

DUV-11

DUV11 OFFLINE RECEIVER TESTS
MD-11-DZDUR-B

EP-DZDUR-B-DL-B

COPYRIGHT © 1977

FICHE 1 OF 1

DEC 1977

digital

MADE IN USA

.REM *

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

PRODUCT CODE: MAINDEC-11-DZDUR-B-D

PRODUCT NAME: DUV11 OFFLINE RECEIVER TESTS

RELEASE DATE: NOV 1977

MAINTAINER : DIAGNOSTICS

*
.REM *

COPYRIGHT (C) 1977
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.

*

. REM *

GENERAL DESCRIPTION

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

1. THE DUV11 OFFLINE RECEIVER TESTS VERIFY THAT THE RECEIVER CHIP/LOGIC WORKS PROPERLY

* . REM *

2. REQUIREMENTS

* . REM *

PDP-11/03 COMPUTER (LSI)

DUV11 SYNCHRONOUS/ISOCRONOUS OPTION

ONE CONSOLE TELETYPE OR EQUIVALENT

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

3. LOADING PROCEDURE

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

STARTING ADDRESS FOR ABSOLUTE LOADER

4K	017500
8K	037500
12K	057500
16K	077500
20K	117500
24K	137500
28K	157500

4. STARTING PROCEDURE

- 4.1 CONTROL SWITCH SETTINGS

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

NOTE: RUNNING UNDER APT-11, THERE IS A USER SWITCH REGISTER CALLED "SUSWR". IN ORDER TO BE FLEXIBLE ON THE AVAILABILITY OF THE H315 CONNECTOR, ONE BIT PASSES STATUS TO APT-11. BIT 0 IN SUSWR 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 DZDUR-B TAPE B" (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 ACTPEG:
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 11/03 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: 160010
VECTOR ADDRESS- DURIV: 770
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 SECTION TESTING
OF THE DEVICE
SEE LISTING FOR DETAILS

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

11. LISTINGS

*
. REM *

*
. REM *

*

523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556

000001

STN=1

```
557 . ENABLE ABS
558
559 ; DUV11 DZDUR-B TAPE B
560 ; COPYRIGHT 1977, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
561
562 ; STARTING PROCEDURE
563 ; TYPE 200G
564 ; PROGRAM WILL TYPE "DUV11 DZDUR-B TAPE B "
565 ; PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
566 ; AT THE END OF A PASS, PROGRAM WILL TYPE "END OF PASS TAPE B"
567 ; AND THEN RESUME TESTING
568
569 . SBTTL BASIC DEFINITIONS
570
571 ; *INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
572 001100 STACK= 1100
573 . EQUIV EMT,ERROR ; BASIC DEFINITION OF ERROR CALL
574 . EQUIV IOT,SCOPE ; BASIC DEFINITION OF SCOPE CALL
575
576 ; *MISCELLANEOUS DEFINITIONS
577 000011 HT= 11 ; CODE FOR HORIZONTAL TAB
578 000012 LF= 12 ; CODE FOR LINE FEED
579 000015 CR= 15 ; CODE FOR CARRIAGE RETURN
580 000200 CRLF= 200 ; CODE FOR CARRIAGE RETURN-LINE FEED
581 177776 PS= 177776 ; PROCESSOR STATUS WORD
582 . EQUIV PS,PSW
583 177774 STKLMT= 177774 ; STACK LIMIT REGISTER
584 177772 PIRQ= 177772 ; PROGRAM INTERRUPT REQUEST REGISTER
585 177570 DSWR= 177570 ; HARDWARE SWITCH REGISTER
586 177570 DDISP= 177570 ; HARDWARE DISPLAY REGISTER
587
588 ; *GENERAL PURPOSE REGISTER DEFINITIONS
589 000000 R0= %0 ; GENERAL REGISTER
590 000001 R1= %1 ; GENERAL REGISTER
591 000002 R2= %2 ; GENERAL REGISTER
592 000003 R3= %3 ; GENERAL REGISTER
593 000004 R4= %4 ; GENERAL REGISTER
594 000005 R5= %5 ; GENERAL REGISTER
595 000006 R6= %6 ; GENERAL REGISTER
596 000007 R7= %7 ; GENERAL REGISTER
597 000006 SP= %6 ; STACK POINTER
598 000007 PC= %7 ; PROGRAM COUNTER
599
600 ; *PRIORITY LEVEL DEFINITIONS
601 000000 PR0= 0 ; PRIORITY LEVEL 0
602 000040 PR1= 40 ; PRIORITY LEVEL 1
603 000100 PR2= 100 ; PRIORITY LEVEL 2
604 000140 PR3= 140 ; PRIORITY LEVEL 3
605 000200 PR4= 200 ; PRIORITY LEVEL 4
606 000240 PR5= 240 ; PRIORITY LEVEL 5
607 000300 PR6= 300 ; PRIORITY LEVEL 6
608 000340 PR7= 340 ; PRIORITY LEVEL 7
609
610 ; *"SWITCH REGISTER" SWITCH DEFINITIONS
611 100000 SW15= 100000
612 040000 SW14= 40000
```

613	020000	SW13=	20000
614	010000	SW12=	10000
615	004000	SW11=	4000
616	002000	SW10=	2000
617	001000	SW09=	1000
618	000400	SW08=	400
619	000200	SW07=	200
620	000100	SW06=	100
621	000040	SW05=	40
622	000020	SW04=	20
623	000010	SW03=	10
624	000004	SW02=	4
625	000002	SW01=	2
626	000001	SW00=	1
627		. EQUIV	SW09, SW9
628		. EQUIV	SW08, SW8
629		. EQUIV	SW07, SW7
630		. EQUIV	SW06, SW6
631		. EQUIV	SW05, SW5
632		. EQUIV	SW04, SW4
633		. EQUIV	SW03, SW3
634		. EQUIV	SW02, SW2
635		. EQUIV	SW01, SW1
636		. EQUIV	SW00, SW0

; *DATA BIT DEFINITIONS (BIT00 TO BIT15)

638		BIT15=	100000
639	100000	BIT14=	40000
640	040000	BIT13=	20000
641	020000	BIT12=	10000
642	010000	BIT11=	4000
643	004000	BIT10=	2000
644	002000	BIT09=	1000
645	001000	BIT08=	400
646	000400	BIT07=	200
647	000200	BIT06=	100
648	000100	BIT05=	40
649	000040	BIT04=	20
650	000020	BIT03=	10
651	000010	BIT02=	4
652	000004	BIT01=	2
653	000002	BIT00=	1
654	000001	. EQUIV	BIT09, BIT9
655		. EQUIV	BIT08, BIT8
656		. EQUIV	BIT07, BIT7
657		. EQUIV	BIT06, BIT6
658		. EQUIV	BIT05, BIT5
659		. EQUIV	BIT04, BIT4
660		. EQUIV	BIT03, BIT3
661		. EQUIV	BIT02, BIT2
662		. EQUIV	BIT01, BIT1
663		. EQUIV	BIT00, BIT0

; *BASIC "CPU" TRAP VECTOR ADDRESSES

666		ERRVEC=	4	;; TIME OUT AND OTHER ERRORS
667	000004	RESVEC=	10	;; RESERVED AND ILLEGAL INSTRUCTIONS
668	000010			

DZDUR-B MACY11 30(1046) 21-SEP-77 09:08 PAGE 16
DZDURB.M11 31-MAY-77 09:48 BASIC DEFINITIONS

SEQ 0014

669	000014	TBITVEC=14	:: "T" BIT
670	000014	TRTVEC= 14	:: TRACE TRAP
671	000014	BPTVEC= 14	:: BREAKPOINT TRAP (BPT)
672	000020	IOTVEC= 20	:: INPUT/OUTPUT TRAP (IOT) **SCOPE**
673	000024	PWRVEC= 24	:: POWER FAIL
674	000030	EMTVEC= 30	:: EMULATOR TRAP (EMT) **ERROR**
675	000034	TRAPVEC=34	:: "TRAP" TRAP
676	000060	TKVEC= 60	:: TTY KEYBOARD VECTOR
677	000064	TPVEC= 64	:: TTY PRINTER VECTOR
678	000240	PIRQVEC=240	:: PROGRAM INTERRUPT REQUEST VECTOR

```
679                                     ; STANDARD INTERRUPT VECTORS
680
681
682                                     . =174
683 000174 000000  DISPREG: 0
684 000176 000000  SWREG: 0
685                                     . =200
686 000200 000167 001746  JMP . START ; GO TO START OF PROGRAM
687
688
689
690                                     . =1100
691 001100 000000  . WORD 0
692 001102 177570  LIGHTS: 177570
693
694
695
696                                     ; PROGRAM CONTROL PARAMETERS
697
698 001104 000000  RETURN: 0
699 001106 000000  NEXT: 0 ; ADDRESS OF NEXT TEST TO BE EXECUTED
700 001110 000000  LOCK: 0 ; ADDRESS FOR LOCK ON CURRENT DATA
701 001112 000000  PASCNT: 0 ; ADDRESS CONTAINING PASS COUNT
702 001114 000000  ERRCNT: 0 ; ERROR COUNT
703 001116 000000  SAVSP: 0 ; STACK POINTER STORAGE
704
705                                     ; PROGRAM VARIABLES
706
707 001120 000020  HOLD: 20 ; TEMPORARY STORAGE=DELAY TIME FOR CABLES
708 001122 000000  SHIFT: 0 ; TEMPORARY STORAGE= # OF SHIFTS PER CHAR
709 001124 000000  COUNT: 0 ; TEMPORARY STORAGE= # OF TIMES A CHAR WILL BE SENT
710 001126 000000  SAVPC: 0 ; PROGRAM COUNTER STORAGE
711 001130 000000  HLD0: 0
712 001132 000000  HLD1: 0
713 001134 000000  HLD2: 0
714 001136 000000  HLD3: 0
715 001140 000000  HLD4: 0
716 001142 000000  HLD5: 0
717 001144 000000  HLD6: 0
718
```

```
719 ;PROGRAM CONVERSATIONAL PARAMETERS
720 001146 377 SYNCNO: . BYTE 377 ;# OF SYNC CHARS REQ'D FOR SYNC'ZATION
721 001147 377 SEXMIT: . BYTE 377 ;SEC XMIT JUMPER "IN"
722 001150 377 SEREC: . BYTE 377 ;SEC REC JUMPER "IN"
723 001151 377 OPTCLR: . BYTE 377 ;OPTIONAL JUMPER CLR "IN"
724 001152 000 MULTD: . BYTE 0 ;NO MULTIPLE DEVICE FLAG
725 001153 377 JMRBY: . BYTE 377 ;EXTERNAL MODEM BYPASS JUMPER "IN"
726 . EVEN
727
728 ;PROGRAM MULTIPLE DEVICE PARAMETERS
729 001154 000000 BASEADD: 0 ;PROG CONTROLLED 1ST DEVICE ADDR
730 001156 000000 KEEPADD: 0 ;SAVED 1ST DEVICE ADDR
731 001160 000000 LASTADD: 0 ;LAST DEVICE RXCSR ADDR
732 001162 000000 BASEIV: 0 ;PROG CONTROLLED IV
733 001164 000000 KEEPIV: 0 ;SAVED INTR VECTOR
734 001166 000000 ACTREG: 0 ;ACTIVE REGISTER , , ,MODIFY THIS
735 ;LOCATION TO DISQUALIFY OR QUALIFY
736 ;DEVICES (1= RUN, , 0= DON'T RUN)
737 001170 000000 ROTADD: 0 ;ROTATING POINTER FOR ACTREG. POINTS
738 ;TO DEVICE PRESENTLY UNDER TEST WHEN RUNNING MULTIPLE DEVICES
739
740 ;PROGRAM CONTROL FLAGS
741
742 001172 000 INIFLG: . BYTE 0 ;PROGRAM INITIALIZATION FLAG
743 001173 000 STFLG: . BYTE 0 ;TEST START FLAG
744 001174 000 LOKFLG: . BYTE 0 ;LOCK ON CURRENT TEST FLAG
745 . EVEN
746 001400 . =1400
747
748
```

```
749
750
751
752 ; INSTRUCTION DEFINITIONS
753
754 005746 PUSH1SP=5746 ; DECREMENT PROCESSOR STACK 1 WORD =TST -(SP)
755 005726 POP1SP=5726 ; INCREMENT PROCESSOR STACK 1 WORD =TST (SP)+
756 010046 PUSHRO=10046 ; SAVE RO ON STACK =MOV RO,-(SP)
757 012600 POPRO=12600 ; RESTORE RO FROM STACK =MOV (SP)+,RO
758 024646 PUSH2SP=24646 ; DECREMENT STACK TWICE =CMP -(SP),-(SP)
759 022626 POP2SP=22626 ; INCREMENT STACK TWICE =CMP (SP)+,(SP)+
760 ; REGISTER DEFINITIONS
761 ; RXCSR BIT DEFINITIONS
762 100000 DSC=BIT15 ; DATA SET CHANGE
763 040000 RING=BIT14 ; RING
764 020000 CTS=BIT13 ; CLR TO SEND
765 010000 CARDET=BIT12 ; CARRIER DETECT
766 004000 REACT=BIT11 ; REC ACTIVE
767 002000 SRD=BIT10 ; SEC REC DATA
768 001000 DSR=BIT9 ; DATA SET RDY
769 000400 STPSYN=BIT8 ; STRIP SYNC
770 000200 RXDONE=BIT7 ; REC DONE
771 000100 RINTEN=BIT6 ; REC INTR ENABLE
772 000040 DSINTE=BIT5 ; DSC INTR ENABLE
773 000020 SYN SCH=BIT4 ; SYNC SEARCH
774 000010 STD=BIT3 ; SEC XMIT DATA
775 000004 RTS=BIT2 ; REQ TO SEND
776 000002 DTR=BIT1 ; DATA TERM RDY
777 000001 VOID=BIT0
778 ; RXDBUF BIT DEFINITIONS
779 100000 RXERR=BIT15 ; REC ERROR
780 040000 OVERRUN=BIT14 ; OVERRUN
781 020000 FRMERR=BIT13 ; FRAME ERROR
782 010000 PARER=BIT12 ; PARITY ERROR
783 ; PARCSR BIT DEFINITIONS
784 001000 PAREN=BIT9 ; PARITY ENABLE
785 000400 EVPAR=BIT8 ; EVEN PARITY SENSE
786 ; PARCSR WRD DEFINITIONS
787 030000 SYNINT=30000 ; SYNC EXTERNAL MODE
788 020000 SYNEXT=20000 ; SYNC INTERNAL MODE
789 000000 ISYMOD=0 ; ISOC MODE
790 000000 FIVE=0 ; WORD LENGTH 5 BITS
791 002000 SIX=2000 ; WORD LENGTH 6 BITS
792 004000 SEVEN=4000 ; WORD LENGTH 7 BITS
793 006000 EIGHT=6000 ; WORD LENGTH 8 BITS
794 000000 NOPAR=0 ; NO PARITY
795 001000 ODDPAR=1000 ; ODD PARITY
796 001400 EVEPAR=1400 ; EVEN PARITY
797 ; TXCSR BIT DEFINITIONS
798 100000 DNA=BIT15 ; DATA NOT AVAILABLE
799 040000 MTDATA=BIT14 ; MAINT DATA
800 020000 CLK=BIT13 ; CLK
801 002000 BITW=BIT10 ; BIT WINDOW
802 000400 MRESET=BIT8 ; MASTER RESET
803 000200 TXDONE=BIT7 ; XMIT DONE
804 000100 TXINTE=BIT6 ; XMIT INTR ENABLE
```

805	000040	DNAINTE=BIT5	;DNA INTR ENAB
806	000020	SEND=BIT4	;SEND
807	000010	HDXEN=BIT3	;HDX/FDX
808	000001	BREAK=BIT0	;BREAK
809		;TXCSR WRD DEFINITIONS	
810	000000	USER=0	;USER MODE
811	004000	MINT=4000	;MAINT INT MODE
812	010000	MEXT=10000	;MAINT EXT MODE
813	014000	SYSTST=14000	;SYSTEM TEST MODE

```

814 . SBTTL COMMON TAGS
815
816 ;*****
817 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
818 ;*USED IN THE PROGRAM.
819
820 001400
821 001400 001400 SCMTAG: =. ; START OF COMMON TAGS
822 001400 000000 ;
823 001402 000 STSTNM: WORD 0 ; CONTAINS THE TEST NUMBER
824 001403 000 SERFLG: BYTE 0 ; CONTAINS ERROR FLAG
825 001404 000000 SICNT: WORD 0 ; CONTAINS SUBTEST ITERATION COUNT
826 001406 000000 SLPADR: WORD 0 ; CONTAINS SCOPE LOOP ADDRESS
827 001410 000000 SLPERR: WORD 0 ; CONTAINS SCOPE RETURN FOR ERRORS
828 001412 000000 SERTTL: WORD 0 ; CONTAINS TOTAL ERRORS DETECTED
829 001414 000 SITEMB: BYTE 0 ; CONTAINS ITEM CONTROL BYTE
830 001415 001 SERMAX: BYTE 1 ; CONTAINS MAX. ERRORS PER TEST
831 001416 000000 SERRPC: WORD 0 ; CONTAINS PC OF LAST ERROR INSTRUCTION
832 001420 000000 SGDADR: WORD 0 ; CONTAINS ADDRESS OF 'GOOD' DATA
833 001422 000000 SBDADR: WORD 0 ; CONTAINS ADDRESS OF 'BAD' DATA
834 001424 000000 SGDDAT: WORD 0 ; CONTAINS 'GOOD' DATA
835 001426 000000 SBDDAT: WORD 0 ; CONTAINS 'BAD' DATA
836 001430 000000 ;
837 001432 000000 ; WORD 0 ; RESERVED--NOT TO BE USED
838 001434 000 SAUTOB: BYTE 0 ; AUTOMATIC MODE INDICATOR
839 001435 000 SINTAG: BYTE 0 ; INTERRUPT MODE INDICATOR
840 001436 000000 ; WORD 0
841 001440 177570 SWR: WORD DSWR ; ADDRESS OF SWITCH REGISTER
842 001442 177570 DISPLAY: WORD DDISP ; ADDRESS OF DISPLAY REGISTER
843 001444 177560 $TKS: 177560 ; TTY KBD STATUS
844 001446 177562 $TKB: 177562 ; TTY KBD BUFFER
845 001450 177564 $TPS: 177564 ; TTY PRINTER STATUS REG. ADDRESS
846 001452 177566 $TPB: 177566 ; TTY PRINTER BUFFER REG. ADDRESS
847 001454 000 $NULL: BYTE 0 ; CONTAINS NULL CHARACTER FOR FILLS
848 001455 002 $FILLS: BYTE 2 ; CONTAINS # OF FILLER CHARACTERS REQUIRED
849 001456 012 $FILLC: BYTE 12 ; INSERT FILL CHARS. AFTER A "LINE FEED"
850 001457 000 $TPFLG: BYTE 0 ; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
851 001460 000000 $REGAD: WORD 0 ; CONTAINS THE ADDRESS FROM
852 ; WHICH ($REGO) WAS OBTAINED
853 001462 000000 $REGO: WORD 0 ; CONTAINS (($REGAD)+0)
854 001464 000000 $REG1: WORD 0 ; CONTAINS (($REGAD)+2)
855 001466 000000 $REG2: WORD 0 ; CONTAINS (($REGAD)+4)
856 001470 000000 $REG3: WORD 0 ; CONTAINS (($REGAD)+6)
857 001472 000000 $REG4: WORD 0 ; CONTAINS (($REGAD)+10)
858 001474 000000 $REG5: WORD 0 ; CONTAINS (($REGAD)+12)
859 001476 000000 $TMP0: WORD 0 ; USER DEFINED
860 001500 000000 $TMP1: WORD 0 ; USER DEFINED
861 001502 000000 $TMP2: WORD 0 ; USER DEFINED
862 001504 000000 $TMP3: WORD 0 ; USER DEFINED
863 001506 000000 $TMP4: WORD 0 ; USER DEFINED
864 001510 000000 $TMP5: WORD 0 ; USER DEFINED
865 001512 000000 $TIMES: 0 ; MAX. NUMBER OF ITERATIONS
866 001514 000000 $ESCAPE: 0 ; ESCAPE ON ERROR ADDRESS
867 001516 177607 000377 $BELL: ASCII <207><377><377> ; CODE FOR BELL
868 001522 077 $QUES: ASCII /?/ ; QUESTION MARK
869 001523 015 $CRLF: ASCII <15> ; CARRIAGE RETURN
  
```

```
870 001524 000012 $LF: .ASCIZ <12> ;:LINE FEED
871 ;:*****
872 .SBTTL APT MAILBOX-ETABLE
873
874 ;:*****
875 .EVEN
876 001526 $MAIL: ;:APT MAILBOX
877 001526 000000 $MSGTY: .WORD AMSGTY ;:MESSAGE TYPE CODE
878 001530 000000 $FATAL: .WORD AFATAL ;:FATAL ERROR NUMBER
879 001532 000000 $TESTN: .WORD ATESTN ;:TEST NUMBER
880 001534 000000 $PASS: .WORD APASS ;:PASS COUNT
881 001536 000000 $DEVCT: .WORD ADEVCT ;:DEVICE COUNT
882 001540 000000 $UNIT: .WORD AUNIT ;:I/O UNIT NUMBER
883 001542 000000 $MSGAD: .WORD AMSGAD ;:MESSAGE ADDRESS
884 001544 000000 $MSGLG: .WORD AMSGLG ;:MESSAGE LENGTH
885 001546 $ETABLE: ;:APT ENVIRONMENT TABLE
886 001546 000 $ENV: .BYTE AENV ;:ENVIRONMENT BYTE
887 001547 000 $ENVM: .BYTE AENVM ;:ENVIRONMENT MODE BITS
888 001550 000000 $SWREG: .WORD ASWREG ;:APT SWITCH REGISTER
889 001552 000000 $USWR: .WORD AUSWR ;:USER SWITCHES
890 001554 000000 $CPUOP: .WORD ACPUOP ;:CPU TYPE, OPTIONS
891 ;* BITS 15-11=CPU TYPE
892 ;* 11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
893 ;* 11/70=06, PDQ=07, Q=10
894 ;* BIT 10=REAL TIME CLOCK
895 ;* BIT 9=FLOATING POINT PROCESSOR
896 ;* BIT 8=MEMORY MANAGEMENT
897 001556 000 $MAMS1: .BYTE AMAMS1 ;:HIGH ADDRESS, M. S. BYTE
898 001557 000 $MTYP1: .BYTE AMTYP1 ;:MEM. TYPE, BLK#1
899 ;* MEM. TYPE BYTE -- (HIGH BYTE)
900 ;* 900 NSEC CORE=001
901 ;* 300 NSEC BIPOLAR=002
902 ;* 500 NSEC MOS=003
903 001560 000000 $MADR1: .WORD AMADR1 ;:HIGH ADDRESS, BLK#1
904 ;* MEM. LAST ADDR. =3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
905 001562 000 $MAMS2: .BYTE AMAMS2 ;:HIGH ADDRESS, M. S. BYTE
906 001563 000 $MTYP2: .BYTE AMTYP2 ;:MEM. TYPE, BLK#2
907 001564 000000 $MADR2: .WORD AMADR2 ;:MEM. LAST ADDRESS, BLK#2
908 001566 000 $MAMS3: .BYTE AMAMS3 ;:HIGH ADDRESS, M. S. BYTE
909 001567 000 $MTYP3: .BYTE AMTYP3 ;:MEM. TYPE, BLK#3
910 001570 000000 $MADR3: .WORD AMADR3 ;:MEM. LAST ADDRESS, BLK#3
911 001572 000 $MAMS4: .BYTE AMAMS4 ;:HIGH ADDRESS, M. S. BYTE
912 001573 000 $MTYP4: .BYTE AMTYP4 ;:MEM. TYPE, BLK#4
913 001574 000000 $MADR4: .WORD AMADR4 ;:MEM. LAST ADDRESS, BLK#4
914 001576 000000 $VECT1: .WORD AVECT1 ;:INTERRUPT VECTOR#1, BUS PRIORITY#1
915 001600 000000 $VECT2: .WORD AVECT2 ;:INTERRUPT VECTOR#2, BUS PRIORITY#2
916 001602 000000 $BASE: .WORD ABASE ;:BASE ADDRESS OF EQUIPMENT UNDER TEST
917 001604 000000 $DEVN: .WORD ADEVN ;:DEVICE MAP
918 001606 000000 $CDW1: .WORD ACDW1 ;:CONTROLLER DESCRIPTION WORD#1
919 001610 000000 $CDW2: .WORD ACDW2 ;:CONTROLLER DESCRIPTION WORD#2
920 001612 000000 $DDW0: .WORD ADDW0 ;:DEVICE DESCRIPTOR WORD#0
921 001614 000000 $DDW1: .WORD ADDW1 ;:DEVICE DESCRIPTOR WORD#1
922 001616 000000 $DDW2: .WORD ADDW2 ;:DEVICE DESCRIPTOR WORD#2
923 001620 000000 $DDW3: .WORD ADDW3 ;:DEVICE DESCRIPTOR WORD#3
924 001622 000000 $DDW4: .WORD ADDW4 ;:DEVICE DESCRIPTOR WORD#4
925 001624 000000 $DDW5: .WORD ADDW5 ;:DEVICE DESCRIPTOR WORD#5
```


942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997

005746
005726
010046
012600
024646
022626

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

100000
040000
020000
010000

001000
000400

030000
020000
000000
000000
002000
004000
006000
000000
001000
0J1400

100000
040000
020000
002000
000400
000200
000100

; INSTRUCTION DEFINITIONS

PUSH1SP=5746 ; DECREMENT PROCESSOR STACK 1 WORD =TST -(SP)
POP1SP=5726 ; INCREMENT PROCESSOR STACK 1 WORD =TST (SP)+
PUSHRO=10046 ; SAVE RO ON STACK =MOV RO,-(SP)
POPPO=12600 ; RESTORE RO FROM STACK =MOV (SP)+,RO
PUSH2SP=24646 ; DECREMENT STACK TWICE =CMP -(SP),-(SP)
POP2SP=22626 ; INCREMENT STACK TWICE =CMP (SP)+,(SP)+

; REGISTER DEFINITIONS

; RXCSR BIT DEFINITIONS

DSC=BIT15 ; DATA SET CHANGE
RING=BIT14 ; RING
CTS=BIT13 ; CLR TO SEND
CARDET=BIT12 ; CARRIER DETECT
REACT=BIT11 ; REC ACTIVE
SRD=BIT10 ; SEC REC DATA
DSR=BIT9 ; DATA SET RDY
STPSYN=BIT8 ; STRIP SYNC
RXDONE=BIT7 ; REC DONE
RINTEN=BIT6 ; REC INTR ENABLE
DSINTE=BIT5 ; DSC INTR ENABLE
SYNSCH=BIT4 ; SYNC SEARCH
STD=BIT3 ; SEC XMIT DATA
RTS=BIT2 ; REQ TO SEND
DTR=BIT1 ; DATA TERM RDY
VOID=BIT0

; RXDBUF BIT DEFINITIONS

RXERR=BIT15 ; REC ERROR
OVRUN=BIT14 ; OVERRUN
FRMERR=BIT13 ; FRAME ERROR
PARER=BIT12 ; PARITY ERROR

; PARCSR BIT DEFINITIONS

PAREN=BIT9 ; PARITY ENABLE
EVPAR=BIT8 ; EVEN PARITY SENSE

; PARCSR WRD DEFINITIONS

SYNINT=30000 ; SYNC EXTERNAL MODE
SYNEXT=20000 ; SYNC INTERNAL MODE
ISYMOD=0 ; ISOC MODE
FIVE=0 ; WORD LENGTH 5 BITS
SIX=2000 ; WORD LENGTH 6 BITS
SEVEN=4000 ; WORD LENGTH 7 BITS
EIGHT=6000 ; WORD LENGTH 8 BITS
NOPAR=0 ; NO PARITY
ODDPAR=1000 ; ODD PARITY
EVEPAR=1400 ; EVEN PARITY

; TXCSR BIT DEFINITIONS

DNA=BIT15 ; DATA NOT AVAILA6LE
MTDATA=BIT14 ; MAINT DATA
CLK=BIT13 ; CLK
BITW=BIT10 ; BIT WINDOW
MRESET=BIT8 ; MASTER RESET
TXDONE=BIT7 ; XMIT DONE
TXINTE=BIT6 ; XMIT INTR ENABLE

998	000040	DNAINTE=BIT5	;DNA INTR ENAB
999	000020	SEND=BIT4	;SEND
1000	000010	HDXEN=BIT3	;HDX/FDX
1001	000001	BREAK=BIT0	;BREAK
1002			;TXCSR WRD DEFINITIONS
1003	000000	USER=0	;USER MODE
1004	004000	MINT=4000	;MAINT INT MODE
1005	010000	MEXT=10000	;MAINT EXT MODE
1006	014000	SYSTST=14000	;SYSTEM TEST MODE

1007
 1008
 1009
 1010
 1011
 1012
 1013
 1014
 1015
 1016
 1017
 1018
 1019
 1020
 1021
 1022
 1023
 1024
 1025
 1026
 1027
 1028
 1029
 1030
 1031
 1032
 1033
 1034
 1035
 1036
 1037
 1038
 1039
 1040
 1041
 1042
 1043
 1044
 1045
 1046
 1047
 1048
 1049
 1050
 1051
 1052
 1053
 1054
 1055
 1056
 1057
 1058
 1059
 1060
 1061
 1062

001652
 001652 001762
 001654 002067
 001656 002116
 001660 002132
 001662 002022
 001664 002067
 001666 002116
 001670 002132
 001672 002043
 001674 002067
 001676 002116
 001700 002132
 001702 001746
 001704 000000
 001706 002126
 001710 002132
 160010
 160011
 160012
 160013
 160012
 160013
 160014
 160015
 160016
 160017
 000770
 000772
 000774
 000776
 020040 051105 047522
 020122 041520 000040
 020040 047503 050115
 051101 051511 047117
 042440 051122 051117
 047440 020116 042522

.SBTTL ERROR POINTER TABLE

;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
 ;*LOCATION SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
 ;*NOTE1: IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC).
 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

;* EM ;;POINTS TO THE ERROR MESSAGE
 ;* DH ;;POINTS TO THE DATA HEADER
 ;* DT ;;POINTS TO THE DATA
 ;* DF ;;POINTS TO THE DATA FORMAT

SERRTB:

;ERROR TABLE

EM1	;ERROR 1	REGISTER ERROR
DH1		
DT1		
DF1		
EM2	;ERROR 2	RECEIVER ERROR
DH1		
DT1		
DF1		
EM3	;ERROR 3	TRANSMITTER ERROR
DH1		
DT1		
DF1		
EM4	;ERROR 4	BIT ERROR (GENERAL)
0		
DT4		
DF1		

;DEFAULT DU ADDRESSES

RXCSR: 160010
 HRXCSR: 160011
 RXDBUF: 160012
 HRXDBUF: 160013
 PARCSR: 160012
 HPARCSR: 160013
 TXCSR: 160014
 HTXCSR: 160015
 TXDBUF: 160016
 HTXDBUF: 160017

;DEFAULT DU VECTORS

DURIV: 770 ;REC INTR VECTOR
 DURIS: 772 ;REC INTR STATUS
 DUTIV: 774 ;XMIT INTR VECTOR
 DUTIS: 776 ;XMIT INTR STATUS

;ERROR MESSAGES

EM4: .ASCIZ / ERROR PC /
 EM1: .ASCIZ / COMPARISON ERROR ON REGISTERS/

1063	002012	044507	052123	051105					
1064	002020	000123							
1065	002022	020040	042522	042503	EM2:	ASCIZ	/	RECEIVER ERROR/	
1066	002030	053111	051105	042440					
1067	002036	051122	051117	000					
1068	002043	040	052040	040522	EM3:	ASCIZ	/	TRANSMITTER ERROR/	
1069	002050	051516	044515	052124					
1070	002056	051105	042440	051122					
1071	002064	051117	000						
1072									
1073	002067	105	051122	041520	DH1:	ASCIZ	/ERRPC	WANTED ACTUAL/	
1074	002074	020040	040527	052116					
1075	002102	042105	020040	041501					
1076	002110	052524	046101	000					
1077		002116							
1078									
1079	002116	001416	001130	001132	DT1:	WORD	SERRPC,HLDD,HLDD1,0		
1080	002124	000000							
1081									
1082	002126	001416	000000		DT4:	WORD	SERRPC,0		
1083									
1084	002132	000	000	000	DF1:	BYTE	0,0,0,0		
1085	002135	000							
1086									
1087									
1088									
1089									
1090									
1091		002136							
1092		000046							
1093	000046	012544							
1094		000052							
1095	000052	000000							
1096		002136							
1097									
1098									
1099									
1100									
1101									
1102		002136							
1103		000024							
1104	000024	000200							
1105		000044							
1106	000044	002136							
1107		002136							
1108									
1109									
1110									
1111									
1112	002136				SAPTHD:				
1113	002136	000000			SHIBTS:	WORD	0	;; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.	
1114	002140	001526			SMBADR:	WORD	SMAIL	;; ADDRESS OF APT MAILBOX (BITS 0-15)	
1115	002142	000010			STSTM:	WORD	10	;; RUN TIM OF LONGEST TEST	
1116	002144	000010			SPASTM:	WORD	10	;; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)	
1117	002146	000000			SUNITM:	WORD		;; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT	
1118	002150	000052				WORD	SETEND-SMAIL/2	;; LENGTH MAILBOX-ETABLE (WORDS)	

; DATA HEADERS FOR ERROR MESSAGES
ASCIZ /ERRPC WANTED ACTUAL/

; DATA TABLES FOR ERROR MESSAGES
WORD SERRPC,HLDD,HLDD1,0

EVEN
SBTTL ACT11 HOOKS

;; *****
HOOKS REQUIRED BY ACT11
SSVPC= ; SAVE PC
=46 ; ;1)SET LOC. 46 TO ADDRESS OF SENDAD IN .SEOP
SENDAD
=52 ; ;2)SET LOC. 52 TO ZERO
WORD 0 ; ; RESTORE PC
=SSVPC
SBTTL APT PARAMETER BLOCK

;; *****
SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
;; *****
SX= ; ; SAVE CURRENT LOCATION
=24 ; ; SET POWER FAIL TO POINT TO START OF PROGRAM
200 ; ; FOR APT START UP
=44 ; ; POINT TO APT INDIRECT ADDRESS PNTR.
SAPTHDR ; ; POINT TO APT HEADER BLOCK
= SX ; ; RESET LOCATION COUNTER
;; *****
SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
INTERFACE SPEC.

```

1119
1120
1121 ;PROGRAM INITIALIZATION
1122 ;LOCK OUT INTERRUPTS
1123 ;SET UP PROCESSOR STACK
1124 ;SET UP POWER FAIL VECTOR
1125 ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
1126 ;TYPE TITLE MESSAGE
1127
1128 002152 . START:
1129 . SBTTL INITIALIZE THE COMMON TAGS
1130 . ;CLEAR THE COMMON TAGS ($CMTAG) AREA
1131 002152 012706 001400 MOV #SCMTAG,R6 ;FIRST LOCATION TO BE CLEARED
1132 002156 005026 CLR (R6)+ ;CLEAR MEMORY LOCATION
1133 002160 022706 001440 CMP #SWR,R6 ;DONE?
1134 002164 001374 BNE -6 ;LOOP BACK IF NO
1135 002166 012706 001100 MOV ##STACK,SP ;SETUP THE STACK POINTER
1136 . ;INITIALIZE A FEW VECTORS
1137 002172 012737 016170 000020 MOV #SSCOPE,@#IOTVEC ;IOT VECTOR FOR SCOPE ROUTINE
1138 002200 012737 000340 000022 MOV #340,@#IOTVEC+2 ;LEVEL 7
1139 002206 012737 014060 000030 MOV #SEERROR,@#EMTVEC ;EMT VECTOR FOR ERROR ROUTINE
1140 002214 012737 000340 000032 MOV #340,@#EMTVEC+2 ;LEVEL 7
1141 002222 012737 016524 000034 MOV #STRAP,@#TRAPVEC ;TRAP VECTOR FOR TRAP CALLS
1142 002230 012737 000340 000036 MOV #340,@#TRAPVEC+2;LEVEL 7
1143 002236 012737 014662 000024 MOV #SPWRDN,@#PWRVEC ;POWER FAILURE VECTOR
1144 002244 012737 000340 000026 MOV #340,@#PWRVEC+2 ;LEVEL 7
1145 002252 005067 177234 CLR STIMES ;INITIALIZE NUMBER OF ITERATIONS
1146 002256 005067 177232 CLR SESCAPE ;CLEAR THE ESCAPE ON ERROR ADDRESS
1147 002262 112767 000001 177125 MOVB #1,SERMAX ;ALLOW ONE ERROR PER TEST
1148 002270 012767 002270 177110 MOV #.,SLPADR ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1149 002276 012767 002276 177104 MOV #.,SLPERR ;SETUP THE ERROR LOOP ADDRESS
1150 . ;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1151 . ;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1152 002304 013746 000004 MOV @#ERRVEC,-(SP) ;SAVE ERROR VECTOR
1153 002310 012737 002344 000004 MOV #64$,@#ERRVEC ;SET UP ERROR VECTOR
1154 002316 012767 177570 177114 MOV #DSWR,SWR ;SETUP FOR A HARDWARE SWICH REGISTER
1155 002324 012767 177570 177110 MOV #DDISP,DISPLAY ;AND A HARDWARE DISPLAY REGISTER
1156 002332 022777 177777 177100 CMP #-1,@SWR ;TRY TO REFERENCE HARDWARE SWR
1157 002340 001012 BNE 66$ ;BRANCH IF NO TIMEOUT TRAP OCCURRED
1158 . ;AND THE HARDWARE SWR IS NOT = -1
1159 002342 000403 BR 65$ ;BRANCH IF NO TIMEOUT
1160 002344 012716 002352 64$: MOV #65$, (SP) ;SET UP FOR TRAP RETURN
1161 002350 000002 RTI
1162 002352 012767 000176 177060 65$: MOV #SWREG,SWR ;POINT TO SOFTWARE SWR
1163 002360 012767 000174 177054 MOV #DISPREG,DISPLAY
1164 002366 012637 000004 66$: MOV (SP)+,@#ERRVEC ;RESTORE ERROR VECTOR
1165
1166 002372 005067 177136 CLR $PASS ;CLEAR PASS COUNT
1167 002376 132767 000200 177143 BITB #APTSIZE,$ENVM ;TEST USER SIZE UNDER APT
1168 002404 001403 BEQ 67$ ;YES,USE NON-APT SWITCH
1169 002406 012767 001550 177024 MOV #SSWREG,SWR ;NO,USE APT SWITCH REGISTER
1170 002414 67$:
1171 002414 012706 001100 MOV #STACK,SP ;SET STACK
1172 002420 106427 000340 MTPS #340 ;LOCK INTERRUPTS
1173 002424 012737 014662 000024 MOV #.PFAIL,@#24 ;SET UP POWER FAIL VECTOR
1174 002432 105067 176535 CLRB STFLG ;CLEAR START FLAG

```

INITIALIZE THE COMMON TAGS

1175	002436	005067	176450		CLR	PASCNT		; CLEAR PASS COUNT
1176	002442	105067	176735		CLRB	\$ERFLC		; CLEAR ERROR FLAG
1177	002446	005067	176740		CLR	\$ERTTL		; CLEAR ERROR COUNT
1178	002452	005067	176740		CLR	\$ERRPC		; CLEAR LAST ERROR POINTER
1179	002456	012767	000001	176716	MOV	#1, \$TSTNM		; SET UP FOR TEST 1
1180	002464	012767	002152	176412	MOV	#. START, RETURN		; SET UP FOR POWER FAIL BEFORE
1181								; TESTING STARTS
1182	002472	013746	000006		MOV	@#6, -(SP)		
1183	002476	013746	000004		MOV	@#4, -(SP)		
1184	002502	012737	002516	000004	MOV	#1\$, @#4		
1185	002510	005777	176724		TST	@SWR		
1186	002514	000407			BR	2\$		
1187	002516	012767	000176	176714	15:	MOV	#SWREG, SWR	
1188	002524	012767	000174	176710		MOV	#DISPREG, DISPLAY	
1189	002532	022626				CMP	(SP)+, (SP)+	
1190	002534	012637	000004		25:	MOV	(SP)+, @#4	
1191	002540	012637	000006			MOV	(SP)+, @#6	
1192	002544	022767	000176	176666		CMP	#SWREG, SWR	
1193	002552	001007				BNE	3\$	
1194	002554	005737	000042			TST	@#42	; CHECK FOR CHAIN
1195	002560	001402				BEQ	33\$	
1196	002562	000167	000522			JMP	. BEGIN	
1197	002566	004767	010054		33\$:	JSR	PC, CNTLU	
1198	002572	105767	176374		3\$:	TSTB	INIFLG	; HAS INITIALIZATION BEEN PERFORMED
1199	002576	001004				BNE	ONCE	
1200	002600	104401	01E022			TYPE	, MTITLE	; TYPE TITLE MESSAGE
1201	002604	105167	176362			COMB	INIFLG	; IF NOT SET FLAG AND DO
1202	002610	105767	176732		ONCE:	TSTB	\$ENV	; APT CONTROL?
1203	002614	001410				BEQ	11\$; BR IF NO
1204	002616	032767	000001	176726		BIT	#1, \$USWR	; EXTENAL JUMPER ON?
1205	002624	001002				BNE	12\$; NO
1206	002626	105067	176321			CLRB	JMRBY	; CLEAR FLAG
1207	002632	000167	000452		12\$:	JMP	. BEGIN	; GO DO IT
1208	002636	032777	000001	176574	11\$:	BIT	#SW00, @SWR	; RESELECT VECTOR & CONTROL REG?
1209	002644	001002				BNE	1\$	
1210	002646	000167	000436			JMP	. BEGIN	
1211	002652	012700	000300		1\$:	MOV	#300, R0	; RESTORE VECTOR AREA TO TRAPCATCHER
1212	002656	012701	000302			MOV	#302, R1	; START AT LOCATION 300
1213	002662	012702	000004			MOV	#4, R2	
1214	002666	010110			25:	MOV	R1, (R0)	
1215	002670	005011				CLR	(R1)	
1216	002672	060200				ADD	R2, R0	
1217	002674	060201				ADD	R2, R1	
1218	002676	022701	001000			CMP	#1000, R1	; END AT LOCATION 776
1219	002702	002771				BLT	2\$	
1220	002704	104406				INSTR		; OUTPUT MESSAGE & GET INPUT STRING
1221	002706	015070				MREGAD		; MESSAGE
1222	002710	104410				PARAM		; CONVERT STRING
1223	002712	160000				160000		; LOW LIMIT
1224	002714	167776				167776		; HIGH LIMIT
1225	002716	017020				DUBASE		; STORE AT THIS LOCATION
1226	002720	001			BYTE	1		; MASK
1227	002721	001			BYTE	1		; HOW MANY TIMES + 2
1228	002722	016767	014072	176226		MOV	DUBASE, KEEPADD	; SAVE
1229	002730	004767	013732			JSR	PC, DUADDR	
1230	002734	016767	176216	176212		MOV	KEEPADD, BASEADD	; RESTORE FOR ROTATION

```

1231 002742 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1232 002744 015055 MVECTO ;MESSAGE
1233 002746 104410 PARAM ;CONVERT STRING
1234 002750 000300 300 ;LOW LIMIT
1235 002752 000776 776 ;HIGH LIMIT
1236 002754 001736 DURIV ;STORE AT THIS LOCATION
1237 002756 001 . BYTE 1 ;MASK
1238 002757 004 . BYTE 4 ;HOW MANY TIMES + 2
1239 002760 016767 176752 176176 MOV DURIV,KEEPIV ;SAVE
1240 002766 016767 176744 176166 MOV DURIV,BASEIV ;SET UP FOR ROTATION
1241 002774 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1242 002776 015120 MMULT ;MESSAGE
1243 003000 104414 SETFLG ;SET FLAG BASED UPON INPUT STRING
1244 003002 001152 MULTD ;THIS FLAG
1245 003004 105767 176142 TSTB MULTD ;ARE THERE MULTIPLE DEVICES
1246 ;ON THE SYSTEM ?
1247 003010 100406 BMI BBB ;YES,ASK NEXT QUESTION
1248 003012 005067 176150 CLR ACTREG
1249 003016 005067 176146 CLR ROTADD
1250 003022 000167 000140 JMP OUTMUL ;JUMP AROUND NEXT QUESTION
1251 003026 BBB:
1252 003026 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1253 003030 015147 MLASTD ;MESSAGE
1254 003032 104410 PARAM ;CONVERT STRING
1255 003034 160000 160000 ;LOW LIMIT
1256 003036 167776 167776 ;HIGH LIMIT
1257 003040 001160 LASTADD ;STORE AT THIS LOCATION
1258 003042 001 . BYTE 1 ;MASK
1259 003043 001 . BYTE 1 ;HOW MANY TIMES + 2
1260 ;THE FOLLOWING ROUTINE SETS UP ACTREG FOR THE FIRST TIME
1261 003044 012767 000001 176116 1$: MOV #1,ROTADD ;SET UP POINTER
1262 003052 005067 176110 CLR ACTREG ;CLR ACTIVE REGISTER
1263 003056 056767 176106 176102 2$: BIS ROTADD,ACTREG ;MAKE THIS DEVICE ACTIVE
1264 003064 000241 CLC
1265 003066 006167 176076 ROL ROTADD ;SET UP POINTER
1266 003072 103421 BCS 3$ ;ARE YOU OUT OF RANGE ?
1267 003074 062767 000010 176052 ADD #10,BASEADD ;SET UP BASE ADDRESS
1268 003102 026767 176052 176044 CMP LASTADD,BASEADD ;IS THIS THE LAST DEVICE ?
1269 003110 101362 BHI 2$ ;NO DO IT AGAIN
1270 003112 056767 176052 176046 BIS ROTADD,ACTREG ;THIS ASSUMES THAT THERE ARE AT
1271 ;LEAST TWO DEVICES WHEN YOU ANSWER YES TO
1272 ;MULTIPLE DEVICE QUESTION
1273 003120 012767 000001 176042 4$: MOV #1,ROTADD ;SET UP FOR LATER USE IN END OF PASS ROUTINE
1274 003126 016767 176024 176020 MOV KEEPADD,BASEADD ;DITTO
1275 003134 000414 BR OUTMUL ;CONTINUE QUESTIONS
1276 003136 016767 176014 176010 3$: MOV KEEPADD,BASEADD ;RESTORE
1277 003144 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1278 003146 015243 MRANGE ;MESSAGE
1279 003150 104410 PARAM ;CONVERT STRING
1280 003152 160000 160000 ;LOW LIMIT
1281 003154 167776 167776 ;HIGH LIMIT
1282 003156 001160 LASTADD ;STORE AT THIS LOCATION
1283 003160 001 . BYTE 1 ;MASK
1284 003161 001 . BYTE 1 ;HOW MANY TIMES + 2
1285 003162 000167 177656 JMP 1$ ;DO IT AGAIN
1286 003166 012767 000340 013466 OUTMUL: MOV #340,DUPRT

```

INITIALIZE THE COMMON TAGS

```

1287 003174 004767 013412      JSR      PC,DULEV
1288                          ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
1289                          ;BUFFER TO THE CHARACTERS "1" AND "2"
1290                          ;IF THE CHARACTER IS "1" CLEAR THE FLAG
1291                          ;IF THE CHARACTER IS "2" SET THE FLAG
1292 003200                      AAA:
1293 003200 104406              INSTR    ;OUTPUT MESSAGE & GET INPUT STRING
1294 003202 015461              MSYNC   ;MESSAGE
1295 003204 122767 000061 012610 3$: CMPB   #'1,INBUF      ;IS IT "1" ?
1296 003212 001003              BNE    1$
1297 003214 105067 175726      CLRB   SYNCNO ;000
1298 003220 000412              BR     4$
1299 003222 122767 000062 012572 1$: CMPB   #'2,INBUF      ;IS IT "2" ?
1300 003230 001004              BNE    2$
1301 003232 112767 177777 175706 MOVB   #-1,SYNCNO    ;377
1302 003240 000402              BR     4$
1303 003242 104407              2$:  INSTR  ;RETRY
1304 003244 000757              BR     3$
1305 003246 000240              4$:  NOP
1306 003250 104406              INSTR    ;OUTPUT MESSAGE & GET INPUT STRING
1307 003252 015527              MWIRE6 ;MESSAGE
1308 003254 104414              SETFLG ;SET FLAG BASED UPON INPUT STRING
1309 003256 001147              SEXMIT ;THIS FLAG
1310 003260 104406              INSTR    ;OUTPUT MESSAGE & GET INPUT STRING
1311 003262 015600              MWIRE5 ;MESSAGE
1312 003264 104414              SETFLG ;SET FLAG BASED UPON INPUT STRING
1313 003266 001150              SEREC  ;THIS FLAG
1314 003270 104406              INSTR    ;OUTPUT MESSAGE & GET INPUT STRING
1315 003272 015650              MWIRE4 ;MESSAGE
1316 003274 104414              SETFLG ;SET FLAG BASED UPON INPUT STRING
1317 003276 001151              OPTCLR ;THIS FLAG
1318 003300 104406              INSTR    ;OUTPUT MESSAGE & GET INPUT STRING
1319 003302 015727              MEXTJ  ;MESSAGE
1320 003304 104414              SETFLG ;SET FLAG BASED UPON INPUT STRING
1321 003306 001153              JMRBY  ;THIS FLAG
1322
1323                          ;TEST START AND RESTART
1324
1325 003310 012706 001100      .BEGIN: MOV    #STACK,SP      ;SET UP STACK
1326 003314 106427 000340      MTPS   #340          ;LOCK OUT INTERRUPTS
1327 003320 032777 000002 176112 BIT    #SW01,@SWR     ;IF SW01=1, GET STARTING PC
1328 003326 001413              BEQ    3$
1329 003330 104406              INSTR    ;OUTPUT MESSAGE & GET INPUT STRING
1330 003332 015413              MTSTPC ;MESSAGE
1331 003334 104410              PARAM  ;CONVERT STRING
1332 003336 003374              TST1   ;LOW LIMIT
1333 003340 017500              17500 ;HIGH LIMIT
1334 003342 001402              $TSTNM ;STORE AT THIS LOCATION
1335 003344 001              .BYTE  1           ;MASK
1336 003345 001              .BYTE  1           ;HOW MANY TIMES + 2
1337 003346 016767 176030 175530 MOV    $TSTNM,RETURN
1338 003354 000403              BR     4$
1339 003356 012767 003374 175520 3$: MOV    #TST1,RETURN    ;START AT TEST 1
1340 003364 104401 015407      4$:  TYPE  ,MR          ;TYPE R
1341 003370 000177 175510      JMP    @RETURN        ;START TESTING
1342

```

```
1343 ; ; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1344 ; ; RECEIVER SECTION, IT USES THE ERROR FLAGS
1345 ; ; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1346 ; ; (OVRRUN, RXERR)
1347 ; ; MODE: ISYMOD
1348 ; ; LENGTH: FIVE
1349 ; ; CHAR: 37
1350 ; ;
1351 ; ; *****
1352 003374 000004 TST1: SCOPE
1353 003376 052777 000400 176322 BIS #MRESET,@TXCSR ; MASTER RESET
1354 003404 012777 000000 176310 MOV #ISYMOD,@PARCSR ; SET THE MODE
1355 003412 052777 000400 176306 BIS #MRESET,@TXCSR ; MASTER RESET
1356
1357 ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1358 003420 012777 064001 176300 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
1359
1360 ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
1361 003426 012777 000000 176266 MOV #ISYMOD!FIVE!NOPAR!D,@PARCSR
1362 003434 052777 000020 176250 BIS #SYNSCH,@RXCSR ; SET SYNC SEARCH
1363 ; POKE CLK TO GET RECEIVER INTO SYNCROIZATION...
1364 003442 042777 020000 176256 BIC #CLK,@TXCSR ; POKE CLK DOWN
1365 003450 052777 020000 176250 BIS #CLK,@TXCSR ; POKE CLK UP
1366 ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1367 003456 042777 020000 176242 BIC #CLK,@TXCSR ; POKE CLK DOWN
1368 003464 052777 020000 176234 BIS #CLK,@TXCSR ; POKE CLK UP
1369 003472 016703 176220 MOV RXDBUF,R3 ; SET UP FOR ERROR MESSAGE
1370 003476 012700 000037 MOV #37,R0 ; EXPECTED
1371 003502 012767 000007 175412 MOV #7,SHIFT ; # OF SHIFTS
1372 003510 012767 000176 175762 MOV #176,$TMP1 ; DATA CHAR
1373 003516 004767 013300 JSR PC,RPOKE ; SHIFT IN THIS CHAR
1374 003522 105777 176164 TSTB @RXCSR ; RXDONE ?
1375 003526 100401 BMI .+4
1376 003530 104004 ERROR 4 ; RXDONE SHOULD BE SET
1377 003532 017701 176160 MOV @RXDBUF,R1 ; ACTUAL
1378 003536 020001 CMP R0,R1 ; COMPARE EXPECTED VS. ACTUAL
1379 003540 001401 BEQ .+4
1380 003542 104002 ERROR 2 ; RECEIVED DATA DID NOT MATCH
1381 ; ; EXPECTED DATA - CHECK MAINT DATA
1382 ; ; OR RECEIVER LOGIC
1383 003544 012767 000007 175350 MOV #7,SHIFT ; # OF SHIFTS
1384 003552 012767 000176 175720 MOV #176,$TMP1 ; DATA CHAR
1385 003560 004767 013236 JSR PC,RPOKE ; SHIFT IN THIS CHAR
1386 ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1387 003564 012767 000007 175330 MOV #7,SHIFT ; # OF SHIFTS
1388 003572 012767 000176 175700 MOV #176,$TMP1 ; DATA CHAR
1389 003600 004767 013216 JSR PC,RPOKE ; SHIFT IN THIS CHAR
1390 003604 012700 140037 MOV #140000!37,R0 ; EXPECTED DATA PLUS
1391 ; ; RXERR & OVRRUN
1392 003610 017701 176102 MOV @RXDBUF,R1 ; ACTUAL
1393 003614 020001 CMP R0,R1 ; COMPARE EXP VS. ACT
1394 003616 001401 BEQ .+4
1395 003620 104002 ERROR 2 ; SPECIFICALLY LOOK AT RXERR &
1396 ; ; OVRRUN BITS... THEY BOTH SHOULD BE SET
1397
1398 ; ; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
```

```

1399                                     ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
1400                                     ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1401                                     ;; (OVRUN, RXERR)
1402                                     ;; MODE: ISYMOD
1403                                     ;; LENGTH: FIVE
1404                                     ;; CHAR: 0
1405                                     ;;
1406                                     ;; *****
1407 003622 000004 TST2: SCOPE
1408 003624 052777 000400 176074 BIS #MRESET,@TXCSR ; MASTER RESET
1409 003632 012777 000000 176062 MOV #ISYMOD,@PARCSR ; SET THE MODE
1410 003640 052777 000400 176060 BIS #MRESET,@TXCSR ; MASTER RESET
1411
1412 ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1413 003646 012777 064001 176052 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
1414
1415 ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
1416 003654 012777 000000 176040 MOV #ISYMOD!FIVE!NOPAR!0,@PARCSR
1417 003662 052777 000020 176022 BIS #SYNSCH,@RXCSR ; SET SYNC SEARCH
1418 ; POKE CLK TO GET RECEIVER INTO SYNCRIZATION...
1419 003670 042777 020000 176030 BIC #CLK,@TXCSR ; POKE CLK DOWN
1420 003676 052777 020000 176022 BIS #CLK,@TXCSR ; POKE CLK UP
1421 ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1422 003704 042777 020000 176014 BIC #CLK,@TXCSR ; POKE CLK DOWN
1423 003712 052777 020000 176006 BIS #CLK,@TXCSR ; POKE CLK UP
1424 003720 016703 17E772 MOV RXDBUF,R3 ; SET UP FOR ERROR MESSAGE
1425 003724 012700 000000 MOV #0,R0 ; EXPECTED
1426 003730 012767 000007 175164 MOV #7,SHIFT ; # OF SHIFTS
1427 003736 012767 000100 175534 MOV #100,$TMP1 ; DATA CHAR
1428 003744 004767 013052 JSR PC,RPOKE ; SHIFT IN THIS CHAR
1429 003750 105777 175736 TSTB @RXCSR ; RXDONE ?
1430 003754 100401 BMI +4
1431 003756 104004 ERROR 4 ; RXDONE SHOULD BE SET
1432 003760 017701 175732 MOV @RXDBUF,R1 ; ACTUAL
1433 003764 020001 CMP R0,R1 ; COMPARE EXPECTED VS. ACTUAL
1434 003766 001401 BEQ +4
1435 003770 104002 ERROR 2 ; RECEIVED DATA DID NOT MATCH
1436 ; EXPECTED DATA - CHECK MAINT DATA
1437 ; OR RECEIVER LOGIC
1438 003772 012767 000007 175122 MOV #7,SHIFT ; # OF SHIFTS
1439 004000 012767 000100 175472 MOV #100,$TMP1 ; DATA CHAR
1440 004006 004767 013010 JSR PC,RPOKE ; SHIFT IN THIS CHAR
1441 ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1442 004012 012767 000007 175102 MOV #7,SHIFT ; # OF SHIFTS
1443 004020 012767 000100 175452 MOV #100,$TMP1 ; DATA CHAR
1444 004026 004767 012770 JSR PC,RPOKE ; SHIFT IN THIS CHAR
1445 004032 012700 140000 MOV #140000!0,R0 ; EXPECTED DATA PLUS
1446 ; RXERR & OVRUN
1447 004036 017701 175654 MOV @RXDBUF,R1 ; ACTUAL
1448 004042 020001 CMP R0,R1 ; COMPARE EXP VS. ACT
1449 004044 001401 BEQ +4
1450 004046 104002 ERROR 2 ; SPECIFICALLY LOOK AT RXERR &
1451 ; OVRUN BITS... THEY BOTH SHOULD BE SET
1452
1453 ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1454 ;; RECEIVER SECTION, IT USES THE ERROR FLAGS

```

```

1455                ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1456                ;; (OVRUN, RXERR)
1457                ;; MODE: ISYMOD
1458                ;; LENGTH: SIX
1459                ;; CHAR: 25
1460                ;;
1461                ;; *****
1462 004050 000004 TST3: SCOPE
1463 004052 052777 000400 175646 BIS #MRESET, @TXCSR ; MASTER RESET
1464 004060 012777 000000 175634 MOV #ISYMOD, @PARCSR ; SET THE MODE
1465 004066 052777 000400 175632 BIS #MRESET, @TXCSR ; MASTER RESET
1466
1467                ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1468 004074 012777 064001 175624 MOV #MTDATA!CLK!MINT!BREAK, @TXCSR
1469
1470                ; SET MODE, # OF BITS, PARITY SENSE, & LOAD SYNC REG
1471 004102 012777 002000 175612 MOV #ISYMOD!SIX!NOPAR!0, @PARCSR
1472 004110 052777 000020 175574 BIS #SYNSCH, @RXCSR ; SET SYNC SEARCH
1473                ; POKE CLK TO GET RECEIVER INTO SYNCHRONIZATION...
1474 004116 042777 020000 175602 BIC #CLK, @TXCSR ; POKE CLK DOWN
1475 004124 052777 020000 175574 BIS #CLK, @TXCSR ; POKE CLK UP
1476                ; POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
1477 004132 042777 020000 175566 BIC #CLK, @TXCSR ; POKE CLK DOWN
1478 004140 052777 020000 175560 BIS #CLK, @TXCSR ; POKE CLK UP
1479 004146 016703 175544 MOV RXDBUF, R3 ; SET UP FOR ERROR MESSAGE
1480 004152 012700 000025 MOV #25, R0 ; EXPECTED
1481 004156 012767 000010 174736 MOV #8, SHIFT ; # OF SHIFTS
1482 004164 012767 000252 175306 MOV #252, $TMP1 ; DATA CHAR
1483 004172 004767 012624 JSR PC, RPOKE ; SHIFT IN THIS CHAR
1484 004176 105777 175510 TSTB @RXCSR ; RXDONE ?
1485 004202 100401 BMI .+4
1486 004204 104004 ERROR 4 ; RXDONE SHOULD BE SET
1487 004206 017701 175504 MOV @RXDBUF, R1 ; ACTUAL
1488 004212 020001 CMP R0, R1 ; COMPARE EXPECTED VS. ACTUAL
1489 004214 001401 BEQ .+4
1490 004216 104002 ERROR 2 ; RECEIVED DATA DID NOT MATCH
1491                ; EXPECTED DATA - CHECK MAINT DATA
1492                ; OR RECEIVER LOGIC
1493 004220 012767 000010 174674 MOV #8, SHIFT ; # OF SHIFTS
1494 004226 012767 000252 175244 MOV #252, $TMP1 ; DATA CHAR
1495 004234 004767 012562 JSR PC, RPOKE ; SHIFT IN THIS CHAR
1496                ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1497 004240 012767 000010 174654 MOV #8, SHIFT ; # OF SHIFTS
1498 004246 012767 000252 175224 MOV #252, $TMP1 ; DATA CHAR
1499 004254 004767 012542 JSR PC, RPOKE ; SHIFT IN THIS CHAR
1500 004260 012700 140025 MOV #140000!25, R0 ; EXPECTED DATA PLUS
1501                ; RXERR & OVRUN
1502 004264 017701 175426 MOV @RXDBUF, R1 ; ACTUAL
1503 004270 020001 CMP R0, R1 ; COMPARE EXP VS. ACT
1504 004272 001401 BEQ .+4
1505 004274 104002 ERROR 2 ; SPECIFICALLY LOOK AT RXERR &
1506                ; OVRUN BITS... THEY BOTH SHOULD BE SET
1507
1508                ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1509                ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
1510                ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1511
    
```

```
1511 ; (OVRRUN, RXERR)
1512 ; MODE: ISYMOD
1513 ; LENGTH: SIX
1514 ; CHAR: 52
1515 ;
1516 ; *****
1517 004276 000004 TST4: SCOPE
1518 004300 052777 000400 175420 BIS #MRESET, @TXCSR ; MASTER RESET
1519 004306 012777 000000 175406 MOV #ISYMOD, @PARCSR ; SET THE MODE
1520 004314 052777 000400 175404 BIS #MRESET, @TXCSR ; MASTER RESET
1521
1522 ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1523 004322 012777 064001 175376 MOV #MTDATA!CLK!MINT!BREAK, @TXCSR
1524
1525 ; SET MODE, # OF BITS, PARITY SENSE, & LOAD SYNC REG
1526 004330 012777 002000 175364 MOV #ISYMOD!SIX!NOPAR!0, @PARCSR
1527 004336 052777 000020 175346 BIS #SYNSCH, @RXCSR ; SET SYNC SEARCH
1528 ; POKE CLK TO GET RECEIVER INTO SYNCRIZATION...
1529 004344 042777 020000 175354 BIC #CLK, @TXCSR ; POKE CLK DOWN
1530 004352 052777 020000 175346 BIS #CLK, @TXCSR ; POKE CLK UP
1531 ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1532 004360 042777 020000 175340 BIC #CLK, @TXCSR ; POKE CLK DOWN
1533 004366 052777 020000 175332 BIS #CLK, @TXCSR ; POKE CLK UP
1534 004374 016703 175316 MOV RXDBUF, R3 ; SET UP FOR ERROR MESSAGE
1535 004400 012700 000052 MOV #52, R0 ; EXPECTED
1536 004404 012767 000010 174510 MOV #8, SHIFT ; # OF SHIFTS
1537 004412 012767 000324 175060 MOV #324, STMP1 ; DATA CHAR
1538 004420 004767 012376 JSR PC, RPOKE ; SHIFT IN THIS CHAR
1539 004424 105777 175262 TSTB @RXCSR ; RXDONE ?
1540 004430 100401 BMI .+4
1541 004432 104004 ERROR 4 ; RXDONE SHOULD BE SET
1542 004434 017701 175256 MOV @RXDBUF, R1 ; ACTUAL
1543 004440 020001 CMP R0, R1 ; COMPARE EXPECTED VS. ACTUAL
1544 004442 001401 BEQ .+4
1545 004444 104002 ERROR 2 ; RECEIVED DATA DID NOT MATCH
1546 ; EXPECTED DATA - CHECK MAINT DATA
1547 ; OR RECEIVER LOGIC
1548 004446 012767 000010 174446 MOV #8, SHIFT ; # OF SHIFTS
1549 004454 012767 000324 175016 MOV #324, STMP1 ; DATA CHAR
1550 004462 004767 012334 JSR PC, RPOKE ; SHIFT IN THIS CHAR
1551 ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1552 004466 012767 000010 174426 MOV #8, SHIFT ; # OF SHIFTS
1553 004474 012767 000324 174776 MOV #324, STMP1 ; DATA CHAR
1554 004502 004767 012314 JSR PC, RPOKE ; SHIFT IN THIS CHAR
1555 004506 012700 140052 MOV #140000!52, R0 ; EXPECTED DATA PLUS
1556 ; RXERR & OVRRUN
1557 004512 017701 175200 MOV @RXDBUF, R1 ; ACTUAL
1558 004516 020001 CMP R0, R1 ; COMPARE EXP VS. ACT
1559 004520 001401 BEQ .+4
1560 004522 104002 ERROR 2 ; SPECIFICALLY LOOK AT RXERR &
1561 ; OVRRUN BITS... THEY BOTH SHOULD BE SET
1562
1563 ; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1564 ; RECEIVER SECTION, IT USES THE ERROR FLAGS
1565 ; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1566 ; (OVRRUN, RXERR)
```

INITIALIZE THE COMMON TAGS

```

1567          ;; MODE: ISYMOD
1568          ;; LENGTH: SIX
1569          ;; CHAR: 77
1570          ;;
1571          ;; *****
1572 004524 000004          TST5: SCOPE
1573 004526 052777 000400 175172      BIS      #MRESET,@TXCSR ; MASTER RESET
1574 004534 012777 000000 175160      MOV      #ISYMOD,@PARCSR ; SET THE MODE
1575 004542 052777 000400 175156      BIS      #MRESET,@TXCSR ; MASTER RESET
1576
1577          ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1578 004550 012777 064001 175150      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
1579
1580          ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
1581 004556 012777 002000 175136      MOV      #ISYMOD!SIX!NOPAR!0,@PARCSR
1582 004564 052777 000020 175120      BIS      #SYNSCH,@RXCSR ; SET SYNC SEARCH
1583          ; POKE CLK TO GET RECEIVER INTO SYNCROIZATION...
1584 004572 042777 020000 175126      BIC      #CLK,@TXCSR ; POKE CLK DOWN
1585 004600 052777 020000 175120      BIS      #CLK,@TXCSR ; POKE CLK UP
1586          ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1587 004606 042777 020000 175112      BIC      #CLK,@TXCSR ; POKE CLK DOWN
1588 004614 052777 020000 175104      BIS      #CLK,@TXCSR ; POKE CLK UP
1589 004622 016703 175070          MOV      RXDBUF,R3 ; SET UP FOR ERROR MESSAGE
1590 004626 012700 000077          MOV      #77,R0 ; EXPECTED
1591 004632 012767 000010 174262      MOV      #8,SHIFT ; # OF SHIFTS
1592 004640 012767 000376 174632      MOV      #376,$TMP1 ; DATA CHAR
1593 004646 004767 012150          JSR      PC,RPOKE ; SHIFT IN THIS CHAR
1594 004652 105777 175034          TSTB    @RXCSR ; RXDONE ?
1595 004656 100401          BMI     .+4
1596 004660 104004          ERROR  4 ; RXDONE SHOULD BE SET
1597 004662 017701 175030          MOV      @RXDBUF,R1 ; ACTUAL
1598 004666 020001          CMP     R0,R1 ; COMPARE EXPECTED VS ACTUAL
1599 004670 001401          BEQ     .+4
1600 004672 104002          ERROR  2 ; RECEIVED DATA DID NOT MATCH
1601          ; EXPECTED DATA - CHECK MAINT DATA
1602          ; OR RECEIVER LOGIC
1603 004674 012767 000010 174220      MOV      #8,SHIFT ; # OF SHIFTS
1604 004702 012767 000376 174570      MOV      #376,$TMP1 ; DATA CHAR
1605 004710 004767 012106          JSR      PC,RPOKE ; SHIFT IN THIS CHAR
1606          ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1607 004714 012767 000010 174200      MOV      #8,SHIFT ; # OF SHIFTS
1608 004722 012767 000376 174550      MOV      #376,$TMP1 ; DATA CHAR
1609 004730 004767 012066          JSR      PC,RPOKE ; SHIFT IN THIS CHAR
1610 004734 012700 140077          MOV      #140000!77,R0 ; EXPECTED DATA PLUS
1611          ; RXERR & OVRUN
1612 004740 017701 174752          MOV      @RXDBUF,R1 ; ACTUAL
1613 004744 020001          CMP     R0,R1 ; COMPARE EXP VS. ACT
1614 004746 001401          BEQ     .+4
1615 004750 104002          ERROR  2 ; SPECIFICALLY LOOK AT RXERR &
1616          ; OVRUN BITS... THEY BOTH SHOULD BE SET
1617
1618          ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1619          ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
1620          ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1621          ;; (OVRUN,RXERR)
1622          ;; MODE: ISYMOD
    
```

```
1623          ;; LENGTH: SIX
1624          ;; CHAR: 0
1625          ;;
1626          ;; *****
1627 004752 000004          TST6: SCOPE
1628 004754 052777 000400 174744      BIS      #MRESET,@TXCSR ; MASTER RESET
1629 004762 012777 000000 174732      MOV      #ISYMOD,@PARCSR ; SET THE MODE
1630 004770 052777 000400 174730      BIS      #MRESET,@TXCSR ; MASTER RESET
1631
1632          ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1633 004776 012777 064001 174722      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
1634
1635          ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
1636 005004 012777 002000 174710      MOV      #ISYMOD!SIX!NOPAR!0,@PARCSR
1637 005012 052777 000020 174672      BIS      #SYNSCH,@RXCSR ; SET SYNC SEARCH
1638          ; POKE CLK TO GET RECEIVER INTO SYNCRIZATION...
1639 005020 042777 020000 174700      BIC      #CLK,@TXCSR ; POKE CLK DOWN
1640 005026 052777 020000 174672      BIS      #CLK,@TXCSR ; POKE CLK UP
1641          ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1642 005034 042777 020000 174664      BIC      #CLK,@TXCSR ; POKE CLK DOWN
1643 005042 052777 020000 174656      BIS      #CLK,@TXCSR ; POKE CLK UP
1644 005050 016703 174642          MOV      RXDBUF,R3 ; SET UP FOR ERROR MESSAGE
1645 005054 012700 000000          MOV      #0,R0 ; EXPECTED
1646 005060 012767 000010 174034      MOV      #8,SHIFT ; # OF SHIFTS
1647 005066 012767 000200 174404      MOV      #200,$TMP1 ; DATA CHAR
1648 005074 004767 011722          JSR      PC,RPOKE ; SHIFT IN THIS CHAR
1649 005100 105777 174606          TSTB    @RXCSR ; RXDONE ?
1650 005104 100401          BMI     .+4
1651 005106 104004          ERROR   4 ; RXDONE SHOULD BE SET
1652 005110 017701 174602          MOV     @RXDBUF,R1 ; ACTUAL
1653 005114 020001          CMP     R0,R1 ; COMPARE EXPECTED VS. ACTUAL
1654 005116 001401          BEQ     .+4
1655 005120 104002          ERROR   2 ; RECEIVED DATA DID NOT MATCH
1656          ; EXPECTED DATA - CHECK MAINT DATA
1657          ; OR RECEIVER LOGIC
1658 005122 012767 000010 173772      MOV     #8,SHIFT ; # OF SHIFTS
1659 005130 012767 000200 174342      MOV     #200,$TMP1 ; DATA CHAR
1660 005136 004767 011660          JSR     PC,RPOKE ; SHIFT IN THIS CHAR
1661          ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1662 005142 012767 000010 173752      MOV     #8,SHIFT ; # OF SHIFTS
1663 005150 012767 000200 174322      MOV     #200,$TMP1 ; DATA CHAR
1664 005156 004767 011640          JSR     PC,RPOKE ; SHIFT IN THIS CHAR
1665 005162 012700 140000          MOV     #140000!0,R0 ; EXPECTED DATA PLUS
1666          ; RXERR & OVRRUN
1667 005166 017701 174524          MOV     @RXDBUF,R1 ; ACTUAL
1668 005172 020001          CMP     R0,R1 ; COMPARE EXP VS. ACT
1669 005174 001401          BEQ     .+4
1670 005176 104002          ERROR   2 ; SPECIFICALLY LOOK AT RXERR &
1671          ; OVRRUN BITS... THEY BOTH SHOULD BE SET
1672
1673          ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1674          ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
1675          ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1676          ;; (OVRRUN, RXERR)
1677          ;; MODE: ISYMOD
1678          ;; LENGTH: SEVEN
```

INITIALIZE THE COMMON TAGS

```
1679          ;; CHAR: 125
1680          ;;
1681          ;; *****
1682 005200 000004          TST7: SCOPE
1683 005202 052777 000400 174516      BIS      #MRESET,@TXCSR ; MASTER RESET
1684 005210 012777 000000 174504      MOV      #ISYMOD,@PARCSR ; SET THE MODE
1685 005216 052777 000400 174502      BIS      #MRESET,@TXCSR ; MASTER RESET
1686
1687          ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1688 005224 012777 064001 174474      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
1689
1690          ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
1691 005232 012777 004000 174462      MOV      #ISYMOD!SEVEN!NOPAR!0,@PARCSR
1692 005240 052777 000020 174444      BIS      #SYNSCH,@RXCSR ; SET SYNC SEARCH
1693          ; POKE CLK TO GET RECEIVER INTO SYNCRIZATION. . .
1694 005246 042777 020000 174452      BIC      #CLK,@TXCSR ; POKE CLK DOWN
1695 005254 052777 020000 174444      BIS      #CLK,@TXCSR ; POKE CLK UP
1696          ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1697 005262 042777 020000 174436      BIC      #CLK,@TXCSR ; POKE CLK DOWN
1698 005270 052777 020000 174430      BIS      #CLK,@TXCSR ; POKE CLK UP
1699 005276 016703 174414          MOV      RXDBUF,R3 ; SET UP FOR ERROR MESSAGE
1700 005302 012700 000125          MOV      #125,R0 ; EXPECTED
1701 005306 012767 000011 173606      MOV      #9,SHIFT ; # OF SHIFTS
1702 005314 012767 000652 174156      MOV      #652,$TMP1 ; DATA CHAR
1703 005322 004767 011474          JSR      PC,RPOKE ; SHIFT IN THIS CHAR
1704 005326 105777 174360          TSTB    @RXCSR ; RXDONE ?
1705 005332 100401          BMI     .+4
1706 005334 104004          ERROR  4 ; RXDONE SHOULD BE SET
1707 005336 017701 174354          MOV      @RXDBUF,R1 ; ACTUAL
1708 005342 020001          CMP     R0,R1 ; COMPARE EXPECTED VS. ACTUAL
1709 005344 001401          BEQ     .+4
1710 005346 104002          ERROR  2 ; RECEIVED DATA DID NOT MATCH
1711          ; EXPECTED DATA - CHECK MAINT DATA
1712          ; OR RECEIVER LOGIC
1713 005350 012767 000011 173544      MOV      #9,SHIFT ; # OF SHIFTS
1714 005356 012767 000652 174114      MOV      #652,$TMP1 ; DATA CHAR
1715 005364 004767 011432          JSR      PC,RPOKE ; SHIFT IN THIS CHAR
1716          ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1717 005370 012767 000011 173524      MOV      #9,SHIFT ; # OF SHIFTS
1718 005376 012767 000652 174074      MOV      #652,$TMP1 ; DATA CHAR
1719 005404 004767 011412          JSR      PC,RPOKE ; SHIFT IN THIS CHAR
1720 005410 012700 140125          MOV      #140000!125,R0 ; EXPECTED DATA PLUS
1721          ; RXERR & OVRRUN
1722 005414 017701 174276          MOV      @RXDBUF,R1 ; ACTUAL
1723 005420 020001          CMP     R0,R1 ; COMPARE EXP VS. ACT
1724 005422 001401          BEQ     .+4
1725 005424 104002          ERROR  2 ; SPECIFICALLY LOOK AT RXERR &
1726          ; OVRRUN BITS. . . THEY BOTH SHOULD BE SET
1727
1728          ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1729          ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
1730          ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1731          ;; (OVRRUN, RXERR)
1732          ;; MODE: ISYMOD
1733          ;; LENGTH: SEVEN
1734          ;; CHAR: 52
```

INITIALIZE THE COMMON TAGS

```
1735  
1736  
1737 005426 000004  
1738 005430 052777 000400 174270  
1739 005436 012777 000000 174256  
1740 005444 052777 000400 174254  
1741  
1742  
1743 005452 012777 064001 174246  
1744  
1745  
1746 005460 012777 004000 174234  
1747 005466 052777 000020 174216  
1748  
1749 005474 042777 020000 174224  
1750 005502 052777 020000 174216  
1751  
1752 005510 042777 020000 174210  
1753 005516 052777 020000 174202  
1754 005524 016703 174166  
1755 005530 012700 000052  
1756 005534 012767 000011 173360  
1757 005542 012767 000524 173730  
1758 005550 004767 011246  
1759 005554 105777 174132  
1760 005560 100401  
1761 005562 104004  
1762 005564 017701 174126  
1763 005570 020001  
1764 005572 001401  
1765 005574 104002  
1766  
1767  
1768 005576 012767 000011 173316  
1769 005604 012767 000524 173666  
1770 005612 004767 011204  
1771  
1772 005616 012767 000011 173276  
1773 005624 012767 000524 173646  
1774 005632 004767 011164  
1775 005636 012700 140052  
1776  
1777 005642 017701 174050  
1778 005646 020001  
1779 005650 001401  
1780 005652 104002  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1790
```

```
;;  
;*****  
TST10: SCOPE  
BIS #MRESET,@TXCSR ;MASTER RESET  
MOV #ISYMOD,@PARCSR ;SET THE MODE  
BIS #MRESET,@TXCSR ;MASTER RESET  
;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE  
MOV #MTDATA!CLK!MINT!BREAK,@TXCSR  
;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG  
MOV #ISYMOD!SEVEN!NOPAR!0,@PARCSR  
BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH  
;POKE CLK TO GET RECEIVER INTO SYNCROIZATION...  
BIC #CLK,@TXCSR ;POKE CLK DOWN  
BIS #CLK,@TXCSR ;POKE CLK UP  
;POKE CLK TO GET LOGIC INTO SYNCRONIZATION  
BIC #CLK,@TXCSR ;POKE CLK DOWN  
BIS #CLK,@TXCSR ;POKE CLK UP  
MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE  
MOV #52,R0 ;EXPECTED  
MOV #9,SHIFT ;# OF SHIFTS  
MOV #524,$TMP1 ;DATA CHAR  
JSR PC,RPOKE ;SHIFT IN THIS CHAR  
TSTB @RXCSR ;RXDONE ?  
BMI +4  
ERROR 4 ;RXDONE SHOULD BE SET  
MOV @RXDBUF,R1 ;ACTUAL  
CMP R0,R1 ;COMPARE EXPECTED VS. ACTUAL  
BEQ +4  
ERROR 2 ;RECEIVED DATA DID NOT MATCH  
;EXPECTED DATA - CHECK MAINT DATA  
;OR RECEIVER LOGIC  
MOV #9,SHIFT ;# OF SHIFTS  
MOV #524,$TMP1 ;DATA CHAR  
JSR PC,RPOKE ;SHIFT IN THIS CHAR  
;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF  
MOV #9,SHIFT ;# OF SHIFTS  
MOV #524,$TMP1 ;DATA CHAR  
JSR PC,RPOKE ;SHIFT IN THIS CHAR  
MOV #140000!52,R0 ;EXPECTED DATA PLUS  
;RXERR & OVRUN  
MOV @RXDBUF,R1 ;ACTUAL  
CMP R0,R1 ;COMPARE EXP VS. ACT  
BEQ +4  
ERROR 2 ;SPECIFICALLY LOOK AT RXERR &  
;OVRUN BITS... THEY BOTH SHOULD BE SET  
; THIS TEST VERIFYS WORD LENGTH SELECT OF THE  
; RECEIVER SECTION, IT USES THE ERROR FLAGS  
; TO DETERMINE THAT IT WAS SELECTED CORRECTLY  
; (OVRUN,RXERR)  
; MODE: ISYMOD  
; LENGTH: SEVEN  
; CHAR: 177  
;;
```

INITIALIZE THE COMMON TAGS

```
1791 ; ; *****
1792 005654 000004 TST11: SCOPE
1793 005656 052777 000400 174042 BIS #MRESET,@TXCSR ; MASTER RESET
1794 005664 012777 000000 174030 MOV #ISYMOD,@PARCSR ; SET THE MODE
1795 005672 052777 000400 174026 BIS #MRESET,@TXCSR ; MASTER RESET
1796
1797 ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1798 005700 012777 064001 174020 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
1799
1800 ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
1801 005706 012777 004000 174006 MOV #ISYMOD!SEVEN!NOPAR!0,@PARCSR
1802 005714 052777 000020 173770 BIS #SYNSCH,@RXCSR ; SET SYNC SEARCH
1803 ; POKE CLK TO GET RECEIVER INTO SYNCRIZATION. ...
1804 005722 042777 020000 173776 BIC #CLK,@TXCSR ; POKE CLK DOWN
1805 005730 052777 020000 173770 BIS #CLK,@TXCSR ; POKE CLK UP
1806 ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1807 005736 042777 020000 173762 BIC #CLK,@TXCSR ; POKE CLK DOWN
1808 005744 052777 020000 173754 BIS #CLK,@TXCSR ; POKE CLK UP
1809 005752 016703 173740 MOV RXDBUF,R3 ; SET UP FOR ERROR MESSAGE
1810 005756 012700 000177 MOV #177,R0 ; EXPECTED
1811 005762 012767 000011 173132 MOV #9,SHIFT ; # OF SHIFTS
1812 005770 012767 000776 173502 MOV #776,$TMP1 ; DATA CHAR
1813 005776 004767 011020 JSR PC,RPOKE ; SHIFT IN THIS CHAR
1814 006002 105777 173704 TSTB @RXCSR ; RXDONE ?
1815 006006 100401 BMI +4
1816 006010 104004 ERROR 4 ; RXDONE SHOULD BE SET
1817 006012 017701 173700 MOV @RXDBUF,R1 ; ACTUAL
1818 006016 020001 CMP R0,R1 ; COMPARE EXPECTED VS. ACTUAL
1819 006020 001401 BEQ +4
1820 006022 104002 ERROR 2 ; RECEIVED DATA DID NOT MATCH
1821 ; EXPECTED DATA - CHECK MAINT DATA
1822 ; OR RECEIVER LOGIC
1823 006024 012767 000011 173070 MOV #9,SHIFT ; # OF SHIFTS
1824 006032 012767 000776 173440 MOV #776,$TMP1 ; DATA CHAR
1825 006040 004767 010756 JSR PC,RPOKE ; SHIFT IN THIS CHAR
1826 ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1827 006044 012767 000011 173050 MOV #9,SHIFT ; # OF SHIFTS
1828 006052 012767 000776 173420 MOV #776,$TMP1 ; DATA CHAR
1829 006060 004767 010736 JSR PC,RPOKE ; SHIFT IN THIS CHAR
1830 006064 012700 140177 MOV #140000!177,R0 ; EXPECTED DATA PLUS
1831 ; RXERR & OVRUN
1832 006070 017701 173622 MOV @RXDBUF,R1 ; ACTUAL
1833 006074 020001 CMP R0,R1 ; COMPARE EXP VS. ACT
1834 006076 001401 BEQ +4
1835 006100 104002 ERROR 2 ; SPECIFICALLY LOOK AT RXERR &
1836 ; OVRUN BITS... THEY BOTH SHOULD BE SET
1837
1838 ; ; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1839 ; ; RECEIVER SECTION, IT USES THE ERROR FLAGS
1840 ; ; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1841 ; ; (OVRUN, RXERR)
1842 ; ; MODE: ISYMOD
1843 ; ; LENGTH: SEVEN
1844 ; ; CHAR: 0
1845 ; ;
1846 ; ; *****
```

INITIALIZE THE COMMON TAGS

```
1847 006102 000004 TST12: SCOPE
1848 006104 052777 000400 173614 BIS #MRESET,@TXCSR ;MASTER RESET
1849 006112 012777 000000 173602 MOV #ISYMOD,@PARCSR ;SET THE MODE
1850 006120 052777 000400 173600 BIS #MRESET,@TXCSR ;MASTER RESET
1851
1852 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
1853 006126 012777 064001 173572 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
1854
1855 ;SET MODE , # OF BITS,PARITY SENSE,&LOAD SYNC REG
1856 006134 012777 004000 173560 MOV #ISYMOD!SEVEN!NOPAR!0,@PARCSR
1857 006142 052777 000020 173542 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
1858 ;POKE CLK TO GET RECEIVER INTO SYNCRIZATION...
1859 006150 042777 020000 173550 BIC #CLK,@TXCSR ;POKE CLK DOWN
1860 006156 052777 020000 173542 BIS #CLK,@TXCSR ;POKE CLK UP
1861 ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1862 006164 042777 020000 173534 BIC #CLK,@TXCSR ;POKE CLK DOWN
1863 006172 052777 020000 173526 BIS #CLK,@TXCSR ;POKE CLK UP
1864 006200 016703 173512 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
1865 006204 012700 000000 MOV #0,R0 ;EXPECTED
1866 006210 012767 000011 172704 MOV #9,SHIFT ;# OF SHIFTS
1867 006216 012767 000400 173254 MOV #400,$TMP1 ;DATA CHAR
1868 006224 004767 010572 JSR PC,RPOKE ;SHIFT IN THIS CHAR
1869 006230 105777 173456 TSTB @RXCSR ;RXDONE ?
1870 006234 100401 BMI +4
1871 006236 104004 ERROR 4 ;RXDONE SHOULD BE SET
1872 006240 017701 173452 MOV @RXDBUF,R1 ;ACTUAL
1873 006244 020001 CMP R0,R1 ;COMPARE EXPECTED VS. ACTUAL
1874 006246 001401 BEQ +4
1875 006250 104002 ERROR 2 ;RECEIVED DATA DID NOT MATCH
1876 ;EXPECTED DATA - CHECK MAINT DATA
1877 ;OR RECEIVER LOGIC
1878 006252 012767 000011 172642 MOV #9,SHIFT ;# OF SHIFTS
1879 006260 012767 000400 173212 MOV #400,$TMP1 ;DATA CHAR
1880 006266 004767 010530 JSR PC,RPOKE ;SHIFT IN THIS CHAR
1881 ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1882 006272 012767 000011 172622 MOV #9,SHIFT ;# OF SHIFTS
1883 006300 012767 000400 173172 MOV #400,$TMP1 ;DATA CHAR
1884 006306 004767 010510 JSR PC,RPOKE ;SHIFT IN THIS CHAR
1885 006312 012700 140000 MOV #140000!0,R0 ;EXPECTED DATA PLUS
1886 ;RXERR & OVRRUN
1887 006316 017701 173374 MOV @RXDBUF,R1 ;ACTUAL
1888 006322 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
1889 006324 001401 BEQ +4
1890 006326 104002 ERROR 2 ;SPECIFICALLY LOOK AT RXERR &
1891 ;OVRRUN BITS... THEY BOTH SHOULD BE SET
1892
1893 ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1894 ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
1895 ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1896 ;; (OVRRUN, RXERR)
1897 ;; MODE: ISYMOD
1898 ;; LENGTH: EIGHT
1899 ;; CHAR: 125
1900 ;;
1901 ;; *****
1902 006330 000004 TST13: SCOPE
```

INITIALIZE THE COMMON TAGS

```
1903 006332 052777 000400 173366      BIS      #MRESET,@TXCSR ; MASTER RESET
1904 006340 012777 000000 173354      MOV      #ISYMOD,@PARCSR ; SET THE MODE
1905 006346 052777 000400 173352      BIS      #MRESET,@TXCSR ; MASTER RESET
1906
1907 ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1908 006354 012777 064001 173344      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
1909
1910 ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
1911 006362 012777 006000 173332      MOV      #ISYMOD!EIGHT!NOPAR!0,@PARCSR
1912 006370 052777 000020 173314      BIS      #SYNSCH,@RXCSR ; SET SYNC SEARCH
1913 ; POKE CLK TO GET RECEIVER INTO SYNCROIZATION...
1914 006376 042777 020000 173322      BIC      #CLK,@TXCSR ; POKE CLK DOWN
1915 006404 052777 020000 173314      BIS      #CLK,@TXCSR ; POKE CLK UP
1916 ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1917 006412 042777 020000 173306      BIC      #CLK,@TXCSR ; POKE CLK DOWN
1918 006420 052777 020000 173300      BIS      #CLK,@TXCSR ; POKE CLK UP
1919 006426 016703 173264      MOV      RXDBUF,R3 ; SET UP FOR ERROR MESSAGE
1920 006432 012700 000125      MOV      #125,R0 ; EXPECTED
1921 006436 012767 000012 172456      MOV      #10,SHIFT ; # OF SHIFTS
1922 006444 012767 001252 173026      MOV      #1252,$TMP1 ; DATA CHAR
1923 006452 004767 010344      JSR      PC,RPOKE ; SHIFT IN THIS CHAR
1924 006456 105777 173230      TSTB    @RXCSR ; RXDONE ?
1925 006462 100401      BMI     .+4
1926 006464 104004      ERROR   4 ; RXDONE SHOULD BE SET
1927 006466 017701 173224      MOV     @RXDBUF,R1 ; ACTUAL
1928 006472 020001      CMP     R0,R1 ; COMPARE EXPECTED VS. ACTUAL
1929 006474 001401      BEQ     .+4
1930 006476 104002      ERROR   2 ; RECEIVED DATA DID NOT MATCH
1931 ; EXPECTED DATA - CHECK MAINT DATA
1932 ; OR RECEIVER LOGIC
1933 006500 012767 000012 172414      MOV     #10,SHIFT ; # OF SHIFTS
1934 006506 012767 001252 172764      MOV     #1252,$TMP1 ; DATA CHAR
1935 006514 004767 010302      JSR     PC,RPOKE ; SHIFT IN THIS CHAR
1936 ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1937 006520 012767 000012 172374      MOV     #10,SHIFT ; # OF SHIFTS
1938 006526 012767 001252 172744      MOV     #1252,$TMP1 ; DATA CHAR
1939 006534 004767 010262      JSR     PC,RPOKE ; SHIFT IN THIS CHAR
1940 006540 012700 140125      MOV     #140000!125,R0 ; EXPECTED DATA PLUS
1941 ; RXERR & OVRUN
1942 006544 017701 173146      MOV     @RXDBUF,R1 ; ACTUAL
1943 006550 020001      CMP     R0,R1 ; COMPARE EXP VS. ACT
1944 006552 001401      BEQ     .+4
1945 006554 104002      ERROR   2 ; SPECIFICALLY LOOK AT RXERR &
1946 ; OVRUN BITS... THEY BOTH SHOULD BE SET
1947
1948 ; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
1949 ; RECEIVER SECTION, IT USES THE ERROR FLAGS
1950 ; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
1951 ; (OVRUN,RXERR)
1952 ; MODE: ISYMOD
1953 ; LENGTH: EIGHT
1954 ; CHAR: 252
1955 ;
1956 ; *****
1957 006556 000004      TST14: SCOPE
1958 006560 052777 000400 173140      BIS     #MRESET,@TXCSR ; MASTER RESET
```

INITIALIZE THE COMMON TAGS

```
1959 006566 012777 000000 173126      MOV    #ISYMOD,@PARCSR ;SET THE MODE
1960 006574 052777 000400 173124      BIS    #MRESET,@TXCSR ;MASTER RESET
1961
1962                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
1963 006602 012777 064001 173116      MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
1964
1965                                     ;SET MODE , # OF BITS,PARITY SENSE,&LOAD SYNC REG
1966 006610 012777 006000 173104      MOV    #ISYMOD!EIGHT!NOPAR!0,@PARCSR
1967 006616 052777 000020 173066      BIS    #SYNSCH,@RXCSR ;SET SYNC SEARCH
1968                                     ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION...
1969 006624 042777 020000 173074      BIC    #CLK,@TXCSR ;POKE CLK DOWN
1970 006632 052777 020000 173066      BIS    #CLK,@TXCSR ;POKE CLK UP
1971                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1972 006640 042777 020000 173060      BIC    #CLK,@TXCSR ;POKE CLK DOWN
1973 006646 052777 020000 173052      BIS    #CLK,@TXCSR ;POKE CLK UP
1974 006654 016703 173036      MOV    RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
1975 006660 012700 000252      MOV    #252,R0 ;EXPECTED
1976 006664 012767 000012 172230      MOV    #10,SHIFT ;# OF SHIFTS
1977 006672 012767 001524 172600      MOV    #1524,$TMP1 ;DATA CHAR
1978 006700 004767 010116      JSR    PC,RPOKE ;SHIFT IN THIS CHAR
1979 006704 105777 173002      TSTB   @RXCSR ;RXDONE ?
1980 006710 100401      BMI    .+4
1981 006712 104004      ERROR  4 ;RXDONE SHOULD BE SET
1982 006714 017701 172776      MOV    @RXDBUF,R1 ;ACTUAL
1983 006720 020001      CMP    R0,R1 ;COMPARE EXPECTED VS. ACTUAL
1984 006722 001401      BEQ    .+4
1985 006724 104002      ERROR  2 ;RECEIVED DATA DID NOT MATCH
1986                                     ;EXPECTED DATA - CHECK MAINT DATA
1987                                     ;OR RECEIVER LOGIC
1988 006726 012767 000012 172166      MOV    #10,SHIFT ;# OF SHIFTS
1989 006734 012767 001524 172536      MOV    #1524,$TMP1 ;DATA CHAR
1990 006742 004767 010054      JSR    PC,RPOKE ;SHIFT IN THIS CHAR
1991                                     ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
1992 006746 012767 000012 172146      MOV    #10,SHIFT ;# OF SHIFTS
1993 006754 012767 001524 172516      MOV    #1524,$TMP1 ;DATA CHAR
1994 006762 004767 010034      JSR    PC,RPOKE ;SHIFT IN THIS CHAR
1995 006766 012700 140252      MOV    #140000!252,R0 ;EXPECTED DATA PLUS
1996                                     ;RXERR & OVRUN
1997 006772 017701 172720      MOV    @RXDBUF,R1 ;ACTUAL
1998 006776 020001      CMP    R0,R1 ;COMPARE EXP VS. ACT
1999 007000 001401      BEQ    .+4
2000 007002 104002      ERROR  2 ;SPECIFICALLY LOOK AT RXERR &
2001                                     ;OVRUN BITS. . THEY BOTH SHOULD BE SET
2002
2003                                     ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2004                                     ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
2005                                     ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2006                                     ;; (OVRUN,RXERR)
2007                                     ;; MODE: ISYMOD
2008                                     ;; LENGTH: EIGHT
2009                                     ;; CHAR: 377
2010                                     ;;
2011                                     ;; *****
2012 007004 000004      TST15  SCOPE
2013 007006 052777 000400 172712      BIS    #MRESET,@TXCSR ;MASTER RESET
2014 007014 012777 000000 172700      MOV    #ISYMOD,@PARCSR ;SET THE MODE
```

INITIALIZE THE COMMON TAGS

```
2015 007022 052777 000400 172676     BIS    #MRESET,@TXCSR ; MASTER RESET
2016
2017 ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2018 007030 012777 064001 172670     MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
2019
2020 ; SET MODE, # OF BITS, PARITY SENSE, & LOAD SYNC REG
2021 007036 012777 006000 172656     MOV    #ISYMOD!EIGHT!NOPAR!0,@PARCSR
2022 007044 052777 000020 172640     BIS    #SYNSCH,@RXCSR ; SET SYNC SEARCH
2023 ; POKE CLK TO GET RECEIVER INTO SYNCROIZATION...
2024 007052 042777 020000 172646     BIC    #CLK,@TXCSR ; POKE CLK DOWN
2025 007060 052777 020000 172640     BIS    #CLK,@TXCSR ; POKE CLK UP
2026 ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2027 007066 042777 020000 172632     BIC    #CLK,@TXCSR ; POKE CLK DOWN
2028 007074 052777 020000 172624     BIS    #CLK,@TXCSR ; POKE CLK UP
2029 007102 016703 172610     MOV    RXDBUF,R3 ; SET UP FOR ERROR MESSAGE
2030 007106 012700 000377     MOV    #377,R0 ; EXPECTED
2031 007112 012767 000012 172002     MOV    #10,SHIFT ; # OF SHIFTS
2032 007120 012767 001776 172352     MOV    #1776,$TMP1 ; DATA CHAR
2033 007126 004767 007670     JSR    PC,RPOKE ; SHIFT IN THIS CHAR
2034 007132 105777 172554     TSTB   @RXCSR ; RXDONE ?
2035 007136 100401     BMI    .+4
2036 007140 104004     ERROR  4 ; RXDONE SHOULD BE SET
2037 007142 017701 172550     MOV    @RXDBUF,R1 ; ACTUAL
2038 007146 020001     CMP    R0,R1 ; COMPARE EXPECTED VS. ACTUAL
2039 007150 001401     BEQ    .+4
2040 007152 104002     ERROR  2 ; RECEIVED DATA DID NOT MATCH
2041 ; EXPECTED DATA - CHECK MAINT DATA
2042 ; OR RECEIVER LOGIC
2043 007154 012767 000012 171740     MOV    #10,SHIFT ; # OF SHIFTS
2044 007162 012767 001776 172310     MOV    #1776,$TMP1 ; DATA CHAR
2045 007170 004767 007626     JSR    PC,RPOKE ; SHIFT IN THIS CHAR
2046 ; NOW SHIFT IN A SECOND CHAPACTER WITHOUT READING RXDBUF
2047 007174 012767 000012 171720     MOV    #10,SHIFT ; # OF SHIFTS
2048 007202 012767 001776 172270     MOV    #1776,$TMP1 ; DATA CHAR
2049 007210 004767 007606     JSR    PC,RPOKE ; SHIFT IN THIS CHAR
2050 007214 012700 140377     MOV    #140000!377,R0 ; EXPECTED DATA PLUS
2051 ; RXERR & OVRRUN
2052 007220 017701 172472     MOV    @RXDBUF,R1 ; ACTUAL
2053 007224 020001     CMP    R0,R1 ; COMPARE EXP VS. ACT
2054 007226 001401     BEQ    .+4
2055 007230 104002     ERROR  2 ; SPECIFICALLY LOOK AT RXERR &
2056 ; OVRRUN BITS... THEY BOTH SHOULD BE SET
2057
2058 ; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2059 ; RECEIVER SECTION, IT USES THE ERROR FLAGS
2060 ; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2061 ; (OVRRUN, RXERR)
2062 ; MODE: ISYMOD
2063 ; LENGTH: EIGHT
2064 ; CHAR: 0
2065 ;
2066 ; *****
2067 007232 000004     TST16: SCOPE
2068 007234 052777 000400 172464     BIS    #MRESET,@TXCSR ; MASTER RESET
2069 007242 012777 000000 172452     MOV    #ISYMOD,@PARCSR ; SET THE MODE
2070 007250 052777 000400 172450     BIS    #MRESET,@TXCSR ; MASTER RESET
```

```
2071
2072 ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2073 007256 012777 064001 172442 MOV #MTDATA!CLK!MINT!BREAK, @TXCSR
2074
2075 ; SET MODE, # OF BITS, PARITY SENSE, & LOAD SYNC REG
2076 007264 012777 006000 172430 MOV #ISYMOD!EIGHT!NOPAR!D, @PARCSR
2077 007272 052777 000020 172412 BIS #SYNSCH, @RXCSR ; SET SYNC SEARCH
2078 ; POKE CLK TO GET RECEIVER INTO SYNCHRONIZATION...
2079 007300 042777 020000 172420 BIC #CLK, @TXCSR ; POKE CLK DOWN
2080 007306 052777 020000 172412 BIS #CLK, @TXCSR ; POKE CLK UP
2081 ; POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2082 007314 042777 020000 172404 BIC #CLK, @TXCSR ; POKE CLK DOWN
2083 007322 052777 020000 172376 BIS #CLK, @TXCSR ; POKE CLK UP
2084 007330 016703 172362 MOV RXDBUF, R3 ; SET UP FOR ERROR MESSAGE
2085 007334 012700 000000 MOV #D, R0 ; EXPECTED
2086 007340 012767 000012 171554 MOV #10, SHIFT ; # OF SHIFTS
2087 007346 012767 001000 172124 MOV #1000, $TMP1 ; DATA CHAR
2088 007354 004767 007442 JSR PC, RPOKE ; SHIFT IN THIS CHAR
2089 007360 105777 172326 TSTB @RXCSR ; RXDONE ?
2090 007364 100401 BMI +4
2091 007366 104004 ERROR 4 ; RXDONE SHOULD BE SET
2092 007370 017701 172322 MOV @RXDBUF, R1 ; ACTUAL
2093 007374 020001 CMP R0, R1 ; COMPARE EXPECTED VS. ACTUAL
2094 007376 001401 BEQ +4
2095 007400 104002 ERROR 2 ; RECEIVED DATA DID NOT MATCH
2096 ; EXPECTED DATA - CHECK MAINT DATA
2097 ; OR RECEIVER LOGIC
2098 007402 012767 000012 171512 MOV #10, SHIFT ; # OF SHIFTS
2099 007410 012767 001000 172062 MOV #1000, $TMP1 ; DATA CHAR
2100 007416 004767 007400 JSR PC, RPOKE ; SHIFT IN THIS CHAR
2101 ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2102 007422 012767 000012 171472 MOV #10, SHIFT ; # OF SHIFTS
2103 007430 012767 001000 172042 MOV #1000, $TMP1 ; DATA CHAR
2104 007436 004767 007360 JSR PC, RPOKE ; SHIFT IN THIS CHAR
2105 007442 012700 140000 MOV #140000!D, R0 ; EXPECTED DATA PLUS
2106 ; RXERR & OVRUN
2107 007446 017701 172244 MOV @RXDBUF, R1 ; ACTUAL
2108 007452 020001 CMP R0, R1 ; COMPARE EXP VS. ACT
2109 007454 001401 BEQ +4
2110 007456 104002 ERROR 2 ; SPECIFICALLY LOOK AT RXERR &
2111 ; OVRUN BITS... THEY BOTH SHOULD BE SET
2112
2113 ; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2114 ; RECEIVER SECTION, IT USES THE ERROR FLAGS
2115 ; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2116 ; (OVRUN, RXERR)
2117 ; R : SYNEXT
2118 ; LENGTH: FIVE
2119 ; CHAR: 25
2120
2121 ; *****
2122 007460 000004 TST17: SCOPE
2123 007462 052777 000400 172236 BIS #MRESET, @TXCSR ; MASTER RESET
2124 007470 012777 020000 172224 MOV #SYNEXT, @PARCSR ; SET THE MODE
2125 007476 052777 000400 172222 BIS #MRESET, @TXCSR ; MASTER RESET
2126
```

INITIALIZE THE COMMON TAGS

```
2127 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2128 007504 012777 064001 172214 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2129
2130 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2131 007512 012777 020000 172202 MOV #SYNEXT!FIVE!NOPAR!G,@PARCSR
2132 007520 052777 000020 172164 BIS #SYNSCH,@RXCSR ;SET SEARCH SYNC
2133 ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2134 007526 042777 020000 172172 BIC #CLK,@TXCSR ;POKE CLK DOWN
2135 007534 052777 020000 172164 BIS #CLK,@TXCSR ;POKE CLK UP
2136 007542 016703 172150 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2137 007546 012700 000025 MOV #25,R0 ;EXPECTED
2138 007552 012767 000005 171342 MOV #5,SHIFT ;# OF SHIFTS
2139 007560 012767 000025 171712 MOV #25,$TMP1 ;DATA CHAR
2140 007566 004767 007230 JSR PC,RPOKE ;SHIFT IN THIS CHAR
2141 007572 105777 172114 TSTB @RXCSR ;RXDONE ?
2142 007576 100401 BMI .+4
2143 007600 104004 ERROR 4 ;RXDONE SHOULD BE SET
2144 007602 017701 172110 MOV @RXDBUF,R1 ;ACTUAL
2145 007606 020001 CMP R0,R1 ;COMPARE EXPECTED VS. ACTUAL
2146 007610 001401 BEQ .+4
2147 007612 104002 ERROR 2 ;RECEIVED DATA DID NOT MATCH
2148 ;EXPECTED DATA - CHECK MAINT DATA
2149 ;OR RECEIVER LOGIC
2150 007614 012767 000005 171300 MOV #5,SHIFT ;# OF SHIFTS
2151 007622 012767 000025 171650 MOV #25,$TMP1 ;DATA CHAR
2152 007630 004767 007166 JSR PC,RPOKE ;SHIFT IN THIS CHAR
2153 ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2154 007634 012767 000005 171260 MOV #5,SHIFT ;# OF SHIFTS
2155 007642 012767 000025 171630 MOV #25,$TMP1 ;DATA CHAR
2156 007650 004767 007146 JSR PC,RPOKE ;SHIFT IN THIS CHAR
2157 007654 012700 140025 MOV #140000!25,R0 ;EXPECTED DATA PLUS
2158 ;RXERR & OVRRUN
2159 007660 017701 172032 MOV @RXDBUF,R1 ;ACTUAL
2160 007664 020001 CMP R0,R1 ;COMPARE EXP VS. ACT
2161 007666 001401 BEQ .+4
2162 007670 104002 ERROR 2 ;SPECIFICALLY LOOK AT RXERR &
2163 ;OVRRUN BITS... THEY BOTH SHOULD BE SET
2164
2165 ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2166 ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
2167 ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2168 ;; (OVRRUN, RXERR)
2169 ;; MODE: SYNEXT
2170 ;; LENGTH: FIVE
2171 ;; CHAR: 12
2172 ;;
2173 ;*****
2174 007672 000004 TST20: SCOPE
2175 007674 052777 000400 172024 BIS #MRESET,@TXCSR ;MASTER RESET
2176 007702 012777 020000 172012 MOV #SYNEXT,@PARCSR ;SET THE MODE
2177 007710 052777 000400 172010 BIS #MRESET,@TXCSR ;MASTER RESET
2178
2179 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2180 007716 012777 064001 172002 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2181
2182 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
```

```
2183 007724 012777 020000 171770      MOV      #SYNEXT!FIVE!NOPAR!0,@PARCSR
2184 007732 052777 000020 171752      BIS      #SYNSCH,@RXCSR ;SET SEARCH SYNC
2185                                     ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2186 007740 042777 020000 171760      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2187 007746 052777 020000 171752      BIS      #CLK,@TXCSR ;POKE-CLK UP
2188 007754 016703 171736      MOV      RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2189 007760 012700 000012      MOV      #12,R0 ;EXPECTED
2190 007764 012767 000005 171130      MOV      #5,SHIFT ;# OF SHIFTS
2191 007772 012767 000012 171500      MOV      #12,$TMP1 ;DATA CHAR
2192 010000 004767 007015      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
2193 010004 105777 171702      TSTB    @RXCSR ;RXDONE ?
2194 010010 100401      BMI     .+4
2195 010012 104004      ERROR   4 ;RXDONE SHOULD BE SET
2196 010014 017701 171676      MOV      @RXDBUF,R1 ;ACTUAL
2197 010020 020001      CMP     R0,R1 ;COMPARE EXPECTED VS. ACTUAL
2198 010022 001401      BEQ     .+4
2199 010024 104002      ERROR   2 ;RECEIVED DATA DID NOT MATCH
2200                                     ;EXPECTED DATA - CHECK MAINT DATA
2201                                     ;OR RECEIVER LOGIC
2202 010026 012767 000005 171066      MOV      #5,SHIFT ;# OF SHIFTS
2203 010034 012767 000012 171436      MOV      #12,$TMP1 ;DATA CHAR
2204 010042 004767 006754      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
2205                                     ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2206 010046 012767 000005 171046      MOV      #5,SHIFT ;# OF SHIFTS
2207 010054 012767 000012 171416      MOV      #12,$TMP1 ;DATA CHAR
2208 010062 004767 006734      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
2209 010066 012700 140012      MOV      #140000!12,R0 ;EXPECTED DATA PLUS
2210                                     ;RXERR & OVRUN
2211 010072 017701 171620      MOV      @RXDBUF,R1 ;ACTUAL
2212 010076 020001      CMP     R0,R1 ;COMPARE EXP VS. ACT
2213 010100 001401      BEQ     .+4
2214 010102 104002      ERROR   2 ;SPECIFICALLY LOOK AT RXERR &
2215                                     ;OVRUN BITS... THEY BOTH SHOULD BE SET
2216
2217                                     ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2218                                     ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
2219                                     ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2220                                     ;; (OVRUN,RXERR)
2221                                     ;; MODE: SYNEXT
2222                                     ;; LENGTH: FIVE
2223                                     ;; CHAR: 37
2224                                     ;;
2225                                     ;; *****
2226 010104 000004      TST21:  SCOPE
2227 010106 052777 000400 171612      BIS      #MRESET,@TXCSR ;MASTER RESET
2228 010114 012777 020000 171600      MOV      #SYNEXT,@PARCSR ;SET THE MODE
2229 010122 052777 000400 171576      BIS      #MRESET,@TXCSR ;MASTER RESET
2230
2231                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2232 010130 012777 064001 171570      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
2233
2234                                     ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2235 010136 012777 020000 171556      MOV      #SYNEXT!FIVE!NOPAR!0,@PARCSR
2236 010144 052777 000020 171540      BIS      #SYNSCH,@RXCSR ;SET SEARCH SYNC
2237                                     ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2238 010152 042777 020000 171546      BIC      #CLK,@TXCSR ;POKE CLK DOWN
```

INITIALIZE THE COMMON TAGS

```
2239 010160 052777 020000 171540      BIS      #CLK,@TXCSR      ;POKE CLK UP
2240 010166 016703 171524      MOV      RXDBUF,R3      ;SET UP FOR ERROR MESSAGE
2241 010172 012700 000037      MOV      #37,R0      ;EXPECTED
2242 010176 012767 000005 170716      MOV      #5,SHIFT      ;# OF SHIFTS
2243 010204 012767 000037 171266      MOV      #37,$TMP1      ;DATA CHAR
2244 010212 004767 006604      JSR      PC,RPOKE      ;SHIFT IN THIS CHAR
2245 010216 105777 171470      TSTB     @RXCSR      ;RXDONE ?
2246 010222 100401      BMI      +4
2247 010224 104004      ERROR    4      ;RXDONE SHOULD BE SET
2248 010226 017701 171464      MOV      @RXDBUF,R1      ;ACTUAL
2249 010232 020001      CMP      R0,R1      ;COMPARE EXPECTED VS. ACTUAL
2250 010234 001401      BEQ      +4
2251 010236 104002      ERROR    2      ;RECEIVED DATA DID NOT MATCH
2252      ;EXPECTED DATA - CHECK MAINT DATA
2253      ;OR RECEIVER LOGIC
2254 010240 012767 000005 170654      MOV      #5,SHIFT      ;# OF SHIFTS
2255 010246 012767 000037 171224      MOV      #37,$TMP1      ;DATA CHAR
2256 010254 004767 006542      JSR      PC,RPOKE      ;SHIFT IN THIS CHAR
2257      ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2258 010260 012767 000005 170634      MOV      #5,SHIFT      ;# OF SHIFTS
2259 010266 012767 000037 171204      MOV      #37,$TMP1      ;DATA CHAR
2260 010274 004767 006522      JSR      PC,RPOKE      ;SHIFT IN THIS CHAR
2261 010300 012700 140037      MOV      #140000!37,R0      ;EXPECTED DATA PLUS
2262      ;RXERR & OVRRUN
2263 010304 017701 171406      MOV      @RXDBUF,R1      ;ACTUAL
2264 010310 020001      CMP      R0,R1      ;COMPARE EXP VS. ACT
2265 010312 001401      BEQ      +4
2266 010314 104002      ERROR    2      ;SPECIFICALLY LOOK AT RXERR &
2267      ;OVRRUN BITS... THEY BOTH SHOULD BE SET
2268
2269      ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2270      ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
2271      ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2272      ;; (OVRRUN, RXERR)
2273      ;; MODE: SYNEXT
2274      ;; LENGTH: FIVE
2275      ;; CHAR: 0
2276      ;;
2277      ;; *****
2278 010316 000004      TST2:    SCOPE
2279 010320 052777 000400 171400      BIS      #MRESET,@TXCSR ;MASTER RESET
2280 010326 012777 020000 171366      MOV      #SYNEXT,@PARCSR ;SET THE MODE
2281 010334 052777 000400 171364      BIS      #MRESET,@TXCSR ;MASTER RESET
2282
2283      ;SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2284 010342 012777 064001 171356      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
2285
2286      ;SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2287 010350 012777 020000 171344      MOV      #SYNEXT!FIVE!NOPAR!0,@PARCSR
2288 010356 052777 000020 171326      BIS      #SYNSCH,@RXCSR ;SET SEARCH SYNC
2289
2290      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2290 010364 042777 020000 171334      BIC      #CLK,@TXCSR      ;POKE CLK DOWN
2291 010372 052777 020000 171326      BIS      #CLK,@TXCSR      ;POKE CLK UP
2292 010400 016703 171312      MOV      RXDBUF,R3      ;SET UP FOR ERROR MESSAGE
2293 010404 012700 000000      MOV      #0,R0      ;EXPECTED
2294 010410 012767 000005 170504      MOV      #5,SHIFT      ;# OF SHIFTS
```

INITIALIZE THE COMMON TAGS

```
2295 010416 012767 000000 171054      MOV    #0,STMP1      ;DATA CHAR
2296 010424 004767 006372              JSR    PC,RPOKE      ;SHIFT IN THIS CHAR
2297 010430 105777 171256              TSTB   @RXCSR ;RXDONE ?
2298 010434 100401              BMI    .+4
2299 010436 104004              ERROR  4 ;RXDONE SHOULD BE SET
2300 010440 017701 171252      MOV    @RXDBUF,R1    ;ACTUAL
2301 010444 020001              CMP    R0,R1 ;COMPARE EXPECTED VS. ACTUAL
2302 010446 001401              BEQ    .+4
2303 010450 104002              ERROR  2 ;RECEIVED DATA DID NOT MATCH
2304                                ;EXPECTED DATA - CHECK MAINT DATA
2305                                ;OR RECEIVER LOGIC
2306 010452 012767 000005 170442      MOV    #5,SHIFT      ;# OF SHIFTS
2307 010460 012767 000000 171012      MOV    #0,STMP1      ;DATA CHAR
2308 010466 004767 006330              JSR    PC,RPOKE      ;SHIFT IN THIS CHAR
2309                                ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2310 010472 012767 000005 170422      MOV    #5,SHIFT      ;# OF SHIFTS
2311 010500 012767 000000 170772      MOV    #0,STMP1      ;DATA CHAR
2312 010506 004767 006310              JSR    PC,RPOKE      ;SHIFT IN THIS CHAR
2313 010512 012700 140000      MOV    #140000!0,R0  ;EXPECTED DATA PLUS
2314                                ;RXERR & OVRRUN
2315 010516 017701 171174      MOV    @RXDBUF,R1    ;ACTUAL
2316 010522 020001              CMP    R0,R1 ;COMPARE EXP VS. ACT
2317 010524 001401              BEQ    .+4
2318 010526 104002              ERROR  2 ;SPECIFICALLY LOOK AT RXERR &
2319                                ;OVRRUN BITS... THEY BOTH SHOULD BE SET
2320
2321                                ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2322                                ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
2323                                ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2324                                ;; (OVRRUN, RXERR)
2325                                ;; MODE: SYNEXT
2326                                ;; LENGTH: SIX
2327                                ;; CHAR: 25
2328                                ;;
2329                                ;; *****
2330 010530 000004      TST23: SCOPE
2331 010532 052777 000400 171166      BIS    #MRESET,@TXCSR ;MASTER RESET
2332 010540 012777 020000 171154      MOV    #SYNEXT,@PARCSR ;SET THE MODE
2333 010546 052777 000400 171152      BIS    #MRESET,@TXCSR ;MASTER RESET
2334
2335                                ;SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2336 010554 012777 064001 171144      MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
2337
2338                                ;SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2339 010562 012777 022000 171132      MOV    #SYNEXT!SIX!NOPAR!0,@PARCSR
2340 010570 052777 000020 171114      BIS    #SYNSCH,@RXCSR ;SET SEARCH SYNC
2341                                ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2342 010576 042777 020000 171122      BIC    #CLK,@TXCSR ;POKE CLK DOWN
2343 010604 052777 020000 171114      BIS    #CLK,@TXCSR ;POKE CLK UP
2344 010612 016703 171100      MOV    RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2345 010616 012700 000025      MOV    #25,R0 ;EXPECTED
2346 010622 012767 000006 170272      MOV    #6,SHIFT      ;# OF SHIFTS
2347 010630 012767 000025 170642      MOV    #25,STMP1     ;DATA CHAR
2348 010636 004767 006160              JSR    PC,RPOKE      ;SHIFT IN THIS CHAR
2349 010642 105777 171044              TSTB   @RXCSR ;RXDONE ?
2350 010646 100401              BMI    .+4
```

```

2351 010650 104004          ERROR 4 ;RXDONE SHOULD BE SET
2352 010652 017701 171040  MOV @RXDBUF,R1 ;ACTUAL
2353 010656 020001          CMP R0,R1 ;COMPARE EXPECTED VS. ACTUAL
2354 010660 001401          BEQ .+4
2355 010662 104002          ERROR 2 ;RECEIVED DATA DID NOT MATCH
2356                                ;EXPECTED DATA - CHECK MAINT DATA
2357                                ;OR RECEIVER LOGIC
2358 010664 012767 000006 170230  MOV #6,SHIFT ;# OF SHIFTS
2359 010672 012767 000025 170600  MOV #25,$TMP1 ;DATA CHAR
2360 010700 004767 006115          JSR PC,RPOKE ;SHIFT IN THIS CHAR
2361                                ,NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2362 010704 012767 000006 170210  MOV #6,SHIFT ;# OF SHIFTS
2363 010712 012767 000025 170560  MOV #25,$TMP1 ;DATA CHAR
2364 010720 004767 006076          JSR PC,RPOKE ;SHIFT IN THIS CHAR
2365 010724 012700 140025          MOV #140000!25,R0 ;EXPECTED DATA PLUS
2366                                ;RXERR & OVRRUN
2367 010730 017701 170762          MOV @RXDBUF,R1 ;ACTUAL
2368 010734 020001          CMP R0,R1 ;COMPARE EXP VS. ACT
2369 010736 001401          BEQ .+4
2370 010740 104002          ERROR 2 ;SPECIFICALLY LOOK AT RXERR &
2371                                ;OVRRUN BITS... THEY BOTH SHOULD BE SET
2372
2373                                ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2374                                ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
2375                                ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2376                                ;; (OVRRUN, RXERR)
2377                                ;; MODE: SYNEXT
2378                                ;; LENGTH: SIX
2379                                ;; CHAR: 52
2380                                ;;
2381                                ;; *****
2382 010742 000004          TST24: SCOPE
2383 010744 052777 000400 170754  BIS #MRESET,@TXCSR ;MASTER RESET
2384 010752 012777 020000 170742  MOV #SYNEXT,@PARCSR ;SET THE MODE
2385 010760 052777 000400 170740  BIS #MRESET,@TXCSR ;MASTER RESET
2386
2387                                ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2388 010766 012777 064001 170732  MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2389
2390                                ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2391 010774 012777 022000 170720  MOV #SYNEXT!SIX!NOPAR!0,@PARCSR
2392 011002 052777 000020 170702  BIS #SYNSCH,@RXCSR ;SET SEARCH SYNC
2393                                ; POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2394 011010 042777 020000 170710  BIC #CLK,@TXCSR ;POKE CLK DOWN
2395 011016 052777 020000 170702  BIS #CLK,@TXCSR ;POKE CLK UP
2396 011024 016703 170666          MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2397 011030 012700 000052          MOV #52,R0 ;EXPECTED
2398 011034 012767 000006 170060  MOV #6,SHIFT ;# OF SHIFTS
2399 011042 012767 000052 170430  MOV #52,$TMP1 ;DATA CHAR
2400 011050 004767 005746          JSR PC,RPOKE ;SHIFT IN THIS CHAR
2401 011054 105777 170632          TSTB @RXCSR ;RXDONE ?
2402 011060 100401          BMI .+4
2403 011062 104004          ERROR 4 ;RXDONE SHOULD BE SET
2404 011064 017701 170626          MOV @RXDBUF,R1 ;ACTUAL
2405 011070 020001          CMP R0,R1 ;COMPARE EXPECTED VS. ACTUAL
2406 011072 001401          BEQ .+4
  
```

```

2407 011074 104002          ERROR 2          ;RECEIVED DATA DID NOT MATCH
2408                                ;EXPECTED DATA - CHECK MAINT DATA
2409                                ;OR RECEIVER LOGIC
2410 011076 012767 000006 170016  MOV #6,SHIFT          ;# OF SHIFTS
2411 011104 012767 000052 170366  MOV #52,$TMP1         ;DATA CHAR
2412 011112 004767 005704          JSR PC,RPOKE          ;SHIFT IN THIS CHAR
2413                                ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2414 011116 012767 000006 167776  MOV #6,SHIFT          ;# OF SHIFTS
2415 011124 012767 000052 170346  MOV #52,$TMP1         ;DATA CHAR
2416 011132 004767 005664          JSR PC,RPOKE          ;SHIFT IN THIS CHAR
2417 011136 012700 140052          MOV #140000!52,RO     ;EXPECTED DATA PLUS
2418                                ;RXERR & OVRRUN
2419 011142 017701 170550          MOV @RXDBUF,R1        ;ACTUAL
2420 011146 020001          CMP RO,R1             ;COMPARE EXP VS. ACT
2421 011150 001401          BEQ .+4
2422 011152 104002          ERROR 2          ;SPECIFICALLY LOOK AT RXERR &
2423                                ;OVRRUN BITS... THEY BOTH SHOULD BE SET
2424
2425                                ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2426                                ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
2427                                ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2428                                ;; (OVRRUN,RXERR)
2429                                ;; MODE: SYNEXT
2430                                ;; LENGTH: SIX
2431                                ;; CHAR: 77
2432                                ;;
2433                                ;*****
2434 011154 000004          TST25: SCOPE
2435 011156 052777 000400 170542  BIS #MRESET,@TXCSR   ;MASTER RESET
2436 011164 012777 020000 170530  MOV #SYNEXT,@PARCSR  ;SET THE MODE
2437 011172 052777 000400 170526  BIS #MRESET,@TXCSR   ;MASTER RESET
2438
2439                                ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2440 011200 012777 064001 170520  MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2441
2442                                ;SET MODE , # OF BITS,PARITY SENSE,&LOAD SYNC REG
2443 011206 012777 022000 170506  MOV #SYNEXT!SIX!NOPAR!0,@PARCSR
2444 011214 052777 000020 170470  BIS #SYNSCH,@RXCSR   ;SET SEARCH SYNC
2445                                ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2446 011222 042777 020000 170476  BIC #CLK,@TXCSR      ;POKE CLK DOWN
2447 011230 052777 020000 170470  BIS #CLK,@TXCSR      ;POKE CLK UP
2448 011236 016703 170454          MOV RXDBUF,R3        ;SET UP FOR ERROR MESSAGE
2449 011242 012700 000077          MOV #77,RO           ;EXPECTED
2450 011246 012767 000006 167646  MOV #6,SHIFT          ;# OF SHIFTS
2451 011254 012767 000077 170216  MOV #77,$TMP1         ;DATA CHAR
2452 011262 004767 005534          JSR PC,RPOKE          ;SHIFT IN THIS CHAR
2453 011266 105777 170420          TSTB @RXCSR          ;RXDONE ?
2454 011272 100401          BMI .+4
2455 011274 104004          ERROR 4          ;RXDONE SHOULD BE SET
2456 011276 017701 170414          MOV @RXDBUF,R1        ;ACTUAL
2457 011302 020001          CMP RO,R1             ;COMPARE EXPECTED VS. ACTUAL
2458 011304 001401          BEQ .+4
2459 011306 104002          ERROR 2          ;RECEIVED DATA DID NOT MATCH
2460                                ;EXPECTED DATA - CHECK MAINT DATA
2461                                ;OR RECEIVER LOGIC
2462 011310 012767 000006 167604  MOV #6,SHIFT          ;# OF SHIFTS
  
```

INITIALIZE THE COMMON TAGS

```
2463 011316 012767 000077 170154      MOV    #77,$TMP1      ;DATA CHAR
2464 011324 004767 005472              JSR    PC,RPOKE       ;SHIFT IN THIS CHAR
2465                                ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2466 011330 012767 000006 167564      MOV    #6,SHIFT       ;# OF SHIFTS
2467 011336 012767 000077 170134      MOV    #77,$TMP1     ;DATA CHAR
2468 011344 004767 005452              JSR    PC,RPOKE       ;SHIFT IN THIS CHAR
2469 011350 012700 140077              MOV    #140000!77,RO  ;EXPECTED DATA PLUS
2470                                ;RXERR & OVRRUN
2471 011354 017701 170336              MOV    @RXDBUF,R1    ;ACTUAL
2472 011360 020001                      CMP    RO,R1         ;COMPARE EXP VS. ACT
2473 011362 001401                      BEQ    .+4
2474 011364 104002                      ERROR  2             ;SPECIFICALLY LOOK AT RXERR &
2475                                ;OVRRUN BITS... THEY BOTH SHOULD BE SET
2476
2477                                ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2478                                ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
2479                                ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2480                                ;; (OVRRUN,RXERR)
2481                                ;; MODE: SYNEXT
2482                                ;; LENGTH: SIX
2483                                ;; CHAR: 0
2484                                ;;
2485                                ;; *****
2486 011366 000004                      TST26: SCOPE
2487 011370 052777 000400 170330      BIS    #MRESET,@TXCSR ;MASTER RESET
2488 011376 012777 020000 170316      MOV    #SYNEXT,@PARCSR ;SET THE MODE
2489 011404 052777 000400 170314      BIS    #MRESET,@TXCSR ;MASTER RESET
2490
2491                                ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2492 011412 012777 064001 170306      MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
2493
2494                                ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2495 011420 012777 022000 170274      MOV    #SYNEXT!SIX!NOPAR!0,@PARCSR
2496 011426 052777 000020 170256      BIS    #SYNSCH,@RXCSR ;SET SEARCH SYNC
2497                                ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2498 011434 042777 020000 170264      BIC    #CLK,@TXCSR   ;POKE CLK DOWN
2499 011442 052777 020000 170256      BIS    #CLK,@TXCSR   ;POKE CLK UP
2500 011450 016703 170242              MOV    RXDBUF,R3     ;SET UP FOR ERROR MESSAGE
2501 011454 012700 000000              MOV    #0,RO        ;EXPECTED
2502 011460 012767 000006 167434      MOV    #6,SHIFT     ;# OF SHIFTS
2503 011466 012767 000000 170004      MOV    #0,$TMP1     ;DATA CHAR
2504 011474 004767 005322              JSR    PC,RPOKE     ;SHIFT IN THIS CHAR
2505 011500 105777 170206              TSTB  @RXCSR        ;RXDONE ?
2506 011504 100401                      BMI    .+4
2507 011506 104004                      ERROR  4             ;RXDONE SHOULD BE SET
2508 011510 017701 170202              MOV    @RXDBUF,R1   ;ACTUAL
2509 011514 020001                      CMP    RO,R1        ;COMPARE EXPECTED VS. ACTUAL
2510 011516 001401                      BEQ    .+4
2511 011520 104002                      ERROR  2             ;RECEIVED DATA DID NOT MATCH
2512                                ;EXPECTED DATA - CHECK MAINT DATA
2513                                ;OR RECEIVER LOGIC
2514 011522 012767 000006 167372      MOV    #6,SHIFT     ;# OF SHIFTS
2515 011530 012767 000000 167742      MOV    #0,$TMP1     ;DATA CHAR
2516 011536 004767 005260              JSR    PC,RPOKE     ;SHIFT IN THIS CHAR
2517                                ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2518 011542 012767 000006 167352      MOV    #6,SHIFT     ;# OF SHIFTS
```



```

2575 012000 017701 167712      MOV    @RXDBUF,R1      ;ACTUAL
2576 012004 020001      CMP    RD,R1          ;COMPARE EXP VS. ACT
2577 012006 001401      BEQ    +4
2578 012010 104002      ERROR  2              ;SPECIFICALLY LOOK AT RXERR &
2579                                ;OVRRUN BITS... THEY BOTH SHOULD BE SET
2580
2581                                ;; THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2582                                ;; RECEIVER SECTION, IT USES THE ERROR FLAGS
2583                                ;; TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2584                                ;; (OVRRUN, RXERR)
2585                                ;; MODE: SYNEXT
2586                                ;; LENGTH: SEVEN
2587                                ;; CHAR: 52
2588                                ;;
2589                                ;; *****
2590 012012 000004      TST30: SCOPE
2591 012014 052777 000400 167704  BIS    #MRESET,@TXCSR ;MASTER RESET
2592 012022 012777 020000 167672  MOV    #SYNEXT,@PARCSR ;SET THE MODE
2593 012030 052777 000400 167670  BIS    #MRESET,@TXCSR ;MASTER RESET
2594
2595                                ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2596 012036 012777 064001 167662  MOV    #MTDATA!CLK!MINT!BREAK,@TXCSR
2597
2598                                ; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2599 012044 012777 024000 167650  MOV    #SYNEXT!SEVEN!NOPAR!0,@PARCSR
2600 012052 052777 000020 167632  BIS    #SYNSCH,@RXCSR ;SET SEARCH SYNC
2601                                ; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2602 012060 042777 020000 167640  BIC    #CLK,@TXCSR    ;POKE CLK DOWN
2603 012066 052777 020000 167632  BIS    #CLK,@TXCSR    ;POKE CLK UP
2604 012074 016703 167616  MOV    RXDBUF,R3      ;SET UP FOR ERROR MESSAGE
2605 012100 012700 000052  MOV    #52,RD         ;EXPECTED
2606 012104 012767 000007 167010  MOV    #7,SHIFT       ;# OF SHIFTS
2607 012112 012767 000052 167360  MOV    #52,$TMP1      ;DATA CHAR
2608 012120 004767 004676  JSR    PC,RPOKE       ;SHIFT IN THIS CHAR
2609 012124 105777 167562  TSTB   @RXCSR ;RXDONE ?
2610 012130 100401  BMI    +4
2611 012132 104004  ERROR  4              ;RXDONE SHOULD BE SET
2612 012134 017701 167556  MOV    @RXDBUF,R1     ;ACTUAL
2613 012140 020001  CMP    RD,R1          ;COMPARE EXPECTED VS. ACTUAL
2614 012142 001401  BEQ    +4
2615 012144 104002  ERROR  2              ;RECEIVED DATA DID NOT MATCH
2616                                ;EXPECTED DATA - CHECK MAINT DATA
2617                                ;OR RECEIVER LOGIC
2618 012146 012767 000007 166746  MOV    #7,SHIFT       ;# OF SHIFTS
2619 012154 012767 000052 167316  MOV    #52,$TMP1      ;DATA CHAR
2620 012162 004767 004634  JSR    PC,RPOKE       ;SHIFT IN THIS CHAR
2621                                ; NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2622 012166 012767 000007 166726  MOV    #7,SHIFT       ;# OF SHIFTS
2623 012174 012767 000052 167276  MOV    #52,$TMP1      ;DATA CHAR
2624 012202 004767 004614  JSR    PC,RPOKE       ;SHIFT IN THIS CHAR
2625 012206 012700 140052  MOV    #140000!52,RD  ;EXPECTED DATA PLUS
2626                                ;RXERR & OVRRUN
2627 012212 017701 167500  MOV    @RXDBUF,R1     ;ACTUAL
2628 012216 020001  CMP    RD,R1          ;COMPARE EXP VS. ACT
2629 012220 001401  BEQ    +4
2630 012222 104002  ERROR  2              ;SPECIFICALLY LOOK AT RXERR &

```

```

2631 ;OVRUN BITS... THEY BOTH SHOULD BE SET
2632
2633
2634 ;END OF PASS
2635 ;TYPE NAME OF TEST
2636 ;UPDATE PASS COUNT
2637 ;CHECK FOR EXIT TO ACT-11
2638 ;RESTART TEST
2639
2640 012224 000004 .EOP: SCOPE
2641 012226 004767 000340 JSR PC,CKSWR
2642 012232 104401 TYPE ;TYPE NAME OF TEST
2643 012234 015362 MEPASS
2644 012236 104413 012470 CONVRT ,OUTCRY
2645 012242 104401 015201 TYPE ,DEVICE
2646 012246 105767 166700 TSTB MULTD ;ARE YOU RUNNING MULTIPLE DEVICES ?
2647 012252 001511 BEQ CCC ;NO, JUMP AROUND
2648 012254 005767 166706 TST ACTREG ;ARE ANY DEVICES ACTIVE ?
2649 012260 001007 BNE RUNIT ;YES
2650 012262 104401 015213 TYPE ,MCOV ;NO
2651 012266 016700 166674 MOV ACTREG,RO ;DISPLAY ACTREG
2652 012272 000000 HALT ;SELECT SOMETHING TO RUN @ ACTREG:
2653 ;SELECT SWITCHES & HIT CONTINUE (PUT SW00 =1)
2654 012274 000167 167652 JMP START ;START OVER AGAIN..... YOU DESELECTED EVERYTHING
2655 012300 062767 000010 166646 RUNIT: ADD #10,BASEADD ;NEXT BLOCK (ADDRESSES)
2656 012306 062767 000010 166646 ZERO: ADD #10,BASEIV ;NEXT BLOCK (VECTORS)
2657 012314 000241 CLC
2658 012316 006167 166646 ROL ROTADD ;UP DATE ROTATING POINTER
2659 012322 103410 BCS Z5 ;IS IT THE LAST DEVICE
2660 ;TO BE TESTED IN THIS PASS ?
2661 012324 036767 166640 166634 BIT ROTADD,ACTREG ;TEST THIS DEVICE FOR ACTIVE STATUS
2662 012332 001762 BEQ RUNIT ;IF NOT ACTIVE, TRY NEXT ADDRESS
2663 012334 004767 000034 JSR PC,REPLAY ;CALCULATE NEW PARAMETERS
2664 012340 000167 000210 JMP RESTRT ;YES IT WAS ACTIVE, TEST THIS DEVICE
2665 012344 012767 000001 166616 Z5: MOV #1,ROTADD ;OK!,NOW SET UP ROTATING
2666 ;POINTER FOR NEXT MULTIPLE PASS
2667 012352 016767 166600 166574 MOV KEEPADD,BASEADD ;RESTORE BASE ADDRESS
2668 012360 016767 166600 166574 MOV KEEPIV,BASEIV ;RESTORE BASE INTERRUPT VECTORS
2669 012366 004767 000002 JSR PC,REPLAY ;CALC NEW PARAMETERS
2670 012372 000441 BR CCC ;JUMP AROUND REPLAY
2671 012374 016767 166554 004416 REPLAY: MOV BASEADD,DUBASE ;SET UP FOR NEW ADDRESSES
2672 012402 004767 004260 JSR PC,DUADDR ;CREATE NEW ADDRESSES
2673 012406 016767 166550 167322 MOV BASEIV,DURIV ;CREATE DURIV
2674 012414 062767 000002 166540 ADD #2,BASEIV
2675 012422 016767 166534 167310 MOV BASEIV,DURIS ;CREATE DURIS
2676 012430 062767 000002 166524 ADD #2,BASEIV
2677 012436 016767 166520 167276 MOV BASEIV,DUTIV ;CREATE DUTIV
2678 012444 062767 000002 166510 ADD #2,BASEIV
2679 012452 016767 166504 167264 MOV BASEIV,DUTIS ;CREATE DUTIS
2680 012460 016767 167252 166474 MOV DURIV,BASEIV ;RESTORE
2681 012466 000207 RTS PC
2682
2683 012470 000001 OUTCRY: 1
2684 012472 006 002 . BYTE 6,2
2685 012474 001712 RXCSR
2686
  
```

```

2687 012476          CCC:
2688 012476 005067 166700      CLR      $TSTNM      ; CLEAR TEST NUMBER
2689 012502 005067 166710      CLR      $ERRPC      ; CLEAR LAST ERROR PC
2690 012506 005067 166671      CLR      $ERFLG      ; CLEAR ERROR FLAG
2691 012512 005267 166374      INC      PASCNT      ; UPDATE PASS COUNT
2692 012516 016767 166370 166356  MOV      PASCNT, LIGHTS ; DISPLAY PASS COUNT
2693 012524 016767 166362 167002  MOV      PASCNT, $PASS ; PASS COUNT TO APT
2694 012532 013701 000042      MOV      @#42, R1    ; CHECK FOR ACT-11 OR DDP
2695 012536 001406          BEQ      RESTRT      ; IF NO CONTINUE TESTING
2696 012540 000005          RESET
2697 012542 000005          RESET
2698 012544 004711          SENDAD: JSR      PC, (R1)
2699 012546 000240          NOP
2700 012550 000240          NOP
2701 012552 000240          NOP
2702 012554          RESTRT:
2703 012554 012767 003376 166624  MOV      #TST1+2, $LPADR ; LOAD LAST ADDR
2704 012562 004767 000004      JSR      PC, CKSWR
2705 012566 000167 170516      JMP      .BEGIN
2706
2707          ; CHECK SWITCH REGISTER ROUTINE.
2708          ; CHECKS TO ALLOW FOR < G > TO ALLOW
2709          ; THE CHANGING OF LOCATION 176
2710
2711 012572 005737 000042          CKSWR: TST      @#42
2712 012576 001040          BNE      OUT
2713 012600 022767 000176 166632  CMP      #SWREG, SWR ; SOFTWARE SWR PRESENT?
2714 012606 001034          BNE      OUT ; NO--LEAVE
2715 012610 105777 166630          TSTB   @$TKS ; CHECK TTY READY
2716 012614 100031          BPL      OUT ; NO--LEAVE
2717 012616 017767 166624 000422  MOV      @$TKB, .MSG ; GET CHARACTER
2718 012624 042767 177600 000414  BIC      #177600, .MSG ; STRIP JUNK
2719 012632 122767 000007 000406  CMPB   #7, .MSG ; IS IT < G > ?
2720 012640 001017          BNE      OUT ; NO
2721 012642 104401 015767          TYPE   , MCNTG
2722 012646 005137 012706          CNTLU: COM     @#RDSW
2723 012652 104401 015777          TYPE   , MMSWR
2724 012656 104413          CONVRT
2725 012660 012710          SWREGL
2726 012662 104406 016010          INSTR, MMNEW
2727 012666 104410          PARAM
2728 012670 000000          0
2729 012672 177777          177777
2730 012674 000176          SWREG
2731 012676 000 001          . BYTE 0, 1
2732 012700 005037 012706          OUT:  CLR     @#RDSW
2733 012704 000207          RTS      PC
2734 012706 000000          RDSW:  . WORD 0
2735 012710 000001          SWREGL: 1
2736 012712 006 002          . BYTE 6, 2
2737 012714 000176          SWREG
2738
2739 012716 000005          5
2740
2741          ; CHECK FOR FREEZE ON CURRENT DATA
2742

```

```

2743 012720 004767 177646 .SCOP1: JSR PC,CKSWR
2744 012724 032777 001000 166506 BIT #SW09,ASWR
2745 012732 001402 BEQ 1$
2746 012734 016716 166150 MOV LOCK,(SP)
2747 012740 000002 1$: RTI
2748 .SBTTL TYPE ROUTINE
2749
2750 ;*****
2751 ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2752 ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2753 ;*NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2754 ;*NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2755 ;*NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2756 ;*
2757 ;*CALL:
2758 ;*1) USING A TRAP INSTRUCTION
2759 ;* TYPE ,MESADR ;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2760 ;*OR
2761 ;* TYPE
2762 ;* MESADR
2763 ;*
2764
2765 012742 105767 166511 $TYPE: TSTB $TPFLG ;IS THERE A TERMINAL?
2766 012746 100002 BPL 1$ ;BR IF YES
2767 012750 000000 HALT ;HALT HERE IF NO TERMINAL
2768 012752 000430 BR 3$ ;LEAVE
2769 012754 010046 1$: MOV RO,-(SP) ;SAVE RO
2770 012756 017600 000002 MOV @2(SP),RO ;GET ADDRESS OF ASCIZ STRING
2771 012762 122767 000001 166556 CMPB #APTENV,$ENV ;RUNNING IN APT MODE
2772 012770 001011 BNE 62$ ;NO,GO CHECK FOR APT CONSOLE
2773 012772 132767 000100 166547 BITB #APTPOOL,$ENVM ;SPOOL MESSAGE TO APT
2774 013000 001405 BEQ 62$ ;NO,GO CHECK FOR CONSOLE
2775 013002 010067 000004 MOV RO,61$ ;SETUP MESSAGE ADDRESS FOR APT
2776 013006 004767 000006 JSR PC,$ATY3 ;SPOOL MESSAGE TO APT
2777 013012 000000 61$: .WORD 0 ;MESSAGE ADDRESS
2778 013014 132767 000040 166525 62$: BITB #APTCSUP,$ENVM ;APT CONSOLE SUPPRESSED
2779 013022 001003 BNE 60$ ;YES,SKIP TYPE OUT
2780 013024 112046 2$: MOVB (RO)+,-(SP) ;PUSH CHARACTER TO BE TYPED ONTO STACK
2781 013026 001005 BNE 4$ ;BR IF IT ISN'T THE TERMINATOR
2782 013030 005726 TST (SP)+ ;IF TERMINATOR POP IT OFF THE STACK
2783 013032 012600 60$: MOV (SP)+,RO ;RESTORE RO
2784 013034 062716 000002 3$: ADD #2,(SP) ;ADJUST RETURN PC
2785 013040 000002 RTI ;RETURN
2786 013042 122716 000011 4$: CMPB #HT,(SP) ;BRANCH IF <HT>
2787 013046 001430 BEQ 8$
2788 013050 122716 000200 CMPB #CRLF,(SP) ;BRANCH IF NOT <CRLF>
2789 013054 001006 BNE 5$
2790 013056 005726 TST (SP)+ ;POP <CR><LF> EQUIV
2791 013060 104401 TYPE ;TYPE A CR AND LF
2792 013062 001523 $CRLF
2793 013064 105067 000130 CLRB $CHARCNT ;CLEAR CHARACTER COUNT
2794 013070 000755 BR 2$ ;GET NEXT CHARACTER
2795 013072 004767 000056 5$: JSR PC,$TYPEC ;GO TYPE THIS CHARACTER
2796 013076 126726 166354 6$: CMPB $FILLC,(SP)+ ;IS IT TIME FOR FILLER CHARS.?
2797 013102 001350 BNE 2$ ;IF NO GO GET NEXT CHAR.
2798 013104 016746 166344 MOV $NULL,-(SP) ;GET # OF FILLER CHARS. NEEDED

```

```

2799
2800 013110 105366 000001      7$:  DECB  1(SP)      ;; AND THE NULL CHAR.
2801 013114 002770              BLT   6$           ;; DOES A NULL NEED TO BE TYPED?
2802 013116 004767 000032      JSR   PC,$TYPEC    ;; BR IF NO--GO POP THE NULL OFF OF STACK
2803 013122 105367 000072      DECB  $CHARCNT     ;; GO TYPE A NULL
2804 013126 000770              BR    7$           ;; DO NOT COUNT AS A COUNT
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
  
```

; HORIZONTAL TAB PROCESSOR

```

2808 013130 112716 000040      8$:  MOV   #' ,(SP)    ;; REPLACE TAB WITH SPACE
2809 013134 004767 000014      9$:  JSR   PC,$TYPEC    ;; TYPE A SPACE
2810 013140 132767 000007 000052  BITB  #7,$CHARCNT    ;; BRANCH IF NOT AT
2811 013146 001372              BNE   9$           ;; TAB STOP
2812 013150 005726              TST   (SP)+        ;; POP SPACE OFF STACK
2813 013152 000724              BR    2$           ;; GET NEXT CHARACTER
2814 013154 105777 166270      $TYPEC: TSTB @ $TPS    ;; WAIT UNTIL PRINTER IS READY
2815 013160 100375              BPL   $TYPEC
2816 013162 116677 000002 166262  MOV   2(SP),@ $TPB  ;; LOAD CHAR TO BE TYPED INTO DATA REG.
2817 013170 122766 000015 000002  CMPB  #CR,2(SP)     ;; IS CHARACTER A CARRIAGE RETURN?
2818 013176 001003              BNE   1$           ;; BRANCH IF NO
2819 013200 105067 000014      CLRB  $CHARCNT     ;; YES--CLEAR CHARACTER COUNT
2820 013204 000406              BR    $TYPEX
2821 013206 122766 000012 000002  1$:  CMPB  #LF,2(SP)   ;; IS CHARACTER A LINE FEED?
2822 013214 001402              BEQ   $TYPEX       ;; BRANCH IF YES
2823 013216 105227              INCB  (PC)+        ;; COUNT THE CHARACTER
2824 013220 000000      $CHARCNT: .WORD 0  ;; CHARACTER COUNT STORAGE
2825 013222 000207      $TYPEX:  RTS      PC
2826
2827
2828
2829
  
```

; ASCII STRING INPUT ROUTINE

```

2830 013224 017667 000000 000014  INSTR: MOV   @ (SP),.MSG  ;; PICK UP MESSAGE
2831 013232 062716 000002          ADD   #2,(SP)        ;; JUMP AROUND MESSAGE FOR RTI
2832 013236 105767 166304          TSTB  $ENV          ;; APT CONTROL
2833 013242 001036          BNE   INSTR2       ;; YES NO TYPE
2834 013244 104401          .INST1: TYPE
2835 013246 000000          .MSG:  0
2836 013250 012704 016022          MOV   #INBUF,R4    ;; GET STARTING LOC OF INBUF
2837 013254 012703 000007          MOV   #7,R3        ;; MAX # OF CHARS
2838 013260 105777 166160      1$:  TSTB  @ $TKS      ;; TTY FLAG
2839 013264 100375          BPL   1$
2840 013266 117714 166154          MOV   @ $TKB,(R4)  ;; TAKE CHAR
2841 013272 142714 000200          BICB  #200,(R4)   ;; STRIP
2842 013276 121427 000025          CMPB  (R4),#25     ;; IS IT < G>
2843 013302 001760          BEQ   .INST1
2844 013304 122427 000015          CMPB  (R4)+,#15    ;; CHECK FOR CR
2845 013310 001413          BEQ   INSTR2
2846 013312 105777 166132      2$:  TSTB  @ $TPS      ;; TEST FLAG
2847 013316 100375          BPL   2$
2848 013320 117777 166122 166124  MOV   @ $TKB,@ $TPB ;; ECHO CHARACTER
2849 013326 005303          DEC   R3           ;; DID YOU TYPE TOO MANY CHARS ?
2850 013330 001353          BNE   1$
2851 013332 104401          .INSTE: TYPE
2852 013334 015307          MQM   ;?
2853 013336 000742          BR    .INST1      ;; RETRY
2854 013340 000002      INSTR2: RTI
  
```

```

2855
2856           ; CONVERT ASCII STRING TO OCTAL
2857
2858 013342 011605   .PARAM: MOV      (SP),R5 ; PUT CONTENTS OF SP INTO R5
2859 013344 012567 000162   MOV      (R5)+,LOLIM ; PUT LOW LIMIT INTO LOLIM
2860 013350 012567 000160   MOV      (R5)+,HILIM ; PUT HIGH LIMIT INTO HILIM
2861 013354 012567 000156   MOV      (R5)+,DEVADR ; PUT STORE LOC INTO DEVADR
2862 013360 112567 000154   MOV      (R5)+,LOBITS ; PUT MASK INTO LOBITS
2863 013364 112567 000151   MOV      (R5)+,ADRCNT ; PUT COUNT INTO ADRCNT
2864 013370 010516   MOV      R5,(SP) ; RESTORE RETURN ADDR ON STACK FOR RTI
2865 013372 005005   PARAM1: CLR      R5
2866 013374 012704 016022   MOV      #INBUF,R4
2867 013400 122714 000015   CMPB    #15,(R4) ; CR ?
2868 013404 001420   BEQ     PARERR ; YOU TYPED CR TOO SOON !
2869 013406 121427 000060   15:    CMPB    (R4),#60 ; LOW LIMIT ASCII 0
2870 013412 002415   BLT     PARERR
2871 013414 121427 000067   CMPB    (R4),#67 ; HIGH LIMIT ASCII 7
2872 013420 003012   BGT     PARERR
2873 013422 142714 000060   BICB    #60,(R4) ; CONVERT TO OCTAL
2874 013426 152405   BISB    (R4)+,R5 ; STORE AWAY ITS AN OK CHAR
2875 013430 122714 000015   CMPB    #15,(R4) ; CR ?
2876 013434 001414   BEQ     LIMITS ; NOW CHECK FOR HIGH & LOW LIMIT CONDS
2877 013436 006305   ASL     R5 ; ALLOCATE ROOM FOR NEXT CHAR
2878 013440 006305   ASL     R5
2879 013442 006305   ASL     R5
2880 013444 000760   BR      15
2881 013446 122714 000015   PARERR: CMPB    #15,(R4) ; CR?
2882 013452 001003   BNE     1205
2883 013454 005737 012706   TST     @#RDSW ; CK SWR USED
2884 013460 001023   BNE     PARTI
2885 013462 104407   1205:  INSTER ; RETRY
2886 013464 000742   BR      PARAM1
2887
2888           ; TEST TO SEE IF NUMBER IS WITHIN LIMITS
2889
2890 013466 020567 000042   LIMITS: CMP     R5,HILIM
2891 013472 101365   BHI     PARERR ; THE # IS TOO HIGH
2892 013474 020567 000032   CMP     R5,LOLIM
2893 013500 103762   BLO     PARERR ; THE # IS TOO LOW
2894 013502 136705 000032   BITB    LOBITS,R5 ; TEST BY MASKING THE #
2895 013506 001357   BNE     PARERR
2896
2897           ; STORE NUMBER AT SPECIFIED ADDRESS
2898
2899 013510 016704 000022   15:    MOV     DEVADR,R4 ; GET STARTING ADDR OF
2900 013514 010524   MOV     R5,(R4)+ ; STORE AT THIS ADDR
2901 013516 062705 000002   ADD     #2,R5
2902 013522 105367 000013   DECB   ADRCNT ; HOW MANY TIMES + 2 ?
2903 013526 001372   BNE     15
2904 013530 000002   PARTI: RTI
2905 013532 000000   LOLIM: 0
2906 013534 000000   HILIM: 0
2907 013536 000000   DEVADR: 0
2908 013540 000000   LOBITS: 0
2909           ADRCNT=LOBITS+1
2910

```

```

2911                                     ;SAVE PC OF TEST THAT FAILED AND RO-R5
2912
2913 013542 016667 000004 165356 .SAV05: MOV     4(SP),SAVPC
2914
2915                                     ;SAVE RO-R5
2916
2917 013550 010567 165720          SV05:  MOV     R5,$REG5
2918 013554 010467 165712          MOV     R4,$REG4
2919 013560 010367 165704          MOV     R3,$REG3
2920 013564 010267 165675          MOV     R2,$REG2
2921 013570 010167 165670          MOV     R1,$REG1
2922 013574 010067 165662          MOV     R0,$REG0
2923 013600 000002          RTI
2924
2925                                     ;RESTORE RO-R5
2926
2927 013602 016700 165654          .RES05: MOV     $REG0,R0
2928 013606 016701 165652          MOV     $REG1,R1
2929 013612 016702 165650          MOV     $REG2,R2
2930 013616 016703 165646          MOV     $REG3,R3
2931 013622 016704 165644          MOV     $REG4,R4
2932 013626 016705 165642          MOV     $REG5,R5
2933 013632 000002          RTI
2934
2935                                     ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
2936
2937 013634 104401          .CONVR: TYPE
2938 013636 015313          MCRLF   ;CR LF
2939 013640 017601 000000          MOV     @ (SP),R1          ;PICK UP DATA POINTER
2940 013644 062716 000002          ADD     #2,(SP) ;SET UP SP FOR RTI
2941 013650 012167 000130          MOV     (R1)+,WRDCNT      ;PICK UP # OF WORDS FROM TABLE
2942 013654 112167 000126          1$:    MOVB   (R1)+,CHRCNT   ;PICK UP # OF CHARS FROM TABLE
2943 013660 112167 000123          MOVB   (R1)+,SPACNT      ;PICK UP # OF SPACES FROM TABLE
2944 013664 013167 000120          MOV     @ (R1)+,BINWRD   ;PICK UP ADDRESS OF MSG
2945                                     ;FROM TABLE
2946 013670 016704 000114          2$:    MOV     BINWRD,R4      ;SAVE
2947 013674 116705 000106          MOVB   CHRCNT,R5         ;SAVE
2948 013700 012700 016064          MOV     #TEMP,R0         ;STARTING ADDRESS OF TEMP BLOCK
2949 013704 010403          3$:    MOV     R4,R3          ;SAVE
2950 013706 042703 177770          BIC     #177770,R3       ;CLR OUT UPPER BITS .. SAVE CHAR
2951 013712 062703 000260          ADD     #260,R3 ;CONVERT TO ASCII
2952 013716 110320          MOVB   R3,(R0)+         ;STORE AWAY
2953 013720 006204          ASR    R4              ;SHIFT FOR NEXT #
2954 013722 006204          ASR    R4              ;DITTO
2955 013724 006204          ASR    R4              ;DITTO
2956 013726 005305          DEC    R5              ;DEC CHAR COUNT
2957 013730 001365          BNE    3$             ;DO IT AGAIN ?
2958 013732 012703 016126          MOV     #MDATA,R3       ;STARTING ADDRESS OF MDATA BLOCK
2959 013736 114023          4$:    MOVB   -(R0),(R3)+     ;REVERSE THE ORDER OF NUMBERS
2960 013740 105367 000042          DECB   CHRCNT ;DEC CHAR COUNT
2961 013744 001374          BNE    4$             ;DO IT AGAIN ?
2962 013746 105767 000035          TSTB   SPACNT ;HOW MANY SPACES ?
2963 013752 001405          BEQ    6$             ;TYPE # IF BR =0
2964 013754 112723 000240          5$:    MOVB   #240,(R3)+     ;"SPACE" IN ASCII
2965 013760 105367 000023          DECB   SPACNT ;DEC # OF SPACE COUNT
2966 013764 001373          BNE    5$             ;DO IT AGAIN ?

```

```

2967 013766 105013      65:   CLRB   (R3)   ; INSERT "0" FOR TTY OUTPUT ROUTINE
2968 013770 104401      TYPE
2969 013772 016126      MDATA ; THIS MESSAGE
2970 013774 005367 000004 DEC   WRDCNT ; HOW MANY #'S ?
2971 014000 001325      BNE   1$      ; DO THIS ROUTINE AGAIN IF NOT EQUAL TO 0
2972 014002 000002      RTI   ; RETURN TO PROGRAM
2973 014004 000000      WRDCNT: 0
2974 014006 000000      CHRCNT: 0
2975          014007      SPACNT=CHRCNT+1
2976 014010 000000      BINWRD: 0
2977
2978          ; COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
2979          ; BUFFER TO THE CHARACTERS "N" AND "Y".
2980          ; IF THE CHARACTER IS "N" CLEAR THE FLAG
2981          ; IF THE CHARACTER IS "Y" SET THE FLAG
2982
2983 014012 017605 000000  ..SETFLG: MOV   @ (SP), R5
2984 014016 122767 000116 001776  CMPB  #'N, INBUF ; IS IT "N" ?
2985 014024 001002      BNE   1$
2986 014026 105015      CLRB  (R5)      ; 000
2987 014030 000406      BR    2$
2988 014032 122767 000131 001762 15:  CMPB  #'Y, INBUF ; IS IT "Y" ?
2989 014040 001005      BNE   3$
2990 014042 112715 177777      MOVB  #-1, (R5) ; 377
2991 014046 062716 000002 25:  ADD   #2, (SP)
2992 014052 000002      RTI
2993 014054 104407      35:  INSTER ; RETRY
2994 014056 000755      BR    SETFLG
2995          .SBTTL  ERROR HANDLER ROUTINE
2996
2997          ; *****
2998          ; *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
2999          ; *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3000          ; *AND GO TO SAVIT ON ERROR
3001          ; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3002          ; *SW15=1      HALT ON ERROR
3003          ; *SW13=1      INHIBIT ERROR TYPEOUTS
3004          ; *SW10=1      BELL ON ERROR
3005          ; *SW09=1      LOOP ON ERROR
3006          ; *CALL
3007          ; *      ERROR  N      ;; ERROR=EMT AND N=ERROR ITEM NUMBER
3008
3009 014060      $ERROR:
3010 014060 105267 165317 75:  INCB  $ERFLG      ;; SET THE ERROR FLAG
3011 014064 001775      BEQ   7$          ;; DON'T LET THE FLAG GO TO ZERO
3012 014066 016777 165310 165346  MOV   $TSTNM, @DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
3013 014074 032777 002000 165336  BIT   #BIT10, @SWR    ;; BELL ON ERROR?
3014 014102 001402      BEQ   1$          ;; NO - SKIP
3015 014104 104401 001516      TYPE  , $BELL      ;; RING BELL
3016 014110 005267 165276 15:  INC   $ERTTL      ;; COUNT THE NUMBER OF ERRORS
3017 014114 011667 165276      MOV   (SP), $ERRPC  ;; GET ADDRESS OF ERROR INSTRUCTION
3018 014120 162767 000002 165270  SUB   #2, $ERRPC
3019 014126 117767 165264 165260  MOVB  @ $ERRPC, $ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
3020 014134 032777 020000 165276  BIT   #BIT13, @SWR    ;; SKIP TYPEOUT IF SET
3021 014142 001004      BNE   20$        ;; SKIP TYPEOUTS
3022 014144 004767 000072      JSR   PC, SAVIT    ;; GO TO USER ERROR ROUTINE

```

```

3023 014150 104401 001523          TYPE      , $CRLF
3024 014154
3025 014154 122767 000001 165364 20$:  CMPB    #APTENV, $ENV    ;; RUNNING IN APT MODE
3026 014162 001007          BNE      2$              ;; NO, SKIP APT ERROR REPORT
3027 014164 116767 165224 000004  MOVB    $ITEMB, 21$     ;; SET ITEM NUMBER AS ERROR NUMBER
3028 014172 004767 000016'        JSR     PC, $ATY4       ;; REPORT FATAL ERROR TO APT
3029 014176      000          .BYTE   0
3030 014177      000          .BYTE   0
3031 014200 000777          BR      22$             ;; APT ERROR LOOP
3032 014202 005777 165232 25$:  TST    @SWR            ;; HALT ON ERROR
3033 014206 100001          BPL     3$              ;; SKIP IF CONTINUE
3034 014210 000000          HALT
3035 014212 032777 001000 165220 3$:  BIT    #BIT09, @SWR     ;; LOOP ON ERROR SWITCH SET?
3036 014220 001402          BEQ    4$              ;; BR IF NO
3037 014222 016716 165162        MOV    $LPERR, (SP)     ;; FUDGE RETURN FOR LOOPING
3038 014226 005767 165262 4$:  TST    $ESCAPE         ;; CHECK FOR AN ESCAPE ADDRESS
3039 014232 001402          BEQ    5$              ;; BR IF NONE
3040 014234 016716 165254        MOV    $ESCAPE, (SP)   ;; FUDGE RETURN ADDRESS FOR ESCAPE
3041 014240 5$:
3042 014240 000002          RTI                ;; RETURN
3043 014242 010067 164662  SAVIT:  MOV    R0, HLD0
3044 014246 010167 164660        MOV    R1, HLD1
3045 014252 010267 164656        MOV    R2, HLD2
3046 014256 010367 164654        MOV    R3, HLD3
3047 014262 010467 164652        MOV    R4, HLD4
3048 014266 010567 164650        MOV    R5, HLD5
3049 014272 016767 165104 164644  MOV    $TSTNM, HLD6

```

SBTTL ERROR MESSAGE TYPEOUT ROUTINE

```

3051
3052
3053      ;; *****
3054      ;; *THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
3055      ;; *ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
3056      ;; *AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
3057
3058      $ERRTYP:
3059 014300 104401 001523          TYPE      , $CRLF    ;; "CARRIAGE RETURN" & "LINE FEED"
3060 014304 010046          MOV    R0, -(SP)     ;; SAVE R0
3061 014306 005000          CLR    R0            ;; PICKUP THE ITEM INDEX
3062 014310 153700 001414        BISB   @#$ITEMB, R0
3063 014314 001004          BNE    1$            ;; IF ITEM NUMBER IS ZERO, JUST
3064          .WORD   0          ;; TYPE THE PC OF THE ERROR
3065 014316 016746 165074        MOV    $ERRPC, -(SP)  ;; SAVE $ERRPC FOR TYPEOUT
3066          .WORD   0          ;; ERROR ADDRESS
3067 014322 104402          TYPOC          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3068 014324 000426          BR      6$            ;; GET OUT
3069 014326 005300 15$:  DEC    R0              ;; ADJUST THE INDEX SO THAT IT WILL
3070 014330 006300          ASL    R0            ;; WORK FOR THE ERROR TABLE
3071 014332 006300          ASL    R0
3072 014334 006300          ASL    R0
3073 014336 062700 001652        ADD    #$ERRTB, R0    ;; FORM TABLE POINTER
3074 014342 012067 000004        MOV    (R0)+, 2$     ;; PICKUP "ERROR MESSAGE" POINTER
3075 014346 001404          BEQ    3$            ;; SKIP TYPEOUT IF NO POINTER
3076 014350 104401          TYPE
3077 014352 000000 2$:  .WORD   0            ;; "ERROR MESSAGE" POINTER GOES HERE
3078 014354 104401 001523          TYPE      , $CRLF    ;; "CARRIAGE RETURN" & "LINE FEED"

```

```

3079 014360 012067 000004 3$: MOV (R0)+,4$ ;;PICKUP "DATA HEADER" POINTER
3080 014364 001404 BEQ 5$ ;;SKIP TYPEOUT IF 0
3081 014366 104401 TYPE ;;TYPE THE "DATA HEADER"
3082 014370 000000 4$: .WORD 0 ;;"DATA HEADER" POINTER GOES HERE
3083 014372 104401 001523 TYPE , $CRLF ;;"CARRIAGE RETURN" & "LINE FEED"
3084 014376 011000 5$: MOV (R0),R0 ;;PICKUP "DATA TABLE" POINTER
3085 014400 001004 BNE 7$ ;;GO TYPE THE DATA
3086 014402 012600 6$: MOV (SP)+,R0 ;;RESTORE R0
3087 014404 104401 001523 TYPE , $CRLF ;;"CARRIAGE RETURN" & "LINE FEED"
3088 014410 000207 RTS PC ;;RETURN
3089 014412 7$:
3090 014412 013046 MOV @ (R0)+, -(SP) ;;SAVE @ (R0)+ FOR TYPEOUT
3091 014414 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3092 014416 005710 TST (R0) ;;IS THERE ANOTHER NUMBER?
3093 014420 001770 BEQ 6$ ;;BR IF NO
3094 014422 104401 014430 TYPE , 8$ ;;TYPE TWO(2) SPACES
3095 014426 000771 BR 7$ ;;LOOP
3096 014430 020040 000 8$: .ASCIZ / / ;;TWO(2) SPACES
3097 014434 .EVEN
3098 .SBTTL BINARY TO OCTAL (ASCII) AND TYPE
3099
3100 ;;*****
3101 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3102 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
3103 ;*STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3104 ;*CALL:
3105 ;* MOV NUM, -(SP) ;;NUMBER TO BE TYPED
3106 ;* TYPOS ;;CALL FOR TYPEOUT
3107 ;* .BYTE N ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3108 ;* .BYTE M ;;M=1 OR 0
3109 ;* ;;1=TYPE LEADING ZEROS
3110 ;* ;;0=SUPPRESS LEADING ZEROS
3111 ;*
3112 ;*STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3113 ;*STYPOS OR STYPOC
3114 ;*CALL:
3115 ;* MOV NUM, -(SP) ;;NUMBER TO BE TYPED
3116 ;* TYPON ;;CALL FOR TYPEOUT
3117 ;*
3118 ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3119 ;*CALL:
3120 ;* MOV NUM, -(SP) ;;NUMBER TO BE TYPED
3121 ;* TYPOC ;;CALL FOR TYPEOUT
3122
3123 014434 017646 000000 000211 STYPOS: MOV @ (SP), -(SP) ;;PICKUP THE MODE
3124 014440 116667 000001 MOVB 1(SP), $OFILL ;;LOAD ZERO FILL SWITCH
3125 014446 112667 000207 MOVB (SP)+, $OMODE+1 ;;NUMBER OF DIGITS TO TYPE
3126 014452 062716 000002 ADD #2, (SP) ;;ADJUST RETURN ADDRESS
3127 014456 000406 BR STYPON
3128 014460 112767 000001 000171 STYPOC: MOVB #1, $OFILL ;;SET THE ZERO FILL SWITCH
3129 014466 112767 000006 000165 MOVB #6, $OMODE+1 ;;SET FOR SIX(6) DIGITS
3130 014474 112767 000005 000154 STYPON: MOVB #5, $OCNT ;;SET THE ITERATION COUNT
3131 014502 010346 MOV R3, -(SP) ;;SAVE R3
3132 014504 010446 MOV R4, -(SP) ;;SAVE R4
3133 014506 010546 MOV R5, -(SP) ;;SAVE R5
3134 014510 116704 000145 MOVB $OMODE+1, R4 ;;GET THE NUMBER OF DIGITS TO TYPE

```

3135	014514	005404		NEG	R4	
3136	014516	062704	000006	ADD	#6,R4	::SUBTRACT IT FOR MAX. ALLOWED
3137	014522	110467	000132	MOVB	R4,\$OMODE	::SAVE IT FOR USE
3138	014526	116704	000125	MOVB	\$OFILL,R4	::GET THE ZERO FILL SWITCH
3139	014532	016605	000012	MOV	12(SP),R5	::PICKUP THE INPUT NUMBER
3140	014536	005003		CLR	R3	::CLEAR THE OUTPUT WORD
3141	014540	006105		15: ROL	R5	::ROTATE MSB INTO "C"
3142	014542	000404		BR	3\$::GO DO MSB
3143	014544	006105		25: ROL	R5	::FORM THIS DIGIT
3144	014546	006105		ROL	R5	
3145	014550	006105		ROL	R5	
3146	014552	010503		MOV	R5,R3	
3147	014554	006103		35: ROL	R3	::GET LSB OF THIS DIGIT
3148	014556	105367	000076	DECB	\$OMODE	::TYPE THIS DIGIT?
3149	014562	100016		BPL	7\$::BR IF NO
3150	014564	042703	177770	BIC	#177770,R3	::GET RID OF JUNK
3151	014570	001002		BNE	4\$::TEST FOR 0
3152	014572	005704		TST	R4	::SUPPRESS THIS 0?
3153	014574	001403		BEQ	5\$::BR IF YES
3154	014576	005204		45: INC	R4	::DON'T SUPPRESS ANYMORE 0'S
3155	014600	052703	000060	BIS	#'0,R3	::MAKE THIS DIGIT ASCII
3156	014604	052703	000040	55: BIS	#',R3	::MAKE ASCII IF NOT ALREADY
3157	014610	110367	000040	MOVB	R3,8\$::SAVE FOR TYPING
3158	014614	104401	014654	TYPE	,8\$::GO TYPE THIS DIGIT
3159	014620	105367	000032	75: DECB	\$OCNT	::COUNT BY 1
3160	014624	003347		BGT	2\$::BR IF MORE TO DO
3161	014626	002402		BLT	6\$::BR IF DONE
3162	014630	005204		INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
3163	014632	000744		BR	2\$::GO DO THE LAST DIGIT
3164	014634	012605		65: MOV	(SP)+,R5	::RESTORE R5
3165	014636	012604		MOV	(SP)+,R4	::RESTORE R4
3166	014640	012603		MOV	(SP)+,R3	::RESTORE R3
3167	014642	016666	000002 000004	MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
3168	014650	012616		MOV	(SP)+,(SP)	
3169	014652	000002		RTI		::RETURN
3170	014654	000		85: .BYTE	0	::STORAGE FOR ASCII DIGIT
3171	014655	000		.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
3172	014656	000		\$OCNT:	.BYTE 0	::OCTAL DIGIT COUNTER
3173	014657	000		\$OFILL:	.BYTE 0	::ZERO FILL SWITCH
3174	014660	000000		\$OMODE:	.WORD 0	::NUMBER OF DIGITS TO TYPE
3175						::ENTER HERE ON POWER FAILURE
3176						
3177						
3178	014662			SPWRDN:		
3179	014662	010046		.PFAIL:	MOV R0,-(SP)	::SAVE R0-R5 ON PROCESSOR STACK
3180	014664	010146		MOV	R1,-(SP)	
3181	014666	010246		MOV	R2,-(SP)	
3182	014670	010346		MOV	R3,-(SP)	
3183	014672	010446		MOV	R4,-(SP)	
3184	014674	010546		MOV	R5,-(SP)	
3185	014676	016746	163122	MOV	24,-(SP)	
3186	014702	010667	164210	MOV	SP,SAVSP	::SAVE STACK POINTER
3187	014706	012767	014720 163110	MOV	#RESTART,24	::SET UP FOR POWER UP TRAP
3188	014714	000000		HALT		::HALT ON POWER DOWN NORMAL
3189	014716	000777		BR		
3190						

```

3191                                     ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
3192
3193 014720 016706 164172          RESTAR: MOV      SAVSP, SP          ;RESTORE STACK POINTER
3194 014724 012605                MOV      (SP)+, R5          ;RESTORE R0-R5
3195 014726 012604                MOV      (SP)+, R4
3196 014730 012603                MOV      (SP)+, R3
3197 014732 012602                MOV      (SP)+, R2
3198 014734 012601                MOV      (SP)+, R1
3199 014736 012600                MOV      (SP)+, R0
3200 014740 012767 014662 163056  MOV      #. PFAIL, 24          ;SET UP FOR POWER FAILURE
3201 014746 106427 000340          MTPS     #340
3202 014752 012706 001100          MOV      #STACK, SP
3203 014756 005067 001102          CLR      TEMP
3204 014762 005267 001076          INC      TEMP
3205 014766 001375                BNE      -4
3206 014770 104413                CONVRT
3207 014772 015014                PFTAB
3208 014774 104401                TYPE
3209 014776 015316                MPFAIL
3210 015000 005067 164377          CLR      $ERFLG
3211 015004 005067 164406          CLR      $ERRPC
3212 015010 000177 164070          JMP      @RETURN
3213 015014 000001                PFTAB: 1
3214 015016 006 002                .BYTE 6, 2
3215 015020 000207                RETURN
3216 015022 005015 042012 053125  MTITLE: .ASCIZ <15><12><12>/DUV11 DZDUR-B TAPE B /<15><12>
3217 015030 030461 042040 042132
3218 015036 051125 041055 052040
3219 015044 050101 020105 020102
3220 015052 005015 000
3221 015055 015 053012 041505  MVECTO: .ASCIZ <15><12>/VEC ADD- /
3222 015062 040440 042104 000055
3223 015070 005015 051461 020124  MREGAD: .ASCIZ <15><12>/1ST DEV: REC CSR ADD- /
3224 015076 042504 035126 051040
3225 015104 041505 041440 051123
3226 015112 040440 042104 000055
3227 015120 005015 052515 052114  MMULT: .ASCIZ <15><12>/MULT DEV ? (Y OR N)- /
3228 015126 042040 053105 037440
3229 015134 024040 020131 051117
3230 015142 047040 026451 000
3231 015147 015 046012 051501  MLASTD: .ASCIZ <15><12>/LAST DEV: REC CSR ADDR- /
3232 015154 020124 042504 035126
3233 015162 051040 041505 041440
3234 015170 051123 040440 042104
3235 015176 026522 000
3236 015201 075 042504 044526  DEVICE: .ASCIZ /=DEVICE /
3237 015206 042503 020040 000
3238 015213 015 051412 046105  MCOW: .ASCIZ <15><12>/SELECT TO RUN @ACTREG /
3239 015220 041505 020124 047524
3240 015226 051040 047125 040040
3241 015234 041501 051124 043505
3242 015242 000
3243 015243 015 047412 043126  MRANGE: .ASCIZ <15><12>/OVFLO: RETYPE LAST DEV RXCSR ADDS- /
3244 015250 047514 051072 052105
3245 015256 050131 020105 040514
3246 015264 052123 042040 053105

```

3247	015272	051040	041530	051123
3248	015300	040440	042104	026523
3249	015306	000		
3250	015307	040	037440	000
3251	015313	015	000012	
3252	015316	043120	044501	026114
3253	015324	020040	042522	052123
3254	015332	051101	020124	052101
3255	015340	052040	051505	020124
3256	015346	047111	050040	047522
3257	015354	051107	051505	000123
3258	015362	005015	047105	020104
3259	015370	043117	050040	051501
3260	015376	020123	040524	042520
3261	015404	041040	000	
3262	015407	015	051012	000
3263	015413	015	052012	051505
3264	015420	020124	041520	000055
3265	015426	005015	047514	045503
3266	015434	047440	020116	052040
3267	015442	051505	037524	024040
3268	015450	020131	051117	047040
3269	015456	026451	000	
3270	015461	015	021412	047440
3271	015466	020106	054523	041516
3272	015474	041440	040510	051522
3273	015502	051440	046105	041505
3274	015510	042524	020104	020050
3275	015516	020061	051117	031040
3276	015524	026451	000	
3277	015527	015	044412	020123
3278	015534	042523	020103	046530
3279	015542	052111	051440	044527
3280	015550	041524	020110	032505
3281	015556	026465	020062	047111
3282	015564	020077	054450	047440
3283	015572	020122	024516	000055
3284	015600	005015	051511	051440
3285	015606	041505	051040	041505
3286	015614	051440	044527	041524
3287	015622	020110	032505	026465
3288	015630	020063	047111	020077
3289	015636	054450	047440	020122
3290	015644	024516	000055	
3291	015650	005015	051511	047440
3292	015656	052120	041440	051114
3293	015664	042440	040516	046102
3294	015672	020105	053523	052111
3295	015700	044103	042440	032465
3296	015706	030455	044440	037516
3297	015714	024040	020131	051117
3298	015722	047040	026451	000
3299	015727	015	005012	031510
3300	015734	032461	041440	047117
3301	015742	042516	052103	051117
3302	015750	047440	020116	024077

MQM: . ASCII / ?/
 MCRLF: . ASCII <15><12>
 MPFAIL: . ASCII /PFAIL, RESTART AT TEST IN PROGRESS/

MEPASS: . ASCII <15><12>/END OF PASS TAPE B/

MR: . ASCII <15><12>/R/
 MTSTPC: . ASCII <15><12>/TEST PC-/

MLOCK: . ASCII <15><12>/LOCK ON TEST? (Y OR N)-/

MSYNC: . ASCII <15><12>/# OF SYNC CHARS SELECTED (1 OR 2)-/

MWIRE6: . ASCII <15><12>/IS SEC XMIT SWITCH E55-2 IN? (Y OR N)-/

MWIRE5: . ASCII <15><12>/IS SEC REC SWITCH E55-3 IN? (Y OR N)-/

MWIRE4: . ASCII <15><12>/IS OPT CLR ENABLE SWITCH E55-1 IN? (Y OR N)-/

MEXTJ: . ASCII <15><12><12>/H315 CONNECTOR ON ?(Y OR N)-/

```

3303 015756 020131 051117 047040
3304 015764 026451 000
3305 015767 015 020012 043536 MCNTG: .ASCIZ <15><12>/ G /
3306 015774 020040 000
3307 015777 040 053523 036522 MMSWR: .ASCIZ / SWR= /
3308 016004 020040 000040
3309 016010 020040 047040 053505 MMNEW: .ASCIZ / NEW= /
3310 016016 020075 000040
3311 .EVEN
3312
3313 ;BUFFERS FOR INPUT-OUTPUT
3314
3315 016022 000000 INBUF: 0
3316 016064 000000 . = +40
3317 016064 000000 TEMP: 0
3318 016126 000000 . = +40
3319 016126 000000 MDATA: 0
3320 016170 000000 . = +40
3321 .SBTTL SCOPE HANDLER ROUTINE
3322
3323 ;*****
3324 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3325 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG. (DISPLAY<7: 0>)
3326 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15: 08>
3327 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3328 ;*SW14=1 LOOP ON TEST
3329 ;*SW11=1 INHIBIT ITERATIONS
3330 ;*SW09=1 LOOP ON ERROR
3331 ;*SW08=1 LOOP ON TEST IN SWR<7: 0>
3332 ;*CALL
3333 ;* SCOPE ;:SCOPE=10T
3334
3335 016170 $SCOPE:
3336
3337 ;SCOPE LOOP AND INTERATION HANDLER
3338
3339 016170 .SCOPE:
3340 016170 004767 174376 JSR PC,CKSWR
3341 016174 005067 163216 CLR $ERRPC ;CLEAR LAST ERROR PC
3342 016200 022716 003376 CMP #TST1+2, (SP) ;IS SCOPE AT BEGINING OF TEST 1?
3343 016204 001422 BEQ $XTSTR ;YES NO LOOP.
3344
3345 016206 032777 040000 163224 TTST: BIT #BIT14, @SWR ;THIS CODE IS FOR TESTING FOR BIT 14
3346 016214 001412 BEQ 1$ ;ON LSI WHICH SYSMAC CANNOT HANDLE
3347 016216 016767 163160 163162 MOV $TSTNM, $LPADR
3348 016224 000406 BR 1$
3349 016226 105777 163212 TSTB @5TKS ;KEYBOARD DONE?
3350 016232 100123 BPL $OVER ;BR IF NO
3351 016234 017766 163206 177776 MOV @5TKB, -2(SP) ;CLEAR DONE BIT
3352 016242 032777 040000 163170 1$: BIT #BIT14, @SWR ;LOOP ON PRESENT TEST?
3353 016250 001114 BNE $OVER ;YES IF SW14=1
3354 ;####START OF CODE FOR THE XOR TESTER####
3355 016252 000416 $XTSTR: BR 6$ ;IF RUNNING ON THE "XOR" TESTER CHANGE
3356 ;THIS INSTRUCTION TO A "NOP" (NOP=240)
3357 016254 013746 000004 MOV @#ERRVEC, -(SP) ;SAVE THE CONTENTS OF THE ERROR VECTOR
3358 016260 012737 016300 000004 MOV #5$, @#ERRVEC ;SET FOR TIMEOUT
  
```

```

3359 016266 005737 177060          TST      @#177060          ;; TIME OUT ON XOR?
3360 016272 012637 000004          MOV      (SP)+, @#ERRVEC ;; RESTORE THE ERROR VECTOR
3361 016276 000463                   BR       $$VLAD           ;; GO TO THE NEXT TEST
3362 016300 022626          5$:     CMP      (SP)+, (SP)+ ;; CLEAR THE STACK AFTER A TIME OUT
3363 016302 012637 000004          MOV      (SP)+, @#ERRVEC ;; RESTORE THE ERROR VECTOR
3364 016306 000423                   BR       7$              ;; LOOP ON THE PRESENT TEST
3365 016310                   6$:     ; #####END OF CODE FOR THE XOR TESTER#####
3366 016310 032777 000400 163122          BIT      #BIT08, @SWR     ;; LOOP ON SPEC. TEST?
3367 016316 001404                   BEQ      2$              ;; BR IF NO
3368 016320 127767 163114 163054          CMPB    @SWR, $STNM      ;; ON THE RIGHT TEST?   SWR<7: 0>
3369 016326 001465                   BEQ      $OVER          ;; BR IF YES
3370 016330 105767 163047          2$:     TSTB    $ERFLG      ;; HAS AN ERROR OCCURRED?
3371 016334 001421                   BEQ      3$              ;; BR IF NO
3372 016336 126767 163053 163037          CMPB    $ERMAX, $ERFLG  ;; MAX. ERRORS FOR THIS TEST OCCURRED?
3373 016344 101015                   BHI      3$              ;; BR IF NO
3374 016346 032777 001000 163064          BIT      #BIT09, @SWR     ;; LOOP ON ERROR?
3375 016354 001404                   BEQ      4$              ;; BR IF NO
3376 016356 016767 163026 163022          7$:     MOV      $LPERR, $LPADR ;; SET LOOP ADDRESS TO LAST SCOPE
3377 016364 000446                   BR       $OVER          ;;
3378 016366 105067 163011          4$:     CLRB    $ERFLG      ;; ZERO THE ERROR FLAG
3379 016372 005067 163114          CLR     $TIMES          ;; CLEAR THE NUMBER OF ITERATIONS TO MAKE
3380 016376 000415                   BR       1$              ;; ESCAPE TO THE NEXT TEST
3381 016400 032777 004000 163032          3$:     BIT      #BIT11, @SWR ;; INHIBIT ITERATIONS?
3382 016406 001011                   BNE      1$              ;; BR IF YES
3383 016410 005767 163120          TST     $PASS          ;; IF FIRST PASS OF PROGRAM
3384 016414 001406                   BEQ      1$              ;; INHIBIT ITERATIONS
3385 016416 005267 162762          INC     $ICNT          ;; INCREMENT ITERATION COUNT
3386 016422 026767 163064 162754          CMP     $TIMES, $ICNT   ;; CHECK THE NUMBER OF ITERATIONS MADE
3387 016430 002024                   BGE     $OVER          ;; BR IF MORE ITERATION REQUIRED
3388 016432 012767 000001 162744          1$:     MOV     #1, $ICNT  ;; REINITIALIZE THE ITERATION COUNTER
3389 016440 016767 000056 163044          MOV     $MXCNT, $TIMES ;; SET NUMBER OF ITERATIONS TO DO
3390 016446 105267 162730          $$VLAD: INCB    $STNM     ;; COUNT TEST NUMBERS
3391 016452 116767 162724 163052          MOVB   $STNM, $TESTN   ;; SET TEST NUMBER IN APT MAILBOX
3392 016460 011667 162722          MOV     (SP), $LPADR   ;; SAVE SCOPE LOOP ADDRESS
3393 016464 011667 162720          MOV     (SP), $LPERR   ;; SAVE ERROR LOOP ADDRESS
3394 016470 005067 163020          CLR     $ESCAPE       ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
3395 016474 112767 000001 162713          MOVB   #1, $ERMAX     ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3396 016502 016777 162674 162732          $OVER: MOV     $STNM, @DISPLAY ;; DISPLAY TEST NUMBER
3397 016510 016716 162672          MOV     $LPADR, (SP)   ;; FUDGE RETURN ADDRESS
3398 016514 000002          4$:     RTI
3399 016516 001407          BRW:    1407
3400 016520 000432          BRX:    432
3401 016522 000005          $MXCNT: 5              ;; MAX. NUMBER OF ITERATIONS
3402          .SBTTL TRAP DECODER
3403
3404          ;; *****
3405          ;; *THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3406          ;; *AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3407          ;; *OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3408          ;; *GO TO THAT ROUTINE.
3409
3410 016524 010046          STRAP: MOV     R0, -(SP) ;; SAVE R0
3411 016526 016600 000002          MOV     2(SP), R0     ;; GET TRAP ADDRESS
3412 016532 005740          TST     -(R0)         ;; BACKUP BY 2
3413 016534 111000          MOVB   (R0), R0      ;; GET RIGHT BYTE OF TRAP
3414 016536 006300          ASL    R0             ;; POSITION FOR INDEXING
    
```

```
3415 016540 016000 016560      MOV    $TRPAD(R0),R0    ;; INDEX TO TABLE
3416 016544 000200              RTS    R0              ;; GO TO ROUTINE
3417
3418
3419                      ;; THIS IS USE TO HANDLE THE "GETPRI" MACRO
3420
3421 016546 011646      STRAP2: MOV    (SP),-(SP)    ;; MOVE THE PC DOWN
3422 016550 016666 000004 000002  MOV    4(SP),2(SP)    ;; MOVE THE PSW DOWN
3423 016556 000002      RTI              ;; RESTORE THE PSW
3424
3425      .SBTTL  TRAP TABLE
3426
3427      ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3428      ;*BY THE "TRAP" INSTRUCTION.
3429
3430      ;          ROUTINE
3431      ;          -----
3432 016560 016546      STRPAD: .WORD  $TRAP2
3433 016562 012742      $TYPE   ;; CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
3434 016564 014460      $TYPOC  ;; CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3435 016566 014434      $TYPOS  ;; CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
3436 016570 014474      $TYPON  ;; CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
3437
3438
3439 016572 012720      .SCOP1  ;; CALL=SCOP1     TRAP+5(104405)
3440 016574 013224      .INSTR  ;; CALL=INSTR     TRAP+6(104406)
3441 016576 013332      .INSTER ;; CALL=INSTER   TRAP+7(104407)
3442 016600 013342      .PARAM  ;; CALL=PARAM    TRAP+10(104410)
3443 016602 013542      .SAVOS  ;; CALL=SAVOS    TRAP+11(104411)
3444 016604 013602      .RESOS  ;; CALL=RESOS    TRAP+12(104412)
3445 016606 013634      .CONVRT ;; CALL=CONVRT   TRAP+13(104413)
3446 016610 014012      .SETFLG ;; CALL=SETFLG   TRAP+14(104414)
3447      ;*****
3448      ;UTILITIES
3449      ;*****
3450
3451      ; THIS UTILITY CALCULATES PRIORITY LEVEL
3452 016612 006367 000044      DULEV: ASL    DUPRT    ; SHIFT LEFT
3453 016616 006367 000040      ASL    DUPRT    ;
3454 016622 006367 000034      ASL    DUPRT    ;
3455 016626 006367 000030      ASL    DUPRT    ;
3456 016632 006367 000024      ASL    DUPRT    ;
3457 016636 016767 000020 000020  MOV    DUPRT,LESS1    ; MOVE THIS TO LESS1
3458 016644 162767 000001 000012  SUB    #1,LESS1      ; CREATE LESS1
3459 016652 042767 000037 000004  BIC    #37,LESS1     ; CLEAR TNZVC
3460 016660 000207      RTS    PC
3461 016662 000240      DUPRT: PR5
3462 016664 000200      LESS1: PR4    ; LEVEL TO ALLOW INTERRUPTS
3463
3464      ; NEW DU ADDRESSES
3465 016666 016767 000126 163016  DUADDR: MOV    DUBASE,RXCSR    ; XXX0
3466 016674 005267 000120      INC    DUBASE
3467 016700 016767 000114 163006  MOV    DUBASE,HRXCSR  ; XXX1
3468 016706 005267 000106      INC    DUBASE
3469 016712 016767 000102 162776  MOV    DUBASE,RXDBUF  ; XXX2
3470 016720 016767 000074 162774  MOV    DUBASE,PARCSR  ; XXX2
```

```

3471 016726 005267 000066      INC      DUBASE
3472 016732 016767 000062 162760    MOV      DUBASE,HRXDBUF ;XXX3
3473 016740 016767 000054 162756    MOV      DUBASE,HPARCSR ;XXX3
3474 016746 005267 000046      INC      DUBASE
3475 016752 016767 000042 162746    MOV      DUBASE,TXCSR   ;XXX4
3476 016760 005267 000034      INC      DUBASE
3477 016764 016767 000030 162736    MOV      DUBASE,HTXCSR  ;XXX5
3478 016772 005267 000022      INC      DUBASE
3479 016776 016767 000016 162726    MOV      DUBASE,TXDBUF  ;XXX6
3480 017004 005267 000010      INC      DUBASE
3481 017010 016767 000004 162716    MOV      DUBASE,HTXDBUF ;XXX7
3482 017016 000207      RTS      PC
3483 017020 000000      DUBASE: 0
3484
3485
3486
3487
3488 017022 042777 040000 162676    RPOKE:  BIC      #MTDATA,@TXCSR
3489 017030 005067 162446      CLR      $TMP2
3490 017034 006067 162440      ROR      $TMP1 ;FORCE CARRY
3491 017040 006067 162436      ROR      $TMP2 ;PICK UP CARRY IN BIT 15
3492 017044 006267 162432      ASR      $TMP2 ;SHIFT INTO BIT 14
3493 017050 042767 100000 162424    BIC      #BIT15,$TMP2 ;CLR BIT 15
3494 017056 056777 162420 162642    BIS      $TMP2,@TXCSR ;POKE MAINT DATA
3495 017064 042777 020000 162634    BIC      #CLK,@TXCSR ;POKE CLK
3496 017072 052777 020000 162626    BIS      #CLK,@TXCSR ;
3497 017100 005367 162016      DEC      SHIFT
3498 017104 001346      BNE      RPOKE
3499 017106 000207      RTS      PC
3500
3501 017110 016767 162364 162364    ODD8:  MOV      $TMP1,$TMP2 ;SAVE TEMP1
3502 017116 005067 162362      CLR      $TMP3
3503 017122 012727 000010      MOV      #8,(PC)+
3504 017126 000000      4$:  0
3505 017130 006067 162346      1$:  ROR      $TMP2
3506 017134 005567 162344      ADC      $TMP3
3507 017140 005367 177762      DEC      4$
3508 017144 001371      BNE      1$
3509 017146 006067 162332      ROR      $TMP3
3510 017152 103404      BCS      2$
3511 017154 052767 000400 162316    BIS      #BIT8,$TMP1 ;SET ODD PARITY
3512 017162 000403      BR       3$
3513 017164 042767 000400 162306    2$:  BIC      #BIT8,$TMP1 ;CLR EVEN PARITY
3514
3515 017172 000207      3$:  ;$TMP1 NOW HAS ODD PARITY CHARACTER
3516
3517
3518 017174 016767 162300 162300    ;THIS ROUTINE CALCULATES EVEN PARITY FOR AN 8 BIT CHARACTER
3519 017202 005067 162276    EVENS: MOV      $TMP1,$TMP2 ;SAVE TEMP1
3520 017206 012727 000010      CLR      $TMP3
3521 017212 000000      4$:  MOV      #8,(PC)+
3522 017214 006067 162262      1$:  0
3523 017220 005567 162260      ROR      $TMP2
3524 017224 005367 177762      ADC      $TMP3
3525 017230 001371      DEC      4$
3526 017232 006067 162246      BNE      1$
3526 017232 006067 162246      ROR      $TMP3
    
```

3527	017236	103004				BCC	2\$	
3528	017240	052767	000400	162232		BIS	#BIT8,\$TMP1	;SET EVEN PARITY
3529	017246	000403				BR	3\$	
3530	017250	042767	000400	162222	2\$:	BIC	#BIT8,\$TMP1	;CLR ODD PARITY
3531								; \$TMP1 NOW HAS EVEN PARITY CHARACTER
3532	017256	000207			3\$:	RTS	PC	
3533								
3534	017260	062716	000002		TRPREG:	ADD	#2,(SP)	;ALLOW IT TO "CRUNCH" INTO HLT BACK
3535								; IN MAIN PART OF THE PROGRAM
3536	017264	000002				RTI		
3537		000001			.END			

AAA	003200	1292#								
ABASE =	000000	875	916							
ACDW1 =	000000	875	918							
ACDW2 =	000000	875	919							
ACPUOP=	000000	875	890							
ACTREG	001166	734#	1248*	1262*	1263*	1270*	2648	2651	2661	
ADDW0 =	000000	875	920							
ADDW1 =	000000	875	921							
ADDW10=	000000	875	930							
ADDW11=	000000	875	931							
ADDW12=	000000	875	932							
ADDW13=	000000	875	933							
ADDW14=	000000	875	934							
ADDW15=	000000	875	935							
ADDW2 =	000000	875	922							
ADDW3 =	000000	875	923							
ADDW4 =	000000	875	924							
ADDW5 =	000000	875	925							
ADDW6 =	000000	875	926							
ADDW7 =	000000	875	927							
ADDW8 =	000000	875	928							
ADDW9 =	000000	875	929							
ADEVCT=	000000	875	881							
ADEVN =	000000	875	917							
ADRCNT=	013541	2863*	2902*	2909#						
AENV =	000000	875	886							
AENVN =	000000	875	887							
AFATAL=	000000	875	878							
AMADR1=	000000	875	903							
AMADR2=	000000	875	907							
AMADR3=	000000	875	910							
AMADR4=	000000	875	913							
AMAMS1=	000000	875	897							
AMAMS2=	000000	875	905							
AMAMS3=	000000	875	908							
AMAMS4=	000000	875	911							
AMSGAD=	000000	875	883							
AMSGLG=	000000	875	884							
AMSGTY=	000000	875	877							
AMTYP1=	000000	875	898							
AMTYP2=	000000	875	906							
AMTYP3=	000000	875	909							
AMTYP4=	000000	875	912							
APASS =	000000	875	880							
APRIOR=	000000	875								
APTCSU=	000040	535#	2778							
APTENV=	000001	535#	2771	3025						
APTSIZ=	000200	535#	1167							
APTSP0=	000100	535#	2773							
ASWREG=	000000	875	888							
AESTN=	000000	875	879							
AUNIT =	000000	875	882							
AUSWR =	000000	875	889							
AVECT1=	000000	875	914							
AVECT2=	000000	875	915							
BASEAD	001154	729#	1230*	1267*	1268	1274*	1276*	2655*	2667*	2671

CROSS REFERENCE TABLE -- USER SYMBOLS

DF1	002132	1026	1030	1034	1038	1084#													
DH1	002067	1024	1028	1032	1073#														
DISPLA	001442	842#	1155*	1163*	1188*	3012*	3396*												
DISPRE	000174	683#	1163	1188															
DNA =	100000	798#	991#																
DNAINT=	000040	805#	998#																
DSC =	100000	762#	955#																
DSINTE=	000040	772#	965#																
DSR =	001000	768#	961#																
DSWR =	177570	585#	841	1154															
DTR =	000002	776#	969#																
DT1	002116	1025	1029	1033	1079#														
DT4	002126	1037	1082#																
DUADDR	016666	1229	2672	3465#															
DUBASE	017020	1225	1228	2671*	3465	3466*	3467	3468*	3469	3470	3471*	3472	3473	3474*					
		3475	3476*	3477	3478*	3479	3480*	3481	3483#										
DULEV	016612	1287	3452#																
DUPRT	016662	1286*	3452*	3453*	3454*	3455*	3456*	3457	3461#										
DURIS	001740	1053#	2675*																
DURIV	001736	1052#	1236	1239	1240	2673*	2680												
DUTIS	001744	1055#	2679*																
DUTIV	001742	1054#	2677*																
EIGHT =	006000	793#	986#	1911	1966	2021	2076												
EMTVEC=	000030	674#	1139*	1140*															
EM1	001762	1023	1059#																
EM2	002022	1027	1065#																
EM3	002043	1031	1068#																
EM4	001746	1035	1057#																
ERRCNT	001114	702#																	
ERRVEC=	000004	667#	1152	1153*	1164*	3357	3358*	3360*	3363*										
EVEN8	017174	3518#																	
EVEPAR=	001400	796#	989#																
EVPAR =	000400	785#	978#																
FIVE =	000000	790#	983#	1361	1416	2131	2183	2235	2287										
FRMERR=	020000	781#	974#																
GNS =	***** U	3433	3434	3435	3436	3439	3440	3441	3442	3443	3444	3445	3446						
HDXEN =	000010	807#	1000#																
HILIM	013534	2860*	2890	2906#															
HLD0	001130	711#	1079	3043*															
HLD1	001132	712#	1079	3044*															
HLD2	001134	713#	3045*																
HLD3	001136	714#	3046*																
HLD4	001140	715#	3047*																
HLD5	001142	716#	3048*																
HLD6	001144	717#	3049*																
HOLD	001120	707#																	
HPARCS	001724	1046#	3473*																
HRXCSR	001714	1042#	3467*																
HRXDBU	001720	1044#	3472*																
HT =	000011	577#	2786	2827															
HTXCSR	001730	1048#	3477*																
HTXDBU	001734	1050#	3481*																
INBUF	016022	1295	1299	2836	2866	2984	2988	3315#											
INIFLG	001172	742#	1198	1201*															
INSTER=	104407	1303	2885	2993	3441#														
INSTR =	104406	1220	1231	1241	1252	1277	1293	1306	1310	1314	1318	1329	2726	3440#					

CROSS REFERENCE TABLE -- USER SYMBOLS

\$N = 000000	533#	2634#												
\$NULL 001454	847#	2798	2827											
\$NWTST= 000000	1351#	1406#	1461#	1516#	1571#	1626#	1681#	1736#	1791#	1846#	1901#	1956#	2011#	
	2066#	2121#	2173#	2225#	2277#	2329#	2381#	2433#	2485#	2537#	2589#			
\$OCNT 014656	3130*	3159*	3172#											
\$OMODE 014660	3125*	3129*	3134	3137*	3148*	3174#								
\$OVER 016502	3350	3353	3369	3377	3387	3396#								
\$PASS 001534	880#	1166*	2693*	3383	3402									
\$PASTM 002144	1116#													
\$PWDRN 014662	1143	3178#												
\$QUES 001522	868#	2827	3043											
\$RDCHR= ***** U	3439													
\$RDDEC= ***** U	3439													
\$RDLIN= ***** U	3439													
\$RDOCT= ***** U	3439													
\$REGAD 001460	851#													
\$REGO 001462	853#	2922*	2927											
\$REG1 001464	854#	2921*	2928											
\$REG2 001466	855#	2920*	2929											
\$REG3 001470	856#	2919*	2930											
\$REG4 001472	857#	2918*	2931											
\$REG5 001474	858#	2917*	2932											
\$R2A = ***** U	3439													
\$SAVRE= ***** U	3439													
\$SCOPE 016170	1137	3335#												
\$SETUP= 000017	1119#	1136	1137	1139	1141	1143	1145	1146	1148	3010	3035	3042	3336	
\$STUP = 177777	1119#													
\$SVLAD 016446	3361	3390#												
\$SVPC = 002136	1091#	1096												
\$SWR = 177400	524#	865	866	867	1145	1146	1148	1149	1353	1408	1463	1518	1573	
	1628	1683	1738	1793	1848	1903	1958	2013	2068	2123	2175	2227	2279	
	2331	2383	2435	2487	2539	2591	3001	3002	3003	3004	3005	3013	3020	
	3032	3035	3043	3327	3328	3329	3330	3331	3352	3364	3366	3367	3370	
	3371	3372	3379	3380	3381	3393	3396	3401						
\$SWREG 001550	888#	1169												
\$SWRMK= 000000	3331	3332	3368											
\$TESTN 001532	879#	3391*												
\$TIMES 001512	865#	1145*	3379*	3386	3389*	3401								
\$TKB 001446	844#	2717	2840	2848	3351									
\$TKS 001444	843#	2715	2838	3349										
\$TMP0 001476	859#													
\$TMP1 001500	860#	1372*	1384*	1388*	1427*	1439*	1443*	1482*	1494*	1498*	1537*	1549*	1553*	
	1592*	1604*	1608*	1647*	1659*	1663*	1702*	1714*	1718*	1757*	1769*	1773*	1812*	
	1824*	1828*	1867*	1879*	1883*	1922*	1934*	1938*	1977*	1989*	1993*	2032*	2044*	
	2048*	2087*	2099*	2103*	2139*	2151*	2155*	2191*	2203*	2207*	2243*	2255*	2259*	
	2295*	2307*	2311*	2347*	2359*	2363*	2399*	2411*	2415*	2451*	2463*	2467*	2503*	
	2515*	2519*	2555*	2567*	2571*	2607*	2619*	2623*	3490*	3501	3511*	3513*	3518	
	3528*	3530*												
\$TMP2 001502	861#	3489*	3491*	3492*	3493*	3494	3501*	3505*	3518*	3522*				
\$TMP3 001504	862#	3502*	3506*	3509*	3519*	3523*	3526*							
\$TMP4 001506	863#													
\$TMP5 001510	864#													
\$TN = 000031	535#	1351	1353#	1406	1408#	1461	1463#	1516	1518#	1571	1573#	1626	1628#	
	1681	1683#	1736	1738#	1791	1793#	1846	1848#	1901	1903#	1956	1958#	2011	
	2013#	2066	2068#	2121	2123#	2173	2175#	2225	2227#	2277	2279#	2329	2331#	
	2381	2383#	2433	2435#	2485	2487#	2537	2539#	2589	2591#				

.SCMTA	524#	814
.SEOP	524#	
.SERRO	524#	2995
.SERRT	524#	3051
.SPOWE	524#	
.SSCOP	524#	3321
.STRAP	524#	3402
.STYPE	524#	2748
.STYPO	524#	3098

.ABS. 017266 000

ERRORS DETECTED: 0

DZDURB, DZDURB/SOL/CRF=DZDUR1/EQ: RUNB, DZDUR2, DZDURB
RUN-TIME: 21 12 1 SECONDS
RUN-TIME RATIO: 234/35=6.6
CORE USED: 30K (59 PAGES)

AK

G 7