DH11

RELIABILITY TEST SINGLE

MD-11-DZDHN-C

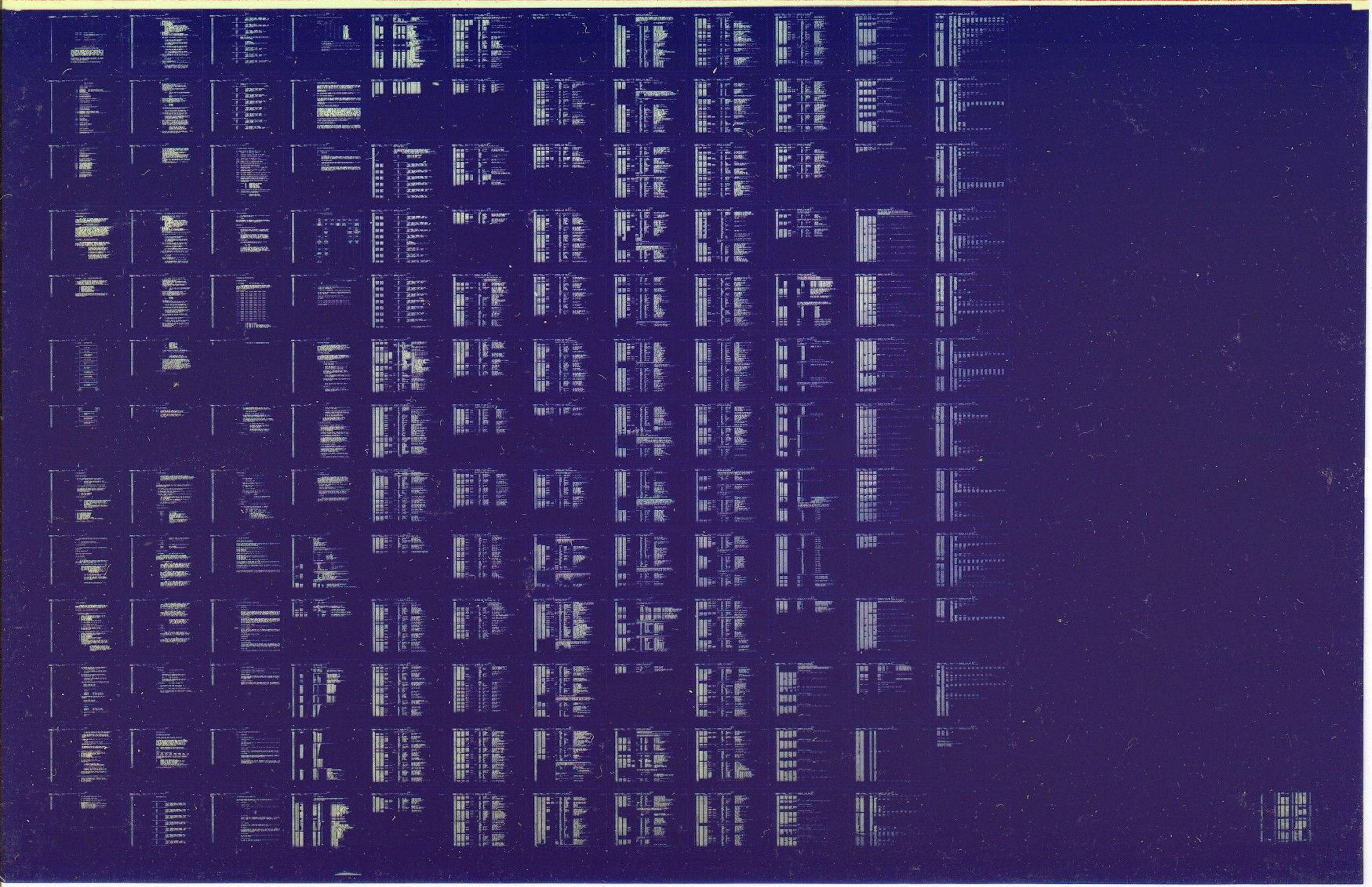
EP-DZDHN-C-DL-B

FICHE 1 OF 1

JUN 1977

digital

MADE IN USA



```
.REM @
```

PRODUCT CODE: MAINDEC-11-DZDHN-C-D

PRODUCT NAME: DH11 DATA RELIABILITY TESTS / SINGLE LINE ECHO AND PATTERNS/CABLE TESTS

DATE: 01-MAY-1977

MAINTAINER: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

1.0 GENERAL PROGRAM DESCRIPTION

1.1 PROGRAM PURPOSE

"MD-11-DZDHN" 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 SINGLE 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.

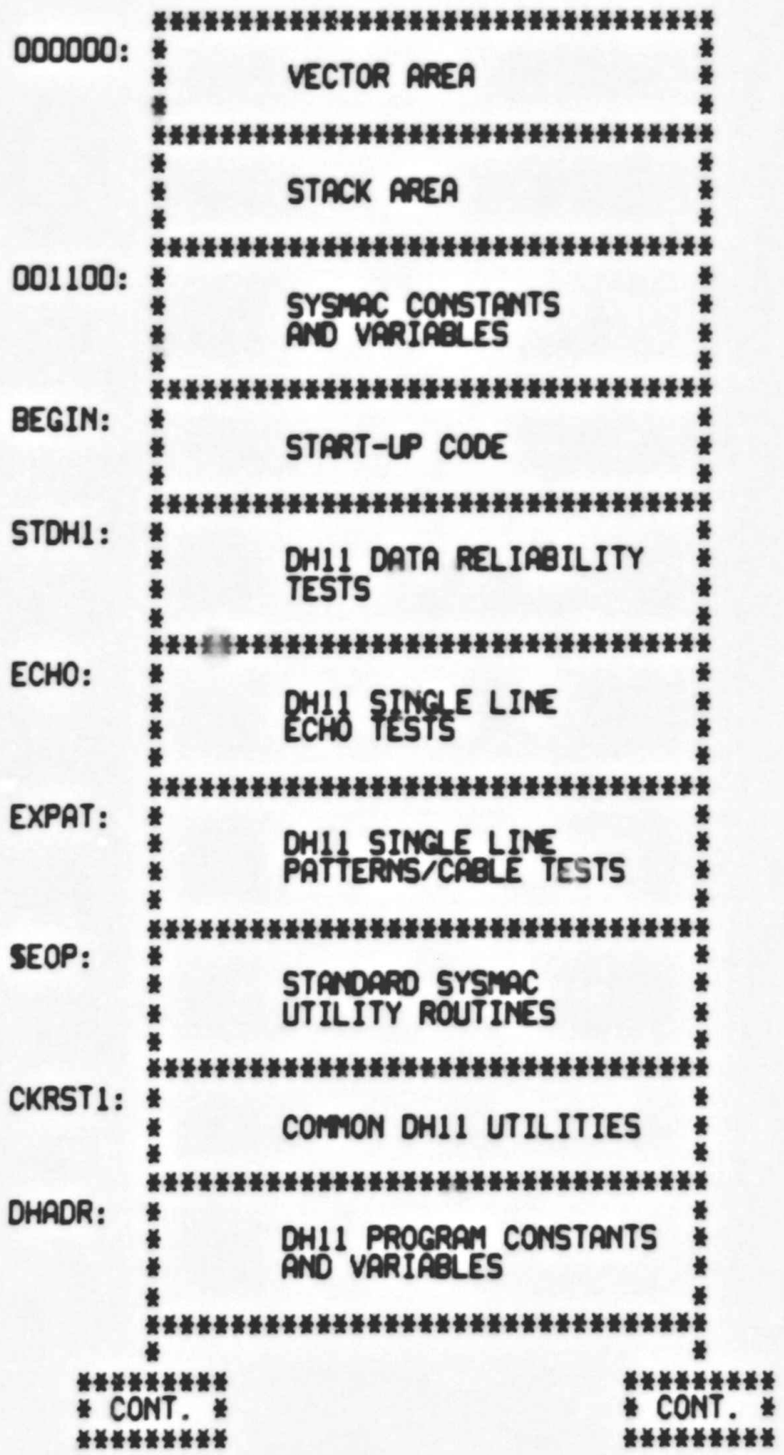
1.1.3 SUBPROGRAM 3 DH11 DATA PATTERNS/CABLE TESTS

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 MD-11-DZDHN 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

MD-11-DZDHN 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

MD-11-DZDHN 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 POP11 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 MD-11-DZDHN.
2. BOOT THE SYSTEM TO LOAD THE MONITOR
3. ONCE LOADED THE "XXDP" MONITOR PRINTS AN INTRODUCTORY MESSAGE AND RESPONDS WITH A "."
4. TYPE: "DZDHNB" 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
```

```
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
```

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.

3.0 ERROR INFORMATION

3.1 ERROR REPORTING PROCEDURES

3.1.1 STANDARD SYSMAC.SML ERROR REPORTING CONVENTIONS

THE PROGRAM UTILIZES THE STANDARD PDP11 DIAGNOSTICS ERROR UTILITIES. THE TEST ROUTINE CALLS THESE UTILITIES USING AN "ERROR N" INSTRUCTION (CODED EMT) WHERE "N" IS THE NUMBER OF THE ERROR MESSAGE. THE UTILITY ROUTINE USES "N" TO ACCESS THE PROPER ERROR INFORMATION VIA THE ERROR TABLE DESCRIBED IN SECTION 3.1.2 BELOW. EACH MESSAGE RESULTS IN THREE LINES OF TYPEOUT AS FOLLOWS:

LINE 1 A BRIEF DESCRIPTION OF THE FAILING FUNCTION
LINE 2 LABELS TO IDENTIFY THE DATA TYPED ON LINE 3
LINE 3 THE ACTUAL ERROR DATA (UP TO 8 OCTAL OR DECIMAL NO.S)

EXAMPLE:

```
SYSTEM CONTROL REGISTER ERROR
  (PC)    (PS)    (SP)    TEST   DEVADR  REGADR   WAS     S/B
002720  000002  001074  000003  160020  160020  000000  000001
```

THE ERROR TABLE ITEMS SHOWN IN THE NEXT SECTION DESCRIBE ALL THE ERROR MESSAGES WITHIN MD-11-DZDHN AND ARE INTERPRETED AS FOLLOWS:

EM ADDRESS OF THE MESSAGE FOR LINE 1
DH ADDRESS OF THE DATA HEADER MESSAGE FOR LINE 2
DT ADDRESS OF THE TABLE OF ADDRESSES THAT POINT TO THE DATA WORDS TO BE PRINTED
DF ADDRESS THAT POINTS TO THE DATA DESCRIPTOR TABLE THAT DEFINES WHETHER AN ITEM IS OCTAL OR DECIMAL. IF THIS ENTRY IS "0" ALL DATA WORDS ARE IN OCTAL.

SECTION 3.1.3 DEFINES THE MEANING OF THE MNEUMONICS USED 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             ;"SERRPC,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             ;"SERRPC,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             ;"SERRPC,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             ;"SERRPC,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             ;"SERRPC,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             ;"SERRPC,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             ;"SERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
                                DF2             ;PRINT ALL OCTAL

                        ;ERROR TABLE ITEM FOR ERROR 10
```

```
                        EM10            ;"INVALID DATA IN SILO - DROPPED LINE #  "
                        DH2             ;  (PC)    CURLPR   CHAR #   WASADR   SHBADR    WAS     S/B"
                        DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
                        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 11

                        EM11            ;"DATA ERROR - LINE #  "
                        DH2             ;  (PC)    CURLPR   CHAR #   WASADR   SHBADR    WAS     S/B"
                        DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
                        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 12

                        EM12            ;"TEST TIMEOUT - DROPPED LINE #  "
                        DH3             ;"  (PC)    CURLPR   RTOTAL   XTOTAL   RDONE"
                        DT3             ;"$ERRPC,CURLPR,$TMP0,$TMP1RDONE"
                        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 13

                        NOTE:   ERROR 13 IS CALLED TO PRINT EACH LINE OF DATA IN THE
                                ERROR STATISTICS TABLE. IT PRINTS ONLY DATA WITHOUT ANY
                                MESSAGE OR DATA HEADERS.


                        0               ;NO MESSAGE
                        0               ;NO DATA HEADER
                        DT4             ;$TMP0,$TMP1,$TMP2,$TMP3,$TMP4,$TMP5,$TMP6
                        DF1             ;PRINT ALL DECIMAL

;ERROR TABLE ITEM FOR ERROR 14

                        EM14            ;"BUS ERROR TRAP TO 04"
                        DH4             ;"  (PC)     (PS)     (SP)    TRAPPC   TRAPPS"
                        DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2"
                        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 15

                        EM15            ;"RSVD INSTR TRAP TO 10"
                        DH4             ;"  (PC)     (PS)     (SP)    TRAPPC   TRAPPS"
                        DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2"
                        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 16

                        EM16            ;"SINGLE LINE ECHO TEST - INTR WAIT TIMEOUT"
                        DH5             ;"  (PC)    DEVADR    LINE     (SCR)  CURLPR   EXFLAG"
                        DT6             ;$ERRPC,$REG1,LINE,$TMP0,CURLPR,EXFLAG"
                        DF2             ;PRINT ALL OCTAL
```

```
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             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
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             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
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             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
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             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
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             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
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             ;$ERRORPC,DHADR,LINE,CURLPR,$REG0
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            ;$ERRPC,DHADR,LINE,CURLPR,$REG0,$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                                    F       IS THE TOTAL NO. OF FRAMING ERRORS IN DECIMAL
1425                                    O       IS THE TOTAL NO. OF OVERRUN ERRORS IN DECIMAL
1426
```

```
1427                    NOTES:  1.) IF A LINE WAS DROPPED DURING THE TEST DUE TO AN
1428                                UNRECOVERABLE READ OR WRITE ERROR THE MESSAGE SHOWN
1429                                BELOW WILL REPLACE THE NORMAL ERROR STATISTICS ENTRY:
1430
1431                                    "LINE #NN WAS DROPPED"
1432
1433                                    WHERE "NN" IS THE LINE NO. IN OCTAL.
1434
1435                            2.) IF THE PRINTOUT IS INVOKED BY TYPING AN "S", THE
1436                                "RTOTAL" AND "XTOTAL" ENTRIES MAY OR MAY NOT BE EQUAL
1437                                DEPENDING UPON WHEN THE PROGRAM "SAW" THE "S".
1438
1439                            3.) AFTER PRINTING THE ERROR STATISTICS TABLE, THE PROGRAM
1440                                WILL RESTART AND BEGIN TESTING THE NEXT DH11 IN SEQUENCE.
1441                                IF ONLY ONE DH11 IS SELECTED FOR TEST OR SR14=1, THE SAME DH11
1442                                WILL BE TESTED AGAIN.
1443
1444                    B. SUB-PROGRAM 2        DH11 SINGLE LINE ECHO TESTS
1445
1446                            1) SEND MODE:   THE DISPLAY ON THE DH11 TEST TERMINAL
1447                                            SHOULD MATCH THE BUFFER TYPED IN ON
1448                                            THE CONSOLE TERMINAL.
1449
1450                            2) ECHO MODE:   THE CHARACTERS ECHOED ON THE DH11 TEST
1451                                            TERMINAL SHOULD MATCH THE CHARACTERS
1452                                            TYPED ON THE TEST TERMINAL KEYBOARD.
1453
1454                    C. SUB-PROGRAM 3        DH11 SINGLE LINE PATTERNS/CABLE TESTS
1455
1456                            (NONE PROVIDED)
1457
```

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 #NNNNN"

   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

```
                                        12      STORAGE INTERRUPT ENABLE

                                                THIS BIT, WHEN SET, PERMITS THE SETTING OF BIT 14 TO GENERATE AN INTERRU

                                        13      TRANSMITTER AND NON-EX-MEM INTERRUPT ENABLE

                                                THIS BIT, WHEN SET, PERMITS THE SETTING OF BIT 10 OR 15 TO GENERATE AN I

                                        14      STORAGE INTERRUPT

                                                THIS BIT IS SET WHEN THE RECEIVER SCANNER FINDS A RECEIVER HOLDING BUFFE
                                                STORE THAT CHARACTER IN THE SILO, AND CANNOT DO SO BECAUSE OF A LACK OF
                                                AN INTERRUPT REQUEST IF BIT 12 IS SET. THIS BIT IS READ ONLY, EXCEPT IN
                                                IT IS READ/WRITE.

                                        15      TRANSMITTER INTERRUPT

                                                THIS BIT IS SET WHEN THE DH11 CONCLUDES AN NPR CYCLE THAT INCREMENTED A
                                                CHARACTER IN A MESSAGE BUFFER WAS LOADED INTO A UART TRANSMITTER HOLDING
                                                REQUEST IF BIT 13 IS SET.  THIS BIT IS READ/WRITE. (IT IS SET DURING AN
```

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
08-11	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 XO4

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	BIT 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.

```
1720                    SPEED TABLE FOR RECEIVER AND TRANSMITTER SPEEDS:
1721
1722                                            BIT
1723
1724                    TRANSMITTER     13      12      11      10
1725
1726                    RECEIVER        9       8       7       6
1727                                    --      --      --      --
1728
1729                                    0       0       0       0       ZERO BAUD
1730                                    0       0       0       1       50 BAUDS
1731                                    0       0       1       0       75 BAUDS
1732                                    0       0       1       1       110 BAUDS
1733                                    0       1       0       0       134.5 BAUDS
1734                                    0       1       0       1       150 BAUDS
1735                                    0       1       1       0       200 BAUDS
1736                                    0       1       1       1       300 BAUDS
1737                                    1       0       0       0       600 BAUDS
1738                                    1       0       0       1       1200 BAUDS
1739                                    1       0       1       0       1800 BAUDS
1740                                    1       0       1       1       2400 BAUDS
1741                                    1       1       0       0       4800 BAUDS
1742                                    1       1       0       1       9600 BAUDS
1743                                    1       1       1       0       EXTERNAL INPUT A
1744                                    1       1       1       1       EXTERNAL INPUT B
1745
1746            14      HALF DUPLEX/FULL DUPLEX
1747
1748                    IF THIS BIT IS SET, THIS LINE WILL OPERATE IN HALF-DUPLEX MODE. IF NOT S
1749                    IN FULL-DUPLEX MODE.
1750
1751                    IN THIS APPLICATION HALF-DUPLEX MEANS THAT THE DH11 RECEIVER IS BLINDED
1752
1753            15      AUTO-ECHO ENABLE
1754
1755                    WHEN THIS BIT IS SET, CHARACTERS RECEIVED ON THIS LINE WILL BE HARDWARE
1756                    FURTHER DETAILS.
1757
```

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
	CABLE						
ROW A	UNIBUS CONNECTOR (NOTE #3)	NPR CNTL	REG & BYTE CNT	CURRENT ADDRS & ADDRS	SYSTEM CNTL & 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)						

```
1894                    !       !       !       !       !       !       !       !
1895                    ----------------------------------------------------------
1896
1897                                    FIGURE 2-4  DH11 MODULE UTILIZATION DIAGRAM
1898                                                        PAGE 2
1899
1900
1901            NOTES:
1902
1903                 1.  IF END OF BUS, REPLACE M920 WITH M930.
1904
1905                 2.  IF LAST UNIT IN BASIC BOX, REPLACE M920 WITH BC11A CABLE WHEN
1906                     EXPANDING TO PERIPHERAL BOX.
1907
1908                 3.  IF FIRST UNIT IN EXPANDER BOX, REPLACE M920 WITH BC11A CABLE.
1909
1910                 4.  E02 MUST BE G727 GRANT CONTINUITY IF MODEM CONTROL MODULE SET
1911                     IS  NOT  INSTALLED.   * DENOTES DM11-BB MODEM CONTROL OPTION,
1912                     WITH DH11-AA OR AC.
1913
1914                 5.  MODULE SLOTS PROVIDE FOR ADDITIONAL CLOCK RATES.
1915
1916                 6.  FOR DIAGNOSTIC CHECKOUT OF DH11-AA, AB, OR AC, REPLACES  M971
1917                     WITH M974.
1918
1919                 7.  THIS SLOT CONTAINS MODEM CONTROL MODULE M7807 WITH DH11-AD.
1920
1921                 8.  THIS SLOT CONTAINS MODEM CONTROL MODULE M7808 WITH DH11-AD.
1922
1923                 9.  THIS SLOT CONTAINS EIA CONVERTER AND  PRIORITY  MODULE  M5906
1924                     FOR DH11-AD OR AE.
1925
```

6.0 MAINTENANCE PROCEDURES

THIS SECTION OUTLINES SOME GENERAL TECHNIQUES FOR USING MD-11-DZDHN 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 MAINTENANCE 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 MD-DZDHN 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 MD-11-DZDHN 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  MAINDEC-11-DZDHN-C
                        ;*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
                                $SVPC=.                 ;SAVE PC
                                .=46
                                120000                  ;;1)SET LOC.46 TO ADDRESS OF 120000
                                .=52
                                .WORD   0               ;;2)SET LOC.52 TO ZERO
                                .=$SVPC                 ;; RESTORE PC
                        .SBTTL  APT PARAMETER BLOCK

                        ;;*****************************************************************
                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                        ;;*****************************************************************
                                .$X=.   ;;SAVE CURRENT LOCATION
                                .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
                                200     ;;FOR APT START UP
                                .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
                                $APTHDR ;;POINT TO APT HEADER BLOCK
                                .=.$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
```

```
                                        .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
                                .SBTTL  TRAP CATCHER

                                        .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                                        .=174
000174  000000                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  016166                  JMP     @#INPARX        ;;JUMP TO STARTING ADDRESS OF PROGRAM

000204  000137  001626                  JMP     @#BEGIN         ;BEGIN EXECUTION WITH DEFAULT PARAMETERS
000210  000137  016200                  JMP     @#INPARC        ;INPUT PARAMETERS - DEVICE SELECTION ONLY
000214  000137  004706                  JMP     @#ECHO          ;GO START LINE ECHO TESTS
000220  000137  006106                  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  000000                $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
 2371  001330  000000                $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
 2372  001332  000000                $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
 2373  001334  000000                $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
 2374  001336  000000                $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
 2375  001340  000000                $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
 2376  001342  000000                $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
 2377  001344  000000                $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
 2378  001346  000000                $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
 2379  001350  000000                $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
 2380  001352  000000                $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
 2381  001354  000000                $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
 2382
 2383
 2384  001356                        $ETEND:
 2385
```

```
                                .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

001356                          $ERRTB:

                                ;ERROR TABLE ITEM FOR ERROR 1

001356  022442                          EM1             ;"NON EX MEMORY ERROR - DROPPED LINE #  "
001360  022511                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001362  022566                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001364  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 2

001366  022614                          EM2             ;"TRANSMITTER FALSE INTERRUPT - DROPPED LINE#  "
001370  022511                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001372  022566                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001374  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 3

001376  022673                          EM3             ;"BUFFER ACTIVE REGISTER ERROR - DROPPED LINE #  "
001400  022511                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001402  022566                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001404  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 4

001406  022753                          EM4             ;"BYTE COUNT REGISTER ERROR - DROPPED LINE #  "
001410  022511                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001412  022566                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001414  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 5

001416  023030                          EM5             ;"CURRENT ADDRESS REGISTER ERROR - DROPPED LINE #  "
001420  022511                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001422  022566                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001424  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 6

001426  023112                          EM6             ;"SILO OVERFLOW ERROR - DROPPED LINE #  "
001430  022511                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001432  022566                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
```

```
2442  001434  022604                  DF2             ;PRINT ALL OCTAL
2443
2444                                ;ERROR TABLE ITEM FOR ERROR 7
2445
2446  001436  023161                  EM7             ;"RECEIVER FALSE INTERRUPT - LINE #  "
2447  001440  022511                  DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
2448  001442  022566                  DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
2449  001444  022604                  DF2             ;PRINT ALL OCTAL
2450
2451                                ;ERROR TABLE ITEM FOR ERROR 10
2452
2453  001446  023235                  EM10            ;"INVALID DATA IN SILO - DROPPED LINE #  "
2454  001450  023305                  DH2             ; (PC)   CURLPR  CHAR #  WASADR  SHBADR   WAS     S/B"
2455  001452  023372                  DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
2456  001454  022604                  DF2             ;PRINT ALL OCTAL
2457
2458                                ;ERROR TABLE ITEM FOR ERROR 11
2459
2460  001456  023412                  EM11            ;"DATA ERROR - LINE #  "
2461  001460  023305                  DH2             ; (PC)   CURLPR  CHAR #  WASADR  SHBADR   WAS     S/B"
2462  001462  023372                  DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
2463  001464  022604                  DF2             ;PRINT ALL OCTAL
2464
2465                                ;ERROR TABLE ITEM FOR ERROR 12
2466
2467  001466  023440                  EM12            ;"TEST TIMEOUT - DROPPED LINE #  "
2468  001470  023500                  DH3             ;" (PC)   CURLPR  RTOTAL  XTOTAL  RDONE"
2469  001472  023546                  DT3             ;"$ERRPC,CURLPR,$TMP0,$TMP1RDONE"
2470  001474  022604                  DF2             ;PRINT ALL OCTAL
2471
2472                                ;ERROR TABLE ITEM FOR ERROR 13
2473
2474  001476  000000                  0               ;NO MESSAGE
2475  001500  000000                  0               ;NO DATA HEADER
2476  001502  023562                  DT4             ;$TMP0,$TMP1,$TMP2,$TMP3,$TMP4,$TMP5,$TMP6
2477  001504  023602                  DF1             ;PRINT ALL DECIMAL
2478
2479                                ;ERROR TABLE ITEM FOR ERROR 14
2480
2481  001506  023612                  EM14            ;"BUS ERROR TRAP TO 04"
2482  001510  023637                  DH4             ;" (PC)    (PS)    (SP)   TRAPPC  TRAPPS"
2483  001512  023706                  DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2"
2484  001514  022604                  DF2             ;PRINT ALL OCTAL
2485
2486                                ;ERROR TABLE ITEM FOR ERROR 15
2487
2488  001516  023722                  EM15            ;"RSVD INSTR TRAP TO 10"
2489  001520  023637                  DH4             ;" (PC)    (PS)    (SP)   TRAPPC  TRAPPS"
2490  001522  023706                  DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2"
2491  001524  022604                  DF2             ;PRINT ALL OCTAL
2492
2493                                ;ERROR TABLE ITEM FOR ERROR 16
2494
2495  001526  023750                  EM16            ;"SINGLE LINE ECHO TEST - INTR WAIT TIMEOUT"
2496  001530  024022                  DH5             ;" (PC)   DEVADR   LINE    (SCR)  CURLPR  EXFLAG"
2497  001532  024102                  DT6             ;$ERRPC,$REG1,LINE,$TMP0,CURLPR,EXFLAG"
```

```
001534  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 17

001536  024120                          EM17            ;"ALTERNATING 1/0 PATTERN TEST DONE"
001540  024162                          DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001542  024232                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
001544  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 20

001546  024246                          EM20            ;"BINARY UP COUNT PATTERN TEST DONE"
001550  024162                          DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001552  024232                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
001554  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 21

001556  024310                          EM21            ;"BINARY DOWN COUNT PATTERN TEST DONE"
001560  024162                          DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001562  024232                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
001564  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 22

001566  024354                          EM22            ;"RANDOM DATA PATTERN TEST DONE"
001570  024162                          DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001572  024232                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
001574  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 23

001576  024412                          EM23            ;SINGLE CHAR PATTERN TEST DONE"
001600  024162                          DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001602  024232                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
001604  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 24

001606  024450                          EM24            ;"TYPED BUFFER PATTERN TEST DONE"
001610  024162                          DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001612  024232                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
001614  022604                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 25

001616  024507                          EM25            ;"DATA PATTERNS TEST TIMEOUT"
001620  024542                          DH7             ;" (PC) DEVADR   LINE   CURLPR  ICOUNT  PATCDE"
001622  024622                          DT10            ;$ERRPC,DHADR,LINE,CURLPR,$REGO,$REG1
001624  022604                          DF2             ;PRINT ALL OCTAL


001626  005000          BEGIN:  CLR     R0              ;INIT R0 TO INDICATE DEFAULT PARAMETERS
001630  005067  020060          CLR     VCFLG           ;INIT VECTOR ADDR SET UP FLAG
```

```
2554  001634  005067  020506          CLR     DPFLG           ;CLEAR DATA PATTERNS TEST FLAG
2555  001640  005067  020514          CLR     RETFLG          ;CLEAR ECHO TEST RETURN FLAG
2556  001644  005067  020466  BEGINA: CLR     TITFLG          ;INIT TITLE MESSAGE FLAG
2557                          .SBTTL  INITIALIZE THE COMMON TAGS
2558                          ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
2559  001650  012706  001100          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
2560  001654  005026                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
2561  001656  022706  001140          CMP     #SWR,R6 ;;DONE?
2562  001662  001374                  BNE     .-6             ;;LOOP BACK IF NO
2563  001664  012706  001100          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
2564                          ;;INITIALIZE A FEW VECTORS
2565  001670  012737  011112  000020  MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
2566  001676  012737  000340  000022  MOV     #340,@#IOTVEC+2 ;;LEVEL 7
2567  001704  012737  011356  000030  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
2568  001712  012737  000340  000032  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
2569  001720  012737  014326  000034  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
2570  001726  012737  000340  000036  MOV     #340,@#TRAPVEC+2;LEVEL 7
2571  001734  012737  014412  000024  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
2572  001742  012737  000340  000026  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
2573  001750  005067  177246          CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
2574  001754  005067  177244          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2575  001760  112767  000001  177127  MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
2576  001766  012767  001766  177112  MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2577  001774  012767  001774  177106  MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
2578                          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
2579                          ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
2580  002002  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
2581  002006  012737  002042  000004  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
2582  002014  012767  177570  177116  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
2583  002022  012767  177570  177112  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
2584  002030  022777  177777  177102  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
2585  002036  001012                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
2586                                                          ;;AND  THE HARDWARE SWR IS NOT = -1
2587  002040  000403                  BR      65$             ;;BRANCH IF NO TIMEOUT
2588  002042  012716  002050  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
2589  002046  000002                  RTI
2590  002050  012767  000176  177062  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
2591  002056  012767  000174  177056          MOV     #DISPREG,DISPLAY
2592  002064  012637  000004  66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
2593
2594  002070  005067  177144          CLR     $PASS           ;;CLEAR PASS COUNT
2595  002074  132767  000200  177151  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
2596  002102  001403                  BEQ     67$             ;;YES,USE NON-APT SWITCH
2597  002104  012767  001254  177026  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
2598  002112                  67$:
2599                          .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
2600  002112  005737  000042          TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
2601  002116  001012                  BNE     68$             ;;BRANCH IF YES
2602  002120  126727  177126  000001  CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
2603  002126  001406                  BEQ     68$             ;;BRANCH IF YES
2604  002130  026727  177004  000176  CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
2605  002136  001005                  BNE     69$             ;;BRANCH IF NO
2606  002140  104406                  GTSWR                   ;;GET SOFT-SWR SETTINGS
2607  002142  000403                  BR      69$
2608  002144  112767  000001  176762  68$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
2609  002152                  69$:
```

```
2610  002152  012767  016762  175624  START1: MOV     #BUSER,ERRVEC   ;SET UP THE BUS ERROR VECTOR
2611  002160  012767  000340  175620          MOV     #340,ERRVEC+2
2612  002166  012767  017024  175614          MOV     #RESERR,RESVEC  ;SET UP THE RSVD INSTR VECTOR
2613  002174  012767  000340  175610          MOV     #340,RESVEC+2
2614  002202  005767  020130                  TST     TITFLG          ;HAVE WE TYPED TITLE ONCE ?
2615  002206  001012                          BNE     1$              ;BR IF YES
2616  002210  104401                          TYPE                    ;GO TYPE PROGRAM TITLE
2617  002212  024640                          TITLE
2618  002214  005167  020116                  COM     TITFLG          ;SET FLAG - TYPE TITLE ONLY ONCE PER LOAD
2619  002220  032777  000001  176712          BIT     #BIT0,@SWR      ;DO WE WANT TO AUTOSIZE?
2620  002226  001002                          BNE     1$              ;BRANCH IF NOT.
2621  002230  004767  012716                  JSR     PC,AUTOSZ       ;GO AUTOSIZE.
2622  002234  005767  017454          1$:     TST     VCFLG           ;START AT 200 ??
2623  002240  001413                          BEQ     13$             ;BR IF NOT
2624  002242  032777  000001  176670          BIT     #BIT0,@SWR      ;ARE PARAMETERS TO BE INPUT MANUALLY?
2625  002250  001003                          BNE     9$              ;BRANCH IF YES
2626  002252  016700  017664                  MOV     ADRVEC,R0       ;OTHERWISE, GET ADDRESSES BETWEEN VECOTRS FROM AUTOSIZER
2627  002256  000402                          BR      10$
2628  002260  004767  013634          9$:     JSR     PC,INPARA       ;GO ASK FOR PARAMETERS
2629  002264  005067  017424          10$:    CLR     VCFLG           ;RE INIT START FLAG
2630  002270  005767  020064          13$:    TST     RETFLG          ;RETURN TO ECHO TESTS ?
2631  002274  001402                          BEQ     11$             ;BR IF NOT
2632  002276  000167  002426                  JMP     ECHO1           ;RETURN TO ECHO TEST START-UP
2633  002302  005767  020040          11$:    TST     DPFLG           ;RETURN TO DATA PATTERNS TEST ?
2634  002306  001402                          BEQ     12$             ;BR IF NOT
2635  002310  000167  003614                  JMP     EXPAT1          ;GO BACK TO DATA PATTERNS TESTS
2636  002314  005700                  12$:    TST     R0              ;USE DEFAULT PARAMETERS ?
2637  002316  001407                          BEQ     START2          ;BR IF YES
2638  002320  022700  177777                  CMP     #-1,R0          ;CHANGE DH SELECT PARAM ONLY ?
2639  002324  001002                          BNE     2$              ;BR IF NOT
2640  002326  000167  013762                  JMP     INPAR3          ;GO ASK FOR SELECT PARAM.
2641  002332  000167  013666          2$:     JMP     INPAR           ;GO ASK FOR ALL PARAMETERS
2642
2643  002336  012767  021612  017764  START2: MOV     #DHADTB-2,ADPTR ;GET POINTER TO ADDRESS TABLE
2644  002344  012767  021652  017760          MOV     #DHVCTB-2,VCPTR ;GET POINTER TO VECTOR TABLE
2645  002352  012767  021714  017754          MOV     #BRLVL-2,BRPTR  ;GET POINTER TO BR LEVEL TABLE
2646  002360  012767  177777  017562          MOV     #-1,DHNUM       ;START WITH DH #00
2647  002366  012767  000001  017132          MOV     #1,SELMSK       ;SET UP DH11 BIT TEST MARKER
2648
2649  002374  005267  017550          RESTRT: INC     DHNUM           ;GENERATE DH11 DEV NUMBER
2650  002400  062767  000002  017722          ADD     #2,ADPTR        ;UPDATE TABLE POINTERS
2651  002406  062767  000002  017716          ADD     #2,VCPTR
2652  002414  062767  000002  017712          ADD     #2,BRPTR
2653  002422  036767  017100  017100          BIT     SELMSK,DHSEL    ;TEST FOR SELECTED DH11
2654  002430  001004                          BNE     RSTRTA          ;BR IF SELECTED FOR TEST
2655  002432  006367  017070          REST1:  ASL     SELMSK          ;SHIFT MARKER TO TEST NEXT DH11
2656  002436  001737                          BEQ     START2          ;BR IF 16 TESTED - START OVER
2657  002440  000755                          BR      RESTRT          ;GO TEST IF THIS ONE SELECTED
2658  002442  017767  017662  017052  RSTRTA: MOV     @ADPTR,DHADR    ;SET UP DH11 ADDRESS
2659  002450  017767  017656  017046          MOV     @VCPTR,DHVCT    ;SET UP THE DH11 VECTOR ENTRY
2660  002456  017767  017652  017462          MOV     @BRPTR,DHRLVL   ;GET BR LEVEL VALUES
2661  002464  004567  012354                  JSR     R5,SUNUM        ;GO SET DH NUMBER IN THE MESSAGE BUFFER
2662  002470  022150                          DHNUM
2663  002472  024742                          TITLE2+20
2664  002474  104401                          TYPE                    ;GO PRINT "TESTING DH11 #XX"
2665  002476  024722                          TITLE2
```

```
2666  002500  012767  002500  176400          MOV     #.,$LPADR       ;INIT SCOPE RETURN
2667                                ;;**************************************************************
2668                                ;*TEST 1         SUB-PROGRAM 1 - DATA RELIABILITY TESTS
2669                                ;;**************************************************************
2670  002506  000004               TST1:   SCOPE
2671  002510  012767  000001  176504          MOV     #1,$TIMES       ;;DO 1 ITERATION
2672  002516  004767  014414        STDH1:  JSR     PC,CLSTAT       ;GO CLEAR THE STATISTICS TABLES
2673  002522  004767  014774                JSR     PC,KYBD1        ;GO SET UP FOR KEYBOARD INTRS.
2674  002526  005067  017006                CLR     QUICK           ;INIT THE QUICK TEST FLAG
2675  002532  005067  017000                CLR     DRPLIN          ;INIT DROPPED LINE FLAGS
2676  002536  005067  017410                CLR     LINE            ;INIT LINE NO. TO DO
2677  002542  016702  016756                MOV     DHVCT,R2        ;SET UP THE VECTORS
2678  002546  012722  003550                MOV     #RINT1,(R2)+    ;GO TO RINT1 ON RCVR INTR
2679  002552  116722  017370                MOVB    DHRLVL,(R2)+    ;
2680  002556  105722                       TSTB    (R2)+
2681  002560  012722  003044                MOV     #TINT1,(R2)+    ;GO TO TINT1 ON XMITTR INTR
2682  002564  116712  017357                MOVB    DHTLVL,(R2)
2683  002570  016701  016726                MOV     DHADR,R1        ;SET UP DEVICE ADDRESS
2684
2685  002574  004767  012152        NEWLIN: JSR     PC,SELINE       ;GO SELECT NEW LINE FOR TEST
2686  002600  000401                       BR      1$              ;BR IF TESTED ALL SELECTED LINES
2687  002602  000402                       BR      2$              ;BR IF NOT DONE
2688  002604  000167  001616        1$:     JMP     PRSTAT          ;GO CHECK AND REPORT STATISTICS
2689  002610  004567  012230        2$:     JSR     R5,SUNUM        ;PUT LINE NO. IN MSG
2690  002614  022152                       LINE
2691  002616  025602                       STMSG2+20
2692  002620  104401                       TYPE                    ;TYPE LINE NO. BEING TESTED
2693  002622  025562                       STMSG2
2694  002624  012767  021544  016746        MOV     #LPRTAB,LPRPTR  ;SET UP LPR TABLE POINTER
2695
2696  002632  004567  014316        NEWLPR: JSR     R5,SETLPR       ;GO SET UP LPR CONSTANT
2697  002636  000756                       BR      NEWLIN          ;BR IF DONE ALL BAUD RATES AT THIS LINE
2698  002640  012767  177760  016734        MOV     #177760,CHRCNT  ;INIT CHAR COUNT
2699  002646  012767  177777  016730        MOV     #-1,CLSEL       ;INIT CHR LNGTH SELECT CODE
2700
2701  002654  005067  016662                CLR     QUICKX          ;INIT QUICK TEST EXIT FLAG
2702  002660  004567  014354        NEWCL:  JSR     R5,SETCL        ;GO SET UP THE CHAR LENGTH
2703  002664  000762                       BR      NEWLPR          ;BR IF DONE ALL FOUR LENGTHS
2704  002666  005067  016714                CLR     PARBIT          ;INIT PARITY SELECT BIT CODE
2705
2706  002672  004567  014502        NEWPAR: JSR     R5,SETPAR       ;GO SET PARITY SELECT
2707  002676  000770                       BR      NEWCL           ;BR IF DONE ALL COMBOS
2708
2709  002700  004767  011750        DHST1:  JSR     PC,DHSET1       ;GO SET UP FOR TESTING THIS LINE
2710  002704  056761  016624  000012        BIS     LINMSK,BAR(R1)  ;ACTIVATE THE SELECTED LINE
2711  002712  004767  016510                JSR     PC,CHPS1        ;GO CLEAR PSW
2712
2713
2714  002716  012767  000200  017414        MOV     #200,TIMEA      ;INIT TIMER A
2715  002724  005067  017412                CLR     TIMEB           ;INIT TIMER B
2716  002730  022767  000003  016652  1$:     CMP     #3,RDONE        ;BOTH RCVR AND XMITTR DONE ?
2717  002736  001435                       BEQ     3$              ;BR IF YES
2718  002740  004767  014150                JSR     PC,TIMEIT       ;CALL THE TIMER
2719  002744  000771                       BR      1$              ;TIMER STEPS AROUND THIS BRANCH IF
2720                                                                ;TIMEOUT OCCURS
2721
```

```
 2722  002746  052777  004000  016546          BIS     #BIT11,@DHADR   ;CLEAR OUT THE DH11
 2723  002754  116700  017172                  MOVB    LINE,R0         ;GET LINE NO.
 2724  002760  006300                          ASL     R0              ;FORM TABLE INDEX
 2725  002762  016067  030202  176212          MOV     RTOTAL(R0),$TMP0;SAVE XMITTED COUNT
 2726  002770  016067  030242  176206          MOV     XTOTAL(R0),$TMP1;SAVE THE RCVD COUNT
 2727  002776  004567  012042                  JSR     R5,SUNUM        ;PUT LINE NO. IN MESSAGE
 2728  003002  022152                          LINE
 2729  003004  023475                          EM12+35
 2730  003006  012767  003016  176074          MOV     #2$,$LPERR      ;SET UP ERROR LOOP RETURN
 2731  003014  104012                          ERROR   12              ;LINE FAILED TO FINISH ON TIME - HUNG
 2732  003016  056767  016512  016512  2$:     BIS     LINMSK,DRPLIN   ;SET DROP FLAG
 2733  003024  012706  001100                  MOV     #STACK,SP       ;RESET STACK POINTER
 2734  003030  000661                          BR      NEWLIN          ;GO TRY ANOTHER LINE
 2735
 2736
 2737  003032  012711  004000          3$:     MOV     #BIT11,(R1)     ;CLEAR THE WORLD OUT IN THE DH11
 2738  003036  004767  001050                  JSR     PC,CKER         ;GO UPDATE THE DATA ERROR TABLES
 2739  003042  000713                          BR      NEWPAR          ;GO TRY NEXT PARITY COMBINATION
```

```
2740
2741                                    ;TRANSMITTER INTERRUPT SERVICE ROUTINE ONE
2742
2743  003044  032711  002000    TINT1:  BIT     #BIT10,(R1)     ;NON EX MEM ERROR ??
2744  003050  001432                    BEQ     2$              ;BR IF NOT
2745
2746  003052  011103                    MOV     (R1),R3         ;SAVE THE SCR
2747  003054  004767  016362            JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2748  003060  012711  004000            MOV     #BIT11,(R1)     ;CLEAR OUT THE DH11
2749  003064  116704  017062            MOVB    LINE,R4         ;SET UP THE S/B DATA
2750  003070  042703  175760            BIC     #175760,R3      ;CLEAR OUT SUPERFLUOUS BITS
2751  003074  010102                    MOV     R1,R2           ;SET UP REGADR
2752  003076  004767  012026            JSR     PC,SUER1        ;GO SET UP ERROR INFO
2753  003102  004567  011736            JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2754  003106  022152                    LINE
2755  003110  022506                    EM1+44
2756  003112  012767  003122  175770    MOV     #1$,SLPERR      ;SET UP THE ERROR LOOP RETURN
2757  003120  104001                    ERROR   1               ;NON EX MEM ERROR
2758  003122  022626            1$:     CMP     (SP)+,(SP)+     ;POP THE STACK
2759  003124  056767  016404  016404    BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2760  003132  000167  177436            JMP     NEWLIN          ;GO TRY NEXT LINE
2761
2762  003136  011103            2$:     MOV     (R1),R3         ;GET THE SCR REG CONTENTS
2763  003140  100433                    BMI     4$              ;BR IF XMIT DONE SET
2764
2765  003142  004767  016274            JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2766  003146  012711  004000            MOV     #BIT11,(R1)     ;CLEAR THE DH11 - FATAL ERROR
2767  003152  012704  100000            MOV     #BIT15,R4       ;SET UP S/B DATA
2768  003156  156704  016770            BISB    LINE,R4
2769  003162  042703  077760            BIC     #77760,R3       ;CLEAR OUT SUPERFLUOUS BITS
2770  003166  010102                    MOV     R1,R2           ;SET UP REGADR
2771  003170  004767  011734            JSR     PC,SUER1        ;GO SET UP ERROR INFO
2772  003174  004567  011644            JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2773  003200  022152                    LINE
2774  003202  022670                    EM2+54
2775  003204  012767  003214  175676    MOV     #3$,SLPERR      ;SET UP ERROR LOOP RETURN
2776  003212  104002                    ERROR   2               ;XMITTR FALSE INTERRUPT
2777  003214  022626            3$:     CMP     (SP)+,(SP)+     ;POP THE STACK
2778  003216  056767  016312  016312    BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2779  003224  000167  177344            JMP     NEWLIN          ;GO TRY NEXT LINE
2780
2781  003230  005761  000012    4$:     TST     BAR(R1)         ;DID BAR BIT CLEAR ??
2782  003234  001432                    BEQ     6$              ;BR IF YES
2783
2784  003236  004767  016200            JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2785  003242  016103  000012            MOV     BAR(R1),R3      ;GET THE WAS DATA
2786  003246  012711  004000            MOV     #BIT11,(R1)     ;CLEAR THE DH11
2787  003252  005004                    CLR     R4              ;SET UP S/B DATA
2788  003254  010102                    MOV     R1,R2           ;SET UP REGADR
2789  003256  062702  000012            ADD     #BAR,R2
2790  003262  004767  011642            JSR     PC,SUER1        ;GO SET UP ERROR INFO
2791  003266  004567  011552            JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2792  003272  022152                    LINE
2793  003274  022750                    EM3+55
2794  003276  012767  003306  175604    MOV     #5$,SLPERR      ;SAVE THE ERROR LOOP RETURN
2795  003304  104003                    ERROR   3               ;BUFFER ACTIVE REG FAILED TO CLEAR
```

```
2796  003306  022626                  5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
2797  003310  056767  016220  016220          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2798  003316  000167  177252                  JMP     NEWLIN          ;GO TRY NEXT LINE
2799
2800  003322  005761  000010          6$:     TST     BCR(R1)         ;DID BYTE COUNT GO TO ZERO ??
2801  003326  001432                  BEQ     8$              ;BR IF YES
2802
2803  003330  004767  016106                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2804  003334  016103  000010                  MOV     BCR(R1),R3      ;GET THE WAS DATA
2805  003340  012711  004000                  MOV     #BIT11,(R1)     ;CLEAR THE DH11
2806  003344  005004                  CLR     R4              ;SET UP S/B DATA
2807  003346  010102                  MOV     R1,R2           ;SET UP REGADR
2808  003350  062702  000010                  ADD     #BCR,R2
2809  003354  004767  011550                  JSR     PC,SUER1        ;GO SET UP THE ERROR INFO
2810  003360  004567  011460                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2811  003364  022152                  LINE
2812  003366  023025                  EM4+52
2813  003370  012767  003400  175512          MOV     #7$,$LPERR      ;SET UP ERROR LOOP RETURN
2814  003376  104004                  ERROR   4               ;BYTE COUNT REG FAILED TO GO TO 000000
2815  003400  022626          7$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
2816  003402  056767  016126  016126          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2817  003410  000167  177160                  JMP     NEWLIN          ;GO TRY NEXT LINE
2818
2819  003414  016103  000006          8$:     MOV     CAR(R1),R3      ;GET THE WAS DATA
2820  003420  016704  016156                  MOV     CHRCNT,R4       ;SET UP S/B DATA
2821  003424  005404                  NEG     R4
2822  003426  062704  032776                  ADD     #TBUF,R4
2823  003432  020304                  CMP     R3,R4           ;WAS CAR CORRECT ??
2824  003434  001425                  BEQ     10$             ;BR IF YES
2825
2826  003436  004767  016000                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
2827  003442  010102                  MOV     R1,R2           ;SET UP REGADR
2828  003444  062702  000006                  ADD     #CAR,R2
2829  003450  004767  011454                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
2830  003454  004567  011364                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
2831  003460  022152                  LINE
2832  003462  023107                  EM5+57
2833  003464  012767  003474  175416          MOV     #9$,$LPERR      ;SET UP THE ERROR RETURN
2834  003472  104005                  ERROR   5               ;CURRENT ADDRESS REG NOT CORRECT
2835  003474  022626          9$:     CMP     (SP)+,(SP)+     ;POP THE STACK
2836  003476  056767  016032  016032          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
2837  003504  000167  177064                  JMP     NEWLIN          ;GO TRY NEXT LINE
2838
2839  003510                          10$:
2840  003510  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2841  003512  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2842  003514  016703  016062                  MOV     CHRCNT,R3
2843  003520  005403                  NEG     R3              ;CHAR COUNT IN R3
2844  003522  116704  016424                  MOVB    LINE,R4         ;GET LINE NO.
2845  003526  006304                  ASL     R4              ;DOUBLE IT
2846  003530  060364  030242                  ADD     R3,XTOTAL(R4)   ;UPDATE TOTAL XMIT COUNT
2847  003534  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
2848  003536  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
2849  003540  052767  000001  016042          BIS     #BIT0,RDONE     ;SET XMIT DONE FLAG
2850  003546  000002                  RTI                     ;RETURN TO WAIT LOOP
```

```
                                        ;RECEIVER INTERRUPT SERVICE ROUTINE ONE
                                        ;
003550  032711  040000        RINT1:  BIT     #BIT14,(R1)     ;SILO OVERFLOW ERROR ??
003554  001431                 BEQ     2$              ;BR IF NOT
                                        ;
003556  004767  015660          2$:     JSR     PC,CHPS2        ;GO LOCK OUT INTRS
003562  011103                  MOV     (R1),R3         ;GET THE WAS DATA
003564  012711  004000          MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
003570  042703  177760          BIC     #177760,R3      ;CLEAR JUNK
003574  116704  016352          MOVB    LINE,R4
003600  004767  011324          JSR     PC,SUER1        ;GO SET UP ERROR INFO
003604  004567  011234          JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
003610  022152                  LINE
003612  023156                  EM6+44
003614  012767  003624  175266  MOV     #1$,$LPERR      ;SET UP ERROR LOOP RETURN
003622  104006                  ERROR   6               ;SILO OVERFLOW - BAD,BAD,BAD !!!
003624  022626          1$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
003626  056767  015702  015702  BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
003634  000167  176734          JMP     NEWLIN          ;GO TRY NEXT LINE
                                        ;
003640  105711          2$:     TSTB    (R1)            ;CHAR AVAIL SET ??
003642  100434                  BMI     4$              ;BR IF YES
                                        ;
003644  004767  015572          JSR     PC,CHPS2        ;GO LOCK OUT INTRS
003650  011103                  MOV     (R1),R3         ;GET WAS DATA
003652  042703  177560          BIC     #177560,R3      ;CLEAN IT UP
003656  012711  004000          MOV     #BIT11,(R1)     ;NOW CLEAR DH11
003662  012704  000200          MOV     #BIT07,R4       ;SET UP S/B DATA
003666  156704  016260          BISB    LINE,R4
003672  010102                  MOV     R1,R2           ;SET UP REGADR
003674  004767  011230          JSR     PC,SUER1        ;GO SET UP ERROR INFO
003700  004567  011140          JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
003704  022152                  LINE
003706  023232                  EM7+51
003710  012767  003720  175172  MOV     #3$,$LPERR      ;SET UP THE ERROR LOOP RETURN
003716  104007                  ERROR   7               ;RECEIVER FALSE INTERRUPT
003720  022626          3$:     CMP     (SP)+,(SP)+     ;POP GOES THE SP
003722  056767  015606  015606  BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
003730  000167  176640          JMP     NEWLINE         ;GO TRY NEXT LINE
                                        ;
003734  016167  000002  175240  4$:     MOV     NRC(R1),$TMPO   ;SAVE THE DATA RECEIVED
003742  100431                  BMI     6$              ;BR IF IT WAS VALID DATA
                                        ;
003744  004767  015472          JSR     PC,CHPS2        ;GO LOCK OUT INTRS
003750  012711  004000          MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
003754  162767  030516  024530  SUB     #RBUF,RBFPTR    ;WHICH CHAR WAS IT ??
003762  016702  024524          MOV     RBFPTR,R2       ;SAVE CHAR NUMBER
003766  004767  011132          JSR     PC,SUER2        ;GO SET UP ERROR INFO
003772  004567  011046          JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
003776  022152                  LINE
004000  023302                  EM10+45
004002  012767  004012  175100  MOV     #5$,$LPERR      ;SET UP ERROR RETURN
004010  104010                  ERROR   10              ;RECEIVED INVALID DATA
004012  022626          5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
004014  056767  015514  015514  BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
004022  000167  176546          JMP     NEWLIN          ;GO TRY ANOTHER LINE
```

```
2907
2908  004026                          6$:
2909  004026  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2910  004030  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2911  004032  016777  175144  024452        MOV     $TMPO,@RBFPTR   ;STORE CHAR IN THE BUFFER
2912  004040  062767  000002  024444        ADD     #2,RBFPTR       ;UPDATE THE POINTER
2913  004046  026767  015540  024436        CMP     RBFEND,RBFPTR   ;END OF BUFFER ??
2914  004054  001013                        BNE     7$              ;BR IF NOT
2915  004056  016703  015520                MOV     CHRCNT,R3       ;GET CHAR COUNT
2916  004062  005403                        NEG     R3
2917  004064  116704  016062                MOVB    LINE,R4         ;GET THE LINE NO.
2918  004070  006304                        ASL     R4              ;DOUBLE IT
2919  004072  060364  030202                ADD     R3,RTOTAL(R4)   ;UPDATE TOTAL RECEIVED COUNT
2920  004076  052767  000002  015504        BIS     #BIT1,RDONE     ;SET THE RCVR DONE BLAG
2921  004104                          7$:
2922  004104  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
2923  004106  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
2924  004110  000002                        RTI                     ;RETURN TO WAIT LOOP
```

```
                                        ;THIS ROUTINE IS CALLED TO CHECK THE RECEIVED DATA, REPORT ALL ERRORS,
                                        ;AND UPDATE THE STATISTICS TABLE ENTRIES FOR ALL LINES ACTIVE
                                        
004112 012767 030516 024372  CKER:  MOV   #RBUF,RBFPTR     ;SET UP POINTERS
004120 012767 032776 024366          MOV   #TBUF,TBFPTR
004126 012767 030302 024346          MOV   #DATERR,DEPTR    ;SET UP POINTERS TO STATISTICS TABLES
004134 012767 030342 024342          MOV   #PARERR,PEPTR
004142 012767 030402 024336          MOV   #OVRERR,ORPTR
004150 012767 030442 024332          MOV   #FRMERR,FRPTR
004156 116705 015770                 MOVB  LINE,R5          ;GET LINE NO. AND DOUBLE IT
004162 006305                        ASL   R5
004164 060567 024312                 ADD   R5,DEPTR         ;POINT TO CORRECT LINE ENTRY IN TABLE
004170 060567 024310                 ADD   R5,PEPTR
004174 060567 024306                 ADD   R5,ORPTR
004200 060567 024304                 ADD   R5,FRPTR
                                        
004204 117704 024304         1$:     MOVB  @TBFPTR,R4       ;GET THE S/B DATA
004210 000304                        SWAB  R4               ;PUT LINE NO. IN HIGH BYTE
004212 105004                        CLRB  R4
004214 156704 015732                 BISB  LINE,R4
004220 000304                        SWAB  R4
004222 052704 100000                 BIS   #BIT15,R4        ;AND FINALLY THE VALID DATA BIT
004226 017703 024260                 MOV   @RBFPTR,R3       ;GET THE WAS DATA
004232 020304                        CMP   R3,R4            ;WAS = S/B ?????
004234 001435                        BEQ   3$               ;BR IF YES
                                        
004236 010367 174756                 MOV   R3,$TMP7         ;SAVE THE WAS DATA
004242 010146                        MOV   R1,-(SP)         ;SAVE THE DEVADR
004244 016701 024242                 MOV   RBFPTR,R1        ;GET THE SBADR
004250 016702 024240                 MOV   TBFPTR,R2        ;GET THE WASADR
004254 010200                        MOV   R2,R0            ;GET XMIT BUFFER ADDR
004256 162700 032776                 SUB   #TBUF,R0         ;GENERATE CHAR #
004262 004767 000066                 JSR   PC,UPDER         ;GO CHECK AND UPDATE THE DATA ERROR TABLE
004266 004767 010632                 JSR   PC,SUER2         ;GO SET UP ERROR INFO
004272 004567 010546                 JSR   R5,SUNUM         ;GO PUT LINE NO. IN MSG
004276 022152                        LINE
004300 023435                        EM11+23
004302 012767 004312 174600          MOV   #2$,$LPERR       ;SET UP ERROR RETURN
004310 104011                        ERROR 11               ;DATA COMPARE ERROR OR PARITY,FRAMING
                                                            ;OR OVERRUN
004312 012601                2$:     MOV   (SP)+,R1         ;RESTORE THE DEVADR
004314 032767 070000 174676          BIT   #70000,$TMP7     ;ANY PARITY,OVERRUN, OR FRAMING ERROR
004322 001402                        BEQ   3$               ;BR IF NOT
                                        
004324 004767 000036                 JSR   PC,SOFT          ;GO TAKE CARE OF SOFT ERROR REPORT
                                        
004330 005267 024160         3$:     INC   TBFPTR           ;UPDATE POINTERS
004334 062767 000002 024150          ADD   #2,RBFPTR
004342 026767 015244 024142          CMP   RBFEND,RBFPTR    ;COMPARED ALL CHARS ??
004350 001315                        BNE   1$               ;BR IF NOT
                                        
004352 000207                        RTS   PC               ;RETURN TO WAIT LOOP
                                        
004354 120304                UPDER:  CMPB  R3,R4            ;DATA BYTES CORRECT ??
004356 001402                        BEQ   1$               ;BR IF YES
004360 005277 024116                 INC   @DEPTR           ;COUNT THE DATA ERROR
```

```
2981   004364   000207                1$:     RTS     PC              ;RETURN
2982
2983   004366   006367   174626       SOFT:   ASL     $TMP7           ;TEST FOR OVERRUN ERRORS
2984   004372   100002                       BPL     1$              ;BR IF NONE
2985   004374   005277   024106              INC     @ORPTR          ;COUNT IT
2986   004400   006367   174614       1$:     ASL     $TMP7           ;TEST FOR FRAMING ERRORS
2987   004404   100002                       BPL     2$              ;BR IF NONE
2988   004406   005277   024076              INC     @FRPTR          ;COUNT IT
2989   004412   006367   174602       2$:     ASL     $TMP7           ;TEST FOR PARITY ERRORS
2990   004416   100002                       BPL     3$              ;BR IF NONE
2991   004420   005277   024060              INC     @PEPTR          ;COUNT IT
2992   004424   000207               3$:     RTS     PC              ;RETURN
2993
2994
```

```
                                        ;THIS ROUTINE IS CALLED TO PRINT OUT THE TEST STATISTICS
                                        ;
004426  012767  000001  015100  PRSTAT: MOV     #1,LINMSK       ;SET UP BIT TEST MARKER
004434  005001                        CLR     R1              ;R1 CONTAINS THE LINE NO.
004436  004567  010402                JSR     R5,SUNUM        ;GO SET UP DH11 # IN STAT MESSAGE
                                        ;
004442  022150                        DHNUM
004444  025543                        STMSG1+6
004446  104401                        TYPE                    ;GO TYPE THE STATISTICS HEADER
004450  025535                        STMSG1
004452  104401                        TYPE                    ;TYPE HEADER
004454  025640                        STMSG4
                                        ;
004456  036767  015052  015052  1$:   BIT     LINMSK,DRPLIN   ;DID THIS LINE GET DROPPED ?
004464  001411                        BEQ     2$              ;BR IF NOT
                                        ;
004466  010167  174526                MOV     R1,$TMP7        ;SAVE THE LINE NO.
004472  004567  010346                JSR     R5,SUNUM        ;GO PUT LINE NO. IN MESSAGE
004476  001220                        $TMP7
004500  025617                        STMSG3+10
004502  104401                        TYPE
004504  025607                        STMSG3
004506  000436                        BR      3$              ;GO TEST NEXT LINE
                                        ;
004510  010102                2$:     MOV     R1,R2           ;SET UP R2 WITH TABLE INDEX
004512  006302                        ASL     R2
004514  036767  015014  015010        BIT     LINMSK,LINSEL   ;WAS THIS LINE SELECTED ??
004522  001430                        BEQ     3$              ;BR IF NOT
004524  010167  174452                MOV     R1,$TMP0        ;SET UP THE ERROR INFORMATION FROM
                                        ;THE TABLES INTO THE MESSAGE POINTERS
004530  016267  030202  174446        MOV     RTOTAL(R2),$TMP1
004536  016267  030242  174442        MOV     XTOTAL(R2),$TMP2
004544  016267  030302  174436        MOV     DATERR(R2),$TMP3
004552  016267  030342  174432        MOV     PARERR(R2),$TMP4
004560  016267  030402  174426        MOV     OVRERR(R2),$TMP5
004566  016267  030442  174422        MOV     FRMERR(R2),$TMP6
004574  012767  004604  174306        MOV     #3$,SLPERR      ;RETURN TO 3$ AFTER PRINTING LINE
004602  104013                        ERROR   13
004604  005201                3$:     INC     R1              ;STEP TO NEXT LINE
004606  006367  014722                ASL     LINMSK          ;SHIFT THE MARKER
004612  001401                        BEQ     ENDA            ;BR IF ALL LINES REPORTED
004614  000720                        BR      1$              ;GO BACK AND DO THIS LINE
```

```
3038  004616  000004                    ENDA:   SCOPE
3039  004620  105067  174256                    CLRB    $TSTNM          ;RE-INIT TEST NUMBER FOR NEXT PASS
3040  004624  012767  000240  004136            MOV     #240,$EOP       ;NOP THE SCOPE IN ENDPASS ROUTINE
3041  004632  005267  015312                    INC     DHNUM           ;GENERATE NEW DH11 NUMBER
3042  004636  062767  000002  015464            ADD     #2,ADPTR        ;UPDATE THE TABLE POINTERS
3043  004644  062767  000002  015460            ADD     #2,VCPTR
3044  004652  062767  000002  015454            ADD     #2,BRPTR
3045  004660  006367  014642                    ASL     SELMSK          ;SHIFT MARKER TO TEST NEXT DH11
3046  004664  001002                            BNE     1$              ;BR IF NOT TESTED ALL DH11'S
3047  004666  000167  004076                    JMP     $EOP            ;JUMP TO EOP IF WE HAVE
3048  004672  036767  014630  014630  1$:       BIT     SELMSK,DHSEL    ;IS THIS DH11 SELECTED ?
3049  004700  001746                            BEQ     ENDA            ;BR IF NOT
3050  004702  000167  175534                    JMP     RSTRTA          ;GO TEST THIS DH11
```

```
                        .SBTTL SUB-PROGRAM 2 - SINGLE LINE ECHO/CABLE TESTS

                        ;               ************************************
                        ;               *  SINGLE LINE ECHO TESTS          *
                        ;               ************************************

004706  012767  177777  015444  ECHO:   MOV     #-1,RETFLG      ;SET RETURN FLAG - COME BACK
004714  005067  015426          CLR     DPFLG           ;CLEAR PATTERNS TEST FLAG
004720  005067  014770          CLR     VCFLG           ;INIT VECTOR SETUP FLAG
004724  000167  174714          JMP     BEGINA          ;TO "ECHO1" AFTER SETUP

004730  012767  160020  014564  ECHO1:  MOV     #160020,DHADR   ;SET UP DH11 DEFAULT ADDRESS
004736  012767  000330  014560          MOV     #330,DHVCT      ;SET UP DH11 DEFAULT VECTOR
004744  104401                  TYPE                    ;PRINT I.D. MESSAGE
004746  026252                  ECMSG1                  ;"SINGLE LINE ECHO TEST - CONNECT
                                                        ;TERMINAL TO TEST LINE"
004750  000167  011250          JMP     INPAR           ;GO SET UP DEVICE AND VECTOR
                                                        ;ADDRESSES - COME BACK TO "ECHO2"

004754  104401          ECHO2:  TYPE                    ;GO ASK FOR TTY INPUT
004756  026351                  ECMSG2                  ;"LINE # (00 - 17 OCTAL)"
004760  104412                  RDOCT                   ;INPUT LINE NO. FM TTY
004762  012667  015164          MOV     (SP)+,LINE      ;GET NO. TYPED
004766  042767  177760  015156  BIC     #177760,LINE    ;CLEAR JUNK
004774  016702  015152          MOV     LINE,R2         ;GET LINE NO.
005000  005202                  INC     R2              ;CORRECT FOR SHIFT ROUTINE
005002  012767  000001  014524  MOV     #1,LINMSK       ;INIT LINE SELECT BIT MASK
005010  005302          1$:     DEC     R2              ;COUNT ONE LINE CHECKED
005012  001403                  BEQ     2$              ;BR IF DONE
005014  006367  014514          ASL     LINMSK          ;SHIFT SELECT BIT
005020  000773                  BR      1$              ;GO COUNT IT
005022  004767  013204  2$:     JSR     PC,LPRIN        ;GO ASK FOR AND SET UP LINE PARAMETERS

005026  005767  015314          TST     DPFLG           ;DATA PATTERNS TEST ?
005032  001401                  BEQ     3$              ;BR IF NOT
005034  000207                  RTS     PC              ;RETURN TO PATTERNS TEST

005036  105067  021201  3$:     CLRB    EC2             ;CLEAR ECHO BUFFER
005042  104401                  TYPE
005044  027373                  SNMSG1                  ;"SEND MODE - Y OR N ?"
005046  104410          4$:     RDCHR                   ;GET CHAR TYPED
005050  012600                  MOV     (SP)+,R0        ;GET CHAR TYPED
005052  122700  000015          CMPB    #15,R0          ;WAS IT A <CR> ?
005056  001405                  BEQ     5$              ;BR IF YES
005060  110067  021157          MOVB    R0,EC2          ;ECHO WHAT WAS TYPED
005064  104401                  TYPE
005066  026243                  EC2
005070  000766                  BR      4$              ;GO WAIT FOR TERMINATOR
005072  105767  021145  5$:     TSTB    EC2             ;<CR> ONLY ??
005076  001412                  BEQ     ECHO3           ;BR IF YES
005100  122767  000116  021135  CMPB    #116,EC2        ;WAS IT AN "N" ??
005106  001406                  BEQ     ECHO3           ;BR IF YES
005110  122767  000131  021125  CMPB    #131,EC2        ;WAS IT A "Y" ??
005116  001347                  BNE     3$              ;BR IF NOT ASK AGAIN
005120  000167  000574          JMP     SENDP1          ;GO TO SEND ROUTINE
```

```
                                ECHO3:  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
                                        CLR     CEXIT           ;INIT CONTROL-C EXIT FLAG
                                        CLR     EXFLAG          ;CLEAR TEST EXIT FLAGS
                                        MOV     #120240,DHRLVL  ;INIT FOR BR LEVEL 5
                                        MOV     DHADR,R1        ;SET UP DEVICE ADDRESS
                                        MOV     #BIT11,(R1)     ;CLEAR THE SELECTED DH11
                                        MOV     DHVCT,R0        ;GET THE FIRST VECTOR ADDRESS
                                        MOV     #RINT2,(R0)+    ;SET UP THE VECTORS
                                        MOVB    DHRLVL,(R0)
                                        TST     (R0)+
                                        MOV     #TINT2,(R0)+
                                        MOVB    DHTLVL,(R0)
                                        MOV     LINE,(R1)       ;SET THE LINE SELECT BITS
                                        MOV     #TBUF,R2        ;INIT BUFFER POINTER
                                        BIS     #BIT06,(R1)     ;ENABLE  RCVR INTRS
                                        MOV     CURLPR,LPR(R1)  ;SET UP LINE PARAMETERS
                                        JSR     R5,SUNUM        ;PUT LINE NO. IN MESSAGE
                                        LINE
                                        ECMSG3+20
                                        TYPE                    ;GIVE DIRECTIONS
                                        ECMSG3                  ;"GO TYPE CHARS ON TEST TERMINAL"
                                        TYPE                    ;PRINT DIRECTIONS
                                        ECMSG4                  ;"[CONTROL-C TO EXIT] [CONTROL-E TO ECHO BUFFER]"
                                        JSR     PC,CHPS1        ;GO CLEAR PSW

                                DHWAIT: MOV     #200,TIMEA      ;INIT TIMER "A"
                                        CLR     TIMEB           ;INIT TIMER "B"
                                1$:     TST     CEXIT           ;CONTROL-C EXIT ??
                                        BNE     2$              ;BR IF YES
                                        JSR     PC,TIMEIT       ;CALL TIMER
                                        BR      1$              ;BR IF NO TIMEOUT

                                        MOV     R1,$REG1        ;SAVE DEVADR
                                        MOV     (R1),$TMP0      ;SAVE CONTENT OF SCR
                                        BIS     #BIT11,(R1)     ;CLEAR OUT THE DH11
                                        MOV     #2$,$LPERR      ;SET ERROR LOOP RETURN
                                        ERROR   16              ;REPORT RCVR WAIT TIMEOUT
                                2$:     JMP     ECHO2           ;GO RESTART
```

```
3147                                        ;TRANSMITTER INTERRUPT SERVICE ROUTINE TWO
3148
3149  005330  042711  120000        TINT2:  BIC     #BIT15+BIT13,(R1)       ;DISABLE XMIT INTRS
3150  005334  022767  000001 015030        CMP     #1,EXFLAG               ;CONTROL-C FLAG ?
3151  005342  001437                    BEQ     2$                      ;BR IF YES
3152  005344  022767  000003 015020        CMP     #3,EXFLAG               ;WAS BUFFER JUST DUMPED ?
3153  005352  001437                    BEQ     3$                      ;BR IF YES
3154  005354  022767  000002 015010        CMP     #2,EXFLAG               ;CONTROL-E FLAG ?
3155  005362  001403                    BEQ     1$                      ;BR IF YES
3156  005364  020227  034126              CMP     R2,#TBUF+600.           ;BUFFER FULL ?
3157  005370  002434                    BLT     31$                     ;BR IF NOT
3158  005372  012767  000003 014772 1$:      MOV     #3,EXFLAG               ;SET DUMP FLAG
3159  005400  162702  032776              SUB     #TBUF,R2                ;SET UP BYTE COUNT REG
3160  005404  005402                    NEG     R2
3161  005406  010261  000010              MOV     R2,BCR(R1)
3162  005412  012761  032776 000006        MOV     #TBUF,CAR(R1)           ;SET UP CURRENT ADDR REG
3163  005420  012767  000200 014712        MOV     #200,TIMEA              ;INIT TIMER
3164  005426  052711  020000              BIS     #BIT13,(R1)             ;ENABLE XMITTR INTR
3165  005432  016761  014076 000012        MOV     LINMSK,BAR(R1)          ;ACTIVATE LINE
3166  005440  000415                    BR      4$                      ;GO EXIT
3167
3168  005442  012767  177777 014674 2$:      MOV     #-1,CEXIT               ;SET CONTROL-C EXIT
3169  005450  000411                    BR      4$                      ;GO EXIT
3170
3171  005452  012702  032776        3$:     MOV     #TBUF,R2                ;RESET ECHO BUFFER P[OPINTER
3172  005456  005067  014710              CLR     EXFLAG                  ;INIT EXIT FLAG
3173  005462  012767  000200 014650 31$:     MOV     #200,TIMEA              ;INIT TIMER AGAIN
3174  005470  052711  000100              BIS     #BIT06,(R1)             ;ENABLE RCVR INTR
3175
3176  005474  000002                4$:     RTI                             ;RETURN TO MAINLINE
3177
3178
```

```
                                        ;RECEIVER INTERRUPT SERVICE ROUTINE TWO
                                        
005476  042711  000100          RINT2:  BIC     #BIT06,(R1)             ;DISABLE RCVR INTR
005502  016167  000002  173472          MOV     NRC(R1),$TMP0           ;SAVE THE DATA TYPED
005510  042767  177600  173464          BIC     #177600,$TMP0           ;CLEAR HIGH BYTE
005516  022767  000003  173456          CMP     #3,$TMP0                ;CONTROL-C TYPED ??
005524  001015                          BNE     1$                      ;BR IF NOT
                                        
005526  112767  000136  014640          MOVB    #136,ECBUF              ;SET UP TO ECHO CONTROL-C
005534  112767  000103  014633          MOVB    #103,ECBUF+1
005542  012761  177776  000010          MOV     #-2,BCR(R1)             ;SET UP BCR REG
005550  012767  000001  014614          MOV     #1,EXFLAG               ;SET CONTROL-C FLAG
005556  000444                          BR      4$                      ;GO OUT PUT CHAR TYPED
                                        
005560  022767  000005  173414  1$:     CMP     #5,$TMP0                ;WAS IT A CONTROL-E ??
005566  001021                          BNE     2$                      ;BR IF NOT
                                        
005570  112767  000136  014576          MOVB    #136,ECBUF              ;SET UP TO ECHO CONTROL-E
005576  112767  000105  014571          MOVB    #105,ECBUF+1
005604  112722  000136                  MOVB    #136,(R2)+              ;PUT IN THE ECHO BUFFER
005610  112722  000105                  MOVB    #105,(R2)+
005614  012761  177776  000010          MOV     #-2,BCR(R1)             ;SET UP BYTE COUNT REG
005622  012767  000002  014542          MOV     #2,EXFLAG               ;SET CONTROL-E FLAG
005630  000417                          BR      4$                      ;GO EXIT
                                        
005632  022767  000012  173342  2$:     CMP     #12,$TMP0               ;WAS IT A LINE FEED ??
005640  001003                          BNE     3$                      ;BR IF NOT
005642  004767  012060                  JSR     PC,LDFILL               ;GO LOAD FILLERS
005646  000410                          BR      4$                      ;GO EXIT
                                        
005650  116767  173326  014516  3$:     MOVB    $TMP0,ECBUF             ;SET UP CHAR TO ECHO
005656  116722  173320                  MOVB    $TMP0,(R2)+
005662  012761  177777  000010          MOV     #-1,BCR(R1)             ;OUTPUT ONE CHAR ONLY
005670  012761  022374  000006  4$:     MOV     #ECBUF,CAR(R1)          ;SET UP CURRENT ADDR REG
005676  012767  000200  014434          MOV     #200,TIMEA              ;INIT TIMER AGAIN
005704  052711  020000                  BIS     #BIT13,(R1)             ;ENABLE XMITTR INTR
005710  016761  013620  000012          MOV     LINMSK,BAR(R1)          ;ACTIVATE THE LINE
005716  000002                          RTI                             ;RETURN TO MAINLINE
```

```
3220  005720  104401                SENDP1: TYPE                    ;ASK FOR DIRECTIONS
3221  005722  027423                        SNMSG2                  ;"TYPE SEND BUFFER - TERMINATE WITH CONTROL-C"
3222  005724  012705  032776                MOV     #TBUF,R5        ;SET UP BUFFER POINTER
3223  005730  104410                1$:     RDCHR                   ;GET CHAR
3224  005732  012600                        MOV     (SP)+,R0
3225  005734  110067  020303                MOVB    R0,EC2          ;ECHO CHAR
3226  005740  104401                        TYPE
3227  005742  026243                        EC2
3228  005744  022700  000003                CMP     #3,R0           ;WAS IT A CONTROL-C ??
3229  005750  001421                        BEQ     4$              ;BR IF YES
3230
3231  005752  026727  013624  000400        CMP     CHRCNT,#256.    ;BUFFER FULL ??
3232  005760  003015                        BGT     4$              ;BR IF YES
3233  005762  022700  000012                CMP     #12,R0          ;WAS IT A LINE FEED ?
3234  005766  001010                        BNE     3$              ;BR IF NOT
3235
3236  005770  110025                        MOVB    R0,(R5)+        ;LOAD CHAR TYPED
3237  005772  116704  014442                MOVB    FILLB,R4        ;GET FILLER COUNT
3238  005776  116725  014434        2$:     MOVB    FILLA,(R5)+     ;LOAD A FILLER
3239  006002  005304                        DEC     R4              ;COUNT IT
3240  006004  001374                        BNE     2$              ;BR IF NOT DONE
3241  006006  000750                        BR      1$              ;GET SOME MORE INPUT
3242
3243  006010  110025                3$:     MOVB    R0,(R5)+        ;LOAD BUFFER
3244  006012  000746                        BR      1$              ;GO GET SOME MORE
3245
3246  006014  004767  011630        4$:     JSR     PC,SENDP2       ;GO XMIT THE BUFFER
3247  006020  105067  020217        5$:     CLRB    EC2             ;CLEAR ECHO BUFFER
3248  006024  104401                        TYPE
3249  006026  027503                        SNMSG3                  ;"CHANGE PARAMETERS- Y OR N"
3250  006030  104410                6$:     RDCHR
3251  006032  012600                        MOV     (SP)+,R0        ;GET CHAR
3252  006034  122700  000015                CMPB    #15,R0          ;WAS IT A <CR> HE TYPED ??
3253  006040  001405                        BEQ     7$              ;BR IF IT WAS
3254  006042  110067  020175                MOVB    R0,EC2          ;ECHO IT
3255  006046  104401                        TYPE
3256  006050  026243                        EC2
3257  006052  000766                        BR      6$              ;GO WAIT FOR TERMINATOR
3258
3259  006054  105767  020163        7$:     TSTB    EC2             ;<CR> ONLY ??
3260  006060  001717                        BEQ     SENDP1          ;BR IF YES
3261  006062  122767  000116  020153        CMPB    #116,EC2        ;DID HE SAY NO ??
3262  006070  001713                        BEQ     SENDP1          ;BR IF HE DID
3263  006072  122767  000131  020143        CMPB    #131,EC2        ;DID HE SAY YES ??
3264  006100  001347                        BNE     5$              ;GO ASK ALL OVER AGAIN
3265  006102  000167  176646                JMP     ECHO2           ;GO ASK FOR NEW PARAMETERS
```

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

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

006106  012767  177777  014232  EXPAT:  MOV     #-1,DPFLG       ;SET PATTERNS TEST FLAG
006114  005067  014240          CLR     RETFLG          ;CLR ECHO TESTS FLAG
006120  005067  013570          CLR     VCFLG           ;CLEAR VECTOR SETUP FLAG
006124  000167  173514          JMP     BEGINA          ;GO SET UP RETURN TO "EXPAT1"

006130  012767  160020  013364  EXPAT1: MOV     #160020,DHADR   ;SET UP DEFAULT DH11 ADDR
006136  012767  000330  013360          MOV     #330,DHVCT      ;AND VECTOR TOO
006144  104401                        TYPE
006146  027542                        DPMSG1                  ;"DATA PATTERNS TESTS - CONNECT TEST JUMPAR"
006150  000167  010050          JMP     INPAR           ;GO GET SOME PARAMETERS RETURN TO EXPAT2

006154  004767  176574  EXPAT2: JSR     PC,ECHO2        ;GO GET REST OF THE PARAMETERS
006160  004767  013170          JSR     PC,SUCLMK       ;GO SET UP CHAR LENGTH MASK
006164  104401          1$:     TYPE
006166  027620                        DPMSG2                  ;"BUFFER SIZE ? (1-512)
006170  104413                        RDDEC                   ;GET THE SIZE TYPED
006172  012600                        MOV     (SP)+,R0
006174  001406                        BEQ     2$              ;BR IF DEFAULT TO 256. <CR>

006176  020027  001001          CMP     R0,#513.        ;TOO BIG ?
006202  002405                  BLT     3$              ;BR IF NOT
006204  104401                  TYPE
006206  027652                  DPMSG3                  ;"INVALID SIZE - TRY AGAIN"
006210  000765                  BR      1$              ;GO ASK AGAIN

006212  012700  000400  2$:     MOV     #256.,R0        ;DEFAULT TO 256. BYTE BUFFER
006216  005400          3$:     NEG     R0              ;MAKE IT NEG BYTE COUNT
006220  010067  013356          MOV     R0,CHRCNT       ;SAVE IT FOR TEST

006224  012767  120240  013714  MOV     #120240,DHRLVL  ;SET BR LEVELS TO BR5
006232  016700  013266          MOV     DHVCT,R0        ;SET UP VECTORS
006236  012720  010266          MOV     #RINT3,(R0)+
006242  116710  013700          MOVB    DHRLVL,(R0)
006246  005720                  TST     (R0)+
006250  012720  007644          MOV     #TINT3,(R0)+
006254  116710  013667          MOVB    DHTLVL,(R0)
```

```
3308  006260  104407                    EXPAT3: CKSWR                   ;TEST FOR CHANGE IN SOFT-SWR
3309  006262  005067  014066                    CLR     PATFLG          ;CLEAR <CR> SEQUENCE FLAG
3310  006266  105067  017751                    CLRB    EC2             ;CLEAR ECHO BUFFER
3311  006272  016701  013224                    MOV     DHADR,R1        ;INIT R1 TO POINT TO SCR REG
3312  006276  104401                            TYPE
3313  006300  027707                            DPMSG4                  ;"PATTERN TYPE ? (A,U,D,R,S, OR B)"
3314  006302  104410                    7$:     RDCHR
3315  006304  012600                            MOV     (SP)+,R0        ;GET WHAT HE TYPED
3316  006306  120027  000015                    CMPB    R0,#15          ;WAS IT A <CR> ??
3317  006312  001407                            BEQ     9$              ;BR IF YES
3318  006314  010067  014032                    MOV     R0,DATPAT
3319  006320  110067  017717                    MOVB    R0,EC2          ;ECHO IT
3320  006324  104401                            TYPE
3321  006326  026243                            EC2
3322  006330  000764                            BR      7$              ;GO WAIT FOR TERMINATOR
3323
3324  006332  104401                    9$:     TYPE
3325  006334  027761                            DPMSG5                  ;"SET SR07=1 TO LOCK ON TEST PATTERN"
3326  006336  105767  017701                    TSTB    EC2             ;<CR> ONLY ??
3327  006342  001005                            BNE     8$              ;BR IF NOT
3328  006344  012767  000015  014002            MOV     #15,PATFLG
3329  006352  000167  000506                    JMP     DPATCR          ;GO SEQUENCE A,U,D,R PATTERNS
3330
3331  006356  022767  000101  013766    8$:     CMP     #101,DATPAT     ;ALTERNATING 1/0 ?
3332  006364  001002                            BNE     1$              ;BR IF NOT
3333  006366  000167  000102                    JMP     DPATA           ;GO DO IT
3334
3335  006372  022767  000125  013752    1$:     CMP     #125,DATPAT     ;UP COUNT PATTERN ?
3336  006400  001002                            BNE     2$              ;BR IF NOT
3337  006402  000167  000164                    JMP     DPATU           ;GO DO IT
3338
3339  006406  022767  000104  013736    2$:     CMP     #104,DATPAT     ;DOWN COUNT PATTERN ?
3340  006414  001002                            BNE     3$              ;BR IF NOT
3341  006416  000167  000246                    JMP     DPATD           ;GO DO IT
3342
3343  006422  022767  000122  013722    3$:     CMP     #122,DATPAT     ;RANDOM PATTERN ?
3344  006430  001002                            BNE     4$              ;BR IF NOT
3345  006432  000167  000330                    JMP     DPATR           ;GO DO IT
3346
3347  006436  022767  000123  013706    4$:     CMP     #123,DATPAT     ;SINGLE CHAR PATTERN ?
3348  006444  001002                            BNE     5$              ;BR IF NOT
3349  006446  000167  000474                    JMP     DPATS           ;GO DO IT
3350
3351  006452  022767  000102  013672    5$:     CMP     #102,DATPAT     ;TYPE IN BUFFER ?
3352  006460  001002                            BNE     6$              ;BR IF NOT
3353  006462  000167  000620                    JMP     DPATB           ;GO DO IT
3354
3355  006466  104401                    6$:     TYPE
3356  006470  030023                            DPMSG6                  ;"INVALID PATTERN - TRY AGAIN"
3357  006472  000672                            BR      EXPAT3          ;GO ASK AGAIN
3358
```

```
 3359  006474  005067  013650          DPATA:  CLR     DATCNT          ;INIT ITERATION COUNTER
 3360  006500  004767  012432          1$:     JSR     PC,SUPATA       ;GO SET UP THE PATTERN
 3361  006504  004767  001016                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
 3362  006510  005267  013634                  INC     DATCNT          ;COUNT IT
 3363  006514  026767  013642  013626          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES
 3364  006522  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
 3365
 3366  006524  016767  013620  172430          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
 3367  006532  005067  013612                  CLR     DATCNT          ;INIT COUNTER
 3368  006536  012767  006546  172344          MOV     #2$,$LPERR      ;COME BACK TO 2$
 3369  006544  104017                          ERROR   17              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
 3370  006546  022767  000015  013600  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
 3371  006554  001001                          BNE     3$              ;BR IF NOT
 3372  006556  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
 3373  006560  105777  172354          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
 3374  006564  100745                          BMI     1$              ;BR IF YES
 3375  006566  000167  177466                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
 3376
 3377  006572  005067  013552          DPATU:  CLR     DATCNT          ;INIT ITERATION COUNTER
 3378  006576  004767  012362          1$:     JSR     PC,SUPATU       ;GO SET UP THE PATTERN
 3379  006602  004767  000720                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
 3380  006606  005267  013536                  INC     DATCNT          ;COUNT IT
 3381  006612  026767  013544  013530          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES ?
 3382  006620  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
 3383
 3384  006622  016767  013522  172332          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
 3385  006630  005067  013514                  CLR     DATCNT          ;INIT COUNTER
 3386  006634  012767  006644  172246          MOV     #2$,$LPERR      ;COME BACK TO 2$
 3387  006642  104020                          ERROR   20              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
 3388  006644  022767  000015  013502  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
 3389  006652  001001                          BNE     3$              ;BR IF NOT
 3390  006654  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
 3391  006656  105777  172256          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
 3392  006662  100745                          BMI     1$              ;BR IF YES
 3393  006664  000167  177370                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
 3394
 3395  006670  005067  013454          DPATD:  CLR     DATCNT          ;INIT ITERATION COUNTER
 3396  006674  004767  012314          1$:     JSR     PC,SUPATD       ;GO SET UP THE PATTERN
 3397  006700  004767  000622                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
 3398  006704  005267  013440                  INC     DATCNT          ;COUNT IT
 3399  006710  026767  013446  013432          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES
 3400  006716  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
 3401
 3402  006720  016767  013424  172234          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
 3403  006726  005067  013416                  CLR     DATCNT          ;INIT COUNTER
 3404  006732  012767  006742  172150          MOV     #2$,$LPERR      ;COME BACK TO 2$
 3405  006740  104021                          ERROR   21              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
 3406  006742  022767  000015  013404  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
 3407  006750  001001                          BNE     3$              ;BR IF NOT
 3408  006752  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
 3409  006754  105777  172160          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
 3410  006760  100745                          BMI     1$              ;BR IF YES
 3411  006762  000167  177272                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
 3412
 3413  006766  005067  013356          DPATR:  CLR     DATCNT          ;INIT ITERATION COUNTER
 3414  006772  004767  012250          1$:     JSR     PC,SUPATR       ;GO SET UP THE PATTERN
```

```
3415  006776  004767  000524          JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
3416  007002  005267  013342          INC     DATCNT          ;COUNT IT
3417  007006  026767  013350  013334  CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES
3418  007014  001366                  BNE     1$              ;BR IF NOT DO IT AGAIN
3419
3420  007016  016767  013326  172136  MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
3421  007024  005067  013320          CLR     DATCNT          ;INIT COUNTER
3422  007030  012767  007040  172052  MOV     #2$,$LPERR      ;COME BACK TO 2$
3423  007036  104022                  ERROR   22              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
3424  007040  022767  000015  013306  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
3425  007046  001001                  BNE     3$              ;BR IF NOT
3426  007050  000207                  RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
3427  007052  105777  172062          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
3428  007056  100745                  BMI     1$              ;BR IF YES
3429  007060  000167  177174          JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
3430
3431  007064  012767  000101  013260  DPATCR: MOV     #101,DATPAT     ;FLAG 1/0 PATTERN
3432  007072  004767  177376          JSR     PC,DPATA        ;CALL FOR 1/0 PATTERN
3433  007076  012767  000125  013246          MOV     #125,DATPAT     ;FLAG UP COUNT PATTERN
3434  007104  004767  177462          JSR     PC,DPATU        ;CALL FOR UP COUNT PATTERN
3435  007110  012767  000104  013234          MOV     #104,DATPAT     ;FLAG DOWN COUNT PATTERN
3436  007116  004767  177546          JSR     PC,DPATD        ;CALL FOR DOWN COUNT PATTERN
3437  007122  012767  000122  013222          MOV     #122,DATPAT     ;FLAG RANDOM DATA PATTERN
3438  007130  004767  177632          JSR     PC,DPATR        ;CALL FO RANDOM PATTERN
3439  007134  105777  172000          TSTB    @SWR            ;LOCK ON ALL FOUR PATTERNS
3440  007140  100751                  BMI     DPATCR          ;BR IF YES
3441  007142  000167  177112          JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3442
3443  007146  105067  017071  DPATS:  CLRB    EC2             ;CLEAR THE ECHO BUFFER
3444  007152  005067  013172          CLR     DATCNT          ;INIT ITERATION COUNTER
3445  007156  104401                  TYPE
3446  007160  030063                  DPMSG7                  ;"TYPE SINGLE TEST CHAR"
3447  007162  104410          3$:     RDCHR                   ;GET CHAR
3448  007164  012600                  MOV     (SP)+,R0        ;GET WHAT HE TYPED
3449  007166  122700  000015          CMPB    #15,R0          ;WAS IT A <CR> ??
3450  007172  001407                  BEQ     4$              ;BR IF YES
3451  007174  010067  013156          MOV     R0,SINGLE       ;SAVE IT FOR LOADING BUFFER
3452  007200  110067  017037          MOVB    R0,EC2          ;ECHO IT ON TTY
3453  007204  104401                  TYPE
3454  007206  026243                  EC2
3455  007210  000764                  BR      3$              ;GO WAIT FOR TERMINATOR
3456
3457  007212  105767  017025  4$:     TSTB    EC2             ;WAS SINGLE CHAR A <CR> ??
3458  007216  001003                  BNE     1$              ;BR IF NOT A <CR> ONLY
3459  007220  012767  000015  013130          MOV     #15,SINGLE      ;SET UP TO LOAD ALL <CR>'S
3460  007226  004767  012074  1$:     JSR     PC,SUPATS       ;GO SET IT UP IN BUFFER
3461  007232  004767  000270          JSR     PC,DHST2        ;GO EXECUTE IT ON DH11
3462  007236  005267  013106          INC     DATCNT          ;COUNT ONE TIME
3463  007242  026767  013114  013100  CMP     PATLIM,DATCNT   ;DONE REQUIRED ITERATIONS ?
3464  007250  001366                  BNE     1$              ;BR IF NOT
3465
3466  007252  016767  013072  171702          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
3467  007260  005067  013064          CLR     DATCNT          ;INIT ITERATION COUNTER
3468  007264  012767  007274  171616          MOV     #2$,$LPERR      ;COME BACK TO 2$ ALWAYS
3469  007272  104023                  ERROR   23              ;REPORT DONE SINGLE CHAR PATTERN
3470  007274  105777  171640  2$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
```

```
3471  007300  100752                BMI     1$              ;BR IF YES
3472  007302  000167  176752        JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3473
```

```
3474  007306  005067  013036          DPATB:  CLR     DATCNT          ;INIT ITERATION COUNTER
3475  007312  104401                          TYPE
3476  007314  030116                          DPMSGA                  ;"TYPE IN BUFFER - TERMINATE WITH CONTROL-C"
3477  007316  012705  032776                  MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
3478  007322  104410                  1$:     RDCHR                   ;GET A CHAR
3479  007324  012667  013026                  MOV     (SP)+,SINGLE    ;SAVE IT
3480  007330  116767  013022  016705          MOVB    SINGLE,EC2      ;ECHO IT
3481  007336  104401                          TYPE
3482  007340  026243                          EC2
3483
3484  007342  022767  000003  013006          CMP     #3,SINGLE       ;WAS IT A CONTROL-C ??
3485  007350  001423                          BEQ     3$              ;BR IF YES
3486
3487  007352  020527  033777                  CMP     R5,#TBUF+513.   ;BUFFER FULL ??
3488  007356  001420                          BEQ     3$              ;BR IF YES
3489
3490  007360  022767  000012  012770          CMP     #12,SINGLE      ;WAS IT A LINE FEED ??
3491  007366  001011                          BNE     2$              ;BR IF NOT
3492
3493  007370  016700  013044                  MOV     FILLB,R0        ;LOAD LF PLUS FILLERS
3494  007374  116725  012756                  MOVB    SINGLE,(R5)+
3495  007400  116725  013032          11$:    MOVB    FILLA,(R5)+     ;LOAD A FILLER CHAR
3496  007404  005300                          DEC     R0
3497  007406  001374                          BNE     11$             ;BR TILL REQUIRED FILLERS LOADED
3498  007410  000744                          BR      1$              ;GO ASK FOR ANOTHER CHAR
3499
3500  007412  116725  012740          2$:     MOVB    SINGLE,(R5)+    ;LOAD IT IN BUFFER
3501  007416  000741                          BR      1$              ;GO GET NEXT CHAR
3502
3503  007420  112767  000136  016617  3$:     MOVB    #136,EC3                ;ECHO CONTROL-C
3504  007426  112767  000103  016612          MOVB    #103,EC3+1
3505  007434  104401                          TYPE
3506  007436  026245                          EC3
3507  007440  162705  032776                  SUB     #TBUF,R5        ;SET UP CHAR COUNT
3508  007444  005405                          NEG     R5
3509  007446  010567  012130                  MOV     R5,CHRCNT
3510  007452  004767  000050          4$:     JSR     PC,DHST2        ;GO EXECUTE PATTERN
3511  007456  005267  012666                  INC     DATCNT
3512  007462  026767  012674  012660          CMP     PATLIM,DATCNT   ;DONE REQUIRED ITERATIONS
3513  007470  001370                          BNE     4$              ;BR IF NOT
3514
3515  007472  016767  012652  171462          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
3516  007500  005067  012644                  CLR     DATCNT          ;INIT ITERATION COUNTER
3517  007504  012767  007514  171376          MOV     #5$,$LPERR      ;RETURN TO 5$
3518  007512  104024                          ERROR   24              ;DONE REQUIRED ITERATIONS
3519  007514  105777  171420          5$:     TSTB    @SWR            ;LOCK ON THIS BUFFER ??
3520  007520  100754                          BMI     4$              ;BR IF YES
3521  007522  000167  176532                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3522
```

```
3523  007526  004767  005122          DHST2:  JSR     PC,DHSET1       ;GO SET UP THE DH11
3524  007532  056761  011776  000012          BIS     LINMSK,BAR(R1)  ;ACTIVATE THE LINE
3525  007540  004767  011662                  JSR     PC,CHPS1        ;GO CLEAR PSW
3526
3527  007544  012767  000200  012566  PTWAIT: MOV     #200,TIMEA      ;INIT TIMERS
3528  007552  005067  012564                  CLR     TIMEB
3529  007556  005767  012026          1$:     TST     RDONE           ;DONE ENTIRE PATTERN ?
3530  007562  001023                          BNE     3$              ;BR IF YES
3531  007564  004767  007324                  JSR     PC,TIMEIT       ;CALL THE TIMER
3532  007570  000772                          BR      1$              ;EXECUTED IF NO TIMEOUT
3533
3534  007572  012777  004000  011722          MOV     #BIT11,@DHADR   ;CLEAR THE DH11
3535  007600  016767  012544  171354          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNTER
3536  007606  016767  012540  171350          MOV     DATPAT,$REG1    ;SAVE PATTERN TYPE
3537  007614  012767  007624  171266          MOV     #2$,SLPERR      ;SET UP ERROR RETURN
3538  007622  104025                          ERROR   25              ;DATA PATTERNS TEST TIMEOUT ERROR
3539
3540  007624  005726          2$:     TST     (SP)+           ;FIX STACK SINCE WE ARE SKIPPING RTS
3541  007626  000167  176426                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3542
3543  007632  052711  004000          3$:     BIS     #BIT11,(R1)     ;CLEAR THE DH11
3544  007636  004767  000740                  JSR     PC,CKERDP       ;GO CHECK DATA BUFFERS
3545  007642  000207                          RTS     PC              ;RETURN TO CONTROL ROUTINE
3546
3547
```

```
3548
3549                                    ;TRANSMITTER INTERRUPT SERVICE ROUTINE THREE
3550
3551  007644  032711  002000      TINT3:  BIT     #BIT10,(R1)     ;NON EX MEM ERROR ??
3552  007650  001430                      BEQ     2$              ;BR IF NOT
3553
3554  007652  011103                      MOV     (R1),R3         ;SAVE THE SCR
3555  007654  004767  011562              JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3556  007660  012711  004000              MOV     #BIT11,(R1)     ;CLEAR OUT THE DH11
3557  007664  116704  012262              MOVB    LINE,R4         ;SET UP THE S/B DATA
3558  007670  042703  175760              BIC     #175760,R3      ;CLEAR OUT SUPERFLUOUS BITS
3559  007674  010102                      MOV     R1,R2           ;SET UP REGADR
3560  007676  004767  005226              JSR     PC,SUER1        ;GO SET UP ERROR INFO
3561  007702  004567  005136              JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3562  007706  022152                      LINE
3563  007710  022506                      EM1+44
3564  007712  012767  007722  171170      MOV     #1$,$LPERR      ;SET UP THE ERROR LOOP RETURN
3565  007720  104001                      ERROR   1               ;NON EX MEM ERROR
3566  007722  022626              1$:     CMP     (SP)+,(SP)+     ;POP THE STACK
3567  007724  005726                      TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3568  007726  000167  176326              JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3569
3570  007732  011103              2$:     MOV     (R1),R3         ;GET THE SCR REG CONTENTS
3571  007734  100431                      BMI     4$              ;BR IF XMIT DONE SET
3572
3573  007736  004767  011500              JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3574  007742  012711  004000              MOV     #BIT11,(R1)     ;CLEAR THE DH11 - FATAL ERROR
3575  007746  012704  100000              MOV     #BIT15,R4       ;SET UP S/B DATA
3576  007752  156704  012174              BISB    LINE,R4
3577  007756  042703  077760              BIC     #77760,R3       ;CLEAR OUT SUPERFLUOUS BITS
3578  007762  010102                      MOV     R1,R2           ;SET UP REGADR
3579  007764  004767  005140              JSR     PC,SUER1        ;GO SET UP ERROR INFO
3580  007770  004567  005050              JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3581  007774  022152                      LINE
3582  007776  022670                      EM2+54
3583  010000  012767  010010  171102      MOV     #3$,$LPERR      ;SET UP ERROR LOOP RETURN
3584  010006  104002                      ERROR   2               ;XMITTR FALSE INTERRUPT
3585  010010  022626              3$:     CMP     (SP)+,(SP)+     ;POP THE STACK
3586  010012  005726                      TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3587  010014  000167  176240              JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
3588
3589  010020  005761  000012      4$:     TST     BAR(R1)         ;DID BAR BIT CLEAR ??
3590  010024  001430                      BEQ     6$              ;BR IF YES
3591
3592  010026  004767  011410              JSR     PC,CHPS2        ;GO LOCK OUT INTRS
3593  010032  016103  000012              MOV     BAR(R1),R3      ;GET THE WAS DATA
3594  010036  012711  004000              MOV     #BIT11,(R1)     ;CLEAR THE DH11
3595  010042  005004                      CLR     R4              ;SET UP S/B DATA
3596  010044  010102                      MOV     R1,R2           ;SET UP REGADR
3597  010046  062702  000012              ADD     #BAR,R2
3598  010052  004767  005052              JSR     PC,SUER1        ;GO SET UP ERROR INFO
3599  010056  004567  004762              JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
3600  010062  022152                      LINE
3601  010064  022750                      EM3+55
3602  010066  012767  010076  171014      MOV     #5$,$LPERR      ;SAVE THE ERROR LOOP RETURN
3603  010074  104003                      ERROR   3               ;BUFFER ACTIVE REG FAILED TO CLEAR
```

```
3604  010076  022626                  5$:   CMP   (SP)+,(SP)+    ;POP GOES THE STACK
3605  010100  005726                        TST   (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3606  010102  000167  176152              JMP   EXPAT3            ;GO ASK FOR NEW PATTERN
3607
3608  010106  005761  000010        6$:   TST   BCR(R1)          ;DID BYTE COUNT GO TO ZERO ??
3609  010112  001430                      BEQ   8$                ;BR IF YES
3610
3611  010114  004767  011322              JSR   PC,CHPS2          ;GO LOCK OUT INTRS
3612  010120  016103  000010              MOV   BCR(R1),R3        ;GET THE WAS DATA
3613  010124  012711  004000              MOV   #BIT11,(R1)       ;CLEAR THE DH11
3614  010130  005004                      CLR   R4                ;SET UP S/B DATA
3615  010132  010102                      MOV   R1,R2             ;SET UP REGADR
3616  010134  062702  000010              ADD   #BCR,R2
3617  010140  004767  004764              JSR   PC,SUER1          ;GO SET UP THE ERROR INFO
3618  010144  004567  004674              JSR   R5,SUNUM          ;GO SET UP LINE NO. IN MSG
3619  010150  022152                      LINE
3620  010152  023025                      EM4+52
3621  010154  012767  010164  170726      MOV   #7$,$LPERR        ;SET UP ERROR LOOP RETURN
3622  010162  104004                      ERROR 4                 ;BYTE COUNT REG FAILED TO GO TO 000000
3623  010164  022626                  7$:   CMP   (SP)+,(SP)+     ;POP GOES THE STACK
3624  010166  005726                        TST   (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3625  010170  000167  176064              JMP   EXPAT3            ;GO ASK FOR NEW PATTERN
3626
3627  010174  016103  000006        8$:   MOV   CAR(R1),R3       ;GET THE WAS DATA
3628  010200  016704  011376              MOV   CHRCNT,R4         ;SET UP S/B DATA
3629  010204  005404                      NEG   R4
3630  010206  062704  032776              ADD   #TBUF,R4
3631  010212  020304                      CMP   R3,R4             ;WAS CAR CORRECT ??
3632  010214  001423                      BEQ   10$               ;BR IF YES
3633
3634  010216  004767  011220              JSR   PC,CHPS2          ;GO LOCK OUT INTRS
3635  010222  010102                      MOV   R1,R2             ;SET UP REGADR
3636  010224  062702  000006              ADD   #CAR,R2
3637  010230  004767  004674              JSR   PC,SUER1          ;GO SET UP ERROR INFO
3638  010234  004567  004604              JSR   R5,SUNUM          ;GO SET UP LINE NO. IN MSG
3639  010240  022152                      LINE
3640  010242  023107                      EM5+57
3641  010244  012767  010254  170636      MOV   #9$,$LPERR        ;SET UP THE ERROR RETURN
3642  010252  104005                      ERROR 5                 ;CURRENT ADDRESS REG NOT CORRECT
3643  010254  022626                  9$:   CMP   (SP)+,(SP)+     ;POP THE STACK
3644  010256  005726                        TST   (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
3645  010260  000167  175774              JMP   EXPAT3            ;GO ASK FOR NEW PATTERN
3646
3647  010264  000002                  10$:  RTI
```

```
                                        ;RECEIVER INTERRUPT SERVICE ROUTINE THREE
                                        ;
010266  032711  040000          RINT3:  BIT     #BIT14,(R1)     ;SILO OVERFLOW ERROR ??
010272  001427                          BEQ     2$              ;BR IF NOT
                                        ;
010274  004767  011142                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
010300  011103                          MOV     (R1),R3         ;GET WAS DATA
010302  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
010306  042703  177760                  BIC     #177760,R3      ;CLEAR JUNK
010312  116704  011634                  MOVB    LINE,R4
010316  004767  004606                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
010322  004567  004516                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
010326  022152                          LINE
010330  023156                          EM6+44
010332  012767  010342  170550          MOV     #1$,$LPERR      ;SET UP ERROR LOOP RETURN
010340  104006                          ERROR   6               ;SILO OVERFLOW - BAD,BAD,BAD !!!
010342  022626                  1$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
010344  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
010346  000167  175706                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
                                        ;
010352  105711                  2$:     TSTB    (R1)            ;CHAR AVAIL SET ??
010354  100432                          BMI     4$              ;BR IF YES
                                        ;
010356  004767  011060                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
010362  011103                          MOV     (R1),R3         ;GET WAS DATA
010364  042703  177560                  BIC     #177560,R3      ;CLEAN IT UP
010370  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR DH11
010374  012704  000200                  MOV     #BIT07,R4       ;SET UP S/B DATA
010400  156704  011546                  BISB    LINE,R4
010404  010102                          MOV     R1,R2           ;SET UP REGADR
010406  004767  004516                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
010412  004567  004426                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
010416  022152                          LINE
010420  023232                          EM7+51
010422  012767  010432  170460          MOV     #3$,$LPERR      ;SET UP THE ERROR LOOP RETURN
010430  104007                          ERROR   7               ;RECEIVER FALSE INTERRUPT
010432  022626                  3$:     CMP     (SP)+,(SP)+     ;POP GOES THE SP
010434  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
010436  000167  175616                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
                                        ;
010442  016167  000002  170532  4$:     MOV     NRC(R1),$TMP0   ;SAVE THE DATA RECEIVED
010450  100427                          BMI     6$              ;BR IF IT WAS VALID DATA
                                        ;
010452  004767  010764                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
010456  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
010462  162767  030516  020022          SUB     #RBUF,RBFPTR    ;WHICH CHAR WAS IT ??
010470  016702  020016                  MOV     RBFPTR,R2       ;SAVE CHAR NUMBER
010474  004767  004424                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
010500  004567  004340                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
010504  022152                          LINE
010506  023302                          EM10+45
010510  012767  010520  170372          MOV     #5$,$LPERR      ;SET UP ERROR RETURN
010516  104010                          ERROR   10              ;RECEIVED INVALID DATA
010520  022626                  5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
010522  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
010524  000167  175530                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
```

```
3704
3705  010530  016777  170446  017754  6$:   MOV   $TMP0,@RBFPTR   ;STORE CHAR IN THE BUFFER
3706  010536  062767  000001  017746        ADD   #1,RBFPTR       ;UPDATE THE POINTER
3707  010544  026767  011042  017740        CMP   RBFEND,RBFPTR   ;END OF BUFFER ??
3708  010552  001407                        BEQ   7$              ;BR IF YES
3709  010554  062767  000001  017730        ADD   #1,RBFPTR
3710  010562  026767  011024  017722        CMP   RBFEND,RBFPTR
3711  010570  001003                        BNE   8$              ;BR IF NOT DONE
3712  010572  052767  000001  011010  7$:   BIS   #1,RDONE        ;SET SOFTWARE DONE FLAG
3713  010600  000002            8$:   RTI                         ;RETURN TO WAIT LOOP
```

```
                                ;THIS ROUTINE IS CALLED TO CHECK THE RECEIVED DATA AND REPORT ALL ERRORS
                                ;FOR THE DATA PATTERNS TESTS
                
010602  012767  030516  017702  CKERDP: MOV     #RBUF,RBFPTR    ;SET UP POINTERS
010610  012767  032776  017676          MOV     #TBUF,TBFPTR
                
010616  117704  017672          1$:     MOVB    @TBFPTR,R4      ;GET THE S/B DATA
010622  000304                          SWAB    R4              ;PUT LINE NO. IN HIGH BYTE
010624  105004                          CLRB    R4
010626  156704  011320                  BISB    LINE,R4
010632  000304                          SWAB    R4
010634  052704  100000                  BIS     #BIT15,R4       ;AND FINALLY THE VALID DATA BIT
010640  046704  011524                  BIC     CLMSK,R4        ;MASK OFF BITS NOT XMITTED
010644  017703  017642                  MOV     @RBFPTR,R3      ;GET THE WAS DATA
010650  020304                          CMP     R3,R4           ;WAS = S/B ?????
010652  001425                          BEQ     3$              ;BR IF YES
                
010654  010367  170340                  MOV     R3,$TMP7        ;SAVE THE WAS DATA
010660  010146                          MOV     R1,-(SP)        ;SAVE THE DEVADR
010662  016701  017624                  MOV     RBFPTR,R1       ;GET THE SBADR
010666  016702  017622                  MOV     TBFPTR,R2       ;GET THE WASADR
010672  010200                          MOV     R2,R0           ;GET XMIT BUFFER ADDR
010674  162700  032776                  SUB     #TBUF,R0        ;GENERATE CHAR #
010700  004767  004220                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
010704  004567  004134                  JSR     R5,SUNUM        ;GO PUT LINE NO. IN MSG
010710  022152                          LINE
010712  023435                          EM11+23
010714  012767  010724  170166          MOV     #2$,$LPERR      ;SET UP ERROR RETURN
010722  104011                          ERROR   11              ;DATA COMPARE ERROR OR PARITY,FRAMING
                                                                ;OR OVERRUN
010724  012601                  2$:     MOV     (SP)+,R1        ;RESTORE THE DEVADR
                
010726  005267  017562          3$:     INC     TBFPTR          ;UPDATE POINTERS
010732  062767  000001  017552          ADD     #1,RBFPTR
010740  026767  010646  017544          CMP     RBFEND,RBFPTR   ;COMPARED ALL CHARS ??
010746  001407                          BEQ     4$              ;BR IF YES
010750  062767  000001  017534          ADD     #1,RBFPTR       ;UPDATE IT AGAIN
010756  026767  010630  017526          CMP     RBFEND,RBFPTR   ;DONE YET ?
010764  001314                          BNE     1$              ;BR IF NOT
                
010766  000207                  4$:     RTS     PC              ;RETURN TO WAIT LOOP
```

```
                                        .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

 010770                                 $EOP:
 010770 000004                                  SCOPE
 010772 005067 170104                           CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 010776 005067 170220                           CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 011002 005267 170232                           INC     $PASS           ;;INCREMENT THE PASS NUMBER
 011006 042767 100000 170224                    BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 011014 005327                                  DEC     (PC)+           ;;LOOP?
 011016 000001                          $EOPCT: .WORD   1
 011020 003022                                  BGT     $DOAGN          ;;YES
 011022 012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 011024 000001                          $ENDCT: .WORD   1
 011026 011016                                          $EOPCT
 011030 104401 011075                           TYPE    ,$ENDMG         ;;TYPE "END PASS #"
 011034 016746 170200                           MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 011040 104405                                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 011042 104401 011072                           TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 011046 013700 000042                   $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 011052 001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 011054 000005                                  RESET                   ;;CLEAR THE WORLD
 011056 004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 011060 000240                                  NOP                     ;;SAVE ROOM
 011062 000240                                  NOP                     ;;FOR
 011064 000240                                  NOP                     ;;ACT11
 011066                                 $DOAGN:
 011066 000137                                  JMP     @(PC)+          ;;RETURN
 011070 002336                          $RTNAD: .WORD   START2
 011072    377    377    000            $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 011075    015 042412 042116            $ENDMG: .ASCIZ  <15><12>/END PASS #/
 011102 050040 051501 020123
 011110 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

 011112                                 $SCOPE:
 011112 104407                                  CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
 011114 005067 011032                           CLR     LINE            ;INIT LINE COUNTER
 011120 032777 040000 170012            1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
 011126 001104                                  BNE     $OVER           ;;YES IF SW14=1
```

```
3812                                    ;#####START OF CODE FOR THE XOR TESTER#####
3813  011130  000416                    $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
3814                                                                    ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
3815  011132  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
3816  011136  012737  011156  000004            MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
3817  011144  005737  177060                    TST     @#177060        ;;TIME OUT ON XOR?
3818  011150  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
3819  011154  000453                    BR      $SVLAD          ;;GO TO THE NEXT TEST
3820  011156  022626                    5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
3821  011160  012637  000004                    MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
3822  011164  000413                    BR      7$              ;;LOOP ON THE PRESENT TEST
3823  011166                            6$:;#####END OF CODE FOR THE XOR TESTER#####
3824  011166  105767  167711            2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
3825  011172  001421                    BEQ     3$              ;;BR IF NO
3826  011174  126767  167715  167701            CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
3827  011202  101015                    BHI     3$              ;;BR IF NO
3828  011204  032777  001000  167726            BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
3829  011212  001404                    BEQ     4$              ;;BR IF NO
3830  011214  016767  167670  167664    7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
3831  011222  000446                    BR      $OVER
3832  011224  105067  167653            4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
3833  011230  005067  167766                    CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
3834  011234  000415                    BR      1$              ;;ESCAPE TO THE NEXT TEST
3835  011236  032777  004000  167674    3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
3836  011244  001011                    BNE     1$              ;;BR IF YES
3837  011246  005767  167766                    TST     $PASS           ;;IF FIRST PASS OF PROGRAM
3838  011252  001406                    BEQ     1$              ;;      INHIBIT ITERATIONS
3839  011254  005267  167624                    INC     $ICNT           ;;INCREMENT ITERATION COUNT
3840  011260  026767  167736  167616            CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
3841  011266  002024                    BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
3842  011270  012767  000001  167606    1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
3843  011276  016767  000052  167716            MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
3844  011304  105267  167572            $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
3845  011310  116767  167566  167720            MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
3846  011316  011667  167564                    MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
3847  011322  011667  167562                    MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
3848  011326  005067  167672                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
3849  011332  112767  000001  167555            MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3850  011340  016777  167536  167574    $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
3851  011346  016716  167534                    MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
3852  011352  000002                    RTI                             ;;FIXES PS
3853  011354  000010                    $MXCNT: 10                      ;;MAX. NUMBER OF ITERATIONS
3854                                    .SBTTL  ERROR HANDLER ROUTINE
3855
3856                                    ;;******************************************************************
3857                                    ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3858                                    ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3859                                    ;*AND GO TO $ERRTYP ON ERROR
3860                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3861                                    ;*SW15=1         HALT ON ERROR
3862                                    ;*SW13=1         INHIBIT ERROR TYPEOUTS
3863                                    ;*SW09=1         LOOP ON ERROR
3864                                    ;*CALL
3865                                    ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3866
3867  011356                            $ERROR:
```

```
                    CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
011356  104407
011360  105267  167517      7$:   INCB    $ERFLG          ;;SET THE ERROR FLAG
011364  001775                    BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
011366  016777  167510  167546    MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
011374  005267  167512            INC     $ERTTL          ;;INC THE ERROR COUNT
011400  011667  167512            MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
011404  162767  000002  167504    SUB     #2,$ERRPC
011412  117767  167500  167474    MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
011420  032777  020000  167512    BIT     #BIT13,@$SWR    ;;SKIP TYPEOUT IF SET
011426  001004                    BNE     20$             ;;SKIP TYPEOUTS
011430  004767  000074            JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
011434  104401  001227            TYPE    ,$CRLF
011440                      20$:
011440  122767  000001  167604    CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
011446  001007                    BNE     2$              ;;NO,SKIP APT ERROR REPORT
011450  116767  167440  000004    MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
011456  004767  001174            JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
011462     000      21$:   .BYTE   0
011463     000             .BYTE   0
011464  000777      22$:   BR      22$             ;;APT ERROR LOOP
011466  005777  167446      2$:   TST     @$SWR           ;;HALT ON ERROR
011472  100002                    BPL     3$              ;;SKIP IF CONTINUE
011474  000000                    HALT                    ;;HALT ON ERROR!
011476  104407                    CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
011500  032777  001000  167432  3$:   BIT     #BIT09,@$SWR    ;;LOOP ON ERROR SWITCH SET?
011506  001402                    BEQ     4$              ;;BR IF NO
011510  016716  167374            MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
011514  005767  167504      4$:   TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
011520  001402                    BEQ     5$              ;;BR IF NONE
011522  016716  167476            MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
011526                      5$:
011526  000002                    RTI                     ;;RETURN
                    .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                    ;;**************************************************************
                    ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                    ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                    ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.

011530              $ERRTYP:
011530  104401  001227            TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
011534  010046                    MOV     R0,-(SP)        ;;SAVE R0
011536  005000                    CLR     R0              ;;PICKUP THE ITEM INDEX
011540  153700  001114            BISB    @#$ITEMB,R0
011544  001004                    BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
                                                        ;;TYPE THE PC OF THE ERROR
011546  016746  167344            MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                        ;;ERROR ADDRESS
011552  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
011554  000445                    BR      10$             ;;GET OUT
011556  005300      1$:   DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
011560  006300                    ASL     R0              ;;      WORK FOR THE ERROR TABLE
011562  006300                    ASL     R0
011564  006300                    ASL     R0
011566  062700  001356            ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
011572  012067  000004            MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
```

```
3924  011576  001404                  BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
3925  011600  104401                  TYPE                    ;;TYPE THE "ERROR MESSAGE"
3926  011602  000000            2$:   .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
3927  011604  104401  001227        TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
3928  011610  012067  000004  3$:   MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
3929  011614  001404                  BEQ     5$              ;;SKIP TYPEOUT IF 0
3930  011616  104401                  TYPE                    ;;TYPE THE "DATA HEADER"
3931  011620  000000            4$:   .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
3932  011622  104401  001227        TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
3933  011626  010146            5$:   MOV     R1,-(SP)        ;;SAVE R1
3934  011630  012001                  MOV     (R0)+,R1        ;;PICKUP "DATA TABLE" POINTER
3935  011632  001415                  BEQ     9$              ;;BR IF NO DATA TO BE TYPED
3936  011634  012000                  MOV     (R0)+,R0        ;;PICKUP "DATA FORMAT" POINTER
3937  011636  105720            6$:   TSTB    (R0)+           ;;"OCTAL" OR "DECIMAL"
3938  011640  001003                  BNE     7$              ;;BR IF DECIMAL
3939  011642  013146                  MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3940  011644  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3941  011646  000402                  BR      8$
3942  011650                    7$:
3943  011650  013146                  MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
3944  011652  104405                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
3945  011654  005711            8$:   TST     (R1)            ;;IS THERE ANOTHER NUMBER?
3946  011656  001403                  BEQ     9$              ;;BR IF NO
3947  011660  104401  011700        TYPE    ,11$            ;;TYPE TWO(2) SPACES
3948  011664  000764                  BR      6$              ;;LOOP
3949
3950  011666  012601            9$:   MOV     (SP)+,R1        ;;RESTORE R1
3951  011670  012600            10$:  MOV     (SP)+,R0        ;;RESTORE R0
3952  011672  104401  001227        TYPE    $CRLF           ;;"CARRIAGE RETURN" & "LINE FEED"
3953  011676  000207                  RTS     PC              ;;RETURN
3954  011700  020040      000   11$:  .ASCIZ  / /             ;;TWO(2) SPACES
3955          011704                .EVEN
3956                            .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3957
3958                            ;;**************************************************************
3959                            ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3960                            ;*OCTAL (ASCII) NUMBER AND TYPE IT.
3961                            ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3962                            ;*CALL:
3963                            ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3964                            ;*      TYPOS                   ;;CALL FOR TYPEOUT
3965                            ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3966                            ;*      .BYTE   M               ;;M=1 OR 0
3967                            ;*                                      ;;1=TYPE LEADING ZEROS
3968                            ;*                                      ;;0=SUPPRESS LEADING ZEROS
3969                            ;*
3970                            ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3971                            ;*$TYPOS OR $TYPOC
3972                            ;*CALL:
3973                            ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3974                            ;*      TYPON                   ;;CALL FOR TYPEOUT
3975                            ;*
3976                            ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3977                            ;*CALL:
3978                            ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3979                            ;*      TYPOC                   ;;CALL FOR TYPEOUT
```

```
 011704  017646  000000          $TYPOS: MOV     @(SP),-(SP)             ;;PICKUP THE MODE
 011710  116667  000001  000211          MOVB    1(SP),$0FILL            ;;LOAD ZERO FILL SWITCH
 011716  112667  000207                  MOVB    (SP)+,$OMODE+1          ;;NUMBER OF DIGITS TO TYPE
 011722  062716  000002                  ADD     #2,(SP)                 ;;ADJUST RETURN ADDRESS
 011726  000406                          BR      $TYPON
 011730  112767  000001  000171  $TYPOC: MOVB    #1,$0FILL               ;;SET THE ZERO FILL SWITCH
 011736  112767  000006  000165          MOVB    #6,$OMODE+1             ;;SET FOR SIX(6) DIGITS
 011744  112767  000005  000154  $TYPON: MOVB    #5,$OCNT                ;;SET THE ITERATION COUNT
 011752  010346                          MOV     R3,-(SP)                ;;SAVE R3
 011754  010446                          MOV     R4,-(SP)                ;;SAVE R4
 011756  010546                          MOV     R5,-(SP)                ;;SAVE R5
 011760  116704  000145                  MOVB    $OMODE+1,R4             ;;GET THE NUMBER OF DIGITS TO TYPE
 011764  005404                          NEG     R4
 011766  062704  000006                  ADD     #6,R4                   ;;SUBTRACT IT FOR MAX. ALLOWED
 011772  110467  000132                  MOVB    R4,$OMODE               ;;SAVE IT FOR USE
 011776  116704  000125                  MOVB    $0FILL,R4               ;;GET THE ZERO FILL SWITCH
 012002  016605  000012                  MOV     12(SP),R5               ;;PICKUP THE INPUT NUMBER
 012006  005003                          CLR     R3                      ;;CLEAR THE OUTPUT WORD
 012010  006105                  1$:     ROL     R5                      ;;ROTATE MSB INTO "C"
 012012  000404                          BR      3$                      ;;GO DO MSB
 012014  006105                  2$:     ROL     R5                      ;;FORM THIS DIGIT
 012016  006105                          ROL     R5
 012020  006105                          ROL     R5
 012022  010503                          MOV     R5,R3
 012024  006103                  3$:     ROL     R3                      ;;GET LSB OF THIS DIGIT
 012026  105367  000076                  DECB    $OMODE                  ;;TYPE THIS DIGIT?
 012032  100016                          BPL     7$                      ;;BR IF NO
 012034  042703  177770                  BIC     #177770,R3              ;;GET RID OF JUNK
 012040  001002                          BNE     4$                      ;;TEST FOR 0
 012042  005704                          TST     R4                      ;;SUPPRESS THIS 0?
 012044  001403                          BEQ     5$                      ;;BR IF YES
 012046  005204                  4$:     INC     R4                      ;;DON'T SUPPRESS ANYMORE 0'S
 012050  052703  000060                  BIS     #'0,R3                  ;;MAKE THIS DIGIT ASCII
 012054  052703  000040          5$:     BIS     #' ,R3                  ;;MAKE ASCII IF NOT ALREADY
 012060  110367  000040                  MOVB    R3,8$                   ;;SAVE FOR TYPING
 012064  104401  012124                  TYPE    ,8$                     ;;GO TYPE THIS DIGIT
 012070  105367  000032          7$:     DECB    $OCNT                   ;;COUNT BY 1
 012074  003347                          BGT     2$                      ;;BR IF MORE TO DO
 012076  002402                          BLT     6$                      ;;BR IF DONE
 012100  005204                          INC     R4                      ;;INSURE LAST DIGIT ISN'T A BLANK
 012102  000744                          BR      2$                      ;;GO DO THE LAST DIGIT
 012104  012605                  6$:     MOV     (SP)+,R5                ;;RESTORE R5
 012106  012604                          MOV     (SP)+,R4                ;;RESTORE R4
 012110  012603                          MOV     (SP)+,R3                ;;RESTORE R3
 012112  016666  000002  000004          MOV     2(SP),4(SP)             ;;SET THE STACK FOR RETURNING
 012120  012616                          MOV     (SP)+,(SP)
 012122  000002                          RTI                             ;;RETURN
 012124     000                  8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
 012125     000                          .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
 012126     000                  $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
 012127     000                  $0FILL: .BYTE   0                       ;;ZERO FILL SWITCH
 012130  000000                  $OMODE: .WORD   0                       ;;NUMBER OF DIGITS TO TYPE
                                 .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE


                                 ;;*******************************************************************
```

```
                                        ;*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
                                        $TYPDS:
012132                                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
012132  010046                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
012134  010146                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
012136  010246                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
012140  010346                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
012142  010546                                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
012144  012746  020200                         MOV     20(SP),R5       ;;GET THE INPUT NUMBER
012150  016605  000020                         BPL     1$              ;;BR IF INPUT IS POS.
012154  100004                                  NEG     R5              ;;MAKE THE BINARY NUMBER POS.
012156  005405                                  MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
012160  112766  000055  000001                  CLR     R0              ;;ZERO THE CONSTANTS INDEX
012166  005000                          1$:     MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
012170  012703  012346                         MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
012174  112723  000040                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
012200  005002                          3$:     MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
012202  016001  012336                         SUB     R1,R5           ;;FORM THIS BCD DIGIT
012206  160105                          4$:     BLT     4$              ;;BR IF DONE
012210  002402                                  INC     R2              ;;INCREASE THE BCD DIGIT BY 1
012212  005202                                  BR      3$
012214  000774                          5$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
012216  060105                                  TST     R2              ;;CHECK IF BCD DIGIT=0
012220  005702                                  BNE     5$              ;;FALL THROUGH IF 0
012222  001002                                  TSTB    (SP)            ;;STILL DOING LEADING 0'S?
012224  105716                                  BMI     7$              ;;BR IF YES
012226  100407                          6$:     ASLB    (SP)            ;;MSD?
012230  106316                                  BCC     6$              ;;BR IF NO
012232  103003                                  MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
012234  116663  000001  177777                  BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
012242  052702  000060                  7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
012246  052702  000040                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
012252  110223                                  TST     (R0)+           ;;JUST INCREMENTING
012254  005720                                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
012256  020027  000010                         BLT     2$              ;;GO DO THE NEXT DIGIT
012262  002746                                  BGT     8$              ;;GO TO EXIT
012264  003002                                  MOV     R5,R2           ;;GET THE LSD
012266  010502                                  BR      6$              ;;GO CHANGE TO ASCII
012270  000764                          8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
012272  105726                                  BPL     9$              ;;BR IF NO
012274  100003                                  MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
012276  116663  177777  177776          9$:     CLRB    (R3)            ;;SET THE TERMINATOR
012304  105013                                  MOV     (SP)+,R5        ;;POP STACK INTO R5
012306  012605                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
012310  012603                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
012312  012602                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
012314  012601                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
012316  012600                                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
012320  104401  012346
```

```
                                        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
                                        MOV     (SP)+,(SP)
                                        RTI                     ;;RETURN TO USER
                                $DTBL:  10000.
                                        1000.
                                        100.
                                        10.
                                $DBLK:  .BLKW   4
                                .SBTTL  TYPE ROUTINE

                                ;;**************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*      TYPE
                                ;*      MESADR
                                ;*

012356  105767  166575  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
012362  100002                 BPL     1$              ;;BR IF YES
012364  000000                 HALT                    ;;HALT HERE IF NO TERMINAL
012366  000430                 BR      3$              ;;LEAVE
012370  010046          1$:    MOV     R0,-(SP)        ;;SAVE R0
012372  017600  000002         MOV     2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
012376  122767  000001  166646 CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
012404  001011                 BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
012406  132767  000100  166637 BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
012414  001405                 BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
012416  010067  000004         MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
012422  004767  000220         JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
012426  000000          61$:   .WORD   0               ;;MESSAGE ADDRESS
012430  132767  000040  166615 62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
012436  001003                 BNE     60$             ;;YES,SKIP TYPE OUT
012440  112046          2$:    MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
012442  001005                 BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
012444  005726                 TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
012446  012600          60$:   MOV     (SP)+,R0        ;;RESTORE R0
012450  062716  000002         3$:    ADD     #2,(SP)         ;;ADJUST RETURN PC
012454  000002                 RTI                     ;;RETURN
012456  122716  000011         4$:    CMPB    #HT,(SP)        ;;BRANCH IF <HT>
012462  001430                 BEQ     8$
012464  122716  000200                 CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
012470  001006                 BNE     5$
012472  005726                 TST     (SP)+           ;;POP  <CR><LF> EQUIV
012474  104401                 TYPE                    ;;TYPE A CR AND LF
012476  001227                 $CRLF
012500  105067  000130         CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
012504  000755                 BR      2$              ;;GET NEXT CHARACTER
012506  004767  000056         5$:    JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
```

```
4148  012512  126726  166440          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
4149  012516  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
4150  012520  016746  166430                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
4151                                                                  ;;AND THE NULL CHAR.
4152  012524  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
4153  012530  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
4154  012532  004767  000032                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
4155  012536  105367  000072                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
4156  012542  000770                          BR      7$              ;;LOOP
4157
4158                                  ;HORIZONTAL TAB PROCESSOR
4159
4160  012544  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
4161  012550  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
4162  012554  132767  000007  000052          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
4163  012562  001372                          BNE     9$              ;;TAB STOP
4164  012564  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
4165  012566  000724                          BR      2$              ;;GET NEXT CHARACTER
4166  012570  105777  166354          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
4167  012574  100375                          BPL     $TYPEC
4168  012576  116677  000002  166346          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
4169  012604  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
4170  012612  001003                          BNE     1$              ;;BRANCH IF NO
4171  012614  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
4172  012620  000406                          BR      $TYPEX          ;;EXIT
4173  012622  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
4174  012630  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
4175  012632  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
4176  012634  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
4177  012636  000207          $TYPEX: RTS     PC
4178
4179                                  .SBTTL  APT COMMUNICATIONS ROUTINE
4180
4181                                  ;;**************************************************************************
4182  012640  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
4183  012646  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
4184  012654  000403                          BR      $ATYC
4185  012656  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
4186  012664                          $ATYC:
4187  012664  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4188  012666  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4189  012670  105767  000206                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
4190  012674  001450                          BEQ     5$              ;;IF NOT:  BR
4191  012676  122767  000001  166346          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
4192  012704  001031                          BNE     3$              ;;IF NOT:  BR
4193  012706  132767  000100  166337          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
4194  012714  001425                          BEQ     3$              ;;IF NOT:  BR
4195  012716  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
4196  012722  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
4197  012730  005767  166276          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
4198  012734  001375                          BNE     1$              ;;IF NOT:  WAIT
4199  012736  010067  166304                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
4200  012742  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
4201  012744  001376                          BNE     2$
4202  012746  166700  166274                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
4203  012752  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
```

```
4204  012754  010067  166270              MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
4205  012760  012767  000004  166244      MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
4206  012766  000413                      BR      5$
4207  012770  017667  000004  000016 3$:  MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
4208  012776  062766  000002  000004      ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
4209  013004  016746  164766              MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
4210  013010  004767  177342              JSR     PC,$TYPE        ;;CALL TYPE MACRO
4211  013014  000000              4$:     .WORD   0
4212  013016                      5$:
4213  013016  105767  000062      10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
4214  013022  001416                      BEQ     12$             ;;IF NOT:  BR
4215  013024  005767  166222              TST     $ENV            ;;RUNNING UNDER APT?
4216  013030  001413                      BEQ     12$             ;;IF NOT:  BR
4217  013032  005767  166174      11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
4218  013036  001375                      BNE     11$             ;;IF NOT:  WAIT
4219  013040  017667  000004  166166      MOV     @4(SP),$FATAL   ;;GET ERROR #
4220  013046  062766  000002  000004      ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
4221  013054  005267  166152              INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
4222  013060  105067  000020      12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
4223  013064  105067  000013              CLRB    $LFLG           ;;CLEAR LOG FLAG
4224  013070  105067  000006              CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
4225  013074  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
4226  013076  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
4227  013100  000207                      RTS     PC              ;;RETURN
4228  013102     000  $MFLG:  .BYTE   0          ;;MESSG. FLAG
4229  013103     000  $LFLG:  .BYTE   0          ;;LOG FLAG
4230  013104     000  $FFLG:  .BYTE   0          ;;FATAL FLAG
4231          013106          .EVEN
4232          000200  APTSIZE=200
4233          000001  APTENV=001
4234          000100  APTSPOOL=100
4235          000040  APTCSUP=040
4236                  .SBTTL  TTY INPUT ROUTINE
4237
4238                  ;;**********************************************************************
4239                  .ENABL  LSB
4240
4241                  ;;**********************************************************************
4242                  ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
4243                  ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
4244                  ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
4245                  ;*WHEN OPERATING IN TTY FLAG MODE.
4246  013106  022767  000176  166024 $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
4247  013114  001074                      BNE     15$             ;;BRANCH IF NO
4248  013116  105777  166022              TSTB    @$TKS           ;;CHAR THERE?
4249  013122  100071                      BPL     15$             ;;IF NO, DON'T WAIT AROUND
4250  013124  117746  166016              MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
4251  013130  042716  177600              BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
4252  013134  022726  000007              CMP     #7,(SP)+        ;;IS IT A CONTROL G?
4253  013140  001062                      BNE     15$             ;;NO, RETURN TO USER
4254  013142  126727  165766  000001      CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
4255  013150  001456                      BEQ     15$             ;;BRANCH IF YES
4256
4257  013152  104401  013761              TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
4258  013156  104401  013766 $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
4259  013162  016746  165010              MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
```

```
4260  013166  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4261  013170  104401  013777                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
4262  013174  005046                19$:     CLR     -(SP)           ;;CLEAR COUNTER
4263  013176  005046                        CLR     -(SP)           ;;THE NEW SWR
4264  013200  105777  165740        7$:     TSTB    @$TKS           ;;CHAR THERE?
4265  013204  100375                        BPL     7$              ;;IF NOT TRY AGAIN
4266
4267  013206  117746  165734                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
4268  013212  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
4269
4270
4271
4272  013216  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
4273  013222  001005                        BNE     10$             ;;BRANCH IF NOT
4274  013224  104401  013754                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
4275  013230  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
4276  013234  000757                        BR      19$             ;;LET'S TRY IT AGAIN
4277
4278
4279  013236  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
4280  013242  001022                        BNE     16$             ;;BRANCH IF NO
4281  013244  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
4282  013250  001403                        BEQ     11$             ;;BRANCH IF YES
4283  013252  016677  000002  165660        MOV     2(SP),@SWR      ;;SAVE NEW SWR
4284  013260  062706  000006        11$:    ADD     #6,SP           ;;CLEAR UP STACK
4285  013264  104401  001227        14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
4286  013270  126727  165641  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
4287  013276  001003                        BNE     15$             ;;BRANCH IF NOT
4288  013300  012777  000100  165636        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
4289  013306  000002                15$:    RTI                     ;;RETURN
4290  013310  004767  177254        16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
4291  013314  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
4292  013320  002420                        BLT     18$             ;;BRANCH IF YES
4293  013322  021627  000067                CMP     (SP),#67        ;;CHAR > 7?
4294  013326  003015                        BGT     18$             ;;BRANCH IF YES
4295  013330  042726  000060                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
4296  013334  005766  000002                TST     2(SP)           ;;IS THIS THE FIRST CHAR
4297  013340  001403                        BEQ     17$             ;;BRANCH IF YES
4298  013342  006316                        ASL     (SP)            ;;NO, SHIFT PRESENT
4299  013344  006316                        ASL     (SP)            ;;    CHAR OVER TO MAKE
4300  013346  006316                        ASL     (SP)            ;;    ROOM FOR NEW ONE.
4301  013350  005266  000002        17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
4302  013354  056616  177776                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
4303  013360  000707                        BR      7$              ;;GET THE NEXT ONE
4304  013362  104401  001226        18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
4305  013366  000720                        BR      20$             ;;SIMULATE CONTROL-U
4306                                .DSABL  LSB
4307
4308
4309                                ;;**********************************************************************
4310                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
4311                                ;*CALL:
4312                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
4313                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
4314                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
4315                                ;
```

```
4316
4317  013370  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
4318  013372  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
4319  013400  105777  165540          1$:     TSTB    @$TKS           ;;WAIT FOR
4320  013404  100375                          BPL     1$              ;;A CHARACTER
4321  013406  117766  165534  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
4322  013414  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
4323  013422  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
4324  013430  001013                          BNE     3$              ;;BRANCH IF NO
4325  013432  105777  165506          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
4326  013436  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
4327  013440  117746  165502                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
4328  013444  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
4329  013450  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
4330  013454  001366                          BNE     2$              ;;IF NOT DISCARD IT
4331  013456  000750                          BR      1$              ;;YES, RESUME
4332  013460  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
4333  013466  002407                          BLT     4$              ;;BRANCH IF YES
4334  013470  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
4335  013476  003003                          BGT     4$              ;;BRANCH IF YES
4336  013500  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
4337  013506  000002                  4$:     RTI                     ;;GO BACK TO USER
4338                                  ;;**************************************************************
4339                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
4340                                  ;*CALL:
4341                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
4342                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
4343                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
4344
4345  013510  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
4346  013512  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
4347  013514  012703  013744          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
4348  013520  022703  013754          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
4349  013524  101456                          BLOS    4$              ;;BR IF YES
4350  013526  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
4351  013530  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
4352  013532  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
4353  013536  001022                          BNE     5$              ;;BR IF NO
4354  013540  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
4355  013542  001007                          BNE     6$              ;;BR IF NO
4356  013544  112767  000134  000170          MOVB    #'\,9$          ;;TYPE A BACK SLASH
4357  013552  104401  013742                  TYPE    ,9$
4358  013556  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
4359  013562  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
4360  013564  020327  013744                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
4361  013570  103434                          BLO     4$              ;;BR IF YES
4362  013572  111367  000144                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
4363  013576  104401  013742                  TYPE    ,9$             ;;GO TYPE
4364  013602  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
4365  013604  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
4366  013606  001406                          BEQ     7$              ;;BR IF NO
4367  013610  112767  000134  000124          MOVB    #'\,9$          ;;TYPE A BACK SLASH
4368  013616  104401  013742                  TYPE    ,9$
4369  013622  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
4370  013624  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
4371  013630  001003                          BNE     8$              ;;BR IF NO
```

```
 4372  013632  104401  013754            TYPE    $CNTLU          ;;TYPE A CONTROL "U"
 4373  013636  000726                    BR      1$              ;;GO START OVER
 4374  013640  122713  000022    8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
 4375  013644  001011                    BNE     3$              ;;BRANCH IF NO
 4376  013646  105013                    CLRB    (R3)            ;;CLEAR THE CHARACTER
 4377  013650  104401  001227            TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
 4378  013654  104401  013744            TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
 4379  013660  000717                    BR      2$              ;;GO PICKUP ANOTHER CHACTER
 4380  013662  104401  001226    4$:     TYPE    ,$QUES          ;;TYPE A '?'
 4381  013666  000712                    BR      1$              ;;CLEAR THE BUFFER AND LOOP
 4382  013670  111367  000046    3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 4383  013674  104401  013742            TYPE    ,9$
 4384  013700  122723  000015            CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 4385  013704  001305                    BNE     2$              ;;LOOP IF NOT RETURN
 4386  013706  105063  177777            CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 4387  013712  104401  001230            TYPE    ,$LF            ;;TYPE A LINE FEED
 4388  013716  005726                    TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
 4389  013720  012603                    MOV     (SP)+,R3        ;;RESTORE R3
 4390  013722  011646                    MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 4391  013724  016666  000004  000002    MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 4392  013732  012766  013744  000004    MOV     #$TTYIN,4(SP)
 4393  013740  000002                    RTI                     ;;RETURN
 4394  013742     000            9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
 4395  013743     000                    .BYTE   0               ;;TERMINATOR
 4396  013744  000010            $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
 4397  013754  052536  005015     000    $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
 4398  013761     136  006507  000012    $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
 4399  013766  005015  053523  020122    $MSWR:  .ASCIZ  <15><12>/SWR = /
 4400  013774  020075     000
 4401  013777     040  047040  053505    $MNEW:  .ASCIZ  /  NEW = /
 4402  014004  036440  000040
 4403                                    .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
 4404
 4405                                    ;;**********************************************************************
 4406                                    ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
 4407                                    ;*CHANGE IT TO BINARY.
 4408                                    ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
 4409                                    ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
 4410                                    ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
 4411                                    ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
 4412                                    ;*CALL:
 4413                                    ;*      RDOCT                   ;;READ AN OCTAL NUMBER
 4414                                    ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
 4415                                    ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
 4416
 4417  014010  011646            $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
 4418  014012  016666  000004  000002    MOV     4(SP),2(SP)     ;;INPUT NUMBER
 4419  014020  010046                    MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 4420  014022  010146                    MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 4421  014024  010246                    MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 4422  014026  104411            1$:     RDLIN                   ;;READ AN ASCIZ LINE
 4423  014030  012600                    MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
 4424  014032  010067  000100            MOV     R0,5$           ;;AND SAVE IT
 4425  014036  005001                    CLR     R1              ;;CLEAR DATA WORD
 4426  014040  005002                    CLR     R2
 4427  014042  112046            2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
```

```
                        BEQ     3$              ;;IF ZERO GET OUT
                        CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
                        BGT     4$              ;;IS AN OCTAL DIGIT
                        CMPB    #'7,(SP)
                        BLT     4$
                        ASL     R1              ;;*2
                        ROL     R2
                        ASL     R1              ;;*4
                        ROL     R2
                        ASL     R1              ;;*8
                        ROL     R2
                        BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
                        ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
                        BR      2$              ;;LOOP
                3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
                        MOV     R1,12(SP)       ;;SAVE THE RESULT
                        MOV     R2,$HIOCT
                        MOV     (SP)+,R2        ;;POP STACK INTO R2
                        MOV     (SP)+,R1        ;;POP STACK INTO R1
                        MOV     (SP)+,R0        ;;POP STACK INTO R0
                        RTI                     ;;RETURN
                4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
                        CLRB    (R0)            ;;SET A TERMINATOR
                        TYPE                    ;;TYPE UP THRU THE BAD CHAR.
                5$:     .WORD   0
                        TYPE    $QUES           ;;"?" "CR" & "LF"
                        BR      1$              ;;TRY AGAIN
                $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
                .SBTTL  READ A DECIMAL NUMBER FROM THE TTY

                ;;**************************************************************
                ;*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
                ;*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
                ;*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
                ;*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
                ;*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
                ;*POSITIVE 32767 TO NEGATIVE 32768.
                ;*CALL:
                ;*      RDDEC                   ;;READ A DECIMAL NUMBER
                ;*      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
                ;
                $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
                        MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                1$:     RDLIN                   ;;READ AN ASCIZ LINE
                        MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
                        MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
                        CLR     -(SP)           ;;CLEAR DATA WORD
                        CLR     R2              ;;SIGN SET POSITIVE
                        CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
                        BNE     2$              ;;BR IF NO MINUS SIGN
                        MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
                2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
```

```
4484  014214  001424                        BEQ     3$              ;;GET OUT IF ZERO
4485  014216  122701  000060                CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
4486  014222  003032                        BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
4487  014224  122701  000071                CMPB    #'9,R1
4488  014230  002427                        BLT     5$
4489  014232  032716  170000                BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
4490  014236  001024                        BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
4491  014240  006316                        ASL     (SP)            ;;*2
4492  014242  011646                        MOV     (SP),-(SP)      ;;SAVE FOR LATER
4493  014244  006316                        ASL     (SP)            ;;*4
4494  014246  006316                        ASL     (SP)            ;;*8.
4495  014250  062616                        ADD     (SP)+,(SP)      ;;*10.
4496  014252  102416                        BVS     5$              ;;OVERFLOW ISN'T ALLOWED
4497  014254  162701  000060                SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
4498  014260  060116                        ADD     R1,(SP)         ;;ADD IN THIS DIGIT
4499  014262  102412                        BVS     5$              ;;OVERFLOW ISN'T ALLOWED
4500  014264  000752                        BR      2$              ;;LOOP
4501  014266  005702                3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
4502  014270  001401                        BEQ     4$              ;;BR IF NO
4503  014272  005416                        NEG     (SP)            ;;YES--NEGATE THE NUMBER
4504  014274  012666  000012        4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
4505  014300  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
4506  014302  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
4507  014304  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
4508  014306  000002                        RTI                     ;;RETURN
4509
4510  014310  005726                5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
4511  014312  105010                        CLRB    (R0)            ;;SET A TERMINATOR
4512  014314  104401                        TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
4513  014316  000000                6$:     .WORD   0               ;;POINTER GOES HERE
4514  014320  104401  001226                TYPE    ,$QUES          ;;"?" "CR" &"LF"
4515  014324  000720                        BR      1$              ;;TRY AGAIN
4516                                .SBTTL  TRAP DECODER
4517
4518                                ;;**************************************************************
4519                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
4520                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
4521                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
4522                                ;*GO TO THAT ROUTINE.
4523
4524  014326  010046                $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
4525  014330  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
4526  014334  005740                        TST     -(R0)           ;;BACKUP BY 2
4527  014336  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
4528  014340  006300                        ASL     R0              ;;POSITION FOR INDEXING
4529  014342  016000  014362                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
4530  014346  000200                        RTS     R0              ;;GO TO ROUTINE
4531
4532
4533                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
4534
4535  014350  011646                $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
4536  014352  016666  000004  000002        MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
4537  014360  000002                        RTI                     ;;RESTORE THE PSW
4538
4539                                .SBTTL  TRAP TABLE
```

```
                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE "TRAP" INSTRUCTION.
                                ;
                                ;       ROUTINE
                                ;       -------
014362  014350                  $TRPAD: .WORD   $TRAP2
014364  012356                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
014366  011730                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
014370  011704                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
014372  011744                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
014374  012132                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

014376  013156                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

014400  013106                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
014402  013370                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
014404  013510                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
014406  014010                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
014410  014150                          $RDDEC  ;;CALL=RDDEC    TRAP+13(104413) READ A DECIMAL NUMBER FROM TTY
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;**************************************************************
                                ;POWER DOWN ROUTINE
014412  012737  014556  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
014420  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
014426  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
014430  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
014432  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
014434  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
014436  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
014440  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
014442  017746  164472                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
014446  010667  000110                  MOV     SP,$SAVR6       ;;SAVE SP
014452  012737  014464  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
014460  000000                          HALT
014462  000776                          BR      .-2             ;;HANG UP

                                ;;**************************************************************
                                ;POWER UP ROUTINE
014464  012737  014556  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
014472  016706  000064                  MOV     $SAVR6,SP       ;;GET SP
014476  005067  000060                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
014502  005267  000054          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
014506  001375                          BNE     1$              ;;OF  WORD
014510  012677  164424                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
014514  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
014516  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
014520  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
014522  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
014524  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
014526  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
014530  012737  014412  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
014536  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
014544  104401                          TYPE                    ;REPORT THE POWER FAILURE
014546  014564                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
```

```
 4596  014550  012716                    MOV     (PC)+,(SP)      ;;RESTART AT CKRST1
 4597  014552  014574            $PWRAD: .WORD   CKRST1          ;;RESTART ADDRESS
 4598  014554  000002                    RTI
 4599  014556  000000            $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
 4600  014560  000776                    BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
 4601  014562  000000            $SAVR6: 0                       ;;PUT THE SP HERE
 4602  014564  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
 4603  014572  000122
 4604                                    .EVEN
```

```
                        ;;*****************************************************************
                        ;COMMON DH11 SERVICE ROUTINES
                        ;;*****************************************************************

                        ;THESE ROUTINES DETERMINE RESTART ADDRESS AFTER SYSTEM ERROR
                        ;(BUS ERROR,RSVD INSTR ERROR, OR POWER FAIL)

014574 005767 005546    CKRST1: TST     DPFLG           ;IN PATTERNS TEST ?
014600 001005                   BNE     1$              ;BR IF YES
014602 005767 005552            TST     RETFLG          ;IN ECHO TEST ?
014606 001004                   BNE     2$              ;BR IF YES
014610 000167 165626            JMP     RSTRTA          ;GO RESTART RELIABILITY TESTS
014614 000167 171266    1$:     JMP     EXPAT           ;GO TO PATTERNS TESTS
014620 000167 170062    2$:     JMP     ECHO            ;GO TO ECHO TESTS

014624 005767 005516    CKRST2: TST     DPFLG           ;IN PATTERNS TEST ?
014630 001005                   BNE     1$              ;BR IF YES
014632 005767 005522            TST     RETFLG          ;IN ECHO TEST ?
014636 001004                   BNE     2$              ;BR IF YES
014640 000167 165566            JMP     REST1           ;GO RESTART RELIABILITY TESTS
014644 000167 171236    1$:     JMP     EXPAT           ;GO TO PATTERNS TESTS
014650 000167 170032    2$:     JMP     ECHO            ;GO TO ECHO TESTS


                        ;THIS ROUTINE IS CALLED TO SET UP THE DH11 PARAMETERS PRIOR TO TEST

014654 012711 004000    DHSET1: MOV     #BIT11,(R1)     ;CLEAR THE DH11 UNDER TEST
014660 004767 004556            JSR     PC,CHPS2        ;GO LOCK OUT INTRS
014664 012711 030100            MOV     #30100,(R1)     ;ENABLE INTERRUPTS ON XMIT DONE
                                                        ;NON-EX MEM, DATA AVAIL, OR SILO OVFLW
014670 156711 005256            BISB    LINE,(R1)       ;SELECT THE LINE NO.
014674 005067 004710            CLR     RDONE           ;CLEAR SOFTWARE DONE FLAG
014700 012767 030516 013604     MOV     #RBUF,RBFPTR    ;SET UP RCVR BUFFER POINTER
014706 012767 030516 004676     MOV     #RBUF,RBFEND    ;MARK END OF THIS BUFFER
014714 016705 004662            MOV     CHRCNT,R5       ;GET CHAR COUNT
014720 005405                   NEG     R5              ;MAKE IT POSITIVE
014722 060567 004664            ADD     R5,RBFEND
014726 016761 004644 000004     MOV     CURLPR,LPR(R1)  ;LOAD THE LPR REG
014734 016761 004642 000010     MOV     CHRCNT,BCR(R1)  ;LOAD THE BYTE COUNT REG
014742 012761 032776 000006     MOV     #TBUF,CAR(R1)   ;LOAD CURRENT ADDRESS REG
014750 000207                   RTS     PC              ;RETURN


                        ;THIS ROUTINE IS CALLED TO SELECT A NEW LINE NO. BASED ON THE
                        ;VALUE OF THE LINE SELECTION PARAMETER

                                ;CALLING SEQUENCE:

                                ;JSR     PC,SELINE      ;CALL THE ROUTINE
                                ;BR      1$             ;EXIT BRANCH-ROUTINE MOVES THE RETURN
                                                        ;PC AROUND THIS BR IF MORE LINES ARE
                                                        ;YET TO BE TESTED

014752 105767 005175    SELINE: TSTB    LINE+1          ;FIRST TIME THROUGH FOR ANY TEST ?
014756 001010                   BNE     1$              ;BR IF NOT
014760 105167 005167            COMB    LINE+1          ;SET ENTRY FLAG
014764 012767 000001 004542     MOV     #1,LINMSK       ;INIT SELECT TEST MASK TO TEST LINE 00
```

```
4661  014772  105067  005154            CLRB    LINE            ;START WITH LINE #00
4662  014776  000405                    BR      2$              ;GO TEST FOR LINE #00
4663  015000  105267  005146    1$:     INCB    LINE            ;GENERATE NEW LINE NO.
4664  015004  006367  004524            ASL     LINMSK          ;SHIFT SELECT MASK TO TEST NXT LINE
4665  015010  001407                    BEQ     3$              ;RETURN TO EXIT BRANCH - ALL LINES DONE
4666  015012  036767  004516  004512    BIT     LINMSK,LINSEL   ;IS THE LINE SELECTED FOR TEST ??
4667  015020  001767                    BEQ     1$              ;BR IF NOT
4668  015022  062716  000002            ADD     #2,(SP)         ;MOVE RETURN PC AROUND EXIT BRANCH
4669  015026  000402                    BR      4$              ;RETURN TO TEST SELECTED LINE
4670  015030  005067  005116    3$:     CLR     LINE            ;INIT ENTRY FLAG AND LINE NO. TO 000
4671  015034  142777  000017  004460    BICB    #17,@DHADR      ;INIT LINE SELECT BITS IN "SCR"
4672  015042  000207            4$:     RTS     PC              ;RETURN TO CALLING TEST
4673
4674                          ;THIS ROUTINE IS CALLED TO CONVERT EITHER THE "DH" NUMBER OR THE
4675                          ;"LINE" NUMBER TO TWO ASCII CHARACTERS AND MOVE THEM INTO A
4676                          ;PARTICULAR MESSAGE BUFFER FOR ERROR REPORTING
4677
4678                                  ;CALLING SEQUENCE
4679
4680                                  ;JSR     R5,SUNUM       ;CALL TO THIS ROUTINE
4681                                  ;ADDR1                  ;ADDRESS OF THE NUMBER TO BE CONVERTED
4682                                  ;ADDR2                  ;ADDRESS OF THE MSG BUFFER SLOT
4683
4684  015044                  SUNUM:
4685  015044  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4686  015046  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4687  015050  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4688  015052  012500                  MOV     (R5)+,R0        ;GET ADDRESS OF NUMBER
4689  015054  012501                  MOV     (R5)+,R1        ;GET MSG BUFFER ADDR
4690  015056  111000                  MOVB    (R0),R0         ;GET NO. TO BE CONVERTED
4691  015060  010002                  MOV     R0,R2           ;SAVE IT IN R2
4692  015062  006202                  ASR     R2              ;SHIFT MSD TO LSD POSITION
4693  015064  006202                  ASR     R2
4694  015066  006202                  ASR     R2
4695  015070  042702  177770         BIC     #177770,R2      ;CLR JUNK BITS
4696  015074  062702  000060         ADD     #60,R2          ;MAKE IT ASCII
4697  015100  110221                  MOVB    R2,(R1)+        ;PUT IT IN MSG BUFFER
4698  015102  042700  177770         BIC     #177770,R0      ;CLR JUNK FROM LSD
4699  015106  062700  000060         ADD     #60,R0          ;MAKE IT ASCII
4700  015112  110011                  MOVB    R0,(R1)         ;PUT LSD IN THE BUFFER
4701  015114  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
4702  015116  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
4703  015120  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
4704  015122  000205                  RTS     R5              ;RETURN TO CALLER
4705
4706                          ;THIS ROUTINE IS CALLED TO SET UP THE ERROR INFORMATION IN THE
4707                          ;MESSAGE BUFFERS
4708
4709  015124  010067  164032  SUER2:  MOV     R0,$REG0        ;STORE THE REGS IN CORE
4710  015130  010167  164030  SUER1:  MOV     R1,$REG1
4711  015134  010267  164026          MOV     R2,$REG2
4712  015140  010367  164024          MOV     R3,$REG3
4713  015144  010467  164022          MOV     R4,$REG4
4714  015150  000207                  RTS     PC              ;RETURN TO REPORT ERROR
4715
4716                          ;THIS ROUTINE AUTOSIZES THE SYSTEM TO DETERMINE THE ADDRESSES AND
```

```
                                        ;VECTORS OF THE DH11'S AND DM11-BB'S.

015152  010046                AUTOSZ: MOV     R0,-(SP)
015154  005003                        CLR     R3
015156  012702  021756                MOV     #DHADRS,R2
015162  005022                25$:    CLR     (R2)+           ;CLEAR DH TABLES.
015164  005203                        INC     R3
015166  020327  000102                CMP     R3,#102         ;HAVE WE CLEARED ALL ENTRIES?
015172  001373                        BNE     25$             ;BRANCH IF NOT.
015174  013746  000004                MOV     @#4,-(SP)       ;SAVE TRAP VECTOR.
015200  012737  015306  000004        MOV     #4$,@#4         ;SETUP FOR NON-EXISTENT MEMORY TRAP.
015206  012703  022050                MOV     #DMADRS,R3      ;SETUP DM ADDRESS TABLE POINTER.
015212  012702  021756                MOV     #DHADRS,R2      ;SET UP DH ADDRESS TABLE POINTER.

015216  012701  160020                MOV     #160020,R1      ;R1=FIRST ADDRESS TO BE TESTED.

015222  005711                1$:     TST     (R1)            ;SEE IF ADDRESS IN R1 RESPONDS.
015224  005761  000016                TST     16(R1)          ;CHECK TO SEE IF DEVICE IS MODULO 20.
015230  052711  004000                BIS     #4000,(R1)      ;IF IT IS, CONTINUE
                                                              ;AND CHECK TO SEE
015234  052711  001000                BIS     #1000,(R1)      ;IF THIS ADDRESS CONTAINS
015240  052711  002000                BIS     #2000,(R1)      ;A DH-11.
015244  032711  003000                BIT     #3000,(R1)      ;CHECK TO INSURE THESE BITS SET.
015250  001410                        BEQ     3$              ;IF NOT, BRANCH.
                                                              ;SET THE MAINTENANCE BIT, THE NON-
015252  052711  000400                BIS     #400,(R1)       ;EXISTENT MEMORY BIT AND THE CLEAR
                                                              ;NON-EXISTENT MEMORY INTERRUPT BIT.
015256  032711  002400                BIT     #2400,(R1)      ;IS THIS A DH-11? (BITS 8 AND 10 SHOULD
                                                              ;CLEAR IF THIS IS A DH11.)

015262  001003                        BNE     3$              ;IF NOT, CHECK TO SEE IF THIS IS A DM11-BB.
015264  042711  001000                BIC     #1000,(R1)      ;CLEAR MAINTENANCE BIT.
015270  010122                        MOV     R1,(R2)+        ;SAVE THE ADDRESS IN THE DH ADR TABLE.

015272  020127  163760        3$:     CMP     R1,#163760      ;HAVE WE REACHED THE TOP OF THE FLOATING ADDRESSES.
015276  001406                        BEQ     5$              ;IF YES, GET OUT.
015300  062701  000020                ADD     #20,R1          ;IF NOT, UPDATE ADDRESS AND
015304  000746                        BR      1$              ;GO CHECK IT.

015306  012716  015272        4$:     MOV     #3$,(SP)        ;IF DH ADDRESS DOES NOT RESPOND, GO TO 3$.
015312  000002                        RTI

                                        ;TEST FOR DM11 BB ADDRESS

015314  012737  015346  000004  5$:     MOV     #6$,@#4         ;SETUP FOR NON-EXISTENT MEMORY TRAP.
015322  012701  170500                MOV     #170500,R1      ;R1=FIRST ADDRESS TO BE TESTED.
015326  005711                21$:    TST     (R1)            ;SEE IF ADDRESS RESPONDS.
015330  010123                        MOV     R1,(R3)+        ;IF IT DOES, THIS IS A DM11-BB,
                                                              ;SO SAVE THE ADDRESS.
015332  020127  170670        23$:    CMP     R1,#170670      ;HAVE WE REACHED THE TOP OF THE DM11 ADDRESSES?
015336  001406                        BEQ     22$             ;IF YES, GET OUT.
015340  062701  000010                ADD     #10,R1          ;IF NOT, UPDATE ADDRESS AND
015344  000770                        BR      21$             ;GO CHECK IT.

015346  012716  015332        6$:     MOV     #23$,(SP)       ;IF DM ADDRESS DOES NOT RESPOND, GO TO 23$.
```

```
                                        RTI
                                
                        22$:    MOV     (SP)+,@#4       ;RESTORE TRAP VECTOR.
                                SUB     #DHADRS,R2      ;HAVE WE FOUND ANY DH11'S AT ALL?
                                BNE     7$              ;IF YES, BRANCH
                                TYPE    ,MSG1           ;NO DH11'S WERE FOUND,
                                HALT
                                
                        7$:     ASR     R2              ;R2 NOW CONTAINS THE NUMBER
                                CLR     R0              ;OF DH'S FOUND.
                        8$:     ROL     R0              ;FILL R0 WITH 1'S
                                INC     R0              ;CORRESPONDING TO
                                DEC     R2              ;THE NUMBER OF DH'S
                                TST     R2              ;FOUND.
                                BNE     8$
                                MOV     R0,$DHSEL       ;$DHSEL CONTAINS THE DH SELECTION PARAMETER.
                                                        ;IE. ALL DH'S FOUND WILL BE TESTED.
                                
                        ;FIND DH VECTOR:
                                MOV     #DHADRS,R2      ;SETUP POINTER TO BEGINNING OF DH
                                MOV     #DHVEC,R5       ;ADDRESS TABLE AND VECTOR TABLE.
                                MOV     #340,@#IOTVEC+2 ;SET IOT TRAP PRIORITY TO 7.
                                MOV     #12$,@#IOTVEC   ;SETUP IOT TRAP VECTOR.
                                MOV     #300,R3         ;START OF FLOATING VECTORS
                                MOV     #302,R4         ;PC OF IOT INSTR.
                                
                        9$:     MOV     R4,(R3)+        ;FILL VECTOR AREA WITH ADDRESS
                                                        ;OF NEXT INSTR (.+2)
                                MOV     #4,(R4)+        ;NEXT INSTRUCTION IS AN IOT TRAP.
                                CMP     (R3)+,(R4)+     ;UPDATE R3+R4.
                                CMP     R4,#1000        ;HAVE WE REACHED TO TOP OF THE
                                                        ;VECTOR SPACE?
                                BLOS    9$              ;IF NOT, REPEAT PROCESS.
                                
                        10$:    TST     (R2)            ;HAVE WE CHECK ALL DH'S?
                                BEQ     13$             ;IF YES, GET OUT + CHECK FOR DM11 BB'S VECTORS.
                                
                                CLR     PS              ;ZERO CPU PRIORITY.
                                BIS     #1000,@(R2)     ;SET MAINTENANCE BIT
                                BIS     #300,@(R2)      ;ATTEMPT TO CAUSE RECEIVER
                                                        ;INTERRUPT.
                                CLR     R0
                                
                        11$:    INC     R0              ;WAIT...
                                BNE     11$
                                TYPE    ,MSG2           ;ERROR MSG-NO DH RECEIVER INTERRUPT OCCURRED.
                                BIS     #4000,@(R2)     ;DO A MASTER CLEAR
                                BIC     #1000,@(R2)     ;CLEAR MAINTENANCE BIT
                                BR      10$
                                
                        12$:    MOV     (SP),R1
                                BIC     #7,R1           ;CLEAR GARBAGE.
                                MOV     R1,(R5)+        ;SAVE VECTOR ADDRESS.
                                CMP     (SP)+,(SP)+     ;POP STACK
                                MOV     #10$,(SP)       ;SETUP FOR RETURN.
                                BIS     #4000,@(R2)     ;DO A MASTER CLEAR
```

```
 4829  015570  042732  001000            BIC     #1000,@(R2)+    ;CLEAR MAINTENANCE BIT.
 4830  015574  000002                    RTI
 4831
 4832                                    ;FIND DM11 BB VECTORS:
 4833
 4834  015576  012702  022050    13$:    MOV     #DMADRS,R2      ;SET POINTERS TO BEGINNING OF
 4835  015602  012705  022106            MOV     #DMVEC,R5       ;ADR TABLE & VECTOR TABLE.
 4836  015606  012737  015670  000020    MOV     #16$,@#IOTVEC   ;SET IOT TRAP VECTOR.
 4837
 4838  015614  005712            14$:    TST     (R2)            ;HAVE WE CHECKED ALL DM'S?
 4839  015616  001441                    BEQ     17$             ;IF YES, GET OUT.
 4840  015620  005067  162152            CLR     PS              ;ZERO CPU PRIORITY
 4841  015624  052772  001000  000000    BIS     #1000,@(R2)     ;SET MAINTENANCE BIT.
 4842  015632  052772  000300  000000    BIS     #300,@(R2)      ;ATTEMPT TO CAUSE INTERRUPT.
 4843  015640  005000                    CLR     R0
 4844
 4845  015642  005200            15$:    INC     R0              ;WAIT....
 4846  015644  001376                    BNE     15$
 4847  015646  104401  026026            TYPE    ,MSG3           ;ERROR MSG - NO DM11-BB INTERRUPT OCCURRED.
 4848  015652  052772  004000  000000    BIS     #4000,@(R2)     ;CLEAR BITS PREVIOUSLY SET.
 4849  015660  042772  001000  000000    BIC     #1000,@(R2)     ;CLEAR MAINTENANCE BIT.
 4850  015666  000752                    BR      14$
 4851
 4852  015670  011601            16$:    MOV     (SP),R1
 4853  015672  162701  000004            SUB     #4,R1           ;CALCULATE VECTOR ADDRESS.
 4854  015676  010125                    MOV     R1,(R5)+        ;SAVE VECTOR ADDRESS.
 4855  015700  022626                    CMP     (SP)+,(SP)+     ;POP STACK.
 4856  015702  012716  015614            MOV     #14$,(SP)       ;SETUP FOR RETURN.
 4857  015706  052772  004000  000000    BIS     #4000,@(R2)     ;CLEAR BITS PREVIOUSLY SET.
 4858  015714  042732  001000            BIC     #1000,@(R2)+    ;CLEAR MAINTENANCE BIT AND
 4859                                                            ;POINT TO NEXT DM11-BB ADDRESS.
 4860  015720  000002                    RTI
 4861
 4862  015722  012737  011112  000020    MOV     #$SCOPE,@#IOTVEC  ;RESTORE IOT VECTOR FOR SCOPE ROUTINE.
 4863  015730  012600            17$:    MOV     (SP)+,R0        ;RESTORE R0.
 4864  015732  012703  000300            MOV     #300,R3         ;START OF FLOATING VECTORS.
 4865  015736  012704  000302            MOV     #302,R4
 4866
 4867  015742  010423            18$:    MOV     R4,(R3)+        ;FILL VECTOR AREA WITH ADDRESS OF NEXT
 4868                                                            ;INSTRUCTION (.+2).
 4869  015744  012724  000000            MOV     #0,(R4)+        ;NEXT INSTRUCTION IS A HALT.
 4870  015750  022324                    CMP     (R3)+,(R4)+     ;UPDATE R3 & R4.
 4871  015752  020427  001000            CMP     R4,#1000        ;ARE WE DONE?
 4872  015756  101771                    BLOS    18$             ;IF NOT, REPEAT UNTIL ADDRESSES
 4873                                                            ;377 TO 777 ARE DONE.
 4874  015760  013701  022014            MOV     @#DHVEC,R1      ;LET R1 POINT TO 1ST DH VECTOR ADDRESS.
 4875  015764  005737  022016            TST     @#DHVEC+2       ;IS THERE MORE THAN ONE ENTRY?
 4876  015770  001403                    BEQ     26$             ;BRANCH IF NO.
 4877  015772  163701  022016            SUB     @#DHVEC+2,R1    ;DETERMINE NUMBER OF ADDRESSES
 4878                                                            ;BETWEEN DH VECTORS (10(8) OR 20(8)).
 4879  015776  005401                    NEG     R1              ;MAKE IT POSITIVE.
 4880  016000  010167  004136    26$:    MOV     R1,ADRVEC       ;SAVE THAT NUMBER.
 4881  016004  032777  000002  163126    BIT     #BIT1,@SWR      ;SHOULD DEVICE MAP BE TYPED OUT?
 4882  016012  001441                    BEQ     20$             ;IF NOT, RETURN.
 4883  016014  104401                    TYPE                    ;TYPEOUT MAP OF DH & DM11-BB'S
 4884  016016  026073                    DEVMAP                  ;FOUND.
```

```
 4885  016020  012701  021756          MOV     #DHADRS,R1      ;R1=BEGINNING OF DH ADDRESS TABLE.
 4886  016024  012702  022014          MOV     #DHVEC,R2       ;R2=BEGINNING OF DH VECTOR TABLE.
 4887  016030  012703  022050          MOV     #DMADRS,R3      ;R3=BEGINNING OF DM11-BB ADDRESS TABLE.
 4888  016034  012704  022106          MOV     #DMVEC,R4       ;R4=BEGINNING OF DM11-BB VECTOR TABLE.
 4889  016040  005005                  CLR     R5              ;CLEAR TABLE LINE COUNTER
 4890
 4891  016042  012146          19$:    MOV     (R1)+,-(SP)     ;MOVE DATA TO BE TYPED
 4892  016044  104403                  TYPOS                   ;TYPE DATA
 4893  016046     006                  .BYTE   6
 4894  016047     001                  .BYTE   1
 4895  016050  012246                  MOV     (R2)+,-(SP)     ;MOVE DATA TO BE TYPED
 4896  016052  104403                  TYPOS                   ;TYPE DATA
 4897  016054     005                  .BYTE   5
 4898  016055     000                  .BYTE   0
 4899  016056  104401  026067          TYPE    ,SPACE
 4900  016062  012346                  MOV     @(R3)+,-(SP)    ;MOVE DATA TO BE TYPED.
 4901  016064  104403                  TYPOS                   ;TYPE DATA.
 4902  016066     006                  .BYTE   6
 4903  016067     001                  .BYTE   1
 4904  016070  104401  026067          TYPE    ,SPACE
 4905  016074  012446                  MOV     @(R4)+,-(SP)    ;MOVE DATA TO BE TYPED.
 4906  016076  104403                  TYPOS                   ;TYPE DATA.
 4907  016100     005                  .BYTE   5
 4908  016101     000                  .BYTE   0
 4909  016102  104401                  TYPE                    ;TYPE A CARRIAGE RETURN & LINE FEED.
 4910  016104  001227                  $CRLF
 4911  016106  005711                  TST     (R1)            ;HAVE WE TYPED ALL DH ENTRIES?
 4912  016110  001354                  BNE     19$             ;IF NOT, DO IT AGAIN.
 4913  016112  104401  001227          TYPE    ,$CRLF
 4914  016116  000207          20$:    RTS     PC              ;IF YES, GO BACK TO MAIN PROGRAM.
 4915
 4916                                  ;THIS ROUTINE IS USED TO ACCEPT INPUT PARAMETERS FROM THE CONSOLE
 4917                                  ;TELETYPE
 4918
 4919  016120  104401          INPARA: TYPE
 4920  016122  025440                  VCWC                    ;"ASK FOR NO. ADDRESSES BETWEEN VECTORS"
 4921  016124  104412                  RDOCT                   ;READ OCTAL NO. FM TTY
 4922  016126  012600                  MOV     (SP)+,R0        ;GET THE NO. HE TYPED
 4923  016130  001407                  BEQ     3$              ;BR IF HE TYPED <CR>
 4924  016132  022700  000010          CMP     #10,R0          ;10(8) ADDRESSES BETWEEN VECTORS ?
 4925  016136  001406                  BEQ     4$              ;BR IF YES
 4926  016140  022700  000020          CMP     #20,R0          ;20(8) ADDRESSES BETWEEN VECTORS ??
 4927  016144  001403                  BEQ     4$              ;BR IF YES
 4928  016146  000764                  BR      INPARA          ;ASK ALL OVER AGAIN
 4929  016150  012700  000020  3$:     MOV     #20,R0          ;SET UP CONSTANT FOR 20(8) ADDRESSES
 4930  016154  005067  004166  4$:     CLR     DPFLG           ;CLEAR PATTERNS TESTS FLAG
 4931  016160  005067  004174          CLR     RETFLG          ;INIT ECHO TEST RETURN FLAG
 4932  016164  000207                  RTS     PC              ;RETURN TO CALLER
 4933
 4934  016166  012767  177777  003520  INPARX: MOV     #-1,VCFLG       ;SET START AT 200 FLAG
 4935  016174  000167  163444          JMP     BEGINA          ;GO START UP
 4936  016200  012700  177777  INPARC: MOV     #-1,R0          ;SET FLAG IN R0
 4937  016204  005067  003504          CLR     VCFLG           ;INIT VECTOR SET UP FLAG
 4938  016210  005067  004132          CLR     DPFLG           ;CLEAR PATTERNS TESTS FLAG
 4939  016214  005067  004140          CLR     RETFLG          ;INIT ECHO TEST RETURN FLAG
 4940  016220  000167  163420          JMP     BEGINA          ;GO ASK FOR SELECT PARAMETER
```

```

016224  013701  021756          INPAR:  MOV     @#DHADRS,R1     ;MOVE ADDRESS OF FIRST DH INTO R1.
016230  032777  000001  162702          BIT     #BIT0,@SWR      ;ARE PARAMETERS TO BE INPUT MANUALLY?
016236  001405                          BEQ     2$              ;BRANCH IF NOT.
016240  104401                  1$:     TYPE                    ;ASK FOR DEVICE ADDRESS
016242  024747                          INMSG1
016244  104412                          RDOCT                   ;READ IN WHAT IS TYPED
016246  012601                          MOV     (SP)+,R1        ;GET THE NO. HE TYPED
016250  001403                          BEQ     INPAR1          ;BR IF DEFAULT
016252  004767  000176          2$:     JSR     PC,CHKADR       ;GO CHECK VALIDITY OF THE ADDR
016256  000770                          BR      1$              ;ERROR BRANCH

016260  013701  022014          INPAR1: MOV     @#DHVEC,R1      ;MOVE FIRST DH VECTOR INTO R1.
016264  032777  000001  162646          BIT     #BIT0,@SWR      ;ARE PARAMETERS TO BE INPUT MANUALLY?
016272  001405                          BEQ     2$              ;BRANCH IF NOT.
016274  104401                  1$:     TYPE                    ;ASK FOR VECTOR ADDRESS
016276  025013                          INMSG2
016300  104412                          RDOCT                   ;READ IN WHAT HE TYPES
016302  012601                          MOV     (SP)+,R1        ;GET THE ADDRESS
016304  001403                          BEQ     INPAR3          ;BR IF DEFAULT
016306  004767  000306          2$:     JSR     PC,CHKVCT       ;GO CHECK VALIDITY OF VECTOR
016312  000770                          BR      1$              ;ERROR BRANCH

016314  005767  004040          INPAR3: TST     RETFLG          ;LINE ECHO TESTS ?
016320  001402                          BEQ     1$              ;BR IF NOT
016322  000167  166426                  JMP     ECHO2           ;RETURN TO LINE ECHO TESTS
016326  005767  004014          1$:     TST     DPFLG           ;DATA PATTERNS TESTS ACTIVE ??
016332  001402                          BEQ     2$              ;BR IF NOT
016334  000167  167614                  JMP     EXPAT2          ;GO BACK TO PATTERNS TESTS
016340  013701  022144          2$:     MOV     @#SDHSEL,R1     ;MOVE DEVICE SELECTION PARAMETER INTO R1.
016344  005700                          TST     R0              ;START AT 210?
016346  100404                          BMI     4$              ;BRANCH IF YES.
016350  032777  000001  162562          BIT     #BIT0,@SWR      ;IS PARAMETER TO BE INPUT MANUALLY?
016356  001405                          BEQ     3$              ;BRANCH IF NOT.
016360  104401                  4$:     TYPE                    ;ASK FOR DEVICE SELECTION PARAMETER
016362  025062                          INMSG3
016364  104412                          RDOCT                   ;READ IN WHAT HE TYPES
016366  012601                          MOV     (SP)+,R1        ;GET THE SELECT PARAMETER
016370  001402                          BEQ     INPAR4          ;BR IF  DEFAULT
016372  010167  003132          3$:     MOV     R1,DHSEL        ;SET UP DH11 SELECTION PARAMETER
016376  012767  177777  003126  INPAR4: MOV     #-1,LINSEL      ;INIT FOR ALL 16. LINES
016404  032777  000001  162526          BIT     #BIT0,@SWR      ;IS LINE SELECT PARAMETER TO BE INPUT MANUALLY?
016412  001407                          BEQ     1$              ;BRANCH IF NO.
016414  104401                          TYPE                    ;ASK FOR LINE SELECT PARAMETER
016416  025376                          INMSGA
016420  104412                          RDOCT                   ;READ WHAT HE TYPES
016422  012601                          MOV     (SP)+,R1        ;GET IT OFF STACK
016424  001402                          BEQ     1$              ;BR IF DEFAULT
016426  010167  003100                  MOV     R1,LINSEL       ;SET LINE SELECT PARAMETER
016432  032777  000400  162500  1$:     BIT     #BIT8,@SWR      ;HALT AFTER SET UP ??
016440  001403                          BEQ     EXPAR           ;BR IF  NOT
016442  104401                          TYPE                    ;TYPE CONTINUE MESSAGE PRIOR TO HALTING
016444  025326                          INMSG7
016446  000000                          HALT                    ;DEPRESS CONTINUE TO RESUME TESTING
016450  000167  163662          EXPAR:  JMP     START2          ;GO START UP THE PROGRAM

```

```
 4997
 4998  016454  020127  160020          CHKADR: CMP     R1,#160020      ;IS ADDRESS ABOVE OR EQUAL TO LOW LIMIT
 4999  016460  002001                          BGE     1$              ;BR IF YES
 5000  016462  000453                          BR      4$              ;BR IF NOT
 5001  016464  020127  160420            1$:   CMP     R1,#160420      ;IS IT BELOW THE HIGH LIMIT?
 5002  016470  002401                          BLT     2$              ;BR IF YES
 5003  016472  000447                          BR      4$              ;BR IF NOT
 5004  016474  032701  000017            2$:   BIT     #17,R1          ;CORRECT BOUNDARY ?
 5005  016500  001044                          BNE     4$              ;BR IF NOT
 5006  016502  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN PC AROUND ERROR BRANCH
 5007  016506  005767  003646                  TST     RETFLG          ;ARE WE IN ECHO TESTS ?
 5008  016512  001403                          BEQ     21$             ;BR IF NOT
 5009  016514  010167  003002                  MOV     R1,DHADR        ;SET UP DH11 DEVICE ADDRESS
 5010  016520  000436                          BR      5$              ;CONTINUE
 5011  016522  005767  003620            21$:  TST     DPFLG           ;PATTERNS TESTS ACTIVE ??
 5012  016526  001403                          BEQ     22$             ;BR IF NOT
 5013  016530  010167  002766                  MOV     R1,DHADR        ;SET UP DEVICE ADDRESS
 5014  016534  000430                          BR      5$              ;CONTINUE
 5015  016536  012702  021614            22$:  MOV     #DHADTB,R2      ;POINT TO BEGIN OF ADDR TABLE
 5016  016542  032777  000001  162370        BIT     #BIT0,@#SWR     ;ARE WE AUTOSIZING?
 5017  016550  001011                          BNE     3$              ;BRANCH IF NOT.
 5018  016552  012703  021756                  MOV     #DHADRS,R3      ;POINT TO BEGINNING OF AUTOSIZER
 5019                                                                  ;DH ADDRESS TABLE.
 5020  016556  016704  003362                  MOV     @DHSEL,R4
 5021  016562  012322                    6$:   MOV     (R3)+,@(R2)+    ;MOVE CONTENTS OF AUTOSIZER DH TABLE
 5022                                                                  ;TO THE TABLE USED BY PROGRAM.
 5023  016564  006204                          ASR     R4
 5024  016566  005704                          TST     R4              ;HAVE WE MOVED ALL TABLE ENTRIES?
 5025  016570  001374                          BNE     6$              ;BRANCH IF NOT--ONE MORE TIME.
 5026  016572  000411                          BR      5$              ;RETURN TO INPUT ROUTINES.
 5027  016574  010122                    3$:   MOV     R1,(R2)+        ;SET UP A TABLE ENTRY
 5028  016576  062701  000020                  ADD     #20,R1          ;GENERATE NEXT DH11 ADDR
 5029  016602  022702  021654                  CMP     #DHADTB+40,R2   ;END OF TABLE ?
 5030  016606  001372                          BNE     3$              ;BR IF NOT
 5031  016610  000402                          BR      5$              ;RETURN TO INPUT ROUTINES
 5032  016612  104401                    4$:   TYPE                    ;TELL HIM HE GOOFED
 5033  016614  025133                          INMSG4
 5034  016616  000207                    5$:   RTS     PC              ;RETURN TO INPUT ROUTINES
 5035
 5036  016620  020127  000300          CHKVCT: CMP     R1,#300         ;IS ADDRESS ABOVE OR EQUAL TO LOW LIMIT
 5037  016624  002001                          BGE     1$              ;BR IF YES
 5038  016626  000452                          BR      4$              ;BR IF NOT
 5039  016630  020127  001000            1$:   CMP     R1,#1000        ;IS IT BELOW THE HIGH LIMIT?
 5040  016634  002401                          BLT     2$              ;BR IF YES
 5041  016636  000446                          BR      4$              ;BR IF NOT
 5042  016640  032701  000007            2$:   BIT     #7,R1           ;CORRECT BOUNDARY ?
 5043  016644  001043                          BNE     4$              ;BR IF NOT
 5044  016646  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN PC AROUND ERROR BRANCH
 5045  016652  005767  003502                  TST     RETFLG          ;ARE WE IN ECHO TESTS ?
 5046  016656  001403                          BEQ     21$             ;BR IF NOT
 5047  016660  010167  002640                  MOV     R1,DHVCT        ;SET UP DH11 VECTOR ADDR
 5048  016664  000435                          BR      5$              ;CONTINUE
 5049  016666  005767  003454            21$:  TST     DPFLG           ;PATTERNS TESTS ACTIVE ??
 5050  016672  001403                          BEQ     22$             ;BR IF NOT
 5051  016674  010167  002624                  MOV     R1,DHVCT        ;SET UP DEVICE VECTOR
 5052  016700  000427                          BR      5$              ;CONTINUE
```

```
5053  016702  012702  021654          22$:    MOV     #DHVCTB,R2      ;POINT TO BEGIN OF VECTOR TABLE
5054  016706  032777  000001  162224          BIT     #BIT0,@$SWR     ;ARE WE AUTOSIZING?
5055  016714  001011                          BNE     3$              ;BRANCH IF NOT.
5056  016716  012703  022014                  MOV     #DHVEC,R3       ;POINT TO BEGINING OF AUTOSIZER
5057                                                                  ;DH VECTOR TABLE.
5058  016722  016704  003216                  MOV     $DHSEL,R4
5059  016726  012322                  6$:     MOV     (R3)+,(R2)+     ;MOVE CONTENTS OF AUTOSIZER VECTOR
5060                                                                  ;TABLE TO TABLE USED BY PROGRAM.
5061  016730  006204                          ASR     R4
5062  016732  005704                          TST     R4              ;HAVE WE MOVED ALL TABLE ENTRIES?
5063  016734  001374                          BNE     6$              ;BRANCH IF NOT--ONE MORE TIME.
5064  016736  000410                          BR      5$              ;RETURN TO INPUT ROUTINES.
5065  016740  010122                  3$:     MOV     R1,(R2)+        ;SET UP A TABLE ENTRY
5066  016742  060001                          ADD     R0,R1           ;GENERATE NEXT DH11 ADDR
5067  016744  022702  021714                  CMP     #DHVCTB+40,R2   ;END OF TABLE ?
5068  016750  001373                          BNE     3$              ;BR IF NOT
5069  016752  000402                          BR      5$              ;RETURN TO INPUT ROUTINES
5070  016754  104401                  4$:     TYPE                    ;TELL HIM HE GOOFED
5071  016756  025204                          INMSG5
5072  016760  000207                  5$:     RTS     PC              ;RETURN TO INPUT ROUTINES
5073
5074                                  ;THESE TWO ROUTINES SERVICE UNEXPECTED BUS ERROR AND RSVD INSTR TRAPS
5075
5076  016762  010667  162210          BUSER:  MOV     SP,$REG6        ;SAVE THE SP
5077  016766  012667  162172                  MOV     (SP)+,$REG1     ;GET THE TRAP PC
5078  016772  012667  162170                  MOV     (SP)+,$REG2     ;GET THE TRAP PSW
5079  016776  012706  001100                  MOV     #STACK,SP       ;RESET THE STACK POINTER
5080  017002  012767  017012  162100          MOV     #1$,$LPERR      ;ALWAYS COME BACK TO 1$
5081  017010  104014                          ERROR   14              ;UNEXPECTED BUS ERROR TRAP
5082  017012  000005                  1$:     RESET                   ;PREPARE TO RESTART
5083  017014  004767  002406                  JSR     PC,CHPS1        ;GO CLEAR PSW
5084  017020  000167  175600                  JMP     CKRST2          ;GO RESTART THE PROGRAM
5085
5086  017024  010667  162146          RESERR: MOV     SP,$REG6        ;SAVE THE SP
5087  017030  012667  162130                  MOV     (SP)+,$REG1     ;GET THE TRAP PC
5088  017034  012667  162126                  MOV     (SP)+,$REG2     ;GET THE TRAP PSW
5089  017040  012706  001100                  MOV     #STACK,SP       ;RESET THE STACK POINTER
5090  017044  012767  017054  162036          MOV     #1$,$LPERR      ;ALWAYS COME BACK TO 1$
5091  017052  104015                          ERROR   15              ;UNEXPECTED RSVD INSTR ERROR TRAP
5092  017054  000005                  1$:     RESET                   ;PREPARE TO RESTART
5093  017056  004767  002344                  JSR     PC,CHPS1        ;GO CLEAR PSW
5094  017062  000167  175536                  JMP     CKRST2          ;GO RESTART THE PROGRAM
5095
5096                                  ;THIS ROUTINE IS CALLED WHEN A TEST NEEDS TO RESTORE THE TRAP
5097                                  ;CATCHER IN THE DH11 VECTOR
5098
5099  017066  016703  002432          RESTRP: MOV     DHVCT,R3        ;GET VECTOR ADDRESS
5100  017072  010313                          MOV     R3,(R3)         ;RESTORE THE TRAP CATCHER
5101  017074  062723  000002                  ADD     #2,(R3)+
5102  017100  005023                          CLR     (R3)+
5103  017102  010313                          MOV     R3,(R3)
5104  017104  062723  000002                  ADD     #2,(R3)+
5105  017110  005023                          CLR     (R3)+
5106  017112  000207                          RTS     PC              ;RETURN TO CALLING TEST
5107
5108                                  ;THIS ROUTINE CALLED BY ANY TEST THAT NEEDS A TIMING WAIT LOOP
```

```
                                ;"TIMEA" IS INITIALIZED BY THE CALLING ROUTINE TO THE MINIMUM REQUIRED
                                ;VALUE AND "TIMEB" IS CLEARED TO 000000. IF A TIME OUT OCCURS THIS
                                ;ROUTINE WILL MOVE THE RETURN PC AROUND THE "LOOP" BRANCH BACK IN
                                ;THE ROUTINE THAT CALLED IT TO ALLOW REPORTING AN ERROR MESSAGE

017114  005267  003222          TIMEIT: INC     TIMEB           ;COUNT B
017120  001005                          BNE     1$              ;BR IF NOT ZERO
017122  005367  003212                  DEC     TIMEA           ;COUNT TIME A
017126  001002                          BNE     1$              ;BR IF NO TIMEOUT
017130  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN PC TO ALLOW ERROR REPORT
017134  000207                  1$:     RTS     PC              ;RETURN TO THE CALLING TEST


                                ;THIS ROUTINE IS CALLED TO CLEAR ALL ENTRIES IN THE STATISTICS TABLES

017136  012705  030202          CLSTAT: MOV     #RTOTAL,R5      ;SET UP POINTER TO BEGINNING
017142  005025                  1$:     CLR     (R5)+           ;CLEAR ONE WORD
017144  022705  030502                  CMP     #RTOTAL+192.,R5 ;CLEARED ALL ENTRIES ??
017150  001374                          BNE     1$              ;BR IF NOT
017152  000207                          RTS     PC


                                ;THIS ROUTINE IS CALLED TO RETRIEVE A NEW LPR CONSTANT
                                ;FROM THE LPR TABLE (LPRTAB)

                                ;CALLING SEQUENCE:

                                ;       JSR     R5,SETLPR       ;CALL
                                ;       BR      NEWLIN          ;EXIT BRANCH - EXECUTED AFTER ALL
                                                                ;13 BAUD RATES EXERCISED

017154  022767  021576  002416  SETLPR: CMP     #CURLPR,LPRPTR  ;DONE ALL 13. ENTRIES ??
017162  001425                          BEQ     3$              ;BR IF YES
017164  017767  002410  002404          MOV     @LPRPTR,CURLPR  ;GET THE LPR CONSTANT
017172  105777  161742                  TSTB    @SWR            ; QUICK TEST ?
017176  100410                          BMI     1$              ;BR IF NOT - SUPPLY THE WHOLE THING
017200  022767  033500  002370          CMP     #33500,CURLPR   ;9600 BAUD TEST ??
017206  001404                          BEQ     1$              ;BR IF YES
017210  012767  177777  002322          MOV     #-1,QUICK       ;SET QUICK TEST FLAG
017216  000402                          BR      2$              ;CONTINUE
017220  005067  002314          1$:     CLR     QUICK           ;DO FULL TESTING AT 9600. BAUD
017224  062767  000002  002346  2$:     ADD     #2,LPRPTR       ;UPDATE THE TABLE POINTER
017232  062705  000002                  ADD     #2,R5           ;MOVE PC AROUND ERROR BRANCH
017236  000205                  3$:     RTS     R5              ;RETURN


                                ;THIS ROUTINE IS CALLED TO SETUP THE CHAR LENGTH SELECT BITS AND
                                ;LOAD THE OUTPUT DATA BUFFER

                                ;CALLING SEQUENCE:

                                ;       JSR     R5,SETCL        ;CALL
                                ;       BR      NEWLPR          ;EXIT BRANCH AFTER ALL FOUR LNGTHS TESTED

017240  005767  002276          SETCL:  TST     QUICKX          ;EXIT AFTER ONLY ONE CHAR LNGTH ?
017244  001034                          BNE     2$              ;BR IF YES
017246  005267  002332                  INC     CLSEL           ;GENERATE NEW CHAR LNGTH SELECT CODE
```

```
5165  017252  022767  000004  002324          CMP     #4,CLSEL        ;DONE FOUR OF THEM ??
5166  017260  001436                          BEQ     2$              ;BR IF YES
5167  017262  005767  002252                  TST     QUICK           ;QUICK TEST FLAG SET ?
5168  017266  001407                          BEQ     1$              ;BR IF NOT
5169  017270  005267  002246                  INC     QUICKX          ;SET QUICK TEST EXIT FLAG
5170  017274  005067  002304                  CLR     CLSEL           ;DO ONLY 5 BIT CHARS
5171  017300  012767  177760  002274          MOV     #177760,CHRCNT  ;DO  ONLY 32. CHAR BUFFER
5172  017306  042767  000003  002262  1$:     BIC     #3,CURLPR       ;SET UP THE CURRENT LPR
5173  017314  056767  002264  002254          BIS     CLSEL,CURLPR
5174  017322  006367  002254                  ASL     CHRCNT          ;GENERATE CHAR COUNT
5175  017326  004767  000006                  JSR     PC,SUBUF1       ;GO SET UP THE OUTPUT BUFFER
5176  017332  062705  000002                  ADD     #2,R5           ;MOVE PC AROUND EXIT BRANCH
5177  017336  000205                  2$:     RTS     R5              ;RETURN
5178
5179                                          ;THIS ROUTINE IS CALLED TO LOAD THE OUTPUT DATA BUFFER WITH THE
5180                                          ;REQUIRED BINARY COUNT PATTERN
5181
5182                                          ;CALLING SEQUENCE:
5183
5184                                          ;       JSR     PC,SUBUF1       ;CALL
5185
5186  017340                                  SUBUF1:
5187  017340  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
5188  017342  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
5189  017344  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
5190  017346  005004                          CLR     R4              ;INIT CHAR GENERATOR
5191  017350  016703  002226                  MOV     CHRCNT,R3       ;SET UP LOAD COUNT
5192  017354  012702  032776                  MOV     #TBUF,R2        ;SET UP BUFFER POINTER
5193  017360  110422                  1$:     MOVB    R4,(R2)+        ;LOAD A CHAR
5194  017362  005204                          INC     R4              ;GENERATE NEXT CHAR
5195  017364  005203                          INC     R3              ;COUNT ONE LOADED
5196  017366  001374                          BNE     1$              ;BR TIL BUFFER FULL
5197  017370  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
5198  017372  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
5199  017374  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
5200  017376  000207                          RTS     PC              ;RETURN
5201
5202                                          ;THIS ROUTINE IS CALLED TO SET UP THE PARITY SELECT BITS
5203                                          ;IN THE CURRENT LPR TEST CONSTANT
5204
5205                                          ;CALLING SEQUENCE:
5206
5207                                          ;       JSR     R5,SETPAR       ;CALL
5208                                          ;       BR      NEWCL           ;EXIT BRANCH
5209
5210  017400  022767  177777  002200  SETPAR: CMP     #-1,PARBIT      ;DONE ALL PARITY COMBOS ?
5211  017406  001444                          BEQ     5$              ;BR IF YES
5212  017410  005767  002124                  TST     QUICK           ;QUICK TEST FLAG SET ?
5213  017414  001403                          BEQ     1$              ;BR IF NOT
5214  017416  012767  000060  002162          MOV     #60,PARBIT      ;CHECK ODD PARITY ONLY
5215  017424  042767  000060  002144  1$:     BIC     #60,CURLPR      ;SET PARITY SELECT BITS
5216  017432  056767  002150  002136          BIS     PARBIT,CURLPR
5217  017440  005767  002142                  TST     PARBIT          ;SELECT BITS 00 ?
5218  017444  001004                          BNE     2$              ;BR IF NOT
5219  017446  012767  000020  002132          MOV     #20,PARBIT      ;SET SELECT BITS TO 01
5220  017454  000417                          BR      4$              ;EXIT
```

```
5221  017456  022767  000020  002122  2$:     CMP     #20,PARBIT      ;SELECT BITS 10 ?
5222  017464  001004                          BNE     3$              ;BR IF NOT
5223  017466  012767  000060  002112          MOV     #60,PARBIT      ;MAKE SELECT BITS 11
5224  017474  000407                          BR      4$              ;EXIT
5225  017476  022767  000060  002102  3$:     CMP     #60,PARBIT      ;SELECT BITS 11 ?
5226  017504  001005                          BNE     5$              ;BR IF NOT
5227  017506  012767  177777  002072          MOV     #-1,PARBIT      ;SET EXIT FLAG
5228  017514  062705  000002          4$:     ADD     #2,R5           ;MOVE RETURN PC AROUND EXIT BRANCH
5229  017520  000205                  5$:     RTS     R5              ;RETURN
5230
5231                                  ;THIS ROUTINE IS CALLED TO SET UP FOR KEYBOARD INTERRUPTS
5232
5233  017522  012767  017546  160330  KYBD1:  MOV     #KYBD2,60       ;SET UP THE INPUT VECTOR
5234  017530  012767  000340  160324          MOV     #340,62
5235  017536  012767  000100  160014          MOV     #100,177560     ;ENABLE KYBD INTRS
5236  017544  000207                          RTS     PC              ;RETURN TO START TESTING
5237
5238                                  ;THIS ROUTINE SERVICES THE KEYBOARD INTERRUPT AND LOOKS FOR AN "S"
5239                                  ;BEING TYPED TO INDICATE ABORT AND PRINT STATISTICS
5240
5241  017546  117746  161374          KYBD2:  MOVB    @$TKB,-(SP)     ;GET CHAR TYPED
5242  017552  142716  000200                  BICB    #200,(SP)       ;CLEAR UNWANTED BITS
5243  017556  122716  000123                  CMPB    #123,(SP)       ;WAS AN "S" TYPED ?
5244  017562  001420                          BEQ     1$              ;BR IF YES
5245  017564  022767  000176  161346          CMP     #SWREG,SWR      ;USING SOFTWARE SWR?
5246  017572  001024                          BNE     2$              ;BRANCH IF YES
5247  017574  126727  161334  000001          CMPB    $AUTOB,#1       ;RUNNING IN AUTO MODE?
5248  017602  001420                          BEQ     2$              ;BRANCH IF YES
5249  017604  122716  000007                  CMPB    #7,(SP)         ;IS IT A <↑G>
5250  017610  001015                          BNE     2$              ;BRANCH IF NO
5251  017612  005726                          TST     (SP)+           ;POP
5252  017614  104401  013761                  TYPE    ,$CNTLG         ;TYPE <↑G>
5253  017620  000167  173332                  JMP     $GTSWR
5254  017624  005726          1$:     TST     (SP)+           ;POP
5255  017626  000005                          RESET                   ;ZAP THE WORLD
5256  017630  012706  001100                  MOV     #STACK,SP       ;RESET THE SP
5257  017634  004767  001566                  JSR     PC,CHPS1        ;GO CLEAR PSW
5258  017640  000167  164562                  JMP     PRSTAT          ;GO DUMP THE STATISTICS
5259  017644  005726          2$:     TST     (SP)+           ;POP
5260  017646  000002                          RTI                     ;RETURN AND FORGET IT
5261
5262                                  ;THIS ROUTINE SENDS A TEST BUFFER TO REMOTE DH11 LINE
5263
5264  017650  016701  001646          SENDP2: MOV     DHADR,R1                ;SET UP DH SCR ADDR
5265  017654  012711  004000                  MOV     #BIT11,(R1)             ;CLEAR THE DH11
5266  017660  016711  002266                  MOV     LINE,(R1)               ;SET LINE SELECT
5267  017664  162705  032776                  SUB     #TBUF,R5                ;SET UP BYTE COUNT
5268  017670  005405                          NEG     R5
5269  017672  010561  000010                  MOV     R5,BCR(R1)
5270  017676  012761  032776  000006          MOV     #TBUF,CAR(R1)           ;SET CURRENT ADDRESS
5271  017704  016761  001666  000004          MOV     CURLPR,LPR(R1)          ;SET LINE PARAMETERS
5272  017712  016761  001616  000012          MOV     LINMSK,BAR(R1)          ;ACTIVATE THE LINE
5273
5274  017720  005711          1$:     TST     (R1)                    ;DONE TRANSMITTING ??
5275  017722  100376                  BPL     1$                      ;BR IF NOT
5276  017724  000207                  RTS     PC                      ;RETURN TO CONTROL ROUTINE "SENDP1"
```

```
                                        ;THIS ROUTINE IS CALLED TO LOAD FILLERS INTO ECHO BUFFER
                                        ;
017726  116704  002506        LDFILL: MOVB    FILLB,R4                ;GET COUNT OF FILLERS
017732  012703  022375                MOV     #ECBUF+1,R3             ;SET UP BUFFER POINTER
017736  116767  161240  002430        MOVB    $TMPO,ECBUF             ;STORE LF CHAR
017744  116722  161232                MOVB    $TMPO,(R2)+             ;IN ECHO BUFFER TOO
017750  116723  002462        1$:     MOVB    FILLA,(R3)+             ;LOAD A FILLER CHAR
017754  116722  002456                MOVB    FILLA,(R2)+
017760  005304                        DEC     R4                      ;COUNT IT
017762  001372                        BNE     1$                      ;BR TIL REQUIRED COUNT LOADED
017764  116704  002450                MOVB    FILLB,R4                ;SET UP BYTE COUNT REG
017770  005204                        INC     R4
017772  005404                        NEG     R4
017774  010461  000010                MOV     R4,BCR(R1)              ;LOAD BCR REG
020000  000207                        RTS     PC                      ;RETURN TO RINT2
                                        ;
                                        ;THIS ROUTINE IS CALLED TO SET UP XMITTER SPEED
                                        ;
020002  104401                INXSP:  TYPE                            ;ASK USER TO TYPE SPEED
020004  026631                        XSMSG1                          ;"TRANSMITTER SPEED ?"
020006  012767  022156  002140  1$:     MOV     #XSPTAB,XSPTR           ;SET UP TABLE POINTER
020014  042767  036000  001554        BIC     #36000,CURLPR           ;INIT SPEED SELECT BITS
020022  104413                        RDDEC                           ;READ SPEED HE TYPED
020024  005716                        TST     (SP)                    ;DEFAULT TO 9600. BAUD ?
020026  001426                        BEQ     4$                      ;BR IF YES
020030  027716  002120        2$:     CMP     @XSPTR,(SP)             ;TYPED ENTRY MATCH TABLE ENTRY ?
020034  001010                        BNE     3$                      ;BR IF NOT
020036  062767  000002  002110        ADD     #2,XSPTR                ;POINT TO SELECT BITS IN TABLE
020044  057767  002104  001524        BIS     @XSPTR,CURLPR           ;SET SPEED SELECT BITS
020052  005726                        TST     (SP)+                   ;FIX STACK
020054  000417                        BR      5$                      ;CONTINUE
                                        ;
020056  062767  000004  002070  3$:     ADD     #4,XSPTR                ;POINT TO NEXT ENTRY
020064  022767  022242  002062        CMP     #XSPTAB+52.,XSPTR       ;END OF TABLE ??
020072  001356                        BNE     2$                      ;BR IF NOT
020074  104401                        TYPE                            ;ERROR MESSAGE
020076  026657                        XSMSG2                          ;"INVALID XMITR SPEED - TRY AGAIN"
020100  005726                        TST     (SP)+                   ;FIX THE SP
020102  000741                        BR      1$                      ;GO TRY AGAIN
                                        ;
020104  052767  032000  001464  4$:     BIS     #32000,CURLPR           ;SET UP DEFAULT TO 9600. BAUD
020112  005726                        TST     (SP)+                   ;FIX STACK POINTER
                                        ;
020114  000207                5$:     RTS     PC                      ;RETURN TO CALLER
                                        ;
                                        ;THIS ROUTINE IS CALLED TO SET UP RECEIVER SPEED
                                        ;
020116  104401                INRSP:  TYPE                            ;ASK USER TO TYPE SPEED
020120  026722                        RSMSG1                          ;"RECEIVER SPEED ?"
020122  012767  022244  002112  1$:     MOV     #RSPTAB,RSPTR           ;SET UP TABLE POINTER
020130  042767  001700  001440        BIC     #1700,CURLPR            ;INIT SPEED SELECT BITS
020136  104413                        RDDEC                           ;READ SPEED HE TYPED
020140  005716                        TST     (SP)                    ;DEFAULT TO 9600. BAUD ?
020142  001426                        BEQ     4$                      ;BR IF YES
020144  027716  002072        2$:     CMP     @RSPTR,(SP)             ;TYPED ENTRY MATCH TABLE ENTRY ?
```

```
5333  020150  001010                        BNE     3$                ;BR IF NOT
5334  020152  062767  000002  002062        ADD     #2,RSPTR          ;POINT TO SELECT BITS IN TABLE
5335  020160  057767  002056  001410        BIS     @RSPTR,CURLPR     ;SET SPEED SELECT BITS
5336  020166  005726                        TST     (SP)+             ;FIX STACK
5337  020170  000417                        BR      5$                ;CONTINUE
5338
5339  020172  062767  000004  002042  3$:   ADD     #4,RSPTR          ;POINT TO NEXT ENTRY
5340  020200  022767  022330  002034        CMP     #RSPTAB+52.,RSPTR ;END OF TABLE ??
5341  020206  001356                        BNE     2$                ;BR IF NOT
5342  020210  104401                        TYPE                      ;ERROR MESSAGE
5343  020212  026745                        RSMSG2                    ;"INVALID RCVR SPEED - TRY AGAIN"
5344  020214  005726                        TST     (SP)+             ;FIX THE SP
5345  020216  000741                        BR      1$                ;GO TRY AGAIN
5346
5347  020220  052767  001500  001350  4$:   BIS     #1500,CURLPR      ;SET UP DEFAULT TO 9600. BAUD
5348  020226  005726                        TST     (SP)+             ;FIX STACK POINTER
5349
5350  020230  000207                  5$:   RTS     PC                ;RETURN TO CALLER
5351
5352
5353                                  ;THIS ROUTINE IS CALLED TO SET UP LINE PARAMETERS FM KYBD
5354
5355  020232  105067  006005        LPRIN:  CLRB    EC2               ;CLEAR ECHO BUFFER
5356  020236  104401                        TYPE
5357  020240  026557                        LPMSG                     ;"DO YOU WANT TO CHANGE "LPR"?"
5358  020242  104410                  1$:   RDCHR
5359  020244  012600                        MOV     (SP)+,R0          ;GET WHAT HE TYPED
5360  020246  122700  000015                CMPB    #15,R0            ;WAS IT A <CR> ??
5361  020252  001405                        BEQ     2$                ;BR IF YES
5362  020254  110067  005763                MOVB    R0,EC2            ;ECHO WHAT HE TYPED
5363  020260  104401                        TYPE
5364  020262  026243                        EC2
5365  020264  000766                        BR      1$                ;GO WAIT FOR TERMINATOR
5366
5367  020266  105767  005751          2$:   TSTB    EC2               ;<CR> ONLY ??
5368  020272  001411                        BEQ     3$                ;BR IF YES
5369  020274  122767  000116  005741        CMPB    #116,EC2          ;WAS IT A "NO" ??
5370  020302  001405                        BEQ     3$                ;BR IF IT WAS
5371  020304  122767  000131  005731        CMPB    #131,EC2          ;WAS IT A "YES" ??
5372  020312  001347                        BNE     LPRIN             ;GO ASK ALL OVER AGAIN
5373  020314  000407                        BR      4$                ;BR IF IT WAS "YES"
5374  020316  005767  001254          3$:   TST     CURLPR            ;HAS LPR BEEN SET UP AT ALL ?
5375  020322  001016                        BNE     5$                ;BR IF YES USE PREVIOUS LPR
5376  020324  012767  033503  001244        MOV     #33503,CURLPR     ;SET DEFAULT 9600 BAUD,8 BITS NO PARITY
5377  020332  000412                        BR      5$                ;CONTINUE
5378  020334  004767  177442          4$:   JSR     PC,INXSP          ;GO INPUT AND SET UP XMIT SPEED
5379  020340  004767  177552                JSR     PC,INRSP          ;GO INPUT AND SET UP RCVR SPEED
5380  020344  004767  000022                JSR     PC,INCL           ;GO INPUT AND SET UP CHAR LENGTH
5381  020350  004767  000162                JSR     PC,INSB           ;GO INPUT AND SET UP NO. OF STOP BITS
5382  020354  004767  000274                JSR     PC,INPB           ;GO INPUT AND SET UP PARITY SELECTION
5383  020360  004767  000410          5$:   JSR     PC,INFCHR         ;GO INPUT AND SET UP FILLER CHAR
5384  020364  004767  000474                JSR     PC,INFCNT         ;GO INPUT AND SET UP FILLER COUNT
5385  020370  000207                        RTS     PC                ;RETURN TO CALLER
5386
5387                                  ;THIS ROUTINE IS CALLED TO SET UP CHAR LENCTH BITS
5388
```

```
5389  020372  105067  005645          INCL:   CLRB    EC2             ;CLEAR THE ECHO BUFFER
5390  020376  104401                          TYPE                    ;ASK FOR INPUT
5391  020400  027010                          CLMSG1                  ;"CHAR LENGTH - 6,7, OR 8 ?"
5392  020402  042767  000003  001166  1$:     BIC     #3,CURLPR       ;INIT CHAR LENGTH SELECT BITR
5393  020410  104410                          ROCHR                   ;GET THE CHAR HE TYPED
5394  020412  012600                          MOV     (SP)+,R0        ;GET WHAT HE TYPED
5395  020414  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR> ??
5396  020420  001405                          BEQ     11$             ;BR IF IT WAS
5397  020422  110067  005615                  MOVB    R0,EC2          ;ECHO WHAT HE TYPED
5398  020426  104401                          TYPE
5399  020430  026243                          EC2
5400  020432  000763                          BR      1$              ;GO WAIT FOR TERMINATOR
5401  020434  105767  005603          11$:    TSTB    EC2             ;<CR> ONLY ??
5402  020440  001432                          BEQ     4$              ;BR IF YES
5403  020442  142767  000060  005573          BICB    #60,EC2         ;STRIP ASCII
5404  020450  122767  000006  005565          CMPB    #6,EC2          ;6 BITS ?
5405  020456  001004                          BNE     2$              ;BR IF NOT
5406  020460  052767  000001  001110          BIS     #1,CURLPR       ;SET UP FOR 6 BIT CHARS
5407  020466  000422                          BR      5$              ;CONTINUE
5408  020470  122767  000007  005545  2$:     CMPB    #7,EC2          ;7 BITS ?
5409  020476  001004                          BNE     3$              ;BR IF NOT
5410  020500  052767  000002  001070          BIS     #2,CURLPR       ;SET UP FOR 7 BIT CHARS
5411  020506  000412                          BR      5$              ;CONTINUE
5412  020510  122767  000010  005525  3$:     CMPB    #8.,EC2         ;8 BITS ?
5413  020516  001403                          BEQ     4$              ;BR IF YES
5414  020520  104401                          TYPE                    ;ERROR MESSAGE
5415  020522  027046                          CLMSG2                  ;"INVALID CHAR LENGTH = TRY AGAIN"
5416  020524  000722                          BR      INCL            ;GO TRY AGAIN
5417  020526  052767  000003  001042  4$:     BIS     #3,CURLPR       ;SET UP FOR 8 BIT CHARS
5418  020534  000207                  5$:     RTS     PC              ;RETURN TO CALLER
5419
5420                                          ;THIS ROUTINE IS CALLED TO SET UP NO. OF STOP BITS
5421
5422  020536  105067  005501          INSB:   CLRB    EC2             ;CLEAR ECHO BUFFER
5423  020542  104401                          TYPE                    ;ASK FOR INPUT
5424  020544  027112                          SBMSG1                  ;"NO. OF STOP BITS - 1 OR 2 ?"
5425  020546  104410                  1$:     ROCHR                   ;GET CHAR TYPED
5426  020550  012600                          MOV     (SP)+,R0        ;GET WHAT HE TYPED
5427  020552  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR>
5428  020556  001405                          BEQ     11$             ;BR IF YES
5429  020560  110067  005457                  MOVB    R0,EC2          ;ECHO WHAT HE TYPED
5430  020564  104401                          TYPE
5431  020566  026243                          EC2
5432  020570  000766                          BR      1$              ;GO WAIT FOR TERMINATOR
5433  020572  105767  005445          11$:    TSTB    EC2             ;<CR> ONLY ??
5434  020576  001422                          BEQ     3$              ;BR IF YES
5435  020600  142767  000060  005435          BICB    #60,EC2         ;CLEAR ASCII JUNK
5436  020606  122767  000002  005427          CMPB    #2,EC2          ;2 STOP BITS ?
5437  020614  001004                          BNE     2$              ;BR IF NOT
5438  020616  052767  000004  000752          BIS     #4,CURLPR       ;SET UP FOR TWO STOP BITS
5439  020624  000412                          BR      4$              ;CONTINUE
5440  020626  122767  000001  005407  2$:     CMPB    #1,EC2          ;ONE STOP BIT ?
5441  020634  001403                          BEQ     3$              ;BR IF YES
5442  020636  104401                          TYPE                    ;ERROR MESSAGE
5443  020640  027151                          SBMSG2                  ;"INVALID NO. STOP BITS - TRY AGAIN"
5444  020642  000735                          BR      INSB            ;GO TRY AGAIN
```

```
5445  020644  042767  000004  000724  3$:     BIC     #4,CURLPR       ;SET UP FOR ONE STOP BIT
5446  020652  000207                  4$:     RTS     PC              ;RETURN TO CALLER
5447
5448                                  ;THIS ROUTINE IS CALLED TO SET UP PARITY SELECT BITS
5449
5450  020654  105067  005363          INPB:   CLRB    EC2             ;CLEAR ECHO BUFFER
5451  020660  104401                          TYPE                    ;ASK FOR INPUT
5452  020662  027217                          PBMSG1                  ;"PARITY - E,O, OR <CR> ?"
5453  020664  042767  000060  000704  1$:     BIC     #60,CURLPR      ;INIT FOR NO PARITY CHECKING
5454  020672  104410                          RDCHR                   ;GET CHAR TYPED
5455  020674  012600                          MOV     (SP)+,R0        ;GET WHAT HE TYPED
5456  020676  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR> ??
5457  020702  001405                          BEQ     11$             ;BR IF IT WAS
5458  020704  110067  005333                  MOVB    R0,EC2          ;ECHO THE CHAR TYPED
5459  020710  104401                          TYPE
5460  020712  026243                          EC2
5461  020714  000763                          BR      1$              ;GO WAIT FOR TERMINATOR
5462  020716  105767  005321          11$:    TSTB    EC2             ;<CR> ONLY ??
5463  020722  001423                          BEQ     4$              ;BR IF YES
5464  020724  122767  000105  005311          CMPB    #105,EC2        ;EVEN PARITY ??
5465  020732  001004                          BNE     2$              ;BR IF NOT
5466  020734  052767  000060  000634          BIS     #60,CURLPR      ;SET UP FOR EVEN PARITY
5467  020742  000413                          BR      4$              ;CONTINUE
5468  020744  122767  000117  005271  2$:     CMPB    #117,EC2        ;ODD PARITY
5469  020752  001004                          BNE     3$              ;BR IF NOT
5470  020754  052767  000020  000614          BIS     #20,CURLPR      ;SET UP FOR ODD PARITY
5471  020762  000403                          BR      4$              ;CONTINUE
5472  020764  104401                  3$:     TYPE                    ;ERROR MESSAGE
5473  020766  027265                          PBMSG2                  ;"INVALID PARITY - TRY AGAIN"
5474  020770  000731                          BR      INPB            ;GO TRY AGAIN
5475  020772  000207                  4$:     RTS     PC              ;RETURN TO CALLER
5476
5477                                  ;THIS ROUTINE IS CALLED TO SET UP "FILL" CHAR
5478
5479  020774  105067  005243          INFCHR: CLRB    EC2             ;CLEAR ECHO BUFFER
5480  021000  005067  001432                  CLR     FILLA           ;INIT TEMP STORAGE FOR CHAR
5481  021004  104401                          TYPE                    ;GO ASK FOR FILLER CHAR
5482  021006  027324                          FILC1                   ;"FILL CHAR ?"
5483  021010  005067  001420          1$:     CLR     DHFILL          ;ININ FILL LOCATION
5484  021014  104410                          RDCHR                   ;GET CHAR TYPED
5485  021016  012600                          MOV     (SP)+,R0        ;GET WHAT HE TYPED
5486  021020  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR> ??
5487  021024  001405                          BEQ     2$              ;BR IF YES
5488  021026  110067  005211                  MOVB    R0,EC2          ;ECHO WHAT HE TYPED
5489  021032  104401                          TYPE
5490  021034  026243                          EC2
5491  021036  000764                          BR      1$              ;GO WAIT FOR TERMINATOR
5492
5493  021040  105767  005177          2$:     TSTB    EC2             ;<CR> ONLY ??
5494  021044  001403                          BEQ     3$              ;BR IF YES
5495  021046  116767  005171  001361          MOVB    EC2,DHFILL+1    ;SET UP FILL CHAR
5496  021054  116767  001355  001354  3$:     MOVB    DHFILL+1,FILLA  ;SAVE FILL CHAR
5497  021062  000207                          RTS     PC              ;RETURN TO CALLER
5498
5499                                  ;THIS ROUTINE IS CALLED TO SET UP "FILL" COUNT
5500
```

```
5501  021064  005067  001350          INFCNT: CLR     FILLB           ;INIT TEMP. STORAGE FOR COUNT
5502  021070  104401                          TYPE                    ;ASK FOR COUNT
5503  021072  027352                          FILC2                   ;"FILL COUNT ?"
5504  021074  104412                          RDOCT                   ;GET OCTAL NO. TYPED
5505  021076  005716                          TST     (SP)            ;DEFAULT TO ONE ?
5506  021100  001403                          BEQ     1$              ;BR IF YES
5507  021102  111667  001326                  MOVB    (SP),DHFILL     ;SET UP COUNT TYPED
5508  021106  000403                          BR      2$              ;CONTINUE
5509  021110  112767  000001  001316  1$:     MOVB    #1,DHFILL       ;SET UP FOR  1 FILLER
5510  021116  005726                  2$:     TST     (SP)+           ;FIX THE SP
5511  021120  142767  000360  001306          BICB    #360,DHFILL     ;LIMIT COUNT TO 15. MAX
5512  021126  116767  001302  001304          MOVB    DHFILL,FILLB    ;SAVE IT FOR LATER
5513  021134  000207                          RTS     PC              ;RETURN TO CALLER
5514                          ;THIS ROUTINE CALLED TO SET UP ALTERNATING 1/0 PATTERN
5515
5516  021136  004767  000246          SUPATA: JSR     PC,CLALL        ;GO CLEAR XMIT AND RCV BUFFERS
5517  021142  016700  000434                  MOV     CHRCNT,R0               ;GET CHAR COUNT
5518  021146  012705  032776                  MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
5519  021152  112725  000252          1$:     MOVB    #252,(R5)+      ;LOAD A BYTE
5520  021156  005200                          INC     R0              ;COUNT IT
5521  021160  001374                          BNE     1$              ;BR TILL BUFFER FULL
5522  021162  000207                          RTS     PC              ;RETURN TO "DPATA" ROUTINE
5523
5524                          ;THIS ROUTINE IS CALLED TO SET UP UP COUNT PATTERN
5525
5526  021164  004767  000220          SUPATU: JSR     PC,CLALL        ;GO CLEAR BUFFERS
5527  021170  016700  000406                  MOV     CHRCNT,R0       ;GET COUNT OF CHARS TO LOAD
5528  021174  012705  032776                  MOV     #TBUF,R5        ;POINT TO XMITTR BUFFER
5529  021200  005004                          CLR     R4              ;INIT CHAR GENERATOR
5530  021202  110425                  1$:     MOVB    R4,(R5)+        ;LOAD ONE BYTE
5531  021204  105204                          INCB    R4              ;GENERATE NEXT BYTE
5532  021206  005200                          INC     R0              ;COUNT IT
5533  021210  001374                          BNE     1$              ;BR TIL BUFFER FULL
5534  021212  000207                          RTS     PC              ;RETURN TO "DPATU" ROUTINE
5535
5536                          ;THIS ROUTINE IS CALLED TO SET UP DOWN COUNT PATTERN
5537
5538  021214  004767  000170          SUPATD: JSR     PC,CLALL        ;CLEAR THE BUFFERS
5539  021220  016700  000356                  MOV     CHRCNT,R0       ;SET UP COUNT TO LOAD
5540  021224  012705  032776                  MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
5541  021230  012704  000377                  MOV     #377,R4         ;INIT CHAR GENERATOR
5542  021234  110425                  1$:     MOVB    R4,(R5)+        ;LOAD ONE BYTE
5543  021236  105304                          DECB    R4              ;GENERATE NEW CHAR
5544  021240  005200                          INC     R0              ;COUNT IT
5545  021242  001374                          BNE     1$              ;BR TIL BUFFER FULL
5546  021244  000207                          RTS     PC              ;RETURN TO "DPATA" ROUTINE
5547
5548                          ;THIS ROUTINE CALLED TO LOAD RANDOM DATA PATTERN
5549
5550  021246  004767  000136          SUPATR: JSR     PC,CLALL        ;GO CLEAR BUFFERS
5551  021252  016700  000324                  MOV     CHRCNT,R0       ;SET UP COUNT TO LOAD
5552  021256  012705  032776                  MOV     #TBUF,R5        ;POINT TO XMITTR BUFFER
5553  021262  012767  125252  001074          MOV     #125252,RANA    ;INIT RANDOM NUMBER GENERATOR
5554
5555  021270  066767  001070  001070  1$:     ADD     RANA,RANB       ;GENERATE RANDOM NO.
5556  021276  005567  001062                  ADC     RANA
```

```
5557  021302  066767  001060  001054          ADD     RANB,RANA
5558  021310  005567  001052                  ADC     RANB
5559
5560  021314  116725  001044                  MOVB    RANA,(R5)+      ;LOAD A BYTE
5561  021320  005200                          INC     R0              ;COUNT IT
5562  021322  001363                          BNE     1$              ;BR TIL BUFFER FULL
5563  021324  000207                          RTS     PC              ;RETURN TO "DPATR" ROUTINE
5564
5565                                  ;THIS ROUTINE LOADS A SINGLE CHAR THROUGHOUT BUFFER
5566
5567  021326  004767  000056          SUPATS: JSR     PC,CLALL        ;GO CLEAR BUFFERS
5568  021332  016700  000244                  MOV     CHRCNT,R0       ;INIT CHAR COUNTER
5569  021336  012705  032776                  MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
5570  021342  116725  001010          1$:     MOVB    SINGLE,(R5)+    ;LOAD ONE CHAR
5571  021346  005200                          INC     R0              ;COUNT IT
5572  021350  001374                          BNE     1$              ;BR TIL BUFFER FULL
5573  021352  000207                          RTS     PC              ;RETURN TO "DPATS" ROUTINE
5574
5575                                  ;THIS ROUTINE CALLED TO INIT CHAR LENGTH MASK FOR PATTERNS TESTS
5576
5577  021354  016700  000216          SUCLMK: MOV     CURLPR,R0       ;GET CURRENT "LPR"
5578  021360  012767  000340  001002          MOV     #340,CLMSK      ;INIT FOR 5 BIT CHARS
5579  021366  042700  177774                  BIC     #177774,R0      ;MASK OFF ALL BUT CL BITS
5580  021372  005700                  1$:     TST     R0              ;DONE SETUP ?
5581  021374  001404                          BEQ     2$              ;BR IF YES
5582  021376  106367  000766                  ASLB    CLMSK           ;SHIFT MASK LEFT
5583  021402  005300                          DEC     R0              ;COUNT IT
5584  021404  000772                          BR      1$              ;GO SEE IF ITS RIGHT ON
5585  021406  000207                  2$:     RTS     PC              ;RETURN TO CALLER
5586                                  ;ROUTINE TO CLEAR XMIT AND RECEIVER BUFFERS
5587
5588  021410  012700  032776          CLALL:  MOV     #TBUF,R0        ;SET UP POINTER
5589  021414  005020                  1$:     CLR     (R0)+           ;CLEAR A WORD
5590  021416  022700  034126                  CMP     #ENBUFS,R0      ;DONE ALL LOCATIONS ?
5591  021422  001374                          BNE     1$              ;BR IF NOT
5592  021424  000207                          RTS     PC
```

```
5593                                    ;THIS ROUTINE IS CALLED TO SET PSW PRIORITY TO 000 IN ORDER
5594                                    ;TO BE LSI11 COMPATIBLE
5595
5596  021426  012746  000000          CHPS1:  MOV     #0,-(SP)        ;NEW PSW
5597  021432  012746  021440                  MOV     #1$,-(SP)       ;NEW PC
5598  021436  000002                          RTI                     ;CHANGE PSW
5599  021440  000207                  1$:     RTS     PC              ;RETURN TO CALLING TEST
5600
5601                                    ;THIS ROUTINE DOES THE SAME THING EXCEPT IT SET THE PSW
5602                                    ;PRIORITY TO 340 (LEVEL 7 ) TO LOCK OUT INTRS
5603
5604  021442  012746  000340          CHPS2:  MOV     #340,-(SP)      ;NEW PSW
5605  021446  012746  021454                  MOV     #1$,-(SP)       ;NEW PC
5606  021452  000002                          RTI                     ;CHANGE THE PSW
5607  021454  000207                  1$:     RTS     PC              ;RETURN TO CALLING TEST
5608
5609                                    ;THIS ROUTINE IS ALSO FOR LSI11 COMPATIBILITY AND IT IS CALLED
5610                                    ;TO SAVE THE PSW IN "$TMPO"
5611
5612  021456  005046                  SAPS:   CLR     -(SP)           ;TEMP STORAGE TO SAVE PSW
5613  021460  016746  156350                  MOV     34,-(SP)        ;SAVE TRAP VECTOR POINTER
5614  021464  012767  021474  156342          MOV     #1$,34          ;GO TO 1$ ON TRAP
5615  021472  104400                          TRAP                    ;GO TO IT
5616  021474  016666  000002  000006  1$:     MOV     2(SP),6(SP)     ;GET PSW SAVED
5617  021502  012716  021510                  MOV     #2$,(SP)        ;GO TO 2$ ON RTI
5618  021506  000002                          RTI
5619  021510  012667  156320          2$:     MOV     (SP)+,34        ;RESTORE VECTOR
5620  021514  012667  157462                  MOV     (SP)+,$TMPO     ;FINALLY SAVE PSW IN $TMPO
5621  021520  000207                          RTS     PC
5622
5623
```

```
                        .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 REG.
000014                      BKR=14          ;INDEX CONST. TO ACCESS BREAK CONTROL REG.
000016                      SSR=16          ;INDEX CONST. TO ACCESS SILO STATUS REG.

021522  000000    DHADR:  0               ;HOLDS THE "SCR" ADDRESS OF THE DH11 UNDER TEST
021524  000000    DHVCT:  0               ;HOLDS THE 1ST VECTOR ADDRESS OF THE DH11 UNDER TEST
021526  000000    SELMSK: 0               ;BIT TST MARKER FOR SELECTING DH11'S
021530  000001    DHSEL:  1               ;SPECIFIES DH11'S SELECTED FOR TEST
021532  177777    LINSEL: 177777          ;SPECIFIES LINES TO TEST
021534  000000    LINMSK: 0               ;MARKER USED TO TEST FOR LINES TO TEST
021536  000000    DRPLIN: 0               ;DROPPED LINE FLAGS

021540  000000    QUICK:  0               ;QUICK TEST FLAG - ALLOWS SINGLE PATTERN TEST
                                          ;ON ALL TESTS NOT USING 9600. BAUD
021542  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

021544  033500    LPRTAB: 33500           ;9600 BAUD
021546  004200            4200            ;75 BAUD
021550  006300            6300            ;110 BAUD
021552  010400            10400           ;134.5 BAUD
021554  012500            12500           ;150 BAUD
021556  014600            14600           ;200 BAUD
021560  016700            16700           ;300 BAUD
021562  021000            21000           ;600 BAUD
021564  023100            23100           ;1200 BAUD
021566  025200            25200           ;1800 BAUD
021570  027300            27300           ;2400 BAUD
021572  031400            31400           ;4800 BAUD
021574  002100            2100            ;50 BAUD

021576  000000    CURLPR: 0               ;CONTAINS CURRENT "LPR" CONSTANT

021600  000000    LPRPTR: 0               ;CONTAINS POINTER TO LPR TABLE
021602  000000    CHRCNT: 0               ;LOADED WITH CURRENT CHAR COUNT

021604  000000    CLSEL:  0               ;CHAR LENGTH SELECT PARAMETER
021606  000000    PARBIT: 0               ;PARITY SELECT PARAMETER

021610  000000    RDONE:  0               ;SOFTWARE DONE FLAG
```

```
                                RBFEND: 0                       ;HOLDS END OF BUFFER ADDRESS

                                ;DH11 ADDRESS TABLE - THIS TABLE CONTAINS THE "SCR" ADDRESS FOR UP TO
                                ;SIXTEEN DH11'S

                                DHADTB: 160020          ;ADDRESS OF FIRST DH11
                                        160040          ;ADDRESS OF SECOND DH11
                                        160060
                                        160100
                                        160120
                                        160140
                                        160160
                                        160200
                                        160220
                                        160240
                                        160260
                                        160300
                                        160320
                                        160340
                                        160360
                                        160400          ;ADDRESS OF THE LAST DH11

                                ;DH11 VECTOR TABLE - THIS TABLE CONTAINS THE VECTOR ADDRESSES FOR UP
                                ;TO SIXTEEN DH11'S

                                DHVCTB: 330             ;ADDRESS OF VECTOR FOR FIRST DH11
                                        350             ;ADDRESS OF VECTOR FOR SECOND DH11
                                        370
                                        410
                                        430
                                        450
                                        470
                                        510
                                        530
                                        550
                                        570
                                        610
                                        630
                                        650
                                        670
                                        710             ;ADDRESS OF VECTOR FOR LAST DH11

                                VCFLG:  0               ;VECTOR DISPLACEMENT FLAG


                                ;BR PRIORITY LEVEL TABLE - THIS TABLE CONTAINS THE PRIORITY LEVELS
                                ;FOR UP TO SIXTEEN DH11'S - THE RCVR LEVEL IS STORED IN THE LOW BYTE
                                ;AND THE XMTTR LEVEL IN THE HIGH BYTE

                                BRLVL:  120240          ;BRLEVELS FOR FIRST DH11
                                        120240          ;BR LEVELS FOR SECOND DH11
                                        120240
                                        120240
                                        120240
                                        120240
                                        120240
```

```
5736   021734  120240                          120240
5737   021736  120240                          120240
5738   021740  120240                          120240
5739   021742  120240                          120240
5740   021744  120240                          120240
5741   021746  120240                          120240
5742   021750  120240                          120240
5743   021752  120240                          120240
5744   021754  120240                          120240          ;BR LEVELS FOR LAST DH11
5745
5746                                   ;THIS DH ADDRESS TABLE IS FILLED BY THE AUTOSIZER.
5747
5748   021756                          DHADRS:
5749   021756  000000                  .WORD   0
5750   021760  000000                  .WORD   0
5751   021762  000000                  .WORD   0
5752   021764  000000                  .WORD   0
5753   021766  000000                  .WORD   0
5754   021770  000000                  .WORD   0
5755   021772  000000                  .WORD   0
5756   021774  000000                  .WORD   0
5757   021776  000000                  .WORD   0
5758   022000  000000                  .WORD   0
5759   022002  000000                  .WORD   0
5760   022004  000000                  .WORD   0
5761   022006  000000                  .WORD   0
5762   022010  000000                  .WORD   0
5763   022012  000000                  .WORD   0
5764
5765                                   ;THIS DH VECTOR TABLE IS FILLED BY THE AUTOSIZER.
5766
5767   022014                          DHVEC:
5768   022014  000000                  .WORD   0
5769   022016  000000                  .WORD   0
5770   022020  000000                  .WORD   0
5771   022022  000000                  .WORD   0
5772   022024  000000                  .WORD   0
5773   022026  000000                  .WORD   0
5774   022030  000000                  .WORD   0
5775   022032  000000                  .WORD   0
5776   022034  000000                  .WORD   0
5777   022036  000000                  .WORD   0
5778   022040  000000                  .WORD   0
5779   022042  000000                  .WORD   0
5780   022044  000000                  .WORD   0
5781   022046  000000                  .WORD   0
5782
5783                                   ;THIS DM ADDRESS TABLE IF FILLED BY THE AUTOSIZER.
5784
5785   022050                          DMADRS:
5786   022050  000000                  .WORD   0
5787   022052  000000                  .WORD   0
5788   022054  000000                  .WORD   0
5789   022056  000000                  .WORD   0
5790   022060  000000                  .WORD   0
5791   022062  000000                  .WORD   0
```

```
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0

                                ;THIS DM VECTOR TABLE IS FILLED BY THE AUTOSIZER.

                                DMVEC:
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0
                                .WORD   0

                                ADRVEC: 0               ;ADDRESSES BETWEEN VECTORS - FILLED BY THE AUTOSIZER.
                                $DHSEL: 0               ;DEVICE SELECT PARAMETER - FILLED BY THE AUTOSIZER.
                                DHRLVL: .BYTE   0               ;BR LEVEL FOR RCVR
                                DHTLVL: .BYTE   0               ;BR LEVEL FOR XMITTER
                                DHNUM:  0               ;CONTAINS NUMBER OF THE DH11 UNDER TEST
                                LINE:   0               ;CONTINES NUMBER OF THE LINE UNDER TEST
                                ;TABLES USED TO SELECT XMITTR AND RCVR SPEEDS

                                ;THE TABLES CONSIST OF 13. TWO WORD ENTRIES - ONE FOR EACH
                                ;ALLOWABLE BAUD RATE. THE FIRST WORD IS THE ACTUAL BAUD RATE
                                ;IN DECIMAL AND THE SECOND WORD IS THE ENCODED BINARY WORD
                                ;THAT SETS THAT BAUD RATE IN THE "LPR"

                                XSPTR:  0                       ;CONTAINS POINTER TO FOLLOWING TABLE

                                XSPTAB: 50.                     ;50. BAUD
                                        2000
                                        75.                     ;75. BAUD
                                        4000
                                        110.                    ;110. BAUD
                                        6000
                                        1345.                   ;134.5 BAUD
                                        10000
                                        150.                    ;150. BAUD
                                        12000
                                        200.                    ;200. BAUD
                                        14000
                                        300.                    ;300 BAUD
```

```
5848  022210  016000                        16000
5849  022212  001130                        600.                    ;600. BAUD
5850  022214  020000                        20000
5851  022216  002260                        1200.                   ;1200. BAUD
5852  022220  022000                        22000
5853  022222  003410                        1800.                   ;1800. BAUD
5854  022224  024000                        24000
5855  022226  004540                        2400.                   ;2400. BAUD
5856  022230  026000                        26000
5857  022232  011300                        4800.                   ;4800. BAUD
5858  022234  030000                        30000
5859  022236  022600                        9600.                   ;9600. BAUD
5860  022240  032000                        32000
5861  022242  000000              RSPTR:    0                       ;CONTAINS POINTER TO FOLLOWING TABLE
5862
5863  022244  000062              RSPTAB:   50.                     ;50. BAUD
5864  022246  000100                        100
5865  022250  000113                        75.                     ;75. BAUD
5866  022252  000200                        200
5867  022254  000156                        110.                    ;110. BAUD
5868  022256  000300                        300
5869  022260  002501                        1345.                   ;134.5 BAUD
5870  022262  000400                        400
5871  022264  000226                        150.                    ;150. BAUD
5872  022266  000500                        500
5873  022270  000310                        200.                    ;200. BAUD
5874  022272  000600                        600
5875  022274  000454                        300.                    ;300 BAUD
5876  022276  000700                        700
5877  022300  001130                        600.                    ;600. BAUD
5878  022302  001000                        1000
5879  022304  002260                        1200.                   ;1200. BAUD
5880  022306  001100                        1100
5881  022310  003410                        1800.                   ;1800. BAUD
5882  022312  001200                        1200
5883  022314  004540                        2400.                   ;2400. BAUD
5884  022316  001300                        1300
5885  022320  011300                        4800.                   ;4800. BAUD
5886  022322  001400                        1400
5887  022324  022600                        9600.                   ;9600. BAUD
5888  022326  001500                        1500
5889
5890                          ;ADDRESS POINTERS TO SET UP TABLES WHEN INPUTTING PARAMETERS
5891
5892  022330  000000              ADPTR:    0                       ;POINTS TO ADDRESS TABLE
5893  022332  000000              VCPTR:    0                       ;POINTS TO VECTOR TABLE
5894  022334  000000              BRPTR:    0                       ;POINTS TO BR LEVEL TABLE
5895
5896  022336  000000              TITFLG:   0                       ;FLAG TO ALLOW PRINTING TITLE ONLY ONCE
5897  022340  000000              TIMEA:    0                       ;GENERAL PURPOSE TIMERS
5898  022342  000000              TIMEB:    0
5899
5900  022344  000000              CEXIT:    0                       ;CONTROL-C EXIT FLAG FM ECHO TESTS
5901  022346  000000              DPFLG:    0                       ;PATTERNS TEST FLAG
5902  022350  000000              DATCNT:   0                       ;ITERATION COUNTER FOR PATTERNS TEST
5903  022352  000000              DATPAT:   0                       ;FLAGS TYPE PATTERN
```

```
5904   022354  000000                  PATFLG: 0                       ;DATA PATTERNS (CR) SEQUENCE FLAG
5905   022356  000000                  SINGLE: 0                       ;HOLDS SINGLE CHAR TEST PATTERN
5906   022360  000000                  RETFLG: 0                       ;ECHO TEST RETURN FLAG FM SETUP
5907   022362  000012                  PATLIM: 10.                     ;PATTERNS TESTS ITERATION COUNT
5908   022364  000000                  RANA:   0                       ;RANDOM NO. ACCUMULATORS
5909   022366  000000                  RANB:   0
5910   022370  000000                  CLMSK:  0                       ;CHAR LENGTH BIT CLR MASK
5911   022372  000000                  EXFLAG: 0                       ;ECHO TEST EXIT FLAGS
5912   022374  000020                  ECBUF:  .BLKW   16.             ;DATA BUFFER FOR SINGLE LINE ECHO TEST
5913   022434  000000                  DHFILL: 0                       ;FILL CHAR AND COUNT FOR SINGLE LINE
5914                                                                   ;ECHO TESTS
5915   022436  000000                  FILLA:  0                       ;TEMP STORAGE FOR FILLER CHAR
5916   022440  000000                  FILLB:  0                       ;SAME FOR COUNT
5917
5918
```

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

                                ;INFORMATION FOR MESSAGE 1

022442 047516 020116 054105     EM1:    .ASCIZ  'NON EX MEMORY ERROR - DROPPED LINE #  '
022450 046440 046505 051117
022456 020131 051105 047522
022464 020122 020055 051104
022472 050117 042520 020104
022500 044514 042516 021440
022506 020040    000
022511    040 050050 024503     DH1:    .ASCIZ  ' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B'
022516 020040 041440 051125
022524 050114 020122 042040
022532 053105 042101 020122
022540 051040 043505 042101
022546 020122 020040 040527
022554 020123 020040 020040
022562 027523 000102
                                .EVEN
022566 001116 021576 001164     DT1:    .WORD   $ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4,0
022574 001166 001170 001172
022602 000000
022604    000    000    000     DF2:    .BYTE   0,0,0,0,0,0,0,0
022607    000    000    000
022612    000    000

                                ;INFORMATION FOR MESSAGE 2

022614 051124 047101 046523     EM2:    .ASCIZ  'TRANSMITTER FALSE INTERRUPT - DROPPED LINE #  '
022622 052111 042524 020122
022630 040506 051514 020105
022636 047111 042524 051122
022644 050125 020124 020055
022652 051104 050117 042520
022660 020104 044514 042516
022666 021440 020040    000

                                ;INFORMATION FOR MESSAGE 3

022673    102 043125 042506     EM3:    .ASCIZ  'BUFFER ACTIVE REGISTER ERROR - DROPPED LINE #  '
022700 020122 041501 044524
022706 042526 051040 043505
022714 051511 042524 020122
022722 051105 047522 020122
022730 020055 051104 050117
022736 042520 020104 044514
022744 042516 021440 020040
022752    000

                                ;INFORMATION FOR MESSAGE 4
```

```
022753     102  052131  020105  EM4:   .ASCIZ  'BYTE COUNT REGISTER ERROR - DROPPED LINE #  '
022760  047503  047125  020124
022766  042522  044507  052123
022774  051105  042440  051122
023002  051117  026440  042040
023010  047522  050120  042105
023016  046040  047111  020105
023024  020043  000040

                                ;INFORMATION FOR MESSAGE 5

023030  052503  051122  047105  EM5:   .ASCIZ  'CURRENT ADDRESS REGISTER ERROR - DROPPED LINE #  '
023036  020124  042101  051104
023044  051505  020123  042522
023052  044507  052123  051105
023060  042440  051122  051117
023066  026440  042040  047522
023074  050120  042105  046040
023102  047111  020105  020043
023110  000040

                                ;INFORMATION FOR MESSAGE 6

023112  044523  047514  047440  EM6:   .ASCIZ  'SILO OVERFLOW ERROR - DROPPED LINE #  '
023120  042526  043122  047514
023126  020127  051105  047522
023134  020122  020055  051104
023142  050117  042520  020104
023150  044514  042516  021440
023156  020040     000

                                ;INFORMATION FOR MESSAGE 7

023161     122  041505  044505  EM7:   .ASCIZ  'RECEIVER FALSE INTERRUPT - DROPPED LINE #  '
023166  042526  020122  040506
023174  051514  020105  047111
023202  042524  051122  050125
023210  020124  020055  051104
023216  050117  042520  020104
023224  044514  042516  021440
023232  020040     000

                                ;INFORMATION FOR MESSAGE 10

023235     111  053116  046101  EM10:  .ASCIZ  'INVALID DATA IN SILO - DROPPED LINE #  '
023242  042111  042040  052101
023250  020101  047111  051440
023256  046111  020117  020055
023264  051104  050117  042520
023272  020104  044514  042516
023300  021440  020040     000
023305     040  050050  024503  DH2:   .ASCIZ  ' (PC)   CURLPR  CHAR #  WASADR  SHBADR   WAS     S/B'
023312  020040  041440  051125
023320  050114  020122  041440
023326  040510  020122  020043
023334  053440  051501  042101
```

```
023342  020122  051440  041110
023350  042101  020122  020040
023356  040527  020123  020040
023364  020040  027523  000102
                                .EVEN
023372  001116  021576  001162  DT2:    .WORD   $ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4,0
023400  001164  001166  001170
023406  001172  000000

                                ;INFORMATION FOR MESSAGE 11

023412  040504  040524  042440  EM11:   .ASCIZ  'DATA ERROR - LINE #  '
023420  051122  051117  026440
023426  046040  047111  020105
023434  020043  000040

                                ;INFORMATION FOR MESSAGE 12

023440  042524  052123  052040  EM12:   .ASCIZ  'TEST TIMEOUT - DROPPED LINE #  '
023446  046511  047505  052125
023454  026440  042040  047522
023462  050120  042105  046040
023470  047111  020105  020043
023476  000040
023500  024040  041520  020051  DH3:    .ASCIZ  ' (PC)   CURLPR  RTOTAL  XTOTAL  RDONE'
023506  020040  052503  046122
023514  051120  020040  052122
023522  052117  046101  020040
023530  052130  052117  046101
023536  020040  042122  047117
023544  000105
                                .EVEN
023546  001116  021576  001202  DT3:    .WORD   $ERRPC,CURLPR,$TMP0,$TMP1,RDONE,0
023554  001204  021610  000000

                                ;INFORMATION FOR MESSAGE 13

023562  001202  001204  001206  DT4:    .WORD   $TMP0,$TMP1,$TMP2,$TMP3,$TMP4,$TMP5,$TMP6,0
023570  001210  001212  001214
023576  001216  000000
023602     000     001     001  DF1:    .BYTE   0,1,1,1,1,1,1,0
023605     001     001     001
023610     001     000

                                ;INFORMATION FOR MESSAGE 14

023612  052502  020123  051105  EM14:   .ASCIZ  'BUS ERROR TRAP TO 04'
023620  047522  020122  051124
023626  050101  052040  020117
023634  032060     000
023637     040  050050  024503  DH4:    .ASCIZ  ' (PC)    (PS)    (SP)   TRAPPC  TRAPPS'
023644  020040  020040  050050
023652  024523  020040  020040
023660  051450  024520  020040
023666  052040  040522  050120
023674  020103  052040  040522
```

```
023702 050120 000123
                                .EVEN
023706 001116 001202 001176     DT5:    .WORD   $ERRPC,$TMP0,$REG6,$REG1,$REG2,0
023714 001164 001166 000000

                                ;INFORMATION FOR MESSAGE 15

023722 051522 042126 044440     EM15:   .ASCIZ  'RSVD INSTR TRAP TO 10'
023730 051516 051124 052040
023736 040522 020120 047524
023744 030440 000060

                                ;INFORMATION FOR MESSAGE 16

023750 044523 043516 042514     EM16:   .ASCIZ  'SINGLE LINE ECHO TEST - INTR WAIT TIMEOUT'
023756 046040 047111 020105
023764 041505 047510 052040
023772 051505 020124 020055
024000 047111 051124 053440
024006 044501 020124 044524
024014 042515 052517 000124
024022 024040 041520 020051     DH5:    .ASCIZ  ' (PC)   DEVADR   LINE    (SCR)  CURLPR  EXFLAG'
024030 020040 042504 040526
024036 051104 020040 046040
024044 047111 020105 020040
024052 024040 041523 024522
024060 020040 052503 046122
024066 051120 020040 054105
024074 046106 043501    000
       024102                   .EVEN
024102 001116 001164 022152     DT6:    .WORD   $ERRPC,$REG1,LINE,$TMP0,CURLPR,EXFLAG,0
024110 001202 021576 022372
024116 000000


                                ;INFORMATION FOR MESSAGE 17

024120 046101 042524 047122     EM17:   .ASCIZ  'ALTERNATING 1/0 PATTERN TEST DONE'
024126 052101 047111 020107
024134 027461 020060 040520
024142 052124 051105 020116
024150 042524 052123 042040
024156 047117 000105
024162 024040 041520 020051     DH6:    .ASCIZ  ' (PC)   DEVADR   LINE   CURLPR  ICOUNT'
024170 020040 042504 040526
024176 051104 020040 046040
024204 047111 020105 020040
024212 052503 046122 051120
024220 020040 041511 052517
024226 052116    000
       024232                   .EVEN
024232 001116 021522 022152     DT7:    .WORD   $ERRPC,DHADR,LINE,CURLPR,$REG0,0
024240 021576 001162 000000

                                ;INFORMATION FOR MESSAGE 20
```

```
                                        ;INFORMATION FOR MESSAGE 20
024246  044502  040516  054522  EM20:   .ASCIZ  'BINARY UP COUNT PATTERN TEST DONE'
024254  052440  020120  047503
024262  047125  020124  040520
024270  052124  051105  020116
024276  042524  052123  042040
024304  047117  000105
                                        ;INFORMATION FOR MESSAGE 21
024310  044502  040516  054522  EM21:   .ASCIZ  'BINARY DOWN COUNT PATTERN TEST DONE'
024316  042040  053517  020116
024324  047503  047125  020124
024332  040520  052124  051105
024340  020116  042524  052123
024346  042040  047117  000105
                                        ;INFORMATION FOR MESSAGE 22
024354  040522  042116  046517  EM22:   .ASCIZ  'RANDOM DATA PATTERN TEST DONE'
024362  042040  052101  020101
024370  040520  052124  051105
024376  020116  042524  052123
024404  042040  047117  000105
                                        ;INFORMATION FOR MESSAGE 23
024412  044523  043516  042514  EM23:   .ASCIZ  'SINGLE CHAR PATTERN TEST DONE'
024420  041440  040510  020122
024426  040520  052124  051105
024434  020116  042524  052123
024442  042040  047117  000105
                                        ;INFORMATION    FOR MESSAGE 24
024450  054524  042520  020104  EM24:   .ASCIZ  'TYPED BUFFER PATTERN TEST DONE'
024456  052502  043106  051105
024464  050040  052101  042524
024472  047122  052040  051505
024500  020124  047504  042516
024506      000
                                        ;INFORMATION FOR MESSAGE 25
024507      104 052101  020101  EM25:   .ASCIZ  'DATA PATTERNS TEST TIMEOUT'
024514  040520  052124  051105
024522  051516  052040  051505
024530  020124  044524  042515
024536  052517  000124
024542  024040  041520  020051  DH7:    .ASCIZ  ' (PC)   DEVADR   LINE   CURLPR  ICOUNT  PATCDE'
024550  020040  042504  040526
024556  051104  020040  046040
024564  047111  020105  020040
024572  052503  046122  051120
024600  020040  041511  052517
024606  052116  020040  040520
```

```
 6199  024614  041524  042504     000
 6200          024622                    .EVEN
 6201  024622  001116  021522  022152  DT10:   .WORD   $ERRPC,DHADR,LINE,CURLPR,$REG0,$REG1,0
 6202  024630  021576  001162  001164
 6203  024636  000000
 6204
 6205                                    .EVEN
```

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

024640  005015  040515  047111  TITLE:  .ASCIZ  <15><12>'MAINDEC-11-DZDHN-B DH11 DATA RELIABILITY TEST'<15><12>
024646  042504  026503  030461
024654  042055  042132  047110
024662  041055  042040  030510
024670  020061  040504  040524
024676  051040  046105  040511
024704  044502  044514  054524
024712  052040  051505  006524
024720  000012
024722  005015  042524  052123  TITLE2: .ASCIZ  <15><12>'TESTING DH11 #  '<15><12>
024730  047111  020107  044104
024736  030461  021440  020040
024744  005015      000
024747      015  052012  050131  INMSG1: .ASCIZ  <15><12>'TYPE SCR ADDRESS FOR FIRST DH11'<15><12>
024754  020105  041523  020122
024762  042101  051104  051505
024770  020123  047506  020122
024776  044506  051522  020124
025004  044104  030461  005015
025012      000
025013      015  052012  050131  INMSG2: .ASCIZ  <15><12>'TYPE VECTOR ADDRESS FOR FIRST DH11'<15><12>
025020  020105  042526  052103
025026  051117  040440  042104
025034  042522  051523  043040
025042  051117  043040  051111
025050  052123  042040  030510
025056  006461  000012
025062  005015  054524  042520  INMSG3: .ASCIZ  <15><12>'TYPE DH11 DEVICE SELECTION PARAMETER'<15><12>
025070  042040  030510  020061
025076  042504  044526  042503
025104  051440  046105  041505
025112  044524  047117  050040
025120  051101  046501  052105
025126  051105  005015      000
025133      015  044412  053116  INMSG4: .ASCIZ  <15><12>'INVALID DH11 SCR ADDRESS - TRY AGAIN'<15><12>
025140  046101  042111  042040
025146  030510  020061  041523
025154  020122  042101  051104
025162  051505  020123  020055
025170  051124  020131  043501
025176  044501  006516  000012
025204  005015  047111  040526  INMSG5: .ASCIZ  <15><12>'INVALID DH11 VECTOR ADDRESS - TRY AGAIN'<15><12>
025212  044514  020104  044104
025220  030461  053040  041505
025226  047524  020122  042101
025234  051104  051505  020123
025242  020055  051124  020131
025250  043501  044501  006516
025256  000012
025260  005015  047531  020125  INMSG6: .ASCIZ  <15><12>'YOU MUST SELECT AT LEAST ONE DH11'<15><12>
025266  052515  052123  051440
```

```
 6262  025274  046105  041505  020124
 6263  025302  052101  046040  040505
 6264  025310  052123  047440  042516
 6265  025316  042040  030510  006461
 6266  025324  000012
 6267  025330  005015  042504  051120  INMSG7: .ASCIZ  <15><12>'DEPRESS "CONTINUE" TO START TESTING'<15><12>
 6268  025336  051505  020123  041442
 6269  025344  047117  044524  052516
 6270  025350  021105  052040  020117
 6271  025356  052123  051101  020124
 6272  025364  042524  052123  047111
 6273  025372  006507  000012
 6274  025376  005015  054524  042520  INMSGA: .ASCIZ  <15><12>'TYPE LINE SELECTION PARAMETER'<15><12>
 6275  025404  046040  047111  020105
 6276  025412  042523  042514  052103
 6277  025420  047511  020116  040520
 6278  025426  040522  042515  042524
 6279  025434  006522  000012
 6280
 6281  025440  005015  054524  042520  VCWC:   .ASCIZ  <15><12>'TYPE NO. OF ADDRESSES (OCTAL) BETWEEN VECTORS (10 OR 20)'<15><1
 6282  025446  047040  027117  047440
 6283  025454  020106  042101  051104
 6284  025462  051505  042523  020123
 6285  025470  047450  052103  046101
 6286  025476  020051  042502  053524
 6287  025504  042505  020116  042526
 6288  025512  052103  051117  020123
 6289  025520  030450  020060  051117
 6290  025526  031040  024460  005015
 6291  025534     000
 6292  025535     015  042012  020110  STMSG1: .ASCIZ  <15><12>'DH #   STATISTICS:'
 6293  025542  020043  020040  052123
 6294  025550  052101  051511  044524
 6295  025556  051503  000072
 6296  025562  005015  042524  052123  STMSG2: .ASCIZ  <15><12>'TESTING LINE #  '<15><12>
 6297  025570  047111  020107  044514
 6298  025576  042516  021440  020040
 6299  025604  005015     000
 6300  025607     015  046012  047111  STMSG3: .ASCIZ  <15><12>'LINE #   WAS DROPPED'<15><12>
 6301  025614  020105  020043  020040
 6302  025622  040527  020123  051104
 6303  025630  050117  042520  006504
 6304  025636  000012
 6305  025640  005015  044514  042516  STMSG4: .ASCIZ  <15><12>'LINE #  RTOTAL  XTOTAL  DATERR  PARERR  FRMERR  OVRERR'<15><12>
 6306  025646  021440  020040  052122
 6307  025654  052117  046101  020040
 6308  025662  052130  052117  046101
 6309  025670  020040  040504  042524
 6310  025676  051122  020040  040520
 6311  025704  042522  051122  020040
 6312  025712  051106  042515  051122
 6313  025720  020040  053117  042522
 6314  025726  051122  005015     000
 6315                          ;MESSAGES USED BY THE AUTOSIZER.
 6316  025733     116  020117  044104  MSG1:   .ASCIZ  /NO DH11'S WERE FOUND/<15><12>
 6317  025740  030461  051447  053440
```

```
 6318  025746  051105  020105  047506
 6319  025754  047125  006504  000012
 6320  025762  047516  042040  020110  MSG2:   .ASCIZ  /NO DH RECEIVER INTERRUPT OCCURRED/<15><12>
 6321  025770  042522  042503  053111
 6322  025776  051105  044440  052116
 6323  026004  051105  052522  052120
 6324  026012  047440  041503  051125
 6325  026020  042522  006504  000012
 6326  026026  047516  042040  030515  MSG3:   .ASCIZ  /NO DM11-BB INTERRUPT OCCURRED./<15><12>
 6327  026034  026461  041102  044440
 6328  026042  052116  051105  052522
 6329  026050  052120  047440  041503
 6330  026056  051125  042522  027104
 6331  026064  005015      000
 6332  026067      040  020040      000  SPACE:  .ASCIZ  /   /
 6333  026073      015  042012  030510  DEVMAP: .ASCII  <15><12>/DH11, DM11-BB DEVICE MAP:/<15><12>
 6334  026100  026061  042040  030515
 6335  026106  026461  041102  042040
 6336  026114  053105  041511  020105
 6337  026122  040515  035120  005015
 6338  026130  005015  044104  030461          .ASCII  <15><12>/DH11   DH11   DM11-BB   DM11-BB/
 6339  026136  020040  042040  030510
 6340  026144  020061  020040  046504
 6341  026152  030461  041055  020102
 6342  026160  020040  046504  030461
 6343  026166  041055      102
 6344  026171      015  040412  051104          .ASCIZ  <15><12>/ADRS   VECT    ADRS      VECT/<15><12><15><12>
 6345  026176  020123  020040  042526
 6346  026204  052103  020040  020040
 6347  026212  042101  051522  020040
 6348  026220  020040  020040  042526
 6349  026226  052103  005015  005015
 6350  026234      000
 6351                                  ;MESSAGES FOR INPUTTING PARAMETERS TO ECHO TESTS
 6352
 6353  026235      040  020040  005015  EC:     .ASCIZ  '   '<15><12>
 6354  026242      000
 6355  026243      040      000          EC2:    .ASCIZ  ' '
 6356  026245      040  006440  000012  EC3:    .ASCIZ  '  '<15><12>
 6357  026252  005015  044523  043516  ECMSG1: .ASCIZ  <15><12>'SINGLE LINE ECHO TEST - CONNECT TERMINAL TO DH11 TEST LINE'<15>
 6358  026260  042514  046040  047111
 6359  026266  020105  041505  047510
 6360  026274  052040  051505  020124
 6361  026302  020055  047503  047116
 6362  026310  041505  020124  042524
 6363  026316  046522  047111  046101
 6364  026324  052040  020117  044104
 6365  026332  030461  052040  051505
 6366  026340  020124  044514  042516
 6367  026346  005015      000
 6368
 6369
 6370  026351      015  052012  050131  ECMSG2: .ASCIZ  <15><12>'TYPE LINE # (00 - 17 OCTAL)'
 6371  026356  020105  044514  042516
 6372  026364  021440  024040  030060
 6373  026372  026440  030440  020067
```

```
 6374  026400  041460  040524  024514
 6375  026406     000
 6376
 6377  026407     015  052012  051505  ECMSG3: .ASCIZ  <15><12>'TESTING LINE #   - GO TYPE IN ON TEST LINE'<15><12>
 6378  026414  044524  043516  046040
 6379  026422  047111  020105  020043
 6380  026430  020040  020055  047507
 6381  026436  052040  050131  020105
 6382  026444  047111  047440  020116
 6383  026452  042524  052123  046040
 6384  026460  047111  006505  000012
 6385
 6386  026466  005015  054524  042520  ECMSG4: .ASCIZ  <15><12>'TYPE: [CONTROL-C TO EXIT] [CONTROL-E TO ECHO BUFFER]'<15><12>
 6387  026474  020072  041533  047117
 6388  026502  051124  046117  041455
 6389  026510  052040  020117  054105
 6390  026516  052111  020135  041533
 6391  026524  047117  051124  046117
 6392  026532  042455  052040  020117
 6393  026540  041505  047510  041040
 6394  026546  043125  042506  056522
 6395  026554  005015     000
 6396  026557     015  042012  020117  LPMSG:  .ASCIZ  <15><12>'DO YOU WANT TO CHANGE "LPR" (Y OR N) ? '
 6397  026564  047531  020125  040527
 6398  026572  052116  052040  020117
 6399  026600  044103  047101  042507
 6400  026606  021040  050114  021122
 6401  026614  024040  020131  051117
 6402  026622  047040  020051  020077
 6403  026630     000
 6404  026631     015  052012  040522  XSMSG1: .ASCIZ  <15><12>'TRANSMITTER SPEED ?'
 6405  026636  051516  044515  052124
 6406  026644  051105  051440  042520
 6407  026652  042105  037440     000
 6408
 6409  026657     015  044412  053116  XSMSG2: .ASCIZ  <15><12>'INVALID XMIT SPEED - TRY AGAIN'<15><12>
 6410  026664  046101  042111  054040
 6411  026672  044515  020124  050123
 6412  026700  042505  020104  020055
 6413  026706  051124  020131  043501
 6414  026714  044501  006516  000012
 6415
 6416  026722  005015  042522  042503  RSMSG1: .ASCIZ  <15><12>'RECEIVER SPEED ?'
 6417  026730  053111  051105  051440
 6418  026736  042520  042105  037440
 6419  026744     000
 6420
 6421  026745     015  044412  053116  RSMSG2: .ASCIZ  <15><12>'INVALID RCVR SPEED - TRY AGAIN'<15><12>
 6422  026752  046101  042111  051040
 6423  026760  053103  020122  050123
 6424  026766  042505  020104  020055
 6425  026774  051124  020131  043501
 6426  027002  044501  006516  000012
 6427
 6428  027010  005015  044103  051101  CLMSG1: .ASCIZ  <15><12>'CHAR LENGTH (6, 7, OR 8) ? '
 6429  027016  046040  047105  052107
```

```
 6430  027024  020110  033050  020054
 6431  027032  026067  047440  020122
 6432  027040  024470  037440  000040
 6433
 6434  027046  005015  047111  040526  CLMSG2: .ASCIZ  <15><12>'INVALID CHAR LENGTH - TRY AGAIN'<15><12>
 6435  027054  044514  020104  044103
 6436  027062  051101  046040  047105
 6437  027070  052107  020110  020055
 6438  027076  051124  020131  043501
 6439  027104  044501  006516  000012
 6440
 6441  027112  005015  047516  020056  SBMSG1: .ASCIZ  <15><12>'NO. OF STOP BITS (1 OR 2) ? '
 6442  027120  043117  051440  047524
 6443  027126  020120  044502  051524
 6444  027134  024040  020061  051117
 6445  027142  031040  020051  020077
 6446  027150     000
 6447
 6448  027151     015  044412  053116  SBMSG2: .ASCIZ  <15><12>'INVALID NO. STOP BITS - TRY AGAIN'<15><12>
 6449  027156  046101  042111  047040
 6450  027164  027117  051440  047524
 6451  027172  020120  044502  051524
 6452  027200  026440  052040  054522
 6453  027206  040440  040507  047111
 6454  027214  005015     000
 6455
 6456  027217     015  050012  051101  PBMSG1: .ASCIZ  <15><12>'PARITY SELECTION (E, O, OR <CR>) ? '
 6457  027224  052111  020131  042523
 6458  027232  042514  052103  047511
 6459  027240  020116  042450  020054
 6460  027246  026117  047440  020122
 6461  027254  041474  037122  020051
 6462  027262  020077     000
 6463
 6464  027265     015  044412  053116  PBMSG2: .ASCIZ  <15><12>'INVALID PARITY - TRY AGAIN'<15><12>
 6465  027272  046101  042111  050040
 6466  027300  051101  052111  020131
 6467  027306  020055  051124  020131
 6468  027314  043501  044501  006516
 6469  027322  000012
 6470
 6471  027324  005015  044506  046114  FILC1:  .ASCIZ  <15><12>'FILLER CHARACTER ? '
 6472  027332  051105  041440  040510
 6473  027340  040522  052103  051105
 6474  027346  037440  000040
 6475
 6476  027352  005015  044506  046114  FILC2:  .ASCIZ  <15><12>'FILLER COUNT ?'
 6477  027360  051105  041440  052517
 6478  027366  052116  037440     000
 6479  027373     015  051412  047105  SNMSG1: .ASCIZ  <15><12>'SEND MODE -(Y OR N)  '
 6480  027400  020104  047515  042504
 6481  027406  026440  054450  047440
 6482  027414  020122  024516  020040
 6483  027422     000
 6484  027423     015  052012  050131  SNMSG2: .ASCIZ  <15><12>'TYPE SEND BUFFER - TERMINATE WITH CONTROL-C'<15><12>
 6485  027430  020105  042523  042116
```

```
 6486  027436  041040  043125  042506
 6487  027444  020122  020055  042524
 6488  027452  046522  047111  052101
 6489  027460  020105  044527  044124
 6490  027466  041440  047117  051124
 6491  027474  046117  041455  005015
 6492  027502     000
 6493  027503     015  041412  040510  SNMSG3: .ASCIZ  <15><12>'CHANGE PARAMETERS (Y OR N)? '
 6494  027510  043516  020105  040520
 6495  027516  040522  042515  042524
 6496  027524  051522  024040  020131
 6497  027532  051117  047040  037451
 6498  027540  000040
 6499
```

```
                                        ;MESSAGES FO INPUTIING PAIIERNS IESI PARAMEIERS
027542  005015  040504  040524  DPMSG1: .ASCIZ  <15><12>'DATA PATTERNS TESTS - CONNECT TEST JUMPER'<15><12>
027550  050040  052101  042524
027556  047122  020123  042524
027564  052123  020123  020055
027572  047503  047116  041505
027600  020124  042524  052123
027606  045040  046525  042520
027614  006522  000012
027620  005015  052502  043106  DPMSG2: .ASCIZ  <15><12>'BUFFER SIZE ? (1 - 512)'
027626  051105  051440  055111
027634  020105  020077  030450
027642  026440  032440  031061
027650  000051
027652  005015  047111  040526  DPMSG3: .ASCIZ  <15><12>'INVALID SIZE - TRY AGAIN'<15><12>
027660  044514  020104  044523
027666  042532  026440  052040
027674  054522  040440  040507
027702  047111  005015      000
027707      015  050012  052101  DPMSG4: .ASCIZ  <15><12>'PATTERN TYPE ? (A,U,D,R,S,B OR <CR>) '<15><12>
027714  042524  047122  052040
027722  050131  020105  020077
027730  040450  052454  042054
027736  051054  051454  041054
027744  047440  020122  041474
027752  037122  020051  005015
027760      000
027761      015  051412  052105  DPMSG5: .ASCIZ  <15><12>'SET SR07=1 TO LOCK ON PATTERN'<15><12>
027766  051440  030122  036467
027774  020061  047524  046040
030002  041517  020113  047117
030010  050040  052101  042524
030016  047122  005015      000
030023      015  044412  053116  DPMSG6: .ASCIZ  <15><12>'INVALID PATTERN - TRY AGAIN'<15><12>
030030  046101  042111  050040
030036  052101  042524  047122
030044  026440  052040  054522
030052  040440  040507  047111
030060  005015      000
030063      015  052012  050131  DPMSG7: .ASCIZ  <15><12>'TYPE SINGLE TEST CHAR '<15><12>
030070  020105  044523  043516
030076  042514  052040  051505
030104  020124  044103  051101
030112  006440  000012
030116  005015  054524  042520  DPMSGA: .ASCIZ  <15><12>'TYPE IN TEST BUFFER - TERMINATE WITH CONTROL-C'<15><12>
030124  044440  020116  042524
030132  052123  041040  043125
030140  042506  020122  020055
030146  042524  046522  047111
030154  052101  020105  044527
030162  044124  041440  047117
030170  051124  046117  041455
030176  005015      000
030202                          .EVEN
                                ;ERROR STATISTICS TABLES
```

```
6556  030202  000020                  RTOTAL: .BLKW   16.             ;TOTAL CHARS RCVD PER LINE
6557  030242  000020                  XTOTAL: .BLKW   16.             ;TOTAL CHARS XMITTED PER LINE
6558  030302  000020                  DATERR: .BLKW   16.             ;TOTAL DATA COMPARE ERRORS PER LINE
6559  030342  000020                  PARERR: .BLKW   16.             ;TOTAL PARITY ERRORS PER LINE
6560  030402  000020                  OVRERR: .BLKW   16.             ;TOTAL OVERRUN ERRORS PER LINE
6561  030442  000020                  FRMERR: .BLKW   16.             ;TOTAL FRAMING ERRORS PER LINE
6562
6563                                  ;STATISTICS TABLES POINTERS
6564
6565  030502  000000                  DEPTR:  0                       ;CONTAINS POINTERS TO STAT TABLES
6566  030504  000000                  PEPTR:  0
6567  030506  000000                  ORPTR:  0
6568  030510  000000                  FRPTR:  0
6569
6570  030512  000000                  RBFPTR: 0                       ;CONTAINS INPUT BUFFER POINTER
6571  030514  000000                  TBFPTR: 0                       ;CONTAINS OUTPUT BUFFER POINTER
6572
6573
6574                                  ;600. WORD RECEIVER INPUT BUFFER
6575
6576  030516  001130                  RBUF:   .BLKW   600.
6577
6578
6579                                  ;600(10) BYTE TRANSMITTER OUTPUT DATA BUFFER
6580
6581                                  .EVEN
6582  032776  001130                  TBUF:   .BLKB   600.
6583
6584  034126  000000                  ENBUFS: 0                       ;MARK END OF BUFFERS
6585
6586          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    2376
ADDW11= 000000          2321    2377
ADDW12= 000000          2321    2378
ADDW13= 000000          2321    2379
ADDW14= 000000          2321    2380
ADDW15= 000000          2321    2381
ADDW2 = 000000          2321    2368
ADDW3 = 000000          2321    2369
ADDW4 = 000000          2321    2370
ADDW5 = 000000          2321    2371
ADDW6 = 000000          2321    2372
ADDW7 = 000000          2321    2373
ADDW8 = 000000          2321    2374
ADDW9 = 000000          2321    2375
ADEVCT= 000000          2321    2327
ADEVM = 000000          2321    2363
ADPTR   022330          2643*   2650*   2658    3042*   5892#
ADRVEC  022142          2626    4880*   5820#
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          4130    4235#
APTENV= 000001          3881    4123    4191    4233#
APTSIZ= 000200          2595    4232#
APTSPD= 000100          4125    4193    4234#
ASWREG= 000000          2321    2334
ATESTN= 000000          2321    2325
AUNIT = 000000          2321    2328
AUSWR = 000000          2321    2335
AUTOSZ  015152          2621    4719#
AVECT1= 000000          2321    2360
AVECT2= 000000          2321    2361
BAR   = 000012          2710*   2781    2785    2789    3165*   3215*   3524*   3589    3593    3597    5272*   5633#
```

N11


```
BCR    = 000010  2800  2804  2808  3161* 3189* 3200* 3211* 3608  3612  3616  4643* 5269* 5291*
                 5632#
BEGIN    001626  2143  2552#
BEGINA   001644  2556#  3060  3275  4935  4940
BIT0   = 000001  2242#  2619  2624  2849  4943  4954  4973  4982  5016  5054
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  3121  3174  3181
BIT07  = 000200  2225#  2235  2878  3675
BIT08  = 000400  2224#  2234
BIT09  = 001000  2223#  2233  3828  3892
BIT1   = 000002  2241#  2920  4881
BIT10  = 002000  2222#  2743  3551
BIT11  = 004000  2221#  2722  2737  2748  2766  2786  2805  2858  2877  2895  3112  3141  3534
                 3543  3556  3574  3594  3613  3655  3674  3692  3835  4631  5265
BIT12  = 010000  2220#
BIT13  = 020000  2219#  3149  3164  3214  3876
BIT14  = 040000  2218#  2853  3650  3810
BIT15  = 100000  2217#  2767  2946  3149  3575  3725
BIT2   = 000004  2240#
BIT3   = 000010  2239#
BIT4   = 000020  2238#
BIT5   = 000040  2237#
BIT6   = 000100  2236#
BIT7   = 000200  2235#
BIT8   = 000400  2234#  4990
BIT9   = 001000  2233#
BKR    = 000014  5634#
BPTVEC= 000014  2249#
BRLVL    021716  2645  5729#
BRPTR    022334  2645*  2652*  2660  3044*  5894#
BUSER    016762  2610  5076#
CAR    = 000006  2819  2828  3162* 3212* 3627  3636  4644* 5270* 5631#
CEXIT    022344  3108*  3134  3168* 5900#
CHKAOR   016454  4950  4998#
CHKVCT   016620  4961  5036#
CHPS1    021426  2711  3130  3525  5083  5093  5257  5596#
CHPS2    021442  2747  2765  2784  2803  2826  2856  2874  2894  3107  3555  3573  3592  3611
                 3634  3653  3671  3691  4632  5604#
CHRCNT   021602  2698*  2820  2842  2915  3231  3299* 3509* 3628  4639  4643  5171* 5174* 5191
                 5517  5527  5539  5551  5568  5674#
CKER     004112  2738  2928#
CKERDP   010602  3544  3717#
CKRST1   014574  4597  4612#
CKRST2   014624  4620#  5084  5094
CKSWR  = 104407  3308  3808  3868  3891  4555#
CLALL    021410  5516  5526  5538  5550  5567  5588#
CLMSG1   027010  5391  6428#
CLMSG2   027046  5415  6434#
CLMSK    022370  3726  5578*  5582*  5910#
CLSEL    021604  2699*  5164*  5165  5170*  5173  5676#
CLSTAT   017136  2672  5125#
```

```
CR     = 000015  2157#  4169   4179
CRLF   = 000200  2158#  4140   4179
CURLPR  021576   3122   4642   5140   5142*  5145   5172*  5173*  5215*  5216*  5271   5299*  5306*  5318*
                 5328*  5335*  5347*  5374   5376*  5392*  5406*  5410*  5417*  5438*  5445*  5453*  5466*
                 5470*  5577   5671#  5942   6036   6063   6117   6138   6201
DATCNT  022350   3359*  3362*  3363   3366   3367*  3377*  3380*  3381   3384   3385*  3395*  3398*  3399
                 3402   3403*  3413*  3416*  3417   3420   3421*  3444*  3462*  3463   3466   3467*  3474*
                 3511*  3512   3515   3516*  3535   5902#
DATERR  030302   2930   3027   6558#
DATPAT  022352   3318*  3331   3335   3339   3343   3347   3351   3431*  3433*  3435*  3437*  3536   5903#
DDISP  = 177570  2164#  2285   2583
DEPTR   030502   2930*  2936*  2980*  6565#
DEVMAP  026073   4884   6333#
DF1     023602   2477   6071#
DF2     022604   2407   2414   2421   2428   2435   2442   2449   2456   2463   2470   2484   2491   2498
                 2505   2512   2519   2526   2533   2540   2547   5945#
DHADR   021522   2658*  2683   2722*  3062*  3111   3277*  3311   3534*  4671*  5009*  5013*  5264   5637#
                 6138   6201
DHADRS  021756   4721   4729   4776   4792   4885   4942   5018   5748#
DHADTB  021614   2643   5015   5029   5685#
DHFILL  022434   5483*  5495*  5496   5507*  5509*  5511*  5512   5913#
DHNUM   022150   2646*  2649*  2662   3001   3041*  5824#
DHRLVL  022146   2660*  2679   3110*  3115   3301*  3304   5822#
DHSEL   021530   2653   3048   4980*  5640#
DHSET1  014654   2709   3523   4631#
DHST1   002700   2709#
DHST2   007526   3361   3379   3397   3415   3461   3510   3523#
DHTLVL  022147   2682   3118   3307   5823#
DHVCT   021524   2659*  2677   3063*  3113   3278*  3302   5047*  5051*  5099   5638#
DHVCTB  021654   2644   5053   5067   5705#
DHVEC   022014   4793   4874   4875   4877   4886   4953   5056   5767#
DHWAIT  005252   3132#
DH1     022511   2405   2412   2419   2426   2433   2440   2447   5933#
DH2     023305   2454   2461   6026#
DH3     023500   2468   6055#
DH4     023637   2482   2489   6081#
DH5     024022   2496   6108#
DH6     024162   2503   2510   2517   2524   2531   2538   6130#
DH7     024542   2545   6192#
DISPLA  001142   2285#  2583*  2591*  3850*  3871*
DISPRE  000174   2138#  2591
DMADRS  022050   4728   4834   4887   5785#
DMVEC   022106   4835   4888   5804#
DPATA   006474   3333   3359#  3432
DPATB   007306   3353   3474#
DPATCR  007064   3329   3431#  3440
DPATD   006670   3341   3395#  3436
DPATR   006766   3345   3413#  3438
DPATS   007146   3349   3443#
DPATU   006572   3337   3377#  3434
DPFLG   022346   2554*  2633   3058*  3084   3272*  4612   4620   4930*  4938*  4967   5011   5049   5901#
DPMSGA  030116   3476   6544#
DPMSG1  027542   3280   6501#
DPMSG2  027620   3286   6509#
DPMSG3  027652   3294   6514#
DPMSG4  027707   3313   6519#
```

```
DPMSG5  027761          3325    6527#
DPMSG6  030023          3356    6533#
DPMSG7  030063          3446    6539#
DRPLIN  021536          2675*   2732*   2759*   2778*   2797*   2816*   2836*   2868*   2888*   2905*   3008    5643#
DSWR  = 177570          2163#   2284    2582
DT1     022566          2406    2413    2420    2427    2434    2441    2448    5942#
DT10    024622          2546    6201#
DT2     023372          2455    2462    6036#
DT3     023546          2469    6063#
DT4     023642          2476    6068#
DT5     023706          2483    2490    6089#
DT6     024102          2497    6117#
DT7     024232          2504    2511    2518    2525    2532    2539    6138#
EC      026235          6353#
ECBUF   022374          3187*   3188*   3196*   3197*   3209*   3212    5281    5282*   5912#
ECHO    004706          2145    3057#   4618    4626
ECHO1   004730          2632    3062#
ECHO2   004754          3070#   3144    3265    3283    4966
ECHO3   005124          3100    3102    3107#
ECMSG1  026252          3065    6357#
ECMSG2  026351          3071    6370#
ECMSG3  026407          3125    3127    6377#
ECMSG4  026466          3129    6386#
EC2     026243          3088*   3095*   3097    3099    3101    3103    3225*   3227    3247*   3254*   3256    3259    3261
                        3263    3310*   3319*   3321    3326    3443*   3452*   3454    3457    3480*   3482    5355*   5362*
                        5364    5367    5369    5371    5389*   5397*   5399    5401    5403*   5404    5408    5412    5422*
                        5429*   5431    5433    5435*   5436    5440    5450*   5458*   5460    5462    5464    5468    5479*
                        5488*   5490    5493    5495    6355#
EC3     026245          3503*   3504*   3506    6356#
EMTVEC= 000030          2252#   2567*   2568*
EM1     022442          2404    2755    3563    5926#
EM10    023235          2453    2901    3698    6019#
EM11    023412          2460    2961    3740    6042#
EM12    023440          2467    2729    6049#
EM14    023612          2481    6077#
EM15    023722          2488    6094#
EM16    023750          2495    6101#
EM17    024120          2502    6124#
EM2     022614          2411    2774    3582    5951#
EM20    024246          2509    6143#
EM21    024310          2516    6152#
EM22    024354          2523    6161#
EM23    024412          2530    6169#
EM24    024450          2537    6177#
EM25    024507          2544    6187#
EM3     022673          2418    2793    3601    5963#
EM4     022753          2425    2812    3620    5975#
EM5     023030          2432    2832    3640    5986#
EM6     023112          2439    2864    3661    5998#
EM7     023161          2446    2884    3681    6008#
ENBUFS  034126          5590    6584#
ENDA    004616          3035    3038#   3049
ERRVEC= 000004          2245#   2580    2581*   2592*   2610*   2611*   3815    3816*   3818*   3821*
EXFLAG  022372          3109*   3150    3152    3154    3158*   3172*   3190*   3201*   5911#   6117
EXPAR   016450          4991    4995#
EXPAT   006106          2146    3272#   4617    4625
```

```
EXPAT1  006130          2635    3277#
EXPAT2  006154          3283#   4969
EXPAT3  006260          3308#   3357    3375    3393    3411    3429    3441    3472    3521    3541    3568    3587    3606
                        3625    3645    3666    3686    3703
FILC1   027324          5482    6471#
FILC2   027352          5503    6476#
FILLA   022436          3238    3495    5284    5285    5480*   5496*   5915#
FILLB   022440          3237    3493    5280    5288    5501*   5512*   5916#
FRMERR  030442          2933    3030    6561#
FRPTR   030510          2933*   2939*   2988*   6568#
GNS     = ******  U     2137    4547    4548    4549    4550    4551    4553    4555    4556    4557    4558    4559
GTSWR   = 104406        2606    4553#
HT      = 000011        2155#   4138    4179
INCL    020372          5380    5389#   5416
INFCHR  020774          5383    5479#
INFCNT  021064          5384    5501#
INMSGA  025376          4985    6274#
INMSG1  024747          4946    6224#
INMSG2  025013          4957    6231#
INMSG3  025062          4976    6238#
INMSG4  025133          5033    6245#
INMSG5  025204          5071    6252#
INMSG6  025260          6260#
INMSG7  025326          4993    6267#
INPAR   016224          2641    3067    3281    4942#
INPARA  016120          2628    4919#   4928
INPARC  016200          2144    4936#
INPARX  016166          2141    4934#
INPAR1  016260          4949    4953#
INPAR3  016314          2640    4960    4964#
INPAR4  016376          4979    4981#
INPB    020654          5382    5450#   5474
INRSP   020116          5325#   5379
INSB    020536          5381    5422#   5444
INXSP   020002          5296#   5378
IOTVEC= 000020          2250#   2565*   2566*   4794*   4795*   4836*   4862*
KYBD1   017522          2673    5233#
KYBD2   017546          5233    5241#
LDFILL  017726          3206    5280#
LF      = 000012        2156#   4173    4179
LINE    022152          2676*   2690    2723    2728    2749    2754    2768    2773    2792    2811    2831    2844    2860
                        2863    2879    2883    2900    2917    2934    2944    2960    3073*   3074*   3075    3119    3124
                        3557    3562    3576    3581    3600    3619    3639    3657    3660    3676    3680    3697    3723
                        3739    3809*   4635    4657    4659*   4661*   4663*   4670*   5266    5825#   6117    6138    6201
LINMSK  021534          2710    2732    2759    2778    2797    2816    2836    2868    2888    2905    2997*   3008    3021
                        3034*   3077*   3080*   3165    3215    3524    4660*   4664*   4666    5272    5642#
LINSEL  021532          3021    4666    4981*   4989*   5641#
LPMSG   026557          5357    6396#
LPR     = 000004        3122*   4642*   5271*   5630#
LPRIN   020232          3082    5355#   5372
LPRPTR  021600          2694*   5140    5142    5150*   5673#
LPRTAB  021544          2694    5657#
MSG1    025733          4778    6316#
MSG2    025762          4818    6320#
MSG3    026026          4847    6326#
NEWCL   002660          2702#   2707
```

```
NEWLIN  002574          2685#   2697    2734    2760    2779    2798    2817    2837    2869    2889    2906
NEWLPR  002632          2696#   2703
NEWPAR  002672          2706#   2739
NRC     = 000002        2891    3182    3688    5629#
ORPTR   030506          2932*   2938*   2985*   6567#
OVRERR  030402          2932    3029    6560#
PARBIT  021606          2704*   5210    5214*   5216    5217    5219*   5221    5223*   5225    5227*   5677#
PARERR  030342          2931    3028    6559#
PATFLG  022354          3309*   3328*   3370    3388    3406    3424    5904#
PATLIM  022362          3363    3381    3399    3417    3463    3512    5907#
PBMSG1  027217          5452    6456#
PBMSG2  027265          5473    6464#
PEPTR   030504          2931*   2937*   2991*   6566#
PIRQ    = 177772        2162#
PIRQVE= 000240          2256#
PRSTAT  004426          2688    2997#   5258
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    4810*   4840*
PSW     = 177776        2160#
PTWAIT  007544          3527#
PWRVEC= 000024          2251#   2571*   2572*   4564*   4565*   4574*   4580*   4592*   4593*
QUICK   021540          2674*   5147*   5149*   5167    5212    5645#
QUICKX  021542          2701*   5162    5169*   5647#
RAMA    022364          5553*   5555    5556*   5557*   5560    5908#
RAMB    022366          5555*   5557    5558*   5909#
RBFEND  021612          2913    2973    3707    3710    3748    3751    4638*   4641*   5680#
RBFPTR  030512          2896*   2897    2911*   2912*   2913    2928*   2947    2953    2972*   2973    3693*   3694    3705*
                        3706*   3707    3709*   3710    3717*   3727    3733    3747*   3748    3750*   3751    4637*   6570#
RBUF    030516          2896    2928    3693    3717    4637    4638    6576#
RDCHR = 104410          3091    3223    3250    3314    3447    3478    4350    4556#   5358    5393    5425    5454    5484
RDDEC = 104413          3287    4559#   5300    5329
RDLIN = 104411          4422    4475    4557#
RDOCT = 104412          3072    4558#   4921    4947    4958    4977    4986    5504
RDONE   021610          2716    2849*   2920*   3529    3712*   4636*   5679#   6063
RESERR  017024          2612    5086#
RESTRP  017066          5099#
RESTRT  002374          2649#   2657
REST1   002432          2655#   4624
RESVEC= 000010          2246#   2612*   2613*
RETFLG  022360          2555*   2630    3057*   3273*   4614    4622    4931*   4939*   4964    5007    5045    5906#
RINT1   003550          2678    2853#
RINT2   005476          3114    3181#
RINT3   010266          3303    3650#
RSMSG1  026722          5326    6416#
RSMSG2  026745          5343    6421#
RSPTAB  022244          5327    5340    5863#
RSPTR   022242          5327*   5332    5334*   5335    5339*   5340    5861#
RSTRTA  002442          2654    2658#   3050    4616
RTOTAL  030202          2725    2919*   3025    5125    5127    6556#
```

```
SAPS    021456  5612#
SBMSG1  027112  5424    6441#
SBMSG2  027151  5443    6448#
SELINE  014752  2685    4657#
SELMSK  021526  2647#   2653    2655#   3045#   3048    5639#
SENDP1  005720  3105    3220#   3260    3262
SENDP2  017650  3246    5264#
SETCL   017240  2702    5162#
SETLPR  017154  2696    5140#
SETPAR  017400  2706    5210#
SINGLE  022356  3451#   3459#   3479#   3480    3484    3490    3494    3500    5570    5905#
SNMSG1  027373  3090    6479#
SNMSG2  027423  3221    6484#
SNMSG3  027503  3249    6493#
SOFT    004366  2969    2983#
SPACE   026067  4899    4904    6332#
SSR     =000016 5635#
STACK   =001100 2150#   2563    2733    5079    5089    5256
START1  002152  2610#
START2  002336  2637    2643#   2656    3789    4995
STDH1   002516  2672#
STKLMT= 177774  2161#
STMSG1  025535  3002    3004    6292#
STMSG2  025562  2691    2693    6296#
STMSG3  025607  3014    3016    6300#
STMSG4  025640  3006    6305#
SUBUF1  017340  5175    5186#
SUCLMK  021354  3284    5577#
SUER1   015130  2752    2771    2790    2809    2829    2861    2881    3560    3579    3598    3617    3637    3658
                3678    4710#
SUER2   015124  2898    2958    3695    3737    4709#
SUNUM   015044  2661    2689    2727    2753    2772    2791    2810    2830    2862    2882    2899    2959    2999
                3012    3123    3561    3580    3599    3618    3638    3659    3679    3696    3738    4684#
SUPATA  021136  3360    5516#
SUPATD  021214  3396    5538#
SUPATR  021246  3414    5550#
SUPATS  021326  3460    5567#
SUPATU  021164  3378    5526#
SWR     001140  2284#   2561    2582#   2584    2590#   2597#   2604    2619    2624    3373    3391    3409    3427
                3439    3470    3519    3810    3828    3835    3876    3888    3892    4246    4283#   4572    4585#
                4881    4943    4954    4973    4982    4990    5016    5054    5143    5245
SWREG   000176  2139#   2590    2604    4246    4259    5245
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  030514          2929*   2941    2954    2971*   3718*   3720    3734    3746*   6571#
TBITVE= 000014          2247#
TBUF    032776          2822    2929    2956    3120    3156    3159    3162    3171    3222    3477    3487    3507    3630
                        3718    3736    4644    5192    5267    5270    5518    5528    5540    5552    5569    5588    6582#
TIMEA   022340          2714*   3132*   3163*   3173*   3213*   3527*   5116*   5897#
TIMEB   022342          2715*   3133*   3528*   5114*   5898#
TIMEIT  017114          2718    3136    3531    5114#
TINT1   003044          2681    2743#
TINT2   005330          3117    3149#
TINT3   007644          3306    3551#
TITFLG  022336          2556*   2614    2618*   5896#
TITLE   024640          2617    6211#
TITLE2  024722          2663    2665    6220#
TKVEC =  000060         2254#
TPVEC =  000064         2255#
TRAPVE=  000034         2253#   2569*   2570*
TRTVEC=  000014         2248#
TST1    002506          2670#
TYPDS =  104405         3778    3944    4551#
TYPE  =  104401         2616    2664    2692    3003    3005    3015    3064    3070    3089    3096    3126    3128    3220
                        3226    3248    3255    3279    3285    3293    3312    3320    3324    3355    3445    3453    3475
                        3481    3505    3776    3779    3879    3908    3925    3927    3930    3932    3947    3952    4016
                        4091    4143    4257    4258    4261    4274    4285    4304    4357    4363    4368    4372    4377
                        4378    4380    4383    4387    4451    4453    4512    4514    4547#   4594    4778    4818    4847
                        4883    4899    4904    4909    4913    4919    4945    4956    4975    4984    4992    5032    5070
                        5252    5296    5313    5325    5342    5356    5363    5390    5398    5414    5423    5430    5442
                        5451    5459    5472    5481    5489    5502
TYPOC =  104402         3916    3940    4260    4548#
TYPON =  104404         4550#
TYPOS =  104403         4549#   4892    4896    4901    4906
UPDER   004354          2957    2978#
VCFLG   021714          2553*   2622    2629*   3059*   3274*   4934*   4937*   5722#
VCPTR   022332          2644*   2651*   2659    3043*   5893#
VCWC    025440          4920    6281#
XSMSG1  026631          5297    6404#
XSMSG2  026657          5314    6409#
XSPTAB  022156          5298    5311    5835#
XSPTR   022154          5298*   5303    5305*   5306    5310*   5311    5833#
XTOTAL  037242          2726    2846*   3026    6557#
$APTHD  000000          2118    2124#
$ASTAT= ******  U       4213    4228
$ATYC   012664          4184    4186#
$ATY1   012640          4182#
$ATY3   012646          4128    4183#
```

```
$ATY4   012656          3884    4185#
$AUTOB  001134          2281#   2608*   4254    4403    5247
$BASE   001306          2362#
$BDADR  001122          2276#
$BDDAT  001126          2278#
$CDW1   001312          2364#
$CDW2   001314          2365#
$CHARC  012634          4145*   4155*   4162    4171*   4176#
$CKSWR  013106          4246#   4555
$CMTAG  001100          2264#   2558    2559    2567    2573    2574
$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  013761          4257    4398#   5252
$CNTLU  013754          4274    4372    4397#
$CPUOP  001260          2336#
$CRLF   001227          2315#   3879    3900    3908    3927    3932    3952    4144    4179    4285    4377    4397    4456
                        4516    4910    4913
$DBLK   012346          4057    4091    4099#
$DDW0   001316          2366#
$DDW1   001320          2367#
$DDW10  001342          2376#
$DDW11  001344          2377#
$DDW12  001346          2378#
$DDW13  001350          2379#
$DDW14  001352          2380#
$DDW15  001354          2381#
$DDW2   001322          2368#
$DDW3   001324          2369#
$DDW4   001326          2370#
$DDW5   001330          2371#
$DDW6   001332          2372#
$DDW7   001334          2373#
$DDW8   001336          2374#
$DDW9   001340          2375#
$DEVCT  001242          2327#
$DEVM   001310          2363#
$DHSEL  022144          4788*   4970    5020    5058    5821#
$DORGN  011066          3772    3781    3787#
$DTBL   012336          4060    4095#
$ENDAD  011056          3783#
$ENDCT  011024          3774#
$ENDMG  011075          3776    3791#
$ENULL  011072          3779    3790#
$ENV    001252          2332#   2602    3881    4123    4191    4215
$ENVM   001253          2333#   2595    4125    4130    4193
$EOP    010770          3040*   3047    3764#
$EOPCT  011016          3771#   3775
$ERFLG  001103          2267#   3799    3824    3826    3832*   3854    3869*   3900
$ERMAX  001115          2273#   2575*   3826    3849*   3854
$ERROR  011356          2567    3867#
$ERRPC  001116          2274#   3873*   3874*   3875    3900    3914    5942    6036    6063    6089    6117    6138    6201
$ERRTB  001356          2400#   3922
$ERRTY  011530          3878    3907#
$ERTTL  001112          2271#   3872*   3900
```

```
$ESCAP  001224          2313#   2574*   3848*   3895    3897    3900
$ETABL  001252          2331#
$ETEND  001356          2130    2384#
$FATAL  001234          2324#   4219*
$FFLG   013104          4182*   4185*   4213    4222*   4230#
$FILLC  001156          2292#   4148    4179
$FILLS  001155          2291#   4179
$GDAOR  001120          2275#
$GDDAT  001124          2277#
$GET42  011046          3780#
$GTSWR  013156          4258#   4553    5253
$HD   = 000001          2089    2090
$HIBTS  000000          2125#
$HIOCT  014146          4444*   4455#
$ICNT   001104          2268#   3839*   3840    3842*   3853
$ILLUP  014556          4564    4580    4599#
$INTAG  001135          2282#   4286    4403
$ITEMB  001114          2272#   3875*   3883    3900    3911
$LF     001230          2316#   3900    4179    4387    4397    4456    4516
$LFLG   013103          4223*   4229#
$LPADR  001106          2269#   2576*   2666*   3830*   3846*   3851    3853
$LPERR  001110          2270#   2577*   2730*   2756*   2775*   2794*   2813*   2833*   2865*   2885*   2902*   2962*   3031*
                        3142*   3368*   3386*   3404*   3422*   3468*   3517*   3537*   3564*   3583*   3602*   3621*   3641*
                        3662*   3682*   3699*   3741*   3830    3847*   3853    3894    5080*   5090*
$MADR1  001264          2349#
$MADR2  001270          2353#
$MADR3  001274          2356#
$MADR4  001300          2359#
$MAIL   001232          2126    2130    2322#   2594    2602    3845    3881    4123
$MAMS1  001262          2343#
$MAMS2  001266          2351#
$MAMS3  001272          2354#
$MAMS4  001276          2357#
$MBADR  000002          2126#
$MFLG   013102          4183*   4189    4224*   4228#
$MNEW   013777          4261    4401#
$MSGAD  001246          2329#   4199*   4202
$MSGLG  001250          2330#   4204*
$MSGTY  001232          2323#   4197    4205*   4217    4221*
$MSWR   013766          4258    4399#
$MTYP1  001263          2344#
$MTYP2  001267          2352#
$MTYP3  001273          2355#
$MTYP4  001277          2358#
$MXCNT  011354          3843    3853#
$NULL   001154          2290#   4150    4179
$NWTST= 000001          2667#
$OCNT   012126          3988*   4017*   4030#
$OMODE  012130          3983*   3987*   3992    3995*   4006*   4032#
$OVER   011340          3811    3831    3841    3850#
$PASS   001240          2326#   2594*   3768*   3769*   3777    3790    3837    3854
$PASTM  000006          2128#
$POWER  014564          4595    4602#
$PWRAD  014552          4597#
$PWRDN  014412          2571    4564#   4592
$PWRMG  014546          4595#
```

```
$PWRUP  014464          4574    4580#
$QUES   001226          2314#   3900    4179    4304    4380    4397    4453    4456    4514    4516
$RDCHR  013370          4317#   4556
$RDDEC  014150          4470#   4559
$RDLIN  013510          4345#   4557
$RDOCT  014010          4417#   4558
$RDSZ =  000010         4338#
$REGAD  001160          2294#
$REG0   001162          2296#   3366*   3384*   3402*   3420*   3466*   3515*   3535*   4709*   6036    6138    6201
$REG1   001164          2297#   3139*   3536*   4710*   5077*   5087*   5942    6036    6089    6117    6201
$REG2   001166          2298#   4711*   5078*   5088*   5942    6036    6089
$REG3   001170          2299#   4712*   5942    6036
$REG4   001172          2300#   4713*   5942    6036
$REG5   001174          2301#
$REG6   001176          2302#   5076*   5086*   6089
$REG7   001200          2303#
$RTNAD  011070          3789#
$R2A  =  ******  U      4560
$SAVRE=  ******  U      4560
$SAVR6  014562          4573*   4581    4582*   4583*   4601#
$SCOPE  011112          2565    3807#   4862
$SETUP=  000117         2557#   2564    2565    2567    2569    2571    2573    2574    2576    2599    3766    3808    3868
                        3891    3899    4241    4403
$STUP =  177777         2557#
$SVLAD  011304          3819    3844#
$SVPC =  000000         2103#   2108
$SWR  =  165000         2076#   2089    2094    2095    2096    2097    2098    2099    2312    2313    2314    2573    2574
                        2576    2577    2671    3761    3767    3782    3788    3790    3800    3801    3802    3803    3804
                        3810    3822    3824    3825    3826    3833    3834    3835    3847    3850    3853    3860    3861
                        3862    3863    3872    3876    3888    3892    3900    4598
$SWREG  001254          2334#   2597
$SWRMK=  000000         3804
$TESTN  001236          2325#   3845*
$TIMES  001222          2312#   2573*   2671*   3767*   3833*   3840    3843*   3853
$TKB    001146          2287#   4239    4250    4267    4321    4327    5241
$TKS    001144          2286#   4239    4248    4264    4288*   4319    4325
$TMP0   001202          2304#   2725*   2891*   2911    3023*   3140*   3182*   3183*   3184    3193    3204    3209    3210
                        3688*   3705    5282    5283    5620*   6063    6068    6089    6117
$TMP1   001204          2305#   2726*   3025*   6063    6068
$TMP2   001206          2306#   3026*   6068
$TMP3   001210          2307#   3027*   6068
$TMP4   001212          2308#   3028*   6068
$TMP5   001214          2309#   3029*   6068
$TMP6   001216          2310#   3030*   6068
$TMP7   001220          2311#   2951*   2966    2983*   2986*   2989*   3011*   3013    3731*
$TN   =  000002         2089#   2667    2671#
$TPB    001152          2289#   4168*   4179
$TPFLG  001157          2293#   4117    4179
$TPS    001150          2288#   4166    4179
$TRAP   014326          2569    4524#
$TRAP2  014350          4535#   4546
$TRP  =  000014         4539#   4548#   4549#   4550#   4551#   4552#   4553    4554#   4555    4556#   4557#   4558#   4559#
                        4560#
$TRPAD  014362          4529    4546#
$TSTM   000004          2127#
$TSTNM  001102          2266#   3039*   3766*   3799    3844*   3845    3850    3854    3871    3900
```

```
$TTYIN  013744          4347    4348    4360    4378    4392    4396#
$TYPBN= ****** U        4552
$TYPOS  012132          4045#   4551
$TYPE   012356          4117#   4210    4539    4547
$TYPEC  012570          4147    4154    4161    4166#   4167    4290
$TYPEX  012636          4172    4174    4177#
$TYPOC  011730          3986#   4548
$TYPON  011744          3985    3988#   4550
$TYPOS  011704          3981#   4549
$UNIT   001244          2328#
$UNITM  000010          2129#
$USWR   001256          2335#
$VECT1  001302          2360#
$VECT2  001304          2361#
$XTSTR  011130          3813#
$$GET4= 000000          3782#
$OFILL  012127          3982*   3986*   3996    4031#
$40CAT= ****** U        3810    3878
.       = 034130        2103    2104#   2106#   2108#   2114    2115#   2117#   2119#   2133#   2137#   2263#   2317    2562
                        2576    2577    2666    3790    3794    3853    3854    3900    3955#   4099#   4179    4231#   4239
                        4396#   4397    4403    4456    4516    4576    4600    5912#   6116#   6137#   6200#   6553#   6556#
                        6557#   6558#   6559#   6560#   6561#   6576#   6582#
.$ASTA= ****** U        4183    4186
.$X     = 000000        2114#   2119
```

```
COMMEN  2257#
ENDCOM  2257#
ERROR   2151#  2731  2757  2776  2795  2814  2834  2866  2886  2903  2963  3032  3143  3369  3387
        3405   3423  3469  3518  3538  3565  3584  3603  3622  3642  3663  3683  3700  3742  5081
        5091
ESCAPE  2257#
GETPRI  2257#
GETSWR  2077#  2257#  2599
MULT    2257#
NEWTST  2257#  2667
POP     2257#  2847  2921  4086  4225  4226  4445  4505  4585  4586  4701  5197
PUSH    2257#  2839  2908  4045  4186  4188  4209  4419  4472  4566  4572  4684  5186
REPORT  2257#
SCOPE   2152#  2670  3038  3765
SETPRI  2257#
SETTRA  4539#  4548  4549  4550  4551  4553  4555  4556  4557  4558  4559
SETUP   2077#  2257#  2557
SKIP    2257#
SLASH   2257#
SPACE   2257#
STARS   2101   2111  2113  2120  2257#  2259  2317  2320  2667  2669  3758  3796  3856  3902  3958
        4035   4102  4181  4238  4241   4309  4338  4405  4458  4518  4562  4578  4605  4607  5625
        5627   5920  5922  6207  6209
SWRSU   2257#  2578#
TRNTRP  4539#
TYPBIN  2257#
TYPDEC  2257#  3777  3942
TYPNAM  2257#
TYPNUM  2257#
TYPOCS  2257#
TYPOCT  2257#  3914  3939  4259
TYPTXT  2257#
$SCMRE  2257#  2296  2297  2298  2299  2300  2301  2302  2303
$SCMTM  2257#  2304  2305  2306  2307  2308  2309  2310  2311
$SESCA  2257#
$SNEWT  2257#  2667
$SSET   4539#  4548  4549  4550  4551  4553  4555  4556  4557  4558  4559
$SSETM  2594#
$SSKIP  2257#
.EQUAT  2077#  2147
.HEADE  2077#  2079
.SETUP  2077#  2557
.SWRHI  2077#  2090
.SWRLO  2077#  2099#
.SACTI  2077#  2099
.SAPTB  2077#  2318#
.SAPTH  2077#  2109
.SAPTY  2077#  4179
.SCATC  2077#  2131
.SCMTA  2077#  2257
.SEOP   2077#  3756
.SERRO  2077#  3854
.SERRT  2077#  3900
.SPOWE  2077#  4560
.SRDDE  2077#  4456
.SRDOC  2077#  4403
```

```
.$READ   2077#   4236
.$SCOP   2077#   3794
.$TRAP   2077#   4516
.$TYPD   2077#   4033
.$TYPE   2077#   4100
.$TYPO   2077#   3956


. ABS.  034130     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DZDHNC.BIN,DZDHNC.LST/CRF/SOL/NL:TOC=DZDHNC.P11
 RUN-TIME: 21 14 1 SECONDS
 RUN-TIME RATIO: 163/38=4.2
 CORE USED:  26K  (52 PAGES)
```

N12