINT TO APT HEADER BLOCK
          000000                        .=.$X   ;;RESET LOCATION COUNTER
                                ;;*****************************************************************
                                ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                ;INTERFACE SPEC.

000000                          $APTHD:
000000    000000                $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
000002    001232                $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
000004    001604                $TSTM:  .WORD   1604    ;;RUN TIM OF LONGEST TEST
000006    001604                $PASTM: .WORD   1604    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
000010    001604                $UNITM: .WORD   1604    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
```

```
 2130  000012  000036                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
 2131                                  .SBTTL  TRAP CATCHER
 2132
 2133          000000                          .=0
 2134                                  ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
 2135                                  ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
 2136                                  ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
 2137          000174                          .=174
 2138  000174  000000                DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
 2139  000176  000000                SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
 2140                                  .SBTTL  STARTING ADDRESS(ES)
 2141  000200  000137  016160                  JMP     @#INPARX        ;;JUMP TO STARTING ADDRESS OF PROGRAM
 2142
 2143  000204  000137  001576                  JMP     @#BEGIN         ;BEGIN EXECUTION WITH DEFAULT PARAMETERS
 2144  000210  000137  016172                  JMP     @#INPARC        ;INPUT PARAMETERS - DEVICE SELECTION ONLY
 2145  000214  000137  004672                  JMP     @#ECHO          ;GO START LINE ECHO TESTS
 2146  000220  000137  006072                  JMP     @#EXPAT         ;GO START DATA PATTERNS TESTS
```

```
                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                          STACK=  1100
                                .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                                ;*MISCELLANEOUS DEFINITIONS
000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                          LF=     12              ;;CODE FOR LINE FEED
000015                          CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                          PS=     177776          ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                          DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                          DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                          R0=     %0              ;;GENERAL REGISTER
000001                          R1      %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3      %3              ;;GENERAL REGISTER
000004                          R4      %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
000006                          R6      %6              ;;GENERAL REGISTER
000007                          R7      %7              ;;GENERAL REGISTER
000006                          SP=     %6              ;;STACK POINTER
000007                          PC=     %7              ;;PROGRAM COUNTER

                                ;*PRIORITY LEVEL DEFINITIONS
000000                          PR0=    0               ;;PRIORITY LEVEL 0
000040                          PR1=    40              ;;PRIORITY LEVEL 1
000100                          PR2=    100             ;;PRIORITY LEVEL 2
000140                          PR3=    140             ;;PRIORITY LEVEL 3
000200                          PR4=    200             ;;PRIORITY LEVEL 4
000240                          PR5=    240             ;;PRIORITY LEVEL 5
000300                          PR6     300             ;;PRIORITY LEVEL 6
000340                          PR7=    340             ;;PRIORITY LEVEL 7

                                ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
000010                          SW03=   10
000004                          SW02=   4
```

```
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
```

```
                                .SBTTL  COMMON TAGS

                                ;;*****************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

            001100                      .=1100
001100                          $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                          .WORD   0
001102     000                  $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103     000                  $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114     000                  $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115     001                  $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134     000                  $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135     000                  $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001146  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
001157     000                  $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE' FLAG (BIT<07>=0-YES)
001160  000000                  $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                                ;;WHICH  ($REG0) WAS OBTAINED
001162  000000                  $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000                  $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  000000                  $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001170  000000                  $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001172  000000                  $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001174  000000                  $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001176  000000                  $REG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
001200  000000                  $REG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
001202  000000                  $TMP0:  .WORD   0               ;;USER DEFINED
001204  000000                  $TMP1:  .WORD   0               ;;USER DEFINED
001206  000000                  $TMP2:  .WORD   0               ;;USER DEFINED
001210  000000                  $TMP3:  .WORD   0               ;;USER DEFINED
001212  000000                  $TMP4:  .WORD   0               ;;USER DEFINED
001214  000000                  $TMP5:  .WORD   0               ;;USER DEFINED
001216  000000                  $TMP6:  .WORD   0               ;;USER DEFINED
001220  000000                  $TMP7:  .WORD   0               ;;USER DEFINED
001222  000000                  $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
```

```
 2313  001224  000000                $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
 2314  001226     077                $QUES:  .ASCII  /?/             ;;QUESTION MARK
 2315  001227     015                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
 2316  001230  000012                $LF:    .ASCIZ  <12>            ;;LINE FEED
 2317                                ;;*****************************************************************
 2318                                .SBTTL  APT MAILBOX-ETABLE
 2319
 2320                                ;;*****************************************************************
 2321                                .EVEN
 2322  001232                        $MAIL:                          ;;APT MAILBOX
 2323  001232  000000                $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
 2324  001234  000000                $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
 2325  001236  000000                $TESTN: .WORD   ATESTN  ;;TEST NUMBER
 2326  001240  000000                $PASS:  .WORD   APASS   ;;PASS COUNT
 2327  001242  000000                $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
 2328  001244  000000                $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
 2329  001246  000000                $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
 2330  001250  000000                $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
 2331  001252                        $ETABLE:                        ;;APT ENVIRONMENT TABLE
 2332  001252     000                $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
 2333  001253     000                $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
 2334  001254  000000                $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
 2335  001256  000000                $USWR:  .WORD   AUSWR   ;;USER SWITCHES
 2336  001260  000000                $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
 2337                                ;*                              BITS 15-11=CPU TYPE
 2338                                ;*                                      11/04=01,11/05=02,11/20=03,11/40-04,11/45-05
 2339                                ;*                                      11/70=06,PDQ=07,Q=10
 2340                                ;*                              BIT 10=REAL TIME CLOCK
 2341                                ;*                              BIT  9=FLOATING POINT PROCESSOR
 2342                                ;*                              BIT  8=MEMORY MANAGEMENT
 2343  001262     000                $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
 2344  001263     000                $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
 2345                                ;*                              MEM.TYPE BYTE   --  (HIGH BYTE)
 2346                                ;*                                      900 NSEC CORE=001
 2347                                ;*                                      300 NSEC BIPOLAR=002
 2348                                ;*                                      500 NSEC MOS=003
 2349  001264  000000                $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
 2350                                ;*                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
 2351  001266     000                $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
 2352  001267     000                $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
 2353  001270  000000                $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
 2354  001272     000                $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
 2355  001273     000                $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
 2356  001274  000000                $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
 2357  001276     000                $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
 2358  001277     000                $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
 2359  001300  000000                $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
 2360  001302  000000                $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
 2361  001304  000000                $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
 2362  001306  000000                $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
 2363  001310  000000                $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
 2364  001312  000000                $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
 2365  001314  000000                $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
 2366  001316  000000                $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
 2367  001320  000000                $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
 2368  001322  000000                $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
```

```
 2369  001324  000000                $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
 2370  001326                        $ETEND:
 2371                                .MEXIT
```

```
                                        .SBTTL  ERROR POINTER TABLE

                                        ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                        ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                        ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                        ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                        ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                        ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                        ;*      DH              ;;POINTS TO THE DATA HEADER
                                        ;*      DT              ;;POINTS TO THE DATA
                                        ;*      DF              ;;POINTS TO THE DATA FORMAT


001326                                  $ERRTB:

                                        ;ERROR TABLE ITEM FOR ERROR 1

                                                EM1             ;'NON EX MEMORY ERROR - DROPPED LINE #  ''
                                                DH1             ;'' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B''
                                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                                DF2             ;PRINT ALL OCTAL

                                        ;ERROR TABLE ITEM FOR ERROR 2

                                                EM2             ;'TRANSMITTER FALSE INTERRUPT - DROPPED LINE#  ''
                                                DH1             ;'' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B''
                                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                                DF2             ;PRINT ALL OCTAL

                                        ;ERROR TABLE ITEM FOR ERROR 3

                                                EM3             ;'BUFFER ACTIVE REGISTER ERROR - DROPPED LINE #  ''
                                                DH1             ;'' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B''
                                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                                DF2             ;PRINT ALL OCTAL

                                        ;ERROR TABLE ITEM FOR ERROR 4

                                                EM4             ;'BYTE COUNT REGISTER ERROR - DROPPED LINE #  ''
                                                DH1             ;'' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B''
                                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                                DF2             ;PRINT ALL OCTAL

                                        ;ERROR TABLE ITEM FOR ERROR 5

                                                EM5             ;'CURRENT ADDRESS REGISTER ERROR - DROPPED LINE #  ''
                                                DH1             ;'' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B''
                                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                                DF2             ;PRINT ALL OCTAL

                                        ;ERROR TABLE ITEM FOR ERROR 6

                                                EM6             ;'SILO OVERFLOW ERROR - DROPPED LINE #  ''
                                                DH1             ;'' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B''
                                                DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3 $REG4

001326  022434
001330  022503
001332  022560
001334  022576

001336  022606
001340  022503
001342  022560
001344  022576

001346  022665
001350  022503
001352  022560
001354  022576

001356  022745
001360  022503
001362  022560
001364  022576

001366  023022
001370  022503
001372  022560
001374  022576

001376  023104
001400  022503
001402  022560
```

```
2428  001404  022576                        DF2             ;PRINT ALL OCTAL
2429
2430                                  ;ERROR TABLE ITEM FOR ERROR 7
2431
2432  001406  023153                        EM7             ;'RECEIVER FALSE INTERRUPT - LINE #  ''
2433  001410  022503                        DH1             ;'' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B''
2434  001412  022560                        DT1             ;'$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
2435  001414  022576                        DF2             ;PRINT ALL OCTAL
2436
2437                                  ;ERROR TABLE ITEM FOR ERROR 10
2438
2439  001416  023227                        EM10            ;''INVALID DATA IN SILO - DROPPED LINE #  ''
2440  001420  023277                        DH2             ; (PC)   CURLPR  CHAR #  WASADR  SHBADR   WAS     S/B''
2441  001422  023364                        DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
2442  001424  022576                        DF2             ;PRINT ALL OCTAL
2443
2444                                  ;ERROR TABLE ITEM FOR ERROR 11
2445
2446  001426  023404                        EM11            ;'DATA ERROR - LINE #  ''
2447  001430  023277                        DH2             ; (PC)   CURLPR  CHAR #  WASADR  SHBADR   WAS     S/B''
2448  001432  023364                        DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
2449  001434  022576                        DF2             ;PRINT ALL OCTAL
2450
2451                                  ;ERROR TABLE ITEM FOR ERROR 12
2452
2453  001436  023432                        EM12            ;''TEST TIMEOUT - DROPPED LINE #  ''
2454  001440  023472                        DH3             ;'' (PC)   CURLPR  RTOTAL  XTOTAL  RDONE''
2455  001442  023540                        DT3             ;'$ERRPC,CURLPR,$TMP0,$TMP1RDONE''
2456  001444  022576                        DF2             ;PRINT ALL OCTAL
2457
2458                                  ;ERROR TABLE ITEM FOR ERROR 13
2459
2460  001446  000000                        0               ;NO MESSAGE
2461  001450  000000                        0               ;NO DATA HEADER
2462  001452  023554                        DT4             ;$TMP0,$TMP1,$TMP2,$TMP3,$TMP4,$TMP5,$TMP6
2463  001454  023574                        DF1             ;PRINT ALL DECIMAL
2464
2465                                  ;ERROR TABLE ITEM FOR ERROR 14
2466
2467  001456  023604                        EM14            ;'BUS ERROR TRAP TO 04''
2468  001460  023631                        DH4             ;'' (PC)    (PS)    (SP)   TRAPPC  TRAPPS''
2469  001462  023700                        DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2''
2470  001464  022576                        DF2             ;PRINT ALL OCTAL
2471
2472                                  ;ERROR TABLE ITEM FOR ERROR 15
2473
2474  001466  023714                        EM15            ;'RSVD INSTR TRAP TO 10''
2475  001470  023631                        DH4             ;'' (PC)    (PS)    (SP)   TRAPPC  TRAPPS''
2476  001472  023700                        DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2''
2477  001474  022576                        DF2             ;PRINT ALL OCTAL
2478
2479                                  ;ERROR TABLE ITEM FOR ERROR 16
2480
2481  001476  023742                        EM16            ;''SINGLE LINE ECHO TEST - INTR WAIT TIMEOUT''
2482  001500  024014                        DH5             ;'' (PC)   DEVADR   LINE    (SCR)  CURLPR  EXFLAG''
2483  001502  024074                        DT6             ;$ERRPC,$REG1,LINE,$TMP0,CURLPR,EXFLAG''
```

```
 2484  001504  022576                        DF2             ;PRINT ALL OCTAL
 2485
 2486                                ;ERROR TABLE ITEM FOR ERROR 17
 2487
 2488  001506  024112                        EM17            ;''ALTERNATING 1/0 PATTERN TEST DONE''
 2489  001510  024154                        DH6             ;'' (PC)   DEVADR   LINE   CURLPR  ICOUNT''
 2490  001512  024224                        DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
 2491  001514  022576                        DF2             ;PRINT ALL OCTAL
 2492
 2493                                ;ERROR TABLE ITEM FOR ERROR 20
 2494
 2495  001516  024240                        EM20            ;''BINARY UP COUNT PATTERN TEST DONE''
 2496  001520  024154                        DH6             ;'' (PC)   DEVADR   LINE   CURLPR  ICOUNT''
 2497  001522  024224                        DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
 2498  001524  022576                        DF2             ;PRINT ALL OCTAL
 2499
 2500                                ;ERROR TABLE ITEM FOR ERROR 21
 2501
 2502  001526  024302                        EM21            ;''BINARY DOWN COUNT PATTERN TEST DONE''
 2503  001530  024154                        DH6             ;'' (PC)   DEVADR   LINE   CURLPR  ICOUNT''
 2504  001532  024224                        DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
 2505  001534  022576                        DF2             ;PRINT ALL OCTAL
 2506
 2507                                ;ERROR TABLE ITEM FOR ERROR 22
 2508
 2509  001536  024346                        EM22            ;''RANDOM DATA PATTERN TEST DONE''
 2510  001540  024154                        DH6             ;'' (PC)   DEVADR   LINE   CURLPR  ICOUNT''
 2511  001542  024224                        DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
 2512  001544  022576                        DF2             ;PRINT ALL OCTAL
 2513
 2514                                ;ERROR TABLE ITEM FOR ERROR 23
 2515
 2516  001546  024404                        EM23            ;''SINGLE CHAR PATTERN TEST DONE''
 2517  001550  024154                        DH6             ;'' (PC)   DEVADR   LINE   CURLPR  ICOUNT''
 2518  001552  024224                        DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
 2519  001554  022576                        DF2             ;PRINT ALL OCTAL
 2520
 2521                                ;ERROR TABLE ITEM FOR ERROR 24
 2522
 2523  001556  024442                        EM24            ;''TYPED BUFFER PATTERN TEST DONE''
 2524  001560  024154                        DH6             ;'' (PC)   DEVADR   LINE   CURLPR  ICOUNT''
 2525  001562  024224                        DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
 2526  001564  022576                        DF2             ;PRINT ALL OCTAL
 2527
 2528                                ;ERROR TABLE ITEM FOR ERROR 25
 2529
 2530  001566  024501                        EM25            ;''DATA PATTERNS TEST TIMEOUT''
 2531  001570  024534                        DH7             ;'' (PC) DEVADR   LINE   CURLPR  ICOUNT  PATCDE''
 2532  001572  024614                        DT10            ;$ERRPC,DHADR,LINE,CURLPR,$REG0,$REG1
 2533  001574  022576                        DF2             ;PRINT ALL OCTAL
 2534
 2535
 2536
 2537
 2538  001576  005000                BEGIN:  CLR     R0              ;INIT R0 TO INDICATE DEFAULT PARAMETERS
 2539  001600  005067  020102                CLR     VCFLG           ;INIT VECTOR ADDR SET UP FLAG
```

```
2540  001604  005067  020530          CLR     DPFLG           ;CLEAR DATA PATTERNS TEST FLAG
2541  001610  005067  020536          CLR     RETFLG          ;CLEAR ECHO TEST RETURN FLAG
2542  001614  005067  020510  BEGINA: CLR     TITFLG          ;INIT TITLE MESSAGE FLAG
2543                          .SBTTL  INITIALIZE THE COMMON TAGS
2544                          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2545  001620  012706  001100          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
2546  001624  005026                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
2547  001626  022706  001140          CMP     #SWR,R6 ;;DONE?
2548  001632  001374                  BNE     .-6             ;;LOOP BACK IF NO
2549  001634  012706  001100          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
2550                          ;;INITIALIZE A FEW VECTORS
2551  001640  012737  011076  000020  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2552  001646  012737  000340  000022  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
2553  001654  012737  011342  000030  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2554  001662  012737  000340  000032  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
2555  001670  012737  014312  000034  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2556  001676  012737  000340  000036  MOV     #340,@#TRAPVEC+2;LEVEL 7
2557  001704  012737  014376  000024  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
2558  001712  012737  000340  000026  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
2559  001720  005067  177276          CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
2560  001724  005067  177274          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2561  001730  112767  000001  177157  MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
2562  001736  012767  001736  177142  MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2563  001744  012767  001744  177136  MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
2564                          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2565                          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2566  001752  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
2567  001756  012737  002012  000004  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
2568  001764  012767  177570  177146  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
2569  001772  012767  177570  177142  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
2570  002000  022777  177777  177132  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
2571  002006  001012                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2572                                                          ;;AND  THE HARDWARE SWR IS NOT  -1
2573  002010  000403                  BR      65$             ;;BRANCH IF NO TIMEOUT
2574  002012  012716  002020  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
2575  002016  000002                  RTI
2576  002020  012767  000176  177112  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
2577  002026  012767  000174  177106          MOV     #DISPREG,DISPLAY
2578  002034  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
2579
2580  002040  005067  177174          CLR     $PASS           ;;CLEAR PASS COUNT
2581  002044  132767  000200  177201  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
2582  002052  001403                  BEQ     67$             ;;YES,USE NON-APT SWITCH
2583  002054  012767  001254  177056  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
2584  002062                  67$:
2585                          .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
2586  002062  005737  000042          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
2587  002066  001012                  BNE     68$             ;;BRANCH IF YES
2588  002070  126727  177156  000001  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
2589  002076  001406                  BEQ     68$             ;;BRANCH IF YES
2590  002100  026727  177034  000176  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
2591  002106  001005                  BNE     69$             ;;BRANCH IF NO
2592  002110  104406                  GTSWR                   ;;GET SOFT-SWR SETTINGS
2593  002112  000403                  BR      69$
2594  002114  112767  000001  177012  68$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
2595  002122                  69$:
```

```
2596  002122  012767  016754  175654  START1: MOV     #BUSER,ERRVEC   ;SET UP THE BUS ERROR VECTOR
2597  002130  012767  000340  175650          MOV     #340,ERRVEC+2
2598  002136  012767  017016  175644          MOV     #RESERR,RESVEC  ;SET UP THE RSVD INSTR VECTOR
2599  002144  012767  000340  175640          MOV     #340,RESVEC+2
2600  002152  005767  020152                  TST     TITFLG          ;HAVE WE TYPED TITLE ONCE ?
2601  002156  001012                          BNE     1$              ;BR IF YES
2602  002160  104401                          TYPE                    ;GO TYPE PROGRAM TITLE
2603  002162  024632                          TITLE
2604  002164  005167  020140                  COM     TITFLG          ;SET FLAG - TYPE TITLE ONLY ONCE PER LOAD
2605  002170  032777  000001  176742          BIT     #BIT0,@SWR      ;DO WE WANT TO AUTOSIZE?
2606  002176  001002                          BNE     1$              ;BRANCH IF NOT.
2607  002200  004767  012740                  JSR     PC,AUTOSZ       ;GO AUTOSIZE.
2608  002204  005767  017476          1$:     TST     VCFLG           ;START AT 200 ??
2609  002210  001413                          BEQ     13$             ;BR IF NOT
2610  002212  032777  000001  176720          BIT     #BIT0,@SWR      ;ARE PARAMETERS TO BE INPUT MANUALLY?
2611  002220  001003                          BNE     9$              ;BRANCH IF YES
2612  002222  016700  017706                  MOV     ADRVEC,R0       ;OTHERWISE, GET ADDRESSES BETWEEN VECOTRS FROM AUTOSIZER
2613  002226  000402                          BR      10$
2614  002230  004767  013656          9$:     JSR     PC,INPARA       ;GO ASK FOR PARAMETERS
2615  002234  005067  017446          10$:    CLR     VCFLG           ;RE INIT START FLAG
2616  002240  005767  020106          13$:    TST     RETFLG          ;RETURN TO ECHO TESTS ?
2617  002244  001402                          BEQ     11$             ;BR IF NOT
2618  002246  000167  002442                  JMP     ECHO1           ;RETURN TO ECHO TEST START-UP
2619  002252  005767  020062          11$:    TST     DPFLG           ;RETURN TO DATA PATTERNS TEST ?
2620  002256  001402                          BEQ     12$             ;BR IF NOT
2621  002260  000167  003630                  JMP     EXPAT1          ;GO BACK TO DATA PATTERNS TESTS
2622  002264  005700                  12$:    TST     R0              ;USE DEFAULT PARAMETERS ?
2623  002266  001407                          BEQ     START2          ;BR IF YES
2624  002270  022700  177777                  CMP     #-1,R0          ;CHANGE DH SELECT PARAM ONLY ?
2625  002274  001002                          BNE     2$              ;BR IF NOT
2626  002276  000167  014004                  JMP     INPAR3          ;GO ASK FOR SELECT PARAM.
2627  002302  000167  013710          2$:     JMP     INPAR           ;GO ASK FOR ALL PARAMETERS
2628
2629  002306  012767  021604  020006  START2: MOV     #DHADTB-2,ADPTR ;GET POINTER TO ADDRESS TABLE
2630  002314  012767  021644  020002          MOV     #DHVCTB-2,VCPTR ;GET POINTER TO VECTOR TABLE
2631  002322  012767  021706  017776          MOV     #BRLVL-2,BRPTR  ;GET POINTER TO BR LEVEL TABLE
2632  002330  012767  177777  017604          MOV     #-1,DHNUM       ;START WITH DH #00
2633  002336  012767  000001  017154          MOV     #1,SELMSK       ;SET UP DH11 BIT TEST MARKER
2634
2635  002344  005267  017572          RESTRT: INC     DHNUM           ;GENERATE DH11 DEV NUMBER
2636  002350  062767  000002  017744          ADD     #2,ADPTR        ;UPDATE TABLE POINTERS
2637  002356  062767  000002  017740          ADD     #2,VCPTR
2638  002364  062767  000002  017734          ADD     #2,BRPTR
2639  002372  036767  017122  017122          BIT     SELMSK,DHSEL    ;TEST FOR SELECTED DH11
2640  002400  001004                          BNE     RSTRTA          ;BR IF SELECTED FOR TEST
2641  002402  006367  017112          REST1:  ASL     SELMSK          ;SHIFT MARKER TO TEST NEXT DH11
2642  002406  001737                          BEQ     START2          ;BR IF 16 TESTED - START OVER
2643  002410  000755                          BR      RESTRT          ;GO TEST IF THIS ONE SELECTED
2644  002412  017767  017704  017074  RSTRTA: MOV     @ADPTR,DHADR    ;SET UP DH11 ADDRESS
2645  002420  017767  017700  017070          MOV     @VCPTR,DHVCT    ;SET UP THE DH11 VECTOR ENTRY
2646  002426  017767  017674  017504          MOV     @BRPTR,DHRLVL   ;GET BR LEVEL VALUES
2647  002434  004567  012376                  JSR     R5,SUNUM        ;GO SET DH NUMBER IN THE MESSAGE BUFFER
2648  002440  022142                          DHNUM
2649  002442  024721                          TITLE2+20
2650  002444  104401                          TYPE                    ;GO PRINT "TESTING DH11 #XX"
2651  002446  024701                          TITLE2
```

```
2652  002450  012767  002450  176430          MOV     #.,$LPADR       ;INIT SCOPE RETURN
2653                                  ;;*****************************************************************
2654                                  ;*TEST 1         SUB-PROGRAM 1 - DATA RELIABILITY TESTS
2655                                  ;;*****************************************************************
2656  002456  000004                  TST1:   SCOPE
2657  002460  012767  000001  176534          MOV     #1,$TIMES       ;;DO 1 ITERAT'ON
2658  002466  004767  014436          STDH1:  JSR     PC,CLSTAT       ;GO CLEAR THE STATISTICS TABLES
2659  002472  004767  015016                  JSR     PC,KYBD1        ;GO SET UP FOR KEYBOARD INTRS.
2660  002476  005067  017030                  CLR     QUICK           ;INIT THE QUICK TEST FLAG
2661  002502  005067  017022                  CLR     DRPLIN          ;INIT DROPPED LINE FLAGS
2662  002506  005067  017432                  CLR     LINE            ;INIT LINE NO. TO 00
2663  002512  016702  017000                  MOV     DHVCT,R2        ;SET UP THE VECTORS
2664  002516  012722  003534                  MOV     #RINT1,(R2)+    ;GO TO RINT1 ON RCVR INTR
2665  002522  116722  017412                  MOVB    DHRLVL,(R2)+    ;
2666  002526  105722                          TSTB    (R2)+
2667  002530  012722  003030                  MOV     #TINT1,(R2)+    ;GO TO TINT1 ON XMITTR INTR
2668  002534  116712  017401                  MOVB    DHTLVL,(R2)
2669  002540  016701  016750                  MOV     DHADR,R1        ;SET UP DEVICE ADDRESS
2670
2671  002544  004767  012166          NEWLIN: JSR     PC,SELINE       ;GO SELECT NEW LINE FOR TEST
2672  002550  000401                          BR      1$              ;BR IF TESTED ALL SELECTED LINES
2673  002552  000410                          BR      2$              ;BR IF NOT DONE
2674  002554  122767  000001  176470  1$:     CMPB    #1,     $ENV    ; UNDER APT?
2675  002562  001002                          BNE     3$              ; NO-BRANCH
2676  002564  000167  002012                  JMP     ENDA            ; YES- SKIP STATISTICS REPORT
2677  002570  000167  001616          3$:     JMP     PRSTAT          ;GO CHECK AND REPORT STATISTICS
2678  002574  004567  012236          2$:     JSR     R5,SUNUM        ;PUT LINE NO. IN MSG
2679  002600  022144                          LINE
2680  002602  025561                          STMSG2+20
2681  002604  104401                          TYPE                    ;TYPE LINE NO. BEING TESTED
2682  002606  025541                          STMSG2
2683  002610  012767  021536  016754          MOV     #LPRTAB,LPRPTR  ;SET UP LPR TABLE POINTER
2684
2685  002616  004567  014324          NEWLPR: JSR     R5,SETLPR       ;GO SET UP LPR CONSTANT
2686  002622  000750                          BR      NEWLIN          ;BR IF DONE ALL BAUD RATES AT THIS LINE
2687  002624  012767  177760  016742          MOV     #177760,CHRCNT  ;INIT CHAR COUNT
2688  002632  012767  177777  016736          MOV     #-1,CLSEL       ;INIT CHR LNGTH SELECT CODE
2689
2690  002640  005067  016670                  CLR     QUICKX          ;INIT QUICK TEST EXIT FLAG
2691  002644  004567  014362          NEWCL:  JSR     R5,SETCL        ;GO SET UP THE CHAR LENGTH
2692  002650  000762                          BR      NEWLPR          ;BR IF DONE ALL FOUR LENGTHS
2693  002652  005067  016722                  CLR     PARBIT          ;INIT PARITY SELECT BIT CODE
2694
2695  002656  004567  014510          NEWPAR: JSR     R5,SETPAR       ;GO SET PARITY SELECT
2696  002662  000770                          BR      NEWCL           ;BR IF DONE ALL COMBOS
2697
2698  002664  004767  011750          DHST1:  JSR     PC,DHSET1       ;GO SET UP FOR TESTING THIS LINE
2699  002670  056761  016632  000012          BIS     LINMSK,BAR(R1)  ;ACTIVATE THE SELECTED LINE
2700  002676  004767  016516                  JSR     PC,CHPS1        ;GO CLEAR PSW
2701
2702
2703  002702  012767  000200  017422          MOV     #200,TIMEA      ;INIT TIMER A
2704  002710  005067  017420                  CLR     TIMEB           ;INIT TIMER B
2705  002714  022767  000003  016660  1$:     CMP     #3,RDONE        ;BOTH RCVR AND XMITTR DONE ?
2706  002722  001435                          BEQ     3$              ;BR IF YES
2707  002724  004767  014156                  JSR     PC,TIMEIT       ;CALL THE TIMER
```

```
 2708  002730  000771                          BR      1$              ;TIMER STEPS AROUND THIS BRANCH IF
 2709                                                                  ;TIMEOJT OCCURS
 2710
 2711  002732  052777  004000  016554          BIS     #BIT11,@DHADR   ;CLEAR OUT THE DH11
 2712  002740  116700  017200                  MOVB    LINE,R0         ;GET LINE NO.
 2713  002744  006300                          ASL     R0              ;FORM TABLE INDEX
 2714  002746  016067  030160  176226          MOV     RTOTAL(R0),$TMP0;SAVE XMITTED COUNT
 2715  002754  016067  030220  176222          MOV     XTOTAL(R0),$TMP1;SAVE THE RCVD COUNT
 2716  002762  004567  012050                  JSR     R5,SUNUM        ;PUT LINE NO. IN MESSAGE
 2717  002766  022144                          LINE
 2718  002770  023467                          EM12+35
 2719  002772  012767  003002  176110          MOV     #2$,$LPERR      ;SET UP ERROR LOOP RETURN
 2720  003000  104012                          ERROR   12              ;LINE FAILED TO FINISH ON TIME - HUNG
 2721  003002  056767  016520  016520  2$:     BIS     LINMSK,DRPLIN   ;SET DROP FLAG
 2722  003010  012706  001100                  MOV     #STACK,SP       ;RESET STACK POINTER
 2723  003014  000653                          BR      NEWLIN          ;GO TRY ANOTHER LINE
 2724
 2725
 2726  003016  012711  004000          5$:     MOV     #BIT11,(R1)     ;CLEAR THE WORLD OUT IN THE DH11
 2727  003022  004767  001050                  JSR     PC,CKER         ;GO UPDATE THE DATA ERROR TABLES
 2728  003026  000713                          BR      NEWPAR          ;GO TRY NEXT PARITY COMBINATION
```

```
2729
2730                                    ;TRANSMITTER INTERRUPT SERVICE ROUTINE ONE
2731
2732  003030  032711  002000            TINT1:  BIT     #BIT10,(R1)     ;NON EX MEM ERROR ??
2733  003034  001432                            BEQ     2$              ;BR IF NOT
2734
2735  003036  011103                            MOV     (R1),R3         ;SAVE THE SCR
2736  003040  004767  016370                    JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2737  003044  012711  004000                    MOV     #BIT11,(R1)     ;CLEAR OUT THE DH11
2738  003050  116704  017070                    MOVB    LINE,R4         ;SET UP THE S/B DATA
2739  003054  042703  175760                    BIC     #175760,R3      ;CLEAR OUT SUPERFLUOUS BITS
2740  003060  010102                            MOV     R1,R2           ;SET UP REGADR
2741  003062  004767  012034                    JSR     PC,SUER1        ;GO SET UP ERROR INFO
2742  003066  004567  011744                    JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2743  003072  022144                            LINE
2744  003074  022500                            EM1+44
2745  003076  012767  003106  176004            MOV     #1$,$LPERR      ;SET UP THE ERROR LOOP RETURN
2746  003104  104001                            ERROR   1               ;NON EX MEM ERROR
2747  003106  022626                    1$:     CMP     (SP)+,(SP)+     ;POP THE STACK
2748  003110  056767  016412  016412            BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2749  003116  000167  177422                    JMP     NEWLIN          ;GO TRY NEXT LINE
2750
2751  003122  011103                    2$:     MOV     (R1),R3         ;GET THE SCR REG CONTENTS
2752  003124  100433                            BMI     4$              ;BR IF XMIT DONE SET
2753
2754  003126  004767  016302                    JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2755  003132  012711  004000                    MOV     #BIT11,(R1)     ;CLEAR THE DH11 - FATAL ERROR
2756  003136  012704  100000                    MOV     #BIT15,R4       ;SET UP S/B DATA
2757  003142  156704  016776                    BISB    LINE,R4
2758  003146  042703  077760                    BIC     #77760,R3       ;CLEAR OUT SUPERFLUOUS BITS
2759  003152  010102                            MOV     R1,R2           ;SET UP REGADR
2760  003154  004767  011742                    JSR     PC,SUER1        ;GO SET UP ERROR INFO
2761  003160  004567  011652                    JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2762  003164  022144                            LINE
2763  003166  022662                            EM2+54
2764  003170  012767  003200  175712            MOV     #3$,$LPERR      ;SET UP ERROR LOOP RETURN
2765  003176  104002                            ERROR   2               ;XMITTR FALSE INTERRUPT
2766  003200  022626                    3$:     CMP     (SP)+,(SP)+     ;POP THE STACK
2767  003202  056767  016320  016320            BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2768  003210  000167  177330                    JMP     NEWLIN          ;GO TRY NEXT LINE
2769
2770  003214  005761  000012            4$:     TST     BAR(R1)         ;DID BAR BIT CLEAR ??
2771  003220  001432                            BEQ     6$              ;BR IF YES
2772
2773  003222  004767  016206                    JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2774  003226  016103  000012                    MOV     BAR(R1),R3      ;GET THE WAS DATA
2775  003232  012711  004000                    MOV     #BIT11,(R1)     ;CLEAR THE DH11
2776  003236  005004                            CLR     R4              ;SET UP S/B DATA
2777  003240  010102                            MOV     R1,R2           ;SET UP REGADR
2778  003242  062702  000012                    ADD     #BAR,R2
2779  003246  004767  011650                    JSR     PC,SUER1        ;GO SET UP ERROR INFO
2780  003252  004567  011560                    JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2781  003256  022144                            LINE
2782  003260  022742                            EM3+55
2783  003262  012767  003272  175620            MOV     #5$,$LPERR      ;SAVE THE ERROR LOOP RETURN
2784  003270  104003                            ERROR   3               ;BUFFER ACTIVE REG FAILED TO CLEAR
```

```
2785  003272  022626                  5$:   CMP     (SP)+,(SP)+     ;POP GOES THE STACK
2786  003274  056767  016226  016226        BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2787  003302  000167  177236              JMP     NEWLIN          ;GO TRY NEXT LINE
2788
2789  003306  005761  000010        6$:   TST     BCR(R1)         ;DID BYTE COUNT GO TO ZERO ??
2790  003312  001432                      BEQ     8$              ;BR IF YES
2791
2792  003314  004767  016114              JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2793  003320  016103  000010              MOV     BCR(R1),R3      ;GET THE WAS DATA
2794  003324  012711  004000              MOV     #BIT11,(R1)     ;CLEAR THE DH11
2795  003330  005004                      CLR     R4              ;SET UP S/B DATA
2796  003332  010102                      MOV     R1,R2           ;SET UP REGADR
2797  003334  062702  000010              ADD     #BCR,R2
2798  003340  004767  011556              JSR     PC,SUER1        ;GO SET UP THE ERROR INFO
2799  003344  004567  011466              JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2800  003350  022144                      LINE
2801  003352  023017                      EM4+52
2802  003354  012767  003364  175526      MOV     #7$,$LPERR      ;SET UP ERROR LOOP RETURN
2803  003362  104004                      ERROR   4               ;BYTE COUNT REG FAILED TO GO TO 000000
2804  003364  022626              7$:   CMP     (SP)+,(SP)+     ;POP GOES THE STACK
2805  003366  056767  016134  016134      BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2806  003374  000167  177144              JMP     NEWLIN          ;GO TRY NEXT LINE
2807
2808  003400  016103  000006        8$:   MOV     CAR(R1),R3      ;GET THE WAS DATA
2809  003404  016704  016164              MOV     CHRCNT,R4       ;SET UP S/B DATA
2810  003410  005404                      NEG     R4
2811  003412  062704  032754              ADD     #TBUF,R4
2812  003416  020304                      CMP     R3,R4           ;WAS CAR CORRECT ??
2813  003420  001425                      BEQ     10$             ;BR IF YES
2814
2815  003422  004767  016006              JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2816  003426  010102                      MOV     R1,R2           ;SET UP REGADR
2817  003430  062702  000006              ADD     #CAR,R2
2818  003434  004767  011462              JSR     PC,SUER1        ;GO SET UP ERROR INFO
2819  003440  004567  011372              JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2820  003444  022144                      LINE
2821  003446  023101                      EM5+57
2822  003450  012767  003460  175432      MOV     #9$,$LPERR      ;SET UP THE ERROR RETURN
2823  003456  104005                      ERROR   5               ;CURRENT ADDRESS REG NOT CORRECT
2824  003460  022626              9$:   CMP     (SP)+,(SP)+     ;POP THE STACK
2825  003462  056767  016040  016040      BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2826  003470  000167  177050              JMP     NEWLIN          ;GO TRY NEXT LINE
2827
2828  003474                        10$:
2829  003474  010346                      MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2830  003476  010446                      MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2831  003500  016703  016070              MOV     CHRCNT,R3
2832  003504  005403                      NEG     R3              ;CHAR COUNT IN R3
2833  003506  116704  016432              MOVB    LINE,R4         ;GET LINE NO.
2834  003512  006304                      ASL     R4              ;DOUBLE IT
2835  003514  060364  030220              ADD     R3,XTOTAL(R4)   ;UPDATE TOTAL XMIT COUNT
2836  003520  012604                      MOV     (SP)+,R4        ;;POP STACK INTO R4
2837  003522  012603                      MOV     (SP)+,R3        ;;POP STACK INTO R3
2838  003524  052767  000001  016050      BIS     #BIT0,RDONE     ;SET XMIT DONE FLAG
2839  003532  000002                      RTI                     ;RETURN TO WAIT LOOP
```

```
2840                                    ;RECEIVER INTERRUPT SERVICE ROUTINE ONE
2841
2842  003534  032711  040000          RINT1:  BIT     #BIT14,(R1)     ;SILO OVERFLOW ERROR ??
2843  003540  001431                          BEQ     2$              ;BR IF NOT
2844
2845  003542  004767  015666                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2846  003546  011103                          MOV     (R1),R3         ;GET THE WAS DATA
2847  003550  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
2848  003554  042703  177760                  BIC     #177760,R3      ;CLEAR JUNK
2849  003560  116704  016360                  MOVB    LINE,R4
2850  003564  004767  011332                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
2851  003570  004567  011242                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2852  003574  022144                          LINE
2853  003576  023150                          EM6+44
2854  003600  012767  003610  175302          MOV     #1$,$LPERR      ;SET UP ERROR LOOP RETURN
2855  003606  104006                          ERROR   6               ;SILO OVERFLOW - BAD,BAD,BAD !!!
2856  003610  022626                  1$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
2857  003612  056767  015710  015710          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2858  003620  000167  176720                  JMP     NEWLIN          ;GO TRY NEXT LINE
2859
2860  003624  105711                  2$:     TSTB    (R1)            ;CHAR AVAIL SET ??
2861  003626  100434                          BMI     4$              ;BR IF YES
2862
2863  003630  004767  015600                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2864  003634  011103                          MOV     (R1),R3         ;GET WAS DATA
2865  003636  042703  177560                  BIC     #177560,R3      ;CLEAN IT UP
2866  003642  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR DH11
2867  003646  012704  000200                  MOV     #BIT07,R4       ;SET UP S/B DATA
2868  003652  156704  016266                  BISB    LINE,R4
2869  003656  010102                          MOV     R1,R2           ;SET UP REGADR
2870  003660  004767  011236                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
2871  003664  004567  011146                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2872  003670  022144                          LINE
2873  003672  023224                          EM7+51
2874  003674  012767  003704  175206          MOV     #3$,$LPERR      ;SET UP THE ERROR LOOP RETURN
2875  003702  104007                          ERROR   7               ;RECEIVER FALSE INTERRUPT
2876  003704  022626                  3$:     CMP     (SP)+,(SP)+     ;POP GOES THE SP
2877  003706  056767  015614  015614          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2878  003714  000167  176624                  JMP     NEWLINE         ;GO TRY NEXT LINE
2879
2880  003720  016167  000002  175254  4$:     MOV     NRC(R1),$TMP0   ;SAVE THE DATA RECEIVED
2881  003726  100431                          BMI     6$              ;BR IF IT WAS VALID DATA
2882
2883  003730  004767  015500                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2884  003734  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
2885  003740  162767  030474  024522          SUB     #RBUF,RBFPTR    ;WHICH CHAR WAS IT ??
2886  003746  016702  024516                  MOV     RBFPTR,R2       ;SAVE CHAR NUMBER
2887  003752  004767  011140                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
2888  003756  004567  011054                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2889  003762  022144                          LINE
2890  003764  023274                          EM10+45
2891  003766  012767  003776  175114          MOV     #5$,$LPERR      ;SET UP ERROR RETURN
2892  003774  104010                          ERROR   10              ;RECEIVED INVALID DATA
2893  003776  022626                  5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
2894  004000  056767  015522  015522          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2895  004006  000167  176532                  JMP     NEWLIN          ;GO TRY ANOTHER LINE
```

```
 2896
 2897  004012                                  6$:
 2898  004012  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 2899  004014  010446                                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 2900  004016  016777  175160  024444                  MOV     $TMP0,@RBFPTR   ;STORE CHAR IN THE BUFFER
 2901  004024  062767  000002  024436                  ADD     #2,RBFPTR       ;UPDATE THE POINTER
 2902  004032  026767  015546  024430                  CMP     RBFEND,RBFPTR   ;END OF BUFFER ??
 2903  004040  001013                                  BNE     7$              ;BR IF NOT
 2904  004042  016703  015526                          MOV     CHRCNT,R3       ;GET CHAR COUNT
 2905  004046  005403                                  NEG     R3
 2906  004050  116704  016070                          MOVB    LINE,R4         ;GET THE LINE NO.
#2907  004054  006304                                  ASL     R4              ;DOUBLE IT
#2908  004056  060364  030160                          ADD     R3,RTOTAL(R4)   ;UPDATE TOTAL RECEIVED COUNT
 2909  004062  052767  000002  015512                  BIS     #BIT1,RDONE     ;SET THE RCVR DONE BLAG
 2910  004070                                  7$:
 2911  004070  012604                                  MOV     (SP)+,R4        ;;POP STACK INTO R4
 2912  004072  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
 2913  004074  000002                                  RTI                     ;RETURN TO WAIT LOOP
```

```
2914                                    ;THIS ROUTINE IS CALLED TO CHECK THE RECEIVED DATA, REPORT ALL ERRORS,
2915                                    ;AND UPDATE THE STATISTICS TABLE ENTRIES FOR ALL LINES ACTIVE
2916
2917  004076  012767  030474  024364   CKER:   MOV     #RBUF,RBFPTR    ;SET UP POINTERS
2918  004104  012767  032754  024360           MOV     #TBUF,TBFPTR
2919  004112  012767  030260  024340           MOV     #DATERR,DEPTR   ;SET UP POINTERS TO STATISTICS TABLES
2920  004120  012767  030320  024334           MOV     #PARERR,PEPTR
2921  004126  012767  030360  024330           MOV     #OVRERR,ORPTR
2922  004134  012767  030420  024324           MOV     #FRMERR,FRPTR
2923  004142  116705  015776                   MOVB    LINE,R5         ;GET LINE NO. AND DOUBLE IT
2924  004146  006305                           ASL     R5
2925  004150  060567  024304                   ADD     R5,DEPTR        ;POINT TO CORRECT LINE ENTRY IN TABLE
2926  004154  060567  024302                   ADD     R5,PEPTR
2927  004160  060567  024300                   ADD     R5,ORPTR
2928  004164  060567  024276                   ADD     R5,FRPTR
2929
2930  004170  117704  024276           1$:     MOVB    @TBFPTR,R4      ;GET THE S/B DATA
2931  004174  000304                           SWAB    R4              ;PUT LINE NO. IN HIGH BYTE
2932  004176  105004                           CLRB    R4
2933  004200  156704  015740                   BISB    LINE,R4
2934  004204  000304                           SWAB    R4
2935  004206  052704  100000                   BIS     #BIT15,R4       ;AND FINALLY THE VALID DATA BIT
2936  004212  017703  024252                   MOV     @RBFPTR,R3      ;GET THE WAS DATA
2937  004216  020304                           CMP     R3,R4           ;WAS = S/B ?????
2938  004220  001435                           BEQ     3$              ;BR IF YES
2939
2940  004222  010367  174772                   MOV     R3,$TMP7        ;SAVE THE WAS DATA
2941  004226  010146                           MOV     R1,-(SP)        ;SAVE THE DEVADR
2942  004230  016701  024234                   MOV     RBFPTR,R1       ;GET THE SBADR
2943  004234  016702  024232                   MOV     TBFPTR,R2       ;GET THE WASADR
2944  004240  010200                           MOV     R2,R0           ;GET XMIT BUFFER ADDR
2945  004242  162700  032754                   SUB     #TBUF,R0        ;GENERATE CHAR #
2946  004246  004767  000066                   JSR     PC,UPDER        ;GO CHECK AND UPDATE THE DATA ERROR TABLE
2947  004252  004767  010640                   JSR     PC,SUER2        ;GO SET UP ERROR INFO
2948  004256  004567  010554                   JSR     R5,SUNUM        ;GO PUT LINE NO. IN MSG
2949  004262  022144                           LINE
2950  004264  023427                           EM11+23
2951  004266  012767  004276  174614           MOV     #2$,$LPERR      ;SET UP ERROR RETURN
2952  004274  104011                           ERROR   11              ;DATA COMPARE ERROR OR PARITY,FRAMING
2953                                                                   ;OR OVERRUN
2954  004276  012601                   2$:     MOV     (SP)+,R1        ;RESTORE THE DEVADR
2955  004300  032767  070000  174712           BIT     #70000,$TMP7    ;ANY PARITY,OVERRUN, OR FRAMING ERROR
2956  004306  001402                           BEQ     3$              ;BR IF NOT
2957
2958  004310  004767  000036                   JSR     PC,SOFT         ;GO TAKE CARE OF SOFT ERROR REPORT
2959
2960  004314  005267  024152           3$:     INC     TBFPTR          ;UPDATE POINTERS
2961  004320  062767  000002  024142           ADD     #2,RBFPTR
2962  004326  026767  015252  024134           CMP     RBFEND,RBFPTR   ;COMPARED ALL CHARS ??
2963  004334  001315                           BNE     1$              ;BR IF NOT
2964
2965  004336  000207                           RTS     PC              ;RETURN TO WAIT LOOP
2966
2967  004340  120304                   UPDER:  CMPB    R3,R4           ;DATA BYTES CORRECT ??
2968  004342  001402                           BEQ     1$              ;BR IF YES
2969  004344  005277  024110                   INC     @DEPTR          ;COUNT THE DATA ERROR
```

```
(ZDHN-D MA(Y11 30A(1052)  27-DEC-78  15:31  PAGE 68
C:ZDHND.P11    27-DEC-78  15:28          T1      SUB-PROGRAM 1 - DATA RELIABILITY TESTS                    SEQ 0067

 2970  004350  000207                  1$:     RTS     PC              ;RETURN
 2971
 2972  004352  006367  174642          SOFT:   ASL     $TMP7           ;TEST FOR OVERRUN ERRORS
 2973  004356  100002                          BPL     1$              ;BR IF NONE
 2974  004360  005277  024100                  INC     @ORPTR          ;COUNT IT
 2975  004364  006367  174630          1$:     ASL     $TMP7           ;TEST FOR FRAMING ERRORS
 2976  004370  100002                          BPL     2$              ;BR IF NONE
 2977  004372  005277  024070                  INC     @FRPTR          ;COUNT IT
 2978  004376  006367  174616          2$:     ASL     $TMP7           ;TEST FOR PARITY ERRORS
 2979  004402  100002                          BPL     3$              ;BR IF NONE
 2980  004404  005277  024052                  INC     @PEPTR          ;COUNT IT
 2981  004410  000207                  3$:     RTS     PC              ;RETURN
 2982
 2983
```

```
2984                                    ;THIS ROUTINE IS CALLED TO PRINT OUT THE TEST STATISTICS
2985
2986  004412  012767  000001  005106  PRSTAT: MOV     #1,LINMSK       ;SET UP BIT TEST MARKER
2987  004420  005001                          CLR     R1              ;R1 CONTAINS THE LINE NO.
2988  004422  004567  010410                  JSR     R5,SUNUM        ;GO SET UP DH11 # IN STAT MESSAGE
2989
2990  004426  022142                          DHNUM
2991  004430  025522                          STMSG1+6
2992  004432  104401                          TYPE                    ;GO TYPE THE STATISTICS HEADER
2993  004434  025514                          STMSG1
2994  004436  104401                          TYPE                    ;TYPE HEADER
2995  004440  025617                          STMSG4
2996
2997  004442  036767  015060  015060  1$:     BIT     LINMSK,DRPLIN   ;DID THIS LINE GET DROPPED ?
2998  004450  001411                          BEQ     2$              ;BR IF NOT
2999
3000  004452  010167  174542                  MOV     R1,$TMP7        ;SAVE THE LINE NO.
3001  004456  004567  010354                  JSR     R5,SUNUM        ;GO PUT LINE NO. IN MESSAGE
3002  004462  001220                          $TMP7
3003  004464  025576                          STMSG3+10
3004  004466  104401                          TYPE
3005  004470  025566                          STMSG3
3006  004472  000436                          BR      3$              ;GO TEST NEXT LINE
3007
3008  004474  010102                  2$:     MOV     R1,R2           ;SET UP R2 WITH TABLE INDEX
3009  004476  006302                          ASL     R2
3010  004500  036767  015022  015016          BIT     LINMSK,LINSEL   ;WAS THIS LINE SELECTED ??
3011  004506  001430                          BEQ     3$              ;BR IF NOT
3012  004510  010167  174466                  MOV     R1,$TMP0        ;SET UP THE ERROR INFORMATION FROM
3013                                          ;THE TABLES INTO THE MESSAGE POINTERS
3014  004514  016267  030160  174462          MOV     RTOTAL(R2),$TMP1
3015  004522  016267  030220  174456          MOV     XTOTAL(R2),$TMP2
3016  004530  016267  030260  174452          MOV     DATERR(R2),$TMP3
3017  004536  016267  030320  174446          MOV     PARERR(R2),$TMP4
3018  004544  016267  030360  174442          MOV     OVRERR(R2),$TMP5
3019  004552  016267  030420  174436          MOV     FRMERR(R2),$TMP6
3020  004560  012767  004570  174322          MOV     #3$,$LPERR      ;RETURN TO 3$ AFTER PRINTING LINE
3021  004566  104013                          ERROR   13
3022  004570  005201                  3$:     INC     R1              ;STEP TO NEXT LINE
3023  004572  006367  014730                  ASL     LINMSK          ;SHIFT THE MARKER
3024  004576  001401                          BEQ     ENDA            ;BR IF ALL LINES REPORTED
3025  004600  000720                          BR      1$              ;GO BACK AND DO THIS LINE
3026
```

```
 3027  004602  000004                    ENDA:   SCOPE
 3028  004604  105067  174272                    CLRB    $TSTNM          ;RE-INIT TEST NUMBER FOR NEXT PASS
 3029  004610  012767  000240  004136            MOV     #240,$EOP       ;NOP THE SCOPE IN ENDPASS ROUTINE
 3030  004616  005267  015320                    INC     DHNUM           ;GENERATE NEW DH11 NUMBER
 3031  004622  062767  000002  015472            ADD     #2,ADPTR        ;UPDATE THE TABLE POINTERS
 3032  004630  062767  000002  015466            ADD     #2,VCPTR
 3033  004636  062767  000002  015462            ADD     #2,BRPTR
 3034  004644  006367  014650                    ASL     SELMSK          ;SHIFT MARKER TO TEST NEXT DH11
 3035  004650  001002                            BNE     1$              ;BR IF NOT TESTED ALL DH11'S
 3036  004652  000167  004076                    JMP     $EOP            ;JUMP TO EOP IF WE HAVE
 3037  004656  036767  014636  014636  1$:       BIT     SELMSK,DHSEL    ;IS THIS DH11 SELECTED ?
 3038  004664  001746                            BEQ     ENDA            ;BR IF NOT
 3039  004666  000167  175520                    JMP     RSTRTA          ;GO TEST THIS DH11
```

```
3040                                    .SBTTL SUB-PROGRAM 2 - SINGLE LINE ECHO/CABLE TESTS
3041
3042                                    ;       ************************************
3043                                    ;       *  SINGLE LINE ECHO TESTS          *
3044                                    ;       ************************************
3045
3046  004672  012767  177777  015452  ECHO:   MOV     #-1,RETFLG      ;SET RETURN FLAG - COME BACK
3047  004700  005067  015434           CLR     DPFLG           ;CLEAR PATTERNS TEST FLAG
3048  004704  005067  014776           CLR     VCFLG           ;INIT VECTOR SETUP FLAG
3049  004710  000167  174700           JMP     BEGINA          ;TO 'ECHO1' AFTER SETUP
3050
3051  004714  012767  160020  014572  ECHO1:  MOV     #160020,DHADR   ;SET UP DH11 DEFAULT ADDRESS
3052  004722  012767  000330  014566           MOV     #330,DHVCT      ;SET UP DH11 DEFAULT VECTOR
3053  004730  104401                           TYPE                    ;PRINT I.D. MESSAGE
3054  004732  026231                           ECMSG1                  ;'SINGLE LINE ECHO TEST - CONNECT
3055                                                                   ;TERMINAL TO TEST LINE'
3056  004734  000167  011256           JMP     INPAR           ;GO SET UP DEVICE AND VECTOR
3057                                                           ;ADDRESSES - COME BACK TO 'ECHO2'
3058
3059  004740  104401                   ECHO2:  TYPE                    ;GO ASK FOR TTY INPUT
3060  004742  026330                           ECMSG2                  ;'LINE # (00 - 17 OCTAL)'
3061  004744  104412                           RDOCT                   ;INPUT LINE NO. FM TTY
3062  004746  012667  015172           MOV     (SP)+,LINE      ;GET NO. TYPED
3063  004752  042767  177760  015164   BIC     #177760,LINE    ;CLEAR JUNK
3064  004760  016702  015160           MOV     LINE,R2         ;GET LINE NO.
3065  004764  005202                   INC     R2              ;CORRECT FOR SHIFT ROUTINE
3066  004766  012767  000001  014532   MOV     #1,LINMSK       ;INIT LINE SELECT BIT MASK
3067  004774  005302                   1$:     DEC     R2              ;COUNT ONE LINE CHECKED
3068  004776  001403                   BEQ     2$              ;BR IF DONE
3069  005000  006367  014522           ASL     LINMSK          ;SHIFT SELECT BIT
3070  005004  000773                   BR      1$              ;GO COUNT IT
3071  005006  004767  013212           2$:     JSR     PC,LPRIN        ;GO ASK FOR AND SET UP LINE PARAMETERS
3072
3073  005012  005767  015322           TST     DPFLG           ;DATA PATTERNS TEST ?
3074  005016  001401                   BEQ     3$              ;BR IF NOT
3075  005020  000207                   RTS     PC              ;RETURN TO PATTERNS TEST
3076
3077  005022  105067  021174           3$:     CLRB    EC2             ;CLEAR ECHO BUFFER
3078  005026  104401                           TYPE
3079  005030  027352                           SNMSG1                  ;'SEND MODE - Y OR N ?'
3080  005032  104410                   4$:     RDCHR                   ;GET CHAR TYPED
3081  005034  012600                           MOV     (SP)+,R0        ;GET CHAR TYPED
3082  005036  122700  000015           CMPB    #15,R0          ;WAS IT A <CR> ?
3083  005042  001405                   BEQ     5$              ;BR IF YES
3084  005044  110067  021152           MOVB    R0,EC2          ;ECHO WHAT WAS TYPED
3085  005050  104401                   TYPE
3086  005052  026222                   EC2
3087  005054  000766                   BR      4$              ;GO WAIT FOR TERMINATOR
3088  005056  105767  021140           5$:     TSTB    EC2             ;<CR> ONLY ??
3089  005062  001412                   BEQ     ECHO3           ;BR IF YES
3090  005064  122767  000116  021130   CMPB    #116,EC2        ;WAS IT AN 'N' ??
3091  005072  001406                   BEQ     ECHO3           ;BR IF YES
3092  005074  122767  000131  021120   CMPB    #131,EC2        ;WAS IT A 'Y' ??
3093  005102  001347                   BNE     3$              ;BR IF NOT ASK AGAIN
3094  005104  000167  000574           JMP     SENDP1          ;GO TO SEND ROUTINE
3095
```

```
 3096  005110  004767  014320          ECHO3:  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
 3097  005114  005067  015216                  CLR     CEXIT           ;INIT CONTROL-C EXIT FLAG
 3098  005120  005067  015240                  CLR     EXFLAG          ;CLEAR TEST EXIT FLAGS
 3099  005124  012767  120240  015006          MOV     #120240,DHRLVL  ;INIT FOR BR LEVEL 5
 3100  005132  016701  014356                  MOV     DHADR,R1        ;SET UP DEVICE ADDRESS
 3101  005136  012711  004000                  MOV     #BIT11,(R1)     ;CLEAR THE SELECTED DH11
 3102  005142  016700  014350                  MOV     DHVCT,R0        ;GET THE FIRST VECTOR ADDRESS
 3103  005146  012720  005462                  MOV     #RINT2,(R0)+    ;SET UP THE VECTORS
 3104  005152  116710  014762                  MOVB    DHRLVL,(R0)
 3105  005156  005720                          TST     (R0)+
 3106  005160  012720  005314                  MOV     #TINT2,(R0)+
 3107  005164  116710  014751                  MOVB    DHTLVL,(R0)
 3108  005170  016711  014750                  MOV     LINE,(R1)       ;SET THE LINE SELECT BITS
 3109  005174  012702  032754                  MOV     #TBUF,R2        ;INIT BUFFER POINTER
 3110  005200  052711  000100                  BIS     #BIT06,(R1)     ;ENABLE  RCVR INTRS
 3111  005204  016761  014360  000004          MOV     CURLPR,LPR(R1)  ;SET UP LINE PARAMETERS
 3112  005212  004567  007620                  JSR     R5,SUNUM        ;PUT LINE NO. IN MESSAGE
 3113  005216  022144                          LINE
 3114  005220  026406                          ECMSG3+20
 3115  005222  104401                          TYPE                    ;GIVE DIRECTIONS
 3116  005224  026366                          ECMSG3                  ;''GO TYPE CHARS ON TEST TERMINAL''
 3117  005226  104401                          TYPE                    ;PRINT DIRECTIONS
 3118  005230  026445                          ECMSG4                  ;''[CONTROL-C TO EXIT] [CONTROL-E TO ECHO BUFFER]''
 3119  005232  004767  014162                  JSR     PC,CHPS1        ;GO CLEAR PSW
 3120
 3121  005236  012767  000200  015066  DHWAIT: MOV     #200,TIMEA      ;INIT TIMER 'A'
 3122  005244  005067  015064                  CLR     TIMEB           ;INIT TIMER 'B'
 3123  005250  005767  015062          1$:     TST     CEXIT           ;CONTROL-C EXIT ??
 3124  005254  001015                          BNE     2$              ;BR IF YES
 3125  005256  004767  011624                  JSR     PC,TIMEIT       ;CALL TIMER
 3126  005262  000772                          BR      1$              ;BR IF NO TIMEOUT
 3127
 3128  005264  010167  173674                  MOV     R1,$REG1        ;SAVE DEVADR
 3129  005270  011167  173706                  MOV     (R1),$TMP0      ;SAVE CONTENT OF SCR
 3130  005274  052711  004000                  BIS     #BIT11,(R1)     ;CLEAR OUT THE DH11
 3131  005300  012767  005310  173602          MOV     #2$,$LPERR      ;SET ERROR LOOP RETURN
 3132  005306  104016                          ERROR   16              ;REPORT RCVR WAIT TIMEOUT
 3133  005310  000167  177424          2$:     JMP     ECHO2           ;GO RESTART
 3134
 3135
```

```
3136                                    ;TRANSMITTER INTERRUPT SERVICE ROUTINE TWO
3137
3138  005314  042711  120000            TINT2:  BIC     #BIT15+BIT13,(R1)       ;DISABLE XMIT INTRS
3139  005320  022767  000001  015036            CMP     #1,EXFLAG               ;CONTROL-C FLAG ?
3140  005326  001437                            BEQ     2$                      ;BR IF YES
3141  005330  022767  000003  015026            CMP     #3,EXFLAG               ;WAS BUFFER JUST DUMPED ?
3142  005336  001437                            BEQ     3$                      ;BR IF YES
3143  005340  022767  000002  015016            CMP     #2,EXFLAG               ;CONTROL-E FLAG ?
3144  005346  001403                            BEQ     1$                      ;BR IF YES
3145  005350  020227  034104                    CMP     R2,#TBUF+600.           ;BUFFER FULL ?
3146  005354  002434                            BLT     31$                     ;BR IF NOT
3147  005356  012767  000003  015000  1$:       MOV     #3,EXFLAG               ;SET DUMP FLAG
3148  005364  162702  032754                    SUB     #TBUF,R2                ;SET UP BYTE COUNT REG
3149  005370  005402                            NEG     R2
3150  005372  010261  000010                    MOV     R2,BCR(R1)
3151  005376  012761  032754  000006            MOV     #TBUF,CAR(R1)           ;SET UP CURRENT ADDR REG
3152  005404  012767  000200  014720            MOV     #200,TIMEA              ;INIT TIMER
3153  005412  052711  020000                    BIS     #BIT13,(R1)             ;ENABLE XMITTR INTR
3154  005416  016761  014104  000012            MOV     LINMSK,BAR(R1)          ;ACTIVATE LINE
3155  005424  000415                            BR      4$                      ;GO EXIT
3156
3157  005426  012767  177777  014702  2$:       MOV     #-1,CEXIT               ;SET CONTROL-C EXIT
3158  005434  000411                            BR      4$                      ;GO EXIT
3159
3160  005436  012702  032754            3$:     MOV     #TBUF,R2                ;RESET ECHO BUFFER POINTER
3161  005442  005067  014716                    CLR     EXFLAG                  ;INIT EXIT FLAG
3162  005446  012767  000200  014656    31$:    MOV     #200,TIMEA              ;INIT TIMER AGAIN
3163  005454  052711  000100                    BIS     #BIT06,(R1)             ;ENABLE RCVR INTR
3164
3165  005460  000002            4$:     RTI                                     ;RETURN TO MAINLINE
3166
3167
```

```
3168                                    ;RECEIVER INTERRUPT SERVICE ROUTINE TWO
3169
3170  005462  042711  000100            RINT2:  BIC     #BIT06,(R1)             ;DISABLE RCVR INTR
3171  005466  016167  000002  173506            MOV     NRC(R1),$TMP0           ;SAVE THE DATA TYPED
3172  005474  042767  177600  173500            BIC     #177600,$TMP0           ;CLEAR HIGH BYTE
3173  005502  022767  000003  173472            CMP     #3,$TMP0                ;CONTROL-C TYPED ??
3174  005510  001015                            BNE     1$                      ;BR IF NOT
3175
3176  005512  112767  000136  014646            MOVB    #136,ECBUF              ;SET UP TO ECHO CONTROL-C
3177  005520  112767  000103  014641            MOVB    #103,ECBUF+1
3178  005526  012761  177776  000010            MOV     #-2,BCR(R1)             ;SET UP BCR REG
3179  005534  012767  000001  014622            MOV     #1,EXFLAG               ;SET CONTROL-C FLAG
3180  005542  000444                            BR      4$                      ;GO OUT PUT CHAR TYPED
3181
3182  005544  022767  000005  173430  1$:       CMP     #5,$TMP0                ;WAS IT A CONTROL-E ??
3183  005552  001021                            BNE     2$                      ;BR IF NOT
3184
3185  005554  112767  000136  014604            MOVB    #136,ECBUF              ;SET UP TO ECHO CONTROL-E
3186  005562  112767  000105  014577            MOVB    #105,ECBUF+1
3187  005570  112722  000136                    MOVB    #136,(R2)+              ;PUT IN THE ECHO BUFFER
3188  005574  112722  000105                    MOVB    #105,(R2)+
3189  005600  012761  177776  000010            MOV     #-2,BCR(R1)             ;SET UP BYTE COUNT REG
3190  005606  012767  000002  014550            MOV     #2,EXFLAG               ;SET CONTROL-E FLAG
3191  005614  000417                            BR      4$                      ;GO EXIT
3192
3193  005616  022767  000012  173356  2$:       CMP     #12,$TMP0               ;WAS IT A LINE FEED ??
3194  005624  001003                            BNE     3$                      ;BR IF NO!
3195  005626  004767  012066                    JSR     PC,LDFILL               ;GO LOAD FILLERS
3196  005632  000410                            BR      4$                      ;GO EXIT
3197
3198  005634  116767  173342  014524  3$:       MOVB    $TMP0,ECBUF             ;SET UP CHAR TO ECHO
3199  005642  116722  173334                    MOVB    $TMP0,(R2)+
3200  005646  012761  177777  000010            MOV     #-1,BCR(R1)             ;OUTPUT ONE CHAR ONLY
3201  005654  012761  022366  000006  4$:       MOV     #ECBUF,CAR(R1)          ;SET UP CURRENT ADDR REG
3202  005662  012767  000200  014442            MOV     #200,TIMEA              ;INIT TIMER AGAIN
3203  005670  052711  020000                    BIS     #BIT13,(R1)             ;ENABLE XMITTR INTR
3204  005674  016761  013626  000012            MOV     LINMSK,BAR(R1)          ;ACTIVATE THE LINE
3205  005702  000002                            RTI                             ;RETURN TO MAINLINE
3206
3207
3208
```

```
3209  005704  104401                  SENDP1: TYPE                    ;ASK FOR DIRECTIONS
3210  005706  027402                          SNMSG2                  ;''TYPE SEND BUFFER - TERMINATE WITH CONTROL-C''
3211  005710  012705  032754                  MOV     #TBUF,R5        ;SET UP BUFFER POINTER
3212  005714  104410                  1$:     RDCHR                   ;GET CHAR
3213  005716  012600                          MOV     (SP)+,R0
3214  005720  110067  020276                  MOVB    R0,EC2          ;ECHO CHAR
3215  005724  104401                          TYPE
3216  005726  026222                          EC2
3217  005730  022700  000003                  CMP     #3,R0           ;WAS IT A CONTROL-C ??
3218  005734  001421                          BEQ     4$              ;BR IF YES
3219
3220  005736  026727  013632  000400          CMP     CHRCNT,#256.    ;BUFFER FULL ??
3221  005744  003015                          BGT     4$              ;BR IF YES
3222  005746  022700  000012                  CMP     #12,R0          ;WAS IT A LINE FEED ?
3223  005752  001010                          BNE     3$              ;BR IF NOT
3224
3225  005754  110025                          MOVB    R0,(R5)+        ;LOAD CHAR TYPED
3226  005756  116704  014450                  MOVB    FILLB,R4        ;GET FILLER COUNT
3227  005762  116725  014442          2$:     MOVB    FILLA,(R5)+     ;LOAD A FILLER
3228  005766  005304                          DEC     R4              ;COUNT IT
3229  005770  001374                          BNE     2$              ;BR IF NOT DONE
3230  005772  000750                          BR      1$              ;GET SOME MORE INPUT
3231
3232  005774  110025                  3$:     MOVB    R0,(R5)+        ;LOAD BUFFER
3233  005776  000746                          BR      1$              ;GO GET SOME MORE
3234
3235  006000  004767  011636          4$:     JSR     PC,SENDP2       ;GO XMIT THE BUFFER
3236  006004  105067  020212          5$:     CLRB    EC2             ;CLEAR ECHO BUFFER
3237  006010  104401                          TYPE
3238  006012  027462                          SNMSG3                  ;''CHANGE PARAMETERS- Y OR N''
3239  006014  104410                  6$:     RDCHR
3240  006016  012600                          MOV     (SP)+,R0        ;GET CHAR
3241  006020  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR> HE TYPED ??
3242  006024  001405                          BEQ     7$              ;BR IF IT WAS
3243  006026  110067  020170                  MOVB    R0,EC2          ;ECHO IT
3244  006032  104401                          TYPE
3245  006034  026222                          EC2
3246  006036  000766                          BR      6$              ;GO WAIT FOR TERMINATOR
3247
3248  006040  105767  020156          7$:     TSTB    EC2             ;<CR> ONLY ??
3249  006044- 001717                          BEQ     SENDP1          ;BR IF YES
3250  006046  122767  000116  020146          CMPB    #116,EC2        ;DID HE SAY NO ??
3251  006054  001713                          BEQ     SENDP1          ;BR IF HE DID
3252  006056  122767  000131  020136          CMPB    #131,EC2        ;DID HE SAY YES ??
3253  006064  001347                          BNE     5$              ;GO ASK ALL OVER AGAIN
3254  006066  000167  176646                  JMP     ECHO2           ;GO ASK FOR NEW PARAMETERS
```

```
                        .SBTTL SUB-PROGRAM THREE - DATA PATTERNS TESTS

                        ;                       ****************************
                        ;                       * DATA PATTERNS TESTS      *
                        ;                       ****************************

006072 012767 177777 014240 EXPAT:  MOV   #-1,DPFLG      ;SET PATTERNS TEST FLAG
006100 005067 014246        CLR     RETFLG       ;CLR ECHO TESTS FLAG
006104 005067 013576        CLR     VCFLG        ;CLEAR VECTOR SETUP FLAG
006110 000167 173500        JMP     BEGINA       ;GO SET UP RETURN TO 'EXPAT1'

006114 012767 160020 013372 EXPAT1: MOV   #160020,DHADR  ;SET UP DEFAULT DH11 ADDR
006122 012767 000330 013366 MOV     #330,DHVCT   ;AND VECTOR TOO
006130 104401               TYPE
006132 027521               DPMSG1               ;'DATA PATTERNS TESTS - CONNECT TEST JUMPAR'
006134 000167 010056        JMP     INPAR        ;GO GET SOME PARAMETERS RETURN TO EXPAT2

006140 004767 176574 EXPAT2: JSR    PC,ECHO2     ;GO GET REST OF THE PARAMETERS
006144 004767 013176        JSR     PC,SUCLMK    ;GO SET UP CHAR LENGTH MASK
006150 104401        1$:    TYPE
006152 027577               DPMSG2               ;'BUFFER SIZE ? (1-512)
006154 104413               RDDEC                ;GET THE SIZE TYPED
006156 012600               MOV     (SP)+,R0
006160 001406               BEQ     2$           ;BR IF DEFAULT TO 256. <CR>

006162 020027 001001        CMP     R0,#513.     ;TOO BIG ?
006166 002405               BLT     3$           ;BR IF NOT
006170 104401               TYPE
006172 027631               DPMSG3               ;'INVALID SIZE - TRY AGAIN'
006174 000765               BR      1$           ;GO ASK AGAIN

006176 012700 000400 2$:    MOV     #256.,R0     ;DEFAULT TO 256. BYTE BUFFER
006202 005400        3$:    NEG     R0           ;MAKE IT NEG BYTE COUNT
006204 010067 013364        MOV     R0,CHRCNT    ;SAVE IT FOR TEST

006210 012767 120240 013722 MOV     #120240,DHRLVL  ;SET BR LEVELS TO BR5
006216 016700 013274        MOV     DHVCT,R0     ;SET UP VECTORS
006222 012720 010252        MOV     #RINT3,(R0)+
006226 116710 013706        MOVB    DHRLVL,(R0)
006232 005720               TST     (R0)+
006234 012720 007630        MOV     #TINT3,(R0)+
006240 116710 013675        MOVB    DHTLVL,(R0)
```

```
3297  006244  104407                    EXPAT3: CKSWR                   ;TEST FOR CHANGE IN SOFT-SWR
3298  006246  005067  014074                    CLR     PATFLG          ;CLEAR <CR> SEQUENCE FLAG
3299  006252  105067  017744                    CLRB    EC2             ;CLEAR ECHO BUFFER
3300  006256  016701  013232                    MOV     DHADR,R1        ;INIT R1 TO POINT TO SCR REG
3301  006262  104401                            TYPE
3302  006264  027666                            DPMSG4                  ;'PATTERN TYPE ? (A,U,D,R,S, OR B)''
3303  006266  104410                    7$:     RDCHR
3304  006270  012600                            MOV     (SP)+,R0        ;GET WHAT HE TYPED
3305  006272  120027  000015                    CMPB    R0,#15          ;WAS IT A <CR> ??
3306  006276  001407                            BEQ     9$              ;BR IF YES
3307  006300  010067  014040                    MOV     R0,DATPAT
3308  006304  110067  017712                    MOVB    R0,EC2          ;ECHO IT
3309  006310  104401                            TYPE
3310  006312  026222                            EC2
3311  006314  000764                            BR      7$              ;GO WAIT FOR TERMINATOR
3312
3313  006316  104401                    9$:     TYPE
3314  006320  027740                            DPMSG5                  ;''SET SR07=1 TO LOCK ON TEST PATTERN''
3315  006322  105767  017674                    TSTB    EC2             ;<CR> ONLY ??
3316  006326  001005                            BNE     8$              ;BR IF NOT
3317  006330  012767  000015  014010            MOV     #15,PATFLG
3318  006336  000167  000506                    JMP     DPATCR          ;GO SEQUENCE A,U,D,R PATTERNS
3319
3320  006342  022767  000101  013774    8$:     CMP     #101,DATPAT     ;ALTERNATING 1/0 ?
3321  006350  001002                            BNE     1$              ;BR IF NOT
3322  006352  000167  000102                    JMP     DPATA           ;GO DO IT
3323
3324  006356  022767  000125  013760    1$:     CMP     #125,DATPAT     ;UP COUNT PATTERN ?
3325  006364  001002                            BNE     2$              ;BR IF NOT
3326  006366  000167  000164                    JMP     DPATU           ;GO DO IT
3327
3328  006372  022767  000104  013744    2$:     CMP     #104,DATPAT     ;DOWN COUNT PATTERN ?
3329  006400  001002                            BNE     3$              ;BR IF NOT
3330  006402  000167  000246                    JMP     DPATD           ;GO DO IT
3331
3332  006406  022767  000122  013730    3$:     CMP     #122,DATPAT     ;RANDOM PATTERN ?
3333  006414  001002                            BNE     4$              ;BR IF NOT
3334  006416  000167  000330                    JMP     DPATR           ;GO DO IT
3335
3336  006422  022767  000123  013714    4$:     CMP     #123,DATPAT     ;SINGLE CHAR PATTERN ?
3337  006430  001002                            BNE     5$              ;BR IF NOT
3338  006432  000167  000474                    JMP     DPATS           ;GO DO IT
3339
3340  006436  022767  000102  013700    5$:     CMP     #102,DATPAT     ;TYPE IN BUFFER ?
3341  006444  001002                            BNE     6$              ;BR IF NOT
3342  006446  000167  000620                    JMP     DPATB           ;GO DO IT
3343
3344  006452  104401                    6$:     TYPE
3345  006454  030002                            DPMSG6                  ;''INVALID PATTERN - TRY AGAIN''
3346  006456  000672                            BR      EXPAT3          ;GO ASK AGAIN
3347
```

```
3348  006460  005067  013656          DPATA:  CLR     DATCNT          ;INIT ITERATION COUNTER
3349  006464  004767  012440          1$:     JSR     PC,SUPATA       ;GO SET UP THE PATTERN
3350  006470  004767  001016                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
3351  006474  005267  013642                  INC     DATCNT          ;COUNT IT
3352  006500  026767  013650  013634          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES
3353  006506  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
3354
3355  006510  016767  013626  172444          MOV     DATCNT,$REGO    ;SAVE ITERATION COUNT
3356  006516  005067  013620                  CLR     DATCNT          ;INIT COUNTER
3357  006522  012767  006532  172360          MOV     #2$,$LPERR      ;COME BACK TO 2$
3358  006530  104017                          ERROR   17              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
3359  006532  022767  000015  013606  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
3360  006540  001001                          BNE     3$              ;BR IF NOT
3361  006542  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
3362  006544  105777  172370          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
3363  006550  100745                          BMI     1$              ;BR IF YES
3364  006552  000167  177466                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
3365
3366  006556  005067  013560          DPATU:  CLR     DATCNT          ;INIT ITERATION COUNTER
3367  006562  004767  012370          1$:     JSR     PC,SUPATU       ;GO SET UP THE PATTERN
3368  006566  004767  000720                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
3369  006572  005267  013544                  INC     DATCNT          ;COUNT IT
3370  006576  026767  013552  013536          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES ?
3371  006604  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
3372
3373  006606  016767  013530  172346          MOV     DATCNT,$REGO    ;SAVE ITERATION COUNT
3374  006614  005067  013522                  CLR     DATCNT          ;INIT COUNTER
3375  006620  012767  006630  172262          MOV     #2$,$LPERR      ;COME BACK TO 2$
3376  006626  104020                          ERROR   20              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
3377  006630  022767  000015  013510  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
3378  006636  001001                          BNE     3$              ;BR IF NOT
3379  006640  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
3380  006642  105777  172272          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
3381  006646  100745                          BMI     1$              ;BR IF YES
3382  006650  000167  177370                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
3383
3384  006654  005067  013462          DPATD:  CLR     DATCNT          ;INIT ITERATION COUNTER
3385  006660  004767  012322          1$:     JSR     PC,SUPATD       ;GO SET UP THE PATTERN
3386  006664  004767  000622                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
3387  006670  005267  013446                  INC     DATCNT          ;COUNT IT
3388  006674  026767  013454  013440          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES
3389  006702  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
3390
3391  006704  016767  013432  172250          MOV     DATCNT,$REGO    ;SAVE ITERATION COUNT
3392  006712  005067  013424                  CLR     DATCNT          ;INIT COUNTER
3393  006716  012767  006726  172164          MOV     #2$,$LPERR      ;COME BACK TO 2$
3394  006724  104021                          ERROR   21              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
3395  006726  022767  000015  013412  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
3396  006734  001001                          BNE     3$              ;BR IF NOT
3397  006736  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
3398  006740  105777  172174          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
3399  006744  100745                          BMI     1$              ;BR IF YES
3400  006746  000167  177272                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
3401
3402  006752  005067  013364          DPATR:  CLR     DATCNT          ;INIT ITERATION COUNTER
3403  006756  004767  012256          1$:     JSR     PC,SUPATR       ;GO SET UP THE PATTERN
```

```
3404  006762  004767  000524                JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
3405  006766  005267  013350                INC     DATCNT          ;COUNT IT
3406  006772  026767  013356  013342        CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES
3407  007000  001366                        BNE     1$              ;BR IF NOT DO IT AGAIN
3408
3409  007002  016767  013334  172152        MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
3410  007010  005067  013326                CLR     DATCNT          ;INIT COUNTER
3411  007014  012767  007024  172066        MOV     #2$,$LPERR      ;COME BACK TO 2$
3412  007022  104022                        ERROR   22              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
3413  007024  022767  000015  013314  2$:   CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
3414  007032  001001                        BNE     3$              ;BR IF NOT
3415  007034  000207                        RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
3416  007036  105777  172076          3$:   TSTB    @SWR            ;LOCK ON THIS PATTERN ??
3417  007042  100745                        BMI     1$              ;BR IF YES
3418  007044  000167  177174                JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
3419
3420  007050  012767  000101  013266  DPATCR: MOV    #101,DATPAT     ;FLAG 1/0 PATTERN
3421  007056  004767  177376                JSR     PC,DPATA        ;CALL FOR 1/0 PATTERN
3422  007062  012767  000125  013254        MOV     #125,DATPAT     ;FLAG UP COUNT PATTERN
3423  007070  004767  177462                JSR     PC,DPATU        ;CALL FOR UP COUNT PATTERN
3424  007074  012767  000104  013242        MOV     #104,DATPAT     ;FLAG DOWN COUNT PATTERN
3425  007102  004767  177546                JSR     PC,DPATD        ;CALL FOR DOWN COUNT PATTERN
3426  007106  012767  000122  013230        MOV     #122,DATPAT     ;FLAG RANDOM DATA PATTERN
3427  007114  004767  177632                JSR     PC,DPATR        ;CALL FO RANDOM PATTERN
3428  007120  105777  172014                TSTB    @SWR            ;LOCK ON ALL FOUR PATTERNS
3429  007124  100751                        BMI     DPATCR          ;BR IF YES
3430  007126  000167  177112                JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3431
3432  007132  105067  017064          DPATS: CLRB    EC2             ;CLEAR THE ECHO BUFFER
3433  007136  005067  013200                CLR     DATCNT          ;INIT ITERATION COUNTER
3434  007142  104401                        TYPE
3435  007144  030042                        DPMSG7                  ;'TYPE SINGLE TEST CHAR'
3436  007146  104410                  3$:   RDCHR                   ;GET CHAR
3437  007150  012600                        MOV     (SP)+,R0        ;GET WHAT HE TYPED
3438  007152  122700  000015                CMPB    #15,R0          ;WAS IT A <CR> ??
3439  007156  001407                        BEQ     4$              ;BR IF YES
3440  007160  010067  013164                MOV     R0,SINGLE       ;SAVE IT FOR LOADING BUFFER
3441  007164  110067  017032                MOVB    R0,EC2          ;ECHO IT ON TTY
3442  007170  104401                        TYPE
3443  007172  026222                        EC2
3444  007174  000764                        BR      3$              ;GO WAIT FOR TERMINATOR
3445
3446  007176  105767  017020          4$:   TSTB    EC2             ;WAS SINGLE CHAR A <CR> ??
3447  007202  001003                        BNE     1$              ;BR IF NOT A <CR> ONLY
3448  007204  012767  000015  013136        MOV     #15,SINGLE      ;SET UP TO LOAD ALL <CR>'S
3449  007212  004767  012102            1$: JSR     PC,SUPATS       ;GO SET IT UP IN BUFFER
3450  007216  004767  000270                JSR     PC,DHST2        ;GO EXECUTE IT ON DH11
3451  007222  005267  013114                INC     DATCNT          ;COUNT ONE TIME
3452  007226  026767  013122  013106        CMP     PATLIM,DATCNT   ;DONE REQUIRED ITERATIONS ?
3453  007234  001366                        BNE     1$              ;BR IF NOT
3454
3455  007236  016767  013100  171716        MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
3456  007244  005067  013072                CLR     DATCNT          ;INIT ITERATION COUNTER
3457  007250  012767  007260  171632        MOV     #2$,$LPERR      ;COME BACK TO 2$ ALWAYS
3458  007256  104023                        ERROR   23              ;REPORT DONE SINGLE CHAR PATTERN
3459  007260  105777  171654          2$:   TSTB    @SWR            ;LOCK ON THIS PATTERN ??
```

```
3460  007264  100752                    BMI     1$              ;BR IF YES
3461  007266  000167  176752            JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3462
```

```
3463  007272  005067  013044          DPATB:  CLR     DATCNT          ;INIT ITERATION COUNTER
3464  007276  104401                          TYPE
3465  007300  030075                          DPMSGA                  ;''TYPE IN BUFFER - TERMINATE WITH CONTROL-C''
3466  007302  012705  032754                  MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
3467  007306  104410                  1$:     RDCHR                   ;GET A CHAR
3468  007310  012667  013034                  MOV     (SP)+,SINGLE    ;SAVE IT
3469  007314  116767  013030  016700          MOVB    SINGLE,EC2      ;ECHO IT
3470  007322  104401                          TYPE
3471  007324  026222                          EC2
3472
3473  007326  022767  000003  013014          CMP     #3,SINGLE       ;WAS IT A CONTROL-C ??
3474  007334  001423                          BEQ     3$              ;BR IF YES
3475
3476  007336  020527  033755                  CMP     R5,#TBUF+513.   ;BUFFER FULL ??
3477  007342  001420                          BEQ     3$              ;BR IF YES
3478
3479  007344  022767  000012  012776          CMP     #12,SINGLE      ;WAS IT A LINE FEED ??
3480  007352  001011                          BNE     2$              ;BR IF NOT
3481
3482  007354  016700  013052                  MOV     FILLB,R0        ;LOAD LF PLUS FILLERS
3483  007360  116725  012764                  MOVB    SINGLE,(R5)+
3484  007364  116725  013040          11$:    MOVB    FILLA,(R5)+     ;LOAD A FILLER CHAR
3485  007370  005300                          DEC     R0
3486  007372  001374                          BNE     11$             ;BR TILL REQUIRED FILLERS LOADED
3487  007374  000744                          BR      1$              ;GO ASK FOR ANOTHER CHAR
3488
3489  007376  116725  012746          2$:     MOVB    SINGLE,(R5)+    ;LOAD IT IN BUFFER
3490  007402  000741                          BR      1$              ;GO GET NEXT CHAR
3491
3492  007404  112767  000136  016612  3$:     MOVB    #136,EC3                ;ECHO CONTROL-C
3493  007412  112767  000103  016605          MOVB    #103,EC3+1
3494  007420  104401                          TYPE
3495  007422  026224                          EC3
3496  007424  162705  032754                  SUB     #TBUF,R5        ;SET UP CHAR COUNT
3497  007430  005405                          NEG     R5
3498  007432  010567  012136                  MOV     R5,CHRCNT
3499  007436  004767  000050          4$:     JSR     PC,DHST2        ;GO EXECUTE PATTERN
3500  007442  005267  012674                  INC     DATCNT
3501  007446  026767  012702  012666          CMP     PATLIM,DATCNT   ;DONE REQUIRED ITERATIONS
3502  007454  001370                          BNE     4$              ;BR IF NOT
3503
3504  007456  016767  012660  171476          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
3505  007464  005067  012652                  CLR     DATCNT          ;INIT ITERATION COUNTER
3506  007470  012767  007500  171412          MOV     #5$,$LPERR      ;RETURN TO 5$
3507  007476  104024                          ERROR   24              ;DONE REQUIRED ITERATIONS
3508  007500  105777  171434          5$:     TSTB    @SWR            ;LOCK ON THIS BUFFER ??
3509  007504  100754                          BMI     4$              ;BR IF YES
3510  007506  000167  176532                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3511
```

```
 3512  007512  004767  005122          DHST2:  JSR     PC,DHSET1       ;GO SET UP THE DH11
 3513  007516  056761  012004  000012          BIS     LINMSK,BAR(R1)  ;ACTIVATE THE LINE
 3514  007524  004767  011670                  JSR     PC,CHPS1        ;GO CLEAR PSW
 3515
 3516  007530  012767  000200  012574  PTWAIT: MOV     #200,TIMEA      ;INIT TIMERS
 3517  007536  005067  012572                  CLR     TIMEB
 3518  007542  005767  012034          1$:     TST     RDONE           ;DONE ENTIRE PATTERN ?
 3519  007546  001023                          BNE     3$              ;BR IF YES
 3520  007550  004767  007332                  JSR     PC,TIMEIT       ;CALL THE TIMER
 3521  007554  000772                          BR      1$              ;EXECUTED IF NO TIMEOUT
 3522
 3523  007556  012777  004000  011730          MOV     #BIT11,@DHADR   ;CLEAR THE DH11
 3524  007564  016767  012552  171370          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNTER
 3525  007572  016767  012546  171364          MOV     DATPAT,$REG1    ;SAVE PATTERN TYPE
 3526  007600  012767  007610  171302          MOV     #2$,$LPERR      ;SET UP ERROR RETURN
 3527  007606  104025                          ERROR   25              ;DATA PATTERNS TEST TIMEOUT ERROR
 3528
 3529  007610  005726                  2$:     TST     (SP)+           ;FIX STACK SINCE WE ARE SKIPPING RTS
 3530  007612  000167  176426                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
 3531
 3532  007616  052711  004000          3$:     BIS     #BIT11,(P1)     ;CLEAR THE DH11
 3533  007622  004767  000740                  JSR     PC,CKERDP       ;GO CHECK DATA BUFFERS
 3534  007626  000207                          RTS     PC              ;RETURN TO CONTROL ROUTINE
 3535
 3536
```

```
3537
3538                                    ;TRANSMITTER INTERRUPT SERVICE ROUTINE THREE
3539
3540  007630  032711  002000          T:NT3:  BIT     #BIT10,(R1)     ;NON EX MEM ERROR ??
3541  007634  001430                          BEQ     2$              ;BR IF NOT
3542
3543  007636  011103                          MOV     (R1),R3         ;SAVE THE SCR
3544  007640  004767  011570                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3545  007644  012711  004000                  MOV     #BIT11,(R1)     ;CLEAR OUT THE DH11
3546  007650  116704  012270                  MOVB    LINE,R4         ;SET UP THE S/B DATA
3547  007654  042703  175760                  BIC     #175760,R3      ;CLEAR OUT SUPERFLUOUS BITS
3548  007660  010102                          MOV     R1,R2           ;SET UP REGADR
3549  007662  004767  005234                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
3550  007666  004567  005144                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3551  007672  022144                          LINE
3552  007674  022500                          EM1+44
3553  007676  012767  007706  171204          MOV     #1$,$LPERR      ;SET UP THE ERROR LOOP RETURN
3554  007704  104001                          ERROR   1               ;NON EX MEM ERROR
3555  007706  022626                  1$:     CMP     (SP)+,(SP)+     ;POP THE STACK
3556  007710  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3557  007712  000167  176326                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3558
3559  007716  011103                  2$:     MOV     (R1),R3         ;GET THE SCR REG CONTENTS
3560  007720  100431                          BMI     4$              ;BR IF XMIT DONE SET
3561
3562  007722  004767  011506                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3563  007726  012711  004000                  MOV     #BIT11,(R1)     ;CLEAR THE DH11 - FATAL ERROR
3564  007732  012704  100000                  MOV     #BIT15,R4       ;SET UP S/B DATA
3565  007736  156704  012202                  BISB    LINE,R4
3566  007742  042703  077760                  BIC     #77760,R3       ;CLEAR OUT SUPERFLUOUS BITS
3567  007746  010102                          MOV     R1,R2           ;SET UP REGADR
3568  007750  004767  005146                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
3569  007754  004567  005056                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3570  007760  022144                          LINE
3571  007762  022662                          EM2+54
3572  007764  012767  007774  171116          MOV     #3$,$LPERR      ;SET UP ERROR LOOP RETURN
3573  007772  104002                          ERROR   2               ;XMITTR FALSE INTERRUPT
3574  007774  022626                  3$:     CMP     (SP)+,(SP)+     ;POP THE STACK
3575  007776  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3576  010000  000167  176240                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3577
3578  010004  005761  000012          4$:     TST     BAR(R1)         ;DID BAR BIT CLEAR ??
3579  010010  001430                          BEQ     6$              ;BR IF YES
3580
3581  010012  004767  011416                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3582  010016  016103  000012                  MOV     BAR(R1),R3      ;GET THE WAS DATA
3583  010022  012711  004000                  MOV     #BIT11,(R1)     ;CLEAR THE DH11
3584  010026  005004                          CLR     R4              ;SET UP S/B DATA
3585  010030  010102                          MOV     R1,R2           ;SET UP REGADR
3586  010032  062702  000012                  ADD     #BAR,R2
3587  010036  004767  005060                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
3588  010042  004567  004770                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3589  010046  022144                          LINE
3590  010050  022742                          EM3+55
3591  010052  012767  010062  171030          MOV     #5$,$LPERR      ;SAVE THE ERROR LOOP RETURN
3592  010060  104003                          ERROR   3               ;BUFFER ACTIVE REG FAILED TO CLEAR
```

```
3593  010062  022626                5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
3594  010064  005726                        TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3595  010066  000167  176152               JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3596
3597  010072  005761  0000 0         6$:     TST     BCR(R1)         ;DID BYTE COUNT GO TO ZERO ??
3598  010076  001430                        BEQ     8$              ;BR IF YES
3599
3600  010100  004767  011330               JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3601  010104  016103  000010               MOV     BCR(R1),R3      ;GET THE WAS DATA
3602  010110  012711  004000               MOV     #BIT11,(R1)     ;CLEAR THE DH11
3603  010114  005004                        CLR     R4              ;SET UP S/B DATA
3604  010116  010102                        MOV     R1,R2           ;SET UP REGADR
3605  010120  062702  000010               ADD     #BCR,R2
3606  010124  004767  004772               JSR     PC,SUER1        ;GO SET UP THE ERROR INFO
3607  010130  004567  004702               JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3608  010134  022144                        LINE
3609  010136  023017                        EM4+52
3610  010140  012767  010150  170742       MOV     #7$,$LPERR      ;SET UP ERROR LOOP RETURN
3611  010146  104004                        ERROR   4               ;BYTE COUNT REG FAILED TO GO TO 000000
3612  010150  022626                7$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
3613  010152  005726                        TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3614  010154  000167  176064               JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3615
3616  010160  016103  000006         8$:     MOV     CAR(R1),R3      ;GET THE WAS DATA
3617  010164  016704  011404               MOV     CHRCNT,R4       ;SET UP S/B DATA
3618  010170  005404                        NEG     R4
3619  010172  062704  032754               ADD     #TBUF,R4
3620  010176  020304                        CMP     R3,R4           ;WAS CAR CORRECT ??
3621  010200  001423                        BEQ     10$             ;BR IF YES
3622
3623  010202  004767  011226               JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3624  010206  010102                        MOV     R1,R2           ;SET UP REGADR
3625  010210  062702  000006               ADD     #CAR,R2
3626  010214  004767  004702               JSR     PC,SUER1        ;GO SET UP ERROR INFO
3627  010220  004567  004612               JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3628  010224  022144                        LINE
3629  010226  023101                        EM5+57
3630  010230  012767  010240  170652       MOV     #9$,$LPERR      ;SET UP THE ERROR RETURN
3631  010236  104005                        ERROR   5               ;CURRENT ADDRESS REG NOT CORRECT
3632  010240  022626                9$:     CMP     (SP)+,(SP)+     ;POP THE STACK
3633  010242  005726                        TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3634  010244  000167  175774               JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3635
3636  010250  000002                10$:    RTI
```

```
3637                                        ;RECEIVER INTERRUPT SERVICE ROUTINE THREE
3638
3639  010252  032711  040000          RINT3:  BIT     #BIT14,(R1)     ;SILO OVERFLOW ERROR ??
3640  010256  001427                          BEQ     2$              ;BR IF NOT
3641
3642  010260  004767  011150                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3643  010264  011103                          MOV     (R1),R3         ;GET THE WAS DATA
3644  010266  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
3645  010272  042703  177760                  BIC     #177760,R3      ;CLEAR JUNK
3646  010276  116704  011642                  MOVB    LINE,R4
3647  010302  004767  004614                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
3648  010306  004567  004524                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3649  010312  022144                          LINE
3650  010314  023150                          EM6+44
3651  010316  012767  010326  170564          MOV     #1$,$LPERR      ;SET UP ERROR LOOP RETURN
3652  010324  104006                          ERROR   6               ;SILO OVERFLOW - BAD,BAD,BAD !!!
3653  010326  022626                  1$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
3654  010330  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3655  010332  000167  175706                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3656
3657  010336  105711                  2$:     TSTB    (R1)            ;CHAR AVAIL SET ??
3658  010340  100432                          BMI     4$              ;BR IF YES
3659
3660  010342  004767  011066                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3661  010346  011103                          MOV     (R1),R3         ;GET WAS DATA
3662  010350  042703  177560                  BIC     #177560,R3      ;CLEAN IT UP
3663  010354  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR DH11
3664  010360  012704  000200                  MOV     #BIT07,R4       ;SET UP S/B DATA
3665  010364  156704  011554                  BISB    LINE,R4
3666  010370  010102                          MOV     R1,R2           ;SET UP REGADR
3667  010372  004767  004524                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
3668  010376  004567  004434                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3669  010402  022144                          LINE
3670  010404  023224                          EM7+51
3671  010406  012767  010416  170474          MOV     #3$,$LPERR      ;SET UP THE ERROR LOOP RETURN
3672  010414  104007                          ERROR   7               ;RECEIVER FALSE INTERRUPT
3673  010416  022626                  3$:     CMP     (SP)+,(SP)+     ;POP GOES THE SP
3674  010420  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3675  010422  000167  175616                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3676
3677  010426  016167  000002  170546  4$:     MOV     NRC(R1),$TMP0   ;SAVE THE DATA RECEIVED
3678  010434  100427                          BMI     6$              ;BR IF IT WAS VALID DATA
3679
3680  010436  004767  010772                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3681  010442  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
3682  010446  162767  030474  020014          SUB     #RBUF,RBFPTR    ;WHICH CHAR WAS IT ??
3683  010454  016702  020010                  MOV     RBFPTR,R2       ;SAVE CHAR NUMBER
3684  010460  004767  004432                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
3685  010464  004567  004346                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3686  010470  022144                          LINE
3687  010472  023274                          EM10+45
3688  010474  012767  010504  170406          MOV     #5$,$LPERR      ;SET UP ERROR RETURN
3689  010502  104010                          ERROR   10              ;RECEIVED INVALID DATA
3690  010504  022626                  5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
3691  010506  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3692  010510  000167  175530                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
```

```
 3693
 3694  010514  016777  170462  017746  6$:     MOV     $TMP0,@RBFPTR   ;STORE CHAR IN THE BUFFER
 3695  010522  062767  000001  017740          ADD     #1,RBFPTR       ;UPDATE THE POINTER
 3696  010530  026767  011050  017732          CMP     RBFEND,RBFPTR   ;END OF BUFFER ??
 3697  010536  001407                          BEQ     7$              ;BR IF YES
 3698  010540  062767  000001  017722          ADD     #1,RBFPTR
 3699  010546  026767  011032  017714          CMP     RBFEND,RBFPTR
 3700  010554  001003                          BNE     8$              ;BR IF NOT DONE
 3701  010556  052767  000001  011016  7$:     BIS     #1,RDONE        ;SET SOFTWARE DONE FLAG
 3702  010564  000002                  8$:     RTI                     ;RETURN TO WAIT LOOP
```

```
CZDHN-D MACY11 30A(1052)  27-DEC-78  15:31  PAGE 87
CZDHND.P11    27-DEC-78 15:28          SUB-PROGRAM THREE - DATA PATTERNS TESTS                                SEQ 0086

 3703                                 ;THIS ROUTINE IS CALLED TO CHECK THE RECEIVED DATA AND REPORT ALL ERRORS
 3704                                 ;FOR THE DATA PATTERNS TESTS
 3705
 3706  010566  012767  030474  017674 CKERDP: MOV     #RBUF,RBFPTR    ;SET UP POINTERS
 3707  010574  012767  032754  017670         MOV     #TBUF,TBFPTR
 3708
 3709  010602  117704  017664         1$:     MOVB    @TBFPTR,R4      ;GET THE S/B DATA
 3710  010606  000304                          SWAB    R4              ;PUT LINE NO. IN HIGH BYTE
 3711  010610  105004                          CLRB    R4
 3712  010612  156704  011326                  BISB    LINE,R4
 3713  010616  000304                          SWAB    R4
 3714  010620  052704  100000                  BIS     #BIT15,R4       ;AND FINALLY THE VALID DATA BIT
 3715  010624  046704  011532                  BIC     CLMSK,R4        ;MASK OFF BITS NOT XMITTED
 3716  010630  017703  017634                  MOV     @RBFPTR,R3      ;GET THE WAS DATA
 3717  010634  020304                          CMP     R3,R4           ;WAS = S/B ?????
 3718  010636  001425                          BEQ     3$              ;BR IF YES
 3719
 3720  010640  010367  170354                  MOV     R3,$TMP7        ;SAVE THE WAS DATA
 3721  010644  010146                          MOV     R1,-(SP)        ;SAVE THE DEVADR
 3722  010646  016701  017616                  MOV     RBFPTR,R1       ;GET THE SBADR
 3723  010652  016702  017614                  MOV     TBFPTR,R2       ;GET THE WASADR
 3724  010656  010200                          MOV     R2,R0           ;GET XMIT BUFFER ADDR
 3725  010660  162700  032754                  SUB     #TBUF,R0        ;GENERATE CHAR #
 3726  010664  004767  004226                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
 3727  010670  004567  004142                  JSR     R5,SUNUM        ;GO PUT LINE NO. IN MSG
 3728  010674  022144                          LINE
 3729  010676  023427                          EM11+23
 3730  010700  012767  010710  170202         MOV     #2$,$LPERR      ;SET UP ERROR RETURN
 3731  010706  104011                          ERROR   11              ;DATA COMPARE ERROR OR PARITY,FRAMING
 3732                                                                  ;OR OVERRUN
 3733  010710  012601                  2$:     MOV     (SP)+,R1        ;RESTORE THE DEVADR
 3734
 3735  010712  005267  017554          3$:     INC     TBFPTR          ;UPDATE POINTERS
 3736  010716  062767  000001  017544          ADD     #1,RBFPTR
 3737  010724  026767  010654  017536          CMP     RBFEND,RBFPTR   ;COMPARED ALL CHARS ??
 3738  010732  001407                          BEQ     4$              ;BR IF YES
 3739  010734  062767  000001  017526          ADD     #1,RBFPTR       ;UPDATE IT AGAIN
 3740  010742  026767  010636  017520          CMP     RBFEND,RBFPTR   ;DONE YET ?
 3741  010750  001314                          BNE     1$              ;BR IF NOT
 3742
 3743  010752  000207                  4$:     RTS     PC              ;RETURN TO WAIT LOOP
 3744
```

```
                                        .SBTTL  END OF PASS ROUTINE
                                        ;;**********************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO START2

010754                                  $EOP:
010754  000004                                SCOPE
010756  005067  170120                       CLR     $TSTNM          ;;ZERO THE TEST NUMBER
010762  005067  170234                       CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
010766  005267  170246                       INC     $PASS           ;;INCREMENT THE PASS NUMBER
010772  042767  100000  170240               BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
011000  005327                       $EOPCT: DEC     (PC)+           ;;LOOP?
011002  000001                               .WORD   1
011004  003022                               BGT     $DOAGN          ;;YES
011006  012737                               MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
011010  000001                       $ENDCT: .WORD   1
011012  011002                               $EOPCT
011014  104401  011061                       TYPE    ,$ENDMG         ;;TYPE 'END PASS #'
011020  016746  170214                       MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
011024  104405                               TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
011026  104401  011056                       TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
011032  013700  000042               $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
011036  001405                               BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
011040  000005                               RESET                   ;;CLEAR THE WORLD
011042  004710                       $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
011044  000240                               NOP                     ;;SAVE ROOM
011046  000240                               NOP                     ;;FOR
011050  000240                               NOP                     ;;ACT11
011052                                  $DOAGN:
011052  000137                               JMP     @(PC)+          ;;RETURN
011054  002306                       $RTNAD: .WORD   START2
011056     377     377     000       $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
011061     015  042412  042116       $ENDMG: .ASCIZ  <15><12>/END PASS #/
011066  050040  051501  020123
011074  000043
                                        .SBTTL  SCOPE HANDLER ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW14=1         LOOP ON TEST
                                        ;*SW11=1         INHIBIT ITERATIONS
                                        ;*SW09=1         LOOP ON ERROR
                                        ;*CALL
                                        ;*      SCOPE           ;;SCOPE=IOT

011076                                  $SCOPE:
011076  104407                               CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
011100  005067  011040                       CLR     LINE            ;INIT LINE COUNTER
011104  032777  040000  170026       1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
011112  001104                               BNE     $OVER           ;;YES IF SW14=1
```

```
3801                                    ;#####START OF CODE FOR THE XOR TESTER#####
3802  011114  000416                    $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
3803                                                                    ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
3804  011116  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
3805  011122  012737  011142  000004            MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
3806  011130  005737  177060                    TST     @#177060        ;;TIME OUT ON XOR?
3807  011134  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
3808  011140  000453                            BR      $SVLAD          ;;GO TO THE NEXT TEST
3809  011142  022626                    5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
3810  011144  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
3811  011150  000413                            BR      7$              ;;LOOP ON THE PRESENT TEST
3812  011152                            6$:;#####END OF CODE FOR THE XOR TESTER#####
3813  011152  105767  167725            2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
3814  011156  001421                            BEQ     3$              ;;BR IF NO
3815  011160  126767  167731  167715            CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
3816  011166  101015                            BHI     3$              ;;BR IF NO
3817  011170  032777  001000  167742            BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
3818  011176  001404                            BEQ     4$              ;;BR IF NO
3819  011200  016767  167704  167700  7$:      MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
3820  011206  000446                            BR      $OVER
3821  011210  105067  167667            4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
3822  011214  005067  170002                    CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
3823  011220  000415                            BR      1$              ;;ESCAPE TO THE NEXT TEST
3824  011222  032777  004000  167710  3$:      BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
3825  011230  001011                            BNE     1$              ;;BR IF YES
3826  011232  005767  170002                    TST     $PASS           ;;IF FIRST PASS OF PROGRAM
3827  011236  001406                            BEQ     1$              ;;      INHIBIT ITERATIONS
3828  011240  005267  167640                    INC     $ICNT           ;;INCREMENT ITERATION COUNT
3829  011244  026767  167752  167632            CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
3830  011252  002024                            BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
3831  011254  012767  000001  167622  1$:      MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
3832  011262  016767  000052  167732            MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
3833  011270  105267  167606            $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
3834  011274  116767  167602  167734            MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
3835  011302  011667  167600                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
3836  011306  011667  167576                    MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
3837  011312  005067  167706                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
3838  011316  112767  000001  167571            MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3839  011324  016777  167552  167610  $OVER:   MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
3840  011332  016716  167550                    MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
3841  011336  000002                            RTI                     ;;FIXES PS
3842  011340  000010                    $MXCNT: 10                      ;;MAX. NUMBER OF ITERATIONS
3843                                    .SBTTL  ERROR HANDLER ROUTINE
3844
3845                                    ;;**************************************************************
3846                                    ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3847                                    ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3848                                    ;*AND GO TO $ERRTYP ON ERROR
3849                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3850                                    ;*SW15=1         HALT ON ERROR
3851                                    ;*SW13=1         INHIBIT ERROR TYPEOUTS
3852                                    ;*SW09=1         LOOP ON ERROR
3853                                    ;*CALL
3854                                    ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3855
3856  011342                            $ERROR:
```

```
 3857  011342  104407                          CKSWR                       ;;TEST FOR CHANGE IN SOFT-SWR
 3858  011344  105267  167533          7$:     INCB    $ERFLG              ;;SET THE ERROR FLAG
 3859  011350  001775                          BEQ     7$                  ;;DON'T LET THE FLAG GO TO ZERO
 3860  011352  016777  167524  167562          MOV     $TSTNM,@DISPLAY     ;;DISPLAY TEST NUMBER AND ERROR FLAG
 3861  011360  005267  167526                  INC     $ERTTL              ;;INC THE ERROR COUNT
 3862  011364  011667  167526                  MOV     (SP),$ERRPC         ;;GET ADDRESS OF ERROR INSTRUCTION
 3863  011370  162767  000002  167520          SUB     #2,$ERRPC
 3864  011376  117767  167514  167510          MOVB    @$ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE
 3865  011404  032777  020000  167526          BIT     #BIT13,@SWR         ;;SKIP TYPEOUT IF SET
 3866  011412  001004                          BNE     20$                 ;;SKIP TYPEOUTS
 3867  011414  004767  000074                  JSR     PC,$ERRTYP          ;;GO TO USER ERROR ROUTINE
 3868  011420  104401  001227                  TYPE    ,$CRLF
 3869  011424                          20$:
 3870  011424  122767  000001  167620          CMPB    #APTENV,$ENV        ;;RUNNING IN APT MODE
 3871  011432  001007                          BNE     2$                  ;;NO,SKIP APT ERROR REPORT
 3872  011434  116767  167454  000004          MOVB    $ITEMB,21$          ;;SET ITEM NUMBER AS ERROR NUMBER
 3873  011442  004767  001174                  JSR     PC,$ATY4            ;;REPORT FATAL ERROR TO APT
 3874  011446     000                  21$:     .BYTE   0
 3875  011447     000                          .BYTE   0
 3876  011450  000777                  22$:     BR      22$                 ;;APT ERROR LOOP
 3877  011452  005777  167462          2$:     TST     @SWR                ;;HALT ON ERROR
 3878  011456  100002                          BPL     3$                  ;;SKIP IF CONTINUE
 3879  011460  000000                          HALT                        ;;HALT ON ERROR!
 3880  011462  104407                          CKSWR                       ;;TEST FOR CHANGE IN SOFT-SWR
 3881  011464  032777  001000  167446  3$:     BIT     #BIT09,@SWR         ;;LOOP ON ERROR SWITCH SET?
 3882  011472  001402                          BEQ     4$                  ;;BR IF NO
 3883  011474  016716  167410                  MOV     $LPERR,(SP)         ;;FUDGE RETURN FOR LOOPING
 3884  011500  005767  167520          4$:     TST     $ESCAPE             ;;CHECK FOR AN ESCAPE ADDRESS
 3885  011504  001402                          BEQ     5$                  ;;BR IF NONE
 3886  011506  016716  167512                  MOV     $ESCAPE,(SP)        ;;FUDGE RETURN ADDRESS FOR ESCAPE
 3887  011512                          5$:
 3888  011512  000002                          RTI                         ;;RETURN
 3889                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 3890
 3891                                  ;;**************************************************************************
 3892                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 3893                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
 3894                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 3895
 3896  011514                          $ERRTYP:
 3897  011514  104401  001227                  TYPE    ,$CRLF              ;;"CARRIAGE RETURN" & "LINE FEED"
 3898  011520  010046                          MOV     R0,-(SP)            ;;SAVE R0
 3899  011522  005000                          CLR     R0                  ;;PICKUP THE ITEM INDEX
 3900  011524  153700  001114                  BISB    @#$ITEMB,R0
 3901  011530  001004                          BNE     1$                  ;;IF ITEM NUMBER IS ZERO, JUST
 3902                                                                      ;;TYPE THE PC OF THE ERROR
 3903  011532  016746  167360                  MOV     $ERRPC,-(SP)        ;;SAVE $ERRPC FOR TYPEOUT
 3904                                                                      ;;ERROR ADDRESS
 3905  011536  104402                          TYPOC                       ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 3906  011540  000445                          BR      10$                 ;;GET OUT
 3907  011542  005300                  1$:     DEC     R0                  ;;ADJUST THE INDEX SO THAT IT WILL
 3908  011544  006300                          ASL     R0                  ;;      WORK FOR THE ERROR TABLE
 3909  011546  006300                          ASL     R0
 3910  011550  006300                          ASL     R0
 3911  011552  062700  001326                  ADD     #$ERRTB,R0          ;;FORM TABLE POINTER
 3912  011556  012067  000004                  MOV     (R0)+,2$            ;;PICKUP "ERROR MESSAGE" POINTER
```

ERROR MESSAGE TYPEOUT ROUTINE SEQ 0090

```
3913  011562  001404                        BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
3914  011564  104401                        TYPE                    ;;TYPE THE 'ERROR MESSAGE'
3915  011566  000000                2$:     .WORD   0               ;;'ERROR MESSAGE' POINTER GOES HERE
3916  011570  104401  001227                TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
3917  011574  012067  000004        3$:     MOV     (R0)+,4$        ;;PICKUP 'DATA HEADER' POINTER
3918  011600  001404                        BEQ     5$              ;;SKIP TYPEOUT IF 0
3919  011602  104401                        TYPE                    ;;TYPE THE 'DATA HEADER'
3920  011604  000000                4$:     .WORD   0               ;;'DATA HEADER' POINTER GOES HERE
3921  011606  104401  001227                TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
3922  011612  010146                5$:     MOV     R1,-(SP)        ;;SAVE R1
3923  011614  012001                        MOV     (R0)+,R1        ;;PICKUP 'DATA TABLE' POINTER
3924  011616  001415                        BEQ     9$              ;;BR IF NO DATA TO BE TYPED
3925  011620  012000                        MOV     (R0)+,R0        ;;PICKUP 'DATA FORMAT' POINTER
3926  011622  105720                6$:     TSTB    (R0)+           ;;'OCTAL' OR 'DECIMAL'
3927  011624  001003                        BNE     7$              ;;BR IF DECIMAL
3928  011626  013146                        MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3929  011630  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3930  011632  000402                        BR      8$
3931  011634                        7$:
3932  011634  013146                        MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3933  011636  104405                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
3934  011640  005711                8$:     TST     (R1)            ;;IS THERE ANOTHER NUMBER?
3935  011642  001403                        BEQ     9$              ;;BR IF NO
3936  011644  104401  011664                TYPE    ,11$            ;;TYPE TWO(2) SPACES
3937  011650  000764                        BR      6$              ;;LOOP
3938
3939  011652  012601                9$:     MOV     (SP)+,R1        ;;RESTORE R1
3940  011654  012600                10$:    MOV     (SP)+,R0        ;;RESTORE R0
3941  011656  104401  001227                TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
3942  011662  000207                        RTS     PC              ;;RETURN
3943  011664  020040     000        11$:    .ASCIZ  /  /            ;;TWO(2) SPACES
3944          011670                        .EVEN
3945                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3946
3947                                ;;**********************************************************************
3948                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3949                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
3950                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3951                                ;*CALL:
3952                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3953                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
3954                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3955                                ;*      .BYTE   M               ;;M=1 OR 0
3956                                ;*                                      ;;1=TYPE LEADING ZEROS
3957                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
3958                                ;*
3959                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3960                                ;*$TYPOS OR $TYPOC
3961                                ;*CALL:
3962                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3963                                ;*      TYPON                   ;;CALL FOR TYPEOUT
3964                                ;*
3965                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3966                                ;*CALL:
3967                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3968                                ;*      TYPOC                   ;;CALL FOR TYPEOUT
```

```
 3969
 3970  011670  017646  000000          $TYPOS: MOV     @(SP),-(SP)      ;;PICKUP THE MODE
 3971  011674  116667  000001  000211          MOVB    1(SP),$OFILL     ;;LOAD ZERO FILL SWITCH
 3972  011702  112667  000207                  MOVB    (SP)+,$OMODE+1   ;;NUMBER OF DIGITS TO TYPE
 3973  011706  062716  000002                  ADD     #2,(SP)          ;;ADJUST RETURN ADDRESS
 3974  011712  000406                          BR      $TYPON
 3975  011714  112767  000001  000171 $TYPOC: MOVB    #1,$OFILL        ;;SET THE ZERO FILL SWITCH
 3976  011722  112767  000006  000165          MOVB    #6,$OMODE+1      ;;SET FOR SIX(6) DIGITS
 3977  011730  112767  000005  000154 $TYPON: MOVB    #5,$OCNT         ;;SET THE ITERATION COUNT
 3978  011736  010346                          MOV     R3,-(SP)         ;;SAVE R3
 3979  011740  010446                          MOV     R4,-(SP)         ;;SAVE R4
 3980  011742  010546                          MOV     R5,-(SP)         ;;SAVE R5
 3981  011744  116704  000145                  MOVB    $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
 3982  011750  005404                          NEG     R4
 3983  011752  062704  000006                  ADD     #6,R4            ;;SUBTRACT IT FOR MAX. ALLOWED
 3984  011756  110467  000132                  MOVB    R4,$OMODE        ;;SAVE IT FOR USE
 3985  011762  116704  000125                  MOVB    $OFILL,R4        ;;GET THE ZERO FILL SWITCH
 3986  011766  016605  000012                  MOV     12(SP),R5        ;;PICKUP THE INPUT NUMBER
 3987  011772  005003                          CLR     R3               ;;CLEAR THE OUTPUT WORD
 3988  011774  006105                  1$:     ROL     R5               ;;ROTATE MSB INTO "C"
 3989  011776  000404                          BR      3$               ;;GO DO MSB
 3990  012000  006105                  2$:     ROL     R5               ;;FORM THIS DIGIT
 3991  012002  006105                          ROL     R5
 3992  012004  006105                          ROL     R5
 3993  012006  010503                          MOV     R5,R3
 3994  012010  006103                  3$:     ROL     R3               ;;GET LSB OF THIS DIGIT
 3995  012012  105367  000076                  DECB    $OMODE           ;;TYPE THIS DIGIT?
 3996  012016  100016                          BPL     7$               ;;BR IF NO
 3997  012020  042703  177770                  BIC     #177770,R3       ;;GET RID OF JUNK
 3998  012024  001002                          BNE     4$               ;;TEST FOR 0
 3999  012026  005704                          TST     R4               ;;SUPPRESS THIS 0?
 4000  012030  001403                          BEQ     5$               ;;BR IF YES
 4001  012032  005204                  4$:     INC     R4               ;;DON'T SUPPRESS ANYMORE 0'S
 4002  012034  052703  000060                  BIS     #'0,R3           ;;MAKE THIS DIGIT ASCII
 4003  012040  052703  000040          5$:     BIS     #' ,R3           ;;MAKE ASCII IF NOT ALREADY
 4004  012044  110367  000040                  MOVB    R3,8$            ;;SAVE FOR TYPING
 4005  012050  104401  012110                  TYPE    ,8$              ;;GO TYPE THIS DIGIT
 4006  012054  105367  000032          7$:     DECB    $OCNT            ;;COUNT BY 1
 4007  012060  003347                          BGT     2$               ;;BR IF MORE TO DO
 4008  012062  002402                          BLT     6$               ;;BR IF DONE
 4009  012064  005204                          INC     R4               ;;INSURE LAST DIGIT ISN'T A BLANK
 4010  012066  000744                          BR      2$               ;;GO DO THE LAST DIGIT
 4011  012070  012605                  6$:     MOV     (SP)+,R5         ;;RESTORE R5
 4012  012072  012604                          MOV     (SP)+,R4         ;;RESTORE R4
 4013  012074  012603                          MOV     (SP)+,R3         ;;RESTORE R3
 4014  012076  016666  000002  000004          MOV     2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
 4015  012104  012616                          MOV     (SP)+,(SP)
 4016  012106  000002                          RTI                      ;;RETURN
 4017  012110     000          8$:     .BYTE   0                ;;STORAGE FOR ASCII DIGIT
 4018  012111     000                  .BYTE   0                ;;TERMINATOR FOR TYPE ROUTINE
 4019  012112     000          $OCNT:  .BYTE   0                ;;OCTAL DIGIT COUNTER
 4020  012113     000          $OFILL: .BYTE   0                ;;ZERO FILL SWITCH
 4021  012114  000000          $OMODE: .WORD   0                ;;NUMBER OF DIGITS TO TYPE
 4022                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 4023
 4024                          ;;**********************************************************************
```

```
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                        ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                        ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                        ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                        ;*REPLACED WITH SPACES.
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                        ;*      TYPDS                   ;;GO TO THE ROUTINE
012116                                  $TYPDS:
012116  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
012120  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
012122  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
012124  010346                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
012126  010546                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
012130  012746  020200                        MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
012134  016605  000020                        MOV     20(SP),R5       ;;GET THE INPUT NUMBER
012140  100004                                BPL     1$              ;;BR IF INPUT IS POS.
012142  005405                                NEG     R5              ;;MAKE THE BINARY NUMBER POS.
012144  112766  000055  000001                MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
012152  005000                          1$:   CLR     R0              ;;ZERO THE CONSTANTS INDEX
012154  012703  012332                        MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
012160  112723  000040                        MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
012164  005002                          2$:   CLR     R2              ;;CLEAR THE BCD NUMBER
012166  016001  012322                        MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
012172  160105                          3$:   SUB     R1,R5           ;;FORM THIS BCD DIGIT
012174  002402                                BLT     4$              ;;BR IF DONE
012176  005202                                INC     R2              ;;INCREASE THE BCD DIGIT BY 1
012200  000774                                BR      3$
012202  060105                          4$:   ADD     R1,R5           ;;ADD BACK THE CONSTANT
012204  005702                                TST     R2              ;;CHECK IF BCD DIGIT=0
012206  001002                                BNE     5$              ;;FALL THROUGH IF 0
012210  105716                                TSTB    (SP)            ;;STILL DOING LEADING 0'S?
012212  100407                                BMI     7$              ;;BR IF YES
012214  106316                          5$:   ASLB    (SP)            ;;MSD?
012216  103003                                BCC     6$              ;;BR IF NO
012220  116663  000001  177777                MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
012226  052702  000060                  6$:   BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
012232  052702  000040                  7$:   BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
012236  110223                                MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
012240  005720                                TST     (R0)+           ;;JUST INCREMENTING
012242  020027  000010                        CMP     R0,#10          ;;CHECK THE TABLE INDEX
012246  002746                                BLT     2$              ;;GO DO THE NEXT DIGIT
012250  003002                                BGT     8$              ;;GO TO EXIT
012252  010502                                MOV     R5,R2           ;;GET THE LSD
012254  000764                                BR      6$              ;;GO CHANGE TO ASCII .
012256  105726                          8$:   TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
012260  100003                                BPL     9$              ;;BR IF NO
012262  116663  177777  177776                MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
012270  105013                          9$:   CLRB    (R3)            ;;SET THE TERMINATOR
012272  012605                                MOV     (SP)+,R5        ;;POP STACK INTO R5
012274  012603                                MOV     (SP)+,R3        ;;POP STACK INTO R3
012276  012602                                MOV     (SP)+,R2        ;;POP STACK INTO R2
012300  012601                                MOV     (SP)+,R1        ;;POP STACK INTO R1
012302  012600                                MOV     (SP)+,R0        ;;POP STACK INTO R0
012304  104401  012332                        TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
```

```
 4081  012310  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJJST THE STACK
 4082  012316  012616                          MOV     (SP)+,(SP)
 4083  012320  000002                          RTI                     ;;RETURN TO USER
 4084  012322  023420                  $DTBL:  10000.
 4085  012324  001750                          1000.
 4086  012326  000144                          100.
 4087  012330  000012                          10.
 4088  012332  000004                  $DBLK:  .BLKW   4
 4089                                  .SBTTL  TYPE ROUTINE
 4090
 4091                                  ;;**********************************************************************
 4092                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 4093                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 4094                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 4095                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 4096                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 4097                                  ;*
 4098                                  ;*CALL:
 4099                                  ;*1) USING A TRAP INSTRUCTION
 4100                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 4101                                  ;*OR
 4102                                  ;*      TYPE
 4103                                  ;*      MESADR
 4104                                  ;*
 4105
 4106  012342  105767  166611          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 4107  012346  100002                          BPL     1$              ;;BR IF YES
 4108  012350  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
 4109  012352  000430                          BR      3$              ;;LEAVE
 4110  012354  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
 4111  012356  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 4112  012362  122767  000001  166662          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
 4113  012370  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
 4114  012372  132767  000100  166653          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
 4115  012400  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
 4116  012402  010067  000004                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
 4117  012406  004767  000220                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
 4118  012412  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
 4119  012414  132767  000040  166631  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
 4120  012422  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
 4121  012424  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 4122  012426  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 4123  012430  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 4124  012432  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
 4125  012434  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 4126  012440  000002                          RTI                     ;;RETURN
 4127  012442  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 4128  012446  001430                          BEQ     8$
 4129  012450  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 4130  012454  001006                          BNE     5$
 4131  012456  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
 4132  012460  104401                          TYPE                    ;;TYPE A CR AND LF
 4133  012462  001227                          $CRLF
 4134  012464  105067  000130                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 4135  012470  000755                          BR      2$              ;;GET NEXT CHARACTER
 4136  012472  004767  000056          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
```

```
 4137  012476  126726  166454          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 4138  012502  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
 4139  012504  016746  166444                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
 4140                                                                  ;;AND THE NULL CHAR.
 4141  012510  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 4142  012514  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 4143  012516  004767  000032                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
 4144  012522  105367  000072                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 4145  012526  000770                          BR      7$              ;;LOOP
 4146
 4147                                  ;HORIZONTAL TAB PROCESSOR
 4148
 4149  012530  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 4150  012534  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 4151  012540  132767  000007  000052          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 4152  012546  001372                          BNE     9$              ;;TAB STOP
 4153  012550  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
 4154  012552  000724                          BR      2$              ;;GET NEXT CHARACTER
 4155  012554  105777  166370          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 4156  012560  100375                          BPL     $TYPEC
 4157  012562  116677  000002  166362          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 4158  012570  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 4159  012576  001003                          BNE     1$              ;;BRANCH IF NO
 4160  012600  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 4161  012604  000406                          BR      $TYPEX          ;;EXIT
 4162  012606  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 4163  012614  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 4164  012616  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 4165  012620  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 4166  012622  000207          $TYPEX: RTS     PC
 4167
 4168                                  .SBTTL  APT COMMUNICATIONS ROUTINE
 4169
 4170                                  ;;**********************************************************************
 4171  012624  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
 4172  012632  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
 4173  012640  000403                          BR      $ATYC
 4174  012642  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
 4175  012650                          $ATYC:
 4176  012650  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 4177  012652  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 4178  012654  105767  000206                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
 4179  012660  001450                          BEQ     5$              ;;IF NOT:  BR
 4180  012662  122767  000001  166362          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
 4181  012670  001031                          BNE     3$              ;;IF NOT:  BR
 4182  012672  132767  000100  166353          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
 4183  012700  001425                          BEQ     3$              ;;IF NOT:  BR
 4184  012702  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
 4185  012706  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 4186  012714  005767  166312          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
 4187  012720  001375                          BNE     1$              ;;IF NOT:  WAIT
 4188  012722  010067  166320                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
 4189  012726  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
 4190  012730  001376                          BNE     2$
 4191  012732  166700  166310                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
 4192  012736  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
```

```
 4193  012740  010067  166304                 MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
 4194  012744  012767  000004  166260         MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
 4195  012752  000413                         BR      5$
 4196  012754  017667  000004  000016  3$:    MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
 4197  012762  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
 4198  012770  016746  165002                 MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
 4199  012774  004767  177342                 JSR     PC,$TYPE        ;;CALL TYPE MACRO
 4200  013000  000000                 4$:     .WORD   0
 4201  013002                          5$:
 4202  013002  105767  000062         10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
 4203  013006  001416                         BEQ     12$             ;;IF NOT:  BR
 4204  013010  005767  166236                 TST     $ENV            ;;RUNNING UNDER APT?
 4205  013014  001413                         BEQ     12$             ;;IF NOT:  BR
 4206  013016  005767  166210         11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
 4207  013022  001375                         BNE     11$             ;;IF NOT:  WAIT
 4208  013024  017667  000004  166202         MOV     @4(SP),$FATAL   ;;GET ERROR #
 4209  013032  062766  000002  000004         ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
 4210  013040  005267  166166                 INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
 4211  013044  105067  000020         12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
 4212  013050  105067  000013                 CLRB    $LFLG           ;;CLEAR LOG FLAG
 4213  013054  105067  000006                 CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
 4214  013060  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
 4215  013062  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
 4216  013064  000207                         RTS     PC              ;;RETURN
 4217  013066     000                 $MFLG:  .BYTE   0               ;;MESSG. FLAG
 4218  013067     000                 $LFLG:  .BYTE   0               ;;LOG FLAG
 4219  013070     000                 $FFLG:  .BYTE   0               ;;FATAL FLAG
 4220          013072                         .EVEN
 4221          000200                 APTSIZE=200
 4222          000001                 APTENV=001
 4223          000100                 APTSPOOL=100
 4224          000040                 APTCSUP=040
 4225                                 .SBTTL  TTY INPUT ROUTINE
 4226
 4227                                 ;;**********************************************************************
 4228                                 .ENABL  LSB
 4229
 4230                                 ;;**********************************************************************
 4231                                 ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 4232                                 ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
 4233                                 ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
 4234                                 ;*WHEN OPERATING IN TTY FLAG MODE.
 4235  013072  022767  000176  166040 $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
 4236  013100  001074                         BNE     15$             ;;BRANCH IF NO
 4237  013102  105777  166036                 TSTB    @$TKS           ;;CHAR THERE?
 4238  013106  100071                         BPL     15$             ;;IF NO, DON'T WAIT AROUND
 4239  013110  117746  166032                 MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
 4240  013114  042716  177600                 BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
 4241  013120  022726  000007                 CMP     #7,(SP)+        ;;IS IT A CONTROL G?
 4242  013124  001062                         BNE     15$             ;;NO, RETURN TO USER
 4243  013126  126727  166002  000001         CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
 4244  013134  001456                         BEQ     15$             ;;BRANCH IF YES
 4245
 4246  013136  104401  013745                 TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
 4247  013142  104401  013752         $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
 4248  013146  016746  165024                 MOV     SWRFG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
```

```
 4249  013152  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4250  013154  104401  013763                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
 4251  013160  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
 4252  013162  005046                          CLR     -(SP)           ;;THE NEW SWR
 4253  013164  105777  165754          7$:     TSTB    @$TKS           ;;CHAR THERE?
 4254  013170  100375                          BPL     7$              ;;IF NOT TRY AGAIN
 4255
 4256  013172  117746  165750                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
 4257  013176  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 4258
 4259
 4260
 4261  013202  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
 4262  013206  001005                          BNE     10$             ;;BRANCH IF NOT
 4263  013210  104401  013740                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
 4264  013214  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
 4265  013220  000757                          BR      19$             ;;LET'S TRY IT AGAIN
 4266
 4267
 4268  013222  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
 4269  013226  001022                          BNE     16$             ;;BRANCH IF NO
 4270  013230  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
 4271  013234  001403                          BEQ     11$             ;;BRANCH IF YES
 4272  013236  016677  000002  165674          MOV     2(SP),@SWR      ;;SAVE NEW SWR
 4273  013244  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
 4274  013250  104401  001227          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
 4275  013254  126727  165655  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
 4276  013262  001003                          BNE     15$             ;;BRANCH IF NOT
 4277  013264  012777  000100  165652          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
 4278  013272  000002                  15$:    RTI                     ;;RETURN
 4279  013274  004767  177254          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
 4280  013300  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
 4281  013304  002420                          BLT     18$             ;;BRANCH IF YES
 4282  013306  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
 4283  013312  003015                          BGT     18$             ;;BRANCH IF YES
 4284  013314  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
 4285  013320  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
 4286  013324  001403                          BEQ     17$             ;;BRANCH IF YES
 4287  013326  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
 4288  013330  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
 4289  013332  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
 4290  013334  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
 4291  013340  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
 4292  013344  000707                          BR      7$              ;;GET THE NEXT ONE
 4293  013346  104401  001226          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
 4294  013352  000720                          BR      20$             ;;SIMULATE CONTROL-U
 4295                                  .DSABL  LSB
 4296
 4297
 4298                                  ;;*****************************************************************
 4299                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 4300                                  ;*CALL:
 4301                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
 4302                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 4303                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
 4304                                  ;
```

```
 4305
 4306  013354  011646                    $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
 4307  013356  016666  000004  000002            MOV     4(SP),2(SP)     ;;SAVE THE PS
 4308  013364  105777  165554            1$:     TSTB    @$TKS           ;;WAIT FOR
 4309  013370  100375                            BPL     1$              ;;A CHARACTER
 4310  013372  117766  165550  000004            MOVB    @$TKB,4(SP)     ;;READ THE TTY
 4311  013400  042766  177600  000004            BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
 4312  013406  026627  000004  000023            CMP     4(SP),#23       ;;IS IT A CONTROL-S?
 4313  013414  001013                            BNE     3$              ;;BRANCH IF NO
 4314  013416  105777  165522            2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
 4315  013422  100375                            BPL     2$              ;;LOOP UNTIL ITS THERE
 4316  013424  117746  165516                    MOVB    @$TKB,-(SP)     ;;GET CHARACTER
 4317  013430  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 4318  013434  022627  000021                    CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 4319  013440  001366                            BNE     2$              ;;IF NOT DISCARD IT
 4320  013442  000750                            BR      1$              ;;YES, RESUME
 4321  013444  026627  000004  000140    3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
 4322  013452  002407                            BLT     4$              ;;BRANCH IF YES
 4323  013454  026627  000004  000175            CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
 4324  013462  003003                            BGT     4$              ;;BRANCH IF YES
 4325  013464  042766  000040  000004            BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
 4326  013472  000002                    4$:     RTI                     ;;GO BACK TO USER
 4327                                    ;;**********************************************************************
 4328                                    ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 4329                                    ;*CALL:
 4330                                    ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 4331                                    ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 4332                                    ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 4333
 4334  013474  010346                    $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 4335  013476  005046                            CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
 4336  013500  012703  013730            1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 4337  013504  022703  013740            2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
 4338  013510  101456                            BLOS    4$              ;;BR IF YES
 4339  013512  104410                            RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 4340  013514  112613                            MOVB    (SP)+,(R3)      ;;GET CHARACTER
 4341  013516  122713  000177            10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 4342  013522  001022                            BNE     5$              ;;BR IF NO
 4343  013524  005716                            TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
 4344  013526  001007                            BNE     6$              ;;BR IF NO
 4345  013530  112767  000134  000170            MOVB    #'\,9$          ;;TYPE A BACK SLASH
 4346  013536  104401  013726                    TYPE    ,9$
 4347  013542  012716  177777                    MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
 4348  013546  005303                    6$:     DEC     R3              ;;BACKUP BY ONE
 4349  013550  020327  013730                    CMP     R3,#$TTYIN      ;;STACK EMPTY?
 4350  013554  103434                            BLO     4$              ;;BR IF YES
 4351  013556  111367  000144                    MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
 4352  013562  104401  013726                    TYPE    ,9$             ;;GO TYPE
 4353  013566  000746                            BR      2$              ;;GO READ ANOTHER CHAR.
 4354  013570  005716                    5$:     TST     (SP)            ;;RUBOUT KEY SET?
 4355  013572  001406                            BEQ     7$              ;;BR IF NO
 4356  013574  112767  000134  000124            MOVB    #'\,9$          ;;TYPE A BACK SLASH
 4357  013602  104401  013726                    TYPE    ,9$
 4358  013606  005016                            CLR     (SP)            ;;CLEAR THE RUBOUT KEY
 4359  013610  122713  000025            7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
 4360  013614  001003                            BNE     8$              ;;BR IF NO
```

```
 4361  013616  104401  013740                TYPE    ,$CNTLU         ;;TYPE A CONTROL 'U'
 4362  013622  000726                        BR      1$              ;;GO START OVER
 4363  013624  122713  000022        8$:     CMPB    #22,(R3)        ;;IS CHARACTER A '^R'?
 4364  013630  001011                        BNE     3$              ;;BRANCH IF NO
 4365  013632  105013                        CLRB    (R3)            ;;CLEAR THE CHARACTER
 4366  013634  104401  001227                TYPE    ,$CRLF          ;;TYPE A 'CR' & 'LF'
 4367  013640  104401  013730                TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
 4368  013644  000717                        BR      2$              ;;GO PICKUP ANOTHER CHACTER
 4369  013646  104401  001226        4$:     TYPE    ,$QUES          ;;TYPE A '?'
 4370  013652  000712                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
 4371  013654  111367  000046        3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 4372  013660  104401  013726                TYPE    ,9$
 4373  013664  122723  000015                CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 4374  013670  001305                        BNE     2$              ;;LOOP IF NOT RETURN
 4375  013672  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 4376  013676  104401  001230                TYPE    ,$LF            ;;TYPE A LINE FEED
 4377  013702  005726                        TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
 4378  013704  012603                        MOV     (SP)+,R3        ;;RESTORE R3
 4379  013706  011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 4380  013710  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 4381  013716  012766  013730  000004        MOV     #$TTYIN,4(SP)
 4382  013724  000002                        RTI                     ;;RETURN
 4383  013726     000        9$:     .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
 4384  013727     000                .BYTE   0                       ;;TERMINATOR
 4385  013730  000010        $TTYIN: .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
 4386  013740  052536  005015   000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U'
 4387  013745     136  006507  000012 $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G'
 4388  013752  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
 4389  013760  020075     000
 4390  013763     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
 4391  013770  036440  000040
 4392                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
 4393
 4394                                  ;;**************************************************************
 4395                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
 4396                                  ;*CHANGE IT TO BINARY.
 4397                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
 4398                                  ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A '?' WILL BE TYPED
 4399                                  ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
 4400                                  ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
 4401                                  ;*CALL:
 4402                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
 4403                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
 4404                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
 4405
 4406  013774  011646        $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
 4407  013776  016666  000004  000002        MOV     4(SP),2(SP)     ;;INPUT NUMBER
 4408  014004  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 4409  014006  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 4410  014010  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 4411  014012  104411        1$:     RDLIN                           ;;READ AN ASCIZ LINE
 4412  014014  012600                        MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
 4413  014016  010067  000100                MOV     R0,5$           ;;AND SAVE IT
 4414  014022  005001                        CLR     R1              ;;CLEAR DATA WORD
 4415  014024  005002                        CLR     R2
 4416  014026  112046        2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
```

```
 4417  014030  001420                        BEQ     3$              ;;IF ZERO GET OUT
 4418  014032  122716  000060                CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
 4419  014036  003026                        BGT     4$              ;;IS AN OCTAL DIGIT
 4420  014040  122716  000067                CMPB    #'7,(SP)
 4421  014044  002423                        BLT     4$
 4422  014046  006301                        ASL     R1              ;;*2
 4423  014050  006102                        ROL     R2
 4424  014052  006301                        ASL     R1              ;;*4
 4425  014054  006102                        ROL     R2
 4426  014056  006301                        ASL     R1              ;;*8
 4427  014060  006102                        ROL     R2
 4428  014062  042716  177770                BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
 4429  014066  062601                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
 4430  014070  000756                        BR      2$              ;;LOOP
 4431  014072  005726                3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
 4432  014074  010166  000012                MOV     R1,12(SP)       ;;SAVE THE RESULT
 4433  014100  010267  000026                MOV     R2,$HIOCT
 4434  014104  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 4435  014106  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 4436  014110  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 4437  014112  000002                        RTI                     ;;RETURN
 4438  014114  005726                4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
 4439  014116  105010                        CLRB    (R0)            ;;SET A TERMINATOR
 4440  014120  104401                        TYPE                    ;;TYPE UP THRU THE BAD CHAR.
 4441  014122  000000                5$:     .WORD   0
 4442  014124  104401  001226                TYPE    ,$QUES          ;;'?' 'CR' & 'LF'
 4443  014130  000730                        BR      1$              ;;TRY AGAIN
 4444  014132  000000                $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
 4445                                  .SBTTL  READ A DECIMAL NUMBER FROM THE TTY
 4446
 4447                                  ;;**********************************************************************
 4448                                  ;*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
 4449                                  ;*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
 4450                                  ;*ARE READ A '?' FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
 4451                                  ;*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
 4452                                  ;*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
 4453                                  ;*POSITIVE 32767 TO NEGATIVE 32768.
 4454                                  ;*CALL:
 4455                                  ;*      RDDEC                   ;;READ A DECIMAL NUMBER
 4456                                  ;*      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
 4457                                  ;
 4458
 4459  014134  011646                $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
 4460  014136  016666  000004  000002        MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
 4461  014144  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 4462  014146  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 4463  014150  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 4464  014152  104411                1$:     RDLIN                   ;;READ AN ASCIZ LINE
 4465  014154  012600                        MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
 4466  014156  010067  000120                MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
 4467  014162  005046                        CLR     -(SP)           ;;CLEAR DATA WORD
 4468  014164  005002                        CLR     R2              ;;SIGN SET POSITIVE
 4469  014166  122710  000055                CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
 4470  014172  001001                        BNE     2$              ;;BR IF NO MINUS SIGN
 4471  014174  112002                        MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
 4472  014176  112001                2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
```

```
 4473  014200  001424                        BEQ     3$              ;;GET OUT IF ZERO
 4474  014202  122701  000060                CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
 4475  014206  003032                        BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
 4476  014210  122701  000071                CMPB    #'9,R1
 4477  014214  002427                        BLT     5$
 4478  014216  032716  170000                BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
 4479  014222  001024                        BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
 4480  014224  006316                        ASL     (SP)            ;;*2
 4481  014226  011646                        MOV     (SP),-(SP)      ;;SAVE FOR LATER
 4482  014230  006316                        ASL     (SP)            ;;*4
 4483  014232  006316                        ASL     (SP)            ;;*8.
 4484  014234  062616                        ADD     (SP)+,(SP)      ;;*10.
 4485  014236  102416                        BVS     5$              ;;OVERFLOW ISN'T ALLOWED
 4486  014240  162701  000060                SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
 4487  014244  060116                        ADD     R1,(SP)         ;;ADD IN THIS DIGIT
 4488  014246  102412                        BVS     5$              ;;OVERFLOW ISN'T ALLOWED
 4489  014250  000752                        BR      2$              ;;LOOP
 4490  014252  005702                3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
 4491  014254  001401                        BEQ     4$              ;;BR IF NO
 4492  014256  005416                        NEG     (SP)            ;;YES--NEGATE THE NUMBER
 4493  014260  012666  000012        4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
 4494  014264  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 4495  014266  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 4496  014270  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 4497  014272  000002                        RTI                     ;;RETURN
 4498
 4499  014274  005726                5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
 4500  014276  105010                        CLRB    (R0)            ;;SET A TERMINATOR
 4501  014300  104401                        TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
 4502  014302  000000                6$:     .WORD   0               ;;POINTER GOES HERE
 4503  014304  104401  001226                TYPE    ,$QUES          ;;''?'' ''CR'' &''LF''
 4504  014310  000720                        BR      1$              ;;TRY AGAIN
 4505                                .SBTTL  TRAP DECODER
 4506
 4507                                ;;**********************************************************************
 4508                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE ''TRAP'' INSTRUCTION
 4509                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 4510                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 4511                                ;*GO TO THAT ROUTINE.
 4512
 4513  014312  010046                $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
 4514  014314  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
 4515  014320  005740                        TST     -(R0)           ;;BACKUP BY 2
 4516  014322  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
 4517  014324  006300                        ASL     R0              ;;POSITION FOR INDEXING
 4518  014326  016000  014346                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
 4519  014332  000200                        RTS     R0              ;;GO TO ROUTINE
 4520
 4521
 4522                                ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO
 4523
 4524  014334  011646                $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
 4525  014336  016666  000004  000002        MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
 4526  014344  000002                        RTI                     ;;RESTORE THE PSW
 4527
 4528                                .SBTTL  TRAP TABLE
```

```
                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                        ;*BY THE ''TRAP'' INSTRUCTION.
                                        ;
                                        ;       ROUTINE
                                        ;       -------
014346  014334                  $TRPAD: .WORD   $TRAP2
014350  012342                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
014352  011714                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
014354  011670                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
014356  011730                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
014360  012116                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

014362  013142                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

014364  013072                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
014366  013354                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
014370  013474                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
014372  013774                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
014374  014134                          $RDDEC  ;;CALL=RDDEC    TRAP+13(104413) READ A DECIMAL NUMBER FROM TTY
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;**********************************************************************
                                ;POWER DOWN ROUTINE
014376  012737  014542  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
014404  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
014412  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
014414  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
014416  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
014420  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
014422  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
014424  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
014426  017746  164506                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
014432  010667  000110                  MOV     SP,$SAVR6       ;;SAVE SP
014436  012737  014450  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
014444  000000                          HALT
014446  000776                          BR      .-2             ;;HANG UP

                                ;;**********************************************************************
                                ;POWER UP ROUTINE
014450  012737  014542  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
014456  016706  000064                  MOV     $SAVR6,SP       ;;GET SP
014462  005067  000060                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
014466  005267  000054          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
014472  001375                          BNE     1$              ;;OF  WORD
014474  012677  164440                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
014500  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
014502  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
014504  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
014506  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
014510  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
014512  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
014514  012737  014376  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
014522  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
014530  104401                          TYPE                    ;REPORT THE POWER FAILURE
014532  014550                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
```

```
 4585  014534  012716                          MOV     (PC)+,(SP)      ;;RESTART AT CKRST1
 4586  014536  014560                  $PWRAD: .WORD   CKRST1          ;;RESTART ADDRESS
 4587  014540  000002                          RTI
 4588  014542  000000                  $ILLJP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
 4589  014544  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
 4590  014546  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
 4591  014550  005015  047520  042527  $POWER: .ASCIZ  <15><12>'POWER'
 4592  014556  000122
 4593                                          .EVEN
```

```
 4594                                   ;;*****************************************************************
 4595                                   ;COMMON DH11 SERVICE ROUTINES
 4596                                   ;;*****************************************************************
 4597
 4598                                   ;THESE ROUTINES DETERMINE RESTART ADDRESS AFTER SYSTEM ERROR
 4599                                   ;(BUS ERROR,RSVD INSTR ERROR, OR POWER FAIL)
 4600
 4601  014560  005767  005554          CKRST1: TST     DPFLG           ;IN PATTERNS TEST ?
 4602  014564  001005                          BNE     1$              ;BR IF YES
 4603  014566  005767  005560                  TST     RETFLG          ;IN ECHO TEST ?
 4604  014572  001004                          BNE     2$              ;BR IF YES
 4605  014574  000167  165612                  JMP     RSTRTA          ;GO RESTART RELIABILITY TESTS
 4606  014600  000167  171266          1$:     JMP     EXPAT           ;GO TO PATTERNS TESTS
 4607  014604  000167  170062          2$:     JMP     ECHO            ;GO TO ECHO TESTS
 4608
 4609  014610  005767  005524          CKRST2: TST     DPFLG           ;IN PATTERNS TEST ?
 4610  014614  001005                          BNE     1$              ;BR IF YES
 4611  014616  005767  005530                  TST     RETFLG          ;IN ECHO TEST ?
 4612  014622  001004                          BNE     2$              ;BR IF YES
 4613  014624  000167  165552                  JMP     REST1           ;GO RESTART RELIABILITY TESTS
 4614  014630  000167  171236          1$:     JMP     EXPAT           ;GO TO PATTERNS TESTS
 4615  014634  000167  170032          2$:     JMP     ECHO            ;GO TO ECHO TESTS
 4616
 4617
 4618                                   ;THIS ROUTINE IS CALLED TO SET UP THE DH11 PARAMETERS PRIOR TO TEST
 4619
 4620  014640  012711  004000          DHSET1: MOV     #BIT11,(R1)     ;CLEAR THE DH11 UNDER TEST
 4621  014644  004767  004564                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
 4622  014650  012711  030100                  MOV     #30100,(R1)     ;ENABLE INTERRUPTS ON XMIT DONE
 4623                                                                  ;NON-EX MEM, DATA AVAIL, OR SILO OVFLW
 4624  014654  156711  005264                  BISB    LINE,(R1)       ;SELECT THE LINE NO.
 4625  014660  005067  004716                  CLR     RDONE           ;CLEAR SOFTWARE DONE FLAG
 4626  014664  012767  030474  013576          MOV     #RBUF,RBFPTR    ;SET UP RCVR BUFFER POINTER
 4627  014672  012767  030474  004704          MOV     #RBUF,RBFEND    ;MARK END OF THIS BUFFER
 4628  014700  016705  004670                  MOV     CHRCNT,R5       ;GET CHAR COUNT
 4629  014704  005405                          NEG     R5              ;MAKE IT POSITIVE
 4630  014706  060567  004672                  ADD     R5,RBFEND
 4631  014712  016761  004652  000004          MOV     CURLPR,LPR(R1)  ;LOAD THE LPR REG
 4632  014720  016761  004650  000010          MOV     CHRCNT,BCR(R1)  ;LOAD THE BYTE COUNT REG
 4633  014726  012761  032754  000006          MOV     #TBUF,CAR(R1)   ;LOAD CURRENT ADDRESS REG
 4634  014734  000207                          RTS     PC              ;RETURN
 4635
 4636                                   ;THIS ROUTINE IS CALLED TO SELECT A NEW LINE NO. BASED ON THE
 4637                                   ;VALUE OF THE LINE SELECTION PARAMETER
 4638
 4639                                           ;CALLING SEQUENCE:
 4640
 4641                                           ;JSR     PC,SELINE      ;CALL THE ROUTINE
 4642                                           ;BR      1$             ;EXIT BRANCH-ROUTINE MOVES THE RETURN
 4643                                                                   ;PC AROUND THIS BR IF MORE LINES ARE
 4644                                                                   ;YET TO BE TESTED
 4645
 4646  014736  105767  005203          SELINE: TSTB    LINE+1          ;FIRST TIME THROUGH FOR ANY TEST ?
 4647  014742  001010                          BNE     1$              ;BR IF NOT
 4648  014744  105167  005175                  COMB    LINE+1          ;SET ENTRY FLAG
 4649  014750  012767  000001  004550          MOV     #1,LINMSK       ;INIT SELECT TEST MASK TO TEST LINE 00
```

```
4650  014756  105067  005162                  CLRB    LINE            ;START WITH LINE #00
4651  014762  000410                          BR      2$              ;GO TEST FOR LINE #00
4652  014764  105267  005154          1$:     INCB    LINE            ;GENERATE NEW LINE NO.
4653  014770  116767  005150  164240          MOVB    LINE,   $TESTN  ; MOVE CURRENT LINE #
4654                                                                  ; TO $TESTN SO APT WILL RUN
4655  014776  006367  004524                  ASL     LINMSK          ;SHIFT SELECT MASK TO TEST NXT LINE
4656  015002  001407                          BEQ     3$              ;RETURN TO EXIT BRANCH - ALL LINES DONE
4657  015004  036767  004516  004512  2$:     BIT     LINMSK,LINSEL   ;IS THE LINE SELECTED FOR TEST ??
4658  015012  001764                          BEQ     1$              ;BR IF NOT
4659  015014  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN PC AROUND EXIT BRANCH
4660  015020  000402                          BR      4$              ;RETURN TO TEST SELECTED LINE
4661  015022  005067  005116          3$:     CLR     LINE            ;INIT ENTRY FLAG AND LINE NO. TO 000
4662  015026  142777  000017  004460  4$:     BICB    #17,@DHADR      ;INIT LINE SELECT BITS IN ''SCR''
4663  015034  000207                          RTS     PC              ;RETURN TO CALLING TEST
4664
4665                                  ;THIS ROUTINE IS CALLED TO CONVERT EITHER THE ''DH'' NUMBER OR THE
4666                                  ;''LINE'' NUMBER TO TWO ASCII CHARACTERS AND MOVE THEM INTO A
4667                                  ;PARTICULAR MESSAGE BUFFER FOR ERROR REPORTING
4668
4669                                          ;CALLING SEQUENCE
4670
4671                                          ;JSR     R5,SUNUM       ;CALL TO THIS ROUTINE
4672                                          ;ADDR1                  ;ADDRESS OF THE NUMBER TO BE CONVERTED
4673                                          ;ADDR2                  ;ADDRESS OF THE MSG BUFFER SLOT
4674
4675  015036                          SUNUM:
4676  015036  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4677  015040  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4678  015042  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4679  015044  012500                          MOV     (R5)+,R0        ;GET ADDRESS OF NUMBER
4680  015046  012501                          MOV     (R5)+,R1        ;GET MSG BUFFER ADDR
4681  015050  111000                          MOVB    (R0),R0         ;GET NO. TO BE CONVERTED
4682  015052  010002                          MOV     R0,R2           ;SAVE IT IN R2
4683  015054  006202                          ASR     R2              ;SHIFT MSD TO LSD POSITION
4684  015056  006202                          ASR     R2
4685  015060  006202                          ASR     R2
4686  015062  042702  177770                 BIC     #177770,R2      ;CLR JUNK BITS
4687  015066  062702  000060                 ADD     #60,R2          ;MAKE IT ASCII
4688  015072  110221                          MOVB    R2,(R1)+        ;PUT IT IN MSG BUFFER
4689  015074  042700  177770                 BIC     #177770,R0      ;CLR JUNK FROM LSD
4690  015100  062700  000060                 ADD     #60,R0          ;MAKE IT ASCII
4691  015104  110011                          MOVB    R0,(R1)         ;PUT LSD IN THE BUFFER
4692  015106  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
4693  015110  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
4694  015112  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
4695  015114  000205                          RTS     R5              ;RETURN TO CALLER
4696
4697                                  ;THIS ROUTINE IS CALLED TO SET UP THE ERROR INFORMATION IN THE
4698                                  ;MESSAGE BUFFERS
4699
4700  015116  010067  164040          SUER2:  MOV     R0,$REG0        ;STORE THE REGS IN CORE
4701  015122  010167  164036          SUER1:  MOV     R1,$REG1
4702  015126  010267  164034                  MOV     R2,$REG2
4703  015132  010367  164032                  MOV     R3,$REG3
4704  015136  010467  164030                  MOV     R4,$REG4
4705  015142  000207                          RTS     PC              ;RETURN TO REPORT ERROR
```

```
4706
4707                                    ;THIS ROUTINE AUTOSIZES THE SYSTEM TO DETERMINE THE ADDRESSES AND
4708                                    ;VECTORS OF THE DH11'S AND DM11-BB'S.
4709
4710  015144  010046                    AUTOSZ: MOV     R0,-(SP)
4711  015146  005003                            CLR     R3
4712  015150  012702  021750                    MOV     #DHADRS,R2
4713  015154  005022                    25$:    CLR     (R2)+           ;CLEAR DH TABLES.
4714  015156  005203                            INC     R3
4715  015160  020327  000102                    CMP     R3,#102         ;HAVE WE CLEARED ALL ENTRIES?
4716  015164  001373                            BNE     25$             ;BRANCH IF NOT.
4717  015166  013746  000004                    MOV     @#4,-(SP)       ;SAVE TRAP VECTOR.
4718  015172  012737  015300  000004          MOV     #4$,@#4         ;SETUP FOR NON-EXISTENT MEMORY TRAP.
4719  015200  012703  022042                    MOV     #DMADRS,R3      ;SETUP DM ADDRESS TABLE POINTER.
4720  015204  012702  021750                    MOV     #DHADRS,R2      ;SET UP DH ADDRESS TABLE POINTER.
4721
4722  015210  012701  160020                    MOV     #160020,R1      ;R1-FIRST ADDRESS TO BE TESTED.
4723
4724  015214  005711                    1$:     TST     (R1)            ;SEE IF ADDRESS IN R1 RESPONDS.
4725  015216  005761  000016                    TST     16(R1)          ;CHECK TO SEE IF DEVICE IS MODULO 20.
4726  015222  052711  004000                    BIS     #4000,(R1)      ;IF IT IS, CONTINUE
4727                                            ;AND CHECK TO SEE
4728  015226  052711  001000                    BIS     #1000,(R1)      ;IF THIS ADDRESS CONTAINS
4729  015232  052711  002000                    BIS     #2000,(R1)      ;A DH-11.
4730  015236  032711  003000                    BIT     #3000,(R1)      ;CHECK TO INSURE THESE BITS SET.
4731  015242  001410                            BEQ     3$              ;IF NOT, BRANCH.
4732                                            ;SET THE MAINTENANCE BIT, THE NON-
4733  015244  052711  000400                    BIS     #400,(R1)       ;EXISTENT MEMORY BIT AND THE CLEAR
4734                                            ;NON-EXISTENT MEMORY INTERRUPT BIT.
4735  015250  032711  002400                    BIT     #2400,(R1)      ;IS THIS A DH-11? (BITS 8 AND 10 SHOULD
4736                                            ;CLEAR IF THIS IS A DH11.)
4737
4738  015254  001003                            BNE     3$              ;IF NOT, CHECK TO SEE IF THIS IS A DM11-BB.
4739  015256  042711  001000                    BIC     #1000,(R1)      ;CLEAR MAINTENANCE BIT.
4740  015262  010122                            MOV     R1,(R2)+        ;SAVE THE ADDRESS IN THE DH ADR TABLE.
4741
4742
4743  015264  020127  163760            3$:     CMP     R1,#163760      ;HAVE WE REACHED THE TOP OF THE FLOATING ADDRESSES.
4744  015270  001406                            BEQ     5$              ;IF YES, GET OUT.
4745  015272  062701  000020                    ADD     #20,R1          ;IF NOT, UPDATE ADDRESS AND
4746  015276  000746                            BR      1$              ;GO CHECK IT.
4747
4748  015300  012716  015264            4$:     MOV     #3$,(SP)        ;IF DH ADDRESS DOES NOT RESPOND, GO TO 3$.
4749  015304  000002                            RTI
4750
4751                                    ;TEST FOR DM11 BB ADDRESS
4752
4753  015306  012737  015340  000004    5$:     MOV     #6$,@#4         ;SETUP FOR NON-EXISTENT MEMORY TRAP.
4754  015314  012701  170500                    MOV     #170500,R1      ;R1=FIRST ADDRESS TO BE TESTED.
4755  015320  005711                    21$:    TST     (R1)            ;SEE IF ADDRESS RESPONDS.
4756  015322  010123                            MOV     R1,(R3)+        ;IF IT DOES, THIS IS A DM11-BB,
4757                                            ;SO SAVE THE ADDRESS.
4758  015324  020127  170670            23$:    CMP     R1,#170670      ;HAVE WE REACHED THE TOP OF THE DMTT ADDRESSES?
4759  015330  001406                            BEQ     22$             ;IF YES, GET OUT.
4760  015332  062701  000010                    ADD     #10,R1          ;IF NOT, UPDATE ADDRESS AND
4761  015336  000770                            BR      21$             ;GO CHECK IT.
```

```
 4762
 4763  015340  012716  015324          6$:     MOV     #23$,(SP)       ;IF DM ADDRESS DOES NOT RESPOND, GO TO 23$.
 4764  015344  000002                          RTI
 4765
 4766  015346  012637  000004          22$:    MOV     (SP)+,@#4       ;RESTORE TRAP VECTOR.
 4767  015352  162702  021750                  SUB     #DHADRS,R2      ;HAVE WE FOUND ANY DH11'S AT ALL?
 4768  015356  001003                          BNE     7$              ;IF YES, BRANCH
 4769  015360  104401  025712                  TYPE    ,MSG1           ;NO DH11'S WERE FOUND,
 4770  015364  000000                          HALT
 4771
 4772  015366  006202                  7$:     ASR     R2              ;R2 NOW CONTAINS THE NUMBER
 4773  015370  005000                          CLR     R0              ;OF DH'S FOUND.
 4774  015372  006100                  8$:     ROL     R0              ;FILL R0 WITH 1'S
 4775  015374  005200                          INC     R0              ;CORRESPONDING TO
 4776  015376  005302                          DEC     R2              ;THE NUMBER OF DH'S
 4777  015400  005702                          TST     R2              ;FOUND.
 4778  015402  001373                          BNE     8$
 4779  015404  010067  004526                  MOV     R0,$DHSEL       ;$DHSEL CONTAINS THE DH SELECTION PARAMETER.
 4780                                                                  ;IE. ALL DH'S FOUND WILL BE TESTED.
 4781
 4782                                  ;FIND DH VECTOR:
 4783  015410  012702  021750                  MOV     #DHADRS,R2      ;SETUP POINTER TO BEGINNING OF DH
 4784  015414  012705  022006                  MOV     #DHVEC,R5       ;ADDRESS TABLE AND VECTOR TABLE.
 4785  015420  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;SET IOT TRAP PRIORITY TO 7.
 4786  015426  012737  015536  000020          MOV     #12$,@#IOTVEC   ;SETUP IOT TRAP VECTOR.
 4787  015434  012703  000300                  MOV     #300,R3         ;START OF FLOATING VECTORS
 4788  015440  012704  000302                  MOV     #302,R4         ;PC OF IOT INSTR.
 4789
 4790  015444  010423                  9$:     MOV     R4,(R3)+        ;FILL VECTOR AREA WITH ADDRESS
 4791                                                                  ;OF NEXT INSTR (.+2)
 4792  015446  012724  000004                  MOV     #4,(R4)+        ;NEXT INSTRUCTION IS AN IOT TRAP.
 4793  015452  022324                          CMP     (R3)+,(R4)+     ;UPDATE R3+R4.
 4794  015454  020427  001000                  CMP     R4,#1000        ;HAVE WE REACHED TO TOP OF THE
 4795                                                                  ;VECTOR SPACE?
 4796  015460  101771                          BLOS    9$              ;IF NOT, REPEAT PROCESS.
 4797
 4798  015462  005712                  10$:    TST     (R2)            ;HAVE WE CHECK ALL DH'S?
 4799  015464  001441                          BEQ     13$             ;IF YES, GET OUT + CHECK FOR DM11 BB'S VECTORS.
 4800
 4801  015466  005067  162304                  CLR     PS              ;ZERO CPU PRIORITY.
 4802  015472  052772  001000  000000          BIS     #1000,@(R2)     ;SET MAINTENANCE BIT
 4803  015500  052772  000300  000000          BIS     #300,@(R2)      ;ATTEMPT TO CAUSE RECEIVER
 4804                                                                  ;INTERRUPT.
 4805  015506  005000                          CLR     R0
 4806
 4807  015510  005200                  11$:    INC     R0              ;WAIT...
 4808  015512  001376                          BNE     11$
 4809  015514  104401  025741                  TYPE    ,MSG2           ;ERROR MSG-NO DH RECEIVER INTERRUPT OCCURRED.
 4810  015520  052772  004000  000000          BIS     #4000,@(R2)     ;DO A MASTER CLEAR
 4811  015526  042772  001000  000000          BIC     #1000,@(R2)     ;CLEAR MAINTENANCE BIT
 4812  015534  000752                          BR      10$
 4813
 4814  015536  011601                  12$:    MOV     (SP),R1
 4815  015540  042701  000007                  BIC     #7,R1           ;CLEAR GARBAGE.
 4816  015544  010125                          MOV     R1,(R5)+        ;SAVE VECTOR ADDRESS.
 4817  015546  022626                          CMP     (SP)+,(SP)+     ;POP STACK
```

```
4818  015550  012716  015462           MOV     #10$,(SP)       ;SETUP FOR RETURN.
4819  015554  052772  004000  000000   BIS     #4000,@(R2)     ;DO A MASTER CLEAR
4820  015562  042732  001000           BIC     #1000,@(R2)+    ;CLEAR MAINTENANCE BIT.
4821  015566  000002                   RTI
4822
4823                                  ;FIND DM11 BB VECTORS:
4824
4825  015570  012702  022042   13$:    MOV     #DMADRS,R2      ;SET POINTERS TO BEGINNING OF
4826  015574  012705  022100           MOV     #DMVEC,R5       ;ADR TABLE & VECTOR TABLE.
4827  015600  012737  015662  000020   MOV     #16$,@#IOTVEC   ;SET IOT TRAP VECTOR.
4828
4829  015606  005712           14$:    TST     (R2)            ;HAVE WE CHECKED ALL DM'S?
4830  015610  001441           BEQ     17$             ;IF YES, GET OUT.
4831  015612  005067  162160           CLR     PS              ;ZERO CPU PRIORITY
4832  015616  052772  001000  000000   BIS     #1000,@(R2)     ;SET MAINTENANCE BIT.
4833  015624  052772  000300  000000   BIS     #300,@(R2)      ;ATTEMPT TO CAUSE INTERRUPT.
4834  015632  005000           CLR     R0
4835
4836  015634  005200           15$:    INC     R0              ;WAIT....
4837  015636  001376           BNE     15$
4838  015640  104401  026005           TYPE    ,MSG3           ;ERROR MSG - NO DM11-BB INTERRUPT OCCURRED.
4839  015644  052772  004000  000000   BIS     #4000,@(R2)     ;CLEAR BITS PREVIOUSLY SET.
4840  015652  042772  001000  000000   BIC     #1000,@(R2)     ;CLEAR MAINTENANCE BIT.
4841  015660  000752           BR      14$
4842
4843  015662  011601           16$:    MOV     (SP),R1
4844  015664  162701  000004           SUB     #4,R1           ;CALCULATE VECTOR ADDRESS.
4845  015670  010125           MOV     R1,(R5)+        ;SAVE VECTOR ADDRESS.
4846  015672  022626           CMP     (SP)+,(SP)+     ;POP STACK.
4847  015674  012716  015606           MOV     #14$,(SP)       ;SETUP FOR RETURN.
4848  015700  052772  004000  000000   BIS     #4000,@(R2)     ;CLEAR BITS PREVIOUSLY SET.
4849  015706  042732  001000           BIC     #1000,@(R2)+    ;CLEAR MAINTENANCE BIT AND
4850                                                          ;POINT TO NEXT DM11-BB ADDRESS.
4851  015712  000002           RTI
4852
4853  015714  012737  011076  000020   17$:    MOV     #$SCOPE,@#IOTVEC  ;RESTORE IOT VECTOR FOR SCOPE ROUTINE.
4854  015722  012600           MOV     (SP)+,R0        ;RESTORE R0.
4855  015724  012703  000300           MOV     #300,R3         ;START OF FLOATING VECTORS.
4856  015730  012704  000302           MOV     #302,R4
4857
4858  015734  010423           18$:    MOV     R4,(R3)+        ;FILL VECTOR AREA WITH ADDRESS OF NEXT
4859                                                          ;INSTRUCTION (.+2).
4860  015736  012724  000000           MOV     #0,(R4)+        ;NEXT INSTRUCTION IS A HALT.
4861  015742  022324           CMP     (R3)+,(R4)+     ;UPDATE R3 & R4.
4862  015744  020427  001000           CMP     R4,#1000        ;ARE WE DONE?
4863  015750  101771           BLOS    18$             ;IF NOT, REPEAT UNTIL ADDRESSES
4864                                                          ;377 TO 777 ARE DONE.
4865  015752  013701  022006           MOV     @#DHVEC,R1      ;LET R1 POINT TO 1ST DH VECTOR ADDRESS.
4866  015756  005737  022010           TST     @#DHVEC+2       ;IS THERE MORE THAN ONE ENTRY?
4867  015762  001403           BEQ     26$             ;BRANCH IF NO.
4868  015764  163701  022010           SUB     @#DHVEC+2,R1    ;DETERMINE NUMBER OF ADDRESSES
4869                                                          ;BETWEEN DH VECTORS (10(8) OR 20(8)).
4870  015770  005401           NEG     R1              ;MAKE IT POSITIVE.
4871  015772  010167  004136   26$:    MOV     R1,ADRVEC       ;SAVE THAT NUMBER.
4872  015776  032777  000002  163134   BIT     #BIT1,@SWR      ;SHOULD DEVICE MAP BE TYPED OUT?
4873  016004  001441           BEQ     20$             ;IF NOT, RETURN.
```

```
 4874  016006  104401                        TYPE                    ;TYPEOUT MAP OF DH & DM11-BB'S
 4875  016010  026052                        DEVMAP                  ;FOUND.
 4876  016012  012701  021750                MOV     #DHADRS,R1      ;R1=BEGINNING OF DH ADDRESS TABLE.
 4877  016016  012702  022006                MOV     #DHVEC,R2       ;R2=BEGINNING OF DH VECTOR TABLE.
 4878  016022  012703  022042                MOV     #DMADRS,R3      ;R3=BEGINNING OF DM11-BB ADDRESS TABLE.
 4879  016026  012704  022100                MOV     #DMVEC,R4       ;R4=BEGINNING OF DM11-BB VECTOR TABLE.
 4880  016032  005005                        CLR     R5              ;CLEAR TABLE LINE COUNTER
 4881
 4882  016034  012146                19$:    MOV     (R1)+,-(SP)     ;MOVE DATA TO BE TYPED
 4883  016036  104403                        TYPOS                   ;TYPE DATA
 4884  016040     006                        .BYTE   6
 4885  016041     001                        .BYTE   1
 4886  016042  012246                        MOV     (R2)+,-(SP)     ;MOVE DATA TO BE TYPED
 4887  016044  104403                        TYPOS                   ;TYPE DATA
 4888  016046     005                        .BYTE   5
 4889  016047     000                        .BYTE   0
 4890  016050  104401  026046                TYPE    ,SPACE
 4891  016054  012346                        MOV     (R3)+,-(SP)     ;MOVE DATA TO BE TYPED.
 4892  016056  104403                        TYPOS                   ;TYPE DATA.
 4893  016060     006                        .BYTE   6
 4894  016061     001                        .BYTE   1
 4895  016062  104401  026046                TYPE    ,SPACE
 4896  016066  012446                        MOV     (R4)+,-(SP)     ;MOVE DATA TO BE TYPED.
 4897  016070  104403                        TYPOS                   ;TYPE DATA.
 4898  016072     005                        .BYTE   5
 4899  016073     000                        .BYTE   0
 4900  016074  104401                        TYPE                    ;TYPE A CARRIAGE RETURN & LINE FEED.
 4901  016076  001227                        $CRLF
 4902  016100  005711                        TST     (R1)            ;HAVE WE TYPED ALL DH ENTRIES?
 4903  016102  001354                        BNE     19$             ;IF NOT, DO IT AGAIN.
 4904  016104  104401  001227                TYPE    ,$CRLF
 4905  016110  000207                20$:    RTS     PC              ;IF YES, GO BACK TO MAIN PROGRAM.
 4906
 4907                                        ;THIS ROUTINE IS USED TO ACCEPT INPUT PARAMETERS FROM THE CONSOLE
 4908                                        ;TELETYPE
 4909
 4910  016112  104401                INPARA: TYPE
 4911  016114  025417                        VCWC                    ;"ASK FOR NO. ADDRESSES BETWEEN VECTORS"
 4912  016116  104412                        RDOCT                   ;READ OCTAL NO. FM TTY
 4913  016120  012600                        MOV     (SP)+,R0        ;GET THE NO. HE TYPED
 4914  016122  001407                        BEQ     3$              ;BR IF HE TYPED <CR>
 4915  016124  022700  000010                CMP     #10,R0          ;10(8) ADDRESSES BETWEEN VECTORS ?
 4916  016130  001406                        BEQ     4$              ;BR IF YES
 4917  016132  022700  000020                CMP     #20,R0          ;20(8) ADDRESSES BETWEEN VECTORS ??
 4918  016136  001403                        BEQ     4$              ;BR IF YES
 4919  016140  000764                        BR      INPARA          ;ASK ALL OVER AGAIN
 4920  016142  012700  000020        3$:     MOV     #20,R0          ;SET UP CONSTANT FOR 20(8) ADDRESSES
 4921  016146  005067  004166        4$:     CLR     DPFLG           ;CLEAR PATTERNS TESTS FLAG
 4922  016152  005067  004174                CLR     RETFLG          ;INIT ECHO TEST RETURN FLAG
 4923  016156  000207                        RTS     PC              ;RETURN TO CALLER
 4924
 4925  016160  012767  177777  003520 INPARX: MOV     #-1,VCFLG       ;SET START AT 200 FLAG
 4926  016166  000167  163422                JMP     BEGINA          ;GO START UP
 4927  016172  012700  177777        INPARC: MOV     #-1,R0          ;SET FLAG IN R0
 4928  016176  005067  003504                CLR     VCFLG           ;INIT VECTOR SET UP FLAG
 4929  016202  005067  004132                CLR     DPFLG           ;CLEAR PATTERNS TESTS FLAG
```

```
 4930  016206  005067  004140                CLR     RETFLG          ;INIT ECHO TEST RETURN FLAG
 4931  016212  000167  163376                JMP     BEGINA          ;GO ASK FOR SELECT PARAMETER
 4932
 4933  016216  013701  021750        INPAR:  MOV     @#DHADRS,R1     ;MOVE ADDRESS OF FIRST DH INTO R1.
 4934  016222  032777  000001  162710        BIT     #BIT0,@SWR      ;ARE PARAMETERS TO BE INPUT MANUALLY?
 4935  016230  001405                        BEQ     2$              ;BRANCH IF NOT.
 4936  016232  104401                1$:     TYPE                    ;ASK FOR DEVICE ADDRESS
 4937  016234  024726                        INMSG1
 4938  016236  104412                        RDOCT                   ;READ IN WHAT IS TYPED
 4939  016240  012601                        MOV     (SP)+,R1        ;GET THE NO. HE TYPED
 4940  016242  001403                        BEQ     INPAR1          ;BR IF DEFAULT
 4941  016244  004767  000176        2$:     JSR     PC,CHKADR       ;GO CHECK VALIDITY OF THE ADDR
 4942  016250  000770                        BR      1$              ;ERROR BRANCH
 4943
 4944  016252  013701  022006        INPAR1: MOV     @#DH.EC,R1      ;MOVE FIRST DH VECTOR INTO R1.
 4945  016256  032777  000001  162654        BIT     #BIT0,@SWR      ;ARE PARAMETERS TO BE INPUT MANUALLY?
 4946  016264  001405                        BEQ     2$              ;BRANCH IF NOT.
 4947  016266  104401                1$:     TYPE                    ;ASK FOR VECTOR ADDRESS
 4948  016270  024772                        INMSG2
 4949  016272  104412                        RDOCT                   ;READ IN WHAT HE TYPES
 4950  016274  012601                        MOV     (SP)+,R1        ;GET THE ADDRESS
 4951  016276  001403                        BEQ     INPAR3          ;BR IF DEFAULT
 4952  016300  004767  000306        2$:     JSR     PC,CHKVCT       ;GO CHECK VALIDITY OF VECTOR
 4953  016304  000770                        BR      1$              ;ERROR BRANCH
 4954
 4955  016306  005767  004040        INPAR3: TST     RETFLG          ;LINE ECHO TESTS ?
 4956  016312  001402                        BEQ     1$              ;BR IF NOT
 4957  016314  000167  166420                JMP     ECHO2           ;RETURN TO LINE ECHO TESTS
 4958  016320  005767  004014        1$:     TST     DPFLG           ;DATA PATTERNS TESTS ACTIVE ??
 4959  016324  001402                        BEQ     2$              ;BR IF NOT
 4960  016326  000167  167606                JMP     EXPAT2          ;GO BACK TO PATTERNS TESTS
 4961  016332  013701  022136        2$:     MOV     @#$DHSEL,R1     ;MOVE DEVICE SELECTION PARAMETER INTO R1.
 4962  016336  005700                        TST     R0              ;START AT 210?
 4963  016340  100404                        BMI     4$              ;BRANCH IF YES.
 4964  016342  032777  000001  162570        BIT     #BIT0,@SWR      ;IS PARAMETER TO BE INPUT MANUALLY?
 4965  016350  001405                        BEQ     3$              ;BRANCH IF NOT.
 4966  016352  104401                4$:     TYPE                    ;ASK FOR DEVICE SELECTION PARAMETER
 4967  016354  025041                        INMSG3
 4968  016356  104412                        RDOCT                   ;READ IN WHAT HE TYPES
 4969  016360  012601                        MOV     (SP)+,R1        ;GET THE SELECT PARAMETER
 4970  016362  001402                        BEQ     INPAR4          ;BR IF  DEFAULT
 4971  016364  010167  003132        3$:     MOV     R1,DHSEL        ;SET UP DH11 SELECTION PARAMETER
 4972  016370  012767  177777  003126 INPAR4: MOV     #-1,LINSEL      ;INIT FOR ALL 16. LINES
 4973  016376  032777  000001  162534        BIT     #BIT0,@SWR      ;IS LINE SELECT PARAMETER TO BE INPUT MANUALLY?
 4974  016404  001407                        BEQ     1$              ;BRANCH IF NO.
 4975  016406  104401                        TYPE                    ;ASK FOR LINE SELECT PARAMETER
 4976  016410  025355                        INMSGA
 4977  016412  104412                        RDOCT                   ;READ WHAT HE TYPES
 4978  016414  012601                        MOV     (SP)+,R1        ;GET IT OFF STACK
 4979  016416  001402                        BEQ     1$              ;BR IF DEFAULT
 4980  016420  010167  003100                MOV     R1,LINSEL       ;SET LINE SELECT PARAMETER
 4981  016424  032777  000400  162506 1$:     BIT     #BIT8,@SWR      ;HALT AFTER SET UP ??
 4982  016432  001403                        BEQ     EXPAR           ;BR IF  NOT
 4983  016434  104401                        TYPE                    ;TYPE CONTINUE MESSAGE PRIOR TO HALTING
 4984  016436  025305                        INMSG7
 4985  016440  000000                        HALT                    ;DEPRESS CONTINUE TO RESUME TESTING
```

```
 4986  016442  000167  163640          EXPAR:  JMP     START2          ;GO START UP THE PROGRAM
 4987
 4988
 4989  016446  020127  160020          CHKADR: CMP     R1,#160020      ;IS ADDRESS ABOVE OR EQUAL TO LOW LIMIT
 4990  016452  002001                          BGE     1$              ;BR IF YES
 4991  016454  000453                          BR      4$              ;BR IF NOT
 4992  016456  020127  160420          1$:     CMP     R1,#160420      ;IS IT BELOW THE HIGH LIMIT?
 4993  016462  002401                          BLT     2$              ;BR IF YES
 4994  016464  000447                          BR      4$              ;BR IF NOT
 4995  016466  032701  000017          2$:     BIT     #17,R1          ;CORRECT BOUNDARY ?
 4996  016472  001044                          BNE     4$              ;BR IF NOT
 4997  016474  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN PC AROUND ERROR BRANCH
 4998  016500  005767  003646                  TST     RETFLG          ;ARE WE IN ECHO TESTS ?
 4999  016504  001403                          BEQ     21$             ;BR IF NOT
 5000  016506  010167  003002                  MOV     R1,DHADR        ;SET UP DH11 DEVICE ADDRESS
 5001  016512  000436                          BR      5$              ;CONTINUE
 5002  016514  005767  003620          21$:    TST     DPFLG           ;PATTERNS TESTS ACTIVE ??
 5003  016520  001403                          BEQ     22$             ;BR IF NOT
 5004  016522  010167  002766                  MOV     R1,DHADR        ;SET UP DEVICE ADDRESS
 5005  016526  000430                          BR      5$              ;CONTINUE
 5006  016530  012702  021606          22$:    MOV     #DHADTB,R2      ;POINT TO BEGIN OF ADDR TABLE
 5007  016534  032777  000001  162376          BIT     #BIT0,@SWR      ;ARE WE AUTOSIZING?
 5008  016542  001011                          BNE     3$              ;BRANCH IF NOT.
 5009  016544  012703  021750                  MOV     #DHADRS,R3      ;POINT TO BEGINNING OF AUTOSIZER
 5010                                          ;DH ADDRESS TABLE.
 5011  016550  016704  003362                  MOV     $DHSEL,R4
 5012  016554  012322                  6$:     MOV     (R3)+,(R2)+     ;MOVE CONTENTS OF AUTOSIZER DH TABLE
 5013                                          ;TO THE TABLE USED BY PROGRAM.
 5014  016556  006204                          ASR     R4
 5015  016560  005704                          TST     R4              ;HAVE WE MOVED ALL TABLE ENTRIES?
 5016  016562  001374                          BNE     6$              ;BRANCH IF NOT--ONE MORE TIME.
 5017  016564  000411                          BR      5$              ;RETURN TO INPUT ROUTINES.
 5018  016566  010122                  3$:     MOV     R1,(R2)+        ;SET UP A TABLE ENTRY
 5019  016570  062701  000020                  ADD     #20,R1          ;GENERATE NEXT DH11 ADDR
 5020  016574  022702  021646                  CMP     #DHADTB+40,R2   ;END OF TABLE ?
 5021  016600  001372                          BNE     3$              ;BR IF NOT
 5022  016602  000402                          BR      5$              ;RETURN TO INPUT ROUTINES
 5023  016604  104401                  4$:     TYPE                    ;TELL HIM HE GOOFED
 5024  016606  025112                          INMSG4
 5025  016610  000207                  5$:     RTS     PC              ;RETURN TO INPUT ROUTINES
 5026
 5027  016612  020127  000300          CHKVCT: CMP     R1,#300         ;IS ADDRESS ABOVE OR EQUAL TO LOW LIMIT
 5028  016616  002001                          BGE     1$              ;BR IF YES
 5029  016620  000452                          BR      4$              ;BR IF NOT
 5030  016622  020127  001000          1$:     CMP     R1,#1000        ;IS IT BELOW THE HIGH LIMIT?
 5031  016626  002401                          BLT     2$              ;BR IF YES
 5032  016630  000446                          BR      4$              ;BR IF NOT
 5033  016632  032701  000007          2$:     BIT     #7,R1           ;CORRECT BOUNDARY ?
 5034  016636  001043                          BNE     4$              ;BR IF NOT
 5035  016640  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN PC AROUND ERROR BRANCH
 5036  016644  005767  003502                  TST     RETFLG          ;ARE WE IN ECHO TESTS ?
 5037  016650  001403                          BEQ     21$             ;BR IF NOT
 5038  016652  010167  002640                  MOV     R1,DHVCT        ;SET UP DH11 VECTOR ADDR
 5039  016656  000435                          BR      5$              ;CONTINUE
 5040  016660  005767  003454          21$:    TST     DPFLG           ;PATTERNS TESTS ACTIVE ??
 5041  016664  001403                          BEQ     22$             ;BR IF NOT
```

```
5042  016666  010167  002624            MOV     R1,DHVCT        ;SET UP DEVICE VECTOR
5043  016672  000427                    BR      5$              ;CONTINUE
5044  016674  012702  021646    22$:    MOV     #DHVCTB,R2      ;POINT TO BEGIN OF VECTOR TABLE
5045  016700  032777  000001  162232    BIT     #BIT0,@SWR      ;ARE WE AUTOSIZING?
5046  016706  001011                    BNE     3$              ;BRANCH IF NOT.
5047  016710  012703  022006            MOV     #DHVEC,R3       ;POINT TO BEGINING OF AUTOSIZER
5048                                                            ;DH VECTOR TABLE.
5049  016714  016704  003216            MOV     $DHSEL,R4
5050  016720  012322            6$:     MOV     (R3)+,(R2)+     ;MOVE CONTENTS OF AUTOSIZER VECTOR
5051                                                            ;TABLE TO TABLE USED BY PROGRAM.
5052  016722  006204                    ASR     R4
5053  016724  005704                    TST     R4              ;HAVE WE MOVED ALL TABLE ENTRIES?
5054  016726  001374                    BNE     6$              ;BRANCH IF NOT--ONE MORE TIME.
5055  016730  000410                    BR      5$              ;RETURN TO INPUT ROUTINES.
5056  016732  010122            3$:     MOV     R1,(R2)+        ;SET UP A TABLE ENTRY
5057  016734  060001                    ADD     R0,R1           ;GENERATE NEXT DH11 ADDR
5058  016736  022702  021706            CMP     #DHVCTB+40,R2   ;END OF TABLE ?
5059  016742  001373                    BNE     3$              ;BR IF NOT
5060  016744  000402                    BR      5$              ;RETURN TO INPUT ROUTINES
5061  016746  104401            4$:     TYPE                    ;TELL HIM HE GOOFED
5062  016750  025163                    INMSG5
5063  016752  000207            5$:     RTS     PC              ;RETURN TO INPUT ROUTINES
5064
5065                                    ;THESE TWO ROUTINES SERVICE UNEXPECTED BUS ERROR AND RSVD INSTR TRAPS
5066
5067  016754  010667  162216    BUSER:  MOV     SP,$REG6        ;SAVE THE SP
5068  016760  012667  162200            MOV     (SP)+,$REG1     ;GET THE TRAP PC
5069  016764  012667  162176            MOV     (SP)+,$REG2     ;GET THE TRAP PSW
5070  016770  012706  001100            MOV     #STACK,SP       ;RESET THE STACK POINTER
5071  016774  012767  017004  162106    MOV     #1$,$LPERR      ;ALWAYS COME BACK TO 1$
5072  017002  104014                    ERROR   14              ;UNEXPECTED BUS ERROR TRAP
5073  017004  000005            1$:     RESET                   ;PREPARE TO RESTART
5074  017006  004767  002406            JSR     PC,CHPS1        ;GO CLEAR PSW
5075  017012  000167  175572            JMP     CKRST2          ;GO RESTART THE PROGRAM
5076
5077  017016  010667  162154    RESERR: MOV     SP,$REG6        ;SAVE THE SP
5078  017022  012667  162136            MOV     (SP)+,$REG1     ;GET THE TRAP PC
5079  017026  012667  162134            MOV     (SP)+,$REG2     ;GET THE TRAP PSW
5080  017032  012706  001100            MOV     #STACK,SP       ;RESET THE STACK POINTER
5081  017036  012767  017046  162044    MOV     #1$,$LPERR      ;ALWAYS COME BACK TO 1$
5082  017044  104015                    ERROR   15              ;UNEXPECTED RSVD INSTR ERROR TRAP
5083  017046  000005            1$:     RESET                   ;PREPARE TO RESTART
5084  017050  004767  002344            JSR     PC,CHPS1        ;GO CLEAR PSW
5085  017054  000167  175530            JMP     CKRST2          ;GO RESTART THE PROGRAM
5086
5087                                    ;THIS ROUTINE IS CALLED WHEN A TEST NEEDS TO RESTORE THE TRAP
5088                                    ;CATCHER IN THE DH11 VECTOR
5089
5090  017060  016703  002432    RESTRP: MOV     DHVCT,R3        ;GET VECTOR ADDRESS
5091  017064  010313                    MOV     R3,(R3)         ;RESTORE THE TRAP CATCHER
5092  017066  062723  000002            ADD     #2,(R3)+
5093  017072  005023                    CLR     (R3)+
5094  017074  010313                    MOV     R3,(R3)
5095  017076  062723  000002            ADD     #2,(R3)+
5096  017102  005023                    CLR     (R3)+
5097  017104  000207                    RTS     PC              ;RETURN TO CALLING TEST
```

```
CZDHN-D MACY11 30A(1052)  27-DEC-78  15:31  PAGE 113
CZDHND.P11    27-DEC-78 15:28          POWER DOWN AND UP ROUTINES                                    SEQ 0112

 5098
 5099                                  ;THIS ROUTINE CALLED BY ANY TEST THAT NEEDS A TIMING WAIT LOOP
 5100                                  ;''TIMEA'' IS INITIALIZED BY THE CALLING ROUTINE TO THE MINIMUM REQUIRED
 5101                                  ;VALUE AND ''TIMEB'' IS CLEARED TO 000000. IF A TIME OUT OCCURS THIS
 5102                                  ;ROUTINE WILL MOVE THE RETURN PC AROUND THE 'LOOP'' BRANCH BACK IN
 5103                                  ;THE ROUTINE THAT CALLED IT TO ALLOW REPORTING AN ERROR MESSAGE
 5104
 5105  017106  005267  003222          TIMEIT: INC     TIMEB           ;COUNT B
 5106  017112  001005                          BNE     1$              ;BR IF NOT ZERO
 5107  017114  005367  003212                  DEC     TIMEA           ;COUNT TIME A
 5108  017120  001002                          BNE     1$              ;BR IF NO TIMEOUT
 5109  017122  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN PC TO ALLOW ERROR REPORT
 5110  017126  000207                  1$:     RTS     PC              ;RETURN TO THE CALLING TEST
 5111
 5112
 5113
 5114                                  ;THIS ROUTINE IS CALLED TO CLEAR ALL ENTRIES IN THE STATISTICS TABLES
 5115
 5116  017130  012705  030150          CLSTAT: MOV     #RTOTAL,R5      ;SET UP POINTER TO BEGINNING
 5117  017134  005025                  1$:     CLR     (R5)+           ;CLEAR ONE WORD
 5118  017136  022705  030460                  CMP     #RTOTAL+192.,R5 ;CLEARED ALL ENTRIES ??
 5119  017142  001374                          BNE     1$              ;BR IF NOT
 5120  017144  000207                          RTS     PC
 5121
 5122                                  ;THIS ROUTINE IS CALLED TO RETRIEVE A NEW LPR CONSTANT
 5123                                  ;FROM THE LPR TABLE (LPRTAB)
 5124
 5125                                  ;CALLING SEQUENCE:
 5126
 5127                                  ;       JSR     R5,SETLPR       ;CALL
 5128                                  ;       BR      NEWLIN          ;EXIT BRANCH - EXECUTED AFTER ALL
 5129                                                                  ;13 BAUD RATES EXERCISED
 5130
 5131  017146  022767  021570  002416  SETLPR: CMP     #CURLPR,LPRPTR  ;DONE ALL 13. ENTRIES ??
 5132  017154  001425                          BEQ     3$              ;BR IF YES
 5133  017156  017767  002410  002404          MOV     @LPRPTR,CURLPR  ;GET THE LPR CONSTANT
 5134  017164  105777  161750                  TSTB    @SWR            ; QUICK TEST ?
 5135  017170  100410                          BMI     1$              ;BR IF NOT - SUPPLY THE WHOLE THING
 5136  017172  022767  033500  002370          CMP     #33500,CURLPR   ;9600 BAUD TEST ??
 5137  017200  001404                          BEQ     1$              ;BR IF YES
 5138  017202  012767  177777  002322          MOV     #-1,QUICK       ;SET QUICK TEST FLAG
 5139  017210  000402                          BR      2$              ;CONTINUE
 5140  017212  005067  002314          1$:     CLR     QUICK           ;DO FULL TESTING AT 9600. BAUD
 5141  017216  062767  000002  002346  2$:     ADD     #2,LPRPTR       ;UPDATE THE TABLE POINTER
 5142  017224  062705  000002                  ADD     #2,R5           ;MOVE PC AROUND ERROR BRANCH
 5143  017230  000205                  3$:     RTS     R5              ;RETURN
 5144
 5145                                  ;THIS ROUTINE IS CALLED TO SETUP THE CHAR LENGTH SELECT BITS AND
 5146                                  ;LOAD THE OUTPUT DATA BUFFER
 5147
 5148                                  ;CALLING SEQUENCE:
 5149
 5150                                  ;       JSR     R5,SETCL        ;CALL
 5151                                  ;       BR      NEWLPR          ;EXIT BRANCH AFTER ALL FOUR LNGTHS TESTED
 5152
 5153  017232  005767  002276          SETCL:  TST     QUICKX          ;EXIT AFTER ONLY ONE CHAR LNGTH ?
```

```
5154  017236  001034                        BNE     2$              ;BR IF YES
5155  017240  005267  002332                INC     CLSEL           ;GENERATE NEW CHAR LNGTH SELECT CODE
5156  017244  022767  000004  002324        CMP     #4,CLSEL        ;DONE FOUR OF THEM ??
5157  017252  001426                        BEQ     2$              ;BR IF YES
5158  017254  005767  002252                TST     QUICK           ;QUICK TEST FLAG SET ?
5159  017260  001407                        BEQ     1$              ;BR IF NOT
5160  017262  005267  002246                INC     QUICKX          ;SET QUICK TEST EXIT FLAG
5161  017266  005067  002304                CLR     CLSEL           ;DO ONLY 5 BIT CHARS
5162  017272  012767  177760  002274        MOV     #177760,CHRCNT  ;DO  ONLY 32. CHAR BUFFER
5163  017300  042767  000003  002262  1$:   BIC     #3,CURLPR       ;SET UP THE CURRENT LPR
5164  017306  056767  002264  002254        BIS     CLSEL,CURLPR
5165  017314  006367  002254                ASL     CHRCNT          ;GENERATE CHAR COUNT
5166  017320  004767  000006                JSR     PC,SUBUF1       ;GO SET UP THE OUTPUT BUFFER
5167  017324  062705  000002                ADD     #2,R5           ;MOVE PC AROUND EXIT BRANCH
5168  017330  000205                  2$:   RTS     R5              ;RETURN
5169
5170                                        ;THIS ROUTINE IS CALLED TO LOAD THE OUTPUT DATA BUFFER WITH THE
5171                                        ;REQUIRED BINARY COUNT PATTERN
5172
5173                                        ;CALLING SEQUENCE:
5174
5175                                        ;       JSR     PC,SUBUF1       ;CALL
5176
5177  017332                            SUBUF1:
5178  017332  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
5179  017334  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
5180  017336  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
5181  017340  005004                        CLR     R4              ;INIT CHAR GENERATOR
5182  017342  016703  002226                MOV     CHRCNT,R3       ;SET UP LOAD COUNT
5183  017346  012702  032754                MOV     #TBUF,R2        ;SET UP BUFFER POINTER
5184  017352  110422                  1$:   MOVB    R4,(R2)+        ;LOAD A CHAR
5185  017354  005204                        INC     R4              ;GENERATE NEXT CHAR
5186  017356  005203                        INC     R3              ;COUNT ONE LOADED
5187  017360  001374                        BNE     1$              ;BR TIL BUFFER FULL
5188  017362  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
5189  017364  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
5190  017366  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
5191  017370  000207                        RTS     PC              ;RETURN
5192
5193                                        ;THIS ROUTINE IS CALLED TO SET UP THE PARITY SELECT BITS
5194                                        ;IN THE CURRENT LPR TEST CONSTANT
5195
5196                                        ;CALLING SEQUENCE:
5197
5198                                        ;       JSR     R5,SETPAR       ;CALL
5199                                        ;       BR      NEWCL           ;EXIT BRANCH
5200
5201  017372  022767  177777  002200  SETPAR: CMP     #-1,PARBIT      ;DONE ALL PARITY COMBOS ?
5202  017400  001444                        BEQ     5$              ;BR IF YES
5203  017402  005767  002124                TST     QUICK           ;QUICK TEST FLAG SET ?
5204  017406  001403                        BEQ     1$              ;BR IF NOT
5205  017410  012767  000060  002162        MOV     #60,PARBIT      ;CHECK ODD PARITY ONLY
5206  017416  042767  000060  002144  1$:   BIC     #60,CURLPR      ;SET PARITY SELECT BITS
5207  017424  056767  002150  002136        BIS     PARBIT,CURLPR
5208  017432  005767  002142                TST     PARBIT          ;SELECT BITS 00 ?
5209  017436  001004                        BNE     2$              ;BR IF NOT
```

```
5210  017440  012767  000020  002132          MOV     #20,PARBIT      ;SET SELECT BITS TO 01
5211  017446  000417                          BR      4$              ;EXIT
5212  017450  022767  000020  002122  2$:     CMP     #20,PARBIT      ;SELECT BITS 10 ?
5213  017456  001004                          BNE     3$              ;BR IF NOT
5214  017460  012767  000060  002112          MOV     #60,PARBIT      ;MAKE SELECT BITS 11
5215  017466  000407                          BR      4$              ;EXIT
5216  017470  022767  000060  002102  3$:     CMP     #60,PARBIT      ;SELECT BITS 11 ?
5217  017476  001005                          BNE     5$              ;BR IF NOT
5218  017500  012767  177777  002072          MOV     #-1,PARBIT      ;SET EXIT FLAG
5219  017506  062705  000002          4$:     ADD     #2,R5           ;MOVE RETURN PC AROUND EXIT BRANCH
5220  017512  000205                  5$:     RTS     R5              ;RETURN
5221
5222                                  ;THIS ROUTINE IS CALLED TO SET UP FOR KEYBOARD INTERRUPTS
5223
5224  017514  012767  017540  160336  KYBD1:  MOV     #KYBD2,60       ;SET UP THE INPUT VECTOR
5225  017522  012767  000340  160332          MOV     #340,62
5226  017530  012767  000100  160022          MOV     #100,177560     ;ENABLE KYBD INTRS
5227  017536  000207                          RTS     PC              ;RETURN TO START TESTING
5228
5229                                  ;THIS ROUTINE SERVICES THE KEYBOARD INTERRUPT AND LOOKS FOR AN "S"
5230                                  ;BEING TYPED TO INDICATE ABORT AND PRINT STATISTICS
5231
5232  017540  117746  161402          KYBD2:  MOVB    @$TKB,-(SP)     ;GET CHAR TYPED
5233  017544  142716  000200                  BICB    #200,(SP)       ;CLEAR UNWANTED BITS
5234  017550  122716  000123                  CMPB    #123,(SP)       ;WAS AN "S" TYPED ?
5235  017554  001420                          BEQ     1$              ;BR IF YES
5236  017556  022767  000176  161354          CMP     #SWREG,SWR      ;USING SOFTWARE SWR?
5237  017564  001024                          BNE     2$              ;BRANCH IF YES
5238  017566  126727  161342  000001          CMPB    $AUTOB,#1       ;RUNNING IN AUTO MODE?
5239  017574  001420                          BEQ     2$              ;BRANCH IF YES
5240  017576  122716  000007                  CMPB    #7,(SP)         ;IS IT A <^G>
5241  017602  001015                          BNE     2$              ;BRANCH IF NO
5242  017604  005726                          TST     (SP)+           ;POP
5243  017606  104401  013745                  TYPE    ,$CNTLG         ;TYPE <^G>
5244  017612  000167  173324                  JMP     $GTSWR
5245  017616  005726                  1$:     TST     (SP)+           ;POP
5246  017620  000005                          RESET                   ;ZAP THE WORLD
5247  017622  012706  001100                  MOV     #STACK,SP       ;RESET THE SP
5248  017626  004767  001566                  JSR     PC,CHPS1        ;GO CLEAR PSW
5249  017632  000167  164554                  JMP     PRSTAT          ;GO DUMP THE STATISTICS
5250  017636  005726                  2$:     TST     (SP)+           ;POP
5251  017640  000002                          RTI                     ;RETURN AND FORGET IT
5252
5253                                  ;THIS ROUTINE SENDS A TEST BUFFER TO REMOTE DH11 LINE
5254
5255  017642  016701  001646          SENDP2: MOV     DHADR,R1                ;SET UP DH SCR ADDR
5256  017646  012711  004000                  MOV     #BIT11,(R1)             ;CLEAR THE DH11
5257  017652  016711  002266                  MOV     LINE,(R1)               ;SET LINE SELECT
5258  017656  162705  032754                  SUB     #TBUF,R5                ;SET UP BYTE COUNT
5259  017662  005405                          NEG     R5
5260  017664  010561  000010                  MOV     R5,BCR(R1)
5261  017670  012761  032754  000006          MOV     #TBUF,CAR(R1)           ;SET CURRENT ADDRESS
5262  017676  016761  001666  000004          MOV     CURLPR,LPR(R1)          ;SET LINE PARAMETERS
5263  017704  016761  001616  000012          MOV     LINMSK,BAR(R1)          ;ACTIVATE THE LINE
5264
5265  017712  005711                  1$:     TST     (R1)                    ;DONE TRANSMITTING ??
```

```
5266  017714  100376                  BPL     1$                      ;BR IF NOT
5267  017716  000207                  RTS     PC                      ;RETURN TO CONTROL ROUTINE ''SENDP1''
5268
5269                          ;THIS ROUTINE IS CALLED TO LOAD FILLERS INTO ECHO BUFFER
5270
5271  017720  116704  002506  LDFILL: MOVB    FILLB,R4                ;GET COUNT OF FILLERS
5272  017724  012703  022367          MOV     #ECBUF+1,R3             ;SET UP BUFFER POINTER
5273  017730  116767  161246  002430  MOVB    $TMP0,ECBUF             ;STORE LF CHAR
5274  017736  116722  161240          MOVB    $TMP0,(R2)+             ;IN ECHO BUFFER TOO
5275  017742  116723  002462  1$:     MOVB    FILLA,(R3)+             ;LOAD A FILLER CHAR
5276  017746  116722  002456          MOVB    FILLA,(R2)+
5277  017752  005304                  DEC     R4                      ;COUNT IT
5278  017754  001372                  BNE     1$                      ;BR TIL REQUIRED COUNT LOADED
5279  017756  116704  002450          MOVB    FILLB,R4                ;SET UP BYTE COUNT REG
5280  017762  005204                  INC     R4
5281  017764  005404                  NEG     R4
5282  017766  010461  000010          MOV     R4,BCR(R1)              ;LOAD BCR REG
5283  017772  000207                  RTS     PC                      ;RETURN TO RINT2
5284
5285                          ;THIS ROUTINE IS CALLED TO SET UP XMITTER SPEED
5286
5287  017774  104401          INXSP:  TYPE                            ;ASK USER TO TYPE SPEED
5288  017776  026610                  XSMSG1                          ;''TRANSMITTER SPEED ?''
5289  020000  012767  022150  002140  1$:     MOV     #XSPTAB,XSPTR   ;SET UP TABLE POINTER
5290  020006  042767  036000  001554          BIC     #36000,CURLPR   ;INIT SPEED SELECT BITS
5291  020014  104413                  RDDEC                           ;READ SPEED HE TYPED
5292  020016  005716                  TST     (SP)                    ;DEFAULT TO 9600. BAUD ?
5293  020020  001426                  BEQ     4$                      ;BR IF YES
5294  020022  027716  002120          2$:     CMP     @XSPTR,(SP)     ;TYPED ENTRY MATCH TABLE ENTRY ?
5295  020026  001010                  BNE     3$                      ;BR IF NOT
5296  020030  062767  000002  002110          ADD     #2,XSPTR        ;POINT TO SELECT BITS IN TABLE
5297  020036  057767  002104  001524          BIS     @XSPTR,CURLPR   ;SET SPEED SELECT BITS
5298  020044  005726                  TST     (SP)+                   ;FIX STACK
5299  020046  000417                  BR      5$                      ;CONTINUE
5300
5301  020050  062767  000004  002070  3$:     ADD     #4,XSPTR        ;POINT TO NEXT ENTRY
5302  020056  022767  022234  002062          CMP     #XSPTAB+52.,XSPTR;END OF TABLE ??
5303  020064  001356                  BNE     2$                      ;BR IF NOT
5304  020066  104401                  TYPE                            ;ERROR MESSAGE
5305  020070  026636                  XSMSG2                          ;''INVALID XMITR SPEED - TRY AGAIN''
5306  020072  005726                  TST     (SP)+                   ;FIX THE SP
5307  020074  000741                  BR      1$                      ;GO TRY AGAIN
5308
5309  020076  052767  032000  001464  4$:     BIS     #32000,CURLPR   ;SET UP DEFAULT TO 9600. BAUD
5310  020104  005726                  TST     (SP)+                   ;FIX STACK POINTER
5311
5312  020106  000207          5$:     RTS     PC                      ;RETURN TO CALLER
5313
5314                          ;THIS ROUTINE IS CALLED TO SET UP RECEIVER SPEED
5315
5316  020110  104401          INRSP:  TYPE                            ;ASK USER TO TYPE SPEED
5317  020112  026701                  RSMSG1                          ;''RECEIVER SPEED ?''
5318  020114  012767  022236  002112  1$:     MOV     #RSPTAB,RSPTR   ;SET UP TABLE POINTER
5319  020122  042767  001700  001440          BIC     #1700,CURLPR    ;INIT SPEED SELECT BITS
5320  020130  104413                  RDDEC                           ;READ SPEED HE TYPED
5321  020132  005716                  TST     (SP)                    ;DEFAULT TO 9600. BAUD ?
```

```
 5322  020134  001426                        BEQ     4$              ;BR IF YES
 5323  020136  027716  002072          2$:   CMP     @RSPTR,(SP)     ;TYPED ENTRY MATCH TABLE ENTRY ?
 5324  020142  001010                        BNE     3$              ;BR IF NOT
 5325  020144  062767  000002  002062        ADD     #2,RSPTR        ;POINT TO SELECT BITS IN TABLE
 5326  020152  057767  002056  001410        BIS     @RSPTR,CURLPR   ;SET SPEED SELECT BITS
 5327  020160  005726                        TST     (SP)+           ;FIX STACK
 5328  020162  000417                        BR      5$              ;CONTINUE
 5329
 5330  020164  062767  000004  002042  3$:   ADD     #4,RSPTR        ;POINT TO NEXT ENTRY
 5331  020172  022767  022322  002034        CMP     #RSPTAB+52.,RSPTR;END OF TABLE ??
 5332  020200  001356                        BNE     2$              ;BR IF NOT
 5333  020202  104401                        TYPE                    ;ERROR MESSAGE
 5334  020204  026724                        RSMSG2                  ;''INVALID RCVR SPEED - TRY AGAIN''
 5335  020206  005726                        TST     (SP)+           ;FIX THE SP
 5336  020210  000741                        BR      1$              ;GO TRY AGAIN
 5337
 5338  020212  052767  001500  001350  4$:   BIS     #1500,CURLPR    ;SET UP DEFAULT TO 9600. BAUD
 5339  020220  005726                        TST     (SP)+           ;FIX STACK POINTER
 5340
 5341  020222  000207                  5$:   RTS     PC              ;RETURN TO CALLER
 5342
 5343
 5344                                  ;THIS ROUTINE IS CALLED TO SET UP LINE PARAMETERS FM KYBD
 5345
 5346  020224  105067  005772          LPRIN:  CLRB    EC2             ;CLEAR ECHO BUFFER
 5347  020230  104401                          TYPE
 5348  020232  026536                          LPMSG                   ;'DO YOU WANT TO CHANGE 'LPR'?''
 5349  020234  104410                    1$:   RDCHR
 5350  020236  012600                          MOV     (SP)+,R0        ;GET WHAT HE TYPED
 5351  020240  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR> ??
 5352  020244  001405                          BEQ     2$              ;BR IF YES
 5353  020246  110067  005750                  MOVB    R0,EC2          ;ECHO WHAT HE TYPED
 5354  020252  104401                          TYPE
 5355  020254  026222                          EC2
 5356  020256  000766                          BR      1$              ;GO WAIT FOR TERMINATOR
 5357
 5358  020260  105767  005736          2$:   TSTB    EC2             ;<CR> ONLY ??
 5359  020264  001411                        BEQ     3$              ;BR IF YES
 5360  020266  122767  000116  005726        CMPB    #116,EC2        ;WAS IT A 'NO'' ??
 5361  020274  001405                        BEQ     3$              ;BR IF IT WAS
 5362  020276  122767  000131  005716        CMPB    #131,EC2        ;WAS IT A 'YES'' ??
 5363  020304  001347                        BNE     LPRIN           ;GO ASK ALL OVER AGAIN
 5364  020306  000407                        BR      4$              ;BR IF IT WAS 'YES''
 5365  020310  005767  001254          3$:   TST     CURLPR          ;HAS LPR BEEN SET UP AT ALL ?
 5366  020314  001016                        BNE     5$              ;BR IF YES USE PREVIOUS LPR
 5367  020316  012767  033503  001244        MOV     #33503,CURLPR   ;SET DEFAULT 9600 BAUD,8 BITS NO PARITY
 5368  020324  000412                        BR      5$              ;CONTINUE
 5369  020326  004767  177442          4$:   JSR     PC,INXSP        ;GO INPUT AND SET UP XMIT SPEED
 5370  020332  004767  177552                JSR     PC,INRSP        ;GO INPUT AND SET UP RCVR SPEED
 5371  020336  004767  000022                JSR     PC,INCL         ;GO INPUT AND SET UP CHAR LENGTH
 5372  020342  004767  000162                JSR     PC,INSB         ;GO INPUT AND SET UP NO. OF STOP BITS
 5373  020346  004767  000274                JSR     PC,INPB         ;GO INPUT AND SET UP PARITY SELECTION
 5374  020352  004767  000410          5$:   JSR     PC,INFCHR       ;GO INPUT AND SET UP FILLER CHAR
 5375  020356  004767  000474                JSR     PC,INFCNT       ;GO INPUT AND SET UP FILLER COUNT
 5376  020362  000207                        RTS     PC              ;RETURN TO CALLER
 5377
```

```
 5378                                   ;THIS ROUTINE IS CALLED TO SET UP CHAR LENCTH BITS
 5379
 5380  020364  105067  005632          INCL:  CLRB    EC2             ;CLEAR THE ECHO BUFFER
 5381  020370  104401                         TYPE                    ;ASK FOR INPUT
 5382  020372  026767                         CLMSG1                  ;''CHAR LENGTH - 6,7, OR 8 ?''
 5383  020374  042767  000003  001166  1$:    BIC     #3,CURLPR       ;INIT CHAR LENGTH SELECT BITR
 5384  020402  104410                         RDCHR                   ;GET THE CHAR HE TYPED
 5385  020404  012600                         MOV     (SP)+,R0        ;GET WHAT HE TYPED
 5386  020406  122700  000015                 CMPB    #15,R0          ;WAS IT A <CR> ??
 5387  020412  001405                         BEQ     11$             ;BR IF IT WAS
 5388  020414  110067  005602                 MOVB    R0,EC2          ;ECHO WHAT HE TYPED
 5389  020420  104401                         TYPE
 5390  020422  026222                         EC2
 5391  020424  000763                         BR      1$              ;GO WAIT FOR TERMINATOR
 5392  020426  105767  005570          11$:   TSTB    EC2             ;<CR> ONLY ??
 5393  020432  001432                         BEQ     4$              ;BR IF YES
 5394  020434  142767  000060  005560        BICB    #60,EC2         ;STRIP ASCII
 5395  020442  122767  000006  005552        CMPB    #6,EC2          ;6 BITS ?
 5396  020450  001004                         BNE     2$              ;BR IF NOT
 5397  020452  052767  000001  001110        BIS     #1,CURLPR       ;SET UP FOR 6 BIT CHARS
 5398  020460  000422                         BR      5$              ;CONTINUE
 5399  020462  122767  000007  005532  2$:    CMPB    #7,EC2          ;7 BITS ?
 5400  020470  001004                         BNE     3$              ;BR IF NOT
 5401  020472  052767  000002  001070        BIS     #2,CURLPR       ;SET UP FOR 7 BIT CHARS
 5402  020500  000412                         BR      5$              ;CONTINUE
 5403  020502  122767  000010  005512  3$:    CMPB    #8.,EC2         ;8 BITS ?
 5404  020510  001403                         BEQ     4$              ;BR IF YES
 5405  020512  104401                         TYPE                    ;ERROR MESSAGE
 5406  020514  027025                         CLMSG2                  ;''INVALID CHAR LENGTH   TRY AGAIN''
 5407  020516  000722                         BR      INCL            ;GO TRY AGAIN
 5408  020520  052767  000003  001042  4$:    BIS     #3,CURLPR       ;SET UP FOR 8 BIT CHARS
 5409  020526  000207                 5$:    RTS     PC              ;RETURN TO CALLER
 5410
 5411                                   ;THIS ROUTINE IS CALLED TO SET UP NO. OF STOP BITS
 5412
 5413  020530  105067  005466          INSB:  CLRB    EC2             ;CLEAR ECHO BUFFER
 5414  020534  104401                         TYPE                    ;ASK FOR INPUT
 5415  020536  027071                         SBMSG1                  ;'NO. OF STOP BITS - 1 OR 2 ?''
 5416  020540  104410                 1$:    RDCHR                   ;GET CHAR TYPED
 5417  020542  012600                         MOV     (SP)+,R0        ;GET WHAT HE TYPED
 5418  020544  122700  000015                 CMPB    #15,R0          ;WAS IT A <CR>
 5419  020550  001405                         BEQ     11$             ;BR IF YES
 5420  020552  110067  005444                 MOVB    R0,EC2          ;ECHO WHAT HE TYPED
 5421  020556  104401                         TYPE
 5422  020560  026222                         EC2
 5423  020562  000766                         BR      1$              ;GO WAIT FOR TERMINATOR
 5424  020564  105767  005432          11$:   TSTB    EC2             ;<CR> ONLY ??
 5425  020570  001422                         BEQ     3$              ;BR IF YES
 5426  020572  142767  000060  005422        BICB    #60,EC2         ;CLEAR ASCII JUNK
 5427  020600  122767  000002  005414        CMPB    #2,EC2          ;2 STOP BITS ?
 5428  020606  001004                         BNE     2$              ;BR IF NOT
 5429  020610  052767  000004  000752        BIS     #4,CURLPR       ;SET UP FOR TWO STOP BITS
 5430  020616  000412                         BR      4$              ;CONTINUE
 5431  020620  122767  000001  005374  2$:    CMPB    #1,EC2          ;ONE STOP BIT ?
 5432  020626  001403                         BEQ     3$              ;BR IF YES
 5433  020630  104401                         TYPE                    ;ERROR MESSAGE
```

```
5434  020632  027130                        SBMSG2                  ;''INVALID NO. STOP BITS - TRY AGAIN''
5435  020634  000735                        BR      INSB            ;GO TRY AGAIN
5436  020636  042767  000004  000724  3$:   BIC     #4,CURLPR       ;SET UP FOR ONE STOP BIT
5437  020644  000207                  4$:   RTS     PC              ;RETURN TO CALLER
5438
5439                                        ;THIS ROUTINE IS CALLED TO SET UP PARITY SELECT BITS
5440
5441  020646  105067  005350        INPB:   CLRB    EC2             ;CLEAR ECHO BUFFER
5442  020652  104401                        TYPE                    ;ASK FOR INPUT
5443  020654  027176                        PBMSG1                  ;''PARITY - E,O, OR <CR> ?''
5444  020656  042767  000060  000704  *$:   BIC     #60,CURLPR      ;INIT FOR NO PARITY CHECKING
5445  020664  104410                        RDCHR                   ;GET CHAR TYPED
5446  020666  012600                        MOV     (SP)+,R0        ;GET WHAT HE TYPED
5447  020670  122700  000015                CMPB    #15,R0          ;WAS IT A <CR> ??
5448  020674  001405                        BEQ     11$             ;BR IF IT WAS
5449  020676  110067  005320                MOVB    R0,EC2          ;ECHO THE CHAR TYPED
5450  020702  104401                        TYPE
5451  020704  026222                        EC2
5452  020706  000763                        BR      1$              ;GO WAIT FOR TERMINATOR
5453  020710  105767  005306        11$:    TSTB    EC2             ;<CR> ONLY ??
5454  020714  001423                        BEQ     4$              ;BR IF YES
5455  020716  122767  000105  005276        CMPB    #105,EC2        ;EVEN PARITY ??
5456  020724  001004                        BNE     2$              ;BR IF NOT
5457  020726  052767  000060  000634        BIS     #60,CURLPR      ;SET UP FOR EVEN PARITY
5458  020734  000413                        BR      4$              ;CONTINUE
5459  020736  122767  000117  005256  2$:   CMPB    #117,EC2        ;ODD PARITY
5460  020744  001004                        BNE     3$              ;BR IF NOT
5461  020746  052767  000020  000614        BIS     #20,CURLPR      ;SET UP FOR ODD PARITY
5462  020754  000403                        BR      4$              ;CONTINUE
5463  020756  104401                  3$:   TYPE                    ;ERROR MESSAGE
5464  020760  027244                        PBMSG2                  ;''INVALID PARITY - TRY AGAIN''
5465  020762  000731                        BR      INPB            ;GO TRY AGAIN
5466  020764  000207                  4$:   RTS     PC              ;RETURN TO CALLER
5467
5468                                        ;THIS ROUTINE IS CALLED TO SET UP ''FILL'' CHAR
5469
5470  020766  105067  005230        INFCHR: CLRB    EC2             ;CLEAR ECHO BUFFER
5471  020772  005067  001432                CLR     FILLA           ;INIT TEMP STORAGE FOR CHAR
5472  020776  104401                        TYPE                    ;GO ASK FOR FILLER CHAR
5473  021000  027303                        FILC1                   ;''FILL CHAR ?''
5474  021002  005067  001420          1$:   CLR     DHFILL          ;ININ FILL LOCATION
5475  021006  104410                        RDCHR                   ;GET CHAR TYPED
5476  021010  012600                        MOV     (SP)+,R0        ;GET WHAT HE TYPED
5477  021012  122700  000015                CMPB    #15,R0          ;WAS IT A <CR> ??
5478  021016  001405                        BEQ     2$              ;BR IF YES
5479  021020  110067  005176                MOVB    R0,EC2          ;ECHO WHAT HE TYPED
5480  021024  104401                        TYPE
5481  021026  026222                        EC2
5482  021030  000764                        BR      1$              ;GO WAIT FOR TERMINATOR
5483
5484  021032  105767  005164          2$:   TSTB    EC2             ;<CR> ONLY ??
5485  021036  001403                        BEQ     3$              ;BR IF YES
5486  021040  116767  005156  001361        MOVB    EC2,DHFILL+1    ;SET UP FILL CHAR
5487  021046  116767  001355  001354  3$:   MOVB    DHFILL+1,FILLA  ;SAVE FILL CHAR
5488  021054  000207                        RTS     PC              ;RETURN TO CALLER
5489
```

```
5490                                    ;THIS ROUTINE IS CALLED TO SET UP "FILL" COUNT
5491
5492  021056  005067  001350            INFCNT: CLR     FILLB           ;INIT TEMP, STORAGE FOR COUNT
5493  021062  104401                            TYPE                    ;ASK FOR COUNT
5494  021064  027331                            FILC2                   ;"FILL COUNT ?"
5495  021066  104412                            RDOCT                   ;GET OCTAL NO. TYPED
5496  021070  005716                            TST     (SP)            ;DEFAULT TO ONE ?
5497  021072  001403                            BEQ     1$              ;BR IF YES
5498  021074  111667  001326                    MOVB    (SP),DHFILL     ;SET UP COUNT TYPED
5499  021100  000403                            BR      2$              ;CONTINUE
5500  021102  112767  000001  001316    1$:     MOVB    #1,DHFILL       ;SET UP FOR  1 FILLER
5501  021110  005726                    2$:     TST     (SP)+           ;FIX THE SP
5502  021112  142767  000360  001306            BICB    #360,DHFILL     ;LIMIT COUNT TO 15. MAX
5503  021120  116767  001302  001304            MOVB    DHFILL,FILLB    ;SAVE IT FOR LATER
5504  021126  000207                            RTS     PC              ;RETURN TO CALLER
5505                                    ;THIS ROUTINE CALLED TO SET UP ALTERNATING 1/0 PATTERN
5506
5507  021130  004767  000246            SUPATA: JSR     PC,CLALL        ;GO CLEAR XMIT AND RCV BUFFERS
5508  021134  016700  000434                    MOV     CHRCNT,R0               ;GET CHAR COUNT
5509  021140  012705  032754                    MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
5510  021144  112725  000252            1$:     MOVB    #252,(R5)+      ;LOAD A BYTE
5511  021150  005200                            INC     R0              ;COUNT IT
5512  021152  001374                            BNE     1$              ;BR TILL BUFFER FULL
5513  021154  000207                            RTS     PC              ;RETURN TO "DPATA" ROUTINE
5514
5515                                    ;THIS ROUTINE IS CALLED TO SET UP UP COUNT PATTERN
5516
5517  021156  004767  000220            SUPATU: JSR     PC,CLALL        ;GO CLEAR BUFFERS
5518  021162  016700  000406                    MOV     CHRCNT,R0       ;GET COUNT OF CHARS TO LOAD
5519  021166  012705  032754                    MOV     #TBUF,R5        ;POINT TO XMITTR BUFFER
5520  021172  005004                            CLR     R4              ;INIT CHAR GENERATOR
5521  021174  110425                    1$:     MOVB    R4,(R5)+        ;LOAD ONE BYTE
5522  021176  105204                            INCB    R4              ;GENERATE NEXT BYTE
5523  021200  005200                            INC     R0              ;COUNT IT
5524  021202  001374                            BNE     1$              ;BR TIL BUFFER FULL
5525  021204  000207                            RTS     PC              ;RETURN TO "DPATU" ROUTINE
5526
5527                                    ;THIS ROUTINE IS CALLED TO SET UP DOWN COUNT PATTERN
5528
5529  021206  004767  000170            SUPATD: JSR     PC,CLALL        ;CLEAR THE BUFFERS
5530  021212  016700  000356                    MOV     CHRCNT,R0       ;SET UP COUNT TO LOAD
5531  021216  012705  032754                    MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
5532  021222  012704  000377                    MOV     #377,R4         ;INIT CHAR GENERATOR
5533  021226  110425                    1$:     MOVB    R4,(R5)+        ;LOAD ONE BYTE
5534  021230  105304                            DECB    R4              ;GENERATE NEW CHAR
5535  021232  005200                            INC     R0              ;COUNT IT
5536  021234  001374                            BNE     1$              ;BR TIL BUFFER FULL
5537  021236  000207                            RTS     PC              ;RETURN TO "DPATA" ROUTINE
5538
5539                                    ;THIS ROUTINE CALLED TO LOAD RANDOM DATA PATTERN
5540
5541  021240  004767  000136            SUPATR: JSR     PC,CLALL        ;GO CLEAR BUFFERS
5542  021244  016700  000324                    MOV     CHRCNT,R0       ;SET UP COUNT TO LOAD
5543  021250  012705  032754                    MOV     #TBUF,R5        ;POINT TO XMITTR BUFFER
5544  021254  012767  125252  001074            MOV     #125252,RANA    ;INIT RANDOM NUMBER GENERATOR
5545
```

```
5546  021262  066767  001070  001070  1$:     ADD     RANA,RANB       ;GENERATE RANDOM NO.
5547  021270  005567  001062          ADC     RANA
5548  021274  066767  001060  001054  ADD     RANB,RANA
5549  021302  005567  001052          ADC     RANB
5550
5551  021306  116725  001044          MOVB    RANA,(R5)+      ;LOAD A BYTE
5552  021312  005200                  INC     R0              ;COUNT IT
5553  021314  001362                  BNE     1$              ;BR TIL BUFFER FULL
5554  021316  000207                  RTS     PC              ;RETURN TO 'DPATR' ROUTINE
5555
5556                          ;THIS.ROUTINE LOADS A SINGLE CHAR THROUGHOUT BUFFER
5557
5558  021320  004767  000056  SUPATS: JSR     PC,CLALL        ;GO CLEAR BUFFERS
5559  021324  016700  000244          MOV     CHRCNT,R0       ;INIT CHAR COUNTER
5560  021330  012705  032754          MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
5561  021334  116725  001010  1$:     MOVB    SINGLE,(R5)+    ;LOAD ONE CHAR
5562  021340  005200                  INC     R0              ;COUNT IT
5563  021342  001374                  BNE     1$              ;BR TIL BUFFER FULL
5564  021344  000207                  RTS     PC              ;RETURN TO 'DPATS' ROUTINE
5565
5566                          ;THIS ROUTINE CALLED TO INIT CHAR LENGTH MASK FOR PATTERNS TESTS
5567
5568  021346  016700  000216  SUCLMK: MOV     CURLPR,R0       ;GET CURRENT 'LPR'
5569  021352  012767  000340  001002  MOV     #340,CLMSK      ;INIT FOR 5 BIT CHARS
5570  021360  042700  177774          BIC     #177774,R0      ;MASK OFF ALL BUT CL BITS
5571  021364  005700          1$:     TST     R0              ;DONE SETUP ?
5572  021366  001404                  BEQ     2$              ;BR IF YES
5573  021370  106367  000766          ASLB    CLMSK           ;SHIFT MASK LEFT
5574  021374  005300                  DEC     R0              ;COUNT IT
5575  021376  000772                  BR      1$              ;GO SEE IF ITS RIGHT ON
5576  021400  000207          2$:     RTS     PC              ;RETURN TO CALLER
5577
5578                          ;ROUTINE TO CLEAR XMIT AND RECEIVER BUFFERS
5579  021402  012700  032754  CLALL:  MOV     #TBUF,R0        ;SET UP POINTER
5580  021406  005020          1$:     CLR     (R0)+           ;CLEAR A WORD
5581  021410  022700  034104          CMP     #ENBUFS,R0      ;DONE ALL LOCATIONS ?
5582  021414  001374                  BNE     1$              ;BR IF NOT
5583  021416  000207                  RTS     PC
```

POWER DOWN AND UP ROUTINES SEQ 0121

```
                                        ;THIS ROUTINE IS CALLED TO SET PSW PRIORITY TO 000 IN ORDER
                                        ;TO BE LSI11 COMPATIBLE
021420  012746  000000          CHPS1:  MOV     #0,-(SP)        ;NEW PSW
021424  012746  021432                  MOV     #1$,-(SP)       ;NEW PC
021430  000002                          RTI                     ;CHANGE PSW
021432  000207                  1$:     RTS     PC              ;RETURN TO CALLING TEST

                                        ;THIS ROUTINE DOES THE SAME THING EXCEPT IT SET THE PSW
                                        ;PRIORITY TO 340 (LEVEL 7 ) TO LOCK OUT INTRS

021434  012746  000340          CHPS2:  MOV     #340,-(SP)      ;NEW PSW
021440  012746  021446                  MOV     #1$,-(SP)       ;NEW PC
021444  000002                          RTI                     ;CHANGE THE PSW
021446  000207                  1$:     RTS     PC              ;RETURN TO CALLING TEST

                                        ;THIS ROUTINE IS ALSO FOR LSI11 COMPATIBILITY AND IT IS CALLED
                                        ;TO SAVE THE PSW IN '$TMP0'

021450  005046                  SAPS:   CLR     -(SP)           ;TEMP STORAGE TO SAVE PSW
021452  016746  156356                  MOV     34,-(SP)        ;SAVE TRAP VECTOR POINTER
021456  012767  021466  156350          MOV     #1$,34          ;GO TO 1$ ON TRAP
021464  104400                          TRAP                    ;GO TO IT
021466  016666  000002  000006  1$:     MOV     2(SP),6(SP)     ;GET PSW SAVED
021474  012716  021502                  MOV     #2$,(SP)        ;GO TO 2$ ON RTI
021500  000002                          RTI
021502  012667  156326          2$:     MOV     (SP)+,34        ;RESTORE VECTOR
021506  012667  157470                  MOV     (SP)+,$TMP0     ;FINALLY SAVE PSW IN $TMP0
021512  000207                          RTS     PC
```

```
CZDHN-D MA(V11 30A(1052)  27-DEC-78  15:31  PAGE 123
CZDHND.P11    27-DEC-78 15:28          DH11 PROGRAM CONSTANTS AND VARIABLES                                SEQ 0122

                                       .SBTTL DH11 PROGRAM CONSTANTS AND VARIABLES
                                       ;;*****************************************************************
                                       ;ADDITIONAL PROGRAM CONSTANTS AND VARIABLES
                                       ;;*****************************************************************

            000002                             NRC=2            ;INDEX CONST. TO ACCESS NEXT RCVD CHAR REG
            000004                             LPR=4            ;INDEX CONST. TO ACCESS LINE PARAMETER REG.
            000006                             CAR=6            ;INDEX CONST. TO ACCESS CURRENT ADDRESS REG.
            000010                             BCR=10           ;INDEX CONST. TO ACCESS BYTE COUNT REG.
            000012                             BAR=12           ;INDEX CONST. TO ACCESS BUFFER ACTIVE REC.
            000014                             BKR=14           ;INDEX CONST. TO ACCESS BREAK CONTROL REG.
            000016                             SSR=16           ;INDEX CONST. TO ACCESS SILO STATUS REG.

021514      000000                 DHADR:  0                    ;HOLDS THE "SCR" ADDRESS OF THE DH11 UNDER TEST
021516      000000                 DHVCT:  0                    ;HOLDS THE 1ST VECTOR ADDRESS OF THE DH11 UNDER TEST
021520      000000                 SELMSK: 0                    ;BIT TST MARKER FOR SELECTING DH11'S
021522      000001                 DHSEL:  1                    ;SPECIFIES DH11'S SELECTED FOR TEST
021524      177777                 LINSEL: 177777               ;SPECIFIES LINES TO TEST
021526      000000                 LINMSK: 0                    ;MARKER USED TO TEST FOR LINES TO TEST
021530      000000                 DRPLIN: 0                    ;DROPPED LINE FLAGS

021532      000000                 QUICK:  0                    ;QUICK TEST FLAG - ALLOWS SINGLE PATTERN TEST
                                                                ;ON ALL TESTS NOT USING 9600. BAUD
021534      000000                 QUICKX: 0                    ;ALLOWS SUB-TEST EXIT DURING QUICK TEST



                                       ;THIS TABLE CONTAINS THIRTEEN CONSTANTS USED TO ESTABLISH
                                       ;THE INITIAL LINE PARAMETERS FOR THE THIRTEEN PROGRAMMABLE BAUD
                                       ;RATES - EACH PARAMETER INITIALLY SPECIFIES NO PARITY CHECKING
                                       ;AND A CHARACTER LENGTH OF FIVE BITS

021536      033500                 LPRTAB: 33500                ;9600 BAUD
021540      004200                         4200                 ;75 BAUD
021542      006300                         6300                 ;110 BAUD
021544      010400                         10400                ;134.5 BAUD
021546      012500                         12500                ;150 BAUD
021550      014600                         14600                ;200 BAUD
021552      016700                         16700                ;300 BAUD
021554      021000                         21000                ;600 BAUD
021556      023100                         23100                ;1200 BAUD
021560      025200                         25200                ;1800 BAUD
021562      027300                         27300                ;2400 BAUD
021564      031400                         31400                ;4800 BAUD
021566      002100                         2100                 ;50 BAUD

021570      000000                 CURLPR: 0                    ;CONTAINS CURRENT "LPR" CONSTANT

021572      000000                 LPRPTR: 0                    ;CONTAINS POINTER TO LPR TABLE
021574      000000                 CHRCNT: 0                    ;LOADED WITH CURRENT CHAR COUNT

021576      000000                 CLSEL:  0                    ;CHAR LENGTH SELECT PARAMETER
021600      000000                 PARBIT: 0                    ;PARITY SELECT PARAMETER

021602      000000                 PDONE:  0                    ;SOFTWARE DONE FLAG
```

```
5671  021604  000000                  RBFEND: 0                       ;HOLDS END OF BUFFER ADDRESS
5672
5673                                  ;DH11 ADDRESS TABLE - THIS TABLE CONTAINS THE ''SCR'' ADDRESS FOR UP TO
5674                                  ;SIXTEEN DH11'S
5675
5676  021606  160020                  DHADTB: 160020          ;ADDRESS OF FIRST DH11
5677  021610  160040                          160040          ;ADDRESS OF SECOND DH11
5678  021612  160060                          160060
5679  021614  160100                          160100
5680  021616  160120                          160120
5681  021620  160140                          160140
5682  021622  160160                          160160
5683  021624  160200                          160200
5684  021626  160220                          160220
5685  021630  160240                          160240
5686  021632  160260                          160260
5687  021634  160300                          160300
5688  021636  160320                          160320
5689  021640  160340                          160340
5690  021642  160360                          160360
5691  021644  160400                          160400          ;ADDRESS OF THE LAST DH11
5692
5693                                  ;DH11 VECTOR TABLE - THIS TABLE CONTAINS THE VECTOR ADDRESSES FOR UP
5694                                  ;TO SIXTEEN DH11'S
5695
5696  021646  000330                  DHVCTB: 330             ;ADDRESS OF VECTOR FOR FIRST DH11
5697  021650  000350                          350             ;ADDRESS OF VECTOR FOR SECOND DH11
5698  021652  000370                          370
5699  021654  000410                          410
5700  021656  000430                          430
5701  021660  000450                          450
5702  021662  000470                          470
5703  021664  000510                          510
5704  021666  000530                          530
5705  021670  000550                          550
5706  021672  000570                          570
5707  021674  000610                          610
5708  021676  000630                          630
5709  021700  000650                          650
5710  021702  000670                          670
5711  021704  000710                          710             ;ADDRESS OF VECTOR FOR LAST DH11
5712
5713  021706  000000                  VFLG:   0               ;VECTOR DISPLACEMENT FLAG
5714
5715
5716
5717                                  ;BR PRIORITY LEVEL TABLE - THIS TABLE CONTAINS THE PRIORITY LEVELS
5718                                  ;FOR UP TO SIXTEEN DH11'S - THE RCVR LEVEL IS STORED IN THE LOW BYTE
5719                                  ;AND THE XMTTR LEVEL IN THE HIGH BYTE
5720  021710  120240                  BRLVL:  120240          ;BRLEVELS FOR FIRST DH11
5721  021712  120240                          120240          ;BR LEVELS FOR SECOND DH11
5722  021714  120240                          120240
5723  021716  120240                          120240
5724  021720  120240                          120240
5725  021722  120240                          120240
5726  021724  120240                          120240
```

```
 5727  021726  120240                          120240
 5728  021730  120240                          120240
 5729  021732  120240                          120240
 5730  021734  120240                          120240
 5731  021736  120240                          120240
 5732  021740  120240                          120240
 5733  021742  120240                          120240
 5734  021744  120240                          120240
 5735  021746  120240                          120240          ;BR LEVELS FOR LAST DH11
 5736
 5737                                  ;THIS DH ADDRESS TABLE IS FILLED BY THE AUTOSIZER.
 5738
 5739  021750                          DHADRS:
 5740  021750  000000                  .WORD   0
 5741  021752  000000                  .WORD   0
 5742  021754  000000                  .WORD   0
 5743  021756  000000                  .WORD   0
 5744  021760  000000                  .WORD   0
 5745  021762  000000                  .WORD   0
 5746  021764  000000                  .WORD   0
 5747  021766  000000                  .WORD   0
 5748  021770  000000                  .WORD   0
 5749  021772  000000                  .WORD   0
 5750  021774  000000                  .WORD   0
 5751  021776  000000                  .WORD   0
 5752  022000  000000                  .WORD   0
 5753  022002  000000                  .WORD   0
 5754  022004  000000                  .WORD   0
 5755
 5756                                  ;THIS DH VECTOR TABLE IS FILLED BY THE AUTOSIZER.
 5757
 5758  022006                          DHVEC:
 5759  022006  000000                  .WORD   0
 5760  022010  000000                  .WORD   0
 5761  022012  000000                  .WORD   0
 5762  022014  000000                  .WORD   0
 5763  022016  000000                  .WORD   0
 5764  022020  000000                  .WORD   0
 5765  022022  000000                  .WORD   0
 5766  022024  000000                  .WORD   0
 5767  022026  000000                  .WORD   0
 5768  022030  000000                  .WORD   0
 5769  022032  000000                  .WORD   0
 5770  022034  000000                  .WORD   0
 5771  022036  000000                  .WORD   0
 5772  022040  000000                  .WORD   0
 5773
 5774                                  ;THIS DM ADDRESS TABLE IF FILLED BY THE AUTOSIZER.
 5775
 5776  022042                          DMADRS:
 5777  022042  000000                  .WORD   0
 5778  022044  000000                  .WORD   0
 5779  022046  000000                  .WORD   0
 5780  022050  000000                  .WORD   0
 5781  022052  000000                  .WORD   0
 5782  022054  000000                  .WORD   0
```

```
 5783  022056  000000                .WORD   0
 5784  022060  000000                .WORD   0
 5785  022062  000000                .WORD   0
 5786  022064  000000                .WORD   0
 5787  022066  000000                .WORD   0
 5788  022070  000000                .WORD   0
 5789  022072  000000                .WORD   0
 5790  022074  000000                .WORD   0
 5791  022076  000000                .WORD   0
 5792
 5793                                ;THIS DM VECTOR TABLE IS FILLED BY THE AUTOSIZER.
 5794
 5795  022100                        DMVEC:
 5796  022100  000000                .WORD   0
 5797  022102  000000                .WORD   0
 5798  022104  000000                .WORD   0
 5799  022106  000000                .WORD   0
 5800  022110  000000                .WORD   0
 5801  022112  000000                .WORD   0
 5802  022114  000000                .WORD   0
 5803  022116  000000                .WORD   0
 5804  022120  000000                .WORD   0
 5805  022122  000000                .WORD   0
 5806  022124  000000                .WORD   0
 5807  022126  000000                .WORD   0
 5808  022130  000000                .WORD   0
 5809  022132  000000                .WORD   0
 5810
 5811  022134  000000                ADRVEC: 0               ;ADDRESSES BETWEEN VECTORS - FILLED BY THE AUTOSIZER.
 5812  022136  000000                $DHSEL: 0               ;DEVICE SELECT PARAMETER - FILLED BY THE AUTOSIZER.
 5813  022140     000                DHRLVL: .BYTE   0               ;BR LEVEL FOR RCVR
 5814  022141     000                DHTLVL: .BYTE   0               ;BR LEVEL FOR XMITTER
 5815  022142  000000                DHNUM:  0               ;CONTAINS NUMBER OF THE DH11 UNDER TEST
 5816  022144  000000                LINE:   0               ;CONTINES NUMBER OF THE LINE UNDER TEST
 5817                                ;TABLES USED TO SELECT XMITTR AND RCVR SPEEDS
 5818
 5819                                ;THE TABLES CONSIST OF 13. TWO WORD ENTRIES - ONE FOR EACH
 5820                                ;ALLOWABLE BAUD RATE. THE FIRST WORD IS THE ACTUAL BAUD RATE
 5821                                ;IN DECIMAL AND THE SECOND WORD IS THE ENCODED BINARY WORD
 5822                                ;THAT SETS THAT BAUD RATE IN THE 'LPR'
 5823
 5824  022146  000000                XSPTR:  0                       ;CONTAINS POINTER TO FOLLOWING TABLE
 5825
 5826  022150  000062                XSPTAB: 50.                     ;50. BAUD
 5827  022152  002000                        2000
 5828  022154  000113                        75.                     ;75. BAUD
 5829  022156  004000                        4000
 5830  022160  000156                        110.                    ;110. BAUD
 5831  022162  006000                        6000
 5832  022164  002501                        1345.                   ;134.5 BAUD
 5833  022166  010000                        10000
 5834  022170  000226                        150.                    ;150. BAUD
 5835  022172  012000                        12000
 5836  022174  000310                        200.                    ;200. BAUD
 5837  022176  014000                        14000
 5838  022200  000454                        300.                    ;300 BAUD
```

```
 5839  022202  016000                        16000
 5840  022204  001130                        600.                ;600. BAUD
 5841  022206  020000                        20000
 5842  022210  002260                        1200.               ;1200. BAUD
 5843  022212  022000                        22000
 5844  022214  003410                        1800.               ;1800. BAUD
 5845  022216  024000                        24000
 5846  022220  004540                        2400.               ;2400. BAUD
 5847  022222  026000                        26000
 5848  022224  011300                        4800.               ;4800. BAUD
 5849  022226  030000                        30000
 5850  022230  022600                        9600.               ;9600. BAUD
 5851  022232  032000                        32000
 5852  022234  000000              RSPTR:    0                   ;CONTAINS POINTER TO FOLLOWING TABLE
 5853
 5854  022236  000062              RSPTAB:   50.                 ;50. BAUD
 5855  022240  000100                        100
 5856  022242  000113                        75.                 ;75. BAUD
 5857  022244  000200                        200
 5858  022246  000156                        110.                ;110. BAUD
 5859  022250  000300                        300
 5860  022252  002501                        1345.               ;134.5 BAUD
 5861  022254  000400                        400
 5862  022256  000226                        150.                ;150. BAUD
 5863  022260  000500                        500
 5864  022262  000310                        200.                ;200. BAUD
 5865  022264  000600                        600
 5866  022266  000454                        300.                ;300 BAUD
 5867  022270  000700                        700
 5868  022272  001130                        600.                ;600. BAUD
 5869  022274  001000                        1000
 5870  022276  002260                        1200.               ;1200. BAUD
 5871  022300  001100                        1100
 5872  022302  003410                        1800.               ;1800. BAUD
 5873  022304  001200                        1200
 5874  022306  004540                        2400.               ;2400. BAUD
 5875  022310  001300                        1300
 5876  022312  011300                        4800.               ;4800. BAUD
 5877  022314  001400                        1400
 5878  022316  022600                        9600.               ;9600. BAUD
 5879  022320  001500                        1500
 5880
 5881                          ;ADDRESS POINTERS TO SET UP TABLES WHEN INPUTTING PARAMETERS
 5882
 5883  022322  000000              ADPTR:    0                   ;POINTS TO ADDRESS TABLE
 5884  022324  000000              VCPTR:    0                   ;POINTS TO VECTOR TABLE
 5885  022326  000000              BRPTR:    0                   ;POINTS TO BR LEVEL TABLE
 5886
 5887  022330  000000              TITFLG:   0                   ;FLAG TO ALLOW PRINTING TITLE ONLY ONCE
 5888  022332  000000              TIMEA:    0                   ;GENERAL PURPOSE TIMERS
 5889  022334  000000              TIMEB:    0
 5890
 5891  022336  000000              CEXIT:    0                   ;CONTROL-C EXIT FLAG FM ECHO TESTS
 5892  022340  000000              DPFLG:    0                   ;PATTERNS TEST FLAG
 5893  022342  000000              DATCNT:   0                   ;ITERATION COUNTER FOR PATTERNS TEST
 5894  022344  000000              DATPAT:   0                   ;FLAGS TYPE PATTERN
```

```
 5895  022346  000000                PATFLG: 0                       ;DATA PATTERNS <CR> SEQUENCE FLAG
 5896  022350  000000                SINGLE: 0                       ;HOLDS SINGLE CHAR TEST PATTERN
 5897  022352  000000                RETFLG: 0                       ;ECHO TEST RETURN FLAG FM SETUP
 5898  022354  000012                PATLIM: 10.                     ;PATTERNS TESTS ITERATION COUNT
 5899  022356  000000                RANA:   0                       ;RANDOM NO. ACCUMULATORS
 5900  022360  000000                RANB:   0
 5901  022362  000000                CLMSK:  0                       ;CHAR LENGTH BIT CLR MASK
 5902  022364  000000                EXFLAG: 0                       ;ECHO TEST EXIT FLAGS
 5903  022366  000020                ECBUF:  .BLKW   16.             ;DATA BUFFER FOR SINGLE LINE ECHO TEST
 5904  022426  000000                DHFILL: 0                       ;FILL CHAR AND COUNT FOR SINGLE LINE
 5905                                                                ;ECHO TESTS
 5906  022430  000000                FILLA:  0                       ;TEMP STORAGE FOR FILLER CHAR
 5907  022432  000000                FILLB:  0                       ;SAME FOR COUNT
 5908
 5909
```

```
                                        .SBTTL STANDARD ERROR MESSAG BUFFERS
                                        ;;*****************************************  ********************
                                        ;ERROR MESSAGE INFORMATION - MESSAGE BUFFERS AND POINTERS
                                        ;;*****************************************************************

                                        ;INFORMATION FOR MESSAGE 1

022434  047516  020116  054105  EM1:    .ASCIZ  'NON EX MEMORY ERROR - DROPPED LINE #  '
022442  046440  046505  051117
022450  020131  051105  047522
022456  020122  020055  051104
022464  050117  042520  020104
022472  044514  042516  021440
022500  020040     000
022503     040  050050  024503  DH1:    .ASCIZ  ' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B'
022510  020040  041440  051125
022516  050114  020122  042040
022524  053105  042101  020122
022532  051040  043505  042101
022540  020122  020040  040527
022546  020123  020040  020040
022554  027523  000102
                                        .EVEN
022560  001116  021570  001164  DT1:    .WORD   $ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4,0
022566  001166  001170  001172
022574  000000
022576     000     000     000  DF2:    .BYTE   0,0,0,0,0,0,0,0
022601     000     000     000
022604     000     000

                                        ;INFORMATION FOR MESSAGE 2

022606  051124  047101  046523  EM2:    .ASCIZ  'TRANSMITTER FALSE INTERRUPT - DROPPED LINE #  '
022614  052111  042524  020122
022622  040506  051514  020105
022630  047111  042524  051122
022636  050125  020124  020055
022644  051104  050117  042520
022652  020104  044514  042516
022660  021440  020040     000

                                        ;INFORMATION FOR MESSAGE 3

022665     102  043125  042506  EM3:    .ASCIZ  'BUFFER ACTIVE REGISTER ERROR - DROPPED LINE #  '
022672  020122  041501  044524
022700  042526  051040  043505
022706  051511  042524  020122
022714  051105  047522  020122
022722  020055  051104  050117
022730  042520  020104  044514
022736  042516  021440  020040
022744     000

                                        ;INFORMATION FOR MESSAGE 4
```

```
 5966  022745     102  052131  020105  EM4:    .ASCIZ  'BYTE COUNT REGISTER ERROR - DROPPED LINE #  '
 5967  022752  047503  047125  020124
 5968  022760  042522  044507  052123
 5969  022766  051105  042440  051122
 5970  022774  051117  026440  042040
 5971  023002  047522  050120  042105
 5972  023010  046040  047111  020105
 5973  023016  020043  000040
 5974
 5975                                  ;INFORMATION FOR MESSAGE 5
 5976
 5977  023022  052503  051122  047105  EM5:    .ASCIZ  'CURRENT ADDRESS REGISTER ERROR - DROPPED LINE #  '
 5978  023030  020124  042101  051104
 5979  023036  051505  020123  042522
 5980  023044  044507  052123  051105
 5981  023052  042440  051122  051117
 5982  023060  026440  042040  047522
 5983  023066  050120  042105  046040
 5984  023074  047111  020105  020043
 5985  023102  000040
 5986
 5987                                  ;INFORMATION FOR MESSAGE 6
 5988
 5989  023104  044523  047514  047440  EM6:    .ASCIZ  'SILO OVERFLOW ERROR - DROPPED LINE #  '
 5990  023112  042526  043122  047514
 5991  023120  020127  051105  047522
 5992  023126  020122  020055  051104
 5993  023134  050117  042520  020104
 5994  023142  044514  042516  021440
 5995  023150  020040     000
 5996
 5997                                  ;INFORMATION FOR MESSAGE 7
 5998
 5999  023153     122  041505  044505  EM7:    .ASCIZ  'RECEIVER FALSE INTERRUPT - DROPPED LINE #  '
 6000  023160  042526  020122  040506
 6001  023166  051514  020105  047111
 6002  023174  042524  051122  050125
 6003  023202  020124  020055  051104
 6004  023210  050117  042520  020104
 6005  023216  044514  042516  021440
 6006  023224  020040     000
 6007
 6008                                  ;INFORMATION FOR MESSAGE 10
 6009
 6010  023227     111  053116  046101  EM10:   .ASCIZ  'INVALID DATA IN SILO - DROPPED LINE #  '
 6011  023234  042111  042040  052101
 6012  023242  020101  047111  051440
 6013  023250  046111  020117  020055
 6014  023256  051104  050117  042520
 6015  023264  020104  044514  042516
 6016  023272  021440  020040     000
 6017  023277     040  050050  024503  DH2:    .ASCIZ  ' (PC)   CURLPR  CHAR #  WASADR  SHBADR   WAS     S/B'
 6018  023304  020040  041440  051125
 6019  023312  050114  020122  041440
 6020  023320  040510  020122  020043
 6021  023326  053440  051501  042101
```

```
6022  023334  020122  051440  041110
6023  023342  042101  020122  020040
6024  023350  040527  020123  020040
6025  023356  020040  027523  000102
6026                                  .EVEN
6027  023364  001116  021570  001162  DT2:    .WORD   $ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4,0
6028  023372  001164  001166  001170
6029  023400  001172  000000
6030
6031                                  ;INFORMATION FOR MESSAGE 11
6032
6033  023404  040504  040524  042440  EM11:   .ASCIZ  'DATA ERROR - LINE #  '
6034  023412  051122  051117  026440
6035  023420  046040  047111  020105
6036  023426  020043  000040
6037
6038                                  ;INFORMATION FOR MESSAGE 12
6039
6040  023432  042524  052123  052040  EM12:   .ASCIZ  'TEST TIMEOUT - DROPPED LINE #  '
6041  023440  046511  047505  052125
6042  023446  026440  042040  047522
6043  023454  050120  042105  046040
6044  023462  047111  020105  020043
6045  023470  000040
6046  023472  024040  041520  020051  DH3:    .ASCIZ  ' (PC)   CURLPR  RTOTAL  XTOTAL  RDONE'
6047  023500  020040  052503  046122
6048  023506  051120  020040  052122
6049  023514  052117  046101  020040
6050  023522  052130  052117  046101
6051  023530  020040  042122  047117
6052  023536  000105
6053                                  .EVEN
6054  023540  001116  021570  001202  DT3:    .WORD   $ERRPC,CURLPR,$TMP0,$TMP1,RDONE,0
6055  023546  001204  021602  000000
6056
6057                                  ;INFORMATION FOR MESSAGE 13
6058
6059  023554  001202  001204  001206  DT4:    .WORD   $TMP0,$TMP1,$TMP2,$TMP3,$TMP4,$TMP5,$TMP6,0
6060  023562  001210  001212  001214
6061  023570  001216  000000
6062  023574     000     001     001  DF1:    .BYTE   0,1,1,1,1,1,1,0
6063  023577     001     001     001
6064  023602     001     000
6065
6066                                  ;INFORMATION FOR MESSAGE 14
6067
6068  023604  052502  020123  051105  EM14:   .ASCIZ  'BUS ERROR TRAP TO 04'
6069  023612  047522  020122  051124
6070  023620  050101  052040  020117
6071  023626  032060     000
6072  023631     040  050050  024503  DH4:    .ASCIZ  ' (PC)    (PS)    (SP)   TRAPPC  TRAPPS'
6073  023636  020040  020040  050050
6074  023644  024523  020040  020040
6075  023652  051450  024520  020040
6076  023660  052040  040522  050120
6077  023666  020103  052040  040522
```

```
 6078  023674  050120  000123
 6079                                  .EVEN
 6080  023700  001116  001202  001176  DT5:    .WORD   $ERRPC,$TMPO,$REG6,$REG1,$REG2,0
 6081  023706  001164  001166  000000
 6082
 6083                                  ;INFORMATION FOR MESSAGE 15
 6084
 6085  023714  051522  042126  044440  EM15:   .ASCIZ  'RSVD INSTR TRAP TO 10'
 6086  023722  051516  051124  052040
 6087  023730  040522  020120  047524
 6088  023736  030440  000060
 6089
 6090                                  ;INFORMATION FOR MESSAGE 16
 6091
 6092  023742  044523  043516  042514  EM16:   .ASCIZ  'SINGLE LINE ECHO TEST - INTR WAIT TIMEOUT'
 6093  023750  046040  047111  020105
 6094  023756  041505  047510  052040
 6095  023764  051505  020124  020055
 6096  023772  047111  051124  053440
 6097  024000  044501  020124  044524
 6098  024006  042515  052517  000124
 6099  024014  024040  041520  020051  DH5:    .ASCIZ  ' (PC)  'DEVADR   LINE    (SCR)  CURLPR  EXFLAG'
 6100  024022  020040  042504  040526
 6101  024030  051104  020040  046040
 6102  024036  047111  020105  020040
 6103  024044  024040  041523  024522
 6104  024052  020040  052503  046122
 6105  024060  051120  020040  054105
 6106  024066  046106  043501     000
 6107          024074                  .EVEN
 6108  024074  001116  001164  022144  DT6:    .WORD   $ERRPC,$REG1,LINE,$TMPO,CURLPR,EXFLAG,0
 6109  024102  001202  021570  022364
 6110  024110  000000
 6111
 6112
 6113                                  ;INFORMATION FOR MESSAGE 17
 6114
 6115  024112  046101  042524  047122  EM17:   .ASCIZ  'ALTERNATING 1/0 PATTERN TEST DONE'
 6116  024120  052101  047111  020107
 6117  024126  027461  020060  040520
 6118  024134  052124  051105  020116
 6119  024142  042524  052123  042040
 6120  024150  047117  000105
 6121  024154  024040  041520  020051  DH6:    .ASCIZ  ' (PC)   DEVADR   LINE   CURLPR  ICOUNT'
 6122  024162  020040  042504  040526
 6123  024170  051104  020040  046040
 6124  024176  047111  020105  020040
 6125  024204  052503  046122  051120
 6126  024212  020040  041511  052517
 6127  024220  052116     000
 6128          024224                  .EVEN
 6129  024224  001116  021514  022144  DT7:    .WORD   $ERRPC,DHADR,LINE,CURLPR,$REG0,0
 6130  024232  021570  001162  000000
 6131
 6132                                  ;INFORMATION FOR MESSAGE 20
 6133
```

```
 6134  024240  044502  040516  054522  EM20:   .ASCIZ  'BINARY UP COUNT PATTERN TEST DONE'
 6135  024246  052440  020120  047503
 6136  024254  047125  020124  040520
 6137  024262  052124  051105  020116
 6138  024270  042524  052123  042040
 6139  024276  047117  000105
 6140
 6141                                  ;INFORMATION FOR MESSAGE 21
 6142
 6143  024302  044502  040516  054522  EM21:   .ASCIZ  'BINARY DOWN COUNT PATTERN TEST DONE'
 6144  024310  042040  053517  020116
 6145  024316  047503  047125  020124
 6146  024324  040520  052124  051105
 6147  024332  020116  042524  052123
 6148  024340  042040  047117  000105
 6149
 6150                                  ;INFORMATION FOR MESSAGE 22
 6151
 6152  024346  040522  042116  046517  EM22:   .ASCIZ  'RANDOM DATA PATTERN TEST DONE'
 6153  024354  042040  052101  020101
 6154  024362  040520  052124  051105
 6155  024370  020116  042524  052123
 6156  024376  042040  047117  000105
 6157
 6158                                  ;INFORMATION FOR MESSAGE 23
 6159
 6160  024404  044523  043516  042514  EM23:   .ASCIZ  'SINGLE CHAR PATTERN TEST DONE'
 6161  024412  041440  040510  020122
 6162  024420  040520  052124  051105
 6163  024426  020116  042524  052123
 6164  024434  042040  047117  000105
 6165
 6166                                  ;INFORMATION    FOR MESSAGE 24
 6167
 6168  024442  054524  042520  020104  EM24:   .ASCIZ  'TYPED BUFFER PATTERN TEST DONE'
 6169  024450  052502  043106  051105
 6170  024456  050040  052101  042524
 6171  024464  047122  052040  051505
 6172  024472  020124  047504  042516
 6173  024500     000
 6174
 6175
 6176                                  ;INFORMATION FOR MESSAGE 25
 6177
 6178  024501     104  052101  020101  EM25:   .ASCIZ  'DATA PATTERNS TEST TIMEOUT'
 6179  024506  040520  052124  051105
 6180  024514  051516  052040  051505
 6181  024522  020124  044524  042515
 6182  024530  052517  000124
 6183  024534  024040  041520  020051  DH7:    .ASCIZ  ' (PC)'  DEVADR   LINE   CURLPR  ICOUNT  PATCDE'
 6184  024542  020040  042504  040526
 6185  024550  051104  020040  046040
 6186  024556  047111  020105  020040
 6187  024564  052503  046122  051120
 6188  024572  020040  041511  052517
 6189  024600  052116  020040  040520
```

```
 6190  024606  041524  042504      000
 6191          024614                      .EVEN
 6192  024614  001116  021514  022144  DT10:   .WORD   $ERRPC,DHADR,LINE,CURLPR,$REG0,$REG1,0
 6193  024622  021570  001162  001164
 6194  024630  000000
 6195
 6196                                      .EVEN
```

```
                                        .SBTTL MISCELANEOUS TABLES AND  MESSAGE AND DATA BUFFERS
                                        ;;*****************************************************************
                                        ;MISCELLANEOUS MESSAGES
                                        ;;*****************************************************************

024632  005015  055103  044104  TITLE:  .ASCIZ  <15><12>'CZDHN-D DH11 DATA RELIABILITY TEST'<15><12>
024640  026516  020104  044104
024646  030461  042040  052101
024654  020101  042522  044514
024662  041101  046111  052111
024670  020131  042524  052123
024676  005015     000
024701     015  052012  051505  TITLE2: .ASCIZ  <15><12>'TESTING DH11 #  '<15><12>
024706  044524  043516  042040
024714  030510  020061  020043
024722  006440  000012
024726  005015  054524  042520  INMSG1: .ASCIZ  <15><12>'TYPE SCR ADDRESS FOR FIRST DH11'<15><12>
024734  051440  051103  040440
024742  042104  042522  051523
024750  043040  051117  043040
024756  051111  052123  042040
024764  030510  006461  000012
024772  005015  054524  042520  INMSG2: .ASCIZ  <15><12>'TYPE VECTOR ADDRESS FOR FIRST DH11'<15><12>
025000  053040  041505  047524
025006  020122  042101  051104
025014  051505  020123  047506
025022  020122  044506  051522
025030  020124  044104  030461
025036  005015     000
025041     015  052012  050131  INMSG3: .ASCIZ  <15><12>'TYPE DH11 DEVICE SELECTION PARAMETER'<15><12>
025046  020105  044104  030461
025054  042040  053105  041511
025062  020105  042523  042514
025070  052103  047511  020116
025076  040520  040522  042515
025104  042524  006522  000012
025112  005015  047111  040526  INMSG4: .ASCIZ  <15><12>'INVALID DH11 SCR ADDRESS - TRY AGAIN'<15><12>
025120  044514  020104  044104
025126  030461  051440  051103
025134  040440  042104  042522
025142  051523  026440  052040
025150  054522  040440  040507
025156  047111  005015     000
025163     015  044412  053116  INMSG5: .ASCIZ  <15><12>'INVALID DH11 VECTOR ADDRESS - TRY AGAIN'<15><12>
025170  046101  042111  042040
025176  030510  020061  042526
025204  052103  051117  040440
025212  042104  042522  051523
025220  026440  052040  054522
025226  040440  040507  047111
025234  005015     000
025237     015  054412  052517  INMSG6: .ASCIZ  <15><12>'YOU MUST SELECT AT LEAST ONE DH11'<15><12>
025244  046440  051525  020124
025252  042523  042514  052103
025260  040440  020124  042514
025266  051501  020124  047117
```

```
CZDHN-D MACY11 30A(1052)  27-DEC-78  15:31  PAGE 136
CZDHND.P11    27-DEC-78 15:28          MISCELANEOUS TABLES AND  MESSAGE AND DATA BUFFERS                                SEQ 0135

 6253  025274  020105  044104  030461
 6254  025302  005015     000
 6255  025305     015  042012  050105  INMSG7: .ASCIZ  <15><12>'DEPRESS ''CONTINUE'' TO START TESTING'<15><12>
 6256  025312  042522  051523  021040
 6257  025320  047503  052116  047111
 6258  025326  042525  020042  047524
 6259  025334  051440  040524  052122
 6260  025342  052040  051505  044524
 6261  025350  043516  005015     000
 6262  025355     015  052012  050131  INMSGA: .ASCIZ  <15><12>'TYPE LINE SELECTION PARAMETER'<15><12>
 6263  025362  020105  044514  042516
 6264  025370  051440  046105  041505
 6265  025376  044524  047117  050040
 6266  025404  051101  046501  052105
 6267  025412  051105  005015     000
 6268
 6269  025417     015  052012  050131  VCWC:   .ASCIZ  <15><12>'TYPE NO. OF ADDRESSES (OCTAL) BETWEEN VECTORS (10 OR 20)'<15><1
 6270  025424  020105  047516  020056
 6271  025432  043117  040440  042104
 6272  025440  042522  051523  051505
 6273  025446  024040  041517  040524
 6274  025454  024514  041040  052105
 6275  025462  042527  047105  053040
 6276  025470  041505  047524  051522
 6277  025476  024040  030061  047440
 6278  025504  020122  030062  006451
 6279  025512  000012
 6280  025514  005015  044104  021440  STMSG1: .ASCIZ  <15><12>'DH #   STATISTICS:'
 6281  025522  020040  051440  040524
 6282  025530  044524  052123  041511
 6283  025536  035123     000
 6284  025541     015  052012  051505  STMSG2: .ASCIZ  <15><12>'TESTING LINE #  '<15><12>
 6285  025546  044524  043516  046040
 6286  025554  047111  020105  020043
 6287  025562  006440  000012
 6288  025566  005015  044514  042516  STMSG3: .ASCIZ  <15><12>'LINE #   WAS DROPPED'<15><12>
 6289  025574  021440  020040  053440
 6290  025602  051501  042040  047522
 6291  025610  050120  042105  005015
 6292  025616     000
 6293  025617     015  046012  047111  STMSG4: .ASCIZ  <15><12>'LINE #  RTOTAL  XTOTAL  DATERR  PARERR  FRMERR  OVRERR'<15><12>
 6294  025624  020105  020043  051040
 6295  025632  047524  040524  020114
 6296  025640  054040  047524  040524
 6297  025646  020114  042040  052101
 6298  025654  051105  020122  050040
 6299  025662  051101  051105  020122
 6300  025670  043040  046522  051105
 6301  025676  020122  047440  051126
 6302  025704  051105  006522  000012
 6303                          ;MESSAGES USED BY THE AUTOSIZER.
 6304  025712  047516  042040  030510  MSG1:   .ASCIZ  /NO DH11'S WERE FOUND/<15><12>
 6305  025720  023461  020123  042527
 6306  025726  042522  043040  052517
 6307  025734  042116  005015     000
 6308  025741     116  020117  044104  MSG2:   .ASCIZ  /NO DH RECEIVER INTERRUPT OCCURRED/<15><12>
```

```
CZDHN-D MACY11 30A(1052)  27-DEC-78  15:31  PAGE 137
CZDHND.P11    27-DEC-78 15:28          MISCELANEOUS TABLES AND  MESSAGE AND DATA BUFFERS                                SEQ 0136

 6309  025746  051040  041505  044505
 6310  025754  042526  020122  047111
 6311  025762  042524  051122  050125
 6312  025770  020124  041517  052503
 6313  025776  051122  042105  005015
 6314  026004     000
 6315  026005     116  020117  046504  MSG3:    .ASCIZ  /NO DM11-BB INTERRUPT OCCURRED./<15><12>
 6316  026012  030461  041055  020102
 6317  026020  047111  042524  051122
 6318  026026  050125  020124  041517
 6319  026034  052503  051122  042105
 6320  026042  006456  000012
 6321  026046  020040  000040          SPACE:   .ASCIZ  /   /
 6322  026052  005015  044104  030461  DEVMAP:  .ASCII  <15><12>/DH11, DM11-BB DEVICE MAP:/<15><12>
 6323  026060  020054  046504  030461
 6324  026066  041055  020102  042504
 6325  026074  044526  042503  046440
 6326  026102  050101  006472     012
 6327  026107     015  042012  030510          .ASCII  <15><12>/DH11   DH11   DM11-BB   DM11-BB/
 6328  026114  020061  020040  044104
 6329  026122  030461  020040  042040
 6330  026130  030515  026461  041102
 6331  026136  020040  042040  030515
 6332  026144  026461  041102
 6333  026150  005015  042101  051522          .ASCIZ  <15><12>/ADRS   VECT    ADRS      VECT/<15><12><15><12>
 6334  026156  020040  053040  041505
 6335  026164  020124  020040  040440
 6336  026172  051104  020123  020040
 6337  026200  020040  053040  041505
 6338  026206  006524  006412  000012
 6339                                  ;MESSAGES FOR INPUTTING PARAMETERS TO ECHO TESTS
 6340
 6341  026214  020040  006440  000012  EC:      .ASCIZ  '   '<15><12>
 6342  026222  000040                  EC2:     .ASCIZ  ' '
 6343  026224  020040  005015     000  EC3:     .ASCIZ  '  '<15><12>
 6344  026231     015  051412  047111  ECMSG1:  .ASCIZ  <15><12>'SINGLE LINE ECHO TEST - CONNECT TERMINAL TO DH11 TEST LINE'<15>
 6345  026236  046107  020105  044514
 6346  026244  042516  042440  044103
 6347  026252  020117  042524  052123
 6348  026260  026440  041440  047117
 6349  026266  042516  052103  052040
 6350  026274  051105  044515  040516
 6351  026302  020114  047524  042040
 6352  026310  030510  020061  042524
 6353  026316  052123  046040  047111
 6354  026324  006505  000012
 6355
 6356
 6357  026330  005015  054524  042520  ECMSG2.  .ASCIZ  <15><12>'TYPE LINE # (00 - 17 OCTAL)'
 6358  026336  046040  047111  020105
 6359  026344  020043  030050  020060
 6360  026352  020055  033461  030040
 6361  026360  052103  046101  000051
 6362
 6363  026366  005015  042524  052123  ECMSG3:  .ASCIZ  <15><12>'TESTING LINE #   - GO TYPE IN ON TEST LINE'<15><12>
 6364  026374  047111  020107  044514
```

```
 6365  026402  042516  021440  020040
 6366  026410  026440  043440  020117
 6367  026416  054524  042520  044440
 6368  026424  020116  047117  052040
 6369  026432  051505  020124  044514
 6370  026440  042516  005015     000
 6371
 6372  026445     015  052012  050131  ECMSG4: .ASCIZ  <15><12>'TYPE: [CONTROL-C TO EXIT] [CONTROL-E TO ECHO BUFFER]'<15><12>
 6373  026452  035105  055440  047503
 6374  026460  052116  047522  026514
 6375  026466  020103  047524  042440
 6376  026474  044530  056524  055440
 6377  026502  047503  052116  047522
 6378  026510  026514  020105  047524
 6379  026516  042440  044103  020117
 6380  026524  052502  043106  051105
 6381  026532  006535  000012
 6382  026536  005015  047504  054440  LPMSG:  .ASCIZ  <15><12>'DO YOU WANT TO CHANGE 'LPR'' (Y OR N) ? '
 6383  026544  052517  053440  047101
 6384  026552  020124  047524  041440
 6385  026560  040510  043516  020105
 6386  026566  046042  051120  020042
 6387  026574  054450  047440  020122
 6388  026602  024516  037440  000040
 6389  026610  005015  051124  047101  XSMSG1: .ASCIZ  <15><12>'TRANSMITTER SPEED ?'
 6390  026616  046523  052111  000224
 6391  026624  020122  050123  042505
 6392  026632  020104  000077
 6393
 6394  026636  005015  047111  040526  XSMSG2: .ASCIZ  <15><12>'INVALID XMIT SPEED - TRY AGAIN'<15><12>
 6395  026644  044514  020104  046530
 6396  026652  052111  051440  042520
 6397  026660  042105  026440  052040
 6398  026666  054522  040440  040507
 6399  026674  047111  005015     000
 6400
 6401  026701     015  051012  041505  RSMSG1: .ASCIZ  <15><12>'RECEIVER SPEED ?'
 6402  026706  044505  042526  020122
 6403  026714  050123  042505  020104
 6404  026722  000077
 6405
 6406  026724  005015  047111  040526  RSMSG2: .ASCIZ  <15><12>'INVALID RCVR SPEED - TRY AGAIN'<15><12>
 6407  026732  044514  020104  041522
 6408  026740  051126  051440  042520
 6409  026746  042105  026440  052040
 6410  026754  054522  040440  040507
 6411  026762  047111  005015     000
 6412
 6413  026767     015  041412  040510  CLMSG1: .ASCIZ  <15><12>'CHAR LENGTH (6, 7, OR 8) ? '
 6414  026774  020122  042514  043516
 6415  027002  044124  024040  026066
 6416  027010  033440  020054  051117
 6417  027016  034040  020051  020077
 6418  027024     000
 6419
 6420  027025     015  044412  053116  CLMSG2: .ASCIZ  <15><12>'INVALID CHAR LENGTH - TRY AGAIN'<15><12>
```

```
 6421  027032  046101  042111  041440
 6422  027040  040510  020122  042514
 6423  027046  043516  044124  026440
 6424  027054  052040  054522  040440
 6425  027062  040507  047111  005015
 6426  027070     000
 6427
 6428  027071     015  047012  027117  SBMSG1: .ASCIZ  <15><12>'NO. OF STOP BITS (1 OR 2) ? '
 6429  027076  047440  020106  052123
 6430  027104  050117  041040  052111
 6431  027112  020123  030450  047440
 6432  027120  020122  024462  037440
 6433  027126  000040
 6434
 6435  027130  005015  047111  040526  SBMSG2: .ASCIZ  <15><12>'INVALID NO. STOP BITS - TRY AGAIN'<15><12>
 6436  027136  044514  020104  047516
 6437  027144  020056  052123  050117
 6438  027152  041040  052111  020123
 6439  027160  020055  051124  020131
 6440  027166  043501  044501  006516
 6441  027174  000012
 6442
 6443  027176  005015  040520  044522  PBMSG1: .ASCIZ  <15><12>'PARITY SELECTION (E, O, OR <CR>) ? '
 6444  027204  054524  051440  046105
 6445  027212  041505  044524  047117
 6446  027220  024040  026105  047440
 6447  027226  020054  051117  036040
 6448  027234  051103  024476  037440
 6449  027242  000040
 6450
 6451  027244  005015  047111  040526  PBMSG2: .ASCIZ  <15><12>'INVALID PARITY - TRY AGAIN'<15><12>
 6452  027252  044514  020104  040520
 6453  027260  044522  054524  026440
 6454  027266  052040  054522  040440
 6455  027274  040507  047111  005015
 6456  027302     000
 6457
 6458  027303     015  043012  046111  FILC1:  .ASCIZ  <15><12>'FILLER CHARACTER ? '
 6459  027310  042514  020122  044103
 6460  027316  051101  041501  042524
 6461  027324  020122  020077     000
 6462
 6463  027331     015  043012  046111  FILC2:  .ASCIZ  <15><12>'FILLER COUNT ?'
 6464  027336  042514  020122  047503
 6465  027344  047125  020124  000077
 6466  027352  005015  042523  042116  SNMSG1: .ASCIZ  <15><12>'SEND MODE -(Y OR N)  '
 6467  027360  046440  042117  020105
 6468  027366  024055  020131  051117
 6469  027374  047040  020051  000040
 6470  027402  005015  054524  042520  SNMSG2: .ASCIZ  <15><12>'TYPE SEND BUFFER - TERMINATE WITH CONTROL-C'<15><12>
 6471  027410  051440  047105  020104
 6472  027416  052502  043106  051105
 6473  027424  026440  052040  051105
 6474  027432  044515  040516  042524
 6475  027440  053440  052111  020110
 6476  027446  047503  052116  047522
```

MISCELANEOUS TABLES AND MESSAGE AND DATA BUFFERS

SEQ 0139

```
 6477  027454  026514  006503  000012
 6478  027462  005015  044103  047101  SNMSG3: .ASCIZ  <15><12>'CHANGE PARAMETERS (Y OR N)? '
 6479  027470  042507  050040  051101
 6480  027476  046501  052105  051105
 6481  027504  020123  054450  047440
 6482  027512  020122  024516  020077
 6483  027520     000
 6484
```

```
 6485                                   ;MESSAGES FO INPUTTING PATTERNS TEST PARAMETERS
 6486  027521    015  042012  052101  DPMSG1: .ASCIZ  <15><12>'DATA PATTERNS TESTS - CONNECT TEST JUMPER'<15><12>
 6487  027526 020101  040520  052124
 6488  027534 051105  051516  052040
 6489  027542 051505  051524  026440
 6490  027550 041440  047117  042516
 6491  027556 052103  052040  051505
 6492  027564 020124  052512  050115
 6493  027572 051105  005015     000
 6494  027577    015  041012  043125  DPMSG2: .ASCIZ  <15><12>'BUFFER SIZE ? (1 - 512)'
 6495  027604 042506  020122  044523
 6496  027612 042532  037440  024040
 6497  027620 020061  020055  030465
 6498  027626 024462     000
 6499  027631    015  044412  053116  DPMSG3: .ASCIZ  <15><12>'INVALID SIZE - TRY AGAIN'<15><12>
 6500  027636 046101  042111  051440
 6501  027644 055111  020105  020055
 6502  027652 051124  020131  043501
 6503  027660 044501  006516  000012
 6504  027666 005015  040520  052124  DPMSG4: .ASCIZ  <15><12>'PATTERN TYPE ? (A,U,D,R,S,B OR <CR>) '<15><12>
 6505  027674 051105  020116  054524
 6506  027702 042520  037440  024040
 6507  027710 026101  026125  026104
 6508  027716 026122  026123  020102
 6509  027724 051117  036040  051103
 6510  027732 024476  006440  000012
 6511  027740 005015  042523  020124  DPMSG5: .ASCIZ  <15><12>'SET SR07=1 TO LOCK ON PATTERN'<15><12>
 6512  027746 051123  033460  030475
 6513  027754 052040  020117  047514
 6514  027762 045503  047440  020116
 6515  027770 040520  052124  051105
 6516  027776 006516  000012
 6517  030002 005015  047111  040526  DPMSG6: .ASCIZ  <15><12>'INVALID PATTERN - TRY AGAIN'<15><12>
 6518  030010 044514  020104  040520
 6519  030016 052124  051105  020116
 6520  030024 020055  051124  020131
 6521  030032 043501  044501  006516
 6522  030040 000012
 6523  030042 005015  054524  042520  DPMSG7: .ASCIZ  <15><12>'TYPE SINGLE TEST CHAR '<15><12>
 6524  030050 051440  047111  046107
 6525  030056 020105  042524  052123
 6526  030064 041440  040510  020122
 6527  030072 005015     000
 6528  030075    015  052012  050131  DPMSGA: .ASCIZ  <15><12>'TYPE IN TEST BUFFER - TERMINATE WITH CONTROL-C'<15><12>
 6529  030102 020105  047111  052040
 6530  030110 051505  020124  052502
 6531  030116 043106  051105  026440
 6532  030124 052040  051105  044515
 6533  030132 040516  042524  053440
 6534  030140 052111  020110  047503
 6535  030146 052116  047522  026514
 6536  030154 006503  000012
 6537                                   .EVEN
 6538                                   ;ERROR STATISTICS TABLES
 6539
 6540  030160 000020                   RTOTAL: .BLKW   16.              ;TOTAL CHARS RCVD PER LINE
```

```
 6541  030220  000020                XTOTAL: .BLKW   16.             ;TOTAL CHARS XMITTED PER LINE
 6542  030260  000020                DATERR: .BLKW   16.             ;TOTAL DATA COMPARE ERRORS PER LINE
 6543  030320  000020                PARERR: .BLKW   16.             ;TOTAL PARITY ERRORS PER LINE
 6544  030360  000020                OVRERR: .BLKW   16.             ;TOTAL OVERRUN ERRORS PER LINE
 6545  030420  000020                FRMERR: .BLKW   16.             ;TOTAL FRAMING ERRORS PER LINE
 6546
 6547                                ;STATISTICS TABLES POINTERS
 6548
 6549  030460  000000                DEPTR:  0                       ;CONTAINS POINTERS TO STAT TABLES
 6550  030462  000000                PEPTR:  0
 6551  030464  000000                ORPTR:  0
 6552  030466  000000                FRPTR:  0
 6553
 6554  030470  000000                RBFPTR: 0                       ;CONTAINS INPUT BUFFER POINTER
 6555  030472  000000                TBFPTR: 0                       ;CONTAINS OUTPUT BUFFER POINTER
 6556
 6557
 6558                                ;600. WORD RECEIVER INPUT BUFFER
 6559
 6560  030474  001130                RBUF:   .BLKW   600.
 6561
 6562
 6563                                ;600(10) BYTE TRANSMITTER OUTPUT DATA BUFFER
 6564
 6565                                .EVEN
 6566  032754  001130                TBUF:   .BLKB   600.
 6567
 6568  034104  000000                ENBUFS: 0                       ;MARK END OF BUFFERS
 6569
 6570          000001                .END
```

```
ABASE = 000000            2321    2362
ACDW1 = 000000            2321    2364
ACDW2 = 000000            2321    2365
ACPUOP= 000000            2321    2336
ADDW0 = 000000            2321    2366
ADDW1 = 000000            2321    2367
ADDW10= 000000            2321
ADDW11= 000000            2321
ADDW12= 000000            2321
ADDW13= 000000            2321
ADDW14= 000000            2321
ADDW15= 000000            2321
ADDW2 = 000000            2321    2368
ADDW3 = 000000            2321    2369
ADDW4 = 000000            2321
ADDW5 = 000000            2321
ADDW6 = 000000            2321
ADDW7 = 000000            2321
ADDW8   000000            2321
ADDW9 = 000000            2321
ADEVCT= 000000            2321    2327
ADEVM = 000000            2321    2363
ADPTR   022322            2629*   2636*   2644    3031*   5883#
ADRVEC  022134            2612    4871*   5811#
AENV  = 000000            2321    2332
AENVM - 000000            2321    2333
AFATAL= 000000            2321    2324
AMADR1- 000000            2321    2349
AMADR2  000000            2321    2353
AMADR3= 000000            2321    2356
AMADR4= 000000            2321    2359
AMAMS1= 000000            2321    2343
AMAMS2= 000000            2321    2351
AMAMS3= 000000            2321    2354
AMAMS4= 000000            2321    2357
AMSGAD- 000000            2321    2329
AMSGLG= 000000            2321    2330
AMSGTY= 000000            2321    2323
AMTYP1= 000000            2321    2344
AMTYP2- 000000            2321    2352
AMTYP3= 000000            2321    2355
AMTYP4- 000000            2321    2358
APASS - 000000            2321    2326
APRIOR= 000000            2321
APTCSU= 000040            4119    4224#
APTENV= 000001            3870    4112    4180    4222#
APTSIZ- 000200            2581    4221#
APTSPO- 000100            4114    4182    4223#
ASWREG= 000000            2321    2334
ATESTN= 000000            2321    2325
AUNIT = 000000            2321    2328
AUSWR = 000000            2321    2335
AUTOSZ  015144            2607    4710#
AVECT1- 000000            2321    2360
AVECT2- 000000            2321    2361
BAR     000012            2699*   2770    2774    2778    3154*   3204*   3513*   3578    3582    3586    5263*   5624#
```

```
BCR     = 000010        2789    2793    2797    3150*   3178*   3189*   3200*   3597    3601    3605    4652*   5260*   5282*
                        5623#
BEGIN     001576        2143    2538#
BEGINA    001614        2542#   3049    3264    4926    4931
BIT0    = 000001        2242#   2605    2610    2838    4934    4945    4964    4973    5007    5045
BIT00   = 000001        2232#   2242
BIT01   = 000002        2231#   2241
BIT02   = 000004        2230#   2240
BIT03   = 000010        2229#   2239
BIT04   = 000020        2228#   2238
BIT05   = 000040        2227#   2237
BIT06   = 000100        2226#   2236    3110    3163    3170
BIT07   = 000200        2225#   2235    2867    3664
BIT08   = 000400        2224#   2234
BIT09   = 001000        2223#   2233    3817    3881
BIT1    = 000002        2241#   2909    4872
BIT10   = 002000        2222#   2732    3540
BIT11   = 004000        2221#   2711    2726    2737    2755    2775    2794    2847    2866    2884    3101    3130    3523
                        3532    3545    3563    3583    3602    3644    3663    3681    3824    4620    5256
BIT12   = 010000        2220#
BIT13     020000        2219#   3138    3153    3203    3865
BIT14   = 040000        2218#   2842    3639    3799
BIT15   = 100000        2217#   2756    2935    3138    3564    3714
BIT2    = 000004        2240#
BIT3      000010        2239#
BIT4      000020        2238#
BIT5    = 000040        2237#
BIT6    = 000100        2236#
BIT7    = 000200        2235#
BIT8      000400        2234#   4981
BIT9    = 001000        2233#
BKR     = 000014        5625#
BPTVEC= 000014          2249#
BRLVL     021710        2631    5720#
BRPTR     022326        2631*   2638*   2646    3033*   5885#
BUSER     016754        2596    5067#
CAR     = 000006        2808    2817    3151*   3201*   3616    3625    4633*   5261*   5622#
CEXIT     022336        3097*   3123    3157*   5891#
CHKADR    016446        4941    4989#
CHKVCT    016612        4952    5027#
CHPS1     021420        2700    3119    3514    5074    5084    5248    5587#
CHPS2     021434        2736    2754    2773    2792    2815    2845    2863    2883    3096    3544    3562    3581    3600
                        3623    3642    3660    3680    4621    5595#
CHRCNT    021574        2687*   2809    2831    2904    3220    3288*   3498*   3617    4628    4632    5162*   5165*   5182
                        5508    5518    5530    5542    5559    5665#
CKER      004076        2727    2917#
CKERDP    010566        3533    3706#
CKRST1    014560        4586    4601#
CKRST2    014610        4609#   5075    5085
CKSWR   = 104407        3297    3797    3857    3880    4544#
CLALL     021402        5507    5517    5529    5541    5558    5579#
CLMSG1    026767        5382    6413#
CLMSG2    027025        5406    6420#
CLMSK     022362        3715    5569*   5573*   5901#
CLSEL     021576        2688*   5155*   5156    5161*   5164    5667#
CLSTAT    017130        2658    5116#
```

```
CR     = 000015  2157#  4158   4168
CRLF   = 000200  2158#  4129   4168
CURLPR  021570   3111   4631   5131   5133*  5136   5163*  5164*  5206*  5207*  5262   5290*  5297*  5309*
                 5319*  5326*  5338*  5365   5367*  5383*  5397*  5401*  5408*  5429*  5436*  5444*  5457*
                 5461*  5568   5662#  5933   6027   6054   6108   6129   6192
DATCNT  022342   3348*  3351*  3352   3355   3356*  3366*  3369*  3370   3373   3374*  3384*  3387*  3388
                 3391   3392*  3402*  3405*  3406   3409   3410*  3433*  3451*  3452   3455   3456*  3463*
                 3500*  3501   3504   3505*  3524   5893#
DATERR  030260   2919   3016   6542#
DATPAT  022344   3307*  3320   3324   3328   3332   3336   3340   3420*  3422*  3424*  3426*  3525   5894#
DDISP  = 177570  2164#  2285   2569
DEPTR   030460   2919*  2925*  2969*  6549#
DEVMAP  026052   4875   6322#
DF1     023574   2463   6062#
DF2     022576   2393   2400   2407   2414   2421   2428   2435   2442   2449   2456   2470   2477   2484
                 2491   2498   2505   2512   2519   2526   2533   5936#
DHADR   021514   2644*  2669   2711*  3051*  3100   3266*  3300   3523*  4662*  5000*  5004*  5255   5628#
                 6129   6192
DHADRS  021750   4712   4720   4767   4783   4876   4933   5009   5739#
DHADTB  021606   2629   5006   5020   5676#
DHFILL  022426   5474*  5486*  5487   5498*  5500*  5502*  5503   5904#
DHNUM   022142   2632*  2635*  2648   2990   3030*  5815#
DHRLVL  022140   2646*  2665   3099*  3104   3290*  3293   5813#
DHSEL   021522   2639   3037   4971*  5631#
DHSET1  014640   2698   3512   4620#
DHST1   002664   2698#
DHST2   007512   3350   3368   3386   3404   3450   3499   3512#
DHTLVL  022141   2668   3107   3296   5814#
DHVCT   021516   2645*  2663   3052*  3102   3267*  3291   5038*  5042*  5090   5629#
DHVCTB  021646   2630   5044   5058   5696#
DHVEC   022006   4784   4865   4866   4868   4877   4944   5047   5758#
DHWAIT  005236   3121#
DH1     022503   2391   2398   2405   2412   2419   2426   2433   5924#
DH2     023277   2440   2447   6017#
DH3     023472   2454   6046#
DH4     023631   2468   2475   6072#
DH5     024014   2482   6099#
DH6     024154   2489   2496   2503   2510   2517   2524   6121#
DH7     024534   2531   6183#
DISPLA  001142   2285#  2569*  2577*  3839*  3860*
DISPRE  000174   2138#  2577
DMADRS  022042   4719   4825   4878   5776#
DMVEC   022100   4826   4879   5795#
DPATA   006460   3322   3348#  3421
DPATB   007272   3342   3463#
DPATCR  007050   3318   3420#  3429
DPATD   006654   3330   3384#  3425
DPATR   006752   3334   3402#  3427
DPATS   007132   3338   3432#
DPATU   006556   3326   3366#  3423
DPFLG   022340   2540*  2619   3047*  3073   3261*  4601   4609   4921*  4929*  4958   5002   5040   589<#
DPMSGA  030075   3465   6528#
DPMSG1  027521   3269   6486#
DPMSG2  027577   3275   6494#
DPMSG3  027631   3283   6499#
DPMSG4  027666   3302   6504#
```

```
DPMSG5  027740          3314    6511#
DPMSG6  030002          3345    6517#
DPMSG7  030042          3435    6523#
DRPLIN  021530          2661*   2721*   2748*   2767*   2786*   2805*   2825*   2857*   2877*   2894*   2997    5634#
DSWR    177570          2163#   2284    2568
DT1     022560          2392    2399    2406    2413    2420    2427    2434    5933#
DT10    024614          2532    6192#
DT2     023364          2441    2448    6027#
DT3     023540          2455    6054#
DT4     023554          2462    6059#
DT5     023700          2469    2476    6080#
DT6     024074          2483    6108#
DT7     024224          2490    2497    2504    2511    2518    2525    6129#
EC      026214          6341#
ECBUF   022366          3176*   3177*   3185*   3186*   3198*   3201    5272    5273*   5903#
ECHO    004672          2145    3046#   4607    4615
ECHO1   004714          2618    3051#
ECHO2   004740          3059#   3133    3254    3272    4957
ECHO3   005110          3089    3091    3096#
ECMSG1  026231          3054    6344#
ECMSG2  026330          3060    6357#
ECMSG3  026366          3114    3116    6363#
ECMSG4  026445          3118    6372#
EC2     026222          3077*   3084*   3086    3088    3090    3092    3214*   3216    3236*   3243*   3245    3248    3250
                        3252    3299*   3308*   3310    3315    3432*   3441*   3443    3446    3469*   3471    5346*   5353*
                        5355    5358    5360    5362    5380*   5388*   5390    5392    5394*   5395    5399    5403    5413*
                        5420*   5422    5424    5426*   5427    5431    5441*   5449*   5451    5453    5455    5459    5470*
                        5479*   5481    5484    5486    6342#
EC3     026224          3492*   3493*   3495    6343#
EMTVEC  000030          2252#   2553*   2554*
EM1     022434          2390    2744    3552    5917#
EM10    023227          2439    2890    3687    6010#
EM11    023404          2446    2950    3729    6033#
EM12    023432          2453    2718    6040#
EM14    023604          2467    6068#
EM15    023714          2474    6085#
EM16    023742          2481    6092#
EM17    024112          2488    6115#
EM2     022606          2397    2763    3571    5942#
EM20    024240          2495    6134#
EM21    024302          2502    6143#
EM22    024346          2509    6152#
EM23    024404          2516    6160#
EM24    024442          2523    6168#
EM25    024501          2530    6178#
EM3     022665          2404    2782    3590    5954#
EM4     022745          2411    2801    3609    5966#
EM5     023022          2418    2821    3629    5977#
EM6     023104          2425    2853    3650    5989#
EM7     023153          2432    2873    3670    5999#
ENBUFS  034104          5581    6568#
ENDA    004602          2676    3024    3027#   3038
ERRVEC- 000004          2245#   2566    2567*   2578*   2596*   2597*   3804    3805*   3807*   3810*
EXFLAG  022364          3098*   3139    3141    3143    3147*   3161*   3179*   3190*   5902#   6108
EXPAR   016442          4982    4986#
EXPAT   006072          2146    3261#   4606    4614
```

```
EXPAT1  006114            2621    3266#
EXPAT2  006140            3272#   4960
EXPAT3  006244            3297#   3346    3364    3382    3400    3418    3430    3461    3510    3530    3557    3576    3595
                          3614    3634    3655    3675    3692
FILC1   027303            5473    6458#
FILC2   027331            5494    6463#
FILLA   022430            3227    3484    5275    5276    5471*   5487*   5906#
FILLB   022432            3226    3482    5271    5279    5492*   5503*   5907#
FRMERR  030420            2922    3019    6545#
FRPTR   030466            2922*   2928*   2977*   6552#
GNS   = ****** U          2137    4536    4537    4538    4539    4540    4542    4544    4545    4546    4547    4548
GTSWR = 104406            2592    4542#
HT    = 000011            2155#   4127    4168
INCL    020364            5371    5380#   5407
INFCHR  020766            5374    5470#
INFCNT  021056            5375    5492#
INMSGA  025355            4976    6262#
INMSG1  024726            4937    6213#
INMSG2  024772            4948    6219#
INMSG3  025041            4967    6226#
INMSG4  025112            5024    6233#
INMSG5  025163            5062    6240#
INMSG6  025237            6248#
INMSG7  025305            4984    6255#
INPAR   016216            2627    3056    3270    4933#
INPARA  016112            2614    4910#   4919
INPARC  016172            2144    4927#
INPARX  016160            2141    4925#
INPAR1  016252            4940    4944#
INPAR3  016306            2626    4951    4955#
INPAR4  016370            4970    4972#
INPB    020646            5373    5441#   5465
INRSP   020110            5316#   5370
INSB    020530            5372    5413#   5435
INXSP   017774            5287#   5369
IOTVEC= 000020            2250#   2551*   2552*   4785*   4786*   4827*   4853*
KYBD1   017514            2659    5224#
KYBD2   017540            5224    5232#
LDFILL  017720            3195    5271#
LF    = 000012            2156#   4162    4168
LINE    022144            2662*   2679    2712    2717    2738    2743    2757    2762    2781    2800    2820    2833    2849
                          2852    2868    2872    2889    2906    2923    2933    2949    3062*   3063*   3064    3108    3113
                          3546    3551    3565    3570    3589    3608    3628    3646    3649    3665    3669    3686    3712
                          3728    3798*   4624    4646    4648*   4650*   4652*   4653    4661*   5257    5816#   6108    6129
                          6192
LINMSK  021526            2699    2721    2748    2767    2786    2805    2825    2857    2877    2894    2986*   2997    3010
                          3023*   3066*   3069*   3154    3204    3513    4649*   4655*   4657    5263    5633#
LINSEL  021524            3010    4657    4972*   4980*   5632#
LPMSG   026536            5348    6382#
LPR   = 000004            3111*   4631*   5262*   5621#
LPRIN   020224            3071    5346#   5363
LPRPTR  021572            2683*   5131    5133    5141*   5664#
LPRTAB  021536            2683    5648#
MSG1    025712            4769    6304#
MSG2    025741            4809    6308#
MSG3    026005            4838    6315#
```

CZDHN-D MACY11 30A(1052) 27-DEC-78 15:31 PAGE 149
CZDHND.P11 27-DEC-78 15:28 CROSS REFERENCE TABLE -- USER SYMBOLS SEQ 0147

```
NEWCL   002644          2691#   2696
NEWLIN  002544          2671#   2686    2723    2749    2768    2787    2806    2826    2858    2878    2895
NEWLPR  002616          2685#   2692
NEWPAR  002656          2695#   2728
NRC   = 000002          2880    3171    3677    5620#
ORPTR   030464          2921*   2927*   2974*   6551#
OVRERR  030360          2921    3018    6544#
PARBIT  021600          2693*   5201    5205*   5207    5208    5210*   5212    5214*   5216    5218*   5668#
PARERR  030320          2920    3017    6543#
PATFLG  022346          3298*   3317*   3359    3377    3395    3413    5895#
PATLIM  022354          3352    3370    3388    3406    3452    3501    5898#
PBMSG1  027176          5443    6443#
PBMSG2  027244          5464    6451#
PEPTR   030462          2920*   2926*   2980*   6550#
PIRQ  = 177772          2162#
PIRQVE- 000240          2256#
PRSTAT  004412          2677    2986#   5249
PR0   = 000000          2179#
PR1   - 000040          2180#
PR2   = 000100          2181#
PR3   = 000140          2182#
PR4   - 000200          2183#
PR5   = 000240          2184#
PR6   = 000300          2185#
PR7   = 000340          2186#
PS    - 177776          2159#   2160    4801*   4831*
PSW   - 177776          2160#
PTWAIT  007530          3516#
PWRVEC= 000024          2251#   2557*   2558*   4553*   4554*   4563*   4569*   4581*   4582*
QUICK   021532          2660*   5138*   5140*   5158    5203    5636#
QUICKX  021534          2690*   5153    5160*   5638#
RANA    022356          5544*   5546    5547*   5548*   5551    5899#
RANB    022360          5546*   5548    5549*   5900#
RBFEND  021604          2902    2962    3696    3699    3737    3740    4627*   4630*   5671#
RBFPTR  030470          2885*   2886    2900*   2901*   2902    2917*   2936    2942    2961*   2962    3682*   3683    3694*
                        3695*   3696    3698*   3699    3706*   3716    3722    3736*   3737    3739*   3740    4626*   6554#
RBUF    030474          2885    2917    3682    3706    4626    4627    6560#
RDCHR = 104410          3080    3212    3239    3303    3436    3467    4339    4545#   5349    5384    5416    5445    5475
RDDEC = 104413          3276    4548#   5291    5320
RDLIN = 104411          4411    4464    4546#
RDOCT - 104412          3061    4547#   4912    4938    4949    4968    4977    5495
RDONE   021602          2705    2838*   2909*   3518    3701*   4625*   5670#   6054
RESERR  017016          2598    5077#
RESTRP  017060          5090#
RESTRT  002344          2635#   2643
REST1   002402          2641#   4613
RESVEC= 000010          2246#   2598*   2599*
RETFLG  022352          2541*   2616    3046*   3262*   4603    4611    4922*   4930*   4955    4998    5036    5897#
RINT1   003534          2664    2842#
RINT2   005462          3103    3170#
RINT3   010252          3292    3639#
RSMSG1  026701          5317    6401#
RSMSG2  026724          5334    6406#
RSPTAB  022236          5318    5331    5854#
RSPTR   022234          5318*   5323    5325*   5326    5330*   5331    5852#
RSTRTA  002412          2640    2644#   3039    4605
```

```
RTOTAL  030160          2714    2908*   3014    5116    5118    6540#
SAPS    021450          5603#
SBMSG1  027071          5415    6428#
SBMSG2  027130          5434    6435#
SELINE  014736          2671    4646#
SELMSK  021520          2633*   2639    2641*   3034*   3037    5630#
SENDP1  005704          3094    3209#   3249    3251
SENDP2  017642          3235    5205#
SETCL   017232          2691    5153#
SETLPR  017146          2685    5131#
SETPAR  017372          2695    5201#
SINGLE  022350          3440*   3448*   3468*   3469    3473    3479    3483    3489    5561    5896#
SNMSG1  027352          3079    6466#
SNMSG2  027402          3210    6470#
SNMSG3  027462          3238    6478#
SOFT    004352          2958    2972#
SPACE   026046          4890    4895    6321#
SSR     = 000016        5626#
STACK   = 001100        2150#   2549    2722    5070    5080    5247
START1  002122          2596#
START2  002306          2623    2629#   2642    3778    4986
STDH1   002466          2658#
STKLMT= 177774          2161#
STMSG1  025514          2991    2993    6280#
STMSG2  025541          2680    2682    6284#
STMSG3  025566          3003    3005    6288#
STMSG4  025617          2995    6293#
SUBUF1  017332          5166    5177#
SUCLMK  021346          3273    5568#
SUER1   015122          2741    2760    2779    2798    2818    2850    2870    3549    3568    3587    3606    3626    3647
                        3667    4701#
SUER2   015116          2887    2947    3684    3726    4700#
SUNUM   015036          2647    2678    2716    2742    2761    2780    2799    2819    2851    2871    2888    2948    2988
                        3001    3112    3550    3569    3588    3607    3627    3648    3668    3685    3727    4675#
SUPATA  021130          3349    5507#
SUPATD  021206          3385    5529#
SUPATR  021240          3403    5541#
SUPATS  021320          3449    5558#
SUPATU  021156          3367    5517#
SWR     001140          2284#   2547    2568*   2570    2576*   2583*   2590    2605    2610    3362    3380    3398    3416
                        3428    3459    3508    3799    3817    3824    3865    3877    3881    4235    4272*   4561    4574*
                        4872    4934    4945    4964    4973    4981    5007    5045    5134    5236
SWREG   000176          2139#   2576    2590    4235    4248    5236
SW0     = 000001        2214#
SW00    = 000001        2204#   2214
SW01    = 000002        2203#   2213
SW02    = 000004        2202#   2212
SW03    = 000010        2201#   2211
SW04    = 000020        2200#   2210
SW05    = 000040        2199#   2209
SW06    = 000100        2198#   2208
SW07    = 000200        2197#   2207
SW08    = 000400        2196#   2206
SW09    = 001000        2195#   2205
SW1     = 000002        2213#
SW10    002000          2194#
```

```
SW11  = 004000          2193#
SW12  = 010000          2192#
SW13  = 020000          2191#
SW14  = 040000          2190#
SW15  = 100000          2189#
SW2   = 000004          2212#
SW3   = 000010          2211#
SW4   = 000020          2210#
SW5   = 000040          2209#
SW6   = 000100          2208#
SW7   = 000200          2207#
SW8   = 000400          2206#
SW9   = 001000          2205#
TBFPTR  030472          2918*  2930   2943   2960*  3707*  3709   3723   3735*  6555#
TBITVE= 000014          2247#
TBUF    032754          2811   2918   2945   3109   3145   3148   3151   3160   3211   3466   3476   3496   3619
                        3707   3725   4633   5183   5258   5261   5509   5519   5531   5543   5560   5579   6566#
TIMEA   022332          2703*  3121*  3152*  3162*  3202*  3516*  5107*  5888#
TIMEB   022334          2704*  3122*  3517*  5105*  5889#
TIMEIT  017106          2707   3125   3520   5105#
TINT1   003030          2667   2732#
TINT2   005314          3106   3138#
TINT3   007630          3295   3540#
TITFLG  022330          2542*  2600   2604*  5887#
TITLE   024632          2603   6202#
TITLE2  024701          2649   2651   6209#
TKVEC = 000060          2254#
TPVEC = 000064          2255#
TRAPVE= 000034          2253#  2555*  2556*
TRTVEC= 000014          2248#
TST1    002456          2656#
TYPDS = 104405          3767   3933   4540#
TYPE  = 104401          2602   2650   2681   2992   2994   3004   3053   3059   3078   3085   3115   3117   3209
                        3215   3237   3244   3268   3274   3282   3301   3309   3313   3344   3434   3442   3464
                        3470   3494   3765   3768   3868   3897   3914   3916   3919   3921   3936   3941   4005
                        4080   4132   4246   4247   4250   4263   4274   4293   4346   4352   4357   4361   4366
                        4367   4369   4372   4376   4440   4442   4501   4503   4536#  4583   4769   4809   4838
                        4874   4890   4895   4900   4904   4910   4936   4947   4966   4975   4983   5023   5061
                        5243   5287   5304   5316   5333   5347   5354   5381   5389   5405   5414   5421   5433
                        5442   5450   5463   5472   5480   5493
TYPOC = 104402          3905   3929   4249   4537#
TYPON = 104404          4539#
TYPOS = 104403          4538#  4883   4887   4892   4897
UPDER   004340          2946   2967#
VCFLG   021706          2539*  2608   2615*  3048*  3263*  4925*  4928*  5713#
VCPTR   022324          2630*  2637*  2645   3032*  5834#
VCWC    025417          4911   6269#
XSMSG1  026610          5288   6389#
XSMSG2  026636          5305   6394#
XSPTAB  022150          5289   5302   5826#
XSPTR   022146          5289*  5294   5296*  5297   5301*  5302   5824#
XTOTAL  030220          2715   2835*  3015   6541#
$APTHD  000000          2118   2124#
$ASTAT= ******          4202   4217
$ATYC   012650          4173   4175#
$ATY1   012624          4171#
```

```
$ATY3   012632          4117    4172#
$ATY4   012642          3873    4174#
$AUTOB  001134          2281#   2594*   4243    4392    5238
$BASE   001306          2362#
$BDADR  001122          2276#
$BDDAT  001126          2278#
$CDW1   001312          2364#
$CDW2   001314          2365#
$CHARC  012620          4134*   4144*   4151    4160*   4165#
$CKSWR  013072          4235#   4544
$CMTAG  001100          2264#   2544    2545    2553    2559    2560
$CM1  = 000010          2296#   2297#   2298#   2299#   2300#   2301#   2302#   2303#   2304#
$CM2  = 000020          2296#   2297#   2298#   2299#   2300#   2301#   2302#   2303#   2304#
$CM3  = 000010          2294#   2296
$CM4  = 000010          2304#   2305#   2306#   2307#   2308#   2309#   2310#   2311#   2312#
$CNTLG  013745          4246    4387#   5243
$CNTLU  013740          4263    4361    4386#
$CPUOP  001260          2336#
$CRLF   001227          2315#   3868    3889    3897    3916    3921    3941    4133    4168    4274    4366    4386    4445
                        4505    4901    4904
$DBLK   012332          4046    4080    4088#
$DDW0   001316          2366#
$DDW1   001320          2367#
$DDW2   001322          2368#
$DDW3   001324          2369#
$DEVCT  001242          2327#
$DEVM   001310          2363#
$DHSEL  022136          4779*   4961    5011    5049    5812#
$DOAGN  011052          3761    3770    3776#
$DTBL   012322          4049    4084#
$ENDAD  011042          3772#
$ENDCT  011010          3763#
$ENDMG  011061          3765    3780#
$FNULL  011056          3768    3779#
$ENV    001252          2332#   2588    2674    3870    4112    4180    4204
$ENVM   001253          2333#   2581    4114    4119    4182
$EOP    010754          3029*   3036    3753#
$EOPCT  011002          3760#   3764
$ERFLG  001103          2267#   3788    3813    3815    3821*   3843    3858*   3889
$ERMAX  001115          2273#   2561*   3815    3838*   3843
$ERROR  011342          2553    3856#
$ERRPC  001116          2274#   3862*   3863*   3864    3889    3903    5933    6027    6054    6080    6108    6129    6192
$ERRTB  001326          2386#   3911
$ERRTY  011514          3867    3896#
$ERTTL  001112          2271#   3861*   3889
$ESCAP  001224          2313#   2560*   3837*   3884    3886    3889
$ETABL  001252          2331#
$ETEND  001326          2130    2370#
$FATAL  001234          2324#   4208*
$FFLG   013070          4171*   4174*   4202    4211*   4219#
$FILLC  001156          2292#   4137    4168
$FILLS  001155          2291#   4168
$GDADR  001120          2275#
$GDDAT  001124          2277#
$GET42  011032          3769#
$GTSWR  013142          4247#   4542    5244
```

```
$HD     = 000001          2089    2090
$HIBTS    000000          2125#
$HIOCT    014132          4433*   4444#
$ICNT     001104          2268#   3828*   3829    3831*   3842
$ILLUP    014542          4553    4569    4588#
$INTAG    001135          2282#   4275    4392
$ITEMB    001114          2272#   3864*   3872    3889    3900
$LF       001230          2316#   3889    4168    4376    4386    4445    4505
$LFLG     013067          4212*   4218#
$LPADR    001106          2269#   2562*   2652*   3819*   3835*   3840    3842
$LPERR    001110          2270#   2563*   2719*   2745*   2764*   2783*   2802*   2822*   2854*   2874*   2891*   2951*   3020*
                          3131*   3357*   3375*   3393*   3411*   3457*   3506*   3526*   3553*   3572*   3591*   3610*   3630*
                          3651*   3671*   3688*   3730*   3819    3836*   3842    3883    5071*   5081*
$MADR1    001264          2349#
$MADR2    001270          2353#
$MADR3    001274          2356#
$MADR4    001300          2359#
$MAIL     001232          2126    2130    2322#   2580    2588    3834    3870    4112
$MAMS1    001262          2343#
$MAMS2    001266          2351#
$MAMS3    001272          2354#
$MAMS4    001276          2357#
$MBADR    000002          2126#
$MFLG     013066          4172*   4178    4213*   4217#
$MNEW     013763          4250    4390#
$MSGAD    001246          2329#   4188*   4191
$MSGLG    001250          2330#   4193*
$MSGTY    001232          2323#   4186    4194*   4206    4210*
$MSWR     013752          4247    4388#
$MTYP1    001263          2344#
$MTYP2    001267          2352#
$MTYP3    001273          2355#
$MTYP4    001277          2358#
$MXCNT    011340          3832    3842#
$NULL     001154          2290#   4139    4168
$NWTST=   000001          2653#
$OCNT     012112          3977*   4006*   4019#
$OMODE    012114          3972*   3976*   3981    3984*   3995*   4021#
$OVER     011324          3800    3820    3830    3839#
$PASS     001240          2326#   2580*   3757*   3758*   3766    3779    3826    3843
$PASTM    000006          2128#
$POWER    014550          4584    4591#
$PWRAD    014536          4586#
$PWRDN    014376          2557    4553#   4581
$PWRMG    014532          4584#
$PWRUP    014450          4563    4569#
$QUES     001226          2314#   3889    4168    4293    4369    4386    4442    4445    4503    4505
$RDCHR    013354          4306#   4545
$RDDEC    014134          4459#   4548
$RDLIN    013474          4334#   4546
$RDOCT    013774          4406#   4547
$RDSZ     000010          4327#
$REGAD    001160          2294#
$REG0     001162          2296#   3355*   3373*   3391*   3409*   3455*   3504*   3524*   4700*   6027    6129    6192
$REG1     001164          2297#   3128*   3525*   4701*   5068*   5078*   5933    6027    6080    6108    6192
$REG2     001166          2298#   4702*   5069*   5079*   5933    6027    6080
```

```
$REG3   001170          2299#   4703*   5933    6027
$REG4   001172          2300#   4704*   5933    6027
$REG5   001174          2301#
$REG6   001176          2302#   5067*   5077*   6080
$REG7   001200          2303#
$RTNAD  011054          3778#
$R2A  = ****** U        4549
$SAVRE= ****** U        4549
$SAVR6  014546          4562*   4570    4571*   4572*   4590#
$SCOPE  011076          2551    3796#   4853
$SETUP= 000117          2543#   2550    2551    2553    2555    2557    2559    2560    2562    2585    3755    3797    3857
                        3880    3888    4230    4392
$STUP = 177777          2543#
$SVLAD  011270          3808    3833#
$SVPC = 000000          2103#   2108
$SWR  = 165000          2076#   2089    2094    2095    2096    2097    2098    2099    2312    2313    2314    2559    2560
                        2562    2563    2657    3750    3756    3771    3777    3779    3789    3790    3791    3792    3793
                        3799    3811    3813    3814    3815    3822    3823    3824    3836    3839    3842    3849    3850
                        3851    3852    3861    3865    3877    3881    3889    4587
$SWREG  001254          2334#   2583
$SWRMK= 000000          3793
$TESTN  001236          2325#   3834*   4653*
$TIMES  001222          2312#   2559*   2657*   3756*   3822*   3829    3832*   3842
$TKB    001146          2287#   4228    4239    4256    4310    4316    5232
$TKS    001144          2286#   4228    4237    4253    4277*   4308    4314
$TMP0   001202          2304#   2714*   2880*   2900    3012*   3129*   3171*   3172*   3173    3182    3193    3198    3199
                        3677*   3694    5273    5274    5611*   6054    6059    6080    6108
$TMP1   001204          2305#   2715*   3014*   6054    6059
$TMP2   001206          2306#   3015*   6059
$TMP3   001210          2307#   3016*   6059
$TMP4   001212          2308#   3017*   6059
$TMP5   001214          2309#   3018*   6059
$TMP6   001216          2310#   3019*   6059
$TMP7   001220          2311#   2940*   2955    2972*   2975*   2978*   3000*   3002    3720*
$TN   = 000002          2089#   2653    2657#
$TPB    001152          2289#   4157*   4168
$TPFLG  001157          2293#   4106    4168
$TPS    001150          2288#   4155    4168
$TRAP   014312          2555    4513#
$TRAP2  014334          4524#   4535
$TRP  = 000014          4528#   4537#   4538#   4539#   4540#   4541#   4542    4543#   4544    4545#   4546#   4547#   4548#
                        4549#
$TRPAD  014346          4518    4535#
$TSTM   000004          2127#
$TSTNM  001102          2266#   3028*   3755*   3788    3833*   3834    3839    3843    3860    3889
$TTYIN  013730          4336    4337    4349    4367    4381    4385#
$TYPBN= ****** U        4541
$TYPDS  012116          4034#   4540
$TYPE   012342          4106#   4199    4528    4536
$TYPEC  012554          4136    4143    4150    4155#   4156    4279
$TYPEX  012622          4161    4163    4166#
$TYPOC  011714          3975#   4537
$TYPON  011730          3974    3977#   4539
$TYPOS  011670          3970#   4538
$UNIT   001244          2328#
$UNITM  000010          2129#
```

```
CZDHN-D MACY11 30A(1052)  27-DEC-78  15:31  PAGE 155
CZDHND.P11    27-DEC-78 15:28          CROSS REFERENCE TABLE -- USER SYMBOLS                                SEQ 0153

$USWR   001256            2335#
$VECT1  001302            2360#
$VECT2  001304            2361#
$XTSTR  011114            3802#
$$GET4= 000000            3771#
$OFILL  012113            3971*   3975*   3985    4020#
$40CAT= ****** U          3799    3867
.       = 034106          2103    2104#   2106#   2108#   2114    2115#   2117#   2119#   2133#   2137#   2263#   2317    2548
                          2562    2563    2652    3779    3783    3842    3843    3889    3944#   4088#   4168    4220#   4228
                          4385#   4386    4392    4445    4505    4565    4589    5903#   6107#   6128#   6191#   6540#   6541#
                          6542#   6543#   6544#   6545#   6560#   6566#
.$ASTA= ****** U          4172    4175
.$X     = 000000          2114#   2119
```

```
COMMEN      1#    2257#
ENDCOM      1#    2257#
ERROR    2151#    2720    2746    2765    2784    2803    2823    2855    2875    2892    2952    3021    3152    3358    3376
          3394    3412    3458    3507    3527    3554    3573    3592    3611    3631    3652    3672    3689    3731    5072
          5082
ESCAPE      1#    2257#
GETPRI      1#    2257#
GETSWR      1#    2077#   2257#   2585
MULT        1#    2257#
NEWTST      1#    2257#   2653
POP         1#    2257#   2836    2910    4075    4214    4215    4434    4494    4574    4575    4692    5188
PUSH        1#    2257#   2828    2897    4034    4175    4177    4198    4408    4461    4555    4561    4675    5177
REPORT      1#    2257#
SCOPE    2152#    2656    3027    3754
SETPRI      1#    2257#
SETTRA   4528#    4537    4538    4539    4540    4542    4544    4545    4546    4547    4548
SETUP       1#    2077#   2257#   2543
SKIP        1#    2257#
SLASH       1#    2257#
SPACE    2257#
STARS       1#    2101    2111    2113    2120    2257#   2259    2317    2320    2653    2655    3747    3785    3845    3891
          3947    4024    4091    4170    4227    4230    4298    4327    4394    4447    4507    4551    4567    4594    4596
          5616    5618    5911    5913    6198    6200
SWRSU       1#    2257#   2564#
TRMTRP   4528#
TYPBIN      1#    2257#
TYPDEC      1#    2257#   3766    3931
TYPNAM      1#    2257#
TYPNUM      1#    2257#
TYPOCS      1#    2257#
TYPOCT      1#    2257#   3903    3928    4248
TYPTXT      1#    2257#
$$CMRE   2257#    2296    2297    2298    2299    2300    2301    2302    2303
$$CMTM   2257#    2304    2305    2306    2307    2308    2309    2310    2311
$$ESCA      1#    2257#
$$NEWT      1#    2257#   2653
$$SET    4528#    4537    4538    4539    4540    4542    4544    4545    4546    4547    4548
$$SETM   2580#
$$SKIP      1#    2257#
.EQUAT      1#    2077#   2147
.HEADE      1#    2077#   2079
.KT11       1#
.SETUP      1#    2077#   2543
.SWRHI      1#    2077#   2090
.SWRLO   2077#    2099#
.$ACT1      1#    2077#   2099
.$APTB      1#    2077#   2318#
.$APTH      1#    2077#   2109
.$APTY      1#    2077#   4168
.$ASTA      1#
.$CATC      1#    2077#   2131
.$CMTA      1#    2077#   2257
.$DB2D      1#
.$DB2O      1#
.$DIV       1#
.$EOP       1#    2077#   3745
```

```
.$ERRO       1#    2077#    3843
.$ERRT       1#    2077#    3889
.$MULT       1#
.$POWE       1#    2077#    4549
.$RAND       1#
.$RDDE       1#    2077#    4445
.$RDOC       1#    2077#    4392
.$READ       1#    2077#    4225
.$R2AZ       1#
.$SAVE       1#
.$SB2D       1#
.$SB2O       1#
.$SCOP       1#    2077#    3783
.$SIZE       1#
.$SUPR       1#
.$TRAP       1#    2077#    4505
.$TYPB       1#
.$TYPD       1#    2077#    4022
.$TYPE       1#    2077#    4089
.$TYPO       1#    2077#    3945
.$40CA       1#
.1170        1#


. ABS.  034106     000


 ERRORS DETECTED:  0

 CZDHND.BIN,CZDHND.SEQ/CRF/SOL/NL:TOC=CZDHND.SML,CZDHND.P11
 RUN-TIME: 15 22 1 SECONDS
 RUN-TIME RATIO: 205/39 5.2
 CORE USED:  33K  (65 PAGES)
```