

DUV11

OFLNE RCVR TMG  
CNDUSAO

AH-T454A-MC  
FICHE 1 OF 1

MAY 1983  
COPYRIGHT © 82-83  
MADE IN USA





.REM \*

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

PRODUCT NAME: CNDUSAO DUV11 OFLNE RCVR TMG

PRODUCT CODE: AC-T453A-MC

PRODUCT DATE: DEC, 1982

MAINTAINER : DIAGNOSTICS SERVICES/ISS

AUTHOR: K.LIND

\*  
.REM \*

COPYRIGHT (C) 1982, 1983  
DIGITAL EQUIPMENT CORPORATION, MAYNARD MASS.

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED UNDER A LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OF RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

\*



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 RECEIVER TIMING TESTS VERIFY THAT THE RECEIVER LOGIC AND ASSOCIATED ERROR FLAGS ASSERT AT THE PROPPER TIME

\*  
.REM \*

2. REQUIREMENTS

PDP-11/21 COMPUTER (LSI)

DUV11 SYNCHRONOUS/ISOCRONOUS OPTION

ONE CONSOLE TELETYPE OR EQUIVALENT

\*  
.REM \*

- 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 AVAILABILITY 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 AVAILABLE.



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 CNDUS-A TAPE C" (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

\*  
.REM \*

\*  
.REM \*



- 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 ADDRESSES  
&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

### 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

- 9.1 THIS PROGRAM PERFORMS THE OFFLINE RECEIVER TIMING TESTING  
OF THE DEVICE  
SEE LISTING FOR DETAILS

\*  
.REM \*

\*  
.REM \*

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

11. REVISION HISTORY  
DZDUSB1 DIAGNOSTIC WAS MODIFIED TO WORK IN SBC 11/21 PROCESSOR  
PRIORITY 340 WAS CHANGED TO 300 WHEREVER ENCOUNTERED  
DEFAULT CSR AND VECTOR ADDRESSES WERE CHANGED  
.INIT CALL ADDED AT THE END OF PROGRAM  
DIAGNOSTIC WAS RENAMED TO CNDUSA0.  
12. LISTINGS

\*



6488  
 6489  
 6490  
 6491  
 6585  
 6586  
 6587  
 6598  
 6612  
 6613  
 6614  
 6675  
 6676  
 6881  
 7009  
 7010

:IN ORDER TO BE SPECIFIC TO SBC 11/21 PROCESSOR  
 :ODT VECTOR ADDRESS AND BEVNT WERE INITIALIZED AND  
 :ALL PRIORITY WORD 340 WERE MODIFIED TO 300.

.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,$MSGAD
(1) ;;PUT ADDR IN MAILBOX
(1) 2$: TSTB (R0)+ ;;FIND END OF MESSAGE
(1) 000104 001376 BNE 2$
(1) 000106 166700 001430 SUB $MSGAD,R0 ;;SUB START OF MESSAGE
(1) 000112 006200 ASR R0 ;;GET MESSAGE LNTH IN WORDS
(1) 000114 010067 001424 MOV R0,$MSGGLT ;;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 012702 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
  
```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 58-1  
 CNDUS2.M11 30-OCT-82 12:20 APT COMMUNICATIONS ROUTINE

SEQ 0012

```

(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)              .EVEN
(1)              APTSIZE=200
(1)              APTENV=001
(1)              APTSPool=100
(1)              APTCSUP=040
7249      000001      $TN=1
7375
7379
7414
7431
7444
7456
7469
  
```



N 1

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 60  
CNDUS2.M11 30-OCT-82 12:20 APT COMMUNICATIONS ROUTINE

SEQ 0013

7492  
7558  
7564  
7700  
7728  
7761  
7802  
7833  
7884  
7932  
7985  
8000  
8012  
8114





CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-1  
 CNDUSA.M11 30-OCT-82 10:55 BASIC DEFINITIONS

SEQ 0015

```

(2)                                     ;*"SWITCH REGISTER" SWITCH DEFINITIONS
(2)      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)                                     ;*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

```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-2  
 CNDUSA.M11 30-OCT-82 10:55 BASIC DEFINITIONS

SEQ 0016

```

(2)          : *BASIC "CPU" TRAP VECTOR ADDRESSES
(2)          ERRVEC= 4          : TIME OUT AND OTHER ERRORS
(2)          RESVEC= 10         : RESERVED AND ILLEGAL INSTRUCTIONS
(2)          TBITVEC=14         : "T" BIT
(2)          TRTVEC= 14         : TRACE TRAP
(2)          BPTVEC= 14         : BREAKPOINT TRAP (BPT)
(2)          IOTVEC= 20         : INPUT/OUTPUT TRAP (IOT) **SCOPE**
(2)          PWRVEC= 24         : POWER FAIL
(2)          EMTVEC= 30         : EMULATOR TRAP (EMT) **ERROR**
(2)          TRAPVEC=34         : "TRAP" TRAP
(2)          TKVEC= 60          : TTY KEYBOARD VECTOR
(2)          TPVEC= 64          : TTY PRINTER VECTOR
(2)          : ***** THE FOLLOWING BREAK VECTOR AND LINE CLOCK VECTOR ARE INCLUDED
(2)          LKVEC= 100         : LINE CLOCK VECTOR
(2)          BRKVEC= 140        : BREAK VECTOR
(2)          PIRQVEC=240        : PROGRAM INTERRUPT REQUEST VECTOR
  
```



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-3  
 CNDUSA.M11 30-OCT-82 10:55 BASIC DEFINITIONS

SEQ 0017

```

(2)                                     ;STANDARD INTERRUPT VECTORS
(2)
(2)
(2) 000174 000174      . =174
(2) 000174 000000      DISPREG:0
(2) 000176 000000      SWREG:0
(2) 000200 000200      . =200
(2) 000200 000167 001746      JMP      .START      ;GO TO START OF PROGRAM
(2)
(2)
(2) 001100 001100      . =1100
(2) 001100 000000      .WORD 0
(2) 001102 177570      LIGHTS:177570
(2)
(2)
(2)                                     ;PROGRAM CONTROL PARAMETERS
(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) 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

```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-4  
 CNDUSA.M11 30-OCT-82 10:55 BASIC DEFINITIONS

SEQ 0018

```

(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)                                     ;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)                                     ;PROGRAM CONTROL FLAGS
(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)                                     .EVEN
(1)                                     .=1400
(2)
(2)

```



```
(2)                                     ;INSTRUCTION DEFINITIONS
(2)
(2)
(2)
(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    PUSHRO=10046          ;SAVE RO ON STACK =MOV RO,-(SP)
(2)    012600    POPRO=12600           ;RESTORE RO FROM STACK =MOV (SP)+,RO
(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    SYN SCH=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    OVR RUN=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
```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-6  
CNDUSA.M11 30-OCT-82 10:55 BASIC DEFINITIONS

SEQ 0020

|     |        |                        |                   |
|-----|--------|------------------------|-------------------|
| (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 000000 SCMTAG: .=. ::START OF COMMON TAGS
(2) 001402 000 $TSTNM: .BYTE 0 ::CONTAINS THE TEST NUMBER
(2) 001403 000 $SERFLG: .BYTE 0 ::CONTAINS ERROR FLAG
(2) 001404 000000 $SICNT: .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 $SERTTL: .WORD 0 ::CONTAINS TOTAL ERRORS DETECTED
(2) 001414 000 $ITEMB: .BYTE 0 ::CONTAINS ITEM CONTROL BYTE
(2) 001415 001 $SERMAX: .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 $STPFLG: .BYTE 0 ::'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
(2) 001460 000000 $REGAD: .WORD 0 ::CONTAINS THE ADDRESS FROM
(2) ::WHICH ($REGO) 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: .ASCII <207><377><377> ::CODE FOR BELL
(2) 001522 077 $QUES: .ASCII /?/ ::QUESTION MARK

```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-8  
 CNDUSA.M11 30-OCT-82 10:55 COMMON TAGS

SEQ 0022

```

(2) 001523 015 $CRLF: .ASCII <15> ::CARRIAGE RETURN
(2) 001524 000012 $LF: .ASCIIZ <12> ::LINE FEED
(3) *****
(3) $BTTL APT MAILBOX-ETABLE
(3)
(4) *****
(3) .EVEN
(3) $MAIL: ::APT MAILBOX
(3) 001526 000000 $MSGTY: .WORD AMSTY ::MESSAGE TYPE CODE
(3) 001526 000000 $FATAL: .WORD AFATAL ::FATAL ERROR NUMBER
(3) 001530 000000 $TESTN: .WORD ATESTN ::TEST NUMBER
(3) 001532 000000 $PASS: .WORD APASS ::PASS COUNT
(3) 001534 000000 $DEVCT: .WORD ADEVCT ::DEVICE COUNT
(3) 001536 000000 $UNIT: .WORD AUNIT ::I/O UNIT NUMBER
(3) 001540 000000 $MSGAD: .WORD AMSGAD ::MESSAGE ADDRESS
(3) 001542 000000 $MSGLG: .WORD AMSLG ::MESSAGE LENGTH
(3) 001544 000000 $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 $DEVN: .WORD ADEVN ::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

```





```
(4)                                     ;INSTRUCTION DEFINITIONS
(4)
(4)
(4)
(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      PUSHRO=10046          ;SAVE RO ON STACK =MOV RO,-(SP)
(4)    012600      POPRO=12600           ;RESTORE RO FROM STACK =MOV (SP)+,RO
(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      SYN SCH=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
```



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-11  
CNDUSA.M11 30-OCT-82 10:55 APT MAILBOX-ETABLE

SEQ 0025

|     |        |                        |                   |
|-----|--------|------------------------|-------------------|
| (4) | 000040 | DNAINTE=BIT5           | ;DNA INTR ENAB    |
| (4) | 000020 | SEND=BIT4              | ;SEND             |
| (4) | 000010 | HDXEN=BIT3             | ;HDX/FDX          |
| (4) | 000001 | BREAK=BIT0             | ;BREAK            |
| (4) |        | ;TXCSK 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 |

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-12  
 CNDUSA.M11 30-OCT-82 10:55 ERROR POINTER TABLE

SEQ 0026

```

(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) 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)
(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)
(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

```



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-13  
 CNDUSA.M11 30-OCT-82 10:55 ERROR POINTER TABLE

SEQ 0027

```

(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)
(1) 002067 105 051122 041520 DH1: ;DATA HEADERS FOR ERROR MESSAGES
(1) 002074 020040 040527 052116 ;ASCIZ /ERRPC WANTED ACTUAL/
(1) 002102 042105 020040 041501
(1) 002110 052524 046101 000
(1) 002116 001416 001130 001132 DT1: .EVEN ;DATA TABLES FOR ERROR MESSAGES
(1) 002124 000000 ;WORD $ERRPC,HLD0,HLD1,0
(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)
(2) .EVEN
(2) .SBTTL ACT11 HOOKS
(3)
(2) ;*****
(2) ;HOOKS REQUIRED BY ACT11
(2) $SVPC=. ;SAVE PC
(2) .=46
(2) 000046 012660 $ENDAD ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
(2) 000052 000052 .=52
(2) 000000 .WORD 0 ;;2)SET LOC.52 TO ZERO
(2) 002136 .=$SVPC ;; RESTORE PC
(2) .SBTTL APT PARAMETER BLOCK
(3)
(2) ;*****
(2) ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
(3) ;*****
(2) .SX=. ;;SAVE CURRENT LOCATION
(2) 000024 .=24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
(2) 000020 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) ;*****
(2) ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
(2) ;INTERFACE SPEC.
(2)
(2) $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 016304 000020      MOV    $$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
(3) 002200 012737 000300 000022      MOV    #PR6,@#IOTVEC+2 ;;LEVEL 6
(3) 002206 012737 014174 000030      MOV    $$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
(3) 002214 012737 000300 000032      MOV    #PR6,@#EMTVEC+2 ;;LEVEL 6
(3)      ;;BIT02
(3) 002222 012737 016640 000034      MOV    #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
(3) 002230 012737 000300 000036      MOV    #PR6,@#TRAPVEC+2;LEVEL 6
(3) 002236 012737 014776 000024      MOV    #SPWRDN,@#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 014776 000024      MOV     #.PFAIL,@#24      ;;SET UP POWER FAIL VECTOR
  
```



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-15  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0029

|     |        |        |        |        |        |                  |  |                                     |
|-----|--------|--------|--------|--------|--------|------------------|--|-------------------------------------|
| (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 | MOV    | #SWREG,SWR       |  |                                     |
| (2) | 002524 | 012767 | 000174 | 176710 | MOV    | #DISPREG,DISPLAY |  |                                     |
| (2) | 002532 | 022626 |        |        | CMP    | (SP)+,(SP)+      |  |                                     |
| (2) | 002534 | 012637 | 000004 |        | 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 | 010170 |        | JSR    | PC,CNTLU         |  |                                     |
| (2) | 002572 | 105767 | 176374 |        | TSTB   | INIFLG           |  | ;HAS INITIALIZATION BEEN PERFORMED  |
| (2) | 002576 | 001004 |        |        | BNE    | ONCE             |  |                                     |
| (2) | 002600 | 104401 | 015136 |        | TYPE   | ,MTITLE          |  | ;TYPE TITLE MESSAGE                 |
| (2) | 002604 | 105167 | 176362 |        | COMB   | INIFLG           |  | ;IF NOT SET FLAG AND DO             |
| (2) | 002610 | 105767 | 176732 |        | 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 |        | JMP    | .BEGIN           |  | ;GO DO IT                           |
| (2) | 002636 | 032777 | 000001 | 176574 | BIT    | #SW00,@SWR       |  | ;RESELECT VECTOR & CONTROL REG?     |
| (2) | 002644 | 001002 |        |        | BNE    | 1\$              |  |                                     |
| (2) | 002646 | 000167 | 000436 |        | JMP    | .BEGIN           |  |                                     |
| (2) | 002652 | 012700 | 000300 |        | MOV    | #300,R0          |  | ;RESTORE VECTOR AREA TO TRAPCATCHER |
| (2) | 002656 | 0127C1 | 000302 |        | MOV    | #302,R1          |  | ;START AT LOCATION 300              |
| (2) | 002662 | 012702 | 000004 |        | MOV    | #4,R2            |  |                                     |
| (2) | 002666 | 010110 |        |        | 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 | 015204 |        |        | MREGAD |                  |  | ;MESSAGE                            |
| (2) | 002710 | 104410 |        |        | PARAM  |                  |  | ;CONVERT STRING                     |
| (2) | 002712 | 174000 |        |        | 174000 |                  |  | ;LOW LIMIT                          |
| (2) | 002714 | 177776 |        |        | 177776 |                  |  | ;HIGH LIMIT                         |
| (2) | 002716 | 017134 |        |        | DUBASE |                  |  | ;STORE AT THIS LOCATION             |
| (2) | 002720 | 001    |        |        | 1      |                  |  | ;MASK                               |
| (2) | 002721 | 001    |        |        | 1      |                  |  | ;HOW MANY TIMES + 2                 |
| (1) | 002722 | 016767 | 014206 | 176226 | MOV    | DUBASE,KEEPADD   |  | ;SAVE                               |
| (1) | 002730 | 004767 | 014046 |        | JSR    | PC,DUADDR        |  |                                     |



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-16  
 CNDUSA.M11 30-OCT-82 10:55 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 | 015171 |        |        | 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    |        |        | 1       | ;MASK                                                      |
| (2) | 002757 | 004    |        |        | 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 | 015234 |        |        | 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 | 015263 |        |        | 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    |        |        | 1       | ;MASK                                                      |
| (2) | 003043 | 001    |        |        | 1       | ;HOW MANY TIMES + 2                                        |
| (1) |        |        |        |        | 1\$:    | THE FOLLOWING ROUTINE SETS UP ACTREG FOR THE FIRST TIME    |
| (1) | 003044 | 012767 | 000001 | 176116 | 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 | 015357 |        |        | 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    |        |        | 1       | ;MASK                                                      |
| (2) | 003161 | 001    |        |        | 1       | ;HOW MANY TIMES + 2                                        |
| (1) | 003162 | 000167 | 177656 |        | 1\$:    | JMP ;DO IT AGAIN                                           |

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-17  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0031

```

(1) 003166 012767 000300 013602 OUTMUL: MOV #300,DUPRT
(1) 003174 004767 013526 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 015575 MSYNC :MESSAGE
(2) 003204 122767 000061 012724 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 012706 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$: NOP
(2) 003250 104406 INSTR :OUTPUT MESSAGE & GET INPUT STRING
(2) 003252 015643 MWIRE6 :MESSAGE
(2) 003254 104414 SETFLG :SET FLAG BASED UPON INPUT STRING
(2) 003256 001147 SEXMIT :THIS FLAG
(2) 003260 104406 INSTR :OUTPUT MESSAGE & GET INPUT STRING
(2) 003262 015714 MWIRE5 :MESSAGE
(2) 003264 104414 SETFLG :SET FLAG BASED UPON INPUT STRING
(2) 003266 001150 SEREC :THIS FLAG
(2) 003270 104406 INSTR :OUTPUT MESSAGE & GET INPUT STRING
(2) 003272 015764 MWIRE4 :MESSAGE
(2) 003274 104414 SETFLG :SET FLAG BASED UPON INPUT STRING
(2) 003276 001151 OPTCLR :THIS FLAG
(2) 003300 104406 INSTR :OUTPUT MESSAGE & GET INPUT STRING
(2) 003302 016043 MEXTJ :MESSAGE
(2) 003304 104414 SETFLG :SET FLAG BASED UPON INPUT STRING
(2) 003306 001153 JMRBY :THIS FLAG
(2)
(2) :TEST START AND RESTART
(2)
(2) 003310 012706 001100 .BEGIN: MOV #STACK,SP :SET UP STACK
(2) 003314 106427 000300 MTPS #300 :LOCK OUT INTERRUPTS
(2) 003320 032777 000002 176112 BIT #SW01,@SWR :IF SW01=1, GET STARTING PC
(2) 003326 001413 BEQ 3$
(3) 003330 104406 INSTR :OUTPUT MESSAGE & GET INPUT STRING
(3) 003332 015527 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$: MOV #TST1,RETURN :START AT TEST 1
(2) 003364 104401 015523 4$: TYPE ,MR :TYPE R
(2) 003370 000177 175510 JMP @RETURN :START TESTING

```





```

(1)                                     ::TO DETERMINE THAT IT WAS SELECTED CORRECTLY
(1)                                     ::(OVRUN,RXERR)
(1)                                     ::MODE:SYNEXT
(1)                                     ::LENGTH:SEVEN
(1)                                     ::CHAR:0
(1)                                     ::
(5)                                     ::*****
(4) 003606 000004                      TST2: SCOPE
(4)                                     ::
(3) 003610 052777 000400 176110        BIS    #MRESET,@TXCSR ;MASTER RESET
(2) 003616 012777 020000 176076        MOV    #SYNEXT,@PARCSR ;SET THE MODE
(3) 003624 052777 000400 176074        BIS    #MRESET,@TXCSR ;MASTER RESET
(2)                                     ::
(2) 003632 012777 064001 176066        ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)                                     MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)                                     ::
(2) 003640 012777 024000 176054        ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(1) 003646 052777 000020 176036        MOV    #SYNEXT!SEVEN!NOPAR!0,@PARCSR
(2)                                     BIS    #SYNSCH,@RXCSR ;SET SEARCH SYNC
(2) 003654 042777 020000 176044        ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
(2) 003662 052777 020000 176036        BIC    #CLK,@TXCSR ;POKE CLK DOWN
(1) 003670 016703 176022                BIS    #CLK,@TXCSR ;POKE CLK UP
(1) 003674 012700 000000                MOV    RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 003700 012767 000007 175214        MOV    #0,R0 ;EXPECTED
(1) 003706 012767 000000 175564        MOV    #7,SHIFT ;# OF SHIFTS
(1) 003714 004767 013216                MOV    #0,$TMP1 ;DATA CHAR
(1) 003720 105777 175766                JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1) 003724 100401                      TSTB   @RXCSR ;RXDONE ?
(1) 003726 104004                      BMI    +4
(1) 003730 017701 175762                ERROR 4 ;RXDONE SHOULD BE SET
(1) 003734 020001                      MOV    @RXDBUF,R1 ;ACTUAL
(1) 003736 001401                      CMP    R0,R1 ;COMPARE EXPECTED VS. ACTUAL
(1) 003740 104002                      BEQ    +4
(1)                                     ERROR 2 ;RECEIVED DATA DID NOT MATCH
(1)                                     ;EXPECTED DATA - CHECK MAINT DATA
(1)                                     ;OR RECEIVER LOGIC
(1) 003742 012767 000007 175152        MOV    #7,SHIFT ;# OF SHIFTS
(1) 003750 012767 000000 175522        MOV    #0,$TMP1 ;DATA CHAR
(1) 003756 004767 013154                JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1)                                     ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
(1) 003762 012767 000007 175132        MOV    #7,SHIFT ;# OF SHIFTS
(1) 003770 012767 000000 175502        MOV    #0,$TMP1 ;DATA CHAR
(1) 003776 004767 013134                JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1) 004002 012700 140000                MOV    #140000!0,R0 ;EXPECTED DATA PLUS
(1)                                     ;RXERR & OVRUN
(1) 004006 017701 175704                MOV    @RXDBUF,R1 ;ACTUAL
(1) 004012 020001                      CMP    R0,R1 ;COMPARE EXP VS. ACT
(1) 004014 001401                      BEQ    +4
(1) 004016 104002                      ERROR 2 ;SPECIFICALLY LOOK AT RXERR &
(1)                                     ;OVRUN BITS...THEY BOTH SHOULD BE SET
(1)                                     ::
(1)                                     ::THIS TEST VERIFYS WORD LENGTH SELECT OF THE
(1)                                     ::RECEIVER SECTION,IT USES THE ERROR FLAGS
(1)                                     ::TO DETERMINE THAT IT WAS SELECTED CORRECTLY
(1)                                     ::(OVRUN,RXERR)
(1)                                     ::MODE:SYNEXT
  
```

8154



```

(1)                                     ::LENGTH:EIGHT
(1)                                     ::CHAR:125
(1)                                     ::
(5) *****
(4) 004020 000004 TST3: SCOPE
(4)
(3) 004022 052777 000400 175676 BIS #MRESET,@TXCSR ;MASTER RESET
(2) 004030 012777 020000 175664 MOV #SYNEXT,@PARCSR ;SET THE MODE
(3) 004036 052777 000400 175662 BIS #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 004044 012777 064001 175654 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 004052 012777 026000 175642 MOV #SYNEXT!EIGHT!NOPAR!0,@PARCSR
(1) 004060 052777 000020 175624 BIS #SYNSCH,@RXCSR ;SET SEARCH SYNC
(2) ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
(2) 004066 042777 020000 175632 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 004074 052777 020000 175624 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 004102 016703 175610 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 004106 012700 000125 MOV #125,R0 ;EXPECTED
(1) 004112 012767 000010 175002 MOV #8,SHIFT ;# OF SHIFTS
(1) 004120 012767 000125 175352 MOV #125,$TMP1 ;DATA CHAR
(1) 004126 004767 013004 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 004132 105777 175554 TSTB @RXCSR ;RXDONE ?
(1) 004136 100401 BMI +4
(1) 004140 104004 ERROR 4 ;RXDONE SHOULD BE SET
(1) 004142 017701 175550 MOV @RXDBUF,R1 ;ACTUAL
(1) 004146 020001 CMP R0,R1 ;COMPARE EXPECTED VS. ACTUAL
(1) 004150 001401 BEQ +4
(1) 004152 104002 ERROR 2 ;RECEIVED DATA DID NOT MATCH
(1) ;EXPECTED DATA - CHECK MAINT DATA
(1) ;OR RECEIVER LOGIC
(1) 004154 012767 000010 174740 MOV #8,SHIFT ;# OF SHIFTS
(1) 004162 012767 000125 175310 MOV #125,$TMP1 ;DATA CHAR
(1) 004170 004767 012742 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
(1) 004174 012767 000010 174720 MOV #8,SHIFT ;# OF SHIFTS
(1) 004202 012767 000125 175270 MOV #125,$TMP1 ;DATA CHAR
(1) 004210 004767 012722 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 004214 012700 140125 MOV #140000!125,R0 ;EXPECTED DATA PLUS
(1) ;RXERR & OVRRUN
(1) 004220 017701 175472 MOV @RXDBUF,R1 ;ACTUAL
(1) 004224 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
(1) 004226 001401 BEQ +4
(1) 004230 104002 ERROR 2 ;SPECIFICALLY LOOK AT RXERR &
(1) ;OVRRUN BITS...THEY BOTH SHOULD BE SET
(1)
(1)
8155 ;THIS TEST VERIFYS WORD LENGTH SELECT OF THE
(1) ;RECEIVER SECTION,IT USES THE ERROR FLAGS
(1) ;TO DETERMINE THAT IT WAS SELECTED CORRECTLY
(1) ;(OVRRUN,RXERR)
(1) ;MODE:SYNEXT
(1) ;LENGTH:EIGHT
(1) ;CHAR:252
(1) ;

```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-21  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0035

```

(5)
(4) 004232 000004
(4)
(3) 004234 052777 000400 175464 BIS #MRESET,@TXCSR ;MASTER RESET
(2) 004242 012777 020000 175452 MOV #SYNEXT,@PARCSR ;SET THE MODE
(3) 004250 052777 000400 175450 BIS #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 004256 012777 064001 175442 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 004264 012777 026000 175430 MOV #SYNEXT!EIGHT!NOPAR!0,@PARCSR
(1) 004272 052777 000020 175412 BIS #SYNSCH,@RXCSR ;SET SEARCH SYNC
(2) ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
(2) 004300 042777 020000 175420 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 004306 052777 020000 175412 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 004314 016703 175376 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 004320 012700 000252 MOV #252,R0 ;EXPECTED
(1) 004324 012767 000010 174570 MOV #8,SHIFT ;# OF SHIFTS
(1) 004332 012767 000252 175140 MOV #252,$TMP1 ;DATA CHAR
(1) 004340 004767 012572 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 004344 105777 175342 TSTB @RXCSR ;RXDONE ?
(1) 004350 100401 BMI .+4
(1) 004352 104004 ERROR 4 ;RXDONE SHOULD BE SET
(1) 004354 017701 175336 MOV @RXDBUF,R1 ;ACTUAL
(1) 004360 020001 CMP R0,R1 ;COMPARE EXPECTED VS. ACTUAL
(1) 004362 001401 BEQ .+4
(1) 004364 104002 ERROR 2 ;RECEIVED DATA DID NOT MATCH
(1) ;EXPECTED DATA - CHECK MAINT DATA
(1) ;OR RECEIVER LOGIC
(1) 004366 012767 000010 174526 MOV #8,SHIFT ;# OF SHIFTS
(1) 004374 012767 000252 175076 MOV #252,$TMP1 ;DATA CHAR
(1) 004402 004767 012530 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
(1) 004406 012767 000010 174506 MOV #8,SHIFT ;# OF SHIFTS
(1) 004414 012767 000252 175056 MOV #252,$TMP1 ;DATA CHAR
(1) 004422 004767 012510 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 004426 012700 140252 MOV #140000!252,R0 ;EXPECTED DATA PLUS
(1) ;RXERR & OVRRUN
(1) 004432 017701 175260 MOV @RXDBUF,R1 ;ACTUAL
(1) 004436 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
(1) 004440 001401 BEQ .+4
(1) 004442 104002 ERROR 2 ;SPECIFICALLY LOOK AT RXERR &
(1) ;OVRRUN BITS...THEY BOTH SHOULD BE SET
(1)
(1) ;THIS TEST VERIFYS WORD LENGTH SELECT OF THE
(1) ;RECEIVER SECTION,IT USES THE ERROR FLAGS
(1) ;TO DETERMINE THAT IT WAS SELECTED CORRECTLY
(1) ;(OVRRUN,RXERR)
(1) ;MODE:SYNEXT
(1) ;LENGTH:EIGHT
(1) ;CHAR:377
(1)
(5)
(4) 004444 000004
(4)

```



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-22  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0036

```

(3) 004446 052777 000400 175252      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 004454 012777 020000 175240      MOV      #SYNEXT,@PARCSR ;SET THE MODE
(3) 004462 052777 000400 175236      BIS      #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 004470 012777 064001 175230      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 004476 012777 026000 175216      MOV      #SYNEXT!EIGHT!NOPAR!0,@PARCSR
(1) 004504 052777 000020 175200      BIS      #SYNSCH,@RXCSR ;SET SEARCH SYNC
(2) ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
(2) 004512 042777 020000 175206      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 004520 052777 020000 175200      BIS      #CLK,@TXCSR ;POKE CLK UP
(1) 004526 016703 175164              MOV      RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 004532 012700 000377              MOV      #377,R0 ;EXPECTED
(1) 004536 012767 000010 174356      MOV      #8,SHIFT ;# OF SHIFTS
(1) 004544 012767 000377 174726      MOV      #377,$TMP1 ;DATA CHAR
(1) 004552 004767 012360              JSR      PC,RPOKE ;SHIFT IN THIS CHAR
(1) 004556 105777 175130              TSTB     @RXCSR ;RXDONE ?
(1) 004562 100401                    BMI      .+4
(1) 004564 104004                    ERROR    4 ;RXDONE SHOULD BE SET
(1) 004566 017701 175124              MOV      @RXDBUF,R1 ;ACTUAL
(1) 004572 020001                    CMP      R0,R1 ;COMPARE EXPECTED VS. ACTUAL
(1) 004574 001401                    BEQ      .+4
(1) 004576 104002                    ERROR    2 ;RECEIVED DATA DID NOT MATCH
(1) ;EXPECTED DATA - CHECK MAINT DATA
(1) ;OR RECEIVER LOGIC
(1) 004600 012767 000010 174314      MOV      #8,SHIFT ;# OF SHIFTS
(1) 004606 012767 000377 174664      MOV      #377,$TMP1 ;DATA CHAR
(1) 004614 004767 012316              JSR      PC,RPOKE ;SHIFT IN THIS CHAR
(1) ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
(1) 004620 012767 000010 174274      MOV      #8,SHIFT ;# OF SHIFTS
(1) 004626 012767 000377 174644      MOV      #377,$TMP1 ;DATA CHAR
(1) 004634 004767 012276              JSR      PC,RPOKE ;SHIFT IN THIS CHAR
(1) 004640 012700 140377              MOV      #140000!377,R0 ;EXPECTED DATA PLUS
(1) ;RXERR & OVRUN
(1) 004644 017701 175046              MOV      @RXDBUF,R1 ;ACTUAL
(1) 004650 020001                    CMP      R0,R1 ;COMPARE EXP VS. ACT
(1) 004652 001401                    BEQ      .+4
(1) 004654 104002                    ERROR    2 ;SPECIFICALLY LOOK AT RXERR &
(1) ;OVRUN BITS...THEY BOTH SHOULD BE SET
(1)
8157 (1) ;:THIS TEST VERIFYS WORD LENGTH SELECT OF THE
(1) ;:RECEIVER SECTION,IT USES THE ERROR FLAGS
(1) ;:TO DETERMINE THAT IT WAS SELECTED CORRECTLY
(1) ;:(OVRUN,RXERR)
(1) ;:MODE:SYNEXT
(1) ;:LENGTH:EIGHT
(1) ;:CHAR:0
(1) ;:
(5) ;*****
(4) 004656 000004                    TST6:   SCOPE
(4)
(3) 004660 052777 000400 175040      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 004666 012777 020000 175026      MOV      #SYNEXT,@PARCSR ;SET THE MODE
(3) 004674 052777 000400 175024      BIS      #MRESET,@TXCSR ;MASTER RESET

```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-23  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0037

```

(2)
(2) 004702 012777 064001 175016 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 004710 012777 026000 175004 MOV #SYNEXT!EIGHT!NOPAR!0,@PARCSR
(1) 004716 052777 000020 174766 BIS #SYNSCH,@RXCSR ;SET SEARCH SYNC
(2) ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
(2) 004724 042777 020000 174774 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 004732 052777 020000 174766 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 004740 016703 174752 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 004744 012700 000000 MOV #0,R0 ;EXPECTED
(1) 004750 012767 000010 174144 MOV #8,,SHIFT ;# OF SHIFTS
(1) 004756 012767 000000 174514 MOV #0,$TMP1 ;DATA CHAR
(1) 004764 004767 012146 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 004770 105777 174716 TSTB @RXCSR ;RXDONE ?
(1) 004774 100401 BMI .+4
(1) 004776 104004 ERROR 4 ;RXDONE SHOULD BE SET
(1) 005000 017701 174712 MOV @RXDBUF,R1 ;ACTUAL
(1) 005004 020001 CMP R0,R1 ;COMPARE EXPECTED VS. ACTUAL
(1) 005006 001401 BEQ .+4
(1) 005010 104002 ERROR 2 ;RECEIVED DATA DID NOT MATCH
(1) ;EXPECTED DATA - CHECK MAINT DATA
(1) ;OR RECEIVER LOGIC
(1) 005012 012767 000010 174102 MOV #8,,SHIFT ;# OF SHIFTS
(1) 005020 012767 000000 174452 MOV #0,$TMP1 ;DATA CHAR
(1) 005026 004767 012104 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
(1) 005032 012767 000010 174062 MOV #8,,SHIFT ;# OF SHIFTS
(1) 005040 012767 000000 174432 MOV #0,$TMP1 ;DATA CHAR
(1) 005046 004767 012064 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 005052 012700 140000 MOV #140000!0,R0 ;EXPECTED DATA PLUS
(1) ;RXERR & OVRUN
(1) 005056 017701 174634 MOV @RXDBUF,R1 ;ACTUAL
(1) 005062 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
(1) 005064 001401 BEQ .+4
(1) 005066 104002 ERROR 2 ;SPECIFICALLY LOOK AT RXERR &
(1) ;OVRUN BITS...THEY BOTH SHOULD BE SET
(1)
8158 ;:THIS TEST VERIFYS WORD LENGTH SELECT OF RECEIVER
(1) ;:SECTION ,IT USES THE ERROR FLAGS TO DETERMINE
(1) ;:THAT IT WAS SELECTED PROPERLY
(1) ;:FRAME ERROR (FRMERR,RXERR)
(1) ;:MODE:ISOC (ISYMOD)
(1) ;:LENGTH:FIVE
(1) ;:CHAR: 25
(1) ;:
(5) ;*****
(4) 005070 000004 TST7: SCOPE
(4)
(3) 005072 052777 000400 174626 BIS #MRESET,@TXCSR ;MASTER RESET
(2) 005100 012777 000000 174614 MOV #ISYMOD,@PARCSR ;SET THE MODE
(3) 005106 052777 000400 174612 BIS #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 005114 012777 064001 174604 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR

```



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-24  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0038

```

(2)
(2)
(2) 005122 012777 000000 174572 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 005130 052777 000020 174554   MOV    #ISYMOD!FIVE!NOPAR!0,@PARCSR
(2)                                BIS    #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)                                ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 005136 042777 020000 174562   BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 005144 052777 020000 174554   BIS    #CLK,@TXCSR ;POKE CLK UP
(2)                                ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 005152 042777 020000 174546   BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 005160 052777 020000 174540   BIS    #CLK,@TXCSR ;POKE CLK UP
(1) 005166 012767 000007 173726   MOV    #7,SHIFT ;# OF SHIFTS
(1) 005174 012767 000052 174276   MOV    #52,$TMP1 ;DATA CHAR
(1)                                ;NOTE: THE ABOVE CHARACTER IS MISSING STOP BIT
(1) 005202 004767 011730          JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1) 005206 016703 174504          MOV    RXDBUF,R3 ;FOR ERROR MESSAGE
(1) 005212 012700 120025          MOV    #RXERR!FRMERR!25,R0 ;EXPECTED
(1) 005216 017701 174474          MOV    @RXDBUF,R1 ;ACTUAL
(1) 005222 020001          CMP    R0,R1 ;COMPARE EXP VS ACT
(1) 005224 001401          BEQ    +4
(1) 005226 104000          ERROR ;FRAME ERROR 4 & RX ERROR SHOULD BE SET
(1)                                ;IF LOWER BYTE DOES NOT MATCH IT
(1)                                ;PROBABLY IS A LENGTH SELECT PROBLEM
(1)
(1)
(1)
8159
(1)                                ;;THIS TEST VERIFYS WORD LENGTH SELECT OF RECEIVER
(1)                                ;;SECTION ,IT USES THE ERROR FLAGS TO DETERMINE
(1)                                ;;THAT IT WAS SELECTED PROPERLY
(1)                                ;;FRAME ERROR (FRMERR,RXERR)
(1)                                ;;MODE:ISOC (ISYMOD)
(1)                                ;;LENGTH:SIX
(1)                                ;;CHAR: 25
(1)                                ;;
(5)                                ;*****
(4) 005230 000004          TST10: SCOPE
(4)
(3) 005232 052777 000400 174466   BIS    #MRESET,@TXCSR ;MASTER RESET
(2) 005240 012777 000000 174454   MOV    #ISYMOD,@PARCSR ;SET THE MODE
(3) 005246 052777 000400 174452   BIS    #MRESET,@TXCSR ;MASTER RESET
(2)
(2)                                ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 005254 012777 064001 174444   MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)                                ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 005262 012777 002000 174432   MOV    #ISYMOD!SIX!NOPAR!0,@PARCSR
(2) 005270 052777 000020 174414   BIS    #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)                                ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 005276 042777 020000 174422   BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 005304 052777 020000 174414   BIS    #CLK,@TXCSR ;POKE CLK UP
(2)                                ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 005312 042777 020000 174406   BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 005320 052777 020000 174400   BIS    #CLK,@TXCSR ;POKE CLK UP
(1) 005326 012767 000010 173566   MOV    #8,SHIFT ;# OF SHIFTS
(1) 005334 012767 000052 174136   MOV    #52,$TMP1 ;DATA CHAR
(1)                                ;NOTE: THE ABOVE CHARACTER IS MISSING STOP BIT
(1) 005342 004767 011570          JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1) 005346 016703 174344          MOV    RXDBUF,R3 ;FOR ERROR MESSAGE
(1) 005352 012700 120025          MOV    #RXERR!FRMERR!25,R0 ;EXPECTED

```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-25  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0039

```

(1) 005356 017701 174334      MOV    @RXDBUF,R1      ;ACTUAL
(1) 005362 020001              CMP    R0,R1      ;COMPARE EXP VS ACT
(1) 005364 001401              BEQ     +4
(1) 005366 104000              ERROR   ;FRAME ERROR    4 & RX ERROR SHOULD BE SET
                                      ;IF LOWER BYTE DOES NOT MATCH IT
                                      ;PROBABLY IS A LENGTH SELECT PROBLEM

```

8160

```

(1)                                     ::THIS TEST VERIFYS WORD LENGTH SELECT OF RECEIVER
(1)                                     ::SECTION ,IT USES THE ERROR FLAGS TO DETERMINE
(1)                                     ::THAT IT WAS SELECTED PROPERLY
(1)                                     ::FRAME ERROR (FRMERR,RXERR)
(1)                                     ::MODE:ISOC (ISYMOD)
(1)                                     ::LENGTH:SEVEN
(1)                                     ::CHAR: 125
(1)                                     ::

```

```

(5)                                     ::*****
(4) 005370 000004              ;ST11: SCOPE
(4)

```

```

(3) 005372 052777 000400 174326      BIS    #MRESET,@TXCSR ;MASTER RESET
(2) 005400 012777 000000 174314      MOV    #ISYMOD,@PARCSR ;SET THE MODE
(3) 005406 052777 000400 174312      BIS    #MRESET,@TXCSR ;MASTER RESET

```

```

(2)                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 005414 012777 064001 174304      MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)

```

```

(2)                                     ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 005422 012777 004000 174272      MOV    #ISYMOD!SEVEN!NOPAR!0,@PARCSR
(2) 005430 052777 000020 174254      BIS    #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)                                     ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....

```

```

(2) 005436 042777 020000 174262      BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 005444 052777 020000 174254      BIS    #CLK,@TXCSR ;POKE CLK UP
(2)

```

```

(2)                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 005452 042777 020000 174246      BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 005460 052777 020000 174240      BIS    #CLK,@TXCSR ;POKE CLK UP
(1) 005466 012767 000011 173426      MOV    #9,SHIFT ;# OF SHIFTS
(1) 005474 012767 000252 173776      MOV    #252,$TMP1 ;DATA CHAR

```

```

(1)                                     ;NOTE: THE ABOVE CHARACTER IS MISSING STOP BIT
(1) 005502 004767 011430              JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1) 005506 016703 174204              MOV    RXDBUF,R3 ;FOR ERROR MESSAGE
(1) 005512 012700 120125              MOV    #RXERR!FRMERR!125,R0 ;EXPECTED
(1) 005516 017701 174174              MOV    @RXDBUF,R1 ;ACTUAL
(1) 005522 020001              CMP    R0,R1 ;COMPARE EXP VS ACT
(1) 005524 001401              BEQ     +4
(1) 005526 104000              ERROR   ;FRAME ERROR    4 & RX ERROR SHOULD BE SET
                                      ;IF LOWER BYTE DOES NOT MATCH IT
                                      ;PROBABLY IS A LENGTH SELECT PROBLEM

```

8161

```

(1)                                     ::THIS TEST VERIFYS WORD LENGTH SELECT OF RECEIVER
(1)                                     ::SECTION ,IT USES THE ERROR FLAGS TO DETERMINE
(1)                                     ::THAT IT WAS SELECTED PROPERLY
(1)                                     ::FRAME ERROR (FRMERR,RXERR)
(1)                                     ::MODE:ISOC (ISYMOD)
(1)                                     ::LENGTH:EIGHT
(1)                                     ::CHAR: 125
(1)                                     ::

```

```

(5)                                     ::*****

```



```

(4) 005530 000004          TST12: SCOPE
(4)
(3) 005532 052777 000400 174166      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 005540 012777 000000 174154      MOV      #ISYMOD,@PARCSR ;SET THE MODE
(3) 005546 052777 000400 174152      BIS      #MRESET,@TXCSR ;MASTER RESET
(2)
(2)
(2) 005554 012777 064001 174144      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)
(2) 005562 012777 006000 174132      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
      MOV      #ISYMOD!EIGHT!NOPAR!0,@PARCSR
(2) 005570 052777 000020 174114      BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
      ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 005576 042777 020000 174122      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 005604 052777 020000 174114      BIS      #CLK,@TXCSR ;POKE CLK UP
      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 005612 042777 020000 174106      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 005620 052777 020000 174100      BIS      #CLK,@TXCSR ;POKE CLK UP
(1) 005626 012767 000012 173266      MOV      #10,,SHIFT ;# OF SHIFTS
(1) 005634 012767 000252 173636      MOV      #252,$TMP1 ;DATA CHAR
      ;NOTE: THE ABOVE CHARACTER IS MISSING STOP BIT
(1) 005642 004767 011270      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
(1) 005646 016703 174044      MOV      RXDBUF,R3 ;FOR ERROR MESSAGE
(1) 005652 012700 120125      MOV      #RXERR!FMERR!125,R0 ;EXPECTED
(1) 005656 017701 174034      MOV      @RXDBUF,R1 ;ACTUAL
(1) 005662 020001      CMP      R0,R1 ;COMPARE EXP VS ACT
(1) 005664 001401      BEQ      +4
(1) 005666 104000      ERROR ;FRAME ERROR 4 & RX ERROR SHOULD BE SET
      ;IF LOWER BYTE DOES NOT MATCH IT
      ;PROBABLY IS A LENGTH SELECT PROBLEM

8162
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(5)
(4) 005670 000004          ;:THIS TEST VERIFYS EVEPAR PARITY SENSE
      ;:OF THE RECEIVER
      ;:MODE:ISOC (ISYMOD)
      ;:PARITY:EVEPAR
      ;:LENGTH:FIVE PLUS PARITY
      ;:CHAR: 25
      ;:
      ;:*****
TST13: SCOPE
(4)
(3) 005672 052777 000400 174026      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 005700 012777 000000 174014      MOV      #ISYMOD,@PARCSR ;SET THE MODE
(3) 005706 052777 000400 174012      BIS      #MRESET,@TXCSR ;MASTER RESET
(2)
(2)
(2) 005714 012777 064001 174004      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)
(2) 005722 012777 001400 173772      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
      MOV      #ISYMOD!FIVE!EVEPAR!0,@PARCSR
(2) 005730 052777 000020 173754      BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
      ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 005736 042777 020000 173762      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 005744 052777 020000 173754      BIS      #CLK,@TXCSR ;POKE CLK UP
      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 005752 042777 020000 173746      BIC      #CLK,@TXCSR ;POKE CLK DOWN

```

```

(2) 005760 052777 020000 173740 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 005766 016703 173724 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 005772 012700 110025 MOV #RXERR!PARER!25,R0 ;EXPECTED
(1) 005776 012767 000010 173116 MOV #8,SHIFT ;# OF SHIFTS
(1) 006004 012767 000252 173466 MOV #252,$TMP1 ;DATA CHAR
(1) 006012 004767 011120 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 006016 105777 173670 TSTB @RXCSR ;RXDONE ?
(1) 006022 100401 BMI +4
(1) 006024 104004 ERROR 4 ;RXDONE SHOULD BE ASSERTED
(1) 006026 017701 173664 MOV @RXDBUF,R1 ;ACTUAL
(1) 006032 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
(1) 006034 001401 BEQ +4
(1) 006036 104000 ERROR ;PARITY ERROR 4 &RXERR SHOULD BE SET
;NOTE THAT THE PARITY BIT SHOULD
;SHOW UP IN THE DATA
;IE. BIT FIVE FOR FIVE LEVEL CODE

8163
(1) ;:THIS TEST VERIFYS EVEPAR PARITY SENSE
(1) ;:OF THE RECEIVER
(1) ;:MODE:ISOC (ISYMOD)
(1) ;:PARITY:EVEPAR
(1) ;:LENGTH:SIX PLUS PARITY
(1) ;:CHAR: 25
(1) ;:
(5) ;:*****
(4) 006040 000004 TST14: SCOPE
(4)
(3) 006042 052777 000400 173656 BIS #MRESET,@TXCSR ;MASTER RESET
(2) 006050 012777 000000 173644 MOV #ISYMOD,@PARCSR ;SET THE MODE
(3) 006056 052777 000400 173642 BIS #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 006064 012777 064001 173634 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 006072 012777 003400 173622 MOV #ISYMOD!SIX!EVEPAR!0,@PARCSR
(2) 006100 052777 000020 173604 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2) ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 006106 042777 020000 173612 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 006114 052777 020000 173604 BIS #CLK,@TXCSR ;POKE CLK UP
(2) ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 006122 042777 020000 173576 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 006130 052777 020000 173570 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 006136 016703 173554 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 006142 012700 110025 MOV #RXERR!PARER!25,R0 ;EXPECTED
(1) 006146 012767 000011 172746 MOV #9,SHIFT ;# OF SHIFTS
(1) 006154 012767 000452 173316 MOV #452,$TMP1 ;DATA CHAR
(1) 006162 004767 010750 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 006166 105777 173520 TSTB @RXCSR ;RXDONE ?
(1) 006172 100401 BMI +4
(1) 006174 104004 ERROR 4 ;RXDONE SHOULD BE ASSERTED
(1) 006176 017701 173514 MOV @RXDBUF,R1 ;ACTUAL
(1) 006202 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
(1) 006204 001401 BEQ +4
(1) 006206 104000 ERROR ;PARITY ERROR 4 &RXERR SHOULD BE SET
;NOTE THAT THE PARITY BIT SHOULD

```



```

(1)                                ;SHOW UP IN THE DATA
(1)                                ;IE. BIT SIX FOR SIX LEVEL CODE
(1)
8164                                ;:THIS TEST VERIFYS EVEPAR PARITY SENSE
(1)                                ;:OF THE RECEIVER
(1)                                ;:MODE:ISOC (ISYMOD)
(1)                                ;:PARITY:EVEPAR
(1)                                ;:LENGTH:SEVEN PLUS PARITY
(1)                                ;:CHAR: 325
(1)                                ;:
(5)                                ;:*****
(4) 006210 000004                  TST15: SCOPE
(4)
(3) 006212 052777 000400 173506    BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 006220 012777 000000 173474    MOV      #ISYMOD,@PARCSR ;SET THE MODE
(3) 006226 052777 000400 173472    BIS      #MRESET,@TXCSR ;MASTER RESET
(2)
(2)                                ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 006234 012777 064001 173464    MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)                                ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 006242 012777 005400 173452    MOV      #ISYMOD!SEVEN!EVEPAR!0,@PARCSR
(2) 006250 052777 000020 173434    BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)                                ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 006256 042777 020000 173442    BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 006264 052777 020000 173434    BIS      #CLK,@TXCSR ;POKE CLK UP
(2)                                ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 006272 042777 020000 173426    BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 006300 052777 020000 173420    BIS      #CLK,@TXCSR ;POKE CLK UP
(1) 006306 016703 173404            MOV      RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 006312 012700 110325            MOV      #RXERR!PARER!325,R0 ;EXPECTED
(1) 006316 012767 000012 172576    MOV      #10,SHIFT ;# OF SHIFTS
(1) 006324 012767 001652 173146    MOV      #1652,$TMP1 ;DATA CHAR
(1) 006332 004767 010600            JSR      PC,RPOKE ;SHIFT IN THIS CHAR
(1) 006336 105777 173350            TSTB     @RXCSR ;RXDONE ?
(1) 006342 100401                    BMI      +4
(1) 006344 104004                    ERROR    4 ;RXDONE SHOULD BE ASSERTED
(1) 006346 017701 173344            MOV      @RXDBUF,R1 ;ACTUAL
(1) 006352 020001                    CMP      R0,R1 ;COMPARE EXP VS. ACT
(1) 006354 001401                    BEQ      +4
(1) 006356 104000                    ERROR    ;PARITY ERROR 4 &RXERR SHOULD BE SET
(1)                                ;NOTE THAT THE PARITY BIT SHOULD
(1)                                ;SHOW UP IN THE DATA
(1)                                ;IE. BIT SEVEN FOR SEVEN LEVEL CODE
(1)
8165                                ;:THIS TEST VERIFYS EVEPAR PARITY SENSE
(1)                                ;:OF THE RECEIVER
(1)                                ;:MODE:ISOC (ISYMOD)
(1)                                ;:PARITY:EVEPAR
(1)                                ;:LENGTH:EIGHT PLUS PARITY
(1)                                ;:CHAR: 125
(1)                                ;:
(5)                                ;:*****
(4) 006360 000004                  TST16: SCOPE
(4)
(3) 006362 052777 000400 173336    BIS      #MRESET,@TXCSR ;MASTER RESET

```

```

(2) 006370 012777 000000 173324      MOV    #ISYMOD,@PARCSR ;SET THE MODE
(3) 006376 052777 000400 173322      BIS    #MRESET,@TXCSR ;MASTER RESET
(2)
(2)
(2) 006404 012777 064001 173314      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)                                MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) 006412 012777 007400 173302      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2)                                MOV    #ISYMOD!EIGHT!EVEPAR!0,@PARCSR
(2) 006420 052777 000020 173264      BIS    #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)                                ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 006426 042777 020000 173272      BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 006434 052777 020000 173264      BIS    #CLK,@TXCSR ;POKE CLK UP
(2)                                ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 006442 042777 020000 173256      BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 006450 052777 020000 173250      BIS    #CLK,@TXCSR ;POKE CLK UP
(1) 006456 016703 173234              MOV    RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 006462 012700 110125              MOV    #RXERR!PARER!125,R0 ;EXPECTED
(1) 006466 012767 000013 172426      MOV    #11,SHIFT ;# OF SHIFTS
(1) 006474 012767 003252 172776      MOV    #3252,$TMP1 ;DATA CHAR
(1) 006502 004767 010430              JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1) 006506 105777 173200              TSTB   @RXCSR ;RXDONE ?
(1) 006512 100401                    BMI    +4
(1) 006514 104004                    ERROR   4 ;RXDONE SHOULD BE ASSERTED
(1) 006516 017701 173174              MOV    @RXDBUF,R1 ;ACTUAL
(1) 006522 020001                    CMP    R0,R1 ;COMPARE EXP VS. ACT
(1) 006524 001401                    BEQ    +4
(1) 006526 104000                    ERROR   ;PARITY ERROR 4 &RXERR SHOULD BE SET

8166
(1)                                ;:THIS TEST VERIFYS ODDPAR PARITY SENSE
(1)                                ;:OF THE RECEIVER
(1)                                ;:MODE:ISOC (ISYMOD)
(1)                                ;:PARITY:ODDPAR
(1)                                ;:LENGTH:FIVE PLUS PARITY
(1)                                ;:CHAR: 65
(1)                                ;:
(5)                                ;:*****
(4) 006530 000004                    TST17: SCOPE
(4)
(3) 006532 052777 000400 173166      BIS    #MRESET,@TXCSR ;MASTER RESET
(2) 006540 012777 000000 173154      MOV    #ISYMOD,@PARCSR ;SET THE MODE
(3) 006546 052777 000400 173152      BIS    #MRESET,@TXCSR ;MASTER RESET
(2)
(2)
(2) 006554 012777 064001 173144      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)                                MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)
(2) 006562 012777 001000 173132      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2)                                MOV    #ISYMOD!FIVE!ODDPAR!0,@PARCSR
(2) 006570 052777 000020 173114      BIS    #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)                                ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 006576 042777 020000 173122      BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 006604 052777 020000 173114      BIS    #CLK,@TXCSR ;POKE CLK UP
(2)                                ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 006612 042777 020000 173106      BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 006620 052777 020000 173100      BIS    #CLK,@TXCSR ;POKE CLK UP
(1) 006626 016703 173064              MOV    RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 006632 012700 110065              MOV    #RXERR!PARER!65,R0 ;EXPECTED

```



INITIALIZE THE COMMON TAGS

```

(1) 006636 012767 000010 172256 MOV #8,SHIFT ;# OF SHIFTS
(1) 006644 012767 000352 172626 MOV #352,$TMP1 ;DATA CHAR
(1) 006652 004767 010260 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 006656 105777 173030 TSTB @RXCSR ;RXDONE ?
(1) 006662 100401 BMI .+4
(1) 006664 104004 ERROR 4 ;RXDONE SHOULD BE ASSERTED
(1) 006666 017701 173024 MOV @RXDBUF,R1 ;ACTUAL
(1) 006672 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
(1) 006674 001401 BEQ .+4
(1) 006676 104000 ERROR ;PARITY ERROR 4 &RXERR SHOULD BE SET
;NOTE THAT THE PARITY BIT SHOULD
;SHOW UP IN THE DATA
;IE. BIT FIVE FOR FIVE LEVEL CODE

8167 ;:THIS TEST VERIFYS ODDPAR PARITY SENSE
;:OF THE RECEIVER
;:MODE:ISOC (ISYMOD)
;:PARITY:ODDPAR
;:LENGTH:SIX PLUS PARITY
;:CHAR: 125
;:
(5) ;:*****
(4) 006700 000004 TST20: SCOPE
(4)
(3) 006702 052777 000400 173016 BIS #MRESET,@TXCSR ;MASTER RESET
(2) 006710 012777 000000 173004 MOV #ISYMOD,@PARCSR ;SET THE MODE
(3) 006716 052777 000400 173002 BIS #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 006724 012777 064001 172774 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 006732 012777 003000 172762 MOV #ISYMOD!SIX!ODDPAR!0,@PARCSR
(2) 006740 052777 000020 172744 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 006746 042777 020000 172752 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 006754 052777 020000 172744 BIS #CLK,@TXCSR ;POKE CLK UP
(2) ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 006762 042777 020000 172736 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 006770 052777 020000 172730 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 006776 016703 172714 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 007002 012700 110125 MOV #RXERR!PARER!125,R0 ;EXPECTED
(1) 007006 012767 000011 172106 MOV #9,SHIFT ;# OF SHIFTS
(1) 007014 012767 000652 172456 MOV #652,$TMP1 ;DATA CHAR
(1) 007022 004767 010110 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 007026 105777 172660 TSTB @RXCSR ;RXDONE ?
(1) 007032 100401 BMI .+4
(1) 007034 104004 ERROR 4 ;RXDONE SHOULD BE ASSERTED
(1) 007036 017701 172654 MOV @RXDBUF,R1 ;ACTUAL
(1) 007042 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
(1) 007044 001401 BEQ .+4
(1) 007046 104000 ERROR ;PARITY ERROR 4 &RXERR SHOULD BE SET
;NOTE THAT THE PARITY BIT SHOULD
;SHOW UP IN THE DATA
;IE. BIT SIX FOR SIX LEVEL CODE

```

```

8168      ::THIS TEST VERIFYS ODDPAR PARITY SENSE
(1)      ::OF THE RECEIVER
(1)      ::MODE:ISOC (ISYMOD)
(1)      ::PARITY:ODDPAR
(1)      ::LENGTH:SEVEN PLUS PARITY
(1)      ::CHAR: 125
(1)      ::
(5)      ::*****
(4) 007050 000004      TST21: SCOPE
(4)
(3) 007052 052777 000400 172646      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 007060 012777 000000 172634      MOV      #ISYMOD,@PARCSR ;SET THE MODE
(3) 007066 052777 000400 172632      BIS      #MRESET,@TXCSR ;MASTER RESET
(2)
(2)      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 007074 012777 064001 172624      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 007102 012777 005000 172612      MOV      #ISYMOD!SEVEN!ODDPAR!0,@PARCSR
(2) 007110 052777 000020 172574      BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)      ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 007116 042777 020000 172602      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 007124 052777 020000 172574      BIS      #CLK,@TXCSR ;POKE CLK UP
(2)      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 007132 042777 020000 172566      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 007140 052777 020000 172560      BIS      #CLK,@TXCSR ;POKE CLK UP
(1) 007146 016703 172544      MOV      RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 007152 012700 110125      MOV      #RXERR!PARER!125,R0 ;EXPECTED
(1) 007156 012767 000012 171736      MOV      #10,SHIFT ;# OF SHIFTS
(1) 007164 012767 001252 172306      MOV      #125,$TMP1 ;DATA CHAR
(1) 007172 004767 007740      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
(1) 007176 105777 172510      TSTB     @RXCSR ;RXDONE ?
(1) 007202 100401      BMI      .+4
(1) 007204 104004      ERROR     4 ;RXDONE SHOULD BE ASSERTED
(1) 007206 017701 172504      MOV      @RXDBUF,R1 ;ACTUAL
(1) 007212 020001      CMP      R0,R1 ;COMPARE EXP VS. ACT
(1) 007214 001401      BEQ      .+4
(1) 007216 104000      ERROR     ;PARITY ERROR 4 &RXERR SHOULD BE SET
(1)      ;NOTE THAT THE PARITY BIT SHOULD
(1)      ;SHOW UP IN THE DATA
(1)      ;IE. BIT SEVEN FOR SEVEN LEVEL CODE

```

```

8169      ::THIS TEST VERIFYS ODDPAR PARITY SENSE
(1)      ::OF THE RECEIVER
(1)      ::MODE:ISOC (ISYMOD)
(1)      ::PARITY:ODDPAR
(1)      ::LENGTH:EIGHT PLUS PARITY
(1)      ::CHAR: 125
(1)      ::
(5)      ::*****
(4) 007220 000004      TST22: SCOPE
(4)
(3) 007222 052777 000400 172476      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 007230 012777 000000 172464      MOV      #ISYMOD,@PARCSR ;SET THE MODE
(3) 007236 052777 000400 172462      BIS      #MRESET,@TXCSR ;MASTER RESET
(2)

```



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-32  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0046

```

(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 007244 012777 064001 172454 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 007252 012777 007000 172442 MOV #ISYMOD!EIGHT!ODDPAR!0,@PARCSR
(2) 007260 052777 000020 172424 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2) ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 007266 042777 020000 172432 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 007274 052777 020000 172424 BIS #CLK,@TXCSR ;POKE CLK UP
(2) ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 007302 042777 020000 172416 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 007310 052777 020000 172410 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 007316 016703 172374 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
(1) 007322 012700 110125 MOV #RXERR!PARER!125,R0 ;EXPECTED
(1) 007326 012767 000013 171566 MOV #11,SHIFT ;# OF SHIFTS
(1) 007334 012767 002252 172136 MOV #2252,$TMP1 ;DATA CHAR
(1) 007342 004767 007570 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 007346 105777 172340 TSTB @RXCSR ;RXDONE ?
(1) 007352 100401 BMI .+4
(1) 007354 104004 ERROR 4 ;RXDONE SHOULD BE ASSERTED
(1) 007356 017701 172334 MOV @RXDBUF,R1 ;ACTUAL
(1) 007362 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
(1) 007364 001401 BEQ .+4
(1) 007366 104000 ERROR ;PARITY ERROR 4 &RXERR SHOULD BE SET
(1)
8170 ;:THIS TEST PERFORMS BINARY DATA CHECK ON THE
(1) ;:RECEIVER
(1) ;:LENGTH:EIGHT PLUS PARITY
(1) ;:MODE:ISYMOD
(1) ;:PARITY:EVEPAR
(1) ;:
(5) ;:*****
(4) 007370 000004 TST23: SCOPE
(4)
(3) 007372 052777 000400 172326 BIS #MRESET,@TXCSR ;MASTER RESET
(2) 007400 012777 000000 172314 MOV #ISYMOD,@PARCSR ;SET THE MODE
(3) 007406 052777 000400 172312 BIS #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 007414 012777 064001 172304 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 007422 012777 007400 172272 MOV #ISYMOD!EIGHT!EVEPAR!0,@PARCSR
(2) 007430 052777 000020 172254 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2) ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 007436 042777 020000 172262 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 007444 052777 020000 172254 BIS #CLK,@TXCSR ;POKE CLK UP
(2) ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 007452 042777 020000 172246 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 007460 052777 020000 172240 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 007466 016703 172224 MOV RXDBUF,R3 ;SET UP ERROR MESSAGE
(1) 007472 005004 CLR R4 ;DATA CHAR
(1) 007474 010400 1$: MOV R4,R0 ;EXPECTED
(1) 007476 012767 000013 171416 MOV #11,SHIFT ;# OF SHIFTS
(1) 007504 010467 171770 MOV R4,$TMP1 ;"TO BE SHIFTED CHARACTER"
(1) 007510 004767 007574 JSR PC,EVEN8 ;CALC PARITY

```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-33  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0047

```

(1) 007514 000241          CLC
(1) 007516 006167 171756    ROL    $TMP1 ;GENERATE START BIT
(1) 007522 052767 002000 171750  BIS    #BIT10,$TMP1 ;GENERATE STOP BIT
; $TMP1 NOW HAS CHARACTER TO BE POKED INTO RECEIVER
(1) 007530 004767 007402    JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1) 007534 017701 172156    MOV    @RXDBUF,R1 ;ACTUAL
(1) 007540 020001          CMP    R0,R1 ;COMPARE EXP VS ACT
(1) 007542 001401          BEQ    +4
(1) 007544 104002          ERROR    2 ;DATA CHARS SHOULD MATCH
; THERE SHOULD BE NO PARITY ERROR
(1) 007546 005204          INC    R4 ;UPGRADE NEXT CHAR
(1) 007550 105704          TSTB    R4 ;LAST CHAR ?
(1) 007552 001350          BNE    1$

8171 ;:THIS TEST PERFORMS BINARY DATA CHECK ON THE
(1) ;:RECEIVER
(1) ;:LENGTH:EIGHT PLUS PARITY
(1) ;:MODE:ISYMOD
(1) ;:PARITY:ODDPAR
(5) ;:*****
(4) 007554 000004          TST24: SCOPE
(4)
(3) 007556 052777 000400 172142    BIS    #MRESET,@TXCSR ;MASTER RESET
(2) 007564 012777 000000 172130    MOV    #ISYMOD,@PARCSR ;SET THE MODE
(3) 007572 052777 000400 172126    BIS    #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 007600 012777 064001 172120    MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 007606 012777 007000 172106    MOV    #ISYMOD!EIGHT!ODDPAR!0,@PARCSR
(2) 007614 052777 000020 172070    BIS    #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2) ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 007622 042777 020000 172076    BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 007630 052777 020000 172070    BIS    #CLK,@TXCSR ;POKE CLK UP
(2) ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 007636 042777 020000 172062    BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 007644 052777 020000 172054    BIS    #CLK,@TXCSR ;POKE CLK UP
(1) 007652 016703 172040          MOV    RXDBUF,R3 ;SET UP ERROR MESSAGE
(1) 007656 005004          CLR    R4 ;DATA CHAR
(1) 007660 010400          IS: MOV    R4,R0 ;EXPECTED
(1) 007662 012767 000013 171232    MOV    #11,SHIFT ;# OF SHIFTS
(1) 007670 010467 171604          MOV    R4,$TMP1 ;"TO BE SHIFTED CHARACTER"
(1) 007674 004767 007324          JSR    PC,ODD8 ;CALC PARITY
(1) 007700 000241          CLC
(1) 007702 006167 171572    ROL    $TMP1 ;GENERATE START BIT
(1) 007706 052767 002000 171564  BIS    #BIT10,$TMP1 ;GENERATE STOP BIT
; $TMP1 NOW HAS CHARACTER TO BE POKED INTO RECEIVER
(1) 007714 004767 007216    JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1) 007720 017701 171772    MOV    @RXDBUF,R1 ;ACTUAL
(1) 007724 020001          CMP    R0,R1 ;COMPARE EXP VS ACT
(1) 007726 001401          BEQ    +4
(1) 007730 104002          ERROR    2 ;DATA CHARS SHOULD MATCH
; THERE SHOULD BE NO PARITY ERROR
(1) 007732 005204          INC    R4 ;UPGRADE NEXT CHAR

```



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-34  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0048

```

(1) 007734 105704      TSTB   R4      ;LAST CHAR ?
(1) 007736 001350      BNE     1$
(1)
8172      ;:THIS TEST PERFORMS BINARY DATA CHECK ON THE
(1)      ;:RECEIVER
(1)      ;:LENGTH:EIGHT PLUS PARITY
(1)      ;:MODE:SYNEXT
(1)      ;:PARITY:EVEPAR
(1)      ;:
(5)      ;:*****
(4) 007740 000004      TST25: SCOPE
(4)
(3) 007742 052777 000400 171756      BIS     #MRESET,@TXCSR ;MASTER RESET
(2) 007750 012777 020000 171744      MOV     #SYNEXT,@PARCSR ;SET THE MODE
(3) 007756 052777 000400 171742      BIS     #MRESET,@TXCSR ;MASTER RESET
(2)
(2)      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 007764 012777 064001 171734      MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 007772 012777 027400 171722      MOV     #SYNEXT!EIGHT!EVEPAR!0,@PARCSR
(1) 010000 052777 000020 171704      BIS     #SYNSCH,@RXCSR ;SET SEARCH SYNC
(2)      ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
(2) 010006 042777 020000 171712      BIC     #CLK,@TXCSR ;POKE CLK DOWN
(2) 010014 052777 020000 171704      BIS     #CLK,@TXCSR ;POKE CLK UP
(1) 010022 016703 171670      MOV     RXDBUF,R3 ;SET UP ERROR MESSAGE
(1) 010026 005004      CLR     R4 ;DATA CHAR
(1) 010030 010400      1$:      MOV     R4,R0 ;EXPECTED
(1) 010032 012767 000011 171062      MOV     #9,SHIFT ;# OF SHIFTS
(1) 010040 010467 171434      MOV     R4,$TMP1 ;"TO BE SHIFTED CHARACTER"
(1) 010044 004767 007240      JSR     PC,EVEN8 ;CALC PARITY
(1)      ;$TMP1 NOW HAS CHARACTER TO BE POKED INTO RECEIVER
(1) 010050 004767 007062      JSR     PC,RPOKE ;SHIFT IN THIS CHAR
(1) 010054 017701 171636      MOV     @RXDBUF,R1 ;ACTUAL
(1) 010060 020001      CMP     R0,R1 ;COMPARE EXP VS ACT
(1) 010062 001401      BEQ     +4
(1) 010064 104002      ERROR  2 ;DATA CHARS SHOULD MATCH
(1)      ;THERE SHOULD BE NO PARITY ERROR
(1) 010066 005204      INC     R4 ;UPGRADE NEXT CHAR
(1) 010070 105704      TSTB   R4 ;LAST CHAR ?
(1) 010072 001356      BNE     1$
(1)
8173      ;:THIS TEST PERFORMS BINARY DATA CHECK ON THE
(1)      ;:RECEIVER
(1)      ;:LENGTH:EIGHT PLUS PARITY
(1)      ;:MODE:SYNEXT
(1)      ;:PARITY:ODDPAR
(1)      ;:
(5)      ;:*****
(4) 010074 000004      TST26: SCOPE
(4)
(3) 010076 052777 000400 171622      BIS     #MRESET,@TXCSR ;MASTER RESET
(2) 010104 012777 020000 171610      MOV     #SYNEXT,@PARCSR ;SET THE MODE
(3) 010112 052777 000400 171606      BIS     #MRESET,@TXCSR ;MASTER RESET
(2)
(2)      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE

```

```

(2) 010120 012777 064001 171600      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)                                     ;SET MODE ,# OF BITS,PARTY SENSE,&LOAD SYNC REG
(2) 010126 012777 027000 171566      MOV      #SYNEXT!EIGHT!ODDPAR!0,@PARCSR
(1) 010134 052777 000020 171550      BIS      #SYNSCH,@RXCSR ;SET SEARCH SYNC
(2)                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 010142 042777 020000 171556      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 010150 052777 020000 171550      BIS      #CLK,@TXCSR ;POKE CLK UP
(1) 010156 016703 171534              MOV      RXDBUF,R3 ;SET UP ERROR MESSAGE
(1) 010162 005004                    CLR      R4 ;DATA CHAR
(1) 010164 010400                    1$:     MOV      R4,R0 ;EXPECTED
(1) 010166 012767 000011 170726      MOV      #9,,SHIFT ;# OF SHIFTS
(1) 010174 010467 171300              MOV      R4,$TMP1 ;'TO BE SHIFTED CHARACTER''
(1) 010200 004767 007020              JSR      PC,ODD8 ;CALC PARITY
(1)                                     ;$TMP1 NOW HAS CHARACTER TO BE POKED INTO RECEIVER
(1) 010204 004767 006726              JSR      PC,RPOKE ;SHIFT IN THIS CHAR
(1) 010210 017701 171502              MOV      @RXDBUF,R1 ;ACTUAL
(1) 010214 020001                    CMP      R0,R1 ;COMPARE EXP VS ACT
(1) 010216 001401                    BEQ      +4
(1) 010220 104002                    ERROR    2 ;DATA CHARS SHOULD MATCH
(1)                                     ;THERE SHOULD BE NO PARITY ERROR
(1) 010222 005204                    INC      R4 ;UPGRADE NEXT CHAR
(1) 010224 105704                    TSTB     R4 ;LAST CHAR ?
(1) 010226 001356                    BNE      1$
(1)                                     ;;THIS TEST CHECKS THE STRIP SYNC FUNCTION
(1)                                     ;;OF THE RECEIVER LOGIC
(1)                                     ;;MODE:ISYMOD
(1)                                     ;;LENGTH:FIVE
(1)                                     ;;NOTE: RXDONE SHOULD NEVER ASSERT
(1)                                     ;;CHAR: 26 (SYNC)
(1)                                     ;;
(5) *****
(4) 010230 000004                    TST27: SCOPE
(4)                                     BIS      #MRESET,@TXCSR ;MASTER RESET
(3) 010232 052777 000400 171466      MOV      #ISYMOD,@PARCSR ;SET THE MODE
(2) 010240 012777 000000 171454      BIS      #MRESET,@TXCSR ;MASTER RESET
(3) 010246 052777 000400 171452
(2)                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 010254 012777 064001 171444      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)                                     ;SET MODE ,# OF BITS,PARTY SENSE,&LOAD SYNC REG
(2) 010262 012777 000026 171432      MOV      #ISYMOD!FIVE!NOPAR!26,@PARCSR
(2) 010270 052777 000020 171414      BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)                                     ;POKE CLK TO GET RECEIVER INTO SYNCRONIZATION....
(2) 010276 042777 020000 171422      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 010304 052777 020000 171414      BIS      #CLK,@TXCSR ;POKE CLK UP
(2)                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 010312 042777 020000 171406      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 010320 052777 020000 171400      BIS      #CLK,@TXCSR ;POKE CLK UP
(1) 010326 052777 000400 171356      BIS      #STPSYN,@RXCSR ;SET STRIP SYNC
(1) 010334 012767 000003 170562      MOV      #3,COUNT ;# OF SYNC CHARS
(1) 010342 012767 000154 171130      1$:     MOV      #154,$TMP1 ;CHAR TO BE SHIFTED
(1) 010350 012767 000007 170544      MOV      #7,SHIFT ;# OF SHIFTS
(1) 010356 004767 006554              JSR      PC,RPOKE ;SHIFT IN THIS CHAR

```



```

(1) 010362 105777 171324      TSTB    @RXCSR    ;RXDONE ?
(1) 010366 100001              BPL      .+4
(1) 010370 104004              ERROR    4          ;RXDONE SHOULD NOT BE ASSERTED
(1) 010372 005367 170526      DEC      COUNT      ;# OF SYNC CHARS
(1) 010376 001361              BNE      1$
(1)
8175
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(5)
(4) 010400 000004              *****
TST30: SCOPE
(4)
(3) 010402 052777 000400 171316      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 010410 012777 000000 171304      MOV      #ISYMOD,@PARCSR ;SET THE MODE
(3) 010416 052777 000400 171302      BIS      #MRESET,@TXCSR ;MASTER RESET
(2)
(2)
(2) 010424 012777 064001 171274      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2)
(2)
(2)
(2) 010432 012777 002026 171262      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 010440 052777 000020 171244      MOV      #ISYMOD!SIX!NOPAR!26,@PARCSR
(2)
(2)
(2) 010446 042777 020000 171252      BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2) 010454 052777 020000 171244      ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2)
(2)
(2) 010462 042777 020000 171236      BIC      #CLK,@TXCSR    ;POKE CLK DOWN
(2) 010470 052777 020000 171230      BIS      #CLK,@TXCSR    ;POKE CLK UP
(1) 010476 052777 000400 171206      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(1) 010504 012767 000003 170412      BIC      #CLK,@TXCSR    ;POKE CLK DOWN
(1) 010512 012767 000254 170760      BIS      #CLK,@TXCSR    ;POKE CLK UP
(1) 010520 012767 000010 170374      BIS      #STPSYN,@RXCSR ;SET STRIP SYNC
(1) 010526 004767 006404      MOV      #3,COUNT      ;# OF SYNC CHARS
(1) 010532 105777 171154      1$: MOV      #254,$TMP1    ;CHAR TO BE SHIFTED
(1) 010536 100001              MOV      #8,SHIFT      ;# OF SHIFTS
(1) 010540 104004              JSR      PC,RPOKE      ;SHIFT IN THIS CHAR
(1) 010542 005367 170356      TSTB    @RXCSR    ;RXDONE ?
(1) 010546 001361              BPL      .+4
(1)
8176
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(1)
(5)
(4) 010550 000004              *****
TST31: SCOPE
(4)
(3) 010552 052777 000400 171146      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 010560 012777 000000 171134      MOV      #ISYMOD,@PARCSR ;SET THE MODE
(3) 010566 052777 000400 171132      BIS      #MRESET,@TXCSR ;MASTER RESET

```

CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-37  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0051

```

(2)
(2) 010574 012777 064001 171124 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 010602 012777 004026 171112 MOV #ISYMOD!SEVEN!NOPAR!26,@PARCSR
(2) 010610 052777 000020 171074 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2) ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 010616 042777 020000 171102 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 010624 052777 020000 171074 BIS #CLK,@TXCSR ;POKE CLK UP
(2) ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 010632 042777 020000 171066 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 010640 052777 020000 171060 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 010646 052777 000400 171036 BIS #STPSYN,@RXCSR ;SET STRIP SYNC
(1) 010654 012767 000003 170242 MOV #3,COUNT ;# OF SYNC CHARS
(1) 010662 012767 000454 170610 1$: MOV #454,$TMP1 ;CHAR TO BE SHIFTED
(1) 010670 012767 000011 170224 MOV #9,$SHIFT ;# OF SHIFTS
(1) 010676 004767 006234 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 010702 105777 171004 TSTB @RXCSR ;RXDONE ?
(1) 010706 100001 BPL +4
(1) 010710 104004 ERROR 4 ;RXDONE SHOULD NOT BE ASSERTED
(1) 010712 005367 170206 DEC COUNT ;# OF SYNC CHARS
(1) 010716 001361 BNE 1$
(1)
8177 ;:THIS TEST CHECKS THE STRIP SYNC FUNCTION
(1) ;:OF THE RECEIVER LOGIC
(1) ;:MODE:ISYMOD
(1) ;:LENGTH:EIGHT
(1) ;:NOTE: RXDONE SHOULD NEVER ASSERT
(1) ;:CHAR: 26 (SYNC)
(1) ;:
(5) ;:*****
(4) 010720 000004 TST32: SCOPE
(4)
(3) 010722 052777 000400 170776 BIS #MRESET,@TXCSR ;MASTER RESET
(2) 010730 012777 000000 170764 MOV #ISYMOD,@PARCSR ;SET THE MODE
(3) 010736 052777 000400 170762 BIS #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 010744 012777 064001 170754 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 010752 012777 006026 170742 MOV #ISYMOD!EIGHT!NOPAR!26,@PARCSR
(2) 010760 052777 000020 170724 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2) ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 010766 042777 020000 170732 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 010774 052777 020000 170724 BIS #CLK,@TXCSR ;POKE CLK UP
(2) ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 011002 042777 020000 170716 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 011010 052777 020000 170710 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 011016 052777 000400 170666 BIS #STPSYN,@RXCSR ;SET STRIP SYNC
(1) 011024 012767 000003 170072 MOV #3,COUNT ;# OF SYNC CHARS
(1) 011032 012767 001054 170440 1$: MOV #1054,$TMP1 ;CHAR TO BE SHIFTED
(1) 011040 012767 000012 170054 MOV #10,$SHIFT ;# OF SHIFTS
(1) 011046 004767 006064 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 011052 105777 170634 TSTB @RXCSR ;RXDONE ?

```



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 61-38  
 CNDUSA.M11 30-OCT-82 10:55 INITIALIZE THE COMMON TAGS

SEQ 0052

```

(1) 011056 100001      BPL      +4
(1) 011060 104004      ERROR    4      ;RXDONE SHOULD NOT BE ASSERTED
(1) 011062 005367 170036 DEC     COUNT ;# OF SYNC CHARS
(1) 011066 001361      BNE      1$

8178      ;:THIS TEST CHECKS THE STRIP SYNC FUNCTION
(1)      ;:OF THE RECEIVER LOGIC
(1)      ;:MODE:SYNEXT
(1)      ;:LENGTH:FIVE
(1)      ;:NOTE: RXDONE SHOULD NEVER ASSERT
(1)      ;:CHAR: 26 (SYNC)
(1)      ;:
(5)      ;:*****
(4) 011070 000004      TST33: SCOPE
(4)
(3) 011072 052777 000400 170626      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 011100 012777 020000 170614      MOV      #SYNEXT,@PARCSR ;SET THE MODE
(3) 011106 052777 000400 170612      BIS      #MRESET,@TXCSR ;MASTER RESET
(2)
(2)      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 011114 012777 064001 170604      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 011122 012777 020026 170572      MOV      #SYNEXT!FIVE!NOPAR!26,@PARCSR
(1) 011130 052777 000020 170554      BIS      #SYNSCH,@RXCSR ;SET SEARCH SYNC
(2)      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 011136 042777 020000 170562      BIC      #CLK,@TXCSR ;POKE CLK DOWN
(2) 011144 052777 020000 170554      BIS      #CLK,@TXCSR ;POKE CLK UP
(1) 011152 052777 000400 170532      BIS      #STPSYN,@RXCSR ;SET STRIP SYNC
(1) 011160 012767 000003 167736      MOV      #3,COUNT ;# OF SYNC CHARS
(1) 011166 012767 000026 170304      1$: MOV      #26,$TMP1 ;CHAR TO BE SHIFTED
(1) 011174 012767 000005 167720      MOV      #5,SHIFT ;# OF SHIFTS
(1) 011202 004767 005730      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
(1) 011206 105777 170500      TSTB     @RXCSR ;RXDONE ?
(1) 011212 100001      BPL      +4
(1) 011214 104004      ERROR    4      ;RXDONE SHOULD NOT BE ASSERTED
(1) 011216 005367 167702      DEC     COUNT ;# OF SYNC CHARS
(1) 011222 001361      BNE      1$

8179      ;:THIS TEST CHECKS THE STRIP SYNC FUNCTION
(1)      ;:OF THE RECEIVER LOGIC
(1)      ;:MODE:SYNEXT
(1)      ;:LENGTH:SIX
(1)      ;:NOTE: RXDONE SHOULD NEVER ASSERT
(1)      ;:CHAR: 26 (SYNC)
(1)      ;:
(5)      ;:*****
(4) 011224 000004      TST34: SCOPE
(4)
(3) 011226 052777 000400 170472      BIS      #MRESET,@TXCSR ;MASTER RESET
(2) 011234 012777 020000 170460      MOV      #SYNEXT,@PARCSR ;SET THE MODE
(3) 011242 052777 000400 170456      BIS      #MRESET,@TXCSR ;MASTER RESET
(2)
(2)      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 011250 012777 064001 170450      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)

```

INITIALIZE THE COMMON TAGS

| Address | Hex Data | Hex Data | Hex Data | Hex Data | Assembly Code                                    |
|---------|----------|----------|----------|----------|--------------------------------------------------|
| (2)     | 011256   | 012777   | 022026   | 170436   | ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG |
| (2)     | 011264   | 052777   | 000020   | 170420   | MOV #SYNEXT!SIX!NOPAR!26,@PARCSR                 |
| (1)     | 011264   | 052777   | 000020   | 170420   | BIS #SYNSCH,@RXCSR ;SET SEARCH SYNC              |
| (2)     | 011272   | 042777   | 020000   | 170426   | ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION      |
| (2)     | 011300   | 052777   | 020000   | 170420   | BIC #CLK,@TXCSR ;POKE CLK DOWN                   |
| (1)     | 011306   | 052777   | 000400   | 170376   | BIS #CLK,@TXCSR ;POKE CLK UP                     |
| (1)     | 011314   | 012767   | 000003   | 167602   | BIS #STPSYN,@RXCSR ;SET STRIP SYNC               |
| (1)     | 011322   | 012767   | 000026   | 170150   | MOV #3,COUNT ;# OF SYNC CHARS                    |
| (1)     | 011330   | 012767   | 000006   | 167564   | 1\$: MOV #26,\$TMP1 ;CHAR TO BE SHIFTED          |
| (1)     | 011336   | 004767   | 005574   |          | MOV #6,SHIFT ;# OF SHIFTS                        |
| (1)     | 011342   | 105777   | 170344   |          | JSR PC,RPOKE ;SHIFT IN THIS CHAR                 |
| (1)     | 011346   | 100001   |          |          | TSTB @RXCSR ;RXDONE ?                            |
| (1)     | 011350   | 104004   |          |          | BPL .+4                                          |
| (1)     | 011352   | 005367   | 167546   |          | ERROR 4 ;RXDONE SHOULD NOT BE ASSERTED           |
| (1)     | 011356   | 001361   |          |          | DEC COUNT ;# OF SYNC CHARS                       |
| (1)     |          |          |          |          | BNE 1\$                                          |
| 8180    |          |          |          |          | ::THIS TEST CHECKS THE STRIP SYNC FUNCTION       |
| (1)     |          |          |          |          | ::OF THE RECEIVER LOGIC                          |
| (1)     |          |          |          |          | ::MODE:SYNEXT                                    |
| (1)     |          |          |          |          | ::LENGTH:SEVEN                                   |
| (1)     |          |          |          |          | ::NOTE: RXDONE SHOULD NEVER ASSERT               |
| (1)     |          |          |          |          | ::CHAR: 26 (SYNC)                                |
| (1)     |          |          |          |          | ::                                               |
| (5)     |          |          |          |          | ::*****                                          |
| (4)     | 011360   | 000004   |          |          | TST35: SCOPE                                     |
| (4)     |          |          |          |          |                                                  |
| (3)     | 011362   | 052777   | 000400   | 170336   | BIS #MRESET,@TXCSR ;MASTER RESET                 |
| (2)     | 011370   | 012777   | 020000   | 170324   | MOV #SYNEXT,@PARCSR ;SET THE MODE                |
| (3)     | 011376   | 052777   | 000400   | 170322   | BIS #MRESET,@TXCSR ;MASTER RESET                 |
| (2)     |          |          |          |          |                                                  |
| (2)     | 011404   | 012777   | 064001   | 170314   | ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE      |
| (2)     |          |          |          |          | MOV #MTDATA!CLK!MINT!BREAK,@TXCSR                |
| (2)     |          |          |          |          |                                                  |
| (2)     | 011412   | 012777   | 024026   | 170302   | ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG |
| (1)     | 011420   | 052777   | 000020   | 170264   | MOV #SYNEXT!SEVEN!NOPAR!26,@PARCSR               |
| (2)     |          |          |          |          | BIS #SYNSCH,@RXCSR ;SET SEARCH SYNC              |
| (2)     | 011426   | 042777   | 020000   | 170272   | ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION      |
| (2)     | 011434   | 052777   | 020000   | 170264   | BIC #CLK,@TXCSR ;POKE CLK DOWN                   |
| (1)     | 011442   | 052777   | 000400   | 170242   | BIS #CLK,@TXCSR ;POKE CLK UP                     |
| (1)     | 011450   | 012767   | 000003   | 167446   | BIS #STPSYN,@RXCSR ;SET STRIP SYNC               |
| (1)     | 011456   | 012767   | 000026   | 170014   | MOV #3,COUNT ;# OF SYNC CHARS                    |
| (1)     | 011464   | 012767   | 000007   | 167430   | 1\$: MOV #26,\$TMP1 ;CHAR TO BE SHIFTED          |
| (1)     | 011472   | 004767   | 005440   |          | MOV #7,SHIFT ;# OF SHIFTS                        |
| (1)     | 011476   | 105777   | 170210   |          | JSR PC,RPOKE ;SHIFT IN THIS CHAR                 |
| (1)     | 011502   | 100001   |          |          | TSTB @RXCSR ;RXDONE ?                            |
| (1)     | 011504   | 104004   |          |          | BPL .+4                                          |
| (1)     | 011506   | 005367   | 167412   |          | ERROR 4 ;RXDONE SHOULD NOT BE ASSERTED           |
| (1)     | 011512   | 001361   |          |          | DEC COUNT ;# OF SYNC CHARS                       |
| (1)     |          |          |          |          | BNE 1\$                                          |
| 8181    |          |          |          |          | ::THIS TEST CHECKS THE STRIP SYNC FUNCTION       |
| (1)     |          |          |          |          | ::OF THE RECEIVER LOGIC                          |
| (1)     |          |          |          |          | ::MODE:SYNEXT                                    |
| (1)     |          |          |          |          | ::LENGTH:EIGHT                                   |
| (1)     |          |          |          |          | ::NOTE: RXDONE SHOULD NEVER ASSERT               |



```

(1)                                     ;;CHAR: 26 (SYNC)
(1)                                     ;;
(5)                                     ;*****
(4) 011514 000004                      TST36: SCOPE
(4)                                     ;;
(3) 011516 052777 000400 170202        BIS    #MRESET,@TXCSR ;MASTER RESET
(2) 011524 012777 020000 170170        MOV    #SYNEXT,@PARCSR ;SET THE MODE
(3) 011532 052777 000400 170166        BIS    #MRESET,@TXCSR ;MASTER RESET
(2)                                     ;;
(2)                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 011540 012777 064001 170160        MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)                                     ;;
(2)                                     ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 011546 012777 026026 170146        MOV    #SYNEXT!EIGHT!NOPAR!26,@PARCSR
(1) 011554 052777 000020 170130        BIS    #SYNSCH,@RXCSR ;SET SEARCH SYNC
(2)                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 011562 042777 020000 170136        BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 011570 052777 020000 170130        BIS    #CLK,@TXCSR ;POKE CLK UP
(1) 011576 052777 000400 170106        BIS    #STPSYN,@RXCSR ;SET STRIP SYNC
(1) 011604 012767 000003 167312        MOV    #3,COUNT ;# OF SYNC CHARS
(1) 011612 012767 000026 167660        1$: MOV    #26,$TMP1 ;CHAR TO BE SHIFTED
(1) 011620 012767 000010 167274        MOV    #8.,SHIFT ;# OF SHIFTS
(1) 011626 004767 005304                JSR    PC,RPOKE ;SHIFT IN THIS CHAR
(1) 011632 105777 170054                TSTB   @RXCSR ;RXDONE ?
(1) 011636 100001                BPL     .+4
(1) 011640 104004                ERROR   4 ;RXDONE SHOULD NOT BE ASSERTED
(1) 011642 005367 167256                DEC    COUNT ;# OF SYNC CHARS
(1) 011646 001361                BNE     1$
(1)                                     ;;
8182                                     ;THIS TEST CHECKS THE STRIP SYNC FUNCTION
(1)                                     ;OF THE RECEIVER LOGIC
(1)                                     ;MODE:SYNINT
(1)                                     ;LENGTH:FIVE
(1)                                     ;NOTE: RXDONE SHOULD NEVER ASSERT
(1)                                     ;CHAR: 26 (SYNC)
(1)                                     ;;
(5)                                     ;*****
(4) 011650 000004                      TST37: SCOPE
(4)                                     ;;
(3) 011652 052777 000400 170046        BIS    #MRESET,@TXCSR ;MASTER RESET
(2) 011660 012777 030000 170034        MOV    #SYNINT,@PARCSR ;SET THE MODE
(3) 011666 052777 000400 170032        BIS    #MRESET,@TXCSR ;MASTER RESET
(2)                                     ;;
(2)                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 011674 012777 064001 170024        MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)                                     ;;
(2)                                     ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 011702 012777 030026 170012        MOV    #SYNINT!FIVE!NOPAR!26,@PARCSR
(2) 011710 052777 000020 167774        BIS    #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)                                     ;POKE CLK TO GET RECEIVER INTO SYNCRONIZATION....
(2) 011716 042777 020000 170002        BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 011724 052777 020000 167774        BIS    #CLK,@TXCSR ;POKE CLK UP
(2)                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 011732 042777 020000 167766        BIC    #CLK,@TXCSR ;POKE CLK DOWN
(2) 011740 052777 020000 167760        BIS    #CLK,@TXCSR ;POKE CLK UP
(1) 011746 052777 000400 167736        BIS    #STPSYN,@RXCSR ;SET STRIP SYNC

```

```

(1) 011754 012767 000003 167142      MOV    #3,COUNT      ;# OF SYNC CHARS
(1) 011762 012767 000026 167510 1$:  MOV    #26,$TMP1    ;CHAR TO BE SHIFTED
(1) 011770 012767 000005 167124      MOV    #5,SHIFT     ;# OF SHIFTS
(1) 011776 004767 005134              JSR     PC,RPOKE     ;SHIFT IN THIS CHAR
(1) 012002 105777 167704              TSTB   @RXCSR ;RXDONE ?
(1) 012006 100001              BPL     .+4
(1) 012010 104004              ERROR   4 ;RXDONE SHOULD NOT BE ASSERTED
(1) 012012 005367 167106              DEC    COUNT ;# OF SYNC CHARS
(1) 012016 001361              BNE     1$

(1)
8183 (1)
(1)
(1)
(1)
(1)
(1)
(1)
(5)
(4) 012020 000004      TST40: SCOPE
(4)
(3) 012022 052777 000400 167676      BIS     #MRESET,@TXCSR ;MASTER RESET
(2) 012030 012777 030000 167664      MOV     #SYNINT,@PARCSR ;SET THE MODE
(3) 012036 052777 000400 167662      BIS     #MRESET,@TXCSR ;MASTER RESET
(2)
(2)
(2) 012044 012777 064001 167654      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
      MOV     #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2)
(2) 012052 012777 032026 167642      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
      MOV     #SYNINT!SIX!NOPAR!26,@PARCSR
(2) 012060 052777 000020 167624      BIS     #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2)
      ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 012066 042777 020000 167632      BIC     #CLK,@TXCSR ;POKE CLK DOWN
(2) 012074 052777 020000 167624      BIS     #CLK,@TXCSR ;POKE CLK UP
(2)
      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 012102 042777 020000 167616      BIC     #CLK,@TXCSR ;POKE CLK DOWN
(2) 012110 052777 020000 167610      BIS     #CLK,@TXCSR ;POKE CLK UP
(1) 012116 052777 000400 167566      BIS     #STPSYN,@RXCSR ;SET STRIP SYNC
(1) 012124 012767 000003 166772      MOV     #3,COUNT      ;# OF SYNC CHARS
(1) 012132 012767 000026 167340 1$:  MOV     #26,$TMP1    ;CHAR TO BE SHIFTED
(1) 012140 012767 000006 166754      MOV     #6,SHIFT     ;# OF SHIFTS
(1) 012146 004767 004764              JSR     PC,RPOKE     ;SHIFT IN THIS CHAR
(1) 012152 105777 167534              TSTB   @RXCSR ;RXDONE ?
(1) 012156 100001              BPL     .+4
(1) 012160 104004              ERROR   4 ;RXDONE SHOULD NOT BE ASSERTED
(1) 012162 005367 166736              DEC    COUNT ;# OF SYNC CHARS
(1) 012166 001361              BNE     1$

(1)
8184 (1)
(1)
(1)
(1)
(1)
(1)
(5)
(4) 012170 000004      TST41: SCOPE
  
```

::THIS TEST CHECKS THE STRIP SYNC FUNCTION  
 ::OF THE RECEIVER LOGIC  
 ::MODE:SYNINT  
 ::LENGTH:SIX  
 ::NOTE: RXDONE SHOULD NEVER ASSERT  
 ::CHAR: 26 (SYNC)

\*\*\*\*\*  
 TST40: SCOPE

;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE  
 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG  
 ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....

;POKE CLK TO GET LOGIC INTO SYNCRONIZATION  
 1\$:

::THIS TEST CHECKS THE STRIP SYNC FUNCTION  
 ::OF THE RECEIVER LOGIC  
 ::MODE:SYNINT  
 ::LENGTH:SEVEN  
 ::NOTE: RXDONE SHOULD NEVER ASSERT  
 ::CHAR: 26 (SYNC)

\*\*\*\*\*  
 TST41: SCOPE



```

(4) 012172 052777 000400 167526 BIS #MRESET,@TXCSR ;MASTER RESET
(3) 012200 012777 030000 167514 MOV #SYNINT,@PARCSR ;SET THE MODE
(3) 012206 052777 000400 167512 BIS #MRESET,@TXCSR ;MASTER RESET
(2)
(2) ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
(2) 012214 012777 064001 167504 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
(2)
(2) ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
(2) 012222 012777 034026 167472 MOV #SYNINT!SEVEN!NOPAR!26,@PARCSR
(2) 012230 052777 000020 167454 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
(2) ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
(2) 012236 042777 020000 167462 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 012244 052777 020000 167454 BIS #CLK,@TXCSR ;POKE CLK UP
(2) ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
(2) 012252 042777 020000 167446 BIC #CLK,@TXCSR ;POKE CLK DOWN
(2) 012260 052777 020000 167440 BIS #CLK,@TXCSR ;POKE CLK UP
(1) 012266 052777 000400 167416 BIS #STPSYN,@RXCSR ;SET STRIP SYNC
(1) 012274 012767 000003 166622 MOV #3,COUNT ;# OF SYNC CHARS
(1) 012302 012767 000026 167170 1$: MOV #26,$TMP1 ;CHAR TO BE SHIFTED
(1) 012310 012767 000007 166604 MOV #7,SHIFT ;# OF SHIFTS
(1) 012316 004767 004614 JSR PC,RPOKE ;SHIFT IN THIS CHAR
(1) 012322 105777 167364 TSTB @RXCSR ;RXDONE ?
(1) 012326 100001 BPL ;+4
(1) 012330 104004 ERROR 4 ;RXDONE SHOULD NOT BE ASSERTED
(1) 012332 005367 166566 DEC COUNT ;# OF SYNC CHARS
(1) 012336 001361 BNE 1$
(1)
8185
(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) 012340 000004 .EOP: SCOPE
(2) 012342 004767 000340 JSR PC,CKSWR
(2) 012346 104401 TYPE ;TYPE NAME OF TEST
(2) 012350 015476 MEPASS
(2) 012352 104413 012604 CONVRT ,OUTCRY
(2) 012356 104401 015315 TYPE ,DEVICE
(2) 012362 105767 166564 TSTB MULTD ;ARE YOU RUNNING MULTIPLE DEVICES ?
(2) 012366 001511 BEQ CCC ;NO,JUMP AROUND
(2) 012370 005767 166572 TST ACTREG ;ARE ANY DEVICES ACTIVE ?
(2) 012374 001007 BNE RUNIT ;YES
(2) 012376 104401 015327 TYPE ,MCOW ;NO
(2) 012402 016700 166560 MOV ACTREG,R0 ;DISPLAY ACTREG
(2) 012406 000000 HALT ;SELECT SOMETHING TO RUN @ ACTREG:
(2) ;SELECT SWITCHES & HIT CONTINUE (PUT SW00 =1)
(2) 012410 000167 167536 JMP .START ;START OVER AGAIN.....YOU DESELECTED EVERYTHING
(2) 012414 062767 000010 166532 RUNIT: ADD #10,BASEADD ;NEXT BLOCK (ADDRESSES)
(2) 012422 062767 000010 166532 ZERO: ADD #10,BASEIV ;NEXT BLOCK (VECTORS)
(2) 012430 000241 CLC
(2) 012432 006167 166532 ROL ROTADD ;UP DATE ROTATING POINTER
(2) 012436 103410 BCS 2$ ;IS IT THE LAST DEVICE
(2) ;TO BE TESTED IN THIS PASS ?

```

[illegible]



```

(2) 012732 017767 166510 000422      MOV    @STKB,.MSG      ;GET CHARACTER
(2) 012740 042767 177600 000414      BIC    #177600,.MSG    ;STRIP JUNK
(2) 012746 122767 000007 000406      CMPB   #7,.MSG        ;IS IT <^G> ?
(2) 012754 001017                BNE     OUT            ;NO
(2) 012756 104401 016103                TYPE   ,MCNTG
(2) 012762 005137 013022      CNTLU:  COM    @#RDSW
(2) 012766 104401 016113                TYPE   ,MMSWR
(2) 012772 104413                CONVRT
(2) 012774 013024                SWREGL
(2) 012776 104406 016124                INSTR,MMNEW
(2) 013002 104410                PARAM
(2) 013004 000000                0
(2) 013006 177777                177777
(2) 013010 000176                SWREG
(2) 013012 000000 001                .BYTE  0,1
(2) 013014 005037 013022      OUT:    CLR    @#RDSW
(2) 013020 000207                RTS     PC
(2) 013022 000000                RDSW:   .WORD  0
(2) 013024 000001                SWREGL: 1
(2) 013026 000006 002                .BYTE  6,2
(2) 013030 000176                SWREG
(2) 013032 000005                5
(2)                                ;CHECK FOR FREEZE ON CURRENT DATA
(2) 013034 004767 177646      .SCOP1: JSR    PC,CKSWR
(2) 013040 032777 001000 166372      BIT    #SW09,@SWR
(2) 013046 001402                BEQ     1$
(2) 013050 016716 166034      MOV    LOCK,(SP)
(2) 013054 000002      1$:    RTI
(2)                                .SBTTL  TYPE ROUTINE
(2)
(2)                                ;*****
(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    MESADR
(2)                                ;*
(2)                                ;*
(2)                                ;*
(2) 013056 105767 166375      $TYPE:  TSTB   $TPFLG      ;;IS THERE A TERMINAL?
(2) 013062 100002                BPL     1$            ;;BR IF YES
(2) 013064 000000                HALT                    ;;HALT HERE IF NO TERMINAL
(2) 013066 000430                BR      3$            ;;LEAVE
(2) 013070 010046      1$:    MOV    R0,-(SP)          ;;SAVE R0
(2) 013072 017600 000002      MOV    @2(SP),R0        ;;GET ADDRESS OF ASCIZ STRING
(2) 013076 122767 000001 166442      CMPB   #APTENV,$ENV    ;;RUNNING IN APT MODE
(2) 013104 001011                BNE     62$            ;;NO,GO CHECK FOR APT CONSOLE

```

```

(2) 013106 132767 000100 166433 BITB #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
(2) 013114 001405 BEQ 62$ ;;NO,GO CHECK FOR CONSOLE
(2) 013116 010067 000004 MOV R0,61$ ;;SETUP MESSAGE ADDRESS FOR APT
(2) 013122 004767 164660 JSR PC,$ATY3 ;;SPOOL MESSAGE TO APT
(2) 013126 000000 61$: .WORD 0 ;;MESSAGE ADDRESS
(2) 013130 132767 000040 166411 62$: BITB #APTC SUP,$ENVM ;;APT CONSOLE SUPPRESSED
(2) 013136 001003 BNE 60$ ;;YES,SKIP TYPE OUT
(2) 013140 112046 2$: MOVB (R0)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
(2) 013142 001005 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
(2) 013144 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
(2) 013146 012600 60$: MOV (SP)+,R0 ;;RESTORE R0
(2) 013150 062716 000002 3$: ADD #2,(SP) ;;ADJUST RETURN PC
(2) 013154 000002 RTI ;;RETURN
(2) 013156 122716 000011 4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
(2) 013162 001430 BEQ 8$
(2) 013164 122716 000200 CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
(2) 013170 001006 BNE 5$
(2) 013172 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
(2) 013174 104401 TYPE ;;TYPE A CR AND LF
(2) 013176 001523 $CRLF
(2) 013200 105067 000130 CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
(2) 013204 000755 BR 2$ ;;GET NEXT CHARACTER
(2) 013206 004767 000056 5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
(2) 013212 126726 166240 6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
(2) 013216 001350 BNE 2$ ;;IF NO GO GET NEXT CHAR.
(2) 013220 016746 166230 MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
(2) 013224 105366 000001 7$: DECB 1(SP) ;;AND THE NULL CHAR.
(2) 013230 002770 BLT 6$ ;;DOES A NULL NEED TO BE TYPED?
(2) 013232 004767 000032 JSR PC,$TYPEC ;;BR IF NO--GO POP THE NULL OFF OF STACK
(2) 013236 105367 000072 DECB $CHARCNT ;;GO TYPE A NULL
(2) 013242 000770 BR 7$ ;;DO NOT COUNT AS A COUNT
(2) ;;LOOP
(2)
(2) ;HORIZONTAL TAB PROCESSOR
(2)
(2) 013244 112716 000040 8$: MOVB #' ,(SP) ;;REPLACE TAB WITH SPACE
(2) 013250 004767 000014 9$: JSR PC,$TYPEC ;;TYPE A SPACE
(2) 013254 132767 000007 000052 BITB #7,$CHARCNT ;;BRANCH IF NOT AT
(2) 013262 001372 BNE 9$ ;;TAB STOP
(2) 013264 005726 TST (SP)+ ;;POP SPACE OFF STACK
(2) 013266 000724 BR 2$ ;;GET NEXT CHARACTER
(2) 013270 105777 166154 $TYPEC: TSTB @ $TPS ;;WAIT UNTIL PRINTER IS READY
(2) 013274 100375 BPL $TYPEC
(2) 013276 116677 000002 166146 MOVB 2(SP),@ $TPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(2)
(2) 013304 122766 000015 000002 CMPB #CR,2(SP) ;;IS CHARACTER A CARRIAGE RETURN?
(2) 013312 001003 BNE 1$ ;;BRANCH IF NO
(2) 013314 105067 000014 CLRB $CHARCNT ;;YES--CLEAR CHARACTER COUNT
(2) 013320 000406 BR $TYPEX ;;EXIT
(2) 013322 122766 000012 000002 1$: CMPB #LF,2(SP) ;;IS CHARACTER A LINE FEED?
(2) 013330 001402 BEQ $TYPEX ;;BRANCH IF YES
(2) 013332 105227 INCB (PC)+ ;;COUNT THE CHARACTER
(2) 013334 000000 $CHARCNT: .WORD 0 ;;CHARACTER COUNT STORAGE
(2) 013336 000207 $TYPEX: RTS PC
(2)
(2)

```



```

(2)
(2)
(2)
(2)
(2) 013340 017667 000000 000014 .INSTR: MOV @ (SP), MSG ; PICK UP MESSAGE
(2) 013346 062716 000002 ADD #2, (SP) ; JUMP AROUND MESSAGE FOR RTI
(2) 013352 105767 166170 TSTB $ENV ; APT CONTROL
(2) 013356 001036 BNE INSTR2 ; YES NO TYPE
(2) 013360 104401 .INST1: TYPE
(2) 013362 000000 .MSG: 0
(2) 013364 012704 016136 MOV #INBUF, R4 ; GET STARTING LOC OF INBUF
(2) 013370 012703 000007 MOV #7, R3 ; MAX # OF CHARS
(2) 013374 105777 166044 1$: TSTB @ $TKS ; TTY FLAG
(2) 013400 100375 BPL 1$
(2) 013402 117714 166040 MOV @ $TKB, (R4) ; TAKE CHAR
(2) 013406 142714 000200 BICB #200, (R4) ; STRIP
(2) 013412 121427 000025 CMPB (R4), #25 ; IS IT <^G>
(2) 013416 001760 BEQ .INST1
(2) 013420 122427 000015 CMPB (R4)+, #15 ; CHECK FOR CR
(2) 013424 001413 BEQ INSTR2
(2) 013426 105777 166016 2$: TSTB @ $TPS ; TEST FLAG
(2) 013432 100375 BPL 2$
(2) 013434 117777 166006 166010 MOVB @ $TKB, @ $TPB ; ECHO CHARACTER
(2) 013442 005303 DEC R3 ; DID YOU TYPE TOO MANY CHARS ?
(2) 013444 001353 BNE 1$
(2) 013446 104401 .INSTE: TYPE
(2) 013450 015423 MQM ; ?
(2) 013452 000742 BR .INST1 ; RETRY
(2) 013454 000002 INSTR2: RTI

; CONVERT ASCII STRING TO OCTAL

(2)
(2)
(2) 013456 011605 .PARAM: MOV (SP), R5 ; PUT CONTENTS OF SP INTO R5
(2) 013460 012567 000162 MOV (R5)+, LOLIM ; PUT LOW LIMIT INTO LOLIM
(2) 013464 012567 000160 MOV (R5)+, HILIM ; PUT HIGH LIMIT INTO HILIM
(2) 013470 012567 000156 MOV (R5)+, DEVADR ; PUT STORE LOC INTO DEVADR
(2) 013474 112567 000154 MOV (R5)+, LOBITS ; PUT MASK INTO LOBITS
(2) 013500 112567 000151 MOV (R5)+, ADCNT ; PUT COUNT INTO ADCNT
(2) 013504 010516 MOV R5, (SP) ; RESTORE RETURN ADDR ON STACK FOR RTI
(2) 013506 005005 PARAM1: CLR R5
(2) 013510 012704 016136 MOV #INBUF, R4
(2) 013514 122714 000015 CMPB #15, (R4) ; CR ?
(2) 013520 001420 BEQ PARERR ; YOU TYPED CR TOO SOON !
(2) 013522 121427 000060 1$: CMPB (R4), #60 ; LOW LIMIT ASCII 0
(2) 013526 002415 BLT PARERR
(2) 013530 121427 000067 CMPB (R4), #67 ; HIGH LIMIT ASCII 7
(2) 013534 003012 BGT PARERR
(2) 013536 142714 000060 BICB #60, (R4) ; CONVERT TO OCTAL
(2) 013542 152405 BISB (R4)+, R5 ; STORE AWAY ITS AN OK CHAR
(2) 013544 122714 000015 CMPB #15, (R4) ; CR ?
(2) 013550 001414 BEQ LIMITS ; NOW CHECK FOR HIGH & LOW LIMIT CONDS
(2) 013552 006305 ASL R5 ; ALLOCATE ROOM FOR NEXT CHAR
(2) 013554 006305 ASL R5
(2) 013556 006305 ASL R5
(2) 013560 000760 BR 1$
(2) 013562 122714 000015 PARERR: CMPB #15, (R4) ; CR?

```

```

(2) 013566 001003
(2) 013570 005737 013022
(2) 013574 001023
(2) 013576 104407
(2) 013600 000742
(2)
(2)
(2) 013602 020567 000042
(2) 013606 101365
(2) 013610 020567 000032
(2) 013614 103762
(2) 013616 136705 000032
(2) 013622 001357
(2)
(2)
(2) 013624 016704 000022
(2) 013630 010524
(2) 013632 062705 000002
(2) 013636 105367 000013
(2) 013642 001372
(2) 013644 000002
(2) 013646 000000
(2) 013650 000000
(2) 013652 000000
(2) 013654 000000
(2) 013655
(2)
(2)
(2) 013656 016667 000004 165242 .SAV05: MOV 4(SP),SAVPC
(2)
(2)
(2)
(2) 013664 010567 165604
(2) 013670 010467 165576
(2) 013674 010367 165570
(2) 013700 010267 165562
(2) 013704 010167 165554
(2) 013710 010067 165546
(2) 013714 000002
(2)
(2)
(2)
(2) 013716 016700 165540
(2) 013722 016701 165536
(2) 013726 016702 165534
(2) 013732 016703 165532
(2) 013736 016704 165530
(2) 013742 016705 165526
(2) 013746 000002
(2)
(2)
(2) 013750 104401

```

```

BNE 120$
TST @#RDSW ;CK SWR USED
BNE PARTI
120$: INSTER ;RETRY
BR PARAM1

;TEST TO SEE IF NUMBER IS WITHIN LIMITS
LIMITS: CMP R5,HILIM
BHI PARERR ;THE # IS TOO HIGH
CMP R5,LOLIM
BLO PARERR ;THE # IS TOO LOW
BITB LOBITS,R5 ;TEST BY MASKINGTHE #
BNE PARERR

;STORE NUMBER AT SPECIFIED ADDRESS
1$: MOV DEVADR,R4 ;GET STARTING ADDR OF
MOV R5,(R4)+ ;STORE AT THIS ADDR
ADD #2,R5
DECB ADRCNT ;HOW MANY TIMES + 2 ?
BNE 1$
PARTI: RTI
LOLIM: 0
HILIM: 0
DEVADR: 0
LOBITS: 0
ADRCNT=LOBITS+1

;SAVE PC OF TEST THAT FAILED AND R0-R5
.SAV05: MOV 4(SP),SAVPC

;SAVE R0-R5
SV05: MOV R5,$REG5
MOV R4,$REG4
MOV R3,$REG3
MOV R2,$REG2
MOV R1,$REG1
MOV R0,$REG0
RTI

;RESTORE R0-R5
.RES05: MOV $REG0,R0
MOV $REG1,R1
MOV $REG2,R2
MOV $REG3,R3
MOV $REG4,R4
MOV $REG5,R5
RTI

;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
.CONVR: TYPE

```



```

(2) 013752 015427          MCRLF      :CR LF
(2) 013754 017601 000000    MOV        @ (SP), R1      :PICK UP DATA POINTER
(2) 013760 062716 000002    ADD        #2, (SP)      :SET UP SP FOR RTI
(2) 013764 012167 000130    MOV        (R1)+, WRDCNT    :PICK UP # OF WORDS FROM TABLE
(2) 013770 112167 000126    1$: MOV     (R1)+, CHRCNT    :PICK UP # OF CHARS FROM TABLE
(2) 013774 112167 000123    MOV     (R1)+, SPACNT    :PICK UP # OF SPACES FROM TABLE
(2) 014000 013167 000120    MOV     @ (R1)+, BINWRD    :PICK UP ADDRESS OF MSG
(2)                                :FROM TABLE
(2) 014004 016704 000114    2$: MOV     BINWRD, R4      :SAVE
(2) 014010 116705 000106    MOV     CHRCNT, R5      :SAVE
(2) 014014 012700 016200    MOV     #TEMP, R0     :STARTING ADDRESS OF TEMP BLOCK
(2) 014020 010403          3$: MOV     R4, R3          :SAVE
(2) 014022 042703 177770    BIC     #177770, R3    :CLR OUT UPPER BITS .. SAVE CHAR
(2) 014026 062703 000260    ADD     #260, R3      :CONVERT TO ASCII
(2) 014032 110320          MOV     R3, (R0)+      :STORE AWAY
(2) 014034 006204          ASR     R4          :SHIFT FOR NEXT #
(2) 014036 006204          ASR     R4          :DITTO
(2) 014040 006204          ASR     R4          :DITTO
(2) 014042 005305          DEC     R5          :DEC CHAR COUNT
(2) 014044 001365          BNE     3$          :DO IT AGAIN ?
(2) 014046 012703 016242    MOV     #MDATA, R3      :STARTING ADDRESS OF MDATA BLOCK
(2) 014052 114023          4$: MOV     -(R0), (R3)+    :REVERSE THE ORDER OF NUMBERS
(2) 014054 105367 000042    DECB     CHRCNT      :DEC CHAR COUNT
(2) 014060 001374          BNE     4$          :DO IT AGAIN ?
(2) 014062 105767 000035    TSTB     SPACNT      :HOW MANY SPACES ?
(2) 014066 001405          BEQ     6$          :TYPE # IF BR =0
(2) 014070 112723 000240    5$: MOV     #240, (R3)+    :"SPACE" IN ASCII
(2) 014074 105367 000023    DECB     SPACNT      :DEC # OF SPACE COUNT
(2) 014100 001373          BNE     5$          :DO IT AGAIN ?
(2) 014102 105013          6$: CLRB     (R3)          :INSERT '0' FOR TTY OUTPUT ROUTINE
(2) 014104 104401          TYPE
(2) 014106 016242          MDATA      :THIS MESSAGE
(2) 014110 005367 000004    DEC     WRDCNT      :HOW MANY #'S ?
(2) 014114 001325          BNE     1$          :DO THIS ROUTINE AGAIN IF NOT EQUAL TO 0
(2) 014116 000002          RTI      :RETURN TO PROGRAM
(2) 014120 000000          WRDCNT: 0
(2) 014122 000000          CHRCNT: 0
(2) 014124 000000          SPACNT=CHRCNT+1
(2)                                BINWRD: 0

                                :COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
                                :BUFFER TO THE CHARACTERS 'N' AND 'Y'.
                                :IF THE CHARACTER IS 'N' CLEAR THE FLAG
                                :IF THE CHARACTER IS 'Y' SET THE FLAG
(2) 014126 017605 000000    .SETFLG:MOV @ (SP), R5
(2) 014132 122767 000116 001776 CMPB     #'N', INBUF      :IS IT 'N' ?
(2) 014140 001002          BNE     1$
(2) 014142 105015          CLRB     (R5)          :000
(2) 014144 000406          BR      2$
(2) 014146 122767 000131 001762 1$: CMPB     #'Y', INBUF      :IS IT 'Y' ?
(2) 014154 001005          BNE     3$
(2) 014156 112715 177777          MOV     #-1, (R5)      :377
(2) 014162 062716 000002          2$: ADD     #2, (SP)
(2) 014166 000002          RTI
(2) 014170 104407          3$: INSTER :RETRY

```

```

(2) 014172 000755      BR      .SETFLG
(2)                   .SBTTL  ERROR HANDLER ROUTINE
(2)
(2) *****
(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) 014174      $ERROR:
(2) 014174 105267 165203 7$:      INCB      $ERFLG      ;;SET THE ERROR FLAG
(2) 014200 001775      BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
(2) 014202 016777 165174 165232      MOV      $TSTNM,$DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(2) 014210 032777 002000 165222      BIT      #BIT10,$SWR      ;;BELL ON ERROR?
(2) 014216 001402      BEQ      1$      ;;NO - SKIP
(2) 014220 104401 001516      TYPE      ,SBELL      ;;RING BELL
(2) 014224 005267 165162      1$:      INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
(2) 014230 011667 165162      MOV      (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
(2) 014234 162767 000002 165154      SUB      #2,$ERRPC
(2) 014242 117767 165150 165144      MOVB     @ $ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CCDE
(2) 014250 032777 020000 165162      BIT      #BIT13,$SWR      ;;SKIP TYPEOUT IF SET
(2) 014256 001004      BNE      20$      ;;SKIP TYPEOUTS
(2) 014260 004767 000072      JSR      PC,SAVIT      ;;GO TO USER ERROR ROUTINE
(2) 014264 104401 001523      TYPE      ,SCRLF
(2) 014270      20$:
(2) 014270 122767 000001 165250      CMPB     #APTENV,$ENV      ;;RUNNING IN APT MODE
(2) 014276 001007      BNE      2$      ;;NO,SKIP APT ERROR REPORT
(2) 014300 116767 165110 000004      MOVB     $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
(2) 014306 004767 163504      JSR      PC,$ATY4      ;;REPORT FATAL ERROR TO APT
(2) 014312 000      21$:      .BYTE      0
(2) 014313 000      .BYTE      0
(2) 014314 000777      22$:      BR      22$      ;;APT ERROR LOOP
(2) 014316 005777 165116      2$:      TST      @SWR      ;;HALT ON ERROR
(2) 014322 100001      BPL      3$      ;;SKIP IF CONTINUE
(2) 014324 000000      HALT      ;;HALT ON ERROR!
(2) 014326 032777 001000 165104      3$:      BIT      #BIT09,$SWR      ;;LOOP ON ERROR SWITCH SET?
(2) 014334 001402      BEQ      4$      ;;BR IF NO
(2) 014336 016716 165046      MOV      $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
(2) 014342 005767 165146      4$:      TST      $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
(2) 014346 001402      BEQ      5$      ;;BR IF NONE
(2) 014350 016716 165140      MOV      $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
(2) 014354      5$:
(2) 014354 000002      RTI      ;;RETURN
(2) 014356 010067 164546      SAVIT:  MOV      R0,HLD0
(2) 014362 010167 164544      MOV      R1,HLD1
(2) 014366 010267 164542      MOV      R2,HLD2
(2) 014372 010367 164540      MOV      R3,HLD3
(2) 014376 010467 164536      MOV      R4,HLD4
(2) 014402 010567 164534      MOV      R5,HLD5
(2) 014406 016767 164770 164530      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)      $ERRTYP:
(3)      014414      104401      001523      TYPE      $CRLF      ;:'CARRIAGE RETURN' & 'LINE FEED'
(3)      014414      104401      001523      MOV      R0,-(SP)      ;:SAVE R0
(3)      014420      010046      CLR      R0      ;:PICKUP THE ITEM INDEX
(3)      014422      005000      BISB     @#$ITEMB,R0
(3)      014424      153700      001414      BNE      1$      ;:IF ITEM NUMBER IS ZERO, JUST
(3)      014430      001004      MOV      $ERRPC,-(SP)      ;:TYPE THE PC OF THE ERROR
(3)      014432      016746      164760      ;:SAVE $ERRPC FOR TYPEOUT
(3)      014432      016746      164760      ;:ERROR ADDRESS
(3)      014436      104402      TYP0C     ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
(3)      014440      000426      BR      6$      ;:GET OUT
(3)      014442      005300      1$:      DEC      R0      ;:ADJUST THE INDEX SO THAT IT WILL
(3)      014444      006300      ASL      R0      ;:WORK FOR THE ERROR TABLE
(3)      014446      006300      ASL      R0
(3)      014450      006300      ASL      R0
(3)      014452      062700      001652      ADD      #$ERRTB,R0      ;:FORM TABLE POINTER
(3)      014456      012067      000004      MOV      (R0)+,2$      ;:PICKUP "ERROR MESSAGE" POINTER
(3)      014462      001404      BEQ      3$      ;:SKIP TYPEOUT IF NO POINTER
(3)      014464      104401      TYPE      ;:TYPE THE "ERROR MESSAGE"
(3)      014466      000000      2$:      .WORD     0      ;:"ERROR MESSAGE" POINTER GOES HERE
(3)      014470      104401      001523      TYPE      $CRLF      ;:'CARRIAGE RETURN' & 'LINE FEED'
(3)      014474      012067      000004      3$:      MOV      (R0)+,4$      ;:PICKUP "DATA HEADER" POINTER
(3)      014500      001404      BEQ      5$      ;:SKIP TYPEOUT IF 0
(3)      014502      104401      TYPE      ;:TYPE THE "DATA HEADER"
(3)      014504      000000      4$:      .WORD     0      ;:"DATA HEADER" POINTER GOES HERE
(3)      014506      104401      001523      TYPE      $CRLF      ;:'CARRIAGE RETURN' & 'LINE FEED'
(3)      014512      011000      5$:      MOV      (R0),R0      ;:PICKUP "DATA TABLE" POINTER
(3)      014514      001004      BNE      7$      ;:GO TYPE THE DATA
(3)      014516      012600      6$:      MOV      (SP)+,R0      ;:RESTORE R0
(3)
(3)      014520      104401      001523      TYPE      $CRLF      ;:'CARRIAGE RETURN' & 'LINE FEED'
(3)      014524      000207      7$:      RTS      PC      ;:RETURN
(3)      014526      013046      MOV      @ (R0)+,-(SP)      ;:SAVE @ (R0)+ FOR TYPEOUT
(3)      014530      104402      TYP0C     ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
(3)      014532      005710      TST      (R0)      ;:IS THERE ANOTHER NUMBER?
(3)      014534      001770      BEQ      6$      ;:BR IF NO
(3)      014536      104401      014544      TYPE      8$      ;:TYPE TWO(2) SPACES
(3)      014542      000771      BR      7$      ;:LOOP
(3)      014544      020040      8$:      .ASCIZ    / /      ;:TWO(2) SPACES
(3)      014550      014550      .EVEN
(2)      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
(2)
(3)      ;*****
(2)      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
(2)      ;*OCTAL (ASCII) NUMBER AND TYPE IT.
(2)      ;*$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)      ;*      $STYPON----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)      ;*      $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)      014550 017646 000000      STYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
(2)      014554 116667 000001 000211      MOV      1(SP), $OFILL      ;;LOAD ZERO FILL SWITCH
(2)      014562 112667 000207      MOV      (SP)+, $OMODE+1      ;;NUMBER OF DIGITS TO TYPE
(2)      014566 062716 000002      ADD      #2, (SP)      ;;ADJUST RETURN ADDRESS
(2)      014572 000406      BR      $TYPON
(2)      014574 112767 000001 000171      STYPOC: MOV      #1, $OFILL      ;;SET THE ZERO FILL SWITCH
(2)      014602 112767 000006 000165      MOV      #6, $OMODE+1      ;;SET FOR SIX(6) DIGITS
(2)      014610 112767 000005 000154      STYPON: MOV      #5, $OCNT      ;;SET THE ITERATION COUNT
(2)      014616 010346      MOV      R3, -(SP)      ;;SAVE R3
(2)      014620 010446      MOV      R4, -(SP)      ;;SAVE R4
(2)      014622 010546      MOV      R5, -(SP)      ;;SAVE R5
(2)      014624 116704 000145      MOV      $OMODE+1, R4      ;;GET THE NUMBER OF DIGITS TO TYPE
(2)      014630 005404      NEG      R4
(2)      014632 062704 000006      ADD      #6, R4      ;;SUBTRACT IT FOR MAX. ALLOWED
(2)      014636 110467 000132      MOV      R4, $OMODE      ;;SAVE IT FOR USE
(2)      014642 116704 000125      MOV      $OFILL, R4      ;;GET THE ZERO FILL SWITCH
(2)      014646 016605 000012      MOV      12(SP), R5      ;;PICKUP THE INPUT NUMBER
(2)      014652 005003      CLR      R3      ;;CLEAR THE OUTPUT WORD
(2)      014654 006105      1$: ROL      R5      ;;ROTATE MSB INTO 'C'
(2)      014656 000404      BR      3$      ;;GO DO MSB
(2)      014660 006105      2$: ROL      R5      ;;FORM THIS DIGIT
(2)      014662 006105      ROL      R5
(2)      014664 006105      ROL      R5
(2)      014666 010503      MOV      R5, R3
(2)      014670 006103      3$: ROL      R3      ;;GET LSB OF THIS DIGIT
(2)      014672 105367 000076      DECB      $OMODE      ;;TYPE THIS DIGIT?
(2)      014676 100016      BPL      7$      ;;BR IF NO
(2)      014700 042703 177770      BIC      #177770, R3      ;;GET RID OF JUNK
(2)      014704 001002      BNE      4$      ;;TEST FOR 0
(2)      014706 005704      TST      R4      ;;SUPPRESS THIS 0?
(2)      014710 001403      BEQ      5$      ;;BR IF YES
(2)      014712 005204      4$: INC      R4      ;;DON'T SUPPRESS ANYMORE 0'S
(2)      014714 052703 000060      BIS      #'0, R3      ;;MAKE THIS DIGIT ASCII
(2)      014720 052703 000040      5$: BIS      #'1, R3      ;;MAKE ASCII IF NOT ALREADY
(2)      014724 110367 000040      MOV      R3, 8$      ;;SAVE FOR TYPING
(2)      014730 104401 014770      TYPE      8$      ;;GO TYPE THIS DIGIT
(2)      014734 105367 000032      7$: DECB      $OCNT      ;;COUNT BY 1
(2)      014740 003347      BGT      2$      ;;BR IF MORE TO DO
  
```



```

(2) 014742 002402          BLT      6$          ;;BR IF DONE
(2) 014744 005204          INC      R4          ;;INSURE LAST DIGIT ISN'T A BLANK
(2) 014746 000744          BR       2$          ;;GO DO THE LAST DIGIT
(2) 014750 012605          6$:  MOV    (SP)+,R5      ;;RESTORE R5
(2) 014752 012604          MOV    (SP)+,R4      ;;RESTORE R4
(2) 014754 012603          MOV    (SP)+,R3      ;;RESTORE R3
(2) 014756 016666 000002 000004 MOV    2(SP),4(SP)  ;;SET THE STACK FOR RETURNING
(2) 014764 012616          MOV    (SP)+,(SP)
(2) 014766 000002          RTI                ;;RETURN
(2) 014770 000          8$:  .BYTE  0          ;;STORAGE FOR ASCII DIGIT
(2) 014771 000          .BYTE  0          ;;TERMINATOR FOR TYPE ROUTINE
(2) 014772 000          $OCNT: .BYTE  0        ;;OCTAL DIGIT COUNTER
(2) 014773 000          $OFILL: .BYTE  0       ;;ZERO FILL SWITCH
(2) 014774 000000          $OMODE: .WORD  0     ;;NUMBER OF DIGITS TO TYPE
(2)                      ;ENTER HERE ON POWER FAILURE
(2)
(2)
(2) 014776          $PWRDN:
(2) 014776 010046          .PFAIL: MOV    R0,-(SP)      ;SAVE R0-R5 ON PROCESSOR STACK
(2) 015000 010146          MOV    R1,-(SP)
(2) 015002 010246          MOV    R2,-(SP)
(2) 015004 010346          MOV    R3,-(SP)
(2) 015006 010446          MOV    R4,-(SP)
(2) 015010 010546          MOV    R5,-(SP)
(2) 015012 016746 163006          MOV    24,-(SP)
(2) 015016 010667 164074          MOV    SP,SAVSP      ;SAVE STACK POINTER
(2) 015022 012767 015034 162774 MOV    #RESTART,24  ;SET UP FOR POWER UP TRAP
(2) 015030 000000          HALT
(2) 015032 000777          BR       .          ;HALT ON POWER DOWN NORMAL
(2)
(2)                      ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
(2)
(2) 015034 016706 164056          RESTAR: MOV    SAVSP,SP      ;RESTORE STACK POINTER
(2) 015040 012605          MOV    (SP)+,R5      ;RESTORE R0-R5
(2) 015042 012604          MOV    (SP)+,R4
(2) 015044 012603          MOV    (SP)+,R3
(2) 015046 012602          MOV    (SP)+,R2
(2) 015050 012601          MOV    (SP)+,R1
(2) 015052 012600          MOV    (SP)+,R0
(2) 015054 012767 014776 162742 MOV    #.PFAIL,24  ;SET UP FOR POWER FAILURE
(2) 015062 106427 000300          MTPS    #300
(2) 015066 012706 001100          MOV    #STACK,SP
(2) 015072 005067 001102          CLR     TEMP
(2) 015076 005267 001076          INC     TEMP
(2) 015102 001375          BNE     .-4
(2) 015104 104413          CONVRT
(2) 015106 015130          PFTAB
(2) 015110 104401          TYPE
(2) 015112 015432          MPFAIL
(2) 015114 005067 164263          CLR     $ERFLG
(2) 015120 005067 164272          CLR     $ERRPC
(2) 015124 000177 163754          JMP     @RETURN
(2) 015130 000001          PFTAB: 1
(2) 015132 006          .BYTE  6,2
(2) 015134 000207          RETURN
(2) 015136 005015 042012 053125 MTITLE: .ASCIIZ <15><12><12>/DUV11 CNDUS-A TAPE C /<15><12>

```

|     |        |        |        |        |                                                               |
|-----|--------|--------|--------|--------|---------------------------------------------------------------|
| (2) | 015144 | 030461 | 041440 | 042116 |                                                               |
| (2) | 015152 | 051525 | 040455 | 052040 |                                                               |
| (2) | 015160 | 050101 | 020105 | 020103 |                                                               |
| (2) | 015166 | 005015 | 000    |        |                                                               |
| (2) | 015171 | 015    | 053012 | 041505 | MVECTO: .ASCIIZ <15><12>/VEC ADD- /                           |
| (2) | 015176 | 040440 | 042104 | 000055 |                                                               |
| (2) | 015204 | 005015 | 051461 | 020124 | MREGAD: .ASCIIZ <15><12>/1ST DEV: REC CSR ADD- /              |
| (2) | 015212 | 042504 | 035126 | 051040 |                                                               |
| (2) | 015220 | 041505 | 041440 | 051123 |                                                               |
| (2) | 015226 | 040440 | 042104 | 000055 |                                                               |
| (2) | 015234 | 005015 | 052515 | 052114 | MMULT: .ASCIIZ <15><12>/MULT DEV ? (Y OR N)- /                |
| (2) | 015242 | 042040 | 053105 | 037440 |                                                               |
| (2) | 015250 | 024040 | 020131 | 051117 |                                                               |
| (2) | 015256 | 047040 | 026451 | 000    |                                                               |
| (2) | 015263 | 015    | 046012 | 051501 | MLASTD: .ASCIIZ <15><12>/LAST DEV: REC CSR ADDR- /            |
| (2) | 015270 | 020124 | 042504 | 035126 |                                                               |
| (2) | 015276 | 051040 | 041505 | 041440 |                                                               |
| (2) | 015304 | 051123 | 040440 | 042104 |                                                               |
| (2) | 015312 | 026522 | 000    |        |                                                               |
| (2) | 015315 | 075    | 042504 | 044526 | DEVICE: .ASCIIZ /=DEVICE /                                    |
| (2) | 015322 | 042503 | 020040 | 000    |                                                               |
| (2) | 015327 | 015    | 051412 | 046105 | MCOW: .ASCIIZ <15><12>/SELECT TO RUN @ACTREG /                |
| (2) | 015334 | 041505 | 020124 | 047524 |                                                               |
| (2) | 015342 | 051040 | 047125 | 040040 |                                                               |
| (2) | 015350 | 041501 | 051124 | 043505 |                                                               |
| (2) | 015356 | 000    |        |        |                                                               |
| (2) | 015357 | 015    | 047412 | 043126 | MRANGE: .ASCIIZ <15><12>/OVFLO:RETYPE LAST DEV RXCSR ADDS- /  |
| (2) | 015364 | 047514 | 051072 | 052105 |                                                               |
| (2) | 015372 | 050131 | 020105 | 040514 |                                                               |
| (2) | 015400 | 052123 | 042040 | 053105 |                                                               |
| (2) | 015406 | 051040 | 041530 | 051123 |                                                               |
| (2) | 015414 | 040440 | 042104 | 026523 |                                                               |
| (2) | 015422 | 000    |        |        |                                                               |
| (2) | 015423 | 040    | 037440 | 000    | MQM: .ASCIIZ / ? /                                            |
| (2) | 015427 | 015    | 000012 |        | MCRLF: .ASCIIZ <15><12>                                       |
| (2) | 015432 | 043120 | 044501 | 026114 | MPFAIL: .ASCIIZ /PFAIL, RESTART AT TEST IN PROGRESS /         |
| (2) | 015440 | 020040 | 042522 | 052123 |                                                               |
| (2) | 015446 | 051101 | 020124 | 052101 |                                                               |
| (2) | 015454 | 052040 | 051505 | 020124 |                                                               |
| (2) | 015462 | 047111 | 050040 | 047522 |                                                               |
| (2) | 015470 | 051107 | 051505 | 000123 |                                                               |
| (2) | 015476 | 005015 | 047105 | 020104 | MEPASS: .ASCIIZ <15><12>/END OF PASS TAPE C /                 |
| (2) | 015504 | 043117 | 050040 | 051501 |                                                               |
| (2) | 015512 | 020123 | 040524 | 042520 |                                                               |
| (2) | 015520 | 041440 | 000    |        |                                                               |
| (2) | 015523 | 015    | 051012 | 000    | MR: .ASCIIZ <15><12>/R /                                      |
| (2) | 015527 | 015    | 052012 | 051505 | MTSTPC: .ASCIIZ <15><12>/TEST PC- /                           |
| (2) | 015534 | 020124 | 041520 | 000055 |                                                               |
| (2) | 015542 | 005015 | 047514 | 045503 | MLOCK: .ASCIIZ <15><12>/LOCK ON TEST? (Y OR N)- /             |
| (2) | 015550 | 047440 | 020116 | 052040 |                                                               |
| (2) | 015556 | 051505 | 037524 | 024040 |                                                               |
| (2) | 015564 | 020131 | 051117 | 047040 |                                                               |
| (2) | 015572 | 026451 | 000    |        |                                                               |
| (2) | 015575 | 015    | 021412 | 047440 | MSYNC: .ASCIIZ <15><12>/# OF SYNC CHARS SELECTED ( 1 OR 2)- / |
| (2) | 015602 | 020106 | 054523 | 041516 |                                                               |
| (2) | 015610 | 041440 | 040510 | 051522 |                                                               |



```

(2) 015616 051440 046105 041505
(2) 015624 042524 020104 020050
(2) 015632 020061 051117 031040
(2) 015640 026451 000
(2) 015643 015 044412 020123 MWIRE6: .ASCIIZ <15><12>/IS SEC XMIT SWITCH E55-2 IN? (Y OR N)-/
(2) 015650 042523 020103 046530
(2) 015656 052111 051440 044527
(2) 015664 041524 020110 032505
(2) 015672 026465 020062 047111
(2) 015700 020077 054450 047440
(2) 015706 020122 024516 000055
(2) 015714 005015 051511 051440 MWIRE5: .ASCIIZ <15><12>/IS SEC REC SWITCH E55-3 IN? (Y OR N)-/
(2) 015722 041505 051040 041505
(2) 015730 051440 044527 041524
(2) 015736 020110 032505 026465
(2) 015744 020063 047111 020077
(2) 015752 054450 047440 020122
(2) 015760 024516 000055
(2) 015764 005015 051511 047440 MWIRE4: .ASCIIZ <15><12>/IS OPT CLP ENABLE SWITCH E55-1 IN? (Y OR N)-/
(2) 015772 052120 041440 051114
(2) 016000 042440 040516 046102
(2) 016006 020105 053523 052111
(2) 016014 044103 042440 032465
(2) 016022 030455 044440 037516
(2) 016030 024040 020131 051117
(2) 016036 047040 026451 000
(2) 016043 015 005012 031510 MEXTJ: .ASCIIZ <15><12><12>/H315 CONNECTOR ON ?(Y OR N)-/
(2) 016050 032461 041440 047117
(2) 016056 042516 052103 051117
(2) 016064 047440 020116 024077
(2) 016072 020131 051117 047040
(2) 016100 026451 000
(2) 016103 015 020012 043536 MCNTG: .ASCIIZ <15><12>/ ^G /
(2) 016110 020040 000
(2) 016113 040 053523 036522 MMSWR: .ASCIIZ / SWR= /
(2) 016120 020040 000040
(2) 016124 020040 047040 053505 MMNEW: .ASCIIZ / NEW= /
(2) 016132 020075 000040
(2)
(2) .EVEN
(2)
(2) ;BUFFERS FOR INPUT-OUTPUT
(2)
(2) INBUF: 0
(2) 016136 000000 .=. +40
(2) 016200 016200 TEMP: 0
(2) 016242 000000 .=. +40
(2) 016242 016242 MDATA: 0
(2) 016304 000000 .=. +40
(3) .SBTTL SCOPE HANDLER ROUTINE STARS
(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)          $SCOPE:
(5)          ;SCOPE LOOP AND INTERATION HANDLER
(5)          .SCOPE:
(5) 016304    JSR      PC,CKSWR
(5) 016304    CLR      $ERRPC          ;CLEAR LAST ERROR PC
(5) 016310    CMP      #TST1+2,(SP)    ;IS SCOPE AT BEGINING OF TEST 1?
(5) 016314    BEQ      $XTSTR          ;YES NO LOOP.
(5) 016320    001422
(5) 016322    032777    040000    163110    TTST:  BIT      #BIT14,$SWR          ;THIS CODE IS FOR TESTING FOR BIT 14
(5) 016330    001412                                ;ON LSI WHICH SYSMAC CANNOT HANDLE
(5) 016332    016767    163044    163046    MOV      $TSTNM,$LPADR
(5) 016340    000406    BR      1$
(5) 016342    105777    163076    TSTB     @STKS          ;KEYBOARD DONE?
(5) 016346    100123    BPL      $OVER          ;BR IF NO
(5) 016350    017766    163072    177776    MOV      @STKB,-2(SP)    ;CLEAR DONE BIT
(3) 016356    032777    040000    163054    1$:  BIT      #BIT14,$SWR    ;LOOP ON PRESENT TEST?
(3) 016364    001114    BNE      $OVER          ;YES IF SW14=1
(3)          ;#####START OF CODE FOR THE XOR TESTER#####
(3) 016366    000416    $XTSTR: BR      6$          ;IF RUNNING ON THE "XOR" TESTER CHANGE
(3)          ;THIS INSTRUCTION TO A "NOP" (NOP=240)
(3) 016370    013746    000004    MOV      @#ERRVEC,-(SP)    ;SAVE THE CONTENTS OF THE ERROR VECTOR
(3) 016374    012737    016414    000004    MOV      #5,$@#ERRVEC    ;SET FOR TIMEOUT
(3) 016402    005737    177060    TST      @#177060          ;TIME OUT ON XOR?
(3) 016406    012637    000004    MOV      (SP)+,$@#ERRVEC    ;RESTORE THE ERROR VECTOR
(3) 016412    000463    BR      $SVLAD          ;GO TO THE NEXT TEST
(3) 016414    022626    5$:  CMP      (SP)+,(SP)+          ;CLEAR THE STACK AFTER A TIME OUT
(3) 016416    012637    000004    MOV      (SP)+,$@#ERRVEC    ;RESTORE THE ERROR VECTOR
(3) 016422    000423    BR      7$          ;LOOP ON THE PRESENT TEST
(3) 016424
(3) 016424    032777    000400    163006    6$:;#####END OF CODE FOR THE XOR TESTER#####
(3) 016432    001404    BIT      #BIT08,$SWR          ;LOOP ON SPEC. TEST?
(3) 016434    127767    163000    162740    BEQ      2$          ;BR IF NO
(3) 016442    001465    CMPB     @SWR,$TSTNM          ;ON THE RIGHT TEST? SWR<7:0>
(3) 016444    105767    162733    2$:  BEQ      $OVER          ;BR IF YES
(3) 016450    001421    TSTB     $ERFLG          ;HAS AN ERROR OCCURRED?
(3) 016452    126767    162737    162723    BEQ      3$          ;BR IF NO
(3) 016460    101015    CMPB     $ERMAX,$ERFLG          ;MAX. ERRORS FOR THIS TEST OCCURRED?
(3) 016462    032777    001000    162750    BHI      3$          ;BR IF NO
(3) 016470    001404    BIT      #BIT09,$SWR          ;LOOP ON ERROR?
(3) 016472    016767    162712    162706    BEQ      4$          ;BR IF NO
(3) 016500    000446    MOV      $LPERR,$LPADR          ;SET LOOP ADDRESS TO LAST SCOPE
(3) 016502    105067    162675    4$:  BR      $OVER          ;ZERO THE ERROR FLAG
(3) 016506    005067    163000    CLRB     $ERFLG          ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(3) 016512    000415    CLR      $TIMES          ;ESCAPE TO THE NEXT TEST
(3) 016514    032777    004000    162716    3$:  BR      1$          ;INHIBIT ITERATIONS?
(3) 016522    001011    BNE      1$          ;BR IF YES
(3) 016524    005767    163004    TST      $PASS          ;IF FIRST PASS OF PROGRAM
(3) 016530    001406    BEQ      1$          ;INHIBIT ITERATIONS
(3) 016532    005267    162646    INC      $ICNT          ;INCREMENT ITERATION COUNT
(3) 016536    026767    162750    162640    CMP      $TIMES,$ICNT    ;CHECK THE NUMBER OF ITERATIONS MADE

```



```

(3) 016544 002024      BGE      $OVER      ;;BR IF MORE ITERATION REQUIRED
(3) 016546 012767 000001 162630 1$:      MOV      #1,$ICNT      ;;REINITIALIZE THE ITERATION COUNTER
(3) 016554 016767 000056 162730      MOV      $MXCNT,$TIMES      ;;SET NUMBER OF ITERATIONS TO DO
(3) 016562 105267 162614      $SVLAD: INCR      $TSTNM      ;;COUNT TEST NUMBERS
(3) 016566 116767 162610 162736      MOV      $TSTNM,$TESTN      ;;SET TEST NUMBER IN APT MAILBOX
(3) 016574 011667 162606      MOV      (SP),$LPADR      ;;SAVE SCOPE LOOP ADDRESS
(3) 016600 011667 162604      MOV      (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
(3) 016604 005067 162704      CLR      $ESCAPE      ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(3) 016610 112767 000001 162577      MOV      #1,$ERMAX      ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(3) 016616 016777 162560 162616 $OVER:  MOV      $TSTNM,@DISPLAY      ;;DISPLAY TEST NUMBER
(3) 016624 016716 162556      MOV      $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
(5) 016630 000002      4$:      RTI
(5) 016632 001407      BRW:      1407
(5) 016634 000432      BRX:      432
(3) 016636 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) 016640 010046      $TRAP:  MOV      R0,-(SP)      ;;SAVE R0
(2) 016642 016600 000002      MOV      2(SP),R0      ;;GET TRAP ADDRESS
(2) 016646 005740      TST      -(R0)      ;;BACKUP BY 2
(2) 016650 111000      MOV      (R0),R0      ;;GET RIGHT BYTE OF TRAP
(2) 016652 006300      ASL      R0      ;;POSITION FOR INDEXING
(2) 016654 016000 016674      MOV      $TRPAD(R0),R0      ;;INDEX TO TABLE
(2) 016660 000200      RTS      R0      ;;GO TO ROUTINE
(2)
(2)      ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
(2)
(2) 016662 011646      $TRAP2: MOV      (SP),-(SP)      ;;MOVE THE PC DOWN
(2) 016664 016666 000004 000002      MOV      4(SP),2(SP)      ;;MOVE THE PSW DOWN
(2) 016672 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) 016674 016662      $TRPAD: .WORD      $TRAP2
(4) 016676 013056      $TYPE      ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
(4) 016700 014574      $TYPOC      ;;CALL=TYPOC      TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
(4) 016702 014550      $TYPOS      ;;CALL=TYPOS      TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
(4) 016704 014610      $TYPON      ;;CALL=TYPON      TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
(2)
(2)
(3) 016706 013034      .SCOP1      ;;CALL=SCOP1      TRAP+5(104405)
(3) 016710 013340      .INSTR      ;;CALL=INSTR      TRAP+6(104406)
(3) 016712 013446      .INSTER      ;;CALL=INSTER      TRAP+7(104407)
(3) 016714 013456      .PARAM      ;;CALL=PARAM      TRAP+10(104410)

```

```

(3) 016716 013656 .SAV05 ::CALL=SAV05 TRAP+11(104411)
(3) 016720 013716 .RES05 ::CALL=RES05 TRAP+12(104412)
(3) 016722 013750 .CONVRT ::CALL=CONVRT TRAP+13(104413)
(3) 016724 014126 .SETFLG ::CALL=SETFLG TRAP+14(104414)
*****
;UTILITIES
*****
;THIS UTILITY CALCULATES PRIORITY LEVEL
DULEV: ASL DUPRT ;SHIFT LEFT
        ASL DUPRT
        ASL DUPRT
        ASL DUPRT
        ASL DUPRT
        MOV DUPRT,LESS1 ;MOVE THIS TO LESS1
        SUB #1,LESS1 ;CREATE LESS1
        BIC #37,LESS1 ;CLEAR TNZVC
        RTS PC
DUPRT: PR5
LESS1: PR4 ;LEVEL TO ALLOW INTERRUPTS

;NEW DU ADDRESSES
DUADDR: MOV DUBASE,RXCSR ;XXX0
        INC DUBASE
        MOV DUBASE,HRXCSR ;XXX1
        INC DUBASE
        MOV DUBASE,RXDBUF ;XXX2
        MOV DUBASE,PARCSR ;XXX2
        INC DUBASE
        MOV DUBASE,HRXDBUF ;XXX3
        MOV DUBASE,HPARCSR ;XXX3
        INC DUBASE
        MOV DUBASE,TXCSR ;XXX4
        INC DUBASE
        MOV DUBASE,HTXCSR ;XXX5
        INC DUBASE
        MOV DUBASE,TXDBUF ;XXX6
        INC DUBASE
        MOV DUBASE,HTXDBUF ;XXX7
        RTS PC
DUBASE: 0

;THIS UTILITY POKES THE MAINT DATA BASED UPON THE
;INFORMATION CONTAINED IN $TMP1 AND IT IS
;SHIFTED IN BY THE CONTENTS OF SHIFT
RPOKE: BIC #MTDATA,@TXCSR
        CLR $TMP2
        ROR $TMP1 ;FORCE CARRY
        ROR $TMP2 ;PICK UP CARRY IN BIT 15
        ASR $TMP2 ;SHIFT INTO BIT 14
        BIC #BIT15,$TMP2 ;CLR BIT 15
        BIS $TMP2,@TXCSR ;POKE MAINT DATA
        BIC #CLK,@TXCSR ;POKE CLK
        BIS #CLK,@TXCSR
        DEC SHIFT
        BNE RPOKE
  
```



```

8238 017222 000207      RTS      PC
8239      ;THIS ROUTINE CALCULATES ODD PARITY FOR AN 8 BIT CHAR
8240 017224 016767 162250 162250 ODD8: MOV      $TMP1,$TMP2      ;SAVE TEMP1
8241 017232 005067 162246      CLR      $TMP3
8242 017236 012727 000010      MOV      #8.,(PC)+
8243 017242 000000      4$:      0
8244 017244 006067 162232      1$:      ROR      $TMP2
8245 017250 005567 162230      ADC      $TMP3
8246 017254 005367 177762      DEC      4$
8247 017260 001371      BNE      1$
8248 017262 006067 162216      ROR      $TMP3
8249 017266 103404      BCS      2$
8250 017270 052767 000400 162202  BIS      #BIT8,$TMP1      ;SET ODD PARITY
8251 017276 000403      BR       3$
8252 017300 042767 000400 162172 2$:      BIC      #BIT8,$TMP1      ;CLR EVEN PARITY
8253      ;$TMP1 NOW HAS ODD PARITY CHARACTER
8254 017306 000207      3$:      RTS      PC
8255
8256      ;THIS ROUTINE CALCULATES EVEN PARITY FOR AN 8 BIT CHARACTER
8257 017310 016767 162164 162164 EVEN8: MOV      $TMP1,$TMP2      ;SAVE TEMP1
8258 017316 005067 162162      CLR      $TMP3
8259 017322 012727 000010      MOV      #8.,(PC)+
8260 017326 000000      4$:      0
8261 017330 006067 162146      1$:      ROR      $TMP2
8262 017334 005567 162144      ADC      $TMP3
8263 017340 005367 177762      DEC      4$
8264 017344 001371      BNE      1$
8265 017346 006067 162132      ROR      $TMP3
8266 017352 103004      BCC      2$
8267 017354 052767 000400 162116  BIS      #BIT8,$TMP1      ;SET EVEN PARITY
8268 017362 000403      BR       3$
8269 017364 042767 000400 162106 2$:      BIC      #BIT8,$TMP1      ;CLR ODD PARITY
8270      ;$TMP1 NOW HAS EVEN PARITY CHARACTER
8271 017372 000207      3$:      RTS      PC
8272
8273 017374 062716 000002      TRPREG: ADD      #2,(SP) ;ALLOW IT TO "CRUNCH" INTO HLT BACK
8274      ;IN MAIN PART OF THE PROGRAM
8275 017400 000002      RTI
8276      ;SAVE POINTER
      POINT=.
      .=100
      $CLKVEC      ;LKVEC HANDLER
      300           ;INTERRUPT HANDLER PRI
      .=140         ;BRKVEC
      170000        ;ODT START ADDRESS
      300           ;PRIORITY
      .=POINT       ;RESTORE POINTER
      $CLKVEC:      TYPE,CLKMES
      HALT
      CLKMES:      .ASCIZ <15><12>/LKVEC INTERRUPT - DISCONNECT LIC /
      (1) 000100 017402
      (1) 000102 000300
      (1) 000140 170000
      (1) 000142 000300
      (1) 017402 104401 017410
      (1) 017406 000000
      (1) 017410 005015 045514 042526
      (1) 017416 020103 047111 042524
      (1) 017424 051122 050125 020124
      (1) 017432 020055 044504 041523
      (1) 017440 047117 042516 052103
      (1) 017446 046040 041524 000040
8277      000001
8278      .END
  
```

;SPECIFIC VECTORS.

|         |        |        |       |
|---------|--------|--------|-------|
| AAA     | 003200 | 8150#  |       |
| ABASE = | 000000 | 8150   |       |
| ACDW1 = | 000000 | 8150   |       |
| ACDW2 = | 000000 | 8150   |       |
| ACPUOP= | 000000 | 8150   |       |
| ACTREG  | 001166 | 8150#* | 8185  |
| ADDW0 = | 000000 | 8150   |       |
| ADDW1 = | 000000 | 8150   |       |
| ADDW10= | 000000 | 8150   |       |
| ADDW11= | 000000 | 8150   |       |
| ADDW12= | 000000 | 8150   |       |
| ADDW13= | 000000 | 8150   |       |
| ADDW14= | 000000 | 8150   |       |
| ADDW15= | 000000 | 8150   |       |
| ADDW2 = | 000000 | 8150   |       |
| ADDW3 = | 000000 | 8150   |       |
| ADDW4 = | 000000 | 8150   |       |
| ADDW5 = | 000000 | 8150   |       |
| ADDW6 = | 000000 | 8150   |       |
| ADDW7 = | 000000 | 8150   |       |
| ADDW8 = | 000000 | 8150   |       |
| ADDW9 = | 000000 | 8150   |       |
| ADEVCT= | 000000 | 8150   |       |
| ADEVN = | 000000 | 8150   |       |
| ADRCNT= | 013655 | 8185#* |       |
| AENV =  | 000000 | 8150   |       |
| AENVN = | 000000 | 8150   |       |
| AFATAL= | 000000 | 8150   |       |
| AMADR1= | 000000 | 8150   |       |
| AMADR2= | 000000 | 8150   |       |
| AMADR3= | 000000 | 8150   |       |
| AMADR4= | 000000 | 8150   |       |
| AMAMS1= | 000000 | 8150   |       |
| AMAMS2= | 000000 | 8150   |       |
| AMAMS3= | 000000 | 8150   |       |
| AMAMS4= | 000000 | 8150   |       |
| AMSGAD= | 000000 | 8150   |       |
| AMSGLG= | 000000 | 8150   |       |
| AMSGTY= | 000000 | 8150   |       |
| AMTYP1= | 000000 | 8150   |       |
| AMTYP2= | 000000 | 8150   |       |
| AMTYP3= | 000000 | 8150   |       |
| AMTYP4= | 000000 | 8150   |       |
| APASS = | 000000 | 8150   |       |
| APRIOR= | 000000 | 8150   |       |
| APTCU=  | 000040 | 7010#  | 8185  |
| APTENV= | 000001 | 7010#  | 8185  |
| APTSIZ= | 000200 | 7010#  | 8150  |
| APTSPO= | 000100 | 7010#  | 8185  |
| ASWREG= | 000000 | 8150   |       |
| ATESTN= | 000000 | 8150   |       |
| AUNIT = | 000000 | 8150   |       |
| AUSWR = | 000000 | 8150   |       |
| AVECT1= | 000000 | 8150   |       |
| AVECT2= | 000000 | 8150   |       |
| BASEAD  | 001154 | 8150#* | 8185* |

















|                |        |       |       |       |       |       |       |       |       |       |       |       |       |  |
|----------------|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|--|
| SW5 = 000040   | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| SW6 = 000100   | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| SW7 = 000200   | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| SW8 = 000400   | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| SW9 = 001000   | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| SYNCNO 001146  | 8150#* |       |       |       |       |       |       |       |       |       |       |       |       |  |
| SYNEXT= 020000 | 8150#  | 8152  | 8153  | 8154  | 8155  | 8156  | 8157  | 8172  | 8173  | 8178  | 8179  | 8180  | 8181  |  |
| SYNINT= 030000 | 8150#  | 8182  | 8183  | 8184  |       |       |       |       |       |       |       |       |       |  |
| SYNSCH= 000020 | 8150#  | 8152  | 8153  | 8154  | 8155  | 8156  | 8157  | 8158  | 8159  | 8160  | 8161  | 8162  | 8163  |  |
|                | 8164   | 8165  | 8166  | 8167  | 8168  | 8169  | 8170  | 8171  | 8172  | 8173  | 8174  | 8175  | 8176  |  |
|                | 8177   | 8178  | 8179  | 8180  | 8181  | 8182  | 8183  | 8184  |       |       |       |       |       |  |
| SYSTST= 014000 | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TBITVE= 000014 | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TEMP 016200    | 8185#* |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TKVEC = 000060 | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TPVEC = 000064 | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TRAPVE= 000034 | 8150#* |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TRPREG 017374  | 8273#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TRTVEC= 000014 | 8150#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST1 003374    | 8150   | 8152# | 8185  |       |       |       |       |       |       |       |       |       |       |  |
| TST10 005230   | 8159#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST11 005370   | 8160#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST12 005530   | 8161#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST13 005670   | 8162#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST14 006040   | 8163#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST15 006210   | 8164#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST16 006360   | 8165#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST17 006530   | 8166#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST2 003606    | 8153#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST20 006700   | 8167#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST21 007050   | 8168#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST22 007220   | 8169#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST23 007370   | 8170#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST24 007554   | 8171#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST25 007740   | 8172#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST26 010074   | 8173#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST27 010230   | 8174#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST3 004020    | 8154#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST30 010400   | 8175#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST31 010550   | 8176#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST32 010720   | 8177#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST33 011070   | 8178#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST34 011224   | 8179#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST35 011360   | 8180#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST36 011514   | 8181#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST37 011650   | 8182#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST4 004232    | 8155#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST40 012020   | 8183#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST41 012170   | 8184#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST5 004444    | 8156#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST6 004656    | 8157#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TST7 005070    | 8158#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TTST 016322    | 8185#  |       |       |       |       |       |       |       |       |       |       |       |       |  |
| TXCSR 001726   | 8150#  | 8152* | 8153* | 8154* | 8155* | 8156* | 8157* | 8158* | 8159* | 8160* | 8161* | 8162* | 8163* |  |
|                | 8164*  | 8165* | 8166* | 8167* | 8168* | 8169* | 8170* | 8171* | 8172* | 8173* | 8174* | 8175* | 8176* |  |
|                | 8177*  | 8178* | 8179* | 8180* | 8181* | 8182* | 8183* | 8184* | 8214* | 8227* | 8233* | 8234* | 8235* |  |



|          |          |        |            |
|----------|----------|--------|------------|
| TXDBUF   | 001732   | 8150#  | 8218*      |
| TXDONE=  | 000200   | 8150#  |            |
| TXINTE=  | 000100   | 8150#  |            |
| TYPE =   | 104401   | 8150   | 8185# 8276 |
| TYPOC =  | 104402   | 8185#  |            |
| TYPON =  | 104404   | 8185#  |            |
| TYPOS =  | 104403   | 8185#  |            |
| USER =   | 000000   | 8150#  |            |
| VOID =   | 000001   | 8150#  |            |
| WRDCNT   | 014120   | 8185#* |            |
| ZERO     | 012422   | 8185#  |            |
| \$APTHD  | 002136   | 8150#  |            |
| \$ASTAT= | ***** U  | 7010   |            |
| \$ATYC   | 000024   | 7010#  |            |
| \$ATY1   | 000000   | 7010#  |            |
| \$ATY3   | 000006   | 7010#  | 8185       |
| \$ATY4   | 000016   | 7010#  | 8185       |
| \$AUTOB  | 001434   | 8150#  |            |
| \$BASE   | 001602   | 8150#  |            |
| \$BDADR  | 001422   | 8150#  |            |
| \$BDDAT  | 001426   | 8150#  |            |
| \$BELL   | 001516   | 8150#  | 8185       |
| \$CDW1   | 001606   | 8150#  |            |
| \$CDW2   | 001610   | 8150#  |            |
| \$CHARC  | 013334   | 8185#* |            |
| \$CKSWR= | ***** U  | 8185   |            |
| \$CLKVE  | 017402   | 8276#  |            |
| \$CMTAG  | 001400   | 8150#  |            |
| \$CM1 =  | 000006   | 8150#  |            |
| \$CM2 =  | 000014   | 8150#  |            |
| \$CM3 =  | 000006   | 8150#  |            |
| \$CM4 =  | 000006   | 8150#  |            |
| \$CPUOP  | 001554   | 8150#  |            |
| \$CRLF   | 001523   | 8150#  | 8185       |
| \$DDW0   | 001612   | 8150#  |            |
| \$DDW1   | 001614   | 8150#  |            |
| \$DDW10  | 001636   | 8150#  |            |
| \$DDW11  | 001640   | 8150#  |            |
| \$DDW12  | 001642   | 8150#  |            |
| \$DDW13  | 001644   | 8150#  |            |
| \$DDW14  | 001646   | 8150#  |            |
| \$DDW15  | 001650   | 8150#  |            |
| \$DDW2   | 001616   | 8150#  |            |
| \$DDW3   | 001620   | 8150#  |            |
| \$DDW4   | 001622   | 8150#  |            |
| \$DDW5   | 001624   | 8150#  |            |
| \$DDW6   | 001626   | 8150#  |            |
| \$DDW7   | 001630   | 8150#  |            |
| \$DDW8   | 001632   | 8150#  |            |
| \$DDW9   | 001634   | 8150#  |            |
| \$DEVCT  | 001536   | 8150#  |            |
| \$EVM    | 001604   | 8150#  |            |
| \$E      | = 000002 | 6838#  |            |
| \$ENDAD  | 012660   | 8150   | 8185#      |
| \$ENV    | 001546   | 7010   | 8150# 8185 |
| \$ENVM   | 001547   | 7010   | 8150# 8185 |

SEQ 0081

|          |         |        |       |
|----------|---------|--------|-------|
| SERFLG   | 001403  | 8150#* | 8185* |
| SERMAX   | 001415  | 8150#* | 8185* |
| \$ERROR  | 014174  | 8150   | 8185# |
| \$ERRPC  | 001416  | 8150#* | 8185* |
| \$ERRTB  | 001652  | 8150#  | 8185  |
| \$ERRTY  | 014414  | 8185#  |       |
| \$ERTTL  | 001412  | 8150#* | 8185* |
| \$ESCAP  | 001514  | 8150#* | 8185* |
| \$ETABL  | 001546  | 8150#  |       |
| \$ETEND  | 001652  | 8150#  |       |
| \$FATAL  | 001530  | 7010*  | 8150# |
| \$FFLG   | 000244  | 7010#* |       |
| \$FILLC  | 001456  | 8150#  | 8185  |
| \$FILLS  | 001455  | 8150#  | 8185  |
| \$GDADR  | 001420  | 8150#  |       |
| \$GDDAT  | 001424  | 8150#  |       |
| \$GTSWR= | ***** U | 8185   |       |
| \$HIBTS  | 002136  | 8150#  |       |
| \$ICNT   | 001404  | 8150#  | 8185* |
| \$INTAG  | 001435  | 8150#  |       |
| \$ITEMB  | 001414  | 8150#  | 8185* |
| \$LF     | 001524  | 8150#  | 8185  |
| \$LFLG   | 000243  | 7010#* |       |
| \$LPADR  | 001406  | 8150#* | 8185* |
| \$LPERR  | 001410  | 8150#* | 8185* |
| \$MADR1  | 001560  | 8150#  |       |
| \$MADR2  | 001564  | 8150#  |       |
| \$MADR3  | 001570  | 8150#  |       |
| \$MADR4  | 001574  | 8150#  |       |
| \$MAIL   | 001526  | 8150#  | 8185  |
| \$MAMS1  | 001556  | 8150#  |       |
| \$MAMS2  | 001562  | 8150#  |       |
| \$MAMS3  | 001566  | 8150#  |       |
| \$MAMS4  | 001572  | 8150#  |       |
| \$MBADR  | 002140  | 8150#  |       |
| \$MFLG   | 000242  | 7010#* |       |
| \$MSGAD  | 001542  | 7010*  | 8150# |
| \$MSGLG  | 001544  | 7010*  | 8150# |
| \$MSGTY  | 001526  | 7010*  | 8150# |
| \$MTYP1  | 001557  | 8150#  |       |
| \$MTYP2  | 001563  | 8150#  |       |
| \$MTYP3  | 001567  | 8150#  |       |
| \$MTYP4  | 001573  | 8150#  |       |
| \$MXCNT  | 016636  | 8185#  |       |
| \$N =    | 000000  | 6837#  | 8185# |
| \$NULL   | 001454  | 8150#  | 8185  |
| \$NWTST= | 000000  | 8152#  | 8153# |
|          |         | 8165#  | 8166# |
|          |         | 8178#  | 8179# |
|          |         | 8185#* |       |
|          |         | 8185#* |       |
|          |         | 8185#  |       |
|          |         | 8150#* | 8185* |
|          |         | 8150#  |       |
|          |         | 8150   | 8185# |
|          |         | 8150#  | 8185  |
| \$OCNT   | 014772  |        |       |
| \$OMODE  | 014774  |        |       |
| \$OVER   | 016616  |        |       |
| \$PASS   | 001534  |        |       |
| \$PASTM  | 002144  |        |       |
| \$PWDRD  | 014776  |        |       |
| \$QUES   | 001522  |        |       |
|          |         | 8154#  | 8155# |
|          |         | 8167#  | 8168# |
|          |         | 8180#  | 8181# |
|          |         |        | 8182# |
|          |         | 8156#  | 8157# |
|          |         | 8169#  | 8170# |
|          |         | 8183#  | 8184# |
|          |         | 8158#  | 8159# |
|          |         | 8171#  | 8172# |
|          |         | 8173#  | 8174# |
|          |         | 8160#  | 8161# |
|          |         | 8175#  | 8176# |
|          |         | 8162#  | 8163# |
|          |         | 8177#  | 8178# |
|          |         |        | 8164# |
|          |         |        | 8165# |
|          |         |        | 8166# |
|          |         |        | 8167# |
|          |         |        | 8168# |
|          |         |        | 8169# |
|          |         |        | 8170# |
|          |         |        | 8171# |
|          |         |        | 8172# |
|          |         |        | 8173# |
|          |         |        | 8174# |
|          |         |        | 8175# |
|          |         |        | 8176# |
|          |         |        | 8177# |
|          |         |        | 8178# |
|          |         |        | 8179# |
|          |         |        | 8180# |
|          |         |        | 8181# |
|          |         |        | 8182# |
|          |         |        | 8183# |
|          |         |        | 8184# |
|          |         |        | 8185# |
|          |         |        | 8186# |
|          |         |        | 8187# |
|          |         |        | 8188# |
|          |         |        | 8189# |
|          |         |        | 8190# |
|          |         |        | 8191# |
|          |         |        | 8192# |
|          |         |        | 8193# |
|          |         |        | 8194# |
|          |         |        | 8195# |
|          |         |        | 8196# |
|          |         |        | 8197# |
|          |         |        | 8198# |
|          |         |        | 8199# |
|          |         |        | 8200# |
|          |         |        | 8201# |
|          |         |        | 8202# |
|          |         |        | 8203# |
|          |         |        | 8204# |
|          |         |        | 8205# |
|          |         |        | 8206# |
|          |         |        | 8207# |
|          |         |        | 8208# |
|          |         |        | 8209# |
|          |         |        | 8210# |
|          |         |        | 8211# |
|          |         |        | 8212# |
|          |         |        | 8213# |
|          |         |        | 8214# |
|          |         |        | 8215# |
|          |         | </     |       |



CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 62-9

CNDUSA.M11 30-OCT-82 10:55

## CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]











CNDUS-A MACY11 30(1046) 14-DEC-82 09:59 PAGE 63-2  
 CNDUSA.M11 30-OCT-82 10:55 CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0086

|        |       |       |       |
|--------|-------|-------|-------|
| .SACT1 | 5064# | 6496# | 8150  |
| .SAPT8 | 5109# | 6496# | 8150# |
| .SAPTH | 5370# | 6496# | 8150  |
| .SAPTY | 5547# | 6496# | 7010  |
| .SASTA | 5417# |       |       |
| .SCATC | 932#  | 6494# |       |
| .SCMTA | 1047# | 6494# | 8150  |
| .SDB2D | 4686# |       |       |
| .SDB20 | 4812# |       |       |
| .SDIV  | 4587# |       |       |
| .SEOP  | 2214# | 6494# |       |
| .SERRO | 2700# | 6495# | 8185  |
| .SERRT | 2896# | 6495# | 8185  |
| .SMULT | 4523# |       |       |
| .SPOWE | 4229# | 6495# |       |
| .SRAND | 4307# |       |       |
| .SRDDE | 3891# |       |       |
| .SRDOC | 3797# |       |       |
| .SREAD | 3395# |       |       |
| .SR2AZ | 4958# |       |       |
| .SSAVE | 3969# |       |       |
| .SSB2D | 4771# |       |       |
| .SSB20 | 4874# |       |       |
| .SSCOP | 2454# | 6495# | 8185  |
| .SSIZE | 4361# |       |       |
| .SSUPR | 4913# |       |       |
| .STRAP | 4073# | 6495# | 8185  |
| .STYPB | 3287# |       |       |
| .STYPD | 3209# |       |       |
| .STYPE | 2985# | 6494# | 8185  |
| .STYPO | 3112# | 6495# | 8185  |
| .S40CA | 972#  |       |       |

. ABS. 017454 000

ERRORS DETECTED: 0

CNDUSA,CNDUSA/CRF/NL:TOC=CNMAC2.SML,CNDUS1.M11,CNDUS2.M11,CNDUSA.M11

RUN-TIME: 18 17 1 SECONDS

RUN-TIME RATIO: 90/37=2.4

CORE USED: 43K (86 PAGES)