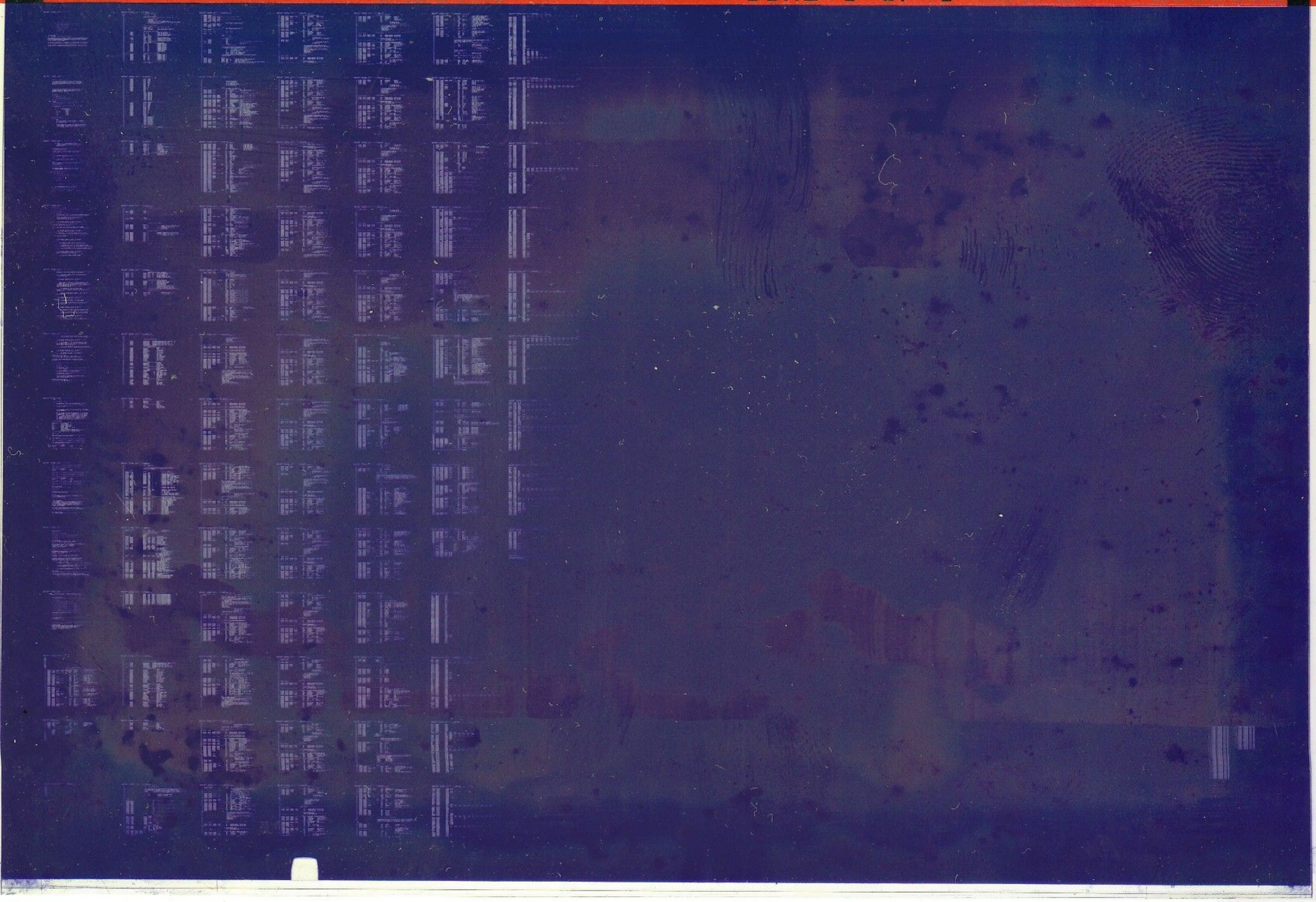
DUV11 OFLNE XMT TSTS
CNDUTAO

AH-T456A-MC
FICHE 1 OF 1

MAY 1983

MADE IN USA



```
                                                                .REM    *

                        I D E N T I F I C A T I O N
                        PRODUCT NAME: CNDUTAO DUV11 OFLNE XMT TSTS
                        PRODUCT CODE:AC-T455A-MC
                        PRODUCT DATE:DEC 1982
                        MAINTAINER  :DIAGNOSTICS SERVICES/ISS
                        AUTHOR: K.LIND
                                                                *
                                                                .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 TRANSMITTER TESTS VERIFY THAT THE TRANSMITTER
SECTION PROVIDES THE CORRECT ERROR FLAGS,AND THAT IT
TRANSMITS CHARACTERS THRU THE BIT WINDOW AT THE CORRECT
NUMBER OF BITS PER CHARACTER.

*

.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 RETARTING 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 CNDUT-A TAPE D" (ONCE ONLY)

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
16XXXX	YYYYYY	ZZZZZZ

WHERE 16XXXX 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
16XXXX	YYYYYY	ZZZZZZ

WHERE 16XXXX 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
16XXXX	YYYYYY	ZZZZZZ

WHERE 16XXXX IS THE ADDRESS OF THE FAILING TRANSMITTER (TXCSR) 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
            16XXXX = DEVICE

            WHERE Y IS THE TAPE LOADED

            WHERE 16XXXX 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
```

K 1

SEQ 0010

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 XMIT 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 TRANSMITTER SECTION TESTING
OF THE DEVICE
SEE LISTING FOR DETAILS

*

.REM *

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

11. REVISION HISTORY
DZDUTB1 DIAGNOSTIC WAS MODIFIED TO WORK IN 11/21 PROCESSOR
PRIORITY340 WAS CHANGED TO 300 WHEREVER ENCOUNTERED
DEFAULT CSR AND VECTOR ADDRESSES WERE CHANGED
.INIT CALL TO CNMAC2.SML ADDED TO INIT 11/21 SPECIFICS.
DIAGNOSTIC WAS RENAMED TO CNDUTA0.

12. LISTINGS

*

```
 6584
 6585
 6586
 6597
 6611
 6612
 6613
 6674
 6675
 6880
 7008
 7009                                   .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  012712                  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
 7248            000001                $TN=1
 7374
 7378
 7413
 7430
 7443
 7455
 7468
```

```
7491
7557
7563
7699
7727
7760
7801
7832
7883
7931
7984
7999
8011
8113
```

```
                                .ENABLE ABS

                                ;DUV11 CNDUT-A TAPE D
                                ;COPYRIGHT 1977, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754

                                        ;STARTING PROCEDURE
                                        ;TYPE 200G
                                        ;PROGRAM WILL TYPE "DUV11 CNDUT-A TAPE D "
                                        ;PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
                                        ;AT THE END OF A PASS, PROGRAM WILL TYPE "END OF PASS TAPE D"
                                        ;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
000C00                          R0=     %0              ;;GENERAL REGISTER
000001                          R1=     %1              ;;GENERAL REGISTER
000002                          R2=     %2              ;;GENERAL REGISTER
000003                          R3=     %3              ;;GENERAL REGISTER
000004                          R4=     %4              ;;GENERAL REGISTER
000005                          R5=     %5              ;;GENERAL REGISTER
000006                          R6=     %6              ;;GENERAL REGISTER
000007                          R7=     %7              ;;GENERAL REGISTER
000006                          SP=     %6              ;;STACK POINTER
000007                          PC=     %7              ;;PROGRAM COUNTER

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

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

```
(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
```

```
7
        (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 CONTRO. 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)  001550  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  012670                          $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  016314  000020             MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
(3)  002200  012737  000300  000022             MOV     #PR6,@#IOTVEC+2 ;;LEVEL 6
(3)  002206  012737  014204  000030             MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
(3)  002214  012737  000300  000032             MOV     #PR6,@#EMTVEC+2 ;;LEVEL 6
(3)                                     ;;BIT02
(3)  002222  012737  016650  000034             MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(3)  002230  012737  000300  000036             MOV     #PR6,@#TRAPVEC+2;LEVEL 6
(3)  002236  012737  015006  000024             MOV     #$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  015006  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)                                                                    ;TESTING 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  010200          33$:      JSR     PC,CNTLU
    (2)  002572  105767  176374          3$:       TSTB    INIFLG          ;HAS INITIALIZATION BEEN PERFORMED
    (2)  002576  001004                            BNE     ONCE
    (2)  002600  104401  015146                    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  015214                            MREGAD  ;MESSAGE
    (2)  002710  104410                            PARAM   ;CONVERT STRING
    (2)  002712  174000                            174000  ;LOW LIMIT
    (2)  002714  177776                            177776  ;HIGH LIMIT
    (2)  002716  017144                            DUBASE  ;STORE AT THIS LOCATION
    (2)  002720     001                  .BYTE     1       ;MASK
    (2)  002721     001                  .BYTE     1       ;HOW MANY TIMES * 2
    (1)  002722  016767  014216  176226            MOV     DUBASE,KEEPADD  ;SAVE
    (1)  002730  004767  014056                    JSR     PC,DUADDR
```

7 CNDUT-A MACY11 30(1046) 14-DEC-82 10:01 PAGE 61-16
CNDUTA.M11 30-OCT-82 11:16 INITIALIZE THE COMMON TAGS SEQ 0030

```
(1)  002734  016767  176216  176212          MOV     KEEPADD,BASEADD ;RESTORE FOR ROTATION
(2)  002742  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  002744  015201                          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  015244                          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  015273                          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  015367                          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  013612  OUTMUL: MOV     #300,DUPRT
(1)  003174  004767  013536                  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  015605                          MSYNC   ;MESSAGE
(2)  003204  122767  000061  012734  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  012716  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$:     NOP
(2)  003250  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003252  015653                          MWIRE6  ;MESSAGE
(2)  003254  104414                          SETFLG  ;SET FLAG BASED UPON INPUT STRING
(2)  003256  001147                          SEXMIT  ;THIS FLAG
(2)  003260  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003262  015724                          MWIRE5  ;MESSAGE
(2)  003264  104414                          SETFLG  ;SET FLAG BASED UPON INPUT STRING
(2)  003266  001150                          SEREC   ;THIS FLAG
(2)  003270  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003272  015774                          MWIRE4  ;MESSAGE
(2)  003274  104414                          SETFLG  ;SET FLAG BASED UPON INPUT STRING
(2)  003276  001151                          OPTCLR  ;THIS FLAG
(2)  003300  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(2)  003302  016053                          MEXTJ   ;MESSAGE
(2)  003304  104414                          SETFLG  ;SET FLAG BASED UPON INPUT STRING
(2)  003306  001153                          JMRBY   ;THIS FLAG
(2)
(2)                                          ;TEST START AND RESTART
(2)
(2)  003310  012706  001100          .BEGIN: MOV     #STACK,SP               ;SET UP STACK
(2)  003314  106427  000300                  MTPS    #300                    ;LOCK OUT INTERRUPTS
(2)  003320  032777  000002  176112          BIT     #SW01,@SWR              ;IF SW01=1, GET STARTING PC
(2)  003326  001413                          BEQ     3$
(3)  003330  104406                          INSTR   ;OUTPUT MESSAGE & GET INPUT STRING
(3)  003332  015537                          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$:     MOV     #TST1,RETURN            ;START AT TEST 1
(2)  003364  104401  015533          4$:     TYPE    ,MR                     ;TYPE R
(2)  003370  000177  175510                  JMP     @RETURN                 ;START TESTING
```

```
 8150
 8151                                           ;;THIS TEST CHECKS THE STRIP SYNC FUNCTION
  (1)                                           ;;OF THE RECEIVER LOGIC
  (1)                                           ;;MODE:SYNINT
  (1)                                           ;;LENGTH:EIGHT
  (1)                                           ;;NOTE: RXDONE SHOULD NEVER ASSERT
  (1)                                           ;;CHAR: 26 (SYNC)
  (1)                                           ;;
  (5)                                           ;;*****************************************************************
  (4)  003374  000004                           TST1:   SCOPE
  (4)
  (3)  003376  052777  000400  176322                   BIS     #MRESET,@TXCSR  ;MASTER RESET
  (2)  003404  012777  030000  176310                   MOV     #SYNINT,@PARCSR ;SET THE MODE
  (3)  003412  052777  000400  176306                   BIS     #MRESET,@TXCSR  ;MASTER RESET
  (2)
  (2)                                           ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
  (2)  003420  012777  064001  176300                   MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
  (2)
  (2)                                           ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
  (2)  003426  012777  036026  176266                   MOV     #SYNINT!EIGHT!NOPAR!26,@PARCSR
  (2)  003434  052777  000020  176250                   BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
  (2)                                                   ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
  (2)  003442  042777  020000  176256                   BIC     #CLK,@TXCSR     ;POKE CLK DOWN
  (2)  003450  052777  020000  176250                   BIS     #CLK,@TXCSR     ;POKE CLK UP
  (2)                                           ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
  (2)  003456  042777  020000  176242                   BIC     #CLK,@TXCSR     ;POKE CLK DOWN
  (2)  003464  052777  020000  176234                   BIS     #CLK,@TXCSR     ;POKE CLK UP
  (1)  003472  052777  000400  176212                   BIS     #STPSYN,@RXCSR  ;SET STRIP SYNC
  (1)  003500  012767  000003  175416                   MOV     #3,COUNT        ;# OF SYNC CHARS
  (1)  003506  012767  000026  175764  1$:      MOV     #26.,$TMP1      ;CHAR TO BE SHIFTED
  (1)  003514  012767  000010  175400                   MOV     #8.,SHIFT       ;# OF SHIFTS
  (1)  003522  004767  013420                           JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
  (1)  003526  105777  176160                           TSTB    @RXCSR  ;RXDONE ?
  (1)  003532  100001                                   BPL     .+4
  (1)  003534  104004                                   ERROR   4       ;RXDONE SHOULD NOT BE ASSERTED
  (1)  003536  005367  175362                           DEC     COUNT   ;# OF SYNC CHARS
  (1)  003542  001361                                   BNE     1$
  (1)
 8152                                           ;;THIS TEST PROVES THAT RXERR FREEZES THE "RECEIVER RESET"
  (1)                                           ;;WHILE IN STRIP SYNC MODE
  (1)                                           ;;THIS TEST FIRST PROVES THAT AUTOMATIC RESETS OCCUR WHEN
  (1)                                           ;;STRIP SYNC IS SET & SYNC CHARACTERS ARE SENT
  (1)                                           ;;.....BUT IF AN ERROR SHOULD OCCUR....THIS AUTOMATIC RESET
  (1)                                           ;;IS DISCOMBOBULATED
  (1)                                           ;;IE. FORCE PARITY ERROR WHILE STRIP SYNC IS SET
  (1)                                           ;;NOTE: NORMALLY THE LOGIC RESETS THE RXDONE &ERROR FLAGS
  (1)                                           ;;PROVIDING THAT ONLY GOOD SYNC CHARACTERS ARE SENT....
  (1)                                           ;;BUT, IF AN RXERR OCCURS   RXDONE PLUS RXERR ARE ASSERTED
  (1)                                           ;;MODE: ISOC (ISYMOD)
  (1)                                           ;;LENGTH: EIGHT
  (1)                                           ;;PARITY: EVEPAR
  (1)                                           ;;CHARACTER EXPECTED:26
  (1)                                           ;;CHARACTER SENT: SYNC CHARACTER
  (1)                                           ;;NOTE: THIS TEST USES ONLY THE RECEIVER LOGIC
  (1)                                           ;;
  (5)                                           ;;*****************************************************************
```

```
    (4)  003544  000004                    TST2:   SCOPE
    (4)
    (3)  003546  052777  000400  176152            BIS     #MRESET,@TXCSR  ;MASTER RESET
    (2)  003554  012777  000000  176140            MOV     #ISYMOD,@PARCSR ;SET THE MODE
    (3)  003562  052777  000400  176136            BIS     #MRESET,@TXCSR  ;MASTER RESET
    (2)
    (2)                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
    (2)  003570  012777  064001  176130            MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
    (2)
    (2)                                     ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
    (2)  003576  012777  007426  176116            MOV     #ISYMOD!EIGHT!EVEPAR!26,@PARCSR
    (1)  003604  016703  176106                    MOV     RXDBUF,R3       ;SET UP FOR ERROR MSG
    (1)  003610  012767  000003  175306            MOV     #3,COUNT        ;# OF TIMES SYNC CHAR WILL BE SENT
    (2)  003616  052777  000020  176066            BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
    (2)                                             ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
    (2)  003624  042777  020000  176074            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
    (2)  003632  052777  020000  176066            BIS     #CLK,@TXCSR     ;POKE CLK UP
    (2)                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
    (2)  003640  042777  020000  176060            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
    (2)  003646  052777  020000  176052            BIS     #CLK,@TXCSR     ;POKE CLK UP
    (1)  003654  052777  000400  176030            BIS     #STPSYN,@RXCSR  ;SET STRIP SYNC
    (1)  003662  012767  000013  175232  2$:       MOV     #11.,SHIFT      ;# OF SHIFTS
    (1)  003670  012767  003054  175602            MOV     #3054,$TMP1     ;SYNC CHAR + START&STOP+ PARITY
    (1)  003676  004767  013244          1$:       JSR     PC,RPOKE        ;SHIFT IN THIS CHARACTER
    (1)  003702  105777  176004                    TSTB    @RXCSR  ;RXDONE = 0 ?
    (1)  003706  100001                            BPL     .+4
    (1)  003710  104004                            ERROR   4       ;RXDONE SHOULD NOT BE SET
    (1)  003712  005367  175206                    DEC     COUNT   ;# OF  SYNC CHARS
    (1)  003716  001361                            BNE     2$      ;GO AGAIN ?
    (1)  003720  012700  000026                    MOV     #26,R0  ;EXPECTED
    (1)  003724  017701  175766                    MOV     @RXDBUF,R1      ;ACTUAL
    (1)                                             ;NOTE THAT THIS IS THE FIRST TIME
    (1)                                             ;RXDBUF IS READ.....THERE SHOULD BE
    (1)                                             ;NO OVER RUN ERROR      4S
    (1)  003730  020001                            CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
    (1)  003732  001401                            BEQ     .+4
    (1)  003734  104002                            ERROR   2       ;DATA CHARS SHOULD COMPARE
    (1)                                             ;THERE SHOULD BE NO RXERR'S
    (1)  003736  012767  000004  175160            MOV     #4,COUNT        ;# OF TIMES
    (1)  003744  012700  110026                    MOV     #RXERR!PARER!26,R0      ;EXPECTED
    (1)  003750  012767  002054  175522            MOV     #2054,$TMP1     ;BAD SYNC CHAR (WRONG PARITY)
    (1)  003756  012767  000013  175136  3$:       MOV     #11.,SHIFT      ;# OF SHIFTS
    (1)  003764  004767  013156                    JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
    (1)  003770  105777  175716                    TSTB    @RXCSR  ;RXDONE = 1?
    (1)  003774  100401                            BMI     .+4
    (1)  003776  104004                            ERROR   4       ;RXDONE SHOULD BE SET
    (1)  004000  017701  175712                    MOV     @RXDBUF,R1      ;ACTUAL DATA
    (1)  004004  020001                            CMP     R0,R1   ;COMPARE EXP VS ACT
    (1)  004006  001401                            BEQ     .+4
    (1)  004010  104000                            ERROR   ;DID THE RESPECTIVE ERROR       4 STOP THE
    (1)                                             ;AUTOMATIC RESSETING OF RXDONE & ERROR FLAGS
    (1)                                             ;.....CHECK THIS.....
    (1)  004012  005367  175106                    DEC     COUNT   ;# OF SYNC CHARS
    (1)  004016  001445                            BEQ     5$      ;FINISHED ? GET OUT OF TEST
    (1)  004020  022767  000003  175076            CMP     #3,COUNT        ;# OF SYNC CHARS
    (1)  004026  001423                            BEQ     6$      ;CHECK FRAME ERROR ?
```

```
(1)  004030  022767  000002  175066          CMP     #2,COUNT        ;# OF SYNC CHARS
(1)  004036  001426                          BEQ     7$      ;CHECK FRAME ERROR & BAD PARITY ?
(1)                                          ;NOPE THEN IT (COUNT) MUST BE = 1  THEREFORE....
(1)  004040  012767  000013  175054          MOV     #11.,SHIFT      ;# OF SHIFTS
(1)  004046  012767  000054  175424          MOV     #54,$TMP1       ;FRAME & PARITY ERROR
(1)  004054  004767  013066                  JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)                                  ;NOW DON'T READ THE RXDBUF TO CREATE OVER RUN
(1)  004060  012767  000054  175412          MOV     #54,$TMP1       ;FRAME & PARITY ERROR
(1)  004066  012700  170026                  MOV     #RXERR!OVRRUN!FRMERR!PARER!26,R0        ;EXPECTED
(1)  004072  000167  177660                  JMP     3$      ;DO IT AGAIN
(1)  004076  012767  001054  175374  6$:     MOV     #1054,$TMP1     ;BAD STOP BIT  FOR FRAME ERROR
(1)  004104  012700  120026                  MOV     #RXERR!FRMERR!26,R0     ;EXPECTED
(1)  004110  000167  177642                  JMP     3$      ;DO IT AGAIN
(1)  004114  012767  000054  175356  7$:     MOV     #54,$TMP1       ;BAD STOP BIT & PARITY
(1)  004122  012700  130026                  MOV     #RXERR!FRMERR!PARER!26,R0       ;EXPECTED
(1)  004126  000167  177624                  JMP     3$      ;DO IT AGAIN
(1)  004132                          5$:
8153                                 ;;THIS TEST PROVES THAT RXERR FREEZES THE "RECEIVER RESET"
(1)                                  ;;WHILE IN STRIP SYNC MODE
(1)                                  ;;THIS TEST FIRST PROVES THAT AUTOMATIC RESETS OCCUR WHEN
(1)                                  ;;STRIP SYNC IS SET & SYNC CHARACTERS ARE SENT
(1)                                  ;;.....BUT IF AN ERROR SHOULD OCCUR....THIS AUTOMATIC RESET
(1)                                  ;;IS DISCOMBOBULATED
(1)                                  ;;IE. FORCE PARITY ERROR WHILE STRIP SYNC IS SET
(1)                                  ;;NOTE: NORMALLY THE LOGIC RESETS THE RXDONE &ERROR FLAGS
(1)                                  ;;PROVIDING THAT ONLY GOOD SYNC CHARACTERS ARE SENT....
(1)                                  ;;BUT, IF AN RXERR OCCURS   RXDONE PLUS RXERR ARE ASSERTED
(1)                                  ;;MODE: ISOC (ISYMOD)
(1)                                  ;;LENGTH: SEVEN
(1)                                  ;;PARITY: EVEPAR
(1)                                  ;;CHARACTER EXPECTED:226
(1)                                                  ;NOTE THAT THE PARITY BIT SHOULD SHOW
(1)                                                  ;UP IN THE DATA     IE. BIT SEVEN FOR
(1)                                                  ;SEVEN LEVEL CODE
(1)                                  ;;CHARACTER SENT: SYNC CHARACTER
(1)                                  ;;NOTE: THIS TEST USES ONLY THE RECEIVER LOGIC
(1)                                  ;;
(5)                                  ;;*****************************************************************
(4)  004132  000004                  TST3:   SCOPE
(4)
(3)  004134  052777  000400  175564          BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  004142  012777  000000  175552          MOV     #ISYMOD,@PARCSR ;SET THE MODE
(3)  004150  052777  000400  175550          BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                  ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)  004156  012777  064001  175542          MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)                                  ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2)  004164  012777  005626  175530          MOV     #ISYMOD!SEVEN!EVEPAR!226,@PARCSR
(1)  004172  016703  175520                  MOV     RXDBUF,R3       ;SET UP FOR ERROR MSG
(1)  004176  012767  000003  174720          MOV     #3,COUNT        ;# OF TIMES SYNC CHAR WILL BE SENT
(2)  004204  052777  000020  175500          BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
(2)                                          ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2)  004212  042777  020000  175506          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  004220  052777  020000  175500          BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)                                  ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
```

```
(2)  004226  042777  020000  175472          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  004234  052777  020000  175464          BIS     #CLK,@TXCSR     ;POKE CLK UP
(1)  004242  052777  000400  175442          BIS     #STPSYN,@RXCSR  ;SET STRIP SYNC
(1)  004250  012767  000012  174644  2$:     MOV     #10.,SHIFT      ;# OF SHIFTS
(1)  004256  012767  001454  175214          MOV     #1454,$TMP1     ;SYNC CHAR + START&STOP+ PARITY
(1)  004264  004767  012656          1$:     JSR     PC,RPOKE        ;SHIFT IN THIS CHARACTER
(1)  004270  105777  175416                  TSTB    @RXCSR  ;RXDONE = 0 ?
(1)  004274  100001                          BPL     .+4
(1)  004276  104004                          ERROR   4       ;RXDONE SHOULD NOT BE SET
(1)  004300  005367  174620                  DEC     COUNT   ;# OF  SYNC CHARS
(1)  004304  001361                          BNE     2$      ;GO AGAIN ?
(1)  004306  012700  000226                  MOV     #226,R0 ;EXPECTED
(1)  004312  017701  175400                  MOV     @RXDBUF,R1      ;ACTUAL
(1)                                                          ;NOTE THAT THIS IS THE FIRST TIME
(1)                                                          ;RXDBUF IS READ.......THERE SHOULD BE
(1)                                                          ;NO OVER RUN ERROR      4S
(1)  004316  020001                          CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
(1)  004320  001401                          BEQ     .+4
(1)  004322  104002                          ERROR   2       ;DATA CHARS SHOULD COMPARE
(1)                                                          ;THERE SHOULD BE NO RXERR'S
(1)  004324  012767  000004  174572          MOV     #4,COUNT        ;# OF TIMES
(1)  004332  012700  110026                  MOV     #RXERR!PARER!26,R0      ;EXPECTED
(1)  004336  012767  001054  175134          MOV     #1054,$TMP1     ;BAD SYNC CHAR (WRONG PARITY)
(1)  004344  012767  000012  174550  3$:     MOV     #10.,SHIFT      ;# OF SHIFTS
(1)  004352  004767  012570                  JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)  004356  105777  175330                  TSTB    @RXCSR  ;RXDONE = 1?
(1)  004362  100401                          BMI     .+4
(1)  004364  104004                          ERROR   4       ;RXDONE SHOULD BE SET
(1)  004366  017701  175324                  MOV     @RXDBUF,R1      ;ACTUAL DATA
(1)  004372  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  004374  001401                          BEQ     .+4
(1)  004376  104000                          ERROR   ;DID THE RESPECTIVE ERROR       4 STOP THE
(1)                                                          ;AUTOMATIC RESSETING OF RXDONE & ERROR FLAGS
(1)                                                          ;.....CHECK THIS.....
(1)                                                          ;NOTE THAT THE PARITY BIT SHOULD
(1)                                                          ;SHOW UP IN THE DATA
(1)                                                          ;IE.  BIT SEVEN FOR SEVEN LEVEL CODE
(1)  004400  005367  174520                  DEC     COUNT   ;# OF SYNC CHARS
(1)  004404  001445                          BEQ     5$      ;FINISHED ? GET OUT OF TEST
(1)  004406  022767  000003  174510          CMP     #3,COUNT        ;# OF SYNC CHARS
(1)  004414  001423                          BEQ     6$      ;CHECK FRAME ERROR ?
(1)  004416  022767  000002  174500          CMP     #2,COUNT        ;# OF SYNC CHARS
(1)  004424  001426                          BEQ     7$      ;CHECK FRAME ERROR & BAD PARITY ?
(1)                                          ;NOPE THEN IT (COUNT) MUST BE = 1  THEREFORE....
(1)  004426  012767  000012  174466          MOV     #10.,SHIFT      ;# OF SHIFTS
(1)  004434  012767  000054  175036          MOV     #54,$TMP1       ;FRAME & PARITY ERROR
(1)  004442  004767  012500                  JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)                              ;NOW DON'T READ THE RXDBUF TO CREATE OVER RUN
(1)  004446  012767  000054  175024          MOV     #54,$TMP1       ;FRAME & PARITY ERROR
(1)  004454  012700  170026                  MOV     #RXERR!OVRRUN!FRMERR!PARER!26,R0        ;EXPECTED
(1)  004460  000167  177660                  JMP     3$      ;DO IT AGAIN
(1)  004464  012767  000454  175006  6$:     MOV     #454,$TMP1      ;BAD STOP BIT  FOR FRAME ERROR
(1)  004472  012700  120226                  MOV     #RXERR!FRMERR!226,R0    ;EXPECTED
(1)  004476  000167  177642                  JMP     3$      ;DO IT AGAIN
(1)  004502  012767  000054  174770  7$:     MOV     #54,$TMP1       ;BAD STOP BIT & PARITY
(1)  004510  012700  130026                  MOV     #RXERR!FRMERR!PARER!26,R0       ;EXPECTED
```

```
  (1)  004514  000167  177624                    JMP     3$      ;DO IT AGAIN
  (1)  004520                              5$:
 8154                                      ;;THIS TEST PROVES THAT RXERR FREEZES THE "RECEIVER RESET"
  (1)                                      ;;WHILE IN STRIP SYNC MODE
  (1)                                      ;;THIS TEST FIRST PROVES THAT AUTOMATIC RESETS OCCUR WHEN
  (1)                                      ;;STRIP SYNC IS SET & SYNC CHARACTERS ARE SENT
  (1)                                      ;;.....BUT IF AN ERROR SHOULD OCCUR....THIS AUTOMATIC RESET
  (1)                                      ;;IS DISCOMBOBULATED
  (1)                                      ;;IE. FORCE PARITY ERROR WHILE STRIP SYNC IS SET
  (1)                                      ;;NOTE: NORMALLY THE LOGIC RESETS THE RXDONE &ERROR FLAGS
  (1)                                      ;;PROVIDING THAT ONLY GOOD SYNC CHARACTERS ARE SENT....
  (1)                                      ;;BUT, IF AN RXERR OCCURS   RXDONE PLUS RXERR ARE ASSERTED
  (1)                                      ;;MODE: ISOC (ISYMOD)
  (1)                                      ;;LENGTH: SIX
  (1)                                      ;;PARITY: EVEPAR
  (1)                                      ;;CHARACTER EXPECTED:126
  (1)                                                  ;NOTE THAT THE PARITY BIT SHOULD SHOW
  (1)                                                  ;UP IN THE DATA     IE. BIT SIX FOR
  (1)                                                  ;SIX LEVEL CODE
  (1)                                      ;;CHARACTER SENT: SYNC CHARACTER
  (1)                                      ;;NOTE: THIS TEST USES ONLY THE RECEIVER LOGIC
  (1)                                      ;;
  (5)                                      ;;**********************************************************************
  (4)  004520  000004                      TST4:   SCOPE
  (4)
  (3)  004522  052777  000400  175176              BIS     #MRESET,@TXCSR  ;MASTER RESET
  (2)  004530  012777  000000  175164              MOV     #ISYMOD,@PARCSR ;SET THE MODE
  (3)  004536  052777  000400  175162              BIS     #MRESET,@TXCSR  ;MASTER RESET
  (2)
  (2)                                      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
  (2)  004544  012777  064001  175154              MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
  (2)
  (2)                                      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
  (2)  004552  012777  003526  175142              MOV     #ISYMOD!SIX!EVEPAR!126,@PARCSR
  (1)  004560  016703  175132                      MOV     RXDBUF,R3       ;SET UP FOR ERROR MSG
  (1)  004564  012767  000003  174332              MOV     #3,COUNT        ;# OF TIMES SYNC CHAR WILL BE SENT
  (2)  004572  052777  000020  175112              BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
  (2)                                              ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
  (2)  004600  042777  020000  175120              BIC     #CLK,@TXCSR     ;POKE CLK DOWN
  (2)  004606  052777  020000  175112              BIS     #CLK,@TXCSR     ;POKE CLK UP
  (2)                                      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
  (2)  004614  042777  020000  175104              BIC     #CLK,@TXCSR     ;POKE CLK DOWN
  (2)  004622  052777  020000  175076              BIS     #CLK,@TXCSR     ;POKE CLK UP
  (1)  004630  052777  000400  175054              BIS     #STPSYN,@RXCSR  ;SET STRIP SYNC
  (1)  004636  012767  000011  174256  2$:         MOV     #9.,SHIFT       ;# OF SHIFTS
  (1)  004644  012767  000654  174626              MOV     #654,$TMP1      ;SYNC CHAR + START&STOP+ PARITY
  (1)  004652  004767  012270              1$:     JSR     PC,RPOKE        ;SHIFT IN THIS CHARACTER
  (1)  004656  105777  175030                      TSTB    @RXCSR  ;RXDONE = 0 ?
  (1)  004662  100001                              BPL     .+4
  (1)  004664  104004                              ERROR   4       ;RXDONE SHOULD NOT BE SET
  (1)  004666  005367  174232                      DEC     COUNT   ;# OF  SYNC CHARS
  (1)  004672  001361                              BNE     2$      ;GO AGAIN ?
  (1)  004674  012700  000126                      MOV     #126,R0 ;EXPECTED
  (1)  004700  017701  175012                      MOV     @RXDBUF,R1      ;ACTUAL
  (1)                                                  ;NOTE THAT THIS IS THE FIRST TIME
  (1)                                                  ;RXDBUF IS READ.......THERE SHOULD BE
```

```
(1)
(1)  004704  020001                          CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
(1)  004706  001401                          BEQ     .+4
(1)  004710  104002                          ERROR   2       ;DATA CHARS SHOULD COMPARE
(1)                                                          ;THERE SHOULD BE NO RXERR'S
(1)  004712  012767  000004  174204          MOV     #4,COUNT        ;# OF TIMES
(1)  004720  012700  110026                  MOV     #RXERR!PARER!26,R0      ;EXPECTED
(1)  004724  012767  000454  174546          MOV     #454,$TMP1      ;BAD SYNC CHAR (WRONG PARITY)
(1)  004732  012767  000011  174162  3$:     MOV     #9.,SHIFT       ;# OF SHIFTS
(1)  004740  004767  012202                  JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)  004744  105777  174742                  TSTB    @RXCSR  ;RXDONE = 1?
(1)  004750  100401                          BMI     .+4
(1)  004752  104004                          ERROR   4       ;RXDONE SHOULD BE SET
(1)  004754  017701  174736                  MOV     @RXDBUF,R1      ;ACTUAL DATA
(1)  004760  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  004762  001401                          BEQ     .+4
(1)  004764  104000                          ERROR   ;DID THE RESPECTIVE ERROR       4 STOP THE
(1)                                                          ;AUTOMATIC RESSETING OF RXDONE & ERROR FLAGS
(1)                                                          ;.....CHECK THIS.....
(1)                                                          ;NOTE THAT THE PARITY BIT SHOULD
(1)                                                          ;SHOW UP IN THE DATA
(1)                                                          ;IE.  BIT SIX FOR SIX LEVEL CODE
(1)  004766  005367  174132                  DEC     COUNT   ;# OF SYNC CHARS
(1)  004772  001445                          BEQ     5$      ;FINISHED ? GET OUT OF TEST
(1)  004774  022767  000003  174122          CMP     #3,COUNT        ;# OF SYNC CHARS
(1)  005002  001423                          BEQ     6$      ;CHECK FRAME ERROR ?
(1)  005004  022767  000002  174112          CMP     #2,COUNT        ;# OF SYNC CHARS
(1)  005012  001426                          BEQ     7$      ;CHECK FRAME ERROR & BAD PARITY ?
(1)                                          ;NOPE THEN IT (COUNT) MUST BE = 1  THEREFORE....
(1)  005014  012767  000011  174100          MOV     #9.,SHIFT       ;# OF SHIFTS
(1)  005022  012767  000054  174450          MOV     #54,$TMP1       ;FRAME & PARITY ERROR
(1)  005030  004767  012112                  JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)                                  ;NOW DON'T READ THE RXDBUF TO CREATE OVER RUN
(1)  005034  012767  000054  174436          MOV     #54,$TMP1       ;FRAME & PARITY ERROR
(1)  005042  012700  170026                  MOV     #RXERR!OVRRUN!FRMERR!PARER!26,R0        ;EXPECTED
(1)  005046  000167  177660                  JMP     3$      ;DO IT AGAIN
(1)  005052  012767  000254  174420  6$:     MOV     #254,$TMP1      ;BAD STOP BIT  FOR FRAME ERROR
(1)  005060  012700  120126                  MOV     #RXERR!FRMERR!126,R0    ;EXPECTED
(1)  005064  000167  177642                  JMP     3$      ;DO IT AGAIN
(1)  005070  012767  000054  174402  7$:     MOV     #54,$TMP1       ;BAD STOP BIT & PARITY
(1)  005076  012700  130026                  MOV     #RXERR!FRMERR!PARER!26,R0       ;EXPECTED
(1)  005102  000167  177624                  JMP     3$      ;DO IT AGAIN
(1)  005106                          5$:
8155                                 ;;THIS TEST PROVES THAT RXERR FREEZES THE "RECEIVER RESET"
(1)                                  ;;WHILE IN STRIP SYNC MODE
(1)                                  ;;THIS TEST FIRST PROVES THAT AUTOMATIC RESETS OCCUR WHEN
(1)                                  ;;STRIP SYNC IS SET & SYNC CHARACTERS ARE SENT
(1)                                  ;;.....BUT IF AN ERROR SHOULD OCCUR....THIS AUTOMATIC RESET
(1)                                  ;;IS DISCOMBOBULATED
(1)                                  ;;IE. FORCE PARITY ERROR WHILE STRIP SYNC IS SET
(1)                                  ;;NOTE: NORMALLY THE LOGIC RESETS THE RXDONE &ERROR FLAGS
(1)                                  ;;PROVIDING THAT ONLY GOOD SYNC CHARACTERS ARE SENT....
(1)                                  ;;BUT, IF AN RXERR OCCURS   RXDONE PLUS RXERR ARE ASSERTED
(1)                                  ;;MODE: ISOC (ISYMOD)
(1)                                  ;;LENGTH: FIVE
(1)                                  ;;PARITY: EVEPAR
```

```
(1)                                 ;;CHARACTER EXPECTED:66
(1)                                                     ;NOTE THAT THE PARITY BIT SHOULD SHOW
(1)                                                     ;UP IN THE DATA     IE. BIT FIVE FOR
(1)                                                     ;FIVE LEVEL CODE
(1)                                 ;;CHARACTER SENT: SYNC CHARACTER
(1)                                 ;;NOTE: THIS TEST USES ONLY THE RECEIVER LOGIC
(1)                                 ;;
(5)                                 ;;**********************************************************************
(4)  005106  000004                 TST5:   SCOPE
(4)
(3)  005110  052777  000400  174610         BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  005116  012777  000000  174576         MOV     #ISYMOD,@PARCSR ;SET THE MODE
(3)  005124  052777  000400  174574         BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)  005132  012777  064001  174566         MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)                                 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2)  005140  012777  001466  174554         MOV     #ISYMOD!FIVE!EVEPAR!66,@PARCSR
(1)  005146  016703  174544                 MOV     RXDBUF,R3       ;SET UP FOR ERROR MSG
(1)  005152  012767  000003  173744         MOV     #3,COUNT        ;# OF TIMES SYNC CHAR WILL BE SENT
(2)  005160  052777  000020  174524         BIS     #SYNSCH,@RXCSR  ;SET SYNC SEARCH
(2)                                         ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2)  005166  042777  020000  174532         BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  005174  052777  020000  174524         BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)                                 ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2)  005202  042777  020000  174516         BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(2)  005210  052777  020000  174510         BIS     #CLK,@TXCSR     ;POKE CLK UP
(1)  005216  052777  000400  174466         BIS     #STPSYN,@RXCSR  ;SET STRIP SYNC
(1)  005224  012767  000010  173670  2$:    MOV     #8.,SHIFT       ;# OF SHIFTS
(1)  005232  012767  000354  174240         MOV     #354,$TMP1      ;SYNC CHAR + START&STOP+ PARITY
(1)  005240  004767  011702         1$:     JSR     PC,RPOKE        ;SHIFT IN THIS CHARACTER
(1)  005244  105777  174442                 TSTB    @RXCSR  ;RXDONE = 0 ?
(1)  005250  100001                         BPL     .+4
(1)  005252  104004                         ERROR   4       ;RXDONE SHOULD NOT BE SET
(1)  005254  005367  173644                 DEC     COUNT   ;# OF  SYNC CHARS
(1)  005260  001361                         BNE     2$      ;GO AGAIN ?
(1)  005262  012700  000066                 MOV     #66,R0  ;EXPECTED
(1)  005266  017701  174424                 MOV     @RXDBUF,R1      ;ACTUAL
(1)                                                     ;NOTE THAT THIS IS THE FIRST TIME
(1)                                                     ;RXDBUF IS READ.......THERE SHOULD BE
(1)                                                     ;NO OVER RUN ERROR      4S
(1)  005272  020001                         CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
(1)  005274  001401                         BEQ     .+4
(1)  005276  104002                         ERROR   2       ;DATA CHARS SHOULD COMPARE
(1)                                                     ;THERE SHOULD BE NO RXERR'S
(1)  005300  012767  000004  173616         MOV     #4,COUNT        ;# OF TIMES
(1)  005306  012700  110026                 MOV     #RXERR!PARER!26,R0      ;EXPECTED
(1)  005312  012767  000254  174160         MOV     #254,$TMP1      ;BAD SYNC CHAR (WRONG PARITY)
(1)  005320  012767  000010  173574  3$:    MOV     #8.,SHIFT       ;# OF SHIFTS
(1)  005326  004767  011614                 JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)  005332  105777  174354                 TSTB    @RXCSR  ;RXDONE = 1?
(1)  005336  100401                         BMI     .+4
(1)  005340  104004                         ERROR   4       ;RXDONE SHOULD BE SET
(1)  005342  017701  174350                 MOV     @RXDBUF,R1      ;ACTUAL DATA
(1)  005346  020001                         CMP     R0,R1   ;COMPARE EXP VS ACT
```

```
(1)  005350  001401                          BEQ     .+4
(1)  005352  104000                          ERROR   ;DID THE RESPECTIVE ERROR       4 STOP THE
(1)                                                  ;AUTOMATIC RESSETING OF RXDONE & ERROR FLAGS
(1)                                                  ;.....CHECK THIS.....
(1)                                                  ;NOTE THAT THE PARITY BIT SHOULD
(1)                                                  ;SHOW UP IN THE DATA
(1)                                                  ;IE.  BIT FIVE FOR FIVE LEVEL CODE
(1)  005354  005367  173544                          DEC     COUNT   ;# OF SYNC CHARS
(1)  005360  001445                                  BEQ     5$      ;FINISHED ? GET OUT OF TEST
(1)  005362  022767  000003  173534                  CMP     #3,COUNT        ;# OF SYNC CHARS
(1)  005370  001423                                  BEQ     6$      ;CHECK FRAME ERROR ?
(1)  005372  022767  000002  173524                  CMP     #2,COUNT        ;# OF SYNC CHARS
(1)  005400  001426                                  BEQ     7$      ;CHECK FRAME ERROR & BAD PARITY ?
(1)                                                  ;NOPE THEN IT (COUNT) MUST BE = 1  THEREFORE....
(1)  005402  012767  000010  173512                  MOV     #8.,SHIFT       ;# OF SHIFTS
(1)  005410  012767  000054  174062                  MOV     #54,$TMP1       ;FRAME & PARITY ERROR
(1)  005416  004767  011524                          JSR     PC,RPOKE        ;SHIFT IN THIS CHAR
(1)                                          ;NOW DON'T READ THE RXDBUF TO CREATE OVER RUN
(1)  005422  012767  000054  174050                  MOV     #54,$TMP1       ;FRAME & PARITY ERROR
(1)  005430  012700  170026                          MOV     #RXERR!OVRRUN!FRMERR!PARER!26,R0        ;EXPECTED
(1)  005434  000167  177660                          JMP     3$      ;DO IT AGAIN
(1)  005440  012767  000154  174032  6$:             MOV     #154,$TMP1      ;BAD STOP BIT  FOR FRAME ERROR
(1)  005446  012700  120066                          MOV     #RXERR!FRMERR!66,R0     ;EXPECTED
(1)  005452  000167  177642                          JMP     3$      ;DO IT AGAIN
(1)  005456  012767  000054  174014  7$:             MOV     #54,$TMP1       ;BAD STOP BIT & PARITY
(1)  005464  012700  130026                          MOV     #RXERR!FRMERR!PARER!26,R0       ;EXPECTED
(1)  005470  000167  177624                          JMP     3$      ;DO IT AGAIN
(1)  005474                                  5$:
8156                                                 ;;THIS TEST VERIFYS WORD LENGTH SELECT OF
(1)                                                  ;;THE TRANSMITTER SECTION,IT USES THE DNA FLAG
(1)                                                  ;;AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
(1)                                                  ;;CORRECTLY
(1)                                                  ;;NOTE: DNA COMES UP ON THE FIRST RISING BIT
(1)                                                  ;;EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
(1)                                                  ;;LOADED INTO TXDBUF
(1)                                                  ;;MODE:SYNINT
(1)                                                  ;;PARITY:NO PARITY
(1)                                                  ;;LENGTH:FIVE
(1)                                                  ;;
(5)                                          ;;**************************************************************
(4)  005474  000004                          TST6:   SCOPE
(4)
(3)  005476  052777  000400  174222                  BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  005504  012777  030000  174210                  MOV     #SYNINT,@PARCSR ;SET THE MODE
(3)  005512  052777  000400  174206                  BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                          ;SET MAINTENANCE MODE & SEND
(2)                                                  ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  005520  012777  004020  174200                  MOV     #MINT!SEND,@TXCSR
(2)
(2)                                          ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  005526  012777  030026  174166                  MOV     #SYNINT!FIVE!NOPAR!26,@PARCSR
(1)  005534  016703  174166                          MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  005540  112777  000021  174164                  MOVB    #21,@TXDBUF     ;LOAD CHAR
(1)  005546  012767  000021  173724                  MOV     #21,$TMP1       ;SHIFTED CHAR
(1)  005554  012767  000005  173340                  MOV     #5,SHIFT        ;# OF SHIFTS
```

```
                                ;POKE CLK TO GET INTO SYNCRONIZATION
005562  052777  020000  174136          BIS     #CLK,@TXCSR     ;POKE CLK UP
005570  042777  020000  174130          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
005576  005000                  1$:     CLR     R0
005600  006067  173674                  ROR     $TMP1   ;FORCE CARRY
005604  103002                          BCC     2$
005606  052700  002000                  BIS     #BITW,R0        ;EQUIV OF BIT WINDOW
005612                          2$:
005612  052777  020000  174106          BIS     #CLK,@TXCSR     ;POKE CLK UP
005620  042777  020000  174100          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
005626  017701  174074                  MOV     @TXCSR,R1       ;ACTUAL
005632  042701  075777                  BIC     #075777,R1      ;SAVE BITW & DNA
005636  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
005640  001401                          BEQ     .+4
005642  104003                          ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
                                                        ;BIT,....ALSO CHECK DNA
005644  005367  173252                  DEC     SHIFT   ;# OF SHIFTS
005650  001352                          BNE     1$      ;DO IT AGAIN ?
                                ;NOW POKE CLK TO SEE DNA
005652  052777  020000  174046          BIS     #CLK,@TXCSR     ;POKE CLK
005660  012700  100000                  MOV     #100000,R0      ;EXPECTED
005664  017701  174036                  MOV     @TXCSR,R1       ;ACTUAL
005670  042701  077777                  BIC     #77777,R1       ;SAVE DNA ONLY
005674  020001                          CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
005676  001401                          BEQ     .+4
005700  104003                          ERROR   3       ;DNA SHOULD BE SET
                                                        ;IF DNA DID NOT SET ,CHECK WORD LENGTH
                                                        ;SELECT LOGIC OF THE TRANSMITTER
005702  005777  174020                  TST     @TXCSR  ;DNA ?
005706  100001                          BPL     .+4
005710  104004                          ERROR   4               ;DNA SHOULD NOT BE SET
                                                        ;IT SHOULD HAVE BEEN CLEARED FROM
                                                        ;PREVIOUS READ

                                ;;THIS TEST VERIFYS WORD LENGTH SELECT OF
                                ;;THE TRANSMITTER SECTION,IT USES THE DNA FLAG
                                ;;AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
                                ;;CORRECTLY
                                ;;NOTE: DNA COMES UP ON THE FIRST RISING BIT
                                ;;EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
                                ;;LOADED INTO TXDBUF
                                ;;MODE:SYNINT
                                ;;PARITY:NO PARITY
                                ;;LENGTH:SIX
                                ;;
                        ;;**************************************************************
005712  000004         TST7:   SCOPE

005714  052777  000400  174004          BIS     #MRESET,@TXCSR  ;MASTER RESET
005722  012777  030000  173772          MOV     #SYNINT,@PARCSR ;SET THE MODE
005730  052777  000400  173770          BIS     #MRESET,@TXCSR  ;MASTER RESET

                                ;SET MAINTENANCE MODE & SEND
                                        ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
005736  012777  004020  173762          MOV     #MINT!SEND,@TXCSR
```

```
28
    (2)
    (2)  005744  012777  032026  173750          ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
    (1)  005752  016703  173750                  MOV     #SYNINT!SIX!NOPAR!26,@PARCSR
    (1)  005756  112777  000021  173746          MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
    (1)  005764  012767  000021  173506          MOVB    #21,@TXDBUF     ;LOAD CHAR
    (1)  005772  012767  000006  173122          MOV     #21,$TMP1       ;SHIFTED CHAR
    (1)                                          MOV     #6,SHIFT        ;# OF SHIFTS
    (1)                                  ;POKE CLK TO GET INTO SYNCRONIZATION
    (2)  006000  052777  020000  173720          BIS     #CLK,@TXCSR     ;POKE CLK UP
    (2)  006006  042777  020000  173712          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
    (1)  006014  005000                  1$:     CLR     R0
    (1)  006016  006067  173456                  ROR     $TMP1   ;FORCE CARRY
    (1)  006022  103002                          BCC     2$
    (1)  006024  052700  002000                  BIS     #BITW,R0        ;EQUIV OF BIT WINDOW
    (1)  006030                          2$:
    (2)  006030  052777  020000  173670          BIS     #CLK,@TXCSR     ;POKE CLK UP
    (2)  006036  042777  020000  173662          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
    (1)  006044  017701  173656                  MOV     @TXCSR,R1       ;ACTUAL
    (1)  006050  042701  075777                  BIC     #075777,R1      ;SAVE BITW & DNA
    (1)  006054  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
    (1)  006056  001401                          BEQ     .+4
    (1)  006060  104003                          ERROR   3               ;BIT WINDOW DID NOT MATCH ACTUAL DATA
    (1)                                                                  ;BIT....ALSO CHECK DNA
    (1)  006062  005367  173034                  DEC     SHIFT   ;# OF SHIFTS
    (1)  006066  001352                          BNE     1$      ;DO IT AGAIN ?
    (1)                                  ;NOW POKE CLK TO SEE DNA
    (1)  006070  052777  020000  173630          BIS     #CLK,@TXCSR     ;POKE CLK
    (1)  006076  012700  100000                  MOV     #100000,R0      ;EXPECTED
    (1)  006102  017701  173620                  MOV     @TXCSR,R1       ;ACTUAL
    (1)  006106  042701  077777                  BIC     #77777,R1       ;SAVE DNA ONLY
    (1)  006112  020001                          CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
    (1)  006114  001401                          BEQ     .+4
    (1)  006116  104003                          ERROR   3               ;DNA SHOULD BE SET
    (1)                                                                  ;IF DNA DID NOT SET ,CHECK WORD LENGTH
    (1)                                                                  ;SELECT LOGIC OF THE TRANSMITTER
    (1)  006120  005777  173602                  TST     @TXCSR  ;DNA ?
    (1)  006124  100001                          BPL     .+4
    (1)  006126  104004                          ERROR   4                       ;DNA SHOULD NOT BE SET
    (1)                                                                  ;IT SHOULD HAVE BEEN CLEARED FROM
    (1)                                                                  ;PREVIOUS READ
    (1)
 8158                                            ;;THIS TEST VERIFYS WORD LENGTH SELECT OF
    (1)                                          ;;THE TRANSMITTER SECTION,IT USES THE DNA FLAG
    (1)                                          ;;AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
    (1)                                          ;;CORRECTLY
    (1)                                          ;;NOTE: DNA COMES UP ON THE FIRST RISING BIT
    (1)                                          ;;EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
    (1)                                          ;;LOADED INTO TXDBUF
    (1)                                          ;;MODE:SYNINT
    (1)                                          ;;PARITY:NO PARITY
    (1)                                          ;;LENGTH:SEVEN
    (1)                                          ;;
    (5)                                  ;;**********************************************************************
    (4)  006130  000004                  TST10:  SCOPE
    (4)
    (3)  006132  052777  000400  173566          BIS     #MRESET,@TXCSR  ;MASTER RESET
    (2)  006140  012777  030000  173554          MOV     #SYNINT,@PARCSR ;SET THE MODE
```

```
29
    (3)  006146  052777  000400  173552          BIS     #MRESET,@TXCSR  ;MASTER RESET
    (2)
    (2)                                  ;SET MAINTENANCE MODE & SEND
    (2)                                          ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
    (2)  006154  012777  004020  173544          MOV     #MINT!SEND,@TXCSR
    (2)
    (2)                                  ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
    (2)  006162  012777  034026  173532          MOV     #SYNINT!SEVEN!NOPAR!26,@PARCSR
    (1)  006170  016703  173532                  MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
    (1)  006174  112777  000021  173530          MOVB    #21,@TXDBUF     ;LOAD CHAR
    (1)  006202  012767  000021  173270          MOV     #21,$TMP1       ;SHIFTED CHAR
    (1)  006210  012767  000007  172704          MOV     #7,SHIFT        ;# OF SHIFTS
    (1)                                  ;POKE CLK TO GET INTO SYNCRONIZATION
    (2)  006216  052777  020000  173502          BIS     #CLK,@TXCSR     ;POKE CLK UP
    (2)  006224  042777  020000  173474          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
    (1)  006232  005000                  1$:     CLR     R0
    (1)  006234  006067  173240                  ROR     $TMP1   ;FORCE CARRY
    (1)  006240  103002                          BCC     2$
    (1)  006242  052700  002000                  BIS     #BITW,R0        ;EQUIV OF BIT WINDOW
    (1)  006246                          2$:
    (2)  006246  052777  020000  173452          BIS     #CLK,@TXCSR     ;POKE CLK UP
    (2)  006254  042777  020000  173444          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
    (1)  006262  017701  173440                  MOV     @TXCSR,R1       ;ACTUAL
    (1)  006266  042701  075777                  BIC     #075777,R1      ;SAVE BITW & DNA
    (1)  006272  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
    (1)  006274  001401                          BEQ     .+4
    (1)  006276  104003                          ERROR   3               ;BIT WINDOW DID NOT MATCH ACTUAL DATA
    (1)                                                                  ;BIT,....ALSO CHECK DNA
    (1)  006300  005367  172616                  DEC     SHIFT   ;# OF SHIFTS
    (1)  006304  001352                          BNE     1$      ;DO IT AGAIN ?
    (1)                                  ;NOW POKE CLK TO SEE DNA
    (1)  006306  052777  020000  173412          BIS     #CLK,@TXCSR     ;POKE CLK
    (1)  006314  012700  100000                  MOV     #100000,R0      ;EXPECTED
    (1)  006320  017701  173402                  MOV     @TXCSR,R1       ;ACTUAL
    (1)  006324  042701  077777                  BIC     #77777,R1       ;SAVE DNA ONLY
    (1)  006330  020001                          CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
    (1)  006332  001401                          BEQ     .+4
    (1)  006334  104003                          ERROR   3               ;DNA SHOULD BE SET
    (1)                                                                  ;IF DNA DID NOT SET ,CHECK WORD LENGTH
    (1)                                                                  ;SELECT LOGIC OF THE TRANSMITTER
    (1)  006336  005777  173364                  TST     @TXCSR  ;DNA ?
    (1)  006342  100001                          BPL     .+4
    (1)  006344  104004                          ERROR   4               ;DNA SHOULD NOT BE SET
    (1)                                                                  ;IT SHOULD HAVE BEEN CLEARED FROM
    (1)                                                                  ;PREVIOUS READ
    (1)
 8159                                    ;;THIS TEST VERIFYS WORD LENGTH SELECT OF
    (1)                                  ;;THE TRANSMITTER SECTION,IT USES THE DNA FLAG
    (1)                                  ;;AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
    (1)                                  ;;CORRECTLY
    (1)                                  ;;NOTE: DNA COMES UP ON THE FIRST RISING BIT
    (1)                                  ;;EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
    (1)                                  ;;LOADED INTO TXDBUF
    (1)                                  ;;MODE:SYNINT
    (1)                                  ;;PARITY:NO PARITY
    (1)                                  ;;LENGTH:EIGHT
```

```
                                        ;;
                                ;;*****************************************************************
006346  000004                 TST11:  SCOPE
                                
006350  052777  000400  173350          BIS     #MRESET,@TXCSR  ;MASTER RESET
006356  012777  030000  173336          MOV     #SYNINT,@PARCSR ;SET THE MODE
006364  052777  000400  173334          BIS     #MRESET,@TXCSR  ;MASTER RESET
                                
                                ;SET MAINTENANCE MODE & SEND
                                        ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
006372  012777  004020  173326          MOV     #MINT!SEND,@TXCSR
                                
                                ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
006400  012777  036026  173314          MOV     #SYNINT!EIGHT!NOPAR!26,@PARCSR
006406  016703  173314                  MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
006412  112777  000021  173312          MOVB    #21,@TXDBUF     ;LOAD CHAR
006420  012767  000021  173052          MOV     #21,$TMP1       ;SHIFTED CHAR
006426  012767  000010  172466          MOV     #8.,SHIFT       ;# OF SHIFTS
                                ;POKE CLK TO GET INTO SYNCRONIZATION
006434  052777  020000  173264          BIS     #CLK,@TXCSR     ;POKE CLK UP
006442  042777  020000  173256          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
006450  005000                  1$:     CLR     R0
006452  006067  173022                  ROR     $TMP1   ;FORCE CARRY
006456  103002                  BCC     2$
006460  052700  002000                  BIS     #BITW,R0        ;EQUIV OF BIT WINDOW
006464                          2$:
006464  052777  020000  173234          BIS     #CLK,@TXCSR     ;POKE CLK UP
006472  042777  020000  173226          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
006500  017701  173222                  MOV     @TXCSR,R1       ;ACTUAL
006504  042701  075777                  BIC     #075777,R1      ;SAVE BITW & DNA
006510  020001                  CMP     R0,R1   ;COMPARE EXP VS ACT
006512  001401                  BEQ     .+4
006514  104003                  ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
                                                ;BIT,....ALSO CHECK DNA
006516  005367  172400                  DEC     SHIFT   ;# OF SHIFTS
006522  001352                  BNE     1$      ;DO IT AGAIN ?
                                ;NOW POKE CLK TO SEE DNA
006524  052777  020000  173174          BIS     #CLK,@TXCSR     ;POKE CLK
006532  012700  100000                  MOV     #100000,R0      ;EXPECTED
006536  017701  173164                  MOV     @TXCSR,R1       ;ACTUAL
006542  042701  077777                  BIC     #77777,R1       ;SAVE DNA ONLY
006546  020001                  CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
006550  001401                  BEQ     .+4
006552  104003                  ERROR   3       ;DNA SHOULD BE SET
                                                ;IF DNA DID NOT SET ,CHECK WORD LENGTH
                                                ;SELECT LOGIC OF THE TRANSMITTER
006554  005777  173146                  TST     @TXCSR  ;DNA ?
006560  100001                  BPL     .+4
006562  104004                  ERROR   4               ;DNA SHOULD NOT BE SET
                                                ;IT SHOULD HAVE BEEN CLEARED FROM
                                                ;PREVIOUS READ
                                
                                ;;THIS TEST VERIFYS WORD LENGTH SELECT OF
                                ;;THE TRANSMITTER SECTION,IT USES THE DNA FLAG
                                ;;AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
                                ;;CORRECTLY
```

```
(1)                                     ;;NOTE: DNA COMES UP ON THE FIRST RISING BIT
(1)                                     ;;EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
(1)                                     ;;LOADED INTO TXDBUF
(1)                                     ;;MODE:SYNEXT
(1)                                     ;;PARITY:NO PARITY
(1)                                     ;;LENGTH:FIVE
(1)                                     ;;
(5)                                     ;;*****************************************************************
(4)  006564  000004                     TST12:  SCOPE
(4)
(3)  006566  052777  000400  173132             BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  006574  012777  020000  173120             MOV     #SYNEXT,@PARCSR ;SET THE MODE
(3)  006602  052777  000400  173116             BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                     ;SET MAINTENANCE MODE & SEND
(2)                                             ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  006610  012777  004020  173110             MOV     #MINT!SEND,@TXCSR
(2)
(2)                                     ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  006616  012777  020026  173076             MOV     #SYNEXT!FIVE!NOPAR!26,@PARCSR
(1)  006624  016703  173076                     MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  006630  112777  000021  173074             MOVB    #21,@TXDBUF     ;LOAD CHAR
(1)  006636  012767  000021  172634             MOV     #21,$TMP1       ;SHIFTED CHAR
(1)  006644  012767  000005  172250             MOV     #5,SHIFT        ;# OF SHIFTS
(1)                                     ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  006652  052777  020000  173046             BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  006660  042777  020000  173040             BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  006666  005000                     1$:     CLR     R0
(1)  006670  006067  172604                     ROR     $TMP1   ;FORCE CARRY
(1)  006674  103002                             BCC     2$
(1)  006676  052700  002000                     BIS     #BITW,R0        ;EQUIV OF BIT WINDOW
(1)  006702                             2$:
(2)  006702  052777  020000  173016             BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  006710  042777  020000  173010             BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  006716  017701  173004                     MOV     @TXCSR,R1       ;ACTUAL
(1)  006722  042701  075777                     BIC     #075777,R1      ;SAVE BITW & DNA
(1)  006726  020001                             CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  006730  001401                             BEQ     .+4
(1)  006732  104003                             ERROR   3               ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                                     ;BIT,....ALSO CHECK DNA
(1)  006734  005367  172162                     DEC     SHIFT   ;# OF SHIFTS
(1)  006740  001352                             BNE     1$      ;DO IT AGAIN ?
(1)                                     ;NOW POKE CLK TO SEE DNA
(1)  006742  052777  020000  172756             BIS     #CLK,@TXCSR     ;POKE CLK
(1)  006750  012700  100000                     MOV     #100000,R0      ;EXPECTED
(1)  006754  017701  172746                     MOV     @TXCSR,R1       ;ACTUAL
(1)  006760  042701  077777                     BIC     #77777,R1       ;SAVE DNA ONLY
(1)  006764  020001                             CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
(1)  006766  001401                             BEQ     .+4
(1)  006770  104003                             ERROR   3               ;DNA SHOULD BE SET
(1)                                                                     ;IF DNA DID NOT SET ,CHECK WORD LENGTH
(1)                                                                     ;SELECT LOGIC OF THE TRANSMITTER
(1)  006772  005777  172730                     TST     @TXCSR  ;DNA ?
(1)  006776  100001                             BPL     .+4
(1)  007000  104004                             ERROR   4               ;DNA SHOULD NOT BE SET
(1)                                                                     ;IT SHOULD HAVE BEEN CLEARED FROM
```

```
                                                ;PREVIOUS READ
                                        ;;THIS TEST VERIFYS WORD LENGTH SELECT OF
                                        ;;THE TRANSMITTER SECTION,IT USES THE DNA FLAG
                                        ;;AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
                                        ;;CORRECTLY
                                        ;;NOTE: DNA COMES UP ON THE FIRST RISING BIT
                                        ;;EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
                                        ;;LOADED INTO TXDBUF
                                        ;;MODE:SYNEXT
                                        ;;PARITY:NO PARITY
                                        ;;LENGTH:SIX
                                        ;;
                                ;;*****************************************************************
007002  000004                    TST13:  SCOPE
007004  052777  000400  172714            BIS     #MRESET,@TXCSR  ;MASTER RESET
007012  012777  020000  172702            MOV     #SYNEXT,@PARCSR ;SET THE MODE
007020  052777  000400  172700            BIS     #MRESET,@TXCSR  ;MASTER RESET
                                ;SET MAINTENANCE MODE & SEND
                                        ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
007026  012777  004020  172672            MOV     #MINT!SEND,@TXCSR
                                ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
007034  012777  022026  172660            MOV     #SYNEXT!SIX!NOPAR!26,@PARCSR
007042  016703  172660                    MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
007046  112777  000021  172656            MOVB    #21,@TXDBUF     ;LOAD CHAR
007054  012767  000021  172416            MOV     #21,$TMP1       ;SHIFTED CHAR
007062  012767  000006  172032            MOV     #6,SHIFT        ;# OF SHIFTS
                                ;POKE CLK TO GET INTO SYNCRONIZATION
007070  052777  020000  172630            BIS     #CLK,@TXCSR     ;POKE CLK UP
007076  042777  020000  172622            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
007104  005000                    1$:     CLR     R0
007106  006067  172366                    ROR     $TMP1   ;FORCE CARRY
007112  103002                            BCC     2$
007114  052700  002000                    BIS     #BITW,R0        ;EQUIV OF BIT WINDOW
007120                            2$:
007120  052777  020000  172600            BIS     #CLK,@TXCSR     ;POKE CLK UP
007126  042777  020000  172572            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
007134  017701  172566                    MOV     @TXCSR,R1       ;ACTUAL
007140  042701  075777                    BIC     #075777,R1      ;SAVE BITW & DNA
007144  020001                            CMP     R0,R1   ;COMPARE EXP VS ACT
007146  001401                            BEQ     .+4
007150  104003                            ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
                                                        ;BIT,....ALSO CHECK DNA
007152  005367  171744                    DEC     SHIFT   ;# OF SHIFTS
007156  001352                            BNE     1$      ;DO IT AGAIN ?
                                ;NOW POKE CLK TO SEE DNA
007160  052777  020000  172540            BIS     #CLK,@TXCSR     ;POKE CLK
007166  012700  100000                    MOV     #100000,R0      ;EXPECTED
007172  017701  172530                    MOV     @TXCSR,R1       ;ACTUAL
007176  042701  077777                    BIC     #77777,R1       ;SAVE DNA ONLY
007202  020001                            CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
007204  001401                            BEQ     .+4
007206  104003                            ERROR   3       ;DNA SHOULD BE SET
```

```
(1)                                                     ;IF DNA DID NOT SET ,CHECK WORD LENGTH
(1)                                                     ;SELECT LOGIC OF THE TRANSMITTER
(1)  007210  005777  172512                     TST     @TXCSR  ;DNA ?
(1)  007214  100001                             BPL     .+4
(1)  007216  104004                             ERROR   4               ;DNA SHOULD NOT BE SET
(1)                                                     ;IT SHOULD HAVE BEEN CLEARED FROM
(1)                                                     ;PREVIOUS READ
(1)
8162                                            ;;THIS TEST VERIFYS WORD LENGTH SELECT OF
(1)                                             ;;THE TRANSMITTER SECTION,IT USES THE DNA FLAG
(1)                                             ;;AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
(1)                                             ;;CORRECTLY
(1)                                             ;;NOTE: DNA COMES UP ON THE FIRST RISING BIT
(1)                                             ;;EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
(1)                                             ;;LOADED INTO TXDBUF
(1)                                             ;;MODE:SYNEXT
(1)                                             ;;PARITY:NO PARITY
(1)                                             ;;LENGTH:SEVEN
(1)                                             ;;
(5)                                     ;;**************************************************************
(4)  007220  000004                     TST14:  SCOPE
(4)
(3)  007222  052777  000400  172476             BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  007230  012777  020000  172464             MOV     #SYNEXT,@PARCSR ;SET THE MODE
(3)  007236  052777  000400  172462             BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                     ;SET MAINTENANCE MODE & SEND
(2)                                             ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  007244  012777  004020  172454             MOV     #MINT!SEND,@TXCSR
(2)
(2)                                     ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  007252  012777  024026  172442             MOV     #SYNEXT!SEVEN!NOPAR!26,@PARCSR
(1)  007260  016703  172442                     MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  007264  112777  000021  172440             MOVB    #21,@TXDBUF     ;LOAD CHAR
(1)  007272  012767  000021  172200             MOV     #21,$TMP1       ;SHIFTED CHAR
(1)  007300  012767  000007  171614             MOV     #7,SHIFT        ;# OF SHIFTS
(1)                                     ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  007306  052777  020000  172412             BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  007314  042777  020000  172404             BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  007322  005000                     1$:     CLR     R0
(1)  007324  006067  172150                     ROR     $TMP1   ;FORCE CARRY
(1)  007330  103002                             BCC     2$
(1)  007332  052700  002000                     BIS     #BITW,R0        ;EQUIV OF BIT WINDOW
(1)  007336                             2$:
(2)  007336  052777  020000  172362             BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  007344  042777  020000  172354             BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  007352  017701  172350                     MOV     @TXCSR,R1       ;ACTUAL
(1)  007356  042701  075777                     BIC     #075777,R1      ;SAVE BITW & DNA
(1)  007362  020001                             CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  007364  001401                             BEQ     .+4
(1)  007366  104003                             ERROR   3               ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                     ;BIT,....ALSO CHECK DNA
(1)  007370  005367  171526                     DEC     SHIFT   ;# OF SHIFTS
(1)  007374  001352                             BNE     1$      ;DO IT AGAIN ?
(1)
(1)                                     ;NOW POKE CLK TO SEE DNA
(1)  007376  052777  020000  172322             BIS     #CLK,@TXCSR     ;POKE CLK
```

```
(1)  007404  012700  100000                MOV     #100000,R0      ;EXPECTED
(1)  007410  017701  172312                MOV     @TXCSR,R1       ;ACTUAL
(1)  007414  042701  077777                BIC     #77777,R1       ;SAVE DNA ONLY
(1)  007420  020001                        CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
(1)  007422  001401                        BEQ     .+4
(1)  007424  104003                        ERROR   3               ;DNA SHOULD BE SET
(1)                                                                ;IF DNA DID NOT SET ,CHECK WORD LENGTH
(1)                                                                ;SELECT LOGIC OF THE TRANSMITTER
(1)  007426  005777  172274                TST     @TXCSR  ;DNA ?
(1)  007432  100001                        BPL     .+4
(1)  007434  104004                        ERROR   4               ;DNA SHOULD NOT BE SET
(1)                                                                ;IT SHOULD HAVE BEEN CLEARED FROM
(1)                                                                ;PREVIOUS READ
(1)
8163                                       ;;THIS TEST VERIFYS WORD LENGTH SELECT OF
(1)                                        ;;THE TRANSMITTER SECTION,IT USES THE DNA FLAG
(1)                                        ;;AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
(1)                                        ;;CORRECTLY
(1)                                        ;;NOTE: DNA COMES UP ON THE FIRST RISING BIT
(1)                                        ;;EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
(1)                                        ;;LOADED INTO TXDBUF
(1)                                        ;;MODE:SYNEXT
(1)                                        ;;PARITY:NO PARITY
(1)                                        ;;LENGTH:EIGHT
(1)                                        ;;
(5)                                    ;;**********************************************************************
(4)  007436  000004                    TST15:  SCOPE
(3)  007440  052777  000400  172260            BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  007446  012777  020000  172246            MOV     #SYNEXT,@PARCSR ;SET THE MODE
(3)  007454  052777  000400  172244            BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                    ;SET MAINTENANCE MODE & SEND
(2)                                            ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  007462  012777  004020  172236            MOV     #MINT!SEND,@TXCSR
(2)
(2)                                    ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  007470  012777  026026  172224            MOV     #SYNEXT!EIGHT!NOPAR!26,@PARCSR
(1)  007476  016703  172224                    MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  007502  112777  000021  172222            MOVB    #21,@TXDBUF     ;LOAD CHAR
(1)  007510  012767  000021  171762            MOV     #21,$TMP1       ;SHIFTED CHAR
(1)  007516  012767  000010  171376            MOV     #8.,SHIFT       ;# OF SHIFTS
(1)                                    ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  007524  052777  020000  172174            BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  007532  042777  020000  172166            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  007540  005000                    1$:     CLR     R0
(1)  007542  006067  171732                    ROR     $TMP1   ;FORCE CARRY
(1)  007546  103002                            BCC     2$
(1)  007550  052700  002000                    BIS     #BITW,R0        ;EQUIV OF BIT WINDOW
(1)  007554                            2$:
(2)  007554  052777  020000  172144            BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  007562  042777  020000  172136            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  007570  017701  172132                    MOV     @TXCSR,R1       ;ACTUAL
(1)  007574  042701  075777                    BIC     #075777,R1      ;SAVE BITW & DNA
(1)  007600  020001                            CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  007602  001401                            BEQ     .+4
```

```
(1)  007604  104003                          ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                         ;BIT,....ALSO CHECK DNA
(1)  007606  005367  171310                   DEC     SHIFT   ;# OF SHIFTS
(1)  007612  001352                           BNE     1$      ;DO IT AGAIN ?
(1)                                ;NOW POKE CLK TO SEE DNA
(1)  007614  052777  020000  172104           BIS     #CLK,@TXCSR     ;POKE CLK
(1)  007622  012700  100000                   MOV     #100000,R0      ;EXPECTED
(1)  007626  017701  172074                   MOV     @TXCSR,R1       ;ACTUAL
(1)  007632  042701  077777                   BIC     #77777,R1       ;SAVE DNA ONLY
(1)  007636  020001                           CMP     R0,R1   ;COMPARE EXPECTED VS ACTUAL
(1)  007640  001401                           BEQ     .+4
(1)  007642  104003                           ERROR   3       ;DNA SHOULD BE SET
(1)                                                         ;IF DNA DID NOT SET ,CHECK WORD LENGTH
(1)                                                         ;SELECT LOGIC OF THE TRANSMITTER
(1)  007644  005777  172056                   TST     @TXCSR  ;DNA ?
(1)  007650  100001                           BPL     .+4
(1)  007652  104004                           ERROR   4               ;DNA SHOULD NOT BE SET
(1)                                                         ;IT SHOULD HAVE BEEN CLEARED FROM
(1)                                                         ;PREVIOUS READ
(1)
8164                                          ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
(1)                                           ;;OF THE TRANSMITTER SECTION.
(1)                                           ;;IT ALSO CHECKS DNA TIMING
(1)                                           ;;MODE:SYNINT
(1)                                           ;;LENGTH:FIVE PLUS PARITY
(1)                                           ;;PARITY:EVEPAR
(1)                                           ;;CHARACTER:25
(1)                                           ;;
(5)                                           ;;*****************************************************************
(4)  007654  000004                           TST16:  SCOPE
(4)
(3)  007656  052777  000400  172042                   BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  007664  012777  030000  172030                   MOV     #SYNINT,@PARCSR ;SET THE MODE
(3)  007672  052777  000400  172026                   BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                           ;SET MAINTENANCE MODE & SEND
(2)                                                   ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  007700  012777  004020  172020                   MOV     #MINT!SEND,@TXCSR
(2)
(2)                                           ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  007706  012777  031426  172006                   MOV     #SYNINT!FIVE!EVEPAR!26,@PARCSR
(1)  007714  016703  172006                   MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  007720  112777  000025  172004           MOVB    #25,@TXDBUF     ;LOAD DATA CHAR
(1)  007726  012767  000065  171544           MOV     #65,$TMP1       ;TO BE SHIFTED CHAR
(1)  007734  012767  000006  171160           MOV     #6,SHIFT        ;# OF SHIFTS
(1)                                           ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  007742  052777  020000  171756           BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  007750  042777  020000  171750           BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  007756  005000                           1$:     CLR     R0
(1)  007760  006067  171514                   ROR     $TMP1   ;FORCE CARRY
(1)  007764  103002                           BCC     2$      ;BR IF CARRY CLR
(1)  007766  052700  002000                   BIS     #BITW,R0        ;EQUIV OF BITW
(1)  007772                                   2$:
(2)  007772  052777  020000  171726           BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  010000  042777  020000  171720           BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  010006  017701  171714                   MOV     @TXCSR,R1       ;ACTUAL
```

```
(1)  010012  042701  075777                BIC     #075777,R1      ;SAVE BITW & DNA
(1)  010016  020001                        CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  010020  001401                        BEQ     .+4
(1)  010022  104003                        ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                        ;BIT....ALSO CHECK DNA
(1)  010024  005367  171072                DEC     SHIFT   ;# OF SHIFTS
(1)  010030  001352                        BNE     1$      ;DO IT AGAIN ?
(1)                                ;NOW POKE CLK TO SEE DNA
(1)  010032  052777  020000  171666        BIS     #CLK,@TXCSR     ;POKE CLK
(1)  010040  012700  100000                MOV     #100000,R0      ;EXPECTED
(1)  010044  017701  171656                MOV     @TXCSR,R1       ;ACTUAL
(1)  010050  042701  077777                BIC     #77777,R1       ;SAVE DNA ONLY
(1)  010054  020001                        CMP     R0,R1           ;COMPARE EXP VS ACT
(1)  010056  001401                        BEQ     .+4
(1)  010060  104003                        ERROR   3       ;DNA SHOULD BE SET
(1)                                                        ;IF DNA DID NOT SET
(1)                                                        ;CHECK WORD LENGTH SELECT LOGIC
(1)
8165                                       ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
(1)                                        ;;OF THE TRANSMITTER SECTION.
(1)                                        ;;IT ALSO CHECKS DNA TIMING
(1)                                        ;;MODE:SYNINT
(1)                                        ;;LENGTH:FIVE PLUS PARITY
(1)                                        ;;PARITY:ODDPAR
(1)                                        ;;CHARACTER:25
(1)                                        ;;
(5)                                ;;**************************************************************
(4)  010062  000004                TST17:  SCOPE
(3)  010064  052777  000400  171634        BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  010072  012777  030000  171622        MOV     #SYNINT,@PARCSR ;SET THE MODE
(3)  010100  052777  000400  171620        BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                ;SET MAINTENANCE MODE & SEND
(2)                                        ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  010106  012777  004020  171612        MOV     #MINT!SEND,@TXCSR
(2)
(2)                                ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  010114  012777  031026  171600        MOV     #SYNINT!FIVE!ODDPAR!26,@PARCSR
(1)  010122  016703  171600                MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  010126  112777  000025  171576        MOVB    #25,@TXDBUF     ;LOAD DATA CHAR
(1)  010134  012767  000025  171336        MOV     #25,$TMP1       ;TO BE SHIFTED CHAR
(1)  010142  012767  000006  170752        MOV     #6,SHIFT        ;# OF SHIFTS
(1)                                ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  010150  052777  020000  171550        BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  010156  042777  020000  171542        BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  010164  005000                1$:     CLR     R0
(1)  010166  006067  171306                ROR     $TMP1   ;FORCE CARRY
(1)  010172  103002                        BCC     2$      ;BR IF CARRY CLR
(1)  010174  052700  002000                BIS     #BITW,R0        ;EQUIV OF BITW
(1)  010200                        2$:
(2)  010200  052777  020000  171520        BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  010206  042777  020000  171512        BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  010214  017701  171506                MOV     @TXCSR,R1       ;ACTUAL
(1)  010220  042701  075777                BIC     #075777,R1      ;SAVE BITW & DNA
(1)  010224  020001                        CMP     R0,R1   ;COMPARE EXP VS ACT
```

```
(1)  010226  001401                          BEQ     .+4
(1)  010230  104003                          ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                          ;BIT,...ALSO CHECK DNA
(1)  010232  005367  170664                  DEC     SHIFT   ;# OF SHIFTS
(1)  010236  001352                          BNE     1$      ;DO IT AGAIN ?
(1)                                  ;NOW POKE CLK TO SEE DNA
(1)  010240  052777  020000  171460          BIS     #CLK,@TXCSR     ;POKE CLK
(1)  010246  012700  100000                  MOV     #100000,R0      ;EXPECTED
(1)  010252  017701  171450                  MOV     @TXCSR,R1       ;ACTUAL
(1)  010256  042701  077777                  BIC     #77777,R1       ;SAVE DNA ONLY
(1)  010262  020001                          CMP     R0,R1           ;COMPARE EXP VS ACT
(1)  010264  001401                          BEQ     .+4
(1)  010266  104003                          ERROR   3       ;DNA SHOULD BE SET
(1)                                                          ;IF DNA DID NOT SET
(1)                                                          ;CHECK WORD LENGTH SELECT LOGIC
(1)
8166                                         ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
(1)                                          ;;OF THE TRANSMITTER SECTION.
(1)                                          ;;IT ALSO CHECKS DNA TIMING
(1)                                          ;;MODE:ISYMOD
(1)                                          ;;LENGTH:FIVE PLUS PARITY
(1)                                          ;;PARITY:EVEPAR
(1)                                          ;;CHARACTER:25
(1)                                          ;;
(5)                                  ;;**********************************************************************
(4)  010270  000004                  TST20:  SCOPE
(4)
(3)  010272  052777  000400  171426          BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  010300  012777  000000  171414          MOV     #ISYMOD,@PARCSR ;SET THE MODE
(3)  010306  052777  000400  171412          BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                  ;SET MAINTENANCE MODE & SEND
(2)                                      ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  010314  012777  004020  171404          MOV     #MINT!SEND,@TXCSR
(2)
(2)                                  ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  010322  012777  001426  171372          MOV     #ISYMOD!FIVE!EVEPAR!26,@PARCSR
(1)  010330  016703  171372                  MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  010334  112777  000025  171370          MOVB    #25,@TXDBUF     ;LOAD DATA CHAR
(1)  010342  012767  000352  171130          MOV     #352,$TMP1      ;TO BE SHIFTED CHAR
(1)  010350  012767  000010  170544          MOV     #8.,SHIFT       ;# OF SHIFTS
(1)                                  ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  010356  052777  020000  171342          BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  010364  042777  020000  171334          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  010372  005000                  1$:     CLR     R0
(1)  010374  006067  171100                  ROR     $TMP1   ;FORCE CARRY
(1)  010400  103002                          BCC     2$      ;BR IF CARRY CLR
(1)  010402  052700  002000                  BIS     #BITW,R0        ;EQUIV OF BITW
(1)  010406                          2$:
(2)  010406  052777  020000  171312          BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  010414  042777  020000  171304          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  010422  017701  171300                  MOV     @TXCSR,R1       ;ACTUAL
(1)  010426  042701  075777                  BIC     #075777,R1      ;SAVE BITW & DNA
(1)  010432  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  010434  001401                          BEQ     .+4
(1)  010436  104003                          ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
```

```
(1)                                                         ;BIT,...ALSO CHECK DNA
(1)  010440  005367  170456                 DEC     SHIFT  ;# OF SHIFTS
(1)  010444  001352                         BNE     1$     ;DO IT AGAIN ?
(1)                                 ;NOW POKE CLK TO SEE DNA
(1)  010446  052777  020000  171252         BIS     #CLK,@TXCSR     ;POKE CLK
(1)  010454  012700  000000                 MOV     #0,R0   ;EXPECTED
(1)  010460  017701  171242                 MOV     @TXCSR,R1       ;ACTUAL
(1)  010464  042701  077777                 BIC     #77777,R1       ;SAVE DNA ONLY
(1)  010470  020001                         CMP     R0,R1           ;COMPARE EXP VS ACT
(1)  010472  001401                         BEQ     .+4
(1)  010474  104003                         ERROR   3               ;DNA SHOULD BE SET
(1)                                                         ;IF DNA DID NOT SET
(1)                                                         ;CHECK WORD LENGTH SELECT LOGIC
(1)
8167                                        ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
(1)                                         ;;OF THE TRANSMITTER SECTION.
(1)                                         ;;IT ALSO CHECKS DNA TIMING
(1)                                         ;;MODE:ISYMOD
(1)                                         ;;LENGTH:FIVE PLUS PARITY
(1)                                         ;;PARITY:ODDPAR
(1)                                         ;;CHARACTER:25
(1)                                         ;;
(5)                                 ;;**************************************************************
(4)  010476  000004                 TST21:  SCOPE
(4)
(3)  010500  052777  000400  171220         BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  010506  012777  000000  171206         MOV     #ISYMOD,@PARCSR ;SET THE MODE
(3)  010514  052777  000400  171204         BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                 ;SET MAINTENANCE MODE & SEND
(2)                                     ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  010522  012777  004020  171176         MOV     #MINT!SEND,@TXCSR
(2)
(2)                                 ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  010530  012777  001026  171164         MOV     #ISYMOD!FIVE!ODDPAR!26,@PARCSR
(1)  010536  016703  171164                 MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  010542  112777  000025  171162         MOVB    #25,@TXDBUF     ;LOAD DATA CHAR
(1)  010550  012767  000252  170722         MOV     #252,$TMP1      ;TO BE SHIFTED CHAR
(1)  010556  012767  000010  170336         MOV     #8.,SHIFT       ;# OF SHIFTS
(1)                                 ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  010564  052777  020000  171134         BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  010572  042777  020000  171126         BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  010600  005000                 1$:     CLR     R0
(1)  010602  006067  170672                 ROR     $TMP1   ;FORCE CARRY
(1)  010606  103002                         BCC     2$      ;BR IF CARRY CLR
(1)  010610  052700  002000                 BIS     #BITW,R0        ;EQUIV OF BITW
(1)  010614                         2$:
(2)  010614  052777  020000  171104         BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  010622  042777  020000  171076         BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  010630  017701  171072                 MOV     @TXCSR,R1       ;ACTUAL
(1)  010634  042701  075777                 BIC     #075777,R1      ;SAVE BITW & DNA
(1)  010640  020001                         CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  010642  001401                         BEQ     .+4
(1)  010644  104003                         ERROR   3               ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                         ;BIT,...ALSO CHECK DNA
(1)  010646  005367  170250                 DEC     SHIFT   ;# OF SHIFTS
```

```
(1)  010652  001352                          BNE     1$      ;DO IT AGAIN ?
(1)                                  ;NOW POKE CLK TO SEE DNA
(1)  010654  052777  020000  171044          BIS     #CLK,@TXCSR     ;POKE CLK
(1)  010662  012700  000000                  MOV     #0,R0   ;EXPECTED
(1)  010666  017701  171034                  MOV     @TXCSR,R1       ;ACTUAL
(1)  010672  042701  077777                  BIC     #77777,R1       ;SAVE DNA ONLY
(1)  010676  020001                          CMP     R0,R1           ;COMPARE EXP VS ACT
(1)  010700  001401                          BEQ     .+4
(1)  010702  104003                          ERROR   3       ;DNA SHOULD BE SET
(1)                                                          ;IF DNA DID NOT SET
(1)                                                          ;CHECK WORD LENGTH SELECT LOGIC
(1)
8168
8169                                         ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
(1)                                          ;;OF THE TRANSMITTER SECTION.
(1)                                          ;;IT ALSO CHECKS DNA TIMING
(1)                                          ;;MODE:SYNINT
(1)                                          ;;LENGTH:SIX PLUS PARITY
(1)                                          ;;PARITY:EVEPAR
(1)                                          ;;CHARACTER:25
(1)                                          ;;
(5)                                  ;;**********************************************************************
(4)  010704  000004                  TST22:  SCOPE
(4)
(3)  010706  052777  000400  171012          BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  010714  012777  030000  171000          MOV     #SYNINT,@PARCSR ;SET THE MODE
(3)  010722  052777  000400  170776          BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                  ;SET MAINTENANCE MODE & SEND
(2)                                          ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  010730  012777  004020  170770          MOV     #MINT!SEND,@TXCSR
(2)
(2)                                  ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  010736  012777  033426  170756          MOV     #SYNINT!SIX!EVEPAR!26,@PARCSR
(1)  010744  016703  170756                  MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  010750  112777  000025  170754          MOVB    #25,@TXDBUF     ;LOAD DATA CHAR
(1)  010756  012767  000125  170514          MOV     #125,$TMP1      ;TO BE SHIFTED CHAR
(1)  010764  012767  000007  170130          MOV     #7,SHIFT        ;# OF SHIFTS
(1)                                  ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  010772  052777  020000  170726          BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  011000  042777  020000  170720          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  011006  005000                  1$:     CLR     R0
(1)  011010  006067  170464                  ROR     $TMP1   ;FORCE CARRY
(1)  011014  103002                          BCC     2$      ;BR IF CARRY CLR
(1)  011016  052700  002000                  BIS     #BITW,R0        ;EQUIV OF BITW
(1)  011022                          2$:
(2)  011022  052777  020000  170676          BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  011030  042777  020000  170670          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  011036  017701  170664                  MOV     @TXCSR,R1       ;ACTUAL
(1)  011042  042701  075777                  BIC     #075777,R1      ;SAVE BITW & DNA
(1)  011046  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  011050  001401                          BEQ     .+4
(1)  011052  104003                          ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                          ;BIT,...ALSO CHECK DNA
(1)  011054  005367  170042                  DEC     SHIFT   ;# OF SHIFTS
(1)  011060  001352                          BNE     1$      ;DO IT AGAIN ?
```

```
                                ;NOW POKE CLK TO SEE DNA
011062  052777  020000  170636          BIS     #CLK,@TXCSR     ;POKE CLK
011070  012700  100000                  MOV     #100000,R0      ;EXPECTED
011074  017701  170626                  MOV     @TXCSR,R1       ;ACTUAL
011100  042701  077777                  BIC     #77777,R1       ;SAVE DNA ONLY
011104  020001                          CMP     R0,R1           ;COMPARE EXP VS ACT
011106  001401                          BEQ     .+4
011110  104003                          ERROR   3               ;DNA SHOULD BE SET
                                                                ;IF DNA DID NOT SET
                                                                ;CHECK WORD LENGTH SELECT LOGIC

                                        ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
                                        ;;OF THE TRANSMITTER SECTION.
                                        ;;IT ALSO CHECKS DNA TIMING
                                        ;;MODE:SYNINT
                                        ;;LENGTH:SIX PLUS PARITY
                                        ;;PARITY:ODDPAR
                                        ;;CHARACTER:25
                                        ;;
                                ;;**************************************************************
011112  000004                  TST23:  SCOPE

011114  052777  000400  170604          BIS     #MRESET,@TXCSR  ;MASTER RESET
011122  012777  030000  170572          MOV     #SYNINT,@PARCSR ;SET THE MODE
011130  052777  000400  170570          BIS     #MRESET,@TXCSR  ;MASTER RESET

                                ;SET MAINTENANCE MODE & SEND
                                        ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
011136  012777  004020  170562          MOV     #MINT!SEND,@TXCSR

                                ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
011144  012777  033026  170550          MOV     #SYNINT!SIX!ODDPAR!26,@PARCSR
011152  016703  170550                  MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
011156  112777  000025  170546          MOVB    #25,@TXDBUF     ;LOAD DATA CHAR
011164  012767  000025  170306          MOV     #25,$TMP1       ;TO BE SHIFTED CHAR
011172  012767  000007  167722          MOV     #7,SHIFT        ;# OF SHIFTS
                                ;POKE CLK TO GET INTO SYNCRONIZATION
011200  052777  020000  170520          BIS     #CLK,@TXCSR     ;POKE CLK UP
011206  042777  020000  170512          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
011214  005000                  1$:     CLR     R0
011216  006067  170256                  ROR     $TMP1   ;FORCE CARRY
011222  103002                          BCC     2$      ;BR IF CARRY CLR
011224  052700  002000                  BIS     #BITW,R0        ;EQUIV OF BITW
011230                          2$:
011230  052777  020000  170470          BIS     #CLK,@TXCSR     ;POKE CLK UP
011236  042777  020000  170462          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
011244  017701  170456                  MOV     @TXCSR,R1       ;ACTUAL
011250  042701  075777                  BIC     #075777,R1      ;SAVE BITW & DNA
011254  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
011256  001401                          BEQ     .+4
011260  104003                          ERROR   3               ;BIT WINDOW DID NOT MATCH ACTUAL DATA
                                                                ;BIT,...ALSO CHECK DNA
011262  005367  167634                  DEC     SHIFT   ;# OF SHIFTS
011266  001352                          BNE     1$      ;DO IT AGAIN ?
                                ;NOW POKE CLK TO SEE DNA
011270  052777  020000  170430          BIS     #CLK,@TXCSR     ;POKE CLK
```

```
(1)  011276  012700  100000                MOV     #100000,R0      ;EXPECTED
(1)  011302  017701  170420                MOV     @TXCSR,R1       ;ACTUAL
(1)  011306  042701  077777                BIC     #77777,R1       ;SAVE DNA ONLY
(1)  011312  020001                        CMP     R0,R1           ;COMPARE EXP VS ACT
(1)  011314  001401                        BEQ     .+4
(1)  011316  104003                        ERROR   3               ;DNA SHOULD BE SET
(1)                                                                ;IF DNA DID NOT SET
(1)                                                                ;CHECK WORD LENGTH SELECT LOGIC
(1)
8171                                       ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
(1)                                        ;;OF THE TRANSMITTER SECTION.
(1)                                        ;;IT ALSO CHECKS DNA TIMING
(1)                                        ;;MODE:ISYMOD
(1)                                        ;;LENGTH:SIX PLUS PARITY
(1)                                        ;;PARITY:EVEPAR
(1)                                        ;;CHARACTER:25
(1)                                        ;;
(5)                                    ;;*****************************************************************
(4)  011320  000004                    TST24:  SCOPE
(4)
(3)  011322  052777  000400  170376            BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  011330  012777  000000  170364            MOV     #ISYMOD,@PARCSR ;SET THE MODE
(3)  011336  052777  000400  170362            BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                    ;SET MAINTENANCE MODE & SEND
(2)                                            ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  011344  012777  004020  170354            MOV     #MINT!SEND,@TXCSR
(2)
(2)                                    ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  011352  012777  003426  170342            MOV     #ISYMOD!SIX!EVEPAR!26,@PARCSR
(1)  011360  016703  170342                    MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  011364  112777  000025  170340            MOVB    #25,@TXDBUF     ;LOAD DATA CHAR
(1)  011372  012767  000652  170100            MOV     #652,$TMP1      ;TO BE SHIFTED CHAR
(1)  011400  012767  000011  167514            MOV     #9.,SHIFT       ;# OF SHIFTS
(1)                                    ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  011406  052777  020000  170312            BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  011414  042777  020000  170304            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  011422  005000                    1$:     CLR     R0
(1)  011424  006067  170050                    ROR     $TMP1   ;FORCE CARRY
(1)  011430  103002                            BCC     2$      ;BR IF CARRY CLR
(1)  011432  052700  002000                    BIS     #BITW,R0        ;EQUIV OF BITW
(1)  011436                            2$:
(2)  011436  052777  020000  170262            BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  011444  042777  020000  170254            BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  011452  017701  170250                    MOV     @TXCSR,R1       ;ACTUAL
(1)  011456  042701  075777                    BIC     #075777,R1      ;SAVE BITW & DNA
(1)  011462  020001                            CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  011464  001401                            BEQ     .+4
(1)  011466  104003                            ERROR   3               ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                                    ;BIT,...ALSO CHECK DNA
(1)  011470  005367  167426                    DEC     SHIFT   ;# OF SHIFTS
(1)  011474  001352                            BNE     1$      ;DO IT AGAIN ?
(1)                                    ;NOW POKE CLK TO SEE DNA
(1)  011476  052777  020000  170222            BIS     #CLK,@TXCSR     ;POKE CLK
(1)  011504  012700  000000                    MOV     #0,R0   ;EXPECTED
(1)  011510  017701  170212                    MOV     @TXCSR,R1       ;ACTUAL
```

```
(1)  011514  042701  077777                BIC     #77777,R1       ;SAVE DNA ONLY
(1)  011520  020001                        CMP     R0,R1           ;COMPARE EXP VS ACT
(1)  011522  001401                        BEQ     .+4
(1)  011524  104003                        ERROR   3               ;DNA SHOULD BE SET
(1)                                                                ;IF DNA DID NOT SET
(1)                                                                ;CHECK WORD LENGTH SELECT LOGIC
(1)
8172                                       ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
(1)                                        ;;OF THE TRANSMITTER SECTION.
(1)                                        ;;IT ALSO CHECKS DNA TIMING
(1)                                        ;;MODE:ISYMOD
(1)                                        ;;LENGTH:SIX PLUS PARITY
(1)                                        ;;PARITY:ODDPAR
(1)                                        ;;CHARACTER:25
(1)                                        ;;
(5)                                    ;;**************************************************************
(4)  011526  000004                    TST25:  SCOPE
(4)
(3)  011530  052777  000400  170170           BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  011536  012777  000000  170156           MOV     #ISYMOD,@PARCSR ;SET THE MODE
(3)  011544  052777  000400  170154           BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                    ;SET MAINTENANCE MODE & SEND
(2)                                            ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  011552  012777  004020  170146           MOV     #MINT!SEND,@TXCSR
(2)
(2)                                    ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  011560  012777  003026  170134           MOV     #ISYMOD!SIX!ODDPAR!26,@PARCSR
(1)  011566  016703  170134                   MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  011572  112777  000025  170132           MOVB    #25,@TXDBUF     ;LOAD DATA CHAR
(1)  011600  012767  000452  167672           MOV     #452,$TMP1      ;TO BE SHIFTED CHAR
(1)  011606  012767  000011  167306           MOV     #9.,SHIFT       ;# OF SHIFTS
(1)                                    ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  011614  052777  020000  170104           BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  011622  042777  020000  170076           BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  011630  005000                    1$:     CLR     R0
(1)  011632  006067  167642                   ROR     $TMP1   ;FORCE CARRY
(1)  011636  103002                           BCC     2$      ;BR IF CARRY CLR
(1)  011640  052700  002000                   BIS     #BITW,R0        ;EQUIV OF BITW
(1)  011644                            2$:
(2)  011644  052777  020000  170054           BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  011652  042777  020000  170046           BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  011660  017701  170042                   MOV     @TXCSR,R1       ;ACTUAL
(1)  011664  042701  075777                   BIC     #075777,R1      ;SAVE BITW & DNA
(1)  011670  020001                           CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  011672  001401                           BEQ     .+4
(1)  011674  104003                           ERROR   3               ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                                ;BIT,...ALSO CHECK DNA
(1)  011676  005367  167220                   DEC     SHIFT   ;# OF SHIFTS
(1)  011702  001352                           BNE     1$      ;DO IT AGAIN ?
(1)                                    ;NOW POKE CLK TO SEE DNA
(1)  011704  052777  020000  170014           BIS     #CLK,@TXCSR     ;POKE CLK
(1)  011712  012700  000000                   MOV     #0,R0   ;EXPECTED
(1)  011716  017701  170004                   MOV     @TXCSR,R1       ;ACTUAL
(1)  011722  042701  077777                   BIC     #77777,R1       ;SAVE DNA ONLY
(1)  011726  020001                           CMP     R0,R1           ;COMPARE EXP VS ACT
```

```
(1)  011730  001401                        BEQ     .+4
(1)  011732  104003                        ERROR   3       ;DNA SHOULD BE SET
(1)                                                        ;IF DNA DID NOT SET
(1)                                                        ;CHECK WORD LENGTH SELECT LOGIC
(1)
8173
8174
8175                                       ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
(1)                                        ;;OF THE TRANSMITTER SECTION.
(1)                                        ;;IT ALSO CHECKS DNA TIMING
(1)                                        ;;MODE:SYNINT
(1)                                        ;;LENGTH:SEVEN PLUS PARITY
(1)                                        ;;PARITY:EVEPAR
(1)                                        ;;CHARACTER:125
(1)                                        ;;
(5)                                    ;;**************************************************************
(4)  011734  000004                    TST26:  SCOPE
(4)
(3)  011736  052777  000400  167762           BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)  011744  012777  030000  167750           MOV     #SYNINT,@PARCSR ;SET THE MODE
(3)  011752  052777  000400  167746           BIS     #MRESET,@TXCSR  ;MASTER RESET
(2)
(2)                                    ;SET MAINTENANCE MODE & SEND
(2)                                            ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
(2)  011760  012777  004020  167740           MOV     #MINT!SEND,@TXCSR
(2)
(2)                                    ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
(2)  011766  012777  035426  167726           MOV     #SYNINT!SEVEN!EVEPAR!26,@PARCSR
(1)  011774  016703  167726                   MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
(1)  012000  112777  000125  167724           MOVB    #125,@TXDBUF    ;LOAD DATA CHAR
(1)  012006  012767  000125  167464           MOV     #125,$TMP1      ;TO BE SHIFTED CHAR
(1)  012014  012767  000010  167100           MOV     #8.,SHIFT       ;# OF SHIFTS
(1)                                    ;POKE CLK TO GET INTO SYNCRONIZATION
(2)  012022  052777  020000  167676           BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  012030  042777  020000  167670           BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  012036  005000                    1$:     CLR     R0
(1)  012040  006067  167434                   ROR     $TMP1   ;FORCE CARRY
(1)  012044  103002                           BCC     2$      ;BR IF CARRY CLR
(1)  012046  052700  002000                   BIS     #BITW,R0        ;EQUIV OF BITW
(1)  012052                            2$:
(2)  012052  052777  020000  167646           BIS     #CLK,@TXCSR     ;POKE CLK UP
(2)  012060  042777  020000  167640           BIC     #CLK,@TXCSR     ;POKE CLK DOWN
(1)  012066  017701  167634                   MOV     @TXCSR,R1       ;ACTUAL
(1)  012072  042701  075777                   BIC     #075777,R1      ;SAVE BITW & DNA
(1)  012076  020001                           CMP     R0,R1   ;COMPARE EXP VS ACT
(1)  012100  001401                           BEQ     .+4
(1)  012102  104003                           ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
(1)                                                        ;BIT,...ALSO CHECK DNA
(1)  012104  005367  167012                   DEC     SHIFT   ;# OF SHIFTS
(1)  012110  001352                           BNE     1$      ;DO IT AGAIN ?
(1)                                    ;NOW POKE CLK TO SEE DNA
(1)  012112  052777  020000  167606           BIS     #CLK,@TXCSR     ;POKE CLK
(1)  012120  012700  100000                   MOV     #100000,R0      ;EXPECTED
(1)  012124  017701  167576                   MOV     @TXCSR,R1       ;ACTUAL
(1)  012130  042701  077777                   BIC     #77777,R1       ;SAVE DNA ONLY
(1)  012134  020001                           CMP     R0,R1           ;COMPARE EXP VS ACT
```

```
   (1)  012136  001401                          BEQ     .+4
   (1)  012140  104003                          ERROR   3       ;DNA SHOULD BE SET
   (1)                                                          ;IF DNA DID NOT SET
   (1)                                                          ;CHECK WORD LENGTH SELECT LOGIC
   (1)
  8176                                          ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
   (1)                                          ;;OF THE TRANSMITTER SECTION.
   (1)                                          ;;IT ALSO CHECKS DNA TIMING
   (1)                                          ;;MODE:SYNINT
   (1)                                          ;;LENGTH:SEVEN PLUS PARITY
   (1)                                          ;;PARITY:ODDPAR
   (1)                                          ;;CHARACTER:125
   (1)                                          ;;
   (5)                                  ;;**************************************************************
   (4)  012142  000004                  TST27:  SCOPE
   (4)
   (3)  012144  052777  000400  167554          BIS     #MRESET,@TXCSR  ;MASTER RESET
   (2)  012152  012777  030000  167542          MOV     #SYNINT,@PARCSR ;SET THE MODE
   (3)  012160  052777  000400  167540          BIS     #MRESET,@TXCSR  ;MASTER RESET
   (2)
   (2)                                  ;SET MAINTENANCE MODE & SEND
   (2)                                          ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
   (2)  012166  012777  004020  167532          MOV     #MINT!SEND,@TXCSR
   (2)
   (2)                                  ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
   (2)  012174  012777  035026  167520          MOV     #SYNINT!SEVEN!ODDPAR!26,@PARCSR
   (1)  012202  016703  167520                  MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
   (1)  012206  112777  000125  167516          MOVB    #125,@TXDBUF    ;LOAD DATA CHAR
   (1)  012214  012767  000325  167256          MOV     #325,$TMP1      ;TO BE SHIFTED CHAR
   (1)  012222  012767  000010  166672          MOV     #8.,SHIFT       ;# OF SHIFTS
   (1)                                  ;POKE CLK TO GET INTO SYNCRONIZATION
   (2)  012230  052777  020000  167470          BIS     #CLK,@TXCSR     ;POKE CLK UP
   (2)  012236  042777  020000  167462          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
   (1)  012244  005000                  1$:     CLR     R0
   (1)  012246  006067  167226                  ROR     $TMP1   ;FORCE CARRY
   (1)  012252  103002                          BCC     2$      ;BR IF CARRY CLR
   (1)  012254  052700  002000                  BIS     #BITW,R0        ;EQUIV OF BITW
   (1)  012260                          2$:
   (2)  012260  052777  020000  167440          BIS     #CLK,@TXCSR     ;POKE CLK UP
   (2)  012266  042777  020000  167432          BIC     #CLK,@TXCSR     ;POKE CLK DOWN
   (1)  012274  017701  167426                  MOV     @TXCSR,R1       ;ACTUAL
   (1)  012300  042701  075777                  BIC     #075777,R1      ;SAVE BITW & DNA
   (1)  012304  020001                          CMP     R0,R1   ;COMPARE EXP VS ACT
   (1)  012306  001401                          BEQ     .+4
   (1)  012310  104003                          ERROR   3       ;BIT WINDOW DID NOT MATCH ACTUAL DATA
   (1)                                                          ;BIT,...ALSO CHECK DNA
   (1)  012312  005367  166604                  DEC     SHIFT   ;# OF SHIFTS
   (1)  012316  001352                          BNE     1$      ;DO IT AGAIN ?
   (1)                                  ;NOW POKE CLK TO SEE DNA
   (1)  012320  052777  020000  167400          BIS     #CLK,@TXCSR     ;POKE CLK
   (1)  012326  012700  100000                  MOV     #100000,R0      ;EXPECTED
   (1)  012332  017701  167370                  MOV     @TXCSR,R1       ;ACTUAL
   (1)  012336  042701  077777                  BIC     #77777,R1       ;SAVE DNA ONLY
   (1)  012342  020001                          CMP     R0,R1           ;COMPARE EXP VS ACT
   (1)  012344  001401                          BEQ     .+4
   (1)  012346  104003                          ERROR   3       ;DNA SHOULD BE SET
```

```
  (1)                                                   ;IF DNA DID NOT SET
  (1)                                                   ;CHECK WORD LENGTH SELECT LOGIC
  (1)
 8177
 8178
  (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)  012350  000004                .EOP: SCOPE
  (2)  012352  004767  000340           JSR     PC,CKSWR
  (2)  012356  104401                   TYPE                                  ;TYPE NAME OF TEST
  (2)  012360  015506                   MEPASS
  (2)  012362  104413  012614           CONVRT  ,OUTCRY
  (2)  012366  104401  015325           TYPE    ,DEVICE
  (2)  012372  105767  166554           TSTB    MULTD   ;ARE YOU RUNNING MULTIPLE DEVICES ?
  (2)  012376  001511                   BEQ     CCC     ;NO,JUMP AROUND
  (2)  012400  005767  166562           TST     ACTREG  ;ARE ANY DEVICES ACTIVE ?
  (2)  012404  001007                   BNE     RUNIT   ;YES
  (2)  012406  104401  015337           TYPE    ,MCOW   ;NO
  (2)  012412  016700  166550           MOV     ACTREG,R0       ;DISPLAY ACTREG
  (2)  012416  000000                   HALT            ;SELECT SOMETHING TO RUN @ ACTREG:
  (2)                                    ;SELECT SWITCHES & HIT CONTINUE (PUT SW00 =1)
  (2)  012420  000167  167526           JMP     .START  ;START OVER AGAIN.....YOU DESELECTED EVERYTHING
  (2)  012424  062767  000010  166522 RUNIT: ADD     #10,BASEADD     ;NEXT BLOCK (ADDRESSES)
  (2)  012432  062767  000010  166522 ZERO:  ADD     #10,BASEIV      ;NEXT BLOCK (VECTORS)
  (2)  012440  000241                   CLC
  (2)  012442  006167  166522           ROL     ROTADD  ;UP DATE ROTATING POINTER
  (2)  012446  103410                   BCS     2$      ;IS IT THE LAST DEVICE
  (2)                                    ;TO BE TESTED IN THIS PASS ?
  (2)  012450  036767  166514  166510   BIT     ROTADD,ACTREG   ;TEST THIS DEVICE FOR ACTIVE STATUS
  (2)  012456  001762                   BEQ     RUNIT   ;IF NOT ACTIVE, TRY NEXT ADDRESS
  (2)  012460  004767  000034           JSR     PC,REPLAY       ;CALCULATE NEW PARAMETERS
  (2)  012464  000167  000210           JMP     RESTRT  ;YES IT WAS ACTIVE,TEST THIS DEVICE
  (2)  012470  012767  000001  166472 2$:    MOV     #1,ROTADD       ;OK!,NOW SET UP  ROTATING
  (2)                                    ;POINTER FOR NEXT MULTIPLE PASS
  (2)  012476  016767  166454  166450   MOV     KEEPADD,BASEADD ;RESTORE BASE ADDRESS
  (2)  012504  016767  166454  166450   MOV     KEEPIV,BASEIV   ;RESTORE BASE INTERRUPT VECTORS
  (2)  012512  004767  000002           JSR     PC,REPLAY       ;CALC NEW PARAMETERS
  (2)  012516  000441                   BR      CCC     ;JUMP AROUND REPLAY
  (2)  012520  016767  166430  004416 REPLAY: MOV     BASEADD,DUBASE  ;SET UP FOR NEW ADDRESSES
  (2)  012526  004767  004260           JSR     PC,DUADDR       ;CREATE NEW ADDRESSES
  (2)  012532  016767  166424  167176   MOV     BASEIV,DURIV    ;CREATE DURIV
  (2)  012540  062767  000002  166414   ADD     #2,BASEIV
  (2)  012546  016767  166410  167164   MOV     BASEIV,DURIS    ;CREATE DURIS
  (2)  012554  062767  000002  166400   ADD     #2,BASEIV
  (2)  012562  016767  166374  167152   MOV     BASEIV,DUTIV    ;CREATE DUTIV
  (2)  012570  062767  000002  166364   ADD     #2,BASEIV
  (2)  012576  016767  166360  167140   MOV     BASEIV,DUTIS    ;CREATE DUTIS
  (2)  012604  016767  167126  166350   MOV     DURIV,BASEIV    ;RESTORE
  (2)  012612  000207                   RTS     PC
  (2)
  (2)  012614  000001                OUTCRY: 1
  (2)  012616     006     002                .BYTE   6,2
```

```
(2)  012620  001712                        RXCSR
(2)
(2)  012622                          CCC:
(2)  012622  005067  166554                 CLR     $TSTNM                  ;CLEAR TEST NUMBER
(2)  012626  005067  166564                 CLR     $ERRPC                  ;CLEAR LAST ERROR PC
(2)  012632  005067  166545                 CLR     $ERFLG                  ;CLEAR ERROR FLAG
(2)  012636  005267  166250                 INC     PASCNT                  ;UPDATE PASS COUNT
(2)  012642  016767  166244  166232        MOV     PASCNT,LIGHTS           ;DISPLAY PASS COUNT
(2)  012650  016767  166236  166656        MOV     PASCNT,$PASS            ;PASS COUNT TO APT
(2)  012656  013701  000042                 MOV     @#42,R1                 ;CHECK FOR ACT-11 OR DDP
(2)  012662  001406                         BEQ     RESTRT          ;IF NO CONTINUE TESTING
(2)  012664  000005                         RESET
(2)  012666  000005                         RESET
(2)  012670  004711                  $ENDAD: JSR     PC,(R1)
(2)  012672  000240                         NOP
(2)  012674  000240                         NOP
(2)  012676  000240                         NOP
(2)  012700                          RESTRT:
(2)  012700  012767  003376  166500        MOV     #TST1+2,$LPADR  ;LOAD LAST ADDR
(2)  012706  004767  000004                 JSR     PC,CKSWR
(2)  012712  000167  170372                 JMP     .BEGIN
(2)
(2)                                         ;CHECK SWITCH REGISTER ROUTINE.
(2)                                         ;CHECKS TO ALLOW FOR <^G> TO ALLOW
(2)                                         ;THE CHANGING OF LOCATION 176
(2)
(2)  012716  005737  000042          CKSWR: TST     @#42
(2)  012722  001040                         BNE     OUT
(2)  012724  022767  000176  166506        CMP     #SWREG,SWR      ;SOFTWARE SWR PRESENT?
(2)  012732  001034                         BNE     OUT             ;NO--LEAVE
(2)  012734  105777  166504                 TSTB    @$TKS           ;CHECK TTY READY
(2)  012740  100031                         BPL     OUT             ;NO--LEAVE
(2)  012742  017767  166500  000422        MOV     @$TKB,..MSG     ;GET CHARACTER
(2)  012750  042767  177600  000414        BIC     #177600,..MSG   ;STRIP JUNK
(2)  012756  122767  000007  000406        CMPB    #7,..MSG        ;IS IT <^G> ?
(2)  012764  001017                         BNE     OUT             ;NO
(2)  012766  104401  016113                 TYPE    ,MCNTG
(2)  012772  005137  013032          CNTLU: COM     @#RDSW
(2)  012776  104401  016123                 TYPE    ,MMSWR
(2)  013002  104413                         CONVRT
(2)  013004  013034                         SWREGL
(2)  013006  104406  016134                 INSTR,MMNEW
(2)  013012  104410                         PARAM
(2)  013014  000000                         0
(2)  013016  177777                         177777
(2)  013020  000176                         SWREG
(2)  013022     000     001          .BYTE  0,1
(2)  013024  005037  013032          OUT:   CLR     @#RDSW
(2)  013030  000207                         RTS     PC
(2)  013032  000000                  RDSW:  .WORD   0
(2)  013034  000001                  SWREGL: 1
(2)  013036     006     002                 .BYTE   6,2
(2)  013040  000176                         SWREG
(2)
(1)  013042  000005                         5
(2)
```

```
(2)                                             ;CHECK FOR FREEZE ON CURRENT DATA
(2)
(2)  013044  004767  177646            .SCOP1: JSR     PC,CKSWR
(2)  013050  032777  001000  166362             BIT     #SW09,@SWR
(2)  013056  001402                             BEQ     1$
(2)  013060  016716  166024                     MOV     LOCK,(SP)
(2)  013064  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)  013066  105767  166365            $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
(2)  013072  100002                             BPL     1$              ;;BR IF YES
(2)  013074  000000                             HALT                    ;;HALT HERE IF NO TERMINAL
(2)  013076  000430                             BR      3$              ;;LEAVE
(2)  013100  010046                     1$:     MOV     R0,-(SP)        ;;SAVE R0
(2)  013102  017600  000002                     MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
(2)  013106  122767  000001  166432             CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(2)  013114  001011                             BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
(2)  013116  132767  000100  166423             BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(2)  013124  001405                             BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
(2)  013126  010067  000004                     MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
(2)  013132  004767  164650                     JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
(2)  013136  000000                     61$:    .WORD   0               ;;MESSAGE ADDRESS
(2)  013140  132767  000040  166401     62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
(2)  013146  001003                             BNE     60$             ;;YES,SKIP TYPE OUT
(2)  013150  112046                     2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(2)  013152  001005                             BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(2)  013154  005726                             TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(2)  013156  012600                     60$:    MOV     (SP)+,R0        ;;RESTORE R0
(2)  013160  062716  000002             3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
(2)  013164  000002                             RTI                     ;;RETURN
(2)  013166  122716  000011             4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(2)  013172  001430                             BEQ     8$
(2)  013174  122716  000200                     CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
(2)  013200  001006                             BNE     5$
(2)  013202  005726                             TST     (SP)+           ;;POP  <CR><LF> EQUIV
(2)  013204  104401                             TYPE                    ;;TYPE A CR AND LF
(2)  013206  001523                             $CRLF
(2)  013210  105067  000130                     CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
(2)  013214  000755                             BR      2$              ;;GET NEXT CHARACTER
(2)  013216  004767  000056             5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(2)  013222  126726  166230             6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
```

```
(2)  013226  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
(2)  013230  016746  166220                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(2)                                                                  ;;AND THE NULL CHAR.
(2)  013234  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(2)  013240  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(2)  013242  004767  000032                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
(2)  013246  105367  000072                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
(2)  013252  000770                          BR      7$              ;;LOOP
(2)
(2)                                  ;HORIZONTAL TAB PROCESSOR
(2)
(2)  013254  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(2)  013260  004767  000014          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
(2)  013264  132767  000007  000052          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(2)  013272  001372                          BNE     9$              ;;TAB STOP
(2)  013274  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
(2)  013276  000724                          BR      2$              ;;GET NEXT CHARACTER
(2)  013300  105777  166144          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
(2)  013304  100375                          BPL     $TYPEC
(2)  013306  116677  000002  166136          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(2)
(2)  013314  122766  000015  000002  CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
(2)  013322  001003                          BNE     1$              ;;BRANCH IF NO
(2)  013324  105067  000014                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
(2)  013330  000406                          BR      $TYPEX          ;;EXIT
(2)  013332  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
(2)  013340  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
(2)  013342  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
(2)  013344  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
(2)  013346  000207          $TYPEX: RTS     PC
(2)
(2)
(2)
(2)
(2)                                  ;ASCII STRING INPUT ROUTINE
(2)
(2)  013350  017667  000000  000014  .INSTR: MOV     @(SP),.MSG      ;PICK UP MESSAGE
(2)  013356  062716  000002                  ADD     #2,(SP)         ;JUMP AROUND MESSAGE FOR RTI
(2)  013362  105767  166160                  TSTB    $ENV            ;APT CONTROL
(2)  013366  001036                          BNE     INSTR2          ;YES NO TYPE
(2)  013370  104401                  .INST1: TYPE
(2)  013372  000000                  .MSG:   0
(2)  013374  012704  016146                  MOV     #INBUF,R4       ;GET STARTING LOC OF INBUF
(2)  013400  012703  000007                  MOV     #7,R3           ;MAX # OF CHARS
(2)  013404  105777  166034          1$:     TSTB    @$TKS   ;TTY FLAG
(2)  013410  100375                          BPL     1$
(2)  013412  117714  166030                  MOVB    @$TKB,(R4)      ;TAKE CHAR
(2)  013416  142714  000200                  BICB    #200,(R4)       ;STRIP
(2)  013422  121427  000025                  CMPB    (R4),#25        ;IS IT <^G>
(2)  013426  001760                          BEQ     .INST1
(2)  013430  122427  000015                  CMPB    (R4)+,#15       ;CHECK FOR CR
(2)  013434  001413                          BEQ     INSTR2
(2)  013436  105777  166006          2$:     TSTB    @$TPS   ;TEST FLAG
(2)  013442  100375                          BPL     2$
(2)  013444  117777  165776  166000          MOVB    @$TKB,@$TPB     ;ECHO CHARACTER
(2)  013452  005303                          DEC     R3      ;DID YOU TYPE TOO MANY CHARS ?
```

```
(2)  013454  001353                        BNE     1$
(2)  013456  104401        .INSTE: TYPE
(2)  013460  015433                MQM     :?
(2)  013462  000742                BR      .INST1  ;RETRY
(2)  013464  000002        INSTR2: RTI
(2)
(2)                                ;CONVERT ASCII STRING TO OCTAL
(2)
(2)  013466  011605        .PARAM: MOV     (SP),R5 ;PUT CONTENTS OF SP INTO R5
(2)  013470  012567  000162       MOV     (R5)+,LOLIM     ;PUT LOW LIMIT INTO LOLIM
(2)  013474  012567  000160       MOV     (R5)+,HILIM     ;PUT HIGH LIMIT INTO HILIM
(2)  013500  012567  000156       MOV     (R5)+,DEVADR    ;PUT STORE LOC INTO DEVADR
(2)  013504  112567  000154       MOVB    (R5)+,LOBITS    ;PUT MASK INTO LOBITS
(2)  013510  112567  000151       MOVB    (R5)+,ADRCNT    ;PUT COUNT INTO ADRCNT
(2)  013514  010516                MOV     R5,(SP) ;RESTORE RETURN ADDR ON STACK FOR RTI
(2)  013516  005005        PARAM1: CLR     R5
(2)  013520  012704  016146       MOV     #INBUF,R4
(2)  013524  122714  000015       CMPB    #15,(R4)        ;CR ?
(2)  013530  001420                BEQ     PARERR  ;YOU TYPED CR TOO SOON !
(2)  013532  121427  000060 1$:    CMPB    (R4),#60        ;LOW LIMIT ASCII 0
(2)  013536  002415                BLT     PARERR
(2)  013540  121427  000067       CMPB    (R4),#67        ;HIGH LIMIT ASCII 7
(2)  013544  003012                BGT     PARERR
(2)  013546  142714  000060       BICB    #60,(R4)        ;CONVERT TO OCTAL
(2)  013552  152405                BISB    (R4)+,R5        ;STORE AWAY ITS AN OK CHAR
(2)  013554  122714  000015       CMPB    #15,(R4)        ;CR ?
(2)  013560  001414                BEQ     LIMITS  ;NOW CHECK FOR HIGH &LOW LIMIT CONDS
(2)  013562  006305                ASL     R5      ;ALLOCATE ROOM FOR NEXT CHAR
(2)  013564  006305                ASL     R5
(2)  013566  006305                ASL     R5
(2)  013570  000760                BR      1$
(2)  013572  122714  000015 PARERR: CMPB    #15,(R4)        ;CR?
(2)  013576  001003                BNE     120$
(2)  013600  005737  013032       TST     @#RDSW          ;CK SWR USED
(2)  013604  001023                BNE     PARTI
(2)  013606  104407        120$:   INSTER  ;RETRY
(2)  013610  000742                BR      PARAM1
(2)
(2)                                ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
(2)
(2)  013612  020567  000042 LIMITS: CMP     R5,HILIM
(2)  013616  101365                BHI     PARERR  ;THE # IS TOO HIGH
(2)  013620  020567  000032       CMP     R5,LOLIM
(2)  013624  103762                BLO     PARERR  ;THE # IS TOO LOW
(2)  013626  136705  000032       BITB    LOBITS,R5       ;TEST BY MASKINGTHE #
(2)  013632  001357                BNE     PARERR
(2)
(2)                                ;STORE NUMBER AT SPECIFIED ADDRESS
(2)
(2)  013634  016704  000022       MOV     DEVADR,R4       ;GET STARTING ADDR OF
(2)  013640  010524        1$:     MOV     R5,(R4)+        ;STORE AT THIS ADDR
(2)  013642  062705  000002       ADD     #2,R5
(2)  013646  105367  000013       DECB    ADRCNT  ;HOW MANY TIMES * 2 ?
(2)  013652  001372                BNE     1$
(2)  013654  000002        PARTI:  RTI
(2)  013656  000000        LOLIM:  0
```

```
(2)  013660  000000                    HILIM:  0
(2)  013662  000000                    DEVADR: 0
(2)  013664  000000                    LOBITS: 0
(2)          013665                    ADRCNT=LOBITS+1
(2)
(2)                                            ;SAVE PC OF TEST THAT FAILED AND R0-R5
(2)
(2)  013666  016667  000004  165232  .SAV05: MOV     4(SP),SAVPC
(2)
(2)                                            ;SAVE R0-R5
(2)
(2)  013674  010567  165574          SV05:   MOV     R5,$REG5
(2)  013700  010467  165566                  MOV     R4,$REG4
(2)  013704  010367  165560                  MOV     R3,$REG3
(2)  013710  010267  165552                  MOV     R2,$REG2
(2)  013714  010167  165544                  MOV     R1,$REG1
(2)  013720  010067  165536                  MOV     R0,$REG0
(2)  013724  000002                          RTI
(2)
(2)                                            ;RESTORE R0-R5
(2)
(2)  013726  016700  165530          .RES05: MOV     $REG0,R0
(2)  013732  016701  165526                  MOV     $REG1,R1
(2)  013736  016702  165524                  MOV     $REG2,R2
(2)  013742  016703  165522                  MOV     $REG3,R3
(2)  013746  016704  165520                  MOV     $REG4,R4
(2)  013752  016705  165516                  MOV     $REG5,R5
(2)  013756  000002                          RTI
(2)
(2)                                            ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
(2)
(2)  013760  104401                  .CONVR: TYPE
(2)  013762  015437                          MCRLF   ;CR  LF
(2)  013764  017601  000000                  MOV     @(SP),R1        ;PICK UP DATA POINTER
(2)  013770  062716  000002                  ADD     #2,(SP) ;SET UP SP FOR RTI
(2)  013774  012167  000130                  MOV     (R1)+,WRDCNT    ;PICK UP # OF WORDS FROM TABLE
(2)  014000  112167  000126          1$:     MOVB    (R1)+,CHRCNT    ;PICK UP # OF CHARS FROM TABLE
(2)  014004  112167  000123                  MOVB    (R1)+,SPACNT    ;PICK UP # OF SPACES FROM TABLE
(2)  014010  013167  000120                  MOV     @(R1)+,BINWRD   ;PICK UP ADDRESS OF  MSG
(2)                                                                  ;FROM TABLE
(2)  014014  016704  000114          2$:     MOV     BINWRD,R4       ;SAVE
(2)  014020  116705  000106                  MOVB    CHRCNT,R5       ;SAVE
(2)  014024  012700  016210                  MOV     #TEMP,R0        ;STARTING ADDRESS OF TEMP BLOCK
(2)  014030  010403                  3$:     MOV     R4,R3           ;SAVE
(2)  014032  042703  177770                  BIC     #177770,R3      ;CLR OUT UPPER BITS .. SAVE CHAR
(2)  014036  062703  000260                  ADD     #260,R3 ;CONVERT TO ASCII
(2)  014042  110320                          MOVB    R3,(R0)+        ;STORE AWAY
(2)  014044  006204                          ASR     R4      ;SHIFT FOR NEXT #
(2)  014046  006204                          ASR     R4      ;DITTO
(2)  014050  006204                          ASR     R4      ;DITTO
(2)  014052  005305                          DEC     R5      ;DEC CHAR COUNT
(2)  014054  001365                          BNE     3$      ;DO IT AGAIN ?
(2)  014056  012703  016252                  MOV     #MDATA,R3       ;STARTING ADDRESS OF MDATA BLOCK
(2)  014062  114023                  4$:     MOVB    -(R0),(R3)+     ;REVERSE THE ORDER OF NUMBERS
(2)  014064  105367  000042                  DECB    CHRCNT  ;DEC CHAR COUNT
(2)  014070  001374                          BNE     4$      ;DO IT AGAIN ?
```

```
(2)  014072  105767  000035              TSTB    SPACNT  ;HOW MANY SPACES ?
(2)  014076  001405                      BEQ     6$      ;TYPE # IF BR =0
(2)  014100  112723  000240      5$:     MOVB    #240,(R3)+      ;"SPACE" IN ASCII
(2)  014104  105367  000023              DECB    SPACNT  ;DEC # OF SPACE COUNT
(2)  014110  001373                      BNE     5$      ;DO IT AGAIN ?
(2)  014112  105013              6$:     CLRB    (R3)    ;INSERT "0" FOR TTY OUTPUT ROUTINE
(2)  014114  104401                      TYPE
(2)  014116  016252                      MDATA   ;THIS MESSAGE
(2)  014120  005367  000004              DEC     WRDCNT  ;HOW MANY #'S ?
(2)  014124  001325                      BNE     1$      ;DO THIS ROUTINE AGAIN IF NOT EQUAL TO 0
(2)  014126  000002                      RTI     ;RETURN TO PROGRAM
(2)  014130  000000              WRDCNT: 0
(2)  014132  000000              CHRCNT: 0
(2)          014133              SPACNT=CHRCNT+1
(2)  014134  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)  014136  017605  000000      .SETFLG:MOV     @(SP),R5
(2)  014142  122767  000116  001776      CMPB    #'N,INBUF       ;IS IT "N" ?
(2)  014150  001002                      BNE     1$
(2)  014152  105015                      CLRB    (R5)    ;000
(2)  014154  000406                      BR      2$
(2)  014156  122767  000131  001762  1$: CMPB    #'Y,INBUF       ;IS IT "Y" ?
(2)  014164  001005                      BNE     3$
(2)  014166  112715  177777              MOVB    #-1,(R5)        ;377
(2)  014172  062716  000002      2$:     ADD     #2,(SP)
(2)  014176  000002                      RTI
(2)  014200  104407              3$:     INSTER  ;RETRY
(2)  014202  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)  014204                      $ERROR:
(2)  014204  105267  165173      7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
(2)  014210  001775                      BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(2)  014212  016777  165164  165222      MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(2)  014220  032777  002000  165212      BIT     #BIT10,@SWR     ;;BELL ON ERROR?
(2)  014226  001402                      BEQ     1$              ;;NO - SKIP
(2)  014230  104401  001516              TYPE    ,$BELL          ;;RING BELL
(2)  014234  005267  165152      1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
(2)  014240  011667  165152              MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
```

```
(2)  014244  162767  000002  165144          SUB     #2,$ERRPC
(2)  014252  117767  165140  165134          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(2)  014260  032777  020000  165152          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
(2)  014266  001004                          BNE     20$             ;;SKIP TYPEOUTS
(2)  014270  004767  000072                  JSR     PC,SAVIT        ;;GO TO USER ERROR ROUTINE
(2)  014274  104401  001523                  TYPE    ,$CRLF
(2)  014300                          20$:
(2)  014300  122767  000001  165240          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
(2)  014306  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
(2)  014310  116767  165100  000004          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(2)  014316  004767  163474                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
(2)  014322     000                  21$:    .BYTE   0
(2)  014323     000                          .BYTE   0
(2)  014324  000777                  22$:    BR      22$             ;;APT ERROR LOOP
(2)  014326  005777  165106          2$:     TST     @SWR            ;;HALT ON ERROR
(2)  014332  100001                          BPL     3$              ;;SKIP IF CONTINUE
(2)  014334  000000                          HALT                    ;;HALT ON ERROR!
(2)  014336  032777  001000  165074  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
(2)  014344  001402                          BEQ     4$              ;;BR IF NO
(2)  014346  016716  165036                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(2)  014352  005767  165136          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(2)  014356  001402                          BEQ     5$              ;;BR IF NONE
(2)  014360  016716  165130                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(2)  014364                          5$:
(2)  014364  000002                          RTI                     ;;RETURN
(2)  014366  010067  164536          SAVIT:  MOV     R0,HLD0
(2)  014372  010167  164534                  MOV     R1,HLD1
(2)  014376  010267  164532                  MOV     R2,HLD2
(2)  014402  010367  164530                  MOV     R3,HLD3
(2)  014406  010467  164526                  MOV     R4,HLD4
(2)  014412  010567  164524                  MOV     R5,HLD5
(2)  014416  016767  164760  164520          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)  014424                          $ERRTYP:
(3)  014424  104401  001523                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(3)  014430  010046                          MOV     R0,-(SP)        ;;SAVE R0
(3)  014432  005000                          CLR     R0              ;;PICKUP THE ITEM INDEX
(3)  014434  153700  001414                  BISB    @#$ITEMB,R0
(3)  014440  001004                          BNE     1$              ;;IF ITEM NUMBER IS ZERO, JUST
(3)                                                                  ;;TYPE THE PC OF THE ERROR
(4)  014442  016746  164750                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
(4)                                                                  ;;ERROR ADDRESS
(4)  014446  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(3)  014450  000426                          BR      6$              ;;GET OUT
(3)  014452  005300                  1$:     DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
(3)  014454  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
(3)  014456  006300                          ASL     R0
(3)  014460  006300                          ASL     R0
(3)  014462  062700  001652                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
```

```
(3)  014466  012067  000004            MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
(3)  014472  001404                     BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
(3)  014474  104401                     TYPE                    ;;TYPE THE "ERROR MESSAGE"
(3)  014476  000000               2$:   .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
(3)  014500  104401  001523             TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(3)  014504  012067  000004       3$:   MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
(3)  014510  001404                     BEQ     5$              ;;SKIP TYPEOUT IF 0
(3)  014512  104401                     TYPE                    ;;TYPE THE "DATA HEADER"
(3)  014514  000000               4$:   .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
(3)  014516  104401  001523             TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(3)  014522  011000               5$:   MOV     (R0),R0         ;;PICKUP "DATA TABLE" POINTER
(3)  014524  001004                     BNE     7$              ;;GO TYPE THE DATA
(3)  014526  012600               6$:   MOV     (SP)+,R0        ;;RESTORE R0
(3)
(3)  014530  104401  001523             TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
(3)  014534  000207                     RTS     PC              ;;RETURN
(3)  014536                       7$:
(4)  014536  013046                     MOV     @(R0)+,-(SP)    ;;SAVE @(R0)+ FOR TYPEOUT
(4)  014540  104402                     TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
(3)  014542  005710                     TST     (R0)            ;;IS THERE ANOTHER NUMBER?
(3)  014544  001770                     BEQ     6$              ;;BR IF NO
(3)  014546  104401  014554             TYPE    ,8$             ;;TYPE TWO(2) SPACES
(3)  014552  000771                     BR      7$              ;;LOOP
(3)  014554  020040     000       8$:   .ASCIZ  /  /            ;;TWO(2) SPACES
(3)          014560                     .EVEN
(2)                               .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(2)
(3)                               ;;**************************************************************
(2)                               ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(2)                               ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(2)                               ;*$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)  014560  017646  000000       $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(2)  014564  116667  000001  000211       MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(2)  014572  112667  000207               MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(2)  014576  062716  000002               ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(2)  014602  000406                       BR      $TYPON
(2)  014604  112767  000001  000171 $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
```

```
(2)  014612  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(2)  014620  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(2)  014626  010346                          MOV     R3,-(SP)        ;;SAVE R3
(2)  014630  010446                          MOV     R4,-(SP)        ;;SAVE R4
(2)  014632  010546                          MOV     R5,-(SP)        ;;SAVE R5
(2)  014634  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(2)  014640  005404                          NEG     R4
(2)  014642  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(2)  014646  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(2)  014652  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(2)  014656  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(2)  014662  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
(2)  014664  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(2)  014666  000404                          BR      3$              ;;GO DO MSB
(2)  014670  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
(2)  014672  006105                          ROL     R5
(2)  014674  006105                          ROL     R5
(2)  014676  010503                          MOV     R5,R3
(2)  014700  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(2)  014702  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
(2)  014706  100016                          BPL     7$              ;;BR IF NO
(2)  014710  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
(2)  014714  001002                          BNE     4$              ;;TEST FOR 0
(2)  014716  005704                          TST     R4              ;;SUPPRESS THIS 0?
(2)  014720  001403                          BEQ     5$              ;;BR IF YES
(2)  014722  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(2)  014724  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(2)  014730  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(2)  014734  110367  000040                  MOVB    R3,8$           ;;SAVE FOR TYPING
(2)  014740  104401  015000                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
(2)  014744  105367  000032          7$:     DECB    $OCNT           ;;COUNT BY 1
(2)  014750  003347                          BGT     2$              ;;BR IF MORE TO DO
(2)  014752  002402                          BLT     6$              ;;BR IF DONE
(2)  014754  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(2)  014756  000744                          BR      2$              ;;GO DO THE LAST DIGIT
(2)  014760  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
(2)  014762  012604                          MOV     (SP)+,R4        ;;RESTORE R4
(2)  014764  012603                          MOV     (SP)+,R3        ;;RESTORE R3
(2)  014766  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(2)  014774  012616                          MOV     (SP)+,(SP)
(2)  014776  000002                          RTI                     ;;RETURN
(2)  015000     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(2)  015001     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(2)  015002     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(2)  015003     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(2)  015004  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
(2)                                          ;ENTER HERE ON POWER FAILURE
(2)
(2)
(2)  015006                          $PWRDN:
(2)  015006  010046                  .PFAIL: MOV     R0,-(SP)                ;SAVE R0-R5 ON PROCESSOR STACK
(2)  015010  010146                          MOV     R1,-(SP)
(2)  015012  010246                          MOV     R2,-(SP)
(2)  015014  010346                          MOV     R3,-(SP)
(2)  015016  010446                          MOV     R4,-(SP)
(2)  015020  010546                          MOV     R5,-(SP)
```

```
(2)  015022  016746  162776                MOV     24,-(SP)
(2)  015026  010667  164064                MOV     SP,SAVSP                ;SAVE STACK POINTER
(2)  015032  012767  015044  162764        MOV     #RESTART,24             ;SET UP FOR POWER UP TRAP
(2)  015040  000000                        HALT                            ;HALT ON POWER DOWN NORMAL
(2)  015042  000777                        BR      .
(2)
(2)                                        ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
(2)
(2)  015044  016706  164046        RESTAR: MOV     SAVSP,SP                ;RESTORE STACK POINTER
(2)  015050  012605                        MOV     (SP)+,R5                ;RESTORE R0-R5
(2)  015052  012604                        MOV     (SP)+,R4
(2)  015054  012603                        MOV     (SP)+,R3
(2)  015056  012602                        MOV     (SP)+,R2
(2)  015060  012601                        MOV     (SP)+,R1
(2)  015062  012600                        MOV     (SP)+,R0
(2)  015064  012767  015006  162732        MOV     #.PFAIL,24              ;SET UP FOR POWER FAILURE
(2)  015072  106427  000300                MTPS    #300
(2)  015076  012706  001100                MOV     #STACK,SP
(2)  015102  005067  001102                CLR     TEMP
(2)  015106  005267  001076                INC     TEMP
(2)  015112  001375                        BNE     .-4
(2)  015114  104413                        CONVRT
(2)  015116  015140                        PFTAB
(2)  015120  104401                        TYPE
(2)  015122  015442                        MPFAIL
(2)  015124  005067  164253                CLR     $ERFLG
(2)  015130  005067  164262                CLR     $ERRPC
(2)  015134  000177  163744                JMP     @RETURN
(2)  015140  000001                PFTAB:  1
(2)  015142     006     002                .BYTE   6,2
(2)  015144  000207                        RETURN
(2)  015146  005015  042012  053125  MTITLE: .ASCIZ  <15><12><12>/DUV11 CNDUT-A TAPE D /<15><12>
(2)  015154  030461  041440  042116
(2)  015162  052125  040455  052040
(2)  015170  050101  020105  020104
(2)  015176  005015     000
(2)  015201     015  053012  041505  MVECTO: .ASCIZ  <15><12>/VEC ADD-/
(2)  015206  040440  042104  000055
(2)  015214  005015  051461  020124  MREGAD: .ASCIZ  <15><12>/1ST DEV: REC CSR ADD-/
(2)  015222  042504  035126  051040
(2)  015230  041505  041440  051123
(2)  015236  040440  042104  000055
(2)  015244  005015  052515  052114  MMULT:  .ASCIZ  <15><12>/MULT DEV ? (Y OR N)-/
(2)  015252  042040  053105  037440
(2)  015260  024040  020131  051117
(2)  015266  047040  026451     000
(2)  015273     015  046012  051501  MLASTD: .ASCIZ  <15><12>/LAST DEV: REC CSR ADDR-/
(2)  015300  020124  042504  035126
(2)  015306  051040  041505  041440
(2)  015314  051123  040440  042104
(2)  015322  026522     000
(2)  015325     075  042504  044526  DEVICE: .ASCIZ  /=DEVICE  /
(2)  015332  042503  020040     000
(2)  015337     015  051412  046105  MCOW:   .ASCIZ  <15><12>/SELECT TO RUN @ACTREG/
(2)  015344  041505  020124  047524
(2)  015352  051040  047125  040040
```

```
(2)  015360  041501  051124  043505
(2)  015366     000
(2)  015367     015  047412  043126  MRANGE: .ASCIZ  <15><12>/OVFLO:RETYPE LAST DEV RXCSR ADDS-/
(2)  015374  047514  051072  052105
(2)  015402  050131  020105  040514
(2)  015410  052123  042040  053105
(2)  015416  051040  041530  051123
(2)  015424  040440  042104  026523
(2)  015432     000
(2)  015433     040  037440     000  MQM:    .ASCIZ  / ?/
(2)  015437     015  000012          MCRLF:  .ASCIZ  <15><12>
(2)  015442  043120  044501  026114  MPFAIL: .ASCIZ  /PFAIL,  RESTART AT TEST IN PROGRESS/
(2)  015450  020040  042522  052123
(2)  015456  051101  020124  052101
(2)  015464  052040  051505  020124
(2)  015472  047111  050040  047522
(2)  015500  051107  051505  000123
(2)  015506  005015  047105  020104  MEPASS: .ASCIZ  <15><12>/END OF PASS TAPE D/
(2)  015514  043117  050040  051501
(2)  015522  020123  040524  042520
(2)  015530  042040     000
(2)  015533     015  051012     000  MR:     .ASCIZ  <15><12>/R/
(2)  015537     015  052012  051505  MTSTPC: .ASCIZ  <15><12>/TEST PC-/
(2)  015544  020124  041520  000055
(2)  015552  005015  047514  045503  MLOCK:  .ASCIZ  <15><12>/LOCK ON  TEST? (Y OR N)-/
(2)  015560  047440  020116  052040
(2)  015566  051505  037524  024040
(2)  015574  020131  051117  047040
(2)  015602  026451     000
(2)  015605     015  021412  047440  MSYNC:  .ASCIZ  <15><12>/# OF SYNC CHARS SELECTED ( 1 OR 2)-/
(2)  015612  020106  054523  041516
(2)  015620  041440  040510  051522
(2)  015626  051440  046105  041505
(2)  015634  042524  020104  020050
(2)  015642  020061  051117  031040
(2)  015650  026451     000
(2)  015653     015  044412  020123  MWIRE6: .ASCIZ  <15><12>/IS SEC XMIT SWITCH E55-2 IN? (Y OR N)-/
(2)  015660  042523  020103  046530
(2)  015666  052111  051440  044527
(2)  015674  041524  020110  032505
(2)  015702  026465  020062  047111
(2)  015710  020077  054450  047440
(2)  015716  020122  024516  000055
(2)  015724  005015  051511  051440  MWIRE5: .ASCIZ  <15><12>/IS SEC REC SWITCH E55-3 IN? (Y OR N)-/
(2)  015732  041505  051040  041505
(2)  015740  051440  044527  041524
(2)  015746  020110  032505  026465
(2)  015754  020063  047111  020077
(2)  015762  054450  047440  020122
(2)  015770  024516  000055
(2)  015774  005015  051511  047440  MWIRE4: .ASCIZ  <15><12>/IS OPT CLR ENABLE SWITCH E55-1 IN? (Y OR N)-/
(2)  016002  052120  041440  051114
(2)  016010  042440  040516  046102
(2)  016016  020105  053523  052111
(2)  016024  044103  042440  032465
(2)  016032  030455  044440  037516
```

```
(2)  016040  024040  020131  051117
(2)  016046  047040  026451     000
(2)  016053     015  005012  031510  MEXTJ:  .ASCIZ  <15><12><12>/H315 CONNECTOR ON ?(Y OR N)-/
(2)  016060  032461  041440  047117
(2)  016066  042516  052103  051117
(2)  016074  047440  020116  024077
(2)  016102  020131  051117  047040
(2)  016110  026451     000
(2)  016113     015  020012  043536  MCNTG:  .ASCIZ  <15><12>/ ^G  /
(2)  016120  020040     000
(2)  016123     040  053523  036522  MMSWR:  .ASCIZ  / SWR=   /
(2)  016130  020040  000040
(2)  016134  020040  047040  053505  MMNEW:  .ASCIZ  /   NEW=  /
(2)  016142  020075  000040
(2)                                  .EVEN
(2)
(2)                                          ;BUFFERS FOR INPUT-OUTPUT
(2)
(2)  016146  000000                  INBUF:  0
(2)          016210                  .=.+40
(2)  016210  000000                  TEMP:   0
(2)          016252                  .=.+40
(2)  016252  000000                  MDATA:  0
(2)          016314                  .=.+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)  016314                          $SCOPE:
(5)
(5)                                          ;SCOPE LOOP AND INTERATION HANDLER
(5)
(5)  016314                          .SCOPE:
(5)  016314  004767  174376                  JSR     PC,CKSWR
(5)  016320  005067  163072                  CLR     $ERRPC          ;CLEAR LAST ERROR PC
(5)  016324  022716  003376                  CMP     #TST1+2,(SP)    ;IS SCOPE AT BEGINING OF TEST 1?
(5)  016330  001422                          BEQ     $XTSTR          ;YES NO LOOP.
(5)
(5)  016332  032777  040000  163100  TTST:   BIT     #BIT14,@SWR     ;THIS CODE IS FOR TESTING FOR BIT 14
(5)  016340  001412                          BEQ     1$              ;ON LSI WHICH SYSMAC CANNOT HANDLE
(5)  016342  016767  163034  163036          MOV     $TSTNM,$LPADR
(5)  016350  000406                          BR      1$
(5)  016352  105777  163066                  TSTB    @$TKS           ;KEYBOARD DONE?
(5)  016356  100123                          BPL     $OVER           ;BR IF NO
(5)  016360  017766  163062  177776          MOV     @$TKB,-2(SP)    ;CLEAR DONE BIT
(3)  016366  032777  040000  163044  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
(3)  016374  001114                          BNE     $OVER           ;;YES IF SW14=1
(3)                                  ;#####START OF CODE FOR THE XOR TESTER#####
```

```
(3)  016376  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
(3)                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
(3)  016400  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
(3)  016404  012737  016424  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
(3)  016412  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
(3)  016416  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(3)  016422  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
(3)  016424  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
(3)  016426  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
(3)  016432  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
(3)  016434                          6$:;#####END OF CODE FOR THE XOR TESTER#####
(3)  016434  032777  000400  162776          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
(3)  016442  001404                          BEQ     2$              ;;BR IF NO
(3)  016444  127767  162770  162730          CMPB    @SWR,$TSTNM     ;;ON THE RIGHT TEST?    SWR<7:0>
(3)  016452  001465                          BEQ     $OVER           ;;BR IF YES
(3)  016454  105767  162723          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
(3)  016460  001421                          BEQ     3$              ;;BR IF NO
(3)  016462  126767  162727  162713          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
(3)  016470  101015                          BHI     3$              ;;BR IF NO
(3)  016472  032777  001000  162740          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
(3)  016500  001404                          BEQ     4$              ;;BR IF NO
(3)  016502  016767  162702  162676  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
(3)  016510  000446                          BR      $OVER
(3)  016512  105067  162665          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(3)  016516  005067  162770                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(3)  016522  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
(3)  016524  032777  004000  162706  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
(3)  016532  001011                          BNE     1$              ;;BR IF YES
(3)  016534  005767  162774                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
(3)  016540  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
(3)  016542  005267  162636                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
(3)  016546  026767  162740  162630          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(3)  016554  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(3)  016556  012767  000001  162620  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(3)  016564  016767  000056  162720          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(3)  016572  105267  162604          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
(3)  016576  116767  162600  162726          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
(3)  016604  011667  162576                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(3)  016610  011667  162574                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(3)  016614  005067  162674                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(3)  016620  112767  000001  162567          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(3)  016626  016777  162550  162606  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
(3)  016634  016716  162546                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(5)  016640  000002                  4$:     RTI
(5)  016642  001407                  BRW:    1407
(5)  016644  000432                  BRX:    432
(3)  016646  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)  016650  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
```

```
(2)  016652  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
(2)  016656  005740                        TST     -(R0)           ;;BACKUP BY 2
(2)  016660  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(2)  016662  006300                        ASL     R0              ;;POSITION FOR INDEXING
(2)  016664  016000  016704                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(2)  016670  000200                        RTS     R0              ;;GO TO ROUTINE
(2)
(2)
(2)                                 ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
(2)
(2)  016672  011646                 $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
(2)  016674  016666  000004  000002        MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
(2)  016702  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)  016704  016672                 $TRPAD: .WORD   $TRAP2
(4)  016706  013066                         $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
(4)  016710  014604                         $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(4)  016712  014560                         $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(4)  016714  014620                         $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(2)
(2)
(3)  016716  013044                         .SCOP1  ;;CALL=SCOP1    TRAP+5(104405)
(3)  016720  013350                         .INSTR  ;;CALL=INSTR    TRAP+6(104406)
(3)  016722  013456                         .INSTER ;;CALL=INSTER   TRAP+7(104407)
(3)  016724  013466                         .PARAM  ;;CALL=PARAM    TRAP+10(104410)
(3)  016726  013666                         .SAV05  ;;CALL=SAV05    TRAP+11(104411)
(3)  016730  013726                         .RES05  ;;CALL=RES05    TRAP+12(104412)
(3)  016732  013760                         .CONVRT ;;CALL=CONVRT   TRAP+13(104413)
(3)  016734  014136                         .SETFLG ;;CALL=SETFLG   TRAP+14(104414)
8179                                ;*****************************************************************
8180                                ;UTILITIES
8181                                ;*****************************************************************
8182
8183                                ;THIS UTILITY CALCULATES PRIORITY LEVEL
8184  016736  006367  000044       DULEV:  ASL     DUPRT   ;SHIFT LEFT
8185  016742  006367  000040               ASL     DUPRT   ;
8186  016746  006367  000034               ASL     DUPRT   ;
8187  016752  006367  000030               ASL     DUPRT   ;
8188  016756  006367  000024               ASL     DUPRT   ;
8189  016762  016767  000020  000020       MOV     DUPRT,LESS1     ;MOVE THIS TO LESS1
8190  016770  162767  000001  000012       SUB     #1,LESS1        ;CREATE LESS1
8191  016776  042767  000037  000004       BIC     #37,LESS1       ;CLEAR TNZVC
8192  017004  000207                       RTS     PC
8193  017006  000240               DUPRT:  PR5
8194  017010  000200               LESS1:  PR4     ;LEVEL TO ALLOW INTERRUPTS
8195
8196                                ;NEW DU ADDRESSES
8197  017012  016767  000126  162672 DUADDR: MOV     DUBASE,RXCSR    ;XXX0
8198  017020  005267  000120               INC     DUBASE
```

```
8199  017024  016767  000114  162662          MOV     DUBASE,HRXCSR   ;XXX1
8200  017032  005267  000106                  INC     DUBASE
8201  017036  016767  000102  162652          MOV     DUBASE,RXDBUF   ;XXX2
8202  017044  016767  000074  162650          MOV     DUBASE,PARCSR   ;XXX2
8203  017052  005267  000066                  INC     DUBASE
8204  017056  016767  000062  162634          MOV     DUBASE,HRXDBUF  ;XXX3
8205  017064  016767  000054  162632          MOV     DUBASE,HPARCSR  ;XXX3
8206  017072  005267  000046                  INC     DUBASE
8207  017076  016767  000042  162622          MOV     DUBASE,TXCSR    ;XXX4
8208  017104  005267  000034                  INC     DUBASE
8209  017110  016767  000030  162612          MOV     DUBASE,HTXCSR   ;XXX5
8210  017116  005267  000022                  INC     DUBASE
8211  017122  016767  000016  162602          MOV     DUBASE,TXDBUF   ;XXX6
8212  017130  005267  000010                  INC     DUBASE
8213  017134  016767  000004  162572          MOV     DUBASE,HTXDBUF  ;XXX7
8214  017142  000207                          RTS     PC
8215  017144  000000                  DUBASE: 0
8216
8217                                          ;THIS UTILITY POKES THE MAINT DATA BASED UPON THE
8218                                          ;INFORMATION CONTAINED IN $TMP1 AND IT IS
8219                                          ;SHIFTED IN BY THE CONTENTS OF SHIFT
8220  017146  042777  040000  162552  RPOKE:  BIC     #MTDATA,@TXCSR
8221  017154  005067  162322                  CLR     $TMP2
8222  017160  006067  162314                  ROR     $TMP1   ;FORCE CARRY
8223  017164  006067  162312                  ROR     $TMP2   ;PICK UP CARRY IN BIT 15
8224  017170  006267  162306                  ASR     $TMP2   ;SHIFT INTO BIT 14
8225  017174  042767  100000  162300          BIC     #BIT15,$TMP2    ;CLR BIT 15
8226  017202  056777  162274  162516          BIS     $TMP2,@TXCSR    ;POKE MAINT DATA
8227  017210  042777  020000  162510          BIC     #CLK,@TXCSR     ;POKE CLK
8228  017216  052777  020000  162502          BIS     #CLK,@TXCSR     ;
8229  017224  005367  161672                  DEC     SHIFT
8230  017230  001346                          BNE     RPOKE
8231  017232  000207                          RTS     PC
8232
8233                                          ;THIS ROUTINE CALCULATES ODD PARITY FOR AN 8 BIT CHAR
8234  017234  016767  162240  162240  ODD8:   MOV     $TMP1,$TMP2     ;SAVE TEMP1
8235  017242  005067  162236                  CLR     $TMP3
8236  017246  012727  000010                  MOV     #8.,(PC)+
8237  017252  000000                  4$:     0
8238  017254  006067  162222          1$:     ROR     $TMP2
8239  017260  005567  162220                  ADC     $TMP3
8240  017264  005367  177762                  DEC     4$
8241  017270  001371                          BNE     1$
8242  017272  006067  162206                  ROR     $TMP3
8243  017276  103404                          BCS     2$
8244  017300  052767  000400  162172          BIS     #BIT8,$TMP1     ;SET ODD PARITY
8245  017306  000403                          BR      3$
8246  017310  042767  000400  162162  2$:     BIC     #BIT8,$TMP1     ;CLR EVEN PARITY
8247                                          ;$TMP1 NOW HAS ODD PARITY CHARACTER
8248  017316  000207                  3$:     RTS     PC
8249
8250                                          ;THIS ROUTINE CALCULATES EVEN PARITY FOR AN 8 BIT CHARACTER
8251  017320  016767  162154  162154  EVEN8:  MOV     $TMP1,$TMP2     ;SAVE TEMP1
8252  017326  005067  162152                  CLR     $TMP3
8253  017332  012727  000010                  MOV     #8.,(PC)+
8254  017336  000000                  4$:     0
```

```
 8255  017340  006067  162136          1$:     ROR     $TMP2
 8256  017344  005567  162134                  ADC     $TMP3
 8257  017350  005367  177762                  DEC     4$
 8258  017354  001371                          BNE     1$
 8259  017356  006067  162122                  ROR     $TMP3
 8260  017362  103004                          BCC     2$
 8261  017364  052767  000400  162106          BIS     #BIT8,$TMP1     ;SET EVEN PARITY
 8262  017372  000403                          BR      3$
 8263  017374  042767  000400  162076  2$:     BIC     #BIT8,$TMP1     ;CLR ODD PARITY
 8264                                          ;$TMP1 NOW HAS EVEN PARITY CHARACTER
 8265  017402  000207                  3$:     RTS     PC
 8266  017404  062716  000002          TRPREG: ADD     #2,(SP) ;ALLOW IT TO "CRUNCH" INTO HLT BACK
 8267                                                          ;IN MAIN PART OF THE PROGRAM
 8268  017410  000002                          RTI
 8269          017412                  POINT=.                 ;SAVE POINTER
  (1)          000100                  .=100
  (1)  000100  017412                  $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)          017412                  .=POINT                 ;RESTORE POINTER
  (1)  017412  104401  017420          $CLKVEC:        TYPE,CLKMES
  (1)  017416  000000                  CLKMES:         HALT
  (1)  017420  005015  045514  042526          .ASCIZ <15><12>/LKVEC INTERRUPT - DISCONNECT LTC /
  (1)  017426  020103  047111  042524
  (1)  017434  051122  050125  020124
  (1)  017442  020055  044504  041523
  (1)  017450  047117  042516  052103
  (1)  017456  046040  041524  000040
 8270          000001                  .END
```

```
AAA     003200          8149#
ABASE = 000000          8149
ACDW1 = 000000          8149
ACDW2 = 000000          8149
ACPUOP= 000000          8149
ACTREG  001166          8149#*  8178
ADDW0 = 000000          8149
ADDW1 = 000000          8149
ADDW10= 000000          8149
ADDW11= 000000          8149
ADDW12= 000000          8149
ADDW13= 000000          8149
ADDW14= 000000          8149
ADDW15= 000000          8149
ADDW2 = 000000          8149
ADDW3 = 000000          8149
ADDW4 = 000000          8149
ADDW5 = 000000          8149
ADDW6 = 000000          8149
ADDW7 = 000000          8149
ADDW8 = 000000          8149
ADDW9 = 000000          8149
ADEVCT= 000000          8149
ADEVM = 000000          8149
ADRCNT= 013665          8178#*
AENV  = 000000          8149
AENVM = 000000          8149
AFATAL= 000000          8149
AMADR1= 000000          8149
AMADR2= 000000          8149
AMADR3= 000000          8149
AMADR4= 000000          8149
AMAMS1= 000000          8149
AMAMS2= 000000          8149
AMAMS3= 000000          8149
AMAMS4= 000000          8149
AMSGAD= 000000          8149
AMSGLG= 000000          8149
AMSGTY= 000000          8149
AMTYP1= 000000          8149
AMTYP2= 000000          8149
AMTYP3= 000000          8149
AMTYP4= 000000          8149
APASS = 000000          8149
APRIOR= 000000          8149
APTCSU= 000040          7009#   8178
APTENV= 000001          7009#   8178
APTSIZ= 000200          7009#   8149
APTSPO= 000100          7009#   8178
ASWREG= 000000          8149
ATESTN= 000000          8149
AUNIT = 000000          8149
AUSWR = 000000          8149
AVECT1= 000000          8149
AVECT2= 000000          8149
BASEAD  001154          8149#*  8178*
```

```
BASEIV  001162          8149#* 8178*
BBB     003026          8149#
BINWRD  014134          8178#*
BITW  = 002000          8149#  8156  8157  8158  8159  8160  8161  8162  8163  8164  8165  8166  8167
                        8169   8170  8171  8172  8175  8176
BIT0  = 000001          8149#
BIT00 = 000001          8149#
BIT01 = 000002          8149#
BIT02 = 000004          8149#
BIT03 = 000010          8149#
BIT04 = 000020          8149#
BIT05 = 000040          8149#
BIT06 = 000100          8149#
BIT07 = 000200          8149#
BIT08 = 000400          8149#  8178
BIT09 = 001000          8149#  8178
BIT1  = 000002          8149#
BIT10 = 002000          8149#  8178
BIT11 = 004000          8149#  8178
BIT12 = 010000          8149#
BIT13 = 020000          8149#  8178
BIT14 = 040000          8149#  8178
BIT15 = 100000          8149#  8225
BIT2  = 000004          8149#
BIT3  = 000010          8149#
BIT4  = 000020          8149#
BIT5  = 000040          8149#
BIT6  = 000100          8149#
BIT7  = 000200          8149#
BIT8  = 000400          8149#  8244  8246  8261  8263
BIT9  = 001000          8149#
BPTVEC= 000014          8149#
BREAK = 000001          8149#  8151  8152  8153  8154  8155
BRKVEC= 000140          8149#
BRW     016642          8178#
BRX     016644          8178#
CARDET= 010000          8149#
CCC     012622          8178#
CHRCNT  014132          8178#*
CKSWR   012716          8178#
CLK   = 020000          8149#  8151  8152  8153  8154  8155  8156  8157  8158  8159  8160  8161  8162
                        8163   8164  8165  8166  8167  8169  8170  8171  8172  8175  8176  8227  8228
CLKMES  017420          8269#
CNTLU   012772          8149   8178#
CONVRT= 104413          8178#
COUNT   001124          8149#  8151* 8152* 8153* 8154* 8155*
CR    = 000015          8149#  8178
CRLF  = 000200          8149#  8178
CTS   = 020000          8149#
DDISP = 177570          8149#
DEVADR  013662          8178#*
DEVICE  015325          8178#
DF1     002132          8149#
DH1     002067          8149#
DISPLA  001442          8149#* 8178*
DISPRE  000174          8149#
```

```
DNA    = 100000          8149#
DNAINT= 000040           8149#
DSC    = 100000          8149#
DSINTE= 000040           8149#
DSR    = 001000          8149#
DSWR   = 177570          8149#
DTR    = 000002          8149#
DT1      002116          8149#
DT4      002126          8149#
DUADDR   017012          8149    8178    8197#
DUBASE   017144          8149    8178*   8197    8198*   8199    8200*   8201    8202    8203*   8204    8205    8206*   8207
                         8208*   8209    8210*   8211    8212*   8213    8215#
DULEV    016736          8149    8184#
DUPRT    017006          8149*   8184*   8185*   8186*   8187*   8188*   8189    8193#
DURIS    001740          8149#   8178*
DURIV    001736          8149#   8178*
DUTIS    001744          8149#   8178*
DUTIV    001742          8149#   8178*
EIGHT  = 006000          8149#   8151    8152    8159    8163
EMTVEC= 000030           8149#*
EM1      001762          8149#
EM2      002022          8149#
EM3      002043          8149#
EM4      001746          8149#
ERRCNT   001114          8149#
ERRVEC= 000004           8149#*  8178*
EVEN8    017320          8251#
EVEPAR= 001400           8149#   8152    8153    8154    8155    8164    8166    8169    8171    8175
EVPAR  = 000400          8149#
FIVE   = 000000          8149#   8155    8156    8160    8164    8165    8166    8167
FRMERR= 020000           8149#   8152    8153    8154    8155
GNS    = ****** U        8178
HDXEN  = 000010          8149#
HILIM    013660          8178#*
HLD0     001130          8149#   8178*
HLD1     001132          8149#   8178*
HLD2     001134          8149#   8178*
HLD3     001136          8149#   8178*
HLD4     001140          8149#   8178*
HLD5     001142          8149#   8178*
HLD6     001144          8149#   8178*
HOLD     001120          8149#
HPARCS   001724          8149#   8205*
HRXCSR   001714          8149#   8199*
HRXDBU   001720          8149#   8204*
HT     = 000011          8149#   8178
HTXCSR   001730          8149#   8209*
HTXDBU   001734          8149#   8213*
INBUF    016146          8149    8178#
INIFLG   001172          8149#*
INSTER= 104407           8149    8178#
INSTR  = 104406          8149    8178#
INSTR2   013464          8178#
IOTVEC= 000020           8149#*
ISYMOD= 000000           8149#   8152    8153    8154    8155    8166    8167    8171    8172
JMRBY    001153          8149#*
```

```
KEEPAD  001156          8149#*  8178
KEEPIV  001164          8149#*  8178
LASTAD  001160          8149#
LESS1   017010          8189*   8190*   8191*   8194#
LF    = 000012          8149#   8178
LIGHTS  001102          8149#   8178*
LIMITS  013612          8178#
LKVEC = 000100          8149#
LOBITS  013664          8178#*
LOCK    001110          8149#   8178
LOKFLG  001174          8149#
LOLIM   013656          8178#*
MCNTG   016113          8178#
MCOW    015337          8178#
MCRLF   015437          8178#
MDATA   016252          8178#
MEPASS  015506          8178#
MEXT  = 010000          8149#
MEXTJ   016053          8149    8178#
MINT  = 004000          8149#   8151    8152    8153    8154    8155    8156    8157    8158    8159    8160    8161    8162
                        8163    8164    8165    8166    8167    8169    8170    8171    8172    8175    8176
MLASTD  015273          8149    8178#
MLOCK   015552          8178#
MMNEW   016134          8178#
MMSWR   016123          8178#
MMULT   015244          8149    8178#
MPFAIL  015442          8178#
MQM     015433          8178#
MR      015533          8149    8178#
MRANGE  015367          8149    8178#
MREGAD  015214          8149    8178#
MRESET= 000400          8149#   8151    8152    8153    8154    8155    8156    8157    8158    8159    8160    8161    8162
                        8163    8164    8165    8166    8167    8169    8170    8171    8172    8175    8176
MSYNC   015605          8149    8178#
MTDATA= 040000          8149#   8151    8152    8153    8154    8155    8220
MTITLE  015146          8149    8178#
MTSTPC  015537          8149    8178#
MULTD   001152          8149#   8178
MVECTO  015201          8149    8178#
MWIRE4  015774          8149    8178#
MWIRE5  015724          8149    8178#
MWIRE6  015653          8149    8178#
NEXT    001106          8149#
NOPAR = 000000          8149#   8151    8156    8157    8158    8159    8160    8161    8162    8163
ODDPAR= 001000          8149#   8165    8167    8170    8172    8176
ODD8    017234          8234#
ODTST = 170000          8149#
ONCE    002610          8149#
OPTCLR  001151          8149#
OUT     013024          8178#
OUTCRY  012614          8178#
OUTMUL  003166          8149#
OVRRUN= 040000          8149#   8152    8153    8154    8155
PARAM = 104410          8149    8178#
PARAM1  013516          8178#
PARCSR  001722          8149#   8151*   8152*   8153*   8154*   8155*   8156*   8157*   8158*   8159*   8160*   8161*   8162*
```

```
                    8163*  8164*  8165*  8166*  8167*  8169*  8170*  8171*  8172*  8175*  8176*  8202*
PAREN = 001000      8149#
PARER = 010000      8149#  8152   8153   8154   8155
PARERR  013572      8178#
PARTI   013654      8178#
PASCNT  001112      8149#* 8178*
PFTAB   015140      8178#
PIRQ  = 177772      8149#
PIRQVE= 000240      8149#
POINT = 017412      8269#
POPR0 = 012600      8149#
POP1SP= 005726      8149#
POP2SP= 022626      8149#
PR0   = 000000      8149#
PR1   = 000040      8149#
PR2   = 000100      8149#
PR3   = 000140      8149#
PR4   = 000200      8149#  8194
PR5   = 000240      8149#  8193
PR6   = 000300      8149#
PR7   = 000340      8149#
PS    = 177776      8149#
PSW   = 177776      8149#
PUSHR0= 010046      8149#
PUSH1S= 005746      8149#
PUSH2S= 024646      8149#
PWRVEC= 000024      8149#*
RDSW    013032      8178#*
RECACT= 004000      8149#
REPLAY  012520      8178#
RESTAR  015044      8178#
RESTRT  012700      8178#
RESVEC= 000010      8149#
RESO5 = 104412      8178#
RETURN  001104      8149#* 8178
RING  = 040000      8149#
RINTEN= 000100      8149#
ROTADD  001170      8149#* 8178*
RPOKE   017146      8151   8152   8153   8154   8155   8220#  8230
RTS   = 000004      8149#
RUNA  = ****** U    5678   5862   6003   6417
RUNB  = ****** U    5703   5871   6010   6426
RUNC  = ****** U    5728   5379   6017   6435
RUND  = 000000      5676#  5753   5888   6024   6444
RUNE  = ****** U    5779   5898   6031   6453
RUNF  = ****** U    5804   5908   6038   6462
RUNIT   012424      8178#
RXCSR   001712      8149#  8151*  8152*  8153*  8154*  8155*  8178   8197*
RXDBUF  001716      8149#  8152   8153   8154   8155   8201*
RXDONE= 000200      8149#
RXERR = 100000      8149#  8152   8153   8154   8155
SAVIT   014366      8178#
SAVPC   001126      8149#  8178*
SAVSP   001116      8149#  8178*
SAVO5 = 104411      8178#
SCOP1 = 104405      8178#
```

```
SEND  = 000020   8149#  8156  8157  8158  8159  8160  8161  8162  8163  8164  8165  8166  8167
                 8169  8170  8171  8172  8175  8176
SEREC   001150   8149#
SETFLG= 104414   8149  8178#
SEVEN = 004000   8149#  8153  8158  8162  8175  8176
SEXMIT  001147   8149#
SHIFT   001122   8149#  8151* 8152* 8153* 8154* 8155* 8156* 8157* 8158* 8159* 8160* 8161* 8162*
                 8163* 8164* 8165* 8166* 8167* 8169* 8170* 8171* 8172* 8175* 8176* 8229*
SIX   = 002000   8149#  8154  8157  8161  8169  8170  8171  8172
SPACNT= 014133   8178#*
SRD   = 002000   8149#
STACK = 001100   8149#  8178
STD   = 000010   8149#
STFLG   001173   8149#*
STKLMT= 177774   8149#
STPSYN= 000400   8149#  8151  8152  8153  8154  8155
SVOS    013674   8178#
SWR     001440   8149#*  8178
SWREG   000176   8149#  8178
SWREGL  013034   8178#
SW0   = 000001   8149#
SW00  = 000001   8149#
SW01  = 000002   8149#
SW02  = 000004   8149#
SW03  = 000010   8149#
SW04  = 000020   8149#
SW05  = 000040   8149#
SW06  = 000100   8149#
SW07  = 000200   8149#
SW08  = 000400   8149#
SW09  = 001000   8149#  8178
SW1   = 000002   8149#
SW10  = 002000   8149#
SW11  = 004000   8149#
SW12  = 010000   8149#
SW13  = 020000   8149#
SW14  = 040000   8149#
SW15  = 100000   8149#
SW2   = 000004   8149#
SW3   = 000010   8149#
SW4   = 000020   8149#
SW5   = 000040   8149#
SW6   = 000100   8149#
SW7   = 000200   8149#
SW8   = 000400   8149#
SW9   = 001000   8149#
SYNCNO  001146   8149#*
SYNEXT= 020000   8149#  8160  8161  8162  8163
SYNINT= 030000   8149#  8151  8156  8157  8158  8159  8164  8165  8169  8170  8175  8176
SYNSCH= 000020   8149#  8151  8152  8153  8154  8155
SYSTST= 014000   8149#
TBITVE= 000014   8149#
TEMP    016210   8178#*
TKVEC = 000060   8149#
TPVEC = 000064   8149#
TRAPVE= 000034   8149#*
```

```
TRPREG  017404          8266#
TRTVEC= 000014          8149#
TST1    003374          8149    8151#   8178
TST10   006130          8158#
TST11   006346          8159#
TST12   006564          8160#
TST13   007002          8161#
TST14   007220          8162#
TST15   007436          8163#
TST16   007654          8164#
TST17   010062          8165#
TST2    003544          8152#
TST20   010270          8166#
TST21   010476          8167#
TST22   010704          8169#
TST23   011112          8170#
TST24   011320          8171#
TST25   011526          8172#
TST26   011734          8175#
TST27   012142          8176#
TST3    004132          8153#
TST4    004520          8154#
TST5    005106          8155#
TST6    005474          8156#
TST7    005712          8157#
TTST    016332          8178#
TXCSR   001726          8149#   8151*   8152*   8153*   8154*   8155*   8156*   8157*   8158*   8159*   8160*   8161*   8162*
                        8163*   8164*   8165*   8166*   8167*   8169*   8170*   8171*   8172*   8175*   8176*   8207*   8220*
                        8226*   8227*   8228*
TXDBUF  001732          8149#   8156*   8157*   8158*   8159*   8160*   8161*   8162*   8163*   8164*   8165*   8166*   8167*
                        8169*   8170*   8171*   8172*   8175*   8176*   8211*
TXDONE= 000200          8149#
TXINTE= 000100          8149#
TYPE  = 104401          8149    8178#   8269
TYPOC = 104402          8178#
TYPON = 104404          8178#
TYPOS = 104403          8178#
USER  = 000000          8149#
VOID  = 000001          8149#
WRDCNT  014130          8178#*
ZERO    012432          8178#
$APTHD  002136          8149#
$ASTAT= ****** U        7009
$ATYC   000024          7009#
$ATY1   000000          7009#
$ATY3   000006          7009#   8178
$ATY4   000016          7009#   8178
$AUTOB  001434          8149#
$BASE   001602          8149#
$BDADR  001422          8149#
$BDDAT  001426          8149#
$BELL   001516          8149#   8178
$CDW1   001606          8149#
$CDW2   001610          8149#
$CHARC  013344          8178#*
$CKSWR= ****** U        8178
```

```
$CLKVE  017412          8269#
$CMTAG  001400          8149#
$CM1   = 000006         8149#
$CM2   = 000014         8149#
$CM3   = 000006         8149#
$CM4   = 000006         8149#
$CPUOP  001554          8149#
$CRLF   001523          8149#   8178
$DDW0   001612          8149#
$DDW1   001614          8149#
$DDW10  001636          8149#
$DDW11  001640          8149#
$DDW12  001642          8149#
$DDW13  001644          8149#
$DDW14  001646          8149#
$DDW15  001650          8149#
$DDW2   001616          8149#
$DDW3   001620          8149#
$DDW4   001622          8149#
$DDW5   001624          8149#
$DDW6   001626          8149#
$DDW7   001630          8149#
$DDW8   001632          8149#
$DDW9   001634          8149#
$DEVCT  001536          8149#
$DEVM   001604          8149#
$E     = 000002         6837#
$ENDAD  012670          8149    8178#
$ENV    001546          7009    8149#   8178
$ENVM   001547          7009    8149#   8178
$ERFLG  001403          8149#*  8178*
$ERMAX  001415          8149#*  8178*
$ERROR  014204          8149    8178#
$ERRPC  001416          8149#*  8178*
$ERRTB  001652          8149#   8178
$ERRTY  014424          8178#
$ERTTL  001412          8149#*  8178*
$ESCAP  001514          8149#*  8178*
$ETABL  001546          8149#
$ETEND  001652          8149#
$FATAL  001530          7009*   8149#
$FFLG   000244          7009#*
$FILLC  001456          8149#   8178
$FILLS  001455          8149#   8178
$GDADR  001420          8149#
$GDDAT  001424          8149#
$GTSWR= ****** U        8178
$HIBTS  002136          8149#
$ICNT   001404          8149#   8178*
$INTAG  001435          8149#
$ITEMB  001414          8149#   8178*
$LF     001524          8149#   8178
$LFLG   000243          7009#*
$LPADR  001406          8149#*  8178*
$LPERR  001410          8149#*  8178*
$MADR1  001560          8149#
```

```
$MADR2  001564            8149#
$MADR3  001570            8149#
$MADR4  001574            8149#
$MAIL   001526            8149#   8178
$MAMS1  001556            8149#
$MAMS2  001562            8149#
$MAMS3  001566            8149#
$MAMS4  001572            8149#
$MBADR  002140            8149#
$MFLG   000242            7009#*
$MSGAD  001542            7009*   8149#
$MSGLG  001544            7009*   8149#
$MSGTY  001526            7009*   8149#
$MTYP1  001557            8149#
$MTYP2  001563            8149#
$MTYP3  001567            8149#
$MTYP4  001573            8149#
$MXCNT  016646            8178#
$N    = 000000            6836#   8178#
$NULL   001454            8149#   8178
$NWTST= 000000            8151#   8152#   8153#   8154#   8155#   8156#   8157#   8158#   8159#   8160#   8161#   8162#   8163#
                          8164#   8165#   8166#   8167#   8169#   8170#   8171#   8172#   8175#   8176#
$OCNT   015002            8178#*
$OMODE  015004            8178#*
$OVER   016626            8178#
$PASS   001534            8149#*  8178*
$PASTM  002144            8149#
$PWRDN  015006            8149    8178#
$QUES   001522            8149#   8178
$RDCHR= ****** U          8178
$RDDEC= ****** U          8178
$RDLIN= ****** U          8178
$RDOCT= ****** U          8178
$REGAD  001460            8149#
$REG0   001462            8149#   8178*
$REG1   001464            8149#   8178*
$REG2   001466            8149#   8178*
$REG3   001470            8149#   8178*
$REG4   001472            8149#   8178*
$REG5   001474            8149#   8178*
$R2A  = ****** U          8178
$SAVRE= ****** U          8178
$SCOPE  016314            8149    8178#
$SETUP= 000017            8149#   8178
$STUP = 177777            8149#
$SVLAD  016572            8178#
$SVPC = 002136            8149#
$SWR  = 177400            6496#   8149    8151    8152    8153    8154    8155    8156    8157    8158    8159    8160    8161
                          8162    8163    8164    8165    8166    8167    8169    8170    8171    8172    8175    8176    8178
$SWREG  001550            8149#
$SWRMK= 000000            8178
$TESTN  001532            8149#   8178*
$TIMES  001512            8149#*  8178*
$TKB    001446            8149#   8178
$TKS    001444            8149#   8178
$TMPO   001476            8149#
```

```
$TMP1   001500          8149#  8151*  8152*  8153*  8154*  8155*  8156*  8157*  8158*  8159*  8160*  8161*  8162*
                        8163*  8164*  8165*  8166*  8167*  8169*  8170*  8171*  8172*  8175*  8176*  8222*  8234
                        8244*  8246*  8251   8261*  8263*
$TMP2   001502          8149#  8221*  8223*  8224*  8225*  8226   8234*  8238*  8251*  8255*
$TMP3   001504          8149#  8235*  8239*  8242*  8252*  8256*  8259*
$TMP4   001506          8149#
$TMP5   001510          8149#
$TN   = 000030          7248#  8151#  8152#  8153#  8154#  8155#  8156#  8157#  8158#  8159#  8160#  8161#  8162#
                        8163#  8164#  8165#  8166#  8167#  8169#  8170#  8171#  8172#  8175#  8176#
$TPB    001452          8149#  8178*
$TPFLG  001457          8149#  8178
$TPS    001450          8149#  8178
$TRAP   016650          8149   8178#
$TRAP2  016672          8178#
$TRP  = 000015          8178#
$TRPAD  016704          8178#
$TSTM   002142          8149#
$TSTNM  001402          8149#* 8178*
$TYPBN= ****** U        8178
$TYPDS= ****** U        8178
$TYPE   013066          7009   8178#
$TYPEC  013300          8178#
$TYPEX  013346          8178#
$TYPOC  014604          8178#
$TYPON  014620          8178#
$TYPOS  014560          8178#
$UNIT   001540          8149#
$UNITM  002146          8149#
$USWR   001552          8149#
$VECT1  001576          8149#
$VECT2  001600          8149#
$XTSTR  016376          8178#
$OFILL  015003          8178#*
$40CAT= ****** U        8178
.     = 017464          7009#  8149#  8151   8152   8153   8154   8155   8156   8157   8158   8159   8160   8161
                        8162   8163   8164   8165   8166   8167   8169   8170   8171   8172   8175   8176   8178#
                        8269#
.BEGIN  003310          8149#  8178
.CONVR  013760          8178#
.EOP    012350          8178#
.INSTE  013456          8178#
.INSTR  013350          8178#
.INST1  013370          8178#
.MSG    013372          8178#*
.PARAM  013466          8178#
.PFAIL  015006          8149   8178#
.RESO5  013726          8178#
.SAVO5  013666          8178#
.SCOPE  016314          8178#
.SCOP1  013044          8178#
.SETFL  014136          8178#
.START  002152          8149#  8178
.$ASTA= ****** U        7009
.$X   = 002136          8149#
```

72

```
COMMEN  1566#  8149#
ENDCOM  1578#  8149#
ERROR   8149#  8151  8152  8153  8154  8155  8156  8157  8158  8159  8160  8161  8162  8163  8164
        8165   8166  8167  8169  8170  8171  8172  8175  8176
ESCAPE  1698#  8149#
GETPRI  1312#  8149#
GETSWR  1771#  8149#
MULT    4484#  8149#
NEWTST  1626#  8149#  8151  8152  8153  8154  8155  8156  8157  8158  8159  8160  8161  8162  8163
        8164   8165  8166  8167  8169  8170  8171  8172  8175  8176
POP     2153#  7009  8149#
PRGEND  8114#  8178
PRGFRT  7249#  8149
PUSH    2145#  7009  8149#
PUSSYF  7985#
REPORT  5463#  6495#  8149#
RSETUP  7444#  8151  8152  8153  8154  8155
SC      6809#  8178
SCOPE   8149#  8151  8152  8153  8154  8155  8156  8157  8158  8159  8160  8161  8162  8163  8164
        8165   8166  8167  8169  8170  8171  8172  8175  8176  8178
SC1     6827#  8178
SETPRI  1279#  8149#
SETTRA  6494#  8178#
SETUP   1337#  8149#
SKIP    1733#  8149#
SLASH   1517#  8149#
SPACE   8149#
STARS   1485#  7009  8149#  8151  8152  8153  8154  8155  8156  8157  8158  8159  8160  8161  8162
        8163   8164  8165  8166  8167  8169  8170  8171  8172  8175  8176  8178
SWRSU   1453#  8149#
TRMTRP  8178#
TSETUP  7456#  8156  8157  8158  8159  8160  8161  8162  8163  8164  8165  8166  8167  8169  8170
        8171   8172  8175  8176
TYPBIN  2088#  8149#
TYPDEC  2058#  8149#
TYPNAM  1826#  8149#
TYPNUM  2025#  8149#
TYPOCS  1978#  8149#
TYPOCT  1941#  8149#  8178
TYPTXT  1894#  8149#
$BEGIN  6718#  8149
$BINAR  7932#
$BUFFE  7123#  8178
$CABLE  8000#
$CATCH  6587#  8149
$CK     6963#  8178
$CLRVE  6733#  8149
$CONVR  7134#  8178
$DNA    7833#  8156  8157  8158  8159  8160  8161  8162  8163
$EOP    6881#  8178
$GETFL  6763#  8149
$GETPA  6753#  8149
$GETSY  6769#  8149
$HEADE  6498#  8149
$INSTR  7010#  8178
$ISOB   7761#
```

```
$MATCH  7570#
$MRR    7431#
$MRRW   7379#
$MRW    7414#
$MSG    7097#  8178
$NEED   8102#  8178
$PARAM  7040#  8178
$PFAIL  7179#  8178
$POKE   7553#  8156  8157  8158  8159  8160  8161  8162  8163  8164  8165  8166  8167  8169  8170
        8171   8172  8175  8176
$POKER  7558#  8151  8152  8153  8154  8155
$RCNET  8012#  8152  8153  8154  8155
$RECAC  7492#
$REG    7222#  8178
$RESET  7375#  8151  8152  8153  8154  8155  8156  8157  8158  8159  8160  8161  8162  8163  8164
        8165   8166  8167  8169  8170  8171  8172  8175  8176
$RXACT  7469#
$SCOPE  6832#  8178
$SCOP1  6998#  8178
$SETFL  6789#  8178
$SETVE  6598#  8149
$START  6676#  8149
$STRIP  7802#  8151
$SYMBO  6514#  8149
$SYNCR  7564#  8151  8152  8153  8154  8155
$TRPAR  7884#  8164  8165  8166  8167  8169  8170  8171  8172  8175  8176
$TSTNO  6839#  8151  8152  8153  8154  8155  8156  8157  8158  8159  8160  8161  8162  8163  8164
        8165   8166  8167  8169  8170  8171  8172  8175  8176
$UNIBU  7358#
$VARIA  6614#  8149
$WORDF  7700#
$WORDO  7650#
$WORDP  7728#
$$CMRE  8149#
$$CMTM  8149#
$$ESCA  1711#  8149#
$$NEWT  1662#  8149#  8151  8152  8153  8154  8155  8156  8157  8158  8159  8160  8161  8162  8163
        8164   8165  8166  8167  8169  8170  8171  8172  8175  8176
$$SET   8178#
$$SETM  8149#
$$SKIP  1746#  8149#
.EQUAT   189#  6494#  8149
.HEADE    61#  6493#
.INIT   5658#  8149#  8269
.SETUP  1213#  6493#  8149
.SWRHI   104#
.$ACT1  5064#  6495#  8149
.$APTB  5109#  6495#  8149#
.$APTH  5370#  6495#  8149
.$APTY  5547#  6495#  7009
.$ASTA  5417#
.$CATC   932#  6493#
.$CMTA  1047#  6493#  8149
.$DB2D  4686#
.$DB2O  4812#
.$DIV   4587#
```

```
.$EOP    2214#   6493#
.$ERRO   2700#   6494#   8178
.$ERRT   2896#   6494#   8178
.$MULT   4523#
.$POWE   4229#   6494#
.$RAND   4307#
.$RDDE   3891#
.$RDOC   3797#
.$READ   3395#
.$R2AZ   4958#
.$SAVE   3969#
.$SB2D   4771#
.$SB2O   4874#
.$SCOP   2454#   6494#   8178
.$SIZE   4361#
.$SUPR   4913#
.$TRAP   4073#   6494#   8178
.$TYPB   3287#
.$TYPD   3209#
.$TYPE   2985#   6493#   8178
.$TYPO   3112#   6494#   8178
.$40CA    972#

. ABS.  017464     000

 ERRORS DETECTED:  0

CNDUTA,CNDUTA/CRF/NL:TOC=CNMAC2.SML,CNDUT1.M11,CNDUT2.M11,CNDUTA.M11
RUN-TIME:  17 16 1 SECONDS
RUN-TIME RATIO: 82/35=2.3
CORE USED:  43K  (86 PAGES)
```