DH11

RELIABILITY DIAGNOSTIC

MD-11-DZDHN-A

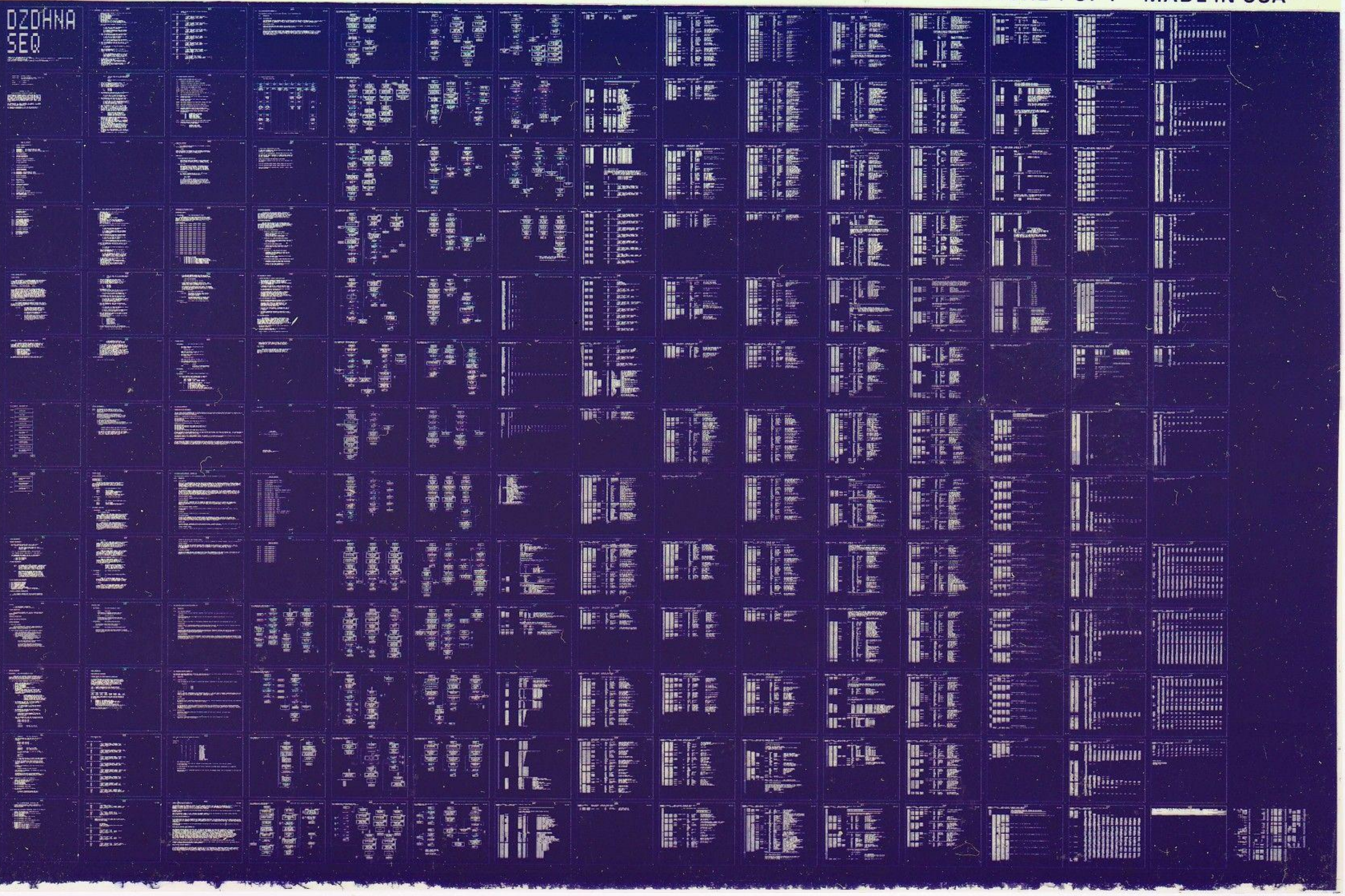
EP-DZDHN-A-DL-A

FEB 1976

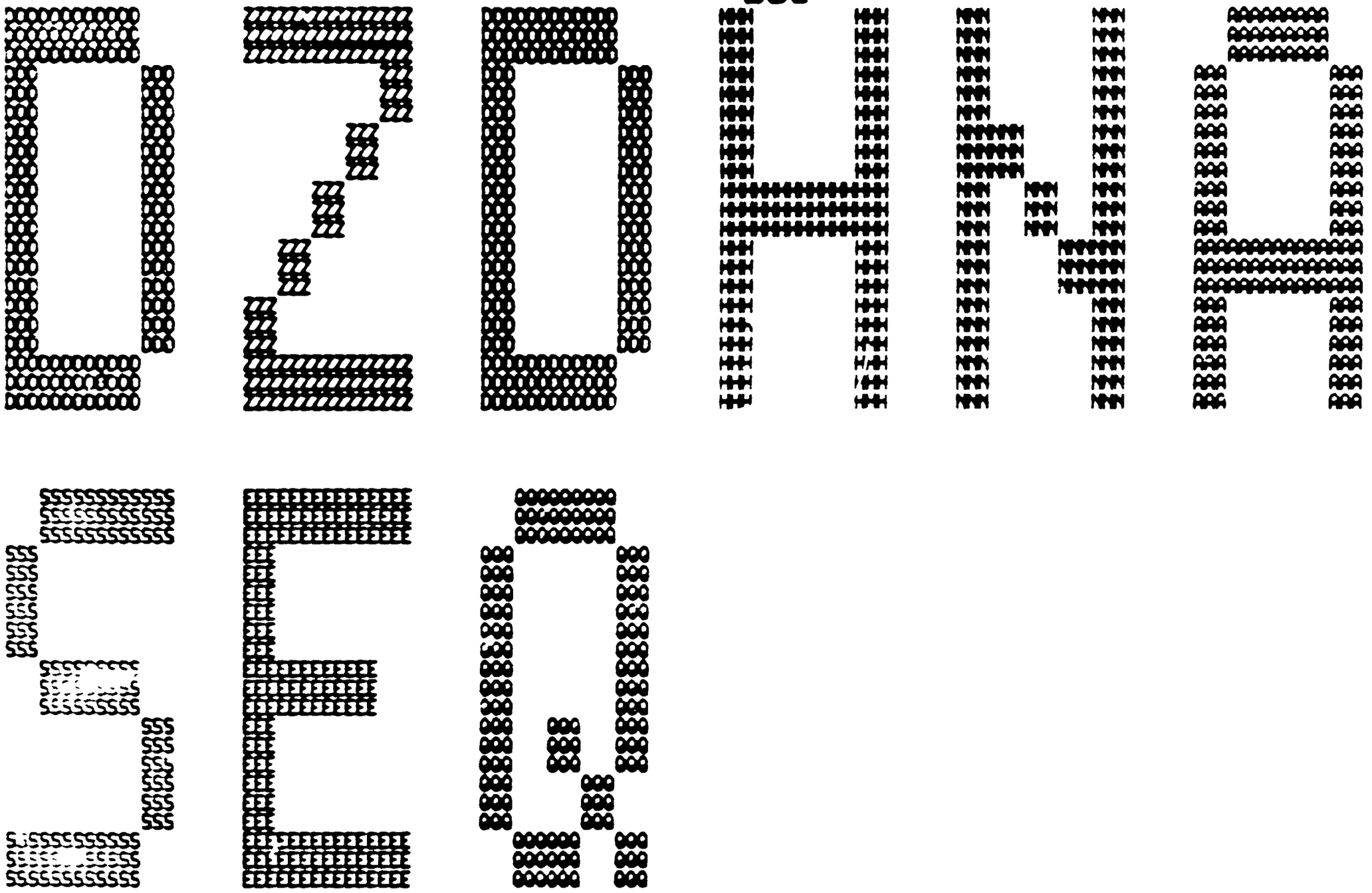
digital

FICHE 1 OF 1

MADE IN USA



B01



LPTSPL Version 101(2107) Running on MTA140
 START User WELLINGHAM,A[1,2] Job DZDHMA Seq. 5761 Date 15-Jan-76 11:38:53 Monitor RX22SD SYS #514/546 *START*
Request created: 15-Jan-76 11:21:39
File: WHTG:DZDHMA.SE0<055>[404,3722] Created: 09-Jan-76 16:08:38 Printed: 15-Jan-76 11:38:55
QUEUE Switches: /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:441 /FORMS:NORMAL

C01

PRODUCT CODE: MAINDEC-11-DZDHA-A SEQ 0001

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

DATE: 21-JANUARY-1976

AUTHOR: E. CROWLEY

MAINTAINED BY: DIAGNOSTIC ENGINEERING

TABLE OF CONTENTS

E01

1.0 GENERAL PROGRAM DESCRIPTION

1.1 PROGRAM PURPOSE

"MD-11-DZDHN-A" 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 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.

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. 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.

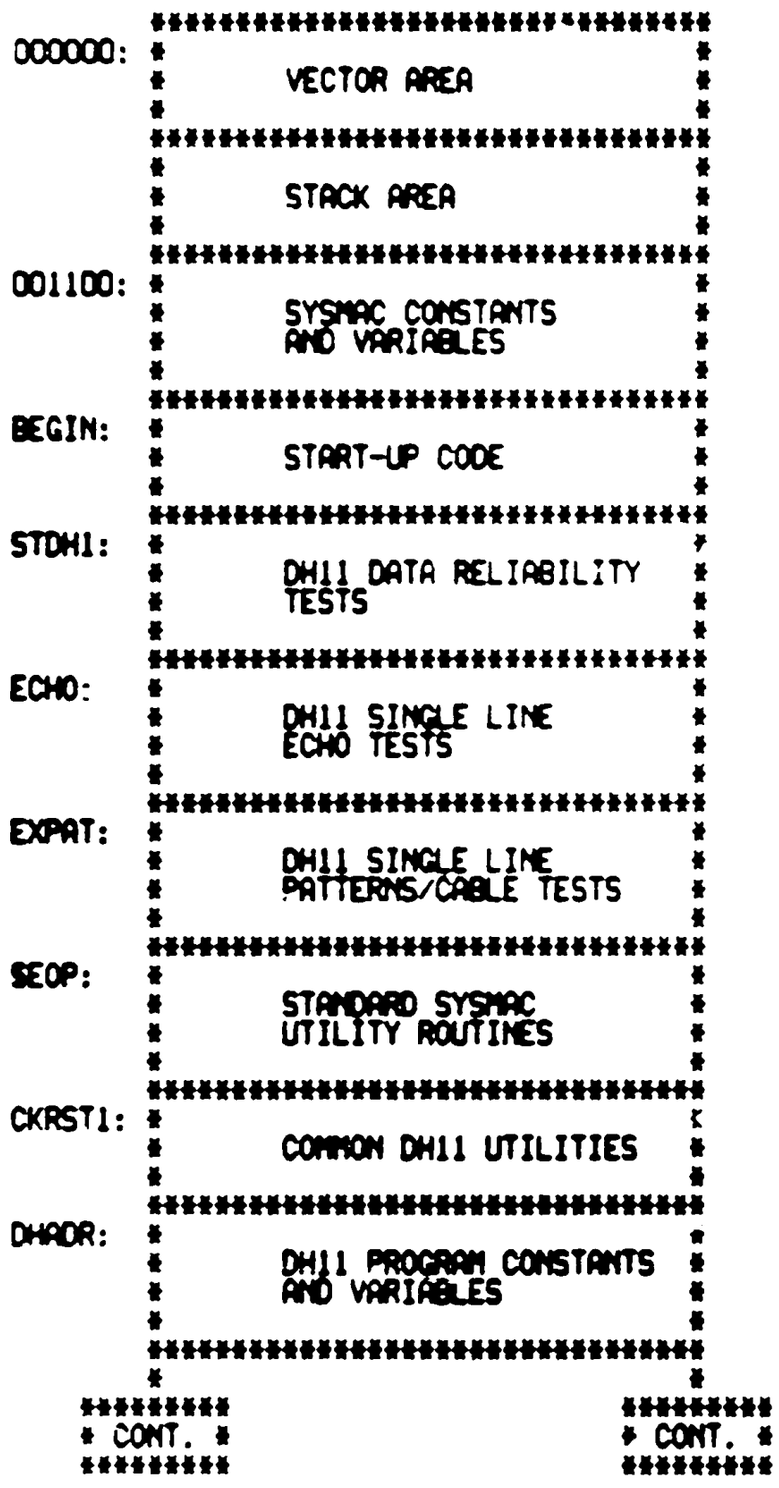
1.1.3 SUBPROGRAM 3 DH11 DATA PATTERNS/CABLE TESTS SEQ 0005

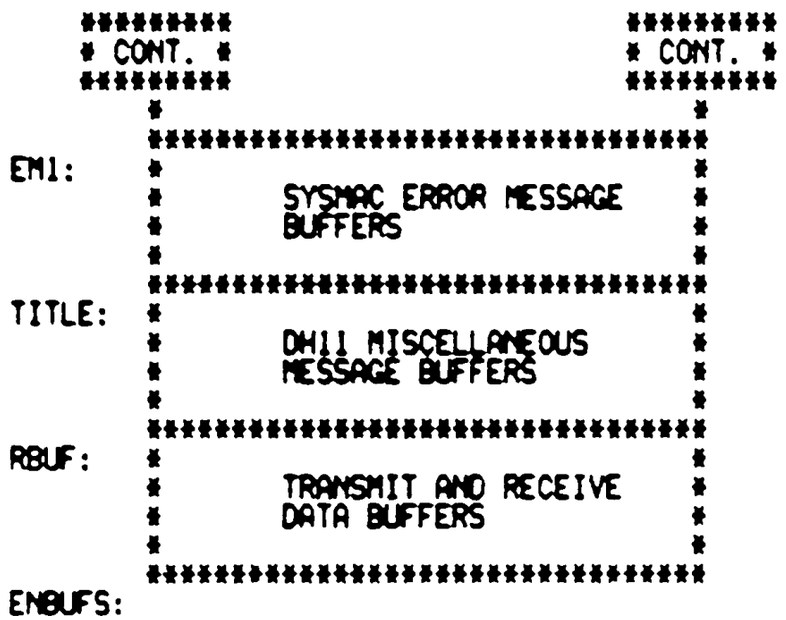
THIS PROGRAM PROVIDES THE MEANS OF TESTING ANY LINE ON ANY DH11 USING AN H315 TEST CONNECTOR TO TERMINATE THE LINE UNDER TEST. THE USER CAN SPECIFY BUFFER SIZE AND LINE PARAMETERS PRIOR TO SELECTING ONE OF THE FOLLOWING DATA PATTERNS FOR TRANSMISSION, RECEPTION, AND ERROR CHECKING:

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.

1.1.4 MC-11-DZDHN-A 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)

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, HOWEVER THE PROGRAM CAN NOT BE RUN AS PART OF A QUICK VERIFY "CHAIN" SINCE IT REQUIRES OPERATOR INTERVENTION TO SPECIFY PARAMETERS.

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 WHERE THE USER WILL BE ASKED TO TYPE IN THE DH11 PARAMETERS FOR HIS SYSTEM. THE PROGRAM CAN NOT BE RUN AS PART OF AN "XXDP" CHAIN UNLESS UPDATED AS DESCRIBED IN SECTION 2.2.2

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-BI 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-A ASSUMES THAT THE FOLLOWING DIAGNOSTICS HAVE BEEN RUN PRIOR TO ITS EXECUTION AND THAT NO ERRORS WERE

DETECTED: SEQ 0009

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

1.5 FAILURE ASSUMPTIONS

MD-11-DZDHN-A ASSUMES THAT THE DH11 HARDCORE VERIFIED BY MD-11-DZDHM-A, THE BASIC DH11 DIAGNOSTIC, IS FUNCTIONING ERROR FREE.

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURES

2.1.2 LOADING PROCEDURES

A. PAPER TAPE SYSTEMS

USE THE STANDARD PDP11 ABSOLUTE LOADER PROCEDURE FOR LOADING PAPER TAPES. AFTER LOADING THE PROGRAM MUST BE MANUALLY STARTED. (REFER TO SECTION 2.1.3)

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-A.
2. BOOT THE SYSTEM TO LOAD THE MONITOR
3. ONCE LOADED THE "XXDP" MONITOR PRINTS AN INTRODUCTORY MESSAGE AND RESPONDS WITH A "."
4. TYPE: "DZDHNA" 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 EXECUTION OF THE DATA RELIABILITY) ASK FOR THE DH11 PARAMETERS.

2.1.3 STARTING PROCEDURES

2.1.3.1 SUB-PROGRAM 1 DH11 DATA RELIABILITY TESTS

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:

A. TO SET UP DH11 SYSTEM PARAMETERS (START AT LOCATION 000200(8))

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

   NOTE: 1. SYSTEMS WHERE THE DM11-BB VECTORS ARE INTERLEAVED WITH THE DH11 VECTORS HAVE EIGHT WORDS BETWEEN VECTORS. (THIS IS THE CASE FOR THE 2040 FRONT END)
   2. STANDARD SYSTEMS HAVE THE DH11 VECTORS CONTIGUOUS WITH A FOUR WORD 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=1 TEST DH11 #02 ETC.

    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=1 TEST LINE #02 ETC.

    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 SR15=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 SR15=1 THE PROGRAM WILL HALT AND TYPE THE FOLLOWING MESSAGE:

    "DEPRESS CONTINUE TO START TESTING"

    WHEN CONTINUE IS DEPRESSED, THE PROGRAM WILL BEGIN TESTING 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.

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

1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE PARTICULAR TEST APPLICATION
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000204(8)
4. DEPRESS LOAD ADDRESS
5. SET THE SR=000000 (WORST CASE TESTING)

   SET THE SR=000200 (QUICK PASS)

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

** THE DEFAULT PARAMETERS ASSUME ONE DH11 WITH THE FOLLOWING ADDRESS ASSIGNMENTS

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

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

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

1. INSTALL THE REQUIRED TEST CONNECTORS FOR THE SPECIFIC TEST APPLICATION.
2. SET THE HALT/ENABLE SWITCH TO THE HALT POSITION
3. SET THE SR=000210
4. DEPRESS LOAD ADDRESS
5. SET THE SR=000000 (WORST CASE TESTING)

   SET THE SR=000200 (QUICK PASS)

   SET THE SR=100000 (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 (A-10) ABOVE.
9. PROGRAM WILL ASK FOR LINE SELECTION PARAMETERS. PROCEED AS IN (A-11) ABOVE.
10. PROGRAM WILL BEGIN EXECUTION AS DESCRIBED IN PARA 2.1.3.1 (A,12) ABOVE

2.1.3 2 SUBPROGRAM 2 DH11 SINGLE LINE ECHO TESTS SEQ 0013

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=000000
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 WORDS BETWEEN VECTORS. TYPE EITHER A "4" OR "10" AS DESCRIBED IN PARA 2.1.3.1 (A7) 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 000.

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.
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. 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 D00214(8).

SE0 0015

2.1.3.3 SUBPROGRAM 3 DH11 SINGLE LINE DATA PATTERNS/CABLE TESTS SEQ 0016

1. TERMINATE THE LINE TO BE TESTED WITH AN M315 TEST CONNECTOR.
2. SET THE HALT/ENABLE SWITCH TO HALT
3. SET THE SR=000220(8)
4. DEPRESS LOAD ADDRESS
5. SET THE SR=000000
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 WORDS BETWEEN VECTORS. TYPE EITHER A "4" OR "10" AS DESCRIBED IN PARA 2.1.3.1 (A 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 TERMINATOR 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).

2.1.4 RESTART PROCEDURES

SAME AS THE STARTING PROCEDURES

SPECIAL ENVIRONMENTS

2.2.1 ACT11/APT11 — THE PROGRAM MAY BE LOADED BY THE ACT11/APT11 SYSTEMS, BUT MAY NOT BE RUN AS PART OF A QUICK VERIFY CHAIN SINCE THE PROGRAM REQUIRES OPERATOR INTERVENTION.

2.2.2 XXDP — ALTHOUGH THE PROGRAM IS NOT DESIGNED TO RUN UNDER CHAIN MODE IN AN "XXDP" ENVIRONMENT, THE USER MAY MODIFY AND UPDATE THE PROGRAM TO CREATE A ".BIC" FILE ON THE "XXDP" MEDIA THAT MAY BE CHAINED USING THE FOLLOWING PROCEDURE:

1) LOAD THE UPDATE PROGRAM (UPD1 OR UPD2)
2) USE UPDATE TO LOAD "DZDHNA.BIN"
3) SET SR1S=1 AND START THE PROGRAM AT LOC 000200(8)
4) TYPE IN THE DH11 PARAMETERS TO MATCH THE SYSTEM TO BE TESTED.
5) AFTER THE PROGRAM HALTS , PATCH LOCATION 202(8) AS FOLLOWS:

   EXAMINE LOCATION 206(8) AND NOTE ITS CONTENTS

   DEPOSIT WHAT IS IN LOCATION 206(8) INTO LOCATION 202(8)

6) NOW RESTART THE UPDATE PROGRAM AND DUMP THE PROGRAM BACK ON TO THE "XXDP" MEDIUM USING THE NAME "DZDHNA.BIC"
7) THIS ".BIC" FILE MAY NOW BE USED TO RUN THE DATA RELIABILITY PROGRAM AS PART OF A CHAIN.

2.3 PROGRAM OPTIONS

2.3.1 CONSOLE SWITCH REGISTER

****
IMPORTANT NOTE
****

FOR ANY CPU THAT HAS NO PHYSICAL SWITCH REGISTER THERE IS A CORE LOCATION TAGGED "SWR:" THAT SERVES AS A SOFTWARE SWITCH REGISTER. THE BITS IN THIS LOCATION SERVE THE SAME FUNCTION AS THE SWITCH REGISTER BITS DESCRIBED BELOW.

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
SR7=1	QUICK VERIFY - DO COMPLETE TESTING ON EACH LINE AT 9600. BAUD ONLY

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

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 ADDRESS 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 CONFIGURATION.

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 (A 10) FOR A DESCRIPTION OF ITS ENCODING.

5) LINE SELECTION PARAMETER (PARA 2.1.3.1 (A11))

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.

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=1 QUICK TEST

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

2. SR7=0 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-A 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  SHAADR   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 EPROR 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
```

```
        NOTE:   ERRORS 17 THRU 24 ARE USED TO REPORT PERFORMANCE
                NOT ERRORS.

;ERROR TABLE ITEM FOR ERROR 17

        EM17            ;"ALTERNATING I/O PATTERN TEST DONE"
        DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
        DT7             ;SERRPC,DHADR,LINE,CURLPR,$REGO
        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 20

        EM20            ;"BINARY UP COUNT PATTERN TEST DONE"
        DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
        DT7             ;SERRPC,DHADR,LINE,CURLPR,$REGO
        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 21

        EM21            ;"BINARY DOWN COUNT PATTERN TEST DONE"
        DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
        DT7             ;SERRPC,DHADR,LINE,CURLPR,$REGO
        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 22

        EM22            ;"RANDOM DATA PATTERN TEST DONE"
        DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
        DT7             ;SERRPC,DHADR,LINE,CURLPR,$REGO
        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 23

        EM23            ;SINGLE CHAR PATTERN TEST DONE"
        DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
        DT7             ;SERRPC,DHADR,LINE,CURLPR,$REGO
        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 24

        EM24            ;"TYPED BUFFER PATTERN TEST DONE"
        DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
        DT7             ;SERRORPC,DHADR,LINE,CURLPR,$REGO
        DF2             ;PRINT ALL OCTAL

;ERROR TABLE ITEM FOR ERROR 25

        EM25            ;"DATA PATTERNS TEST TIMEOUT"
        DH7             ;" (PC) DEVADR   LINE   CURLPR  ICOUNT  PATCDE"
        DT10            ;SERRPC,DHADR,LINE,CURLPR,$REGO,$REG1
        DF2             ;PRINT ALL OCTAL
```

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.

WAS-ADR 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  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO

LLLLLL  RRRRRR  XXXXXX  DDDDDD  PPPPPP  FFFFFF  OOOOOO
```

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

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

"LINE #NN WAS DROPPED"

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

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

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

B. SUB-PROGRAM 2 DH11 SINGLE LINE ECHO TESTS

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

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

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

(NONE PROVIDED)

4.2 PROGRESS REPORTS

A. SUB-PROGRAM 1 DH11 DATA RELIABILITY TESTS

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

   "TESTING DH11 #NN"

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

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

   "TESTING LINE #NN"

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

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

   "END PASS #NNNNNN"

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

B. SUB-PROGRAM 2 DH11 SINGLE LINE ECHO TESTS

(NONE SUPPLIED)

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

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

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

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

THIS TYPEOUT MAY BE INHIBITED BY SETTING SR13=1.

5.0 DH11 DEVICE INFORMATION

SEQ 0032

5.1 ADDRESS AND VECTOR ASSIGNMENTS

THE DH11 USES FLOATING ADDRESSES AND IS LOCATED AFTER DJ11'S IN THE FLOATING ADDRESS SPACE THAT BEGINS AT LOCATION 760 010. BECAUSE THE DH11 HAS EIGHT REGISTERS, IT MUST BE ASSIGNED AN ADDRESS THAT IS A MULTIPLE OF 20 (OCTAL). ALL DH11'S IN A 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 STARTS AT ADDRESS 300. THE VECTORS STARTING AT 300 ARE USED IN THE FOLLOWING ORDER: DC11; KL11/DL11-A, B; DP11; DM11-A; DN11; DM11-BB; DR11-A; DR11-C; PA611 READERS; PA611 PUNCHES; DT11; DX11; DL11-C, D, E; DH11.

THE RECEIVER VECTOR IS THE LOWER NUMBERED VECTOR. THE PRIORITY OF THE RECEIVER AND TRANSMITTER INTERRUPTS ARE INDIVIDUALLY SELECTABLE BY MEANS OF TWO STANDARD POP11 PRIORITY JUMPER PLUGS. BR LEVEL 5 IS STANDARD.

5.2 REGISTER DEFINITION

THE FOLLOWING CHART PRESENTS THE BIT ASSIGNMENTS WITHIN EACH REGISTER: BITS MARKED UNUSED AND WRITE ONLY ARE ALWAYS READ AS ZERO. ATTEMPTING TO WRITE INTO UNUSED OR READ ONLY BITS HAS NO EFFECT ON THOSE BITS. INIT REFERS TO THE INITIALIZE SIGNAL GENERATED BY THE PROCESSOR (E.G. UPON EXECUTION OF A RESET INSTRUCTION). TRANSMIT AND RECEIVE ARE WITH RESPECT TO THE DH11. ALL BITS IN THE ACCOMPANYING DIAGRAMS ARE SHOWN IN THE STATE THEY ASSUME ON POWER CLEAR OR INIT.

5.2.1 THE SYSTEM CONTROL REGISTER - ADDRESS X00

THE SYSTEM CONTROL REGISTER IS A BYTE-ADDRESSABLE REGISTER. THE BIT ASSIGNMENT IS AS FOLLOWS:

BITS	DESCRIPTION

00-03 LINE SELECTION

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

04, 05 MEMORY EXTENSION

THE INFORMATION STORED IN THESE BITS BECOMES BITS 16 AND 17 RESPECTIVELY OF ANY CURRENT ADDRESS LOADED BY THE PROGRAM INTO THE CURRENT ADDRESS REGISTER. THESE BITS ARE READ/WRITE BUT, WHEN READ, REPRESENT ONLY THE STATUS OF BITS 4 AND 5 OF THE SYSTEM CONTROL REGISTER, NOT THE STATUS OF ADDRESS BITS 16 AND 17 OF THE SELECTED LINE. SEE THE SILO STATUS REGISTER FOR FURTHER INFORMATION. THIS ARRANGEMENT PERMITS INTERRUPT SERVICE ROUTINES TO 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 THE SILO EXCEEDS THE "ALARM LEVEL" SPECIFIED BY THE LOW BYTE OF THE SILO STATUS REGISTER. THIS BIT IS READ ONLY, EXCEPT IN MAINTENANCE MODE, WHERE IT IS READ/WRITE. SETTING OF THIS BIT WILL GENERATE AN INTERRUPT REQUEST IF BIT 6 (ABOVE) IS ALSO SET.

08 CLEAR NON-EXISTENT MEMORY INTERRUPT

THIS BIT, WHEN SET, CLEARS THE NON-EXISTENT MEMORY INTERRUPT FLIP-FLOP (BIT 10) AND CLEARS ITSELF. THIS BIT 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 MEMORY LOCATION ON THE UNIBUS AND NO SLAVE SYNC IS RECEIVED IN 20 US. THIS INDICATES THAT THE ADDRESSED LOCATION OR DEVICE DOES NOT EXIST. THIS BIT CAUSES AN INTERRUPT REQUEST IF SET WHILE TRANSMITTER AND NON-EXISTENT MEMORY INTERRUPT ENABLE IS SET. THIS BIT IS READ ONLY, EXCEPT IN MAINTENANCE MODE, WHERE IT IS READ/WRITE.

11 MASTER CLEAR

THIS BIT, WHEN SET, GENERATES "INITIALIZE" WITHIN THE DH11, CLEARING THE SILO, THE UARTS, AND THE REGISTERS. THE EXACT BITS CLEARED ARE DISCUSSED IN THE SECTION ON INITIALIZATION. READ/WRITE.

12 STORAGE INTERRUPT ENABLE

THIS BIT, WHEN SET, PERMITS THE SETTING OF BIT 14 TO GENERATE AN INTERRUPT REQUEST. THIS BIT IS READ/WRITE.

13 TRANSMITTER AND NON-EX-MEM INTERRUPT ENABLE SEQ 0034

THIS BIT, WHEN SET, PERMITS THE SETTING OF BIT 10 OR 15 TO GENERATE AN INTERRUPT REQUEST. THIS BIT IS READ/WRITE.

14 STORAGE INTERRUPT

THIS BIT IS SET WHEN THE RECEIVER SCANNER FINDS A RECEIVER HOLDING BUFFER WITH A CHARACTER IN IT, TRIES TO STORE THAT CHARACTER IN THE SILO, AND CANNOT DO SO BECAUSE OF A LACK OF SPACE. WHEN SET THIS BIT WILL CAUSE AN INTERRUPT REQUEST IF BIT 12 IS SET. THIS BIT IS READ ONLY, EXCEPT IN MAINTENANCE MODE, WHERE IT IS READ/WRITE.

15 TRANSMITTER INTERRUPT

THIS BIT IS SET WHEN THE DH11 CONCLUDES AN NPR CYCLE THAT INCREMENTED A BYTE COUNT TO ZERO, INDICATING THE LAST CHARACTER IN A MESSAGE BUFFER WAS LOADED INTO A UART TRANSMITTER HOLDING REGISTER. THIS BIT WILL CAUSE AN INT' 'PT REQUEST IF BIT 13 IS SET. THIS BIT IS READ/WRITE. (IT IS SET DURING AN NPR CYCLE.)

5.2.2 NEXT RECEIVED CHARACTER REGISTER ADDRESS X02 SEQ 0035

BITS	DESCRIPTION
00-07	NEXT RECEIVED CHARACTER THESE BITS CONTAIN THE NEXT RECEIVED CHARACTER, RIGHT JUSTIFIED. THE LEAST SIGNIFICANT BITS IS BIT 00.
08-11	LINE NUMBER THESE BITS INDICATE THE LINE NUMBER ON WHICH THE NEXT RECEIVED CHARACTER WAS RECEIVED. BIT 8 IS THE LEAST SIGNIFICANT BIT.
12	PARITY ERROR THIS BIT IS SET IF THE PARITY OF THE RECEIVED CHARACTER DOES NOT AGREE WITH THAT DESIGNATED FOR THAT LINE.
13	FRAMING ERROR THIS BIT IS SET IF THE RECEIVER SAMPLES A LINE FOR THE FIRST STOP BIT, AND FINDS THE LINE IN A SPACING CONDITION (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 THAT WAS LOST DUE TO THE INABILITY OF THE RECEIVER SCANNER TO SERVICE THE UART RECEIVER HOLDING BUFFER. REFER TO THE SECTION ON PROGRAMMING FOR FURTHER DETAILS ON DOUBLE-BUFFERED RECEPTION.
15	VALID DATA PRESENT THIS BIT INDICATES THAT THE DATA PRESENTED IN BITS 14-00 IS VALID. IT PERMITS A CHARACTER HANDLING PROGRAM TO TAKE CHARACTERS FROM THE SILO UNTIL IT IS EMPTY. THIS IS DONE BY READING THIS REGISTER AND CHECKING BIT 15 UNTIL A WORD IS OBTAINED FOR WHICH BIT 15 IS A ZERO. THE ENTIRE NEXT RECEIVED CHARACTER REGISTER IS READ-ONLY AND IS ADDRESSABLE ONLY ON A WORD BASIS.

5.2.3 LINE PARAMETER REGISTER ADDRESS X04 SEQ 0036

THIS REGISTER SHOULD BE LOADED ONLY AFTER THE LINE SELECTION BITS OF THE SYSTEM CONTROL REGISTER HAVE BEEN SET TO SELECT THE 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 THE LENGTH (EXCLUDING PARITY) SHOWN:

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 CODE TO TRANSMIT CHARACTERS HAVING TWO STOP MARKS. IF THE LINE IS TRANSMITTING 5-BIT CODE, ASSERTION OF THIS BIT CAUSES THE CHARACTERS TO BE TRANSMITTED WITH 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 APPROPRIATE PARITY BIT AFFIXED, AND CHARACTERS 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 GENERATED ON THIS LINE AND INCOMING CHARACTERS WILL BE EXPECTED TO HAVE ODD PARITY. IF THIS BIT IS NOT SET, BUT BIT 4 IS SET, CHARACTERS OF EVEN PARITY WILL BE GENERATED ON THIS LINE AND INCOMING CHARACTERS WILL BE EXPECTED TO HAVE EVEN PARITY. IF BIT 4 IS NOT SET, THE SETTIN OF THIS BIT IS IMMATERIAL.

06-09 RECEIVER SPEED

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

10-13 TRANSMITTER SPEED

THE STATE OF THESE BITS DETERMINES THE OPERATING SPEED FOR THIS LINE'S TRANSMITTER. THE SPEED TABLE ON THE NEXT PAGE IS APPLICABLE.

SPEED TABLE FOR RECEIVER AND TRANSMITTER SPEEDS: SEQ 0037

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

14 HALF DUPLEX/FULL DUPLEX

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

IN THIS APPLICATION HALF-DUPLEX MEANS THAT THE DH11 RECEIVER IS BLINDED DURING TRANSMISSION OF A CHARACTER.

15 AUTO-ECHO ENABLE

WHEN THIS BIT IS SET, CHARACTERS RECEIVED ON THIS LINE WILL BE HARDWARE ECHOED. SEE THE DISCUSSION OF AUTO-ECHO FOR FURTHER DETAILS.

5.2.4 CURRENT ADDRESS REGISTER ADDRESS X06 SEQ 0038

THIS REGISTER SHOULD BE LOADED ONLY AFTER THE SYSTEM CONTROL REGISTER (SCR) HAS HAD THE APPROPRIATE BITS SET TO SELECT THE DESIRED LINE NUMBER. WHEN THIS REGISTER IS LOADED, ADDRESS BITS 00-15 ARE TRANSFERRED INTO SEMICONDUCTOR MEMORIES IN THE DH11 FROM BITS 00-15 OF THIS REGISTER. ADDRESS BITS 16-17 ARE TRANSFERRED INTO SEMICONDUCTOR 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 BITS 0-3 AND THE READ OR WRITE OF THE CURRENT ADDRESS REGISTER.

WHEN THIS REGISTER IS READ, IT WILL INDICATE THE CURRENT ADDRESS OF THE LINE SELECTED BY THE SYSTEM CONTROL REGISTER. 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 REGISTER SHOULD NOT BE LOADED OR READ WITHOUT FIRST SELECTING A LINE NUMBER BY MEANS OF THE LOWER-ORDER FOUR BITS OF THE SYSTEM CONTROL REGISTER. THIS REGISTER SHOULD BE LOADED WITH THE TWO'S COMPLEMENT OF THE NUMBER OF CHARACTERS (BYTES) TO BE TRANSMITTED ON THAT LINE. THE BYTE COUNT REGISTER IS READ/WRITE.

INTERRUPTS MUST BE INHIBITED OR THE SCR SAVED BETWEEN THE SETTING OF THE SCR BITS 0-3 AND THE READ OR WRITE OF THE BYTE COUNT REGISTER

5.2.6 BUFFER ACTIVE REGISTER (BAR) ADDRESS X12

THIS REGISTER CONTAINS ONE BIT FOR EACH LINE. THE BITS ARE INDIVIDUALLY SET USING BIS INSTRUCTIONS. SETTING A BIT INITIATES TRANSMISSION ON THE ASSOCIATED LINE. THE BIT IS CLEARED BY THE HARDWARE WHEN THE LAST CHARACTER TO BE TRANSMITTED IS LOADED INTO THE TRANSMITTER DATA HOLDING REGISTER OF THE UART FOR THAT LINE. IT SHOULD BE NOTED THAT WHILE THE CLEARING OF A BAR DOES INDICATE THAT A MESSAGE MAY BE SENT, IT DOES NOT INDICATE THAT THE LAST CHARACTERS FROM THE PRECEDING MESSAGE HAVE BEEN COMPLETELY SENT. SPECIFICALLY, TWO MORE CHARACTERS WILL BE SENT AFTER THE BAR BIT CLEARS. THESE ARE THE LAST TWO CHARACTERS OF THE MESSAGE; ONE OF THEM WAS JUST STARTING WHEN THE BAR WAS CLEARED AND ONE WAS THAT FINAL CHARACTER THAT WAS LOADED INTO THE HOLDING REGISTER, THUS CLEARING THE BAR BIT. THIS EFFECT IS A NORMAL CONSEQUENCE OF DOUBLE-BUFFERED TRANSMISSION AND IS MENTIONED HERE FOR THE BENEFIT OF PROGRAMMERS WHO WANT TO WRITE PROGRAMS THAT CONTROL SUCH MODEM LEADS ARE REQUEST TO SEND. REQUEST TO SEND (RTS) SHOULD NOT BE DROPPED UNTIL AT LEAST TWO CHARACTER TIMES AFTER THE BAR BIT FOR A GIVEN LINE CLEARS.

THIS TIMING MAY BE EFFECTED BY SENDING TWO EXTRA (NULL) CHARACTERS IN A MESSAGE AND DROPPING RTS WHEN BAR CLEARS.

CLEARING A BAR BIT SHOULD NOT BE USED TO ABORT TRANSMISSION ON A LINE. RATHER, THE BYTE COUNT FOR THAT LINE SHOULD BE SET 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 WILL IMMEDIATELY GENERATE A BREAK CONDITION ON THE LINE CORRESPONDING TO THAT BIT NUMBER. CLEARING THE BIT WILL TERMINATE THE BREAK CONDITION. THE BREAK CONDITION MAY BE TIMED BY SENDING CHARACTERS DURING THE BREAK INTERVAL, SINCE THESE CHARACTERS WILL NEVER ACTUALLY REACH THE LINE. FURTHER COMMENTS CONCERNING THE TRANSMISSION OF BREAK SIGNALS MAY BE FOUND IN THE BREAK SIGNALS SECTION.

5.2.8 SILO STATUS REGISTER ADDRESS X16 SEQ 0039

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 LOCATION (E.G., 0, 1, 2, 4, 8, 16, OR 32). WHEN THE NUMBER OF CHARACTERS STORED IN THE SILO EXCEEDS THAT NUMBER, AN INTERRUPT REQUEST (SYSTEM CONTROL REGISTER BIT 7) IS GENERATED, IF SYSTEM CONTROL REGISTER BIT 6 IS SET. THESE BITS ARE READ/WRITE.

06-07 READ EXTENDED MEMORY

THESE BITS ARE READ ONLY AND CONTAIN THE A16 AND A17 BITS OF THE CURRENT LINE ADDRESS WHICH THE LINE 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 CHARACTERS IN THE SILO. IT SHOULD BE NOTED THAT THERE ARE SIX BITS, HENCE NUMBERS BETWEEN 0 AND 63 CAN BE REPRESENTED. A FULL SILO HAS 64 ENTRIES AND THE FILL LEVEL APPEARS AS 00000, BUT ONE MAY EASILY TELL THE DIFFERENCE BETWEEN AN EMPTY SILO (00000) AND A FULL SILO (00000) BY CHECKING THE STORAGE OVERFLOW BIT (BIT 14 OF SYSTEM CONTROL). THESE BITS ARE READ ONLY.

5.3 DH11 MODULE ALLOCATION CHART

SEQ 0040

VIEW FROM WIRING SIDE

	SLOT 1	2	3	4	5	6	7	8	9
	M920	M7821	M7277	M7287	M7289	M7821	M7360	M7288	M920
	CABLE								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)	LINE PARAMETER CNTL	UNIBUS CONNECTOR (NOTES #1 & #2)
		M796				M405	M971		
							CABLE		
B		UNIBUS MASTER CNTL				EXTERNAL B CLOCK (NOTE #5)	DATA CABLE (NOTES #6 & #9)		
	M7247	M7247				M7280	M7280		M7279
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		FIFO BUFFER
D									
	M105	M7246							M405
E	* ADDRESS SELECTOR (NOTE #7)	* CONTROL SCAN (NOTES #4 & #8)							EXTERNAL A CLOCK (NOTE #5)
	M7821								M4540
F	* INTR CNTL (NOTE #7)								DH11 DC11 CLOCK

FIGURE 2-4 DH11 MODULE UTILIZATION DIAGRAM

NOTES:

1. IF END OF BUS, REPLACE M920 WITH M930.
2. IF LAST UNIT IN BASIC BOX, REPLACE M920 WITH BC11A CABLE WHEN EXPANDING TO PERIPHERAL BOX.
3. IF FIRST UNIT IN EXPANDER BOX, REPLACE M920 WITH BC11A CABLE.
4. E02 MUST BE G727 GRANT CONTINUITY IF MODEM CONTROL MODULE SET IS NOT INSTALLED. * DENOTES DH11-BB MODEM CONTROL OPTION, WITH DH11-AA OR AC.
5. MODULE SLOTS PROVIDE FOR ADDITIONAL CLOCK RATES.
6. FOR DIAGNOSTIC CHECKOUT OF DH11-AA, AB, OR AC, REPLACES M971 WITH M974.
7. THIS SLOT CONTAINS MODEM CONTROL MODULE M7807 WITH DH11-AD.
8. THIS SLOT CONTAINS MODEM CONTROL MODULE M7808 WITH DH11-AD.
9. THIS SLOT CONTAINS EIA CONVERTER AND PRIORITY MODULE M5906 FOR DH11-AD OR AE.

6.0 MAINTENANCE PROCEDURES

THIS SECTION OUTLINES SOME GENERAL TECHNIQUES FOR USING MD-11DZDHN-A 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-DZDHN-A, 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 ALL LINES WITHOUT DATA CABLES

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

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

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 A87 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.

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-DZDHM-A AND START IT AT LOC 000200(8).
3) TYPE IN THE DESIRED DH11 PARAMETERS - SET THE SR=000000 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=000200(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-A, 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-DZDHM-A AND START IT AT LOC 000220(8) TO RUN THE DATA PATTERNS TESTS.

3) REFER TO PARA 2.1.3.3 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 2.1.3.2 FOR THE OPERATING INSTRUCTIONS FOR THE DH11 ECHO TEST.

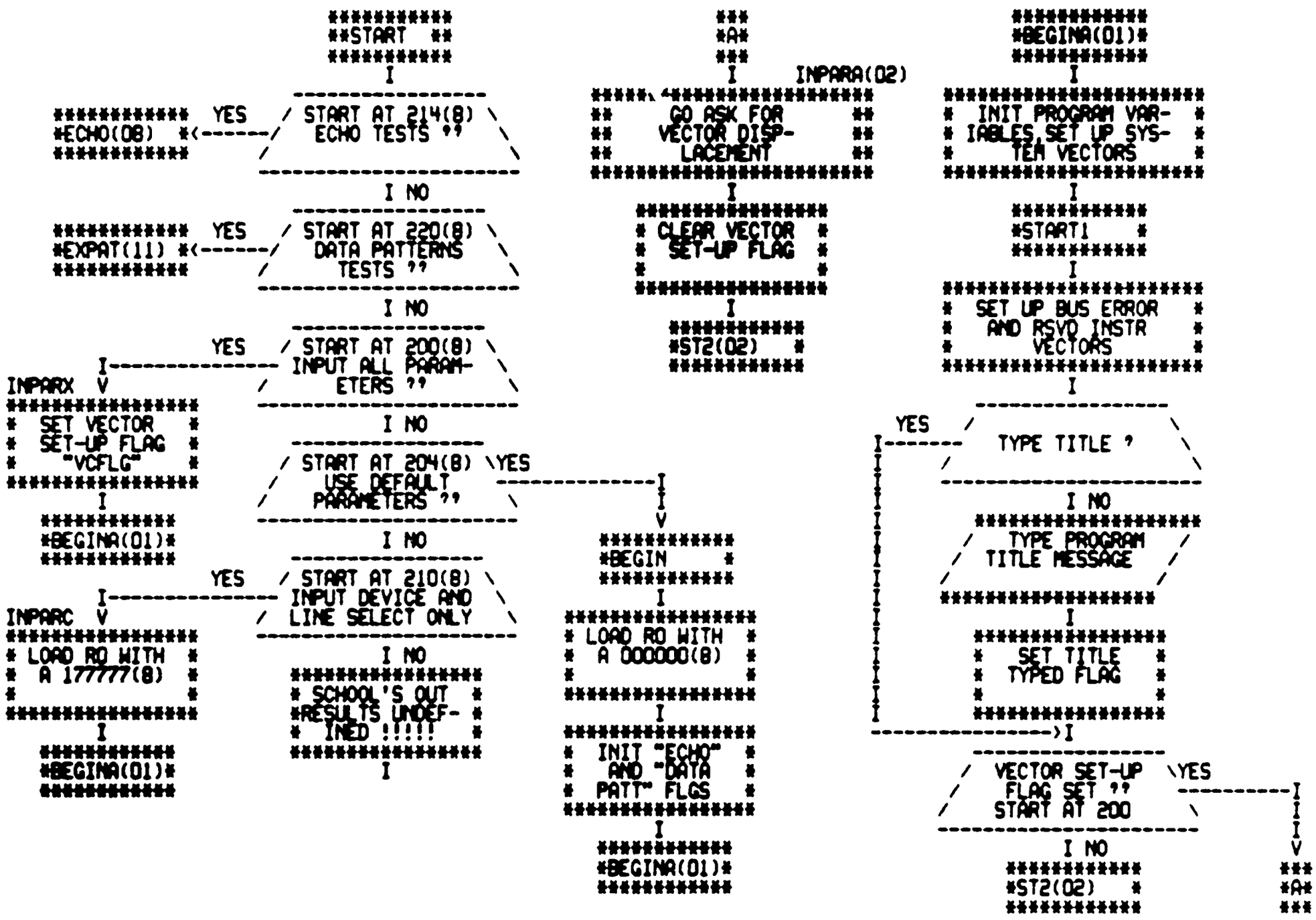
FLOW CHART
****

MD-11-DZDHN-A DH11 DATA RELIABILITY TESTS

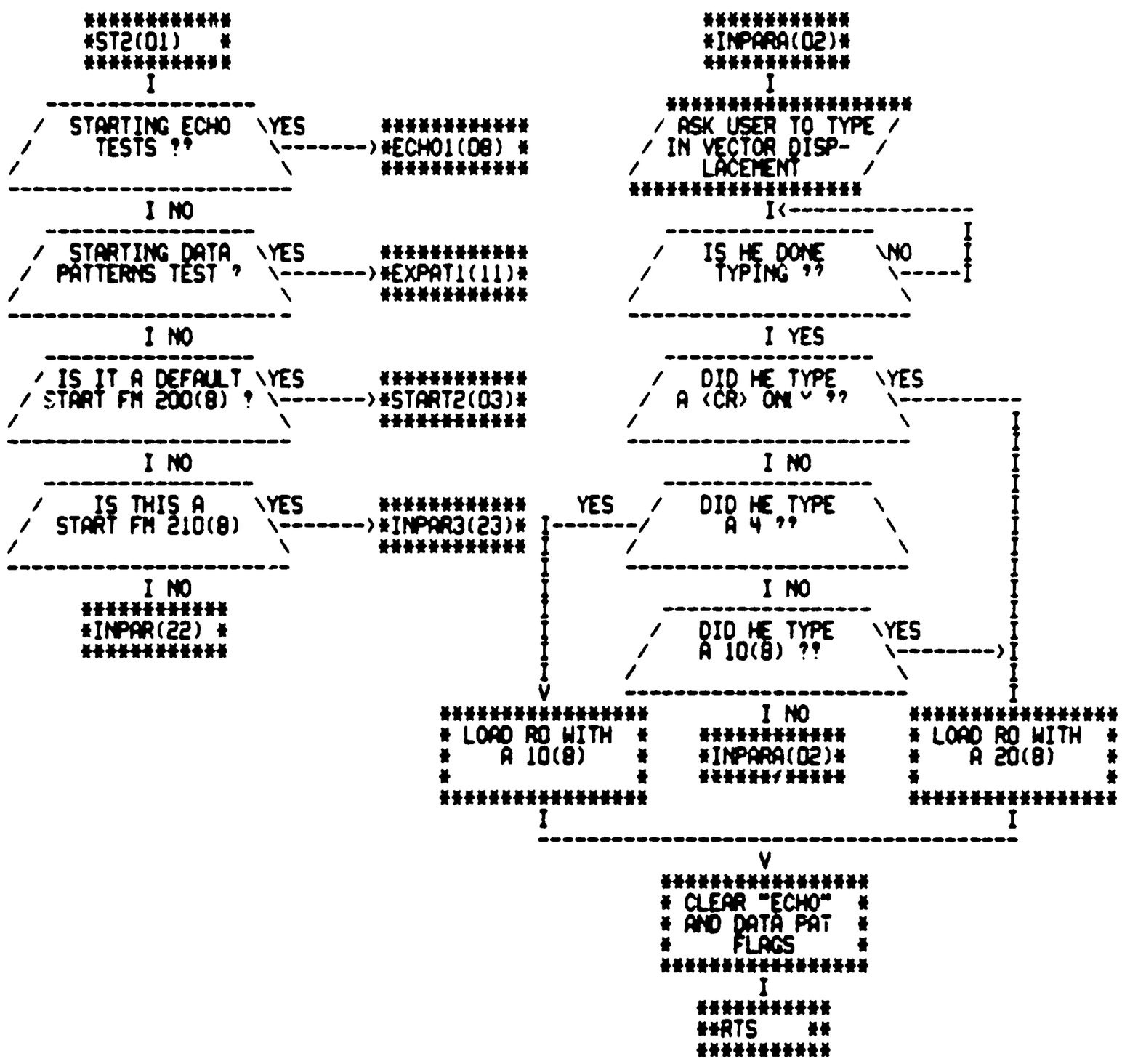
****

TABLE OF CONTENTS
****

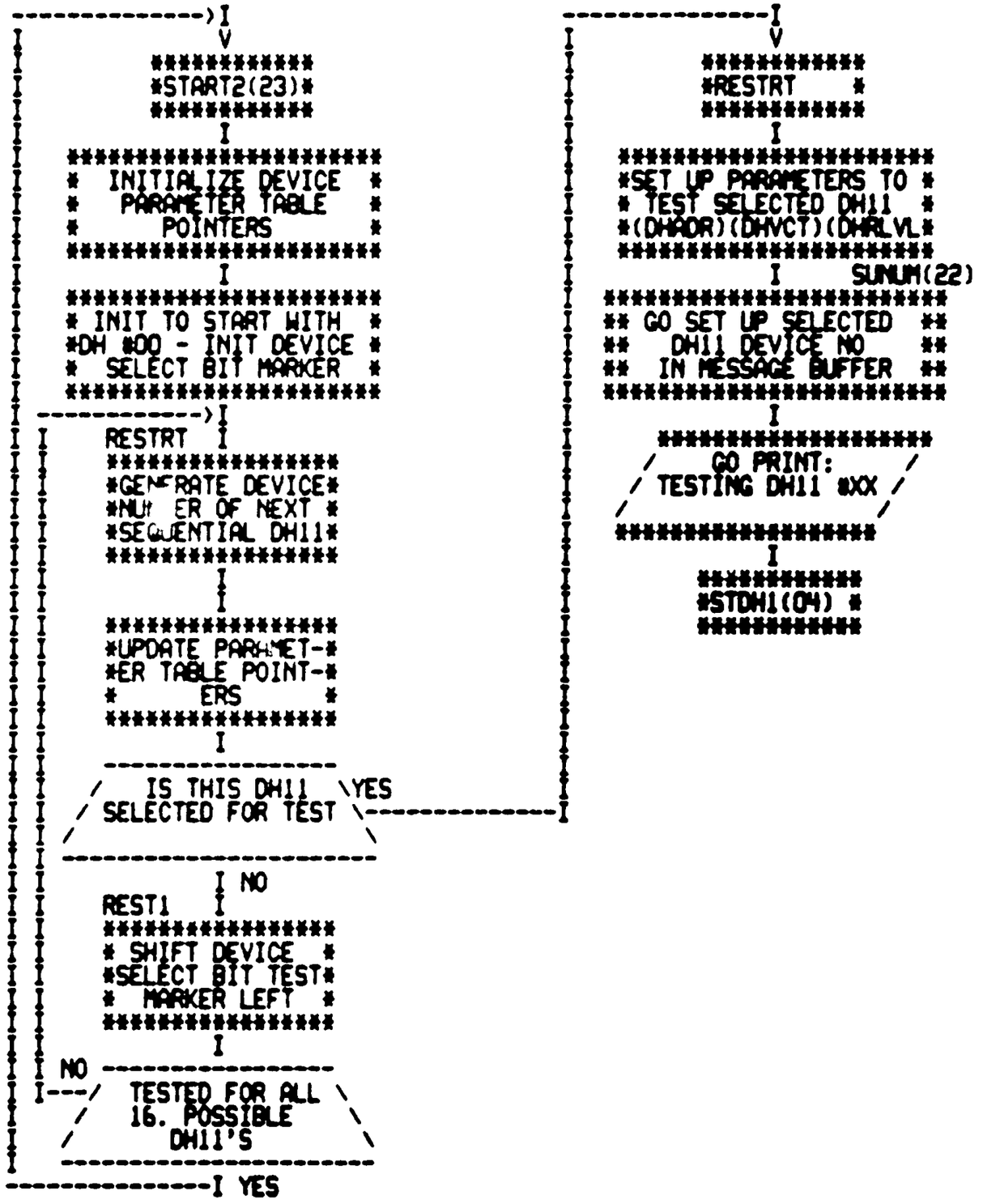
TABLE OF CONTENTS
****

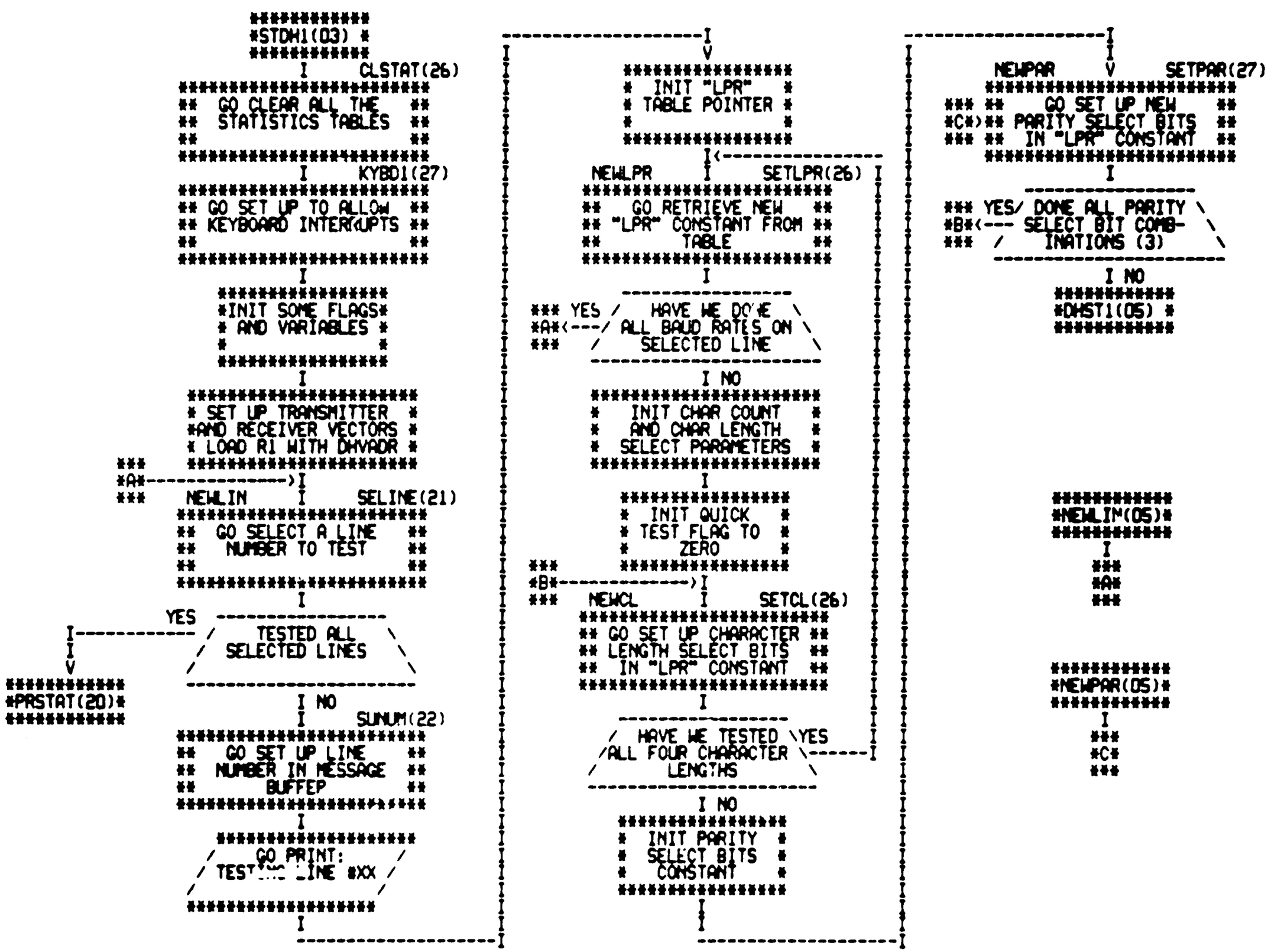

**START **
START AT 214(8) ECHO TESTS ??
YES
*ECHO(08) *
NO
START AT 220(8) DATA PATTERNS TESTS ??
YES
*EXPAT(11) *
NO
START AT 200(8) INPUT ALL PARAM- ETERS ??
YES
INPARX
NO
START AT 204(8) USE DEFAULT PARAMETERS ??
YES
NO
START AT 210(8) INPUT DEVICE AND LINE SELECT ONLY
YES
INPARC
NO
SCHOOL'S OUT
RESULTS UNDEF- INED !!!!!
SET VECTOR SET-UP FLAG "VCFLG"
LOAD R0 WITH A 177777(8)
BEGIN1(01)
*BEGIN *
LOAD R0 WITH A 000000(8)
INIT "ECHO" AND "DATA PATT" FLGS
BEGIN1(01)
A
GO ASK FOR VECTOR DISP- LACEMENT
INPARA(02)
CLEAR VECTOR SET-UP FLAG
*ST2(02) *
BEGIN1(01)
INIT PROGRAM VAR- IABLES,SET UP SYS- TEM VECTORS
*START1 *
SET UP BUS ERROR AND RSVD INSTR VECTORS
YES
TYPE TITLE ?
NO
TYPE PROGRAM TITLE MESSAGE
SET TITLE TYPED FLAG
VECTOR SET-UP FLAG SET ?? START AT 200
YES
NO
*ST2(02) *
A

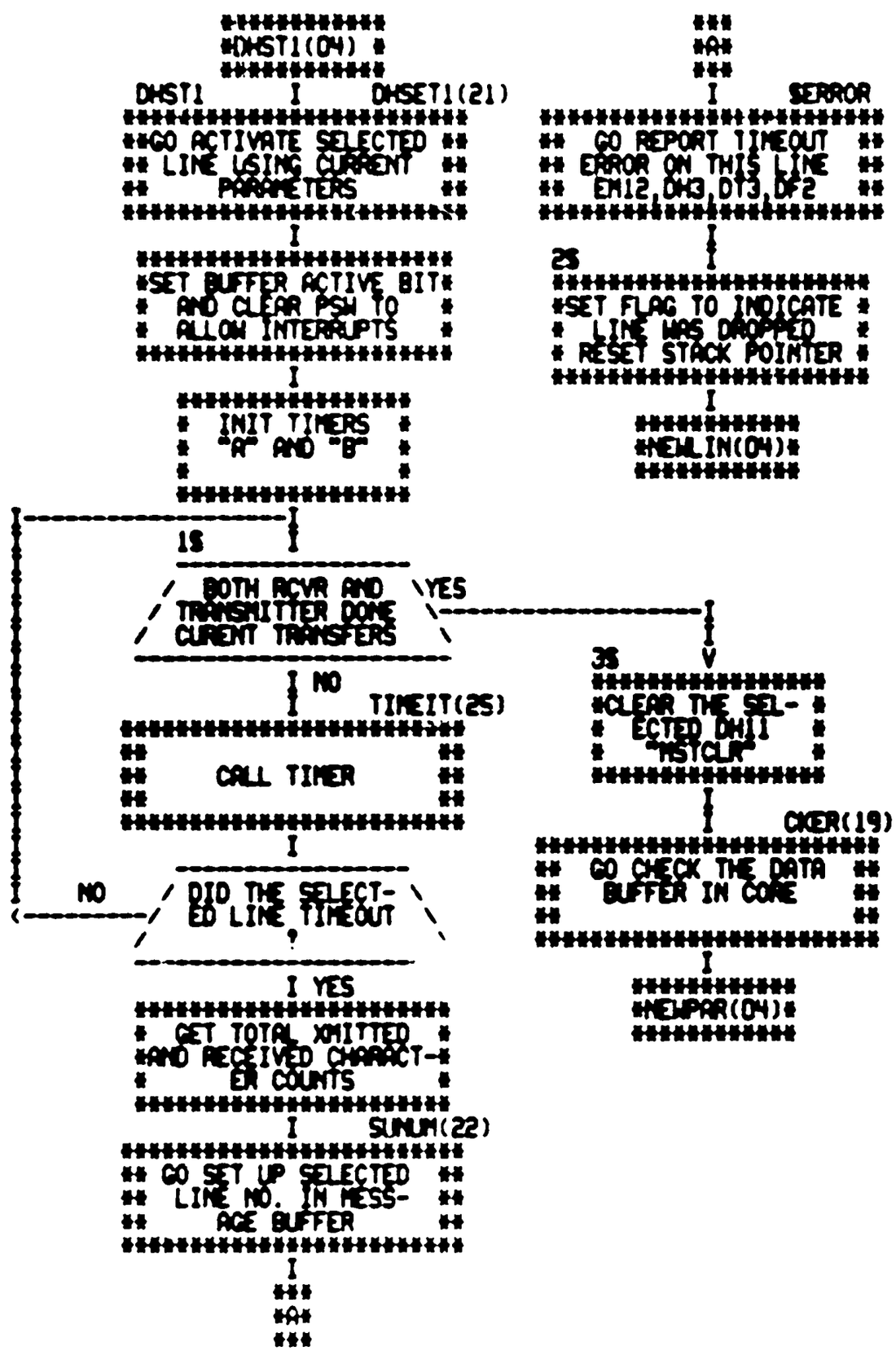
L04

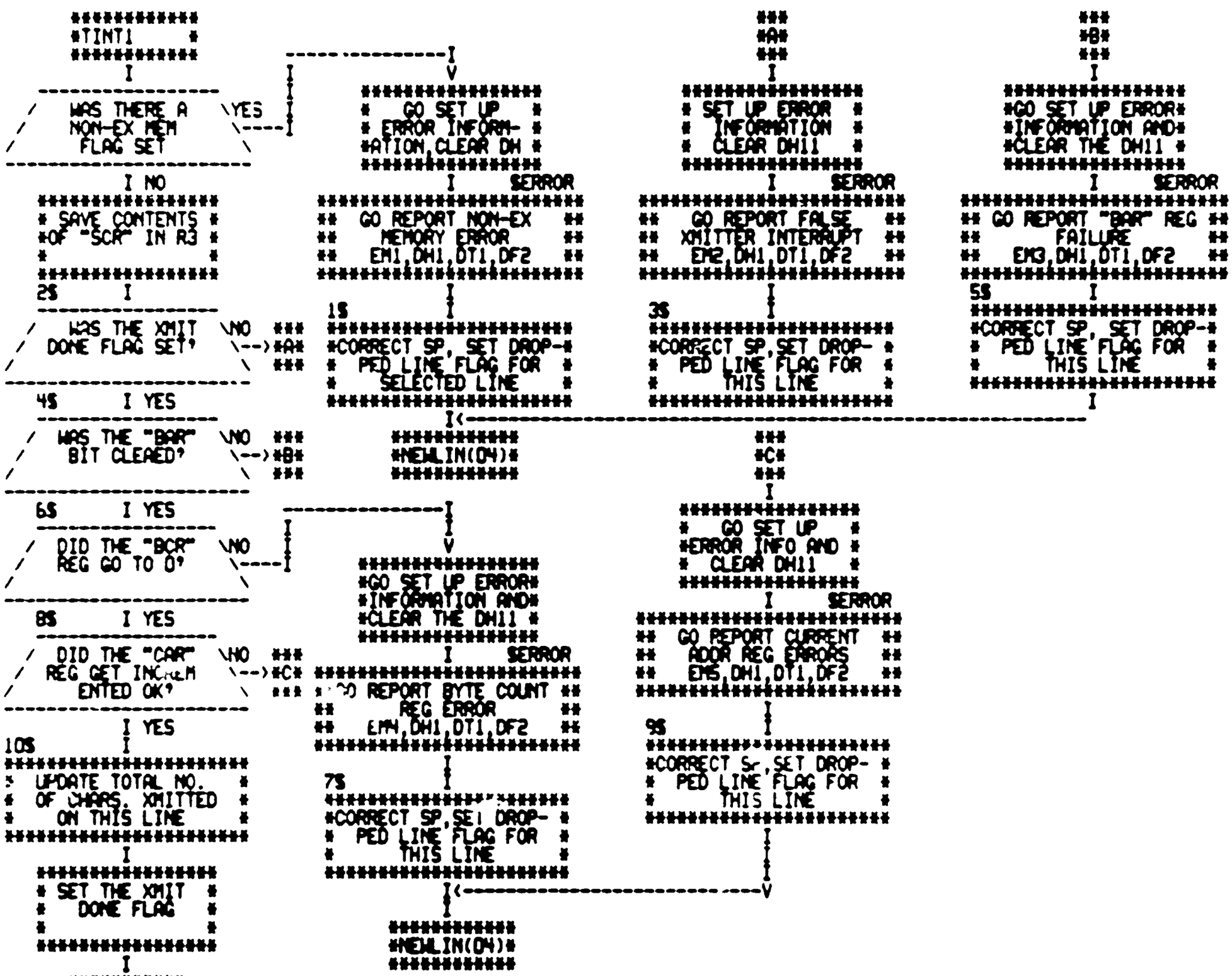

ST2(01)
STARTING ECHO TESTS ??
YES
ECHO1(08)
I NO
STARTING DATA PATTERNS TEST ?
YES
EXPAT1(11)
I NO
IS IT A DEFAULT START FM 200(8) ?
YES
START2(03)
I NO
IS THIS A START FM 210(8)
YES
INPAR3(23)
I NO
INPAR(22)
INPARA(02)
ASK USER TO TYPE IN VECTOR DISP-LACEMENT
IS HE DONE TYPING ??
NO
I YES
DID HE TYPE A <CR> ONLY ??
YES
I NO
DID HE TYPE A 4 ??
YES
I NO
DID HE TYPE A 10(8) ??
YES
I NO
INPARA(02)
* LOAD R0 WITH A 10(8) *
* LOAD R0 WITH A 20(8) *
* CLEAR "ECHO" AND DATA PAT FLAGS *
**RTS **

M04

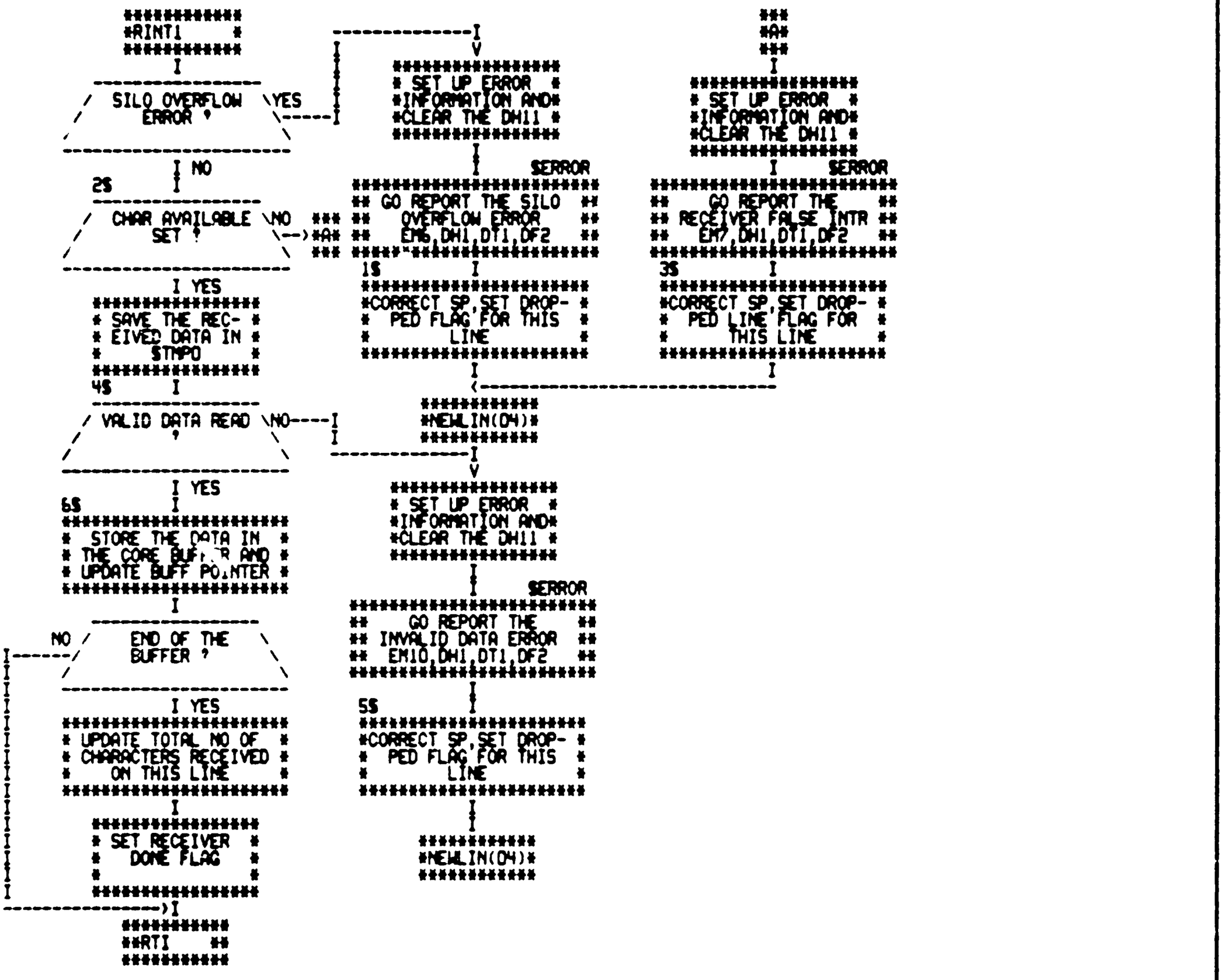

START2(23)
INITIALIZE DEVICE PARAMETER TABLE POINTERS
INIT TO START WITH #DH #00 - INIT DEVICE SELECT BIT MARKER
RESTRT
#GENERATE DEVICE NUMBER OF NEXT SEQUENTIAL DH11
#UPDATE PARAMETER TABLE POINTERS
IS THIS DH11 SELECTED FOR TEST
YES
NO
REST1
SHIFT DEVICE SELECT BIT TEST MARKER LEFT
TESTED FOR ALL 16. POSSIBLE DH11'S
YES
NO
RESTRT
SET UP PARAMETERS TO TEST SELECTED DH11 #(DHADR)(DHVCT)(DHRLVL
SUNUM(22)
GO SET UP SELECTED DH11 DEVICE NO IN MESSAGE BUFFER
GO PRINT: TESTING DH11 #XX
STDH1(04)


STDH1(03)
CLSTAT(26)
** GO CLEAR ALL THE STATISTICS TABLES **
KYBD1(27)
** GO SET UP TO ALLOW KEYBOARD INTERRUPTS **
INIT SOME FLAGS * AND VARIABLES *
* SET UP TRANSMITTER * *AND RECEIVER VECTORS * * LOAD R1 WITH DHVADR *
A
NEWLIN
SELINE(21)
** GO SELECT A LINE NUMBER TO TEST **
YES
TESTED ALL SELECTED LINES
PRSTAT(20)
I NO
SUNUM(22)
** GO SET UP LINE NUMBER IN MESSAGE BUFFER **
GO PRINT: TEST... LINE #XX
A
YES / HAVE WE DONE ALL BAUD RATES ON SELECTED LINE
I NO
* INIT CHAR COUNT AND CHAR LENGTH * * SELECT PARAMETERS *
* INIT QUICK * * TEST FLAG TO * * ZERO *
B
NEWCL
SETCL(26)
** GO SET UP CHARACTER LENGTH SELECT BITS ** ** IN "LPR" CONSTANT **
HAVE WE TESTED ALL FOUR CHARACTER LENGTHS \YES
I NO
* INIT PARITY * * SELECT BITS * * CONSTANT *
* INIT "LPR" * * TABLE POINTER *
NEWLPR
SETLPR(26)
** GO RETRIEVE NEW "LPR" CONSTANT FROM TABLE **
A
YES / HAVE WE DONE ALL BAUD RATES ON SELECTED LINE
I NO
* INIT CHAR COUNT AND CHAR LENGTH * * SELECT PARAMETERS *
* INIT QUICK * * TEST FLAG TO * * ZERO *
B
NEWCL
SETCL(26)
** GO SET UP CHARACTER LENGTH SELECT BITS ** ** IN "LPR" CONSTANT **
HAVE WE TESTED ALL FOUR CHARACTER LENGTHS \YES
I NO
* INIT PARITY * * SELECT BITS * * CONSTANT *
NEWPAR
SETPAR(27)
** GO SET UP NEW PARITY SELECT BITS ** ** IN "LPR" CONSTANT **
C
YES/ DONE ALL PARITY SELECT BIT COMBINATIONS (3)
I NO
DHST1(05)
NEWLIN(05)
NEWPAR(05)
A
C

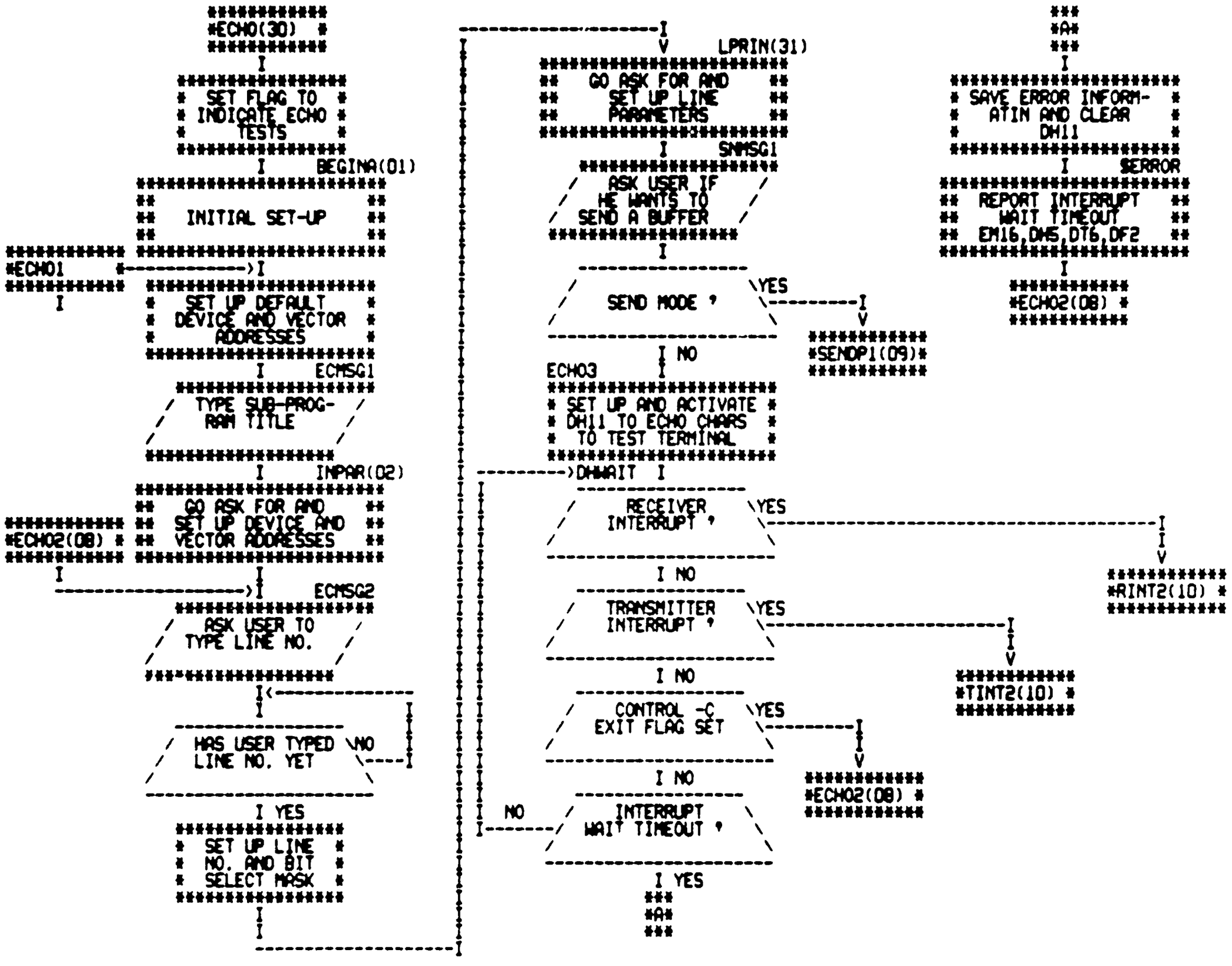

DHST1(04)
DHST1
DHSET1(21)
GO ACTIVATE SELECTED LINE USING CURRENT PARAMETERS
SET BUFFER ACTIVE BIT AND CLEAR PSW TO ALLOW INTERRUPTS
INIT TIMERS "A" AND "B"
1$
BOTH RCVR AND TRANSMITTER DONE CURRENT TRANSFERS
NO
TIMEIT(25)
CALL TIMER
DID THE SELECTED LINE TIMEOUT
NO
YES
GET TOTAL XMITTED AND RECEIVED CHARACT-ER COUNTS
SUMUM(22)
GO SET UP SELECTED LINE NO. IN MESS-AGE BUFFER
A
A
2$
SERROR
GO REPORT TIMEOUT ERROR ON THIS LINE
EM12,DH3,DT3,DF2
SET FLAG TO INDICATE LINE WAS DROPPED
RESET STACK POINTER
NEWLIN(04)
3$
CLEAR THE SEL-ECTED DH11 "MSTCLR"
CKER(19)
GO CHECK THE DATA BUFFER IN CORE
NEWPAR(04)

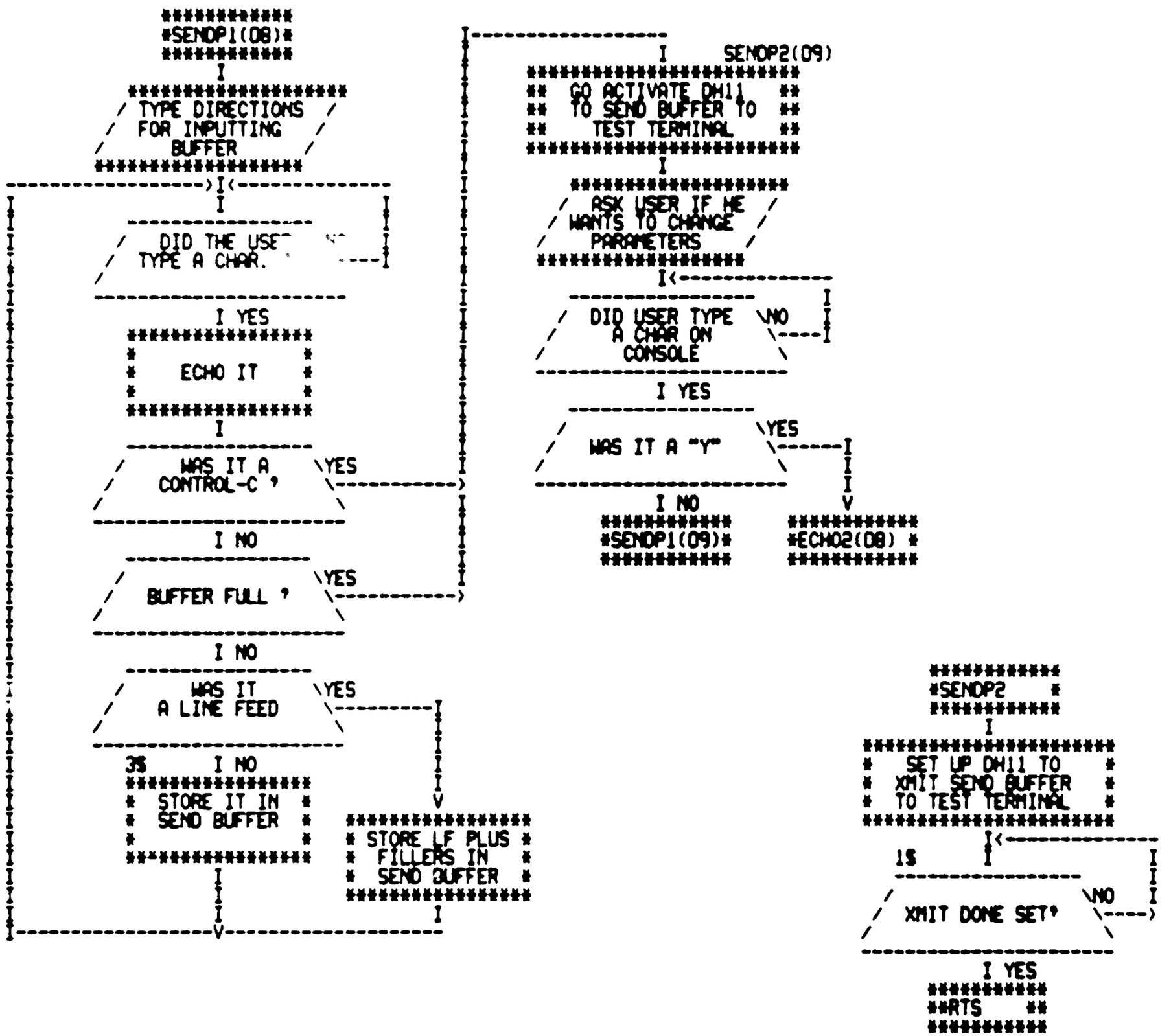

TINTI
WAS THERE A NON-EX MEM FLAG SET
YES
NO
SAVE CONTENTS OF "SCR" IN R3
2$
WAS THE XMIT DONE FLAG SET?
NO
YES
4$
WAS THE "BAR" BIT CLEARED?
NO
YES
6$
DID THE "BCR" REG GO TO 0?
NO
YES
8$
DID THE "CAR" REG GET INCREMENTED OK?
NO
YES
10$
UPDATE TOTAL NO. OF CHARS. XMITTED ON THIS LINE
SET THE XMIT DONE FLAG
RTI
GO SET UP ERROR INFORM-ATION,CLEAR DH
SERROR
GO REPORT NON-EX MEMORY ERROR EM1,DH1,DT1,DF2
1$
CORRECT SP, SET DROP-PED LINE FLAG FOR SELECTED LINE
NEWLIN(04)
GO SET UP ERROR INFORMATION AND CLEAR THE DH11
GO REPORT BYTE COUNT REG ERROR EM4,DH1,DT1,DF2
7$
CORRECT SP,SET DROP-PED LINE FLAG FOR THIS LINE
NEWLIN(04)
A
SET UP ERROR INFORMATION CLEAR DH11
GO REPORT FALSE XMITTER INTERRUPT EM2,DH1,DT1,DF2
3$
CORRECT SP,SET DROP-PED LINE FLAG FOR THIS LINE
C
GO SET UP ERROR INFO AND CLEAR DH11
GO REPORT CURRENT ADDR REG ERRORS EM5,DH1,DT1,DF2
9$
CORRECT SP,SET DROP-PED LINE FLAG FOR THIS LINE
B
GO SET UP ERROR INFORMATION AND CLEAR THE DH11
GO REPORT "BAR" REG FAILURE EM3,DH1,DT1,DF2
5$
CORRECT SP, SET DROP-PED LINE FLAG FOR THIS LINE

DO5

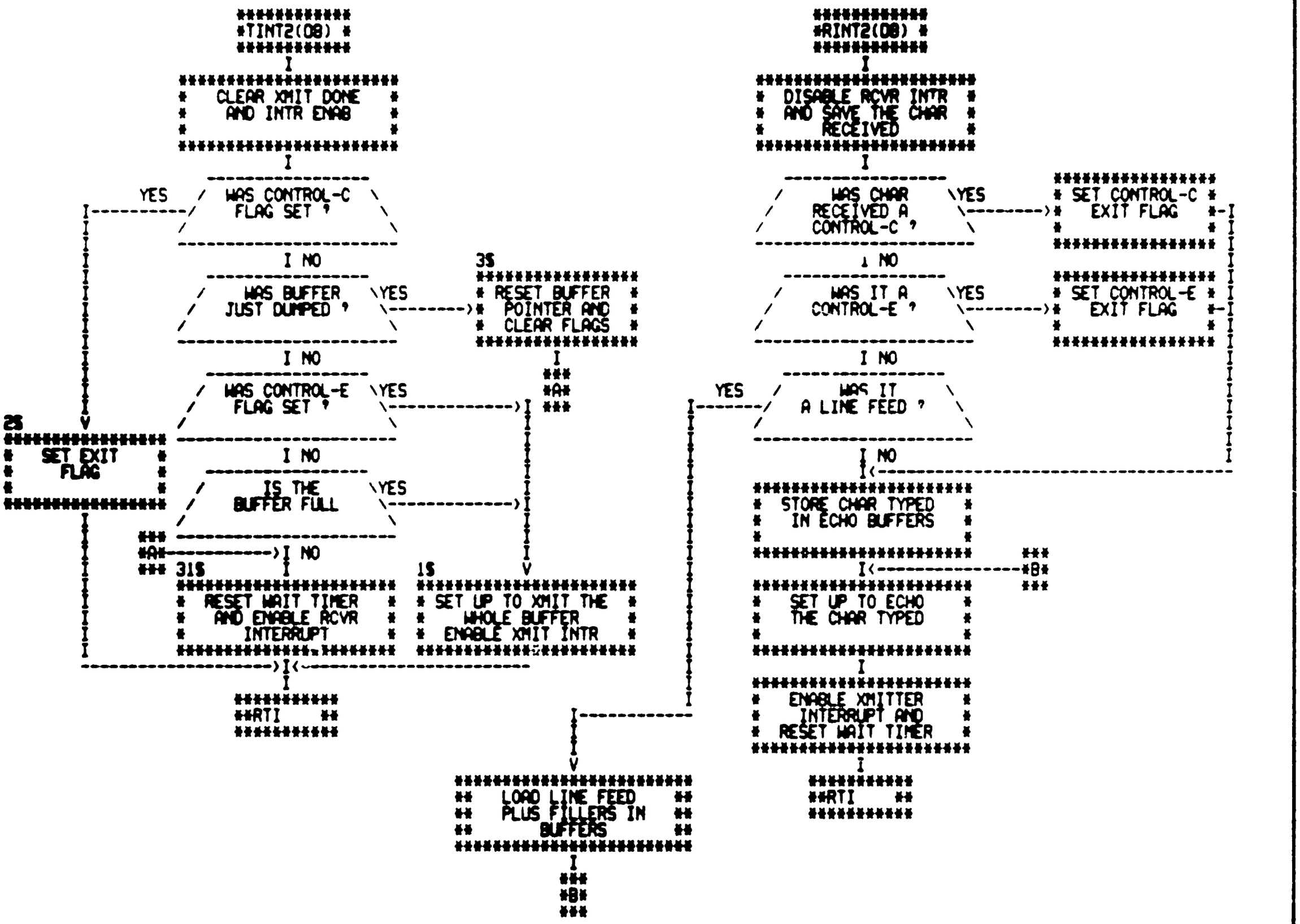

RINTI
SILO OVERFLOW ERROR ?
YES
NO
2$
CHAR AVAILABLE SET ?
NO
YES
A
SAVE THE REC- EIVED DATA IN STMPO
4$
VALID DATA READ ?
NO
YES
6$
STORE THE DATA IN THE CORE BUFFER AND UPDATE BUFF POINTER
END OF THE BUFFER ?
NO
YES
UPDATE TOTAL NO OF CHARACTERS RECEIVED ON THIS LINE
SET RECEIVER DONE FLAG
RTI
SET UP ERROR INFORMATION AND CLEAR THE DH11
SERROR
GO REPORT THE SILO OVERFLOW ERROR EM6,DH1,DT1,DF2
1$
CORRECT SP,SET DROP- PED FLAG FOR THIS LINE
NEWLIN(DH)
SET UP ERROR INFORMATION AND CLEAR THE DH11
SERROR
GO REPORT THE INVALID DATA ERROR EM10,DH1,DT1,DF2
5$
CORRECT SP,SET DROP- PED FLAG FOR THIS LINE
NEWLIN(DH)
A
SET UP ERROR INFORMATION AND CLEAR THE DH11
SERROR
GO REPORT THE RECEIVER FALSE INTR EM7,DH1,DT1,DF2
3$
CORRECT SP,SET DROP- PED LINE FLAG FOR THIS LINE

E05

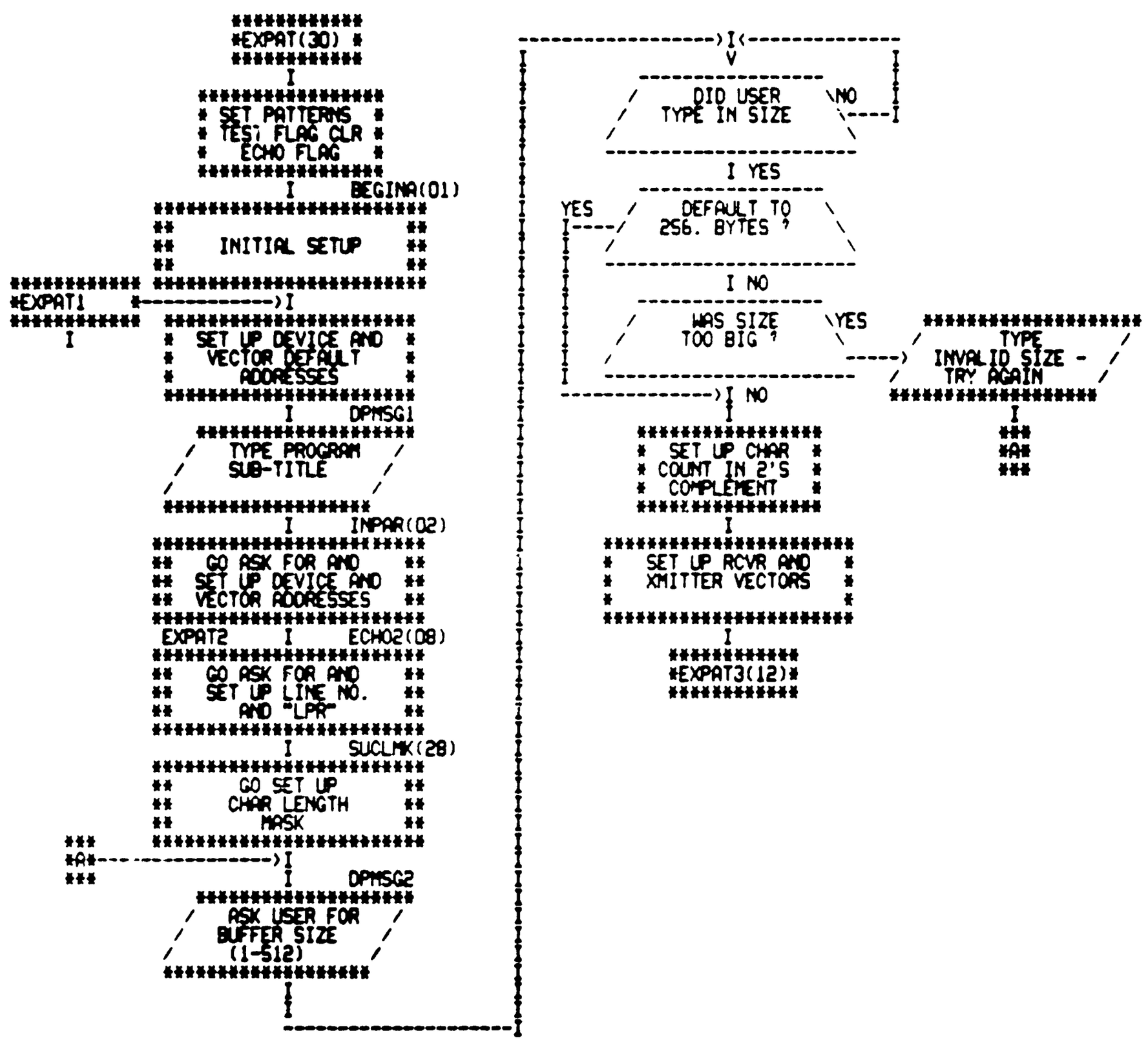

ECHO(30)
SET FLAG TO INDICATE ECHO TESTS
BEGINA(01)
INITIAL SET-UP
ECHO1
SET UP DEFAULT DEVICE AND VECTOR ADDRESSES
ECMSG1
TYPE SUB-PROGRAM TITLE
INPAR(02)
GO ASK FOR AND SET UP DEVICE AND VECTOR ADDRESSES
ECHO2(08)
ECMSG2
ASK USER TO TYPE LINE NO.
HAS USER TYPED LINE NO. YET
NO
YES
SET UP LINE NO. AND BIT SELECT MASK
LPRIN(31)
GO ASK FOR AND SET UP LINE PARAMETERS
SNMSG1
ASK USER IF HE WANTS TO SEND A BUFFER
SEND MODE ?
YES
NO
SENDP1(09)
ECHO3
SET UP AND ACTIVATE DH11 TO ECHO CHARS TO TEST TERMINAL
DHWAIT
RECEIVER INTERRUPT ?
YES
NO
TRANSMITTER INTERRUPT ?
YES
NO
CONTROL -C EXIT FLAG SET
YES
NO
ECHO2(08)
INTERRUPT WAIT TIMEOUT ?
NO
YES
SERROR
SAVE ERROR INFORMATION AND CLEAR DH11
REPORT INTERRUPT WAIT TIMEOUT EM16,DM5,DT6,DF2
ECHO2(08)
RINT2(10)
TINT2(10)
ECHO2(08)

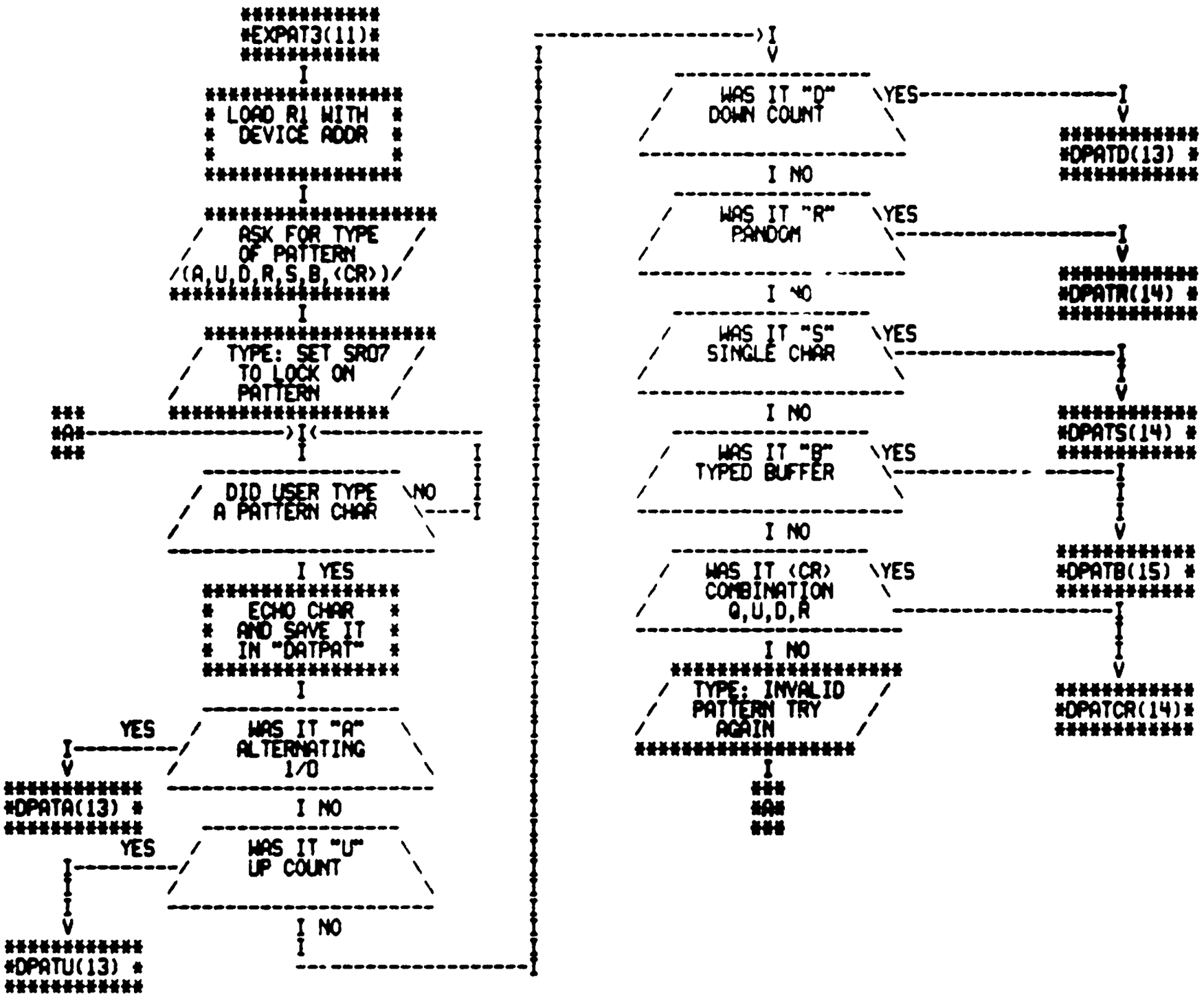

SENDP1(08)
TYPE DIRECTIONS FOR INPUTTING BUFFER
DID THE USER TYPE A CHAR.
YES
ECHO IT
WAS IT A CONTROL-C ?
NO
YES
BUFFER FULL ?
NO
YES
WAS IT A LINE FEED
YES
NO
STORE IT IN SEND BUFFER
STORE LF PLUS FILLERS IN SEND BUFFER
SENDP2(09)
GO ACTIVATE DH11 TO SEND BUFFER TO TEST TERMINAL
ASK USER IF HE WANTS TO CHANGE PARAMETERS
DID USER TYPE A CHAR ON CONSOLE
NO
YES
WAS IT A "Y"
YES
NO
SENDP1(09)
ECHO2(08)
SENDP2
SET UP DH11 TO XMIT SEND BUFFER TO TEST TERMINAL
XMIT DONE SET?
NO
YES
RTS

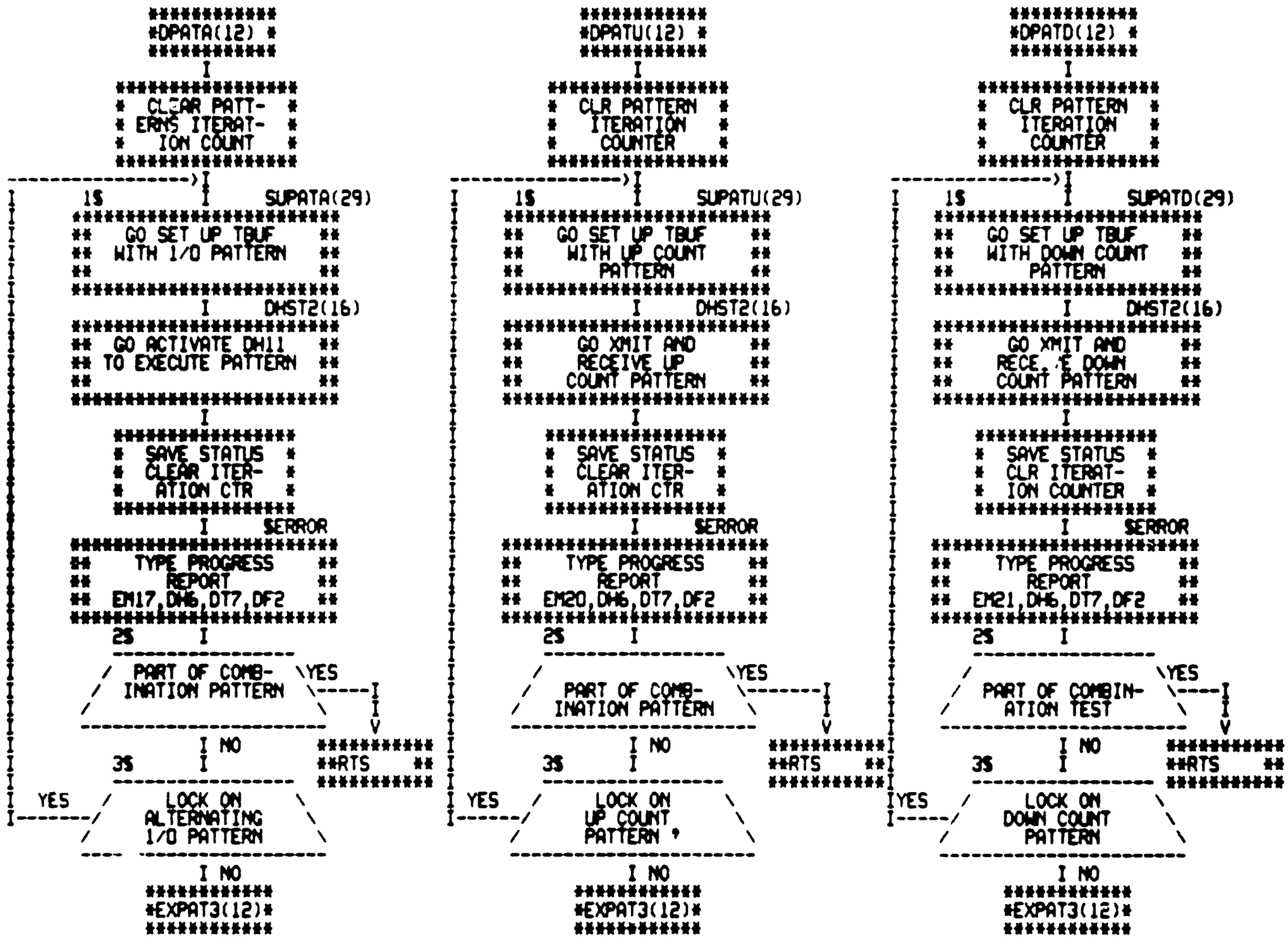
G05

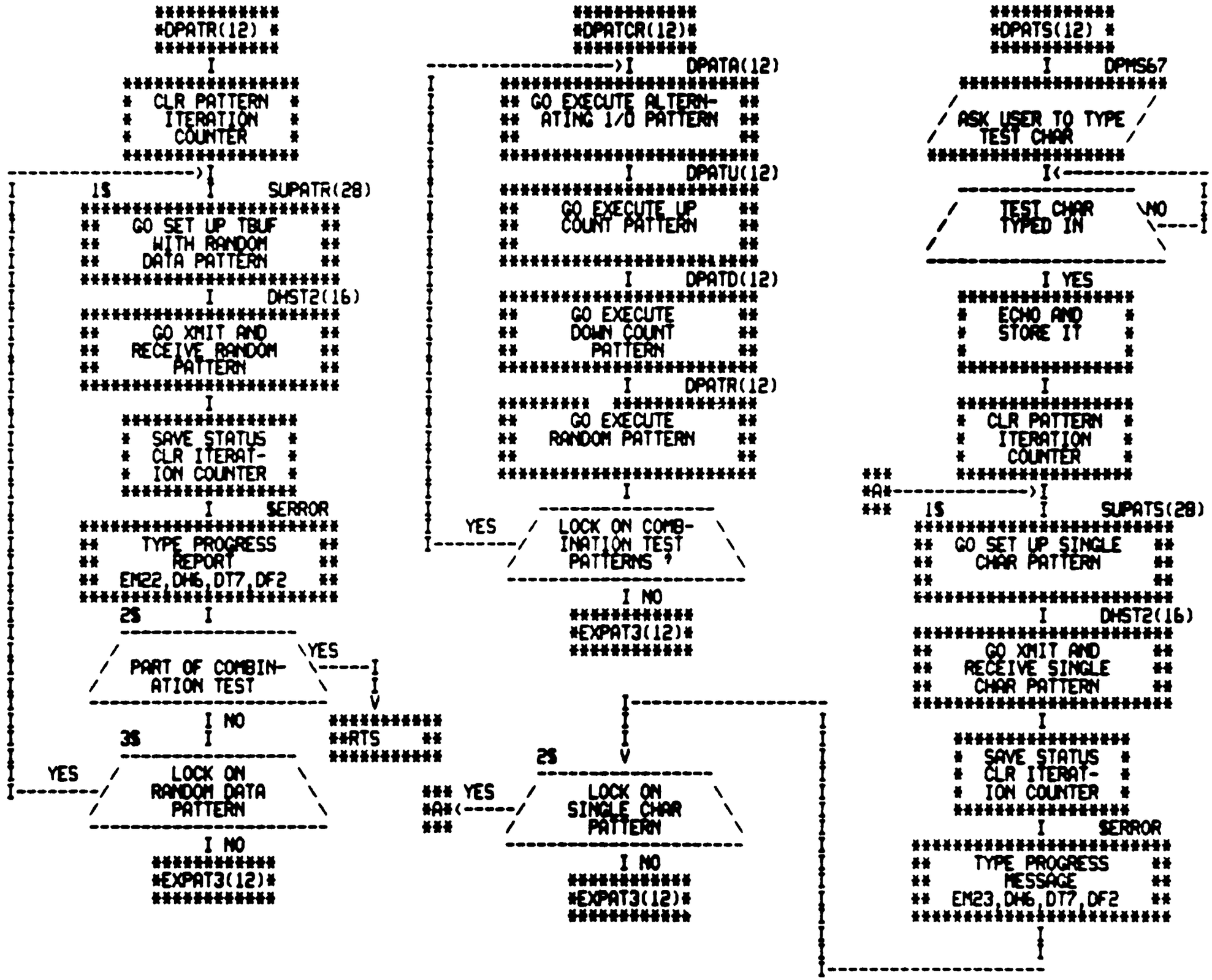

TINT2(08)
CLEAR XMIT DONE AND INTR ENAB
WAS CONTROL-C FLAG SET ?
YES
NO
WAS BUFFER JUST DUMPED ?
YES
NO
3$
RESET BUFFER POINTER AND CLEAR FLAGS
WAS CONTROL-E FLAG SET ?
YES
NO
2$
SET EXIT FLAG
IS THE BUFFER FULL
YES
NO
3$
RESET WAIT TIMER AND ENABLE RCVR INTERRUPT
1$
SET UP TO XMIT THE WHOLE BUFFER ENABLE XMIT INTR
RTI
LOAD LINE FEED PLUS FILLERS IN BUFFERS
RINT2(08)
DISABLE RCVR INTR AND SAVE THE CHAR RECEIVED
WAS CHAR RECEIVED A CONTROL-C ?
YES
SET CONTROL-C EXIT FLAG
NO
WAS IT A CONTROL-E ?
YES
SET CONTROL-E EXIT FLAG
NO
WAS IT A LINE FEED ?
YES
NO
STORE CHAR TYPED IN ECHO BUFFERS
SET UP TO ECHO THE CHAR TYPED
ENABLE XMITTER INTERRUPT AND RESET WAIT TIMER
RTI

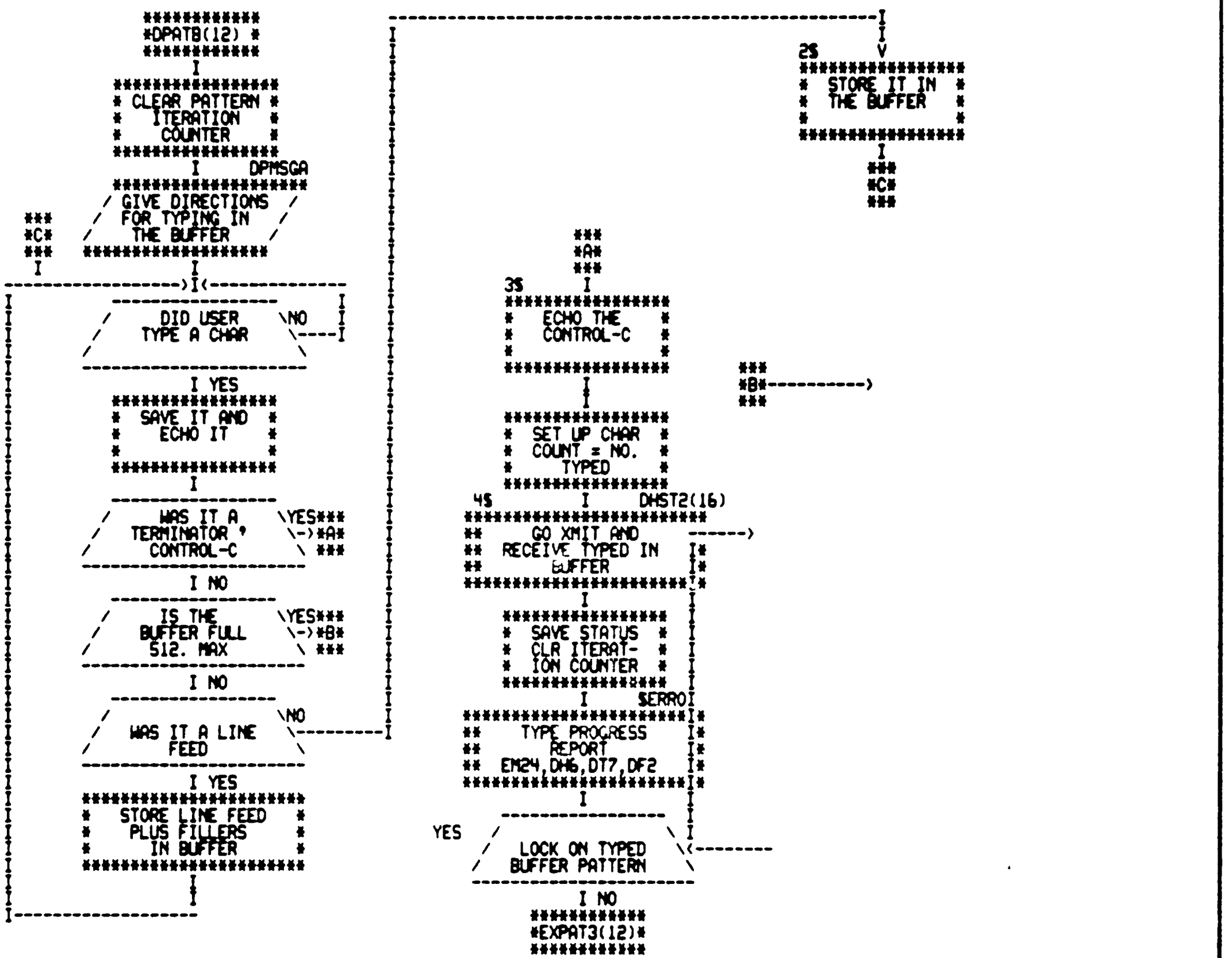
H05

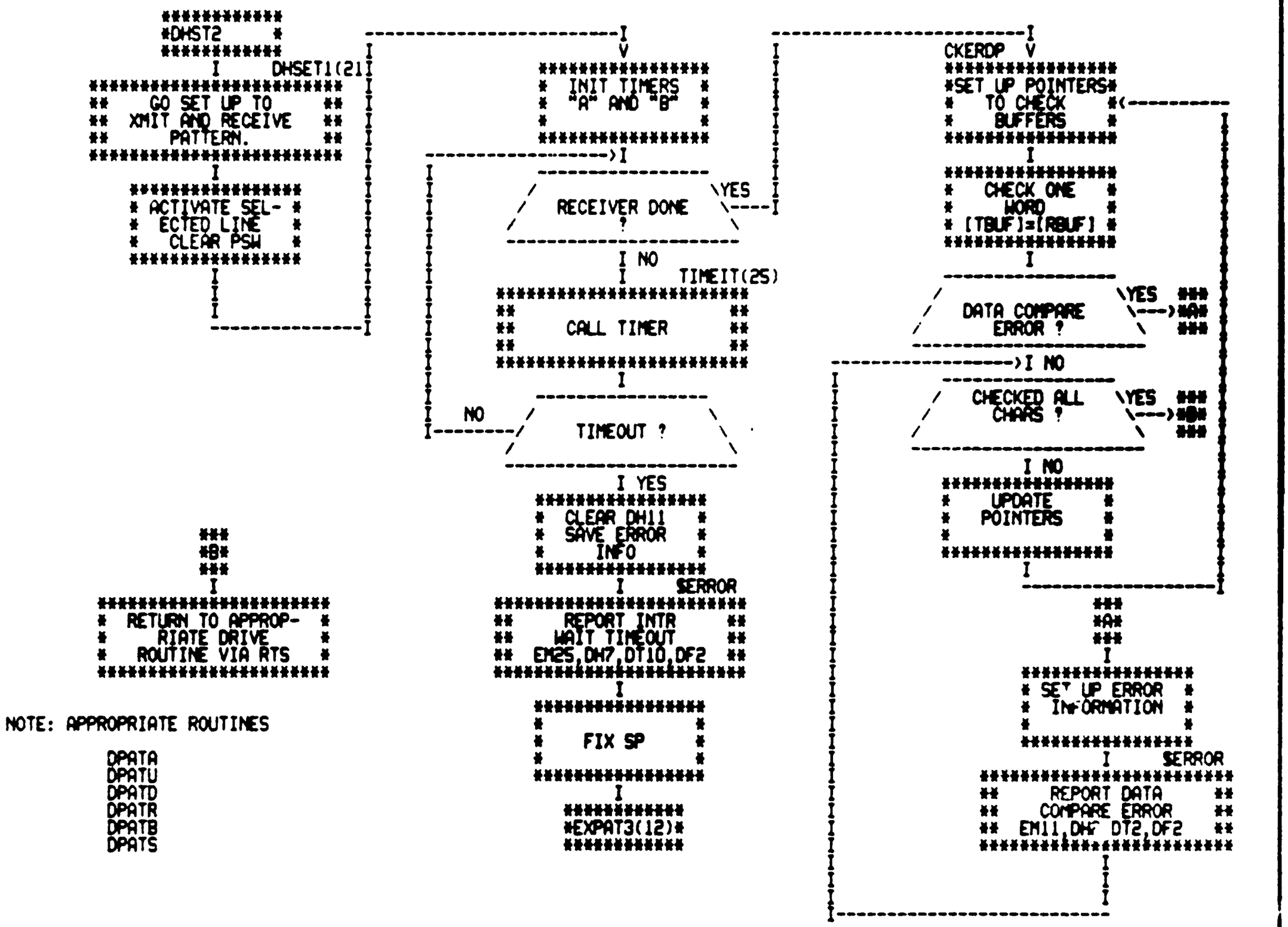

*EXPAT(30) *
SET PATTERNS
TEST FLAG CLR
ECHO FLAG
BEGINA(01)
INITIAL SETUP
*EXPAT1
SET UP DEVICE AND
VECTOR DEFAULT
ADDRESSES
DPMSG1
TYPE PROGRAM
SUB-TITLE
INPAR(02)
GO ASK FOR AND
SET UP DEVICE AND
VECTOR ADDRESSES
EXPAT2
ECHO2(08)
GO ASK FOR AND
SET UP LINE NO.
AND "LPR"
SUCLMK(28)
GO SET UP
CHAR LENGTH
MASK
DPMSG2
ASK USER FOR
BUFFER SIZE
(1-512)
DID USER
TYPE IN SIZE
NO
YES
DEFAULT TO
256. BYTES ?
WAS SIZE
TOO BIG ?
TYPE
INVALID SIZE -
TRY AGAIN
SET UP CHAR
COUNT IN 2'S
COMPLEMENT
SET UP RCVR AND
XMITTER VECTORS
EXPAT3(12)


#EXPAT3(11)#
LOAD R1 WITH DEVICE ADDR
ASK FOR TYPE OF PATTERN
(A,U,D,R,S,B,<CR>)
TYPE: SET SR07 TO LOCK ON PATTERN
#A#
DID USER TYPE A PATTERN CHAR
NO
I YES
ECHO CHAR AND SAVE IT IN "DATPAT"
WAS IT "A" ALTERNATING 1/0
YES
#DPATA(13) #
WAS IT "U" UP COUNT
YES
#DPATU(13) #
I NO
WAS IT "D" DOWN COUNT
YES
#DPATD(13) #
I NO
WAS IT "R" RANDOM
YES
#DPATR(14) #
I NO
WAS IT "S" SINGLE CHAR
YES
#DPATS(14) #
I NO
WAS IT "B" TYPED BUFFER
YES
#DPATB(15) #
I NO
WAS IT <CR> COMBINATION Q,U,D,R
YES
#DPATCR(14)#
I NO
TYPE: INVALID PATTERN TRY AGAIN
#A#


*DPATA(12) *
* CLEAR PATT- *
* ERNS ITERAT- *
* ION COUNT *
1$
SUPATA(29)
** GO SET UP TBUF **
** WITH I/O PATTERN **
DHST2(16)
** GO ACTIVATE DH11 **
** TO EXECUTE PATTERN **
* SAVE STATUS *
* CLEAR ITER- *
* ATION CTR *
SERROR
** TYPE PROGRESS **
** REPORT **
** EM17,DH6,DT7,DF2 **
2$
PART OF COMB- INATION PATTERN
YES
I NO
3$
RTS
LOCK ON ALTERNATING I/O PATTERN
YES
I NO
EXPAT3(12)
*DPATU(12) *
* CLR PATTERN *
* ITERATION *
* COUNTER *
1$
SUPATU(29)
** GO SET UP TBUF **
** WITH UP COUNT **
** PATTERN **
DHST2(16)
** GO XMIT AND **
** RECEIVE UP **
** COUNT PATTERN **
* SAVE STATUS *
* CLEAR ITER- *
* ATION CTR *
SERROR
** TYPE PROGRESS **
** REPORT **
** EM20,DH6,DT7,DF2 **
2$
PART OF COMB- INATION PATTERN
YES
I NO
3$
RTS
LOCK ON UP COUNT PATTERN ?
YES
I NO
EXPAT3(12)
*DPATD(12) *
* CLR PATTERN *
* ITERATION *
* COUNTER *
1$
SUPATD(29)
** GO SET UP TBUF **
** WITH DOWN COUNT **
** PATTERN **
DHST2(16)
** GO XMIT AND **
** RECEIVE DOWN **
** COUNT PATTERN **
* SAVE STATUS *
* CLR ITERAT- *
* ION COUNTER *
SERROR
** TYPE PROGRESS **
** REPORT **
** EM21,DH6,DT7,DF2 **
2$
PART OF COMBIN- ATION TEST
YES
I NO
3$
RTS
LOCK ON DOWN COUNT PATTERN
YES
I NO
EXPAT3(12)

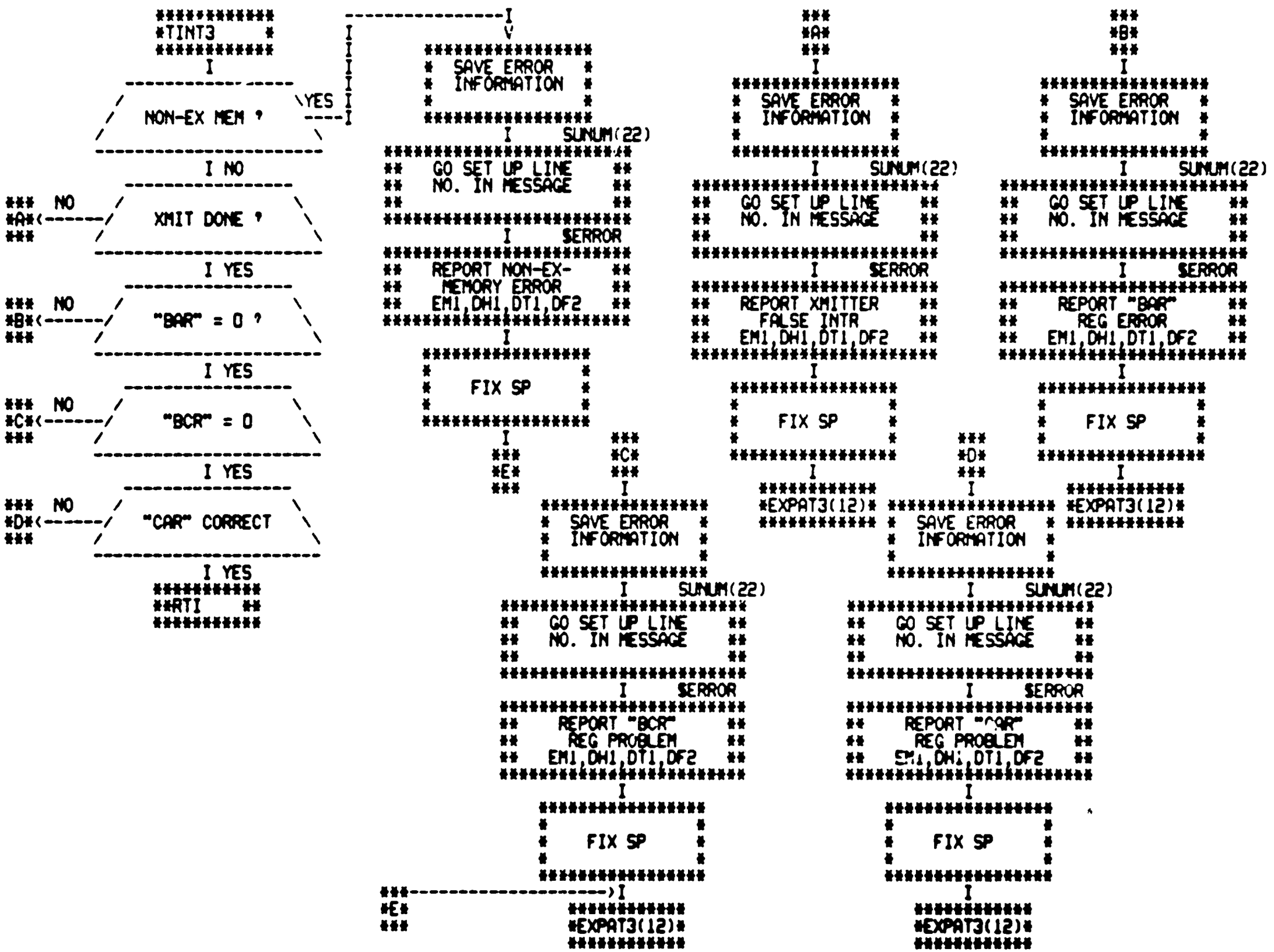

DPATR(12)
CLR PATTERN ITERATION COUNTER
1$
SUPATR(28)
GO SET UP TBUF WITH RANDOM DATA PATTERN
DHST2(16)
GO XMIT AND RECEIVE RANDOM PATTERN
SAVE STATUS CLR ITERAT- ION COUNTER
SERROR
TYPE PROGRESS REPORT EM22,DH6,DT7,DF2
2$
PART OF COMBIN- ATION TEST
YES
NO
3$
LOCK ON RANDOM DATA PATTERN
EXPAT3(12)
DPATCR(12)
DPATA(12)
GO EXECUTE ALTERN- ATING 1/0 PATTERN
DPATU(12)
GO EXECUTE UP COUNT PATTERN
DPATD(12)
GO EXECUTE DOWN COUNT PATTERN
DPATR(12)
GO EXECUTE RANDOM PATTERN
LOCK ON COMB- INATION TEST PATTERNS ?
EXPAT3(12)
RTS
LOCK ON SINGLE CHAR PATTERN
DPATS(12)
DPMS67
ASK USER TO TYPE TEST CHAR
TEST CHAR TYPED IN
YES
NO
ECHO AND STORE IT
CLR PATTERN ITERATION COUNTER
SUPATS(28)
GO SET UP SINGLE CHAR PATTERN
DHST2(16)
GO XMIT AND RECEIVE SINGLE CHAR PATTERN
SAVE STATUS CLR ITERAT- ION COUNTER
SERROR
TYPE PROGRESS MESSAGE EM23,DH6,DT7,DF2

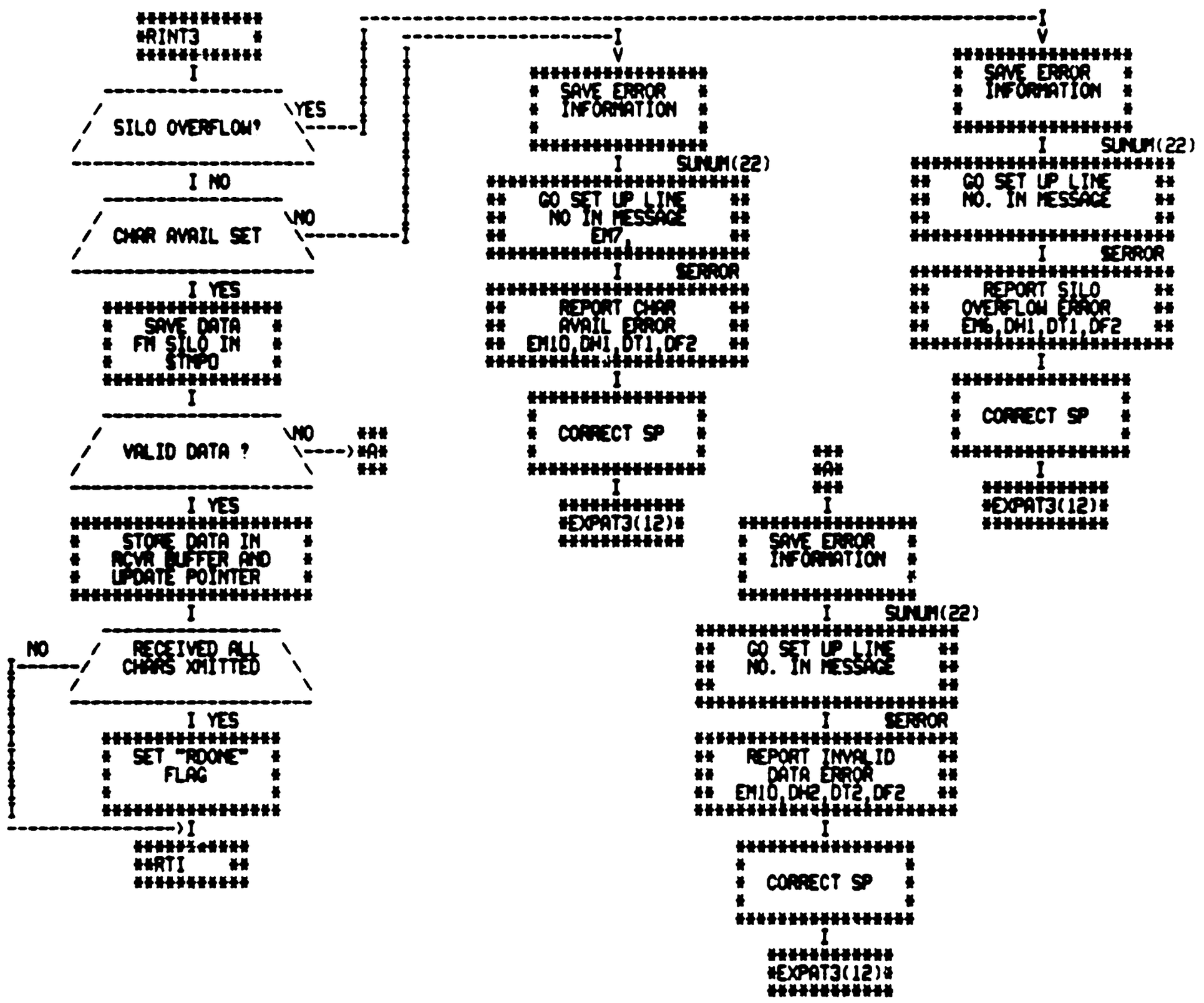

*DPATB(12) *
CLEAR PATTERN ITERATION COUNTER
DPMSGA
GIVE DIRECTIONS FOR TYPING IN THE BUFFER
C
DID USER TYPE A CHAR
NO
YES
SAVE IT AND ECHO IT
WAS IT A TERMINATOR ? CONTROL-C
YES
A
NO
IS THE BUFFER FULL 512. MAX
YES
B
NO
WAS IT A LINE FEED
YES
STORE LINE FEED PLUS FILLERS IN BUFFER
C
STORE IT IN THE BUFFER
A
3$
ECHO THE CONTROL-C
B
SET UP CHAR COUNT = NO. TYPED
4$
DHST2(16)
GO XMIT AND RECEIVE TYPED IN BUFFER
SAVE STATUS CLR ITERAT- ION COUNTER
$ERRO
TYPE PROGRESS REPORT
EM24,DH6,DT7,DF2
YES
LOCK ON TYPED BUFFER PATTERN
NO
EXPAT3(12)

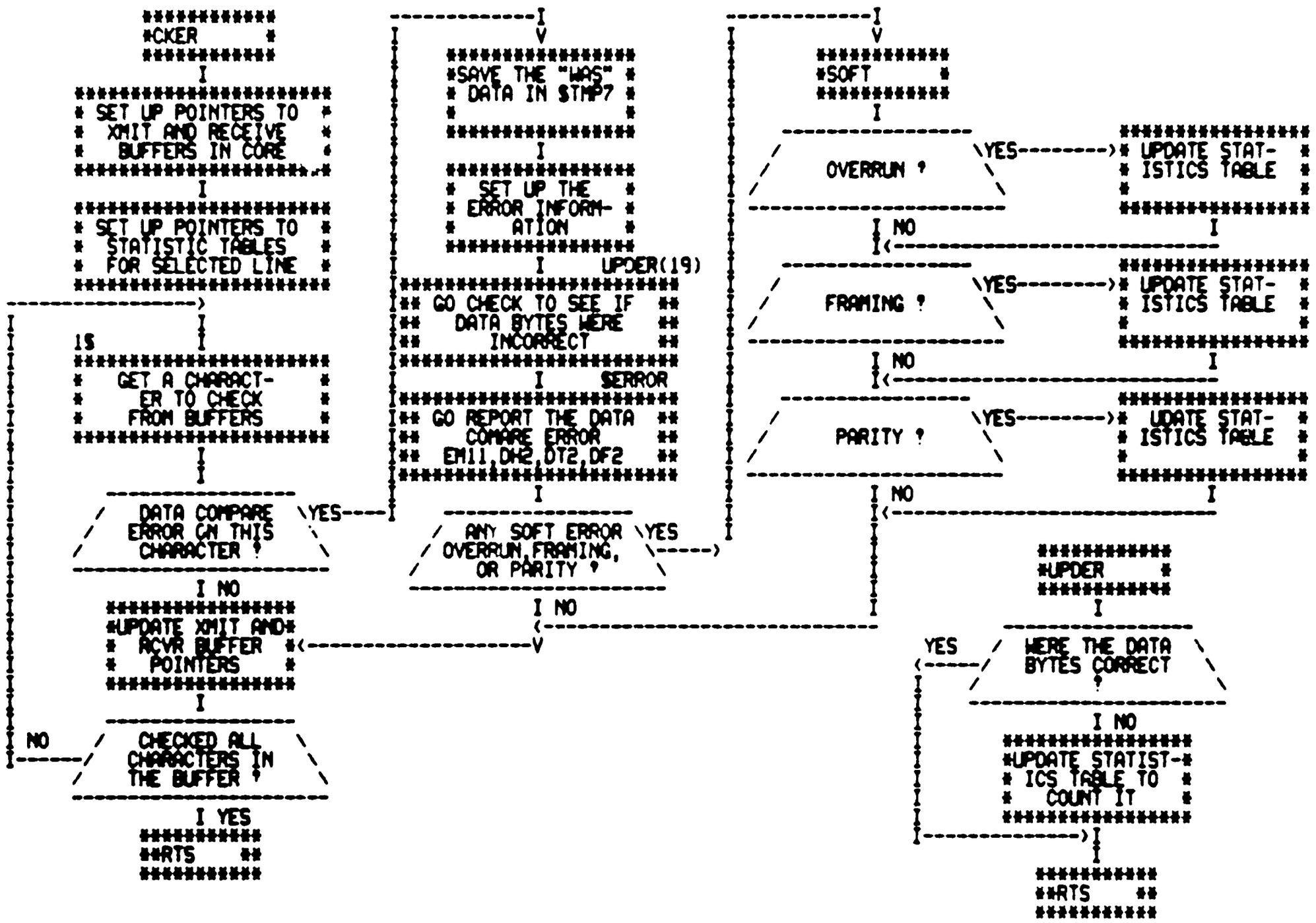


NOTE: APPROPRIATE ROUTINES

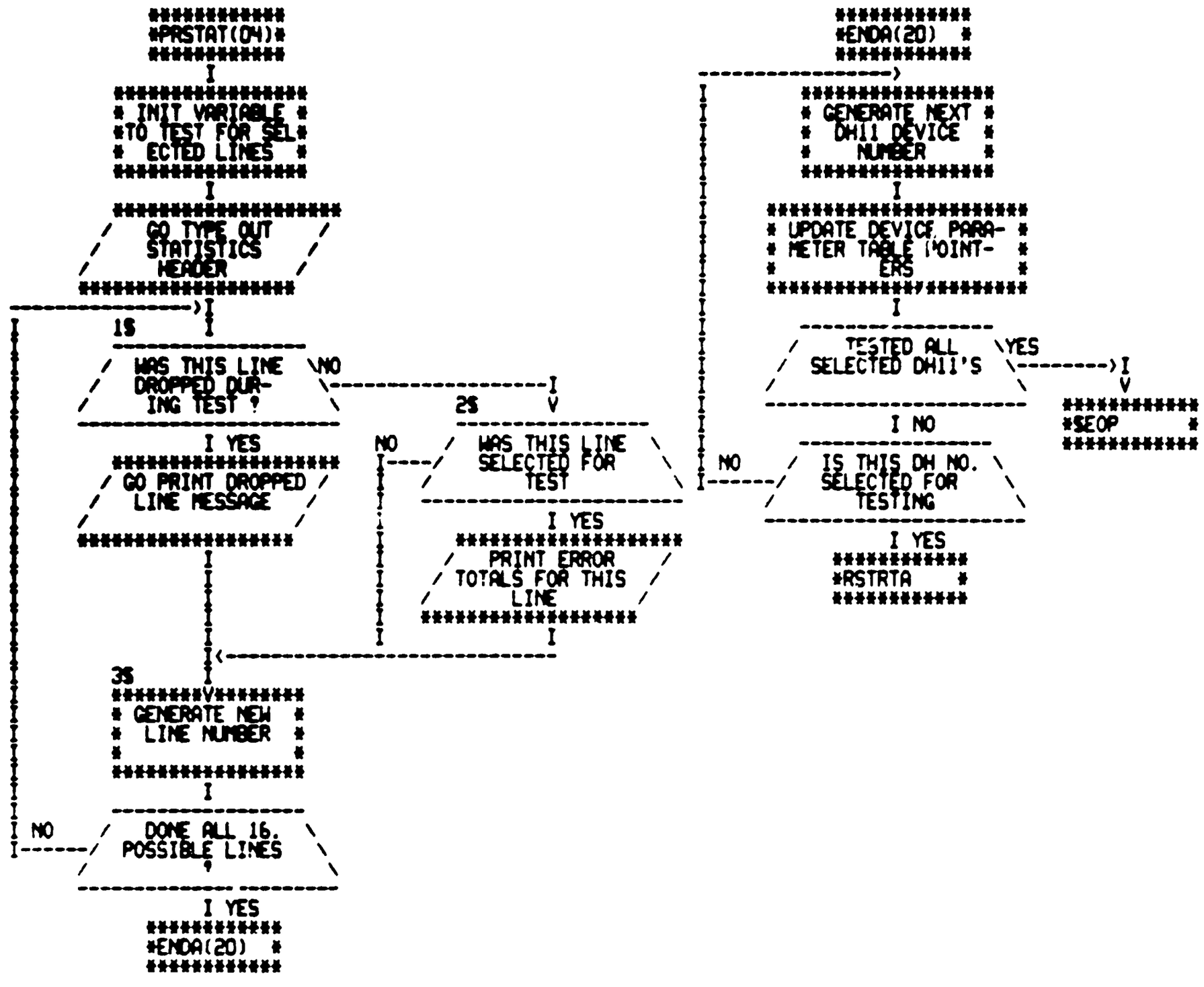
DPATA
DPATU
DPATD
DPATR
DPATB
DPATS

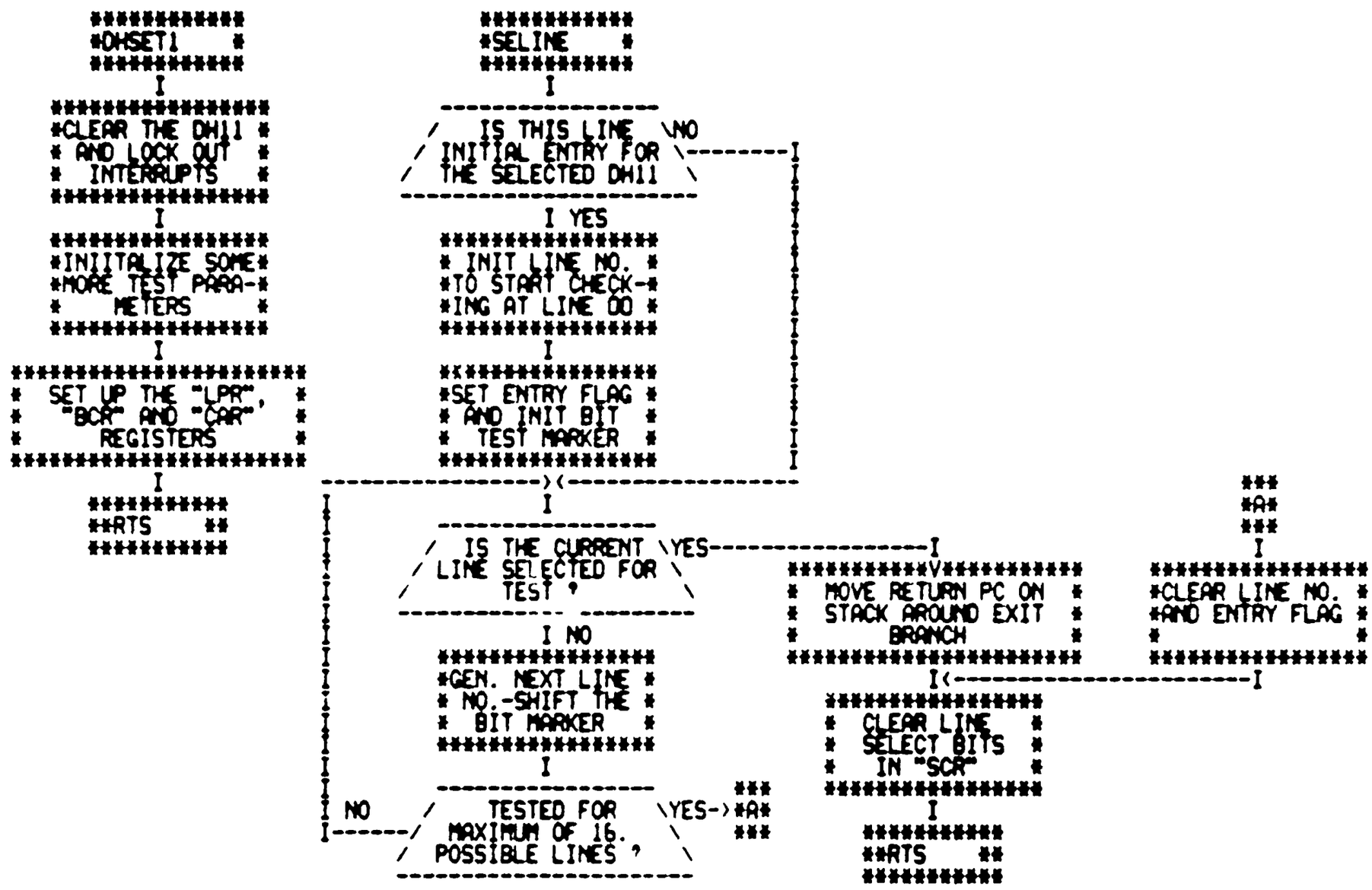

TINT3
NON-EX MEM ?
YES
NO
SAVE ERROR INFORMATION
SUNUM(22)
GO SET UP LINE NO. IN MESSAGE
$ERROR
REPORT NON-EX-MEMORY ERROR EM1,DH1,DT1,DF2
FIX SP
A
XMIT DONE ?
B
"BAR" = 0 ?
REPORT XMITTER FALSE INTR EM1,DH1,DT1,DF2
REPORT "BAR" REG ERROR EM1,DH1,DT1,DF2
C
"BCR" = 0
REPORT "BCR" REG PROBLEM EM1,DH1,DT1,DF2
D
"CAR" CORRECT
REPORT "CAR" REG PROBLEM EM1,DH1,DT1,DF2
E
EXPAT3(12)
RTI

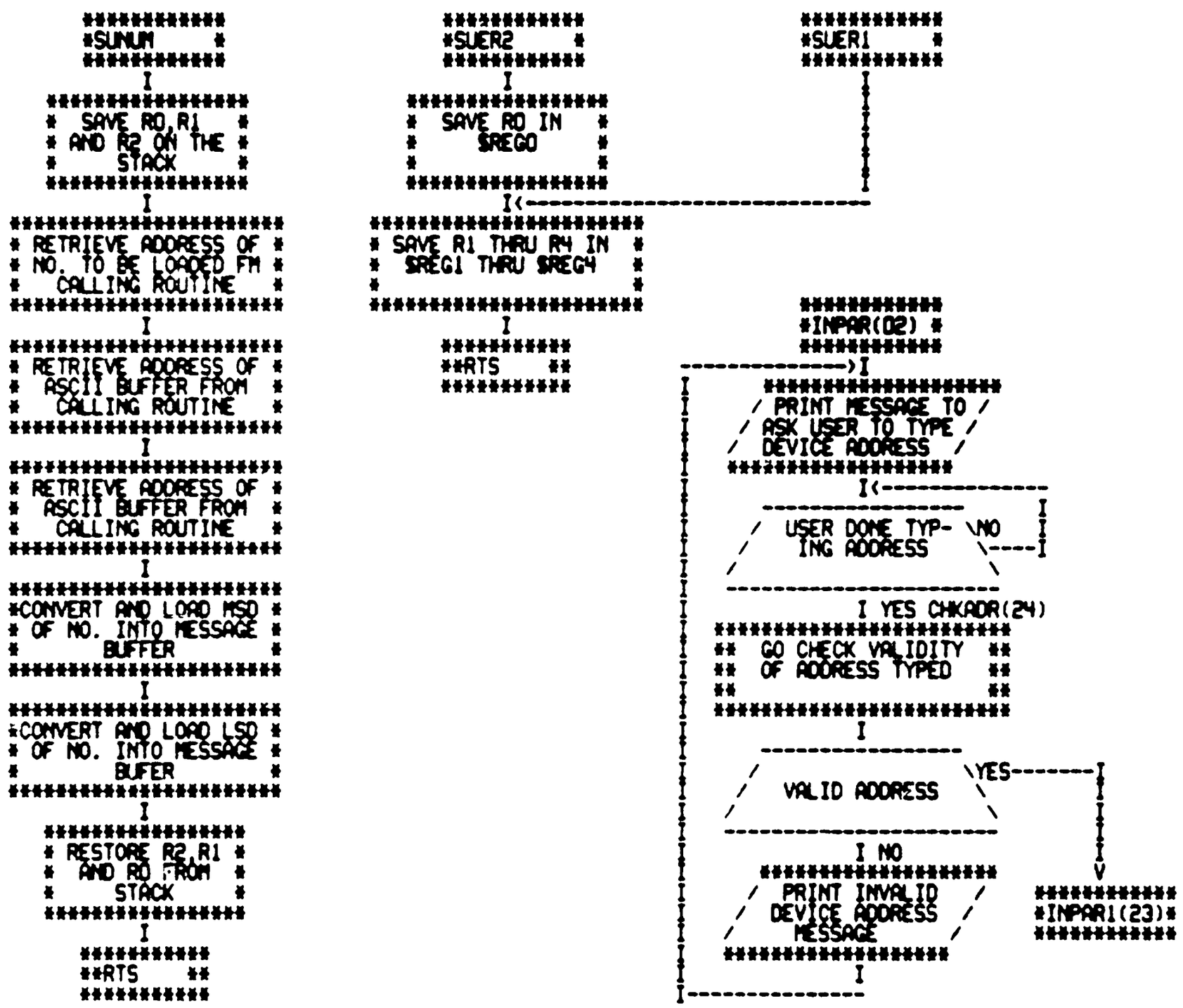

#RINT3
SILO OVERFLOW?
YES
NO
CHAR AVAIL SET
NO
YES
SAVE DATA FM SILO IN $TMPO
VALID DATA ?
NO
#A#
YES
STORE DATA IN RCVR BUFFER AND UPDATE POINTER
RECEIVED ALL CHARS XMITTED
NO
YES
SET "RDONE" FLAG
#RTI
SAVE ERROR INFORMATION
SUNUM(22)
GO SET UP LINE NO IN MESSAGE
EM7
SERROR
REPORT CHAR AVAIL ERROR
EM10,DH1,DT1,DF2
CORRECT SP
#EXPAT3(12)#
SAVE ERROR INFORMATION
SUNUM(22)
GO SET UP LINE NO. IN MESSAGE
SERROR
REPORT SILO OVERFLOW ERROR
EM6,DH1,DT1,DF2
CORRECT SP
#EXPAT3(12)#
SAVE ERROR INFORMATION
SUNUM(22)
GO SET UP LINE NO. IN MESSAGE
SERROR
REPORT INVALID DATA ERROR
EM10,DH2,DT2,DF2
CORRECT SP
#EXPAT3(12)#

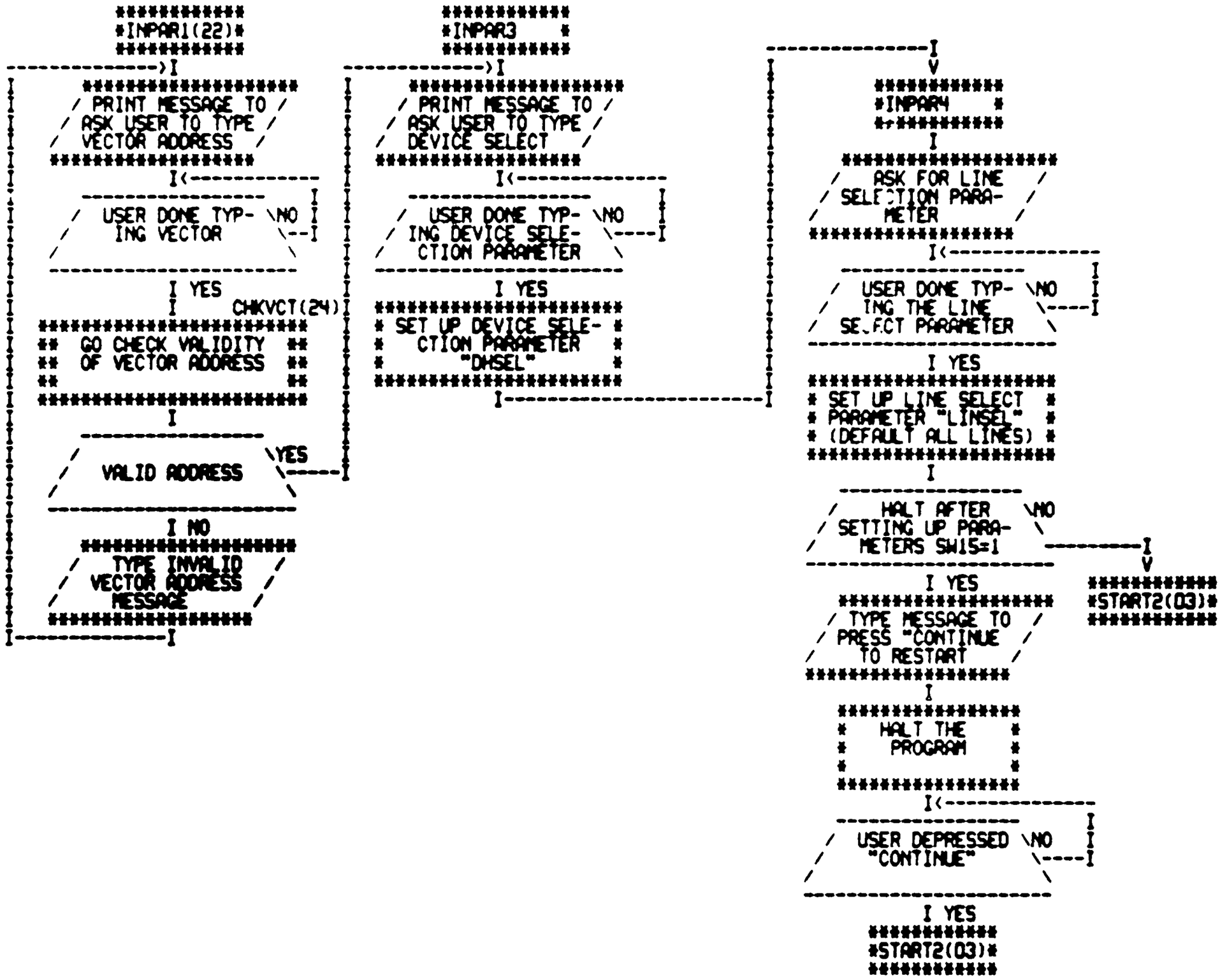

#CKER
SET UP POINTERS TO XMIT AND RECEIVE BUFFERS IN CORE
SET UP POINTERS TO STATISTIC TABLES FOR SELECTED LINE
18
GET A CHARACTER TO CHECK FROM BUFFERS
DATA COMPARE ERROR ON THIS CHARACTER ?
YES
NO
#UPDATE XMIT AND RCVR BUFFER POINTERS
CHECKED ALL CHARACTERS IN THE BUFFER ?
NO
YES
##RTS
SAVE THE "WAS" DATA IN $TMP7
SET UP THE ERROR INFORM- ATION
UPDER(19)
GO CHECK TO SEE IF DATA BYTES WERE INCORRECT
SERROR
GO REPORT THE DATA COMPARE ERROR EM11,DH2,DT2,DF2
ANY SOFT ERROR OVERRUN,FRAMING, OR PARITY ?
YES
NO
#SOFT
OVERRUN ?
YES
UPDATE STAT- ISTICS TABLE
FRAMING ?
YES
UPDATE STAT- ISTICS TABLE
PARITY ?
YES
UPDATE STAT- ISTICS TABLE
NO
#UPDER
WERE THE DATA BYTES CORRECT ?
YES
NO
#UPDATE STATIST-ICS TABLE TO COUNT IT
##RTS

D06

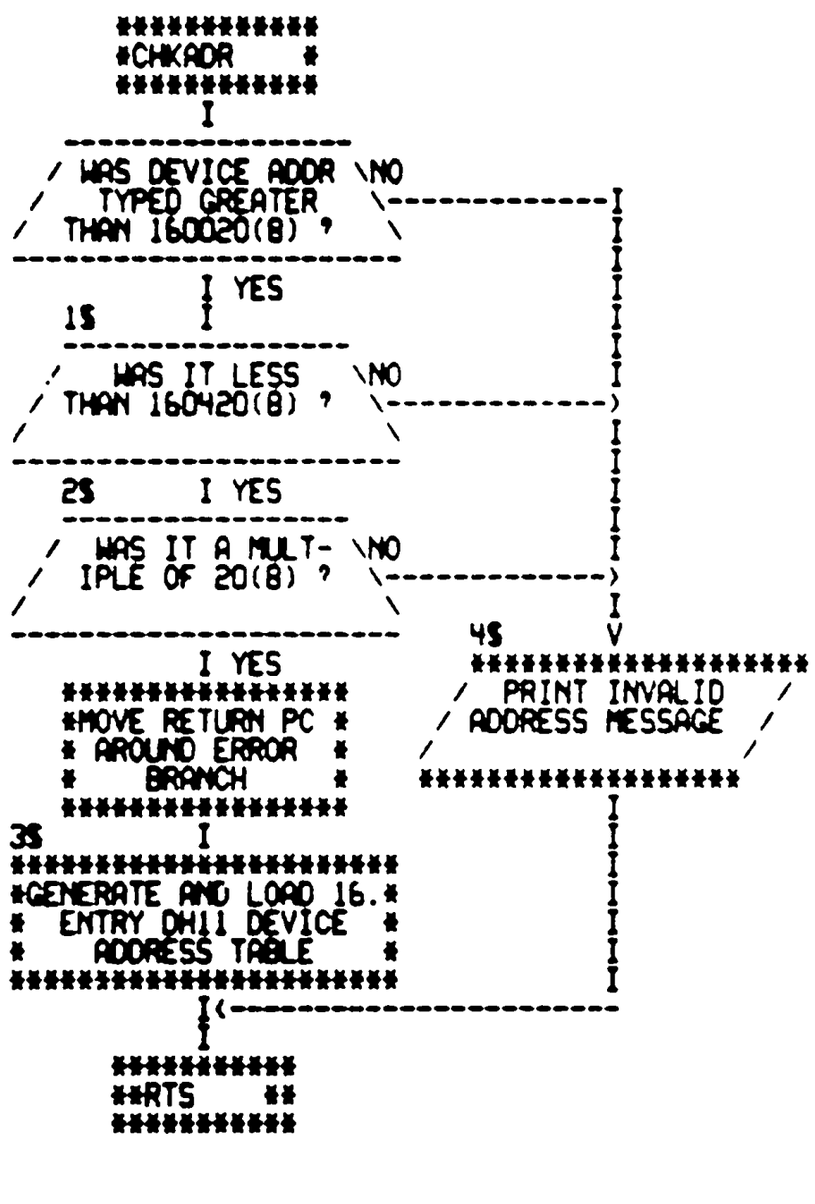

PRSTAT(04)
INIT VARIABLE *TO TEST FOR SELECTED LINES
GO TYPE OUT STATISTICS HEADER
1$
WAS THIS LINE DROPPED DURING TEST ?
NO
YES
GO PRINT DROPPED LINE MESSAGE
2$
WAS THIS LINE SELECTED FOR TEST
NO
YES
PRINT ERROR TOTALS FOR THIS LINE
3$
GENERATE NEW LINE NUMBER
DONE ALL 16. POSSIBLE LINES
NO
YES
*ENDA(20) *
*ENDA(20) *
GENERATE NEXT DH11 DEVICE NUMBER
UPDATE DEVICE PARAMETER TABLE POINTERS
TESTED ALL SELECTED DH11'S
YES
NO
*SEOP *
IS THIS DH NO. SELECTED FOR TESTING
NO
YES
*RSTRTA *

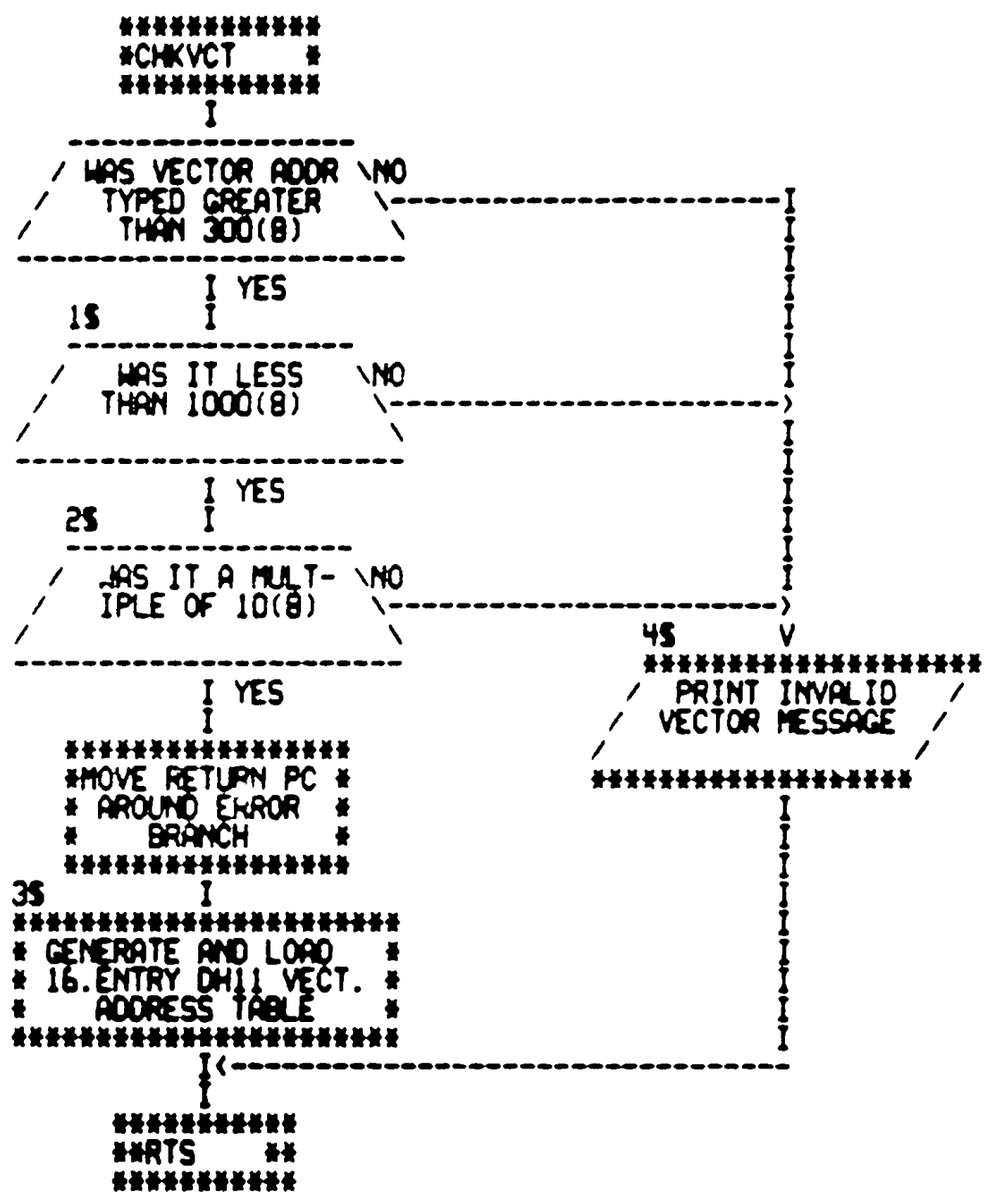

#OHSET1
#CLEAR THE DH11 AND LOCK OUT INTERRUPTS
#INITALIZE SOME MORE TEST PARA-METERS
SET UP THE "LPR", "BCR" AND "CAR" REGISTERS
#RTS
#SELINE
IS THIS LINE INITIAL ENTRY FOR THE SELECTED DH11
I YES
INIT LINE NO. TO START CHECK-ING AT LINE 00
#SET ENTRY FLAG AND INIT BIT TEST MARKER
IS THE CURRENT LINE SELECTED FOR TEST ?
I NO
#GEN. NEXT LINE NO.-SHIFT THE BIT MARKER
TESTED FOR MAXIMUM OF 16. POSSIBLE LINES ?
NO
YES
#A
MOVE RETURN PC ON STACK AROUND EXIT BRANCH
#CLEAR LINE NO. AND ENTRY FLAG
CLEAR LINE SELECT BITS IN "SCR"
#RTS

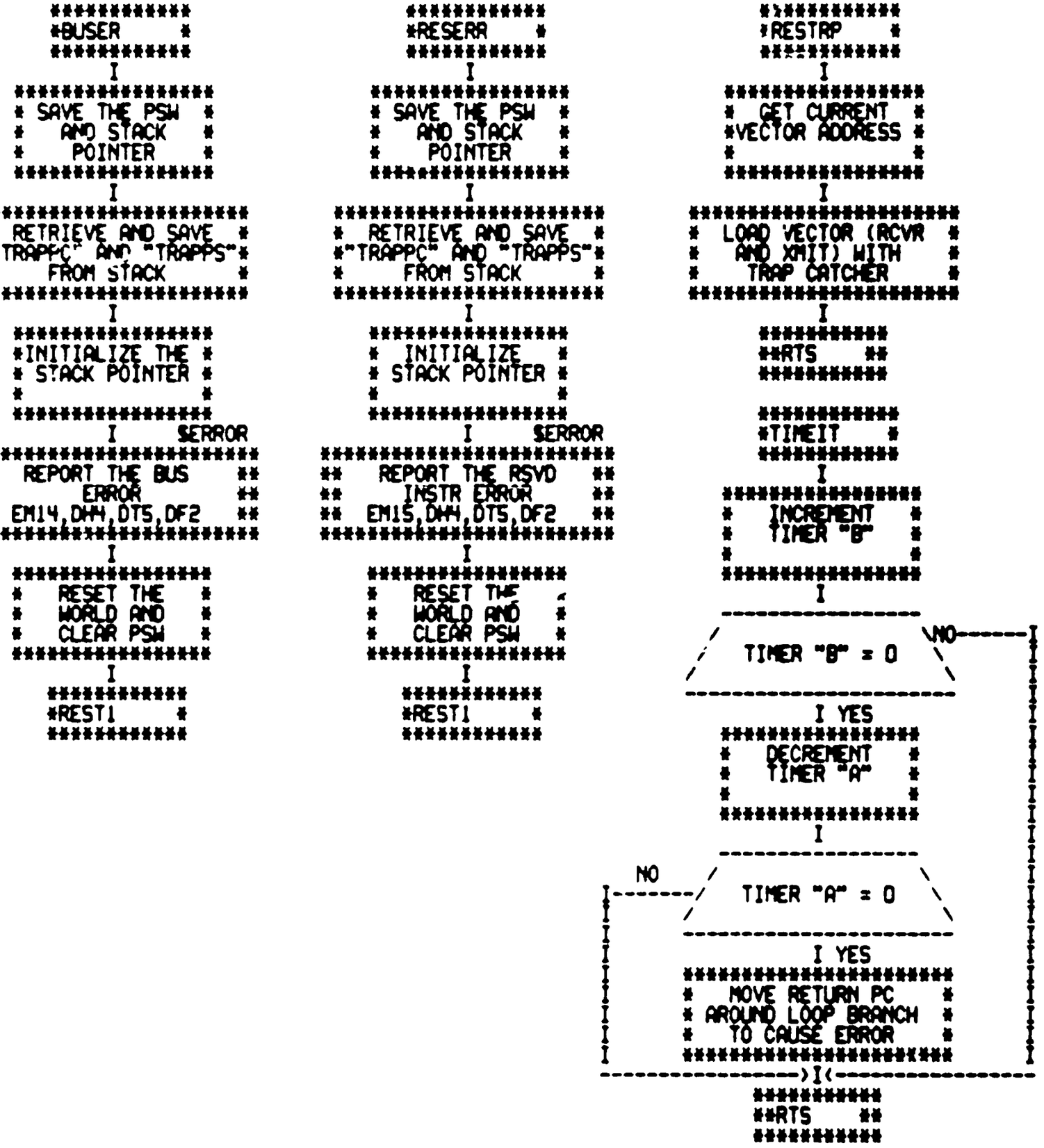

$UNUM
SAVE R0,R1 AND R2 ON THE STACK
RETRIEVE ADDRESS OF NO. TO BE LOADED FM CALLING ROUTINE
RETRIEVE ADDRESS OF ASCII BUFFER FROM CALLING ROUTINE
RETRIEVE ADDRESS OF ASCII BUFFER FROM CALLING ROUTINE
CONVERT AND LOAD MSD OF NO. INTO MESSAGE BUFFER
CONVERT AND LOAD LSD OF NO. INTO MESSAGE BUFER
RESTORE R2,R1 AND R0 FROM STACK
RTS
$UER2
SAVE R0 IN $REG0
SAVE R1 THRU R4 IN $REG1 THRU $REG4
RTS
$UER1
$INPAR(02)
PRINT MESSAGE TO ASK USER TO TYPE DEVICE ADDRESS
USER DONE TYP- ING ADDRESS
NO
YES CHKADR(24)
GO CHECK VALIDITY OF ADDRESS TYPED
VALID ADDRESS
YES
NO
PRINT INVALID DEVICE ADDRESS MESSAGE
$INPAR1(23)

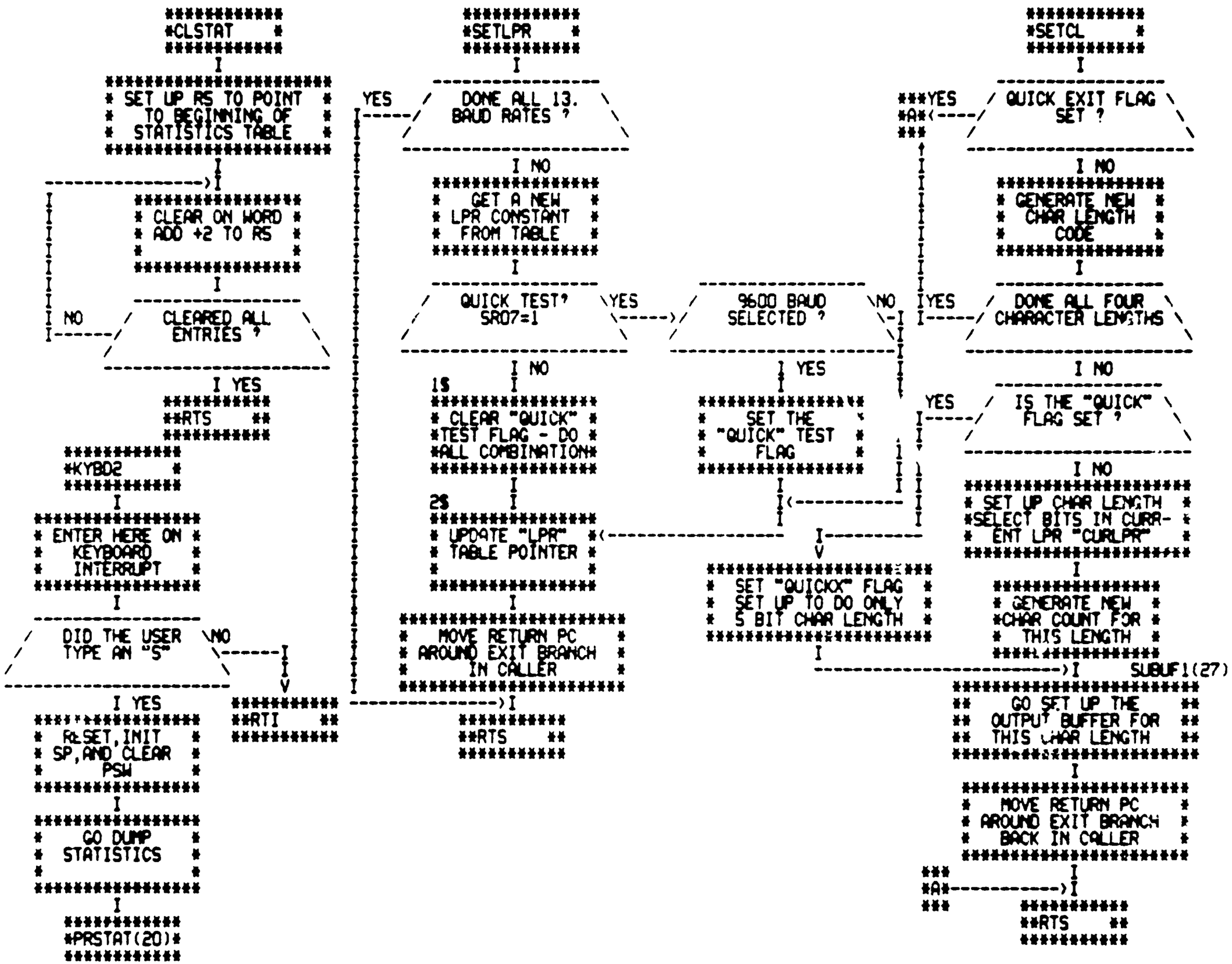

INPAR1(22)
PRINT MESSAGE TO ASK USER TO TYPE VECTOR ADDRESS
USER DONE TYP- ING VECTOR
NO
YES
CHKVCT(24)
GO CHECK VALIDITY OF VECTOR ADDRESS
VALID ADDRESS
YES
NO
TYPE INVALID VECTOR ADDRESS MESSAGE
*INPAR3
PRINT MESSAGE TO ASK USER TO TYPE DEVICE SELECT
USER DONE TYP- ING DEVICE SELE- CTION PARAMETER
NO
YES
SET UP DEVICE SELE- CTION PARAMETER "DHSEL"
*INPAR4
ASK FOR LINE SELECTION PARA- METER
USER DONE TYP- ING THE LINE SELECT PARAMETER
NO
YES
SET UP LINE SELECT PARAMETER "LINSEL" (DEFAULT ALL LINES)
HALT AFTER SETTING UP PARA- METERS SW15=1
NO
YES
TYPE MESSAGE TO PRESS "CONTINUE TO RESTART
HALT THE PROGRAM
USER DEPRESSED "CONTINUE"
NO
YES
START2(03)
START2(03)

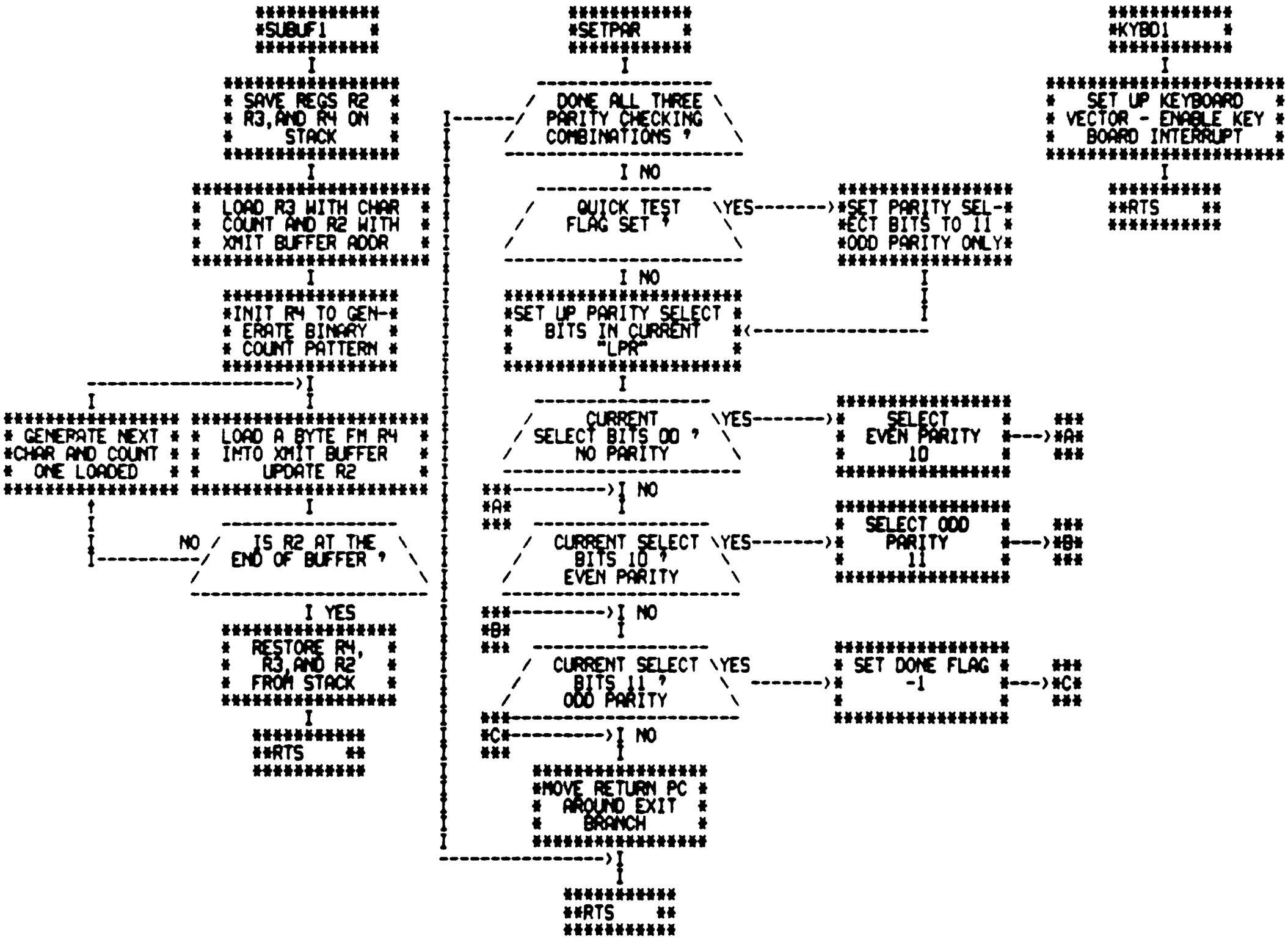
H06

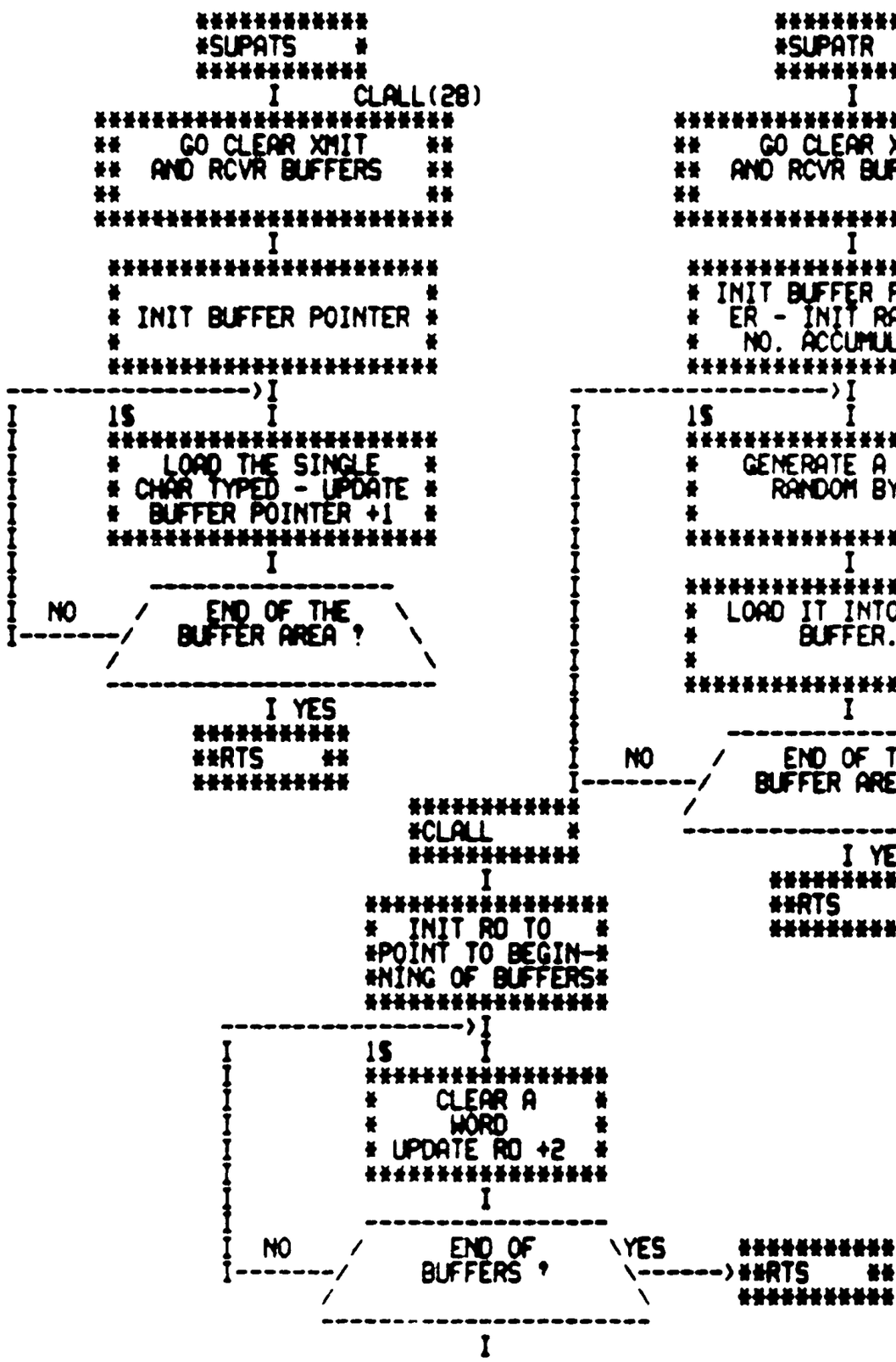

*CHKADR *
WAS DEVICE ADDR TYPED GREATER THAN 160020(8) ?
NO
YES
1$
WAS IT LESS THAN 160420(8) ?
NO
2$
YES
WAS IT A MULT- IPLE OF 20(8) ?
NO
YES
4$
PRINT INVALID ADDRESS MESSAGE
*MOVE RETURN PC * * AROUND ERROR * * BRANCH *
3$
GENERATE AND LOAD 16. * ENTRY DH11 DEVICE * * ADDRESS TABLE *
**RTS **


*CHKVCT *
WAS VECTOR ADDR TYPED GREATER THAN 300(8)
NO
YES
1$
WAS IT LESS THAN 1000(8)
NO
2$
YES
WAS IT A MULT- IPLE OF 10(8)
NO
YES
4$
PRINT INVALID VECTOR MESSAGE
*MOVE RETURN PC * * AROUND ERROR * * BRANCH *
3$
* GENERATE AND LOAD * * 16.ENTRY DH11 VECT. * * ADDRESS TABLE *
**RTS **

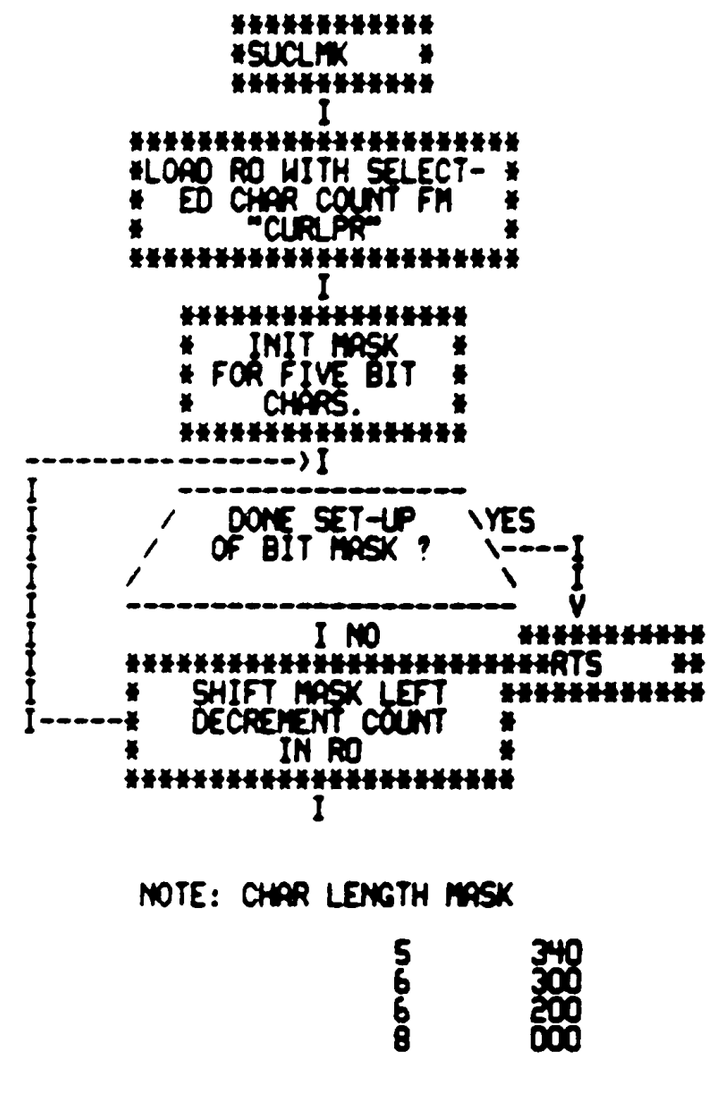

*BUSER *
SAVE THE PSW AND STACK POINTER
RETRIEVE AND SAVE "TRAPPC" AND "TRAPPS" FROM STACK
*INITIALIZE THE STACK POINTER *
$ERROR
REPORT THE BUS ERROR
EM14,DHH,DT5,DF2
RESET THE WORLD AND CLEAR PSW
*REST1 *
*RESERR *
SAVE THE PSW AND STACK POINTER
RETRIEVE AND SAVE "TRAPPC" AND "TRAPPS" FROM STACK
INITIALIZE STACK POINTER
$ERROR
REPORT THE RSVD INSTR ERROR
EM15,DHH,DT5,DF2
RESET THE WORLD AND CLEAR PSW
*REST1 *
*RESTRP *
GET CURRENT VECTOR ADDRESS
LOAD VECTOR (RCVR AND XMIT) WITH TRAP CATCHER
**RTS **
*TIMEIT *
INCREMENT TIMER "B"
TIMER "B" = 0
NO
I YES
DECREMENT TIMER "A"
TIMER "A" = 0
NO
I YES
MOVE RETURN PC AROUND LOOP BRANCH TO CAUSE ERROR
**RTS **

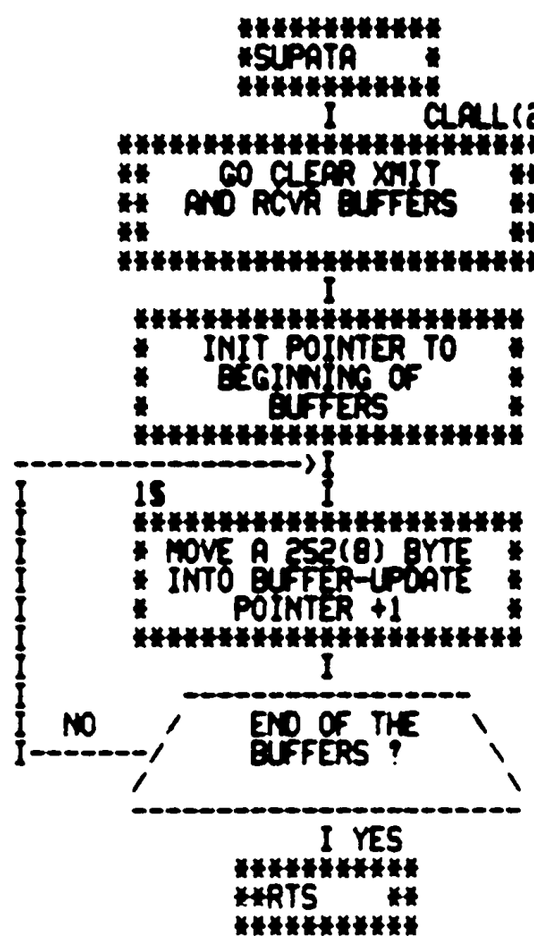

*CLSTAT
SET UP RS TO POINT TO BEGINNING OF STATISTICS TABLE
CLEAR ON WORD
ADD +2 TO RS
CLEARED ALL ENTRIES ?
NO
YES
*RTS
*KYBD2
ENTER HERE ON KEYBOARD INTERRUPT
DID THE USER TYPE AN "S"
NO
YES
RESET, INIT SP, AND CLEAR PSW
GO DUMP STATISTICS
*PRSTAT(20)
*RTI
*SETLPR
DONE ALL 13. BAUD RATES ?
YES
NO
GET A NEW LPR CONSTANT FROM TABLE
QUICK TEST? SR07=1
YES
NO
1$
CLEAR "QUICK" TEST FLAG - DO ALL COMBINATION
2$
UPDATE "LPR" TABLE POINTER
MOVE RETURN PC AROUND EXIT BRANCH IN CALLER
*RTS
9600 BAUD SELECTED ?
NO
YES
SET THE "QUICK" TEST FLAG
SET "QUICKX" FLAG SET UP TO DO ONLY 5 BIT CHAR LENGTH
*SETCL
QUICK EXIT FLAG SET ?
YES
NO
GENERATE NEW CHAR LENGTH CODE
DONE ALL FOUR CHARACTER LENGTHS
YES
NO
IS THE "QUICK" FLAG SET ?
YES
NO
SET UP CHAR LENGTH SELECT BITS IN CURR- ENT LPR "CURLPR"
GENERATE NEW CHAR COUNT FOR THIS LENGTH
SUBUF1(27)
GO SET UP THE OUTPUT BUFFER FOR THIS CHAR LENGTH
MOVE RETURN PC AROUND EXIT BRANCH BACK IN CALLER
*RTS

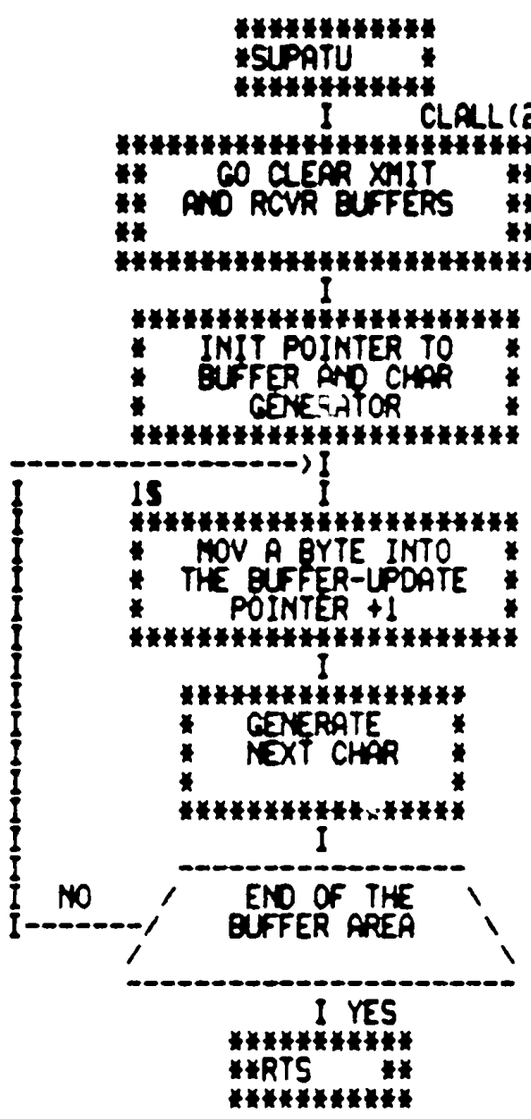

SUBUF1
SAVE REGS R2 R3,AND R4 ON STACK
LOAD R3 WITH CHAR COUNT AND R2 WITH XMIT BUFFER ADDR
INIT R4 TO GENERATE BINARY COUNT PATTERN
GENERATE NEXT CHAR AND COUNT ONE LOADED
LOAD A BYTE FM R4 INTO XMIT BUFFER UPDATE R2
IS R2 AT THE END OF BUFFER ?
NO
YES
RESTORE R4, R3,AND R2 FROM STACK
RTS
SETPAR
DONE ALL THREE PARITY CHECKING COMBINATIONS ?
NO
QUICK TEST FLAG SET ?
YES
SET PARITY SEL-ECT BITS TO 11 ODD PARITY ONLY
SET UP PARITY SELECT BITS IN CURRENT "LPR"
CURRENT SELECT BITS 00 ? NO PARITY
SELECT EVEN PARITY 10
SELECT ODD PARITY 11
CURRENT SELECT BITS 10 ? EVEN PARITY
CURRENT SELECT BITS 11 ? ODD PARITY
SET DONE FLAG -1
MOVE RETURN PC AROUND EXIT BRANCH
RTS
KYBD1
SET UP KEYBOARD VECTOR - ENABLE KEY BOARD INTERRUPT
RTS
A
B
C

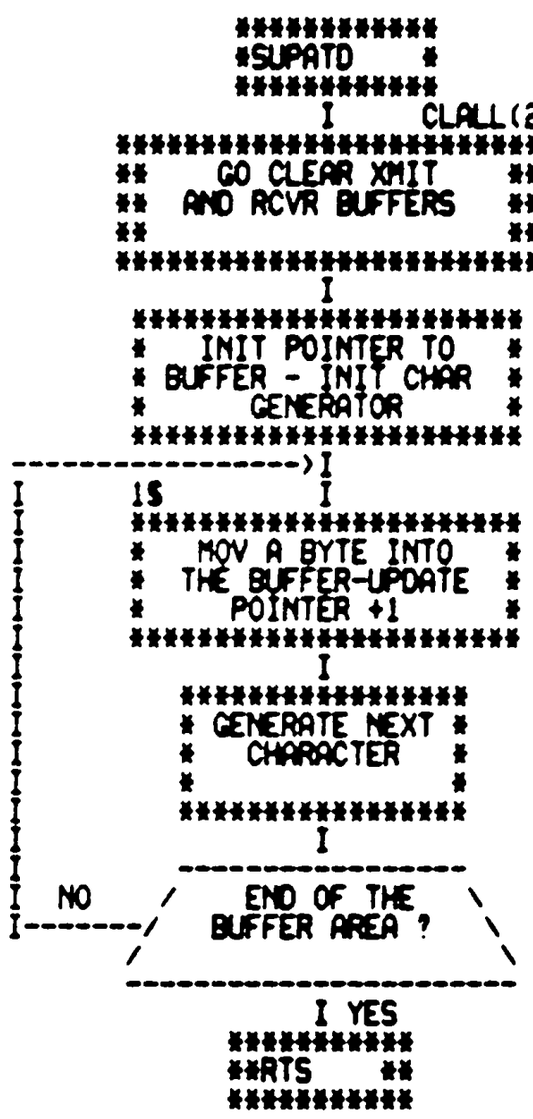

*SUPATS *
CLALL(28)
GO CLEAR XMIT AND RCVR BUFFERS
INIT BUFFER POINTER
LOAD THE SINGLE CHAR TYPED - UPDATE BUFFER POINTER +1
END OF THE BUFFER AREA ?
NO
YES
**RTS **
*CLALL *
INIT RO TO POINT TO BEGIN-NING OF BUFFERS
CLEAR A WORD UPDATE RO +2
END OF BUFFERS ?
NO
YES
**RTS **

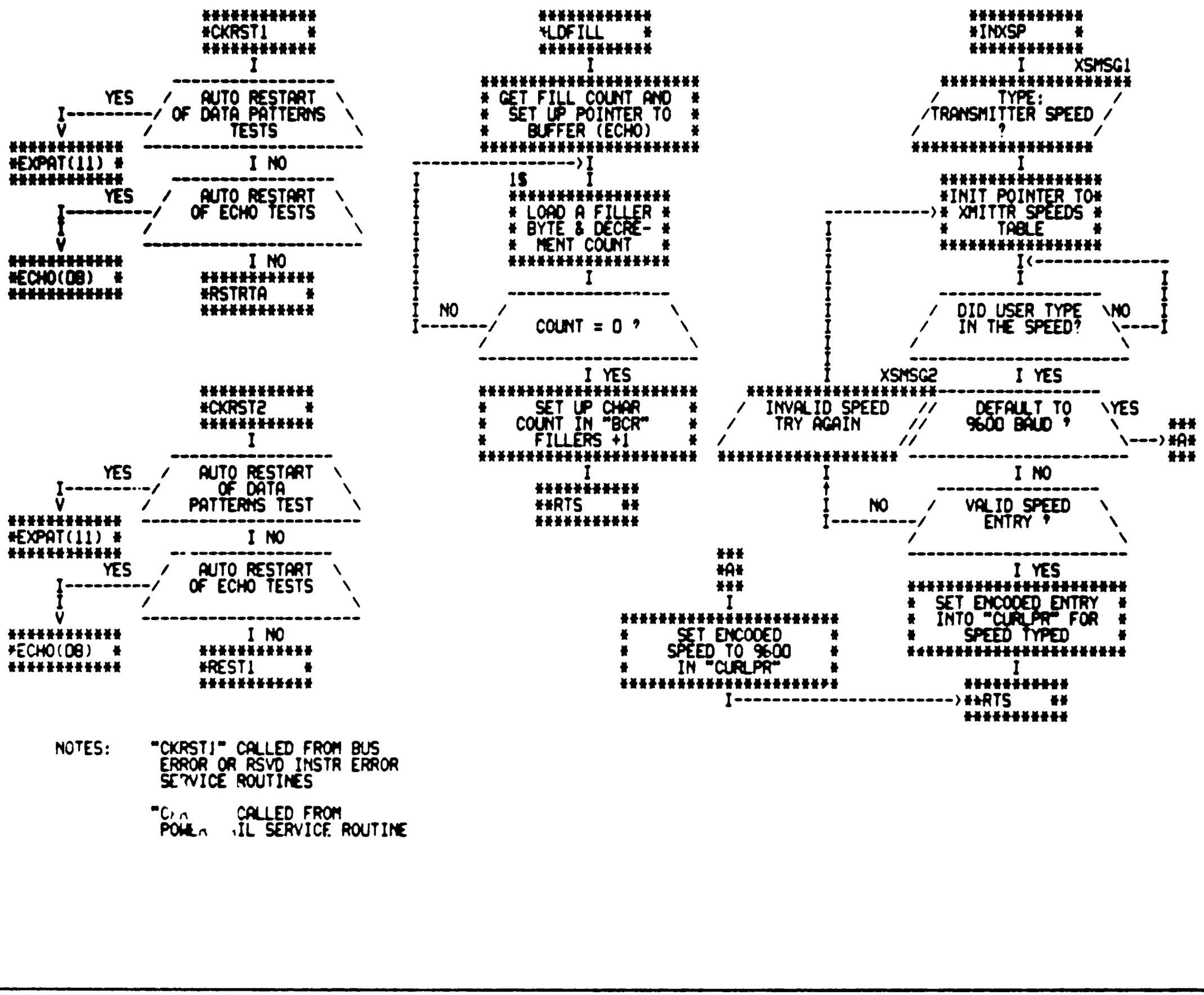
*SUPATR *
CLALL(28)
GO CLEAR XMIT AND RCVR BUFFERS
INIT BUFFER POINT- ER - INIT RANDOM NO. ACCUMULATOR
GENERATE A NEW RANDOM BYTE
LOAD IT INTO XMIT BUFFER.
END OF THE BUFFER AREA ?
NO
YES
**RTS **


*SUCLMK *
LOAD RO WITH SELECT- ED CHAR COUNT FM "CURLPR"
INIT MASK FOR FIVE BIT CHARS.
DONE SET-UP OF BIT MASK ?
YES
NO
**RTS **
SHIFT MASK LEFT DECREMENT COUNT IN RO
NOTE: CHAR LENGTH MASK
5 340
6 300
7 200
8 000


*SUPATA
CLALL(28)
GO CLEAR XMIT AND RCVR BUFFERS
INIT POINTER TO BEGINNING OF BUFFERS
1$
MOVE A 252(8) BYTE INTO BUFFER-UPDATE POINTER +1
NO
END OF THE BUFFERS ?
I YES
**RTS **

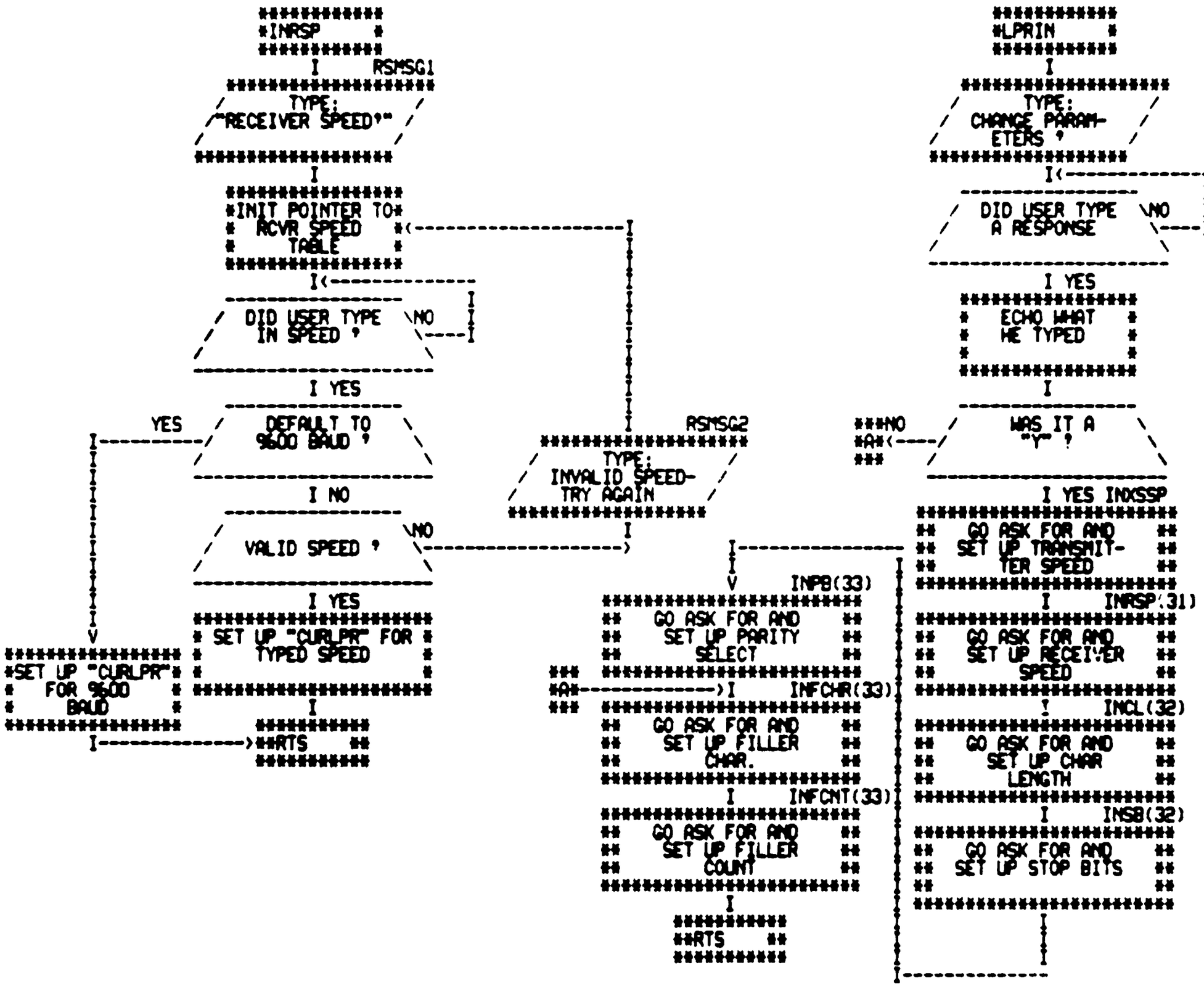

*SUPATU
CLALL(28)
GO CLEAR XMIT AND RCVR BUFFERS
INIT POINTER TO BUFFER AND CHAR GENERATOR
1$
MOV A BYTE INTO THE BUFFER-UPDATE POINTER +1
GENERATE NEXT CHAR
NO
END OF THE BUFFER AREA
I YES
**RTS **

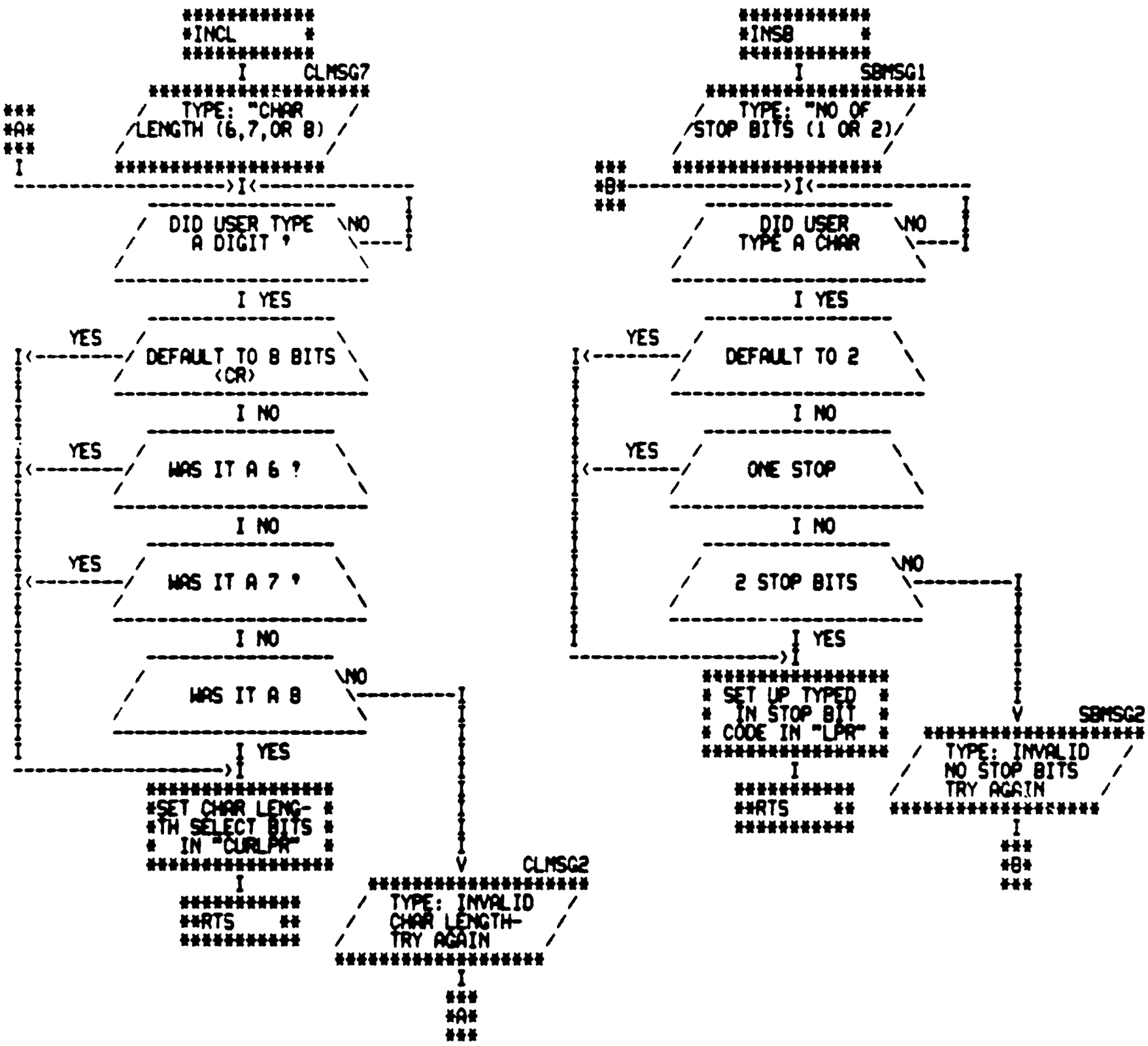

*SUPATD
CLALL(28)
GO CLEAR XMIT AND RCVR BUFFERS
INIT POINTER TO BUFFER - INIT CHAR GENERATOR
1$
MOV A BYTE INTO THE BUFFER-UPDATE POINTER +1
GENERATE NEXT CHARACTER
NO
END OF THE BUFFER AREA ?
I YES
**RTS **

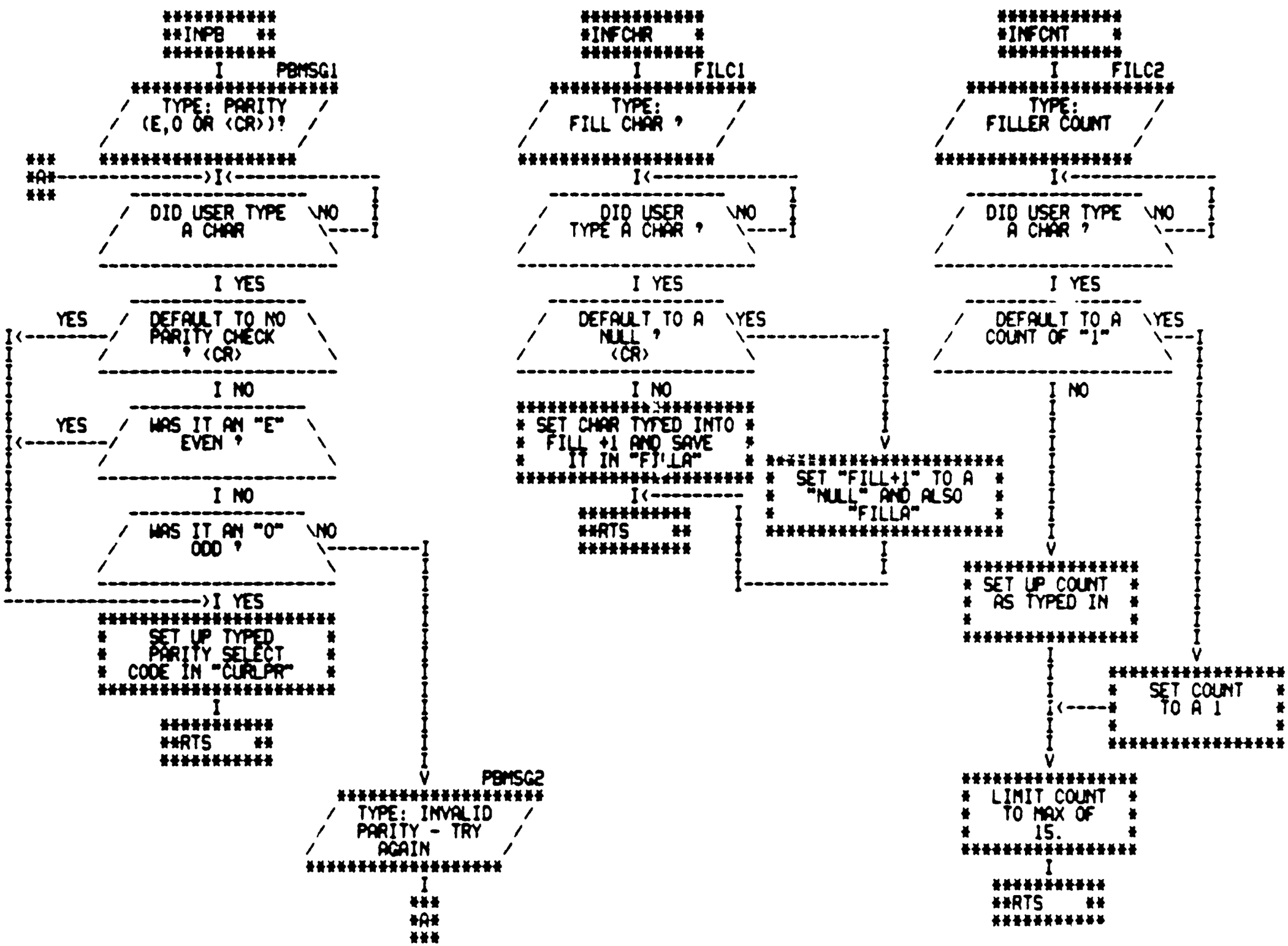


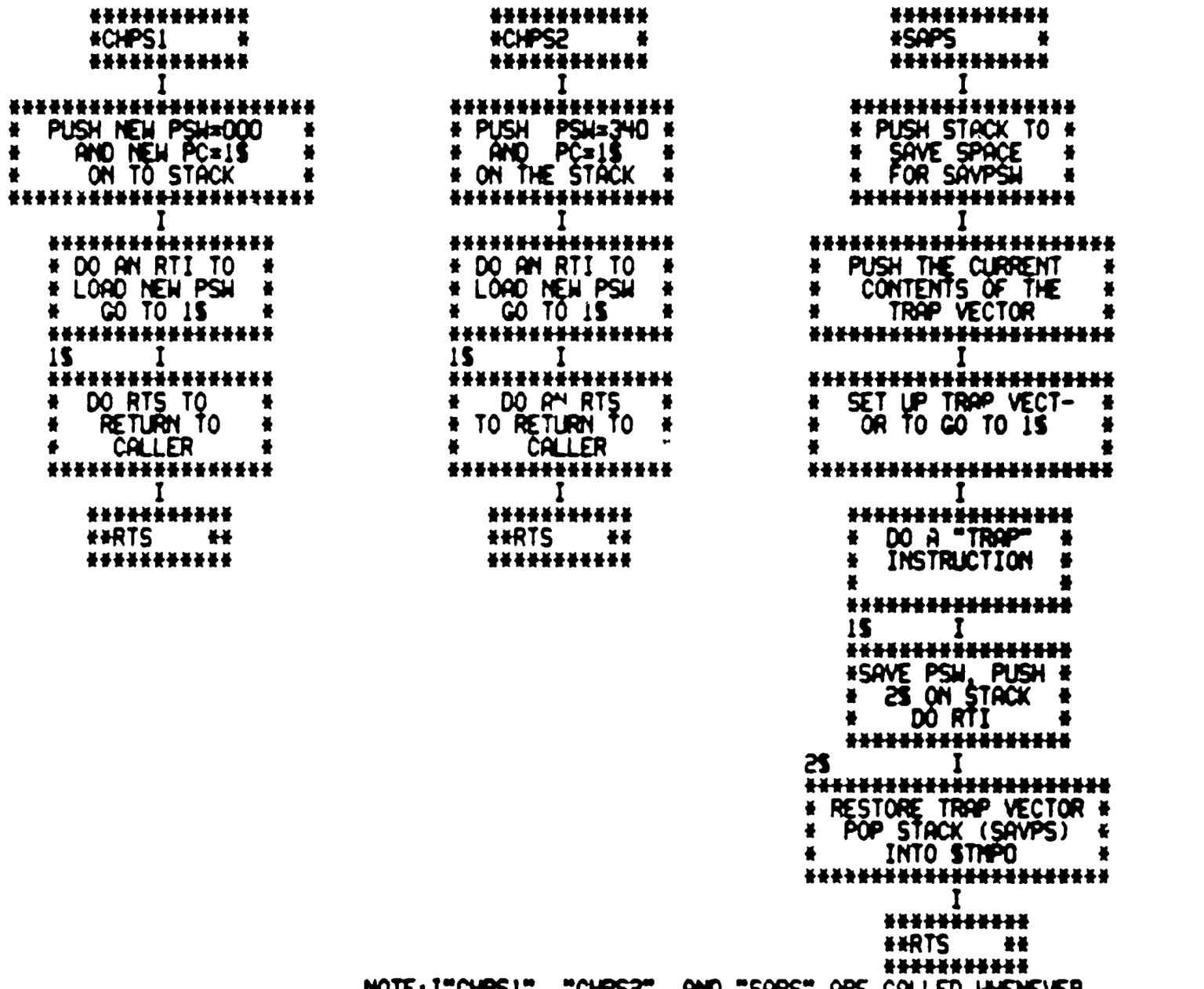
NOTES: "CKRST1" CALLED FROM BUS ERROR OR RSVD INSTR ERROR SERVICE ROUTINES

"C[illegible] CALLED FROM POWER FAIL SERVICE ROUTINE


INRSP
RSMSG1
TYPE: "RECEIVER SPEED?"
INIT POINTER TO * RCVR SPEED TABLE *
DID USER TYPE IN SPEED ?
NO
I YES
DEFAULT TO 9600 BAUD ?
YES
I NO
VALID SPEED ?
NO
I YES
* SET UP "CURLPR" FOR * * TYPED SPEED *
SET UP "CURLPR" FOR 9600 BAUD
RTS
RSMSG2
TYPE: INVALID SPEED- TRY AGAIN
INPB(33)
** GO ASK FOR AND SET UP PARITY SELECT **
INFCHR(33)
** GO ASK FOR AND SET UP FILLER CHAR. **
INFCNT(33)
** GO ASK FOR AND SET UP FILLER COUNT **
RTS
LPRIN
TYPE: CHANGE PARAM-ETERS ?
DID USER TYPE A RESPONSE
NO
I YES
* ECHO WHAT HE TYPED *
WAS IT A "Y" ?
***NO
I YES INXSSP
** GO ASK FOR AND SET UP TRANSMIT-TER SPEED **
INRSP(31)
** GO ASK FOR AND SET UP RECEIVER SPEED **
INCL(32)
** GO ASK FOR AND SET UP CHAR LENGTH **
INSB(32)
** GO ASK FOR AND SET UP STOP BITS **


*INCL *
CLMSG7
TYPE: "CHAR
LENGTH (6,7,OR 8)
A
DID USER TYPE A DIGIT ?
NO
YES
DEFAULT TO 8 BITS <CR>
I NO
WAS IT A 6 ?
WAS IT A 7 ?
WAS IT A 8
*SET CHAR LENG- TH SELECT BITS IN "CURLPR"
*RTS **
CLMSG2
TYPE: INVALID CHAR LENGTH- TRY AGAIN
*INSB *
SBMSG1
TYPE: "NO OF STOP BITS (1 OR 2)
B
DID USER TYPE A CHAR
DEFAULT TO 2
ONE STOP
2 STOP BITS
SET UP TYPED IN STOP BIT CODE IN "LPR"
SBMSG2
TYPE: INVALID NO STOP BITS TRY AGAIN


**INPB **
PBMSG1
TYPE: PARITY (E,O OR <CR>)?
DID USER TYPE A CHAR
NO
YES
I YES
DEFAULT TO NO PARITY CHECK ? <CR>
I NO
WAS IT AN "E" EVEN ?
WAS IT AN "O" ODD ?
SET UP TYPED PARITY SELECT CODE IN "CURLPR"
**RTS **
PBMSG2
TYPE: INVALID PARITY - TRY AGAIN
*INFCHR *
FILC1
TYPE: FILL CHAR ?
DID USER TYPE A CHAR ?
DEFAULT TO A NULL ? <CR>
SET CHAR TYPED INTO FILL +1 AND SAVE IT IN "FILLA"
SET "FILL+1" TO A "NULL" AND ALSO "FILLA"
*INFCNT *
FILC2
TYPE: FILLER COUNT
DEFAULT TO A COUNT OF "1"
SET UP COUNT AS TYPED IN
SET COUNT TO A 1
LIMIT COUNT TO MAX OF 15.



NOTE: I "CHPS1", "CHPS2", AND "SAPS" ARE CALLED WHENEVER
THE MAINLINE CODE HAS TO CLEAR THE PSW, LOCK OUT INTRS,
AND SAVE THE PSW RESPECTIVELY

THESE ROUTINES ARE REQUIRED FOR LSI11 COMPATIBILITY

BEGIN	01																
BEGINA	01	01	01	01	08	11											
BUSER	22																
CHKHDR	22	22															
CHKVCT	23	24															
CHPS1	24																
CHPS2	24																
CKER	05	19															
CKEROP	16																
CKRST1	30																
CKRST2	30																
CLALL	28	28	28	29	29	29											
CLMSG2	32																
CLMSG7	32																
CLSTAT	24	26															
DHSET1	05	16	21														
DHST1	26	06	06														
DHST2	13	13	13	14	14	15	16										
DHWAIT	08																
DPATA	12	13	14														
DPATB	12	15															
DPATCR	12	14															
DPATD	12	13	14														
DPATR	12	14	14														
DPATS	12	14															
DPATU	12	13	14														
DPMS67	14																
DPMSG1	11																
DPMSG2	11																
DPMSGA	15																
ECHO	01	08	30	30													
ECHO1	02	08															
ECHO2	08	08	08	09	11												
ECHO3	08																
ECMSG1	28																
ECMSG2	28																
ENDA	20	20															
EXPAT	01	11	30	30													
EXPAT1	02	11															
EXPAT2	11																
EXPAT3	11	12	13	13	13	14	14	14	15	16	17	17	17	17	18	18	18
FILC1	33																
FILC2	33																
INCL	31	32															
INFCHR	31	33															
INFCNT	31	33															
INPAR	02	08	11	22													
INPAR1	22	23															
INPAR3	02	23															
INPAR4	23																
INPARA	01	02	02														
INPARC	01																

INPARX	01																	
INPB	31	33#																
INXSP	31	31																
INXB	31	32																
INXSP	30																	
INXSSP	31																	
KYBD1	04	27																
KYBD2	26																	
LDFILL	30																	
LPRIN	08	31																
NEWCL	04																	
NEWLIN	04	04	05	06	06	07	07											
NEWLPR	04																	
NEWPAR	04	04	05															
PBMSG1	33																	
PBMSG2	33																	
PRSTAT	04	20	26															
RESERR	05																	
RESTI	03	25	25	30														
RESTRP	05																	
RESTRT	03	03																
RINT1	07																	
RINT2	08	10																
RINT3	18																	
RSMSG1	31																	
RSMSG2	31																	
RSTRTA	20	30																
RTI	06	07	10	10	17	18	26											
RTS	02	09	13	13	13	14	19	19	21	21	22	22	24	24	25	25	26	26
	26	27	27	27	28	28	28	28	29	29	29	30	30	31	31	32	32	33
	33	33	34	34	34													
SAPS	34																	
SPMSG1	32																	
SPMSG2	32																	
SELINE	04	21																
SENDP1	08	09	09															
SENDP2	09	09																
SETCL	04	26																
SETLPR	04	26																
SETPAR	04	27																
SNMSG1	08																	
SOFT	19																	
ST2	01	01	02															
START	01#																	
START1	01																	
START2	02	03	23	23														
STDH1	03	04																
SUBUF1	26	27																
SUCLMK	11	28																
SUER1	22																	
SUER2	22																	
SUNUM	03																	

FLOW CHART CROSS REFERENCE LIST

	04	05	17	17	17	17	17	18	18	18	22							
SUPATA	13	29																
SUPATD	13	29																
SUPATR	14	28																
SUPATS	14	28																
SUPATU	13	29																
TIMEIT	05	16	25															
TINT1	08																	
TINT2	08	10																
TINT3	17																	
UPDER	19	19																
XSMSG1	30																	
XSMSG2	30																	
$E%P	20																	
$ERROR	05	06	06	06	06	06	07	07	07	08	13	13	13	14	14	15	16	16
	17	17	17	17	17	18	18	18	19	25	25							
1$	05	06	07	09	10	13	13	13	14	14	19	20	24	24	26	28	28	28
	28	29	29	33	34	34	34											
10$	06																	
2$	05	06	07	10	13	13	13	14	14	15	20	24	24	26	34			
3$	05	06	07	09	10	13	13	13	14	15	20	24	24					
31$	10																	
4$	06	07	15	24	24													
5$	06	07																
6$	06	07																
7$	06																	
8$	06																	
9$	06																	

DZDHNA.P11 TABLE OF CONTENTS

```
                                .NLIST  CND,MD,MC
                                .LIST   TOC,ME,SEQ,BIN
                 165000         $SWR=165000
                                .MCALL  .HEADER,.SCMTAG,.SETUP,.SERROR
                                .MCALL  .SWRHI,.SWRLO,.$CATCH,.EQUATE,.SETUP,.SEOP
                                .MCALL  .$SCOPE,.SERRTYP,.STYPOCT,.STYPDEC,.STYPE,.SPOWER,.STRAP
                                .MCALL  .$READ,.$ROOCT,.$RODEC
                                .MCALL  .SACT11,.SAPTHDR,.SAPTBLS,.SAPTYPE

                                 ENABLE ABS
                                 TITLE  MAINDEC-11-DZDHN-A
                                 COPYRIGHT (C) 1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY ED CROWLEY
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSTEM
                                ;*PACKAGE (MAINDEC-11-DZQAC-B1),AUG 29,1975.
                                ;*
                 000001         $TN=1

                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        14            LOOP ON TEST
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        11            INHIBIT ITERATIONS
                                ;*         9            LOOP ON ERROR
                                ;*****************************************************************

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

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

```
MAINDEC-11-DZDHN-A     MACY11 27(663)  15-DEC-75  09:29  PAGE 1-1                                                    SEQ 0087
DZDHNA.P11      APT PARAMETER BLOCK

   (1)                                          ;INTERFACE SPEC.
   (1)
   (1)  000000                                  $APTHD:
   (1)  000000  000000                          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
   (1)  000002  001230                          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
   (1)  000004  001604                          $TSTM:  .WORD   1604    ;;RUN TIM OF LONGEST TEST
   (1)  000006  001604                          $PASTM: .WORD   1604    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
   (1)  000010  001604                          $UNITM: .WORD   1604    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
   (1)  000012  000052                                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
    16
   (1)                                          .SBTTL  TRAP CATCHER
   (1)
   (1)          000000                                  .=0
   (1)                                          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
   (1)                                          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
   (1)                                          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
   (1)          000174                                  .=174
   (1)  000174  000000                                  DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
   (1)  000176  000000                                  SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
   (1)
   (1)                                          .SBTTL  STARTING ADDRESS(ES)
   (1)  000200  000137  014322                          JMP     @#INPARX                ;;JUMP TO STARTING ADDRESS OF PROGRAM
    17
    18  000204  000137  001624                          JMP     @#BEGIN         ;BEGIN EXECUTION WITH DEFAULT PARAMETERS
    19  000210  000137  014334                          JMP     @#INPARC        ;INPUT PARAMETERS - DEVICE SELECTION ONLY
    20  000214  000137  004604                          JMP     @#ECHO          ;GO START LINE ECHO TESTS
    21  000220  000137  006004                          JMP     @#EXPAT         ;GO START DATA PATTERNS TESTS
```

```
 23
(1)                                 .SBTTL  BASIC DEFINITIONS
(1)
(1)                                 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
(1)          001100                 STACK=  1100
(1)                                 .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
(1)                                 .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
(1)          177776                 PS=     177776          ;;PROCESSOR STATUS WORD
(1)                                 .EQUIV  PS,PSW
(1)          177774                 STKLMT= 177774          ;;STACK LIMIT REGISTER
(1)          177772                 PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
(1)          177570                 DSWR=   177570          ;;HARDWARE SWITCH REGISTER
(1)          177570                 DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
(1)
(1)                                 ;*GENERAL PURPOSE REGISTER DEFINITIONS
(1)          000000                 R0=     %0              ;;GENERAL REGISTER
(1)          000001                 R1=     %1              ;;GENERAL REGISTER
(1)          000002                 R2=     %2              ;;GENERAL REGISTER
(1)          000003                 R3=     %3              ;;GENERAL REGISTER
(1)          000004                 R4=     %4              ;;GENERAL REGISTER
(1)          000005                 R5=     %5              ;;GENERAL REGISTER
(1)          000006                 R6=     %6              ;;GENERAL REGISTER
(1)          000007                 R7=     %7              ;;GENERAL REGISTER
(1)                                 .EQUIV  R6,SP           ;;STACK POINTER
(1)                                 .EQUIV  R7,PC           ;;PROGRAM COUNTER
(1)
(1)                                 ;*PRIORITY LEVEL DEFINITIONS
(1)          000000                 PR0=    0               ;;PRIORITY LEVEL 0
(1)          000040                 PR1=    40              ;;PRIORITY LEVEL 1
(1)          000100                 PR2=    100             ;;PRIORITY LEVEL 2
(1)          000140                 PR3=    140             ;;PRIORITY LEVEL 3
(1)          000200                 PR4=    200             ;;PRIORITY LEVEL 4
(1)          000240                 PR5=    240             ;;PRIORITY LEVEL 5
(1)          000300                 PR6=    300             ;;PRIORITY LEVEL 6
(1)          000340                 PR7=    340             ;;PRIORITY LEVEL 7
(1)
(1)                                 ;*"SWITCH REGISTER" SWITCH DEFINITIONS
(1)          100000                 SW15=   100000
(1)          040000                 SW14=   40000
(1)          020000                 SW13=   20000
(1)          010000                 SW12=   10000
(1)          004000                 SW11=   4000
(1)          002000                 SW10=   2000
(1)          001000                 SW09=   1000
(1)          000400                 SW08=   400
(1)          000200                 SW07=   200
(1)          000100                 SW06=   100
(1)          000040                 SW05=   40
(1)          000020                 SW04=   20
(1)          000010                 SW03=   10
(1)          000004                 SW02=   4
(1)          000002                 SW01=   2
(1)          000001                 SW00=   1
(1)                                 .EQUIV  SW09,SW9
```

```
(1)                             .EQUIV  SW08,SW8
(1)                             .EQUIV  SW07,SW7
(1)                             .EQUIV  SW06,SW6
(1)                             .EQUIV  SW05,SW5
(1)                             .EQUIV  SW04,SW4
(1)                             .EQUIV  SW03,SW3
(1)                             .EQUIV  SW02,SW2
(1)                             .EQUIV  SW01,SW1
(1)                             .EQUIV  SW00,SW0
(1)
(1)                             ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)             100000          BIT15=  100000
(1)             040000          BIT14=  40000
(1)             020000          BIT13=  20000
(1)             010000          BIT12=  10000
(1)             004000          BIT11=  4000
(1)             002000          BIT10=  2000
(1)             001000          BIT09=  1000
(1)             000400          BIT08=  400
(1)             000200          BIT07=  200
(1)             000100          BIT06=  100
(1)             000040          BIT05=  40
(1)             000020          BIT04=  20
(1)             000010          BIT03=  10
(1)             000004          BIT02=  4
(1)             000002          BIT01=  2
(1)             000001          BIT00=  1
(1)                             .EQUIV  BIT09,BIT9
(1)                             .EQUIV  BIT08,BIT8
(1)                             .EQUIV  BIT07,BIT7
(1)                             .EQUIV  BIT06,BIT6
(1)                             .EQUIV  BIT05,BIT5
(1)                             .EQUIV  BIT04,BIT4
(1)                             .EQUIV  BIT03,BIT3
(1)                             .EQUIV  BIT02,BIT2
(1)                             .EQUIV  BIT01,BIT1
(1)                             .EQUIV  BIT00,BIT0
(1)
(1)                             ;*BASIC "CPU" TRAP VECTOR ADDRESSES
(1)             000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
(1)             000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)             000014          TBITVEC=14              ;;"T" BIT
(1)             000014          TRTVEC= 14              ;;TRACE TRAP
(1)             000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
(1)             000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)             000024          PWRVEC= 24              ;;POWER FAIL
(1)             000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
(1)             000034          TRAPVEC=34              ;;"TRAP" TRAP
(1)             000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
(1)             000064          TPVEC=  64              ;;TTY PRINTER VECTOR
(1)             000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
```

```
 24
(1)
(1)
(1)
(1)
(1)
(1)
(1)                 001100                  .=1100
(1)   001100                        $CMTAG:                         ;;START OF COMMON TAGS
(1)   001100 000000                         .WORD   0
(1)   001102    000                 $ESTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
(1)   001103    000                 $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
(1)   001104 000000                 $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
(1)   001106 000000                 $LPADR: .WORD   0               ;;CONTAINS SCO   LOOP ADDRESS
(1)   001110 000000                 $LPERR: .WORD   0               ;;CONTAINS SCO   RETURN FOR ERRORS
(1)   001112 000000                 $ERTTL: .WORD   0               ;;CONTAINS TOTA  ERRORS DETECTED
(1)   001114    000                 $ITEMB: .BYTE   0               ;;CONTAINS ITEM  CONTROL BYTE
(1)   001115    001                 $ERMAX: .BYTE   1               ;;CONTAINS MAX.  ERRORS PER TEST
(1)   001116 000000                 $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
(1)   001120 000000                 $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
(1)   001122 000000                 $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
(1)   001124 000000                 $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
(1)   001126 000000                 $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
(1)   001130 000000                         .WORD   0               ;;RESERVED--NOT TO BE USED
(1)   001132 000000                         .WORD   0
(1)   001134 000000                         .WORD   0
(1)   001136 177570                 SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
(1)   001140 177570                 DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
(1)   001142 177560                 $TKS:   177560                  ;;TTY KBD STATUS
(1)   001144 177562                 $TKB:   177562                  ;;TTY KBD BUFFER
(1)   001146 177564                 $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
(1)   001150 177566                 $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
(1)   001152    000                 $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
(1)   001153    002                 $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
(1)   001154    012                 $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
(1)   001155    000                 $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
(1)   001156 000000                 $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
(1)                                                                 ;;WHICH  ($REGO) WAS OBTAINED
(3)   001160 000000                 $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
(3)   001162 000000                 $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
(3)   001164 000000                 $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
(3)   001166 000000                 $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
(3)   001170 000000                 $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
(3)   001172 000000                 $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
(3)   001174 000000                 $REG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
(3)   001176 000000                 $REG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
(3)   001200 000000                 $TMP0:  .WORD   0               ;;USER DEFINED
(3)   001202 000000                 $TMP1:  .WORD   0               ;;USER DEFINED
(3)   001204 000000                 $TMP2:  .WORD   0               ;;USER DEFINED
(3)   001206 000000                 $TMP3:  .WORD   0               ;;USER DEFINED
(3)   001210 000000                 $TMP4:  .WORD   0               ;;USER DEFINED
(3)   001212 000000                 $TMP5:  .WORD   0               ;;USER DEFINED
(3)   001214 000000                 $TMP6:  .WORD   0               ;;USER DEFINED
(3)   001216 000000                 $TMP7:  .WORD   0               ;;USER DEFINED
```

```
MAINDEC-11-DZDHN-A     MACY11 27(663)  15-DEC-75  09:29  PAGE 1-5                                SEQ 0091
DZDHNA.P11       COMMON TAGS

    (1)  001220  000000                $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
    (1)  001222  000000                $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
    (1)  001224      077               $QUES:  .ASCII  /?/             ;;QUESTION MARK
    (1)  001225      015               $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
    (1)  001226  000012                $LF:    .ASCIZ  <12>            ;;LINE FEED
```

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

                        .SBTTL  APT MAILBOX-ETABLE


                                .EVEN
001230                  $MAIL:                  ;;APT MAILBOX
001230  000000          $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
001232  000000          $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
001234  000000          $TESTN: .WORD   ATESTN  ;;TEST NUMBER
001236  000000          $PASS:  .WORD   APASS   ;;PASS COUNT
001240  000000          $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
001242  000000          $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
001244  000000          $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
001246  000000          $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001250                  $ETABLE:                ;;APT ENVIRONMENT TABLE
001250     000          $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001251     000          $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001252  000000          $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001254  000000          $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001256  000000          $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                        ;*                      BITS 15-11=CPU TYPE
                        ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                        ;*                              11/70=06,PDQ=07,0=10
                        ;*                      BIT 10=REAL TIME CLOCK
                        ;*                      BIT  9=FLOATING POINT PROCESSOR
                        ;*                      BIT  8=MEMORY MANAGEMENT
001260     000          $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001261     000          $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                        ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                        ;*                              900 NSEC CORE=001
                        ;*                              300 NSEC BIPOLAR=002
                        ;*                              500 NSEC MOS=003
001262  000000          $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                        ;*                      MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
001264     000          $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001265     000          $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001266  000000          $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001270     000          $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001271     000          $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001272  000000          $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001274     000          $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001275     000          $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001276  000000          $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001300     000          $VECT1: .BYTE   AVECT1  ;;INTERRUPT VECTOR#1
001301     000          $VECT2: .BYTE   AVECT2  ;;INTERRUPT VECTOR#2
001302     000          $PRIOR: .BYTE   APRIOR  ;;BUS PRIORITY #1,#2
001303     000                  .BYTE   0      ;;SPARE,NOT USED
                                .EVEN
001304  000000          $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001306  000000          $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001310  000000          $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
```

```
(2)  001312  000000                $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
(2)  001314  000000                $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
(2)  001316  000000                $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
(2)  001320  000000                $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
(2)  001322  000000                $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
(2)  001324  000000                $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
(2)  001326  000000                $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
(2)  001330  000000                $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
(2)  001332  000000                $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
(2)  001334  000000                $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
(2)  001336  000000                $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
(2)  001340  000000                $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
(2)  001342  000000                $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
(2)  001344  000000                $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
(2)  001346  000000                $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
(2)  001350  000000                $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
(2)  001352  000000                $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
(2)
(2)
(2)  001354                        $ETEND:
(2)
(1)
(1)                                .SBTTL  ERROR POINTER TABLE
(1)
(1)                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1)                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1)                                ;*LOCATION SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1)                                ;*NOTE1:        IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC).
(1)                                ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1)                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(1)                                ;*      DH              ;;POINTS TO THE DATA HEADER
(1)                                ;*      DT              ;;POINTS TO THE DA
(1)                                ;*      DF              ;;POINTS TO THE DATA FORMAT
(1)
(1)
(1)  001354                        $ERRTB:

                                   ;ERROR TABLE ITEM FOR ERROR 1

28   001354  020166                        EM1             ;"NON EX MEMORY ERROR - DROPPED LINE #  "
29   001356  020235                        DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
30   001360  020312                        DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
31   001362  020330                        DF2             ;PRINT ALL OCTAL

                                   ;ERROR TABLE ITEM FOR ERROR 2

34   001364  020340                        EM2             ;"TRANSMITTER FALSE INTERRUPT - DROPPED LINE#  "
36   001366  020235                        DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
37   001370  020312                        DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
38   001372  020330                        DF2             ;PRINT ALL OCTAL

                                   ;ERROR TABLE ITEM FOR ERROR 3
```

E08


```
001374  020417                          EM3             ;"BUFFER ACTIVE REGISTER ERROR - DROPPED LINE #  "
001376  020235                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001400  020312                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001402  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 4

001404  020477                          EM4             ;"BYTE COUNT REGISTER ERROR - DROPPED LINE #  "
001406  020235                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001410  020312                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001412  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 5

001414  020554                          EM5             ;"CURRENT ADDRESS REGISTER ERROR - DROPPED LINE #  "
001416  020235                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001420  020312                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001422  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 6

001424  020636                          EM6             ;"SILO OVERFLOW ERROR - DROPPED LINE #  "
001426  020235                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001430  020312                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001432  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 7

001434  020705                          EM7             ;"RECEIVER FALSE INTERRUPT - LINE #  "
001436  020235                          DH1             ;" (PC)   CURLPR  DEVADR  REGADR   WAS     S/B"
001440  020312                          DT1             ;"$ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4
001442  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 10

001444  020761                          EM10            ;"INVALID DATA IN SILO - DROPPED LINE #  "
001446  021031                          DH2             ;" (PC)   CURLPR  CHAR #  WASADR  SHBADR   WAS     S/B"
001450  021116                          DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
001452  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 11

001454  021136                          EM11            ;"DATA ERROR - LINE #  "
001456  021031                          DH2             ;" (PC)   CURLPR  CHAR #  WASADR  SHBADR   WAS     S/B"
001460  021116                          DT2             ;$ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4
001462  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 12

001464  021164                          EM12            ;"TEST TIMEOUT - DROPPED LINE #  "
001466  021224                          DH3             ;" (PC)   CURLPR  RTOTAL  XTOTAL  RDONE"
001470  021272                          DT3             ;"$ERRPC,CURLPR,STMP0,STMP1RDONE"
001472  020330                          DF2             ;PRINT ALL OCTAL
```

```
                                ;ERROR TABLE ITEM FOR ERROR 13

001474  000000                          0               ;NO MESSAGE
001476  000000                          0               ;NO DATA HEADER
001500  021306                          DT4             ;$TMP0,$TMP1,$TMP2,$TMP3,$TMP4,$TMP5,$TMP6
001502  021326                          DF1             ;PRINT ALL DECIMAL

                                ;ERROR TABLE ITEM FOR ERROR 14

001504  021336                          EM14            ;"BUS ERROR TRAP TO 04"
001506  021363                          DH4             ;"  (PC)    (PS)    (SP)   TRAPPC  TRAPPS"
001510  021432                          DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2"
001512  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 15

001514  021446                          EM15            ;"RSVD INSTR TRAP TO 10"
001516  021363                          DH4             ;"  (PC)    (PS)    (SP)   TRAPPC  TRAPPS"
001520  021432                          DT5             ;$ERRPC,$TMP0,$REG6,$REG1,$REG2"
001522  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 16

001524  021474                          EM16            ;"SINGLE LINE ECHO TEST - INTR WAIT TIMEOUT"
001526  021546                          DH5             ;"  (PC)   DEVADR   LINE    (SCR)  CURLPR  EXFLAG"
001530  021626                          DT6             ;$ERRPC,$REG1,LINE,$TMP0,CURLPR,EXFLAG"
001532  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 17

001534  021644                          EM17            ;"ALTERNATING I/O PATTERN TEST DONE"
001536  021706                          DH6             ;"  (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001540  021756                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
001542  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 20

001544  021772                          EM20            ;"BINARY UP COUNT PATTERN TEST DONE"
001546  021706                          DH6             ;"  (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001550  021756                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
001552  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 21

001554  022034                          EM21            ;"BINARY DOWN COUNT PATTERN TEST DONE"
001556  021706                          DH6             ;"  (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001560  021756                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
001562  020330                          DF2             ;PRINT ALL OCTAL

                                ;ERROR TABLE ITEM FOR ERROR 22

001564  022100                          EM22            ;"RANDOM DATA PATTERN TEST DONE"
001566  021706                          DH6             ;"  (PC)   DEVADR   LINE   CURLPR  ICOUNT"
001570  021756                          DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REG0
```

```
 150  001572  020330                        DF2             ;PRINT ALL OCTAL
 151
 152                                ;ERROR TABLE ITEM FOR ERROR 23
 153
 154  001574  022136                        EM23            ;SINGLE CHAR PATTERN TEST DONE"
 155  001576  021706                        DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
 156  001600  021756                        DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
 157  001602  020330                        DF2             ;PRINT ALL OCTAL
 158
 159                                ;ERROR TABLE ITEM FOR ERROR 24
 160
 161  001604  022174                        EM24            ;"TYPED BUFFER PATTERN TEST DONE"
 162  001606  021706                        DH6             ;" (PC)   DEVADR   LINE   CURLPR  ICOUNT"
 163  001610  021756                        DT7             ;$ERRPC,DHADR,LINE,CURLPR,$REGO
 164  001612  020330                        DF2             ;PRINT ALL OCTAL
 165
 166                                ;ERROR TABLE ITEM FOR ERROR 25
 167
 168  001614  022233                        EM25            ;"DATA PATTERNS TEST TIMEOUT"
 169  001616  022266                        DH7             ;" (PC) DEVADR   LINE   CURLPR  ICOUNT  PATCDE"
 170  001620  022346                        DT10            ;$ERRPC,DHADR,LINE,CURLPR,$REGO,$REG1
 171  001622  020330                        DF2             ;PRINT ALL OCTAL
 172
 173
 174
 175
 176  001624  005000            BEGIN:  CLR     R0          ;INIT R0 TO INDICATE DEFAULT PARAMETERS
 177  001626  005067  015776            CLR     VCFLG       ;INIT VECTOR ADDR SET UP FLAG
 178  001632  005067  016234            CLR     DPFLG       ;CLEAR DATA PATTERNS TEST FLAG
 179  001636  005067  016242            CLR     RETFLG      ;CLEAR ECHO TEST RETURN FLAG
 180  001642  005067  016214            CLR     TITFLG      ;INIT TITLE MESSAGE FLAG
 182  001646  012706  001100            MOV     #$CMTAG,R6  ;FIRST LOCATION TO BE CLEARED
(1)   001652  005026                    CLR     (R6)+       ;CLEAR MEMORY LOCATION
(1)   001654  022706  001126            CMP     #$BOOAT,R6  ;DONE?
(1)   001660  001374                    BNE     .-6         ;LOOP BACK IF NO
(1)   001662  012706  001100            MOV     #STACK,SP   ;SETUP THE STACK POINTER
(1)   001666  012737  011006  000020    MOV     #$SCOPE,@#IOTVEC ;IOT VECTOR FOR SCOPE ROUTINE
(1)   001674  012737  000340  000022    MOV     #340,@#IOTVEC+2 ;LEVEL 7
(1)   001702  012737  011250  000030    MOV     #$ERROR,@#EMTVEC ;EMT VECTOR FOR ERROR ROUTINE
(1)   001710  012737  000340  000032    MOV     #340,@#EMTVEC+2 ;LEVEL 7
(1)   001716  012737  013460  000034    MOV     #$TRAP,@#TRAPVEC ;TRAP VECTOR FOR TRAP CALLS
(1)   001724  012737  000340  000036    MOV     #340,@#TRAPVEC+2;LEVEL 7
(1)   001732  012737  013524  000024    MOV     #$PWRDN,@#PWRVEC ;POWER FAILURE VECTOR
(1)   001740  012737  000340  000026    MOV     #340,@#PWRVEC+2 ;LEVEL 7
(1)   001746  005067  177246            CLR     $TIMES      ;INITIALIZE NUMBER OF ITERATIONS
(1)   001752  005067  177244            CLR     $ESCAPE     ;CLEAR THE ESCAPE ON ERROR ADDRESS
(1)   001756  112767  000001  177131    MOVB    #1,$ERMAX   ;ALLOW ONE ERROR PER TEST
(1)   001764  012767  001764  177114    MOV     #.,$LPADR   ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
(1)   001772  012767  001772  177110    MOV     #.,$LPERR   ;SETUP THE ERROR LOOP ADDRESS
(2)   002000  013746  000004            MOV     @#4,-(SP)   ;SAVE ERROR VECTOR
(2)   002004  013746  000006            MOV     @#6,-(SP)
(2)   002010  012767  002024  175766    MOV     #64$,4      ;SET UP TIME OUT VECTOR
(2)   002016  005777  177114            TST     @SWR        ;TRY TO REFERENCE HARDWARE SWR
(2)   002022  000407                    BR      65$         ;BRANCH IF NO TIMEOUT TRAP OCCURS
```

H08


```
(2)  002024  012767  000176  177104  64$:    MOV     #SWREG,SWR          ;;POINT TO SOFTWARE SWR
(2)  002032  012767  000174  177100          MOV     #DISPREG,DISPLAY    ;;POINT TO SOFTWARE DISPLAY REG
(2)  002040  022626                          CMP     (SP)+,(SP)+         ;;RESTORE STACK
(2)  002042  012637  000006          65$:    MOV     (SP)+,@#6           ;;RESTORE ERROR VECTOR
(2)  002046  012637  000004                  MOV     (SP)+,@#4
(1)  002052  005067  177160                  CLR     $PASS               ;;CLEAR PASS COUNT
(1)  002056  132767  000200  177165          BITB    #APTSIZE,$ENVM      ;;TEST USER SIZE UNDER APT
(1)  002064  001403                          BEQ     3$                  ;;YES,USE NON-APT SWITCH
(1)  002066  012767  001252  177042          MOV     #$SWREG,SWR         ;;NO,USE APT SWITCH REGISTER
(1)  002074                          3$:
```

```
                                ;SET UP THE BUS ERROR VECTOR
                                ;SET UP THE RSVD INSTR VECTOR
                                ;HAVE WE TYPED TITLE ONCE ?
                                ;BR IF YES
                                ;GO TYPE PROGRAM TITLE
                                ;SET FLAG - TYPE TITLE ONLY ONCE PER LOAD
                                ;START AT 200 ??
                                ;BR IF NOT
                                ;GO ASK FOR PARAMETERS
                                ;RE INIT START FLAG
                                ;RETURN TO ECHO TESTS ?
                                ;BR IF NOT
                                ;RETURN TO ECHO TEST START-UP
                                ;RETURN TO DATA PATTERNS TEST ?
                                ;BR IF NOT
                                ;GO BACK TO DATA PATTERNS TESTS
                                ;USE DEFAULT PARAMETERS ?
                                ;BR IF YES
                                ;CHANGE DH SELECT PARAM ONLY ?
                                ;BR IF NOT
                                ;GO ASK FOR SELECT PARAM.
                                ;GO ASK FOR ALL PARAMETERS

                                ;GET POINTER TO ADDRESS TABLE
                                ;GET POINTER TO VECTOR TABLE
                                ;GET POINTER TO BR LEVEL TABLE
                                ;START WITH DH #00
                                ;SET UP DH11 BIT TEST MARKER

                                ;GENERATE DH11 DEV NUMBER
                                ;UPDATE TABLE POINTERS

                                ;TEST FOR SELECTED DH11
                                ;BR IF SELECTED FOR TEST
                                ;SHIFT MARKER TO TEST NEXT DH11
                                ;BR IF 16 TESTED - START OVER
                                ;GO TEST IF THIS ONE SELECTED
                                ;SET UP DH11 ADDRESS
                                ;SET UP THE DH11 VECTOR ENTRY
                                ;GET BR LEVEL VALUES
                                ;GO SET DH NUMBER IN THE MESSAGE BUFFER

                                ;GO PRINT "TESTING DH11 #XX"
```

```
  236  002370  012767  002370  176510          MOV     #.,$LPADR       ;INIT SCOPE RETURN
  237                                  ;*****************************************************************
  (3)                                  ;*TEST 1        SUB-PROGRAM 1 - DATA RELIABILITY TESTS
  (3)                                  ;*****************************************************************
  (2)  002376  000004                  TST1:   SCOPE
  (1)  002400  012767  000001  176612          MOV     #1,$TIMES       ;;DO 1 ITERATION
  (1)  002406  012767  000001  176620          MOV     #$TN-1,$TESTN   ;;SET TEST NUMBER IN  MAIL BOX
  238  002414  004767  012504          STDH1:  JSR     PC,CLSTAT       ;GO CLEAR THE STATISTICS TABLES
  239  002420  004767  013064                  JSR     PC,KYBD1        ;GO SET UP FOR KEYBOARD INTRS.
  240  002424  005067  015024                  CLR     QUICK           ;INIT THE QUICK TEST FLAG
  241  002430  005067  015016                  CLR     DRPLIN          ;INIT DROPPED LINE FLAGS
  242  002434  005067  015236                  CLR     LINE            ;INIT LINE NO. TO 00
  243  002440  016702  014774                  MOV     DHVCT,R2        ;SET UP THE VECTORS
  244  002444  012722  003446                  MOV     #RINT1,(R2)+    ;GO TO RINT1 ON RCVR INTR
  245  002450  116722  015216                  MOVB    DHRLVL,(R2)+    ;
  246  002454  105722                          TSTB    (R2)+
  247  002456  012722  002742                  MOV     #TINT1,(R2)+    ;GO TO TINT1 ON XMITTR INTR
  248  002462  116712  015205                  MOVB    DHTLVL,(R2)
  249  002466  016701  014744                  MOV     DHADR,R1        ;SET UP DEVICE ADDRESS
  250
  251  002472  004767  011350          NEWLIN: JSR     PC,SELINE       ;GO SELECT NEW LINE FOR TEST
  252  002476  000401                          BR      1$              ;BR IF TESTED ALL SELECTED LINES
  253  002500  000402                          BR      2$              ;BR IF NOT DONE
  254  002502  000167  001616          1$:     JMP     PRSTAT          ;GO CHECK AND REPORT STATISTICS
  255  002506  004567  011426          2$:     JSR     R5,SUNUM        ;PUT LINE NO. IN MSG
  256  002512  017676                          LINE
  257  002514  023321                          STMSG2+20
  258  002516  104400                          TYPE                    ;TYPE LINE NO. BEING TESTED
  259  002520  023301                          STMSG2
  260  002522  012767  017460  014764          MOV     #LPRTAB,LPRPTR  ;SET UP LPR TABLE POINTER
  261
  262  002530  004567  012406          NEWLPR: JSR     R5,SETLPR       ;GO SET UP LPR CONSTANT
  263  002534  000756                          BR      NEWLIN          ;BR IF DONE ALL BAUD RATES AT THIS LINE
  264  002536  012767  177760  014752          MOV     #177760,CHRCNT  ;INIT CHAR COUNT
  265  002544  012767  177777  014746          MOV     #-1,CLSEL       ;INIT CHR LNGTH SELECT CODE
  266
  267  002552  005067  014700                  CLR     QUICKX          ;INIT QUICK TEST EXIT FLAG
  268  002556  004567  012444          NEWCL:  JSR     R5,SETCL        ;GO SET UP THE CHAR LENGTH
  269  002562  000762                          BR      NEWLPR          ;BR IF DONE ALL FOUR LENGTHS
  270  002564  005067  014732                  CLR     PARBIT          ;INIT PARITY SELECT BIT CODE
  271
  272  002570  004567  012572          NEWPAR: JSR     R5,SETPAR       ;GO SET PARITY SELECT
  273  002574  000770                          BR      NEWCL           ;BR IF DONE ALL COMBOS
  274
  275  002576  004767  011146          DHST1:  JSR     PC,DHSET1       ;GO SET UP FOR TESTING THIS LINE
  276  002602  056761  014642  000012          BIS     LINMSK,BAR(R1)  ;ACTIVATE THE SELECTED LINE
  277  002610  004767  014526                  JSR     PC,CHPS1        ;GO CLEAR PSW
  278
  279
  280  002614  012767  000100  015242          MOV     #100,TIMEA      ;INIT TIMER A
  281  002622  005067  015240                  CLR     TIMEB           ;INIT TIMER B
  282  002626  022767  000003  014670  1$:     CMP     #3,RDONE        ;BOTH RCVR AND XMITTR DONE ?
  283  002634  001435                          BEQ     3$              ;BR IF YES
  284  002636  004767  012240                  JSR     PC,TIMEIT       ;CALL THE TIMER
```

```
285  002642  000771                          BR      1$              ;TIMER STEPS AROUND THIS BRANCH IF
286                                                                  ;TIMEOUT OCCURS
287
288  002644  052777  004000  014564          BIS     #BIT11,@DHADR   ;CLEAR OUT THE DH11
289  002652  116700  015020                  MOVB    LINE,R0         ;GET LINE NO.
290  002656  006300                          ASL     R0              ;FORM TABLE INDEX
291  002660  016067  025416  176312          MOV     RTOTAL(R0),$TMP0;SAVE XMITTED COUNT
292  002666  016067  025456  176306          MOV     XTOTAL(R0),$TMP1;SAVE THE RCVD COUNT
293  002674  004567  011240                  JSR     R5,SUNUM        ;PUT LINE NO. IN MESSAGE
294  002700  017676                          LINE
295  002702  021221                          EM12+35
296  002704  012767  002714  176176          MOV     #2$,$LPERR      ;SET UP ERROR LOOP RETURN
297  002712  104012                          ERROR   12              ;LINE FAILED TO FINISH ON TIME - HUNG
298  002714  056767  014530  014530  2$:     BIS     LINMSK,DRPLIN   ;SET DROP FLAG
299  002722  012706  001100                  MOV     #STACK,SP       ;RESET STACK POINTER
300  002726  000661                          BR      NEWLIN          ;GO TRY ANOTHER LINE
301
302
303  002730  012711  004000          3$:     MOV     #BIT11,(R1)     ;CLEAR THE WORLD OUT IN THE DH11
304  002734  004767  001050                  JSR     PC,CKER         ;GO UPDATE THE DATA ERROR TABLES
305  002740  000713                          BR      NEWPAR          ;GO TRY NEXT PARITY COMBINATION
```

```
                                        ;TRANSMITTER INTERRUPT SERVICE ROUTINE ONE
                                        
002742  032711  002000        TINT1:  BIT     #BIT10,(R1)     ;NON EX MEM ERROR ??
002746  001432                        BEQ     2$              ;BR IF NOT
                                        
002750  011103                        MOV     (R1),R3         ;SAVE THE SCR
002752  004767  014400                JSR     PC,CHPS2        ;GO LOCK OUT INTRS
002756  012711  004000                MOV     #BIT11,(R1)     ;CLEAR OUT THE DH11
002762  116704  014710                MOVB    LINE,R4         ;SET UP THE S/B DATA
003766  042703  175760                BIC     #175760,R3      ;CLEAR OUT SUPERFLUOUS BITS
003772  010102                        MOV     R1,R2           ;SET UP REGADR
003774  004767  011224                JSR     PC,SUER1        ;GO SET UP ERROR INFO
003000  004567  011134                JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
003004  017676                        LINE
003006  020232                        EM1+44
003010  012767  003020  176072        MOV     #1$,$LPERR      ;SET UP THE ERROR LOOP RETURN
003016  104001                        ERROR   1               ;NON EX MEM ERROR
003020  022626              1$:       CMP     (SP)+,(SP)+     ;POP THE STACK
003022  056767  014422  014422        BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
003030  000167  177436                JMP     NEWLIN          ;GO TRY NEXT LINE
                                        
003034  011103              2$:       MOV     (R1),R3         ;GET THE SCR REG CONTENTS
003036  100433                        BMI     4$              ;BR IF XMIT DONE SET
                                        
003040  004767  014312                JSR     PC,CHPS2        ;GO LOCK OUT INTRS
003044  012711  004000                MOV     #BIT11,(R1)     ;CLEAR THE DH11 - FATAL ERROR
003050  012704  100000                MOV     #BIT15,R4       ;SET UP S/B DATA
003054  156704  014616                BISB    LINE,R4
003060  042703  077760                BIC     #77760,R3       ;CLEAR OUT SUPERFLUOUS BITS
003064  010102                        MOV     R1,R2           ;SET UP REGADR
003066  004767  011132                JSR     PC,SUER1        ;GO SET UP ERROR INFO
003072  004567  011042                JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
003076  017676                        LINE
003100  020414                        EM2+54
003102  012767  003112  176000        MOV     #3$,$LPERR      ;SET UP ERROR LOOP RETURN
003110  104002                        ERROR   2               ;XMITTR FALSE INTERRUPT
003112  022626              3$:       CMP     (SP)+,(SP)+     ;POP THE STACK
003114  056767  014330  014330        BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
003122  000167  177344                JMP     NEWLIN          ;GO TRY NEXT LINE
                                        
003126  005761  000012        4$:     TST     BAR(R1)         ;DID BAR BIT CLEAR ??
003132  001432                        BEQ     6$              ;BR IF YES
                                        
003134  004767  014216                JSR     PC,CHPS2        ;GO LOCK OUT INTRS
003140  016103  000012                MOV     BAR(R1),R3      ;GET THE LPS DATA
003144  012711  004000                MOV     #BIT11,(R1)     ;CLEAR THE DH11
003150  005004                        CLR     R4              ;SET UP S/B DATA
003152  010102                        MOV     R1,R2           ;SET UP REGADR
003154  062702  000012                ADD     #BAR,R2
003160  004767  011040                JSR     PC,SUER1        ;GO SET UP ERROR INFO
003164  004567  010750                JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
003170  017676                        LINE
003172  020474                        EM3+55
```

```
  362  003174  012767  003204  175706          MOV     #5$,$LPERR      ;SAVE THE ERROR LOOP RETURN
  363  003202  104003                          ERROR   3               ;BUFFER ACTIVE REG FAILED TO CLEAR
  364  003204  022626                  5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
  365  003206  056767  014236  014236          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
  366  003214  000167  177252                  JMP     NEWLIN          ;GO TRY NEXT LINE
  367
  368  003220  005761  000010          6$:     TST     BCR(R1)         ;DID BYTE COUNT GO TO ZERO ??
  369  003224  001432                  BEQ     8$              ;BR IF YES
  370
  371  003226  004767  014124                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
  372  003232  016103  000010                  MOV     BCR(R1),R3      ;GET THE WAS DATA
  373  003236  012711  004000                  MOV     #BIT11,(R1)     ;CLEAR THE DH11
  374  003242  005004                          CLR     R4              ;SET UP S/B DATA
  375  003244  010102                          MOV     R1,R2           ;SET UP REGADR
  376  003246  062702  000010                  ADD     #BCR,R2
  377  003252  004767  010746                  JSR     PC,SUER1        ;GO SET UP THE ERROR INFO
  378  003256  004567  010656                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
  379  003262  017676                          LINE
  380  003264  020551                          EM4+52
  381  003266  012767  003276  175614          MOV     #7$,$LPERR      ;SET UP ERROR LOOP RETURN
  382  003274  104004                          ERROR   4               ;BYTE COUNT REG FAILED TO GO TO 000000
  383  003276  022626                  7$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
  384  003300  056767  014144  014144          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
  385  003306  000167  177160                  JMP     NEWLIN          ;GO TRY NEXT LINE
  386
  387  003312  016103  000006          8$:     MOV     CAR(R1),R3      ;GET THE WAS DATA
  388  003316  016704  014174                  MOV     CHRCNT,R4       ;SET UP S/B DATA
  389  003322  005404                          NEG     R4
  390  003324  062704  030212                  ADD     #TBUF,R4
  391  003330  020304                          CMP     R3,R4           ;WAS CAR CORRECT ??
  392  003334  001425                  BEQ     10$             ;BR IF YES
  393
  394  003334  004767  014016                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
  395  003340  010102                          MOV     R1,R2           ;SET UP REGADR
  396  003342  062702  000006                  ADD     #CAR,R2
  397  003346  004767  010652                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
  398  003352  004567  010562                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
  399  003356  017676                          LINE
  400  003360  020633                          EM5+57
  401  003362  012767  003372  175520          MOV     #9$,$LPERR      ;SET UP THE ERROR RETURN
  402  003370  104005                          ERROR   5               ;CURRENT ADDRESS REG NOT CORRECT
  403  003372  022626                  9$:     CMP     (SP)+,(SP)+     ;POP THE STACK
  404  003374  056767  014050  014050          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
  405  003402  000167  177064                  JMP     NEWLIN          ;GO TRY NEXT LINE
  406
  407  003406                          10$:
  (2)  003406  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (2)  003410  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  408  003412  016703  014100                  MOV     CHRCNT,R3
  409  003416  005403                          NEG     R3              ;CHAR COUNT IN R3
  410  003420  116704  014252                  MOVB    LINE,R4         ;GET LINE NO.
  411  003424  006304                          ASL     R4              ;DOUBLE IT
  412  003426  060364  025456                  ADD     R3,XTOTAL(R4)   ;UPDATE TOTAL XMIT COUNT
  413  003432  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
```

N08


```
    (2)  003434  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
    414  003436  052767  000001  014060        BIS     #BIT0,RDONE     ;SET XMIT DONE FLAG
    415  003444  000002                        RTI                     ;RETURN TO WAIT LOOP
```

```
                                        ;RECEIVER INTERRUPT SERVICE ROUTINE ONE
003446 032711 040000        RINT1:  BIT     #BIT14,(R1)     ;SILO OVERFLOW ERROR ??
003452 001431                       BEQ     2$              ;BR IF NOT

003454 004767 013676                JSR     PC,CHPS2        ;GO LOCK OUT INTRS
003460 011103                       MOV     (R1),R3         ;GET WHS DATA
003462 012711 004000                MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
003466 042703 177760                BIC     #177760,R3      ;CLEAR JUNK
003472 116704 014200                MOVB    LINE,R4
003476 004767 010522                JSR     PC,SUER1        ;GO SET UP ERROR INFO
003502 004567 010432                JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
003506 017676                       LINE
003510 020702                       EM6+44
003512 012767 003522 175370         MOV     #1$,SLPERR      ;SET UP ERROR LOOP RETURN
003520 104006                       ERROR   6               ;SILO OVERFLOW - BAD,BAD,BAD !!!
003522 022626               1$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
003524 056767 013720 013720         BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
003532 000167 176734                JMP     NEWLIN          ;GO TRY NEXT LINE

003536 105711               2$:     TSTB    (R1)            ;CHAR AVAIL SET ??
003540 100434                       BMI     4$              ;BR IF YES

003542 004767 013610                JSR     PC,CHPS2        ;GO LOCK OUT INTRS
003546 011103                       MOV     (R1),R3         ;GET WHS DATA
003550 042703 177560                BIC     #177560,R3      ;CLEAN IT UP
003554 012711 004000                MOV     #BIT11,(R1)     ;NOW CLEAR DH11
003560 012704 000200                MOV     #BIT07,R4       ;SET UP S/B DATA
003564 136704 014106                BISB    LINE,R4
003570 010102                       MOV     R1,R2           ;SET UP REGADR
003572 004767 010426                JSR     PC,SUER1        ;GO SET UP ERROR INFO
003576 004567 010336                JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
003602 017676                       LINE
003604 020750                       EM7+51
003606 012767 003616 175274         MOV     #3$,SLPERR      ;SET UP THE ERROR LOOP RETURN
003614 104007                       ERROR   7               ;RECEIVER FALSE INTERRUPT
003616 022626               3$:     CMP     (SP)+,(SP)+     ;POP GOES THE SP
003620 056767 013624 013624         BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
003626 000167 176640                JMP     NEWLINE         ;GO TRY NEXT LINE

003632 016167 000002 175340 4$:     MOV     NRC(R1),STMP0   ;SAVE THE DATA RECEIVED
003640 100431                       BMI     6$              ;BR IF IT WAS VALID DATA

003642 004767 013510                JSR     PC,CHPS2        ;GO LOCK OUT INTRS
003646 012711 004000                MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
003652 162767 025732 022046         SUB     #RBUF,RBFPTR    ;WHICH CHAR WAS IT ??
003660 016702 022042                MOV     RBFPTR,R2       ;SAVE CHAR NUMBER
003664 004767 010330                JSR     PC,SUER2        ;GO SET UP ERROR INFO
003670 004567 010244                JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
003674 017676                       LINE
003676 021026                       EM10+45
003700 012767 003710 175202         MOV     #5$,SLPERR      ;SET UP ERROR RETURN
003706 104010                       ERROR   10              ;RECEIVED INVALID DATA
003710 022626               5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
```

```
  472  003712  056767  013532  013532          BIS     LINMSK,DRPLIN   ;SET THE DROPPED FLAG FOR THIS LINE
  473  003720  000167  176546                  JMP     NEWLIN          ;GO TRY ANOTHER LINE
  474
  475  003724                          6$:
(2)    003724  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(2)    003726  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  476  003730  016777  175244  021770          MOV     $TMPO,@RBFPTR   ;STORE CHAR IN THE BUFFER
  477  003736  062767  000002  021762          ADD     #2,RBFPTR       ;UPDATE THE POINTER
  478  003744  026767  013556  021754          CMP     RBFEND,RBFPTR   ;END OF BUFFER ??
  479  003752  001013                          BNE     7$              ;BR IF NOT
  480  003754  016703  013536                  MOV     CHRCNT,R3       ;GET CHAR COUNT
  481  003760  005403                          NEG     R3
  482  003762  116704  013710                  MOVB    LINE,R4         ;GET THE LINE NO.
  483  003766  006304                          ASL     R4              ;DOUBLE IT
  484  003770  060364  025416                  ADD     R3,RTOTAL(R4)   ;UPDATE TOTAL RECEIVED COUNT
  485  003774  052767  000002  013522          BIS     #BIT1,RDONE     ;SET THE RCVR DONE FLAG
  486  004002                          7$:
(2)    004002  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
(2)    004004  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
  487  004006  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

004010  012767  025732  021710  CKER:   MOV     #RBUF,RBFPTR    ;SET UP POINTERS
004016  012767  030212  021704          MOV     #TBUF,TBFPTR
004024  012767  025516  021664          MOV     #DATERR,DEPTR   ;SET UP POINTERS TO STATISTICS TABLES
004032  012767  025556  021660          MOV     #PARERR,PEPTR
004040  012767  025616  021654          MOV     #OVRERR,ORPTR
004046  012767  025656  021650          MOV     #FRMERR,FRPTR
004054  116705  013616                  MOVB    LINE,R5         ;GET LINE NO. AND DOUBLE IT
004060  006305                          ASL     R5
004062  060567  021630                  ADD     R5,DEPTR        ;POINT TO CORRECT LINE ENTRY IN TABLE
004066  060567  021626                  ADD     R5,PEPTR
004072  060567  021624                  ADD     R5,ORPTR
004076  060567  021622                  ADD     R5,FRPTR

004102  117704  021622          1$:     MOVB    @TBFPTR,R4      ;GET THE S/B DATA
004106  000304                          SWAB    R4              ;PUT LINE NO. IN HIGH BYTE
004110  105004                          CLRB    R4
004112  156704  013560                  BISB    LINE,R4
004116  000304                          SWAB    R4
004120  052704  100000                  BIS     #BIT15,R4       ;AND FINALLY THE VALID DATA BIT
004124  017703  021576                  MOV     @RBFPTR,R3      ;GET THE WAS DATA
004130  020304                          CMP     R3,R4           ;WAS = S/B ?????
004132  001435                          BEQ     3$              ;BR IF YES

004134  010367  175056                  MOV     R3,$TMP7        ;SAVE THE WAS DATA
004140  010146                          MOV     R1,-(SP)        ;SAVE THE DEVADR
004142  016701  021560                  MOV     RBFPTR,R1       ;GET THE SERADR
004146  016702  021556                  MOV     TBFPTR,R2       ;GET THE WRSADR
004152  010200                          MOV     R2,R0           ;GET XMIT BUFFER ADDR
004154  162700  030212                  SUB     #TBUF,R0        ;GENERATE CHAR #
004160  004767  000066                  JSR     PC,UPDER        ;GO CHECK AND UPDATE THE DATA ERROR TABLE
004164  004767  010030                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
004170  004567  007744                  JSR     R5,SUNUM        ;GO PUT LINE NO. IN MSG
004174  017676                          LINE
004176  021161                          EM11+23
004200  012767  004210  174702          MOV     #2$,$LPERR      ;SET UP ERROR RETURN
004206  104011                          ERROR   11              ;DATA COMPARE ERROR OR PARITY,FRAMING
                                                                ;OR OVERRUN
004210  012601                  2$:     MOV     (SP)+,R1        ;RESTORE THE DEVADR
004212  032767  070000  174776          BIT     #70000,$TMP7    ;ANY PARITY,OVERRUN, OR FRAMING ERROR
004220  001402                          BEQ     3$              ;BR IF NOT

004222  004767  000036                  JSR     PC,SOFT         ;GO TAKE CARE OF SOFT ERROR REPORT

004226  005267  021476          3$:     INC     TBFPTR          ;UPDATE POINTERS
004232  062767  000002  021466          ADD     #2,RBFPTR
004240  026767  013262  021460          CMP     RBFEND,RBFPTR   ;COMPARED ALL CHARS ??
004246  001315                          BNE     1$              ;BR IF NOT

004250  000207                          RTS     PC              ;RETURN TO WAIT LOOP

004252  120304                  UPDER:  CMPB    R3,R4           ;DATA BYTES CORRECT ??
```

E09


```
                                                BEQ     1$              ;BR IF YES
                                                INC     @DEPTR          ;COUNT THE DATA ERROR
                                        1$:     RTS     PC              ;RETURN

                                        SOFT:   ASL     $TMP7           ;TEST FOR OVERRUN ERRORS
                                                BPL     1$              ;BR IF NONE
                                                INC     @ORPTR          ;COUNT IT
                                        1$:     ASL     $TMP7           ;TEST FOR FRAMING ERRORS
                                                BPL     2$              ;BR IF NONE
                                                INC     @FRPTR          ;COUNT IT
                                        2$:     ASL     $TMP7           ;TEST FOR PARITY ERRORS
                                                BPL     3$              ;BR IF NONE
                                                INC     @PEPTR          ;COUNT IT
                                        3$:     RTS     PC              ;RETURN
```

```
561                                     ;THIS ROUTINE IS CALLED TO PRINT OUT THE TEST STATISTICS
562
563  004324  012767  000001  013116  PRSTAT: MOV     #1,LINMSK       ;SET UP BIT TEST MARKER
564  004332  005001                          CLR     R1              ;R1 CONTAINS THE LINE NO.
565  004334  004567  007600                  JSR     R5,SUNUM        ;GO SET UP DH11 # IN STAT MESSAGE
566
567  004340  017674                          DHNUM
568  004342  023262                          STMSG1+6
569  004344  104400                          TYPE                    ;GO TYPE THE STATISTICS HEADER
570  004346  023254                          STMSG1
571  004350  104400                          TYPE                    ;TYPE HEADER
572  004352  023357                          STMSG4
573
574  004354  036767  013070  013070  1$:     BIT     LINMSK,DRPLIN   ;DID THIS LINE GET DROPPED ?
575  004362  001411                          BEQ     2$              ;BR IF NOT
576
577  004364  010167  174626                  MOV     R1,$TMP7        ;SAVE THE LINE NO.
578  004370  004567  007544                  JSR     R5,SUNUM        ;GO PUT LINE NO. IN MESSAGE
579  004374  001216                          $TMP7
580  004376  023336                          STMSG3+10
581  004400  104400                          TYPE
582  004402  023326                          STMSG3
583  004404  000436                          BR      3$              ;GO TEST NEXT LINE
584
585  004406  010102                  2$:     MOV     R1,R2           ;SET UP R2 WITH TABLE INDEX
586  004410  006302                          ASL     R2
587  004412  036767  013032  013026          BIT     LINMSK,LINSEL   ;WAS THIS LINE SELECTED ??
588  004420  001430                          BEQ     3$              ;BR IF NOT
589  004422  010167  174552                  MOV     R1,$TMP0        ;SET UP THE ERROR INFORMATION FROM
                                                                     ;THE TABLES INTO THE MESSAGE POINTERS
590  004426  016267  025416  174546          MOV     RTOTAL(R2),$TMP1
591  004434  016267  025456  174542          MOV     XTOTAL(R2),$TMP2
592  004442  016267  025516  174536          MOV     DATERR(R2),$TMP3
593  004450  016267  025556  174532          MOV     PARERR(R2),$TMP4
594  004456  016267  025616  174526          MOV     OVRERR(R2),$TMP5
595  004464  016267  025636  174522          MOV     FRMERR(R2),$TMP6
596  004472  012767  004502  174410          MOV     #3$,SLPERR      ;RETURN TO 3$ AFTER PRINTING LINE
597  004500  104013                          ERROR   13
598  004502  005201                  3$:     INC     R1              ;STEP TO NEXT LINE
599  004504  006367  012740                  ASL     LINMSK          ;SHIFT THE MARKER
600  004510  001401                          BEQ     .+4             ;BR IF ALL LINES REPORTED
601  004512  000720                          BR      1$              ;GO BACK AND DO THIS LINE
602
603
```

```
  607  004514  000004                      ENDA:   SCOPE
  608  004516  105067  174360                      CLRB    $TSTNM          ;RE-INIT TEST NUMBER FOR NEXT PASS
  609  004522  012767  000240  004134              MOV     #240,SEOP       ;NOP THE SCOPE IN ENDPASS ROUTINE
  610  004530  005267  013140                      INC     DHNUM           ;GENERATE NEW DH11 NUMBER
  611  004534  062767  000002  013312              ADD     #2,ADPTR        ;UPDATE THE TABLE POINTERS
  612  004542  062767  000002  013306              ADD     #2,VCPTR
  613  004550  062767  000002  013302              ADD     #2,BRPTR
  614  004556  006367  012660                      ASL     SELMSK          ;SHIFT MARKER TO TEST NEXT DH11
  615  004562  001002                              BNE     1$              ;BR IF NOT TESTED ALL DH11'S
  616  004564  000167  004074                      JMP     $EOP            ;JUMP TO EOP IF WE HAVE
  617  004570  036767  012646  012646  1$:         BIT     SELMSK,DHSEL    ;IS THIS DH11 SELECTED ?
  618  004576  001746                              BEQ     ENDA            ;BR IF NOT
       004600  000167  175526                      JMP     RSTRTA          ;GO TEST THIS DH11
```

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

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

004604  012767  177777  013272  ECHO:   MOV     #-1,RETFLG      ;SET RETURN FLAG - COME BACK
004612  005067  013254          ECHO:   CLR     DPFLG           ;CLEAR PATTERNS TEST FLAG
004616  005067  013006          ECHO:   CLR     VCFLG           ;INIT VECTOR SETUP FLAG
004622  000167  175014          ECHO:   JMP     BEGINA          ;TO "ECHO1" AFTER SETUP

004626  012767  160020  012602  ECHO1:  MOV     #160020,DHADR   ;SET UP DH11 DEFAULT ADDRESS
004634  012767  000330  012576  ECHO1:  MOV     #330,DHVCT      ;SET UP DH11 DEFAULT VECTOR
004642  104400          ECHO1:  TYPE                            ;PRINT I.D. MESSAGE
004644  023467          ECHO1:  FCMSG1                          ;"SINGLE LINE ECHO TEST - CONNECT
                                                                ;TERMINAL TO TEST LINE"
004646  000167  007506          ECHO1:  JMP     INPAR           ;GO SET UP DEVICE AND VECTOR
                                                                ;ADDRESSES - COME BACK TO "ECHO2"

004652  104400          ECHO2:  TYPE                            ;GO ASK FOR TTY INPUT
004654  023566          ECHO2:  ECMSG2                          ;"LINE # (00 - 17 OCTAL)"
004656  104407          ECHO2:  RDOCT                           ;INPUT LINE NO. FM TTY
004660  012667  013012          ECHO2:  MOV     (SP)+,LINE      ;GET NO. TYPED
004664  042767  177760  013004  ECHO2:  BIC     #177760,LINE    ;CLEAR JUNK
004672  016702  013000          ECHO2:  MOV     LINE,R2         ;GET LINE NO.
004676  005202          ECHO2:  INC     R2                      ;CORRECT FOR SHIFT ROUTINE
004700  012767  000001  012542  ECHO2:  MOV     #1,LINMSK       ;INIT LINE SELECT BIT MASK
004706  005302          1$:     DEC     R2                      ;COUNT ONE LINE CHECKED
004710  001403          1$:     BEQ     2$                      ;BR IF DONE
004712  006367  012532          1$:     ASL     LINMSK          ;SHIFT SELECT BIT
004716  000773          1$:     BR      1$                      ;GO COUNT IT
004720  004767  011222  2$:     JSR     PC,LPRIN                ;GO ASK FOR AND SET UP LINE PARAMETERS

004724  005767  013142          TST     DPFLG                   ;DATA PATTERNS TEST ?
004730  001401          3$:     BEQ     3$                      ;BR IF NOT
004732  000207          3$:     RTS     PC                      ;RETURN TO PATTERNS TEST

00473-  105067  016520  3$:     CLRB    EC2                     ;CLEAR ECHO BUFFER
004740  104400          3$:     TYPE
004742  024610          3$:     SNMSG1                          ;"SEND MODE - Y OR N ?"
004744  104405  4$:     RDCHR                                   ;GET CHAR TYPED
004746  012600          4$:     MOV     (SP)+,R0                ;GET CHAR TYPED
004750  122700  000015          4$:     CMPB    #15,R0          ;WAS IT A <CR> ?
004754  001405          4$:     BEQ     5$                      ;BR IF YES
004756  110067  016476          4$:     MOVB    R0,EC2          ;ECHO WHAT WAS TYPED
004762  104400          4$:     TYPE
004764  023460          4$:     EC2
004766  000766          4$:     BR      4$                      ;GO WAIT FOR TERMINATOR
004770  105767  016464  5$:     TSTB    EC2                     ;<CR> ONLY ??
004774  001412          5$:     BEQ     ECHO3                   ;BR IF YES
004776  122767  000116  016454  5$:     CMPB    #116,EC2        ;WAS IT AN "N" ??
005004  001406          5$:     BEQ     ECHO3                   ;BR IF YES
005006  122767  000131  016444  5$:     CMPB    #131,EC2        ;WAS IT A "Y" ??
005014  001347          5$:     BNE     3$                      ;BR IF NOT ASK AGAIN
```

```
  675  005016  000167  000574                JMP     SENDP1          ;GO TO SEND ROUTINE
  676
```

```
                                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     #100,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


```

```
                                        ;TRANSMITTER INTERRUPT SERVICE ROUTINE TWO
                                        
                                        
                        042711  120000  TINT2:  BIC     #BIT15+BIT13,(R1)       ;DISABLE XMIT INTRS
                        022767  000001  012656          CMP     #1,EXFLAG       ;CONTROL-C FLAG ?
                        001437          BEQ     2$              ;BR IF YES
                        022767  000003  012646          CMP     #3,EXFLAG       ;WAS BUFFER JUST DUMPED ?
                        001437          BEQ     3$              ;BR IF YES
                        022767  000002  012636          CMP     #2,EXFLAG       ;CONTROL-E FLAG ?
                        001453          BEQ     1$              ;BR IF YES
                            7   03'342                   CMP     R2,#TBUF+[illegible]    ;BUFFER FULL ?
                            54          BLT     3$              ;BR IF NOT
                        012767  000003  012620  1$:     MOV     #3,EXFLAG       ;SET DUMP FLAG
                                030212          SUB     #TBUF,R2        ;SET UP BYTE COUNT REG
                                                NEG     R2
                            I   000010          MOV     R2,BCR(R1)
                        016761  030212  000006          MOV     #TBUF,CAR(R1)   ;SET UP CURRENT ADDR REG
                        012767  000100  012540          MOV     #100,TIMEA      ;INIT TIMER
                        052711  020000                  BIS     #BIT13,(R1)     ;ENABLE XMITTR INTR
                        016761  012114  000012          MOV     LINMSK,BAR(R1)  ;ACTIVATE LINE
                        000415                  BR      4$              ;GO EXIT
                                
                    740 012767  177777  012522  2$:     MOV     #-1,CEXIT       ;SET CONTROL-C EXIT
                        000411          BR      4$              ;GO EXIT
                                
                        012702  030212  3$:     MOV     #TBUF,R2        ;RESET ECHO BUFFER PIOPINTER
                        005067  012536          CLR     EXFLAG          ;INIT EXIT FLAG
                        012767  000100  012476  31$:    MOV     #100,TIMEA      ;INIT TIMER AGAIN
                        052711  000100          BIS     #BIT06,(R1)     ;ENABLE RCVR INTR
                                
005372  000002                  4$:     RTI                             ;RETURN TO MAINLINE
```

```
                                        ;RECEIVER INTERRUPT SERVICE ROUTINE TWO
                                        
005374 042711 000100          RINT2:  BIC     #BIT06,(R1)             ;DISABLE RCVR INTR
005400 016167 000000  173572          MOV     RBUF(R1),STMP0          ;SAVE THE DATA TYPED
005406 042767 177600  173564          BIC     #177600,STMP0           ;CLEAR HIGH BYTE
005414 022767 000003  173556          CMP     #3,STMP0                ;CONTROL-C TYPED ??
005422 001015                         BNE     1$                      ;BR IF NOT
                                        
005424 112767 000143  012466          MOVB    #143,ECBUF              ;SET UP TO ECHO CONTROL-C
005432 112767 000103  012461          MOVB    #103,ECBUF+1
005440 012761 177776  000010          MOV     #-2,BCR(R1)               ;T UP BCR REG
005446 012767 000001  012442          MOV     #1,EXFLAG               ;SET CONTROL-C FLAG
005454 000444                         BR      4$                      ;GO OUT PUT CHAR TYPED
                                        
005456 022767 000005  173514  1$:     CMP     #5,STMP0                ;WAS IT A CONTROL-E ??
005464 001021                         BNE     2$                      ;BR IF NOT
                                        
005466 112767 000136  012424          MOVB    #136,ECBUF              ;SET UP TO ECHO CONTROL-E
005474 112767 000105  012417          MOVB    #105,ECBUF+1
005502 112722 000136                  MOVB    #136,(R2)+              ;PUT IN THE ECHO BUFFER
005506 112722 000105                  MOVB    #105,(R2)+
005512 012761 177776  000010          MOV     #-2,BCR(R1)             ;SET UP BYTE COUNT REG
005520 012767 000002  012370          MOV     #2,EXFLF ;              ;SET CONTROL-E FLAG
005526 000417                         BR      4$                      ;GO EXIT
                                        
005530 022767 000012  173442  2$:     CMP     #12,STMP0               ;WAS IT A LINE FEED ??
005536 001003                         BNE     3$                      ;BR IF NOT
005540 004767 010076                  JSR     PC,LDFILL               ;GO LOAD FILLERS
005544 000410                         BR      4$                      ;GO EXIT
                                        
005546 116767 173426  012344  3$:     MOVB    STMP0,ECBUF             ;SET UP CHAR TO ECHO
005554 116722 173420                  MOVB    STMP0,(R2)+
005560 012761 177777  000010          MOV     #-1,BCR(R1)             ;OUTPUT ONE CHAR ONLY
005566 012761 020120  000006  4$:     MOV     #ECBUF,CAR(R1)          ;SET UP CURRENT ADDR REG
005574 012767 000100  012262          MOV     #100,TIMEA              ;INIT TIMER AGAIN
005602 052711 020000                  BIS     #BIT13,(R1)             ;ENABLE XMITTR INTR
005606 016761 011636  000012          MOV     LINMSK,BAR(R1)          ;ACTIVATE THE LINE
005614 000002                         RTI                             ;RETURN TO MAINLINE
```

```
005616  104400                  SENDP1: TYPE                    ;ASK FOR DIRECTIONS
005620  024640                          SNMSG2                  ;"TYPE SEND BUFFER - TERMINATE WITH CONTROL-C"
005622  012705  030212                  MOV     #TBUF,R5        ;SET UP BUFFER POINTER
005626  104405                  1$:     RDCHR                   ;GET CHAR
005630  012600                          MOV     (SP)+,R0
005632  110067  015622                  MOVB    R0,EC2          ;ECHO CHAR
005636  104400                          TYPE
005640  023460                          EC2
005642  022700  000003                  CMP     #3,R0           ;WAS IT A CONTROL-C ??
005646  001421                          BEQ     4$              ;BR IF YES

005650  026727  011642  000400          CMP     CHRCNT,#256.    ;BUFFER FULL ??
005656  003015                          BGT     4$              ;BR IF YES
005660  022700  000012                  CMP     #12,R0          ;WAS IT A LINE FEED ?
005664  001010                          BNE     3$              ;BR IF NOT

005666  110025                          MOVB    R0,(R5)+        ;LOAD CHAR TYPED
005670  116704  012270                  MOVB    FILLB,R4        ;GET FILLER COUNT
005674  116725  012262          2$:     MOVB    FILLA,(R5)+     ;LOAD A FILLER
005700  005304                          DEC     R4              ;COUNT IT
005702  001374                          BNE     2$              ;BR IF NOT DONE
005704  000750                          BR      1$              ;GET SOME MORE INPUT

005706  110025                  3$:     MOVB    R0,(R5)+        ;LOAD BUFFER
005710  000746                          BR      1$              ;GO GET SOME MORE

005712  004767  007646          4$:     JSR     PC,SENDP2       ;GO XMIT THE BUFFER
005716  105067  015536          5$:     CLRB    EC2             ;CLEAR ECHO BUFFER
005722  104400                          TYPE
005724  024720                          SNMSG3                  ;"CHANGE PARAMETERS- Y OR N"
005726  104405                  6$:     RDCHR
005730  012600                          MOV     (SP)+,R0        ;GET CHAR
005732  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR> HE TYPED ??
005736  001405                          BEQ     7$              ;BR IF IT WAS
005740  110067  015514                  MOVB    R0,EC2          ;ECHO IT
005744  104400                          TYPE
005746  023460                          EC2
005750  000766                          BR      6$              ;GO WAIT FOR TERMINATOR

005752  105767  015502          7$:     TSTB    EC2             ;<CR> ONLY ??
005756  001717                          BEQ     SENDP1          ;BR IF YES
005760  122767  000116  015472          CMPB    #116,EC2        ;DID HE SAY NO ??
005766  001713                          BEQ     SENDP1          ;BR IF HE DID
005770  122767  000131  015462          CMPB    #131,EC2        ;DID HE SAY YES ??
005776  001347                          BNE     5$              ;GO ASK ALL OVER AGAIN
006000  000167  176646                  JMP     ECHO2           ;GO ASK FOR NEW PARAMETERS
```

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

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

006004 012767 177777 012060 EXPAT:  MOV   #-1,DPFLG       ;SET PATTERNS TEST FLAG
006012 005067 012066        CLR     RETFLG         ;CLR ECHO TESTS FLAG
006016 005067 011606        CLR     VCFLG          ;CLEAR VECTOR SETUP FLAG
006022 000167 173614        JMP     BEGINA         ;GO SET UP RETURN TO "EXPAT1"

006026 012767 160020 011402 EXPAT1: MOV   #160020,DHADR    ;SET UP DEFAULT DH11 ADDR
006034 012767 000330 011376         MOV   #330,DHVCT       ;AND VECTOR TOO
006042 104400                       TYPE
006044 024757                       DPMSG1                 ;"DATA PATTERNS TESTS - CONNECT TEST JUMPAR"
006046 000167 006306                JMP   INPAR            ;GO GET SOME PARAMETERS RETURN TO EXPAT2

006052 004767 176574 EXPAT2: JSR    PC,ECHO2       ;GO GET REST OF THE PARAMETERS
006056 004767 011206         JSR    PC,SUCLMK      ;GO SET UP CHAR LENGTH MASK
006062 104400                1$:    TYPE
006064 025035                       DPMSG2                 ;"BUFFER SIZE ? (1-512)
006066 104410                       RDOEC                  ;GET THE SIZE TYPED
006070 012600                       MOV   (SP)+,R0
006072 001406                       BEQ   2$               ;BR IF DEFAULT TO 256. <CR>

006074 020027 001001                CMP   R0,#513.         ;TOO BIG ?
006100 002405                       BLT   3$               ;BR IF NOT
006102 104400                       TYPE
006104 025067                       DPMSG3                 ;"INVALID SIZE - TRY AGAIN"
006106 000765                       BR    1$               ;GO ASK AGAIN

006110 012700 000400        2$:     MOV   #256.,R0         ;DEFAULT TO 256. BYTE BUFFER
006114 005400               3$:     NEG   R0               ;MAKE IT NEG BYTE COUNT
006116 010067 011374                MOV   R0,CHRCNT        ;SAVE IT FOR TEST

006122 012767 120240 011542         MOV   #120240,DHRLVL   ;SET BR LEVELS TO 6.
006130 016700 011304                MOV   DHVCT,R0         ;SET UP VECTORS
006134 012720 010162                MOV   #RINT3,(R0)+
006140 116710 011526                MOVB  DHRLVL,(R0)
006144 005720                       TST   (R0)+
006146 012720 007540                MOV   #TINT3,(R0)+
006152 116710 011515                MOVB  DHTLVL,(R0)
```

```
 890  006156  005067  011716          EXPAT3: CLR     PATFLG          ;CLEAR <CR> SEQUENCE FLAG
 891  006162  105067  015272                  CLRB    EC2             ;CLEAR ECHO BUFFER
 892  006166  016701  011244                  MOV     DHADR,R1        ;INIT R1 TO POINT TO SCR REG
 893  006172  104400                         TYPE
 894  006174  025124                         DPMSG4                  ;"PATTERN TYPE ? (A,U,D,R,S, OR B)"
 895  006176  104405                  7$:    RDCHR
 896  006200  012600                         MOV     (SP)+,R0        ;GET WHAT HE TYPED
 897  006202  120027  000015                CMPB    R0,#15          ;WAS IT A <CR> ??
 898  006206  001407                         BEQ     9$              ;BR IF YES
 899  006210  010067  011662                 MOV     R0,DATPAT
 900  006214  110067  015240                 MOVB    R0,EC2          ;ECHO IT
 901  006220  104400                         TYPE
 902  006222  023460                         EC2
 903  006224  000764                         BR      7$              ;GO WAIT FOR TERMINATOR
 904
 905  006226  104400                  9$:    TYPE
 906  006230  025176                         DPMSG5                  ;"SET SR07=1 TO LOCK ON TEST PATTERN"
 907  006232  105767  015222                 TSTB    EC2             ;<CR> ONLY ??
 908  006236  001005                         BNE     8$              ;BR IF NOT
 909  006240  012767  000015  011632         MOV     #15,PATFLG
 910  006246  000167  000506                 JMP     DPATCR          ;GO SEQUENCE A,U,D,R PATTERNS
 911
 912  006252  022767  000101  011616  8$:    CMP     #101,DATPAT     ;ALTERNATING 1/0 ?
 913  006260  001002                         BNE     1$              ;BR IF NOT
 914  006262  000167  000102                 JMP     DPATA           ;GO DO IT
 915
 916  006266  022767  000125  011602  1$:    CMP     #125,DATPAT     ;UP COUNT PATTERN ?
 917  006274  001002                         BNE     2$              ;BR IF NOT
 918  006276  000167  000164                 JMP     DPATU           ;GO DO IT
 919
 920  006302  022767  000104  011566  2$:    CMP     #104,DATPAT     ;DOWN COUNT PATTERN ?
 921  006310  001002                         BNE     3$              ;BR IF NOT
 922  006312  000167  000246                 JMP     DPATD           ;GO DO IT

 923  006316  022767  000122  011552  3$:    CMP     #122,DATPAT     ;RANDOM PATTERN ?
 924  006324  001002                         BNE     4$              ;BR IF NOT
 925  006326  000167  000330                 JMP     DPATR           ;GO DO IT

 926  006332  022767  000123  011536  4$:    CMP     #123,DATPAT     ;SINGLE CHAR PATTERN ?
 927  006340  001002                         BNE     5$              ;BR IF NOT
 928  006342  000167  000474                 JMP     DPATS           ;GO DO IT

 929  006346  022767  000102  011522  5$:    CMP     #102,DATPAT     ;TYPE IN BUFFER ?
 930  006354  001002                         BNE     6$              ;BR IF NOT
 931  006356  000167  000620                 JMP     DPATB           ;GO DO IT

 932  006362  104400                  6$:    TYPE
 933  006364  025240                         DPMSG6                  ;"INVALID PATTERN - TRY AGAIN"
 934  006366  000673                         BR      EXPAT3          ;GO ASK AGAIN
 935
```

```
                                ;
006370  005067  011500          DPATA:  CLR     DATCNT          ;INIT ITERATION COUNTER
006374  004767  010452          1$:     JSR     PC,SUPATA       ;GO SET UP THE PATTERN
006400  004767  001016                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
006404  005267  011464                  INC     DATCNT          ;COUNT IT
006410  026767  011472  011456          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES
006416  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
                                ;
006420  016767  011450  172532          MOV     DATCNT,SREG0    ;SAVE ITERATION COUNT
006426  005067  011442                  CLR     DATCNT          ;INIT COUNTER
006432  012767  006442  172450          MOV     #2$,SLPERR      ;COME BACK TO 2$
006440  104017                          ERROR   17              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
006442  022767  000015  011430  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
006450  001001                          BNE     3$              ;BR IF NOT
006452  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
006454  105777  172456          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
006460  100745                          BMI     1$              ;BR IF YES
006462  000167  177470                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
                                ;
006466  005067  011402          DPATU:  CLR     DATCNT          ;INIT ITERATION COUNTER
006472  004767  010402          1$:     JSR     PC,SUPATU       ;GO SET UP THE PATTERN
006476  004767  000720                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
006502  005267  011366                  INC     DATCNT          ;COUNT IT
006506  026767  011374  011360          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES ?
006514  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
                                ;
006516  016767  011352  172434          MOV     DATCNT,SREG0    ;SAVE ITERATION COUNT
006524  005067  011344                  CLR     DATCNT          ;INIT COUNTER
006530  012767  006540  172352          MOV     #2$,SLPERR      ;COME BACK TO 2$
006536  104020                          ERROR   20              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
006540  022767  000015  011332  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
006546  001001                          BNE     3$              ;BR IF NOT
006550  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
006552  105777  172360          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
006556  100745                          BMI     1$              ;BR IF YES
006560  000167  177372                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
                                ;
006564  005067  011304          DPATD:  CLR     DATCNT          ;INIT ITERATION COUNTER
006570  004767  010334          1$:     JSR     PC,SUPATD       ;GO SET UP THE PATTERN
006574  004767  000622                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
006600  005267  011270                  INC     DATCNT          ;COUNT IT
006604  026767  011276  011262          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES
006612  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
                                ;
006614  016767  011254  172336          MOV     DATCNT,SREG0    ;SAVE ITERATION COUNT
006622  005067  011246                  CLR     DATCNT          ;INIT COUNTER
006626  012767  006636  172254          MOV     #2$,SLPERR      ;COME BACK TO 2$
006634  104021                          ERROR   21              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
006636  022767  000015  011234  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
006644  001001                          BNE     3$              ;BR IF NOT
006646  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
006650  105777  172262          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
006654  100745                          BMI     1$              ;BR IF YES
006656  000167  177274                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
                                ;
```

```
  996  006662  005067  011206          DPATR:  CLR     DATCNT          ;INIT ITERATION COUNTER
  997  006666  004767  010270          1$:     JSR     PC,SUPATR       ;GO SET UP THE PATTERN
  998  006672  004767  000524                  JSR     PC,DHST2        ;GO EXECUTE IT ON SELECTED DH11
  999  006676  005267  011172                  INC     DATCNT          ;COUNT IT
 1000  006702  026767  011200  011164          CMP     PATLIM,DATCNT   ;DONE IT ENOUGH TIMES
 1001  006710  001366                          BNE     1$              ;BR IF NOT DO IT AGAIN
 1002
 1003  006712  016767  011156  172240          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
 1004  006720  005067  011150                  CLR     DATCNT          ;INIT COUNTER
 1005  006724  012767  006734  172156          MOV     #2$,$LPERR      ;COME BACK TO 2$
 1006  006732  104022                          ERROR   22              ;REPORT DONE SPECIFIED NO. OF ITERATIONS
 1007  006734  022767  000015  011136  2$:     CMP     #15,PATFLG      ;CYCLING FOUR PATTERNS ?
 1008  006742  001001                          BNE     3$              ;BR IF NOT
 1009  006744  000207                          RTS     PC              ;RETURN TO EXECUTE NEXT PATTERN
 1010  006746  105777  172164          3$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
 1011  006752  100745                          BMI     1$              ;BR IF YES
 1012  006754  000167  177176                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERNS
 1013
 1014  006760  012767  000101  011110  DPATCR: MOV     #101,DATPAT     ;FLAG I/O PATTERN
 1015  006766  004767  177376                  JSR     PC,DPATA        ;CALL FOR I/O PATTERN
 1016  006772  012767  000125  011076          MOV     #125,DATPAT     ;FLAG UP COUNT PATTERN
 1017  007000  004767  177462                  JSR     PC,DPATU        ;CALL FOR UP COUNT PATTERN
 1018  007004  012767  000104  011064          MOV     #104,DATPAT     ;FLAG DOWN COUNT PATTERN
 1019  007012  004767  177546                  JSR     PC,DPATD        ;CALL FOR DOWN COUNT PATTERN
 1020  007016  012767  000122  011052          MOV     #122,DATPAT     ;FLAG RANDOM DATA PATTERN
 1021  007024  004767  177632                  JSR     PC,DPATR        ;CALL FO RANDOM PATTERN
 1022  007030  105777  172102                  TSTB    @SWR            ;LOCK ON ALL FOUR PATTERNS
 1023  007034  100751                          BMI     DPATCR          ;BR IF YES
 1024  007036  000167  177114                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
 1025
 1026  007042  105067  014412          DPATS:  CLRB    EC2             ;CLEAR THE ECHO BUFFER
 1027  007046  005067  011022                  CLR     DATCNT          ;INIT ITERATION COUNTER
 1028  007052  104400                          TYPE
 1029  007054  025300                          DPMSG7                  ;"TYPE SINGLE TEST CHAR"
 1030  007056  104405          3$:     RDCHR                           ;GET CHAR
 1031  007060  012600                          MOV     (SP)+,R0        ;GET WHAT HE TYPED
 1032  007062  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR> ??
 1033  007066  001407                          BEQ     4$              ;BR IF YES
 1034  007070  010067  011006                  MOV     R0,SINGLE       ;SAVE IT FOR LOADING BUFFER
 1035  007074  110067  014360                  MOVB    R0,EC2          ;ECHO IT ON TTY
 1036  007100  104400                          TYPE
 1037  007102  023460                          EC2
 1038  007104  000764                          BR      3$              ;GO WAIT FOR TERMINATOR
 1039
 1040  007106  105767  014346          4$:     TSTB    EC2             ;WAS SINGLE CHAR A <CR> ??
 1041  007112  001003                          BNE     1$              ;BR IF NOT A <CR> ONLY
 1042  007114  012767  000015  010760          MOV     #15,SINGLE      ;SET UP TO LOAD ALL <CR>'S
 1043  007122  004767  010114          1$:     JSR     PC,SUPATS       ;GO SET IT UP IN BUFFER
 1044  007126  004767  000270                  JSR     PC,DHST2        ;GO EXECUTE IT ON DH11
 1045  007132  005267  010736                  INC     DATCNT          ;COUNT ONE TIME
 1046  007136  026767  010744  010730          CMP     PATLIM,DATCNT   ;DONE REQUIRED ITERATIONS ?
 1047  007144  001366                          BNE     1$              ;BR IF NOT
 1048
 1049  007146  016767  010722  172004          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
```

```
1050  007154  005067  010714                  CLR     DATCNT          ;INIT ITERATION COUNTER
1051  007160  012767  007170  171722          MOV     #2$,SLPERR      ;COME BACK TO 2$ ALWAYS
1052  007166  104023                          ERROR   23              ;REPORT DONE SINGLE CHAR PATTERN
1053  007170  105777  171742          2$:     TSTB    @SWR            ;LOCK ON THIS PATTERN ??
1054  007174  100752                          BMI     1$              ;BR IF YES
1055  007176  000167  176754                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
1056
```

```
 1059  007202  005067  010666          DPATB:  CLR     DATCNT          ;INIT ITERATION COUNTER
 1060  007206  104400                          TYPE
 1061  007210  025333                          DPMSGA                  ;"TYPE IN BUFFER - TERMINATE WITH CONTROL-C"
 1062  007212  012705  030212                  MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
 1063  007216  104405                  1$:     RDCHR                   ;GET A CHAR
 1064  007220  012667  010656                  MOV     (SP)+,SINGLE    ;SAVE IT
 1065  007224  116767  010652  014226          MOVB    SINGLE,EC2      ;ECHO IT
 1066  007232  104400                          TYPE
 1067  007234  023460                          EC2
 1068
 1069  007236  022767  000003  010636          CMP     #3,SINGLE       ;WAS IT A CONTROL-C ??
 1070  007244  001423                          BEQ     3$              ;BR IF YES
 1071
 1072  007246  020527  031213                  CMP     R5,#TBUF+513.   ;BUFFER FULL ??
 1073  007252  001420                          BEQ     3$              ;BR IF YES
 1074
 1075  007254  022767  000012  010620          CMP     #12,SINGLE      ;WAS IT A LINE FEED ??
 1076  007262  001011                          BNE     2$              ;BR IF NOT
 1077
 1078  007264  016700  010674                  MOV     FILLB,R0        ;LOAD LF PLUS FILLERS
 1079  007270  116725  010606                  MOVB    SINGLE,(R5)+
 1080  007274  116725  010662          11$:    MOVB    FILLA,(R5)+     ;LOAD A FILLER CHAR
 1081  007300  005300                          DEC     R0
 1082  007302  001374                          BNE     11$             ;BR TILL REQUIRED FILLERS LOADED
 1083  007304  000744                          BR      1$              ;GO ASK FOR ANOTHER CHAR
 1084
 1085  007306  116725  010570          2$:     MOVB    SINGLE,(R5)+    ;LOAD IT IN BUFFER
 1086  007312  000741                          BR      1$              ;GO GET NEXT CHAR
 1087
 1088  007314  112767  000136  014140  3$:     MOVB    #136,EC3                ;ECHO CONTROL-C
 1089  007322  112767  000103  014133          MOVB    #103,EC3+1
 1090  007330  104400                          TYPE
 1091  007332  023462                          EC3
 1092  007334  162705  030212                  SUB     #TBUF,R5        ;SET UP CHAR COUNT
 1093  007340  005405                          NEG     R5
 1094  007342  010567  010150                  MOV     R5,CHRCNT
 1095  007346  004767  000050          4$:     JSR     PC,DHST2        ;GO EXECUTE PATTERN
 1096  007352  005267  010516                  INC     DATCNT
 1097  007356  026767  010524  010510          CMP     PATLIM,DATCNT   ;DONE REQUIRED ITERATIONS
 1098  007364  001370                          BNE     4$              ;BR IF NOT
 1099
 1100  007366  016767  010502  171564          MOV     DATCNT,$REG0    ;SAVE ITERATION COUNT
 1101  007374  005067  010474                  CLR     DATCNT          ;INIT ITERATION COUNTER
 1102  007400  012767  007410  171502          MOV     #5$,$LPERR      ;RETURN TO 5$
 1103  007406  104024                          ERROR   24              ;DONE REQUIRED ITERATIONS
 1104  007410  105777  171522          5$:     TSTB    @SWR            ;LOCK ON THIS BUFFER ??
 1105  007414  100754                          BMI     4$              ;BR IF YES
 1106  007416  000167  176534                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
 1107
```

```
1110  007422  004767  004322          DHST2:  JSR     PC,DHSET1       ;GO SET UP THE DH11
1111  007426  056761  010016  000012          BIS     LINMSK,BAR(R1)  ;ACTIVATE THE LINE
1112  007434  004767  007702                  JSR     PC,CHPS1        ;GO CLEAR PSW
1113
1114  007440  012767  000100  010416  PTWAIT: MOV     #100,TIMEA      ;INIT TIMERS
1115  007446  005067  010414                  CLR     TIMEB
1116  007452  005767  010046          1$:     TST     RDONE           ;DONE ENTIRE PATTERN ?
1117  007456  001023                          BNE     3$              ;BR IF YES
1118  007460  004767  005416                  JSR     PC,TIMEIT       ;CALL THE TIMER
1119  007464  000772                          BR      1$              ;EXECUTED IF NO TIMEOUT
1120
1121  007466  012777  004000  007742          MOV     #BIT11,@DHADR   ;CLEAR THE DH11
1122  007474  016767  010374  171456          MOV     DATCNT,SAVEGO   ;SAVE ITERATION COUNTER
1123  007502  016767  010370  171452          MOV     DATPAT,SAVEGO   ;SAVE PATTERN TYPE
1124  007510  012767  007520  171372          MOV     #2$,SLPEXR      ;SET UP ERROR RETURN
1125  007516  104025                          ERROR   25              ;DATA PATTERNS TEST TIMEOUT ERROR
1126
1127  007520  005726                  2$:     TST     (SP)+           ;FIX STACK SINCE WE ARE SKIPPING RTS
1128  007522  000167  176430                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
1129
1130  007526  052711  004000          3$:     BIS     #BIT11,(R1)     ;CLEAR THE DH11
1131  007532  004767  000740                  JSR     PC,CKERDP       ;GO CHECK DATA BUFFERS
1132  007536  000207                          RTS     PC              ;RETURN TO CONTROL ROUTINE
1133
1134
```

```
                                        ;TRANSMITTER INTERRUPT SERVICE ROUTINE THREE
                                        ;
007540  032711  002000          TINT3:  BIT     #BIT10,(R1)     ;NON EX MEM ERROR ??
007544  001430                  BEQ     2$                      ;BR IF NOT
                                        ;
007546  011103                  MOV     (R1),R3                 ;SAVE THE SCR
007550  004767  007602          JSR     PC,CHPS2                ;GO LOCK OUT INTRS
007554  012711  004000          MOV     #BIT11,(R1)             ;CLEAR OUT THE DH11
007560  116704  010112          MOVB    LINE,R4                 ;SET UP THE S/B DATA
007564  042703  175760          BIC     #175760,R3              ;CLEAR OUT SUPERFLUOUS BITS
007570  010102                  MOV     R1,R2                   ;SET UP REGADR
007572  004767  004426          JSR     PC,SUER1                ;GO SET UP ERROR INFO
007576  004567  004336          JSR     R5,SUNUM                ;GO SET UP LINE NO. IN MSG
007602  017676                  LINE
007604  020232                  EM1+44
007606  012767  007616  171274  MOV     #1$,$LPERR              ;SET UP THE ERROR LOOP RETURN
007614  104001                  ERROR   1                       ;NON EX MEM ERROR
007616  022626          1$:     CMP     (SP)+,(SP)+             ;POP THE STACK
007620  005726                  TST     (SP)+                   ;FIX STACK SINCE NO RTS IS EXECUTED
007622  000167  176330          JMP     EXPAT3                  ;GO ASK FOR NEW PATTERN
                                        ;
007626  011103          2$:     MOV     (R1),R3                 ;GET THE SCR REG CONTENTS
007630  100431                  BMI     4$                      ;BR IF XMIT DONE SET
                                        ;
007632  004767  007520          JSR     PC,CHPS2                ;GO LOCK OUT INTRS
007636  012711  004000          MOV     #BIT11,(R1)             ;CLEAR THE DH11 - FATAL ERROR
007642  012704  100000          MOV     #BIT15,R4               ;SET UP S/B DATA
007646  156704  010024          BISB    LINE,R4
007652  042703  077760          BIC     #77760,R3               ;CLEAR OUT SUPERFLUOUS BITS
007656  010102                  MOV     R1,R2                   ;SET UP REGADR
007660  004767  004340          JSR     PC,SUER1                ;GO SET UP ERROR INFO
007664  004567  004250          JSR     R5,SUNUM                ;GO SET UP LINE NO. IN MSG
007670  017676                  LINE
007672  020414                  EM2+54
007674  012767  007704  171206  MOV     #3$,$LPERR              ;SET UP ERROR LOOP RETURN
007702  104002                  ERROR   2                       ;XMITTR FALSE INTERRUPT
007704  022626          3$:     CMP     (SP)+,(SP)+             ;POP THE STACK
007706  005726                  TST     (SP)+                   ;FIX STACK SINCE NO RTS IS EXECUTED
007710  000167  176242          JMP     EXPAT3                  ;GO ASK FOR NEW PATTERN
                                        ;
007714  005761  000012          4$:     TST     BAR(R1)         ;DID BAR BIT CLEAR ??
007720  001430                  BEQ     6$                      ;BR IF YES
                                        ;
007722  004767  007430          JSR     PC,CHPS2                ;GO LOCK OUT INTRS
007726  016103  000012          MOV     BAR(R1),R3              ;GET THE WAS DATA
007732  012711  004000          MOV     #BIT11,(R1)             ;CLEAR THE DH11
007736  005004                  CLR     R4                      ;SET UP S/B DATA
007740  010102                  MOV     R1,R2                   ;SET UP REGADR
007742  062702  000012          ADD     #BAR,R2
007746  004767  004252          JSR     P:,SUER1                ;GO SET UP ERROR INFO
007752  004567  004162          JSR     R5,SUNUM                ;GO SET UP LINE NO. IN MSG
007756  017676                  LINE
007760  020474                  EM3+55
```

```
1191  007762  012767  007772  171120          MOV     #5$,$LPERR      ;SAVE THE ERROR LOOP RETURN
1192  007770  104003                          ERROR   3               ;BUFFER ACTIVE REG FAILED TO CLEAR
1193  007772  022626                  5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
1194  007774  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
1195  007776  000167  176154                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
1196
1197  010002  005761  000010          6$:     TST     BCR(R1)         ;DID BYTE COUNT GO TO ZERO ??
1198  010006  001430                          BEQ     8$              ;BR IF YES
1199
1200  010010  004767  007342                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
1201  010014  016103  000010                  MOV     BCR(R1),R3      ;GET THE WAS DATA
1202  010020  012711  004000                  MOV     #BIT11,(R1)     ;CLEAR THE DH11
1203  010024  005004                          CLR     R4              ;SET UP S/B DATA
1204  010026  010102                          MOV     R1,R2           ;SET UP REGADR
1205  010030  062702  000010                  ADD     #BCR,R2
1206  010034  004767  004164                  JSR     PC,SUER1        ;GO SET UP THE ERROR INFO
1207  010040  004567  004074                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
1208  010044  017676                          LINE
1209  010046  020551                          EM4+52
1210  010050  012767  010060  171032          MOV     #7$,$LPERR      ;SET UP ERROR LOOP RETURN
1211  010056  104004                          ERROR   4               ;BYTE COUNT REG FAILED TO GO TO 000000
1212  010060  022626                  7$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
1213  010062  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
1214  010064  000167  176066                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
1215
1216  010070  016103  000006          8$:     MOV     CAR(R1),R3      ;GET THE WAS DATA
1217  010074  016704  007416                  MOV     CHRCNT,R4       ;SET UP S/B DATA
1218  010100  005404                          NEG     R4
1219  010102  062704  030212                  ADD     #TBUF,R4
1220  010106  020304                          CMP     R3,R4           ;WAS CAR CORRECT ??
1221  010110  001423                          BEQ     10$             ;BR IF YES
1222
1223  010112  004767  007240                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
1224  010116  010102                          MOV     R1,R2           ;SET UP REGADR
1225  010120  062702  000006                  ADD     #CAR,R2
1226  010124  004767  004074                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
1227  010130  004567  004004                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
1228  010134  017676                          LINE
1229  010136  020633                          EM5+57
1230  010140  012767  010150  170742          MOV     #9$,$LPERR      ;SET UP THE ERROR RETURN
1231  010146  104005                          ERROR   5               ;CURRENT ADDRESS REG NOT CORRECT
1232  010150  022626                  9$:     CMP     (SP)+,(SP)+     ;POP THE STACK
1233  010152  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
1234  010154  000167  175776                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
1235
1236  010160  000002                  10$:    RTI
```

```
                                        ;RECEIVER INTERRUPT SERVICE ROUTINE THREE

010162  032711  040000          RINT3:  BIT     #BIT14,(R1)     ;SILO OVERFLOW ERROR ??
010166  001427                          BEQ     2$              ;BR IF NOT

010170  004767  007162                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
010174  011103                          MOV     (R1),R3         ;GET THE WAS DATA
010176  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
010202  042703  177760                  BIC     #177760,R3      ;CLEAR JUNK
010206  116704  007464                  MOVB    LINE,R4
010212  004767  004006                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
010216  004567  003716                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
010222  017676                          LINE
010224  020702                          EM6+44
010226  012767  010236  170654          MOV     #1$,$LPERR      ;SET UP ERROR LOOP RETURN
010234  104006                          ERROR   6               ;SILO OVERFLOW - BAD,BAD,BAD !!!
010236  022626                  1$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
010240  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
010242  000167  175710                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN

010246  105711          2$:     TSTB    (R1)            ;CHAR AVAIL SET ??
010250  100432                  BMI     4$              ;BR IF YES

010252  004767  007100                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
010256  011103                          MOV     (R1),R3         ;GET WAS DATA
010260  042703  177560                  BIC     #177560,R3      ;CLEAN IT UP
010264  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR DH11
010270  012704  000200                  MOV     #BIT07,R4       ;SET UP S/B DATA
010274  156704  007376                  BISB    LINE,R4
010300  010102                          MOV     R1,R2           ;SET UP REGPR
010302  004767  003716                  JSR     PC,SUER1        ;GO SET UP ERROR INFO
010306  004567  003626                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
010312  017676                          LINE
010314  020756                          EM7+51
010316  012767  010326  170564          MOV     #3$,$LPERR      ;SET UP THE ERROR LOOP RETURN
010324  104007                          ERROR   7               ;RECEIVER FALSE INTERRUPT
010326  022626                  3$:     CMP     (SP)+,(SP)+     ;POP GOES THE SP
010330  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
010332  000167  175620                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN

010336  016167  000002  170634  4$:     MOV     NRC(R1),$TMP0   ;SAVE THE DATA RECEIVED
010344  100427                          BMI     6$              ;BR IF IT WAS VALID DATA

010346  004767  007004                  JSR     PC,CHPS2        ;GO LOCK OUT INTRS
010352  012711  004000                  MOV     #BIT11,(R1)     ;NOW CLEAR THE DH11
010356  162767  025732  015342          SUB     #BUF,RBFPTR     ;WHICH CHAR WAS IT ??
010364  016702  015336                  MOV     RBFPTR,R2       ;SAVE CHAR NUMBER
010370  004767  003624                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
010374  004567  003540                  JSR     R5,SUNUM        ;GO SET UP LINE NO. IN MSG
010400  017676                          LINE
010402  021026                          EM10+45
010404  012767  010414  170476          MOV     #5$,$LPERR      ;SET UP ERROR RETURN
010412  104010                          ERROR   10              ;RECEIVED INVALID DATA
010414  022626                  5$:     CMP     (SP)+,(SP)+     ;POP GOES THE STACK
```

```
1293  010416  005726                          TST     (SP)+           ;FIX STACK SINCE NO RTS IS EXECUTED
1294  010420  000167  175532                  JMP     EXPAT3          ;GO ASK FOR NEW PATTERN
1295
1296  010424  016777  170550  015274  6$:     MOV     $TMP0,@RBFPTR   ;STORE CHAR IN THE BUFFER
1297  010432  062767  000001  015266          ADD     #1,RBFPTR       ;UPDATE THE POINTER
1298  010440  026767  007062  015260          CMP     RBFEND,RBFPTR   ;END OF BUFFER ??
1299  010446  001407                          BEQ     7$              ;BR IF YES
1300  010450  062767  000001  015250          ADD     #1,RBFPTR
1301  010456  026767  007044  015242          CMP     RBFEND,RBFPTR
1302  010464  001003                          BNE     8$              ;BR IF NOT DONE
1303  010466  052767  000001  007030  7$:     BIS     #1,RDONE        ;SET SOFTWARE DONE FLAG
1304  010474  000002                  8$:     RTI                     ;RETURN TO WAIT LOOP
```

```
1307                                        ;THIS ROUTINE IS CALLED TO CHECK THE RECEIVED DATA AND REPORT ALL ERRORS
1308                                        ;FOR THE DATA PATTERNS TESTS
1309
1310  010476  012767  025732  015222  CKEROP: MOV     #RBUF,RBFPTR    ;SET UP POINTERS
1311  010504  012767  030212  015216          MOV     #TBUF,TBFPTR
1312
1313  010512  117704  015212          1$:     MOVB    @TBFPTR,R4      ;GET THE S/B DATA
1314  010516  000304                          SWAB    R4              ;PUT LINE NO. IN HIGH BYTE
1315  010520  105004                          CLRB    R4
1316  010522  156704  007150                  BISB    LINE,R4
1317  010526  000304                          SWAB    R4
1318  010530  052704  100000                  BIS     #BIT15,R4       ;AND FINALLY THE VALID DATA BIT
1319  010534  046704  007354                  BIC     CLMSK,R4        ;MASK OFF BITS NOT XMITTED
1320  010540  017703  015162                  MOV     @RBFPTR,R3      ;GET THE WAS DATA
1321  010544  020304                          CMP     R3,R4           ;WAS = S/B ?????
1322  010546  001425                          BEQ     3$              ;BR IF YES
1323
1324  010550  010367  170442                  MOV     R3,$TMP7        ;SAVE THE WAS DATA
1325  010554  010146                          MOV     R1,-(SP)        ;SAVE THE DEVADR
1326  010556  016701  015144                  MOV     RBFPTR,R1       ;GET THE SBADR
1327  010562  016702  015142                  MOV     TBFPTR,R2       ;GET THE WASADR
1328  010566  010200                          MOV     R2,R0           ;GET XMIT BUFFER ADDR
1329  010570  162700  030212                  SUB     #TBUF,R0        ;GENERATE CHAR #
1330  010574  004767  003420                  JSR     PC,SUER2        ;GO SET UP ERROR INFO
1331  010600  004567  003334                  JSR     R5,SUNUM        ;GO PUT LINE NO. IN MSG
1332  010604  017676                          LINE
1333  010606  021161                          EM11+23
1334  010610  012767  010620  170272          MOV     #2$,$LPERR      ;SET UP ERROR RETURN
1335  010616  104011                          ERROR   11              ;DATA COMPARE ERROR OR PARITY,FRAMING
1336                                                                  ;OR OVERRUN
1337  010620  012601                  2$:     MOV     (SP)+,R1        ;RESTORE THE DEVADR
1338
1339  010622  005267  015102          3$:     INC     TBFPTR          ;UPDATE POINTERS
1340  010626  062767  000001  015072          ADD     #1,RBFPTR
1341  010634  026767  006666  015064          CMP     RBFEND,RBFPTR   ;COMPARED ALL CHARS ??
1342  010642  001407                          BEQ     4$              ;BR IF YES
1343  010644  062767  000001  015054          ADD     #1,RBFPTR       ;UPDATE IT AGAIN
1344  010652  026767  006650  015046          CMP     RBFEND,RBFPTR   ;DONE YET ?
1345  010660  001314                          BNE     1$              ;BR IF NOT
1346
1347  010662  000207                  4$:     RTS     PC              ;RETURN TO WAIT LOOP
1348
```

```
1350                                    ;*****************************************************************
 (1)
 (1)                                    .SBTTL  END OF PASS ROUTINE
 (1)
 (1)                                    ;*INCREMENT THE PASS NUMBER ($PASS)
 (1)                                    ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
 (1)                                    ;*IF THERES A MONITOR GO TO IT
 (1)                                    ;*IF THERE ISN'T JUMP TO START2
 (1)
 (1)  010664                            $EOP:
 (1)  010664  000004                            SCOPE
 (1)  010666  005067  170210                    CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 (1)  010672  005067  170322                    CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 (1)  010676  005267  170334                    INC     $PASS           ;;INCREMENT THE PASS NUMBER
 (1)  010702  042767  100000  170326           BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 (1)  010710  005327                            DEC     (PC)+           ;;LOOP?
 (1)  010712  000001                    $EOPCT: .WORD   1
 (1)  010714  003022                            BGT     $DOAGN          ;;YES
 (1)  010716  012737                            MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 (1)  010720  000001                    $ENDCT: .WORD   1
 (1)  010722  010712                            $EOPCT
 (1)  010724  104400  010766                    TYPE    ,$ENDMG         ;;TYPE "END PASS #"
 (2)  010730  016746  170302                    MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 (2)  010734  104404                            TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 (1)  010736  104400  011003                    TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 (1)  010742                            $GET42:
 (1)
 (1)  010742  013700  000042                    MOV     @#42,R0         ;;GET MONITOR ADDRESS
 (1)  010746  001405                            BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 (1)  010750  000005                            RESET                   ;;CLEAR THE WORLD
 (1)  010752  004710                    $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 (1)  010754  000240                            NOP                     ;;SAVE ROOM
 (1)  010756  000240                            NOP                     ;;FOR
 (1)  010760  000240                            NOP                     ;;ACT11
 (1)  010762                            $DOAGN:
 (1)  010762  000137  002226                    JMP     @#START2                ;;RETURN
 (1)  010766  005015  047105  020104    $ENDMG: .ASCIZ  <15><12>/END PASS #/
 (1)  010774  040520  051523  021440
 (1)  011002     000
 (1)  011003     377     377     000    $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
1351                                    ;*****************************************************************
 (1)
 (1)                                    .SBTTL  SCOPE HANDLER ROUTINE
 (1)
 (1)                                    ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
 (1)                                    ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
 (1)                                    ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
 (1)                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 (1)                                    ;*SW14=1        LOOP ON TEST
 (1)                                    ;*SW11=1        INHIBIT ITERATIONS
 (1)                                    ;*SW09=1        LOOP ON ERROR
 (1)                                    ;*CALL
 (1)                                    ;*      SCOPE           ;;SCOPE=IOT
 (1)
```

```
(1)  011006                          $SCOPE:
(2)  011006  005067  006664          1$:     CLR     LINE            ;INIT LINE COUNTER
(1)  011012  032777  040000  170116          BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(1)  011020  001104                          BNE     $OVER           ;;YES IF SW14=1
(1)                                  ;#####START OF CODE FOR THE XOR TESTER#####
(1)  011022  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(1)                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(1)  011024  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(1)  011030  012737  011050  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(1)  011036  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
(1)  011042  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  011046  000453                          BR      $SVLAD          ;;GO TO THE NEXT TEST
(1)  011050  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(1)  011052  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(1)  011056  000413                          BR      7$              ;;LOOP ON THE PRESENT TEST
(1)  011060                          6$:;#####END OF CODE FOR THE XOR TESTER#####
(1)  011060  105767  170017          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(1)  011064  001421                          BEQ     3$              ;;BR IF NO
(1)  011066  126767  170023  170007          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
(1)  011074  101015                          BHI     3$              ;;BR IF NO
(1)  011076  032777  001000  170032          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
(1)  011104  001404                          BEQ     4$              ;;BR IF NO
(1)  011106  016767  167776  167772  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
(1)  011114  000446                          BR      $OVER
(1)  011116  105067  167761          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  011122  005067  170072                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  011126  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
(1)  011130  032777  004000  170000  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(1)  011136  001011                          BNE     1$              ;;BR IF YES
(1)  011140  005767  170072                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  011144  001406                          BEQ     1$              ;;    INHIBIT ITERATIONS
(1)  011146  005267  167732                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  011152  026767  170042  167724          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  011160  002024                          BGE     $OVER           ;;EQ IF MORE ITERATION REQUIRED
(1)  011162  012767  000001  167714  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  011170  016767  000052  170022          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  011176  105267  167700          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(1)  011202  116767  167674  170024          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
(1)  011210  011667  167672                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  011214  011667  167670                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  011220  005067  167776                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  011224  112767  000001  167663          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  011232  016777  167644  167700  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(1)  011240  016716  167642                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  011244  000002                          RTI                     ;;FIXES PS
(1)  011246  000010                  $MXCNT: 10                      ;;MAX. NUMBER OF ITERATIONS
1352                                 ;*****************************************************************
(1)
(1)                                  .SBTTL  ERROR HANDLER ROUTINE
(1)
(1)                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
(1)                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1)                                  ;*AND GO TO $ERRTYP ON ERROR
(1)                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
```

```
(1)                                     ;*SW15=1        HALT ON ERROR
(1)                                     ;*SW13=1        INHIBIT ERROR TYPEOUTS
(1)                                     ;*SW09=1        LOOP ON ERROR
(1)                                     ;*CALL
(1)                                     ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  011250                             $ERROR:
(1)  011250  105267  167627             7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  011254  001775                             BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  011256  016777  167620  167654             MOV     $ISTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  011264  005267  167622                     INC     $ERTTL          ;;INC THE ERROR COUNT
(1)  011270  011667  167622                     MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  011274  162767  000002  167614             SUB     #2,$ERRPC
(1)  011302  117767  167610  167604             MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  011310  032777  020000  167620             BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(1)  011316  001004                             BNE     20$             ;;SKIP TYPEOUTS
(1)  011320  004767  000072                     JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  011324  104400  001225                     TYPE    ,$CRLF
(1)  011330                             20$:
(1)  011330  122767  000001  167712             CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  011336  001007                             BNE     2$              ;;NO,SKIP APT ERROR REPORT
(1)  011340  116767  167550  000004             MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(1)  011346  004767  001166                     JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
(1)  011352     000                     21$:    .BYTE   0
(1)  011353     000                             .BYTE   0
(1)  011354  000777                     22$:    BR      22$             ;;APT ERROR LOOP
(1)  011356  005777  167554             2$:     TST     @SWR            ;;HALT ON ERROR
(1)  011362  100001                             BPL     3$              ;;SKIP IF CONTINUE
(1)  011364  000000                             HALT                    ;;HALT ON ERROR!
(1)  011366  032777  001000  167542     3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(1)  011374  001402                             BEQ     4$              ;;BR IF NO
(1)  011376  016716  167506                     MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(1)  011402  005767  167614             4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(1)  011406  001402                             BEQ     5$              ;;BR IF NONE
(1)  011410  016716  167606                     MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  011414                             5$:
(1)  011414  000002                             RTI                     ;;RETURN
1353                                    ;******************************************************************
(1)
(1)                                     .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(1)
(1)                                     ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
(1)                                     ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
(1)                                     ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
(1)
(1)  011416                             $ERRTYP:
(1)  011416  104400  001225                     TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  011422  010046                             MOV     R0,-(SP)        ;;SAVE R0
(1)  011424  005000                             CLR     R0              ;;PICKUP THE ITEM INDEX
(1)  011426  153700  001114                     BISB    @#$ITEMB,R0
(1)  011432  001004                             BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(1)                                                                     ;;TYPE THE PC OF THE ERROR
(2)  011434  016746  167456                     MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(2)                                                                     ;;ERROR ADDRESS
```

```
(2)  011440  104401                        TYPOC                ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  011442  000445                        BR      10$          ;;GET OUT
(1)  011444  005300                1$:     DEC     R0           ;;ADJUST THE INDEX SO THAT IT WILL
(1)  011446  006300                        ASL     R0           ;;      WORK FOR THE ERROR TABLE
(1)  011450  006300                        ASL     R0
(1)  011452  006300                        ASL     R0
(1)  011454  062700  001354                ADD     #$ERRTB,R0   ;;FORM TABLE POINTER
(1)  011460  012067  000004                MOV     (R0)+,2$     ;;PICKUP "ERROR MESSAGE" POINTER
(1)  011464  001404                        BEQ     3$           ;;SKIP TYPEOUT IF NO POINTER
(1)  011466  104400                        TYPE                 ;;TYPE THE "ERROR MESSAGE"
(1)  011470  000000                2$:     .WORD   0            ;;"ERROR MESSAGE" POINTER GOES HERE
(1)  011472  104400  001225                TYPE    $CRLF        ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  011476  012067  000004        3$:     MOV     (R0)+,4$     ;;PICKUP "DATA HEADER" POINTER
(1)  011502  001404                        BEQ     5$           ;;SKIP TYPEOUT IF 0
(1)  011504  104400                        TYPE                 ;;TYPE THE "DATA HEADER"
(1)  011506  000000                4$:     .WORD   0            ;;"DATA HEADER" POINTER GOES HERE
(1)  011510  104400  001225                TYPE    $CRLF        ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  011514  010146                5$:     MOV     R1,-(SP)     ;;SAVE R1
(1)  011516  012001                        MOV     (R0)+,R1     ;;PICKUP "DATA TABLE" POINTER
(1)  011520  001415                        BEQ     9$           ;;BR IF NO DATA TO BE TYPED
(1)  011522  012000                        MOV     (R0)+,R0     ;;PICKUP "DATA FORMAT" POINTER
(1)  011524  105720                6$:     TSTB    (R0)+        ;;"OCTAL" OR "DECIMAL"
(1)  011526  001003                        BNE     7$           ;;BR IF DECIMAL
(2)  011530  013146                        MOV     @(R1)+,-(SP) ;;SAVE @(R1)+ FOR TYPEOUT
(2)  011532  104401                        TYPOC                ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(1)  011534  000402                        BR      8$
(2)  011536                        7$:
(2)  011536  013146                        MOV     @(R1)+,-(SP) ;;SAVE @(R1)+ FOR TYPEOUT
(2)  011540  104404                        TYPDS                ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  011542  005711                8$:     TST     (R1)         ;;IS THERE ANOTHER NUMBER?
(1)  011544  001403                        BEQ     9$           ;;BR IF NO
(1)  011546  104400  011566                TYPE    11$          ;;TYPE TWO(2) SPACES
(1)  011552  000764                        BR      6$           ;;LOOP
(1)
(1)  011554  012601                9$:     MOV     (SP)+,R1     ;;RESTORE R1
(1)  011556  012600                10$:    MOV     (SP)+,R0     ;;RESTORE R0
(1)  011560  104400  001225                TYPE    $CRLF        ;;"CARRIAGE RETURN" & "LINE FEED"
(1)  011564  000207                        RTS     PC           ;;RETURN
(1)  011566  020040      000       11$:    .ASCIZ  /  /         ;;TWO(2) SPACES
(1)          011572                        .EVEN
1354                               ;*****************************************************************

(1)
(1)                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(1)
(1)                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(1)                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(1)                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(1)                                ;*CALL:
(1)                                ;*      MOV     NUM,-(SP)    ;;NUMBER TO BE TYPED
(1)                                ;*      TYPOS                ;;CALL FOR TYPEOUT
(1)                                ;*      .BYTE   N            ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(1)                                ;*      .BYTE   M            ;;M=1 OR 0
(1)                                ;*                                   ;;1=TYPE LEADING ZEROS
(1)                                ;*                                   ;;0=SUPPRESS LEADING ZEROS
```

```
(1)                                     ;*
(1)                                     ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(1)                                     ;*$TYPOS OR $TYPOC
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPON                   ;;CALL FOR TYPEOUT
(1)                                     ;*
(1)                                     ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(1)                                     ;*      TYPOC                   ;;CALL FOR TYPEOUT
(1)
(1)  011572  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  011576  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  011604  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  011610  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  011614  000406                          BR      $TYPON
(1)  011616  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(1)  011624  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  011632  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  011640  010346                          MOV     R3,-(SP)        ;;SAVE R3
(1)  011642  010446                          MOV     R4,-(SP)        ;;SAVE R4
(1)  011644  010546                          MOV     R5,-(SP)        ;;SAVE R5
(1)  011646  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  011652  005404                          NEG     R4
(1)  011654  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  011660  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  011664  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  011670  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  011674  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  011676  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  011700  000404                          BR      3$              ;;GO DO MSB
(1)  011702  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  011704  006105                          ROL     R5
(1)  011706  006105                          ROL     R5
(1)  011710  010503                          MOV     R5,R3
(1)  011712  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  011714  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  011720  100016                          BPL     7$              ;;BR IF NO
(1)  011722  042703  177770                  BIC     #^C7,R3         ;;GET RID OF JUNK
(1)  011726  001002                          BNE     4$              ;;TEST FOR 0
(1)  011730  005704                          TST     R4              ;;SUPPRESS THIS 0?
(1)  011732  001403                          BEQ     5$              ;;BR IF YES
(1)  011734  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  011736  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  011742  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  011746  110367  000040                  MOVB    R3,$B           ;;SAVE FOR TYPING
(1)  011752  104400  012012                  TYPE    ,$B             ;;GO TYPE THIS DIGIT
(1)  011756  105367  000032          7$:     DECB    $OCNT           ;;COUNT BY 1
(1)  011762  003347                          BGT     2$              ;;BR IF MORE TO DO
(1)  011764  002402                          BLT     6$              ;;BR IF DONE
(1)  011766  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  011770  000744                          BR      2$              ;;GO DO THE LAST DIGIT
(1)  011772  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
```

```
(1)  011774  012604                        MOV     (SP)+,R4        ;;RESTORE R4
(1)  011776  012603                        MOV     (SP)+,R3        ;;RESTORE R3
(1)  012000  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  012006  012616                        MOV     (SP)+,(SP)
(1)  012010  000002                        RTI                     ;;RETURN
(1)  012012  000                    BS:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  012013  000                            .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  012014  000                    $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  012015  000                    $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  012016  000000                 $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
1355                                ;**************************************************************
(1)
(1)                                 .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
(1)
(1)                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
(1)                                 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
(1)                                 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
(1)                                 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
(1)                                 ;*REPLACED WITH SPACES.
(1)                                 ;*CALL:
(1)                                 ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
(1)                                 ;*      TYPOS                   ;;GO TO THE ROUTINE
(1)
(2)  012020                         $TYPOS:
(3)  012020  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  012022  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  012024  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  012026  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  012030  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  012032  012746  020200                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  012036  016605  000020                 MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  012042  100004                         BPL     1$              ;;BR IF INPUT IS POS.
(1)  012044  005405                         NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  012046  112766  000055  000001         MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  012054  005000                  1$:    CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  012056  012703  012234                 MOV     #$OBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  012062  112723  000040                 MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  012066  005002                  2$:    CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  012070  016001  012224                 MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  012074  160105                  3$:    SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  012076  002402                         BLT     4$              ;;BR IF DONE
(1)  012100  005202                         INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  012102  000774                         BR      3$
(1)  012104  060105                  4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  012106  005702                         TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  012110  001002                         BNE     5$              ;;FALL THROUGH IF 0
(1)  012112  105716                         TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  012114  100407                         BMI     7$              ;;BR IF YES
(1)  012116  106316                  5$:    ASLB    (SP)            ;;MSD?
(1)  012120  103003                         BCC     6$              ;;BR IF NO
(1)  012122  116663  000001  177777         MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)  012130  052702  000060          6$:    BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  012134  052702  000040          7$:    BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  012140  110223                         MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
(1)  012142  005720                    TST     (R0)+           ;;JUST INCREMENTING
(1)  012144  020027  000C.0            CMP     R0,#10.         ;;CHECK THE TABLE INDEX
(1)  012150  002746                    BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  012152  003002                    BGT     8$              ;;GO TO EXIT
(1)  012154  010502                    MOV     R5,R2           ;;GET THE LSD
(1)  012156  000764                    BR      6$              ;;GO CHANGE TO ASCII
(1)  012160  105726                8$: TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  012162  100003                    BPL     9$              ;;BR IF NO
(1)  012164  116663  177777  177776    MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  012172  105013                9$: CLRB    (R3)            ;;SET THE TERMINATOR
(3)  012174  012605                    MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  012176  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  012200  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  012202  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  012204  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  012206  104400  012234            TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)  012212  016666  000002  000004    MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)  012220  012616                    MOV     (SP)+,(SP)
(1)  012222  000002                    RTI                     ;;RETURN TO USER
(1)  012224  023420                $DTBL:  10000.
(1)  012226  001750                        1000.
(1)  012230  000144                        100.
(1)  012232  000012                        10.
(1)  012234  000004                $DBLK:  .BLKW   4
1356                               ;*****************************************************************

(1)                                .SBTTL  TYPE ROUTINE
(1)
(1)                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(1)                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(1)                                ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(1)                                ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(1)                                ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(1)                                ;*
(1)                                ;*CALL:
(1)                                ;*1) USING A TRAP INSTRUCTION
(1)                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(1)                                ;*OR
(1)                                ;*      TYPE
(1)                                ;*      MESADR
(1)                                ;*
(1)
(1)  012244  105767  166705        $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(1)  012250  100002                        BPL     1$              ;;BR IF YES
(1)  012252  000000                        HALT                    ;;HALT HERE IF NO TERMINAL
(1)  012254  000430                        BR      3$              ;;LEAVE
(1)  012256  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
(1)  012260  017600  000002                MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(1)  012264  122767  000001  166756        CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(1)  012272  001011                        BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(1)  012274  132767  000100  166747        BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(1)  012302  001405                        BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(1)  012304  010067  000004                MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(1)  012310  004767  000214                JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
```

```
(1)  012314  000000                    61$:    .WORD   0               ;;MESSAGE ADDRESS
(1)  012316  132767  000040  166725    62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(1)  012324  001003                    BNE     60$                     ;;YES,SKIP TYPE OUT
(1)  012326  112046                    2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(1)  012330  001005                    BNE     4$                      ;;BR IF IT ISN'T THE TERMINATOR
(1)  012332  005726                    TST     (SP)+                   ;;IF TERMINATOR POP IT OFF THE STACK
(1)  012334  012600                    60$:    MOV     (SP)+,R0        ;;RESTORE R0
(1)  012336  062716  000002            3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  012342  000002                    RTI                             ;;RETURN
(1)  012344  122716  000011            4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  012350  001426                    BEQ     8$
(1)  012352  122716  000200                    CMPB    #TCRLF,(SP)     ;;BRANCH IF NOT <CRLF>
(1)  012356  001004                    BNE     5$
(1)  012360  005726                    TST     (SP)+                   ;;POP  <CR><LF> EQUIV
(1)  012362  104400                    TYPE                            ;;TYPE A CR AND LF
(1)  012364  001225                    $CRLF
(1)  012366  000757                    BR      2$                      ;;GET NEXT CHARACTER
(1)  012370  004767  000[illegible]36  5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  012374  126726  166554            6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  012400  001352                    BNE     2$                      ;;IF NO GO GET NEXT CHAR.
(1)  012402  016746  166544                    MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                                    ;;AND THE NULL CHAR.
(1)  012406  105366  000001            7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  012412  002770                    BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  012414  004767  000032                    JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  012420  105367  000072                    DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(1)  012424  000770                    BR      7$                      ;;LOOP
(1)
(1)                                    ;HORIZONTAL TAB PROCESSOR
(1)
(1)  012426  112716  000040            8$:     MOVB    #40,(SP)        ;;REPLACE TAB WITH SPACE
(1)  012432  004767  000014            9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  012436  132767  000007  000052            BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  012444  001372                    BNE     9$                      ;;TAB STOP
(1)  012446  005726                    TST     (SP)+                   ;;POP SPACE OFF STACK
(1)  012450  000726                    BR      2$                      ;;GET NEXT CHARACTER
(1)  012452  105777  166470            $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(1)  012456  100375                    BPL     $TYPEC
(1)  012460  116677  000002  166462            MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  012466  122766  000015  000002            CMPB    #15,2(SP)       ;;BRANCH IF
(1)  012474  001003                    BNE     1$                      ;;NOT <CR>
(1)  012476  105067  000014                    CLRB    $CHARCNT        ;;
(1)  012502  000406                    BR      $TYPEX                  ;;EXIT
(1)  012504  122766  000012  000002    1$:     CMPB    #12,2(SP)       ;;BRANCH IF
(1)  012512  002002                    BGE     $TYPEX                  ;;<LF>
(1)  012514  105227                    INCB    (PC)+                   ;;INC SPACE
(1)  012516  000000                    $CHARCNT:.WORD  0               ;;COUNT
(1)  012520  000207                    $TYPEX: RTS     PC
(1)                                    ;;      EQUATES
(1)                 000011             HT=11
(1)                 000200             TCRLF=200
(1)
1357                                   ;******************************************************************
 (1)
```

```
(1)                                     .SBTTL  APT COMMUNICATIONS ROUTINE
(1)  012522  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG        ;TO REPORT FATAL ERROR
(1)  012530  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG        ;TO TYPE A MESSAGE
(1)  012536  000403                         BR      $ATYC
(1)  012540  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG        ;TO ONLY REPORT FATAL ERROR
(3)  012546                         $ATYC:
(3)  012546  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  012550  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(1)  012552  105767  000206                 TSTB    $MFLG           ;SHOULD TYPE A MESSAGE?
(1)  012556  001450                         BEQ     5$              ;IF NOT:  BR
(1)  012560  122767  000001  166462         CMPB    #APTENV,$ENV    ;OPERATING UNDER APT?
(1)  012564  001031                         BNE     3$              ;IF NOT:  BR
(1)  012570  132767  000100  166453         BITB    #APTSPOOL,$ENVM ;SHOULD SPOOL MESSAGES?
(1)  012576  001425                         BEQ     3$              ;IF NOT:  BR
(1)  012600  017600  000004                 MOV     @4(SP),R0       ;GET MESSAGE ADDR.
(1)  012604  062766  000002  000004         ADD     #2,4(SP)                ;BUMP RETURN ADDR.
(1)  012612  005767  166412         1$:     TST     $MSGTYPE        ;SEE IF DONE W/ LAST XMISSION?
(1)  012616  001375                         BNE     1$              ;IF NOT:  WAIT
(1)  012620  010067  166420                 MOV     R0,$MSGAD       ;PUT ADDR IN MAILBOX
(1)  012624  105720                 2$:     TSTB    (R0)+           ;FIND END OF MESSAGE
(1)  012626  001376                         BNE     2$
(1)  012630  166700  166410                 SUB     $MSGAD,R0       ;SUB START OF MESSAGE
(1)  012634  006200                         ASR     R0              ;GET MESSAGE LNGTH IN WORDS
(1)  012636  010067  166404                 MOV     R0,$MSGLGT      ;PUT LENGTH IN MAILBOX
(1)  012642  012767  000004  166360         MOV     #4,$MSGTYPE     ;TELL APT TO TAKE MSG.
(1)  012650  000413                         BR      5$
(1)  012652  017667  000004  000016  3$:     MOV     @4(SP),4$       ;PUT MSG ADDR IN JSR LINKAGE
(1)  012660  062766  000002  000004         ADD     #2,4(SP)                ;BUMP RETURN ADDRESS
(3)  012666  016746  165104                 MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
(1)  012672  004767  177346                 JSR     PC,$TYPE        ;CALL TYPE MACRO
(1)  012676  000000                 4$:     .WORD   0
(1)  012700                         5$:
(1)  012700  105767  000062         10$:    TSTB    $FFLG           ;SHOULD REPORT FATAL ERROR?
(1)  012704  001416                         BEQ     12$             ;IF NOT:  BR
(1)  012706  005767  166336                 TST     $ENV            ;RUNNING UNDER APT?
(1)  012712  001413                         BEQ     12$             ;IF NOT:  BR
(1)  012714  005767  166310         11$:    TST     $MSGTYPE        ;FINISHED LAST MESSAGE?
(1)  012720  001375                         BNE     11$             ;IF NOT:  WAIT
(1)  012722  017667  000004  166302         MOV     @4(SP),$FATAL   ;GET ERROR #
(1)  012730  062766  000002  000004         ADD     #2,4(SP)                ;BUMP RETURN ADDR.
(1)  012736  005267  166266                 INC     $MSGTYPE        ;TELL APT TO TAKE ERROR
(1)  012742  105067  000020         12$:    CLRB    $FFLG           ;CLEAR FATAL FLAG
(1)  012746  105067  000013                 CLRB    $LFLG           ;CLEAR LOG FLAG
(1)  012752  105067  000006                 CLRB    $MFLG           ;CLEAR MESSAGE FLAG
(3)  012756  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  012760  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  012762  000207                         RTS     PC              ;RETURN
(1)
(1)  012764     000                 $MFLG:  .BYTE   0               ;MESSG. FLAG
(1)  012765     000                 $LFLG:  .BYTE   0               ;LOG FLAG
(1)  012766     000                 $FFLG:  .BYTE   0               ;FATAL FLAG
(1)          012770                         .EVEN
(1)          000200                 APTSIZE=200
(1)          000001                 APTENV=001
```

```
   (1)            000100                   APTSPOOL=100
   (1)            000040                   APTCSUP=040
  1358                                     ;*****************************************************************
   (1)
   (1)                                     .SBTTL  TTY INPUT ROUTINE
   (1)
   (1)                                     ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
   (1)                                     ;*CALL:
   (1)                                     ;*      RDCHR                       ;;INPUT A SINGLE CHARACTER FROM THE TTY
   (1)                                     ;*      RETURN HERE                 ;;CHARACTER IS ON THE STACK
   (1)                                     ;
   (1)
   (1)  012779  011646                     $RDCHR: MOV     (SP),-(SP)          ;;PUSH DOWN THE PC
   (1)  012772  016666  000004  000002             MOV     4(SP),2(SP)         ;;SAVE THE PS
   (1)  013000  105777  166136              1$:    TSTB    @$TKS               ;;WAIT FOR
   (1)  013004  100375                              BPL     1$                  ;;A CHARACTER
   (1)  013006  117766  166132  000004             MOVB    @$TKB,4(SP)         ;;READ THE TTY
   (1)  013014  042766  177600  000004             BIC     #^C<177>,4(SP)      ;;GET RID OF JUNK IF ANY
   (1)  013022  000002                              RTI                         ;;GO BACK TO USER
   (2)                                     ;*****************************************************************
   (1)                                     ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
   (1)                                     ;*CALL:
   (1)                                     ;*      RDLIN                       ;;INPUT A STRING FROM THE TTY
   (1)                                     ;*      RETURN HERE                 ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
   (1)                                     ;*                                  ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
   (1)
   (1)  013024  010346                     $RDLIN: MOV     R3,-(SP)            ;;SAVE R3
   (1)  013026  012703  013132              1$:    MOV     #$TTYIN,R3          ;;GET ADDRESS
   (1)  013032  022703  013142              2$:    CMP     #$TTYIN+8.,R3       ;;BUFFER FULL?
   (1)  013036  101405                              BLOS    4$                  ;;BR IF YES
   (1)  013040  104405                              RDCHR                       ;;GO READ ONE CHARACTER FROM THE TTY
   (1)  013042  112613                              MOVB    (SP)+,(R3)          ;;GET CHARACTER
   (1)  013044  122713  000177                      CMPB    #177,(R3)           ;;IS IT A RUBOUT
   (1)  013050  001003                              BNE     3$                  ;;SKIP IF NOT
   (1)  013052  104400  001224              4$:    TYPE    $QUES               ;;TYPE A '?'
   (1)  013056  000763                              BR      1$                  ;;CLEAR THE BUFFER AND LOOP
   (1)  013060  111367  000044              3$:    MOVB    (R3),9$             ;;ECHO THE CHARACTER
   (1)  013064  104400  013130                      TYPE    ,9$
   (1)  013070  122723  000015                      CMPB    #15,(R3)+           ;;CHECK FOR RETURN
   (1)  013074  001356                              BNE     2$                  ;;LOOP IF NOT RETURN
   (1)  013076  105063  177777                      CLRB    -1(R3)              ;;CLEAR RETURN (THE 15)
   (1)  013102  104400  001226                      TYPE    ,$LF                ;;TYPE A LINE FEED
   (1)  013106  012603                              MOV     (SP)+,R3            ;;RESTORE R3
   (1)  013110  011646                              MOV     (SP),-(SP)          ;;ADJUST THE STACK AND PUT ADDRESS OF THE
   (1)  013112  016666  000004  000002              MOV     4(SP),2(SP)         ;;      FIRST ASCII CHARACTER ON IT
   (1)  013120  012766  013132  000004              MOV     #$TTYIN,4(SP)
   (1)  013126  000002                              RTI                         ;;RETURN
   (1)  013130     000              9$:    .BYTE   0                           ;;STORAGE FOR ASCII CHAR. TO TYPE
   (1)  013131     000                     .BYTE   0                           ;;TERMINATOR
   (1)  013132  000010              $TTYIN: .BLKB   8.                          ;;RESERVE 8 BYTES FOR TTY INPUT
  1359                                     ;*****************************************************************
   (1)
   (1)                                     .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
   (1)
```

```
(1)                                     ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
(1)                                     ;*CHANGE IT TO BINARY.
(1)                                     ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
(1)                                     ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
(1)                                     ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
(1)                                     ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
(1)                                     ;*CALL:
(1)                                     ;*      RDOCT                   ;;READ AN OCTAL NUMBER
(1)                                     ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
(1)                                     ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
(1)
(1)  013142  011646                     $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
(1)  013144  016666  000004  000002             MOV     4(SP),2(SP)     ;;INPUT NUMBER
(3)  013152  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  013154  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  013156  010246                             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  013160  104406                     1$:     RDLIN                   ;;READ AN ASCIZ LINE
(1)  013162  012600                             MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
(1)  013164  010067  000100                     MOV     R0,5$           ;;AND SAVE IT
(1)  013170  005001                             CLR     R1              ;;CLEAR DATA WORD
(1)  013172  005002                             CLR     R2
(1)  013174  112046                     2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
(1)  013176  001420                             BEQ     3$              ;;IF ZERO GET OUT
(1)  013200  122716  000060                     CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
(1)  013204  003026                             BGT     4$              ;;IS AN OCTAL DIGIT
(1)  013206  122716  000067                     CMPB    #'7,(SP)
(1)  013212  002423                             BLT     4$
(1)  013214  006301                             ASL     R1              ;;*2
(1)  013216  006102                             ROL     R2
(1)  013220  006301                             ASL     R1              ;;*4
(1)  013222  006102                             ROL     R2
(1)  013224  006301                             ASL     R1              ;;*8
(1)  013226  006102                             ROL     R2
(1)  013230  042716  177770                     BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
(1)  013234  062601                             ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
(1)  013236  000756                             BR      2$              ;;LOOP
(1)  013240  005726                     3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
(1)  013242  010166  000012                     MOV     R1,12(SP)       ;;SAVE THE RESULT
(1)  013246  010267  000026                     MOV     R2,$HIOCT
(3)  013252  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  013254  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  013256  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  013260  000002                             RTI                     ;;RETURN
(1)  013262  005726                     4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
(1)  013264  105010                             CLRB    (R0)            ;;SET A TERMINATOR
(1)  013266  104400                             TYPE                    ;;TYPE UP THRU THE BAD CHAR.
(1)  013270  000000                     5$:     .WORD   0
(1)  013272  104400  001224                     TYPE    ,$QUES          ;;"?" "CR" & "LF"
(1)  013276  000730                             BR      1$              ;;TRY AGAIN
(1)  013300  000000                     $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
1360                                    ;******************************************************************
(1)
(1)                                     .SBTTL  READ A DECIMAL NUMBER FROM THE TTY
(1)
```

```
(1)                                     ;*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
(1)                                     ;*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
(1)                                     ;*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
(1)                                     ;*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
(1)                                     ;*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
(1)                                     ;*POSITIVE 32767 TO NEGATIVE 32768.
(1)                                     ;*CALL:
(1)                                     ;*      RDDEC                   ;;READ A DECIMAL NUMBER
(1)                                     ;*      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
(1)                                     ;
(1)
(1)  013302  011646                     $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
(1)  013304  016666  000004  000002             MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
(3)  013312  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  013314  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  013316  010246                             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  013320  104406                     1$:     RDLIN                   ;;READ AN ASCIZ LINE
(1)  013322  012600                             MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
(1)  013324  010067  000120                     MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
(1)  013330  005046                             CLR     -(SP)           ;;CLEAR DATA WORD
(1)  013332  005002                             CLR     R2              ;;SIGN SET POSITIVE
(1)  013334  122710  000055                     CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
(1)  013340  001001                             BNE     2$              ;;BR IF NO MINUS SIGN
(1)  013342  112002                             MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
(1)  013344  112001                     2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
(1)  013346  001424                             BEQ     3$              ;;GET OUT IF ZERO
(1)  013350  122701  000060                     CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
(1)  013354  003032                             BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
(1)  013356  122701  000071                     CMPB    #'9,R1
(1)  013362  002427                             BLT     5$
(1)  013364  032716  170000                     BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
(1)  013370  001024                             BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
(1)  013372  006316                             ASL     (SP)            ;;*2
(1)  013374  011646                             MOV     (SP),-(SP)      ;;SAVE FOR LATER
(1)  013376  006316                             ASL     (SP)            ;;*4
(1)  013400  006316                             ASL     (SP)            ;;*8.
(1)  013402  062616                             ADD     (SP)+,(SP)      ;;*10.
(1)  013404  102416                             BVS     5$              ;;OVERFLOW ISN'T ALLOWED
(1)  013406  162701  000060                     SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
(1)  013412  060116                             ADD     R1,(SP)         ;;ADD IN THIS DIGIT
(1)  013414  102412                             BVS     5$              ;;O  RFLOW ISN'T ALLOWED
(1)  013416  000752                             BR      2$              ;;LOOP
(1)  013420  005702                     3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
(1)  013422  001401                             BEQ     4$              ;;BR IF NO
(1)  013424  005416                             NEG     (SP)            ;;YES--NEGATE THE NUMBER
(1)  013426  012666  000012             4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
(3)  013432  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  013434  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  013436  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  013440  000002                             RTI                     ;;RETURN
(1)
(1)  013442  005726                     5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
(1)  013444  105010                             CLRB    (R0)            ;;SET A TERMINATOR
(1)  013446  104400                             TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
```

```
   (1)  013450  ^^0000                  6$:     .WORD   0               ;;POINTER GOES HERE
   (1)  013452    1400  001224                  TYPE    $QUES           ;;"?" "CR" &"LF"
   (1)  013456  ..J720                  BR      1$                      ;;TRY AGAIN
  136;                                  ;*************************************************************************
   (1)
   (1)                                  .SBTTL  TRAP DECODER
   (1)
   (1)                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
   (1)                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
   (1)                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
   (1)                                  ;*GO TO THAT ROUTINE.
   (1)
   (1)  013460  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
   (1)  013462  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
   (1)  013466  005740                          TST     -(R0)           ;;BACKUP BY 2
   (1)  013470  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
   (1)  013472  006300                          ASL     R0              ;;POSITION FOR INDEXING
   (1)  013474  016000  013502                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
   (1)  013500  000200                          RTS     R0              ;;GO TO ROUTINE
   (1)
   (3)
   (3)                                  .SBTTL  TRAP TABLE
   (3)
   (3)                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
   (3)                                  ;*BY THE "TRAP" INSTRUCTION.
   (3)
   (3)                                  ;       ROUTINE
   (3)                                  ;       -------
   (3)  013502                          $TRPAD:
   (3)  013502  012244                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
   (3)  013504  011616                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
   (3)  013506  011572                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
   (3)  013510  011632                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
   (3)  013512  012020                          $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)
   (3)  013514  012770                          $RDCHR  ;;CALL=RDCHR    TRAP+5(104405)  TTY TYPEIN CHARACTER ROUTINE
   (3)  013516  013024                          $RDLIN  ;;CALL=RDLIN    TRAP+6(104406)  TTY TYPEIN STRING ROUTINE
   (3)  013520  013142                          $RDOCT  ;;CALL=RDOCT    TRAP+7(104407)  READ AN OCTAL NUMBER FROM TTY
   (3)  013522  013302                          $RDDEC  ;;CALL=RDDEC    TRAP+10(104410) READ A DECIMAL NUMBER FROM TTY
 1362                                   ;*************************************************************************
   (1)
   (1)                                  .SBTTL  POWER DOWN AND UP ROUTINES
   (1)
   (1)                                  ;POWER DOWN ROUTINE
   (1)  013524  012737  013652  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
   (1)  013532  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
   (3)  013540  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
   (3)  013542  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
   (3)  013544  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
   (3)  013546  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
   (3)  013550  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
   (3)  013552  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
   (1)  013554  010667  000076                  MOV     SP,$SAVR6       ;;SAVE SP
   (1)  013560  012737  013572  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
   (1)  013566  000000                          HALT
```

```
(1)  013570  000776                        BR      .-2             ;;HANG UP
(1)
(1)                                 ;POWER UP ROUTINE
(1)  013572  016706  000060        $PWRUP: MOV     $SAVR6,SP       ;;GET SP
(1)  013576  005067  000054                CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
(1)  013602  005267  000050        1$:     INC     $SAVR6          ;;WAIT FOR THE INC
(1)  013606  001375                        BNE     1$              ;;OF  WORD
(3)  013610  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  013612  012604                        MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  013614  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  013616  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  013620  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  013622  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  013624  012737  013524  000024        MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
(1)  013632  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
(1)  013640  104400                        TYPE                    ;REPORT THE POWER FAILURE
(1)  013642  013660                $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
(1)  013644  012716                        MOV     (PC)+,(SP)      ;;RESTART AT CKRST1
(1)  013646  013670                $PWRAD: .WORD   CKRST1          ;;RESTART ADDRESS
(1)  013650  000002                        RTI
(1)  013652  000000                $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
(1)  013654  000776                        BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
(1)  013656  000000                $SAVR6: 0                       ;;PUT THE SP HERE
(1)  013660  005015  047520  042527 $POWER: .ASCIZ  <15><12>"POWER"
(1)  013666  000122
(1)                                        .EVEN
```

```
1365                                    ;*****************************************************
1366                                    ;COMMON DH11 SERVICE ROUTINES
1367                                    ;*****************************************************
1368
1369                                    ;THESE ROUTINES DETERMINE RESTART ADDRESS AFTER SYSTEM ERROR
1370                                    ;(BUS ERROR,RSVD INSTR ERROR, OR POWER FAIL)
1371
1372  013670  005767  004176      CKRST1: TST     DPFLG           ;IN PATTERNS TEST ?
1373  013674  001005                      BNE     1$              ;BR IF YES
1374  013676  005767  004202              TST     RETFLG          ;IN ECHO TEST ?
1375  013702  001004                      BNE     2$              ;BR IF YES
1376  013704  000167  166422              JMP     RSTRTA          ;GO RESTART RELIABILITY TESTS
1377  013710  000167  172070      1$:     JMP     EXPAT           ;GO TO PATTERNS TESTS
1378  013714  000167  170664      2$:     JMP     ECHO            ;GO TO ECHO TESTS
1379
1380  013720  005767  004146      CKRST2: TST     DPFLG           ;IN PATTERNS TEST ?
1381  013724  001005                      BNE     1$              ;BR IF YES
1382  013726  005767  004152              TST     RETFLG          ;IN ECHO TEST ?
1383  013732  001004                      BNE     2$              ;BR IF YES
1384  013734  000167  166362              JMP     RESTI           ;GO RESTART RELIABILITY TESTS
1385  013740  000167  172040      1$:     JMP     EXPAT           ;GO TO PATTERNS TESTS
1386  013744  000167  170634      2$:     JMP     ECHO            ;GO TO ECHO TESTS
1387
1388
1389                                    ;THIS ROUTINE IS CALLED TO SET UP THE DH11 PARAMETERS PRIOR TO TEST
1390
1391  013750  012711  004000      DHSET1: MOV     #BIT11,(R1)     ;CLEAR THE DH11 UNDER TEST
1392  013754  004767  003376              JSR     PC,CHPS2        ;GO LOCK OUT INTRS
1393  013760  012711  030100              MOV     #30100,(R1)     ;ENABLE INTERRUPTS ON XMIT DONE
1394                                                              ;NON-EX MEM, DATA AVAIL, OR SILO OVFLW
1395  013764  156711  003706              BISB    LINE,(R1)       ;SELECT THE LINE NO.
1396  013770  005067  003530              CLR     RDONE           ;CLEAR SOFTWARE DONE FLAG
1397  013774  012767  025732  011724      MOV     #RBUF,RBFPTR    ;SET UP RCVR BUFFER POINTER
1398  014002  012767  025732  003516      MOV     #RBUF,RBFEND    ;MARK END OF THIS BUFFER
1399  014010  016705  003502              MOV     CHRCNT,RS       ;GET CHAR COUNT
1400  014014  005405                      NEG     RS              ;MAKE IT POSITIVE
1401  014016  060567  003504              ADD     RS,RBFEND
1402  014022  016761  003464  000004      MOV     CURLPR,LPR(R1)  ;LOAD THE LPR REG
1403  014030  016761  003462  000010      MOV     CHRCNT,BCR(R1)  ;LOAD THE BYTE COUNT REG
1404  014036  012761  030212  000006      MOV     #TBUF,CAR(R1)   ;LOAD CURRENT ADDRESS REG
1405  014044  000207                      RTS     PC              ;RETURN
1406
1407                                    ;THIS ROUTINE IS CALLED TO SELECT A NEW LINE NO. BASED ON THE
1408                                    ;VALUE OF THE LINE SELECTION PARAMETER
1409
1410                                            ;CALLING SEQUENCE:
1411
1412                                            ;JSR     PC,SELINE      ;CALL THE ROUTINE
1413                                            ;BR      1$             ;EXIT BRANCH-ROUTINE MOVES THE RETURN
1414                                                                    ;PC AROUND THIS BR IF MORE LINES ARE
1415                                                                    ;YET TO BE TESTED
1416
1417  014046  105767  003625      SELINE: TSTB    LINE+1          ;FIRST TIME THROUGH FOR ANY TEST ?
1418  014052  001010                      BNE     1$              ;BR IF NOT
```

```
1419  014054  105167  003617                      COMB    LINE+1          ;SET ENTRY FLAG
1420  014060  012767  000001  003362              MOV     #1,LINMSK       ;INIT SELECT TEST MASK TO TEST LINE 00
1421  014066  105067  003604                      CLRB    LINE            ;START WITH LINE #00
      014072  000405                              BR      2$              ;GO TEST FOR LINE #00
      014074  105267  003576          1$:         INCB    LINE            ;GENERATE NEW LINE NO.
      014100  006367  003344                      ASL     LINMSK          ;SHIFT SELECT MASK TO TEST NXT LINE
      014104  001407                              BEQ     3$              ;RETURN TO EXIT BRANCH - ALL LINES DONE
      014106  036767  003336  003332  2$:         BIT     LINMSK,LINSEL   ;IS THE LINE SELECTED FOR TEST ??
      014114  001767                              BEQ     1$              ;BR IF NOT
      014116  062716  000002                      ADD     #2,(SP)         ;MOVE RETURN PC AROUND EXIT BRANCH
      014122  000402                              BR      4$              ;RETURN TO TEST SELECTED LINE
      014124  005067  003546          3$:         CLR     LINE            ;INIT ENTRY FLAG AND LINE NO. TO 000
      014130  142777  000017  003300  4$:         BICB    #17,@DHADR      ;INIT LINE SELECT BITS IN "SCR"
      014136  000207                              RTS     PC              ;RETURN TO CALLING TEST

                                  ;THIS ROUTINE IS CALLED TO CONVERT EITHER THE "DM" NUMBER OR THE
                                  ;"LINE" NUMBER TO TWO ASCII CHARACTERS AND MOVE THEM INTO A
                                  ;PARTICULAR MESSAGE BUFFER FOR ERROR REPORTING

                                          ;CALLING SEQUENCE

                                          ;JSR     R5,SUNUM       ;CALL TO THIS ROUTINE
                                          ;ADDR1                  ;ADDRESS OF THE NUMBER TO BE CONVERTED
                                          ;ADDR2                  ;ADDRESS OF THE MSG BUFFER SLOT

      014140                      SUNUM:
(2)   014140  010046                      MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(2)   014142  010146                      MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(2)   014144  010246                      MOV     R2,-(SP)        ;;PUSH R2 ON STACK
1445  014146  012500                      MOV     (R5)+,R0        ;GET ADDRESS OF NUMBER
1446  014150  012501                      MOV     (R5)+,R1        ;GET MSG BUFFER ADDR
1447  014152  111000                      MOVB    (R0),R0         ;GET NO. TO BE CONVERTED
1448  014154  010002                      MOV     R0,R2           ;SAVE IT IN R2
1449  014156  006202                      ASR     R2              ;SHIFT MSD TO LSD POSITION
1450  014160  006202                      ASR     R2
1451  014162  006202                      ASR     R2
1452  014164  042702  177770             BIC     #177770,R2      ;CLR JUNK BITS
1453  014170  062702  000060             ADD     #60,R2          ;MAKE IT ASCII
1454  014174  110221                      MOVB    R2,(R1)+        ;PUT IT IN MSG BUFFER
1455  014176  042700  177770             BIC     #177770,R0      ;CLR JUNK FROM LSD
1456  014202  062700  000060             ADD     #60,R0          ;MAKE IT ASCII
1457  014206  110011                      MOVB    R0,(R1)         ;PUT LSD IN THE BUFFER
1458  014210  012602                      MOV     (SP)+,R2        ;;POP STACK INTO R2
(2)   014212  012601                      MOV     (SP)+,R1        ;;POP STACK INTO R1
(2)   014214  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
1459  014216  000205                      RTS     R5              ;RETURN TO CALLER
1460
1461                              ;THIS ROUTINE IS CALLED TO SET UP THE ERROR INFORMATION IN THE
1462                              ;MESSAGE BUFFERS
1463
1464  014220  010067  164734      SUER2:  MOV     R0,$REG0        ;STORE THE REGS IN CORE
1465  014224  010167  164732      SUER1:  MOV     R1,$REG1
1466  014230  010267  164730              MOV     R2,$REG2
1467  014234  010367  164726              MOV     R3,$REG3
```

```
1468  014240  010467  164724                    MOV     R4,$REG4
1469  014244  000207                            RTS     PC              ;RETURN TO REPORT ERROR
1470
1471                                    ;THIS ROUTINE IS USED TO ACCEPT INPUT PARAMETERS FROM THE CONSOLE
1472                                    ;TELETYPE
1473
1474  014246  104400                    INPARA: TYPE
1475  014250  023164                            VCMC                    ;"ASK FOR NO. WORDS BETWEEN VECTORS"
1476  014252  104407                            RDOCT                   ;READ OCTAL NO. FM TTY
1477  014254  012600                            MOV     (SP)+,R0        ;GET THE NO. HE TYPED
1478  014256  001407                            BEQ     2$              ;BR IF HE TYPED <CR>
1479  014260  022700  000004                    CMP     #4,R0           ;FOUR WORDS BETWEEN VECTORS ?
1480  014264  001404                            BEQ     2$              ;BR IF YES
1481  014266  022700  000010                    CMP     #10,R0          ;8. WORDS BETWEEN VECTORS ??
1482  014272  001404                            BEQ     3$              ;BR IF YES
1483  014274  000764                            BR      INPARA          ;ASK ALL OVER AGAIN
1484  014276  012700  000010            2$:     MOV     #10,R0          ;SET UP CONSTANT IN R0 FOR 4 WORDS
1485  014302  000402                            BR      4$              ;CONTINUE
1486  014304  012700  000020            3$:     MOV     #20,R0          ;SET UP CONSTANT FOR 8. WORDS
1487  014310  005067  003556            4$:     CLR     DPFLG           ;CLEAR PATTERNS TESTS FLAG
1488  014314  005067  003564                    CLR     RETFLG          ;INIT ECHO TEST RETURN FLAG
1489  014320  000207                            RTS     PC              ;RETURN TO CALLER
1490
1491  014322  012767  177777  003300    INPARX: MOV     #-1,VCFLG       ;SET START AT 200 FLAG
1492  014330  000167  165306                    JMP     BEGINA          ;GO START UP
1493  014334  012700  177777            INPARC: MOV     #-1,R0          ;SET FLAG IN R0
1494  014340  005067  003264                    CLR     VCFLG           ;INIT VECTOR SET UP FLAG
1495  014344  005067  003522                    CLR     DPFLG           ;CLEAR PATTERNS TESTS FLAG
1496  014350  005067  003530                    CLR     RETFLG          ;INIT ECHO TEST RETURN FLAG
1497  014354  000167  165262                    JMP     BEGINA          ;GO ASK FOR SELECT PARAMETER
1498
1499  014360  104400                    INPAR:  TYPE                    ;ASK FOR DEVICE ADDRESS
1500  014362  022473                            INMSG1
1501  014364  104407                            RDOCT                   ;READ IN WHAT IS TYPED
1502  014366  012601                            MOV     (SP)+,R1        ;GET THE NO. HE TYPED
1503  014370  001403                            BEQ     INPAR1          ;BR IF DEFAULT
1504  014372  004767  000130                    JSR     PC,CHKADR       ;GO CHECK VALIDITY OF THE ADDR
1505  014376  000770                            BR      INPAR           ;ERROR BRANCH
1506
1507  014400  104400                    INPAR1: TYPE                    ;ASK FOR VECTOR ADDRESS
1508  014402  022537                            INMSG2
1509  014404  104407                            RDOCT                   ;READ IN WHAT HE TYPES
1510  014406  012601                            MOV     (SP)+,R1        ;GET THE ADDRESS
1511  014410  001403                            BEQ     INPAR3          ;BR IF DEFAULT
1512  014412  004767  000222                    JSR     PC,CHKVCT       ;GO CHECK VALIDITY OF VECTOR
1513  014416  000770                            BR      INPAR1          ;ERROR BRANCH
1514
1515  014420  005767  003460    INPAR3: TST     RETFLG                  ;LINE ECHO TESTS ?
1516  014424  001402                            BEQ     1$              ;BR IF NOT
1517  014426  000167  170220                    JMP     ECHO2           ;RETURN TO LINE ECHO TESTS
1518  014432  005767  003434            1$:     TST     DPFLG           ;DATA PATTERNS TESTS ACTIVE ??
1519  014436  001402                            BEQ     2$              ;BR IF NOT
1520  014440  000167  171406                    JMP     EXPAT2          ;GO BACK TO PATTERNS TESTS
1521  014444  104400            2$:     TYPE                            ;ASK FOR DEVICE SELECTION PARAMETER
```

```
1522  014446  022606                          INMSG3
1523  014450  104407                          RDOCT
1524  014452  012601                          MOV     (SP)+,R1        ;GET THE SELECT PARAMETER
1525  014454  001402                          BEQ     INPAR4          ;BR IF  DEFAULT
1526  014456  010167  002762                  MOV     R1,DHSEL        ;SET UP DH11 SELECTION PARAMETER
1527  014462  012767  177777  002756  INPAR4: MOV     #-1,LINSEL      ;INIT FOR ALL 16. LINES
1528  014470  104400                          TYPE                    ;ASK FOR LINE SELECT PARAMETER
1529  014472  023122                          INMSGA
1530  014474  104407                          RDOCT                   ;READ WHAT HE TYPES
1531  014476  012601                          MOV     (SP)+,R1        ;GET IT OFF STACK
1532  014500  001402                          BEQ     1$              ;BR IF DEFAULT
1533  014502  010167  002740                  MOV     R1,LINSEL       ;SET LINE SELECT PARAMETER
1534  014506  005777  164424          1$:     TST     @SWR            ;HALT AFTER SET UP ??
1535  014512  100003                          BPL     EXPAR           ;BR IF  NOT
1536  014514  104400                          TYPE                    ;TYPE CONTINUE MESSAGE PRIOR TO HALTING
1537  014516  023052                          INMSG7
1538  014520  000000                          HALT                    ;DEPRESS CONTINUE TO RESUME TESTING
1539  014522  000167  165500          EXPAR:  JMP     START2          ;GO START UP THE PROGRAM
1540
1541
1542  014526  021127  160020          CHKADR: CMP     R1,#160020      ;IS ADDRESS ABOVE OR EQUAL TO LOW LIMIT
1543  014532  002001                          BGE     1$              ;BR IF YES
1544  014534  000436                          BR      4$              ;BR IF NOT
1545  014536  020127  160420          1$:     CMP     R1,#160420      ;IS IT BELOW THE HIGH LIMIT?
1546  014542  002401                          BLT     2$              ;BR IF YES
1547  014544  000432                          BR      4$              ;BR IF NOT
1548  014546  032701  000017          2$:     BIT     #17,R1          ;CORRECT BOUNDARY ?
1549  014552  001027                          BNE     4$              ;BR IF NOT
1550  014554  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN PC AROUND ERROR BRANCH
1551  014560  005767  003320                  TST     RETFLG          ;ARE WE IN ECHO TESTS ?
1552  014564  001403                          BEQ     21$             ;BR IF NOT
1553  014566  010167  002644                  MOV     R1,DHADR        ;SET UP DH11 DEVICE ADDRESS
1554  014572  000421                          BR      5$              ;CONTINUE
1555  014574  005767  003272          21$:    TST     DPFLG           ;PATTERNS TESTS ACTIVE ??
1556  014600  001403                          BEQ     22$             ;BR IF NOT
1557  014602  010167  002630                  MOV     R1,DHADR        ;SET UP DEVICE ADDRESS
1558  014606  000413                          BR      5$              ;CONTINUE
1559  014610  012702  017530          22$:    MOV     #DHADTB,R2      ;POINT TO BEGIN OF ADDR TABLE
1560  014614  010122                  3$:     MOV     R1,(R2)+        ;SET UP A TABLE ENTRY
1561  014616  062701  000020                  ADD     #20,R1          ;GENERATE NEXT DH11 ADDR
1562  014622  022702  017570                  CMP     #DHADTB+40,R2   ;END OF TABLE ?
1563  014626  001372                          BNE     3$              ;BR IF NOT
1564  014630  000402                          BR      5$              ;RETURN TO INPUT ROUTINES
1565  014632  104400                  4$:     TYPE                    ;TELL HIM HE GOOFED
1566  014634  022657                          INMSG4
1567  014636  000207                  5$:     RTS     PC              ;RETURN TO INPUT ROUTINES
1568
1569  014640  020127  000300          CHKVCT: CMP     R1,#300         ;IS ADDRESS ABOVE OR EQUAL TO LOW LIMIT
1570  014644  002001                          BGE     1$              ;BR IF YES
1571  014646  000435                          BR      4$              ;BR IF NOT
1572  014650  020127  001000          1$:     CMP     R1,#1000        ;IS IT BELOW THE HIGH LIMIT?
1573  014654  002401                          BLT     2$              ;BR IF YES
1574  014656  000431                          BR      4$              ;BR IF NOT
1575  014660  032701  000007          2$:     BIT     #7,R1           ;CORRECT BOUNDARY ?
```

```
1576  014664  001026                        BNE     4$              ;BR IF NOT
1577  014666  062716  000002                ADD     #2,(SP)         ;MOVE RETURN PC AROUND ERROR BRANCH
1578  014672  005767  003206                TST     RETFLG          ;ARE WE IN ECHO TESTS ?
1579  014676  001403                        BEQ     21$             ;BR IF NOT
1580  014700  010167  002534                MOV     R1,DHVCT        ;SET UP DH11 VECTOR ADDR
1581  014704  000420                        BR      5$              ;CONTINUE
1582  014706  005767  003160        21$:    TST     DPPFLG          ;PATTERNS TESTS ACTIVE ??
1583  014712  001403                        BEQ     22$             ;BR IF NOT
1584  014714  010167  002520                MOV     R1,DHVCT        ;SET UP DEVICE VECTOR
1585  014720  000412                        BR      5$              ;CONTINUE
1586  014722  012702  017570        22$:    MOV     #DHVCTB,R2      ;POINT TO BEGIN OF VECTOR TABLE
1587  014726  010122                3$:     MOV     R1,(R2)+        ;SET UP A TABLE ENTRY
1588  014730  060001                        ADD     R0,R1           ;GENERATE NEXT DH11 ADDR
1589  014732  022702  017630                CMP     #DHVCTB+40,R2   ;END OF TABLE ?
1590  014736  001373                        BNE     3$              ;BR IF NOT
1591  014740  000402                        BR      5$              ;RETURN TO INPUT ROUTINES
1592  014742  104400                4$:     TYPE                    ;TELL HIM HE GOOFED
1593  014744  022730                        INMSGS
1594  014746  000207                5$:     RTS     PC              ;RETURN TO INPUT ROUTINES
1595
1596                                        ;THESE TWO ROUTINES SERVICE UNEXPECTED BUS ERROR AND RSVD INSTR TRAPS
1597
1598  014750  010667  164220        BUSER:  MOV     SP,$REG6        ;SAVE THE SP
1599  014754  012667  164202                MOV     (SP)+,$REG1     ;GET THE TRAP PC
1600  014760  012667  164200                MOV     (SP)+,$REG2     ;GET THE TRAP PSW
1601  014764  012706  001100                MOV     #STACK,SP       ;RESET THE STACK POINTER
1602  014770  012767  015000  164112        MOV     #1$,$LPERR      ;ALWAYS COME BACK TO 1$
1603  014776  104014                        ERROR   14              ;UNEXPECTED BUS ERROR TRAP
1604  015000  000005                1$:     RESET                   ;PREPARE TO RESTART
1605  015002  004767  002334                JSR     PC,CHF          ;GO CLEAR PSW
1606  015006  000167  176706                JMP     CKRST2          ;GO RESTART THE PROGRAM
1607
1608  015012  010667  164156        RESERR: MOV     SP,$REG6        ;SAVE THE SP
1609  015016  012667  164140                MOV     (SP)+,$REG1     ;GET THE TRAP PC
1610  015022  012667  164136                MOV     (SP)+,$REG2     ;GET THE TRAP PSW
1611  015026  012706  001100                MOV     #STACK,SP       ;RESET THE STACK POINTER
1612  015032  012767  015042  164050        MOV     #1$,$LPERR      ;ALWAYS COME BACK TO 1$
1613  015040  104015                        ERROR   15              ;UNEXPECTED RSVD INSTR ERROR TRAP
1614  015042  000005                1$:     RESET                   ;PREPARE TO RESTART
1615  015044  004767  002272                JSR     PC,CHPS1        ;GO CLEAR PSW
1616  015050  000167  176644                JMP     CKRST2          ;GO RESTART THE PROGRAM
1617
1618                                        ;THIS ROUTINE IS CALLED WHEN A TEST NEEDS TO RESTORE THE TRAP
1619                                        ;CATCHER IN THE DH11 VECTOR
1620
1621  015054  016703  002360        RESTRP: MOV     DHVCT,R3        ;GET VECTOR ADDRESS
1622  015060  010313                        MOV     R3,(R3)         ;RESTORE THE TRAP CATCHER
1623  015062  062723  000002                ADD     #2,(R3)+
1624  015066  005023                        CLR     (R3)+
1625  015070  010313                        MOV     R3,(R3)
1626  015072  062723  000002                ADD     #2,(R3)+
1627  015076  005023                        CLR     (R3)+
1628  015100  000207                        RTS     PC              ;RETURN TO CALLING TEST
1629
```

```
                                        ;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

015102  005267  002760          TIMEIT: INC     TIMEB           ;COUNT B
015106  001005                          BNE     1$              ;BR IF NOT ZERO
015110  005367  002750                  DEC     TIMEA           ;COUNT TIME A
015114  001002                          BNE     1$              ;BR IF NO TIMEOUT
015116  062716  000002                  ADD     #2,(SP)         ;MOVE RETURN PC TO ALLOW ERROR REPORT
015122  000207                  1$:     RTS     PC              ;RETURN TO THE CALLING TEST



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

015124  012705  025416          CLSTAT: MOV     #RTOTAL,R5      ;SET UP POINTER TO BEGINNING
015130  005025                  1$:     CLR     (R5)+           ;CLEAR ONE WORD
015132  022705  025716                  CMP     #RTOTAL+192.,R5 ;CLEARED ALL ENTRIES ??
015136  001374                          BNE     1$              ;BR IF NOT
015140  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

015142  022767  017512  012344  SETLPR: CMP     #CURLPR,LPRPTR  ;DONE ALL 13. ENTRIES ??
015150  001425                          BEQ     3$              ;BR IF YES
015152  017767  002336  002332          MOV     @LPRPTR,CURLPR  ;GET THE LPR CONSTANT
015160  105777  163752                  TSTB    @#SWR           ; QUICK TEST ?
015164  100010                          BPL     1$              ;BR IF NOT - SUPPLY THE WHOLE THING
015166  022767  033500  002316          CMP     #33500,CURLPR   ;9600 BAUD TEST ??
015174  001404                          BEQ     1$              ;BR IF YES
015176  012767  177777  002250          MOV     #-1,QUICK       ;SET QUICK TEST FLAG
015204  000402                          BR      2$              ;CONTINUE
015206  005067  002242          1$:     CLR     QUICK           ;DO FULL TESTING AT 9600. BAUD
015212  062767  000002  002274  2$:     ADD     #2,LPRPTR       ;UPDATE THE TABLE POINTER
015220  062705  000002                  ADD     #2,R5           ;MOVE PC AROUND ERROR BRANCH
015224  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
```

```
 1684  015226  005767  002224          SETCL:  TST     QUICKX          ;EXIT AFTER ONLY ONE CHAR LNGTH ?
 1685  015232  001034                          BNE     2$              ;BR IF YES
 1686  015234  005267  002260                  INC     CLSEL           ;GENERATE NEW CHAR LNGTH SELECT CODE
 1687  015240  022767  000004  002252          CMP     #4,CLSEL        ;DONE FOUR OF THEM ??
 1688  015246  001426                          BEQ     2$              ;BR IF YES
 1689  015250  005767  002200                  TST     QUICK           ;QUICK TEST FLAG SET ?
 1690  015254  001407                          BEQ     1$              ;BR IF NOT
 1691  015256  005267  002174                  INC     QUICKX          ;SET QUICK TEST EXIT FLAG
 1692  015262  005067  002232                  CLR     CLSEL           ;DO ONLY 5 BIT CHARS
 1693  015266  012767  177760  002222          MOV     #177760,CHRCNT  ;DO  ONLY 32. CHAR BUFFER
 1694  015274  042767  000003  002210  1$:     BIC     #3,CURLPR       ;SET UP THE CURRENT LPR
 1695  015302  056767  002212  002202          BIS     CLSEL,CURLPR
 1696  015310  006367  002202                  ASL     CHRCNT          ;GENERATE CHAR COUNT
 1697  015314  004767  000006                  JSR     PC,SUBUF1       ;GO SET UP THE OUTPUT BUFFER
 1698  015320  062705  000002                  ADD     #2,R5           ;MOVE PC AROUND EXIT BRANCH
 1699  015324  000205                  2$:     RTS     R5              ;RETURN
 1700
 1701                                          ;THIS ROUTINE IS CALLED TO LOAD THE OUTPUT DATA BUFFER WITH THE
 1702                                          ;REQUIRED BINARY COUNT PATTERN
 1703
 1704                                          ;CALLING SEQUENCE:
 1705
 1706                                          ;       JSR     PC,SUBUF1       ;CALL
 1707
 1708  015326                          SUBUF1:
   (2) 015326  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
   (2) 015330  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
   (2) 015332  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 1709  015334  005004                          CLR     R4              ;INIT CHAR GENERATOR
 1710  015336  016703  002154                  MOV     CHRCNT,R3       ;SET UP LOAD COUNT
 1711  015342  012702  030212                  MOV     #TBUF,R2        ;SET UP BUFFER POINTER
 1712  015346  110422                  1$:     MOVB    R4,(R2)+        ;LOAD A CHAR
 1713  015350  005204                          INC     R4              ;GENERATE NEXT CHAR
 1714  015352  005203                          INC     R3              ;COUNT ONE LOADED
 1715  015354  001374                          BNE     1$              ;BR TIL BUFFER FULL
 1716  015356  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
   (2) 015360  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
   (2) 015362  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 1717  015364  000207                          RTS     PC              ;RETURN
 1718
 1719                                          ;THIS ROUTINE IS CALLED TO SET UP THE PARITY SELECT BITS
 1720                                          ;IN THE CURRENT LPR TEST CONSTANT
 1721
 1722                                          ;CALLING SEQUENCE:
 1723
 1724                                          ;       JSR     R5,SETPAR       ;CALL
 1725                                          ;       BR      NEXCL           ;EXIT BRANCH
 1726
 1727  015366  022767  177777  002126  SETPAR: CMP     #-1,PARBIT      ;DONE ALL PARITY COMBOS ?
 1728  015374  001444                          BEQ     5$              ;BR IF YES
 1729  015376  005767  002052                  TST     QUICK           ;QUICK TEST FLAG SET ?
 1730  015402  001403                          BEQ     1$              ;BR IF NOT
 1731  015404  012767  000060  002110          MOV     #60,PARBIT      ;CHECK ODD PARITY ONLY
 1732  015412  042767  000060  002072  1$:     BIC     #60,CURLPR      ;SET PARITY SELECT BITS
```

```
 1733  015420  056767  002076  002064          BIS     PARBIT,CURLPR
 1734  015426  005767  002070                  TST     PARBIT          ;SELECT BITS 00 ?
 1735  015432  001004                          BNE     2$              ;BR IF NOT
 1736  015434  012767  000020  002060          MOV     #20,PARBIT      ;SET SELECT BITS TO 01
 1737  015442  000417                          BR      4$              ;EXIT
 1738  015444  022767  000020  002050  2$:     CMP     #20,PARBIT      ;SELECT BITS 10 ?
 1739  015452  001004                          BNE     3$              ;BR IF NOT
 1740  015454  012767  000060  002040          MOV     #60,PARBIT      ;MAKE SELECT BITS 11
 1741  015462  000407                          BR      4$              ;EXIT
 1742  015464  022767  000060  002030  3$:     CMP     #60,PARBIT      ;SELECT BITS 11 ?
 1743  015472  001005                          BNE     5$              ;BR IF NOT
 1744  015474  012767  177777  002020          MOV     #-1,PARBIT      ;SET EXIT FLAG
 1745  015502  062705  000002          4$:     ADD     #2,R5           ;MOVE RETURN PC AROUND EXIT BRANCH
 1746  015506  000205                  5$:     RTS     R5              ;RETURN
 1747
 1748                                  ;THIS ROUTINE IS CALLED TO SET UP FOR KEYBOARD INTERRUPTS
 1749
 1750  015510  012767  015534  162342  KYBD1:  MOV     #KYBD2,60       ;SET UP THE INPUT VECTOR
 1751  015516  012767  000340  162336          MOV     #340,62
 1752  015524  012767  000100  162026          MOV     #100,177560     ;ENABLE KYBD INTRS
 1753  015532  000207                          RTS     PC              ;RETURN TO START TESTING
 1754
 1755                                  ;THIS ROUTINE SERVICES THE KEYBOARD INTERRUPT AND LOOKS FOR AN "S"
 1756                                  ;BEING TYPED TO INDICATE ABORT AND PRINT STATISTICS
 1757
 1758  015534  122767  000323  162020  KYBD2:  CMPB    #323,177562     ;WAS AN "S" TYPED ?
 1759  015542  001401                          BEQ     1$              ;BR IF YES
 1760  015544  000002                          RTI                     ;RETURN AND FORGET IT
 1761  015546  000005                  1$:     RESET                   ;ZAP THE WORLD
 1762  015550  012706  001100                  MOV     #STACK,SP       ;RESET THE SP
 1763  015554  004767  001562                  JSR     PC,CHPS1        ;GO CLEAR PSW
 1764  015560  000167  166540                  JMP     PRSTAT          ;GO DUMP THE STATISTICS
 1765
 1766                                  ;THIS ROUTINE SENDS A TEST BUFFER TO REMOTE DH11 LINE
 1767
 1768  015564  016701  001646          SENDP2: MOV     DHADR,R1                ;SET UP DH SCR ADDR
 1769  015570  012711  004000                  MOV     #BIT11,(R1)             ;CLEAR THE DH11
 1770  015574  016711  002076                  MOV     LINE,(R1)               ;SET LINE SELECT
 1771  015600  162705  030212                  SUB     #TBUF,R5                ;SET UP BYTE COUNT
 1772  015604  005405                          NEG     R5
 1773  015606  010561  000010                  MOV     R5,BCR(R1)
 1774  015612  012761  030212  000006          MOV     #TBUF,CAR(R1)           ;SET CURRENT ADDRESS
 1775  015620  016761  001666  000004          MOV     CURLPR,LPR(R1)          ;SET LINE PARAMETERS
 1776  015626  016761  001616  000012          MOV     LINMSK,BAR(R1)          ;ACTIVATE THE LINE
 1777
 1778  015634  005711                  1$:     TST     (R1)                    ;DONE TRANSMITTING ??
 1779  015636  100376                          BPL     1$                      ;BR IF NOT
 1780  015640  000207                          RTS     PC                      ;RETURN TO CONTROL ROUTINE "SENDP1"
 1781
 1782                                  ;THIS ROUTINE IS CALLED TO LOAD FILLERS INTO ECHO BUFFER
 1783
 1784  015642  116704  002316          LDFILL: MOVB    FILLB,R4                ;GET COUNT OF FILLERS
 1785  015646  012703  020121                  MOV     #ECBUF+1,R3             ;SET UP BUFFER POINTER
 1786  015652  116767  163322  002240          MOVB    $TMP0,ECBUF             ;STORE LF CHAR
```

```
1787  015660  116722  163314          MOVB    $TMPO,(R2)+     ;IN ECHO BUFFER TOO
1788  015664  116723  002272  1$:     MOVB    FILLA,(R3)+     ;LOAD A FILLER CHAR
1789  015670  116722  002266          MOVB    FILLA,(R2)+
1790  015674  005304                  DEC     R4              ;COUNT IT
1791  015676  001372                  BNE     1$              ;BR TIL REQUIRED COUNT LOADED
1792  015700  116704  002260          MOVB    FILLB,R4        ;SET UP BYTE COUNT REG
1793  015704  005204                  INC     R4
1794  015706  005404                  NEG     R4
1795  015710  010461  000010          MOV     R4,BCR(R1)      ;LOAD BCR REG
1796  015714  000207                  RTS     PC              ;RETURN TO RINT2
1797
1798                          ;THIS ROUTINE IS CALLED TO SET UP XMITTER SPEED
1799
1800  015716  104400          INXSP:  TYPE                    ;ASK USER TO TYPE SPEED
1801  015720  024046                  XSMSG1                  ;"TRANSMITTER SPEED ?"
1802  015722  012767  017702  001750  1$:     MOV     #XSPTAB,XSPTR   ;SET UP TABLE POINTER
1803  015730  042767  036000  001554          BIC     #36000,CURLPR   ;INIT SPEED SELECT BITS
1804  015736  104410                  RDDEC                   ;READ SPEED HE TYPED
1805  015740  005716                  TST     (SP)            ;DEFAULT TO 9600. BAUD ?
1806  015742  001426                  BEQ     4$              ;BR IF YES
1807  015744  027716  001730          2$:     CMP     @XSPTR,(SP)     ;TYPED ENTRY MATCH TABLE ENTRY ?
1808  015750  001010                  BNE     3$              ;BR IF NOT
1809  015752  062767  000002  001720          ADD     #2,XSPTR        ;POINT TO SELECT BITS IN TABLE
1810  015760  057767  001714  001524          BIS     @XSPTR,CURLPR   ;SET SPEED SELECT BITS
1811  015766  005726                  TST     (SP)+           ;FIX STACK
1812  015770  000417                  BR      5$              ;CONTINUE
1813
1814  015772  062767  000004  001700  3$:     ADD     #4,XSPTR        ;POINT TO NEXT ENTRY
1815  016000  022767  017766  001672          CMP     #XSPTAB+52.,XSPTR;END OF TABLE ??
1816  016006  001356                  BNE     2$              ;BR IF NOT
1817  016010  104400                  TYPE                    ;ERROR MESSAGE
1818  016012  024074                  XSMSG2                  ;"INVALID XMITR SPEED - TRY AGAIN"
1819  016014  005726                  TST     (SP)+           ;FIX THE SP
1820  016016  000741                  BR      1$              ;GO TRY AGAIN
1821
1822  016020  052767  032000  001464  4$:     BIS     #32000,CURLPR   ;SET UP DEFAULT TO 9600. BAUD
1823  016026  005726                  TST     (SP)+           ;FIX STACK POINTER
1824
1825  016030  000207          5$:     RTS     PC              ;RETURN TO CALLER
1826
1827                          ;THIS ROUTINE IS CALLED TO SET UP RECEIVER SPEED
1828
1829  016032  104400          INRSP:  TYPE                    ;ASK USER TO TYPE SPEED
1830  016034  024137                  RSMSG1                  ;"RECEIVER SPEED ?"
1831  016036  012767  017770  001722  1$:     MOV     #RSPTAB,RSPTR   ;SET UP TABLE POINTER
1832  016044  042767  001700  001440          BIC     #1700,CURLPR    ;INIT SPEED SELECT BITS
1833  016052  104410                  RDDEC                   ;READ SPEED HE TYPED
1834  016054  005716                  TST     (SP)            ;DEFAULT TO 9600. BAUD ?
1835  016056  001426                  BEQ     4$              ;BR IF YES
1836  016060  027716  001702          2$:     CMP     @RSPTR,(SP)     ;TYPED ENTRY MATCH TABLE ENTRY ?
1337  016064  001010                  BNE     3$              ;BR IF NOT
1838  016066  062767  000002  001672          ADD     #2,RSPTR        ;POINT TO SELECT BITS IN TABLE
1839  016074  057767  001666  001410          BIS     @RSPTR,CURLPR   ;SET SPEED SELECT BITS
1840  016102  005726                  TST     (SP)+           ;FIX STACK
```

```
1841  016104  000417                          BR      5$              ;CONTINUE
1842
1843  016106  062767  000004  001652  3$:     ADD     #4,RSPTR        ;POINT TO NEXT ENTRY
1844  016114  022767  020054  001644          CMP     #RSPTAB+52.,RSPTR;END OF TABLE ??
1845  016122  001356                          BNE     2$              ;BR IF NOT
1846  016124  104400                          TYPE                    ;ERROR MESSAGE
1847  016126  024162                          RSMSG2                  ;"INVALID RCVR SPEED - TRY AGAIN"
1848  016130  005726                          TST     (SP)+           ;FIX THE SP
1849  016132  000741                          BR      1$              ;GO TRY AGAIN
1850
1851  016134  052767  001500  001350  4$:     BIS     #1500,CURLPR    ;SET UP DEFAULT TO 9600. BAUD
1852  016142  005726                          TST     (SP)+           ;FIX STACK POINTER
1853
1854  016144  000207                  5$:     RTS     PC              ;RETURN TO CALLER
1855
1856
1857                                  ;THIS ROUTINE IS CALLED TO SET UP LINE PARAMETERS FM KYBD
1858
1859  016146  105067  005306          LPRIN:  CLRB    EC2             ;CLEAR ECHO BUFFER
1860  016152  104400                          TYPE
1861  016154  023774                          LPMSG                   ;"DO YOU WANT TO CHANGE "LPR"?"
1862  016156  104405                  1$:     RDCHR
1863  016160  012600                          MOV     (SP)+,R0        ;GET WHAT HE TYPED
1864  016162  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR> ??
1865  016166  001405                          BEQ     2$              ;BR IF YES
1866  016170  110067  005264                  MOVB    R0,EC2          ;ECHO WHAT HE TYPED
1867  016174  104400                          TYPE
1868  016176  000234                          EC2
1869  016200  000766                          BR      1$              ;GO WAIT FOR TERMINATOR
1870
1871  016202  105767  005252          2$:     TSTB    EC2             ;<CR> ONLY ??
1872  016206  001411                          BEQ     3$              ;BR IF YES
1873  016210  122767  000116  005242          CMPB    #116,EC2        ;WAS IT A "NO" ??
1874  016216  001405                          BEQ     3$              ;BR IF IT WAS
1875  016220  122767  000131  005232          CMPB    #131,EC2        ;WAS IT A "YES" ??
1876  016226  001347                          BNE     LPRIN           ;GO ASK ALL OVER AGAIN
1877  016230  000407                          BR      4$              ;BR IF IT WAS "YES"
1878  016232  005767  001254          3$:     TST     CURLPR          ;HAS LPR BEEN SET UP AT ALL ?
1879  016236  001016                          BNE     5$              ;BR IF YES USE PREVIOUS LPR
1880  016240  012767  033503  001244          MOV     #33503,CURLPR   ;SET DEFAULT 9600 BAUD,8 BITS NO PARITY
1881  016246  000412                          BR      5$              ;CONTINUE
1882  016250  004767  177442          4$:     JSR     PC,INXSP        ;GO INPUT AND SET UP XMIT SPEED
1883  016254  004767  177552                  JSR     PC,INRSP        ;GO INPUT AND SET UP RCVR SPEED
1884  016260  004767  000022                  JSR     PC,INCL         ;GO INPUT AND SET UP CHAR LENGTH
1885  016264  004767  000162                  JSR     PC,INSB         ;GO INPUT AND SET UP NO. OF STOP BITS
1886  016270  004767  000274                  JSR     PC,INPB         ;GO INPUT AND SET UP PARITY SELECTION
1887  016274  004767  000410          5$:     JSR     PC,INFCHR       ;GO INPUT AND SET UP FILLER CHAR
1888  016300  004767  000474                  JSR     PC,INFCNT       ;GO INPUT AND SET UP FILLER COUNT
1889  016304  000207                          RTS     PC              ;RETURN TO CALLER
1890
1891                                  ;THIS ROUTINE IS CALLED TO SET UP CHAR LENGTH BITS
1892
1893  016306  105067  005146          INCL:   CLRB    EC2             ;CLEAR THE ECHO BUFFER
1894  016312  104400                          TYPE                    ;ASK FOR INPUT
```

```
1895  016314  024225                          CLMSG1                  ;"CHAR LENGTH - 6,7, OR 8 ?"
1896  016316  042767  000003  001166  1$:     BIC     #3,CURLPR       ;INIT CHAR LENGTH SELECT BITR
1897  016324  104405                          RDCHR                   ;GET THE CHAR HE TYPED
1898  016326  012600                          MOV     (SP)+,R0        ;GET WHAT HE TYPED
1899  016330  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR> ??
1900  016334  001405                          BEQ     11$             ;BR IF IT WAS
1901  016336  110067  005116                  MOVB    R0,EC2          ;ECHO WHAT HE TYPED
1902  016342  104400                          TYPE
1903  016344  023460                          EC2
1904  016346  000763                          BR      1$              ;GO WAIT FOR TERMINATOR
1905  016350  105767  005104          11$:    TSTB    EC2             ;<CR> ONLY ??
1906  016354  001432                          BEQ     4$              ;BR IF YES
1907  016356  142767  000060  005074          BICB    #60,EC2         ;STRIP ASCII
1908  016364  122767  000006  005066          CMPB    #6,EC2          ;6 BITS ?
1909  016372  001004                          BNE     2$              ;BR IF NOT
1910  016374  052767  000001  001110          BIS     #1,CURLPR       ;SET UP FOR 6 BIT CHARS
1911  016402  000422                          BR      5$              ;CONTINUE
1912  016404  122767  000007  005046  2$:     CMPB    #7 EC2          ;7 BITS ?
1913  016412  001004                          BNE     3$              ;BR IF NOT
1914  016414  052767  000002  001070          BIS     #2,CURLPR       ;SET UP FOR 7 BIT CHARS
1915  016422  000412                          BR      5$              ;CONTINUE
1916  016424  122767  000010  005026  3$:     CMPB    #8.,EC2         ;8 BITS ?
1917  016432  001403                          BEQ     4$              ;BR IF YES
1918  016434  104400                          TYPE                    ;ERROR MESSAGE
1919  016436  024263                          CLMSG2                  ;"INVALID CHAR LENGTH = TRY AGAIN"
1920  016440  000722                          BR      INCL            ;GO TRY AGAIN
1921  016442  052767  000003  001042  4$:     BIS     #3,CURLPR       ;SET UP FOR 8 BIT CHARS
1922  016450  000207                  5$:     RTS     PC              ;RETURN TO CALLER
1923
1924                                          ;THIS ROUTINE IS CALLED TO SET UP NO. OF STOP BITS
1925
1926  016452  105067  005002          INSB:   CLRB    EC2             ;CLEAR ECHO BUFFER
1927  016456  104400                          TYPE                    ;ASK FOR INPUT
1928  016460  024327                          SBMSG1                  ;"NO. OF STOP BITS - 1 OR 2 ?"
1929  016462  104405                  1$:     RDCHR                   ;GET CHAR TYPED
1930  016464  012603                          MOV     (SP)+,R0        ;GET WHAT HE TYPED
1931  016466  122700  000015                  CMPB    #15,R0          ;WAS IT A <CR>
1932  016472  001405                          BEQ     11$             ;BR IF YES
1933  016474  110067  004760                  MOVB    R0,EC2          ;ECHO WHAT HE TYPED
1934  016500  104400                          TYPE
1935  016502  023460                          EC2
1936  016504  000766                          BR      1$              ;GO WAIT FOR TERMINATOR
1937  016506  105767  004746          11$:    TSTB    EC2             ;<CR> ONLY ??
1938  016512  001422                          BEQ     3$              ;BR IF YES
1939  016514  142767  000060  004736          BICB    #60,EC2         ;CLEAR ASCII JUNK
1940  016522  122767  000002  004730          CMPB    #2,EC2          ;2 STOP BITS ?
1941  016530  001004                          BNE     2$              ;BR IF NOT
1942  016532  052767  000004  000752          BIS     #4,CURLPR       ;SET UP FOR TWO STOP BITS
1943  016540  000412                          BR      4$              ;CONTINUE
1944  016542  122767  000001  004710  2$:     CMPB    #1,EC2          ;ONE STOP BIT ?
1945  016550  001403                          BEQ     3$              ;BR IF YES
1946  016552  104400                          TYPE                    ;ERROR MESSAGE
1947  016554  024366                          SBMSG2                  ;"INVALID NO. STOP BITS - TRY AGAIN"
1948  016556  000735                          BR      INSB            ;GO TRY AGAIN
```

```
1949  016560  042767  000004  000724  3$:     BIC     #4,CURLPR       ;SET UP FOR ONE STOP BIT
1950  016566  000207                  4$:     RTS     PC              ;RETURN TO CALLER
1951
1952                                  ;THIS ROUTINE IS CALLED TO SET UP PARITY SELECT BITS
1953
1954  016570  105067  004664          INPB:   CLRB    EC2             ;CLEAR ECHO BUFFER
1955  016574  104400                  TYPE                            ;ASK FOR INPUT
1956  016576  024434                  PBMSG1                          ;"PARITY - E,O, OR <CR> ?"
1957  016600  042767  000060  000704  1$:     BIC     #60,CURLPR      ;INIT FOR NO PARITY CHECKING
1958  016606  104405                  RDCHR                           ;GET CHAR TYPED
1959  016610  012600                  MOV     (SP)+,R0                ;GET WHAT HE TYPED
1960  016612  122700  000015          CMPB    #15,R0                  ;WAS IT A <CR> ??
1961  016616  001405                  BEQ     11$                     ;BR IF IT WAS
1962  016620  110067  004634          MOVB    R0,EC2                  ;ECHO THE CHAR TYPED
1963  016624  104400                  TYPE
1964  016626  023460                  EC2
1965  016630  000763                  BR      1$                      ;GO WAIT FOR TERMINATOR
1966  016632  105767  004622          11$:    TSTB    EC2             ;<CR> ONLY ??
1967  016636  001423                  BEQ     4$                      ;BR IF YES
1968  016640  122767  000105  004612  CMPB    #105,EC2                ;EVEN PARITY ??
1969  016646  001004                  BNE     2$                      ;BR IF NOT
1970  016650  052767  000060  000634  BIS     #60,CURLPR              ;SET UP FOR EVEN PARITY
1971  016656  000413                  BR      4$                      ;CONTINUE
1972  016660  122767  000117  004572  2$:     CMPB    #117,EC2        ;ODD PARITY
1973  016666  001004                  BNE     3$                      ;BR IF NOT
1974  016670  052767  000020  000614  BIS     #20,CURLPR              ;SET UP FOR ODD PARITY
1975  016676  000403                  BR      4$                      ;CONTINUE
1976  016700  104400                  3$:     TYPE                    ;ERROR MESSAGE
1977  016702  024502                  PBMSG2                          ;"INVALID PARITY - TRY AGAIN"
1978  016704  000731                  BR      INPB                    ;GO TRY AGAIN
1979  016706  000207                  4$:     RTS     PC              ;RETURN TO CALLER
1980
1981                                  ;THIS ROUTINE IS CALLED TO SET UP "FILL" CHAR
1982
1983  016710  105067  004544          INFCHR: CLRB    EC2             ;CLEAR ECHO BUFFER
1984  016714  005067  001242          CLR     FILLA                   ;INIT TEMP STORAGE FOR CHAR
1985  016720  104400                  TYPE                            ;GO ASK FOR FILLER CHAR
1986  016722  024541                  FILC1                           ;"FILL CHAR ?"
1987  016724  005067  001230          1$:     CLR     DHFILL          ;ININ FILL LOCATION
1988  016730  104405                  RDCHR                           ;GET CHAR TYPED
1989  016732  012600                  MOV     (SP)+,R0                ;GET WHAT HE TYPED
1990  016734  122700  000015          CMPB    #15,R0                  ;WAS IT A <CR> ??
1991  016740  001405                  BEQ     2$                      ;BR IF YES
1992  016742  110067  004512          MOVB    R0,EC2                  ;ECHO WHAT HE TYPED
1993  016746  104400                  TYPE
1994  016750  023460                  EC2
1995  016752  000764                  BR      1$                      ;GO WAIT FOR TERMINATOR
1996
1997  016754  105767  004500          2$:     TSTB    EC2             ;<CR> ONLY ??
1998  016760  001403                  BEQ     3$                      ;BR IF YES
1999  016762  116767  004472  001171  MOVB    EC2,DHFILL+1            ;SET UP FILL CHAR
2000  016770  116767  001165  001164  3$:     MOVB    DHFILL+1,FILLA  ;SAVE FILL CHAR
2001  016776  000207                  RTS     PC                      ;RETURN TO CALLER
2002
```

```
                                        ;THIS ROUTINE IS CALLED TO SET UP "FILL" COUNT
                                        
017000  005067  001160          INFCNT: CLR     FILLB           ;INIT TEMP, STORAGE FOR COUNT
017004  104400                          TYPE                    ;ASK FOR COUNT
017006  024567                          FILC2                   ;"FILL COUNT ?"
017010  104407                          RDOCT                   ;GET OCTAL NO. TYPED
017012  005716                          TST     (SP)            ;DEFAULT TO ONE ?
017014  001403                          BEQ     1$              ;BR IF YES
017016  111667  001136                  MOVB    (SP),DHFILL     ;SET UP COUNT TYPED
017022  000403                          BR      2$              ;CONTINUE
017024  112767  000001  001126  1$:     MOVB    #1,DHFILL       ;SET UP FOR  1 FILLER
017032  005726                  2$:     TST     (SP)+           ;FIX THE SP
017034  142767  000360  001116          BICB    #360,DHFILL     ;LIMIT COUNT TO 15. MAX
017042  116767  001112  001114          MOVB    DHFILL,FILLB    ;SAVE IT FOR LATER
017050  000207                          RTS     PC              ;RETURN TO CALLER
                                        ;THIS ROUTINE CALLED TO SET UP ALTERNATING 1/0 PATTERN
                                        
017052  004767  000246          SUPATA: JSR     PC,CLALL        ;GO CLEAR XMIT AND RCV BUFFERS
017056  016700  000434                  MOV     CHRCNT,R0               ;GET CHAR COUNT
017062  012705  030212                  MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
017066  112725  000252          1$:     MOVB    #252,(R5)+      ;LOAD A BYTE
017072  005200                          INC     R0              ;COUNT IT
017074  001374                          BNE     1$              ;BR TILL BUFFER FULL
017076  000207                          RTS     PC              ;RETURN TO "DPATA" ROUTINE
                                        
                                        ;THIS ROUTINE IS CALLED TO SET UP UP COUNT PATTERN
                                        
017100  004767  000220          SUPATU: JSR     PC,CLALL        ;GO CLEAR BUFFERS
017104  016700  000406                  MOV     CHRCNT,R0       ;GET COUNT OF CHARS TO LOAD
017110  012705  030212                  MOV     #TBUF,R5        ;POINT TO XMITTR BUFFER
017114  005004                          CLR     R4              ;INIT CHAR GENERATOR
017116  110425          1$:     MOVB    R4,(R5)+        ;LOAD ONE BYTE
017120  105204                          INCB    R4              ;GENERATE NEXT BYTE
017122  005200                          INC     R0              ;COUNT IT
017124  001374                          BNE     1$              ;BR TIL BUFFER FULL
017126  000207                          RTS     PC              ;RETURN TO "DPATU" ROUTINE
                                        
                                        ;THIS ROUTINE IS CALLED TO SET UP DOWN COUNT PATTERN
                                        
017130  004767  000170          SUPATD: JSR     PC,CLALL        ;CLEAR THE BUFFERS
017134  016700  000356                  MOV     CHRCNT,R0       ;SET UP COUNT TO LOAD
017140  012705  030212                  MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
017144  012704  000377                  MOV     #377,R4         ;INIT CHAR GENERATOR
017150  110425          1$:     MOVB    R4,(R5)+        ;LOAD ONE BYTE
017152  105304                          DECB    R4              ;GENERATE NEW CHAR
017154  005200                          INC     R0              ;COUNT IT
017156  001374                          BNE     1$              ;BR TIL BUFFER FULL
017160  000207                          RTS     PC              ;RETURN TO "DPATA" ROUTINE
                                        
                                        ;THIS ROUTINE CALLED TO LOAD RANDOM DATA PATTERN
                                        
017162  004767  000136          SUPATR: JSR     PC,CLALL        ;GO CLEAR BUFFERS
017166  016700  000324                  MOV     CHRCNT,R0       ;SET UP COUNT TO LOAD
017172  012705  030212                  MOV     #TBUF,R5        ;POINT TO XMITTR BUFFER
```

```
2057  017176  012767  125252  000704          MOV     #125252,RANA    ;INIT RANDOM NUMBER GENERATOR
2058
2059  017204  066767  000700  000700  1$:     ADD     RANA,RANB       ;GENERATE RANDOM NO.
2060  017212  005567  000672                  ADC     RANA
2061  017216  066767  000670  000664          ADD     RANB,RANA
2062  017224  005567  000662                  ADC     RANB
2063
2064  017230  116725  000654                  MOVB    RANA,(R5)+      ;LOAD A BYTE
2065  017234  005200                          INC     R0              ;COUNT IT
2066  017236  001362                          BNE     1$              ;BR TIL BUFFER FULL
2067  017240  000207                          RTS     PC              ;RETURN TO "DPATR" ROUTINE
2068
2069                          ;THIS ROUTINE LOADS A SINGLE CHAR THROUGHOUT BUFFER
2070
2071  017242  004767  000056  SUPATS: JSR     PC,CLALL        ;GO CLEAR BUFFERS
2072  017246  016700  000244          MOV     CHRCNT,R0       ;INIT CHAR COUNTER
2073  017252  012705  030212          MOV     #TBUF,R5        ;POINT TO XMIT BUFFER
2074  017256  116725  000620  1$:     MOVB    SINGLE,(R5)+    ;LOAD ONE CHAR
2075  017262  005200                  INC     R0              ;COUNT IT
2076  017264  001374                  BNE     1$              ;BR TIL BUFFER FULL
2077  017266  000207                  RTS     PC              ;RETURN TO "DPATS" ROUTINE
2078
2079                          ;THIS ROUTINE CALLED TO INIT CHAR LENGTH MASK FOR PATTERNS TESTS
2080
2081  017270  016700  000216  SUCLMK: MOV     CURLPR,R0       ;GET CURRENT "LPR"
2082  017274  012767  000340  000612          MOV     #340,CLMSK      ;INIT FOR 5 BIT CHARS
2083  017302  042700  177774                  BIC     #177774,R0      ;MASK OFF ALL BUT CL BITS
2084  017306  005700  1$:     TST     R0              ;DONE SETUP ?
2085  017310  001404                  BEQ     2$              ;BR IF YES
2086  017312  106367  000576                  ASLB    CLMSK           ;SHIFT MASK LEFT
2087  017316  005300                  DEC     R0              ;COUNT IT
2088  017320  000772                  BR      1$              ;GO SEE IF ITS RIGHT ON
2089  017322  000207  2$:     RTS     PC              ;RETURN TO CALLER
2090                          ;ROUTINE TO CLEAR XMIT AND RECEIVER BUFFERS
2091
2092  017324  012700  030212  CLALL:  MOV     #TBUF,R0        ;SET UP POINTER
2093  017330  005020  1$:     CLR     (R0)+           ;CLEAR A WORD
2094  017332  022700  031342          CMP     #ENBUFS,R0      ;DONE ALL LOCATIONS ?
2095  017336  001374                  BNE     1$              ;BR IF NOT
2096  017340  000207                  RTS     PC
```

```
                                        ;THIS ROUTINE IS CALLED TO SET PSW PRIORITY TO 000 IN ORDER
                                        ;TO BE LSI11 COMPATIBLE
                                        
017342  012746  000000          CHPS1:  MOV     #0,-(SP)        ;NEW PSW
017346  012746  017354                  MOV     #1$,-(SP)       ;NEW PC
017352  000002                          RTI                     ;CHANGE PSW
017354  000207                  1$:     RTS     PC              ;RETURN TO CALLING TEST

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

017356  012746  000340          CHPS2:  MOV     #340,-(SP)      ;NEW PSW
017362  012746  017370                  MOV     #1$,-(SP)       ;NEW PC
017366  000002                          RTI                     ;CHANGE THE PSW
017370  000207                  1$:     RTS     PC              ;RETURN TO CALLING TEST

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

017372  005046                  SAPS:   CLR     -(SP)           ;TEMP STORAGE TO SAVE PSW
017374  016746  160434                  MOV     34,-(SP)        ;SAVE TRAP VECTOR POINTER
017400  012767  017410  160426          MOV     #1$,34          ;GO TO 1$ ON TRAP
017406  104400                          TRAP                    ;GO TO IT
017410  016666  000002  000006  1$:     MOV     2(SP),6(SP)     ;GET PSW SAVED
017416  012716  017424                  MOV     #2$,(SP)        ;GO TO 2$ ON RTI
017422  000002                          RTI
017424  012667  160404          2$:     MOV     (SP)+,34        ;RESTORE VECTOR
017430  012667  161544                  MOV     (SP)+,$TMPO     ;FINALLY SAVE PSW IN $TMPO
017434  000207                          RTS     PC
```

C13


```
                                 .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.

017436  000000                   DHADR:  0                ;HOLDS THE "SCR" ADDRESS OF THE DH11 UNDER TEST
017440  000000                   DHVCT:  0                ;HOLDS THE 1ST VECTOR ADDRESS OF THE DH11 UNDER TEST
017442  000000                   SELMSK: 0                ;BIT TST MARKER FOR SELECTING DH11'S
017444  000001                   DHSEL:  1                ;SPECIFIES DH11'S SELECTED FOR TEST
017446  177777                   LINSEL: 177777           ;SPECIFIES LINES TO TEST
017450  000000                   LINMSK: 0                ;MARKER USED TO TEST FOR LINES TO TEST
017452  000000                   DRPLIN: 0                ;DROPPED LINE FLAGS

017454  000000                   QUICK:  0                ;QUICK TEST FLAG - ALLOWS SINGLE PATTERN TEST
                                                          ;ON ALL TESTS NOT USING 9600. BAUD
017456  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

017460  033500                   LPRTAB: 33500                    ;9600 BAUD
017462  004200                           4200                     ;75 BAUD
017464  006300                           6300                     ;110 BAUD
017466  010400                           10400                    ;134.5 BAUD
017470  012500                           12500                    ;150 BAUD
017472  014600                           14600                    ;200 BAUD
017474  016700                           16700                    ;300 BAUD
017476  021000                           21000                    ;600 BAUD
017500  023100                           23100                    ;1200 BAUD
017502  025200                           25200                    ;1800 BAUD
017504  027300                           27300                    ;2400 BAUD
017506  031400                           31400                    ;4800 BAUD
017510  002100                           2100                     ;50 BAUD

017512  000000                   CURLPR: C                        ;CONTAINS CURRENT "LPR" CONSTANT

017514  000000                   LPRPTR: 0                        ;CONTAINS POINTER TO LPR TABLE
017516  000000                   CHRCNT: 0                        ;LOADED WITH CURRENT CHAR COUNT

017520  000000                   CLSEL:  0                        ;CHAR LENGTH SELECT PARAMETER
017522  000000                   PARBIT: 0                        ;PARITY SELECT PARAMETER
```

```
017524  000000  ROONE:  0                       ;SOFTWARE DONE FLAG
017526  000000  RBFEND: 0                       ;HOLDS END OF BUFFER ADDRESS

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

017530  160020  DHADTB: 160020          ;ADDRESS OF FIRST DH11
017532  160040           160040          ;ADDRESS OF SECOND DH11
017534  160060           160060
017536  160100           160100
017540  160120           160120
017542  160140           160140
017544  160160           160160
017546  160200           160200
017550  160220           160220
017552  160240           160240
017554  160260           160260
017556  160300           160300
017560  160320           160320
017562  160340           160340
017564  160360           160360
017566  160400           160400          ;ADDRESS OF THE LAST DH11

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

017570  000330  DHVCTB: 330             ;ADDRESS OF VECTOR FOR FIRST DH11
017572  000350           350             ;ADDRESS OF VECTOR FOR SECOND DH11
017574  000370           370
017576  000410           410
017600  000430           430
017602  000450           450
017604  000470           470
017606  000510           510
017610  000530           530
017612  000550           550
017614  000570           570
017616  000610           610
017620  000630           630
017622  000650           650
017624  000670           670
017626  000710           710             ;ADDRESS OF VECTOR FOR LAST DH11

017630  000000  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

017632  120240  VL:    120240          ;BRLEVELS FOR FIRST DH11
017634  120240           120240          ;BR LEVELS FOR SECOND DH11
017636  120240           120240
```

```
017640 120240                  120240
017642 120240                  120240
017644 120240                  120240
017646 120240                  120240
017650 120240                  120240
017652 120240                  120240
017654 120240                  120240
017656 120240                  120240
017660 120240                  120240
017662 120240                  120240
017664 120240                  120240
017666 120240                  120240
017670 120240                  120240         ;BR LEVELS FOR LAST DH11
017672    000        DHRLVL: .BYTE   0                ;BR LEVEL FOR RCVR
017673    000        DHTLVL: .BYTE   0                ;BR LEVEL FOR XMITTER
017674 000000        DHNUM:  0                ;CONTAINS NUMBER OF THE DH11 UNDER TEST
017676 000000        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"

017700 000000        XSPTR:  0                        ;CONTAINS POINTER TO FOLLOWING TABLE

017702 000062        XSPTAB: 50.                      ;50. BAUD
017704 002000                2000
017706 000113                75.                      ;75. BAUD
017710 004000                4000
017712 000156                110.                     ;110. BAUD
017714 006000                6000
017716 002501                1345.                    ;134.5 BAUD
017720 010000                10000
017722 000226                150.                     ;150. BAUD
017724 012000                12000
017726 000310                200.                     ;200. BAUD
017730 014000                14000
017732 000454                300.                     ;300 BAUD
017734 016000                16000
017736 001130                600.                     ;600. BAUD
017740 020000                20000
017742 002260                1200.                    ;1200. BAUD
017744 022000                22000
017746 003410                1800.                    ;1800. BAUD
017750 024000                24000
017752 004540                2400.                    ;2400. BAUD
017754 026000                26000
017756 011300                4800.                    ;4800. BAUD
017760 030000                30000
017762 022600                9600.                    ;9600. BAUD
017764 032000                32000
017766 000000        RSPTR:  0                        ;CONTAINS POINTER TO FOLLOWING TABLE
```

```
                                RSPTAB: 50.                     ;50. BAUD
                                        100
                                        75.                     ;75. BAUD
                                        200
                                        110.                    ;110. BAUD
                                        300
                                        1345.                   ;134.5 BAUD
                                        400
                                        150.                    ;150. BAUD
                                        500
                                        200.                    ;200. BAUD
                                        600
                                        300.                    ;300 BAUD
                                        700
                                        600.                    ;600. BAUD
                                        1000
                                        1200.                   ;1200. BAUD
                                        1100
                                        1800.                   ;1800. BAUD
                                        1200
                                        2400.                   ;2400. BAUD
                                        1300
                                        4800.                   ;4800. BAUD
                                        1400
                                        9600.                   ;9600. BAUD
                                        1500

                                ;ADDRESS POINTERS TO SET UP TABLES WHEN INPUTTING PARAMETERS

020054  000000                  ADPTR:  0                       ;POINTS TO ADDRESS TABLE
020056  000000                  VCPTR:  0                       ;POINTS TO VECTOR TABLE
020060  000000                  BRPTR:  0                       ;POINTS TO BR LEVEL TABLE

020062  000000                  TITFLG: 0                       ;FLAG TO ALLOW PRINTING TITLE ONLY ONCE
020064  000000                  TIMEA:  0                       ;GENERAL PURPOSE TIMERS
020066  000000                  TIMEB:  0

020070  000000                  CEXIT:  0                       ;CONTROL-C EXIT FLAG FM ECHO TESTS
020072  000000                  DPFLG:  0                       ;PATTERNS TEST FLAG
020074  000000                  DATCNT: 0                       ;ITERATION COUNTER FOR PATTERNS TEST
020076  000000                  DATPAT: 0                       ;FLAGS TYPE PATTERN
020100  000000                  PATFLG: 0                       ;DATA PATTERNS <CR> SEQUENCE FLAG
020102  000000                  SINGLE: 0                       ;HOLDS SINGLE CHAR TEST PATTERN
020104  000000                  RETFLG: 0                       ;ECHO TEST RETURN FLAG FM SETUP
020106  000012                  PATLIM: 10.                     ;PATTERNS TESTS ITERATION COUNT
020110  000000                  RANA:   0                       ;RANDOM NO. ACCUMULATORS
020112  000000                  RANB:   0
020114  000000                  CLMSK:  0                       ;CHAR LENGTH BIT CLR MASK
020116  000000                  EXFLAG: 0                       ;ECHO TEST EXIT FLAGS
020120  000020                  ECBUF:  .BLKW   16.             ;DATA BUFFER FOR SINGLE LINE ECHO TEST
020160  000000                  DHFILL: 0                       ;FILL CHAR AND COUNT FOR SINGLE LINE
                                                                ;ECHO TESTS
020162  000000                  FILLA:  0                       ;TEMP STORAGE FOR FILLER CHAR
020164  000000                  FILLB:  0                       ;SAME FOR COUNT
```

```
2346
2347
```

```
2349                                    .SBTTL STANDARD ERROR MESSAG BUFFERS
2350                                    ;*****************************************************************
2351                                    ;ERROR MESSAGE INFORMATION - MESSAGE BUFFERS AND POINTERS
2352                                    ;*****************************************************************
2353
2354                                    ;INFORMATION FOR MESSAGE 1
2355
2356  020166 047516 020116 054105  EM1:    .ASCIZ  'NON EX MEMORY ERROR - DROPPED LINE #  '
       020174 046440 046505 051117
       020202 020131 051105 047522
       020210 020122 020055 051104
       020216 050117 042520 020104
       020224 044514 042516 021440
       020232 020040    000
2357  020235    040 050050 024503  DH1:    .ASCIZ  ' (PC)   CURLPR  DEVADR  REGADR   WAS     S/B'
       020242 020040 041440 051125
       020250 050114 020122 042040
       020256 053105 042101 020122
       020264 051040 043505 042101
       020272 020122 020040 040527
       020300 020123 020040 020040
       020306 027523 000102
2358                                    .EVEN
2359  020312 001116 017512 000162  DT1:    .WORD   $ERRPC,CURLPR,$REG1,$REG2,$REG3,$REG4,0
       020320 001164 001166 001170
       020326 000000
2360  020330    000    000    000  DF2:    .BYTE   0,0,0,0,0,0,0,0
       020333    000    000    000
       020336    000    000
2361
2362                                    ;INFORMATION FOR MESSAGE 2
2363
2364  020340 051124 047101 046523  EM2:    .ASCIZ  'TRANSMITTER FALSE INTERRUPT - DROPPED LINE #  '
       020346 052111 042524 020122
       020354 040506 051514 020105
       020362 047111 042524 051122
       020370 050125 020124 020055
       020376 051104 050117 042520
       020404 020104 044514 042516
       020412 021440 020040    000
2365
2366
2367                                    ;INFORMATION FOR MESSAGE 3
2368
2369  020417    102 043125 042506  EM3:    .ASCIZ  'BUFFER ACTIVE REGISTER ERROR - DROPPED LINE #  '
       020424 020122 041501 044524
       020432 042526 051040 043505
       020440 051511 042524 020122
       020446 051105 047522 020122
       020454 020055 051104 050117
       020462 042520 020104 044514
       020470 042516 021440 020040
       020476    000
237C
```

```
                                        ;INFORMATION FOR MESSAGE 4
                                        ;
                                        ;
020477  000102  052131  020105  EM4:    .ASCIZ  'BYTE COUNT REGISTER ERROR - DROPPED LINE #  '
020504  047503  047125  020124
020512  042522  044507  052123
020520  051105  042440  051122
020526  051117  026440  042040
020534  047522  050120  042105
020542  046040  047111  020105
020550  020043  000040
                                        ;
                                        ;INFORMATION FOR MESSAGE 5
                                        ;
                                        ;
020554  052503  051122  047105  EM5:    .ASCIZ  'CURRENT ADDRESS REGISTER ERROR - DROPPED LINE #  '
020562  020124  042101  051104
020570  051505  020123  042522
020576  044507  052123  051105
020604  042440  051122  051117
020612  026440  042040  047522
020620  050120  042105  046040
020626  047111  020105  020043
020634  000040
                                        ;
                                        ;INFORMATION FOR MESSAGE 6
                                        ;
020636  044523  047514  047440  EM6:    .ASCIZ  'SILO OVERFLOW ERROR - DROPPED LINE #  '
020644  042526  043122  047514
020652  020127  051105  047522
020660  020122  020055  051104
020666  050117  042520  020104
020674  044514  042516  021440
020702  020040     000
                                        ;
                                        ;INFORMATION FOR MESSAGE 7
                                        ;
020705     122  041505  044505  EM7:    .ASCIZ  'RECEIVER FALSE INTERRUPT - DROPPED LINE #  '
020712  042526  020122  040506
020720  051514  020105  047111
020726  042524  051122  050124
020734  020124  020055  051104
020742  050117  042520  020104
020750  044514  042516  021440
020756  020040     000
                                        ;
                                        ;INFORMATION FOR MESSAGE 10
                                        ;
                                        ;
020761     111  053116  046101  EM10:   .ASCIZ  'INVALID DATA IN SILO - DROPPED LINE #  '
020766  042111  042040  052101
020774  020101  047111  051440
021002  046111  020117  020055
021010  051104  050117  042520
021016  020104  044514  042516
021024  021440  020040     000
021031     040  050050  024503  DH2:    .ASCIZ  ' (PC)   CURLPR  CHAR #  WASADR  SHBADR   WAS     S/B'
```

```
                021036  020040  041440  051125
                021044  050114  020122  041440
                021052  040510  020122  020043
                021060  053440  051501  042101
                021066  020122  051440  041110
                021074  042101  020122  020040
                021102  040527  020123  020040
                021110  020040  027523  000102
2391                                            .EVEN
2392            021116  001116  017512  001160  DT2:    .WORD   $ERRPC,CURLPR,$REG0,$REG1,$REG2,$REG3,$REG4,0
                021124  001162  001164  001166
                021132  001170  000000
2393
2394                                            ;INFORMATION FOR MESSAGE 11
2395
2396            021136  040504  040524  042440  EM11:   .ASCIZ  'DATA ERROR - LINE #  '
                021144  051122  051117  026440
                021152  046040  047111  020105
                021160  020043  000040
2397
2398                                            ;INFORMATION FOR MESSAGE 12
2399
2400            021164  042524  052123  052040  EM12:   .ASCIZ  'TEST TIMEOUT - DROPPED LINE #  '
                021172  046511  047505  052125
                021200  026440  042040  047522
                021206  050120  044105  046040
                021214  047111  020105  020043
                021222  000040
2401            021224  020040  041520  020051  DH3:    .ASCIZ  ' (PC)   CURLPR  RTOTAL  XTOTAL  RDONE'
                021232  020040  052503  046122
                021240  051120  020040  052122
                021246  052117  046101  020040
                021254  052130  052117  046101
                021262  020040  042122  047117
                021270  000105
2402                                            .EVEN
2403            021272  001116  017512  001200  DT3:    .WORD   $ERRPC,CURLPR,STMP0,STMP1,RDONE,0
                021300  001202  017524  000000
2404
2405                                            ;INFORMATION FOR MESSAGE 13
2406
2407            021306  001200  001202  001204  DT4:    .WORD   STMP0,STMP1,STMP2,STMP3,STMP4,STMP5,STMP6,0
                021314  001206  001210  001212
                021322  001214  000000
2408            021326     000     001     001  DF1:    .BYTE   0,1,1,1,1,1,1,0
                021331     001     001     001
                021334     001     000
2409
2410                                            ;INFORMATION FOR MESSAGE 14
2411
2412            021336  052502  020123  051105  EM14:   .ASCIZ  'BUS ERROR TRAP TO 04'
                021344  047522  020122  051124
                021352  050101  052040  020117
                021360  032060     000
```

```
2413  021363  000040  052040  024503  DH4:    .ASCIZ  ' (PC)    (PS)    (SP)   TRAPPC  TRAPPS'
      021370  020050  020050  050050
      021376  024523  020040  030040
      021404  051450  024520  020042
      021412  052040  040522  050120
      021420  020103  052040  040522
      021426  050120  000123
2414                                  .EVEN
2415  021432  001116  001200  001174  DT5:    .WORD   $ERRPC,$TMPO,$REG6,$REG1,$REG2,0
      021440  001162  001164  000000
2416
2417                                  ;INFORMATION FOR MESSAGE 15
2418
2419  021446  051522  042126  044440  EM15:   .ASCIZ  'RSVD INSTR TRAP TO 10'
      021454  051516  051124  052040
      021462  040522  020120  047524
      021470  030440  000060
2420
2421                                  ;INFORMATION FOR MESSAGE 16
2422
2423  021474  044523  043516  042514  EM16:   .ASCIZ  'SINGLE LINE ECHO TEST - INTR WAIT TIMEOUT'
      021502  046040  047111  020105
      021510  041505  047510  052040
      021516  051505  020124  020055
      021524  047111  051124  053440
      021532  044501  020124  044524
      021540  042515  052517  000124
2424  021546  024040  041520  020051  DH5:    .ASCIZ  ' (PC)   DEVADR   LINE    (SCR)  CURLPR  EXFLAG'
      021554  020040  042504  040526
      021562  051104  020040  046040
      021570  047111  020105  020040
      021576  024040  041523  024522
      021604  020040  052503  046122
      021612  051120  020040  054105
      021620  046106  043501     000
2425          021626                  .EVEN
2426  021626  001116  001162  017676  DT6:    .WORD   $ERRPC,$REG1,LINE,$TMPO,CURLPR,EXFLAG,0
      021634  001200  017512  020116
      021642  000000
2427
2428
2429                                  ;INFORMATION FOR MESSAGE 17
2430
2431  021644  046101  042524  047122  EM17:   .ASCIZ  'ALTERNATING 1/0 PATTERN TEST DONE'
      021652  052101  047111  020107
      021660  027461  020060  040520
      021666  052124  051105  020116
      021674  042524  052123  042040
      021702  047117  000105
2432  021706  024040  041520  020051  DH6:    .ASCIZ  ' (PC)   DEVADR   LINE   CURLPR  ICOUNT'
      021714  020040  042504  040526
      021722  051104  020040  046040
      021730  047111  020105  020040
      021736  052503  046122  051120
```

```
        021744  020040  041511  052517
        021752  052116     000
2433            021756                  .EVEN
2434    021756  001116  017436  017676  DT7:    .WORD   $ERRPC,DHADR,LINE,CURLPR,$REGO,0
        021764  017512  001160  000000
2435
2436                                    ;INFORMATION FOR MESSAGE 20
2437
2438    021772  044502  040516  054522  EM20:   .ASCIZ  'BINARY UP COUNT PATTERN TEST DONE'
        022000  052440  020120  047503
        022006  047125  020124  040520
        022014  052124  051105  020116
        022022  042524  052123  042040
        022030  047117  000105
2439
2440                                    ;INFORMATION FOR MESSAGE 21
2441
2442    022034  044502  040516  054522  EM21:   .ASCIZ  'BINARY DOWN COUNT PATTERN TEST DONE'
        022042  042040  053517  020116
        022050  047503  047125  020124
        022056  040520  052124  051105
        022064  020116  042524  052123
        022072  042040  047117  000105
2443
2444                                    ;INFORMATION FOR MESSAGE 22
2445
2446    022100  040522  042116  046517  EM22:   .ASCIZ  'RANDOM DATA PATTERN TEST DONE'
        022106  042040  052101  020101
        022114  040520  052124  051105
        022122  020116  042524  052123
        022130  042040  047117  000105
2447
2448                                    ;INFORMATION FOR MESSAGE 23
2449
2450    022136  044523  043516  042514  EM23:   .ASCIZ  'SINGLE CHAR PATTERN TEST DONE'
        022144  041440  040510  020122
        022152  040520  052124  051105
        022160  020116  042524  052123
        022166  042040  047117  000105
2451
2452                                    ;INFORMATION    FOR MESSAGE 24
2453
2454    022174  054524  042520  020104  EM24:   .ASCIZ  'TYPED BUFFER PATTERN TEST DONE'
        022202  052502  043106  051105
        022210  050040  052101  042524
        022216  047122  052040  051505
        022224  020124  047504  042516
        022232     000
2455
2456
2457                                    ;INFORMATION FOR MESSAGE 25
2458
2459    022233     104  052101  020101  EM25:   .ASCIZ  'DATA PATTERNS TEST TIMEOUT'
        022240  040520  052124  051105
```

```
        022246  051516  052040  051505
        022254  020124  044524  042515
        022262  052517  000124
 2460   022266  024040  041520  020051  DI-7:    .ASCIZ  ' (PC)   DEVADR   LINE   CURLPR  ICOUNT  PATCDE'
        022274  020040  042504  040526
        022302  051104  020040  040340
        022310  047111  020105  020040
        022316  052503  046122  051120
        022324  020040  041511  052517
        022332  052116  020040  040520
        022340  041524  042504     000
 2461           022346                  .EVEN
 2462   022344  001116  017436  017676  DTIO:    .WORD   SERRPC,DHADR,LINE,CURLPR,$REGO,$REG1,0
        022354    7512  001160  001162
        022362  000000
 2463
 2464                                   .EVEN
```

```
2467                                    .SBTTL MISCELANEOUS TABLES AND  MESSAGE AND DATA BUFFERS
2468                                    ;*****************************************************************
2469                                    ;MISCELLANEOUS MESSAGES
2470                                    ;*****************************************************************
2471
2472  022364  005015  040515  047111  TITLE:  .ASCIZ  <15><12>'MAINDEC-11-DZDHN-A DH11 DATA RELIABILITY TEST'<15><12>
      022372  042504  026503  030461
      022400  042055  042132  047110
      022406  040455  042040  030510
      022414  020061  040504  040524
      022422  051040  046105  040511
      022430  044502  044514  054524
      022436  052040  051505  006524
      022444  000012
2473  022446  005015  042524  052123  TITLE2: .ASCIZ  <15><12>'TESTING DH11 #  '<15><12>
      022454  047111  020107  044104
      022462  030461  021440  020040
      022470  005015     000
2474  022473     015  052012  050131  INMSG1: .ASCIZ  <15><12>'TYPE SCR ADDRESS FOR FIRST DH11'<15><12>
      022500  020105  041523  020122
      022506  042101  051104  051505
      022514  020123  047506  020122
      022522  044506  051522  020124
      022530  044104  030461  005015
      022536     000
2475  022537     015  052012  050131  INMSG2: .ASCIZ  <15><12>'TYPE VECTOR ADDRESS FOR FIRST DH11'<15><12>
      022545  020105  042524  052103
      022553  051117  040440  042104
      022561  042522  051523  043040
      022567  051117  040440  051111
      022575  052123  042040  030510
      023003  006461  000012
2476  023011  005015  054524  042520  INMSG3: .ASCIZ  <15><12>'TYPE DH11 DEVICE SELECTION PARAMETER'<15><12>
      023017  042040  030510  020061
      023025  042504  044524  042503
      023033  051440  046105  041505
      023041  044524  047117  050040
      023047  051101  046501  052103
      023055  051105  005015     000
2477  023063     015  044412  053116  INMSG4: .ASCIZ  <15><12>'INVALID DH11 SCR ADDRESS - TRY AGAIN'<15><12>
      023071  046101  042111  042040
      023077  030510  020061  041523
      023105  020122  042101  051104
      023113  051505  020123  020055
      023121  051124  020131  047524
      023127  044501  006516  000012
2478  023135  005015  047111  040526  INMSG5: .ASCIZ  <15><12>'INVALID DH11 VECTOR ADDRESS - TRY AGAIN'<15><12>
      023143  044514  020104  044104
      023151  030461  053040  041505
      023157  047524  020122  042101
      023165  051104  051505  020123
      023173  020055  051124  020131
      023201  043501  044501  006516
      023207  000012
```

```
 2479  023004  005015  047531  020125  INMSG6: .ASCIZ  <15><12>'YOU MUST SELECT AT LEAST ONE DH11'<15><12>
       023012  052515  052123  051440
       023020  046105  041505  020124
       023026  052101  046040  040505
       023034  052123  047440  042516
       023042  071040  030510  006461
       023050  000012

 2480  023052  005015  042504  051120  INMSG7: .ASCIZ  <15><12>'DEPRESS "CONTINUE" TO START TESTING'<15><12>
       023060  051505  020123  041442
       023066  047117  044524  052516
       023074  021105  052040  020117
       023102  052123  051101  020124
       023110  042524  052123  047111
       023116  000007  000012

 2481  023122  005015  054524  042520  INMSGA: .ASCIZ  <15><12>'TYPE LINE SELECTION PARAMETER'<15><12>
       023130  046040  047111  020105
       023136  042523  042514  052103
       023144  047511  020116  040520
       023152  040522  042515  042524
       023160  006522  000012

 2482
 2483  023164  005015  054524  042520  VCWC:   .ASCIZ  <15><12>'TYPE NO. OF WORDS (OCTAL) BETWEEN VECTORS (4 OR 10)'<15><12>
       023172  047040  027117  047440
       023200  020105  047527  042122
       023206  020123  047450  052103
       023214  046101  020051  042502
       023222  053524  042505  020116
       023230  042524  052103  051117
       023236  020123  032120  047440
       023244  020122  030061  006451
       023252  000012

 2484  023254  005015  041104  021440  STMSG1: .ASCIZ  <15><12>'DH #   STATISTICS:'
       023262  020040  051440  040524
       023270  044524  051123  041511
       023276  035123  000000

 2485  023301  000015  042012  051505  STMSG2: .ASCIZ  <15><12>'TESTING LINE #  '<15><12>
       023306  044524  043516  046040
       023314  047111  020105  020043
       023322  006440  000012

 2486  023326  005015  044514  042516  STMSG3: .ASCIZ  <15><12>'LINE #   WAS DROPPED'<15><12>
       023334  021440  020040  053440
       023342  051501  042040  047522
       023350  050120  042105  005015
       023356  000000

 2487  023357  000015  046012  047111  STMSG4: .ASCIZ  <15><12>'LINE #  RTOTAL  XTOTAL  DATERR  PARERR  FRMERR  OVRERR'<15><12>
       023364  020105  020043  051040
       023372  047524  040524  020114
       023400  054040  047524  040524
       023406  020114  042040  052101
       023414  051105  020122  050040
       023422  051101  051105  020122
       023430  043040  046522  051105
       023436  020122  047440  051126
       023444  051105  006522  000012
```

```
                                        ;MESSAGES FOR INPUTTING PARAMETERS TO ECHO TESTS
                023462  020040  006440  000012  EC:     .ASCIZ  '  '<15><12>
                023466  000040                  EC2:    .ASCIZ  ' '
                023470  020040  005015  000     EC3:    .ASCIZ  '  '<15><12>
2493            023474  015     051412  047111  ECMSG1: .ASCIZ  <15><12>'SINGLE LINE ECHO TEST - CONNECT TERMINAL TO DH11 TEST LINE'<15>
                023500  046107  020105  044514
                023504  042516  021440  044103
                023510  020117  042524  062123
                023514  020440  041440  047117
                023520  042516  052103  052040
                023524  051105  044516  044516
                023530  020114  047524  042040
                023534  030510  020061  042524
                023540  052123  046040  047111
                023544  006505  000012
2494
2495
2496            023546  005015  054524  042520  ECMSG2: .ASCIZ  <15><12>'TYPE LINE # (00 - 17 OCTAL)'
                023554  046040  047111  020105
                023560  020043  030050  020060
                023564  020055  033461  030040
                023570  052103  046101  000051
2497
2498            023574  005015  042524  052123  ECMSG3: .ASCIZ  <15><12>'TESTING LINE #   - GO TYPE IN ON TEST LINE'<15><12>
                023602  047111  020107  044514
                023606  042516  021440  020040
                023612  020440  043440  020117
                023616  054524  042520  044440
                023622  020116  047117  052040
                023626  051505  020124  044514
                023632  042516  005015  000
2499
2500            023636  015     052012  050131  ECMSG4: .ASCIZ  <15><12>'TYPE: [CONTROL-C TO EXIT] [CONTROL-E TO ECHO BUFFER]'<15><12>
                023642  035105  055440  047503
                023646  052116  047522  026514
                023652  020103  047524  042440
                023656  044523  055124  055440
                023662  047503  052116  047522
                023666  026514  020105  047524
                023672  044523  044103  020117
                023676  020440  043105  061105
                023702  006536  000012
2501            023706  005015  047504  054440  LPMSG:  .ASCIZ  <15><12>'DO YOU WANT TO CHANGE "LPR" (Y OR N) ? '
                023712  052117  053440  047101
                023716  020124  047524  041440
                023722  040510  043516  020105
                023726  046040  051120  020040
                023732  054450  047440  020122
                023736  042516  037440  000040
2502            023742  005015  051124  047101  XSMSG1: .ASCIZ  <15><12>'TRANSMITTER SPEED ?'
                023746  046523  052111  042524
                023752  020122  050123  042505
                023756  020104  000077
```

```
2503
2504  024074  005015  047111  040526  XSMSG2: .ASCIZ  <15><12>'INVALID XMIT SPEED - TRY AGAIN'<15><12>
      024102  044514  020104  046520
      024110  052111  051440  042520
      024116  042105  026440  052040
      024124  054522  040440  040507
      024132  047111  005015     000
2505
2506  024137     015  051012  041505  RSMSG1: .ASCIZ  <15><12>'RECEIVER SPEED ?'
      024144  044505  042526  020122
      024152  050123  042505  020104
      024160  000077
2507
2508  024162  005015  047111  040526  RSMSG2: .ASCIZ  <15><12>'INVALID RCVR SPEED - TRY AGAIN'<15><12>
      024170  044514  020104  041522
      024176  051126  051440  042520
      024204  042105  026440  052040
      024212  054522  040440  040507
      024220  047111  005015     000
2509
2510  024225     015  041412  040510  CLMSG1: .ASCIZ  <15><12>'CHAR LENGTH (6, 7, OR 8) ? '
      024232  020122  042514  043516
      024240  044124  024040  026C66
      024246  033440  020054  051117
      024254  034040  020051  020077
      024262     000
2511
2512  024263     015  044412  053116  CLMSG2: .ASCIZ  <15><12>'INVALID CHAR LENGTH - TRY AGAIN'<15><12>
      024270  046101  042111  041440
      024276  040510  020122  042514
      024304  043516  044124  026440
      024312  052040  054522  040440
      024320  040507  047111  005015
      024326     000
2513
2514  024327     015  047012  027117  SBMSG1: .ASCIZ  <15><12>'NO. OF STOP BITS (1 OR 2) ? '
      024334  047440  020106  052123
      024342  050117  041040  052111
      024350  020123  030450  047440
      024356  020122  024462  037440
      024364  000040
2515
2516  024366  005015  047111  040526  SBMSG2: .ASCIZ  <15><12>'INVALID NO. STOP BITS - TRY AGAIN'<15><12>
      024374  044514  020104  047516
      024402  020056  052123  050117
      024410  041040  052111  020123
      024416  020055  051124  020131
      024424  043501  044501  006516
      024432  000012
2517
2518  024434  005015  040520  044522  PBMSG1: .ASCIZ  <15><12>'PARITY SELECTION (E, O, OR <CR>) ? '
      024442  054524  051440  046105
      024450  041505  044524  047117
      024456  024040  026105  047440
```

```
                024464  020054  051117  036040
                024472  051103  024476  037440
                024500  000040

2519
2520    024502  005015  047111  040526  PBMSG2: .ASCIZ  <15><12>'INVALID PARITY - TRY AGAIN'<15><12>
                024510  044514  020104  040520
                024516  044522  054524  026440
                024524  052040  054522  040440
                024532  040507  047111  005015
                024540      000

2521
2522    024541      015  043012  046111  FILC1:  .ASCIZ  <15><12>'FILLER CHARACTER ? '
                024546  042514  020122  044103
                024554  051101  041501  042524
                024562  020122  020077      000

2523
2524    024567      015  043012  046111  FILC2:  .ASCIZ  <15><12>'FILLER COUNT ?'
                024574  042514  020122  047503
                024602  047124  020124  000077
2525    024610  005015  042522  042116  SNMSG1: .ASCIZ  <15><12>'SEND MODE -(Y OR N)  '
                024616  046440  043117  020105
                024624  024505  020131  051117
                024632  047040  020051  000040
2526    024640  005015  054524  042520  SNMSG2: .ASCIZ  <15><12>'TYPE SEND BUFFER - TERMINATE WITH CONTROL-C'<15><12>
                024646  051440  047105  020104
                024654  052502  043106  051105
                024662  026440  052040  051105
                024670  044515  040516  042524
                024676  053440  052111  020110
                024704  047503  052116  047522
                024712  026514  006503  000012
2527    024720  005015  044103  047101  SNMSG3: .ASCIZ  <15><12>'CHANGE PARAMETERS (Y OR N)? '
                024726  042507  050040  051101
                024734  046501  052105  051105
                024742  020123  054450  047440
                024750  020122  024516  020077
                024756      000
2528
```

```
2531                                        ;MESSAGES FO INPUTTING PATTERNS TEST PARAMETERS
2532  024757 000015 042012 052101  DPMSG1: .ASCIZ  <15><12>'DATA PATTERNS TESTS - CONNECT TEST JUMPER'<15><12>
      024764 020101 040520 052124
      024772 051105 051516 052040
      025000 051505 051524 026440
      025006 041440 047117 042516
      025014 052103 052040 051505
      025022 020124 052012 050115
      025030 051105 050015 000012
2533  025036 000015 041012 043125  DPMSG2: .ASCIZ  <15><12>'BUFFER SIZE ? (1 - 512)'
      025044 042506 020122 044523
      025052 047440 027440 026440
      025060 020061 020055 030465
      025066 044440 000051
2534  025074 000015 044412 053116  DPMSG3: .ASCIZ  <15><12>'INVALID SIZE - TRY AGAIN'<15><12>
      025102 046101 042111 051440
      025110 051111 020105 020055
      025116 051124 020131 043501
      025124 044501 006516 000012
2535  025132 006015 040520 052124  DPMSG4: .ASCIZ  <15><12>'PATTERN TYPE ? (A,U,D,R,S,B OR <CR>) '<15><12>
      025140 051105 020116 054524
      025146 042520 027440 026440
      025154 026101 026122 026104
      025162 026122 026123 020102
      025170 051117 026040 051103
      025176 044476 006440 000012
2536  026004 006015 042523 020124  DPMSG5: .ASCIZ  <15><12>'SET SR07=1 TO LOCK ON PATTERN'<15><12>
      026012 051123 033460 030475
      026020 052040 020117 047514
      026026 045503 047440 020116
      026034 040520 052124 051105
      026042 006516 000012
2537  026050 006015 047111 040526  DPMSG6: .ASCIZ  <15><12>'INVALID PATTERN - TRY AGAIN'<15><12>
      026056 044514 020104 040520
      026064 052124 051105 020116
      026072 020055 051124 020131
      026100 043501 044501 006516
      026106 000012
2538  026114 006015 054524 042520  DPMSG7: .ASCIZ  <15><12>'TYPE SINGLE TEST CHAR '<15><12>
      026122 051440 047111 046107
      026130 020105 042524 052123
      026136 041440 040510 020122
      026144 006015 000
2539  026151 000015 052012 050131  DPMSGA: .ASCIZ  <15><12>'TYPE IN TEST BUFFER - TERMINATE WITH CONTROL-C'<15><12>
      026157 020105 047111 052040
      026165 051505 020124 052502
      026173 043106 051105 026440
      026201 052040 051105 044515
      026207 040516 042524 053440
      026215 052111 020110 047503
      026223 052116 047522 026514
      026231 006503 000012
2540                                        .EVEN
2541                                        ;ERROR STATISTICS TABLES
```

```
025416  000020                  RTOTAL: .BLKW   16.             ;TOTAL CHARS RCVD PER LINE
025436  000020                  XTOTAL: .BLKW   16.             ;TOTAL CHARS XMITTED PER LINE
025516  000020                  DATERR: .BLKW   16.             ;TOTAL DATA COMPARE ERRORS PER LINE
025556  000020                  PAPERR: .BLKW   16.             ;TOTAL PARITY ERRORS PER LINE
025616  000020                  OVRERR: .BLKW   16.             ;TOTAL OVERRUN ERRORS PER LINE
025656  000020                  FRMERR: .BLKW   16.             ;TOTAL FRAMING ERRORS PER LINE

                                ;STATISTICS TABLES POINTERS

025716  000000                  DEPTR:  0                       ;CONTAINS POINTERS TO STAT TABLES
025720  000000                  PEPTR:  0
025722  000000                  ORPTR:  0
025724  000000                  FRPTR:  0

025726  000000                  RBFPTR: 0                       ;CONTAINS INPUT BUFFER POINTER
025730  000000                  TBFPTR: 0                       ;CONTAINS OUTPUT BUFFER POINTER


                                ;600. WORD RECEIVER INPUT BUFFER

025732  001130                  RBUF:   .BLKW   600.


                                ;600(10) BYTE TRANSMITTER OUTPUT DATA BUFFER

                                .EVEN
030212  001130                  TBUF:   .BLKB   600.

031342  000000                  ENBUFS: 0                       ;MARK END OF BUFFERS

        000001                  .END
```

```
ABASE = 000000
ACOM1 = 000000
ACOM2 = 000000
ACPTOP= 000000
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDX2 = 000000
ADDX3 = 000000
ADDX4 = 000000
ADDX5 = 000000
ADDX6 = 000000
ADDX7 = 000000
ADDX8 = 000000
ADDX9 = 000000
ADEVCT= 000000
ADEVM = 000000
ADPTR   020054          211*    218*    226     611*    2321#
AENV  = 000000
AENVM = 000000
AFATAL= 000000
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMMS1 = 000000
AMMS2 = 000000
AMMS3 = 000000
AMMS4 = 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
APASS = 000000
APRIOR= 000000
APTCSL= 000040          1356    1357#
APTENV= 000001          1352    1356    1357#
APTSIZ= 000200           182    1357#
APTSPC= 000100          1356    1357#
ASAVEG= 000000
ATESTM= 000000
AUNIT = 000000
AUSAR = 000000
AVECT1= 000000
AVECT2= 000000
BAR   = 000012          276#    349     353     357     739#    791#    1111#   1178    1182    1186    1776#   2139#
```

```
BCR    = 000010          368    372    376    735*   765*   776*   787*   1197   1201   1205   1403*  1773*  1795*
                        2138*
BEGIN    001624           18    176*
BEGINA   001642          180*   630    855   1492   1497
BIT0   = 000001           23*   414
BIT00  = 000001           23*
BIT01  = 000002           23*
BIT02  = 000004           23*
BIT03  = 000010           23*
BIT04  = 000020           23*
BIT05  = 000040           23*
BIT06  = 000100           23*   693    748    757
BIT07  = 000200           23*   445   1266
BIT08  = 000400           23*
BIT09  = 001000           23*  1351   1352
BIT1   = 000002           23*   485
BIT10  = 002000           23*   311   1140
BIT11  = 004000           23*   288    303    316    334    354    373    425    444    462    684    713    1121
                         1130  1145   1163   1183   1202   1246   1265   1283   1351   1391   1769
BIT12  = 010000           23*
BIT13  = 020000           23*   723    738    790   1352
BIT14  = 040000           23*   420   1241   1351
BIT15  = 100000           23*   335    511    723   1164   1318
BIT2   = 000004           23*
BIT3   = 000010           23*
BIT4   = 000020           23*
BIT5   = 000040           23*
BIT6   = 000100           23*
BIT7   = 000200           23*
BIT8   = 000400           23*
BIT9   = 001000           23*
BKR    = 000014         2140*
BPTVEC= 000014           23*
BRLVL    017632          213   2235*
BRPTR    020060          213*   220*   228    613*  2323*
BUSER    014750          185   1593*
CAR    = 000006          387    396    736*   788*  1216   1225   1404*  1774*  2137*
CEXIT    020070          690*   706    742*  2329*
CHKADR   014526         1504   1542*
CHKVCT   014640         1512   1569*
CHPS1    017342          277    702   1112   1605   1615   1763   2101*
CHPS2    017356          315    333    352    371    394    423    441    461    679   1144   1162   1181   1200
                        1223   1244   1262   1282   1392   2109*
CHRCNT   017516          264*   388    408    480    809    879*  1094*  1217   1399   1403   1693*  1696*  1710
                        2021   2031   2043   2055   2072   2180*
CKER     004010          304    403*
CKEROP   010476         1131   1310*
CKRST1   013670         1362   1372*
CKRST2   013720         1380*  1606   1616
CLALL    017324         2020   2030   2042   2054   2071   2092*
CLMSG1   024225         1895   2510*
CLMSG2   024263         1919   2512*
CLMSK    020114         1319   2082*  2086*  2339*
CLSEL    017520          265*  1686*  1687   1692*  1695   2182*
```

DZDHNA.P11 CROSS REFERENCE TABLE

```
DSWR   = 177570            23*    24
DT1      020312            30     37     44     51     58     65     72   2359*
DT10     022346           170   2462*
DT2      021116            79     86   2392*
DT3      021272            93   2403*
DT4      021306           100   2407*
DT5      021432           107    114   2415*
DT6      021626           121   2426*
DT7      021756           128    135    142    149    156    163   2434*
EC       023462          2490*
ECBUF    020120           763*   764*   772*   773*   785*   788   1785   1786*  2341*
ECHO     004604            20   627*  1378   1386
ECHO1    004626           200   632*
ECHO2    004652           640*   716    843    863   1517
ECHO3    004722           670    672    679*
ECMSG1   023467           635   2493*
ECMSG2   023566           641   2496*
ECMSG3   023624           697    699   2498*
ECMSG4   023703           701   2500*
EC2      023460           658*   665*   667    669    671    673    803*   805    825*   832*   834    837    839
                          841    891*   900*   902    907   1026*  1035*  1037   1040   1065*  1067   1859*  1866*
                         1868   1871   1873   1875   1893*  1901*  1903   1905   1907*  1908   1912   1916   1926*
                         1933*  1935   1937   1939*  1940   1944   1954*  1962*  1964   1966   1968   1972   1983*
                         1992*  1994   1997   1999   2491*
EC3      023462          1088*  1089*  1091   2492*
EMTVEC=  000030            23*   182*
EM1      020166            28    323   1152   2356*
EM10     020761            77    468   1289   2389*
EM11     021136            84    526   1333   2396*
EM12     021164            91    295   2400*
EM14     021336           105   2412*
EM15     021446           112   2419*
EM16     021474           119   2423*
EM17     021644           126   2431*
EM2      020340            35    342   1171   2364*
EM20     021772           133   2438*
EM21     022034           140   2442*
EM22     022100           147   2446*
EM23     022136           154   2450*
EM24     022174           161   2454*
EM25     022233           168   2459*
EM3      020417            42    361   1190   2369*
EM4      020477            49    380   1209   2373*
EM5      020554            56    400   1229   2377*
EM6      020636            63    431   1252   2381*
EM7      020705            70    451   1272   2385*
ENBUFS   031342          2094   2[illegible]1*
ENDA     004514           601   [illegible]7*   618
ERRVEC=  000004            23*   165*   186*  1351*
EXFLAG   020[illegible]16  681*   724    726    728    732*   746*   766*   777*  2340*  2426
EXPAR    014522          1535   1539*
EXPAT    006004            21   852*  1377   1385
EXPAT1   006026           203   857*
EXPAT2   006052           863*  1520
```

```
EXPAT3  006156            890#    938     958     976     994    1012    1024    1055    1106    1128    1157    1176    1195
                         1214    1234    1257    1277    1294
FILC1   024541           1986    2522#
FILC2   024567           2007    2524#
FILLA   020162            816    1080    1788    1789    1984*   2000*   2344#
FILLB   020164            815    1078    1784    1792    2005*   2016*   2345#
FRMERR  025656            498     596    2548#
FRPTR   025724            498*    504*    553*   2555#
GNS   = ****** U           16    1361
INCL    016306           1884    1893#   1920
INFCHR  016710           1887    1983#
INFCNT  017000           1888    2005#
INMSG0  023126           1529    2481#
INMSG1  023143           1500    2474#
INMSG2  023543           1508    2475#
INMSG3  023606           1522    2476#
INMSG4  023657           1566    2477#
INMSG5  022730           1593    2478#
INMSG6  023004           2479#
INMSG7  023052           1537    2480#
INPAR   014360            209     637     861    1499#   1505
INPARA  014246            196    1474#   1483
INPARC  014334             19    1493#
INPARX  014322             16    1491#
INPAR1  014400           1503    1507#   1513
INPAR3  014420            208    1511    1515#
INPAR4  014462           1525    1527#
INPB    016570           1886    1954#   1978
INRSP   016032           1829#   1883
INSB    016452           1885    1926#   1948
INXSP   015716           1800#   1882
IOTVEC= 000020             23#    182*
KYBD1   015510            239    1750#
KYBD2   015534           1750    1758#
LDFILL  015642            782    1784#
LINE    017676            242*    256     289     294     317     322     336     341     360     379     399     410     427
                          430     446     450     467     482     499     509     525     643*    644*    645     691     696
                         1146    1151    1165    1170    1189    1208    1228    1248    1251    1267    1271    1288    1316
                         1332    1351*   1395    1417    1419*   1421*   1423*   1430*   1770    2254#   2426    2434    2462
LINMSK  017450            276     298     327     346     365     384     404     435     455     472     563*    574     587
                          600*    647*    650*    739     791    1111    1420*   1424*   1426    1776    2148#
LINSEL  017446            587    1426    1527*   1533*   2147#
LPMSG   023774           1861    2501#
LPR   = 000004            694*   1402*   1775*   2136#
LPRIN   016146            652    1859#   1876
LPRPTR  017514            260*   1662    1664    1672*   2179#
LPRTAB  017460            260    2163#
NEWCL   002556            268#    273
NEWLIN  002472            251#    263     300     328     347     366     385     405     436     456     473
NEWLPR  002530            262#    269
NEWPAR  002570            272#    305
NRC   = 000002            458     758    1279    2135#
ORPTR   025722            497*    503*    550*   2554#
OVRERR  025616            537     595    2547#
```

```
PARBIT  017522   270*  1727  1731*  1733  1734  1736*  1738  1740*  1742  1744*  2183#
PARERR  025556   496   594   2546#
PATFLG  020100   890*  909*  953    971   989   1007   2333#
PATLIM  020106   946   964   982    1000  1046  1097   2336#
PBMSG1  024434  1956  2518#
PBMSG2  024502  1977  2520#
PC     =X000007   23#  196*  238*  239*  251*  275*  277*  284*  304*  315*  320*  333*  339*
                352*  358*  371*  377*  394*  397*  423*  428*  441*  448*  461*  465*  522*
                523*  534*  541*  546*  557*  652*  656*  679*  702*  708*  782*  824*  863*
                864*  943*  944*  955*  961*  962*  973*  979*  980*  991*  997*  998*  1009*
               1015*  1017*  1019*  1021*  1043*  1044*  1095*  1110*  1112*  1118*  1131*  1132*  1144*
               1149*  1162*  1168*  1181*  1187*  1200*  1206*  1223*  1226*  1244*  1249*  1262*  1269*
               1282*  1286*  1330*  1347*  1350*  1352*  1353*  1356*  1357*  1362   1392*  1405*  1432*
               1469*  1489*  1504*  1512*  1567*  1594*  1605*  1615*  1628*  1641*  1651*  1697*  1717*
               1753*  1763*  1780*  1796*  1825*  1854*  1882*  1883*  1884*  1885*  1886*  1887*  1888*
               1889*  1922*  1950*  1979*  2001*  2017*  2020*  2026*  2030*  2038*  2042*  2050*  2054*
               2067*  2071*  2077*  2089*  2096*  2104*  2112*  2126*
PEPTR   025720   496*  502*  556*  2553#
PIRQ   = 177772   23#
PIRQVE = 000240   23#
PRSTAT  004324   254   563#  1764
PR0    = 000000   23#
PR1    = 000040   23#
PR2    = 000100   23#
PR3    = 000140   23#
PR4    = 000200   23#
PR5    = 000240   23#
PR6    = 000300   23#
PR7    = 000340   23#
PS     = 177776   23#
PSW    = 177776   23#
PTWAIT  007440  1114#
PWRVEC= 000024   23#  182*  1362*
QUICK   017454   240*  1669*  1671*  1689  1729  2151#
QUICKX  017456   267*  1684   1691*  2153#
RAMA    020110  2057*  2059   2060*  2061*  2064  2337#
RAMB    020112  2059*  2061   2062*  2338#
RBEND   017526   478   538   1298   1301   1341  1344  1398*  1401*  2186#
RBFPTR  025726   463*  464   476*   477*   478   493*  512    518    537*  538   1284*  1285  1296*
                1297*  1298  1300*  1301   1310* 1320  1326   1340*  1341  1343*  1344  1397*  2557#
RBUF    025732   463   493   1284   1310   1397  1348  2563#
RDCHR  = 104405   661   801   828    895    1030  1063  1358   1361#  1862  1897  1929  1958  1988
RDDEC  = 104410   867  1361#  1804   1833
RDLIN  = 104406  1359  1360  1361#
RDOCT  = 104407   642  1361#  1476   1501   1509  1523  1530   2008
RDONE   017524   282   414*  485*   1116   1503* 1396* 2185#  2403
RESERR  015012   187  1608#
RESTRP  015054  1621#
RESTRT  002264   217#  225
REST1   002322   223#  1384
RESVEC= 000010    23#  187*  188*
RETFLG  020104   179*  198   627*   853*   1374  1382  1488*  1496*  1515  1551  1578  2335#
RINT1   003446   244   420#
RINT2   005374   686   757#
```

DZDHNA.P11 CROSS REFERENCE TABLE

```
SBMSG2  024366          1947    2516#
SC_INC  014046           251    1417#
SCLMSK  0174             215*    221     223*    614*    617     2145#
SC   P1 005616           675     798#    838     840
SCNOP2  015564           824    1768#
SCTCL   015226           268    1684#
SCTLPR  015142           262    1662#
SCTPAR  015366           272    1727#
SINGLE  020102          1034*   1042*   1064*   1065    1069    1075    1079    1085    2074    2334#
SNMSG1  024610           860    2335#
SNMSG2  024640           799    2336#
SNMSG3  024720           827    2337#
SOFT    004264           524     548#
SP      =000006           23#    182*    299*    326     345     364     383     403     407*    413     434     454     471
                         475*    486     517*    530     643     662     802     829     868     896    1031    1064    1127
                        1155    1156    1174    1175    1193    1194    1212    1213    1232    1233    1255    1256    1275
                        1276    1292    1293    1325*   1337    1350*   1351*   1352*   1353*   1354*   1355*   1356*   1357*
                        1358*   1359*   1360*   1361*   1362*   1428*   1444*   1458    1477    1502    1510    1524    1531
                        1550*   1577*   1598    1599    1600    1601*   1608    1609    1610    1611*   1640*   1708*   1716
                        1762*   1805    1807    1811    1819    1823    1834    1836    1840    1848    1852    1863    1898
                        1930    1959    1989    2009    2011    2014    2101*   2102*   2109*   2110*   2117*   2118*   2121*
                        2122*   2124    2125
SSR     = 000016        2141#
STACK   = 001100          23#    182     299     1601    1611    1762
START1  002074           185#
START2  002226           205     211#    224     1350    1539
STDH1   002414           238#
STKLMT= 177774            23#
STMSG1  023254           568     570    2484#
STMSG2  023301           557     559    2485#
STMSG3  023326           580     582    2486#
STMSG4  023367           572    2487#
SUBUF1  015326          1697    1708#
SUCLNK  017270           864    2081#
SUER1   014224           320     339     358     377     397     428     448    1149    1168    1187    1206    1226    1249
                        1269    1465#
SUER2   014220           465     523    1286    1330    1464#
SUNUM   014140           236     255     283     321     340     359     378     398     429     449     466     524     565
                         578     695    1150    1169    1188    1207    1227    1250    1270    1287    1331    1444#
SUPATA  017052           943    2020#
SUPATD  017130           979    2042#
SUPATR  017162           997    2054#
SUPATS  017242          1043    2071#
SUPATU  017100           961    2030#
SWR     001136            24#    182#    956     974     992     1010    1022    1053    1104    1351    1352    1534    1665
SWREG   000176            16#    182
SW0     =000001           23#
SW00    =000001           23#
SW01    =000002           23#
SW02    =000004           23#
SW03    =000010           23#
SW04    =000020           23#
SW05    =000040           23#
SW06    =000100           23#
```

DZDHNA.P11 CROSS REFERENCE TABLE

```
SW07  = 000200            23#
SW08  = 000400            23#
SW09  = 001000            23#
SW1   = 000002            23#
SW10  = 002000            23#
SW11  = 004000            23#
SW12  = 010000            23#
SW13  = 020000            23#
SW14  = 040000            23#
SW15  = 100000            23#
SW2   = 000004            23#
SW3   = 000010            23#
SW4   = 000020            23#
SW5   = 000040            23#
SW6   = 000100            23#
SW7   = 000200            23#
SW8   = 000400            23#
SW9   = 001000            23#
TBFPTR  025730           494#   506    519    536#  1311#  1313   1327   1339#  2558#
TBITVE= 000014            23#
TBUF    030212           390    494    521    692    730    733    736    745    800   1062   1072   1092   1219
                        1311   1329   1404   1711   1771   1774   2022   2038   2044   2056   2073   2092   2569#
TCRLF = 000200          1356#
THT   = 000011          1356#
TIMEA   020064           280#   704#   737#   747#   789#  1114#  1638#  2326#
TIMEB   020066           281#   705#  1115#  1636#  2327#
TIMEIT  015102           284    708   1118   1636#
TINT1   002742           247    311#
TINT2   005226           689    723#
TINT3   007540           886   1140#
TITFLG  020062           180#   189    193#  2325#
TITLE   022364           192   2472#
TITLE2  022446           231    233   2473#
TKVEC = 000060            23#
TPVEC = 000064            23#
TRAPVE= 000034            23#   182#
TRTVEC= 000014            23#
TST1    002376           237#
TYPOS = 104404          1350   1353   1361#
TYPE  = 104400           191    232    258    569    571    581    634    640    659    666    698    700    798
                         804    826    833    859    865    873    893    901    905    936   1028   1036   1060
                        1066   1090   1350   1352   1353   1354   1355   1356   1358   1359   1360   1361#  1362
                        1474   1499   1507   1521   1528   1536   1565   1592   1800   1817   1829   1846   1860
                        1867   1894   1902   1918   1927   1934   1946   1955   1963   1976   1985   1993   2006
TYPOC = 104401          1353   1361#
TYPON = 104403          1361#
TYPOS = 104402          1361#
UPDER   004252           522    543#
VCFLG   017630           177#   194    197#   629#   854#  1491#  1494#  2228#
VCPTR   020056           212#   219#   227    612#  2322#
VCWC    023164          1475   2483#
XSMSG1  024046          1801   2502#
XSMSG2  024074          1818   2504#
XSPTAB  017702          1802   1815   2264#
```

```
XSPTR   017700          1802*   1807    1809*   1810    1814*   1815    2262#
XTOTAL  025456           292     412*    592    2544#
$APTHD  000000            15#
$$STAT= ****** U        1357
$ATYC   012546          1357#
$ATYI   012522          1357#
$ATY3   012560          1356    1357#
$ATY4   012540          1352    1357#
$BOSW   001304            24#
$BOOR   001102            24#
$BODAT  001106            24#     182
$CDMI   001310            24#
$CDMO   001312            24#
$CHNAC  012516          1356#*
$CNTAG  001100            24#     182
$CNI  = 000010            24#
$CNO  = 000020            24#
$CSI  = 000010            24#
$CSO  = 000010            24#
$CPUOP  001320            24#
$CRLF   001200            24#     1352    1353    1356    1358    1359    1360
$DBLK   012234          1355#
$DDMO   001314            24#
$DDMI   001316            24#
$DDMI0  001340            24#
$DDMI1  001342            24#
$DDMI2  001344            24#
$DDMI3  001346            24#
$DDMI4  001350            24#
$DDMI5  001352            24#
$DDMO   001354            24#
$DDMI   001356            24#
$DDMS   001360            24#
$DDMS   001362            24#
$DDMS   001364            24#
$DDMS   001366            24#
$DDMS   001370            24#
$DDMS   001372            24#
$DEVCT  001240            24#
$DEVM   001306            24#
$DDMGN  010762          1350#
$DTBL   012224          1355#
$DVDAD  010752          1350#
$DVOCT  010720          1350#
$DVDMG  010766          1350#
$DVNULL 011003          1350#
$DMV    001250            24#     1352    1356    1357
$DMVM   001251            24#     182     1356    1357
$DOP    010664           609*    616     1350#
$DOPCT  010712          1350#
$DRFLG  001103            24#     1351*   1352*
$DRMAX  001115            24#     182*    1351*
$DRROR  011250           182     1352#
$DRRPC  001116            24#     1352*   1353    2359    2392    2403    2415    2426    2434    2462
```

DZDHNA.P11 CROSS REFERENCE TABLE

```
SCRRTB  001354            24#  1353
SERRTY  011416          1352   1353#
SERTTL  001112            24#  1352*
SESCAP  001224            24#   182*  1351*  1352
SFATBL  001250            24#
SFATEN  001354            15     24#
SFATAL  001232            24#  1357*
SFFLG   012764          1357#*
SFILLC  001154            24#  1356
SFILLS  001152            24#  1356
SGDADR  001120            24#
SGDDAT  001124            24#
SGET42  010742          1350#
SHO   = 000001            11
SHIBTS  000000            15#
SHIOCT  013300          1359#*
SICNT   001104            24#  1351*
SILLUP  013652          1362#
SITEMB  001114            24#  1352*  1353
SLF     001226            24#  1352   1356   1358   1359   1360
SLFLG   012765          1357#*
SLPADR  001106            24#   182*   236*  1351*
SLPERR  001110            24#   182*   296*   324*   343*   362*   381*   401*   432*   452*   469*   527*   597*
                         714#   951*   969*   987*  1005*  1051*  1102*  1124*  1153*  1172*  1191*  1210*  1230*
                        1253#  1273*  1290*  1334*  1351*  1352   1602*  1612*
SMADR1  001262            24#
SMADR2  001266            24#
SMADR3  001272            24#
SMADR4  001276            24#
SMAIL   001230            15     24#   182    237    1351   1352   1356
SMDMS1  001260            24#
SMDMS2  001264            24#
SMDMS3  001270            24#
SMDMS4  001274            24#
SMERDR  000002            15#
SMFLG   012764          1357#*
SMSGAD  001244            24#  1357*
SMSGLG  001246            24#  1357*
SMSGTY  001230            24#  1357*
SMTYP1  001261            24#
SMTYP2  001265            24#
SMTYP3  001271            24#
SMTYP4  001275            24#
SMXCNT  011246          1351#
SNULL   001152            24#  1356
SNXTST= 000001           237#
SOCNT   012014          1354#*
SOMODE  012016          1354#*
SOVER   011232          1351#
SPASS   001236            24#   182*  1350*  1351
SPASTM  000006            15#
SPOWER  013660          1362#
SPRIOR  001302            24#
SPWRAD  013646          1362#
```

F15


```
$PWRDN  013524            182    1362#
$PWRMG  013642           1363#
$PWRUP  013572           1362#
$QUES   001224             24#   1352    1356    1358    1359    1360
$RDCHR  012770           1358#   1361
$RDOEC  013302           1360#   1361
$RDOLIN 013024           1358#   1361
$RDOCT  013142           1359#   1361
$RDSZ = 000010           1358#
$REG00  001156             24#
$REG20  001160             24#    949#    967#    985#   1003#   1049#   1100#   1122#   1464#   2392    2434    2462
$REG21  001162             24#    711#   1123#   1465#   1599#   1609#   2359    2392    2415    2426    2462
$REG22  001164             24#   1466#   1600#   1610#   2359    2392    2415
$REG23  001166             24#   1467#   2359    2392
$REG24  001170             24#   1468#   2359    2392
$REG25  001172             24#
$REG26  001174             24#   1598#   1608#   2415
$REG27  001176             24#
$R2A  = ******  U        1361
$SAVRE= ******  U        1361
$SAVR6  013656           1362#*
$SCOPE  011006            182    1351#
$SETUP= 000017            181#    182    1350    1352
$STUP = 177777            181#
$SVLAD  011176           1351#
$SVPC = 000000             14#
$SWR  = 165000              3#     11      12      24     182     237    1350    1351    1352    1362
$  G   001252             24#    182
$  K=  000000           1351
$TESTN  001234             24#    237#   1351#
$TIMES  001220             24#    182#    237#   1350#   1351#
$TKB    001144             24#   1358
$TKS    001142             24#   1358
$TMP0   001200             24#    291#    458#    476     589#    712#    758#    759#    760     769     780     785     786
                         1279#   1296    1786    1787    2125#   2403    2407    2415    2426
$TMP1   001202             24#    292#    591#   2403    2407
$TMP2   001204             24#    592#   2407
$TMP3   001206             24#    593#   2407
$TMP4   001210             24#    594#   2407
$TMP5   001212             24#    595#   2407
$TMP6   001214             24#    596#   2407
$TMP7   001216             24#    516#    531     548#    551#    554#    577#    579    1324#
$TN   = 000002             11#    237#
$TPB    001150             24#   1356#
$TPFLG  001155             24#   1356
$TPS    001146             24#   1356
$TRAP   013460            182    1361#
$TRP  = 000011           1361#
$TRPAD  013502           1361#
$TSTM   000004             15#
$TSTNM  001102             24#    608#   1350#   1351#   1352
$TTYIN  013132           1358#
$TYPBN= ******  U        1361
$TYPDS  012020           1355#   1361
```

DZDHNA.P11 CROSS REFERENCE TABLE

```
$TYPE   012244            1356#   1357    1361
$TYPEC  012452            1356#
$TYPEX  012520            1355#
$TYPOC  011616            1354#   1361
$TYPON  011632            1354#   1361
$TYPOS  011572            1354#   1361
$UNIT   001245              24#
$UNITM  000010              15#
$USWR   001254              24#
$VECT1  001300              24#
$VECT2  001301              24#
$XTSTR  011022            1351#
$OFILL  012015            1354#*
$40CAT= ****** U          1351    1352
.       = 031344            14#     15#     16#     24#     182     236    1350    1351    1352    1353#   1355#   1356    1357#
                          1358#   1359    1360    1362    2341#   2425#  2433#  2461#  2543#  2544#  2545#  2546#  2547#
                          2548#   2563#   2569#
.$ASTA= ****** U          1357
.$$   = 000000              15#
```

DZDHNA.P11 CROSS REFERENCE TABLE

```
COMMEN    23#
ENDCOM    23#
ERROR     23#   297   325   344   363   382   402   433   453   470   528   598   715   952   970
          988  1006  1052  1103  1125  1154  1173  1192  1211  1231  1254  1274  1291  1335  1603
         1613
ESCAPE    23#
MULT      23#
NEWTST    23#   237
POP       23#   413   486  1355  1357  1359  1360  1362  1458  1716
PUSH      23#   407   475  1355  1357  1359  1360  1362  1444  1708
SCOPE     23#   237   607  1350
SETTRA  1361#
SETUP       5#   23#   182
SKIP      23#
SLASH     23#
SPACE     23#
STARS     14    15    23#   24   237  1350  1351  1352  1353  1354  1355  1356  1357  1358  1359
         1360  1361  1362  1365  1367  2131  2133  2350  2352  2468  2470
SWRSU     182#
TRMTRP  1361#
TYPBIN    23#
TYPDEC    23#  1350  1353
TYPNUM    23#
TYPOCS    23#
TYPOCT    23#  1353
TYPTXT    23#
$$CMRE    24#
$$CNTH    24#
$$ESCA    23#
$$NEWT    23#   237
$$SET   1361#
$$SKIP    23#
.EQUAT      5#    23
.HEADE      4#    11
.SETUP      4#   181
.SWRHI      5#    12
.SWRLO      5#    12#    13
.$ACTI      8#    14
.$APTB      8#    24#
.$APTH      8#    15
.$APTY      8#  1357
.$CATC      5#    16
.$CNTA      4#    24
.$EOP       5#  1350
.$ERRO      4#  1352
.$ERRT      6#  1353
.$POKE      6#  1362
.$RDDE      7#  1360
.$RDOC      7#  1359
.$READ      7#  1358
.$SCOP      6#  1351
.$TRAP      6#  1361
.$TYPD      6#  1355
.$TYPE      6#  1356
```

DZDHNA.P11 CROSS REFERENCE TABLE

.STYPO 6# 1354

DZDHNA.P11 CROSS REFERENCE TABLE

```
        724   726   728   730   760   769   780   806   809   811   871   912   916   920   924
        928   932   946   953   964   971   982   989  1000  1007  1046  1069  1072  1075  1097
       1155  1174  1193  1212  1220  1232  1255  1275  1292  1298  1301  1321  1341  1344  1351
       1355  1358  1479  1481  1542  1545  1562  1569  1572  1589  1649  1662  1667  1687  1727
       1738  1742  1807  1815  1836  1844  2094
CMPB    543   663   671   673   830   839   841   897  1032  1351  1352  1356  1357  1358  1359
       1360  1758  1864  1873  1875  1899  1908  1912  1916  1931  1940  1944  1960  1968  1972
       1990
COM     193
COMB   1419
DEC     648   817  1081  1350  1353  1638  1790  2087
DECB   1354  1356  2047
EMT      23
HALT     16  1352  1356  1362  1538
INC     217   536   545   550   553   556   599   610   646   945   963   981   999  1045  1096
       1339  1350  1351  1352  1354  1355  1357  1362  1636  1686  1691  1713  1714  1793  2024
       2036  2048  2065  2075
INCB   1351  1352  1356  1423  2035
IOT      23
JMP      16    18    19    20    21   200   203   208   209   254   328   347   366   385   405
        436   456   473   616   619   630   637   675   716   843   855   861   910   914   918
        922   926   930   934   958   976   994  1012  1024  1055  1106  1128  1157  1176  1195
       1214  1234  1257  1277  1294  1350  1376  1377  1378  1384  1385  1386  1492  1497  1517
       1520  1539  1606  1616  1764
JSR     196   229   238   239   251   255   262   268   272   275   277   284   293   304   315
        320   321   333   339   340   352   358   359   371   377   378   394   397   398   423
        428   429   441   448   449   461   465   466   522   523   524   534   565   578   652
        679   695   702   708   782   824   863   864   943   944   961   962   979   980   997
        998  1015  1017  1019  1021  1043  1044  1095  1110  1112  1118  1131  1144  1149  1150
       1162  1168  1169  1181  1187  1188  1200  1206  1207  1223  1226  1227  1244  1249  1250
       1262  1269  1270  1282  1286  1287  1330  1331  1350  1352  1356  1357  1392  1504  1512
       1605  1615  1697  1763  1882  1883  1884  1885  1886  1887  1888  2020  2030  2042  2054
       2071
MOV     182   185   186   187   188   211   212   213   214   215   226   227   228   236   237
        243   244   247   249   260   264   265   280   291   292   296   299   303   314   316
        319   324   330   334   335   338   343   353   354   356   362   372   373   375   381
        387   388   395   401   407   408   413   424   425   432   442   444   445   447   452
        458   462   464   469   475   476   480   486   493   494   495   496   497   498   512
        516   517   518   519   520   527   530   563   577   585   589   591   592   593   594
        595   596   597   609   627   632   633   643   645   647   662   682   683   684   685
        686   689   691   692   694   704   711   712   714   732   735   736   737   739   742
        745   747   758   765   766   776   777   787   788   799   791     0   802   829   852
        857   858   868   877   879   881   882   883   886   892   896   899   909   949   951
        967   969   985   987  1003  1005  1014  1016  1018  1020  1031  1034  1042  1049  1051
       1062  1064  1078  1094  1100  1102  1114  1121  1122  1123  1124  1143  1145  1148  1153
       1159  1163  1164  1167  1172  1182  1183  1185  1191  1201  1202  1204  1210  1216  1217
       1224  1230  1245  1246  1253  1263  1265  1266  1268  1273  1279  1283  1285  1290  1296
       1310  1311  1320  1324  1325  1326  1327  1328  1334  1337  1350  1351  1352  1353  1354
       1355  1356  1357  1358  1359  1360  1361  1   2  1391  1393  1397  1398  1399  1402  1403
       1404  1420  1444  1445  1446  1448  1458  1464  1465  1466  1467  1468  1477  1484  1486
       1491  1493  1502  1510  1524  1526  1527  1531  1533  1553  1557  1559  1560  1580  1584
       1586  1587  1598  1599  1600  1601  1602  1608  1609  1610  1611  1612  1621  1622  1625
       1647  1664  1669  1693  1708  1710  1711  1716  1731  1736  1740  1744  1750  1751  1752
       1762  1768  1769  1770  1773  1774  1775  1776  1785  1795  1802  1831  1863  1880  1898
```

L15

DZDHNA.P11 CROSS REFERENCE TABLE

```
          1930  1959  1989  2021  2022  2031  2032  2043  2044  2045  2055  2056  2057  2072  2073
          2081  2082  2092  2101  2102  2109  2110  2118  2119  2121  2122  2124  2125
MOVB       182   245   248   289   317   410   427   482   499   506   665   687   690   763   764
           772   773   774   775   785   786   803   814   815   816   821   832   984   887   900
          1045  1065  1079  1080  1085  1088  1089  1146  1248  1313  1351  1352  1354  1355  1356
          1357  1358  1359  1360  1361  1447  1454  1457  1712  1784  1786  1787  1788  1789  1792
          1866  1901  1933  1962  1992  1999  2000  2011  2013  2016  2023  2034  2046  2064  2074
NEG        389   409   481   734   878  1093  1218  1354  1355  1360  1400  1772  1794
NOP       1350
RESET     1350  1604  1614  1761
ROL       1354  1359
RTI        415   487   750   792  1236  1304  1351  1352  1354  1355  1356  1358  1359  1360  1362
          1760  2103  2111  2123
RTS        541   546   557   656   955   973   991  1009  1132  1347  1353  1356  1357  1361  1405
           432  1459  1469  1489  1567  1594  1628  1641  1651  1674  1699  1717  1746  1753  1780
          1798  1825  1854  1889  1922  1950  1979  2001  2017  2026  2038  2050  2067  2077  2089
          2096  2104  2112  2126
SUB        463   521   733  1092  1284  1329  1352  1355  1357  1360  1771
SWAB       507   510  1314  1317
TRAP      1361  2120
TST        182   189   194   198   201   204   349   368   654   688   706   885  1116  1127  1156
          1175  1178  1194  1197  1213  1233  1256  1276  1293  1351  1352  1353  1354  1355  1356
          1357  1359  1360  1361  1372  1374  1380  1382  1515  1518  1534  1551  1555  1578  1582
          1684  1689  1729  1734  1778  1805  1811  1819  1823  1834  1840  1848  1852  1878  2009
          2014  2084
TSTB       246   438   669   837   907   956   974   992  1010  1022  1040  1053  1104  1259  1361
          1353  1355  1356  1357  1358  1417  1665  1871  1905  1937  1966  1997
.ASCII      24
.ASCIZ      24  1350  1353  1362  2356  2357  2364  2369  2373  2377  2381  2395  2389  2390  2396
          2400  2401  2412  2413  2419  2423  2424  2431  2432  2438  2442  2446  2450  2454  2459
          2460  2472  2473  2474  2475  2476  2477  2478  2479  2480  2481  2483  2484  2485  2486
          2487  2490  2491  2492  2493  2496  2498  2500  2501  2502  2504  2506  2513  2510  2512
          2514  2516  2518  2520  2522  2524  2525  2526  2527  2532  2533  2534  2535  2536  2537
          2538  2539
.BLKB     1358  2569
.BLKW     1355  2541  2543  2544  2545  2546  2547  2548  2563
.BYTE       24  1350  1352  1354  1357  1358  2251  2252  2360  2408
.ENABL      10
.END      2573
.ENDC       11    12    14    15    16    23    24   181   182   237  1350  1351  1352  1353  1354
          1355  1356  1357  1358  1359  1360  1361  1362  1365  1367  2131  2133  2350  2352  2468
          2470
.EQUIV      23
.EVEN       24  1353  1357  1362  2358  2391  2402  2414  2425  2433  2461  2464  2540  2568
.IF         11    12    14    15    16    23    24   181   182   237  1350  1351  1352  1353  1354
          1355  1356  1357  1358  1359  1360  1361  1362  1365  1367  2131  2133  2350  2352  2468
          2470
.IFF        12    14    15    23    24   182   237  1350  1351  1352  1353  1354  1355  1356  1357
          1358  1359  1360  1361  1362  1365  1367  2131  2133  2350  2352  2468  2470
.IFT      1351  1352  1358  1359
.IFTF     1351  1352  1358  1359
.IIF        11    12    13    16    24   182  1350  1351  1352  1353  1356  1358  1359  1360  1361
.IRP       181   237   407   413   475   486  1351  1355  1357  1359  1360  1362  1444  1458  1708
          1716
```

DZDHNA.P11 CROSS REFERENCE TABLE

```
.LIST      2    16    23    24   181   182   237  1350  1351  1352  1358  1361
.MACRO    12    24  1361
.MCALL     4     5     6     7     8    23    24   182
.NLIST     1    16    23    24   181   182   237  1350  1351  1352  1358  1361
.PAGE     22    24   184   235   307   417   489   606   620   678   720   754   797   845   889
         941  1058  1109  1136  1238  1306  1349  1364  2348  2465  2466  2530
.REPT     16    24
.SBTTL    12    14    15    16    23    24   237   621   846  1350  1351  1352  1353  1354  1355
        1356  1357  1358  1359  1360  1361  1362  2130  2349  2467
.TITLE    11
.WORD     14    15    16    24  1350  1353  1354  1356  1357  1359  1360  1362  2359  2392  2403
        2407  2415  2426  2434  2462
```

ERRORS DETECTED: 0

*DZDHNA,DZDHNA/CRF=DZDHNA
RUN-TIME: 69 36 7 SECONDS
CORE USED: 23K

Spooler spooling 28 Seconds, 135 KCS, 895 disk reads, 1 disk writes, 211 pages printed