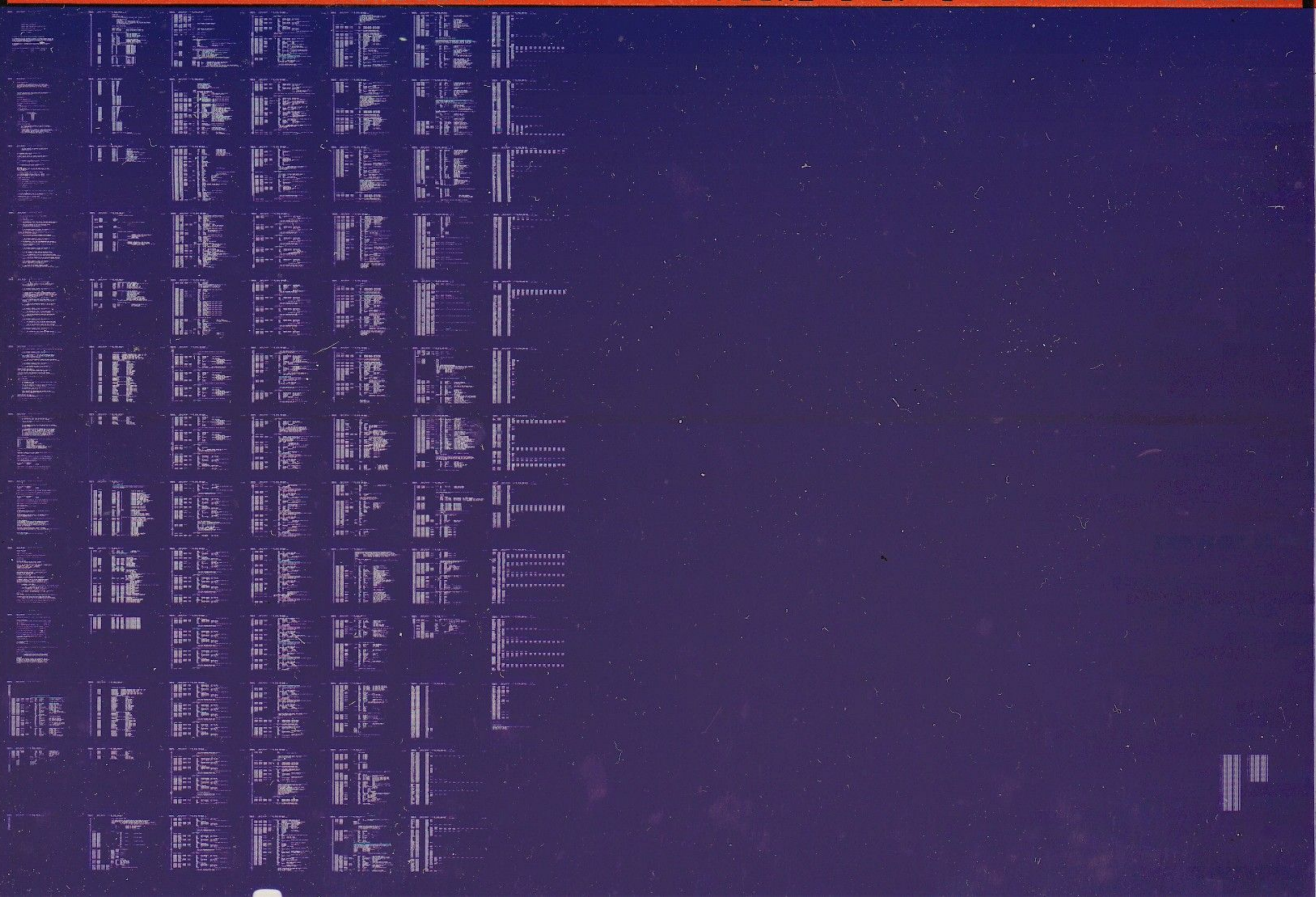
DUV11 OFLNE LGC TSTS
CNDUQAO

AH-T438A-MC
FICHE 1 OF 1



```
.REM    *
```

I D E N T I F I C A T I O N

PRODUCT CODE: AC-T437A-MC

PRODUCT NAME: CNDUQA0 DUV11 OFLNE LGC TSTS

PRODUCT DATE: DEC , 1982

MAINTAINER: DIAGNOSTIC SERVICES/ISS

AUTHOR: J.CARMODY

```
*
.REM    *
```



```
*
```

.REM *

GENERAL DESCRIPTION

THIS DIAGNOSTIC CAN CHAIN 16 DUV11'S. THIS MEANS THAT
16 DEVICES CAN BE SEQUENTIALLY EXERCISED. THE DIAGNOSTIC
MAKES ONE PASS BEFORE PROCEEDING TO THE NEXT DEVICE,
AND CONTINUES EXERCISING ALL DEVICES IN THIS FASHION UNTIL
HALTED.

*
.REM *

1. THE DUV11 OFFLINE LOGIC TESTS VERIFY THAT ALL REGISTERS EXIST
,AND ALL RESPECTIVE BITS CAN BE MASTER CLEARED, READ,
WRITTEN AND/OR READ/WRITTEN

*
.REM *

2. REQUIREMENTS

PDP-11/21 COMPUTER (LSI)

DUV11 SYNCRONOUS/ISOCRONOUS OPTION

ONE CONSOLE TELETYPE OR EQUIVALENT

2.2 STORAGE
THE PROGRAM LOADS INTO 4K OF MEMORY WITH BOOTSTRAP

3. LOADING PROCEDURE

THE STANDARD PROCEDURE FOR LOADING ABSOLUTE BINARY TAPES
IS TO BE USED.

	STARTING ADDRESS FOR ABSOLUTE LOADER
4K	017500
8K	037500
12K	057500
16K	077500
20K	117500
24K	137500
28K	157500

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

NOTE: ALL SWITCHES RESIDE INTERNAL TO THE CPU AT ADDRESS
176. THESE MAY BE SET VIA THE CONSOLE TTY BY DIRECTLY
MODIFYING LOC. 176.

NOTE: RUNNING UNDER APT-11, THERE IS A USER SWITCH REGISTER
CALLED "$USWR". IN ORDER TO BE FLEXIBLE ON THE AVAILIBILITY OF THE
H315 CONNECTOR, ONE BIT PASSES STATUS TO APT-11.
BIT 0 IN $USWR REFLECTS THIS STATUS, A 0 = CONNECTOR
PRESENT, A 1 = CONNECTOR NOT AVAILIBLE.

THE USER CHANGES THE CONTENTS OF THIS LOCATION
WHEN BUILDING THE E TABLE, BY ANSWERING THE
PROMPT "SWITCH 2".

4.1.1 AFTER PROGRAM LOAD (INITIAL PROGRAM START)
ALL CONSOLE SWITCHES DOWN

4.1.2 TO MODIFY DEVICE VECTOR AND CONTROL REGISTER ADDRESSES
AFTER PROGRAM RESTART OR TO RUN MULTIPLE DEVICES

SW00=1

4.1.3 TO START PROGRAM AT SELECTED TEST AFTER A PROGRAM RESTART
(ONLY IN SINGLE DEVICE TESTS)
SW01=1

4.1.4 TO LOCK ON SELECTED TEST AFTER A PROGRAM RESTART
(ONLY IN SINGLE DEVICE TESTS)

SW14=1
NOTE1: IN GENERAL SW01 WILL BE USED WHEN SW14=1 IS USED
NOTE2: WITHOUT SW01=1 "LOCK ON TEST" WILL DEFAULT TO TEST 1

4.2 STARTING ADDRESS

THE STARTING ADDRESS FOR ALL TESTS IS 000200

THE STARTING ADDRESS FOR ALL TESTS IS 000200
THE STARTING ADDRESS TO ENTER A SELECTED TEST IS 000200
THE STARTING ADDRESS TO LOCK ON TEST IS 000200

4.3 PROGRAM AND/OR OPERATOR ACTION

4.3.1 INITIAL PROGRAM START

4.3.1.1 LOAD PROGRAM INTO MEMORY WITH ABSOLUTE LOADER

4.3.1.2 SET SWITCH REGISTER (LOC. 176) TO ZERO.

4.3.1.3 TYPE 200G.

4.3.1.4 PROGRAM WILL START.

4.3.1.5 THE PROGRAM WILL TYPE "DUV11 CZDUQ-C TAPE A" (ONCE ONLY)

.REM *

.REM *

4.3.1.6 THE PROGRAM WILL TYPE "R" TO INDICATE THAT IT IS ABOUT
TO START TESTING ,AND THEN TESTING WILL BEGIN

4.3.2 PROGRAM RESTART WITH ALL SWITCHES DOWN

4.3.2.1 THE PROGRAM WILL TYPE "R" AND WILL COMMENCE TESTING

4.3.3 PROGRAM RESTART WITH SW00=1

4.3.3.1 SET SWITCH REGISTER (LOC. 176) TO A 000001.

4.3.3.2 TYPE 200G.

4.3.3.3 PROGRAM WILL START.

4.3.3.4 THE PROGRAM WILL TYPE '' 1ST DEVICE: RECEIVER CONTROL REGISTER ADDRESS'' AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.3.5 TYPE IN THE ADDRESS OF THE FIRST RECEIVER CONTROL REGISTER ADDRESS OF THE DUV11 TO BE TESTED FOLLOWED BY A <CARRIAGE RETURN>

IF AN INCORRECT ADDRESS IS TYPED ,THE PROGRAM WILL TYPE ''?'' AND WILL THEN REPEAT THE MESSAGE OF 4.3.3.4

4.3.3.6 THE PROGRAM WILL TYPE ''VECTOR ADDRESS-'' AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.3.7 TYPE IN THE BASE RECEIVER INTERRUPT VECTOR ADDRESS FOR THE DUV11 TO BE TESTED FOLLOWED BY A <CARRIAGE RETURN>

IF AN INCORRECT ADDRESS IS TYPED ,THE PROGRAM WILL TYPE ''?'' AND WILL THEN REPEAT THE MESSAGE OF 4.3.3.6

4.3.3.8 THE PROGRAM WILL TYPE ''ARE YOU RUNNING MULTIPLE DEVICES ?'' (Y OR N)-'' AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.3.9 TYPE IN THE APPROPRIATE ANSWER YES OR NO FOLLOWED BY A <CARRIAGE RETURN>

IF AN INCORRECT ANSWER IS GIVEN, THE PROGRAM WILL TYPE ''?'' AND WILL THEN REPEAT THE MESSAGE OF 4.3.3.8

IF A ''NO'' ANSWER IS GIVEN: JUMP TO SECTION 4.3.3.12
IF A ''YES'' ANSWER IS GIVEN:THE NEXT QUESTION IS ASKED

4.3.3.10 THE PROGRAM WILL TYPE ''LAST DEVICE:RECEIVER CONTROL REGISTER ADDRESS-'' AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.3.11 TYPE IN THE ADDRESS OF THE LAST RECEIVER CONTROL REGISTER ADDRESS OF THE DUV11 TO BE TESTED FOLLOWED BY A <CARRIAGE RETURN>

IF AN INCORRECT ANSWER IS TYPED ,THE PROGRAM WILL TYPE ''?'' AND WILL THEN REPEAT THE MESSAGE OF 4.3.3.10
NOTE:ALL ADDRESSES SHALL BE CONTIGUOUS

4.3.3.11.1 IF AN ''OUT OF RANGE'' ADDRESS IS TYPED IE. MORE THAN 16 (10) DEVICES AWAY (UPWARDS).....THE PROGRAM WILL TYPE ''OUT OF RANGE:RETYPE LAST DEVICE RXCSR ADDRESS-''

AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.3.11.2 TYPE IN THE ADDRESS OF THE LAST RECEIVER CONTROL REGISTER ADDRESS OF THE DUV11 TO BE TESTED FOLLOWED BY A <CARRIAGE RETURN>

IF AN INCORRECT ANSWER IS TYPED ,THE PROGRAM WILL TYPE "?" AND WILL REPEAT THE MESSAGE OF 4.3.3.11.1

IF A DEVICE ADDRESS LOWER THAN 1ST DEVICE ADDRESS IS TYPED.....
....SCHOOLS OUT.....THERE IS NO PROTECTION FOR THIS.
THE PROGRAM WILL DEFAULT TO TWO DEVICES ACTIVE (UPWARDS FROM 1ST DEVICE ADDRESS).THE SAME APPLIES TO IDENTICAL ADDRESSES TYPED FOR FIRST AND LAST DEVICE.
OBSERVE LOCATION @ ACTREG: SEE SECTION 7.2

4.3.3.12 THE PROGRAM WILL TYPE "# OF SYNC CHARS SELECTED (1 OR 2)-" AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD. REFER TO MANUAL FOR PROPER SWITCH SETTINGS OF SWITCH E55-4.

4.3.3.13 TYPE IN THE APPROPRIATE ANSWER "1" OR "2" FOLLOWED BY A <CARRIAGE RETURN>.(NOTE:ALL MULTIPLE DEVICES MUST BE THE SAME)

IF AN INCORRECT ANSWER IS TYPED ,THE PROGRAM WILL TYPE "?" AND WILL REPEAT THE MESSAGE OF 4.3.3.12

4.3.3.14 THE PROGRAM WILL TYPE " IS SEC XMIT SWITCH E55-2 ON? (Y OR N)-" AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.3.15 TYPE IN THE APPROPRIATE ANSWER YES OR NO FOLLOWED BY A <CARRIAGE RETURN>.(NOTE THAT ALL MULTIPLE DEVICES MUST BE THE SAME)

IF AN INCORRECT ANSWER IS TYPED ,THE PROGRAM WILL TYPE "?" AND WILL REPEAT THE MESSAGE OF 4.3.3.14

4.3.3.16 THE PROGRAM WILL TYPE "IS SEC REC SWITCH E55-3 ON? (Y OR N)-" AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.3.17 TYPE IN THE APPROPRIATE ANSWER YES OR NO FOLLOWED BY A <CARRIAGE RETURN>. (NOTE: ALL MULTIPLE DEVICES MUST BE THE SAME)

IF AN INCORRECT ANSWER IS TYPED ,THE PROGRAM WILL TYPE "?" AND WILL REPEAT THE MESSAGE OF 4.3.3.16

4.3.3.18 THE PROGRAM WILL TYPE "IS OPT CLR ENABLE SWITCH E55-1 ON? (Y OR N)-" AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.3.19 TYPE IN THE APPROPRIATE ANSWER YES OR NO FOLLOWED

BY A <CARRIAGE RETURN>. (NOTE: ALL MULTIPLE DEVICES MUST BE THE SAME)

IF AN INCORRECT ANSWER IS TYPED ,THE PROGRAM WILL TYPE "?"
AND WILL REPEAT THE MESSAGE OF 4.3.3.18

4.3.3.20 THE PROGRAM WILL TYPE "ARE YOU RUNNING IN MAINT.
MODE EXTERNAL ? AND .....DO YOU HAVE THE EXTERNAL MODEM
BYPASS JUMPER CONNECTOR ON ? (Y OR N)-" AND WAIT FOR AN
INPUT FROM THE TELETYPE KEYBOARD

4.3.3.21 TYPE IN THE APPROPRIATE ANSWER YES OR NO FOLLOWED BY
A <CARRIAGE RETURN>. (NOTE: ALL MULTIPLE DEVICES MUST BE THE SAME)

IF AN INCORRECT ANSWER IS TYPED ,THE PROGRAM WILL TYPE "?"
AND WILL REPEAT THE MESSAGE OF 4.3.3.20

4.3.3.22 THE PROGRAM WILL TYPE "R" TO INDICATE THAT IT
HAS STARTED AND WILL COMMENCE TESTING AT TEST 1

4.3.4 PROGRAM RESTART WITH SW01=1
NOTE: THIS WILL ONLY WORK WHEN A SINGLE DEVICE IS SELECTED
,,,IT WILL NOT WORK IF MULTIPLE DEVICES ARE SELECTED

IF MULTIPLE DEVICES WERE PREVIOUSLY SELECTED,LOAD 000200,
AND SELECT SW00=1 AND ANSWER "NO" TO THE MULTIPLE DEVICE QUESTION
SEE 4.3.3

4.3.4.1 SET SW01=1 IN SWITCH REG (LOC. 176)

4.3.4.2 TYPE 200G.

4.3.4.3 PROGRAM WILL START.

4.3.4.4 THE PROGRAM WILL TYPE "TEST PC-" AND WAIT FOR AN INPUT FROM
THE TELETYPE KEYBOARD

4.3.4.5 TYPE IN THE ADDRESS OF THE TEST AT WHICH THE PROGRAM IS TO
BE STARTED FOLLOWED BY A <CARRIAGE RETURN>

4.3.4.6 THE PROGRAM WILL TYPE "R" TO INDICATE THAT IT HAS STARTED
TESTING AT THE SELECTED TEST

NOTE: CARE MUST BE TAKEN WHEN THIS FEATURE IS USED
,SINCE THERE IS NO PROTECTION AGAINST SELECTING AN ADDRESS
THAT IS IN THE MIDDLE OF A TEST

4.3.5 PROGRAM RESTART WITH SW14 =1
NOTE: THIS WILL ONLY WORK WHEN A SINGLE DEVICE IS SELECTED
SEE NOTE IN 4.3.4 FOR MORE DETAILS

4.3.5.1 SET SW14=1 IN SWITCH REG. (LOC. 176)

4.3.5.2 TYPE 200G.

4.3.5.3 PROGRAM WILL START.

4.3.5.4 THE PROGRAM WILL TYPE "LOCK ON SELECTED TEST ? (Y OR N)-"
AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

4.3.5.5 TYPE IN THE APPROPRIATE ANSWER YES OR NO FOLLOWED BY A
<CARRIAGE RETURN>

IF A NO ANSWER IS GIVEN: THIS LOCK ON TEST WILL BE IGNORED
AND THE PROGRAM WILL TYPE "R" TO INDICATE THAT IT HAS STARTED
TESTING AT TEST 1

4.3.5.6 IF A YES ANSWER WAS GIVEN:THE PROGRAM WILL ACT AS FOLLOWS...
THE PROGRAM WILL TYPE "R" TO INDICATE THAT IT HAS STARTED
TESTING AT TEST 1 AND WILL REMAIN IN TEST 1 UNTIL HALTED
OR IF ANY KEY IS STRUCK ON THE TELETYPE ,THE PROGRAM
WILL FREEZE ON THE NEXT TEST UNTIL A KEY IS STRUCK ON
THE TELETYPE AND SO FORTH THRU THE PROGRAM. IF SW01 =1 IT
WILL PERFORM AS IN SECTION 4.3.4 ALLOWING ONE TO FREEZE
ON A SELECTED TEST RATHER THAN DEFAULTING TO TEST 1

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS (INTERNAL TO THE CPU, ACCESSED VIA LOC. 176).

SW15 =1	HALT ON ERROR
SW14 =1	LOOP ON CURRENT TEST
SW13 =1	INHIBIT ERROR TYPEOUT
SW11 =1	INHIBIT ITERATIONS
SW10 =1	ESCAPE TO NEXT TEST ON ERROR
SW09 =1	LOOP ON ERROR
SW01 =1	RESTART PROGRAM AT SELECTED TEST
SW00 =1	RESELECT VECTOR AND CONTROL REGISTER ADRESSES &PARAMETERS AFTER A PROGRAM RESTART

TO INHIBIT "END OF PASS" TYPEOUT - TURN TELETYPE OFF

6. ERRORS

6.1 ERROR HALTS (UNDER LSI ALL HALT ERRORS RETURN CONTROL TO O.D.T.)
THERE ARE FOUR DISTINCT ERROR TYPEOUTS

6.1.1 PC+2 = ERROR PC
WHERE PC +2 IS THE ADDRESS OF THE CALL TO THE ERROR HANDLER +2

REFER TO THE ABOVE "HLT" IN DIAGNOSTIC FOR ERROR DESCRIPTION

CHECK ADDRESS @ RXCSR: TO LOCATE THE DEVICE PRESENTLY UNDER
TEST WHEN RUNNING MULTIPLE DEVICES

6.1.2 PC +2 = REGISTER ERROR PC

REGISTER	EXPECTED	ACTUAL
1XXXXX	YYYYYY	ZZZZZZ

WHERE 1XXXXX IS THE ADDRESS OF THE FAILING DEVICE REGISTER

WHERE YYYYYY IS THE EXPECTED CONTENTS OF THAT REGISTER

WHERE ZZZZZZ IS THE ACTUAL CONTENTS OF THAT REGISTER

6.1.3 PC +2 = RECEIVER ERROR PC

REGISTER	EXPECTED	ACTUAL
1XXXXX	YYYYYY	ZZZZZZ

WHERE 1XXXXX IS THE ADDRESS OF THE FAILING RECEIVER (RXDBUF) REGISTER

WHERE YYYYYY IS THE EXPECTED DATA CONTENTS OF THAT REGISTER

WHERE ZZZZZZ IS THE ACTUAL DATA CONTENTS OF THAT REGISTER

6.1.4 PC +2 = TRANSMITTER ERROR PC

REGISTER	EXPECTED	ACTUAL
1XXXXX	YYYYYY	ZZZZZZ

WHERE 1XXXXX IS THE ADDRESS OF THE FAILING TRANSMITTER (TXCST) REGISTER

WHERE YYYYYY IS THE EXPECTED CONTENTS OF THAT REGISTER

WHERE ZZZZZZ IS THE ACTUAL CONTENTS OF THAT REGISTER

6.1.5 ERROR DESCRIPTIONS
SEE LISTINGS FOR DETAILS OF ERRORS

6.2 ERROR RECOVERY

6.2.1 SW15 =0
IF THE PROGRAM IS RUN WITH SW15 =0 ,NO OPERATOR ACTION IS
REQUIRED TO CONTINUE TESTING

6.2.2 SW15 =1
IF THE PROGRAM IS RUN WITH SW15 =1 ,TO CONTINUE TESTING
AFTER THE PROGRAM HAS HALTED ,PRESS THE PROCESSOR
CONSOLE ''CONTINUE SWITCH''

NOTE: THE PC + 2 OF THE ''HLT'' WILL BE DISPLAYED IN THE DATA LIGHTS

6.2.3 ILLEGAL INTERRUPTS
IF AN INTERRUPT OCCURS TO A VECTOR ADDRESS NOT SELECTED
DURING PROGRAM INITIALIZATION, THE PROGRAM WILL HALT IN
THE TRAPCATCHER. THE ADDRESS AT WHICH THE PROGRAM
HALTS IS 2 GREATER THAN THE ADDRESS TO WHICH THE INTERRUPT
OCCURED. THE PROGRAM MUST BE RESTARTED AT 000200 TO
RECOVER FROM THIS ERROR.

6.2.4 ADDITIONAL TROUBLESHOOTING AIDS ERRCNT: & PASCNT:
CHECK THESE TWO TAG LOCATIONS FOR TOTAL # OF ERRORS AND PASSES RESPECTIVELY.
LOADING 000200 AND RESTARTING WILL CLEAR THESE LOCATIONS.

6.3 END OF PASS ROUTINE
THIS TYPEOUT IS MENTIONED HERE FOR CONVENIENCE
IT IS IN THE FORM:

```
                END OF PASS TAPE Y
                1XXXXX = DEVICE

                WHERE Y IS THE TAPE LOADED

                WHERE 1XXXXX IS THE DEVICE'S BASE REGISTER ADDRESS

                TO INHIBIT THIS TYPEOUT - TURN TELETYPE OFF

        7.      RESTRICTIONS

        7.1     MULTIPLE DEVICES
                UP TO 16(10) DEVICES MAY BE TESTED. HOWEVER, THEY
                MUST HAVE CONTIGUOUS ADDRESSES AND VECTORS

                NOTE: IF ALL DEVICES UNDER TEST HAVE THE SAME INTERRUPT VECTOR
                YOU CAN CHANGE  "ZERO:    ADD   #10,BASEIV      ;NEXT BLOCK
                (VECTORS)" TO "ZERO:   ADD    #0,BASEIV";
                THEREBY THE VECTOR ADDRESSES WILL NOT BE
                UPDATED AFTER EACH PASS.

        7.2     DISQUALIFYING DEVICES WHEN RUNNING MULTIPLE DEVICES

                WHEN RUNNING MULTIPLE DEVICES    AN ACTIVE BIT IS SET
                FOR EACH DEVICE RUNNING UNDER TEST  IE. BIT 0 FOR
                DEVICE 0   ,BIT 15 FOR DEVICE 15
                TO DISQUALIFY DEVICES:

        7.2.1   IF DEVICE 0 IS TO BE DISQUALIFIED ,SIMPLY RESTART
                PROGRAM WITH SW00 =1 AND OMIT THE FIRST DEVICE.

        7.2.2   IF HOWEVER, DEVICES 1 THRU 15 OR ANY COMBINATION THEREOF
                ARE TO BE DISQUALIFIED....LOAD THE LOCATION OF ACTREG:
                OBSERVE THE ACTIVE BITS (ACTIVE =1, NONACTIVE = 0)
                AND DEPOSIT 0 WHERE THOSE DEVICES ARE TO BE DISQUALIFIED

                7.2.2.1 TO RESTART...TYPE 200G...
                        THE PROGRAM WILL CONTINUE WITH THE DEVICE IT WAS IN BEFORE HALTING.

                7.2.2.2 .......OR .....SET SW00=1 IN SWITCH REG (LOC. 176) AND TYPE 200G....
                        ANSWER THE QUESTION :1ST DEVICE : ETC.....
                         ....THE PROGRAM WILL CONTINUE WITH DEVICE 0

                7.2.2.3 IF ALL DEVICES ARE DISQUALIFIED BY MISTAKE   THE PROGRAM
                        WILL TYPEOUT   AN ERROR MESSAGE.......TYPE 200G.

        7.3     CABLE DELAYS
                NOTE: EXTERNAL LOOP BACK TESTS ONLY (MODEM CABLE WITH H315 CONNECTOR ON)

        7.3.1   TO PROVIDE SUFFICIENT DELAY FOR CLOCK SIGNAL OVER THE CABLE,
                LOCATION "HOLD:" MUST BE MODIFIED TO ACCOMODATE FOR FASTER MACHINES.
                        PRESENTLY "HOLD:" =20  IS SUFFICIENT TIME ON AN 11/21 MACHINE.

                BASICALLY DON'T TRY TO EXCEED 10K TO 12K RATE USING THE EIA DRIVERS

        7.4     TO USE THE "XOR" TESTER ,THE BRANCH AROUND THE "XOR"
```

CODE MUST BE PATCHED TO A "NOP". (SEE LISTINGS FOR DETAILS)

8. DEFAULT PARAMETERS:
1ST DEVICE: RECEIVER CONTROL REGISTER ADDRESS- RXCSR: 174300

VECTOR ADDRESS- DURIV: 330

ARE YOU RUNNING MULTIPLE DEVICES ?- NO MULTD: 0

LAST DEVICE: RECEIVER CONTROL REGISTER ADDRESS- LASTADD: 0

OF SYNC CHARS SELECTED - 2 SYNCNO: 377

IS SEC λMIT SWITCH E55-2 ON?- YES SEXMIT: 377

IS SEC REC SWITCH E55-3 ON?- YES SEREC: 377

IS OPT CLR ENABLE SWITCH E55-1 ON?- YES OPTCLR: 377

DO YOU HAVE THE EXTERNAL MODEM BYPASS JUMPER
CONNECTOR ON (H315)- YES JMRBY: 377

9. PROGRAM DESCRIPTION

*
.REM *

9.1 THIS PROGRAM PERFORMS THE OFFLINE LOGIC BIT BANGING
OF THE DEVICE
SEE LISTING FOR DETAILS

*
.REM *

10. FLOW CHARTS: RECEIVER FLOW,TRANSMITTER FLOW,TRANSMITTER & RECEIVER FLOW

11. CHANGE HISTORY

NOTE: HISTORY BEGINS WITH REV. C0

REV. C0: 1) ALLOW OPERATOR TO SPECIFY ADDRESS > 170000
2) INCREASE DELAY VALUE "HOLD" TO COMPENSATE
FOR 11/23 EXECUTION TIME (FROM 3777 TO 6200)

CNDUQA0
IN ORDER TO BE SPECIFIC TO SBC 11/21 PROCESSOR, REV C0
WAS MODIFIED TO INCLUDE CHANGES FOR PRIORITY 6 INSTEAD OF 7
AND DEFAULT CSR AND VECTOR ADDRESSES AND ALSO .INIT CALL
TO CNMAC2.SML INCLUDED AT THE END TO INITIALIZE 11/21
SPECIFIC VECTORS. THE DIAGNOSTIC WAS RENAMED TO CNDUQA0.

12. LISTINGS

*

```
 6598
 6599
 6600
 6611
 6625
 6626
 6627
 6688
 6689
 6894
 7022
 7023                                   .SBTTL  APT COMMUNICATIONS ROUTINE
  (1)
  (2)                                   ;;*****************************************************************
  (1)  000000  112767  000001  000236  $ATY1:  MOVB    #1,$FFLG          ;;TO REPORT FATAL ERROR
  (1)  000006  112767  000001  000226  $ATY3:  MOVB    #1,$MFLG          ;;TO TYPE A MESSAGE
  (1)  000014  000403                          BR      $ATYC
  (1)  000016  112767  000001  000220  $ATY4:  MOVB    #1,$FFLG          ;;TO ONLY REPORT FATAL ERROR
  (1)  000024                          $ATYC:
  (3)  000024  010046                          MOV     R0,-(SP)          ;;PUSH R0 ON STACK
  (3)  000026  010146                          MOV     R1,-(SP)          ;;PUSH R1 ON STACK
  (1)  000030  105767  000206                  TSTB    $MFLG             ;;SHOULD TYPE A MESSAGE?
  (1)  000034  001450                          BEQ     5$                ;;IF NOT:  BR
  (1)  000036  122767  000001  001502          CMPB    #APTENV,$ENV      ;;OPERATING UNDER APT?
  (1)  000044  001031                          BNE     3$                ;;IF NOT:  BR
  (1)  000046  132767  000100  001473          BITB    #APTSPOOL,$ENVM   ;;SHOULD SPOOL MESSAGES?
  (1)  000054  001425                          BEQ     3$                ;;IF NOT:  BR
  (1)  000056  017600  000004                  MOV     @4(SP),R0         ;;GET MESSAGE ADDR.
  (1)  000062  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
  (1)  000070  005767  001432          1$:     TST     $MSGTYPE          ;;SEE IF DONE W/ LAST XMISSION?
  (1)  000074  001375                          BNE     1$                ;;IF NOT:  WAIT
  (1)  000076  010067  001440                  MOV     R0,$MSGAD
  (1)                                   ;;PUT ADDR IN MAILBOX
  (1)  000102  105720                  2$:     TSTB    (R0)+             ;;FIND END OF MESSAGE
  (1)  000104  001376                          BNE     2$
  (1)  000106  166700  001430                  SUB     $MSGAD,R0         ;;SUB START OF MESSAGE
  (1)  000112  006200                          ASR     R0                ;;GET MESSAGE LNGTH IN WORDS
  (1)  000114  010067  001424                  MOV     R0,$MSGLGT        ;;PUT LENGTH IN MAILBOX
  (1)  000120  012767  000004  001400          MOV     #4,$MSGTYPE       ;;TELL APT TO TAKE MSG.
  (1)  000126  000413                          BR      5$
  (1)  000130  017667  000004  000016  3$:     MOV     @4(SP),4$         ;;PUT MSG ADDR IN JSR LINKAGE
  (1)  000136  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDRESS
  (3)  000144  016746  177626                  MOV     177776,-(SP)      ;;PUSH 177776 ON STACK
  (1)  000150  004767  012666                  JSR     PC,$TYPE          ;;CALL TYPE MACRO
  (1)  000154  000000                  4$:     .WORD   0
  (1)  000156                          5$:
  (1)  000156  105767  000062          10$:    TSTB    $FFLG             ;;SHOULD REPORT FATAL ERROR?
  (1)  000162  001416                          BEQ     12$               ;;IF NOT:  BR
  (1)  000164  005767  001356                  TST     $ENV              ;;RUNNING UNDER APT?
  (1)  000170  001413                          BEQ     12$               ;;IF NOT:  BR
  (1)  000172  005767  001330          11$:    TST     $MSGTYPE          ;;FINISHED LAST MESSAGE?
  (1)  000176  001375                          BNE     11$               ;;IF NOT:  WAIT
  (1)  000200  017667  000004  001322          MOV     @4(SP),$FATAL     ;;GET ERROR #
  (1)  000206  062766  000002  000004          ADD     #2,4(SP)                  ;;BUMP RETURN ADDR.
  (1)  000214  005267  001306                  INC     $MSGTYPE          ;;TELL APT TO TAKE ERROR
```

```
   (1)  000220  105067  000020          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
   (1)  000224  105067  000013                  CLRB    $LFLG           ;;CLEAR LOG FLAG
   (1)  000230  105067  000006                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
   (3)  000234  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
   (3)  000236  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
   (1)  000240  000207                          RTS     PC              ;;RETURN
   (1)  000242     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
   (1)  000243     000          $LFLG:  .BYTE   0
   (1)                                  ;;LOG FLAG
   (1)  000244     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
   (1)          000246                  .EVEN
   (1)          000200          APTSIZE=200
   (1)          000001          APTENV=001
   (1)          000100          APTSPOOL=100
   (1)          000040          APTCSUP=040
 7262           000001          $TN=1
 7388
 7392
 7427
 7444
 7457
 7469
 7482
```

```
7506
7572
7578
7714
7742
7775
7816
7847
7898
7946
7999
8014
8026
8128
```

```
                        .ENABLE ABS

                        ;CNDUQ-A0 DUV11 TAPE A
                        ;COPYRIGHT 1977,1980, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754

                                ;STARTING PROCEDURE
                                ;TYPE 200G
                                ;PROGRAM WILL TYPE "CNDUQ-A0 DUV11 TAPE A "
                                ;PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
                                ;AT THE END OF A PASS, PROGRAM WILL TYPE "END OF PASS TAPE A"
                                ;AND THEN RESUME TESTING

                        .SBTTL  BASIC DEFINITIONS

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

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ::STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
                        ;*****  THE FOLLOWING ODT START ADDRESS FOR SBC 11/21 IS ADDED
170000                  ODTST=  170000
                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

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

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
```

```
(2)  040000          SW14=   40000
(2)  020000          SW13=   20000
(2)  010000          SW12=   10000
(2)  004000          SW11=   4000
(2)  002000          SW10=   2000
(2)  001000          SW09=   1000
(2)  000400          SW08=   400
(2)  000200          SW07=   200
(2)  000100          SW06=   100
(2)  000040          SW05=   40
(2)  000020          SW04=   20
(2)  000010          SW03=   10
(2)  000004          SW02=   4
(2)  000002          SW01=   2
(2)  000001          SW00=   1
(2)                  .EQUIV  SW09,SW9
(2)                  .EQUIV  SW08,SW8
(2)                  .EQUIV  SW07,SW7
(2)                  .EQUIV  SW06,SW6
(2)                  .EQUIV  SW05,SW5
(2)                  .EQUIV  SW04,SW4
(2)                  .EQUIV  SW03,SW3
(2)                  .EQUIV  SW02,SW2
(2)                  .EQUIV  SW01,SW1
(2)                  .EQUIV  SW00,SW0
(2)
(2)                  ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(2)  100000          BIT15=  100000
(2)  040000          BIT14=  40000
(2)  020000          BIT13=  20000
(2)  010000          BIT12=  10000
(2)  004000          BIT11=  4000
(2)  002000          BIT10=  2000
(2)  001000          BIT09=  1000
(2)  000400          BIT08=  400
(2)  000200          BIT07=  200
(2)  000100          BIT06=  100
(2)  000040          BIT05=  40
(2)  000020          BIT04=  20
(2)  000010          BIT03=  10
(2)  000004          BIT02=  4
(2)  000002          BIT01=  2
(2)  000001          BIT00=  1
(2)                  .EQUIV  BIT09,BIT9
(2)                  .EQUIV  BIT08,BIT8
(2)                  .EQUIV  BIT07,BIT7
(2)                  .EQUIV  BIT06,BIT6
(2)                  .EQUIV  BIT05,BIT5
(2)                  .EQUIV  BIT04,BIT4
(2)                  .EQUIV  BIT03,BIT3
(2)                  .EQUIV  BIT02,BIT2
(2)                  .EQUIV  BIT01,BIT1
(2)                  .EQUIV  BIT00,BIT0
(2)
(2)                  ;*BASIC  'CPU'' TRAP VECTOR ADDRESSES
(2)  000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
```

```
   (2)          000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
   (2)          000014                  TBITVEC=14              ;;"T" BIT
   (2)          000014                  TRTVEC= 14              ;;TRACE TRAP
   (2)          000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
   (2)          000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
   (2)          000024                  PWRVEC= 24              ;;POWER FAIL
   (2)          000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
   (2)          000034                  TRAPVEC=34              ;;"TRAP" TRAP
   (2)          000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
   (2)          000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
   (2)                                  ;*****  THE FOLLOWING BREAK VECTOR AND LINE CLOCK VECTOR ARE INCLUDED
   (2)          000100                  LKVEC=  100             ;;LINE CLOCK VECTOR
   (2)          000140                  BRKVEC= 140             ;;BREAK VECTOR
   (2)          000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
```

```
(2)                                     ;STANDARD INTERRUPT VECTORS
(2)
(2)
(2)             000174                  .=174
(2)   000174    000000                  DISPREG:0
(2)   000176    000000                  SWREG:0
(2)             000200                  .=200
(2)   000200    000167  001746                JMP     .START          ;GO TO START OF PROGRAM
(2)
(2)
(2)
(2)             001100                  .=1100
(2)   001100    000000                  .WORD 0
(2)   001102    177570                  LIGHTS:177570
(2)
(2)
(2)
(2)                                     ;PROGRAM CONTROL PARAMETERS
(2)
(2)   001104    000000                  RETURN: 0
(2)   001106    000000                  NEXT:   0                       ;ADDRESS OF NEXT TEST TO BE EXECUTED
(2)   001110    000000                  LOCK:   0                       ;ADDRESS FOR LOCK ON CURRENT DATA
(2)   001112    000000                  PASCNT: 0               ;ADDRESS CONTAINING PASS COUNT
(2)   001114    000000                  ERRCNT: 0                       ;ERROR COUNT
(2)   001116    000000                  SAVSP:  0               ;STACK POINTER STORAGE
(2)
(2)                                     ;PROGRAM VARIABLES
(2)
(2)   001120    000020                  HOLD:   20              ;TEMPORARY STORAGE=DELAY TIME FOR CABLES
(2)   001122    000000                  SHIFT:  0               ;TEMPORARY STORAGE= # OF SHIFTS PER CHAR
(2)   001124    000000                  COUNT:  0               ;TEMPORARY STORAGE= # OF TIMES A CHAR WILL BE SENT
(2)   001126    000000                  SAVPC:  0                       ;PROGRAM COUNTER STORAGE
(2)   001130    000000                  HLD0:   0
(2)   001132    000000                  HLD1:   0
(2)   001134    000000                  HLD2:   0
(2)   001136    000000                  HLD3:   0
(2)   001140    000000                  HLD4:   0
(2)   001142    000000                  HLD5:   0
(2)   001144    000000                  HLD6:   0
(2)
```

```
(2)                                     ;PROGRAM CONVERSATIONAL PARAMETERS
(2)  001146    377              SYNCNO: .BYTE   377     ;# OF SYNC CHARS REQ'D FOR SYNC'ZATION
(2)  001147    377              SEXMIT: .BYTE   377     ;SEC XMIT JUMPER ''IN''
(2)  001150    377              SEREC:  .BYTE   377     ;SEC REC JUMPER ''IN''
(2)  001151    377              OPTCLR: .BYTE   377     ;OPTIONAL JUMPER CLR ''IN''
(2)  001152    000              MULTD:  .BYTE   0       ;NO MULTIPLE DEVICE FLAG
(2)  001153    377              JMRBY:  .BYTE   377     ;EXTERNAL MODEM BYPASS JUMPER ''IN''
(2)                             .EVEN
(2)
(2)                                     ;PROGRAM MULTIPLE DEVICE PARAMETERS
(2)  001154  000000             BASEADD:        0       ;PROG CONTROLLED 1ST DEVICE ADDR
(2)  001156  000000             KEEPADD:        0       ;SAVED 1ST DEVICE ADDR
(2)  001160  000000             LASTADD:        0       ;LAST DEVICE RXCSR ADDR
(2)  001162  000000             BASEIV:         0       ;PROG CONTROLLED IV
(2)  001164  000000             KEEPIV:         0       ;SAVED INTR VECTOR
(2)  001166  000000             ACTREG:         0       ;ACTIVE REGISTER ,,,MODIFY THIS
(2)                                                     ;LOCATION TO DISQUALIFY OR QUALIFY
(2)                                                     ;DEVICES (1= RUN,,,0= DON'T RUN)
(2)  001170  000000             ROTADD:         0       ;ROTATING POINTER FOR ACTREG..POINTS
(2)                                     ;TO DEVICE PRESENTLY UNDER TEST WHEN RUNNING MULTIPLE DEVICES
(2)
(2)                                     ;PROGRAM CONTROL FLAGS
(2)
(2)  001172    000              INIFLG: .BYTE   0               ;PROGRAM INITIALIZATION FLAG
(2)  001173    000              STFLG:  .BYTE   0               ;TEST START FLAG
(2)  001174    000              LOKFLG: .BYTE   0               ;LOCK ON CURRENT TEST FLAG
(2)          001176             .EVEN
(1)          001400             .=1400
(2)
(2)
```

```
(2)
(2)
(2)
(2)                             ;INSTRUCTION DEFINITIONS
(2)
(2)             005746          PUSH1SP=5746    ;DECREMENT PROCESSOR STACK 1 WORD =TST -(SP)
(2)             005726          POP1SP=5726     ;INCREMENT PROCESSOR STACK 1 WORD =TST (SP)+
(2)             010046          PUSHR0=10046    ;SAVE R0 ON STACK =MOV R0,-(SP)
(2)             012600          POPR0=12600     ;RESTORE R0 FROM STACK =MOV (SP)+,R0
(2)             024646          PUSH2SP=24646   ;DECREMENT STACK TWICE =CMP -(SP),-(SP)
(2)             022626          POP2SP=22626    ;INCREMENT STACK TWICE =CMP (SP)+,(SP)+
(2)                             ;REGISTER DEFINITIONS
(2)                                     ;RXCSR BIT DEFINITIONS
(2)             100000          DSC=BIT15               ;DATA SET CHANGE
(2)             040000          RING=BIT14              ;RING
(2)             020000          CTS=BIT13               ;CLR TO SEND
(2)             010000          CARDET=BIT12            ;CARRIER DETECT
(2)             004000          RECACT=BIT11            ;REC ACTIVE
(2)             002000          SRD=BIT10               ;SEC REC DATA
(2)             001000          DSR=BIT9                ;DATA SET RDY
(2)             000400          STPSYN=BIT8             ;STRIP SYNC
(2)             000200          RXDONE=BIT7             ;REC DONE
(2)             000100          RINTEN=BIT6             ;REC INTR ENABLE
(2)             000040          DSINTE=BIT5             ;DSC INTR ENABLE
(2)             000020          SYNSCH=BIT4             ;SYNC SEARCH
(2)             000010          STD=BIT3                ;SEC XMIT DATA
(2)             000004          RTS=BIT2                ;REQ TO SEND
(2)             000002          DTR=BIT1                ;DATA TERM RDY
(2)             000001          VOID=BIT0
(2)                                     ;RXDBUF BIT DEFINITIONS
(2)             100000          RXERR=BIT15             ;REC ERROR
(2)             040000          OVRRUN=BIT14            ;OVERRUN
(2)             020000          FRMERR=BIT13            ;FRAME ERROR
(2)             010000          PARER=BIT12             ;PARITY ERROR
(2)                                     ;PARCSR BIT DEFINITIONS
(2)             001000          PAREN=BIT9              ;PARITY ENABLE
(2)             000400          EVPAR=BIT8              ;EVEN PARITY SENSE
(2)                                     ;PARCSR WRD DEFINITIONS
(2)             030000          SYNINT=30000            ;SYNC EXTERNAL MODE
(2)             020000          SYNEXT=20000            ;SYNC INTERNAL MODE
(2)             000000          ISYMOD=0                ;ISOC MODE
(2)             000000          FIVE=0                  ;WORD LENGTH 5 BITS
(2)             002000          SIX=2000                ;WORD LENGTH 6 BITS
(2)             004000          SEVEN=4000              ;WORD LENGTH 7 BITS
(2)             006000          EIGHT=6000              ;WORD LENGTH 8 BITS
(2)             000000          NOPAR=0                 ;NO PARITY
(2)             001000          ODDPAR=1000             ;ODD PARITY
(2)             001400          EVEPAR=1400             ;EVEN PARITY
(2)                                     ;TXCSR BIT DEFINITIONS
(2)             100000          DNA=BIT15               ;DATA NOT AVAILABLE
(2)             040000          MTDATA=BIT14            ;MAINT DATA
(2)             020000          CLK=BIT13               ;CLK
(2)             002000          BITW=BIT10              ;BIT WINDOW
(2)             000400          MRESET=BIT8             ;MASTER RESET
(2)             000200          TXDONE=BIT7             ;XMIT DONE
(2)             000100          TXINTE=BIT6             ;XMIT INTR ENABLE
```

```
   (2)          000040                  DNAINTE=BIT5            ;DNA INTR ENAB
   (2)          000020                  SEND=BIT4               ;SEND
   (2)          000010                  HDXEN=BIT3              ;HDX/FDX
   (2)          000001                  BREAK=BIT0              ;BREAK
   (2)                                          ;TXCSR WRD DEFINITIONS
   (2)          000000                  USER=0                  ;USER MODE
   (2)          004000                  MINT=4000               ;MAINT INT MODE
   (2)          010000                  MEXT=10000              ;MAINT EXT MODE
   (2)          014000                  SYSTST=14000            ;SYSTEM TEST MODE
```

```
(2)                                 .SBTTL  COMMON TAGS
(2)
(3)                                 ;;*****************************************************************
(2)                                 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
(2)                                 ;*USED IN THE PROGRAM.
(2)
(2)     001400                          .=.
(2)  001400                     $CMTAG:                         ;;START OF COMMON TAGS
(2)  001400  000000                     .WORD   0
(2)  001402     000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
(2)  001403     000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
(2)  001404  000000             $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
(2)  001406  000000             $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
(2)  001410  000000             $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
(2)  001412  000000             $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
(2)  001414     000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
(2)  001415     001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
(2)  001416  000000             $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
(2)  001420  000000             $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
(2)  001422  000000             $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
(2)  001424  000000             $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
(2)  001426  000000             $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
(2)  001430  000000                     .WORD   0              ;;RESERVED--NOT TO BE USED
(2)  001432  000000                     .WORD   0
(2)  001434     000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
(2)  001435     000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
(2)  001436  000000                     .WORD   0
(2)  001440  177570             SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
(2)  001442  177570             DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
(2)  001444  177560             $TKS:   177560                  ;;TTY KBD STATUS
(2)  001446  177562             $TKB:   177562                  ;;TTY KBD BUFFER
(2)  001450  177564             $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
(2)  001452  177566             $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
(2)  001454     000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
(2)  001455     002             $FILLS: .BYTE   2
(2)                                                             ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
(2)  001456     012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
(2)  001457     000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
(2)  001460  000000             $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
(2)                                                             ;;WHICH  ($REG0) WAS OBTAINED
(4)  001462  000000             $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
(4)  001464  000000             $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
(4)  001466  000000             $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
(4)  001470  000000             $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
(4)  001472  000000             $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
(4)  001474  000000             $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
(4)  001476  000000             $TMP0:  .WORD   0               ;;USER DEFINED
(4)  001500  000000             $TMP1:  .WORD   0               ;;USER DEFINED
(4)  001502  000000             $TMP2:  .WORD   0               ;;USER DEFINED
(4)  001504  000000             $TMP3:  .WORD   0               ;;USER DEFINED
(4)  001506  000000             $TMP4:  .WORD   0               ;;USER DEFINED
(4)  001510  000000             $TMP5:  .WORD   0               ;;USER DEFINED
(2)  001512  000000             $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
(2)  001514  000000             $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
(2)  001516  177607  000377     $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
(2)  001522     077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
```

```
(2)  001523     015                 $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
(2)  001524  000012                 $LF:    .ASCIZ  <12>            ;;LINE FEED
(3)                                 ;;*****************************************************************
(3)                                 .SBTTL  APT MAILBOX-ETABLE
(3)
(4)                                 ;;*****************************************************************
(3)                                 .EVEN
(3)  001526                         $MAIL:                          ;;APT MAILBOX
(3)  001526  000000                 $MSGTY: .WORD   AMSGTY   ;;MESSAGE TYPE CODE
(3)  001530  000000                 $FATAL: .WORD   AFATAL   ;;FATAL ERROR NUMBER
(3)  001532  000000                 $TESTN: .WORD   ATESTN   ;;TEST NUMBER
(3)  001534  000000                 $PASS:  .WORD   APASS    ;;PASS COUNT
(3)  001536  000000                 $DEVCT: .WORD   ADEVCT   ;;DEVICE COUNT
(3)  001540  000000                 $UNIT:  .WORD   AUNIT    ;;I/O UNIT NUMBER
(3)  001542  000000                 $MSGAD: .WORD   AMSGAD   ;;MESSAGE ADDRESS
(3)  001544  000000                 $MSGLG: .WORD   AMSGLG   ;;MESSAGE LENGTH
(3)  001546                         $ETABLE:                        ;;APT ENVIRONMENT TABLE
(3)  001546     000                 $ENV:   .BYTE   AENV     ;;ENVIRONMENT BYTE
(3)  001547     000                 $ENVM:  .BYTE   AENVM
(3)                                 ;;ENVIRONMENT MODE BITS
(3)  00.550  000000                 $SWREG: .WORD   ASWREG   ;;APT SWITCH REGISTER
(3)  001552  000000                 $USWR:  .WORD   AUSWR    ;;USER SWITCHES
(3)  001554  000000                 $CPUOP: .WORD   ACPUOP   ;;CPU TYPE,OPTIONS
(3)                                 ;*                              BITS 15-11=CPU TYPE
(3)                                 ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
(3)                                 ;*                                      11/70=06,PDQ=07,Q=10
(3)                                 ;*                              BIT 10=REAL TIME CLOCK
(3)                                 ;*                              BIT  9=FLOATING POINT PROCESSOR
(3)                                 ;*                              BIT  8=MEMORY MANAGEMENT
(3)  001556     000                 $MAMS1: .BYTE   AMAMS1   ;;HIGH ADDRESS,M.S. BYTE
(3)  001557     000                 $MTYP1: .BYTE   AMTYP1   ;;MEM. TYPE,BLK#1
(3)                                 ;*                              MEM.TYPE BYTE   --  (HIGH BYTE)
(3)                                 ;*                                      900 NSEC CORE=001
(3)                                 ;*                                      300 NSEC BIPOLAR=002
(3)                                 ;*                                      500 NSEC MOS=003
(3)  001560  000000                 $MADR1: .WORD   AMADR1   ;;HIGH ADDRESS,BLK#1
(3)                                 ;*                                MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
(3)  001562     000                 $MAMS2: .BYTE   AMAMS2   ;;HIGH ADDRESS,M.S. BYTE
(3)  001563     000                 $MTYP2: .BYTE   AMTYP2   ;;MEM.TYPE,BLK#2
(3)  001564  000000                 $MADR2: .WORD   AMADR2   ;;MEM.LAST ADDRESS,BLK#2
(3)  001566     000                 $MAMS3: .BYTE   AMAMS3   ;;HIGH ADDRESS,M.S.BYTE
(3)  001567     000                 $MTYP3: .BYTE   AMTYP3   ;;MEM.TYPE,BLK#3
(3)  001570  000000                 $MADR3: .WORD   AMADR3   ;;MEM.LAST ADDRESS,BLK#3
(3)  001572     000                 $MAMS4: .BYTE   AMAMS4   ;;HIGH ADDRESS,M.S.BYTE
(3)  001573     000                 $MTYP4: .BYTE   AMTYP4   ;;MEM.TYPE,BLK#4
(3)  001574  000000                 $MADR4: .WORD   AMADR4   ;;MEM.LAST ADDRESS,BLK#4
(3)  001576  000000                 $VECT1: .WORD   AVECT1   ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
(3)  001600  000000                 $VECT2: .WORD   AVECT2   ;;INTERRUPT VECTOR#2BUS PRIORITY#2
(3)  001602  000000                 $BASE:  .WORD   ABASE
(3)                                 ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
(3)  001604  000000                 $DEVM:  .WORD   ADEVM    ;;DEVICE MAP
(3)  001606  000000                 $CDW1:  .WORD   ACDW1    ;;CONTROLLER DESCRIPTION WORD#1
(3)  001610  000000                 $CDW2:  .WORD   ACDW2    ;;CONTROLLER DESCRIPTION WORD#2
(3)  001612  000000                 $DDW0:  .WORD   ADDW0    ;;DEVICE DESCRIPTOR WORD#0
(3)  001614  000000                 $DDW1:  .WORD   ADDW1    ;;DEVICE DESCRIPTOR WORD#1
(3)  001616  000000                 $DDW2:  .WORD   ADDW2    ;;DEVICE DESCRIPTOR WORD#2
```

```
(3)  001620  000000                $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
(3)  001622  000000                $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
(3)  001624  000000                $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
(3)  001626  000000                $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
(3)  001630  000000                $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
(3)  001632  000000                $DDW8:  .WORD   ADDW8   ;;DEVICE DESCRIPTOR WORD#8
(3)  001634  000000                $DDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
(3)  001636  000000                $DDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
(3)  001640  000000                $DDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
(3)  001642  000000                $DDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
(3)  001644  000000                $DDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
(3)  001646  000000                $DDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
(3)  001650  000000                $DDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
(3)
(3)
(3)  001652                        $ETEND:
(3)
(4)
(4)
```

```
(4)
(4)
(4)
(4)                                 ;INSTRUCTION DEFINITIONS
(4)
(4)             005746              PUSH1SP=5746    ;DECREMENT PROCESSOR STACK 1 WORD =TST -(SP)
(4)             005726              POP1SP=5726     ;INCREMENT PROCESSOR STACK 1 WORD =TST (SP)+
(4)             010046              PUSHR0=10046    ;SAVE R0 ON STACK =MOV R0,-(SP)
(4)             012600              POPR0=12600     ;RESTORE R0 FROM STACK =MOV (SP)+,R0
(4)             024646              PUSH2SP=24646   ;DECREMENT STACK TWICE =CMP -(SP),-(SP)
(4)             022626              POP2SP=22626    ;INCREMENT STACK TWICE =CMP (SP)+,(SP)+
(4)                                 ;REGISTER DEFINITIONS
(4)                                         ;RXCSR BIT DEFINITIONS
(4)             100000              DSC=BIT15               ;DATA SET CHANGE
(4)             040000              RING=BIT14              ;RING
(4)             020000              CTS=BIT13               ;CLR TO SEND
(4)             010000              CARDET=BIT12            ;CARRIER DETECT
(4)             004000              RECACT=BIT11            ;REC ACTIVE
(4)             002000              SRD=BIT10               ;SEC REC DATA
(4)             001000              DSR=BIT9                ;DATA SET RDY
(4)             000400              STPSYN=BIT8             ;STRIP SYNC
(4)             000200              RXDONE=BIT7             ;REC DONE
(4)             000100              RINTEN=BIT6             ;REC INTR ENABLE
(4)             000040              DSINTE=BIT5             ;DSC INTR ENABLE
(4)             000020              SYNSCH=BIT4             ;SYNC SEARCH
(4)             000010              STD=BIT3                ;SEC XMIT DATA
(4)             000004              RTS=BIT2                ;REQ TO SEND
(4)             000002              DTR=BIT1                ;DATA TERM RDY
(4)             000001              VOID=BIT0
(4)                                         ;RXDBUF BIT DEFINITIONS
(4)             100000              RXERR=BIT15             ;REC ERROR
(4)             040000              OVRRUN=BIT14            ;OVERRUN
(4)             020000              FRMERR=BIT13            ;FRAME ERROR
(4)             010000              PARER=BIT12             ;PARITY ERROR
(4)                                         ;PARCSR BIT DEFINITIONS
(4)             001000              PAREN=BIT9              ;PARITY ENABLE
(4)             000400              EVPAR=BIT8              ;EVEN PARITY SENSE
(4)                                         ;PARCSR WRD DEFINITIONS
(4)             030000              SYNINT=30000            ;SYNC EXTERNAL MODE
(4)             020000              SYNEXT=20000            ;SYNC INTERNAL MODE
(4)             000000              ISYMOD=0                ;ISOC MODE
(4)             000000              FIVE=0                  ;WORD LENGTH 5 BITS
(4)             002000              SIX=2000                ;WORD LENGTH 6 BITS
(4)             004000              SEVEN=4000              ;WORD LENGTH 7 BITS
(4)             006000              EIGHT=6000              ;WORD LENGTH 8 BITS
(4)             000000              NOPAR=0                 ;NO PARITY
(4)             001000              ODDPAR=1000             ;ODD PARITY
(4)             001400              EVEPAR=1400             ;EVEN PARITY
(4)                                         ;TXCSR BIT DEFINITIONS
(4)             100000              DNA=BIT15               ;DATA NOT AVAILABLE
(4)             040000              MTDATA=BIT14            ;MAINT DATA
(4)             020000              CLK=BIT13               ;CLK
(4)             002000              BITW=BIT10              ;BIT WINDOW
(4)             000400              MRESET=BIT8             ;MASTER RESET
(4)             000200              TXDONE=BIT7             ;XMIT DONE
(4)             000100              TXINTE=BIT6             ;XMIT INTR ENABLE
```

```
   (4)          000040                  DNAINTE=BIT5            ;DNA INTR ENAB
   (4)          000020                  SEND=BIT4               ;SEND
   (4)          000010                  HDXEN=BIT3              ;HDX/FDX
   (4)          000001                  BREAK=BIT0              ;BREAK
   (4)                                          ;TXCSR WRD DEFINITIONS
   (4)          000000                  USER=0                  ;USER MODE
   (4)          004000                  MINT=4000               ;MAINT INT MODE
   (4)          010000                  MEXT=10000              ;MAINT EXT MODE
   (4)          014000                  SYSTST=14000            ;SYSTEM TEST MODE
```

```
    (2)                                 .SBTTL  ERROR POINTER TABLE
    (2)
    (2)                                 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
    (2)                                 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
    (2)                                 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
    (2)                                 ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
    (2)                                 ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
    (2)
    (2)                                 ;*      EM              ;;POINTS TO THE ERROR MESSAGE
    (2)                                 ;*      DH              ;;POINTS TO THE DATA HEADER
    (2)                                 ;*      DT              ;;POINTS TO THE DATA
    (2)                                 ;*      DF              ;;POINTS TO THE DATA FORMAT
    (2)
    (2)
    (2)  001652                         $ERRTB:
    (1)                                         ;ERROR TABLE
    (1)  001652  001762                         EM1     ;ERROR 1        REGISTER ERROR
    (1)  001654  002067                         DH1
    (1)  001656  002116                         DT1
    (1)  001660  002132                         DF1
    (1)  001662  002022                         EM2     ;ERROR 2        RECEIVER ERROR
    (1)  001664  002067                         DH1
    (1)  001666  002116                         DT1
    (1)  001670  002132                         DF1
    (1)  001672  002043                         EM3     ;ERROR 3        TRANSMITTER ERROR
    (1)  001674  002067                         DH1
    (1)  001676  002116                         DT1
    (1)  001700  002132                         DF1
    (1)  001702  001746                         EM4     ;ERROR 4        BIT ERROR (GENERAL)
    (1)  001704  000000                         0
    (1)  001706  002126                         DT4
    (1)  001710  002132                         DF1
    (1)
    (1)                                 ;DEFAULT DU ADDRESSES
    (1)  001712  174300                 RXCSR:  174300
    (1)  001714  174301                 HRXCSR: 174301
    (1)  001716  174302                 RXDBUF: 174302
    (1)  001720  174303                 HRXDBUF:        174303
    (1)  001722  174302                 PARCSR: 174302
    (1)  001724  174303                 HPARCSR:        174303
    (1)  001726  174304                 TXCSR:  174304
    (1)  001730  174305                 HTXCSR: 174305
    (1)  001732  174306                 TXDBUF: 174306
    (1)  001734  174307                 HTXDBUF:        174307
    (1)                                 ;DEFAULT DU VECTORS
    (1)  001736  000330                 DURIV:  330     ;REC INTR VECTOR
    (1)  001740  000332                 DURIS:  332     ;REC INTR STATUS
    (1)  001742  000334                 DUTIV:  334     ;XMIT INTR VECTOR
    (1)  001744  000336                 DUTIS:  336     ;XMIT INTR STATUS
    (1)                                 ;ERROR MESSAGES
    (1)  001746  020040  051105  047522 EM4:    .ASCIZ  /  ERROR PC /
    (1)  001754  020122  041520  000040
    (1)  001762  020040  047503  050115 EM1:    .ASCIZ /  COMPARISON ERROR ON REGISTERS/
    (1)  001770  051101  051511  047117
    (1)  001776  042440  051122  051117
    (1)  002004  047440  020116  042522
```

```
(1)  002012  044507  052123  051105
(1)  002020  000123
(1)  002022  020040  042522  042503  EM2:    .ASCIZ  /  RECEIVER ERROR/
(1)  002030  053111  051105  042440
(1)  002036  051122  051117     000
(1)  002043     040  052040  040522  EM3:    .ASCIZ  /  TRANSMITTER ERROR/
(1)  002050  051516  044515  052124
(1)  002056  051105  042440  051122
(1)  002064  051117     000
(1)                                  ;DATA HEADERS FOR ERROR MESSAGES
(1)  002067     105  051122  041520  DH1:    .ASCIZ  /ERRPC  WANTED  ACTUAL/
(1)  002074  020040  040527  052116
(1)  002102  042105  020040  041501
(1)  002110  052524  046101     000
(1)          002116                  .EVEN
(1)                                  ;DATA TABLES FOR ERROR MESSAGES
(1)  002116  001416  001130  001132  DT1:    .WORD   $ERRPC,HLD0,HLD1,0
(1)  002124  000000
(1)
(1)  002126  001416  000000          DT4:    .WORD   $ERRPC,0
(1)
(1)  002132     000     000     000  DF1:    .BYTE   0,0,0,0
(1)  002135     000
(1)                                  .EVEN
(2)                                  .SBTTL  ACT11 HOOKS
(2)
(3)                                  ;;*****************************************************************
(2)                                  ;HOOKS REQUIRED BY ACT11
(2)          002136                          $SVPC=.                 ;SAVE PC
(2)          000046                          .=46
(2)  000046  012644                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
(2)          000052                          .=52
(2)  000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
(2)          002136                          .=$SVPC                 ;; RESTORE PC
(2)                                  .SBTTL  APT PARAMETER BLOCK
(2)
(3)                                  ;;*****************************************************************
(2)                                  ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(3)                                  ;;*****************************************************************
(2)          002136                          .$X=.   ;;SAVE CURRENT LOCATION
(2)          000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(2)  000024  000200                          200     ;;FOR APT START UP
(2)          000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
(2)  000044  002136                          $APTHDR ;;POINT TO APT HEADER BLOCK
(2)          002136                          .=.$X   ;;RESET LOCATION COUNTER
(3)                                  ;;*****************************************************************
(2)                                  ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(2)                                  ;INTERFACE SPEC.
(2)
(2)  002136                          $APTHD:
(2)  002136  000000                  $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
(2)  002140  001526                  $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
(2)  002142  000010                  $TSTM:  .WORD   10      ;;RUN TIM OF LONGEST TEST
(2)  002144  000010                  $PASTM: .WORD   10      ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
(2)  002146  000000                  $UNITM: .WORD           ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
(2)  002150  000052                          .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
(1)
(2)
(2)                                     ;PROGRAM INITIALIZATION
(2)                                     ;LOCK OUT INTERRUPTS
(2)                                     ;SET UP PROCESSOR STACK
(2)                                     ;SET UP POWER FAIL VECTOR
(2)                                     ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
(2)                                     ;TYPE TITLE MESSAGE
(2)
(2)  002152                         .START:
(3)                                 .SBTTL  INITIALIZE THE COMMON TAGS
(3)                                 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
(3)  002152  012706  001400         MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
(3)  002156  005026                 CLR     (R6)+           ;;CLEAR MEMORY LOCATION
(3)  002160  022706  001440         CMP     #SWR,R6 ;;DONE?
(3)  002164  001374                 BNE     .-6             ;;LOOP BACK IF NO
(3)  002166  012706  001100         MOV     ##STACK,SP              ;;SETUP THE STACK POINTER
(3)                                 ;;INITIALIZE A FEW VECTORS
(3)  002172  012737  016272  000020 MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
(3)  002200  012737  000300  000022 MOV     #PR6,@#IOTVEC+2 ;;LEVEL 6
(3)  002206  012737  014160  000030 MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
(3)  002214  012737  000300  000032 MOV     #PR6,@#EMTVEC+2 ;;LEVEL 6
(3)                                 ;;BIT02
(3)  002222  012737  016626  000034 MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(3)  002230  012737  000300  000036 MOV     #PR6,@#TRAPVEC+2;LEVEL 6
(3)  002236  012737  014762  000024 MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
(3)  002244  012737  000300  000026 MOV     #PR6,@#PWRVEC+2 ;;LEVEL 6
(3)  002252  005067  177234         CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
(3)  002256  005067  177232         CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
(3)  002262  112767  000001  177125 MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
(3)  002270  012767  002270  177110 MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
(3)  002276  012767  002276  177104 MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
(4)                                 ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
(4)                                 ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
(4)  002304  013746  000004         MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
(4)  002310  012737  002344  000004 MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
(4)  002316  012767  177570  177114 MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
(4)  002324  012767  177570  177110 MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
(4)  002332  022777  177777  177100 CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
(4)  002340  001012                 BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
(4)                                                         ;;AND  THE HARDWARE SWR IS NOT = -1
(4)  002342  000403                 BR      65$             ;;BRANCH IF NO TIMEOUT
(4)  002344  012716  002352 64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
(4)  002350  000002                 RTI
(4)  002352  012767  000176  177060 65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
(4)  002360  012767  000174  177054         MOV     #DISPREG,DISPLAY
(4)  002366  012637  000004         66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
(3)
(4)  002372  005067  177136         CLR     $PASS           ;;CLEAR PASS COUNT
(4)  002376  132767  000200  177143 BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
(4)  002404  001403                 BEQ     67$             ;;YES,USE NON-APT SWITCH
(4)  002406  012767  001550  177024 MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
(4)  002414                         67$:
(2)  002414  012706  001100         MOV     #STACK,SP               ;SET STACK
(2)  002420  106427  000300         MTPS    #300                    ;LOCK INTERRUPTS
(2)  002424  012737  014762  000024 MOV     #.PFAIL,@#24            ;SET UP POWER FAIL VECTOR
```

```
(2)  002432  105067  176535                CLRB    STFLG                   ;CLEAR START FLAG
(2)  002436  005067  176450                CLR     PASCNT                  ;CLEAR PASS COUNT
(2)  002442  105067  176735                CLRB    $ERFLG                  ;CLEAR ERROR FLAG
(2)  002446  005067  176740                CLR     $ERTTL                  ;CLEAR ERROR COUNT
(2)  002452  005067  176740                CLR     $ERRPC                  ;CLEAR LAST ERROR POINTER
(2)  002456  012767  000001  176716        MOV     #1,$TSTNM               ;SET UP FOR TEST 1
(2)  002464  012767  002152  176412        MOV     #.START,RETURN          ;SET UP FOR POWER FAIL BEFORE
(2)                                                                        ;TES.ING STARTS
(2)  002472  013746  000006                MOV     @#6,-(SP)
(2)  002476  013746  000004                MOV     @#4,-(SP)
(2)  002502  012737  002516  000004        MOV     #1$,@#4
(2)  002510  005777  176724                TST     @SWR
(2)  002514  000407                        BR      2$
(2)  002516  012767  000176  176714  1$:   MOV     #SWREG,SWR
(2)  002524  012767  000174  176710        MOV     #DISPREG,DISPLAY
(2)  002532  022626                        CMP     (SP)+,(SP)+
(2)  002534  012637  000004          2$:   MOV     (SP)+,@#4
(2)  002540  012637  000006                MOV     (SP)+,@#6
(2)  002544  022767  000176  176666        CMP     #SWREG,SWR
(2)  002552  001007                        BNE     3$
(2)  002554  005737  000042                TST     @#42                    ;CHECK FOR CHAIN
(2)  002560  001402                        BEQ     33$
(2)  002562  000167  000522                JMP     .BEGIN
(2)  002566  004767  010154          33$:  JSR     PC,CNTLU
(2)  002572  105767  176374          3$:   TSTB    INIFLG                  ;HAS INITIALIZATION BEEN PERFORMED
(2)  002576  001004                        BNE     ONCE
(2)  002600  104401  015122                TYPE    ,MTITLE                 ;TYPE TITLE MESSAGE
(2)  002604  105167  176362                COMB    INIFLG                  ;IF NOT SET FLAG AND DO
(2)  002610  105767  176732          ONCE: TSTB    $ENV            ;APT CONTROL?
(2)  002614  001410                        BEQ     11$             ;BR IF NO
(2)  002616  032767  000001  176726        BIT     #1,$USWR        ;EXTENAL JUMPER ON?
(2)  002624  001002                        BNE     12$             ;NO
(2)  002626  105067  176321                CLRB    JMRBY           ;CLEAR FLAG
(2)  002632  000167  000452          12$:  JMP     .BEGIN          ;GO DO IT
(2)  002636  032777  000001  176574  11$:  BIT     #SW00,@SWR      ;RESELECT VECTOR & CONTROL REG?
(2)  002644  001002                        BNE     1$
(2)  002646  000167  000436                JMP     .BEGIN
(2)  002652  012700  000300          1$:   MOV     #300,R0 ;RESTORE VECTOR AREA TO TRAPCATCHER
(2)  002656  012701  000302                MOV     #302,R1 ;START AT LOCATION 300
(2)  002662  012702  000004                MOV     #4,R2
(2)  002666  010110                  2$:   MOV     R1,(R0)
(2)  002670  005011                        CLR     (R1)
(2)  002672  060200                        ADD     R2,R0
(2)  002674  060201                        ADD     R2,R1
(2)  002676  022701  001000                CMP     #1000,R1        ;END AT LOCATION 776
(2)  002702  002771                        BLT     2$
(2)  002704  104406                        INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  002706  015171                        MREGAD  ;MESSAGE
(2)  002710  104410                        PARAM   ;CONVERT STRING
(2)  002712  174000                        174000  ;LOW LIMIT
(2)  002714  177776                        177776  ;HIGH LIMIT
(2)  002716  017122                        DUBASE  ;STORE AT THIS LOCATION
(2)  002720     001                  .BYTE  1       ;MASK
(2)  002721     001                  .BYTE  1       ;HOW MANY TIMES + 2
(1)  002722  016767  014174  176226        MOV     DUBASE,KEEPADD  ;SAVE
(1)  002730  004767  014034                JSR     PC,DUADDR
```

```
(1)  002734  016767  176216  176212          MOV     KEEPADD,BASEADD ;RESTORE FOR ROTATION
(2)  002742  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  002744  015156                          MVECTO  ;MESSAGE
(2)  002746  104410                          PARAM   ;CONVERT STRING
(2)  002750  000300                          300     ;LOW LIMIT
(2)  002752  000776                          776     ;HIGH LIMIT
(2)  002754  001736                          DURIV   ;STORE AT THIS LOCATION
(2)  002756     001                  .BYTE   1       ;MASK
(2)  002757     004                  .BYTE   4       ;HOW MANY TIMES + 2
(1)  002760  016767  176752  176176          MOV     DURIV,KEEPIV    ;SAVE
(1)  002766  016767  176744  176166          MOV     DURIV,BASEIV    ;SET UP FOR ROTATION
(2)  002774  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  002776  015221                          MMULT   ;MESSAGE
(2)  003000  104414                          SETFLG  ;SET FLAG BASED UPON INPUT STRING
(2)  003002  001152                          MULTD   ;THIS FLAG
(1)  003004  105767  176142                  TSTB    MULTD   ;ARE THERE MULTIPLE DEVICES
(1)                                                  ;ON THE SYSTEM ?
(1)  003010  100406                          BMI     BBB     ;YES,ASK NEXT QUESTION
(1)  003012  005067  176150                  CLR     ACTREG
(1)  003016  005067  176146                  CLR     ROTADD
(1)  003022  000167  000140                  JMP     OUTMUL  ;JUMP AROUND NEXT QUESTION
(1)  003026                          BBB:
(2)  003026  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003030  015250                          MLASTD  ;MESSAGE
(2)  003032  104410                          PARAM   ;CONVERT STRING
(2)  003034  174000                          174000  ;LOW LIMIT
(2)  003036  177776                          177776  ;HIGH LIMIT
(2)  003040  001160                          LASTADD ;STORE AT THIS LOCATION
(2)  003042     001                  .BYTE   1       ;MASK
(2)  003043     001                  .BYTE   1       ;HOW MANY TIMES + 2
(1)                                  ;THE FOLLOWING ROUTINE SETS UP ACTREG FOR THE FIRST TIME
(1)  003044  012767  000001  176116  1$:     MOV     #1,ROTADD       ;SET UP POINTER
(1)  003052  005067  176110                  CLR     ACTREG  ;CLR ACTIVE REGISTER
(1)  003056  056767  176106  176102  2$:     BIS     ROTADD,ACTREG   ;MAKE THIS DEVICE ACTIVE
(1)  003064  000241                          CLC
(1)  003066  006167  176076                  ROL     ROTADD  ;SET UP POINTER
(1)  003072  103421                          BCS     3$      ;ARE YOU OUT OF RANGE ?
(1)  003074  062767  000010  176052          ADD     #10,BASEADD     ;SET UP BASE ADDRESS
(1)  003102  026767  176052  176044          CMP     LASTADD,BASEADD ;IS THIS THE LAST DEVICE ?
(1)  003110  101362                          BHI     2$      ;NO DO IT AGAIN
(1)  003112  056767  176052  176046          BIS     ROTADD,ACTREG   ;THIS ASSUMES THAT THERE ARE AT
(1)                                                  ;LEAST TWO DEVICES WHEN YOU ANSWER YES TO
(1)                                                  ;MULTIPLE DEVICE QUESTION
(1)  003120  012767  000001  176042  4$:     MOV     #1,ROTADD       ;SET UP FOR LATER USE IN END OF PASS ROUTINE
(1)  003126  016767  176024  176020          MOV     KEEPADD,BASEADD ;DITTO
(1)  003134  000414                          BR      OUTMUL  ;CONTINUE QUESTIONS
(1)  003136  016767  176014  176010  3$:     MOV     KEEPADD,BASEADD ;RESTORE
(2)  003144  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003146  015344                          MRANGE  ;MESSAGE
(2)  003150  104410                          PARAM   ;CONVERT STRING
(2)  003152  174000                          174000  ;LOW LIMIT
(2)  003154  177776                          177776  ;HIGH LIMIT
(2)  003156  001160                          LASTADD ;STORE AT THIS LOCATION
(2)  003160     001                  .BYTE   1       ;MASK
(2)  003161     001                  .BYTE   1       ;HOW MANY TIMES + 2
(1)  003162  000167  177656                  JMP     1$      ;DO IT AGAIN
```

```
(1)  003166  012767  000300  013570  OUTMUL: MOV     #300,DUPRT
(1)  003174  004767  013514                  JSR     PC,DULEV
(2)                                          ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
(2)                                          ;BUFFER TO THE CHARACTERS "1" AND "2".
(2)                                          ;IF THE CHARACTER IS "1" CLEAR THE FLAG
(2)                                          ;IF THE CHARACTER IS "2" SET THE FLAG
(2)  003200                          AAA:
(2)  003200  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003202  015562                          MSYNC   ;MESSAGE
(2)  003204  122767  000061  012712  3$:     CMPB    #'1,INBUF       ;IS IT "1" ?
(2)  003212  001003                          BNE     1$
(2)  003214  105067  175726                  CLRB    SYNCNO  ;000
(2)  003220  000412                          BR      4$
(2)  003222  122767  000062  012674  1$:     CMPB    #'2,INBUF       ;IS IT "2" ?
(2)  003230  001004                          BNE     2$
(2)  003232  112767  177777  175706          MOVB    #-1,SYNCNO      ;377
(2)  003240  000402                          BR      4$
(2)  003242  104407                  2$:     INSTER  ;RETRY
(2)  003244  000757                          BR      3$
(2)  003246  000240                  4$:     NOP
(2)  003250  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003252  015630                          MWIRE6  ;MESSAGE
(2)  003254  104414                          SETFLG  ;SET FLAG BASED UPON INPUT STRING
(2)  003256  001147                          SEXMIT  ;THIS FLAG
(2)  003260  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003262  015701                          MWIRE5  ;MESSAGE
(2)  003264  104414                          SETFLG  ;SET FLAG BASED UPON INPUT STRING
(2)  003266  001150                          SEREC   ;THIS FLAG
(2)  003270  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003272  015751                          MWIRE4  ;MESSAGE
(2)  003274  104414                          SETFLG  ;SET FLAG BASED UPON INPUT STRING
(2)  003276  001151                          OPTCLR  ;THIS FLAG
(2)  003300  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003302  016030                          MEXTJ   ;MESSAGE
(2)  003304  104414                          SETFLG  ;SET FLAG BASED UPON INPUT STRING
(2)  003306  001153                          JMRBY   ;THIS FLAG
(2)
(2)                                          ;TEST START AND RESTART
(2)
(2)  003310  012706  001100          .BEGIN: MOV     #STACK,SP               ;SET UP STACK
(2)  003314  106427  000300                  MTPS    #300                    ;LOCK OUT INTERRUPTS
(2)  003320  032777  000002  176112          BIT     #SW01,@SWR              ;IF SW01=1, GET STARTING PC
(2)  003326  001413                          BEQ     3$
(3)  003330  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(3)  003332  015514                          MTSTPC  ;MESSAGE
(3)  003334  104410                          PARAM   ;CONVERT STRING
(3)  003336  003374                          TST1    ;LOW LIMIT
(3)  003340  017500                          17500   ;HIGH LIMIT
(3)  003342  001402                          $TSTNM  ;STORE AT THIS LOCATION
(3)  003344     001                  .BYTE   1       ;MASK
(3)  003345     001                  .BYTE   1       ;HOW MANY TIMES * 2
(2)  003346  016767  176030  175530          MOV     $TSTNM,RETURN
(2)  003354  000403                          BR      4$
(2)  003356  012767  003374  175520  3$:     MOV     #TST1,RETURN            ;START AT TEST 1
(2)  003364  104401  015510          4$:     TYPE    ,MR                     ;TYPE R
(2)  003370  000177  175510                  JMP     @RETURN                 ;START TESTING
```

```
 8166
 8167
 8168                                           ;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
  (1)                                           ;;
  (5)                                   ;;**********************************************************************
  (4)  003374  000004                   TST1:   SCOPE
  (4)
  (1)  003376  012737  017362  000004           MOV     #TRPREG,@#4     ;SETUP TRAPCATCHER
  (1)  003404  016737  174670  000006           MOV     PR6,@#6 ;
  (1)  003412  105277  176274                   INCB    @RXCSR          ;TEST THIS REG
  (1)  003416  000401                           BR      .+4             ;IF OK JMP AROUND ERROR
  (1)  003420  104004                           ERROR   4                       ;CHECK DEVICE REG ADDRESSES
  (1)  003422  105277  176266                   INCB    @HRXCSR ;TEST UPPER BYTE THIS REGISTER
  (1)  003426  000401                           BR      .+4             ;IF OK JMP AROUND ERROR
  (1)  003430  104004                           ERROR   4                       ;CHECK DEVICE REG ADDRESSES
  (1)  003432  012737  000006  000004           MOV     #6,@#4          ;RESTORE TRAPCATCHER
  (1)  003440  012737  000000  000006           MOV     #0,@#6          ;
  (1)
 8169  003446  012767  006200  175444           MOV     #6200,HOLD      ;SET LONGER DELAY FOR TEST.     ;;REV. C0
 8170                                           ;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
  (1)                                           ;;
  (5)                                   ;;**********************************************************************
  (4)  003454  000004                   TST2:   SCOPE
  (4)
  (1)  003456  012737  017362  000004           MOV     #TRPREG,@#4     ;SETUP TRAPCATCHER
  (1)  003464  016737  174610  000006           MOV     PR6,@#6 ;
  (1)  003472  105277  176220                   INCB    @RXDBUF         ;TEST THIS REG
  (1)  003476  000401                           BR      .+4             ;IF OK JMP AROUND ERROR
  (1)  003500  104004                           ERROR   4                       ;CHECK DEVICE REG ADDRESSES
  (1)  003502  105277  176212                   INCB    @HRXDBUF        ;TEST UPPER BYTE THIS REGISTER
  (1)  003506  000401                           BR      .+4             ;IF OK JMP AROUND ERROR
  (1)  003510  104004                           ERROR   4                       ;CHECK DEVICE REG ADDRESSES
  (1)  003512  012737  000006  000004           MOV     #6,@#4          ;RESTORE TRAPCATCHER
  (1)  003520  012737  000000  000006           MOV     #0,@#6          ;
  (1)
 8171                                           ;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
  (1)                                           ;;
  (5)                                   ;;**********************************************************************
  (4)  003526  000004                   TST3:   SCOPE
  (4)
  (1)  003530  012737  017362  000004           MOV     #TRPREG,@#4     ;SETUP TRAPCATCHER
  (1)  003536  016737  174536  000006           MOV     PR6,@#6 ;
  (1)  003544  105277  176152                   INCB    @PARCSR         ;TEST THIS REG
  (1)  003550  000401                           BR      .+4             ;IF OK JMP AROUND ERROR
  (1)  003552  104004                           ERROR   4                       ;CHECK DEVICE REG ADDRESSES
  (1)  003554  105277  176144                   INCB    @HPARCSR        ;TEST UPPER BYTE THIS REGISTER
  (1)  003560  000401                           BR      .+4             ;IF OK JMP AROUND ERROR
  (1)  003562  104004                           ERROR   4                       ;CHECK DEVICE REG ADDRESSES
  (1)  003564  012737  000006  000004           MOV     #6,@#4          ;RESTORE TRAPCATCHER
  (1)  003572  012737  000000  000006           MOV     #0,@#6          ;
  (1)
 8172                                           ;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
  (1)                                           ;;
  (5)                                   ;;**********************************************************************
  (4)  003600  000004                   TST4:   SCOPE
  (4)
```

```
    (1)  003602  012737  017362  000004          MOV     #TRPREG,@#4     ;SETUP TRAPCATCHER
    (1)  003610  016737  174464  000006          MOV     PR6,@#6 ;
    (1)  003616  105277  176104                  INCB    @TXCSR          ;TEST THIS REG
    (1)  003622  000401                          BR      .+4             ;IF OK JMP AROUND ERROR
    (1)  003624  104004                          ERROR   4                       ;CHECK DEVICE REG ADDRESSES
    (1)  003626  105277  176076                  INCB    @HTXCSR ;TEST UPPER BYTE THIS REGISTER
    (1)  003632  000401                          BR      .+4             ;IF OK JMP AROUND ERROR
    (1)  003634  104004                          ERROR   4                       ;CHECK DEVICE REG ADDRESSES
    (1)  003636  012737  000006  000004          MOV     #6,@#4          ;RESTORE TRAPCATCHER
    (1)  003644  012737  000000  000006          MOV     #0,@#6          ;
    (1)
  8173                                           ;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
    (1)                                          ;;
    (5)                                  ;;*****************************************************************
    (4)  003652  000004                  TST5:   SCOPE
    (4)
    (1)  003654  012737  017362  000004          MOV     #TRPREG,@#4     ;SETUP TRAPCATCHER
    (1)  003662  016737  174412  000006          MOV     PR6,@#6 ;
    (1)  003670  105277  176036                  INCB    @TXDBUF         ;TEST THIS REG
    (1)  003674  000401                          BR      .+4             ;IF OK JMP AROUND ERROR
    (1)  003676  104004                          ERROR   4                       ;CHECK DEVICE REG ADDRESSES
    (1)  003700  105277  176030                  INCB    @HTXDBUF        ;TEST UPPER BYTE THIS REGISTER
    (1)  003704  000401                          BR      .+4             ;IF OK JMP AROUND ERROR
    (1)  003706  104004                          ERROR   4                       ;CHECK DEVICE REG ADDRESSES
    (1)  003710  012737  000006  000004          MOV     #6,@#4          ;RESTORE TRAPCATCHER
    (1)  003716  012737  000000  000006          MOV     #0,@#6          ;
    (1)
  8174                                           ;;BUS DRIVER TEST
  8175                                           ;;
  8176                                 ;;*****************************************************************
    (3)  003724  000004                  TST6:   SCOPE
    (3)
  8177  003726  022777  000000  175776          CMP     #0,@TXDBUF
  8178  003734  001401                          BEQ     .+4
  8179  003736  104004                          ERROR   4       ;READING TXDBUF SHOULD BE ALL ZERO'S
  8180                                           ;;THIS TEST PERFORMS MASTER RESET TESTING &
    (1)                                          ;;TESTING OF READ/WRITE BIT DTR
    (1)                                          ;;
    (5)                                  ;;*****************************************************************
    (4)  003740  000004                  TST7:   SCOPE
    (4)
    (1)  003742  052777  000002  175742          BIS     #DTR,@RXCSR     ;SET THIS BIT
    (1)  003750  032777  000002  175734          BIT     #DTR,@RXCSR     ;TEST THIS BIT
    (1)  003756  001001                          BNE     .+4     ;BR IF "1"
    (1)  003760  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
    (1)  003762  042777  000002  175722          BIC     #DTR,@RXCSR     ;CLR THIS BIT
    (1)  003770  032777  000002  175714          BIT     #DTR,@RXCSR     ;TEST THIS BIT
    (1)  003776  001401                          BEQ     .+4             ;BR IF "0"
    (1)  004000  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
    (1)                                          ;NOW SET THIS BIT
    (1)  004002  052777  000002  175702          BIS     #DTR,@RXCSR
    (2)  004010  052777  000400  175710          BIS     #MRESET,@TXCSR  ;MASTER RESET
    (1)                                          ;;CHECK EXISTANCE OF OPTIONAL CLEAR JUMPER
    (1)                                          ;;
    (1)  004016  105767  175127                  TSTB    OPTCLR          ;TEST FLAG
    (1)  004022  100006                          BPL     1$              ;OPTIONAL CLR JUMPER IS NOT IN
```

```
   (1)  004024  032777  000002  175660          BIT     #DTR,@RXCSR     ;TEST THIS BIT
   (1)  004032  001401                          BEQ     .+4             ;BR IF "0"
   (1)  004034  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
   (1)  004036  000405                          BR      2$              ;JMP AROUND
   (1)  004040  032777  000002  175644  1$:     BIT     #DTR,@RXCSR     ;TEST THIS BIT
   (1)  004046  001001                          BNE     .+4     ;BR IF "1"
   (1)  004050  104004                          ERROR   4                       ;CHECK OUT OPTIONAL CLR JUMPER
   (1)  004052  000240                  2$:     NOP
   (1)
  8181                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
   (1)                                          ;;TESTING OF READ/WRITE BIT RTS
   (1)                                          ;;
   (5)                                  ;;**************************************************************
   (4)  004054  000004                  TST10:  SCOPE
   (4)
   (1)  004056  052777  000004  175626          BIS     #RTS,@RXCSR     ;SET THIS BIT
   (1)  004064  032777  000004  175620          BIT     #RTS,@RXCSR     ;TEST THIS BIT
   (1)  004072  001001                          BNE     .+4     ;BR IF "1"
   (1)  004074  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
   (1)  004076  042777  000004  175606          BIC     #RTS,@RXCSR     ;CLR THIS BIT
   (1)  004104  032777  000004  175600          BIT     #RTS,@RXCSR     ;TEST THIS BIT
   (1)  004112  001401                          BEQ     .+4             ;BR IF "0"
   (1)  004114  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
   (1)                                          ;NOW SET THIS BIT
   (1)  004116  052777  000004  175566          BIS     #RTS,@RXCSR
   (2)  004124  052777  000400  175574          BIS     #MRESET,@TXCSR  ;MASTER RESET
   (1)                                          ;;CHECK EXISTANCE OF OPTIONAL CLEAR JUMPER
   (1)                                          ;;
   (1)  004132  105767  175013                  TSTB    OPTCLR          ;TEST FLAG
   (1)  004136  100006                          BPL     1$              ;OPTIONAL CLR JUMPER IS NOT IN
   (1)  004140  032777  000004  175544          BIT     #RTS,@RXCSR     ;TEST THIS BIT
   (1)  004146  001401                          BEQ     .+4             ;BR IF "0"
   (1)  004150  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
   (1)  004152  C00405                          BR      2$              ;JMP AROUND
   (1)  004154  032777  000004  175530  1$:     BIT     #RTS,@RXCSR     ;TEST THIS BIT
   (1)  004162  001001                          BNE     .+4     ;BR IF "1"
   (1)  004164  104004                          ERROR   4                       ;CHECK OUT OPTIONAL CLR JUMPER
   (1)  004166  000240                  2$:     NOP
   (1)
  8182                                          ;WAIT FOR CABLE DELAYS
   (1)                                          ;**************************************
   (1)                                          ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
   (1)                                          ;**************************************
   (1)  004170  016702  174724                  MOV     HOLD,R2 ;SET DELAY TIME
   (1)  004174  005302                          DEC     R2
   (1)  004176  001376                          BNE     .-2     ;WAIT THIS TIME
   (1)                                          ;OK NOW FALL THRU AND CONTINUE TESTING.....
   (1)                                          ;EXIT STAGE LEFT....CHINNNG!
  8183                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
   (1)                                          ;;TESTING OF READ/WRITE BIT STD
   (1)                                          ;;
   (5)                                  ;;**************************************************************
   (4)  004200  000004                  TST11:  SCOPE
   (4)
   (1)  004202  052777  000010  175502          BIS     #STD,@RXCSR     ;SET THIS BIT
   (1)  004210  032777  000010  175474          BIT     #STD,@RXCSR     ;TEST THIS BIT
```

```
(1)  004216  001001                          BNE     .+4     ;BR IF "1"
(1)  004220  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
(1)  004222  042777  000010  175462          BIC     #STD,@RXCSR     ;CLR THIS BIT
(1)  004230  032777  000010  175454          BIT     #STD,@RXCSR     ;TEST THIS BIT
(1)  004236  001401                          BEQ     .+4             ;BR IF "0"
(1)  004240  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
(1)                                          ;NOW SET THIS BIT
(1)  004242  052777  000010  175442          BIS     #STD,@RXCSR
(2)  004250  052777  000400  175450          BIS     #MRESET,@TXCSR  ;MASTER RESET
(1)                                          ;;CHECK EXISTANCE OF OPTIONAL CLEAR JUMPER
(1)                                          ;;
(1)  004256  105767  174667                  TSTB    OPTCLR          ;TEST FLAG
(1)  004262  100006                          BPL     1$              ;OPTIONAL CLR JUMPER IS NOT IN
(1)  004264  032777  000010  175420          BIT     #STD,@RXCSR     ;TEST THIS BIT
(1)  004272  001401                          BEQ     .+4             ;BR IF "0"
(1)  004274  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
(1)  004276  000405                          BR      2$              ;JMP AROUND
(1)  004300  032777  000010  175404  1$:     BIT     #STD,@RXCSR     ;TEST THIS BIT
(1)  004306  001001                          BNE     .+4     ;BR IF "1"
(1)  004310  104004                          ERROR   4                       ;CHECK OUT OPTIONAL CLR JUMPER
(1)  004312  000240                  2$:     NOP
(1)
8184                                         ;;THIS TEST PERFORMS MASTER RESET TESTING &
(1)                                          ;;TESTING OF READ/WRITE BIT SYNSCH
(1)                                          ;;
(5)                                  ;;**************************************************************************
(4)  004314  000004                  TST12:  SCOPE
(4)
(1)  004316  052777  000020  175366          BIS     #SYNSCH,@RXCSR  ;SET THIS BIT
(1)  004324  032777  000020  175360          BIT     #SYNSCH,@RXCSR  ;TEST THIS BIT
(1)  004332  001001                          BNE     .+4     ;BR IF "1"
(1)  004334  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
(1)  004336  042777  000020  175346          BIC     #SYNSCH,@RXCSR  ;CLR THIS BIT
(1)  004344  032777  000020  175340          BIT     #SYNSCH,@RXCSR  ;TEST THIS BIT
(1)  004352  001401                          BEQ     .+4             ;BR IF "0"
(1)  004354  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
(1)                                          ;NOW SET THIS BIT
(1)  004356  052777  000020  175326          BIS     #SYNSCH,@RXCSR
(2)  004364  052777  000400  175334          BIS     #MRESET,@TXCSR  ;MASTER RESET
(1)  004372  032777  000020  175312          BIT     #SYNSCH,@RXCSR  ;TEST THIS BIT
(1)  004400  001401                          BEQ     .+4             ;BR IF "0"
(1)  004402  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
(1)
8185                                         ;;THIS TEST PERFORMS MASTER RESET TESTING &
(1)                                          ;;TESTING OF READ/WRITE BIT DSINTE
(1)                                          ;;
(5)                                  ;;**************************************************************************
(4)  004404  000004                  TST13:  SCOPE
(4)
(1)  004406  052777  000040  175276          BIS     #DSINTE,@RXCSR  ;SET THIS BIT
(1)  004414  032777  000040  175270          BIT     #DSINTE,@RXCSR  ;TEST THIS BIT
(1)  004422  001001                          BNE     .+4     ;BR IF "1"
(1)  004424  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
(1)  004426  042777  000040  175256          BIC     #DSINTE,@RXCSR  ;CLR THIS BIT
(1)  004434  032777  000040  175250          BIT     #DSINTE,@RXCSR  ;TEST THIS BIT
(1)  004442  001401                          BEQ     .+4             ;BR IF "0"
```

```
  (1)  004444  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
  (1)                                          ;NOW SET THIS BIT
  (1)  004446  052777  000040  175236          BIS     #DSINTE,@RXCSR
  (2)  004454  052777  000400  175244          BIS     #MRESET,@TXCSR  ;MASTER RESET
  (1)  004462  032777  000040  175222          BIT     #DSINTE,@RXCSR  ;TEST THIS BIT
  (1)  004470  001401                          BEQ     .+4             ;BR IF "0"
  (1)  004472  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
  (1)
 8186                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
  (1)                                          ;;TESTING OF READ/WRITE BIT RINTEN
  (1)                                          ;;
  (5)                                      ;;**********************************************************************
  (4)  004474  000004                      TST14:  SCOPE
  (4)
  (1)  004476  052777  000100  175206          BIS     #RINTEN,@RXCSR  ;SET THIS BIT
  (1)  004504  032777  000100  175200          BIT     #RINTEN,@RXCSR  ;TEST THIS BIT
  (1)  004512  001001                          BNE     .+4     ;BR IF "1"
  (1)  004514  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
  (1)  004516  042777  000100  175166          BIC     #RINTEN,@RXCSR  ;CLR THIS BIT
  (1)  004524  032777  000100  175160          BIT     #RINTEN,@RXCSR  ;TEST THIS BIT
  (1)  004532  001401                          BEQ     .+4             ;BR IF "0"
  (1)  004534  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
  (1)                                          ;NOW SET THIS BIT
  (1)  004536  052777  000100  175146          BIS     #RINTEN,@RXCSR
  (2)  004544  052777  000400  175154          BIS     #MRESET,@TXCSR  ;MASTER RESET
  (1)  004552  032777  000100  175132          BIT     #RINTEN,@RXCSR  ;TEST THIS BIT
  (1)  004560  001401                          BEQ     .+4             ;BR IF "0"
  (1)  004562  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
  (1)
 8187                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
  (1)                                          ;;TESTING OF READ/WRITE BIT STPSYN
  (1)                                          ;;
  (5)                                      ;;**********************************************************************
  (4)  004564  000004                      TST15:  SCOPE
  (4)
  (1)  004566  052777  000400  175116          BIS     #STPSYN,@RXCSR  ;SET THIS BIT
  (1)  004574  032777  000400  175110          BIT     #STPSYN,@RXCSR  ;TEST THIS BIT
  (1)  004602  001001                          BNE     .+4     ;BR IF "1"
  (1)  004604  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
  (1)  004606  042777  000400  175076          BIC     #STPSYN,@RXCSR  ;CLR THIS BIT
  (1)  004614  032777  000400  175070          BIT     #STPSYN,@RXCSR  ;TEST THIS BIT
  (1)  004622  001401                          BEQ     .+4             ;BR IF "0"
  (1)  004624  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
  (1)                                          ;NOW SET THIS BIT
  (1)  004626  052777  000400  175056          BIS     #STPSYN,@RXCSR
  (2)  004634  052777  000400  175064          BIS     #MRESET,@TXCSR  ;MASTER RESET
  (1)  004642  032777  000400  175042          BIT     #STPSYN,@RXCSR  ;TEST THIS BIT
  (1)  004650  001401                          BEQ     .+4             ;BR IF "0"
  (1)  004652  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
  (1)
 8188                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
  (1)                                          ;;TESTING OF READ/WRITE BIT BREAK
  (1)                                          ;;
  (5)                                      ;;**********************************************************************
  (4)  004654  000004                      TST16:  SCOPE
  (4)
```

```
(1)  004656  052777  000001  175042          BIS     #BREAK,@TXCSR   ;SET THIS BIT
(1)  004664  032777  000001  175034          BIT     #BREAK,@TXCSR   ;TEST THIS BIT
(1)  004672  001001                          BNE     .+4     ;BR IF "1"
(1)  004674  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
(1)  004676  042777  000001  175022          BIC     #BREAK,@TXCSR   ;CLR THIS BIT
(1)  004704  032777  000001  175014          BIT     #BREAK,@TXCSR   ;TEST THIS BIT
(1)  004712  001401                          BEQ     .+4             ;BR IF "0"
(1)  004714  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
(1)                                          ;NOW SET THIS BIT
(1)  004716  052777  000001  175002          BIS     #BREAK,@TXCSR
(2)  004724  052777  000400  174774          BIS     #MRESET,@TXCSR  ;MASTER RESET
(1)  004732  032777  000001  174766          BIT     #BREAK,@TXCSR   ;TEST THIS BIT
(1)  004740  001401                          BEQ     .+4             ;BR IF "0"
(1)  004742  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
(1)
8189                                         ;;THIS TEST PERFORMS MASTER RESET TESTING &
(1)                                          ;;TESTING OF READ/WRITE BIT HDXEN
(1)                                          ;;
(5)                                     ;;***************************************************************
(4)  004744  000004                     TST17:  SCOPE
(4)
(1)  004746  052777  000010  174752          BIS     #HDXEN,@TXCSR   ;SET THIS BIT
(1)  004754  032777  000010  174744          BIT     #HDXEN,@TXCSR   ;TEST THIS BIT
(1)  004762  001001                          BNE     .+4     ;BR IF "1"
(1)  004764  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
(1)  004766  042777  000010  174732          BIC     #HDXEN,@TXCSR   ;CLR THIS BIT
(1)  004774  032777  000010  174724          BIT     #HDXEN,@TXCSR   ;TEST THIS BIT
(1)  005002  001401                          BEQ     .+4             ;BR IF "0"
(1)  005004  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
(1)                                          ;NOW SET THIS BIT
(1)  005006  052777  000010  174712          BIS     #HDXEN,@TXCSR
(2)  005014  052777  000400  174704          BIS     #MRESET,@TXCSR  ;MASTER RESET
(1)  005022  032777  000010  174676          BIT     #HDXEN,@TXCSR   ;TEST THIS BIT
(1)  005030  001401                          BEQ     .+4             ;BR IF "0"
(1)  005032  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
(1)
8190                                         ;;THIS TEST PERFORMS MASTER RESET TESTING &
(1)                                          ;;TESTING OF READ/WRITE BIT SEND
(1)                                          ;;
(5)                                     ;;***************************************************************
(4)  005034  000004                     TST20:  SCOPE
(4)
(1)  005036  052777  000020  174662          BIS     #SEND,@TXCSR    ;SET THIS BIT
(1)  005044  032777  000020  174654          BIT     #SEND,@TXCSR    ;TEST THIS BIT
(1)  005052  001001                          BNE     .+4     ;BR IF "1"
(1)  005054  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
(1)  005056  042777  000020  174642          BIC     #SEND,@TXCSR    ;CLR THIS BIT
(1)  005064  032777  000020  174634          BIT     #SEND,@TXCSR    ;TEST THIS BIT
(1)  005072  001401                          BEQ     .+4             ;BR IF "0"
(1)  005074  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
(1)                                          ;NOW SET THIS BIT
(1)  005076  052777  000020  174622          BIS     #SEND,@TXCSR
(2)  005104  052777  000400  174614          BIS     #MRESET,@TXCSR  ;MASTER RESET
(1)  005112  032777  000020  174606          BIT     #SEND,@TXCSR    ;TEST THIS BIT
(1)  005120  001401                          BEQ     .+4             ;BR IF "0"
(1)  005122  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
```

```
  (1)
8191                                    ;;THIS TEST PERFORMS MASTER RESET TESTING &
 (1)                                    ;;TESTING OF READ/WRITE BIT DNAINTE
 (1)                                    ;;
 (5)                                ;;**********************************************************************
 (4)  005124  000004                TST21:  SCOPE
 (4)
 (1)  005126  052777  000040  174572        BIS     #DNAINTE,@TXCSR ;SET THIS BIT
 (1)  005134  032777  000040  174564        BIT     #DNAINTE,@TXCSR ;TEST THIS BIT
 (1)  005142  001001                        BNE     .+4     ;BR IF "1"
 (1)  005144  104004                        ERROR   4                       ;THIS BIT SHOULD BE SET
 (1)  005146  042777  000040  174552        BIC     #DNAINTE,@TXCSR ;CLR THIS BIT
 (1)  005154  032777  000040  174544        BIT     #DNAINTE,@TXCSR ;TEST THIS BIT
 (1)  005162  001401                        BEQ     .+4             ;BR IF "0"
 (1)  005164  104004                        ERROR   4                       ;THIS BIT SHOULD BE CLR
 (1)                                        ;NOW SET THIS BIT
 (1)  005166  052777  000040  174532        BIS     #DNAINTE,@TXCSR
 (2)  005174  052777  000400  174524        BIS     #MRESET,@TXCSR  ;MASTER RESET
 (1)  005202  032777  000040  174516        BIT     #DNAINTE,@TXCSR ;TEST THIS BIT
 (1)  005210  001401                        BEQ     .+4             ;BR IF "0"
 (1)  005212  104004                        ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
 (1)
8192                                    ;;THIS TEST PERFORMS MASTER RESET TESTING &
 (1)                                    ;;TESTING OF READ/WRITE BIT TXINTE
 (1)                                    ;;
 (5)                                ;;**********************************************************************
 (4)  005214  000004                TST22:  SCOPE
 (4)
 (1)  005216  052777  000100  174502        BIS     #TXINTE,@TXCSR  ;SET THIS BIT
 (1)  005224  032777  000100  174474        BIT     #TXINTE,@TXCSR  ;TEST THIS BIT
 (1)  005232  001001                        BNE     .+4     ;BR IF "1"
 (1)  005234  104004                        ERROR   4                       ;THIS BIT SHOULD BE SET
 (1)  005236  042777  000100  174462        BIC     #TXINTE,@TXCSR  ;CLR THIS BIT
 (1)  005244  032777  000100  174454        BIT     #TXINTE,@TXCSR  ;TEST THIS BIT
 (1)  005252  001401                        BEQ     .+4             ;BR IF "0"
 (1)  005254  104004                        ERROR   4                       ;THIS BIT SHOULD BE CLR
 (1)                                        ;NOW SET THIS BIT
 (1)  005256  052777  000100  174442        BIS     #TXINTE,@TXCSR
 (2)  005264  052777  000400  174434        BIS     #MRESET,@TXCSR  ;MASTER RESET
 (1)  005272  032777  000100  174426        BIT     #TXINTE,@TXCSR  ;TEST THIS BIT
 (1)  005300  001401                        BEQ     .+4             ;BR IF "0"
 (1)  005302  104004                        ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
 (1)
8193                                    ;;TEST MAINT MODE BIT 0
8194                                    ;;
8195                                    ;;THIS TEST PERFORMS MASTER RESET TESTING &
 (1)                                    ;;TESTING OF READ/WRITE BIT BIT11
 (1)                                    ;;
 (5)                                ;;**********************************************************************
 (4)  005304  000004                TST23:  SCOPE
 (4)
 (1)  005306  052777  004000  174412        BIS     #BIT11,@TXCSR   ;SET THIS BIT
 (1)  005314  032777  004000  174404        BIT     #BIT11,@TXCSR   ;TEST THIS BIT
 (1)  005322  001001                        BNE     .+4     ;BR IF "1"
 (1)  005324  104004                        ERROR   4                       ;THIS BIT SHOULD BE SET
 (1)  005326  042777  004000  174372        BIC     #BIT11,@TXCSR   ;CLR THIS BIT
```

```
    (1)  005334  032777  004000  174364          BIT     #BIT11,@TXCSR   ;TEST THIS BIT
    (1)  005342  001401                          BEQ     .+4             ;BR IF "0"
    (1)  005344  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
    (1)                                          ;NOW SET THIS BIT
    (1)  005346  052777  004000  174352          BIS     #BIT11,@TXCSR
    (2)  005354  052777  000400  174344          BIS     #MRESET,@TXCSR  ;MASTER RESET
    (1)  005362  032777  004000  174336          BIT     #BIT11,@TXCSR   ;TEST THIS BIT
    (1)  005370  001401                          BEQ     .+4             ;BR IF "0"
    (1)  005372  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
    (1)
   8196                                          ;;TEST MAINT MODE BIT 1
   8197                                          ;;
   8198                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
    (1)                                          ;;TESTING OF READ/WRITE BIT BIT12
    (1)                                          ;;
    (5)                                  ;;**************************************************************************
    (4)  005374  000004                  TST24:  SCOPE
    (4)
    (1)  005376  052777  010000  174322          BIS     #BIT12,@TXCSR   ;SET THIS BIT
    (1)  005404  032777  010000  174314          BIT     #BIT12,@TXCSR   ;TEST THIS BIT
    (1)  005412  001001                          BNE     .+4     ;BR IF "1"
    (1)  005414  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
    (1)  005416  042777  010000  174302          BIC     #BIT12,@TXCSR   ;CLR THIS BIT
    (1)  005424  032777  010000  174274          BIT     #BIT12,@TXCSR   ;TEST THIS BIT
    (1)  005432  001401                          BEQ     .+4             ;BR IF "0"
    (1)  005434  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
    (1)                                          ;NOW SET THIS BIT
    (1)  005436  052777  010000  174262          BIS     #BIT12,@TXCSR
    (2)  005444  052777  000400  174254          BIS     #MRESET,@TXCSR  ;MASTER RESET
    (1)  005452  032777  010000  174246          BIT     #BIT12,@TXCSR   ;TEST THIS BIT
    (1)  005460  001401                          BEQ     .+4             ;BR IF "0"
    (1)  005462  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
    (1)
   8199                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
    (1)                                          ;;TESTING OF READ/WRITE BIT CLK
    (1)                                          ;;
    (5)                                  ;;**************************************************************************
    (4)  005464  000004                  TST25:  SCOPE
    (4)
    (1)  005466  052777  020000  174232          BIS     #CLK,@TXCSR     ;SET THIS BIT
    (1)  005474  032777  020000  174224          BIT     #CLK,@TXCSR     ;TEST THIS BIT
    (1)  005502  001001                          BNE     .+4     ;BR IF "1"
    (1)  005504  104004                          ERROR   4                       ;THIS BIT SHOULD BE SET
    (1)  005506  042777  020000  174212          BIC     #CLK,@TXCSR     ;CLR THIS BIT
    (1)  005514  032777  020000  174204          BIT     #CLK,@TXCSR     ;TEST THIS BIT
    (1)  005522  001401                          BEQ     .+4             ;BR IF "0"
    (1)  005524  104004                          ERROR   4                       ;THIS BIT SHOULD BE CLR
    (1)                                          ;NOW SET THIS BIT
    (1)  005526  052777  020000  174172          BIS     #CLK,@TXCSR
    (2)  005534  052777  000400  174164          BIS     #MRESET,@TXCSR  ;MASTER RESET
    (1)  005542  032777  020000  174156          BIT     #CLK,@TXCSR     ;TEST THIS BIT
    (1)  005550  001401                          BEQ     .+4             ;BR IF "0"
    (1)  005552  104004                          ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
    (1)
   8200                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
    (1)                                          ;;TESTING OF READ/WRITE BIT MTDATA
```

```
(1)                                     ;;
(5)                                     ;;*****************************************************************
(4)  005554  000004                     TST26:  SCOPE
(4)
(1)  005556  052777  040000  174142             BIS     #MTDATA,@TXCSR  ;SET THIS BIT
(1)  005564  032777  040000  174134             BIT     #MTDATA,@TXCSR  ;TEST THIS BIT
(1)  005572  001001                             BNE     .+4     ;BR IF "1"
(1)  005574  104004                             ERROR   4                       ;THIS BIT SHOULD BE SET
(1)  005576  042777  040000  174122             BIC     #MTDATA,@TXCSR  ;CLR THIS BIT
(1)  005604  032777  040000  174114             BIT     #MTDATA,@TXCSR  ;TEST THIS BIT
(1)  005612  001401                             BEQ     .+4             ;BR IF "0"
(1)  005614  104004                             ERROR   4                       ;THIS BIT SHOULD BE CLR
(1)                                             ;NOW SET THIS BIT
(1)  005616  052777  040000  174102             BIS     #MTDATA,@TXCSR
(2)  005624  052777  000400  174074             BIS     #MRESET,@TXCSR  ;MASTER RESET
(1)  005632  032777  040000  174066             BIT     #MTDATA,@TXCSR  ;TEST THIS BIT
(1)  005640  001401                             BEQ     .+4             ;BR IF "0"
(1)  005642  104004                             ERROR   4                       ;CHECK OUT MASTER RESET LOGIC
(1)
8201                                    ;;THIS TEST VERIFYS THAT INIT (RESET) CLEARS BITS IN THE
8202                                    ;;RXCSR & TXCSR
8203                                    ;;
8204                                    ;;*****************************************************************
(3)  005644  000004                     TST27:  SCOPE
(3)
8205  005646  012777  177777  174036             MOV     #177777,@RXCSR  ;SET ALL POSSIBLE BITS
8206  005654  012777  177777  174044             MOV     #177777,@TXCSR  ;DITTO
8207  005662  000005                             RESET
8208  005664  106427  000300                     MTPS    #300    ;RESTORE NON INTERRUPT STATUS
8209  005670  017701  174016                     MOV     @RXCSR,R1       ;SAVE
8210  005674  017702  174026                     MOV     @TXCSR,R2       ;SAVE
8211  005700  105767  173245                     TSTB    OPTCLR  ;IS THE OPTIONAL CLR JUMPER ON ?
8212  005704  100402                             BMI     1$      ;YES
8213  005706  042701  000016                     BIC     #16,R1  ;CLR THE NON RESETABLE BITS
8214  005712  042701  073000          1$:        BIC     #073000,R1      ;CLR ALL NON-CLEARABLE BITS
8215  005716  005701                             TST     R1      ;ARE THEY ALL 0 ?
8216  005720  001401                             BEQ     .+4
8217  005722  104004                             ERROR   4       ;ALL SPECIFIED BITS SHOULD BE CLEAR
8218  005724  042702  002200                     BIC     #002200,R2      ;CLEAR ALL NON-CLEARABLE BITS
8219  005730  005702                             TST     R2      ;ARE THEY ALL 0 ?
8220  005732  001401                             BEQ     .+4
8221  005734  104004                             ERROR   4       ;ALL SPECIFIED BITS SHOULD BE CLEAR
8222                                            ;WAIT FOR CABLE DELAYS
(1)                                             ;*************************************
(1)                                             ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
(1)                                             ;*************************************
(1)  005736  016702  173156                     MOV     HOLD,R2 ;SET DELAY TIME
(1)  005742  005302                             DEC     R2
(1)  005744  001376                             BNE     .-2     ;WAIT THIS TIME
(1)                                             ;OK NOW FALL THRU AND CONTINUE TESTING.....
(1)                                             ;EXIT STAGE LEFT....CHINNNG!
8223
8224                                            ;;THIS TEST PERFORMS MASTER RESET TESTING &
(1)                                             ;;TESTING OF WRITE ONLY BIT MRESET
(1)                                             ;;
(5)                                     ;;*****************************************************************
```

```
  (4)  005746  000004                    TST30:  SCOPE
  (4)
  (1)  005750  052777  000400  173750            BIS     #MRESET,@TXCSR  ;TRY TO SET THIS BIT
  (1)  005756  032777  000400  173742            BIT     #MRESET,@TXCSR  ;TEST THIS BIT
  (1)  005764  001401                            BEQ     .+4             ;BR IF "0"
  (1)  005766  104004                            ERROR   4                       ;THIS BIT SHOULD NOT BE SET
  (2)  005770  052777  000400  173730            BIS     #MRESET,@TXCSR  ;MASTER RESET
  (1)  005776  032777  000400  173722            BIT     #MRESET,@TXCSR  ;TEST THIS BIT
  (1)  006004  001401                            BEQ     .+4             ;BR IF "0"
  (1)  006006  104004                            ERROR   4                       ;THIS BIT SHOULD NOT BE SET
  (1)                                                                    ;CHECK MASTER RESET LOGIC
  (1)
 8225                                    ;;THIS TEST VERIFYS THAT THE RXCSR & TXCSR CAN BE BYTE ADDRESSED (DATOB)
 8226                                    ;;
 8227                                    ;;*****************************************************************
  (3)  006010  000004                    TST31:  SCOPE
  (3)
 8228  006012  052777  000400  173706            BIS     #MRESET,@TXCSR  ;MASTER RESET
 8229  006020  105767  173125                    TSTB    OPTCLR  ;IS THE OPTIONAL CLR JUMPER ON ?
 8230  006024  100405                            BMI     1$      ;YES
 8231  006026  012777  000000  173656            MOV     #0,@RXCSR       ;CLR OUT NON RESETABLE BITS
 8232  006034  005777  173652                    TST     @RXCSR  ;CLR OUT DSC BY READING RXCSR
 8233  006040  152777  000001  173646  1$:       BISB    #BIT0,@HRXCSR   ;SET STRIP SYNC  UPPER BYTE
 8234  006046  017701  173640                    MOV     @RXCSR,R1       ;SAVE RXCSR
 8235  006052  022701  000400                    CMP     #400,R1 ;TEST RXCSR
 8236  006056  001401                            BEQ     .+4
 8237  006060  104004                            ERROR   4       ;ONLY STRIP SYNC SHOULD BE SET
 8238  006062  105077  173624                    CLRB    @RXCSR  ;CLR LOWER BYTE
 8239  006066  017701  173620                    MOV     @RXCSR,R1       ;SAVE RXCSR
 8240  006072  022701  000400                    CMP     #400,R1 ;TEST RXCSR
 8241  006076  001401                            BEQ     .+4
 8242  006100  104004                            ERROR   4       ;ONLY STRIP SYNC SHOULD BE SET
 8243  006102  052777  000400  173616            BIS     #MRESET,@TXCSR  ;MASTER RESET
 8244  006110  152777  000040  173612            BISB    #BIT5,@HTXCSR   ;SET MAINT CLK  UPPER BYTE
 8245  006116  017701  173604                    MOV     @TXCSR,R1       ;SAVE TXCSR
 8246  006122  042701  002000                    BIC     #BITW,R1        ;CLR BIT WINDOW (DEPENDENT
 8247                                                                   ;ON H315 CONNECTOR EXISTANCE)
 8248  006126  022701  020200                    CMP     #20200,R1       ;TEST TXCSR
 8249  006132  001401                            BEQ     .+4
 8250  006134  104004                            ERROR   4       ;ONLY MAINT CLK BIT & TXDONE SHOULD BE SET
 8251  006136  105077  173564                    CLRB    @TXCSR  ;CLR LOWER BYTE
 8252  006142  017701  173560                    MOV     @TXCSR,R1       ;SAVE TXCSR
 8253  006146  042701  002000                    BIC     #BITW,R1        ;CLR BIT WINDOW (DITTO)
 8254  006152  022701  020200                    CMP     #20200,R1       ;TEST TXCSR
 8255  006156  001401                            BEQ     .+4
 8256  006160  104004                            ERROR   4       ;ONLY MAINT CLK BIT & TXDONE SHOULD BE SET
 8257                                            ;;THIS TEST PERFORMS MASTER RESET TESTING &
 8258                                            ;;TESTING OF READ ONLY BIT BITW
 8259                                            ;;MAINT INTERNAL
 8260                                            ;;
 8261                                    ;;*****************************************************************
  (3)  006162  000004                    TST32:  SCOPE
  (3)
 8262  006164  012777  044001  173534            MOV     #MINT!MTDATA!BREAK,@TXCSR       ;SET MAINT INT.,BREAK,
 8263                                                                                   ;&MTDATA
 8264  006172  032777  002000  173526            BIT     #BITW,@TXCSR    ;TEST BITW
```

```
                                        BNE     .+4
                                        ERROR   4               ;BIT WINDOW SHOULD BE SET
                                        BIC     #MTDATA,@TXCSR
                                        MOV     @#1132,R2
                                1$:     DEC     R2
                                        BNE     1$
                                        BIT     #BITW,@TXCSR
                                        BEQ     .+4
                                        ERROR   4               ;BIT SHOULD BE CLR
                                        ;NOW SET THE MTDATA
                                        BIS     #MTDATA,@TXCSR
                                        BIS     #MRESET,@TXCSR  ;MASTER RESET
                                        BIS     #MINT!BREAK,@TXCSR
                                        MOV     @#1132,R2
                                2$:     DEC     R2
                                        BNE     2$
                                        BIT     #BITW,@TXCSR
                                        BEQ     .+4
                                        ERROR   4       ;BITW SHOULD BE CLR BY MASTER RESET
                                        ;;THIS TEST PERFORMS MASTER RESET TESTING &
                                        ;;TESTING OF READ ONLY BIT BITW
                                        ;;MAINT EXTERNAL
                                        ;;
                                ;;**************************************************************
  (3)   006300  000004          TST33:  SCOPE
  (3)
                                        ;TEST TO SEE IF EXTERNAL MODEM BYPASS CONNECTOR
                                        ;IS ON (H315)....IF "NO" JUMP AROUND TEST
        006302  105767  172645          TSTB    JMRBY
        006306  100036                  BPL     1$      ;IT IS NOT ON
        006310  012777  050001  173410  MOV     #MEXT!MTDATA!BREAK,@TXCSR       ;SET MAINT EXT.,BREAK,
                                                                                ;&MTDATA
        006316  032777  002000  173402  BIT     #BITW,@TXCSR    ;TEST BITW
        006324  001001                  BNE     .+4
        006326  104004                  ERROR   4               ;BIT WINDOW SHOULD BE SET
        006330  042777  040000  173370  BIC     #MTDATA,@TXCSR
        006336  032777  002000  173362  BIT     #BITW,@TXCSR
        006344  001401                  BEQ     .+4
        006346  104004                  ERROR   4               ;BIT SHOULD BE CLR
                                        ;NOW SET THE MTDATA
        006350  052777  040000  173350  BIS     #MTDATA,@TXCSR
        006356  052777  000400  173342  BIS     #MRESET,@TXCSR  ;MASTER RESET
        006364  052777  010001  173334  BIS     #MEXT!BREAK,@TXCSR
        006372  032777  002000  173326  BIT     #BITW,@TXCSR
        006400  001401                  BEQ     .+4
        006402  104004                  ERROR   4       ;BITW SHOULD BE CLR BY MASTER RESET
        006404                  1$:
                                        ;;THIS TEST PERFORMS MASTER RESET TESTING &
                                        ;;TESTING OF READ ONLY BIT RXDONE
  (1)                                   ;;
  (1)
  (5)                           ;;**************************************************************
  (4)   006404  000004          TST34:  SCOPE
  (4)
  (2)   006406  052777  000400  173312  BIS     #MRESET,@TXCSR  ;MASTER RESET
```

```
  (1)  006414  032777  000200  173270          BIT     #RXDONE,@RXCSR  ;TEST THIS BIT
  (1)  006422  001401                          BEQ     .+4             ;BR IF "0"
  (1)  006424  104004                          ERROR   4                       ;CHECK MASTER RESET LOGIC
  (1)                                                  ;OR SHORT ON THIS BIT
  (1)
 8313                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
  (1)                                          ;;TESTING OF READ ONLY BIT RECACT
  (1)                                          ;;
  (5)                                  ;;*************************************************************
  (4)  006426  000004                  TST35:  SCOPE
  (4)
  (2)  006430  052777  000400  173270          BIS     #MRESET,@TXCSR  ;MASTER RESET
  (1)  006436  032777  004000  173246          BIT     #RECACT,@RXCSR  ;TEST THIS BIT
  (1)  006444  001401                          BEQ     .+4             ;BR IF "0"
  (1)  006446  104004                          ERROR   4                       ;CHECK MASTER RESET LOGIC
  (1)                                                  ;OR SHORT ON THIS BIT
  (1)
 8314                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
  (1)                                          ;;TESTING OF READ ONLY BIT DSC
  (1)                                          ;;
  (5)                                  ;;*************************************************************
  (4)  006450  000004                  TST36:  SCOPE
  (4)
  (2)  006452  052777  000400  173246          BIS     #MRESET,@TXCSR  ;MASTER RESET
  (1)  006460  032777  100000  173224          BIT     #DSC,@RXCSR     ;TEST THIS BIT
  (1)  006466  001401                          BEQ     .+4             ;BR IF "0"
  (1)  006470  104004                          ERROR   4                       ;CHECK MASTER RESET LOGIC
  (1)                                                  ;OR SHORT ON THIS BIT
  (1)
 8315                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
 8316                                          ;;TESTING OF READ ONLY BIT TXDONE
 8317                                          ;;
 8318                                  ;;*************************************************************
  (3)  006472  000004                  TST37:  SCOPE
  (3)
 8319  006474  052777  000400  173224          BIS     #MRESET,@TXCSR  ;MASTER RESET
 8320  006502  032777  000200  173216          BIT     #TXDONE,@TXCSR  ;TEST THIS BIT
 8321  006510  001001                          BNE     .+4                     ;BR IF "1"
 8322  006512  104004                          ERROR   4               ;CHECK MASTER RESET LOGIC
 8323                                                  ;OR SHORT ON THIS BIT
 8324                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
  (1)                                          ;;TESTING OF READ ONLY BIT DNA
  (1)                                          ;;
  (5)                                  ;;*************************************************************
  (4)  006514  000004                  TST40:  SCOPE
  (4)
  (2)  006516  052777  000400  173202          BIS     #MRESET,@TXCSR  ;MASTER RESET
  (1)  006524  032777  100000  173174          BIT     #DNA,@TXCSR     ;TEST THIS BIT
  (1)  006532  001401                          BEQ     .+4             ;BR IF "0"
  (1)  006534  104004                          ERROR   4                       ;CHECK MASTER RESET LOGIC
  (1)                                                  ;OR SHORT ON THIS BIT
  (1)
 8325                                          ;;THIS TEST PERFORMS MASTER RESET TESTING &
 8326                                          ;;TESTING OF READ  ONLY WORD RECEIVE DATA
 8327                                          ;;
 8328                                  ;;*************************************************************
```

```
   (3)  006536  000004                    TST41:  SCOPE
   (3)
  8329  006540  052777  000400  173160            BIS     #MRESET,@TXCSR  ;MASTER RESET
  8330  006546  016703  173144                    MOV     RXDBUF,R3       ;FOR ERROR MESSAGE
  8331  006552  012700  000377                    MOV     #377,R0         ;EXPECTED
  8332  006556  017701  173134                    MOV     @RXDBUF,R1      ;ACTUAL
  8333  006562  120001                            CMPB    R0,R1
  8334  006564  001401                            BEQ     .+4             ;BR IF "0"
  8335  006566  104002                            ERROR   2               ;REC DATA SHOULD BE ALL 1'S
  8336                                            ;;THIS TEST PERFORMS MASTER RESET TESTING &
   (1)                                            ;;TESTING OF READ ONLY BIT PARER
   (1)                                            ;;
   (5)                                    ;;*****************************************************************
   (4)  006570  000004                    TST42:  SCOPE
   (4)
   (2)  006572  052777  000400  173126            BIS     #MRESET,@TXCSR  ;MASTER RESET
   (1)  006600  032777  010000  173110            BIT     #PARER,@RXDBUF  ;TEST THIS BIT
   (1)  006606  001401                            BEQ     .+4             ;BR IF "0"
   (1)  006610  104004                            ERROR   4                       ;CHECK MASTER RESET LOGIC
   (1)                                                            ;OR SHORT ON THIS BIT
   (1)
  8337                                            ;;THIS TEST PERFORMS MASTER RESET TESTING &
   (1)                                            ;;TESTING OF READ ONLY BIT FRMERR
   (1)                                            ;;
   (5)                                    ;;*****************************************************************
   (4)  006612  000004                    TST43:  SCOPE
   (4)
   (2)  006614  052777  000400  173104            BIS     #MRESET,@TXCSR  ;MASTER RESET
   (1)  006622  032777  020000  173066            BIT     #FRMERR,@RXDBUF ;TEST THIS BIT
   (1)  006630  001401                            BEQ     .+4             ;BR IF "0"
   (1)  006632  104004                            ERROR   4                       ;CHECK MASTER RESET LOGIC
   (1)                                                            ;OR SHORT ON THIS BIT
   (1)
  8338                                            ;;THIS TEST PERFORMS MASTER RESET TESTING &
   (1)                                            ;;TESTING OF READ ONLY BIT OVRRUN
   (1)                                            ;;
   (5)                                    ;;*****************************************************************
   (4)  006634  000004                    TST44:  SCOPE
   (4)
   (2)  006636  052777  000400  173062            BIS     #MRESET,@TXCSR  ;MASTER RESET
   (1)  006644  032777  040000  173044            BIT     #OVRRUN,@RXDBUF ;TEST THIS BIT
   (1)  006652  001401                            BEQ     .+4             ;BR IF "0"
   (1)  006654  104004                            ERROR   4                       ;CHECK MASTER RESET LOGIC
   (1)                                                            ;OR SHORT ON THIS BIT
   (1)
  8339                                            ;;THIS TEST PERFORMS MASTER RESET TESTING &
   (1)                                            ;;TESTING OF READ ONLY BIT RXERR
   (1)                                            ;;
   (5)                                    ;;*****************************************************************
   (4)  006656  000004                    TST45:  SCOPE
   (4)
   (2)  006660  052777  000400  173040            BIS     #MRESET,@TXCSR  ;MASTER RESET
   (1)  006666  032777  100000  173022            BIT     #RXERR,@RXDBUF  ;TEST THIS BIT
   (1)  006674  001401                            BEQ     .+4             ;BR IF "0"
   (1)  006676  104004                            ERROR   4                       ;CHECK MASTER RESET LOGIC
   (1)                                                            ;OR SHORT ON THIS BIT
```

```
 (1)
8340                                    ;;THIS TEST VERIFYS THAT THE DEVICE REGISTER RXCSR
8341                                    ;;IS CLEARED BY MASTER RESET
8342                                ;;**************************************************************
 (3)  006700  000004                TST46:  SCOPE
 (3)
8343  006702  012777  177777  173002        MOV     #17777,@RXCSR  ;SET ALL POSSIBLE BITS
8344  006710  052777  000400  173010        BIS     #MRESET,@TXCSR  ;MASTER RESET
8345  006716  016703  172770               MOV     RXCSR,R3        ;FOR ERROR MESSAGE
8346  006722  017701  172764               MOV     @RXCSR,R1       ;SAVE ACTUAL
8347  006726  105767  172217               TSTB    OPTCLR          ;TEST THE OPT CLR JUMPER FLAG
8348  006732  100010                       BPL     1$              ;NO ,ITS NOT IN
8349  006734  042701  173000               BIC     #173000,R1      ;CLR NON-MASTER RESETTABLE
8350                                        ;BITS(SINCE THESE ARE DEPENDENT ON H315 CONNECTORS EXISTANCE)
8351  006740  012700  000000               MOV     #0,R0   ;EXPECTED
8352  006744  020001                       CMP     R0,R1   ;EXPECTED VS ACTUAL
8353  006746  001401                       BEQ     .+4
8354  006750  104001                       ERROR   1       ;ALL MASTER RESETABLE BITS SHOULD BE CLR
8355  006752  000407                       BR      2$              ;JUMP AROUND
8356  006754  042701  173000       1$:     BIC     #173000,R1      ;CLR NON-MASTER RESETABLE
8357                                        ;BITS(SINCE THESE ARE DEPENDENT ON H315 CONNECTORS EXISTANCE)
8358  006760  012700  000016               MOV     #16,R0          ;EXPECTED
8359  006764  020001                       CMP     R0,R1   ;EXPECTED VS ACTUAL
8360  006766  001401                       BEQ     .+4
8361  006770  104001                       ERROR   1       ;ONLY STD,RTS,DTR BITS SHOULD BE SET
8362                                        ;NOTE THAT STD IS READ =1 INDEPENDENT OF
8363                                        ;SEC XMIT #6 STRAP
8364  006772                       2$:
8365                                    ;WAIT FOR CABLE DELAYS
 (1)                                    ;*************************************
 (1)                                    ;MODIFY 'HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
 (1)                                    ;*************************************
 (1)  006772  016702  172122            MOV     HOLD,R2 ;SET DELAY TIME
 (1)  006776  005302                    DEC     R2
 (1)  007000  001376                    BNE     .-2     ;WAIT THIS TIME
 (1)                                    ;OK NOW FALL THRU AND CONTINUE TESTING.....
 (1)                                    ;EXIT STAGE LEFT....CHINNNG!
8366
8367
8368                                    ;;THIS TEST VERIFYS THAT THE DEVICE REGISTER TXCSR
8369                                    ;;IS CLEARED BY MASTER RESET
8370                                    ;;
8371                                ;;**************************************************************
 (3)  007002  000004                TST47:  SCOPE
 (3)
8372  007004  012777  177777  172714        MOV     #177777,@TXCSR  ;SET ALL POSSIBLE BITS
8373  007012  052777  000400  172706        BIS     #MRESET,@TXCSR  ;MASTER RESET
8374  007020  016703  172702               MOV     TXCSR,R3        ;FOR ERROR MESSAGE
8375  007024  017701  172676               MOV     @TXCSR,R1       ;SAVE ACTUAL
8376  007030  012700  000200               MOV     #200,R0 ;EXPECTED
8377  007034  020001                       CMP     R0,R1   ;EXPECTED VS ACTUAL
8378  007036  001401                       BEQ     .+4
8379  007040  104001                       ERROR   1       ;ONLY TXDONE SHOULD BE SET
8380
8381                                    ;;THIS TEST VERIFYS THAT THE DEVICE REGISTER RXDBUF
8382                                    ;;IS CLEARED BY MASTER RESET
```

```
                                        ;;
                                ;;*****************************************************************
 (3)  007042  000004            TST50:  SCOPE
 (3)
8385  007044  052777  000400  172654    BIS     #MRESET,@TXCSR  ;MASTER RESET
8386  007052  016703  172640            MOV     RXDBUF,R3       ;FOR ERROR MESSAGE
8387  007056  017701  172634            MOV     @RXDBUF,R1      ;SAVE
8388  007062  012700  000377            MOV     #377,R0 ;EXPECTED
8389  007066  020001                     CMP     R0,R1   ;EXPECTED VS ACTUAL
8390  007070  001401                     BEQ     .+4
8391  007072  104002                     ERROR   2       ;ONLY REC DATA BITS SHOULD BE SET
8392                                      ;;THIS TEST VERIFYS BITS RING,CTS,CARDET,SRD,DSR
8393                                      ;;ALSO DSC IS GENERATED WHEN ANY OF THESE BITS ARE SET
8394                                      ;;OR CLEARED.....IT ALSO CHECKS THE MODEM BYPASS
8395                                      ;;JUMPER AND THAT THESE BITS CAN BE READ
8396                                      ;;NOTE: THE MODEM BYPASS JUMPER MUST BE ON (H315)
8397                                      ;;
8398                                ;;*****************************************************************
 (3)  007074  000004            TST51:  SCOPE
 (3)
8399  007076  005077  172610            CLR     @RXCSR  ;TO GET RID OF STD ,RTS,DTR IF OPTCLR JUMPER #4 IS NOT ON
8400  007102  052777  000400  172616    BIS     #MRESET,@TXCSR  ;MASTER RESET
8401                                      ;TEST THAT A "YES" ANSWER WAS GIVEN TO QUESTION IN
8402                                      ;THE MONITOR OR BY DEFAULT
8403                                      ;THIS TEST WILL BE BYPASSED  IF THE EXTERNAL BYPASS
8404                                      ;JUMPER IS NOT INSTALLED
8405  007110  105767  172037            TSTB    JMRBY
8406  007114  100402                     BMI     .+6     ;THE ANSWER WAS YES.....
8407                                                      ; PERFORM THIS TEST
8408  007116  000167  000652            JMP     OUT1    ;JUMP AROUND THIS TEST IF THE ANSWER
8409                                                      ;WAS NO
8410  007122  016703  172564            MOV     RXCSR,R3        ;SET UP FOR ERROR MESSAGE
8411  007126  017701  172560            MOV     @RXCSR,R1       ;ACTUAL
8412  007132  005000                     CLR     R0      ;EXPECTED
8413  007134  005701                     TST     R1      ;IS IT = 0 ?
8414  007136  001401                     BEQ     .+4
8415  007140  104001                     ERROR   1       ;RXCSR SHOULD BE CLR
8416  007142  052777  000002  172542    BIS     #DTR,@RXCSR     ;SET DTR
8417                                      ;WAIT FOR CABLE DELAYS
 (1)                                    ;*****************************************
 (1)                                    ;MODIFY 'HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
 (1)                                    ;*****************************************
 (1)  007150  016702  171744            MOV     HOLD,R2 ;SET DELAY TIME
 (1)  007154  005302                    DEC     R2
 (1)  007156  001376                    BNE     .-2     ;WAIT THIS TIME
 (1)                                    ;OK NOW FALL THRU AND CONTINUE TESTING.....
 (1)                                    ;EXIT STAGE LEFT....CHINNNG!
8418  007160  017701  172526            MOV     @RXCSR,R1       ;ACTUAL
8419  007164  012700  130002            MOV     #130002,R0      ;DSC,CTS,CARDET,DTR
8420  007170  020001                     CMP     R0,R1   ;EXPECTED VS ACTUAL
8421  007172  001401                     BEQ     .+4
8422  007174  104001                     ERROR   1       ;CHECK BYPASS CONNECTOR
8423  007176  017701  172510            MOV     @RXCSR,R1       ;ACTUAL
8424  007202  012700  030002            MOV     #30002,R0       ;CTS,CARDET,DTR
8425  007206  020001                     CMP     R0,R1   ;EXPECTED VS ACTUAL
8426  007210  001401                     BEQ     .+4
```

```
 8427  007212  104001                          ERROR   1       ;PREVIOUS READING OF RXCSR SHOULD
 8428                                                          ;HAVE CLEARED DSC
 8429  007214  052777  000004  172470          BIS     #RTS,@RXCSR
 8430                                          ;WAIT FOR CABLE DELAYS
  (1)                                          ;*************************************
  (1)                                          ;MODIFY 'HOLD:'  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
  (1)                                          ;*************************************
  (1)  007222  016702  171672                  MOV     HOLD,R2 ;SET DELAY TIME
  (1)  007226  005302                          DEC     R2
  (1)  007230  001376                          BNE     .-2     ;WAIT THIS TIME
  (1)                                          ;OK NOW FALL THRU AND CONTINUE TESTING.....
  (1)                                          ;EXIT STAGE LEFT....CHINNNG!
 8431  007232  017701  172454                  MOV     @RXCSR,R1
 8432  007236  012700  170006                  MOV     #170006,R0      ;DSC,RING,CTS,CARDET,RTS,DTR
 8433  007242  020001                          CMP     R0,R1   ;EXPECTED VS ACTUAL
 8434  007244  001401                          BEQ     .+4
 8435  007246  104001                          ERROR   1       ;CHECK BYPASS CONNECTOR
 8436  007250  017701  172436                  MOV     @RXCSR,R1
 8437  007254  012700  070006                  MOV     #70006,R0       ;RING,CTS,CARDET,RTS,DTR
 8438  007260  020001                          CMP     R0,R1   ;EXPECTED VS ACTUAL
 8439  007262  001401                          BEQ     .+4
 8440  007264  104001                          ERROR   1       ;PREVIOUS READING OF RXCSR SHOULD
 8441                                                          ;HAVE CLEARED DSC
 8442  007266  105767  171655                  TSTB    SEXMIT  ;IS SEC XMIT JUMPER IN ?
 8443  007272  100112                          BPL     OUT2    ;NO
 8444  007274  105767  171650                  TSTB    SEREC   ;IS SEC REC JUMPER IN ?
 8445  007300  100163                          BPL     OUT3    ;NO
 8446  007302  052777  000010  172402          BIS     #STD,@RXCSR
 8447                                          ;WAIT FOR CABLE DELAYS
  (1)                                          ;*************************************
  (1)                                          ;MODIFY 'HOLD:'  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
  (1)                                          ;*************************************
  (1)  007310  016702  171604                  MOV     HOLD,R2 ;SET DELAY TIME
  (1)  007314  005302                          DEC     R2
  (1)  007316  001376                          BNE     .-2     ;WAIT THIS TIME
  (1)                                          ;OK NOW FALL THRU AND CONTINUE TESTING.....
  (1)                                          ;EXIT STAGE LEFT....CHINNNG!
 8448  007320  017701  172366                  MOV     @RXCSR,R1
 8449  007324  012700  173016          MASK1:  MOV     #173016,R0      ;DSC,RING,CTS,CARDET,SRD,DSR
 8450                                                          ;STD,RTS,DTR
 8451  007330  020001                          CMP     R0,R1   ;EXPECTED VS ACTUAL
 8452  007332  001401                          BEQ     .+4
 8453  007334  104001                          ERROR   1       ;CHECK BYPASS CONNECTOR
 8454  007336  017701  172350                  MOV     @RXCSR,R1
 8455  007342  012700  073016          MASK2:  MOV     #73016,R0       ;RING,CTS,CARDET,SRD,DSR,STD
 8456                                                          ;RTS,DTR
 8457  007346  020001                          CMP     R0,R1   ;EXPECTED VS ACTUAL
 8458  007350  001401                          BEQ     .+4
 8459  007352  104001                          ERROR   1       ;PREVIOUS READING OF RXCSR SHOULD
 8460                                                          ;HAVE CLEARED DSC
 8461  007354  042777  000002  172330          BIC     #DTR,@RXCSR
 8462                                          ;WAIT FOR CABLE DELAYS
  (1)                                          ;*************************************
  (1)                                          ;MODIFY 'HOLD:'  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
  (1)                                          ;*************************************
  (1)  007362  016702  171532                  MOV     HOLD,R2 ;SET DELAY TIME
```

```
   (1)  007366  005302                          DEC     R2
   (1)  007370  001376                          BNE     .-2     ;WAIT THIS TIME
   (1)                                          ;OK NOW FALL THRU AND CONTINUE TESTING.....
   (1)                                          ;EXIT STAGE LEFT....CHINNNG!
 8463   007372  017701  172314                  MOV     @RXCSR,R1
 8464   007376  012700  143014                  MOV     #143014,R0      ;DSC,RING,SRD,DSR,STD,RTS
 8465   007402  020001                          CMP     R0,R1   ;EXPECTED VS ACTUAL
 8466   007404  001401                          BEQ     .+4
 8467   007406  104001                          ERROR   1       ;DSC SHOULD BE SET
 8468   007410  042777  000004  172274          BIC     #RTS,@RXCSR
 8469                                           ;WAIT FOR CABLE DELAYS
   (1)                                          ;****************************************
   (1)                                          ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
   (1)                                          ;****************************************
   (1)  007416  016702  171476                  MOV     HOLD,R2 ;SET DELAY TIME
   (1)  007422  005302                          DEC     R2
   (1)  007424  001376                          BNE     .-2     ;WAIT THIS TIME
   (1)                                          ;OK NOW FALL THRU AND CONTINUE TESTING.....
   (1)                                          ;EXIT STAGE LEFT....CHINNNG!
 8470   007426  017701  172260                  MOV     @RXCSR,R1
 8471   007432  012700  103010          MASK3:  MOV     #103010,R0      ;DSC,SRD,DSR,STD
 8472   007436  020001                          CMP     R0,R1   ;EXPECTED VS ACTUAL
 8473   007440  001401                          BEQ     .+4
 8474   007442  104001                          ERROR   1       ;DSC SHOULD BE SET
 8475   007444  042777  000010  172240          BIC     #STD,@RXCSR
 8476                                           ;WAIT FOR CABLE DELAYS
   (1)                                          ;****************************************
   (1)                                          ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
   (1)                                          ;****************************************
   (1)  007452  016702  171442                  MOV     HOLD,R2 ;SET DELAY TIME
   (1)  007456  005302                          DEC     R2
   (1)  007460  001376                          BNE     .-2     ;WAIT THIS TIME
   (1)                                          ;OK NOW FALL THRU AND CONTINUE TESTING.....
   (1)                                          ;EXIT STAGE LEFT....CHINNNG!
 8477   007462  017701  172224                  MOV     @RXCSR,R1
 8478   007466  012700  100000                  MOV     #100000,R0      ;DSC
 8479   007472  020001                          CMP     R0,R1   ;EXPECTED VS ACTUAL
 8480   007474  001401                          BEQ     .+4
 8481   007476  104001                          ERROR   1       ;DSC SHOULD BE SET
 8482   007500  017701  172206                  MOV     @RXCSR,R1
 8483   007504  005000                          CLR     R0      ;NONE
 8484   007506  005701                          TST     R1
 8485   007510  001401                          BEQ     .+4
 8486   007512  104001                          ERROR   1       ;DSC SHOULD BE CLEARED FROM PREVIOUS
 8487                                                           ;READING OF RXCSR
 8488   007514  000167  000254                  JMP     OUT1    ;JUMP AROUND
 8489                                   ;THE FOLLOWING ROUTINE HANDLES THE SITUATION WHERE SEC XMIT
 8490                                   ;AND SEC REC JUMPERS ARE NOT ON
 8491   007520  052777  000010  172164  OUT2:   BIS     #STD,@RXCSR
 8492                                           ;WAIT FOR CABLE DELAYS
   (1)                                          ;****************************************
   (1)                                          ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
   (1)                                          ;****************************************
   (1)  007526  016702  171366                  MOV     HOLD,R2 ;SET DELAY TIME
   (1)  007532  005302                          DEC     R2
   (1)  007534  001376                          BNE     .-2     ;WAIT THIS TIME
```

```
6
  (1)                                           ;OK NOW FALL THRU AND CONTINUE TESTING.....
  (1)                                           ;EXIT STAGE LEFT....CHINNNG!
8493  007536  017701  172150                    MOV     @RXCSR,R1       ;ACTUAL
8494  007542  012700  070016                    MOV     #70016,R0       ;EXPECTED: RING ,CTS,CARDET,STD,RTS,DTR
8495  007546  020001                            CMP     R0,R1   ;EXPECTED VS ACTUAL
8496  007550  001401                            BEQ     .+4
8497  007552  104001                            ERROR   1       ;CHECK SEC XMIT & SEC REC JUMPERS
8498  007554  042777  000004  172130            BIC     #RTS,@RXCSR
8499                                            ;WAIT FOR CABLE DELAYS
  (1)                                           ;****************************************
  (1)                                           ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
  (1)                                           ;****************************************
  (1)  007562  016702  171332                   MOV     HOLD,R2 ;SET DELAY TIME
  (1)  007566  005302                           DEC     R2
  (1)  007570  001376                           BNE     .-2     ;WAIT THIS TIME
  (1)                                           ;OK NOW FALL THRU AND CONTINUE TESTING.....
  (1)                                           ;EXIT STAGE LEFT....CHINNNG!
8500  007572  017701  172114                    MOV     @RXCSR,R1       ;ACTUAL
8501  007576  012700  130012                    MOV     #130012,R0      ;DSC,CTS,CARDET,DTR,STD
8502                                            ;NOTE THAT DSC STILL ASSERTS EVEN THO THE SEC XMIT JUMPER # 6 IS NOT ON
8503  007602  020001                            CMP     R0,R1   ;EXPECTED VS ACTUAL
8504  007604  001401                            BEQ     .+4
8505  007606  104001                            ERROR   1       ;CHECK BYPASS CONNECTOR
8506  007610  042777  000002  172074            BIC     #DTR,@RXCSR
8507                                            ;WAIT FOR CABLE DELAYS
  (1)                                           ;****************************************
  (1)                                           ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
  (1)                                           ;****************************************
  (1)  007616  016702  171276                   MOV     HOLD,R2 ;SET DELAY TIME
  (1)  007622  005302                           DEC     R2
  (1)  007624  001376                           BNE     .-2     ;WAIT THIS TIME
  (1)                                           ;OK NOW FALL THRU AND CONTINUE TESTING.....
  (1)                                           ;EXIT STAGE LEFT....CHINNNG!
8508  007626  017701  172060                    MOV     @RXCSR,R1       ;ACTUAL
8509  007632  012700  100010                    MOV     #100010,R0      ;DSC,STD
8510  007636  020001                            CMP     R0,R1   ;EXPECTED VS ACTUAL
8511  007640  001401                            BEQ     .+4
8512  007642  104001                            ERROR   1       ;ONLY DSC & STD  SHOULD BE SET
8513  007644  000167  000124                    JMP     OUT1    ;JUMP AROUND
8514  007650  052777  000010  172034  OUT3:     BIS     #STD,@RXCSR
8515                                            ;WAIT FOR CABLE DELAYS
  (1)                                           ;****************************************
  (1)                                           ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
  (1)                                           ;****************************************
  (1)  007656  016702  171236                   MOV     HOLD,R2 ;SET DELAY TIME
  (1)  007662  005302                           DEC     R2
  (1)  007664  001376                           BNE     .-2     ;WAIT THIS TIME
  (1)                                           ;OK NOW FALL THRU AND CONTINUE TESTING.....
  (1)                                           ;EXIT STAGE LEFT....CHINNNG!
8516  007666  017701  172020                    MOV     @RXCSR,R1       ;ACTUAL
8517  007672  012700  171016                    MOV     #171016,R0      ;EXPECTED: DSC,RING,CTS,CARDET,DSR,STD,RTS,DTR
8518  007676  020001                            CMP     R0,R1   ;EXPECTED VS ACTUAL
8519  007700  001401                            BEQ     .+4
8520  007702  104001                            ERROR   1       ;CHECK SEC REC JUMPER
8521  007704  042777  000004  172000            BIC     #RTS,@RXCSR
8522                                            ;WAIT FOR CABLE DELAYS
```

```
                                        ;****************************************
                                        ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
                                        ;****************************************
007712  016702  171202                  MOV     HOLD,R2 ;SET DELAY TIME
007716  005302                          DEC     R2
007720  001376                          BNE     .-2     ;WAIT THIS TIME
                                        ;OK NOW FALL THRU AND CONTINUE TESTING.....
                                        ;EXIT STAGE LEFT....CHINNNG!
007722  017701  171764                  MOV     @RXCSR,R1       ;ACTUAL
007726  012700  131012                  MOV     #131012,R0      ;EXPECTED: DSC,CTS,CARDET,DSR,STD,DTR
007732  020001                          CMP     R0,R1   ;EXPECTED VS ACTUAL
007734  001401                          BEQ     .+4
007736  104001                          ERROR   1       ;CHECK H315 CONNECTOR
007740  042777  000002  171744          BIC     #DTR,@RXCSR
                                        ;WAIT FOR CABLE DELAYS
                                        ;****************************************
                                        ;MODIFY "HOLD:"  ACCORDINGLY FOR FASTER OR SLOWER  MACHINE
                                        ;****************************************
007746  016702  171146                  MOV     HOLD,R2 ;SET DELAY TIME
007752  005302                          DEC     R2
007754  001376                          BNE     .-2     ;WAIT THIS TIME
                                        ;OK NOW FALL THRU AND CONTINUE TESTING.....
                                        ;EXIT STAGE LEFT....CHINNNG!
007756  017701  171730                  MOV     @RXCSR,R1       ;ACTUAL
007762  012700  101010                  MOV     #101010,R0      ;EXPECTED: DSC,DSR,STD
007766  020001                          CMP     R0,R1   ;EXPECTED VS ACTUAL
007770  001401                          BEQ     .+4
007772  104001                          ERROR   1       ;CHECK H315 CONNECTOR
007774                          OUT1:

                                        ;;THIS TEST VERIFYS THAT RECACT (REC ACTIVE) ASSERTS
                                        ;;IMMED. WHEN SYNC EXTERNAL MODE IS SELECTED
                                        ;;AND SYNC SEARCH IS SET
                                        ;;
                                ;;**************************************************************************
007774  000004                 TST52:   SCOPE

007776  052777  000400  171722          BIS     #MRESET,@TXCSR  ;MASTER RESET
010004  012777  020000  171710          MOV     #SYNEXT,@PARCSR ;SET THE MODE
010012  052777  000400  171706          BIS     #MRESET,@TXCSR  ;MASTER RESET

                                ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
010020  012777  064001  171700          MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR

                                ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
010026  012777  026026  171666          MOV     #SYNEXT!EIGHT!NOPAR!26,@PARCSR
010034  032777  004000  171650          BIT     #RECACT,@RXCSR
010042  001401                          BEQ     .+4
010044  104004                          ERROR   4               ;RECACT SHOULD NOT BE SET
010046  052777  000020  171636          BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
010054  032777  004000  171630          BIT     #RECACT,@RXCSR
010062  001001                          BNE     .+4
010064  104004                          ERROR   4               ;RECACT DID NOT ASSERT
010066  042777  000020  171616          BIC     #SYNSCH,@RXCSR  ;DROP SEARCH SYNC

010074  032777  004000  171610          BIT     #RECACT,@RXCSR  ;IS IT =0?
```

```
   (1)  010102  001401                          BEQ     .+4
   (1)  010104  104004                          ERROR   4       ;RECACT SHOULD BE 0
   (1)
   (1)
  8538                                          ;;THIS TEST VERIFYS THAT RECACT (REC ACTIVE) ASSERTS
   (1)                                          ;;IMMED. WHEN ISOCRONOUS MODE IS SELECTED
   (1)                                          ;;AND SYNC SEARCH IS SET
   (1)                                          ;;
   (5)                                  ;;**************************************************************************
   (4)  010106  000004                  TST53:  SCOPE
   (4)
   (3)  010110  052777  000400  171610          BIS     #MRESET,@TXCSR  ;MASTER RESET
   (2)  010116  012777  000000  171576          MOV     #ISYMOD,@PARCSR ;SET THE MODE
   (3)  010124  052777  000400  171574          BIS     #MRESET,@TXCSR  ;MASTER RESET
   (2)
   (2)                                  ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
   (2)  010132  012777  064001  171566          MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
   (2)
   (2)                                  ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
   (2)  010140  012777  006026  171554          MOV     #ISYMOD!EIGHT!NOPAR!26,@PARCSR
   (1)  010146  032777  004000  171536          BIT     #RECACT,@RXCSR
   (1)  010154  001401                          BEQ     .+4
   (1)  010156  104004                          ERROR   4               ;RECACT SHOULD NOT BE SET
   (1)  010160  052777  000020  171524          BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
   (1)  010166  032777  004000  171516          BIT     #RECACT,@RXCSR
   (1)  010174  001001                          BNE     .+4
   (1)  010176  104004                          ERROR   4               ;RECACT DID NOT ASSERT
   (1)  010200  042777  000020  171504          BIC     #SYNSCH,@RXCSR  ;DROP SEARCH SYNC
   (1)
   (1)  010206  032777  004000  171476          BIT     #RECACT,@RXCSR  ;IS IT =0?
   (1)  010214  001401                          BEQ     .+4
   (1)  010216  104004                          ERROR   4       ;RECACT SHOULD BE 0
   (1)
   (1)
  8539                                          ;;VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
   (1)                                          ;;IN TWO * SYNC CHARS THRU MAINT DATA BIT
   (1)                                          ;;WATCH THE RECACT BIT
   (1)                                          ;;ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
   (1)                                          ;;*: DEPENDENT ON MONITOR.............
   (1)                                          ;;IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
   (1)                                          ;;TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
   (1)                                          ;;ON THE SECOND CHARACTER
   (1)                                          ;;ALSO CHECK THIS CHARACTER IN RXDBUF
   (1)                                          ;;AND CHECK OPERATION OF SYNSCH
   (1)                                          ;;MODE: SYNC INTERNAL
   (1)                                          ;;LENGTH:FIVE
   (1)                                          ;;
   (5)                                  ;;**************************************************************************
   (4)  010220  000004                  TST54:  SCOPE
   (4)
   (3)  010222  052777  000400  171476          BIS     #MRESET,@TXCSR  ;MASTER RESET
   (2)  010230  012777  030000  171464          MOV     #SYNINT,@PARCSR ;SET THE MODE
   (3)  010236  052777  000400  171462          BIS     #MRESET,@TXCSR  ;MASTER RESET
   (2)
   (2)                                  ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
   (2)  010244  012777  064001  171454          MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
```

```
    (2)
    (2)                                         ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
    (2)  010252  012777  030026  171442          MOV     #SYNINT!FIVE!NOPAR!26,@PARCSR
    (1)  010260  016703  171432                 MOV     RXDBUF,R3       ;SET UP FOR ERROR MESSAGE
    (2)  010264  052777  000020  171420          BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
    (2)                                         ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
    (2)  010272  042777  020000  171426          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
    (2)  010300  052777  020000  171420          BIS     #CLK,@TXCSR     ;POKE CLK UP
    (2)                                         ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
    (2)  010306  042777  020000  171412          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
    (2)  010314  052777  020000  171404          BIS     #CLK,@TXCSR     ;POKE CLK UP
    (1)  010322  012767  000002  170574          MOV     #2,COUNT
    (1)  010330  012767  000005  170564  1$:     MOV     #5,SHIFT        ;# OF SHIFTS
    (1)  010336  012767  000026  171134          MOV     #26,$TMP1       ;SYNC CHARACTER
    (1)  010344  004767  006554                 JSR     PC,RPOKE
    (1)  010350  005367  170550                 DEC     COUNT
    (1)  010354  001403                         BEQ     2$
    (1)                                         ;TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
    (1)  010356  105767  170564                 TSTB    SYNCNO
    (1)  010362  100762                         BMI     1$      ;TWO SYNC CHARS
    (1)  010364  105777  171322          2$:    TSTB    @RXCSR  ;CHECK REC DONE BIT
    (1)  010370  100001                         BPL     .+4
    (1)  010372  104004                         ERROR   4               ;RXDONE SHOULD NOT BE ASSERTED
    (1)  010374  032777  004000  171310         BIT     #RECACT,@RXCSR
    (1)  010402  001001                         BNE     .+4
    (1)  010404  104004                         ERROR   4               ;RECACT SHOULD BE ASSERTED
    (1)  010406  012767  000005  170506         MOV     #5,SHIFT
    (1)  010414  012767  000021  171056         MOV     #21,$TMP1       ;ANY CHARACTER
    (1)  010422  004767  006476                 JSR     PC,RPOKE
    (1)  010426  105777  171260                 TSTB    @RXCSR  ;CHECK RXDONE
    (1)  010432  100401                         BMI     .+4
    (1)  010434  104004                         ERROR   4               ;RXDONE SHOULD BE ASSERTED
    (1)  010436  032777  004000  171246         BIT     #RECACT,@RXCSR
    (1)  010444  001001                         BNE     .+4
    (1)  010446  104004                         ERROR   4               ;RECACT SHOULD STILL BE ASSERTED
    (1)  010450  042777  000020  171234         BIC     #SYNSCH,@RXCSR  ;CLR SYNC SEARCH
    (1)  010456  032777  004000  171226         BIT     #RECACT,@RXCSR  ;IT SHOULD DROP IMMEDIATELY
    (1)  010464  001401                         BEQ     .+4
    (1)  010466  104004                         ERROR   4               ;RECACT SHOULD BE CLR
    (1)  010470  105777  171216                 TSTB    @RXCSR  ;RXDONE
    (1)  010474  100401                         BMI     .+4
    (1)  010476  104004                         ERROR   4               ;RXDONE SHOULD STILL BE ASSERTED
    (1)  010500  012700  000021                 MOV     #21,R0  ;EXPECTED DATA
    (1)  010504  017701  171206                 MOV     @RXDBUF,R1      ;ACTUAL DATA
    (1)  010510  020001                         CMP     R0,R1   ;COMPARE EXP VS ACT
    (1)  010512  001401                         BEQ     .+4
    (1)  010514  104002                         ERROR   2               ;DATA CHARS SHOULD COMPARE
    (1)  010516  105777  171170                 TSTB    @RXCSR  ;CHECK RXDONE
    (1)  010522  100001                         BPL     .+4
    (1)  010524  104004                         ERROR   4       ;RXDONE SHOULD BE CLR FROM
    (1)                                         ;PREVIOUS READING OF RXDBUF
    (1)
 8540                                           ;;VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
    (1)                                         ;;IN TWO * SYNC CHARS THRU MAINT DATA BIT
    (1)                                         ;;WATCH THE RECACT BIT
    (1)                                         ;;ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
```

```
(1)                                     ;;*: DEPENDENT ON MONITOR.............
(1)                                     ;;IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
(1)                                     ;;TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
(1)                                     ;;ON THE SECOND CHARACTER
(1)                                     ;;ALSO CHECK THIS CHARACTER IN RXDBUF
(1)                                     ;;AND CHECK OPERATION OF SYNSCH
(1)                                     ;;MODE: SYNC INTERNAL
(1)                                     ;;LENGTH:SIX
(1)                                     ;;
(5)                                     ;;*****************************************************************
(4)  010526  000004                     TST55:  SCOPE
(4)
(3)  010530  052777  000400  171170             BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  010536  012777  030000  171156             MOV     #SYNINT,@PARCSR ;SET THE MODE
(3)  010544  052777  000400  171154             BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)  010552  012777  064001  171146             MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)                                     ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2)  010560  012777  032026  171134             MOV     #SYNINT!SIX!NOPAR!26,@PARCSR
(1)  010566  016703  171124                     MOV     RXDBUF,R3       ;SET UP FOR ERROR MESSAGE
(2)  010572  052777  000020  171112             BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
(2)                                     ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2)  010600  042777  020000  171120             BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  010606  052777  020000  171112             BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2)  010614  042777  020000  171104             BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  010622  052777  020000  171076             BIS     #CLK,@TXCSR     ;POKE CLK UP
(1)  010630  012767  000002  170266             MOV     #2,COUNT
(1)  010636  012767  000006  170256  1$:       MOV     #6,SHIFT        ;# OF SHIFTS
(1)  010644  012767  000026  170626             MOV     #26,$TMP1       ;SYNC CHARACTER
(1)  010652  004767  006246                     JSR     PC,RPOKE
(1)  010656  005367  170242                     DEC     COUNT
(1)  010662  001403                             BEQ     2$
(1)                                     ;TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
(1)  010664  105767  170256                     TSTB    SYNCNO
(1)  010670  100762                             BMI     1$      ;TWO SYNC CHARS
(1)  010672  105777  171014          2$:        TSTB    @RXCSR  ;CHECK REC DONE BIT
(1)  010676  100001                             BPL     .+4
(1)  010700  104004                             ERROR   4               ;RXDONE SHOULD NOT BE ASSERTED
(1)  010702  032777  004000  171002             BIT     #RECACT,@RXCSR
(1)  010710  001001                             BNE     .+4
(1)  010712  104004                             ERROR   4               ;RECACT SHOULD BE ASSERTED
(1)  010714  012767  000006  170200             MOV     #6,SHIFT
(1)  010722  012767  000021  170550             MOV     #21,$TMP1       ;ANY CHARACTER
(1)  010730  004767  006170                     JSR     PC,RPOKE
(1)  010734  105777  170752                     TSTB    @RXCSR  ;CHECK RXDONE
(1)  010740  100401                             BMI     .+4
(1)  010742  104004                             ERROR   4               ;RXDONE SHOULD BE ASSERTED
(1)  010744  032777  004000  170740             BIT     #RECACT,@RXCSR
(1)  010752  001001                             BNE     .+4
(1)  010754  104004                             ERROR   4               ;RECACT SHOULD STILL BE ASSERTED
(1)  010756  042777  000020  170726             BIC     #SYNSCH,@RXCSR  ;CLR SYNC SEARCH
(1)  010764  032777  004000  170720             BIT     #RECACT,@RXCSR  ;IT SHOULD DROP IMMEDIATELY
(1)  010772  001401                             BEQ     .+4
```

```
(1)  010774  104004                    ERROR   4               ;RECACT SHOULD BE CLR
(1)  010776  105777  170710            TSTB    @RXCSR  ;RXDONE
(1)  011002  100401                    BMI     .+4
(1)  011004  104004                    ERROR   4               ;RXDONE SHOULD STILL BE ASSERTED
(1)  011006  012700  000021            MOV     #21,R0  ;EXPECTED DATA
(1)  011012  017701  170700            MOV     @RXDBUF,R1      ;ACTUAL DATA
(1)  011016  020001                    CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  011020  001401                    BEQ     .+4
(1)  011022  104002                    ERROR   2       ;DATA CHARS SHOULD COMPARE
(1)  011024  105777  170662            TSTB    @RXCSR  ;CHECK RXDONE
(1)  011030  100001                    BPL     .+4
(1)  011032  104004                    ERROR   4       ;RXDONE SHOULD BE CLR FROM
(1)                                             ;PREVIOUS READING OF RXDBUF
(1)
8541                                    ;;VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
(1)                                     ;;IN TWO * SYNC CHARS THRU MAINT DATA BIT
(1)                                     ;;WATCH THE RECACT BIT
(1)                                     ;;ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
(1)                                     ;;*: DEPENDENT ON MONITOR.............
(1)                                     ;;IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
(1)                                     ;;TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
(1)                                     ;;ON THE SECOND CHARACTER
(1)                                     ;;ALSO CHECK THIS CHARACTER IN RXDBUF
(1)                                     ;;AND CHECK OPERATION OF SYNSCH
(1)                                     ;;MODE: SYNC INTERNAL
(1)                                     ;;LENGTH:SEVEN
(1)                                     ;;
(5)                             ;;**************************************************************
(4)  011034  000004             TST56:  SCOPE
(4)
(3)  011036  052777  000400  170662            BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  011044  012777  030000  170650            MOV     #SYNINT,@PARCSR ;SET THE MODE
(3)  011052  052777  000400  170646            BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)  011060  012777  064001  170640            MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)                                     ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2)  011066  012777  034026  170626            MOV     #SYNINT!SEVEN!NOPAR!26,@PARCSR
(1)  011074  016703  170616                    MOV     RXDBUF,R3       ;SET UP FOR ERROR MESSAGE
(2)  011100  052777  000020  170604            BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
(2)                                             ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2)  011106  042777  020000  170612            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  011114  052777  020000  170604            BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2)  011122  042777  020000  170576            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  011130  052777  020000  170570            BIS     #CLK,@TXCSR     ;POKE CLK UP
(1)  011136  012767  000002  167760            MOV     #2,COUNT
(1)  011144  012767  000007  167750  1$:       MOV     #7,SHIFT        ;# OF SHIFTS
(1)  011152  012767  000026  170320            MOV     #26,$TMP1       ;SYNC CHARACTER
(1)  011160  004767  005740                    JSR     PC,RPOKE
(1)  011164  005367  167734                    DEC     COUNT
(1)  011170  001403                            BEQ     2$
(1)                                             ;TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
(1)  011172  105767  167750                    TSTB    SYNCNO
(1)  011176  100762                            BMI     1$      ;TWO SYNC CHARS
```

```
    (1)  011200  105777  170506          2$:     TSTB    @RXCSR  ;CHECK REC DONE BIT
    (1)  011204  100001                          BPL     .+4
    (1)  011206  104004                          ERROR   4               ;RXDONE SHOULD NOT BE ASSERTED
    (1)  011210  032777  004000  170474          BIT     #RECACT,@RXCSR
    (1)  011216  001001                          BNE     .+4
    (1)  011220  104004                          ERROR   4               ;RECACT SHOULD BE ASSERTED
    (1)  011222  012767  000007  167672          MOV     #7,SHIFT
    (1)  011230  012767  000021  170242          MOV     #21,$TMP1       ;ANY CHARACTER
    (1)  011236  004767  005662                  JSR     PC,RPOKE
    (1)  011242  105777  170444                  TSTB    @RXCSR  ;CHECK RXDONE
    (1)  011246  100401                          BMI     .+4
    (1)  011250  104004                          ERROR   4               ;RXDONE SHOULD BE ASSERTED
    (1)  011252  032777  004000  170432          BIT     #RECACT,@RXCSR
    (1)  011260  001001                          BNE     .+4
    (1)  011262  104004                          ERROR   4               ;RECACT SHOULD STILL BE ASSERTED
    (1)  011264  042777  000020  170420          BIC     #SYNSCH,@RXCSR  ;CLR SYNC SEARCH
    (1)  011272  032777  004000  170412          BIT     #RECACT,@RXCSR  ;IT SHOULD DROP IMMEDIATELY
    (1)  011300  001401                          BEQ     .+4
    (1)  011302  104004                          ERROR   4               ;RECACT SHOULD BE CLR
    (1)  011304  105777  170402                  TSTB    @RXCSR  ;RXDONE
    (1)  011310  100401                          BMI     .+4
    (1)  011312  104004                          ERROR   4               ;RXDONE SHOULD STILL BE ASSERTED
    (1)  011314  012700  000021                  MOV     #21,R0  ;EXPECTED DATA
    (1)  011320  017701  170372                  MOV     @RXDBUF,R1      ;ACTUAL DATA
    (1)  011324  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
    (1)  011326  001401                          BEQ     .+4
    (1)  011330  104002                          ERROR   2               ;DATA CHARS SHOULD COMPARE
    (1)  011332  105777  170354                  TSTB    @RXCSR  ;CHECK RXDONE
    (1)  011336  100001                          BPL     .+4
    (1)  011340  104004                          ERROR   4               ;RXDONE SHOULD BE CLR FROM
    (1)                                                                  ;PREVIOUS READING OF RXDBUF
    (1)
 8542                                            ;;VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
    (1)                                          ;;IN TWO * SYNC CHARS THRU MAINT DATA BIT
    (1)                                          ;;WATCH THE RECACT BIT
    (1)                                          ;;ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
    (1)                                          ;;*: DEPENDENT ON MONITOR.............
    (1)                                          ;;IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
    (1)                                          ;;TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
    (1)                                          ;;ON THE SECOND CHARACTER
    (1)                                          ;;ALSO CHECK THIS CHARACTER IN RXDBUF
    (1)                                          ;;AND CHECK OPERATION OF SYNSCH
    (1)                                          ;;MODE: SYNC INTERNAL
    (1)                                          ;;LENGTH:EIGHT
    (1)                                          ;;
    (5)                                  ;;**************************************************************
    (4)  011342  000004                  TST57:  SCOPE
    (4)
    (3)  011344  052777  000400  170354          BIS     #MRESET,@TXCSR  ;MASTER RESET
    (2)  011352  012777  030000  170342          MOV     #SYNINT,@PARCSR ;SET THE MODE
    (3)  011360  052777  000400  170340          BIS     #MRESET,@TXCSR  ;MASTER RESET
    (2)
    (2)                                  ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
    (2)  011366  012777  064001  170332          MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
    (2)
    (2)                                  ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
```

```
   (2)  011374  012777  036026  170320          MOV     #SYNINT!EIGHT!NOPAR!26,@PARCSR
   (1)  011402  016703  170310                  MOV     RXDBUF,R3       ;SET UP FOR ERROR MESSAGE
   (2)  011406  052777  000020  170276          BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
   (2)                                          ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
   (2)  011414  042777  020000  170304          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
   (2)  011422  052777  020000  170276          BIS     #CLK,@TXCSR     ;POKE CLK UP
   (2)                                  ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
   (2)  011430  042777  020000  170270          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
   (2)  011436  052777  020000  170262          BIS     #CLK,@TXCSR     ;POKE CLK UP
   (1)  011444  012767  000002  167452          MOV     #2,COUNT
   (1)  011452  012767  000010  167442  1$:     MOV     #8.,SHIFT       ;# OF SHIFTS
   (1)  011460  012767  000026  170012          MOV     #26,$TMP1       ;SYNC CHARACTER
   (1)  011466  004767  005432                  JSR     PC,RPOKE
   (1)  011472  005367  167426                  DEC     COUNT
   (1)  011476  001403                          BEQ     2$
   (1)                                          ;TEST SYNCNO TO SEE HOw MANY SYNC CHARS WERE SELECTED
   (1)  011500  105767  167442                  TSTB    SYNCNO
   (1)  011504  100762                          BMI     1$      ;TWO SYNC CHARS
   (1)  011506  105777  170200          2$:     TSTB    @RXCSR  ;CHECK REC DONE BIT
   (1)  011512  100001                          BPL     .+4
   (1)  011514  104004                          ERROR   4               ;RXDONE SHOULD NOT BE ASSERTED
   (1)  011516  032777  004000  170166          BIT     #RECACT,@RXCSR
   (1)  011524  001001                          BNE     .+4
   (1)  011526  104004                          ERROR   4               ;RECACT SHOULD BE ASSERTED
   (1)  011530  012767  000010  167364          MOV     #8.,SHIFT
   (1)  011536  012767  000021  167734          MOV     #21,$TMP1       ;ANY CHARACTER
   (1)  011544  004767  005354                  JSR     PC,RPOKE
   (1)  011550  105777  170136                  TSTB    @RXCSR  ;CHECK RXDONE
   (1)  011554  100401                          BMI     .+4
   (1)  011556  104004                          ERROR   4               ;RXDONE SHOULD BE ASSERTED
   (1)  011560  032777  004000  170124          BIT     #RECACT,@RXCSR
   (1)  011566  001001                          BNE     .+4
   (1)  011570  104004                          ERROR   4               ;RECACT SHOULD STILL BE ASSERTED
   (1)  011572  042777  000020  170112          BIC     #SYNSCH,@RXCSR  ;CLR SYNC SEARCH
   (1)  011600  032777  004000  170104          BIT     #RECACT,@RXCSR  ;IT SHOULD DROP IMMEDIATELY
   (1)  011606  001401                          BEQ     .+4
   (1)  011610  104004                          ERROR   4               ;RECACT SHOULD BE CLR
   (1)  011612  105777  170074                  TSTB    @RXCSR  ;RXDONE
   (1)  011616  100401                          BMI     .+4
   (1)  011620  104004                          ERROR   4               ;RXDONE SHOULD STILL BE ASSERTED
   (1)  011622  012700  000021                  MOV     #21,R0  ;EXPECTED DATA
   (1)  011626  017701  170064                  MOV     @RXDBUF,R1      ;ACTUAL DATA
   (1)  011632  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
   (1)  011634  001401                          BEQ     .+4
   (1)  011636  104002                          ERROR   2       ;DATA CHARS SHOULD COMPARE
   (1)  011640  105777  170046                  TSTB    @RXCSR  ;CHECK RXDONE
   (1)  011644  100001                          BPL     .+4
   (1)  011646  104004                          ERROR   4       ;RXDONE SHOULD BE CLR FROM
   (1)                                          ;PREVIOUS READING OF RXDBUF
   (1)
8543                                            ;;THIS TEST VERIFYS WORD LENGTH SELECT OF THE
   (1)                                          ;;RECEIVER SECTION,IT USES THE ERROR FLAGS
   (1)                                          ;;TO DETERMINE THAT IT WAS SELECTED CORRECTLY
   (1)                                          ;;(OVRRUN,RXERR)
   (1)                                          ;;MODE:ISYMOD
   (1)                                          ;;LENGTH:FIVE
```

```
(1)                                             ;;CHAR:25
(1)                                             ;;
(5)                                             ;;*****************************************************************
(4)  011650  000004                             TST60:  SCOPE
(4)
(3)  011652  052777  000400  170046                    BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  011660  012777  000000  170034                    MOV     #ISYMOD,@PARCSR ;SET THE MODE
(3)  011666  052777  000400  170032                    BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                             ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)  011674  012777  064001  170024                    MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)                                             ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2)  011702  012777  000000  170012                    MOV     #ISYMOD!FIVE!NOPAR!0,@PARCSR
(2)  011710  052777  000020  167774                    BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
(2)                                                     ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2)  011716  042777  020000  170002                    BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  011724  052777  020000  167774                    BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)                                             ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2)  011732  042777  020000  167766                    BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  011740  052777  020000  167760                    BIS     #CLK,@TXCSR     ;POKE CLK UP
(1)  011746  016703  167744                            MOV     RXDBUF,R3       ;SET UP FOR ERROR MESSAGE
(1)  011752  012700  000025                            MOV     #25,R0  ;EXPECTED
(1)  011756  012767  000007  167136                    MOV     #7,SHIFT        ;# OF SHIFTS
(1)  011764  012767  000152  167506                    MOV     #152,$TMP1      ;DATA CHAR
(1)  011772  004767  005126                            JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)  011776  105777  167710                            TSTB    @RXCSR  ;RXDONE ?
(1)  012002  100401                                    BMI     .+4
(1)  012004  104004                                    ERROR   4       ;RXDONE SHOULD BE SET
(1)  012006  017701  167704                            MOV     @RXDBUF,R1      ;ACTUAL
(1)  012012  020001                                    CMP     R0,R1   ;COMPARE EXPECTED VS. ACTUAL
(1)  012014  001401                                    BEQ     .+4
(1)  012016  104002                                    ERROR   2       ;RECEIVED DATA DID NOT MATCH
(1)                                                                     ;EXPECTED DATA - CHECK MAINT DATA
(1)                                                                     ;OR RECEIVER LOGIC
(1)  012020  012767  000007  167074                    MOV     #7,SHIFT        ;# OF SHIFTS
(1)  012026  012767  000152  167444                    MOV     #152,$TMP1      ;DATA CHAR
(1)  012034  004767  005064                            JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)                                             ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
(1)  012040  012767  000007  167054                    MOV     #7,SHIFT        ;# OF SHIFTS
(1)  012046  012767  000152  167424                    MOV     #152,$TMP1      ;DATA CHAR
(1)  012054  004767  005044                            JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)  012060  012700  140025                            MOV     #140000!25,R0   ;EXPECTED DATA PLUS
(1)                                                                     ;RXERR & OVRRUN
(1)  012064  017701  167626                            MOV     @RXDBUF,R1      ;ACTUAL
(1)  012070  020001                                    CMP     R0,R1   ;COMPARE EXP VS. ACT
(1)  012072  001401                                    BEQ     .+4
(1)  012074  104002                                    ERROR   2       ;SPECIFICALLY LOOK AT RXERR &
(1)                                                                     ;OVRRUN BITS...THEY BOTH SHOULD BE SET
(1)
8544                                            ;;THIS TEST VERIFYS WORD LENGTH SELECT OF THE
(1)                                             ;;RECEIVER SECTION,IT USES THE ERROR FLAGS
(1)                                             ;;TO DETERMINE THAT IT WAS SELECTED CORRECTLY
(1)                                             ;;(OVRRUN,RXERR)
(1)                                             ;;MODE:ISYMOD
(1)                                             ;;LENGTH:FIVE
```

```
(1)                                             ;;CHAR:12
(1)                                             ;;
(5)                                             ;;*****************************************************************
(4)  012076  000004                             TST61:  SCOPE
(4)
(3)  012100  052777  000400  167620                     BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  012106  012777  000000  167606                     MOV     #ISYMOD,@PARCSR ;SET THE MODE
(3)  012114  052777  000400  167604                     BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                             ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)  012122  012777  064001  167576                     MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)                                             ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2)  012130  012777  000000  167564                     MOV     #ISYMOD!FIVE!NOPAR!0,@PARCSR
(2)  012136  052777  000020  167546                     BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
(2)                                                     ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2)  012144  042777  020000  167554                     BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  012152  052777  020000  167546                     BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)                                             ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2)  012160  042777  020000  167540                     BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  012166  052777  020000  167532                     BIS     #CLK,@TXCSR     ;POKE CLK UP
(1)  012174  016703  167516                             MOV     RXDBUF,R3       ;SET UP FOR ERROR MESSAGE
(1)  012200  012700  000012                             MOV     #12,R0  ;EXPECTED
(1)  012204  012767  000007  166710                     MOV     #7,SHIFT        ;# OF SHIFTS
(1)  012212  012767  000124  167260                     MOV     #124,$TMP1      ;DATA CHAR
(1)  012220  004767  004700                             JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)  012224  105777  167462                             TSTB    @RXCSR  ;RXDONE ?
(1)  012230  100401                                     BMI     .+4
(1)  012232  104004                                     ERROR   4       ;RXDONE SHOULD BE SET
(1)  012234  017701  167456                             MOV     @RXDBUF,R1      ;ACTUAL
(1)  012240  020001                                     CMP     R0,R1   ;COMPARE EXPECTED VS. ACTUAL
(1)  012242  001401                                     BEQ     .+4
(1)  012244  104002                                     ERROR   2       ;RECEIVED DATA DID NOT MATCH
(1)                                                                     ;EXPECTED DATA - CHECK MAINT DATA
(1)                                                                     ;OR RECEIVER LOGIC
(1)  012246  012767  000007  166646                     MOV     #7,SHIFT        ;# OF SHIFTS
(1)  012254  012767  000124  167216                     MOV     #124,$TMP1      ;DATA CHAR
(1)  012262  004767  004636                             JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)                                             ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
(1)  012266  012767  000007  166626                     MOV     #7,SHIFT        ;# OF SHIFTS
(1)  012274  012767  000124  167176                     MOV     #124,$TMP1      ;DATA CHAR
(1)  012302  004767  004616                             JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)  012306  012700  140012                             MOV     #140000!12,R0   ;EXPECTED DATA PLUS
(1)                                                                     ;RXERR & OVRRUN
(1)  012312  017701  167400                             MOV     @RXDBUF,R1      ;ACTUAL
(1)  012316  020001                                     CMP     R0,R1   ;COMPARE EXP VS. ACT
(1)  012320  001401                                     BEQ     .+4
(1)  012322  104002                                     ERROR   2       ;SPECIFICALLY LOOK AT RXERR &
(1)                                                                     ;OVRRUN BITS...THEY BOTH SHOULD BE SET
(1)
8545
8546
(2)                                             ;END OF PASS
(2)                                             ;TYPE NAME OF TEST
(2)                                             ;UPDATE PASS COUNT
(2)                                             ;CHECK FOR EXIT TO ACT-11
```

```
(2)                                             ;RESTART TEST
(2)
(2)  012324  000004                     .EOP:   SCOPE
(2)  012326  004767  000340                     JSR     PC,CKSWR
(2)  012332  104401                             TYPE                                    ;TYPE NAME OF TEST
(2)  012334  015463                             MEPASS
(2)  012336  104413  012570                     CONVRT  ,OUTCRY
(2)  012342  104401  015302                     TYPE    ,DEVICE
(2)  012346  105767  166600                     1STB    MULTD   ;ARE YOU RUNNING MULTIPLE DEVICES ?
(2)  012352  001511                             BEQ     CCC     ;NO,JUMP AROUND
(2)  012354  005767  166606                     TST     ACTREG  ;ARE ANY DEVICES ACTIVE ?
(2)  012360  001007                             BNE     RUNIT   ;YES
(2)  012362  104401  015314                     TYPE    ,MCOW   ;NO
(2)  012366  016700  166574                     MOV     ACTREG,R0       ;DISPLAY ACTREG
(2)  012372  000000                             HALT            ;SELECT SOMETHING TO RUN @ ACTREG:
(2)                                             ;SELECT SWITCHES & HIT CONTINUE (PUT SW00 =1)
(2)  012374  000167  167552                     JMP     .START  ;START OVER AGAIN.....YOU DESELECTED EVERYTHING
(2)  012400  062767  000010  166546  RUNI1:    ADD     #10,BASEADD     ;NEXT BLOCK (ADDRESSES)
(2)  012406  062767  000010  166546  ZERO:     ADD     #10,BASEIV      ;NEXT BLOCK (VECTORS)
(2)  012414  000241                             CLC
(2)  012416  006167  166546                     ROL     ROTADD  ;UP DATE ROTATING POINTER
(2)  012422  103410                             BCS     2$      ;IS IT THE LAST DEVICE
(2)                                             ;TO BE TESTED IN THIS PASS ?
(2)  012424  036767  166540  166534             BIT     ROTADD,ACTREG   ;TEST THIS DEVICE FOR ACTIVE STATUS
(2)  012432  001762                             BEQ     RUNIT   ;IF NOT ACTIVE, TRY NEXT ADDRESS
(2)  012434  004767  000034                     JSR     PC,REPLAY       ;CALCULATE NEW PARAMETERS
(2)  012440  000167  000210                     JMP     RESTRT  ;YES IT WAS ACTIVE,TEST THIS DEVICE
(2)  012444  012767  000001  166516  2$:       MOV     #1,ROTADD       ;OK!,NOW SET UP  ROTATING
(2)                                             ;POINTER FOR NEXT MULTIPLE PASS
(2)  012452  016767  166500  166474             MOV     KEEPADD,BASEADD ;RESTORE BASE ADDRESS
(2)  012460  016767  166500  166474             MOV     KEEPIV,BASEIV   ;RESTORE BASE INTERRUPT VECTORS
(2)  012466  004767  000002                     JSR     PC,REPLAY       ;CALC NEW PARAMETERS
(2)  012472  000441                             BR      CCC     ;JUMP AROUND REPLAY
(2)  012474  016767  166454  004420  REPLAY:   MOV     BASEADD,DUBASE  ;SET UP FOR NEW ADDRESSES
(2)  012502  004767  004262                     JSR     PC,DUADDR       ;CREATE NEW ADDRESSES
(2)  012506  016767  166450  167222             MOV     BASEIV,DURIV    ;CREATE DURIV
(2)  012514  062767  000002  166440             ADD     #2,BASEIV
(2)  012522  016767  166434  167210             MOV     BASEIV,DURIS    ;CREATE DURIS
(2)  012530  062767  000002  166424             ADD     #2,BASEIV
(2)  012536  016767  166420  167176             MOV     BASEIV,DUTIV    ;CREATE DUTIV
(2)  012544  062767  000002  166410             ADD     #2,BASEIV
(2)  012552  016767  166404  167164             MOV     BASEIV,DUTIS    ;CREATE DUTIS
(2)  012560  016767  167152  166374             MOV     DURIV,BASEIV    ;RESTORE
(2)  012566  000207                             RTS     PC
(2)
(2)  012570  000001                     OUTCRY: 1
(2)  012572     006     002                     .BYTE   6,2
(2)  012574  001712                             RXCSR
(2)
(2)  012576                              CCC:
(2)  012576  005067  166600                     CLR     $TSTNM                  ;CLEAR TEST NUMBER
(2)  012602  005067  166610                     CLR     $ERRPC                  ;CLEAR LAST ERROR PC
(2)  012606  005067  166571                     CLR     $ERFLG                  ;CLEAR ERROR FLAG
(2)  012612  005267  166274                     INC     PASCNT                  ;UPDATE PASS COUNT
(2)  012616  016767  166270  166256             MOV     PASCNT,LIGHTS           ;DISPLAY PASS COUNT
(2)  012624  016767  166262  166702             MOV     PASCNT,$PASS            ;PASS COUNT TO APT
```

```
(2)  012632  013701  000042            MOV     @#42,R1                 ;CHECK FOR ACT-11 OR DDP
(2)  012636  001406                    BEQ     RESTRT          ;IF NO CONTINUE TESTING
(2)  012640  000005                    RESET
(2)  012642  000005                    RESET
(2)  012644  004711            $ENDAD: JSR     PC,(R1)
(2)  012646  000240                    NOP
(2)  012650  000240                    NOP
(2)  012652  000240                    NOP
(2)  012654                    RESTRT:
(2)  012654  012767  003376  166524    MOV     #TST1+2,$LPADR  ;LOAD LAST ADDR
(2)  012662  004767  000004            JSR     PC,CKSWR
(2)  012666  000167  170416            JMP     .BEGIN
(2)
(2)                                    ;CHECK SWITCH REGISTER ROUTINE.
(2)                                    ;CHECKS TO ALLOW FOR <^G> TO ALLOW
(2)                                    ;THE CHANGING OF LOCATION 176
(2)
(2)  012672  005737  000042    CKSWR:  TST     @#42
(2)  012676  001040                    BNE     OUT
(2)  012700  022767  000176  166532    CMP     #SWREG,SWR      ;SOFTWARE SWR PRESENT?
(2)  012706  001034                    BNE     OUT             ;NO--LEAVE
(2)  012710  105777  166530            TSTB    @$TKS           ;CHECK TTY READY
(2)  012714  100031                    BPL     OUT             ;NO--LEAVE
(2)  012716  017767  166524  000422    MOV     @$TKB,.MSG      ;GET CHARACTER
(2)  012724  042767  177600  000414    BIC     #177600,.MSG    ;STRIP JUNK
(2)  012732  122767  000007  000406    CMPB    #7,.MSG         ;IS IT <^G> ?
(2)  012740  001017                    BNE     OUT             ;NO
(2)  012742  104401  016070            TYPE    ,MCNTG
(2)  012746  005137  013006    CNTLU:  COM     @#RDSW
(2)  012752  104401  016100            TYPE    ,MMSWR
(2)  012756  104413                    CONVRT
(2)  012760  013010                    SWREGL
(2)  012762  104406  016111            INSTR,MMNEW
(2)  012766  104410                    PARAM
(2)  012770  000000                    0
(2)  012772  177777                    177777
(2)  012774  000176                    SWREG
(2)  012776     000     001    .BYTE   0,1
(2)  013000  005037  013006    OUT:    CLR     @#RDSW
(2)  013004  000207                    RTS     PC
(2)  013006  000000            RDSW:   .WORD   0
(2)  013010  000001            SWREGL: 1
(2)  013012     006     002            .BYTE   6,2
(2)  013014  000176                    SWREG
(2)
(1)  013016  000005                    5
(2)
(2)                                    ;CHECK FOR FREEZE ON CURRENT DATA
(2)
(2)  013020  004767  177646    .SCOP1: JSR     PC,CKSWR
(2)  013024  032777  001000  166406    BIT     #SW09,@SWR
(2)  013032  001402                    BEQ     1$
(2)  013034  016716  166050            MOV     LOCK,(SP)
(2)  013040  000002            1$:     RTI
(2)                            .SBTTL  TYPE ROUTINE
(2)
```

```
(3)                                     ;;*****************************************************************
(2)                                     ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
(2)                                     ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
(2)                                     ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
(2)                                     ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
(2)                                     ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
(2)                                     ;*
(2)                                     ;*CALL:
(2)                                     ;*1) USING A TRAP INSTRUCTION
(2)                                     ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
(2)                                     ;*OR
(2)                                     ;*      TYPE
(2)                                     ;*      MESADR
(2)                                     ;*
(2)
(2)  013042  105767  166411     $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(2)  013046  100002                     BPL     1$              ;;BR IF YES
(2)  013050  000000                     HALT                    ;;HALT HERE IF NO TERMINAL
(2)  013052  000430                     BR      3$              ;;LEAVE
(2)  013054  010046              1$:    MOV     R0,-(SP)        ;;SAVE R0
(2)  013056  017600  000002             MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(2)  013062  122767  000001  166456     CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(2)  013070  001011                     BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(2)  013072  132767  000100  166447     BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(2)  013100  001405                     BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(2)  013102  010067  000004             MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(2)  013106  004767  164674             JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(2)  013112  000000              61$:   .WORD   0               ;;MESSAGE ADDRESS
(2)  013114  132767  000040  166425  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(2)  013122  001003                     BNE     60$             ;;YES,SKIP TYPE OUT
(2)  013124  112046              2$:    MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(2)  013126  001005                     BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(2)  013130  005726                     TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(2)  013132  012600              60$:   MOV     (SP)+,R0        ;;RESTORE R0
(2)  013134  062716  000002      3$:    ADD     #2,(SP)         ;;ADJUST RETURN PC
(2)  013140  000002                     RTI                     ;;RETURN
(2)  013142  122716  000011      4$:    CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(2)  013146  001430                     BEQ     8$
(2)  013150  122716  000200             CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(2)  013154  001006                     BNE     5$
(2)  013156  005726                     TST     (SP)+           ;;POP  <CR><LF> EQUIV
(2)  013160  104401                     TYPF                    ;;TYPE A CR AND LF
(2)  013162  001523                     $CRLF
(2)  013164  105067  000130             CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(2)  013170  000755                     BR      2$              ;;GET NEXT CHARACTER
(2)  013172  004767  000056      5$:    JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(2)  013176  126726  166254      6$:    CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(2)  013202  001350                     BNE     2$              ;;IF NO GO GET NEXT CHAR.
(2)  013204  016746  166244             MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(2)                                     ;;AND THE NULL CHAR.
(2)  013210  105366  000001      7$:    DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(2)  013214  002770                     BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(2)  013216  004767  000032             JSR     PC,$TYPEC       ;;GO TYPE A NULL
(2)  013222  105367  000072             DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(2)  013226  000770                     BR      7$              ;;LOOP
(2)
```

```
(2)                                     ;HORIZONTAL TAB PROCESSOR
(2)
(2)  013230  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(2)  013234  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(2)  013240  132767  000007  000052          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(2)  013246  001372                          BNE     9$              ;;TAB STOP
(2)  013250  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
(2)  013252  000724                          BR      2$              ;;GET NEXT CHARACTER
(2)  013254  105777  166170          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(2)  013260  100375                          BPL     $TYPEC
(2)  013262  116677  000002  166162          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(2)
(2)  013270  122766  000015  000002  CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(2)  013276  001003                          BNE     1$              ;;BRANCH IF NO
(2)  013300  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(2)  013304  000406                          BR      $TYPEX          ;;EXIT
(2)  013306  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(2)  013314  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
(2)  013316  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
(2)  013320  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(2)  013322  000207                  $TYPEX: RTS     PC
(2)
(2)
(2)
(2)
(2)                                             ;ASCII STRING INPUT ROUTINE
(2)
(2)  013324  017667  000000  000014  .INSTR: MOV     @(SP),.MSG      ;PICK UP MESSAGE
(2)  013332  062716  000002                  ADD     #2,(SP)         ;JUMP AROUND MESSAGE FOR RTI
(2)  013336  105767  166204                  TSTB    $ENV            ;APT CONTROL
(2)  013342  001036                          BNE     INSTR2          ;YES NO TYPE
(2)  013344  104401                  .INST1: TYPE
(2)  013346  000000                  .MSG:   0
(2)  013350  012704  016124                  MOV     #INBUF,R4       ;GET STARTING LOC OF INBUF
(2)  013354  012703  000007                  MOV     #7,R3           ;MAX # OF CHARS
(2)  013360  105777  166060          1$:     TSTB    @$TKS   ;TTY FLAG
(2)  013364  100375                          BPL     1$
(2)  013366  117714  166054                  MOVB    @$TKB,(R4)      ;TAKE CHAR
(2)  013372  142714  000200                  BICB    #200,(R4)       ;STRIP
(2)  013376  121427  000025                  CMPB    (R4),#25        ;IS IT <^G>
(2)  013402  001760                          BEQ     .INST1
(2)  013404  122427  000015                  CMPB    (R4)+,#15       ;CHECK FOR CR
(2)  013410  001413                          BEQ     INSTR2
(2)  013412  105777  166032          2$:     TSTB    @$TPS   ;TEST FLAG
(2)  013416  100375                          BPL     2$
(2)  013420  117777  166022  166024          MOVB    @$TKB,@$TPB     ;ECHO CHARACTER
(2)  013426  005303                          DEC     R3      ;DID YOU TYPE TOO MANY CHARS ?
(2)  013430  001353                          BNE     1$
(2)  013432  104401                  .INSTE: TYPE
(2)  013434  015410                          MQM     ;?
(2)  013436  000742                          BR      .INST1  ;RETRY
(2)  013440  000002                  INSTR2: RTI
(2)
(2)                                     ;CONVERT ASCII STRING TO OCTAL
(2)
(2)  013442  011605                  .PARAM: MOV     (SP),R5 ;PUT CONTENTS OF SP INTO R5
```

```
(2)  013444  012567  000162                MOV     (R5)+,LOLIM     ;PUT LOW LIMIT INTO LOLIM
(2)  013450  012567  000160                MOV     (R5)+,HILIM     ;PUT HIGH LIMIT INTO HILIM
(2)  013454  012567  000156                MOV     (R5)+,DEVADR    ;PUT STORE LOC INTO DEVADR
(2)  013460  112567  000154                MOVB    (R5)+,LOBITS    ;PUT MASK INTO LOBITS
(2)  013464  112567  000151                MOVB    (R5)+,ADRCNT    ;PUT COUNT INTO ADRCNT
(2)  013470  010516                        MOV     R5,(SP) ;RESTORE RETURN ADDR ON STACK FOR RTI
(2)  013472  005005                PARAM1: CLR     R5
(2)  013474  012704  016124                MOV     #INBUF,R4
(2)  013500  122714  000015                CMPB    #15,(R4)        ;CR ?
(2)  013504  001420                        BEQ     PARERR  ;YOU TYPED CR TOO SOON !
(2)  013506  121427  000060        1$:     CMPB    (R4),#60        ;LOW LIMIT ASCII 0
(2)  013512  002415                        BLT     PARERR
(2)  013514  121427  000067                CMPB    (R4),#67        ;HIGH LIMIT ASCII 7
(2)  013520  003012                        BGT     PARERR
(2)  013522  142714  000060                BICB  - #60,(R4)        ;CONVERT TO OCTAL
(2)  013526  152405                        BISB    (R4)+,R5        ;STORE AWAY ITS AN OK CHAR
(2)  013530  122714  000015                CMPB    #15,(R4)        ;CR ?
(2)  013534  001414                        BEQ     LIMITS  ;NOW CHECK FOR HIGH &LOW LIMIT CONDS
(2)  013536  006305                        ASL     R5      ;ALLOCATE ROOM FOR NEXT CHAR
(2)  013540  006305                        ASL     R5
(2)  013542  006305                        ASL     R5
(2)  013544  000760                        BR      1$
(2)  013546  122714  000015        PARERR: CMPB    #15,(R4)        ;CR?
(2)  013552  001003                        BNE     120$
(2)  013554  005737  013006                TST     @#RDSW          ·CK SWR USED
(2)  013560  001023                        BNE     PARTI
(2)  013562  104407                120$:   INSTER  ;RETRY
(2)  013564  000742                        BR      PARAM1
(2)
(2)                                        ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
(2)
(2)  013566  020567  000042        LIMITS: CMP     R5,HILIM
(2)  013572  101365                        BHI     PARERR  ;THE # IS TOO HIGH
(2)  013574  020567  000032                CMP     R5,LOLIM
(2)  013600  103762                        BLO     PARERR  ;THE # IS TOO LOW
(2)  013602  136705  000032                BITB    LOBITS,R5       ;TEST BY MASKINGTHE #
(2)  013606  001357                        BNE     PARERR
(2)
(2)                                        ;STORE NUMBER AT SPECIFIED ADDRESS
(2)
(2)  013610  016704  000022                MOV     DEVADR,R4       ;GET STARTING ADDR OF
(2)  013614  010524                1$:     MOV     R5,(R4)+        ;STORE AT THIS ADDR
(2)  013616  062705  000002                ADD     #2,R5
(2)  013622  105367  000013                DECB    ADRCNT  ;HOW MANY TIMES + 2 ?
(2)  013626  001372                        BNE     1$
(2)  013630  000002                PARTI:  RTI
(2)  013632  000000                LOLIM:  0
(2)  013634  000000                HILIM:  0
(2)  013636  000000                DEVADR: 0
(2)  013640  000000                LOBITS: 0
(2)          013641                ADRCNT=LOBITS+1
(2)
(2)                                        ;SAVE PC OF TEST THAT FAILED AND R0-R5
(2)
(2)  013642  016667  000004  165256 .SAV05: MOV     4(SP),SAVPC
(2)
```

```
(2)                                     ;SAVE R0-R5
(2)
(2)  013650  010567  165620     SV05:   MOV     R5,$REG5
(2)  013654  010467  165612             MOV     R4,$REG4
(2)  013660  010367  165604             MOV     R3,$REG3
(2)  013664  010267  165576             MOV     R2,$REG2
(2)  013670  010167  165570             MOV     R1,$REG1
(2)  013674  010067  165562             MOV     R0,$REG0
(2)  013700  000002                     RTI
(2)
(2)                                     ;RESTORE R0-R5
(2)
(2)  013702  016700  165554     .RES05: MOV     $REG0,R0
(2)  013706  016701  165552             MOV     $REG1,R1
(2)  013712  016702  165550             MOV     $REG2,R2
(2)  013716  016703  165546             MOV     $REG3,R3
(2)  013722  016704  165544             MOV     $REG4,R4
(2)  013726  016705  165542             MOV     $REG5,R5
(2)  013732  000002                     RTI
(2)
(2)                                     ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
(2)
(2)  013734  104401             .CONVR: TYPE
(2)  013736  015414                     MCRLF   ;CR  LF
(2)  013740  017601  000000             MOV     @(SP),R1        ;PICK UP DATA POINTER
(2)  013744  062716  000002             ADD     #2,(SP) ;SET UP SP FOR RTI
(2)  013750  012167  000130             MOV     (R1)+,WRDCNT    ;PICK UP # OF WORDS FROM TABLE
(2)  013754  112167  000126     1$:     MOVB    (R1)+,CHRCNT    ;PICK UP # OF CHARS FROM TABLE
(2)  013760  112167  000123             MOVB    (R1)+,SPACNT    ;PICK UP # OF SPACES FROM TABLE
(2)  013764  013167  000120             MOV     @(R1)+,BINWRD   ;PICK UP ADDRESS OF  MSG
(2)                                                             ;FROM TABLE
(2)  013770  016704  000114     2$:     MOV     BINWRD,R4       ;SAVE
(2)  013774  116705  000106             MOVB    CHRCNT,R5       ;SAVE
(2)  014000  012700  016166             MOV     #TEMP,R0        ;STARTING ADDRESS OF TEMP BLOCK
(2)  014004  010403             3$:     MOV     R4,R3           ;SAVE
(2)  014006  042703  177770             BIC     #177770,R3      ;CLR OUT UPPER BITS .. SAVE CHAR
(2)  014012  062703  000260             ADD     #260,R3 ;CONVERT TO ASCII
(2)  014016  110320                     MOVB    R3,(R0)+        ;STORE AWAY
(2)  014020  006204                     ASR     R4      ;SHIFT FOR NEXT #
(2)  014022  006204                     ASR     R4      ;DITTO
(2)  014024  006204                     ASR     R4      ;DITTO
(2)  014026  005305                     DEC     R5      ;DEC CHAR COUNT
(2)  014030  001365                     BNE     3$      ;DO IT AGAIN ?
(2)  014032  012703  016230             MOV     #MDATA,R3       ;STARTING ADDRESS OF MDATA BLOCK
(2)  014036  114023             4$:     MOVB    -(R0),(R3)+     ;REVERSE THE ORDER OF NUMBERS
(2)  014040  105367  000042             DECB    CHRCNT  ;DEC CHAR COUNT
(2)  014044  001374                     BNE     4$      ;DO IT AGAIN ?
(2)  014046  105767  000035             TSTB    SPACNT  ;HOW MANY SPACES ?
(2)  014052  001405                     BEQ     6$      ;TYPE # IF BR =0
(2)  014054  112723  000240     5$:     MOVB    #240,(R3)+      ;"SPACE" IN ASCII
(2)  014060  105367  000023             DECB    SPACNT  ;DEC # OF SPACE COUNT
(2)  014064  001373                     BNE     5$      ;DO IT AGAIN ?
(2)  014066  105013             6$:     CLRB    (R3)    ;INSERT "0" FOR TTY OUTPUT ROUTINE
(2)  014070  104401                     TYPE
(2)  014072  016230                     MDATA   ;THIS MESSAGE
(2)  014074  005367  000004             DEC     WRDCNT  ;HOW MANY #'S ?
```

```
(2)  014100  001325                    BNE     1$      ;DO THIS ROUTINE AGAIN IF NOT EQUAL TO 0
(2)  014102  000002                    RTI     ;RETURN TO PROGRAM
(2)  014104  000000                WRDCNT: 0
(2)  014106  000000                CHRCNT: 0
(2)          014107                SPACNT=CHRCNT+1
(2)  014110  000000                BINWRD: 0
(2)
(2)                                        ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
(2)                                        ;BUFFER TO THE CHARACTERS 'N'' AND 'Y''.
(2)                                        ;IF THE CHARACTER IS 'N' CLEAR THE FLAG
(2)                                        ;IF THE CHARACTER IS 'Y' SET THE FLAG
(2)
(2)  014112  017605  000000        .SETFLG:MOV     @(SP),R5
(2)  014116  122767  000116  002000        CMPB    #'N,INBUF       ;IS IT 'N' ?
(2)  014124  001002                        BNE     1$
(2)  014126  105015                        CLRB    (R5)    ;000
(2)  014130  000406                        BR      2$
(2)  014132  122767  000131  001764  1$:    CMPB    #'Y,INBUF       ;IS IT 'Y' ?
(2)  014140  001005                        BNE     3$
(2)  014142  112715  177777                MOVB    #-1,(R5)        ;377
(2)  014146  062716  000002        2$:     ADD     #2,(SP)
(2)  014152  000002                        RTI
(2)  014154  104407                3$:     INSTER  ;RETRY
(2)  014156  000755                        BR      .SETFLG
(2)                                .SBTTL  ERROR HANDLER ROUTINE
(2)
(3)                                ;;**********************************************************************
(2)                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
(2)                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(2)                                ;*AND GO TO SAVIT ON ERROR
(2)                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(2)                                ;*SW15=1        HALT ON ERROR
(2)                                ;*SW13=1        INHIBIT ERROR TYPEOUTS
(2)                                ;*SW10=1        BELL ON ERROR
(2)                                ;*SW09=1        LOOP ON ERROR
(2)                                ;*CALL
(2)                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(2)
(2)  014160                        $ERROR:
(2)  014160  105267  165217        7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
(2)  014164  001775                        BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(2)  014166  016777  165210  165246        MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(2)  014174  032777  002000  165236        BIT     #BIT10,@SWR     ;;BELL ON ERROR?
(2)  014202  001402                        BEQ     1$              ;;NO - SKIP
(2)  014204  104401  001516                TYPE    ,$BELL          ;;RING BELL
(2)  014210  005267  165176        1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
(2)  014214  011667  165176                MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(2)  014220  162767  000002  165170        SUB     #2,$ERRPC
(2)  014226  117767  165164  165160        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(2)  014234  032777  020000  165176        BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(2)  014242  001004                        BNE     20$             ;;SKIP TYPEOUTS
(2)  014244  004767  000072                JSR     PC,SAVIT        ;;GO TO USER ERROR ROUTINE
(2)  014250  104401  001523                TYPE    ,$CRLF
(2)  014254                        20$:
(2)  014254  122767  000001  165264        CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(2)  014262  001007                        BNE     2$              ;;NO,SKIP APT ERROR REPORT
```

```
(2)  014264  116767  165124  000004          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(2)  014272  004767  163520                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
(2)  014276     000                  21$:    .BYTE   0
(2)  014277     000                          .BYTE   0
(2)  014300  000777                  22$:    BR      22$             ;;APT ERROR LOOP
(2)  014302  005777  165132          2$:     TST     @SWR            ;;HALT ON ERROR
(2)  014306  100001                          BPL     3$              ;;SKIP IF CONTINUE
(2)  014310  000000                          HALT                    ;;HALT ON ERROR!
(2)  014312  032777  001000  165120  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(2)  014320  001402                          BEQ     4$              ;;BR IF NO
(2)  014322  016716  165062                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(2)  014326  005767  165162          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(2)  014332  001402                          BEQ     5$              ;;BR IF NONE
(2)  014334  016716  165154                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(2)  014340                          5$:
(2)  014340  000002                          RTI                     ;;RETURN
(2)  014342  010067  164562          SAVIT:  MOV     R0,HLD0
(2)  014346  010167  164560                  MOV     R1,HLD1
(2)  014352  010267  164556                  MOV     R2,HLD2
(2)  014356  010367  164554                  MOV     R3,HLD3
(2)  014362  010467  164552                  MOV     R4,HLD4
(2)  014366  010567  164550                  MOV     R5,HLD5
(2)  014372  016767  165004  164544          MOV     $TSTNM,HLD6
(2)
(3)                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
(3)
(4)                                  ;;**********************************************************************
(3)                                  ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
(3)                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
(3)                                  ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
(3)
(3)  014400                          $ERRTYP:
(3)  014400  104401  001523                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(3)  014404  010046                          MOV     R0,-(SP)        ;;SAVE R0
(3)  014406  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
(3)  014410  153700  001414                  BISB    @#$ITEMB,R0
(3)  014414  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(3)                                                                  ;;TYPE THE PC OF THE ERROR
(4)  014416  016746  164774                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(4)                                                                  ;;ERROR ADDRESS
(4)  014422  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(3)  014424  000426                          BR      6$              ;;GET OUT
(3)  014426  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(3)  014430  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
(3)  014432  006300                          ASL     R0
(3)  014434  006300                          ASL     R0
(3)  014436  062700  001652                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
(3)  014442  012067  000004                  MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(3)  014446  001404                          BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(3)  014450  104401                          TYPE                    ;;TYPE THE "ERROR MESSAGE"
(3)  014452  000000                  2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(3)  014454  104401  001523                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(3)  014460  012067  000004          3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
(3)  014464  001404                          BEQ     5$              ;;SKIP TYPEOUT IF 0
(3)  014466  104401                          TYPE                    ;;TYPE THE "DATA HEADER"
(3)  014470  000000                  4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
```

```
(3)  014472  104401  001523                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(3)  014476  011000                5$:     MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
(3)  014500  001004                        BNE     7$              ;;GO TYPE THE DATA
(3)  014502  012600                6$:     MOV     (SP)+,R0        ;;RESTORE R0
(3)
(3)  014504  104401  001523                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(3)  014510  000207                        RTS     PC              ;;RETURN
(3)  014512                        7$:
(4)  014512  013046                        MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
(4)  014514  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(3)  014516  005710                        TST     (R0)            ;;IS THERE ANOTHER NUMBER?
(3)  014520  001770                        BEQ     6$              ;;BR IF NO
(3)  014522  104401  014530                TYPE    ,8$             ;;TYPE TWO(2) SPACES
(3)  014526  000771                        BR      7$              ;;LOOP
(3)  014530  020040     000        8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
(3)          014534                        .EVEN
(2)                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(2)
(3)                                ;;**************************************************************
(2)                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(2)                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(2)                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
(2)                                ;*CALL:
(2)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(2)                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
(2)                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
(2)                                ;*      .BYTE   M               ;;M=1 OR 0
(2)                                ;*                                      ;;1=TYPE LEADING ZEROS
(2)                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
(2)                                ;*
(2)                                ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
(2)                                ;*$TYPOS OR $TYPOC
(2)                                ;*CALL:
(2)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(2)                                ;*      TYPON                   ;;CALL FOR TYPEOUT
(2)                                ;*
(2)                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
(2)                                ;*CALL:
(2)                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
(2)                                ;*      TYPOC                   ;;CALL FOR TYPEOUT
(2)
(2)  014534  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(2)  014540  116667  000001  000211        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(2)  014546  112667  000207                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(2)  014552  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(2)  014556  000406                        BR      $TYPON
(2)  014560  112767  000001  000171  $TYPOC: MOVB   #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(2)  014566  112767  000006  000165        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(2)  014574  112767  000005  000154  $TYPON: MOVB   #5,$OCNT        ;;SET THE ITERATION COUNT
(2)  014602  010346                        MOV     R3,-(SP)        ;;SAVE R3
(2)  014604  010446                        MOV     R4,-(SP)        ;;SAVE R4
(2)  014606  010546                        MOV     R5,-(SP)        ;;SAVE R5
(2)  014610  116704  000145                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(2)  014614  005404                        NEG     R4
(2)  014616  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(2)  014622  110467  000132                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
```

```
(2)  014626  116704  000125              MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(2)  014632  016605  000012              MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(2)  014636  005003                      CLR     R3              ;;CLEAR THE OUTPUT WORD
(2)  014640  006105                1$:   ROL     R5              ;;ROTATE MSB INTO "C"
(2)  014642  000404                      BR      3$              ;;GO DO MSB
(2)  014644  006105                2$:   ROL     R5              ;;FORM THIS DIGIT
(2)  014646  006105                      ROL     R5
(2)  014650  006105                      ROL     R5
(2)  014652  010503                      MOV     R5,R3
(2)  014654  006103                3$:   ROL     R3              ;;GET LSB OF THIS DIGIT
(2)  014656  105367  000076              DECB    $OMODE          ;;TYPE THIS DIGIT?
(2)  014662  100016                      BPL     7$              ;;BR IF NO
(2)  014664  042703  177770              BIC     #177770,R3      ;;GET RID OF JUNK
(2)  014670  001002                      BNE     4$              ;;TEST FOR 0
(2)  014672  005704                      TST     R4              ;;SUPPRESS THIS 0?
(2)  014674  001403                      BEQ     5$              ;;BR IF YES
(2)  014676  005204                4$:   INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(2)  014700  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(2)  014704  052703  000040        5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(2)  014710  110367  000040              MOVB    R3,8$           ;;SAVE FOR TYPING
(2)  014714  104401  014754              TYPE    ,8$             ;;GO TYPE THIS DIGIT
(2)  014720  105367  000032        7$:   DECB    $OCNT           ;;COUNT BY 1
(2)  014724  003347                      BGT     2$              ;;BR IF MORE TO DO
(2)  014726  002402                      BLT     6$              ;;BR IF DONE
(2)  014730  005204                      INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(2)  014732  000744                      BR      2$              ;;GO DO THE LAST DIGIT
(2)  014734  012605                6$:   MOV     (SP)+,R5        ;;RESTORE R5
(2)  014736  012604                      MOV     (SP)+,R4        ;;RESTORE R4
(2)  014740  012603                      MOV     (SP)+,R3        ;;RESTORE R3
(2)  014742  016666  000002  000004      MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(2)  014750  012616                      MOV     (SP)+,(SP)
(2)  014752  000002                      RTI                     ;;RETURN
(2)  014754     000                8$:   .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(2)  014755     000                      .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(2)  014756     000                $OCNT:.BYTE   0               ;;OCTAL DIGIT COUNTER
(2)  014757     000                $OFILL:.BYTE  0               ;;ZERO FILL SWITCH
(2)  014760  000000                $OMODE:.WORD  0               ;;NUMBER OF DIGITS TO TYPE
(2)                                      ;ENTER HERE ON POWER FAILURE
(2)
(2)
(2)  014762                        $PWRDN:
(2)  014762  010046                .PFAIL: MOV     R0,-(SP)                ;SAVE R0-R5 ON PROCESSOR STACK
(2)  014764  010146                        MOV     R1,-(SP)
(2)  014766  010246                        MOV     R2,-(SP)
(2)  014770  010346                        MOV     R3,-(SP)
(2)  014772  010446                        MOV     R4,-(SP)
(2)  014774  010546                        MOV     R5,-(SP)
(2)  014776  016746  163022               MOV     24,-(SP)
(2)  015002  010667  164110               MOV     SP,SAVSP                ;SAVE STACK POINTER
(2)  015006  012767  015020  163010       MOV     #RESTART,24             ;SET UP FOR POWER UP TRAP
(2)  015014  000000                       HALT                            ;HALT ON POWER DOWN NORMAL
(2)  015016  000777                       BR      .
(2)
(2)                                       ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
(2)
(2)  015020  016706  164072        RESTAR: MOV     SAVSP,SP                ;RESTORE STACK POINTER
```

```
(2)  015024  012605                        MOV     (SP)+,R5                ;RESTORE R0-R5
(2)  015026  012604                        MOV     (SP)+,R4
(2)  015030  012603                        MOV     (SP)+,R3
(2)  015032  012602                        MOV     (SP)+,R2
(2)  015034  012601                        MOV     (SP)+,R1
(2)  015036  012600                        MOV     (SP)+,R0
(2)  015040  012767  014762  162756        MOV     #.PFAIL,24              ;SET UP FOR POWER FAILURE
(2)  015046  106427  000300                MTPS    #300
(2)  015052  012706  001100                MOV     #STACK,SP
(2)  015056  005067  001104                CLR     TEMP
(2)  015062  005267  001100                INC     TEMP
(2)  015066  001375                        BNE     .-4
(2)  015070  104413                        CONVRT
(2)  015072  015114                        PFTAB
(2)  015074  104401                        TYPE
(2)  015076  015417                        MPFAIL
(2)  015100  005067  164277                CLR     $ERFLG
(2)  015104  005067  164306                CLR     $ERRPC
(2)  015110  000177  163770                JMP     @RETURN
(2)  015114  000001                PFTAB:  1
(2)  015116     006     002                .BYTE   6,2
(2)  015120  000207                        RETURN
(2)  015122  005015  041412  042116  MTITLE: .ASCIZ  <15><12><12>/CNDUQ-A0 DUV11 TAPE A /<15><12>
(2)  015130  050525  040455  020060
(2)  015136  052504  030526  020061
(2)  015144  040524  042520  040440
(2)  015152  006440  000012
(2)  015156  005015  042526  020103  MVECTO: .ASCIZ  <15><12>/VEC ADD-/
(2)  015164  042101  026504     000
(2)  015171     015  030412  052123  MREGAD: .ASCIZ  <15><12>/1ST DEV: REC CSR ADD-/
(2)  015176  042040  053105  020072
(2)  015204  042522  020103  051503
(2)  015212  020122  042101  026504
(2)  015220     000
(2)  015221     015  046412  046125  MMULT:  .ASCIZ  <15><12>/MULT DEV ? (Y OR N)-/
(2)  015226  020124  042504  020126
(2)  015234  020077  054450  047440
(2)  015242  020122  024516  000055
(2)  015250  005015  040514  052123  MLASTD: .ASCIZ  <15><12>/LAST DEV: REC CSR ADDR-/
(2)  015256  042040  053105  020072
(2)  015264  042522  020103  051503
(2)  015272  020122  042101  051104
(2)  015300  000055
(2)  015302  042075  053105  041511  DEVICE: .ASCIZ  /=DEVICE  /
(2)  015310  020105  000040
(2)  015314  005015  042523  042514  MCOW:   .ASCIZ  <15><12>/SELECT TO RUN @ACTREG/
(2)  015322  052103  052040  020117
(2)  015330  052522  020116  040500
(2)  015336  052103  042522  000107
(2)  015344  005015  053117  046106  MRANGE: .ASCIZ  <15><12>/OVFLO:RETYPE LAST DEV RXCSR ADDS-/
(2)  015352  035117  042522  054524
(2)  015360  042520  046040  051501
(2)  015366  020124  042504  020126
(2)  015374  054122  051503  020122
(2)  015402  042101  051504  000055
(2)  015410  020040  000077            MQM:    .ASCIZ  /  ?/
```

```
(2)  015414  005015     000          MCRLF:  .ASCIZ  <15><12>
(2)  015417     120  040506  046111  MPFAIL: .ASCIZ  /PFAIL,  RESTART AT TEST IN PROGRESS/
(2)  015424  020054  051040  051505
(2)  015432  040524  052122  040440
(2)  015440  020124  042524  052123
(2)  015446  044440  020116  051120
(2)  015454  043517  042522  051523
(2)  015462     000
(2)  015463     015  042412  042116  MEPASS: .ASCIZ  <15><12>/END OF PASS TAPE A/
(2)  015470  047440  020106  040520
(2)  015476  051523  052040  050101
(2)  015504  020105  000101
(2)  015510  005015  000122          MR:     .ASCIZ  <15><12>/R/
(2)  015514  005015  042524  052123  MTSTPC: .ASCIZ  <15><12>/TEST PC-/
(2)  015522  050040  026503     000
(2)  015527     015  046012  041517  MLOCK:  .ASCIZ  <15><12>/LOCK ON  TEST? (Y OR N)-/
(2)  015534  020113  047117  020040
(2)  015542  042524  052123  020077
(2)  015550  054450  047440  020122
(2)  015556  024516  000055
(2)  015562  005015  020043  043117  MSYNC:  .ASCIZ  <15><12>/# OF SYNC CHARS SELECTED ( 1 OR 2)-/
(2)  015570  051440  047131  020103
(2)  015576  044103  051101  020123
(2)  015604  042523  042514  052103
(2)  015612  042105  024040  030440
(2)  015620  047440  020122  024462
(2)  015626  000055
(2)  015630  005015  051513  051440  MWIRE6: .ASCIZ  <15><12>/IS SEC XMIT SWITCH E55-2 IN? (Y OR N)-/
(2)  015636  041505  054040  044515
(2)  015644  020124  053523  052111
(2)  015652  044103  042440  032465
(2)  015660  031055  044440  037516
(2)  015666  024040  020131  051117
(2)  015674  047040  026451     000
(2)  015701     015  044412  020123  MWIRE5: .ASCIZ  <15><12>/IS SEC REC SWITCH E55-3 IN? (Y OR N)-/
(2)  015706  042523  020103  042522
(2)  015714  020103  053523  052111
(2)  015722  044103  042440  032465
(2)  015730  031455  044440  037516
(2)  015736  024040  020131  051117
(2)  015744  047040  026451     000
(2)  015751     015  044412  020123  MWIRE4: .ASCIZ  <15><12>/IS OPT CLR ENABLE SWITCH E55-1 IN? (Y OR N)-/
(2)  015756  050117  020124  046103
(2)  015764  020122  047105  041101
(2)  015772  042514  051440  044527
(2)  016000  041524  020110  032505
(2)  016006  026465  020061  047111
(2)  016014  020077  054450  047440
(2)  016022  020122  024516  000055
(2)  016030  005015  044012  030463  MEXTJ:  .ASCIZ  <15><12><12>/H315 CONNECTOR ON ?(Y OR N)-/
(2)  016036  020065  047503  047116
(2)  016044  041505  047524  020122
(2)  016052  047117  037440  054450
(2)  016060  047440  020122  024516
(2)  016066  000055
(2)  016070  005015  057040  020107  MCNTG:  .ASCIZ  <15><12>/ ^G  /
```

```
(2)  016076  000040
(2)  016100  051440  051127  020075  MMSWR:  .ASCIZ  / SWR=   /
(2)  016106  020040     000
(2)  016111     040  020040  042516  MMNEW:  .ASCIZ  /   NEW=  /
(2)  016116  036527  020040     000
(2)          016124                 .EVEN
(2)
(2)                                 ;BUFFERS FOR INPUT-OUTPUT
(2)
(2)  016124  000000                 INBUF:  0
(2)          016166                 .=.+40
(2)  016166  000000                 TEMP:   0
(2)          016230                 .=.+40
(2)  016230  000000                 MDATA:  0
(2)          016272                 .=.+40
(3)                                 .SBTTL  SCOPE HANDLER ROUTINESTARS
(3)                                 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
(3)                                 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
(3)                                 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
(3)                                 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(3)                                 ;*SW14=1        LOOP ON TEST
(3)                                 ;*SW11=1        INHIBIT ITERATIONS
(3)                                 ;*SW09=1        LOOP ON ERROR
(3)                                 ;*SW08=1        LOOP ON TEST IN SWR<7:0>
(3)                                 ;*CALL
(3)                                 ;*      SCOPE           ;;SCOPE=IOT
(3)
(3)  0*6272                         $SCOPE:
(5)
(5)                                 ;SCOPE LOOP AND INTERATION HANDLER
(5)
(5)  016272                         .SCOPE:
(5)  016272  004767  174374                 JSR     PC,CKSWR
(5)  016276  005067  163114                 CLR     $ERRPC          ;CLEAR LAST ERROR PC
(5)  016302  022716  003376                 CMP     #TST1+2,(SP)    ;IS SCOPE AT BEGINING OF TEST 1?
(5)  016306  001422                         BEQ     $XTSTR          ;YES NO LOOP.
(5)
(5)  016310  032777  040000  163122  TTST:  BIT     #BIT14,@SWR     ;THIS CODE IS FOR TESTING FOR BIT 14
(5)  016316  001412                         BEQ     1$              ;ON LSI WHICH SYSMAC CANNOT HANDLE
(5)  016320  016767  163056  163060        MOV     $TSTNM,$LPADR
(5)  016326  000406                         BR      1$
(5)  016330  105777  163110                 TSTB    @$TKS           ;KEYBOARD DONE?
(5)  016334  100123                         BPL     $OVER           ;BR IF NO
(5)  016336  017766  163104  177776        MOV     @$TKB,-2(SP)    ;CLEAR DONE BIT
(3)  016344  032777  040000  163066  1$:    BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(3)  016352  001114                         BNE     $OVER           ;;YES IF SW14=1
(3)                                 ;#####START OF CODE FOR THE XOR TESTER#####
(3)  016354  000416                 $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(3)                                                                 ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(3)  016356  013746  000004                 MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(3)  016362  012737  016402  000004        MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(3)  016370  005737  177060                 TST     @#177060        ;;TIME OUT ON XOR?
(3)  016374  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(3)  016400  000463                         BR      $SVLAD          ;;GO TO THE NEXT TEST
(3)  016402  022626                 5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(3)  016404  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
```

```
(3)  016410  000423                          BR      7$                ;;LOOP ON THE PRESENT TEST
(3)  016412                             6$:;#####END OF CODE FOR THE XOR TESTER#####
(3)  016412  032777  000400  163020          BIT     #BIT08,@SWR       ;;LOOP ON SPEC. TEST?
(3)  016420  001404                          BEQ     2$                ;;BR IF NO
(3)  016422  127767  163012  162752          CMPB    @SWR,$TSTNM       ;;ON THE RIGHT TEST?    SWR<7:0>
(3)  016430  001465                          BEQ     $OVER             ;;BR IF YES
(3)  016432  105767  162745          2$:     TSTB    $ERFLG            ;;HAS AN ERROR OCCURRED?
(3)  016436  001421                          BEQ     3$                ;;BR IF NO
(3)  016440  126767  162751  162735          CMPB    $ERMAX,$ERFLG     ;;MAX. ERRORS FOR THIS TEST OCCURRED?
(3)  016446  101015                          BHI     3$                ;;BR IF NO
(3)  016450  032777  001000  162762          BIT     #BIT09,@SWR       ;;LOOP ON ERROR?
(3)  016456  001404                          BEQ     4$                ;;BR IF NO
(3)  016460  016767  162724  162720  7$:     MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
(3)  016466  000446                          BR      $OVER
(3)  016470  105067  162707          4$:     CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
(3)  016474  005067  163012                  CLR     $TIMES            ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(3)  016500  000415                          BR      1$                ;;ESCAPE TO THE NEXT TEST
(3)  016502  032777  004000  162730  3$:     BIT     #BIT11,@SWR       ;;INHIBIT ITERATIONS?
(3)  016510  001011                          BNE     1$                ;;BR IF YES
(3)  016512  005767  163016                  TST     $PASS             ;;IF FIRST PASS OF PROGRAM
(3)  016516  001406                          BEQ     1$                ;;      INHIBIT ITERATIONS
(3)  016520  005267  162660                  INC     $ICNT             ;;INCREMENT ITERATION COUNT
(3)  016524  026767  162762  162652          CMP     $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
(3)  016532  002024                          BGE     $OVER             ;;BR IF MORE ITERATION REQUIRED
(3)  016534  012767  000001  162642  1$:     MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
(3)  016542  016767  000056  162742          MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
(3)  016550  105267  162626          $SVLAD: INCB    $TSTNM            ;;COUNT TEST NUMBERS
(3)  016554  116767  162622  162750          MOVB    $TSTNM,$TESTN     ;;SET TEST NUMBER IN APT MAILBOX
(3)  016562  011667  162620                  MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
(3)  016566  011667  162616                  MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
(3)  016572  005067  162716                  CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(3)  016576  112767  000001  162611          MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(3)  016604  016777  162572  162630  $OVER:  MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
(3)  016612  016716  162570                  MOV     $LPADR,(SP)       ;;FUDGE RETURN ADDRESS
(5)  016616  000002                  4$:     RTI
(5)  016620  001407                  BRW:    1407
(5)  016622  000432                  BRX:    432
(3)  016624  000005                  $MXCNT: 5                         ;;MAX. NUMBER OF ITERATIONS
(2)                                  .SBTTL  TRAP DECODER
(2)
(3)                                  ;;**********************************************************************
(2)                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
(2)                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(2)                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(2)                                  ;*GO TO THAT ROUTINE.
(2)
(2)  016626  010046                  $TRAP:  MOV     R0,-(SP)          ;;SAVE R0
(2)  016630  016600  000002                  MOV     2(SP),R0          ;;GET TRAP ADDRESS
(2)  016634  005740                          TST     -(R0)             ;;BACKUP BY 2
(2)  016636  111000                          MOVB    (R0),R0           ;;GET RIGHT BYTE OF TRAP
(2)  016640  006300                          ASL     R0                ;;POSITION FOR INDEXING
(2)  016642  016000  016662                  MOV     $TRPAD(R0),R0     ;;INDEX TO TABLE
(2)  016646  000200                          RTS     R0                ;;GO TO ROUTINE
(2)
(2)
(2)                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
```

```
(2)
(2)  016650  011646                    $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
(2)  016652  016666  000004  000002            MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
(2)  016660  000002                            RTI                     ;;RESTORE THE PSW
(2)
(4)                                     .SBTTL  TRAP TABLE
(4)
(4)                                     ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
(4)                                     ;*BY THE "TRAP" INSTRUCTION.
(4)
(4)                                     ;       ROUTINE
(4)                                     ;       -------
(4)  016662  016650                    $TRPAD: .WORD   $TRAP2
(4)  016664  013042                            $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
(4)  016666  014560                            $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(4)  016670  014534                            $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(4)  016672  014574                            $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(2)
(2)
(3)  016674  013020                            .SCOP1  ;;CALL=SCOP1    TRAP+5(104405)
(3)  016676  013324                            .INSTR  ;;CALL=INSTR    TRAP+6(104406)
(3)  016700  013432                            .INSTER ;;CALL=INSTER   TRAP+7(104407)
(3)  016702  013442                            .PARAM  ;;CALL=PARAM    TRAP+10(104410)
(3)  016704  013642                            .SAV05  ;;CALL=SAV05    TRAP+11(104411)
(3)  016706  013702                            .RES05  ;;CALL=RES05    TRAP+12(104412)
(3)  016710  013734                            .CONVRT ;;CALL=CONVRT   TRAP+13(104413)
(3)  016712  014112                            .SETFLG ;;CALL=SETFLG   TRAP+14(104414)
8547                                    ;*****************************************************************
8548                                            ;UTILITIES
8549                                    ;*****************************************************************
8550
8551                                    ;THIS UTILITY CALCULATES PRIORITY LEVEL
8552  016714  006367  000044           DULEV:  ASL     DUPRT   ;SHIFT LEFT
8553  016720  006367  000040                   ASL     DUPRT   ;
8554  016724  006367  000034                   ASL     DUPRT   ;
8555  016730  006367  000030                   ASL     DUPRT   ;
8556  016734  006367  000024                   ASL     DUPRT   ;
8557  016740  016767  000020  000020          MOV     DUPRT,LESS1     ;MOVE THIS TO LESS1
8558  016746  162767  000001  000012          SUB     #1,LESS1        ;CREATE LESS1
8559  016754  042767  000037  000004          BIC     #37,LESS1       ;CLEAR TNZVC
8560  016762  000207                          RTS     PC
8561  016764  000240                   DUPRT:  PR5
8562  016766  000200                   LESS1:  PR4     ;LEVEL TO ALLOW INTERRUPTS
8563
8564                                            ;NEW DU ADDRESSES
8565  016770  016767  000126  162714   DUADDR: MOV     DUBASE,RXCSR    ;XXX0
8566  016776  005267  000120                   INC     DUBASE
8567  017002  016767  000114  162704          MOV     DUBASE,HRXCSR   ;XXX1
8568  017010  005267  000106                   INC     DUBASE
8569  017014  016767  000102  162674          MOV     DUBASE,RXDBUF   ;XXX2
8570  017022  016767  000074  162672          MOV     DUBASE,PARCSR   ;XXX2
8571  017030  005267  000066                   INC     DUBASE
8572  017034  016767  000062  162656          MOV     DUBASE,HRXDBUF  ;XXX3
8573  017042  016767  000054  162654          MOV     DUBASE,HPARCSR  ;XXX3
8574  017050  005267  000046                   INC     DUBASE
8575  017054  016767  000042  162644          MOV     DUBASE,TXCSR    ;XXX4
```

```
8576  017062  005267  000034                  INC     DUBASE
8577  017066  016767  000030  162634          MOV     DUBASE,HTXCSR   ;XXX5
8578  017074  005267  000022                  INC     DUBASE
8579  017100  016767  000016  162624          MOV     DUBASE,TXDBUF   ;XXX6
8580  017106  005267  000010                  INC     DUBASE
8581  017112  016767  000004  162614          MOV     DUBASE,HTXDBUF  ;XXX7
8582  017120  000207                          RTS     PC
8583  017122  000000                  DUBASE: 0
8584
8585                                          ;THIS UTILITY POKES THE MAINT DATA BASED UPON THE
8586                                          ;INFORMATION CONTAINED IN $TMP1 AND IT IS
8587                                          ;SHIFTED IN BY THE CONTENTS OF SHIFT
8588  017124  042777  040000  162574  RPOKE:  BIC     #MTDATA,@TXCSR
8589  017132  005067  162344                  CLR     $TMP2
8590  017136  006067  162336                  ROR     $TMP1   ;FORCE CARRY
8591  017142  006067  162334                  ROR     $TMP2   ;PICK UP CARRY IN BIT 15
8592  017146  006267  162330                  ASR     $TMP2   ;SHIFT INTO BIT 14
8593  017152  042767  100000  162322          BIC     #BIT15,$TMP2    ;CLR BIT 15
8594  017160  056777  162316  162540          BIS     $TMP2,@TXCSR    ;POKE MAINT DATA
8595  017166  042777  020000  162532          BIC     #CLK,@TXCSR     ;POKE CLK
8596  017174  052777  020000  162524          BIS     #CLK,@TXCSR     ;
8597  017202  005367  161714                  DEC     SHIFT
8598  017206  001346                          BNE     RPOKE
8599  017210  000207                          RTS     PC
8600
8601                                          ;THIS ROUTINE CALCULATES ODD PARITY FOR AN 8 BIT CHAR
8602  017212  016767  162262  162262  ODD8:   MOV     $TMP1,$TMP2     ;SAVE TEMP1
8603  017220  005067  162260                  CLR     $TMP3
8604  017224  012727  000010                  MOV     #8.,(PC)+
8605  017230  000000                  4$:     0
8606  017232  006067  162244          1$:     ROR     $TMP2
8607  017236  005567  162242                  ADC     $TMP3
8608  017242  005367  177762                  DEC     4$
8609  017246  001371                          BNE     1$
8610  017250  006067  162230                  ROR     $TMP3
8611  017254  103404                          BCS     2$
8612  017256  052767  000400  162214          BIS     #BIT8,$TMP1     ;SET ODD PARITY
8613  017264  000403                          BR      3$
8614  017266  042767  000400  162204  2$:     BIC     #BIT8,$TMP1     ;CLR EVEN PARITY
8615                                          ;$TMP1 NOW HAS ODD PARITY CHARACTER
8616  017274  000207                  3$:     RTS     PC
8617
8618                                          ;THIS ROUTINE CALCULATES EVEN PARITY FOR AN 8 BIT CHARACTER
8619  017276  016767  162176  162176  EVEN8:  MOV     $TMP1,$TMP2     ;SAVE TEMP1
8620  017304  005067  162174                  CLR     $TMP3
8621  017310  012727  000010                  MOV     #8.,(PC)+
8622  017314  000000                  4$:     0
8623  017316  006067  162160          1$:     ROR     $TMP2
8624  017322  005567  162156                  ADC     $TMP3
8625  017326  005367  177762                  DEC     4$
8626  017332  001371                          BNE     1$
8627  017334  006067  162144                  ROR     $TMP3
8628  017340  103004                          BCC     2$
8629  017342  052767  000400  162130          BIS     #BIT8,$TMP1     ;SET EVEN PARITY
8630  017350  000403                          BR      3$
8631  017352  042767  000400  162120  2$:     BIC     #BIT8,$TMP1     ;CLR ODD PARITY
```

```
 8632                                           ;$TMP1 NOW HAS EVEN PARITY CHARACTER
 8633  017360  000207                  3$:      RTS     PC
 8634  017362  062716  000002          TRPREG:  ADD     #2,(SP) ;ALLOW IT TO "CRUNCH" INTO ERROR BACK
 8635                                           ;IN MAIN PART OF THE PROGRAM
 8636  017366  000002                           RTI
 8637          017370                  POINT=.                 ;SAVE POINTER
  (1)          000100                  .=100
  (1)  000100  017370                  $CLKVEC                 ;LKVEC  HANDLER
  (1)  000102  000300                  300                     ;INTERRUPT HANDLER PRI
  (1)          000140                  .=140                   ;BRKVEC
  (1)  000140  170000                  170000                  ;ODT START ADDRESS
  (1)  000142  000300                  300                     ;PRIORITY
  (1)          017370                  .=POINT                 ;RESTORE POINTER
  (1)  017370  104401  017376          $CLKVEC:        TYPE,CLKMES
  (1)  017374  000000                                  HALT
  (1)  017376  005015  045514  042526  CLKMES:  .ASCIZ <15><12>/LKVEC INTERRUPT - DISCONNECT LTC /
  (1)  017404  020103  047111  042524
  (1)  017412  051122  050125  020124
  (1)  017420  020055  044504  041523
  (1)  017426  047117  042516  052103
  (1)  017434  046040  041524  000040
 8638          000001                  .END
```

```
AAA     003200            8165#
ABASE = 000000            8165
ACDW1 = 000000            8165
ACDW2 = 000000            8165
ACPUOP= 000000            8165
ACTREG  001166            8165#*  8546
ADDW0 = 000000            8165
ADDW1 = 000000            8165
ADDW10= 000000            8165
ADDW11= 000000            8165
ADDW12= 000000            8165
ADDW13= 000000            8165
ADDW14= 000000            8165
ADDW15= 000000            8165
ADDW2 = 000000            8165
ADDW3 = 000000            8165
ADDW4 = 000000            8165
ADDW5 = 000000            8165
ADDW6 = 000000            8165
ADDW7 = 000000            8165
ADDW8 = 000000            8165
ADDW9 = 000000            8165
ADEVCT= 000000            8165
ADEVM = 000000            8165
ADRCNT= 013641            8546#*
AENV  = 000000            8165
AENVM = 000000            8165
AFATAL= 000000            8165
AMADR1= 000000            8165
AMADR2= 000000            8165
AMADR3= 000000            8165
AMADR4= 000000            8165
AMAMS1= 000000            8165
AMAMS2= 000000            8165
AMAMS3= 000000            8165
AMAMS4= 000000            8165
AMSGAD= 000000            8165
AMSGLG= 000000            8165
AMSGTY= 000000            8165
AMTYP1= 000000            8165
AMTYP2= 000000            8165
AMTYP3= 000000            8165
AMTYP4= 000000            8165
APASS = 000000            8165
APRIOR= 000000            8165
APTCSU= 000040            7023#   8546
APTENV= 000001            7023#   8546
APTSIZ= 000200            7023#   8165
APTSPO= 000100            7023#   8546
ASWREG= 000000            8165
ATESTN= 000000            8165
AUNIT = 000000            8165
AUSWR = 000000            8165
AVECT1= 000000            8165
AVECT2= 000000            8165
BASEAD  001154            8165#*  8546*
```

```
BASEIV  001162           8165#* 8546*
BBB     003026           8165#
BINWRD  014110           8546#*
BITW  = 002000           8165#  8246  8253  8264  8271  8281  8295  8299  8306
BIT0  = 000001           8165#  8233
BIT00 = 000001           8165#
BIT01 = 000002           8165#
BIT02 = 000004           8165#
BIT03 = 000010           8165#
BIT04 = 000020           8165#
BIT05 = 000040           8165#
BIT06 = 000100           8165#
BIT07 = 000200           8165#
BIT08 = 000400           8165#  8546
BIT09 = 001000           8165#  8546
BIT1  = 000002           8165#
BIT10 = 002000           8165#  8546
BIT11 = 004000           8165#  8195  8546
BIT12 = 010000           8165#  8198
BIT13 = 020000           8165#  8546
BIT14 = 040000           8165#  8546
BIT15 = 100000           8165#  8593
BIT2  = 000004           8165#
BIT3  = 000010           8165#
BIT4  = 000020           8165#
BIT5  = 000040           8165#  8244
BIT6  = 000100           8165#
BIT7  = 000200           8165#
BIT8  = 000400           8165#  8612  8614  8629  8631
BIT9  = 001000           8165#
BPTVEC= 000014           8165#
BREAK = 000001           8165#  8188  8262  8277  8293  8305  8537  8538  8539  8540  8541  8542  8543
                         8544
BRKVEC= 000140           8165#
BRW     016620           8546#
BRX     016622           8546#
CARDET= 010000           8165#
CCC     012576           8546#
CHRCNT  014106           8546#*
CKSWR   012672           8546#
CLK   = 020000           8165#  8199  8537  8538  8539  8540  8541  8542  8543  8544  8595  8596
CLKMES  017376           8637#
CNTLU   012746           8165   8546#
CONVRT= 104413           8546#
COUNT   001124           8165#  8539* 8540* 8541* 8542*
CR    = 000015           8165#  8546
CRLF  = 000200           8165#  8546
CTS   = 020000           8165#
DDISP = 177570           8165#
DEVADR  013636           8546#*
DEVICE  015302           8546#
DF1     002132           8165#
DH1     002067           8165#
DISPLA  001442           8165#* 8546*
DISPRE  000174           8165#
DNA   = 100000           8165#  8324
```

```
DNAINT= 000040           8165#   8191
DSC   = 100000           8165#   8314
DSINTE= 000040           8165#   8185
DSR   = 001000           8165#
DSWR  = 177570           8165#
DTR   = 000002           8165#   8180    8416    8461    8506    8528
DT1     002116           8165#
DT4     002126           8165#
DUADDR  016770           8165    8546    8565#
DUBASE  017122           8165    8546*   8565    8566*   8567    8568*   8569    8570    8571*   8572    8573    8574*   8575
                         8576*   8577    8578*   8579    8580*   8581    8583#
DULEV   016714           8165    8552#
DUPRT   016764           8165*   8552*   8553*   8554*   8555*   8556*   8557    8561#
DURIS   001740           8165#   8546*
DURIV   001736           8165#   8546*
DUTIS   001744           8165#   8546*
DUTIV   001742           8165#   8546*
EIGHT = 006000           8165#   8537    8538    8542
EMTVEC= 000030           8165#*
EM1     001762           8165#
EM2     002022           8165#
EM3     002043           8165#
EM4     001746           8165#
ERRCNT  001114           8165#
ERRVEC= 000004           8165#*  8546*
EVEN8   017276           8619#
EVEPAR= 001400           8165#
EVPAR = 000400           8165#
FIVE  = 000000           8165#   8539    8543    8544
FRMERR= 020000           8165#   8337
GNS   = ****** U         8546
HDXEN = 000010           8165#   8189
HILIM   013634           8546#*
HLD0    001130           8165#   8546*
HLD1    001132           8165#   8546*
HLD2    001134           8165#   8546*
HLD3    001136           8165#   8546*
HLD4    001140           8165#   8546*
HLD5    001142           8165#   8546*
HLD6    001144           8165#   8546*
HOLD    001120           8165#   8169*   8182    8222    8365    8417    8430    8447    8462    8469    8476    8492    8499
                         8507    8515    8522    8529
HPARCS  001724           8165#   8171*   8573*
HRXCSR  001714           8165#   8168*   8233*   8567*
HRXDBU  001720           8165#   8170*   8572*
HT    = 000011           8165#   8546
HTXCSR  001730           8165#   8172*   8244*   8577*
HTXDBU  001734           8165#   8173*   8581*
INBUF   016124           8165    8546#
INIFLG  001172           8165#*
INSTER= 104407           8165    8546#
INSTR = 104406           8165    8546#
INSTR2  013440           8546#
IOTVEC= 000020           8165#*
ISYMOD= 000000           8165#   8538    8543    8544
JMRBY   001153           8165#*  8291    8405
```

```
KEEPAD  001156          8165#*  8546
KEEPIV  001164          8165#*  8546
LASTAD  001160          8165#
LESS1   016766          8557*   8558*   8559*   8562#
LF    = 000012          8165#   8546
LIGHTS  001102          8165#   8546*
LIMITS  013566          8546#
LKVEC = 000100          8165#
LOBITS  013640          8546#*
LOCK    001110          8165#   8546
LOKFLG  001174          8165#
LOLIM   013632          8546#*
MASK1   007324          8449#
MASK2   007342          8455#
MASK3   007432          8471#
MCNTG   016070          8546#
MCOW    015314          8546#
MCRLF   015414          8546#
MDATA   016230          8546#
MEPASS  015463          8546#
MEXT  = 010000          8165#   8293    8305
MEXTJ   016030          8165    8546#
MINT  = 004000          8165#   8262    8277    8537    8538    8539    8540    8541    8542    8543    8544
MLASTD  015250          8165    8546#
MLOCK   015527          8546#
MMNEW   016111          8546#
MMSWR   016100          8546#
MMULT   015221          8165    8546#
MPFAIL  015417          8546#
MQM     015410          8546#
MR      015510          8165    8546#
MRANGE  015344          8165    8546#
MREGAD  015171          8165    8546#
MRESET= 000400          8165#   8180    8181    8183    8184    8185    8186    8187    8188    8189    8190    8191    8192
                        8195    8198    8199    8200    8224    8228    8243    8276    8304    8312    8313    8314    8319
                        8324    8329    8336    8337    8338    8339    8344    8373    8385    8400    8537    8538    8539
                        8540    8541    8542    8543    8544
MSYNC   015562          8165    8546#
MTDATA= 040000          8165#   8200    8262    8267    8275    8293    8298    8303    8537    8538    8539    8540    8541
                        8542    8543    8544    8588
MTITLE  015122          8165    8546#
MTSTPC  015514          8165    8546#
MULTD   001152          8165#   8546
MVECTO  015156          8165    8546#
MWIRE4  015751          8165    8546#
MWIRE5  015701          8165    8546#
MWIRE6  015630          8165    8546#
NEXT    001106          8165#
NOPAR = 000000          8165#   8537    8538    8539    8540    8541    8542    8543    8544
ODDPAR= 001000          8165#
ODD8    017212          8602#
ODTST = 170000          8165#
ONCE    002610          8165#
OPTCLR  001151          8165#   8180    8181    8183    8211    8229    8347
OUT     013000          8546#
OUTCRY  012570          8546#
```

```
OUTMUL  003166            8165#
OUT1    007774            8408    8488    8513    8535#
OUT2    007520            8443    8491#
OUT3    007650            8445    8514#
OVRRUN= 040000            8165#   8338
PARAM = 104410            8165    8546#
PARAM1  013472            8546#
PARCSR  001722            8165#   8171*   8537*   8538*   8539*   8540*   8541*   8542*   8543*   8544*   8570*
PAREN = 001000            8165#
PARER = 010000            8165#   8336
PARERR  013546            8546#
PARTI   013630            8546#
PASCNT  001112            8165#*  8546*
PFTAB   015114            8546#
PIRQ  = 177772            8165#
PIRQVE= 000240            8165#
POINT = 017370            8637#
POPR0 = 012600            8165#
POP1SP= 005726            8165#
POP2SP= 022626            8165#
PR0   = 000000            8165#
PR1   = 000040            8165#
PR2   = 000100            8165#
PR3   = 000140            8165#
PR4   = 000200            8165#   8562
PR5   = 000240            8165#   8561
PR6   = 000300            8165#   8168    8170    8171    8172    8173
PR7   = 000340            8165#
PS    = 177776            8165#
PSW   = 177776            8165#
PUSHR0= 010046            8165#
PUSH1S= 005746            8165#
PUSH2S= 024646            8165#
PWRVEC= 000024            8165#*
RDSW    013006            8546#*
RECACT= 004000            8165#   8313    8537    8538    8539    8540    8541    8542
REPLAY  012474            8546#
RESTAR  015020            8546#
RESTRT  012654            8546#
RESVEC= 000010            8165#
RESO5 = 104412            8546#
RETURN  001104            8165#*  8546
RING  = 040000            8165#
RINTEN= 000100            8165#   8186
ROTADD  001170            8165#*  8546*
RPOKE   017124            8539    8540    8541    8542    8543    8544    8588#   8598
RTS   = 000004            8165#   8181    8429    8468    8498    8521
RUNA  = 000000            5676#   5678    5865    6006    6420
RUNB  = ****** U          5704    5874    6013    6429
RUNC  = ****** U          5729    5882    6020    6438
RUND  = ****** U          5754    5891    6027    6447
RUNE  = ****** U          5779    5901    6034    6456
RUNF  = ****** U          5804    5911    6041    6465
RUNIT   012400            8546#
RXCSR   001712            8165#   8168*   8180*   8181*   8183*   8184*   8185*   8186*   8187*   8205*   8209    8231*   8232
                          8234    8238*   8239    8312    8313    8314    8343*   8345    8346    8399*   8410    8411    8416*
```

```
                    8418    8423    8429*   8431    8436    8446*   8448    8454    8461*   8463    8468*   8470    8475*
                    8477    8482    8491*   8493    8498*   8500    8506*   8508    8514*   8516    8521*   8523    8528*
                    8530    8537*   8538*   8539*   8540*   8541*   8542*   8543*   8544*   8546    8565*
RXDBUF  001716      8165#   8170*   8330    8332    8336    8337    8338    8339    8386    8387    8539    8540    8541
                    8542    8543    8544    8569*
RXDONE= 000200      8165#   8312
RXERR = 100000      8165#   8339
SAVIT   014342      8546#
SAVPC   001126      8165#   8546*
SAVSP   001116      8165#   8546*
SAV05 = 104411      8546#
SCOP1 = 104405      8546#
SEND  = 000020      8165#   8190
SEREC   001150      8165#   8444
SETFLG= 104414      8165    8546#
SEVEN = 004000      8165#   8541
SEXMIT  001147      8165#   8442
SHIFT   001122      8165#   8539*   8540*   8541*   8542*   8543*   8544*   8597*
SIX   = 002000      8165#   8540
SPACNT= 014107      8546#*
SRD   = 002000      8165#
STACK = 001100      8165#   8546
STD   = 000010      8165#   8183    8446    8475    8491    8514
STFLG   001173      8165#*
STKLMT= 177774      8165#
STPSYN= 000400      8165#   8187
SV05    013650      8546#
SWR     001440      8165#*  8546
SWREG   000176      8165#   8546
SWREGL  013010      8546#
SW0   = 000001      8165#
SW00  = 000001      8165#
SW01  = 000002      8165#
SW02  = 000004      8165#
SW03  = 000010      8165#
SW04  = 000020      8165#
SW05  = 000040      8165#
SW06  = 000100      8165#
SW07  = 000200      8165#
SW08  = 000400      8165#
SW09  = 001000      8165#   8546
SW1   = 000002      8165#
SW10  = 002000      8165#
SW11  = 004000      8165#
SW12  = 010000      8165#
SW13  = 020000      8165#
SW14  = 040000      8165#
SW15  = 100000      8165#
SW2   = 000004      8165#
SW3   = 000010      8165#
SW4   = 000020      8165#
SW5   = 000040      8165#
SW6   = 000100      8165#
SW7   = 000200      8165#
SW8   = 000400      8165#
SW9   = 001000      8165#
```

```
SYNCNO  001146          8165#* 8539   8540   8541   8542
SYNEXT= 020000          8165#  8537
SYNINT= 030000          8165#  8539   8540   8541   8542
SYNSCH= 000020          8165#  8184   8537   8538   8539   8540   8541   8542   8543   8544
SYSTST= 014000          8165#
TBITVE= 000014          8165#
TEMP    016166          8546#*
TKVEC = 000060          8165#
TPVEC = 000064          8165#
TRAPVE= 000034          8165#*
TRPREG  017362          8168   8170   8171   8172   8173   8634#
TRTVEC= 000014          8165#
TST1    003374          8165   8168#  8546
TST10   004054          8181#
TST11   004200          8183#
TST12   004314          8184#
TST13   004404          8185#
TST14   004474          8186#
TST15   004564          8187#
TST16   004654          8188#
TST17   004744          8189#
TST2    003454          8170#
TST20   005034          8190#
TST21   005124          8191#
TST22   005214          8192#
TST23   005304          8195#
TST24   005374          8198#
TST25   005464          8199#
TST26   005554          8200#
TST27   005644          8204#
TST3    003526          8171#
TST30   005746          8224#
TST31   006010          8227#
TST32   006162          8261#
TST33   006300          8288#
TST34   006404          8312#
TST35   006426          8313#
TST36   006450          8314#
TST37   006472          8318#
TST4    003600          8172#
TST40   006514          8324#
TST41   006536          8328#
TST42   006570          8336#
TST43   006612          8337#
TST44   006634          8338#
TST45   006656          8339#
TST46   006700          8342#
TST47   007002          8371#
TST5    003652          8173#
TST50   007042          8384#
TST51   007074          8398#
TST52   007774          8537#
TST53   010106          8538#
TST54   010220          8539#
TST55   010526          8540#
TST56   011034          8541#
```

```
TST57   011342            8542#
TST6    003724            8176#
TST60   011650            8543#
TST61   012076            8544#
TST7    003740            8180#
TTST    016310            8546#
TXCSR   001726            8165#   8172*   8180*   8181*   8183*   8184*   8185*   8186*   8187*   8188*   8189*   8190*   8191*
                          8192*   8195*   8198*   8199*   8200*   8206*   8210    8224*   8228*   8243*   8245    8251*   8252
                          8262*   8264    8267*   8271    8275*   8276*   8277*   8281    8293*   8295    8298*   8299    8303*
                          8304*   8305*   8306    8312*   8313*   8314*   8319*   8320    8324*   8329*   8336*   8337*   8338*
                          8339*   8344*   8372*   8373*   8374    8375    8385*   8400*   8537*   8538*   8539*   8540*   8541*
                          8542*   8543*   8544*   8575*   8588*   8594*   8595*   8596*
TXDBUF  001732            8165#   8173*   8177    8579*
TXDONE= 000200            8165#   8320
TXINTE= 000100            8165#   8192
TYPE  = 104401            8165    8546#   8637
TYPOC = 104402            8546#
TYPON = 104404            8546#
TYPOS = 104403            8546#
USER  = 000000            8165#
VOID  = 000001            8165#
WRDCNT  014104            8546#*
ZERO    012406            8546#
$APTHD  002136            8165#
$ASTAT= ****** U          7023
$ATYC   000024            7023#
$ATY1   000000            7023#
$ATY3   000006            7023#   8546
$ATY4   000016            7023#   8546
$AUTOB  001434            8165#
$BASE   001602            8165#
$BDADR  001422            8165#
$BDDAT  001426            8165#
$BELL   001516            8165#   8546
$CDW1   001606            8165#
$CDW2   001610            8165#
$CHARC  013320            8546#*
$CKSWR= ****** U          8546
$CLKVE  017370            8637#
$CMTAG  001400            8165#
$CM1  = 000006            8165#
$CM2  = 000014            8165#
$CM3  = 000006            8165#
$CM4  = 000006            8165#
$CPUOP  001554            8165#
$CRLF   001523            8165#   8546
$DDW0   001612            8165#
$DDW1   001614            8165#
$DDW10  001636            8165#
$DDW11  001640            8165#
$DDW12  001642            8165#
$DDW13  001644            8165#
$DDW14  001646            8165#
$DDW15  001650            8165#
$DDW2   001616            8165#
$DDW3   001620            8165#
```

```
$DDW4   001622            8165#
$DDW5   001624            8165#
$DDW6   001626            8165#
$DDW7   001630            8165#
$DDW8   001632            8165#
$DDW9   001634            8165#
$DEVCT  001536            8165#
$DEVM   001604            8165#
$E    = 000002            6851#
$ENDAD  012644            8165    8546#
$ENV    001546            7023    8165#   8546
$ENVM   001547            7023    8165#   8546
$ERFLG  001403            8165#*  8546*
$ERMAX  001415            8165#*  8546*
$ERROR  014160            8165    8546#
$ERRPC  001416            8165#*  8546*
$ERRTB  001652            8165#   8546
$ERRTY  014400            8546#
$ERTTL  001412            8165#*  8546*
$ESCAP  001514            8165#*  8546*
$ETABL  001546            8165#
$ETEND  001652            8165#
$FATAL  001530            7023*   8165#
$FFLG   000244            7023#*
$FILLC  001456            8165#   8546
$FILLS  001455            8165#   8546
$GDADR  001420            8165#
$GDDAT  001424            8165#
$GTSWR= ****** U          8546
$HIBTS  002136            8165#
$ICNT   001404            8165#   8546*
$INTAG  001435            8165#
$ITEMB  001414            8165#   8546*
$LF     001524            8165#   8546
$LFLG   000243            7023#*
$LPADR  001406            8165#*  8546*
$LPERR  001410            8165#*  8546*
$MADR1  001560            8165#
$MADR2  001564            8165#
$MADR3  001570            8165#
$MADR4  001574            8165#
$MAIL   001526            8165#   8546
$MAMS1  001556            8165#
$MAMS2  001562            8165#
$MAMS3  001566            8165#
$MAMS4  001572            8165#
$MBADR  002140            8165#
$MFLG   000242            7023#*
$MSGAD  001542            7023*   8165#
$MSGLG  001544            7023*   8165#
$MSGTY  001526            7023*   8165#
$MTYP1  001557            8165#
$MTYP2  001563            8165#
$MTYP3  001567            8165#
$MTYP4  001573            8165#
$MXCNT  016624            8546#
```

```
$N     = 000000          6850#  8546#
$NULL    001454          8165#  8546
$NWTST= 000000           8168#  8170#  8171#  8172#  8173#  8176#  8180#  8181#  8183#  8184#  8185#  8186#  8187#
                         8188#  8189#  8190#  8191#  8192#  8195#  8198#  8199#  8200#  8204#  8224#  8227#  8261#
                         8288#  8312#  8313#  8314#  8318#  8324#  8328#  8336#  8337#  8338#  8339#  8342#  8371#
                         8384#  8398#  8537#  8538#  8539#  8540#  8541#  8542#  8543#  8544#
$OCNT    014756          8546#*
$OMODE   014760          8546#*
$OVER    016604          8546#
$PASS    001534          8165#*  8546*
$PASTM   002144          8165#
$PWRDN   014762          8165    8546#
$QUES    001522          8165#   8546
$RDCHR= ****** U         8546
$RDDEC= ****** U         8546
$RDLIN= ****** U         8546
$RDOCT= ****** U         8546
$REGAD   001460          8165#
$REG0    001462          8165#   8546*
$REG1    001464          8165#   8546*
$REG2    001466          8165#   8546*
$REG3    001470          8165#   8546*
$REG4    001472          8165#   8546*
$REG5    001474          8165#   8546*
$R2A   = ****** U         8546
$SAVRE= ****** U         8546
$SCOPE   016272          8165    8546#
$SETUP= 000017           8165#   8546
$STUP  = 177777          8165#
$SVLAD   016550          8546#
$SVPC  = 002136          8165#
$SWR   = 177400          6510#  8165   8168   8170   8171   8172   8173   8176   8180   8181   8183   8184   8185
                         8186   8187   8188   8189   8190   8191   8192   8195   8198   8199   8200   8204   8224
                         8227   8261   8288   8312   8313   8314   8318   8324   8328   8336   8337   8338   8339
                         8342   8371   8384   8398   8537   8538   8539   8540   8541   8542   8543   8544   8546
$SWREG   001550          8165#
$SWRMK= 000000           8546
$TESTN   001532          8165#   8546*
$TIMES   001512          8165#*  8546*
$TKB     001446          8165#   8546
$TKS     001444          8165#   8546
$TMP0    001476          8165#
$TMP1    001500          8165#   8539*  8540*  8541*  8542*  8543*  8544*  8590*  8602   8612*  8614*  8619   8629*
                         8631*
$TMP2    001502          8165#   8589*  8591*  8592*  8593*  8594   8602*  8606*  8619*  8623*
$TMP3    001504          8165#   8603*  8607*  8610*  8620*  8624*  8627*
$TMP4    001506          8165#
$TMP5    001510          8165#
$TN    = 000062          7262#  8168#  8170#  8171#  8172#  8173#  8176#  8180#  8181#  8183#  8184#  8185#  8186#
                         8187#  8188#  8189#  8190#  8191#  8192#  8195#  8198#  8199#  8200#  8204#  8224#  8227#
                         8261#  8288#  8312#  8313#  8314#  8318#  8324#  8328#  8336#  8337#  8338#  8339#  8342#
                         8371#  8384#  8398#  8537#  8538#  8539#  8540#  8541#  8542#  8543#  8544#
$TPB     001452          8165#   8546*
$TPFLG   001457          8165#   8546
$TPS     001450          8165#   8546
$TRAP    016626          8165    8546#
```

```
$TRAP2  016650          8546#
$TRP   = 000015         8546#
$TRPAD  016662          8546#
$TSTM   002142          8165#
$TSTNM  001402          8165#*  8546*
$TYPBN= ****** U        8546
$TYPDS= ****** U        8546
$TYPE   013042          7023    8546#
$TYPEC  013254          8546#
$TYPEX  013322          8546#
$TYPOC  014560          8546#
$TYPON  014574          8546#
$TYPOS  014534          8546#
$UNIT   001540          8165#
$UNITM  002146          8165#
$USWR   001552          8165#
$VECT1  001576          8165#
$VECT2  001600          8165#
$XTSTR  016354          8546#
$OFILL  014757          8546#*
$40CAT= ****** U        8546
.      = 017442         7023#   8165#   8168    8170    8171    8172    8173    8178    8180    8181    8182    8183    8184
                        8185    8186    8187    8188    8189    8190    8191    8192    8195    8198    8199    8200    8216
                        8220    8222    8224    8236    8241    8249    8255    8265    8272    8282    8296    8300    8307
                        8312    8313    8314    8321    8324    8334    8336    8337    8338    8339    8353    8360    8365
                        8378    8390    8406    8414    8417    8421    8426    8430    8434    8439    8447    8452    8458
                        8462    8466    8469    8473    8476    8480    8485    8492    8496    8499    8504    8507    8511
                        8515    8519    8522    8526    8529    8533    8537    8538    8539    8540    8541    8542    8543
                        8544    8546#   8637#
.BEGIN  003310          8165#   8546
.CONVR  013734          8546#
.EOP    012324          8546#
.INSTE  013432          8546#
.INSTR  013324          8546#
.INST1  013344          8546#
.MSG    013346          8546#*
.PARAM  013442          8546#
.PFAIL  014762          8165    8546#
.RESO5  013702          8546#
.SAVO5  013642          8546#
.SCOPE  016272          8546#
.SCOP1  013020          8546#
.SETFL  014112          8546#
.START  002152          8165#   8546
.$ASTA= ****** U        7023
.$X    = 002136         8165#
```

```
COMMEN  1566#  8165#
ENDCOM  1578#  8165#
ERROR   8165#  8168  8170  8171  8172  8173  8179  8180  8181  8183  8184  8185  8186  8187  8188
        8189   8190  8191  8192  8195  8198  8199  8200  8217  8221  8224  8237  8242  8250  8256
        8266   8273  8283  8297  8301  8308  8312  8313  8314  8322  8324  8335  8336  8337  8338
        8339   8354  8361  8379  8391  8415  8422  8427  8435  8440  8453  8459  8467  8474  8481
        8486   8497  8505  8512  8520  8527  8534  8537  8538  8539  8540  8541  8542  8543  8544
ESCAPE  1698#  8165#
GETPRI  1312#  8165#
GETSWR  1771#  8165#
MULT    4484#  8165#
NEWTST  1626#  8165#  8168  8170  8171  8172  8173  8176  8180  8181  8183  8184  8185  8186  8187
        8188   8189  8190  8191  8192  8195  8198  8199  8200  8204  8224  8227  8261  8288  8312
        8313   8314  8318  8324  8328  8336  8337  8338  8339  8342  8371  8384  8398  8537  8538
        8539   8540  8541  8542  8543  8544
POP     2153#  7023  8165#
PRGEND  8129#  8546
PRGFRT  7263#  8165
PUSH    2145#  7023  8165#
PUSSYF  8000#
REPORT  5463#  6509#  8165#
RSETUP  7458#  8537  8538  8539  8540  8541  8542  8543  8544
SC      6823#  8546
SCOPE   8165#  8168  8170  8171  8172  8173  8176  8180  8181  8183  8184  8185  8186  8187  8188
        8189   8190  8191  8192  8195  8198  8199  8200  8204  8224  8227  8261  8288  8312  8313
        8314   8318  8324  8328  8336  8337  8338  8339  8342  8371  8384  8398  8537  8538  8539
        8540   8541  8542  8543  8544  8546
SC1     6841#  8546
SETPRI  1279#  8165#
SETTRA  6508#  8546#
SETUP   1337#  8165#
SKIP    1733#  8165#
SLASH   1517#  8165#
SPACE   8165#
STARS   1485#  7023  8165#  8168  8170  8171  8172  8173  8176  8180  8181  8183  8184  8185  8186
        8187   8188  8189  8190  8191  8192  8195  8198  8199  8200  8204  8224  8227  8261  8288
        8312   8313  8314  8318  8324  8328  8336  8337  8338  8339  8342  8371  8384  8398  8537
        8538   8539  8540  8541  8542  8543  8544  8546
SWRSU   1453#  8165#
TRMTRP  8546#
TSETUP  7470#
TYPBIN  2088#  8165#
TYPDEC  2058#  8165#
TYPNAM  1826#  8165#
TYPNUM  2025#  8165#
TYPOCS  1978#  8165#
TYPOCT  1941#  8165#  8546
TYPTXT  1894#  8165#
$BEGIN  6732#  8165
$BINAR  7947#
$BUFFE  7137#  8546
$CABLE  8015#  8182  8222  8365  8417  8430  8447  8462  8469  8476  8492  8499  8507  8515  8522
        8529
$CATCH  6601#  8165
$CK     6977#  8546
$CLRVE  6747#  8165
```

```
$CONVR  7148#  8546
$DNA    7848#
$EOP    6895#  8546
$GETFL  6777#  8165
$GETPA  6767#  8165
$GETSY  6783#  8165
$HEADE  6512#  8165
$INSTR  7024#  8546
$ISOB   7776#
$MATCH  7585#
$MRR    7445#  8312  8313  8314  8324  8336  8337  8338  8339
$MRRW   7393#  8180  8181  8183  8184  8185  8186  8187  8188  8189  8190  8191  8192  8195  8198
        8199   8200
$MRW    7428#  8224
$MSG    7111#  8546
$NEED   8117#  8546
$PARAM  7054#  8546
$PFAIL  7193#  8546
$POKE   7568#
$POKER  7573#  8539  8540  8541  8542  8543  8544
$RCNET  8027#
$RECAC  7507#  8539  8540  8541  8542
$REG    7236#  8546
$RESET  7389#  8180  8181  8183  8184  8185  8186  8187  8188  8189  8190  8191  8192  8195  8198
        8199   8200  8224  8228  8243  8276  8304  8312  8313  8314  8319  8324  8329  8336  8337
        8338   8339  8344  8373  8385  8400  8537  8538  8539  8540  8541  8542  8543  8544
$RXACT  7483#  8537  8538
$SCOPE  6846#  8546
$SCOP1  7012#  8546
$SETFL  6803#  8546
$SETVE  6612#  8165
$START  6690#  8165
$STRIP  7817#
$SYMBO  6528#  8165
$SYNCR  7579#  8539  8540  8541  8542  8543  8544
$TRPAR  7899#
$TSTNO  6853#  8168  8170  8171  8172  8173  8176  8180  8181  8183  8184  8185  8186  8187  8188
        8189   8190  8191  8192  8195  8198  8199  8200  8204  8224  8227  8261  8288  8312  8313
        8314   8318  8324  8328  8336  8337  8338  8339  8342  8371  8384  8398  8537  8538  8539
        8540   8541  8542  8543  8544
$UNIBU  7372#  8168  8170  8171  8172  8173
$VARIA  6628#  8165
$WORDF  7715#
$WORDO  7665#  8543  8544
$WORDP  7743#
$$CMRE  8165#
$$CMTM  8165#
$$ESCA  1711#  8165#
$$NEWT  1662#  8165#  8168  8170  8171  8172  8173  8176  8180  8181  8183  8184  8185  8186  8187
        8188   8189  8190  8191  8192  8195  8198  8199  8200  8204  8224  8227  8261  8288  8312
        8313   8314  8318  8324  8328  8336  8337  8338  8339  8342  8371  8384  8398  8537  8538
        8539   8540  8541  8542  8543  8544
$$SET   8546#
$$SETM  8165#
$$SKIP  1746#  8165#
.EQUAT   189#  6508#  8165
```

```
.HEADE    61#   6507#
.INIT   5658#   8165#   8637
.SETUP  1213#   6507#   8165
.SWRHI   104#
.$ACT1  5064#   6509#   8165
.$APTB  5109#   6509#   8165#
.$APTH  5370#   6509#   8165
.$APTY  5547#   6509#   7023
.$ASTA  5417#
.$CATC   932#   6507#
.$CMTA  1047#   6507#   8165
.$DB2D  4686#
.$DB2O  4812#
.$DIV   4587#
.$EOP   2214#   6507#
.$ERRO  2700#   6508#   8546
.$ERRT  2856#   6508#   8546
.$MULT  4523#
.$POWE  4229#   6508#
.$RAND  4307#
.$RDDE  3891#
.$RDOC  3797#
.$READ  3395#
.$R2AZ  4958#
.$SAVE  3969#
.$SB2D  4771#
.$SB2O  4874#
.$SCOP  2454#   6508#   8546
.$SIZE  4361#
.$SUPR  4913#
.$TRAP  4073#   6508#   8546
.$TYPB  3287#
.$TYPD  3209#
.$TYPE  2985#   6507#   8546
.$TYPO  3112#   6508#   8546
.$40CA   972#

. ABS.  017442     000

 ERRORS DETECTED:  0

 CNDUQA,CNDUQA/CRF/NL:TOC=CNMAC2.SML,CNDUQ3.M11,CNDUQ4.M11,CNDUQA.M11
 RUN-TIME: 18 18 1 SECONDS
 RUN-TIME RATIO: 121/38=3.1
 CORE USED:  43K  (86 PAGES)
```