

# DUV-11

DUV11 OFF LINE LOGIC TESTS  
MD-11-DZDUQ-B

EP-DZDUQ-B-DL-B  
COPYRIGHT © 1977  
FICHE 1 OF 1

DEC 1977  
**digital**  
MADE IN USA

This microfiche card contains a grid of frames, each representing a logic test. The frames are arranged in approximately 10 rows and 10 columns. Each frame contains a table of data, likely representing test results for various logic components or configurations. The data is organized into columns, with some frames showing headers and multiple rows of test results. The text is small and difficult to read due to the resolution of the scan, but the overall structure is that of a data table.



REM \*

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

PRODUCT CODE MAINDEC-11-DZDUQ-B-D

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

\* REM \*

2 REQUIREMENTS

PDP-11/03 COMPUTER (LSI)

DUV11 SYNCHRONOUS/ISOCRONOUS OPTION

ONE CONSOLE TELETYPE OR EQUIVALENT

2 2 STORAGE  
THE PROGRAM LOADS INTO 4K OF MEMORY WITH BOOTSTRAP

\* REM \*

3 LOADING PROCEDURE

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

STARTING ADDRESS  
FOR ABSOLUTE LOADER

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

4 STARTING PROCEDURE

4 1 CONTROL SWITCH SETTINGS

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

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

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

- 4 1 1 AFTER PROGRAM LOAD (INITIAL PROGRAM START)  
ALL CONSOLE SWITCHES DOWN
- 4 1 2 TO MODIFY DEVICE VECTOR AND CONTROL REGISTER ADDRESSES  
AFTER PROGRAM RESTART OR TO RUN MULTIPLE DEVICES  
  
SW00=1
- 4 1 3 TO START PROGRAM AT SELECTED TEST AFTER A PROGRAM RESTART  
(ONLY IN SINGLE DEVICE TESTS)  
SW01=1
- 4 1 4 TO LOCK ON SELECTED TEST AFTER A PROGRAM RESTART  
(ONLY IN SINGLE DEVICE TESTS)  
  
SW14=1  
NOTE1 IN GENERAL SW01 WILL BE USED WHEN SW14=1 IS USED  
NOTE2 WITHOUT SW01=1 "LOCK ON TEST" WILL DEFAULT TO TEST 1
- 4 2 STARTING ADDRESS  
  
THE STARTING ADDRESS FOR ALL TESTS IS 000200  
  
THE RESTARTING 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 DZDUQ-B TAPE A" (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 PXCSR 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 J1A 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 S 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 T 7 VECTOR ADDRESS NOT SELECTED  
DURING PROGRAM INITIALIZA 4, 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 PFSECTIVELY  
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, BASE IV , NEXT BLOCK  
(VECTORS)" TO "ZERO ADD #0, BASE IV",  
THEREBY THE VECTOR ADDRESSES WILL NOT BE  
UPDATED AFTER EACH PASS

7 2 DISQUALIFYING DEVICES WHEN RUNNING MULTIPLE DEVICES

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

7 2 1 IF DEVICE 0 IS TO BE DISQUALIFIED , SIMPLY RESTART  
PROGRAM WITH SW00 =1 AND OMIT THE FIRST DEVICE

7 2 2 IF HOWEVER, DEVICES 1 THRU 15 OR ANY COMBINATION THEREOF  
ARE TO BE DISQUALIFIED LOAD THE LOCATION OF ACTREG  
OBSERVE THE ACTIVE BITS (ACTIVE =1, NONACTIVE = 0)  
AND DEPOSIT 0 WHERE THOSE DEVICES ARE TO BE DISQUALIFIED

7 2 2 1 TO RESTART TYPE 200G  
THE PROGRAM WILL CONTINUE WITH THE DEVICE IT WAS IN BEFORE HALTING

7 2 2 2 OR SET SW00=1 IN SWITCH REG (LOC 176) AND TYPE 200G  
ANSWER THE QUESTION 1ST DEVICE ETC  
THE PROGRAM WILL CONTINUE WITH DEVICE 0

7 2 2 3 IF ALL DEVICES ARE DISQUALIFIED BY MISTAKE THE PROGRAM  
WILL TYPEOUT AN ERROR MESSAGE TYPE 20GG

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

\*

REM \*

9 1    THIS PROGRAM PERFORMS THE OFFLINE LOG C BIT BANGING  
      OF THE DEVICE  
      SEE LISTING FOR DETAILS

\*

REM \*

10     FLOW CHARTS RECEIVER FLOW, TRANSMITTER FLOW TRANSMITTER & PECEIVER FLOW

11     LISTINGS

\*

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  
557

000001

STN=1



```
558 . ENABLE ABS
559
560 . DUV11 DZDUQ-B TAPE A
561 . COPYRIGHT 1977, DIGITAL EQUIPMENT CORP , MAYNARD, MASS 01754
562
563 . STARTING PROCEDURE
564 . TYPE 200G
565 . PROGRAM WILL TYPE "DUV11 DZDUQ-B TAPE A "
566 . PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
567 . AT THE END OF A PASS, PROGRAM WILL TYPE "END OF PASS TAPE R"
568 . AND THEN RESUME TESTING
569
570 SBTTL BASIC DEFINITIONS
571
572 . *INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
573 001100 STACK= 1100
574 . EQUIV EMT,ERROR ., BASIC DEFINITION OF ERROR CALL
575 . EQUIV IOT,SCOPE ., BASIC DEFINITION OF SCOPE CALL
576
577 . *MISCELLANEOUS DEFINITIONS
578 000011 HT= 11 ., CODE FOR HORIZONTAL TAB
579 000012 LF= 12 ., CODE FOR LINE FEED
580 000015 CR= 15 ., CODE FOR CARRIAGE RETURN
581 000200 CRLF= 200 ., CODE FOR CARRIAGE RETURN-LINE FEED
582 177776 PS= 177776 ., PROCESSOR STATUS WORD
583 . EQUIV PS,PSW
584 177774 STKLMT= 177774 ., STACK LIMIT REGISTER
585 177772 PIRQ= 177772 ., PROGRAM INTERRUPT REQUEST REGISTER
586 177570 DSWR= 177570 ., HARDWARE SWITCH REGISTER
587 177570 , DDISP= 177570 ., HARDWARE DISPLAY REGISTER
588
589 . *GENERAL PURPOSE REGISTER DEFINITIONS
590 000000 R0= %0 ., GENERAL REGISTER
591 000001 R1= %1 ., GENERAL REGISTER
592 000002 R2= %2 ., GENERAL REGISTER
593 000003 R3= %3 ., GENERAL REGISTER
594 000004 R4= %4 ., GENERAL REGISTER
595 000005 R5= %5 ., GENERAL REGISTER
596 000006 R6= %6 ., GENERAL REGISTER
597 000007 R7= %7 ., GENERAL REGISTER
598 000006 SP= %6 ., STACK POINTER
599 000007 PC= %7 ., PROGRAM COUNTER
600
601 . *PRIORITY LEVEL DEFINITIONS
602 000000 PRO= 0 ., PRIORITY LEVEL 0
603 000040 PR1= 40 ., PRIORITY LEVEL 1
604 000100 PR2= 100 ., PRIORITY LEVEL 2
605 000140 PR3= 140 ., PRIORITY LEVEL 3
606 000200 PR4= 200 ., PRIORITY LEVEL 4
607 000240 PR5= 240 ., PRIORITY LEVEL 5
608 000300 PR6= 300 ., PRIORITY LEVEL 6
609 000340 PR7= 340 ., PRIORITY LEVEL 7
610
611 . *"SWITCH REGISTER" SWITCH DEFINITIONS
612 100000 SW15= 100000
613 040000 SW14= 40000
```

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

, \*DATA BIT DEFINITIONS (BIT00 TO BIT15)

639		BIT15=	100000
640	100000	BIT14=	40000
641	040000	BIT13=	20000
642	020000	BIT12=	10000
643	010000	BIT11=	4000
644	004000	BIT10=	2000
645	002000	BIT09=	1000
646	001000	BIT08=	400
647	000400	BIT07=	200
648	000200	BIT06=	100
649	000100	BIT05=	40
650	000040	BIT04=	20
651	000020	BIT03=	10
652	000010	BIT02=	4
653	000004	BIT01=	2
654	000002	BIT00=	1
655	000001	EQUIV	BIT09, BIT9
656		EQUIV	BIT08, BIT8
657		EQUIV	BIT07, BIT7
658		EQUIV	BIT06, BIT6
659		EQUIV	BIT05, BIT5
660		EQUIV	BIT04, BIT4
661		EQUIV	BIT03, BIT3
662		EQUIV	BIT02, BIT2
663		EQUIV	BIT01, BIT1
664		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				
669	000010			

670	000014	TBITVEC=14	;; "T" BIT
671	000014	TRTVEC= 14	;; TRACE TRAP
672	000014	BPTVEC= 14	;; BREAKPOINT TRAP (BPT)
673	000020	IOTVEC= 20	;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
674	000024	PWRVEC= 24	;; POWER FAIL
675	000030	EMTVEC= 30	;; EMULATOR TRAP (EMT) **ERPOR**
676	000034	TRAPVEC=34	;; "TRAP" TRAP
677	000060	TKVEC= 60	;; TTY KEYBOARD VECTOR
678	000064	TPVEC= 64	;; TTY PRINTER VECTOR
679	000240	PIRQVEC=240	;; PROGRAM INTERRUPT REQUEST VECTOR

```

680                                     , STANDARD INTERRUPT VECTORS
681
682
683                                     . =174
684 000174 000000  DISPREG 0
685 000176 000000  SWREG. 0
686                                     =200
687 000200 000167 001746  JMP      START      , GO TO START OF PROGRAM
688
689
690

```

```

691                                     =1100
692 001100 000000  WORD 0
693 001102 177570  LIGHTS 177570
694
695
696

```

, PROGRAM CONTROL PARAMETERS

```

698
699 001104 000000  RETURN 0
700 001106 000000  NEXT 0      , ADDRESS OF NEXT TEST TO BE EXECUTED
701 001110 000000  LOCK 0      , ADDRESS FOR LOCK ON CURRENT DATA
702 001112 000000  PASCNT 0   , ADDRESS CONTAINING PASS COLNT
703 001114 000000  ERRCNT 0   , ERROR COUNT
704 001116 000000  SAVSP 0    , STACK POINTER STORAGE
705

```

, PROGRAM VARIABLES

```

706
707
708 001120 000020  HOLD 20    , TEMPORARY STORAGE=DELAY TIME FOR CABLES
709 001122 000000  SHIFT 0    , TEMPORARY STORAGE= # OF SHIFTS PER CHAR
710 001124 000000  COUNT 0    , TEMPORARY STORAGE= # OF TIMES A CHAR WILL BE SENT
711 001126 000000  SAVPC 0    , PROGRAM COUNTER STORAGE
712 001130 000000  HL00 0
713 001132 000000  HL01 0
714 001134 000000  HL02 0
715 001136 000000  HL03 0
716 001140 000000  HL04 0
717 001142 000000  HL05 0
718 001144 000000  HL06 0
719

```



```

720      ,PROGRAM CONVERSATIONAL PARAMETERS
721 001146 377      SYNCNO  BYTE 377      ;# OF SYNC CHARS REQ'D FOR SYNC'ZATION
722 001147 377      SEXMIT  BYTE 377      ;SEC XMIT JUMPER "IN"
723 001150 377      SEREC   BYTE 377      ;SEC REC JUMPER "IN"
724 001151 377      OPTCLR  BYTE 377      ;OPTIONAL JUMPER CLR "IN"
725 001152 000      MULTD   BYTE 0       ;NO MULTIPLE DEVICE FLAG
726 001153 377      JMRBY   BYTE 377      ;EXTERNAL MODEM BYPASS JUMPER "IN"
727      EVEN
728
729      ,PROGRAM MULTIPLE DEVICE PARAMETERS
730 001154 000000   BASEADD  0       ;PROG CONTROLLED 1ST DEVICE ADDR
731 001156 000000   KEEPADD  0       ;SAVED 1ST DEVICE ADDR
732 001160 000000   LASTADD  0       ;LAST DEVICE RXCSR ADDR
733 001162 000000   BASEIV   0       ;PROG CONTROLLED IV
734 001164 000000   KEEPIV   0       ;SAVED INTR VECTOR
735 001166 000000   ACTREG.  0       ;ACTIVE REGISTER ,,,MODIFY THIS
736      ,LOCATION TO DISQUALIFY OR QUALIFY
737      ,DEVICES (1= RUN,,,0= DON'T RUN)
738 001170 000000   ROTADD.  0       ;ROTATING POINTER FOR ACTREG POINTS
739      ,TO DEVICE PRESENTLY UNDER TEST WHEN RUNNING MULTIPLE DEVICES
740
741      ,PROGRAM CONTROL FLAGS
742
743 001172 000      INIFLG  BYTE 0       ;PROGRAM INITIALIZATION FLAG
744 001173 000      STFLG   BYTE 0       ;TEST START FLAG
745 001174 000      LOKFLG  BYTE 0       ;LOCK ON CURRENT TEST FLAG
746      001176      EVEN
747      001400      =1400
748
749
    
```

750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805

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  
001400  
  
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)  
 POPRO=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  
 RECACT=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  
 SYN SCH=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  
 OVERRUN=BIT14 , OVERRUN  
 FRMERR=BIT13 , FRAME ERROR  
 PARERR=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 AVAILABLE  
 MTDATA=BIT14 , MAINT DATA  
 CLK=BIT13 , CLK  
 BITW=BIT10 , BIT WINDOW  
 MRESET=BIT8 , MASTER RESET  
 TXDONE=BIT7 , XMIT DONE  
 TXINTE=BIT6 , XMIT INTP ENABLE

806	000040	DNAINTE=BIT5	,DNA INTR ENAB
807	000020	SEND=BIT4	,SEND
808	000010	HDXEN=BIT3	,HDX/FDX
809	000001	BREAK=BIT0	,BREAK
810		,TXCSR WRD DEFINITIONS	
811	000000	USER=0	,USER MODE
812	004000	MINT=4000	,MAINT INT MODE
813	010000	MEXT=10000	,MAINT EXT MODE
814	014000	SYSTST=14000	,SYSTEM TEST MODE

Line	Address	Value	Label	Type	Value	Description
815			SBTTL			COMMON TAGS
817						*****
818						*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
819						*USED IN THE PROGRAM
821	001400		=			
822	001400		SCMTAG			START OF COMMON TAGS
823	001400	000000		WORD	0	
824	001402	000	STSTNM	BYTE	0	CONTAINS THE TEST NUMBER
825	001403	000	SERFLG	BYTE	0	CONTAINS ERROR FLAG
826	001404	000000	SICNT	WORD	0	CONTAINS SUBTEST ITERATION COUNT
827	001406	000000	SLPADR	WORD	0	CONTAINS SCOPE LOOP ADDRESS
828	001410	000000	SLPERR	WORD	0	CONTAINS SCOPE RETURN FOR ERRORS
829	001412	000000	SERTTL	WORD	0	CONTAINS TOTAL ERRORS DETECTED
830	001414	000	SITEMB	BYTE	0	CONTAINS ITEM CONTROL BYTE
831	001415	001	SERMAX	BYTE	1	CONTAINS MAX. ERRORS PER TEST
832	001416	000000	SERRPC	WORD	0	CONTAINS PC OF LAST ERROR INSTRUCT ON
833	001420	000000	SGDADR	WORD	0	CONTAINS ADDRESS OF 'GOOD' DATA
834	001422	000000	SBDADR	WORD	0	CONTAINS ADDRESS OF 'BAD' DATA
835	001424	000000	SGDDAT	WORD	0	CONTAINS 'GOOD' DATA
836	001426	000000	SBDDAT	WORD	0	CONTAINS 'BAD' DATA
837	001430	000000		WORD	0	RESERVED--NOT TO BE USED
838	001432	000000		WORD	0	
839	001434	000	SAUTOB	BYTE	0	AUTOMATIC MODE INDICATOR
840	001435	000	SINTAG	BYTE	0	INTERRUPT MODE INDICATOR
841	001436	000000		WORD	0	
842	001440	177570	SWR	WORD	DSWR	ADDRESS OF SWITCH REGISTER
843	001442	177570	D SPLAY	WORD	DDISP	ADDRESS OF DISPLAY REGISTER
844	001444	177560	STKS			TTY KBD STATUS
845	001446	177562	STKB			TTY KBD BUFFER
846	001450	177564	STPS			TTY PRINTER STATUS REG ADDRESS
847	001452	177566	STPB			TTY PRINTER BUFFER REG ADDRESS
848	001454	000	SNULL	BYTE	0	CONTAINS NULL CHARACTER FOR FILLS
849	001455	002	SFILLS	BYTE	2	CONTAINS # OF FILLER CHARACTERS REQUIRED
850	001456	012	SFILLC	BYTE	12	INSERT FILL CHARS AFTER A "LINE FEED"
851	001457	000	STPFLG	BYTE	0	"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
852	001460	000000	SREGAD	WORD	0	CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED
853						
854	001462	000000	SREG0	WORD	0	CONTAINS ((\$REGAD)+0)
855	001464	000000	SREG1	WORD	0	CONTAINS ((\$REGAD)+2)
856	001466	000000	SREG2	WORD	0	CONTAINS ((\$REGAD)+4)
857	001470	000000	SREG3	WORD	0	CONTAINS ((\$REGAD)+6)
858	001472	000000	SREG4	WORD	0	CONTAINS ((\$REGAD)+10)
859	001474	000000	SREG5	WORD	0	CONTAINS ((\$REGAD)+12)
860	001476	000000	STMP0	WORD	0	USER DEFINED
861	001500	000000	STMP1	WORD	0	USER DEFINED
862	001502	000000	STMP2	WORD	0	USER DEFINED
863	001504	000000	STMP3	WORD	0	USER DEFINED
864	001506	000000	STMP4	WORD	0	USER DEFINED
865	001510	000000	STMP5	WORD	0	USER DEFINED
866	001512	000000	STIMES	0		MAX NUMBER OF ITERATIONS
867	001514	000000	SESCAPE	0		ESCAPE ON ERROR ADDRESS
868	001516	177607 000377	SBELL	ASCIZ	<207><377><377>	CODE FOR BELL
869	001522	077	SQUES	ASCII	/?	QUESTION MARK
870	001523	015	SCRLF	ASCII	<15>	CARRIAGE RETURN



871	001524	000012	\$LF	ASCIZ	<12>	;;LINE FEED
872			..	*****		
873			SBTTL	APT MAILBOX-ETABLE		
874						
875			..	*****		
876			EVEN			
877	001526		\$MAIL			;; APT MAILBOX
878	001526	000000	\$MSGTY	WORD	AMSGTY	;; MESSAGE TYPE CODE
879	001530	000000	\$FATAL	WORD	AFATAL	;; FATAL ERROR NUMBER
880	001532	000000	\$TESTN	WORD	ATESTN	;; TEST NUMBER
881	001534	000000	\$PASS	WORD	APASS	;; PASS COUNT
882	001536	000000	\$DEVCT	WORD	ADEVCT	;; DEVICE COUNT
883	001540	000000	\$UNIT	WORD	AUNIT	;; I/O UNIT NUMBER
884	001542	000000	\$MSGAD	WORD	AMSGAD	;; MESSAGE ADDRESS
885	001544	000000	\$MSGLG	WORD	AMSGLG	;; MESSAGE LENGTH
886	001546		\$ETABLE			;; APT ENVIRONMENT TABLE
887	001546	000	\$ENV	BYTE	AENV	;; ENVIRONMENT BYTE
888	001547	000	\$ENVM	BYTE	AENVM	;; ENVIRONMENT MODE BITS
889	001550	000000	\$SWREG	WORD	ASWREG	;; APT SWITCH REGISTER
890	001552	000000	\$USWR	WORD	AUSWR	;; USER SWITCHES
891	001554	000000	\$CPUOP	WORD	ACPUOP	;; CPU TYPE, OPTIONS
892			.*			BITS 15-11=CPU TYPE
893			.*			11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
894			.*			11/70=06, PDQ=07, Q=10
895			.*			BIT 10=REAL TIME CLOCK
896			.*			BIT 9=FLOATING POINT PROCESSOR
897			.*			BIT 8=MEMORY MANAGEMENT
898	001556	000	\$MAMS1	BYTE	AMAMS1	;; HIGH ADDRESS, M. S. BYTE
899	001557	000	\$MTYP1	BYTE	AMTYP1	;; MEM TYPE, BLK#1
900			.*			MEM TYPE BYTE -- (HIGH BYTE)
901			.*			900 NSEC CORE=001
902			.*			300 NSEC BIPOLAR=002
903			.*			500 NSEC MOS=003
904	001560	000000	\$MADR1	WORD	AMADR1	;; HIGH ADDRESS, BLK#1
905			.*			MEM LAST ADDR =3 BYTES. THIS WORD AND LOW OF "TYPE" ABOVE
906	001562	000	\$MAMS2	BYTE	AMAMS2	;; HIGH ADDRESS, M S. BYTE
907	001563	000	\$MTYP2	BYTE	AMTYP2	;; MEM TYPE, BLK#2
908	001564	000000	\$MADR2	WORD	AMADR2	;; MEM LAST ADDRESS, BLK#2
909	001566	000	\$MAMS3	BYTE	AMAMS3	;; HIGH ADDRESS, M S. BYTE
910	001567	000	\$MTYP3	BYTE	AMTYP3	;; MEM TYPE, BLK#3
911	001570	000000	\$MADR3	WORD	AMADR3	;; MEM LAST ADDRESS, BLK#3
912	001572	000	\$MAMS4	BYTE	AMAMS4	;; HIGH ADDRESS, M. S. BYTE
913	001573	000	\$MTYP4	BYTE	AMTYP4	;; MEM TYPE, BLK#4
914	001574	000000	\$MADR4	WORD	AMADR4	;; MEM LAST ADDRESS, BLK#4
915	001576	000000	\$VECT1	WORD	AVECT1	;; INTERRUPT VECTOR#1, BUS PRIORITY#1
916	001600	000000	\$VECT2	WORD	AVECT2	;; INTERRUPT VECTOR#2, BUS PRIORITY#2
917	001602	000000	\$BASE	WORD	ABASE	;; BASE ADDRESS OF EQUIPMENT UNDER TEST
918	001604	000000	\$DEVN	WORD	ADEVN	;; DEVICE MAP
919	001606	000000	\$CDW1	WORD	ACDW1	;; CONTROLLER DESCRIPTION WORD#1
920	001610	000000	\$CDW2	WORD	ACDW2	;; CONTROLLER DESCRIPTION WORD#2
921	001612	000000	\$DDW0	WORD	ADDW0	;; DEVICE DESCRIPTOR WORD#0
922	001614	000000	\$DDW1	WORD	ADDW1	;; DEVICE DESCRIPTOR WORD#1
923	001616	000000	\$DDW2	WORD	ADDW2	;; DEVICE DESCRIPTOR WORD#2
924	001620	000000	\$DDW3	WORD	ADDW3	;; DEVICE DESCRIPTOR WORD#3
925	001622	000000	\$DDW4	WORD	ADDW4	;; DEVICE DESCRIPTOR WORD#4
926	001624	000000	\$DDW5	WORD	ADDW5	;; DEVICE DESCRIPTOR WORD#5



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  
998

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  
001400  
  
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  
OVERRUN=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

999	000040	DNAINTE=BIT5	,DNA INTR ENAB
1000	000020	SEND=BIT4	,SEND
1001	000010	HDXEN=BIT3	,HDX/FDX
1002	000001	BREAK=BIT0	,BREAK
1003		,TXCSR WRD DEFINITIONS	
1004	000000	USER=0	,USER MODE
1005	004000	MINT=4000	,MAINT INT MODE
1006	010000	MEXT=10000	,MAINT EXT MODE
1007	014000	SYSTST=14000	,SYSTEM TEST MODE



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  
 1063

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  
  
 001712 160010  
 001714 160011  
 001716 160012  
 001720 160013  
 001722 160012  
 001724 160013  
 001726 160014  
 001730 160015  
 001732 160016  
 001734 160017  
  
 001736 000770  
 001740 000772  
 001742 000774  
 001744 000776  
  
 001746 020040 051105 047522  
 001754 020122 041520 000040  
 001762 020040 047503 050115  
 001770 051101 051511 047117  
 001776 042440 051122 051117  
 002004 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 \$ITEMB THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT  
 \*NOTE1 IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC)  
 \*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		
FM2	.ERROR 2	RECEIVER FRROR
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  
 HRXCSP 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  
 OUTIV 774 ;XMIT INTR VECTOR  
 OUTIS 776 ;XMIT INTR STATUS

.ERROR MESSAGES

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

1064	002012	044507	052123	051105					
1065	002020	000123							
1066	002022	020040	042522	042503	EM2	ASCIZ	/ RECEIVER ERROR/		
1067	002030	053111	051105	042440					
1068	002036	051122	051117	000					
1069	002043	040	052040	040522	EM3	ASCIZ	/ TRANSMITTER ERROR/		
1070	002050	051516	044515	052124					
1071	002056	051105	042440	051122					
1072	002064	051117	000						
1073							. DATA HEADERS FOR ERROR MESSAGES		
1074	002067	105	051122	041520	DH1	ASCIZ	/ERRPC WANTED ACTUAL/		
1075	002074	020040	040527	052116					
1076	002102	042105	020040	041501					
1077	002110	052524	046101	000					
1078		002116							
1079							. DATA TABLES FOR ERROR MESSAGES		
1080	002116	001416	001130	001132	DT1	WORD	\$ERRPC, HLDO, HLD1, 0		
1081	002124	000000							
1082									
1083	002126	001416	000000		DT4	WORD	\$ERRPC, 0		
1084									
1085	002132	000	000	000	DF1	BYTE	0, 0, 0, 0		
1086	002135	000							
1087									
1088									
1089									
1090									
1091							. *****		
1092		002136					. HOOKS REQUIRED BY ACT11		
1093		000046					\$SVPCL=	. SAVE PC	
1094	000046	012644					=46		
1095		000052					\$ENDAD	. , 1) SET LOC 46 TO ADDRESS OF SENDAD IN SEOP	
1096	000052	000000					=52		
1097		002136					WORD 0	. , 2) SET LOC 52 TO ZERO	
1098							= \$SVPCL	. , RESTORE PC	
1099							SBTTL	APT PARAMETER BLOCK	
1100									
1101							. *****		
1102							. SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT		
1103		002136					. *****		
1104		000024					\$X=	. , SAVE CURRENT LOCATION	
1105	000024	000200					=24	. , SET POWER FAIL TO POINT TO START OF PROGRAM	
1106		000044					200	. , FOR APT START UP	
1107	000044	002136					=44	. , POINT TO APT INDIRECT ADDRESS PNTR	
1108		002136					\$APTHDR	. , POINT TO APT HEADER BLOCK	
1109							= \$X	. , RESET LOCATION COUNTER	
1110							. *****		
1111							. SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC		
1112							. INTERFACE SPEC		
1113	002136						\$APTHD		
1114	002136	000000					\$HIBTS	WORD 0	. , TWO HIGH BITS OF 18 BIT MAILBOX ADDR
1115	002140	001526					\$MADR	WORD \$MAIL	. , ADDRESS OF APT MAILBOX (BITS 0-15)
1116	002142	000010					\$TSTM	WORD 10	. , RUN TIM OF LONGEST TEST
1117	002144	000010					\$PASTM	WORD 10	. , RUN TIME IN SECS OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1118	002146	000000					\$UN!TM	WORD	. , ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1119	002150	000052						WORD	SETEND-\$MAIL/2 . , LENGTH MAILBOX-ETABLE (WORDS)

```

1120
1121
1122          , PROGRAM INITIALIZATION
1123          , LOCK OUT INTERRUPTS
1124          , SET UP PROCESSOR STACK
1125          , SET UP POWER FAIL VECTOR
1126          , CLEAR PROGRAM CONTROL FLAGS AND COUNTS
1127          , TYPE TITLE MESSAGE
1128
1129 002152    START.
1130          SBTTL INITIALIZE THE COMMON TAGS
1131          .. CLEAR THE COMMON TAGS ($CMTAG) AREA
1132 002152 012706 001400    MOV    # $CMTAG, R6          .. FIRST LOCATION TO BE CLEARED
1133 002156 005026          CLR    (R6)+                .. CLEAR MEMORY LOCATION
1134 002160 022706 001440    CMP    # SWR, R6          .. DONE?
1135 002164 001374          BNE    -6                  .. LOOP BACK IF NO
1136 002166 012706 001100    MOV    ## STACK, SP      .. SETUP THE STACK POINTER
1137          .. INITIALIZE A FEW VECTORS
1138 002172 012737 016270 000020    MOV    $$SCOPE, @# IOTVEC ; IOT VECTOR FOR SCOPE ROUTINE
1139 002200 012737 000340 000022    MOV    #340, @# IOTVEC+2 ; LEVEL 7
1140 002206 012737 014160 000030    MOV    $ERROR, @# EMTVEC ; EMT VECTOR FOR ERROR ROUTINE
1141 002214 012737 000340 000032    MOV    #340, @# EMTVEC+2 ; LEVEL 7
1142 002222 012737 016624 000034    MOV    $STRAP, @# TRAPVEC ; TRAP VECTOR FOR TRAP CALLS
1143 002230 012737 000340 000036    MOV    #340, @# TRAPVEC+2, LEVEL 7
1144 002236 012737 014762 000024    MOV    $SPWRDN, @# PWRVEC ; POWER FAILURE VECTOR
1145 002244 012737 000340 000026    MOV    #340, @# PWRVEC+2 ; LEVEL 7
1146 002252 005067 177234          CLR    $TIMES            .. INITIALIZE NUMBER OF ITERATIONS
1147 002256 005067 177232          CLR    $ESCAPE         .. CLEAR THE ESCAPE ON ERROR ADDRESS
1148 002262 112767 000001 177125    MOV    #1, $SERMAX      .. ALLOW ONE ERROR PER TEST
1149 002270 012767 002270 177110    MOV    # , $LPADR      .. INITIALIZE THE LOOP ADDRESS FOR SCOPE
1150 002276 012767 002276 177104    MOV    # , $LPERR      .. SETUP THE ERROR LOOP ADDRESS
1151          .. SIZE FOR A HARDWARE SWITCH REGISTER IF NOT FOUND OR IT IS
1152          .. EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER
1153 002304 013746 000004          MOV    @# ERRVEC, -(SP) ; SAVE ERROR VECTOR
1154 002310 012737 002344 000004    MOV    #64$, @# ERRVEC ; SET UP ERROR VECTOR
1155 002316 012767 177570 177114    MOV    #DSWR, SWR      ; SETUP FOR A HARDWARE SWICH REGISTER
1156 002324 012767 177570 177110    MOV    #DDISP, DISPLAY ; AND A HARDWARE DISPLAY REGISTER
1157 002332 022777 177777 177100    CMP    #-1, @SWR      ; TRY TO REFERENCE HARDWARE SWR
1158 002340 001012          BNE    66$             ; BRANCH IF NO TIMEOUT TRAP OCCURRED
1159          ; AND THE HARDWARE SWR IS NOT = -1
1160 002342 000403          BR    65$             ; BRANCH IF NO TIMEOUT
1161 002344 012716 002352          64$: MOV    #65$, (SP)     ; SET UP FOR TRAP RETURN
1162 002350 000002          RTI
1163 002352 012767 000176 177060 65$ MOV    #SWREG, SWR     ; POINT TO SOFTWARE SWR
1164 002360 012767 000174 177054    MOV    #DISPREG, DISPLAY
1165 002366 012637 000004          66$ MOV    (SP)+, @# ERRVEC ; RESTORE ERROR VECTOR
1166
1167          CLR    $PASS          ; CLEAR PASS COUNT
1168 002372 005067 177136          BITB  #APTSIZE, $ENVM ; TEST USER SIZE UNDER APT
1169 002376 132767 000200 177143    BEQ    67$            ; YES, USE NON-APT SWITCH
1170 002404 001403          MOV    #SSHREG, SWR   ; NO, USE APT SWITCH REGISTER
1171 002406 012767 001550 177024 67$
1172 002414          MOV    #STACK, SP    ; SET STACK
1173 002420 106427 000340          MTPS  #340           ; LOCK INTERPUPTS
1174 002424 012737 014762 000024    MOV    # PFAIL, @#24  ; SET UP POWER FAIL VECTOR
1175 002432 105067 176535          CLRB  $STFLG         ; CLEAR STAPT FLAG
  
```

INITIALIZE THE COMMON TAGS

1176	002436	005067	176450		CLR	PASCNT		; CLEAR PASS COUNT
1177	002442	105067	176735		CLRB	SERFLG		; CLEAR ERROR FLAG
1178	002446	005067	176740		CLR	SERTTL		; CLEAR ERROR COUNT
1179	002452	005067	176740		CLR	SERRPC		; CLEAR LAST EPROR POINTER
1180	002456	012767	000001	176716	MOV	#1, STSTNM		; SET UP FOR TEST 1
1181	002464	012767	002152	176412	MOV	# START, RETURN		; SET UP FOR POWER FAIL BEFORE
1182								; TESTING STARTS
1183	002472	013746	000006		MOV	@#6, -(SP)		
1184	002476	013746	000004		MOV	@#4, -(SP)		
1185	002502	012737	002516	000004	MOV	#15, @#4		
1186	002510	005777	176724		TST	@SWR		
1187	002514	000407			BR	25		
1188	002516	012767	000176	176714	15	MOV	#SWREG, SWR	
1189	002524	012767	000174	176710		MOV	#DISPREG, DISPLAY	
1190	002532	022626			CMP	(SP)+, (SP)+		
1191	002534	012637	000004		25	MOV	(SP)+, @#4	
1192	002540	012637	000006		MOV	(SP)+, @#6		
1193	002544	022767	000176	176666		CMP	#SWREG, SWR	
1194	002552	001007			BNE	35		
1195	002554	005737	000042		TST	@#42		; CHECK FOR CHAIN
1196	002560	001402			BEQ	335		
1197	002562	000167	000522		JMP	BEGIN		
1198	002566	004767	010154		335	JSR	PC, CNTLU	
1199	002572	105767	176374		35	TSTB	INIFIG	; HAS INITIALIZATION BEEN PERFORMED
1200	002576	001004			BNE	ONCE		
1201	002600	104401	01E122		TYPE	, MTITLE		; TYPE TITLE MESSAGE
1202	002604	105167	176362		COMB	INIFLG		; IF NOT SET FLAG AND DO
1203	002610	105767	176732		ONCE	TSTB	SENV	; APT CONTROL?
1204	002614	001410			BEQ	115		; BR IF NO
1205	002616	032767	000001	176726		BIT	#1, \$USWR	; EXTENAL JUMPER ON?
1206	002624	001002			BNE	125		; NO
1207	002626	105067	176321		CLRB	JMRBY		; CLEAR FLAG
1208	002632	000167	000452		125	JMP	BEGIN	; GO DO IT
1209	002636	032777	000001	176574	115	BIT	#SW00, @SWR	; RESELECT VECTOR & CONTROL REG?
1210	002644	001002			BNE	15		
1211	002646	000167	000436		JMP	BEGIN		
1212	002652	012700	000300		15	MOV	#300, R0	; RESTORE VECTOR AREA TO TRAPCATCHER
1213	002656	012701	000302			MOV	#302, R1	; START AT LOCATION 300
1214	002662	012702	000004			MOV	#4, R2	
1215	002666	010110			25	MOV	R1, (R0)	
1216	002670	005011				CLR	(R1)	
1217	002672	060200				ADD	R2, R0	
1218	002674	060201				ADD	R2, R1	
1219	002676	022701	001000			CMP	#1000, R1	; END AT LOCAT ON 776
1220	002702	002771				BLT	25	
1221	002704	104406				INSTR		; OUTPUT MESSAGE & GET INPUT STRING
1222	002706	015170				MREGAD		; MESSAGE
1223	002710	104410				PARAM		; CONVERT STRING
1224	002712	160000				160000		; LOW LIMIT
1225	002714	167776				167776		; HIGH LIMIT
1226	002716	017120				DUBASE		; STORE AT THIS LOCATION
1227	002720	001			BYTE	1		; MASK
1228	002721	001			BYTE	1		; HOW MANY TIMES + 2
1229	002722	016767	014172	176226		MOV	DUBASE, KEEPADD	; SAVE
1230	002730	004767	014032			JSR	PC, DUADDR	
1231	002734	016767	176216	176212		MOV	KEEPADD, BASEADD	; RESTORE FOR ROTATION

INITIALIZE THE COMMON TAGS

1232	002742	104406				INSTR	; OUTPUT MESSAGE & GET INPUT STRING
1233	002744	015155				MVECTO	; MESSAGE
1234	002746	104410				PARAM	; CONVERT STRING
1235	002750	000300				300	; LOW LIMIT
1236	002752	000776				776	; HIGH LIMIT
1237	002754	001736				DURIV	; STORE AT THIS LOCATION
1238	002756	001			BYTE	1	; MASK
1239	002757	004			BYTE	4	; HOW MANY TIMES + 2
1240	002760	016767	176752	176176		MOV	DURIV,KEEPIV ; SAVE
1241	002766	016767	176744	176166		MOV	DURIV,BASEIV ; SET UP FOR ROTATION
1242	002774	104406				INSTR	; OUTPUT MESSAGE & GET INPUT STRING
1243	002776	015220				MMULT	; MESSAGE
1244	003000	104414				SETFLG	; SET FLAG BASED UPON INPUT STRING
1245	003002	001152				MULTD	; THIS FLAG
1246	003004	105767	176142			TSTB	MULTD ; ARE THERE MULTIPLE DEVICES ; ON THE SYSTEM ?
1247							
1248	003010	100406				BMI	BBB ; YES, ASK NEXT QUESTION
1249	003012	005067	176150			CLR	ACTREG
1250	003016	005067	176146			CLR	ROTADD
1251	003022	000167	000140			JMP	OUTMUL ; JUMP AROUND NEXT QUESTION
1252	003026				BBB		
1253	003026	104406				INSTR	; OUTPUT MESSAGE & GET INPUT STRING
1254	003030	015247				MLASTD	; MESSAGE
1255	003032	104410				PARAM	; CONVERT STRING
1256	003034	160000				160000	; LOW LIMIT
1257	003036	167776				167776	; HIGH LIMIT
1258	003040	001160				LASTADD	; STORE AT THIS LOCATION
1259	003042	001			BYTE	1	; MASK
1260	003043	001			BYTE	1	; HOW MANY TIMES + 2
1261							; THE FOLLOWING ROUTINE SETS UP ACTREG FOR THE FIRST TIME
1262	003044	012767	000001	176116	1\$	MOV	#1,ROTADD ; SET UP POINTER
1263	003052	005067	176110			CLR	ACTREG ; CLR ACTIVE REGISTER
1264	003056	056767	176106	176102	2\$	BIS	ROTADD,ACTREG ; MAKE THIS DEVICE ACTIVE
1265	003064	000241				CLC	
1266	003066	006167	176076			ROL	ROTADD ; SET UP POINTER
1267	003072	103421				BCS	3\$ ; ARE YOU OUT OF RANGE ?
1268	003074	062767	000010	176052		ADD	#10,BASEADD ; SET UP BASE ADDRESS
1269	003102	026767	176052	176044		CMF	LASTADD,BASEADD ; IS THIS THE LAST DEVICE ?
1270	003110	101362				BMI	2\$ ; NO DO IT AGAIN
1271	003112	056767	176052	176046		BIS	ACTADD,ACTREG ; THIS ASSUMES THAT THERE ARE AT ; LEAST TWO DEVICES WHEN YOU ANSWER YES TO ; MULTIPLE DEVICE QUESTION
1272							
1273							
1274	003120	012767	000001	176042	4\$	MOV	#1,ROTADD ; SET UP FOR LATER USE IN END OF PASS ROUTINE
1275	003126	016767	176024	176020		MOV	KEEPADD,BASEADD ; DITTO
1276	003134	000414				BR	OUTMUL ; CONTINUE QUESTIONS
1277	003136	016767	176014	176010	3\$	MOV	KEEPADD,BASEADD ; RESTORE
1278	003144	104406				INSTR	; OUTPUT MESSAGE & GET INPUT STRING
1279	003146	015343				MRANGE	; MESSAGE
1280	003150	104410				PARAM	; CONVERT STRING
1281	003152	160000				160000	; LOW LIMIT
1282	003154	167776				167776	; HIGH LIMIT
1283	003156	001160				LASTADD	; STORE AT THIS LOCATION
1284	003160	001			BYTE	1	; MASK
1285	003161	001			BYTE	1	; HOW MANY TIMES + 2
1286	003162	000167	177656			JMP	1\$ ; DO IT AGAIN
1287	003166	012767	000340	013566	OUTMUL	MOV	#340,DUPRT

INITIALIZE THE COMMON TAGS

```

1288 003174 004767 013512 JSR PC,DULEV
1289 ,COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
1290 ,BUFFER TO THE CHARACTERS "1" AND "2"
1291 ,IF THE CHARACTER IS "1" CLEAR THE FLAG
1292 ,IF THE CHARACTER IS "2" SET THE FLAG
1293 003200 AAR
1294 003200 104406 INSTR ,OUTPUT MESSAGE & GET INPUT STR NG
1295 003202 015561 MSYNC ;MESSAGE
1296 003204 122767 000061 012710 3$ CMPB #'1,INBUF ,IS IT "1" ?
1297 003212 001003 BNE 1$
1298 003214 105067 175726 CLRB SYNCNO ,000
1299 003220 000412 BR 4$
1300 003222 122767 000062 012672 1$ CMPB #'2,INBUF ,IS IT "2" ?
1301 003230 001004 BNE 2$
1302 003232 112767 177777 175706 MOVB #-1,SYNCNO ,377
1303 003240 000402 BR 4$
1304 003242 104407 2$ INSTER ,RETRY
1305 003244 000757 BR 3$
1306 003246 000240 4$ NOP
1307 003250 104406 INSTR ,OUTPUT MESSAGE & GET INPUT STRING
1308 003252 015627 MWIRE6 ,MESSAGE
1309 003254 104414 SETFLG ,SET FLAG BASED UPON INPUT STRING
1310 003256 001147 SEXMIT ,THIS FLAG
1311 003260 104406 INSTR ,OUTPUT MESSAGE & GET INPUT STRING
1312 003262 015700 MWIRE5 ,MESSAGE
1313 003264 104414 SETFLG ,SET FLAG BASED UPON INPUT STRING
1314 003266 001150 SEREC ,THIS FLAG
1315 003270 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1316 003272 015750 MWIRE4 ,MESSAGE
1317 003274 104414 SETFLG ,SET FLAG BASED UPON INPUT STRING
1318 003276 001151 OPTCLR ,THIS FLAG
1319 003300 104406 INSTR ,OUTPUT MESSAGE & GET INPUT STRING
1320 003302 016027 MEXTJ ,MESSAGE
1321 003304 104414 SETFLG ,SET FLAG BASED UPON INPUT STRING
1322 003306 001153 JMRBY ,THIS FLAG
1323
1324 ,TEST START AND RESTART
1325
1326 003310 012706 001100 BEGIN MOV #STACK,SP ,SET UP STACK
1327 003314 106427 000340 MTPS #340 LOCK OUT INTERRUPTS
1328 003320 032777 000002 176112 BIT #SW01,DSWR ,IF SW01=1, GET STARTING PC
1329 003326 001413 BEQ 3$
1330 003330 104406 INSTR ,OUTPUT MESSAGE & GET INPUT STRING
1331 003332 015513 MTSTPC ,MESSAGE
1332 003334 104410 PARAM ,CONVERT STRING
1333 003336 003374 TST1 ;LOW LIMIT
1334 003340 017500 17500 ,HIGH LIMIT
1335 003342 001402 $STNM ,STORE AT THIS LOCATION
1336 003344 001 BYTE 1 ,MASK
1337 003345 001 BYTE 1 ,HOW MANY TIMES + 2
1338 003346 016767 176030 175530 MOV $STNM,RETURN
1339 003354 000403 BR 4$
1340 003356 012767 003374 175520 3$ MOV #TST1,RETURN ,START AT TEST 1
1341 003364 104401 015507 4$ TYPE ,MR ,TYPE R
1342 003370 000177 175510 JMP @RETURN ,START TESTING
1343

```

```
1344
1345          ,, THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
1346          ''
1347          ,, *****
1348 003374 000004 TST1 SCOPE
1349 003376 012737 017360 000004 MOV #TRPREG, @#4 , SETUP TRAPCATCHER
1350 003404 016737 174730 000006 MOV PR7, @#6 ,
1351 003412 105277 176274 INCB @RXCSR , TEST THIS REG
1352 003416 000401 BR +4 , IF OK JMP AROUND ERROR
1353 003420 104004 ERROR 4 , CHECK DEVICE REG ADDRESSES
1354 003422 105277 176266 INCB @HRXCSR , TEST UPPER BYTE THIS REGISTER
1355 003426 000401 BR +4 , IF OK JMP AROUND ERROR
1356 003430 104004 ERROR 4 , CHECK DEVICE REG ADDRESSES
1357 003432 012737 000006 000004 MOV #6, @#4 , RESTORE TRAPCATCHER
1358 003440 012737 000000 000006 MOV #0, @#6 ,
1359
1360 003446 012767 003777 175444 MOV #3777, HOLD , SET LONGER DELAY FOR TEST
1361          ,, THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
1362          ''
1363          ,, *****
1364 003454 000004 TST2 SCOPE
1365 003456 012737 017360 000004 MOV #TRPREG, @#4 , SETUP TRAPCATCHER
1366 003464 016737 174650 000006 MOV PR7, @#6 ,
1367 003472 105277 176220 INCB @RXDBUF , TEST THIS REG
1368 003476 000401 BR +4 , IF OK JMP AROUND ERROR
1369 003500 104004 ERROR 4 , CHECK DEVICE REG ADDRESSES
1370 003502 105277 176212 INCB @HRXDBUF , TEST UPPER BYTE THIS REGISTER
1371 003506 000401 BR +4 , IF OK JMP AROUND ERROR
1372 003510 104004 ERROR 4 , CHECK DEVICE REG ADDRESSES
1373 003512 012737 000006 000004 MOV #6, @#4 , RESTORE TRAPCATCHER
1374 003520 012737 000000 000006 MOV #0, @#6 ,
1375
1376          ,, THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
1377          ''
1378          ,, *****
1379 003526 000004 TST3 SCOPE
1380 003530 012737 017360 000004 MOV #TRPREG, @#4 , SETUP TRAPCATCHER
1381 003536 016737 174576 000006 MOV PR7, @#6 ,
1382 003544 105277 176152 INCB @PARCSR , TEST THIS REG
1383 003550 000401 BR +4 , IF OK JMP AROUND ERROR
1384 003552 104004 ERROR 4 , CHECK DEVICE REG ADDRESSES
1385 003554 105277 176144 INCB @HPARCSR , TEST UPPER BYTE THIS REGISTER
1386 003560 000401 BR +4 , IF OK JMP AROUND ERROR
1387 003562 104004 ERROR 4 , CHECK DEVICE REG ADDRESSES
1388 003564 012737 000006 000004 MOV #6, @#4 , RESTORE TRAPCATCHER
1389 003572 012737 000000 000006 MOV #0, @#6 ,
1390
1391          ,, THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
1392          ''
1393          ,, *****
1394 003600 000004 TST4 SCOPE
1395 003602 012737 017360 000004 MOV #TRPREG, @#4 , SETUP TRAPCATCHER
1396 003610 016737 174524 000006 MOV PR7, @#6 ,
1397 003616 105277 176104 INCB @TXCSR , TEST THIS REG
1398 003622 000401 BR +4 , IF OK JMP AROUND ERROR
1399 003624 104004 ERROR 4 , CHECK DEVICE REG ADDRESSES
```

```

1400 003626 105277 176076      INCB  @HTXCSR ,TEST UPPER BYTE THIS REGISTER
1401 003632 000401              BR      +4          , IF OK JMP AROUND ERROR
1402 003634 104004              ERROR  4            ,CHECK DEVICE REG ADDRESSES
1403 003636 012737 000006 000004  MOV    #6,@#4      ;RESTORE TRAPCATCHER
1404 003644 012737 000000 000006  MOV    #0,@#6      ,
1405
1406                          ,, THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
1407                          ,,
1408                          ,, *****
1409 003652 000004              TST5  SCOPE
1410 003654 012737 017360 000004  MOV    #TRPREG,@#4 ,SETUP TRAPCATCHER
1411 003662 016737 174452 000006  MOV    PR7,@#6 ,
1412 003670 105277 176036      INCB  @TXDBUF      ,TEST THIS REG
1413 003674 000401              BR      +4          , IF OK JMP AROUND ERROR
1414 003676 104004              ERROR  4            ,CHECK DEVICE REG ADDRESSES
1415 003700 105277 176030      INCB  @HTXDBUF     ,TEST UPPER BYTE THIS REGISTER
1416 003704 000401              BR      +4          , IF OK JMP AROUND ERROR
1417 003706 104004              ERROR  4            ,CHECK DEVICE REG ADDRESSES
1418 003710 012737 000006 000004  MOV    #6,@#4      ,RESTORE TRAPCATCHER
1419 003716 012737 000000 000006  MOV    #0,@#6      ,
1420
1421                          ,, BUS DRIVER TEST
1422                          ,,
1423                          ,, *****
1424 003724 000004              TST6  SCOPE
1425 003726 022777 000000 175776  CMP    #0,@TXDBUF
1426 003734 001401              BEQ    +4
1427 003736 104004              ERROR  4            ,READING TXDBUF SHOULD BE ALL ZERO'S
1428                          ,, THIS TEST PERFORMS MASTER PESET TESTING &
1429                          ,, TESTING OF READ/WRITE BIT DTR
1430                          ,,
1431                          ,, *****
1432 003740 000004              TST7  SCOPE
1433 003742 052777 000002 175742  BIS    #DTR,@RXCSR ,SET THIS BIT
1434 003750 032777 000002 175734  BIT    #DTR,@RXCSR ,TEST THIS BIT
1435 003756 001001              BNE    +4          ,BR IF "1"
1436 003760 104004              ERROR  4            ,THIS BIT SHOULD BE SET
1437 003762 042777 000002 175722  BIC    #DTR,@RXCSR ,CLR THIS BIT
1438 003770 032777 000002 175714  BIT    #DTR,@RXCSR ,TEST THIS BIT
1439 003776 001401              BEQ    +4          ,BR IF "0"
1440 004000 104004              ERROR  4            ,THIS BIT SHOULD BE CLP
1441                          ,NOW SET THIS BIT
1442 004002 052777 000002 175702  BIS    #DTR,@RXCSR
1443 004010 052777 000400 175710  BIS    #MRESET,@TXCSR ,MASTER RESET
1444                          ,, CHECK EXISTANCE OF OPTIONAL CLEAR JUMPER
1445                          ,,
1446 004016 105767 175127      TSTB  OPTCLR      ,TEST FLAG
1447 004022 100006      BPL    15          ,OPTIONAL CLR JUMPER IS NOT IN
1448 004024 032777 000002 1756c0  BIT    #DTR,@RXCSR ,TEST THIS BIT
1449 004032 001401              BEQ    +4          ,BR IF "0"
1450 004034 104004              ERROR  4            ,CHECK OUT MASTER PESET LOGIC
1451 004036 000405      BR      25          ,JMP AROUND
1452 004040 032777 000002 175644 15  BIT    #DTR,@RXCSR ,TEST THIS BIT
1453 004046 001001              BNE    +4          ,BR IF "1"
1454 004050 104004              ERROR  4            CHECK OUT OPTIONAL CLR JUMPER
1455 004052 000240              25  NOP

```



```

1456
1457      , , THIS TEST PERFORMS MASTER RESET TESTING &
1458      , , TESTING OF READ/WRITE BIT RTS
1459      , ,
1460      , , *****
1461 004054 000004 TST10 SCOPE
1462 004056 052777 000004 175626 BIS #RTS, @RXCSR , SET THIS BIT
1463 004064 032777 000004 175620 BIT #RTS, @RXCSR , TEST THIS BIT
1464 004072 001001 BNE +4 , BR IF "1"
1465 004074 104004 ERROR 4 , THIS BIT SHOULD BE SET
1466 004076 042777 000004 175606 BIC #RTS, @RXCSR , CLR THIS BIT
1467 004104 032777 000004 175600 BIT #RTS, @RXCSR , TEST THIS BIT
1468 004112 001401 BEQ +4 , BR IF "0"
1469 004114 104004 ERROR 4 , THIS BIT SHOULD BE CLR
1470 , NOW SET THIS BIT
1471 004116 052777 000004 175566 BIS #RTS, @RXCSR
1472 004124 052777 000400 175574 BIS #MRESET, @TXCSR , MASTER RESET
1473 , , CHECK EXISTANCE OF OPTIONAL CLEAR JUMPER
1474 , ,
1475 004132 105767 175013 TSTB OPTCLR , TEST FLAG
1476 004136 100006 BPL 15 , OPTIONAL CLR JUMPER IS NOT IN
1477 004140 032777 000004 175544 BIT #RTS, @RXCSR , TEST THIS BIT
1478 004146 001401 BEQ +4 , BR IF "0"
1479 004150 104004 ERROR 4 , CHECK OUT MASTER RESET LOGIC
1480 004152 000405 BR 25 , JMP AROUND
1481 004154 032777 000004 175530 15 BIT #RTS, @RXCSR , TEST THIS BIT
1482 004162 001001 BNE +4 , BR IF "1"
1483 004164 104004 ERROR 4 , CHECK OUT OPTIONAL CLR JUMPER
1484 004166 000240 25 NOP
1485
1486 , WAIT FOR CABLE DELAYS
1487 , *****
1488 , MODIFY "HOLD " ACCORDINGLY FOR FASTER OR SLOWER MACHINE
1489 , *****
1490 004170 016702 174724 MOV HOLD, R2 , SET DELAY TIME
1491 004174 005302 DEC R2
1492 004176 001376 BNE -2 , WAIT THIS TIME
1493 , OK NOW FALL THRU AND CONTINUE TESTING
1494 , EXIT STAGE LEFT CHINNG!
1495 , , THIS TEST PERFORMS MASTER RESET TESTING &
1496 , , TESTING OF READ/WRITE BIT STD
1497 , ,
1498 , , *****
1499 004200 000004 TST11 SCOPE
1500 004202 052777 000010 175502 BIS #STD, @RXCSR , SET THIS BIT
1501 004210 032777 000010 175474 BIT #STD, @RXCSR , TEST THIS BIT
1502 004216 001001 BNE +4 , BR IF "1"
1503 004220 104004 ERROR 4 , THIS BIT SHOULD BE SET
1504 004222 042777 000010 175422 BIC #STD, @RXCSR , CLR THIS BIT
1505 004230 032777 000010 175454 BIT #STD, @RXCSR , TEST THIS BIT
1506 004236 001401 BEQ +4 , BR IF "0"
1507 004240 104004 ERROR 4 , THIS BIT SHOULD BE CLR
1508 , NOW SET THIS BIT
1509 004242 052777 000010 175442 BIS #STD, @RXCSR
1510 004250 052777 000400 175450 BIS #MRESET, @TXCSR , MASTER RESET
1511 , , CHECK EXISTANCE OF OPTIONAL CLEAR JUMPER

```

```
1512
1513 004256 105767 174667      TSTB   OPTCLR      , TEST FLAG
1514 004262 100006              BPL     15         , OPTIONAL CLR JUMPER IS NOT N
1515 004264 032777 000010 175420  BIT    #STD, @RXCSR , TEST THIS BIT
1516 004272 001401              BEQ     +4         , BR IF "0"
1517 004274 104004              ERROR   4         , CHECK OUT MASTER RESET LOGIC
1518 004276 000405              BR      25         ; JMP AROUND
1519 004300 032777 000010 175404 15     BIT    #STD, @RXCSR , TEST THIS BIT
1520 004306 001001              BNE     +4         , BR IF "1"
1521 004310 104004              ERROR   4         , CHECK OUT OPT,ONAL CLR JUMPER
1522 004312 000240              NOP
1523
1524      , , THIS TEST PERFORMS MASTER RESET TESTING &
1525      , , TESTING OF READ/WRITE BIT SYN SCH
1526
1527      , , *****
1528 004314 000004      TST12  SCOPE
1529 004316 052777 000020 175366  BIS    #SYNSCH, @RXCSR , SET THIS BIT
1530 004324 032777 000020 175360  BIT    #SYNSCH, @RXCSR , TEST THIS BIT
1531 004332 001001              BNE     +4         , BR IF "1"
1532 004334 104004              ERROR   4         , THIS BIT SHOULD BE SET
1533 004336 042777 000020 175346  BIC    #SYNSCH, @RXCSR , CLR THIS BIT
1534 004344 032777 000020 175340  BIT    #SYNSCH, @RXCSR , TEST THIS BIT
1535 004352 001401              BEQ     +4         , BR IF "0"
1536 004354 104004              ERROR   4         , THIS BIT SHOULD BE CLP
1537      , NOW SET THIS BIT
1538 004356 052777 000020 175326  BIS    #SYNSCH, @RXCSR
1539 004364 052777 000400 175334  BIS    #MRESET, @TXCSR , MASTER RESET
1540 004372 032777 000020 175312  BIT    #SYNSCH, @RXCSR , TEST THIS BIT
1541 004400 001401              BEQ     +4         , BR IF "0"
1542 004402 104004              ERROR   4         , CHECK OUT MASTER RESET LOGIC
1543
1544      , , THIS TEST PERFORMS MASTER RESET TESTING &
1545      , , TESTING OF READ/WRITE BIT DSINTE
1546
1547      , , *****
1548 004404 000004      TST13  SCOPE
1549 004406 052777 000040 175276  BIS    #DSINTE, @RXCSR , SET THIS BIT
1550 004414 032777 000040 175270  BIT    #DSINTE, @RXCSR , TEST THIS BIT
1551 004422 001001              BNE     +4         , BR IF "1"
1552 004424 104004              ERROR   4         , THIS BIT SHOULD BE SET
1553 004426 042777 000040 175256  BIC    #DSINTE, @RXCSR , CLR THIS BIT
1554 004434 032777 000040 175250  BIT    #DSINTE, @RXCSR , TEST THIS BIT
1555 004442 001401              BEQ     +4         , BR IF "0"
1556 004444 104004              ERROR   4         , THIS BIT SHOULD BE CLP
1557      , NOW SET THIS BIT
1558 004446 052777 000040 175236  BIS    #DSINTE, @RXCSR
1559 004454 052777 000400 175244  BIS    #MRESET, @TXCSR , MASTER RESET
1560 004462 032777 000040 175222  BIT    #DSINTE, @RXCSR , TEST THIS BIT
1561 004470 001401              BEQ     +4         , BR IF "0"
1562 004472 104004              ERROR   4         , CHECK OUT MASTER RESET LOGIC
1563
1564      , , THIS TEST PERFORMS MASTER RESET TESTING &
1565      , , TESTING OF READ/WRITE BIT RINTEN
1566
1567      , , *****
```

```

1568 004474 000004          TST14 SCOPE
1569 004476 052777 000100 175206 BIS      #RINTEN,@RXCSR ;SET THIS BIT
1570 004504 032777 000100 175200 BIT      #RINTEN,@RXCSR ;TEST THIS BIT
1571 004512 001001          BNE      +4      ;BR IF "1"
1572 004514 104004          ERROR    4          ;THIS BIT SHOULD BE SET
1573 004516 042777 000100 175166 BIC      #RINTEN,@RXCSR ;CLR THIS BIT
1574 004524 032777 000100 175160 BIT      #RINTEN,@RXCSR ;TEST THIS BIT
1575 004532 001401          BEQ      +4      ;BR IF "0"
1576 004534 104004          ERROR    4          ;THIS BIT SHOULD BE CLR
1577                                ;NOW SET THIS BIT
1578 004536 052777 000100 175146 BIS      #RINTEN,@RXCSR
1579 004544 052777 000400 175154 BIS      #MRESET,@TXCSR ;MASTER RESET
1580 004552 032777 000100 175132 BIT      #RINTEN,@RXCSR ;TEST THIS BIT
1581 004560 001401          BEQ      +4      ;BR IF "0"
1582 004562 104004          ERROR    4          ;CHECK OUT MASTER RESET LOG C
1583
1584                                ;THIS TEST PERFORMS MASTER RESET TESTING &
1585                                ;TESTING OF READ/WRITE BIT STPSYN
1586
1587                                ;*****
1588 004564 000004          TST15 SCOPE
1589 004566 052777 000400 175116 BIS      #STPSYN,@RXCSR ;SET THIS BIT
1590 004574 032777 000400 175110 BIT      #STPSYN,@RXCSR ;TEST THIS BIT
1591 004602 001001          BNE      +4      ;BR IF "1"
1592 004604 104004          ERROR    4          ;THIS BIT SHOULD BE SET
1593 004606 042777 000400 175076 BIC      #STPSYN,@RXCSR ;CLR THIS BIT
1594 004614 032777 000400 175070 BIT      #STPSYN,@RXCSR ;TEST THIS BIT
1595 004622 001401          BEQ      +4      ;BR IF "0"
1596 004624 104004          ERROR    4          ;THIS BIT SHOULD BE CLR
1597                                ;NOW SET THIS BIT
1598 004626 052777 000400 175056 BIS      #STPSYN,@RXCSR
1599 004634 052777 000400 175064 BIS      #MRESET,@TXCSR ;MASTER RESET
1600 004642 032777 000400 175042 BIT      #STPSYN,@RXCSR ;TEST THIS BIT
1601 004650 001401          BEQ      +4      ;BR IF "0"
1602 004652 104004          ERROR    4          ;CHECK OUT MASTER RESET LOGIC
1603
1604                                ;THIS TEST PERFORMS MASTER RESET TESTING &
1605                                ;TESTING OF READ/WRITE BIT BREAK
1606
1607                                ;*****
1608 004654 000004          TST16 SCOPE
1609 004656 052777 000001 175042 BIS      #BREAK,@TXCSR ;SET THIS BIT
1610 004664 032777 000001 175034 BIT      #BREAK,@TXCSR ;TEST THIS BIT
1611 004672 001001          BNE      +4      ;BR IF "1"
1612 004674 104004          ERROR    4          ;THIS BIT SHOULD BE SET
1613 004676 042777 000001 175022 BIC      #BREAK,@TXCSR ;CLR THIS BIT
1614 004704 032777 000001 175014 BIT      #BREAK,@TXCSR ;TEST THIS BIT
1615 004712 001401          BEQ      +4      ;BR IF "0"
1616 004714 104004          ERROR    4          ;THIS BIT SHOULD BE CLR
1617                                ;NOW SET THIS BIT
1618 004716 052777 000001 175002 BIS      #BREAK,@TXCSR
1619 004724 052777 000400 174774 BIS      #MRESET,@TXCSR ;MASTER RESET
1620 004732 032777 000001 174766 BIT      #BREAK,@TXCSR ;TEST THIS BIT
1621 004740 001401          BEQ      +4      ;BR IF "0"
1622 004742 104004          ERROR    4          ;CHECK OUT MASTER RESET LOGIC
1623
    
```

```
1624                                     ,, THIS TEST PERFORMS MASTER RESET TESTING &
1625                                     ,, TESTING OF READ/WRITE BIT HDXEN
1626                                     ,,
1627                                     ,, *****
1628 004744 000004 TST17 SCOPE
1629 004746 052777 000010 174752 BIS #HDXEN,@TXCSR ,SET THIS BIT
1630 004754 032777 000010 174744 BIT #HDXEN,@TXCSR ,TEST THIS BIT
1631 004762 001001 BNE +4 ,BR IF "1"
1632 004764 104004 ERROR 4 ,THIS BIT SHOULD BE SET
1633 004766 042777 000010 174732 BIC #HDXEN,@TXCSR ,CLR THIS BIT
1634 004774 032777 000010 174724 BIT #HDXEN,@TXCSR ,TEST THIS BIT
1635 005002 001401 BEQ +4 ,BR IF "0"
1636 005004 104004 ERROR 4 ,THIS BIT SHOULD BE CLR
1637 ,NOW SET THIS BIT
1638 005006 052777 000010 174712 BIS #HDXEN,@TXCSR
1639 005014 052777 000400 174704 BIS #MRESET,@TXCSR ,MASTER RESET
1640 005022 032777 000010 174676 BIT #HDXEN,@TXCSR ,TEST THIS BIT
1641 005030 001401 BEQ +4 ,BR IF "0"
1642 005032 104004 ERROR 4 ,CHECK OUT MASTER RESET LOGIC
1643
1644                                     ,, THIS TEST PERFORMS MASTER RESET TESTING &
1645                                     ,, TESTING OF READ/WRITE BIT SEND
1646                                     ,,
1647                                     ,, *****
1648 005034 000004 TST20 SCOPE
1649 005036 052777 000020 174662 BIS #SEND,@TXCSR ,SET THIS BIT
1650 005044 032777 000020 174654 BIT #SEND,@TXCSR ,TEST THIS BIT
1651 005052 001001 BNE +4 ,BR IF "1"
1652 005054 104004 ERROR 4 ,THIS BIT SHOULD BE SET
1653 005056 042777 000020 174642 BIC #SEND,@TXCSR ,CLR THIS BIT
1654 005064 032777 000020 174634 BIT #SEND,@TXCSR ,TEST THIS BIT
1655 005072 001401 BEQ +4 ,BR IF "0"
1656 005074 104004 ERROR 4 ,THIS BIT SHOULD BE CLR
1657 ,NOW SET THIS BIT
1658 005076 052777 000020 174622 BIS #SEND,@TXCSR
1659 005104 052777 000400 174614 BIS #MRESET,@TXCSR ,MASTER RESET
1660 005112 032777 000020 174606 BIT #SEND,@TXCSR ,TEST THIS BIT
1661 005120 001401 BEQ +4 ,BR IF "0"
1662 005122 104004 ERROR 4 ,CHECK OUT MASTER RESET LOGIC
1663
1664                                     ,, THIS TEST PERFORMS MASTER RESET TESTING &
1665                                     ,, TESTING OF READ/WRITE BIT DNAINTE
1666                                     ,,
1667                                     ,, *****
1668 005124 000004 TST21 SCOPE
1669 005126 052777 000040 174572 BIS #DNAINTE,@TXCSR ,SET THIS BIT
1670 005134 032777 000040 174564 BIT #DNAINTE,@TXCSR ,TEST THIS BIT
1671 005142 001001 BNE +4 ,BR IF "1"
1672 005144 104004 ERROR 4 ,THIS BIT SHOULD BE SET
1673 005146 042777 000040 174552 BIC #DNAINTE,@TXCSR ,CLR THIS BIT
1674 005154 032777 000040 174544 BIT #DNAINTE,@TXCSR ,TEST THIS BIT
1675 005162 001401 BEQ +4 ,BR IF "0"
1676 005164 104004 ERROR 4 ,THIS BIT SHOULD BE CLR
1677 ,NOW SET THIS BIT
1678 005166 052777 000040 174532 BIS #DNAINTE,@TXCSR
1679 005174 052777 000400 174524 BIS #MRESET,@TXCSR ,MASTER RESET
```

INITIALIZE THE COMMON TAGS

```
1680 005202 032777 000040 174516 BIT #DNAINTE,@TXCSR ,TEST THIS BIT
1681 005210 001401 BEQ +4 ,BR IF "0"
1682 005212 104004 ERROR 4 ,CHECK OUT MASTER RESET LOGIC
1683
1684 ,, THIS TEST PERFORMS MASTER RESET TESTING &
1685 ,, TESTING OF READ/WRITE BIT TXINTE
1686
1687 ,, *****
1688 005214 000004 TST22 SCOPE
1689 005216 052777 000100 174502 BIS #TXINTE,@TXCSR ,SET THIS BIT
1690 005224 032777 000100 174474 BIT #TXINTE,@TXCSR ,TEST THIS BIT
1691 005232 001001 BNE +4 ,BR IF "1"
1692 005234 104004 ERROR 4 ,THIS BIT SHOULD BE SET
1693 005236 042777 000100 174462 BIC #TXINTE,@TXCSR ,CLR THIS BIT
1694 005244 032777 000100 174454 BIT #TXINTE,@TXCSR ,TEST THIS BIT
1695 005252 001401 BEQ +4 ,BR IF "0"
1696 005254 104004 ERROR 4 ,THIS BIT SHOULD BE CLR
1697 ,NOW SET THIS BIT
1698 005256 052777 000100 174442 BIS #TXINTE,@TXCSR
1699 005264 052777 000400 174434 BIS #MRESET,@TXCSR ,MASTER RESET
1700 005272 032777 000100 174426 BIT #TXINTE,@TXCSR ,TEST THIS BIT
1701 005300 001401 BEQ +4 ,BR IF "0"
1702 005302 104004 ERROR 4 ,CHECK OUT MASTER RESET LOGIC
1703
1704 ,, TEST MAINT MODE BIT 0
1705
1706 ,, THIS TEST PERFORMS MASTER RESET TESTING &
1707 ,, TESTING OF READ/WRITE BIT BIT11
1708
1709 ,, *****
1710 005304 000004 TST23 SCOPE
1711 005306 052777 004000 174412 BIS #BIT11,@TXCSR ,SET THIS BIT
1712 005314 032777 004000 174404 BIT #BIT11,@TXCSR ,TEST THIS BIT
1713 005322 001001 BNE +4 ,BR IF "1"
1714 005324 104004 ERROR 4 ,THIS BIT SHOULD BE SET
1715 005326 042777 004000 174372 BIC #BIT11,@TXCSR ,CLR THIS BIT
1716 005334 032777 004000 174364 BIT #BIT11,@TXCSR ,TEST THIS BIT
1717 005342 001401 BEQ +4 ,BR IF "0"
1718 005344 104004 ERROR 4 ,THIS BIT SHOULD BE CLR
1719 ,NOW SET THIS BIT
1720 005346 052777 004000 174352 BIS #BIT11,@TXCSR
1721 005354 052777 000400 174344 BIS #MRESET,@TXCSR ,MASTER RESET
1722 005362 032777 004000 174336 BIT #BIT11,@TXCSR ,TEST THIS BIT
1723 005370 001401 BEQ +4 ,BR IF "0"
1724 005372 104004 ERROR 4 ,CHECK OUT MASTER PESET LOGIC
1725
1726 ,, TEST MAINT MODE BIT 1
1727
1728 ,, THIS TEST PERFORMS MASTER RESET TESTING &
1729 ,, TESTING OF READ/WRITE BIT BIT12
1730
1731 ,, *****
1732 005374 000004 TST24 SCOPE
1733 005376 052777 010000 174322 BIS #BIT12,@TXCSR ,SET THIS BIT
1734 005404 032777 010000 174314 BIT #BIT12,@TXCSR ,TEST THIS BIT
1735 005412 001001 BNE +4 ,BR IF "1"
```

INITIALIZE THE COMMON TAGS

```
1736 005414 104004 ERROR 4 , THIS BIT SHOULD BE SET
1737 005416 042777 010000 174302 BIC #BIT12,@TXCSR , CLR THIS BIT
1738 005424 032777 010000 174274 BIT #BIT12,@TXCSR , TEST THIS BIT
1739 005432 001401 BEQ +4 , BR IF "0"
1740 005434 104004 ERROR 4 , THIS BIT SHOULD BE CLR
1741 , NOW SET THIS BIT
1742 005436 052777 010000 174262 BIS #BIT12,@TXCSR
1743 005444 052777 000400 174254 BIS #MRESET,@TXCSR , MASTER RESET
1744 005452 032777 010000 174246 BIT #BIT12,@TXCSR , TEST THIS BIT
1745 005460 001401 BEQ +4 , BR IF "0"
1746 005462 104004 ERROR 4 , CHECK OUT MASTER RESET LOGIC
1747
1748 , , THIS TEST PERFORMS MASTER RESET TESTING &
1749 , , TESTING OF READ/WRITE BIT CLK
1750
1751 , , *****
1752 005464 000004 TST25 SCOPE
1753 005466 052777 020000 174232 BIS #CLK,@TXCSR , SET THIS BIT
1754 005474 032777 020000 174224 BIT #CLK,@TXCSR , TEST THIS BIT
1755 005502 001001 BNE +4 , BR IF "1"
1756 005504 104004 ERROR 4 , THIS BIT SHOULD BE SET
1757 005506 042777 020000 174212 BIC #CLK,@TXCSR , CLR THIS BIT
1758 005514 032777 020000 174204 BIT #CLK,@TXCSR , TEST THIS BIT
1759 005522 001401 BEQ +4 , BR IF "0"
1760 005524 104004 ERROR 4 , THIS BIT SHOULD BE CLR
1761 , NOW SET THIS BIT
1762 005526 052777 020000 174172 BIS #CLK,@TXCSR
1763 005534 052777 000400 174164 BIS #MRESET,@TXCSR , MASTER RESET
1764 005542 032777 020000 174156 BIT #CLK,@TXCSR , TEST THIS BIT
1765 005550 001401 BEQ +4 , BR IF "0"
1766 005552 104004 ERROR 4 , CHECK OUT MASTER PESET LOGIC
1767
1768 , , THIS TEST PERFORMS MASTER RESET TESTING &
1769 , , TESTING OF READ/WRITE BIT MTDATA
1770
1771 , , *****
1772 005554 000004 TST26 SCOPE
1773 005556 052777 040000 174142 BIS #MTDATA,@TXCSR , SET THIS BIT
1774 005564 032777 040000 174134 BIT #MTDATA,@TXCSR , TEST THIS BIT
1775 005572 001001 BNE +4 , BR IF "1"
1776 005574 104004 ERROR 4 , THIS BIT SHOULD BE SET
1777 005576 042777 040000 174122 BIC #MTDATA,@TXCSR , CLR THIS BIT
1778 005604 032777 040000 174114 BIT #MTDATA,@TXCSR , TEST THIS BIT
1779 005612 001401 BEQ +4 , BR IF "0"
1780 005614 104004 ERROR 4 , THIS BIT SHOULD BE CLR
1781 , NOW SET THIS BIT
1782 005616 052777 040000 174102 BIS #MTDATA,@TXCSR
1783 005624 052777 000400 174074 BIS #MRESET,@TXCSR , MASTER RESET
1784 005632 032777 040000 174066 BIT #MTDATA,@TXCSR , TEST THIS BIT
1785 005640 001401 BEQ +4 , BR IF "0"
1786 005642 104004 ERROR 4 , CHECK OUT MASTER PESET LOGIC
1787
1788 , , THIS TEST VERIFYS THAT INIT (RESET) CLEARS BITS IN THE
1789 , , RXCSR & TXCSR
1790
1791 , , *****
```

INITIALIZE THE COMMON TAGS

```
1792 005644 000004 TST27 SCOPE
1793 005646 012777 177777 174036 MOV #177777,@RXCSR ;SET ALL POSSIBLE BITS
1794 005654 012777 177777 174044 MOV #177777,@TXCSR ;DITTO
1795 005662 000005 RESET
1796 005664 106427 000340 MTPS #340 ,RESTORE NON INTERRUPT STATUS
1797 005670 017701 174016 MOV @RXCSR,R1 ,SAVE
1798 005674 017702 174026 MOV @TXCSR,R2 ;SAVE
1799 005700 105767 173245 TSTB OPTCLR ,IS THE OPTIONAL CLR JUMPER ON ?
1800 005704 100402 BMI 15 ;YES
1801 005706 042701 000015 BIC #16,R1 ;CLR THE NON RESETABLE BITS
1802 005712 042701 073000 15 BIC #073000,R1 ,CLR ALL NON-CLEARABLE BITS
1803 005716 005701 TST R1 ,ARE THEY ALL 0 ?
1804 005720 001401 BEQ ,+4
1805 005722 104004 ERROR 4 ,ALL SPECIFIED BITS SHOULD BE CLEAR
1806 005724 042702 002200 BIC #002200,R2 ,CLEAR ALL NON-CLEARABLE BITS
1807 005730 005702 TST R2 ,ARE THEY ALL 0 ?
1808 005732 001401 BEQ +4
1809 005734 104004 ERROR 4 ,ALL SPECIFIED BITS SHOULD BE CLEAR
1810 ,WAIT FOR CABLE DELAYS
1811 ,*****
1812 ,MODIFY "HOLD " ACCORDINGLY FOR FASTER OR SLOWER MACHINE
1813 ,*****
1814 005736 016702 173156 MOV HOLD,R2 ,SET DELAY TIME
1815 005742 005302 DEC R2
1816 005744 001376 BNE -2 ,WAIT THIS TIME
1817 ,OK NOW FALL THRU AND CONTINUE TESTING
1818 ,EXIT STAGE LEFT CHINNG'
1819
1820 ,,THIS TEST PERFORMS MASTER PESET TESTING &
1821 ,,TESTING OF WRITE ONLY BIT MRESET
1822 ,,
1823 ,,*****
1824 005746 000004 TST30 SCOPE
1825 005750 052777 000400 173750 BIS #MRESET,@TXCSR ,TRY TO SET THIS BIT
1826 005756 032777 000400 173742 BIT #MRESET,@TXCSR ,TEST THIS BIT
1827 005764 001401 BEQ +4 ,BR IF "0"
1828 005766 104004 ERROR 4 ,THIS BIT SHOULD NOT BE SET
1829 005770 052777 000400 173730 BIS #MRESET,@TXCSR ,MASTER RESET
1830 005776 032777 000400 173722 BIT #MRESET,@TXCSR ,TEST THIS BIT
1831 006004 001401 BEQ +4 ,BR IF "0"
1832 006006 104004 ERROR 4 ,THIS BIT SHOULD NOT BE SET
1833 ,CHECK MASTER RESET LOGIC
1834
1835 ,,THIS TEST VERIFYS THAT THE RXCSR & TXCSR CAN BE BYTE ADDRESSED (DATOB)
1836 ,,
1837 ,,*****
1838 006010 000004 TST31 SCOPE
1839 006012 052777 000400 173706 BIS #MRESET,@TXCSR ,MASTER RESET
1840 006020 105767 173125 TSTB OPTCLR ,IS THE OPTIONAL CLR JUMPER ON ?
1841 006024 100405 BMI 15 ,YES
1842 006026 012777 000000 173656 MOV #0,@RXCSR ,CLR OUT NON RESETABLE BITS
1843 006034 005777 173652 TST @RXCSR ,CLR OUT DSC BY READING RXCSR
1844 006040 152777 000001 173646 15 BISB #010,@RXCSR ,SET STRIP SYNC UPPER BYTE
1845 006046 017701 173640 MOV @RXCSR,R1 ,SAVE RXCSR
1846 006052 022701 000400 CMP #400,R1 ,TEST RXCSR
1847 006056 001401 BEQ +4
```

INITIALIZE THE COMMON TAGS

```
1848 006060 104004 ERROR 4 , ONLY STRIP SYNC SHOULD BE SET
1849 006062 105077 173624 CLRB @RXCSR , CLR LOWER BYTE
1850 006066 017701 173620 MOV @RXCSR, R1 , SAVE RXCSR
1851 006072 022701 000400 CMP #400, R1 ; TEST RXCSR
1852 006076 001401 BEQ +4
1853 006100 104004 ERROR 4 ; ONLY STRIP SYNC SHOULD BE SET
1854 006102 052777 000400 173616 BIS #MRESET, @TXCSR , MASTER RESET
1855 006110 152777 000040 173612 BISB #BITS, @TXCSR ; SET MAINT CLK UPPER BYTE
1856 006116 017701 173604 MOV @TXCSR, R1 ; SAVE TXCSR
1857 006122 042701 002000 BIC #BITW, R1 , CLR BIT WINDOW (DEPENDENT
1858 , ON H315 CONNECTOR EXISTANCE)
1859 006126 022701 020200 CMP #20200, R1 ; TEST TXCSR
1860 006132 001401 BEQ +4
1861 006134 104004 ERROR 4 , ONLY MAINT CLK BIT & TXDONE SHOULD BE SET
1862 006136 105077 173564 CLRB @TXCSR , CLR LOWER BYTE
1863 006142 017701 173560 MOV @TXCSR, R1 , SAVE TXCSR
1864 006146 042701 002000 BIC #BITW, R1 , CLR BIT WINDOW (DITTO)
1865 006152 022701 020200 CMP #20200, R1 , TEST TXCSR
1866 006156 001401 BEQ +4
1867 006160 104004 ERROR 4 , ONLY MAINT CLK BIT & TXDONE SHOULD BE SET
1868 , THIS TEST PERFORMS MASTER RESET TESTING &
1869 , TESTING OF READ ONLY BIT BITW
1870 , MAINT INTERNAL
1871
1872
1873 006162 000004 , , *****
TST32 SCOPE
1874 006164 012777 044001 173534 MOV #MINT!MCDATA!BREAK, @TXCSR , SET MAINT INT , BREAK,
1875 , &MCDATA
1876 006172 032777 002000 173526 BIT #BITW, @TXCSR , TEST BITW
1877 006200 001001 BNE +4
1878 006202 104004 ERROR 4 , BIT WINDOW SHOULD BE SET
1879 006204 042777 040000 173514 BIC #MCDATA, @TXCSR
1880 006212 013702 001132 MOV @#1132, R2
1881 006216 005302 15 DEC R2
1882 006220 001376 BNE 15
1883 006222 032777 002000 173476 BIT #BITW, @TXCSR
1884 006230 001401 BEQ +4
1885 006232 104004 ERROR 4 , BIT SHOULD BE CLR
1886 , NOW SET THE MCDATA
1887 006234 052777 040000 173464 BIS #MCDATA, @TXCSR
1888 006242 052777 000400 173456 BIS #MRESET, @TXCSR , MASTER RESET
1889 006250 052777 004001 173450 BIS #MINT!BREAK, @TXCSR
1890 006256 013702 001132 MOV @#1132, R2
1891 006262 005302 25 DEC R2
1892 006264 001376 BNE 25
1893 006266 032777 002000 173432 BIT #BITW, @TXCSR
1894 006274 001401 BEQ +4
1895 006276 104004 ERROR 4 , BITW SHOULD BE CLR BY MASTER RESET
1896 , THIS TEST PERFORMS MASTER RESET TESTING &
1897 , TESTING OF READ ONLY BIT BITW
1898 , MAINT EXTERNAL
1899
1900 , , *****
1901 006300 000004 TST33 SCOPE
1902 , TEST TO SEE IF EXTERNAL MODEM BYPASS CONNECTOR
1903 , IS ON (H315) IF "NO" JUMP AROUND TEST
```



```
1904 006302 105767 172645 TSTB JMRBY
1905 006306 100036 BPL 15 , IT IS NOT ON
1906 006310 012777 050001 173410 MOV #MEXT!MCDATA!BREAK,@TXCSR , SET MAINT EXT , BREAK,
1907 , &MCDATA
1908 006316 032777 002000 173402 BIT #BITW,@TXCSR ; TEST BITW
1909 006324 001001 BNE +4
1910 006326 104004 ERROR 4 , BIT WINDOW SHOULD BE SET
1911 006330 042777 040000 173370 BIC #MCDATA,@TXCSR
1912 006336 032777 002000 173362 BIT #BITW,@TXCSR
1913 006344 001401 BEQ +4
1914 006346 104004 ERROR 4 , BIT SHOULD BE CLR
1915 , NOW SET THE MCDATA
1916 006350 052777 040000 173350 BIS #MCDATA,@TXCSR
1917 006356 052777 000400 173342 BIS #MRESET,@TXCSR , MASTER RESET
1918 006364 052777 010001 173334 BIS #MEXT!BREAK,@TXCSR
1919 006372 032777 002000 173326 BIT #BITW,@TXCSR
1920 006400 001401 BEQ +4
1921 006402 104004 ERROR 4 , BITW SHOULD BE CLR BY MASTER RESET
1922 006404
1923 15
1924
1925 , , THIS TEST PERFORMS MASTER RESET TESTING &
1926 , , TESTING OF READ ONLY BIT RXDONE
1927 , ,
1928 , , *****
1929 006404 000004 TST34 SCOPE
1930 006406 052777 000400 173312 BIS #MRESET,@TXCSR , MASTER RESET
1931 006414 032777 000200 173270 BIT #RXDONE,@RXCSR , TEST THIS BIT
1932 006422 001401 BEQ +4 , BR IF "0"
1933 006424 104004 ERROR 4 , CHECK MASTER RESET LOGIC
1934 , OR SHORT ON THIS BIT
1935
1936 , , THIS TEST PERFORMS MASTER RESET TESTING &
1937 , , TESTING OF READ ONLY BIT REACT
1938 , ,
1939 , , *****
1940 006426 000004 TST35 SCOPE
1941 006430 052777 000400 173270 BIS #MRESET,@TXCSR , MASTER RESET
1942 006436 032777 004000 173246 BIT #REACT,@RXCSR , TEST THIS BIT
1943 006444 001401 BEQ +4 , BR IF "0"
1944 006446 104004 ERROR 4 , CHECK MASTER RESET LOGIC
1945 , OR SHORT ON THIS BIT
1946
1947 , , THIS TEST PERFORMS MASTER RESET TESTING &
1948 , , TESTING OF READ ONLY BIT DSC
1949 , ,
1950 , , *****
1951 006450 000004 TST36 SCOPE
1952 006452 052777 000400 173246 BIS #MRESET,@TXCSR , MASTER RESET
1953 006460 032777 100000 173224 BIT #DSC,@RXCSR , TEST THIS BIT
1954 006466 001401 BEQ +4 , BR IF "0"
1955 006470 104004 ERROR 4 , CHECK MASTER RESET LOGIC
1956 , OR SHORT ON THIS BIT
1957
1958 , , THIS TEST PERFORMS MASTER RESET TESTING &
1959 , , TESTING OF READ ONLY BIT TXDONE
```

```
1960
1961
1962 006472 000004
1963 006474 052777 000400 173224
1964 006502 032777 000200 173216
1965 006510 001001
1966 006512 104004
1967
1968
1969
1970
1971
1972 006514 000004
1973 006516 052777 000400 173202
1974 006524 032777 100000 173174
1975 006532 001401
1976 006534 104004
1977
1978
1979
1980
1981
1982
1983 006536 000004
1984 006540 052777 000400 173160
1985 006546 016703 173144
1986 006552 012700 000377
1987 006556 017701 173134
1988 006562 120001
1989 006564 001401
1990 006566 104002
1991
1992
1993
1994
1995 006570 000004
1996 006572 052777 000400 173126
1997 006600 032777 010000 173110
1998 006606 001401
1999 006610 104004
2000
2001
2002
2003
2004
2005
2006 006612 000004
2007 006614 052777 000400 173104
2008 006622 032777 020000 173066
2009 006630 001401
2010 006632 104004
2011
2012
2013
2014
2015

,, *****
TST37 SCOPE
BIS #MRESET,@TXCSR ,MASTER RESET
BIT #TXDONE,@TXCSR ,TEST THIS BIT
BNE +4 ,BR IF "1"
ERROR 4 ,CHECK MASTER RESET LOGIC
,OR SHORT ON THIS BIT
,, THIS TEST PERFORMS MASTER RESET TESTING &
,, TESTING OF READ ONLY BIT DNA
,, *****
TST40 SCOPE
BIS #MRESET,@TXCSR ,MASTER RESET
BIT #DNA,@TXCSR ,TEST THIS BIT
BEQ +4 ,BR IF "0"
ERROR 4 ,CHECK MASTER RESET LOGIC
,OR SHORT ON THIS BIT
,, THIS TEST PERFORMS MASTER RESET TESTING &
,, TESTING OF READ ONLY WORD RECEIVE DATA
,, *****
TST41 SCOPE
BIS #MRESET,@TXCSR ,MASTER RESET
MOV RXDBUF,R3 ,FOR ERROR MESSAGE
MOV #377,R0 ,EXPECTED
MOV @RXDBUF,R1 ,ACTUAL
CMPB R0,R1
BEQ +4 ,BR IF "0"
ERROR 2 ,REC DATA SHOULD BE ALL 1'S
,, THIS TEST PERFORMS MASTER RESET TESTING &
,, TESTING OF READ ONLY BIT PARER
,, *****
TST42 SCOPE
BIS #MRESET,@TXCSR ,MASTER RESET
BIT #PARER,@RXDBUF ,TEST THIS BIT
BEQ +4 ,BR IF "0"
ERROR 4 ,CHECK MASTER RESET LOGIC
,OR SHORT ON THIS BIT
,, THIS TEST PERFORMS MASTER RESET TESTING &
,, TESTING OF READ ONLY BIT FMERR
,, *****
TST43 SCOPE
BIS #MRESET,@TXCSR ,MASTER RESET
BIT #FMERR,@RXDBUF ,TEST THIS BIT
BEQ +4 ,BR IF "0"
ERROR 4 ,CHECK MASTER RESET LOGIC
,OR SHORT ON THIS BIT
,, THIS TEST PERFORMS MASTER RESET TESTING &
,, TESTING OF READ ONLY BIT OVPPUN
,,
```

```
2016  
2017 006634 000004  
2018 006636 052777 000400 173062  
2019 006644 032777 040000 173044  
2020 006652 001401  
2021 006654 104004  
2022  
2023  
2024  
2025  
2026  
2027  
2028 006656 000004  
2029 006660 052777 000400 173040  
2030 006666 032777 100000 173022  
2031 006674 001401  
2032 006676 104004  
2033  
2034  
2035  
2036  
2037  
2038 006700 000004  
2039 006702 012777 177777 173002  
2040 006710 052777 000400 173010  
2041 006716 016703 172770  
2042 006722 017701 172764  
2043 006726 105767 172217  
2044 006732 100010  
2045 006734 042701 173000  
2046  
2047 006740 012700 000000  
2048 006744 020001  
2049 006746 001401  
2050 006750 104001  
2051 006752 000407  
2052 006754 042701 173000  
2053  
2054 006760 012700 000016  
2055 006764 020001  
2056 006766 001401  
2057 006770 104001  
2058  
2059  
2060 006772  
2061  
2062  
2063  
2064  
2065 006772 016702 172122  
2066 006776 005302  
2067 007000 001376  
2068  
2069  
2070  
2071
```

```
*****  
TST44 SCOPE  
BIS #MRESET,@TXCSR ,MASTER RESET  
BIT #OVRUN,@RXDBUF ,TEST THIS BIT  
BEQ +4 ,BR IF "0"  
ERROR 4 ,CHECK MASTER RESET LOGIC  
 ,OR SHORT ON THIS BIT  
  
,, THIS TEST PERFORMS MASTER RESET TESTING &  
,, TESTING OF READ ONLY BIT RXERR  
  
*****  
TST45 SCOPE  
BIS #MRESET,@TXCSR ,MASTER RESET  
BIT #RXERR,@RXDBUF ,TEST THIS BIT  
BEQ +4 ,BR IF "0"  
ERROR 4 ,CHECK MASTER RESET LOGIC  
 ,OR SHORT ON THIS BIT  
  
,, THIS TEST VERIFYS THAT THE DEVICE REGISTER RXCSR  
,, IS CLEARED BY MASTER RESET  
  
*****  
TST46 SCOPE  
MOV #177777,@RXCSR ,SET ALL POSSIBLE BITS  
BIS #MRESET,@TXCSR ,MASTER RESET  
MOV RXCSR,R3 ,FOR ERROR MESSAGE  
MOV @RXCSR,R1 ,SAVE ACTUAL  
TSTB OPTCLR ,TEST THE OPT CLR JUMPER FLAG  
BPL 15 ,NO ,ITS NOT IN  
BIC #173000,R1 ,CLR NON-MASTER RESETTABLE  
 ,BITS(SINCE THESE ARE DEPENDENT ON H315 CONNECTORS EXISTANCE)  
MOV #0,R0 ,EXPECTED  
CMP R0,R1 ,EXPECTED VS ACTUAL  
BEQ +4  
ERROR 1 ,ALL MASTER RESETTABLE BITS SHOULD BE CLR  
BR 25 ,JUMP AROUND  
BIC #173000,R1 ,CLR NON-MASTER RESETTABLE  
 ,BITS(SINCE THESE ARE DEPENDENT ON H315 CONNECTORS EXISTANCE)  
MOV #16,R0 ,EXPECTED  
CMP R0,R1 ,EXPECTED VS ACTUAL  
BEQ +4  
ERROR 1 ,ONLY STD,RTS,DTR BITS SHOULD BE SET  
 ,NOTE THAT STD IS READ =1 INDEPENDENT OF  
 ,SEC XMIT #6 STRAP  
  
25  
 ,WAIT FOR CABLE DELAYS  
*****  
 ,MODIFY "HOLD " ACCORDINGLY FOR FASTER OR SLOWER MACHINE  
*****  
MOV HOLD,R2 ,SET DELAY TIME  
DEC R2  
BNE -2 ,WAIT THIS TIME  
 ,OK NOW FALL THRU AND CONTINUE TESTING  
 ,EXIT STAGE LEFT CHINNG'
```

```
2072                                     ;, THIS TEST VERIFYS THAT THE DEVICE REGISTER TXCSR
2073                                     ;, IS CLEARED BY MASTER RESET
2074                                     ;,
2075                                     ;, *****
2076 007002 000004 TST47 SCOPE
2077 007004 012777 177777 172714 MOV #177777, @TXCSR , SET ALL POSSIBLE BITS
2078 007012 052777 000400 172706 BIS #MRESET, @TXCSR , MASTER RESET
2079 007020 016703 172702 MOV TXCSR, R3 ; FOR ERROR MESSAGE
2080 007024 017701 172676 MOV @TXCSR, R1 , SAVE ACTUAL
2081 007030 012700 000203 MOV #200, R0 , EXPECTED
2082 007034 020001 CMP R0, R1 , EXPECTED VS ACTUAL
2083 007036 001401 BEQ +4
2084 007040 104001 ERROR 1 , ONLY TXDONE SHOULD BE SET
2085
2086                                     ;, THIS TEST VERIFYS THAT THE DEVICE REGISTER RXDBUF
2087                                     ;, IS CLEARED BY MASTER RESET
2088                                     ;,
2089                                     ;, *****
2090 007042 000004 TST50 SCOPE
2091 007044 052777 000400 172654 BIS #MRESET, @TXCSR , MASTER RESET
2092 007052 016703 172640 MOV RXDBUF, R3 , FOR EPROR MESSAGE
2093 007056 017701 172634 MOV @RXDBUF, R1 , SAVE
2094 007062 012700 000377 MOV #377, R0 , EXPECTED
2095 007066 020001 CMP R0, R1 , EXPECTED VS ACTUAL
2096 007070 001401 BEQ +4
2097 007072 104002 ERROR 2 , ONLY REC DATA BITS SHOULD BE SET
2098                                     ;, THIS TEST VERIFYS BITS RING, CTS, CARDET, SRD, DSR
2099                                     ;, ALSO DSC IS GENERATED WHEN ANY OF THESE BITS ARE SET
2100                                     ;, OR CLEARED IT ALSO CHECKS THE MODEM BYPASS
2101                                     ;, JUMPER AND THAT THESE BITS CAN BE READ
2102                                     ;, NOTE THE MODEM BYPASS JUMPER MUST BE ON (H315)
2103                                     ;,
2104                                     ;, *****
2105 007074 000004 TST51 SCOPE
2106 007076 005077 172610 CLR @RXCSR , TO GET RID OF STD , RTS, DTR IF OPTCLR JUMPER #4 IS NOT ON
2107 007102 052777 000400 172616 BIS #MRESET, @TXCSR , MASTER RESET
2108                                     ;, TEST THAT A "YES" ANSWER WAS GIVEN TO QUESTION IN
2109                                     ;, THE MONITOR OR BY DEFAULT
2110                                     ;, THIS TEST WILL BE BYPASSED IF THE EXTERNAL BYPASS
2111                                     ;, JUMPER IS NOT INSTALLED
2112 007110 105767 172037 TSTB JMRBY
2113 007114 100402 BMI +6 , THE ANSWER WAS YES
2114                                     ;, PERFORM THIS TEST
2115 007116 000167 000652 JMP OUT1 , JUMP AROUND THIS TEST IF THE ANSWER
2116                                     ;, WAS NO
2117 007122 016703 172564 MOV RXCSR, R3 , SET UP FOR ERROR MESSAGE
2118 007126 017701 172560 MOV @RXCSR, R1 , ACTUAL
2119 007132 005000 CLR R0 , EXPECTED
2120 007134 005701 TST R1 , IS IT = 0 ?
2121 007136 001401 BEQ +4
2122 007140 104001 ERROR 1 , RXCSR SHOULD BE CLR
2123 007142 052777 000002 172542 BIS #DTR, @RXCSR , SET DTR
2124                                     ;, WAIT FOR CABLE DELAYS
2125                                     ;, *****
2126                                     ;, MODIFY "HOLD " ACCORDINGLY FOR FASTER OP SLOWER MACHINE
2127                                     ;, *****
```

```

2128 007150 016702 171744      MOV    HOLD,R2 ,SET DELAY TIME
2129 007154 005302              DEC    R2
2130 007156 001376              BNE    -2      ,WAIT THIS TIME
2131                          ,OK NOW FALL THRU AND CONTINUE TESTING. . . . .
2132                          ,EXIT STAGE LEFT  CHINNG!
2133 007160 017701 172526      MOV    @RXCSR,R1      ,ACTUAL
2134 007164 012700 130002      MOV    #130002,RO     ,DSC,CTS,CARDET,DTR
2135 007170 020001              CMP    RO,R1      ,EXPECTED VS ACTUAL
2136 007172 001401              BEQ    +4
2137 007174 104001              ERROR  1      ,CHECK BYPASS CONNECTOR
2138 007176 017701 172510      MOV    @RXCSR,R1      ,ACTUAL
2139 007202 012700 030002      MOV    #30002,RO     ,CTS,CARDET,DTR
2140 007206 020001              CMP    RO,R1      ,EXPECTED VS ACTUAL
2141 007210 001401              BEQ    +4
2142 007212 104001              ERROR  1      ,PREVIOUS READING OF RXCSR SHOULD
2143                          ,HAVE CLEARED DSC
2144 007214 052777 000004 172470  BIS    #RTS,@RXCSR
2145                          ,WAIT FOR CABLE DELAYS
2146                          ,*****
2147                          ,MODIFY "HOLD." ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2148                          ,*****
2149 007222 016702 171672      MOV    HOLD,R2 ,SET DELAY TIME
2150 007226 005302              DEC    R2
2151 007230 001376              BNE    -2      ,WAIT THIS TIME
2152                          ,OK NOW FALL THRU AND CONTINUE TESTING
2153                          ,EXIT STAGE LEFT  CHINNG!
2154 007232 017701 172454      MOV    @RXCSR,R1
2155 007236 012700 170006      MOV    #170006,RO     ,DSC,RING,CTS,CARDET,RTS,DTR
2156 007242 020001              CMP    RO,R1      ,EXPECTED VS ACTUAL
2157 007244 001401              BEQ    +4
2158 007246 104001              ERROR  1      ,CHECK BYPASS CONNECTOR
2159 007250 017701 172436      MOV    @RXCSR,R1
2160 007254 012700 070006      MOV    #70006,RO     ,RING,CTS,CARDET,RTS,DTR
2161 007260 020001              CMP    RO,R1      ,EXPECTED VS ACTUAL
2162 007262 001401              BEQ    +4
2163 007264 104001              ERROR  1      ,PREVIOUS READING OF RXCSR SHOULD
2164                          ,HAVE CLEARED DSC
2165 007266 105767 171655      TSTB  SEXMIT ,IS SEC XMIT JUMPER IN ?
2166 007272 100112              BPL    OUT2      ,NO
2167 007274 105767 171650      TSTB  SEREC  ,IS SEC REC JUMPER IN ?
2168 007300 100163              BPL    OUT3      ,NO
2169 007302 052777 000010 172402  BIS    #STD,@RXCSR
2170                          ,WAIT FOR CABLE DELAYS
2171                          ,*****
2172                          ,MODIFY "HOLD " ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2173                          ,*****
2174 007310 016702 171604      MOV    HOLD,R2 ,SET DELAY TIME
2175 007314 005302              DEC    R2
2176 007316 001376              BNE    -2      ,WAIT THIS TIME
2177                          ,OK NOW FALL THRU AND CONTINUE TESTING
2178                          ,EXIT STAGE LEFT  CHINNG!
2179 007320 017701 172366      MOV    @RXCSR,R1
2180 007324 012700 173016      MASK1 MOV    #173016,RO     ,DSC,RING,CTS,CARDET SRD,DSP
2181                          ,STD,RTS,DTR
2182 007330 020001              CMP    RO,R1      ,EXPECTED VS ACTUAL
2183 007332 001401              BEQ    +4
  
```

INITIALIZE THE COMMON TAGS

2184	007334	104001			ERROR	1	,CHECK BYPASS CONNECTOR
2185	007336	017701	172350		MOV	@RXCSR,R1	
2186	007342	012700	073016	MASK2	MOV	#73016,R0	,RING,CTS,CARDET,SRD,DSR,STD
2187							,RTS,DTR
2188	007346	020001			CMP	R0,R1	,EXPECTED VS ACTUAL
2189	007350	001401			BEQ	+4	
2190	007352	104001			ERROR	1	,PREVIOUS READING OF RXCSR SHOULD
2191							,HAVE CLEARED DSC
2192	007354	042777	000002	172330	BIC	#DTR,@RXCSR	
2193							,WAIT FOR CABLE DELAYS
2194							,*****
2195							,MODIFY "HOLD" ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2196							,*****
2197	007362	016702	171532		MOV	HOLD,R2	,SET DELAY TIME
2198	007366	005302			DEC	R2	
2199	007370	001376			BNE	-2	,WAIT THIS TIME
2200							,OK NOW FALL THRU AND CONTINUE TESTING
2201							,EXIT STAGE LEFT 'CHINNG'
2202	007372	017701	172314		MOV	@RXCSR,R1	
2203	007376	012700	143014		MOV	#143014,R0	,DSC,RING,SRD,DSR,STD PTS
2204	007402	020001			CMP	R0,R1	,EXPECTED VS ACTUAL
2205	007404	001401			BEQ	+4	
2206	007406	104001			ERROR	1	,DSC SHOULD BE SET
2207	007410	042777	000004	172274	BIC	#RTS,@RXCSR	
2208							,WAIT FOR CABLE DELAYS
2209							,*****
2210							,MODIFY "HOLD" ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2211							,*****
2212	007416	016702	171476		MOV	HOLD,R2	,SET DELAY TIME
2213	007422	005302			DEC	R2	
2214	007424	001376			BNE	-2	,WAIT THIS TIME
2215							,OK NOW FALL THRU AND CONTINUE TESTING
2216							,EXIT STAGE LEFT 'CHINNG'
2217	007426	017701	172260		MOV	@RXCSR,R1	
2218	007432	012700	103010	MASK3	MOV	#103010,R0	,DSC,SRD,DSR,STD
2219	007436	020001			CMP	R0,R1	,EXPECTED VS ACTUAL
2220	007440	001401			BEQ	+4	
2221	007442	104001			ERROR	1	,DSC SHOULD BE SET
2222	007444	042777	000010	172240	BIC	#STD,@RXCSR	
2223							,WAIT FOR CABLE DELAYS
2224							,*****
2225							,MODIFY "HOLD" ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2226							,*****
2227	007452	016702	171442		MOV	HOLD,R2	,SET DELAY TIME
2228	007456	005302			DEC	R2	
2229	007460	001376			BNE	-2	,WAIT THIS TIME
2230							,OK NOW FALL THRU AND CONTINUE TESTING
2231							,EXIT STAGE LEFT 'CHINNG'
2232	007462	017701	172224		MOV	@RXCSR,R1	
2233	007466	012700	100000		MOV	#100000,R0	DSC
2234	007472	020001			CMP	R0,R1	,EXPECTED VS ACTUAL
2235	007474	001401			BEQ	+4	
2236	007476	104001			ERROR	1	,DSC SHOULD BE SET
2237	007500	017701	172206		MOV	@RXCSR,R1	
2238	007504	005000			CLR	R0	,NONE
2239	007506	005701			TST	R1	

INITIALIZE THE COMMON TAGS

```

2240 007510 001401      BEQ      +4
2241 007512 104001      ERROR    1      ,DSC SHOULD BE CLEARED FROM PREVIOUS
2242                                ,READING OF RXCSR
2243 007514 000167 000254      JMP      OUT1    ,JUMP AROUND
2244                                ,THE FOLLOWING ROUTINE HANDLES THE SITUATION WHERE SEC XMIT
2245                                ,AND SEC REC JUMPERS ARE NOT ON
2246 007520 052777 000010 172164  OUT2    BIS      #STD,@RXCSR
2247                                ,WAIT FOR CABLE DELAYS
2248                                ,*****
2249                                ,MODIFY "HOLD." ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2250                                ,*****
2251 007526 016702 171366      MOV      HOLD,R2 ,SET DELAY TIME
2252 007532 005302      DEC      R2
2253 007534 001376      BNE     -2      ,WAIT THIS TIME
2254                                ,OK NOW FALL THRU AND CONTINUE TESTING
2255                                ,EXIT STAGE LEFT 'CHINNG'
2256 007536 017701 172150      MOV      @RXCSR,R1 ,ACTUAL
2257 007542 012700 070016      MOV      #70016,R0 ,EXPECTED RING ,CTS,CARDET,STD,RTS,DTR
2258 007546 020001      CMP      R0,R1 ,EXPECTED VS ACTUAL
2259 007550 001401      BEQ      +4
2260 007552 104001      ERROR    1      ,CHECK SEC XMIT & SEC REC JUMPERS
2261 007554 042777 000004 172130      BIC      #RTS,@RXCSR
2262                                ,WAIT FOR CABLE DELAYS
2263                                ,*****
2264                                ,MODIFY "HOLD " ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2265                                ,*****
2266 007562 016702 171332      MOV      HOLD,R2 ,SET DELAY TIME
2267 007566 005302      DEC      R2
2268 007570 001376      BNE     -2      ,WAIT THIS TIME
2269                                ,OK NOW FALL THRU AND CONTINUE TESTING
2270                                ,EXIT STAGE LEFT 'CHINNG'
2271 007572 017701 172114      MOV      @RXCSR,R1 ,ACTUAL
2272 007576 012700 130012      MOV      #130012,R0 ,DSC,CTS,CARDET,DTP,STD
2273                                ,NOTE THAT DSC STILL ASSERTS EVEN THO THE SEC XMIT JUMPER # 6 IS NOT ON
2274 007602 020001      CMP      R0,R1 ,EXPECTED VS ACTUAL
2275 007604 001401      BEQ      +4
2276 007606 104001      ERROR    1      ,CHECK BYPASS CONNECTOR
2277 007610 042777 000002 172074      BIC      #DTR,@RXCSR
2278                                ,WAIT FOR CABLE DELAYS
2279                                ,*****
2280                                ,MODIFY "HOLD " ACCORDINGLY FOR FASTER OF SLOWER MACHINE
2281                                ,*****
2282 007616 016702 171276      MOV      HOLD,R2 ,SET DELAY TIME
2283 007622 005302      DEC      R2
2284 007624 001376      BNE     -2      ,WAIT THIS TIME
2285                                ,OK NOW FALL THRU AND CONTINUE TESTING
2286                                ,EXIT STAGE LEFT 'CHINNG'
2287 007626 017701 172060      MOV      @RXCSR,R1 ,ACTUAL
2288 007632 012700 100010      MOV      #100010,R0 ,DSC,STD
2289 007636 020001      CMP      R0,R1 ,EXPECTED VS ACTUAL
2290 007640 001401      BEQ      +4
2291 007642 104001      ERROR    1      ,ONLY DSC & STD SHOULD BE SET
2292 007644 000167 000124      JMP      OUT1    ,JUMP AROUND
2293 007650 052777 000010 172034  OUT3    BIS      #STD,@RXCSR
2294                                ,WAIT FOR CABLE DELAYS
2295                                ,*****

```

2296					,MODIFY "HOLD " ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2297					,*****
2298	007656	016702	171236		MOV HOLD,R2 ,SET DELAY TIME
2299	007662	005302			DEC R2
2300	007664	001376			BNE -2 ,WAIT THIS TIME
2301					,OK NOW FALL THRU AND CONTINUE TESTING
2302					,EXIT STAGE LEFT CHINNG'
2303	007666	017701	172020		MOV @RXCSR,R1 ,ACTUAL
2304	007672	012700	171016		MOV #171016,R0 ,EXPECTED DSC,RING,CTS,CARDET,DSR,STD,RTS,DTR
2305	007676	020001			CMP R0,R1 ,EXPECTED VS ACTUAL
2306	007700	001401			BEQ +4
2307	007702	104001			ERROR 1 ,CHECK SEC REC JUMPER
2308	007704	042777	000004	172000	BIC #RTS,@RXCSR
2309					,WAIT FOR CABLE DELAYS
2310					,*****
2311					,MODIFY "HOLD " ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2312					,*****
2313	007712	016702	171202		MOV HOLD,R2 ,SET DELAY TIME
2314	007716	005302			DEC R2
2315	007720	001376			BNE -2 ,WAIT THIS TIME
2316					,OK NOW FALL THRU AND CONTINUE TESTING
2317					,EXIT STAGE LEFT CHINNG'
2318	007722	017701	171764		MOV @RXCSR,R1 ,ACTUAL
2319	007726	012700	131012		MOV #131012,R0 ,EXPECTED DSC,CTS,CARDET,DSR,STD,DTR
2320	007732	020001			CMP R0,R1 ,EXPECTED VS ACTUAL
2321	007734	001401			BEQ +4
2322	007736	104001			ERROR 1 ,CHECK H315 CONNECTOR
2323	007740	042777	000002	171744	BIC #DTR,@RXCSR
2324					,WAIT FOR CABLE DELAYS
2325					,*****
2326					,MODIFY "HOLD " ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2327					,*****
2328	007746	016702	171146		MOV HOLD,R2 ,SET DELAY TIME
2329	007752	005302			DEC R2
2330	007754	001376			BNE -2 ,WAIT THIS TIME
2331					,OK NOW FALL THRU AND CONTINUE TESTING
2332					,EXIT STAGE LEFT CHINNG'
2333	007756	017701	171730		MOV @RXCSR,R1 ,ACTUAL
2334	007762	012700	101010		MOV #101010,R0 ,EXPECTED DSC DSR STD
2335	007766	020001			CMP R0,R1 ,EXPECTED VS ACTUAL
2336	007770	001401			BEQ +4
2337	007772	104001			ERROR 1 ,CHECK H315 CONNECTOR
2338	007774				OUT1
2339					
2340					,, THIS TEST VERIFYS THAT REACT (REC ACT VE) ASSEPTS
2341					,, IMMED WHEN SYNC EXTERNAL MODE IS SELECTED
2342					,, AND SYNC SEARCH IS SET
2343					,,
2344					,,*****
2345	007774	000004			TST52 SCOPE
2346	007776	052777	000400	171722	BIS #MRESET,@TXCSR ,MASTER RESET
2347	010004	012777	020000	171710	MOV #SYNEXT,@PARCSR ,SET THE MODE
2348	010012	052777	000400	171706	BIS #MRESET,@TXCSR ,MASTER RESET
2349					
2350					SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2351	010020	012777	064001	171700	MOV #MTDATA'CLK'MINT'BREAK @TXCSR



```

2352
2353      . SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2354 010026 012777 026026 171666      MOV      #SYNEXT'EIGHT'NOPAR'26, @PARCSR
2355 010034 032777 004000 171650      BIT      #RECACT, @RXCSR
2356 010042 001401                      BEQ      +4
2357 010044 104004                      ERROR    4          ; RECACT SHOULD NOT BE SET
2358 010046 052777 000020 171636      BIS      #SYNSCH, @RXCSR ; SET SYNC SEARCH
2359 010054 032777 004000 171630      BIT      #RECACT, @RXCSR
2360 010062 001001                      BNE      +4
2361 010064 104004                      ERROR    4          ; RECACT DID NOT ASSERT
2362 010066 042777 000020 171616      BIC      #SYNSCH, @RXCSR ; DROP SEARCH SYNC
2363 010074 032777 004000 171610      BIT      #RECACT, @RXCSR ; IS IT =0?
2364 010102 001401                      BEQ      +4
2365 010104 104004                      ERROR    4          ; RECACT SHOULD BE 0
2366
2367      ., THIS TEST VERIFYS THAT RECACT (REC ACTIVE) ASSERTS
2368      ., IMMED WHEN ISOCRONOUS MODE IS SELECTED
2369      ., AND SYNC SEARCH IS SET
2370
2371      ., *****
2372 010106 000004                      TSTB3   SCOPE
2373 010110 052777 000400 171610      BIS      #MRESET, @TXCSR ; MASTER RESET
2374 010116 012777 000000 171576      MOV      #ISYMOD, @PARCSR ; SET THE MODE
2375 010124 052777 000400 171574      BIS      #MRESET, @TXCSR ; MASTER RESET
2376
2377      . SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2378 010132 012777 064001 171566      MOV      #MTDATA'CLK'MINT'BREAK, @TXCSR
2379
2380      . SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2381 010140 012777 006026 171554      MOV      #ISYMOD'EIGHT'NOPAR'26, @PARCSR
2382 010146 032777 004000 171536      BIT      #RECACT, @RXCSR
2383 010154 001401                      BEQ      +4
2384 010156 104004                      ERROR    4          ; RECACT SHOULD NOT BE SET
2385 010160 052777 000020 171524      BIS      #SYNSCH, @RXCSR ; SET SYNC SEARCH
2386 010166 032777 004000 171516      BIT      #RECACT, @RXCSR
2387 010174 001001                      BNE      +4
2388 010176 104004                      ERROR    4          ; RECACT DID NOT ASSERT
2389 010200 042777 000020 171504      BIC      #SYNSCH, @RXCSR ; DROP SEARCH SYNC
2390 010206 032777 004000 171476      BIT      #RECACT, @RXCSR ; IS IT =0?
2391 010214 001401                      BEQ      +4
2392 010216 104004                      ERROR    4          ; RECACT SHOULD BE 0
2393
2394      ., VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
2395      ., IN TWO * SYNC CHARS THRU MAINT DATA BIT
2396      ., WATCH THE RECACT BIT
2397      ., ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
2398      ., * DEPENDENT ON MONITOR
2399      ., IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
2400      ., TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
2401      ., ON THE SECOND CHARACTER
2402      ., ALSO CHECK THIS CHARACTER IN RXDBUF
2403      ., AND CHECK OPERATION OF SYNSCH
2404      ., MODE. SYNC INTERNAL
2405      ., LENGTH FIVE
2406
2407      ., *****

```

2408	010220	000004			TST54	SCOPE	
2409	010222	052777	000400	171476	BIS	#MRESET,@TXCSR	; MASTER RESET
2410	010230	012777	030000	171464	MOV	#SYNINT,@PARCSR	; SET THE MODE
2411	010236	052777	000400	171462	BIS	#MRESET,@TXCSR	; MASTER RESET
2412							
2413							; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2414	010244	012777	064001	171454	MOV	#MTDATA'CLK!MINT!BREAK,@TXCSR	
2415							
2416							; SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2417	010252	012777	030026	171442	MOV	#SYNINT'FIVE'NOPAR'26,@PARCSR	
2418	010260	016703	171432		MOV	RXDBUF,R3	; SET UP FOR ERROR MESSAGE
2419	010264	052777	000020	171420	BIS	#SYNSCH,@RXCSR	; SET SYNC SEARCH
2420							; POKE CLK TO GET RECEIVER INTO SYNCROIZATION
2421	010272	042777	020000	171426	BIC	#CLK,@TXCSR	; POKE CLK DOWN
2422	010300	052777	020000	171420	BIS	#CLK,@TXCSR	; POKE CLK UP
2423							; POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2424	010306	042777	020000	171412	BIC	#CLK,@TXCSR	; POKE CLK DOWN
2425	010314	052777	020000	171404	BIS	#CLK,@TXCSR	; POKE CLK UP
2426	010322	012767	000002	170574	MOV	#2,COUNT	
2427	010330	012767	000005	170564	15	MOV	#5,SHIFT ; # OF SHIFTS
2428	010336	012767	000026	171134	MOV	#26,\$TMP1	; SYNC CHARACTER
2429	010344	004767	006552		JSR	PC,RPOKE	
2430	010350	005367	170550		DEC	COUNT	
2431	010354	001403			BEQ	25	
2432							; TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
2433	010356	105767	170564		TSTB	SYNCNO	
2434	010362	100762			BMI	15	; TWO SYNC CHARS
2435	010364	105777	171322		25	TSTB	@RXCSR ; CHECK REC DONE BIT
2436	010370	100001			BPL	+4	
2437	010372	104004			ERROR	4	; RXDONE SHOULD NOT BE ASSEPTED
2438	010374	032777	004000	171310	BIT	#REACT,@RXCSR	
2439	010402	001001			BNE	+4	
2440	010404	104004			ERROR	4	; REACT SHOULD BE ASSERTED
2441	010406	012767	000005	170506	MOV	#5,SHIFT	
2442	010414	012767	000021	171056	MOV	#21,\$TMP1	; ANY CHARACTER
2443	010422	004767	006474		JSR	PC,RPOKE	
2444	010426	105777	171260		TSTB	@RXCSR	; CHECK RXDONE
2445	010432	100401			BMI	+4	
2446	010434	104004			ERROR	4	; RXDONE SHOULD BE ASSERTED
2447	010436	032777	004000	171246	BIT	#REACT,@RXCSR	
2448	010444	001001			BNE	+4	
2449	010446	104004			ERROR	4	; REACT SHOULD STILL BE ASSERTED
2450	010450	042777	000020	171234	BIC	#SYNSCH,@RXCSR	; CLR SYNC SEARCH
2451	010456	032777	004000	171226	BIT	#REACT,@RXCSR	; IT SHOULD DROP IMMEDIATELY
2452	010464	001401			BEQ	+4	
2453	010466	104004			ERROR	4	; REACT SHOULD BE CLR
2454	010470	105777	171216		TSTB	@RXCSR	; RXDONE
2455	010474	100401			BMI	+4	
2456	010476	104004			ERROR	4	; RXDONE SHOULD STILL BE ASSEPTED
2457	010500	012700	000021		MOV	#21,R0	; EXPECTED DATA
2458	010504	017701	171206		MOV	@RXDBUF,R1	; ACTUAL DATA
2459	010510	020001			CMP	R0,R1	; COMPARE EXP VS ACT
2460	010512	001401			BEQ	+4	
2461	010514	104002			ERROR	2	; DATA CHARS SHOULD COMPARE
2462	010516	105777	171170		TSTB	@RXCSR	; CHECK RXDONE
2463	010522	100001			BPL	+4	

```

2464 010524 104004          ERROR 4          ,RXDONE SHOULD BE CLR FROM
2465                               ,PREVIOUS READING OF RXDBUF
2466
2467                               ,,VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
2468                               ,,IN TWO * SYNC CHARS THRU MAINT DATA BIT
2469                               ,,WATCH THE RECACT BIT
2470                               ,,ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
2471                               ,,* DEPENDENT ON MONITOR.
2472                               ,,IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
2473                               ,,TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
2474                               ,,ON THE SECOND CHARACTER
2475                               ,,ALSO CHECK THIS CHARACTER IN RXDBUF
2476                               ,,AND CHECK OPERATION OF SYNSCH
2477                               ,,MODE SYNC INTERNAL
2478                               ,,LENGTH SIX
2479                               ,,
2480                               ,,*****
2481 010526 000004          TST55  SCOPE
2482 010530 052777 000400 171170          BIS      #MRESET,@TXCSR ,MASTER RESET
2483 010536 012777 030000 171156          MOV      #SYNINT,@PARCSR ,SET THE MODE
2484 010544 052777 000400 171154          BIS      #MRESET,@TXCSR ,MASTER RESET
2485
2486                               ,SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2487 010552 012777 064001 171146          MOV      #MTDATA'CLK'MINT'BREAK,@TXCSR
2488
2489                               ,SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2490 010560 012777 032026 171134          MOV      #SYNINT'SIX'NOPAR'26,@PARCSR
2491 010566 016703 171124          MOV      RXDBUF,R3          ,SET UP FOR ERROR MESSAGE
2492 010572 052777 000020 171112          BIS      #SYNSCH,@RXCSR ,SET SYNC SEARCH
2493                               ,POKE CLK TO GET RECEIVER INTO SYNCROIZATION
2494 010600 042777 020000 171120          BIC      #CLK,@TXCSR ,POKE CLK DOWN
2495 010606 052777 020000 171112          BIS      #CLK,@TXCSP ,POKE CLK UP
2496                               ,POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2497 010614 042777 020000 171104          BIC      #CLK,@TXCSR ,POKE CLK DOWN
2498 010622 052777 020000 171076          BIS      #CLK,@TXCSR ,POKE CLK UP
2499 010630 012767 000002 170266          MOV      #2,COUNT
2500 010636 012767 000006 170256          15      MOV      #6,SHIFT          ,# OF SHIFTS
2501 010644 012767 000026 170626          MOV      #26,$TMP1        ,SYNC CHARACTER
2502 010652 004767 006244          JSP      PC,RPOKE
2503 010656 005367 170242          DEC      COUNT
2504 010662 001403          BEQ      25
2505                               ,TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
2506 010664 105767 170256          TSTB     SYNCNO
2507 010670 100762          BMI      15          ,TWO SYNC CHARS
2508 010672 105777 171014          25      TSTB     @RXCSR ,CHECK REC DONE BIT
2509 010676 100001          BPL      +4
2510 010700 104004          ERROR    4          ,RXDONE SHOULD NOT BE ASSERTED
2511 010702 032777 004000 171002          BIT      #RECACT,@RXCSR
2512 010710 001001          BNE      +4
2513 010712 104004          ERROR    4          ,RECACT SHOULD BE ASSERTED
2514 010714 012767 000006 170200          MOV      #6,SHIFT
2515 010722 012767 000021 170550          MOV      #21,$TMP1        ,ANY CHARACTER
2516 010730 004767 006166          JSR      PC,RPOKE
2517 010734 105777 170752          TSTB     @RXCSR ,CHECK RXDONE
2518 010740 100401          BMI      +4
2519 010742 104004          ERROR    4          ,RXDONE SHOULD BE ASSERTED
  
```

```

2520 010744 032777 004000 170740 BIT #REACT,@RXCSR
2521 010752 001001 BNE +4
2522 010754 104004 ERROR 4 ;REACT SHOULD STILL BE ASSERTED
2523 010756 042777 000020 170726 BIC #SYNSCH,@RXCSR ;CLR SYNC SEARCH
2524 010764 032777 004000 170720 BIT #REACT,@RXCSR ;IT SHOULD DROP IMMEDIATELY
2525 010772 001401 BEQ +4
2526 010774 104004 ERROR 4 ;REACT SHOULD BE CLR
2527 010776 105777 170710 TSTB @RXCSR ,RXDONE
2528 011002 100401 BMI +4
2529 011004 104004 ERROR 4 ;RXDONE SHOULD STILL BE ASSERTED
2530 011006 012700 000021 MOV #21,R0 ;EXPECTED DATA
2531 011012 017701 170700 MOV @RXDBUF,R1 ;ACTUAL DATA
2532 011016 020001 CMP R0,R1 ;COMPARE EXP VS ACT
2533 011020 001401 BEQ +4
2534 011022 104002 ERROR 2 ;DATA CHARS SHOULD COMPARE
2535 011024 105777 170662 TSTB @RXCSR ,CHECK RXDONE
2536 011030 100001 BPL +4
2537 011032 104004 ERROR 4 ;RXDONE SHOULD BE CLR FROM
;PREVIOUS READING OF RXDBUF
2538
2539
2540 ;,VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
2541 ;,IN TWO * SYNC CHARS THRU MAINT DATA BIT
2542 ;,WATCH THE REACT BIT
2543 ;,ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
2544 ;,* DEPENDENT ON MONITOR
2545 ;,IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
2546 ;,TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
2547 ;,ON THE SECOND CHARACTER
2548 ;,ALSO CHECK THIS CHARACTER IN RXDBUF
2549 ;,AND CHECK OPERATION OF SYNSCH
2550 ;,MODE: SYNC INTERNAL
2551 ;,LENGTH: SEVEN
2552
2553 ;,*****
TST56 SCOPE
2554 011034 000004 BIS #MRESET,@TXCSR ;MASTER RESET
2555 011036 052777 000400 170662 MOV #SYNINT,@PARCSR ;SET THE MODE
2556 011044 012777 030000 170650 BIS #MRESET,@TXCSR ;MASTER RESET
2557 011052 052777 000400 170646
2558
2559 ;,SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2560 011060 012777 064001 170640 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2561
2562 ;,SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2563 011066 012777 034026 170626 MOV #SYNINT!SEVEN!NOPAR!26,@PARCSR
2564 011074 016703 170616 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2565 011100 052777 000020 170604 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
2566 ;,POKE CLK TO GET RECEIVER INTO SYNCROIZATION
2567 011106 042777 020000 170612 BIC #CLK,@TXCSR ;POKE CLK DOWN
2568 011114 052777 020000 170604 BIS #CLK,@TXCSR ;POKE CLK UP
2569 ;,POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2570 011122 042777 020000 170576 BIC #CLK,@TXCSR ;POKE CLK DOWN
2571 011130 052777 020000 170570 BIS #CLK,@TXCSR ;POKE CLK UP
2572 011136 012767 000002 167760 MOV #2,COUNT
2573 011144 012767 000007 167750 15 MOV #7,SHIFT ;# OF SHIFTS
2574 011152 012767 000026 170320 MOV #26,$TMP1 ;SYNC CHARACTER
2575 011160 004767 005736 JSR PC,PPOKE

```

INITIALIZE THE COMMON TAGS

2576	011164	005367	167734		DEC	COUNT	
2577	011170	001403			BEQ	25	
2578					, TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED		
2579	011172	105767	167750		TSTB	SYNCNO	
2580	011176	100762			BMI	15	, TWO SYNC CHARS
2581	011200	105777	170506	25	TSTB	@RXCSR	, CHECK REC DONE BIT
2582	011204	100001			BPL	+4	
2583	011206	104004			ERROR	4	, RXDONE SHOULD NOT BE ASSERTED
2584	011210	032777	004000	170474	BIT	#REACT, @RXCSR	
2585	011216	001001			BNE	+4	
2586	011220	104004			ERROR	4	, REACT SHOULD BE ASSERTED
2587	011222	012767	000007	167672	MOV	#7, SHIFT	
2588	011230	012767	000021	170242	MOV	#21, STMP1	, ANY CHARACTER
2589	011236	004767	005660		JSR	PC, RPOKE	
2590	011242	105777	170444		TSTB	@RXCSR	, CHECK RXDONE
2591	011246	100401			BMI	+4	
2592	011250	104004			ERROR	4	, RXDONE SHOULD BE ASSERTED
2593	011252	032777	004000	170432	BIT	#REACT, @RXCSR	
2594	011260	001001			BNE	+4	
2595	011262	104004			ERROR	4	; REACT SHOULD STILL BE ASSERTED
2596	011264	042777	000020	170420	BIC	#SYNSCH, @RXCSR	, CLR SYNC SEARCH
2597	011272	032777	004000	170412	BIT	#REACT, @RXCSR	, IT SHOULD DROP IMMEDIATELY
2598	011300	001401			BEQ	+4	
2599	011302	104004			ERROR	4	, REACT SHOULD BE CLR
2600	011304	105777	170402		TSTB	@RXCSR	, RXDONE
2601	011310	100401			BMI	+4	
2602	011312	104004			ERROR	4	, RXDONE SHOULD STILL BE ASSERTED
2603	011314	012700	000021		MOV	#21, RD	, EXPECTED DATA
2604	011320	017701	170372		MOV	@RXDBUF, R1	, ACTUAL DATA
2605	011324	020001			CMP	RD, R1	, COMPARE EXP VS ACT
2606	011326	001401			BEQ	+4	
2607	011330	104002			ERROR	2	, DATA CHARS SHOULD COMPARE
2608	011332	105777	170354		TSTB	@RXCSR	, CHECK RXDONE
2609	011336	100001			BPL	+4	
2610	011340	104004			ERROR	4	, RXDONE SHOULD BE CLR FROM
2611							, PREVIOUS READING OF RXDBUF
2612							
2613							, VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
2614							, IN TWO * SYNC CHARS THRU MAINT DATA BIT
2615							, WATCH THE REACT BIT
2616							, ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
2617							, * DEPENDENT ON MONITOR
2618							, IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
2619							, TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
2620							, ON THE SECOND CHARACTER
2621							, ALSO CHECK THIS CHARACTER IN RXDBUF
2622							, AND CHECK OPERATION OF SYNSCH
2623							, MODE: SYNC INTERNAL
2624							, LENGTH: EIGHT
2625							,,
2626							,, *****
2627	011342	000004			TST57	SCOPE	
2628	011344	052777	000400	170354	BIS	#MRESET, @TXCSR	, MASTER RESET
2629	011352	012777	030000	170342	MOV	#SYNINT, @PARCSR	, SET THE MODE
2630	011360	052777	000400	170340	BIS	#MRESET, @TXCSR	, MASTER RESET
2631							

```

2632 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2633 011366 012777 064001 170332 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2634
2635 ;SET MODE , # OF BITS,PARITY SENSE,&LOAD SYNC REG
2636 011374 012777 036026 170320 MOV #SYNINT!EIGHT!NOPAR!26,@PARCSR
2637 011402 016703 170310 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2638 011406 052777 000020 170276 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
2639 ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION
2640 011414 042777 020000 170304 BIC #CLK,@TXCSR ;POKE CLK DOWN
2641 011422 052777 020000 170276 BIS #CLK,@TXCSR ;POKE CLK UP
2642 ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2643 011430 042777 020000 170270 BIC #CLK,@TXCSR ;POKE CLK DOWN
2644 011436 052777 020000 170262 BIS #CLK,@TXCSR ;POKE CLK UP
2645 011444 012767 000002 167452 MOV #2,COUNT
2646 011452 012767 000010 167442 1$ MOV #8,SHIFT ;# OF SHIFTS
2647 011460 012767 000020 170012 MOV #26,$TMP1 ;SYNC CHARACTER
2648 011466 004767 005400 JSR PC,RPOKE
2649 011472 005367 167426 DEC COUNT
2650 011476 001403 BEQ 2$
2651 ;TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
2652 011500 105767 167442 TSTB SYNCNO
2653 011504 100762 BMI 1$ ;TWO SYNC CHARS
2654 011506 105777 170200 2$ TSTB @RXCSR ;CHECK REC DONE BIT
2655 011512 100001 BPL +4
2656 011514 104004 ERROR 4 ;RXDONE SHOULD NOT BE ASSERTED
2657 011516 032777 004000 170166 BIT #REACT,@RXCSR
2658 011524 001001 BNE +4
2659 011526 104004 ERROR 4 ;REACT SHOULD BE ASSERTED
2660 011530 012767 000010 167364 MOV #8,SHIFT
2661 011536 012767 000021 167734 MOV #21,$TMP1 ;ANY CHARACTER
2662 011544 004767 005352 JSR PC,RPOKE
2663 011550 105777 170136 TSTB @RXCSR ;CHECK RXDONE
2664 011554 100401 BMI +4
2665 011556 104004 ERROR 4 ;RXDONE SHOULD BE ASSERTED
2666 011560 032777 004000 170124 BIT #REACT,@RXCSR
2667 011566 001001 BNE +4
2668 011570 104004 ERROR 4 ;REACT SHOULD STILL BE ASSERTED
2669 011572 042777 000020 170112 BIC #SYNSCH,@RXCSR ;CLR SYNC SEARCH
2670 011600 032777 004000 170104 BIT #REACT,@RXCSR ;IT SHOULD DROP IMMEDIATELY
2671 011606 001401 BEQ +4
2672 011610 104004 ERROR 4 ;REACT SHOULD BE CLR
2673 011612 105777 170074 TSTB @RXCSR ;RXDONE
2674 011616 100401 BMI +4
2675 011620 104004 ERROR 4 ;RXDONE SHOULD STILL BE ASSERTED
2676 011622 012700 000021 MOV #21,R0 ;EXPECTED DATA
2677 011626 017701 170064 MOV @RXDBUF,R1 ;ACTUAL DATA
2678 011632 020001 CMP R0,R1 ;COMPARE EXP VS ACT
2679 011634 001401 BEQ +4
2680 011636 104002 ERROR 2 ;DATA CHARS SHOULD COMPARE
2681 011640 105777 170046 TSTB @RXCSR ;CHECK RXDONE
2682 011644 100001 BPL +4
2683 011646 104004 ERROR 4 ;RXDONE SHOULD BE CLR FROM
2684 ;PREVIOUS READING OF RXDBUF
2685
2686 ;, THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2687 ;, RECEIVER SECTION, IT USES THE ERROR FLAGS
    
```

```
2688      , , TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2689      , , (OVERRUN, RXERR)
2690      , , MODE ISYMOD
2691      , , LENGTH FIVE
2692      , , CHAR 25
2693      , ,
2694      , , *****
2695 011650 000004 TST60 SCOPE
2696 011652 052777 000400 170046 BIS #MRESET, @TXCSR , MASTER RESET
2697 011660 012777 000000 170034 MOV #ISYMOD, @PARCSR , SET THE MODE
2698 011666 052777 000400 170032 BIS #MRESET, @TXCSR , MASTER RESET
2699
2700      , SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2701 011674 012777 064001 170024 MOV #MTDATA'CLK'MINT'BREAK, @TXCSR
2702
2703      , SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2704 011702 012777 000000 170012 MOV #ISYMOD'FIVE'NOPAR'0, @PARCSR
2705 011710 052777 000020 167774 BIS #SYNSCH, @RXCSR , SET SYNC SEARCH
2706      , POKE CLK TO GET RECEIVER INTO SYNCRIZATION
2707 011716 042777 020000 170002 BIC #CLK, @TXCSR , POKE CLK DOWN
2708 011724 052777 020000 167774 BIS #CLK, @TXCSR , POKE CLK UP
2709      , POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2710 011732 042777 020000 167766 BIC #CLK, @TXCSR , POKE CLK DOWN
2711 011740 052777 020000 167760 BIS #CLK, @TXCSR , POKE CLK UP
2712 011746 016703 167744 MOV RXDBUF, R3 ; SET UP FOR ERROR MESSAGE
2713 011752 012700 000025 MOV #25, R0 , EXPECTED
2714 011756 012767 000007 167136 MOV #7, SHIFT , # OF SHIFTS
2715 011764 012767 000152 167506 MOV #152, $TMP1 , DATA CHAR
2716 011772 004767 005124 JSR PC, RPOKE , SHIFT IN THIS CHAR
2717 011776 105777 167710 TSTB @RXCSR , RXDONE ?
2718 012002 100401 BMI +4
2719 012004 104004 ERROR 4 , RXDONE SHOULD BE SET
2720 012006 017701 167704 MOV @RXDBUF, R1 , ACTUAL
2721 012012 020001 CMP R0, R1 , COMPARE EXPECTED VS ACTUAL
2722 012014 001401 BEQ +4
2723 012016 104002 ERROR 2 , RECEIVED DATA DID NOT MATCH
2724      , EXPECTED DATA - CHECK MAINT DATA
2725      , OR RECEIVER LOGIC
2726 012020 012767 000007 167074 MOV #7, SHIFT , # OF SHIFTS
2727 012026 012767 000152 167444 MOV #152, $TMP1 , DATA CHAR
2728 012034 004767 005062 JSR PC, RPOKE , SHIFT IN THIS CHAR
2729      , NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2730 012040 012767 000007 167054 MOV #7, SHIFT , # OF SHIFTS
2731 012046 012767 000152 167424 MOV #152, $TMP1 , DATA CHAR
2732 012054 004767 005042 JSR PC, RPOKE , SHIFT IN THIS CHAR
2733 012060 012700 140025 MOV #140000'25, R0 , EXPECTED DATA PLUS
2734      , RXERR & OVERRUN
2735 012064 017701 167626 MOV @RXDBUF, R1 , ACTUAL
2736 012070 020001 CMP R0, R1 , COMPARE EXP VS ACT
2737 012072 001401 BEQ +4
2738 012074 104002 ERROR 2 , SPECIFICALLY LOOK AT RXERR &
2739      , OVERRUN BITS THEY BOTH SHOULD BE SET
2740
2741      , , THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2742      , , RECEIVER SECTION, IT USES THE ERROR FLAGS
2743      , , TO DETERMINE THAT IT WAS SELECTED CORRECTLY
```

INITIALIZE THE COMMON TAGS

```
2744      , (OVRUN, RXERR)
2745      , MODE, ISYMOD
2746      , LENGTH, FIVE
2747      , CHAR, 12
2748
2749      , *****
2750 012076 000004 TST61 SCOPE
2751 012100 052777 000400 167620 BIS #MRESET, @TXCSR , MASTER RESET
2752 012106 012777 000000 167606 MOV #ISYMOD, @PARCSR , SET THE MODE
2753 012114 052777 000400 167604 BIS #MRESET, @TXCSR , MASTER RESET
2754
2755 , SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
2756 012122 012777 064001 167576 MOV #MTDATA'CLK'MINT'BREAK, @TXCSR
2757
2758 , SET MODE , # OF BITS, PARITY SENSE, & LOAD SYNC REG
2759 012130 012777 000000 167564 MOV #ISYMOD'FIVE'NOPAR'0, @PARCSR
2760 012136 052777 000020 167546 BIS #SYNSCH, @RXCSR , SET SYNC SEARCH
2761 , POKE CLK TO GET RECEIVER INTO SYNCRIZATION
2762 012144 042777 020000 167554 BIC #CLK, @TXCSR , POKE CLK DOWN
2763 012152 052777 020000 167546 BIS #CLK, @TXCSR , POKE CLK UP
2764 , POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2765 012160 042777 020000 167540 BIC #CLK, @TXCSR , POKE CLK DOWN
2766 012166 052777 020000 167532 BIS #CLK, @TXCSR , POKE CLK UP
2767 012174 016703 167516 MOV RXDBUF, R3 , SET UP FOR ERROR MESSAGE
2768 012200 012700 000012 MOV #12, R0 , EXPECTED
2769 012204 012767 000007 166710 MOV #7, SHIFT , # OF SHIFTS
2770 012212 012767 000124 167260 MOV #124, STMP1 , DATA CHAR
2771 012220 004767 004676 JSR PC, RPOKE , SHIFT IN THIS CHAR
2772 012224 105777 167462 TSTB @RXCSR , RXDONE ?
2773 012230 100401 Bmi +4
2774 012232 104004 ERROR 4 , RXDONE SHOULD BE SET
2775 012234 017701 167456 MOV @RXDBUF, R1 , ACTUAL
2776 012240 020001 CMP R0, R1 , COMPARE EXPECTED VS ACTUAL
2777 012242 001401 BEQ +4
2778 012244 104002 ERROR 2 , RECEIVED DATA DID NOT MATCH
2779 , EXPECTED DATA - CHECK MAINT DATA
2780 , OR RECEIVER LOGIC
2781 012246 012767 000007 166646 MOV #7, SHIFT , # OF SHIFTS
2782 012254 012767 000124 167216 MOV #124, STMP1 , DATA CHAR
2783 012262 004767 004634 JSR PC, RPOKE , SHIFT IN THIS CHAR
2784 , NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2785 012266 012767 000007 166626 MOV #7, SHIFT , # OF SHIFTS
2786 012274 012767 000124 167176 MOV #124, STMP1 , DATA CHAR
2787 012302 004767 004614 JSR PC, RPOKE , SHIFT IN THIS CHAR
2788 012306 012700 140012 MOV #140000'12, R0 , EXPECTED DATA PLUS
2789 , RXERR & OVRUN
2790 012312 017701 167400 MOV @RXDBUF, R1 , ACTUAL
2791 012316 020001 CMP R0, R1 , COMPARE EXP VS ACT
2792 012320 001401 BEQ +4
2793 012322 104002 ERROR 2 , SPECIFICALLY LOOK AT RXERR &
2794 , OVRUN BITS. THEY BOTH SHOULD BE SET
2795
2796
2797
2798 , END OF PASS
2799 , TYPE NAME OF TEST
```





2856	012616	016767	166270	166256		MOV	PASCNT, LIGHTS	, DISPLAY PASS COUNT
2857	012624	016767	166262	166702		MOV	PASCNT, \$PASS	, PASS COUNT TO APT
2858	012632	013701	000042			MOV	@#42, R1	, CHECK FOR ACT-11 OR DDP
2859	012636	001406				BEQ	RESTRT	, IF NO CONTINUE TESTING
2860	012640	000005				RESET		
2861	012642	000005				RESET		
2862	012644	004711			\$ENDAD	JSR	PC, (R1)	
2863	012646	000240				NOP		
2864	012650	000240				NOP		
2865	012652	000240				NOP		
2866	012654				RESTRT			
2867	012654	012767	003376	166524		MOV	#TST1+2, \$LPADR	, LOAD LAST ADDR
2868	012662	004767	000004			JSR	PC, CKSWR	
2869	012666	000167	170416			JMP	BEGIN	
2870								
2871								, CHECK SWITCH REGISTER ROUTINE
2872								, CHECKS TO ALLOW FOR < G > TO ALLOW
2873								, THE CHANGING OF LOCATION 176
2874								
2875	012672	005737	000042		CKSWR	TST	@#42	
2876	012676	001040				BNE	OUT	
2877	012700	022767	000176	166532		CMR	#SWREG, SWR	, SOFTWARE SWR PRESENT?
2878	012706	001034				BNE	OUT	, NO--LEAVE
2879	012710	105777	166530			TSTB	@\$TKS	, CHECK TTY READY
2880	012714	100031				BPL	OUT	, NO--LEAVE
2881	012716	017767	166524	000422		MOV	@\$TKB, MSG	, GET CHARACTER
2882	012724	042767	177600	000414		BIC	#177600, MSG	, STRIP JUNK
2883	012732	122767	000007	000406		CMR	#7, MSG	, IS IT < G > ?
2884	012740	001017				BNE	OUT	, NO
2885	012742	104401	016067			TYPE	, MCNTG	
2886	012746	005137	013006		CNTLU	COM	@#RDSW	
2887	012752	104401	016077			TYPE	, MMSWR	
2888	012756	104413				CONVRT		
2889	012760	013010				SWREG		
2890	012762	104406	016110			INSTR, MMNEW		
2891	012766	104410				PARAM		
2892	012770	000000				0		
2893	012772	177777				177777		
2894	012774	000176				SWREG		
2895	012776	000	001		BYTE	0, 1		
2896	013000	005037	013006		OUT	CLR	@#RDSW	
2897	013004	000207				RTS	PC	
2898	013006	000000			RDSW	WORD	0	
2899	013010	000001			SWPEGL	1		
2900	013012	006	002			BYTE	6, 2	
2901	013014	000176				SWREG		
2902								
2903	013016	000005				5		
2904								
2905								, CHECK FOR FREEZE ON CURRENT DATA
2906								
2907	013020	004767	177646		SCOP1	JSR	PC, CKSWR	
2908	013024	032777	001000	166406		BIT	#SW09, @SWR	
2909	013032	001402				BEQ	1\$	
2910	013034	016716	166050			MOV	LOCK, (SP)	
2911	013040	000002			1\$	RTI		

```

2912          SBTTL  TYPE ROUTINE
2913
2914          , ,*****
2915          , *ROUTINE TO TYPE ASCIZ MESSAGE  MESSAGE MUST TERMINATE WITH A 0 BYTE
2916          , *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED
2917          , *NOTE1          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER
2918          , *NOTE2          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED
2919          , *NOTE3          $FILLC CONTAINS THE CHARACTER TO FILL AFTER
2920          , *
2921          , *CALL
2922          , *1) USING A TRAP INSTRUCTION
2923          , *      TYPE      , MESADR          , , MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2924          , *OR
2925          , *      TYPE
2926          , *      MESADR
2927          , *
2928
2929 013042 105767 166411  STYPE  TSTB  $TPFLG          , , IS THERE A TERMINAL?
2930 013046 100002          EPL  1$          , , BR IF YES
2931 013050 000000          HALT          , , HALT HERE IF NO TERMINAL
2932 013052 000430          BR  3$          , , LEAVE
2933 013054 010046          1$  MOV  RO, -(SP)          , , SAVE PO
2934 013056 017600 000002  MOV  @2(SP), RO          , , GET ADDRESS OF ASCIZ STRING
2935 013062 122767 000001 166456  CMPB #APTENV, $ENV          , , RUNNING IN APT MODE
2936 013070 001011          BNE  62$          , , NO, GO CHECK FOR APT CONSOLE
2937 013072 132767 000100 166447  BITB #APTSPool, $ENV          , , SPOOL MESSAGE TO APT
2938 013100 001405          BEQ  62$          , , NO, GO CHECK FOR CONSOLE
2939 013102 010067 000004          MOV  RO, 61$          , , SETUP MESSAGE ADDRESS FOR APT
2940 013106 004767 000006          JSR  PC, $ATY3          , , SPOOL MESSAGE TO APT
2941 013112 000000          61$  WORD  0          , , MESSAGE ADDRESS
2942 013114 132767 000040 166425  62$  BITB #APTCSUP, $ENV          , , APT CONSOLE SUPPRESSED
2943 013122 001003          BNE  60$          , , YES, SKIP TYPE OUT
2944 013124 112046          2$  MOVB (RO)+, -(SP)          , , PUSH CHARACTER TO BE TYPED ONTO STACK
2945 013126 001005          BNE  4$          , , BR IF IT ISN'T THE TERMINATOR
2946 013130 005726          TST  (SP)+          , , IF TERMINATOR POP IT OFF THE STACK
2947 013132 012600          60$  MOV  (SP)+, RO          , , RESTORE RO
2948 013134 062716 000002          3$  ADD  #2, (SP)          , , ADJUST RETURN PC
2949 013140 000002          RTI          , , RETURN
2950 013142 122716 000011          4$  CMPB #HT, (SP)          , , BRANCH IF <HT>
2951 013146 001430          BEQ  8$          , ,
2952 013150 122716 000200          CMPB #CRLF, (SP)          , , BRANCH IF NOT <CRLF>
2953 013154 001006          BNE  5$          , ,
2954 013156 005726          TST  (SP)+          , , POP <CR><LF> EQUIV
2955 013160 104401          TYPE          , , TYPE A CR AND LF
2956 013162 001523          $CRLF
2957 013164 105067 000130          CLRb $CHARCNT          , , CLEAR CHARACTER COUNT
2958 013170 000755          BR  2$          , , GET NEXT CHARACTER
2959 013172 004767 000056          5$  JSR  PC, $TYPEC          , , GO TYPE THIS CHARACTER
2960 013176 126726 166254          6$  CMPB $FILLC, (SP)+          , , IS IT TIME FOR FILLER CHARS ?
2961 013202 001350          BNE  2$          , , IF NO GO GET NEXT CHAR
2962 013204 016746 166244          MOV  $NULL, -(SP)          , , GET # OF FILLER CHARS NEEDED
2963          , , AND THE NULL CHAR
2964 013210 105366 000001          7$  DECB 1(SP)          , , DOES A NULL NEED TO BE TYPED?
2965 013214 002770          BLT  6$          , , BR IF NO--GO POP THE NULL OFF OF STACK
2966 013216 004767 000032          JSP  PC, $TYPEC          , , GO TYPE A NULL
2967 013222 105367 000072          DECB $CHARCNT          , , DO NOT COUNT AS A COUNT

```

```

2968 013226 000770          BR      75          ..LOOP
2969
2970          .HOR ZONTAL TAB PROCESSOR
2971
2972 013230 112716 000040      8$      MOVB      #' , (SP)      ..REPLACE TAB WITH SPACE
2973 013234 004767 000014      9$      JSR        PC, $TYPEC      ..TYPE A SPACE
2974 013240 132767 000007 000052      BITB      #7, $CHARCNT      ..BRANCH IF NOT AT
2975 013246 001372          BNE      9$          ..TAB STOP
2976 013250 005726          TST      (SP)+          ..POP SPACE OFF STACK
2977 013252 000724          BR        2$          ..GET NEXT CHARACTER
2978 013254 105777 166170      $TYPEC  TSTB      @STPS      ..WAIT UNTIL PRINTER IS READY
2979 013260 100375          BPL      $TYPEC
2980 013262 116677 000002 166162      MOVB      2(SP), @STPB      ..LOAD CHAR TO BE TYPED INTO DATA REG
2981 013270 122766 000015 000002      CMPB      #CR, 2(SP)      ..IS CHARACTER A CARRIAGE RETURN?
2982 013276 001003          BNE      1$          ..BRANCH IF NO
2983 013300 105067 000014          CLRB      $CHARCNT      ..YES--CLEAR CHARACTER COUNT
2984 013304 000406          BR        $TYPEX      ..EXIT
2985 013306 122766 000012 000002 1$      CMPB      #LF, 2(SP)      ..IS CHARACTER A LINE FEED?
2986 013314 001402          BEQ      $TYPEX      ..BRANCH IF YES
2987 013316 105227          INCB      (PC)+          ..COUNT THE CHARACTER
2988 013320 000000      $CHARCNT WORD 0          ..CHAPACTER COUNT STORAGE
2989 013322 000207      $TYPEX  RTS        PC
2990
2991
2992          .ASCII STRING INPUT ROUTINE
2993
2994 013324 017667 000000 000014      NSTR:  MOV        @ (SP), MSG      .PICK UP MESSAGE
2995 013332 062716 000002          ADD      #2, (SP)      .JUMP AROUND MESSAGE FOR RTI
2996 013336 105767 166204          TSTB      $ENV          .APT CONTROL
2997 013342 001036          BNE      INSTR2      .YES NO TYPE
2998 013344 104401          INSTR1 TYPE
2999 013346 000000      MSG      0
3000 013350 012704 016122          MOV      #INBUF, R4      .GET STARTING LOC OF INBUF
3001 013354 012703 000007          MOV      #7, R3          .MAX # OF CHARS
3002 013360 105777 166060      1$      TSTB      @STKS      .TTY FLAG
3003 013364 100375          BPL      1$
3004 013366 117714 166054          MOVB      @STKB, (R4)      .TAKE CHAR
3005 013372 142714 000200          BICB      #200, (R4)      .STRIP
3006 013376 121427 000025          CMPB      (R4), #25      .IS IT < G>
3007 013402 001760          BEQ      INSTR1
3008 013404 122427 000015          CMPB      (R4)+, #15      .CHECK FOR CR
3009 013410 001413          BEQ      INSTR2
3010 013412 105777 166032      2$      TSTB      @STPS      .TEST FLAG
3011 013416 100375          BPL      2$
3012 013420 117777 166022 166024      MOVB      @STKB, @STPB      .ECHO CHARACTER
3013 013426 005303          DEC      R3          .DID YOU TYPE TOO MANY CHARS ?
3014 013430 001353          BNE      1$
3015 013432 104401          INSTR1 TYPE
3016 013434 015407          MQM      , ?
3017 013436 000742          BR        INSTR1      .RETRY
3018 013440 000002      INSTR2 RTI
3019
3020          .CONVERT ASCII STRING TO OCTAL
3021
3022 013442 011605      PAPAM: MOV      (SP), R5      .PUT CONTENTS OF SP INTO R5
3023 013444 012567 000162          MOV      (R5)+, LOLIM      .PUT LOW LIMIT INTO LOLIM
  
```

```

3024 013450 012567 000160      MOV      (R5)+,HILIM      ,PUT HIGH LIMIT INTO HILIM
3025 013454 012567 000156      MOV      (R5)+,DEVADR    ,PUT STORE LOC INTO DEVADP
3026 013460 112567 000154      MOVB     (R5)+,LOBITS    ,PUT MASK INTO LOBITS
3027 013464 112567 000151      MOVB     (R5)+,ADRCNT    ,PUT COUNT INTO ADRCNT
3028 013470 010516              MOV      R5,(SP) ,RESTORE RETURN ADDR ON STACK FOR RTI
3029 013472 005005              PARAM1  CLR      R5
3030 013474 012704 016122              MOV      #INBUF,R4
3031 013500 122714 000015              CMPB     #15,(R4)      ,CR ?
3032 013504 001420              BEQ      PARERR ,YOU TYPED CR TOO SOON !
3033 013506 121427 000060              15      CMPB     (R4),#60      ,LOW LIMIT ASCII 0
3034 013512 002415              BLT      PARERR
3035 013514 121427 000067              CMPB     (R4),#67      ,HIGH LIMIT ASCII 7
3036 013520 003012              BGT      PARERR
3037 013522 142714 000060              BICB     #60,(R4)      ,CONVERT TO OCTAL
3038 013526 152405              BISB     (R4)+,R5      ,STORE AWAY ITS AN OK CHAR
3039 013530 122714 000015              CMPB     #15,(R4)      ,CR ?
3040 013534 001414              BEQ      LIMITS ,NOW CHECK FOR HIGH & LOW LIMIT CONDS
3041 013536 006305              ASL      R5      ,ALLOCATE ROOM FOR NEXT CHAR
3042 013540 006305              ASL      R5
3043 013542 006305              ASL      R5
3044 013544 000760              BR       15
3045 013546 122714 000015              PARERR  CMPB     #15,(R4)      ,CR?
3046 013552 001003              BNE      1205
3047 013554 005737 013006              TST      @#RDSW      ,CK SWR USED
3048 013560 001023              BNE      PARTI
3049 013562 104407              1205    INSTER  ,RETRY
3050 013564 000742              BR       PARAM1
3051
3052              ,TEST TO SEE IF NUMBER IS WITHIN LIMITS
3053
3054 013566 020567 000042              LIMITS  CMP      R5,HILIM
3055 013572 101365              BHI      PARERR ,THE # IS TOO HIGH
3056 013574 020567 000032              CMP      R5,LOLIM
3057 013600 103762              BLO      PARERR ,THE # IS TOO LOW
3058 013602 136705 000032              BITB     LOBITS,R5      ,TEST BY MASKINGTHE #
3059 013606 001357              BNE      PARERR
3060
3061              ,STORE NUMBER AT SPECIFIED ADDRESS
3062
3063 013610 016704 000022              15      MOV      DEVADR,R4      ,GET STARTING ADDR OF
3064 013614 010524              MOV      R5,(R4)+      ,STORE AT THIS ADDR
3065 013616 062705 000002              ADD      #2,R5
3066 013622 105367 000013              DECB     ADRCNT ,HOW MANY TIMES + 2 ?
3067 013626 001372              BNE      15
3068 013630 000002              PARTI   RTI
3069 013632 000000              LOLIM   0
3070 013634 000000              HILIM   0
3071 013636 000000              DEVADR  0
3072 013640 000000              LOBITS  0
3073              ADRCNT=LOBITS+1
3074
3075              ,SAVE PC OF TEST THAT FAILED AND RD-P5
3076
3077 013642 016667 000004 165256  SAV05  MOV      4(SP),SAVPC
3078
3079              ,SAVE R0-R5

```

```

3080
3081 U13650 010567 165620 SV05 MOV R5, $REG5
3082 013654 010467 165612 MOV R4, $REG4
3083 013660 010367 165604 MOV R3, $REG3
3084 013664 010267 165576 MOV R2, $REG2
3085 013670 010167 165570 MOV R1, $REG1
3086 013674 010067 165562 MOV R0, $REG0
3087 013700 000002 RTI
3088
3089 , RESTORE R0-R5
3090
3091 013702 016700 165554 RES05 MOV $REG0, R0
3092 013706 016701 165552 MOV $REG1, R1
3093 013712 016702 165550 MOV $REG2, R2
3094 013716 016703 165546 MOV $REG3, R3
3095 013722 016704 165544 MOV $REG4, R4
3096 013726 016705 165542 MOV $REG5, R5
3097 013732 000002 RTI
3098
3099 , CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
3100
3101 013734 104401 CONVR TYPE
3102 013736 015413 MCRLF , CR LF
3103 013740 017601 000000 MOV @ (SP), R1 , PICK UP DATA POINTER
3104 013744 062716 000002 ADD #2, (SP) , SET UP SP FOR RTI
3105 013750 012167 000130 MOV (R1)+, WPCNT , PICK UP # OF WORDS FROM TABLE
3106 013754 112167 000126 1$ MOV (R1)+, CHRCNT , PICK UP # OF CHARS FROM TABLE
3107 013760 112167 000123 MOV (R1)+, SPACNT , PICK UP # OF SPACES FROM TABLE
3108 013764 013167 000120 MOV @ (R1)+, BINWRD , PICK UP ADDRESS OF MSG
3109 , FROM TABLE
3110 013770 016704 000114 2$ MOV BINWRD, R4 , SAVE
3111 013774 116705 000106 MOV (R4)+, CHRCNT, R5 , SAVE
3112 014000 012700 016164 MOV #TEMP, R0 , STARTING ADDRESS OF TEMP BLOCK
3113 014004 010403 3$ MOV R4, R3 , SAVE
3114 014006 042703 177770 BIC #177770, R3 , CLR OUT UPPER BITS SAVE CHAR
3115 014012 062703 000260 ADD #260, R3 , CONVERT TO ASCII
3116 014016 110320 MOV (R3)+, (R0)+ , STORE AWAY
3117 014020 006204 ASR R4 , SHIFT FOR NEXT #
3118 014022 006204 ASR R4 , DITTO
3119 014024 006204 ASR R4 , DITTO
3120 014026 005305 DEC R5 , DEC CHAR COUNT
3121 014030 001365 BNE 3$ , DO IT AGAIN ?
3122 014032 012703 016226 MOV #MDATA, R3 , STARTING ADDRESS OF MDATA BLOCK
3123 014036 114023 4$ MOV (R0), (R3)+ , REVERSE THE ORDER OF NUMBERS
3124 014040 105367 000042 DECB CHRCNT , DEC CHAR COUNT
3125 014044 001374 BNE 4$ , DO IT AGAIN ?
3126 014046 105767 000035 TSTB SPACNT , HOW MANY SPACES ?
3127 014052 001405 BEQ 6$ , TYPE # IF BR = 0
3128 014054 112723 5$ MOV (R3)+, #240 , "SPACE" IN ASCII
3129 014060 105367 000023 DECB SPACNT , DEC # OF SPACE COUNT
3130 014064 001373 BNE 5$ , DO IT AGAIN ?
3131 014066 105013 6$ CLRB (R3) , INSERT "0" FOR TTY OUTPUT ROUTINE
3132 014070 104401 TYPE
3133 014072 016226 MDATA , THIS MESSAGE
3134 014074 005367 000004 DEC WRDCNT , HOW MANY #'S ?
3135 014100 001325 BNE 1$ , DO THIS ROUTINE AGAIN IF NOT EQUAL TO 0
  
```

```

3136 014102 000002          RTI      , RETURN TO PROGRAM
3137 014104 000000          WROCNT  0
3138 014106 000000          CHRCNT  0
3139          014107          SPACNT=CHRCNT+1
3140 014110 000000          BINWRD  0
3141
3142          , COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
3143          , BUFFER TO THE CHARACTERS "N" AND "Y"
3144          , IF THE CHARACTER IS "N" CLEAR THE FLAG
3145          , IF THE CHARACTER IS "Y" SET THE FLAG
3146
3147 014112 017605 000000    SETFLG  MOV    @ (SP), R5
3148 014116 122767 000116 001776    CMPB   #'N, INBUF      , IS IT "N" ?
3149 014124 001002          BNE    1$
3150 014126 105015          CLRB   (R5)      , 000
3151 014130 000406          BR     2$
3152 014132 122767 000131 001762 1$    CMPB   #'Y, INBUF      , IS IT "Y" ?
3153 014140 001005          BNE    3$
3154 014142 112715 177777          MOVB  #-1, (R5)      , 377
3155 014146 062716 000002          2$    ADD    #2, (SP)
3156 014152 000002          RTI
3157 014154 104407          3$    INSTER , RETRY
3158 014156 000755          BR     SETFLG
3159          SBTTL  ERROR HANDLER ROUTINE
3160
3161          , *****
3162          , *THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3163          , *SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3164          , *AND GO TO SAVIT ON ERROR
3165          , *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE
3166          , *SW15=1      HALT ON ERROR
3167          , *SW13=1      INHIBIT ERPOP TYPEOUTS
3168          , *SW10=1      BELL ON ERROR
3169          , *SW09=1      LOOP ON ERROR
3170          , *CALL
3171          , *      ERROR  N      , , ERROR=EMT AND N=ERROR ITEM NUMBER
3172
3173 014160          SERROR
3174 014160 105267 165217          7$    INCB   SERRFLG      , , SET THE ERROR FLAG
3175 014164 001775          BEQ    7$          , , DON'T LET THE FLAG GO TO ZERO
3176 014166 016777 165210 165246    MOV    $STNM, @DISPLAY , , DISPLAY TEST NUMBER AND ERPOP FLAG
3177 014174 032777 002000 165236    BIT   #BIT10, @SWR     , , BELL ON ERROR?
3178 014202 001402          BEQ    1$          , , NO - SKIP
3179 014204 104401 001516          TYPE  , $BELL        , , RING BELL
3180 014210 005267 165176          1$    INC    $ERTTL      , , COUNT THE NUMBER OF ERRORS
3181 014214 011667 165176          MOV    (SP), $ERRPC    , , GET ADDRESS OF ERROR INSTRUCTION
3182 014220 162767 000002 165170    SUB   #2, $ERRPC
3183 014226 117767 165164 165160    MOVB  @ $ERRPC, $ITEMB , , STRIP AND SAVE THE ERROR ITEM CODE
3184 014234 032777 020000 165176    BIT   #BIT13, @SWR     , , SKIP TYPEOUT IF SET
3185 014242 001004          BNE    20$         , , SKIP TYPEOUTS
3186 014244 004767 000072          JSR   PC, SAVIT      , , GO TO USER ERROR ROUTINE
3187 014250 104401 001523          TYPE  , $CRLF
3188 014254          20$
3189 014254 122767 000001 165264    CMPB  #APTENV, $ENV    , , RUNNING IN APT MODE
3190 014262 001007          BNE    2$          , , NO, SKIP APT ERROR REPORT
3191 014264 116767 165124 000004    MOVB  $ITEMB, 21$     , , SET ITEM NUMBER AS ERPOP NUMBER

```

```

3192 014272 004767 000016' JSR PC, $ATY4 ;; REPORT FATAL ERROR TO APT
3193 014276 000 21$ BYTE 0
3194 014277 000 BYTE 0
3195 014300 000777 22$ BR 22$ ;; APT ERROR LOOP
3196 014302 005777 165132 2$ TST 2$SWR ;; HALT ON ERROR
3197 014306 100001 BPL 3$ ;; SKIP IF CONTINUE
3198 014310 000000 HALT ;; HALT ON ERROR'
3199 014312 032777 001000 165120 3$ BIT #BIT09, 2$SWR ;; LOOP ON ERROR SWITCH SET?
3200 014320 001402 BEQ 4$ ;; BR IF NO
3201 014322 016716 165062 MOV $LPERR, (SP) ;; FUDGE RETURN FOR LOOPING
3202 014326 005767 165162 4$ TST $ESCAPE ;; CHECK FOR AN ESCAPE ADDRESS
3203 014332 001402 BEQ 5$ ;; BR IF NONE
3204 014334 016716 165154 MOV $ESCAPE, (SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE
3205 014340 5$
3206 014340 000002 RTI ;; RETURN
3207 014342 010067 164562 SAVIT MOV R0, HLD0
3208 014346 010167 164560 MOV R1, HLD1
3209 014352 010267 164556 MOV R2, HLD2
3210 014356 010367 164554 MOV R3, HLD3
3211 014362 010467 164552 MOV R4, HLD4
3212 014366 010567 164550 MOV R5, HLD5
3213 014372 016767 165004 164544 MOV $TSTNM, HLD6
    
```

SBTTL ERROR MESSAGE TYPEOUT ROUTINE

```

;; *****
;* THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;* ERROR IS TO BE REPORTED IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
;* AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR
    
```

```

3221 SERRTYP
3222 014400
3223 014400 104401 001523 TYPE , $CPLF ;; "CARRIAGE RETURN" & "LINE FEED"
3224 014404 010046 MOV RO, -(SP) ;; SAVE RO
3225 014406 005000 CLR RO ;; PICKUP THE ITEM INDEX
3226 014410 153700 001414 B1SB 2#$ITEMB, RO
3227 014414 001004 BNE 1$ ;; IF ITEM NUMBER IS ZERO, JUST
3228 ;; TYPE THE PC OF THE ERROR
3229 014416 016746 164774 MOV $ERRPC, -(SP) ;; SAVE $ERRPC FOR TYPEOUT
3230 ;; ERROR ADDRESS
3231 014422 104402 TYPOC ;; GO TYPE--OCTAL ASCII (ALL DIGITS)
3232 014424 000426 BR 6$ ;; GET OUT
3233 014426 005300 1$ DEC RO ;; ADJUST THE INDEX SO THAT IT WILL
3234 014430 006300 ASL RO ;; WORK FOR THE ERROR TABLE
3235 014432 006300 ASL RO
3236 014434 006300 ASL RO
3237 014436 062700 001652 ADD #$ERRTB, RO ;; FORM TABLE POINTER
3238 014442 012067 000004 MOV (RO)+, 2$ ;; PICKUP "ERROR MESSAGE" POINTER
3239 014446 001404 BEQ 3$ ;; SKIP TYPEOUT IF NO POINTER
3240 014450 104401 TYPE ;; TYPE THE "ERROR MESSAGE"
3241 014452 000000 2$ WORD 0 ;; "ERROR MESSAGE" POINTER GOES HERE
3242 014454 104401 001523 TYPE , $CPLF ;; "CARRIAGE RETURN" & "LINE FEED"
3243 014460 012067 000004 3$ MOV (RO)+, 4$ ;; PICKUP "DATA HEADER" POINTER
3244 014464 001404 BEQ 5$ ;; SKIP TYPEOUT IF 0
3245 014466 104401 TYPE ;; TYPE THE "DATA HEADER"
3246 014470 000000 4$ WORD 0 ;; "DATA HEADER" POINTER GOES HERE
3247 014472 104401 001523 TYPE , $CPLF ;; "CARRIAGE RETURN" & "LINE FEED"
    
```



```

3248 014476 011000      5$    MOV    (R0),R0      ;; PICKUP "DATA TABLE" POINTER
3249 014500 001004      BNE    7$          ;; GO TYPE THE DATA
3250 014502 012600      6$    MOV    (SP)+,R0    ;; RESTORE R0
3251 014504 104401 001523  TYPE   , $CRLF     ;; "CARRIAGE RETURN" & "LINE FEED"
3252 014510 000207      RTS    PC          ;; RETURN
3253 014512      7$
3254 014512 013046      MOV    @ (R0)+, -(SP)  ;; SAVE @ (R0)+ FOR TYPEOUT
3255 014514 104402      TYPOC      ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
3256 014516 005710      TST    (R0)        ;; IS THERE ANOTHER NUMBER?
3257 014520 001770      BEQ    6$          ;; BR IF NO
3258 014522 104401 014530  TYPE   , 8$        ;; TYPE TWO(2) SPACES
3259 014526 000771      BR     7$          ;; LOOP
3260 014530 020040 000    8$    .ASCIZ / /        ;; TWO(2) SPACES
3261      014534      EVEN
3262      SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3263
3264      ;, *****
3265      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3266      ;*OCTAL (ASCII) NUMBER AND TYPE IT
3267      ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3268      ;*CALL.
3269      ;*    MOV    NUM, -(SP)      ;; NUMBER TO BE TYPED
3270      ;*    TYPOS      ;; CALL FOR TYPEOUT
3271      ;*    BYTE   N              ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3272      ;*    BYTE   M              ;; M=1 OR 0
3273      ;*                                ;; 1=TYPE LEADING ZEROS
3274      ;*                                ;; 0=SUPPRESS LEADING ZEROS
3275      ;*
3276      ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3277      ;*$TYPOS OR STYPOC
3278      ;*CALL
3279      ;*    MOV    NUM, -(SP)      ;; NUMBER TO BE TYPED
3280      ;*    TYPON      ;; CALL FOR TYPEOUT
3281      ;*
3282      ;*$STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3283      ;*CALL
3284      ;*    MOV    NUM, -(SP)      ;; NUMBER TO BE TYPED
3285      ;*    TYPOC      ;; CALL FOR TYPEOUT
3286
3287 014534 017646 000000  STYPOS  MOV    @ (SP), -(SP)    ;; PICKUP THE MODE
3288 014540 116667 000001  000211  MOVB   1(SP), $OFILL  ;; LOAD ZERO FILL SWITCH
3289 014546 112667 000207      MOVB   (SP)+, $OMODE+1  ;; NUMBER OF DIGITS TO TYPE
3290 014552 062716 000002      ADD    #2, (SP)        ;; ADJUST RETURN ADDRESS
3291 014556 000406      BR     $TYPON
3292 014560 112767 000001 000171  STYPOC. MOVB   #1, $OFILL    ;; SET THE ZERO FILL SWITCH
3293 014566 112767 000006 000165      MOVB   #6, $OMODE+1    ;; SET FOR SIX(6) DIGITS
3294 014574 112767 000005 000154  STYPON  MOVB   #5, $OCNT    ;; SET THE ITERATION COUNT
3295 014602 010346      MOV    R3, -(SP)     ;; SAVE R3
3296 014604 010446      MOV    R4, -(SP)     ;; SAVE R4
3297 014606 010546      MOV    R5, -(SP)     ;; SAVE R5
3298 014610 116704 000145      MOVB   $OMODE+1, R4    ;; GET THE NUMBER OF DIGITS TO TYPE
3299 014614 005404      NEG    R4
3300 014616 062704 000006      ADD    #6, R4         ;; SUBTRACT IT FOR MAX ALLOWED
3301 014622 110467 000132      MOVB   R4, $OMODE     ;; SAVE IT FOR USE
3302 014626 116704 000125      MOVB   $OFILL, R4     ;; GET THE ZERO FILL SWITCH
3303 014632 016605 000012      MOV    12(SP), R5    ;; PICKUP THE INPUT NUMBER

```

3304	014636	005003			CLR	R3	.. CLEAR THE OUTPUT WORD
3305	014640	006105		15	ROL	R5	.. ROTATE MSB INTO "C"
3306	014642	000404			BR	35	.. GO DO MSB
3307	014644	006105		25	ROL	R5	.. FORM THIS DIGIT
3308	014646	006105			ROL	R5	
3309	014650	006105			ROL	R5	
3310	014652	010503			MOV	R5, R3	
3311	014654	006103		35	ROL	R3	.. GET LSB OF THIS DIGIT
3312	014656	105367	000076		DECB	\$OMODE	.. TYPE THIS DIGIT?
3313	014662	100016			BPL	75	.. BR IF NO
3314	014664	042703	177770		BIC	#177770, R3	.. GET RID OF JUNK
3315	014670	001002			BNE	45	.. TEST FOR 0
3316	014672	005704			TST	R4	.. SUPPRESS THIS 0?
3317	014674	001403			BEQ	55	.. BR IF YES
3318	014676	005204		45	INC	R4	.. DON'T SUPPRESS ANYMORE 0'S
3319	014700	052703	000060		BIS	#'0, R3	.. MAKE THIS DIGIT ASCII
3320	014704	052703	000040		BIS	#', R3	.. MAKE ASCII IF NOT ALREADY
3321	014710	110367	000040		MOVB	R3, 85	.. SAVE FOR TYPING
3322	014714	104401	014754		TYPE	, 85	.. GO TYPE THIS DIGIT
3323	014720	105367	000032		DECB	\$OCNT	.. COUNT BY 1
3324	014724	003347			BGT	25	.. BR IF MORE TO DO
3325	014726	002402			BLT	65	.. BR IF DONE
3326	014730	005204			INC	R4	.. INSURE LAST DIGIT ISN'T A BLANK
3327	014732	000744			BR	25	.. GO DO THE LAST DIGIT
3328	014734	012605		65	MOV	(SP)+, R5	.. RESTORE R5
3329	014736	012604			MOV	(SP)+, R4	.. RESTORE R4
3330	014740	012603			MOV	(SP)+, R3	.. RESTORE R3
3331	014742	016666	000002 000004		MOV	2(SP), 4(SP)	.. SET THE STACK FOR RETURNING
3332	014750	012616			MOV	(SP)+, (SP)	
3333	014752	000002			RTI		.. RETURN
3334	014754	000		85	BYTE	0	.. STORAGE FOR ASCII DIGIT
3335	014755	000			BYTE	0	.. TERMINATOR FOR TYPE ROUTINE
3336	014756	000		\$OCNT	BYTE	0	.. OCTAL DIGIT COUNTER
3337	014757	000		\$OFILL	BYTE	0	.. ZERO FILL SWITCH
3338	014760	000000		\$OMODE	WORD	0	.. NUMBER OF DIGITS TO TYPE
3339							.. ENTER HERE ON POWER FAILURE
3340							
3341							
3342	014762				SPWRDN		
3343	014762	010046			PFAIL	MOV R0, -(SP)	.. SAVE R0-R5 ON PROCESSOR STACK
3344	014764	010146				MOV R1, -(SP)	
3345	014766	010246				MOV R2, -(SP)	
3346	014770	010346				MOV R3, -(SP)	
3347	014772	010446				MOV R4, -(SP)	
3348	014774	010546				MOV R5, -(SP)	
3349	014776	016746	163022			MOV 24, -(SP)	
3350	015002	010667	164110			MOV SP, SAVSP	.. SAVE STACK POINTER
3351	015006	012767	015020 163010			MOV #RESTART, 24	.. SET UP FOR POWER UP TRAP
3352	015014	000000			HALT		.. HALT ON POWER DOWN NORMAL
3353	015016	000777			BR		
3354							
3355							.. PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
3356							
3357	015020	016706	164072		RESTAR	MOV SAVSP, SP	.. RESTORE STACK POINTER
3358	015024	012605				MOV (SP)+, R5	.. RESTORE R0-R5
3359	015026	012604				MOV (SP)+, R4	

3360	015030	012603				MOV	(SP)+,R3	
3361	015032	012602				MOV	(SP)+,R2	
3362	015034	012601				MOV	(SP)+,R1	
3363	015036	012600				MOV	(SP)+,R0	
3364	015040	012767	014762	162756		MOV	# PFAIL, 24	, SET UP FOR POWER FAILURE
3365	015046	106427	000340			MTPS	#340	
3366	015052	012706	001100			MOV	#STACK, SP	
3367	015056	005067	001102			CLR	TEMP	
3368	015062	005267	001076			INC	TEMP	
3369	015066	001375				BNE	-4	
3370	015070	104413				CONVRT		
3371	015072	015114				PFTAB		
3372	015074	104401				TYPE		
3373	015076	015416				MPFAIL		
3374	015100	005067	164277			CLR	\$ERFLG	
3375	015104	005067	164306			CLR	\$ERRPC	
3376	015110	000177	163770			JMP	@RETURN	
3377	015114	000001			PFTAB	1		
3378	015116	006	002			BYTE	6, 2	
3379	015120	000207				RETURN		
3380	015122	005015	042012	053125	MTITLE	ASCIZ	<15><12><12>/DUV11 DZDUQ-B TAPE A /<15><12>	
3381	015130	030461	042040	042132				
3382	015136	050525	041055	052040				
3383	015144	050101	020105	020101				
3384	015152	005015	000					
3385	015155	015	052012	041505	MVECTO	ASCIZ	<15><12>/VEC ADD-/	
3386	015162	040440	042104	000055				
3387	015170	005015	051461	020124	MREGAD	ASCIZ	<15><12>/1ST DEV. REC CSR ADD-/	
3388	015176	042504	035126	051040				
3389	015204	041505	041440	051123				
3390	015212	040440	042104	000055				
3391	015220	005015	052515	052114	MMULT	ASCIZ	<15><12>/MULT DEV ? (Y OR N)-/	
3392	015226	042040	053105	037440				
3393	015234	024040	020131	051117				
3394	015242	047040	026451	000				
3395	015247	015	046012	051501	MLASTD	ASCIZ	<15><12>/LAST DEV REC CSR ADDR-/	
3396	015254	020124	042504	035126				
3397	015262	051040	041505	041440				
3398	015270	051123	040440	042104				
3399	015276	026522	000					
3400	015301	075	042504	044526	DEVICE	ASCIZ	/=DEVICE /	
3401	015306	042503	020040	000				
3402	015313	015	051412	046105	MCOV	ASCIZ	<15><12>/SELECT TO RUN @ACTREG/	
3403	015320	041505	020124	047524				
3404	015326	051040	047125	040040				
3405	015334	041501	051124	043505				
3406	015342	000						
3407	015343	015	047412	043126	MRANGE	ASCIZ	<15><12>/OVFLO RETYPE LAST DEV RXCSR ADDS-/	
3408	015350	047514	051072	052105				
3409	015356	050131	020105	040514				
3410	015364	052123	042040	053105				
3411	015372	051040	041530	051123				
3412	015400	040440	042104	026523				
3413	015406	000						
3414	015407	040	037440	000	MQM	ASCIZ	/ ?/	
3415	015413	015	000012		MCRLF	ASCIZ	<15><12>	

DZDUQ-B MACY11 30(1046) 21-SEP-77 09.04 PAGE 69  
DZDUGB M11 31-MAY-77 09.47

BINARY TO OCTAL (ASCII) AND TYPE

SEQ 0067

3416	015416	043120	044501	026114	MPFAIL	ASCIZ	/PFAIL, RESTART AT TEST IN PROGRESS/
3417	015424	020040	042522	052123			
3418	015432	051101	020124	052101			
3419	015440	052040	051505	020124			
3420	015446	047111	050040	047522			
3421	015454	051107	051505	000123			
3422	015462	005015	047105	020104	MEPASS	ASCIZ	<15><12>/END OF PASS TAPE A/
3423	015470	043117	050040	051501			
3424	015476	020123	040524	042520			
3425	015504	040440	000				
3426	015507	015	051012	000	MR	ASCIZ	<15><12>/R/
3427	015513	015	052012	051505	MTSTPC	ASCIZ	<15><12>/TEST PC-/
3428	015520	020124	041520	000055			
3429	015526	005015	047514	045503	MLOCK	ASCIZ	<15><12>/LOCK ON TEST? (Y OR N)-/
3430	015534	047440	020116	052040			
3431	015542	051505	037524	024040			
3432	015550	020131	051117	047040			
3433	015556	026451	000				
3434	015561	015	021412	047440	MSYNC	ASCIZ	<15><12>/# OF SYNC CHARS SELECTED ( 1 OR 2)-/
3435	015566	020106	054523	041516			
3436	015574	041440	040510	051522			
3437	015602	051440	046105	041505			
3438	015610	042524	020104	020050			
3439	015616	020061	051117	031040			
3440	015624	026451	000				
3441	015627	015	044412	020123	MWIRE6	ASCIZ	<15><12>/IS SEC XMIT SWITCH E55-2 IN? (Y OR N)-/
3442	015634	042523	020103	046530			
3443	015642	052111	051440	044527			
3444	015650	041524	020110	032505			
3445	015656	026465	020062	047111			
3446	015664	020077	054450	047440			
3447	015672	020122	024516	000055			
3448	015700	005015	051511	051440	MWIRE5	ASCIZ	<15><12>/IS SEC REC SWITCH E55-3 IN? (Y OR N)-/
3449	015706	041505	051040	041505			
3450	015714	051440	044527	041524			
3451	015722	020110	032505	026465			
3452	015730	020063	047111	020077			
3453	015736	054450	047440	020122			
3454	015744	024516	000055				
3455	015750	005015	051511	047440	MWIRE4	ASCIZ	<15><12>/IS OPT CLP ENABLE SWITCH E55-1 IN? (Y OR N)-/
3456	015756	052120	041440	051114			
3457	015764	042440	040516	046102			
3458	015772	020105	053523	052111			
3459	016000	044103	042440	032465			
3460	016006	030455	044440	037516			
3461	016014	024040	020131	051117			
3462	016022	047040	026451	000			
3463	016027	015	005012	031510	MEXTJ	ASCIZ	<15><12><12>/H315 CONNECTOR ON ?(Y OR N)-/
3464	016034	032461	041440	047117			
3465	016042	042516	052103	051117			
3466	016050	047440	020116	024077			
3467	016056	020131	051117	047040			
3468	016064	026451	000				
3469	016067	015	020012	043536	MCNTG	ASCIZ	<15><12>/ G /
3470	016074	020040	000				
3471	016077	040	053523	036522	MMSWR	ASCIZ	/ SWR= /

```

3472 016104 020040 000040
3473 016110 020040 047040 053505 MMNEW ASLIZ / NEW= /
3474 016116 020075 000040
3475
3476
3477 , BUFFERS FOR INPUT-OUTPUT
3478
3479 016122 000000 INBUF 0
3480 016164 000000 = +40
3481 016164 000000 TEMP 0
3482 016226 000000 = +40
3483 016226 000000 MDATA 0
3484 016270 000000 = +40
3485 SBTTL SCOPE HANDLER ROUTINE
3486
3487 , *****
3488 , *THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS IT WILL INCREMENT
3489 , *AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG (DISPLAY<7 0>)
3490 , *AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15 08>
3491 , *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE
3492 , *SW14=1 LOOP ON TEST
3493 , *SW11=1 INHIBIT ITERATIONS
3494 , *SW09=1 LOOP ON ERROR
3495 , *SW08=1 LOOP ON TEST IN SWR<7 0>
3496 , *CALL
3497 , * SCOPE , , SCOPE=10T
3498
3499 016270 $SCOPE
3500
3501 , SCOPE LOOP AND INTERATION HANDLER
3502
3503 SCOPE
3504 016270 004767 174376 JSR PC,CKSWR
3505 016274 005067 163116 CLR $ERRPC , CLEAR LAST ERROR PC
3506 016300 022716 003376 CMP #TST1+2, (SP) , IS SCOPE AT BEGINING OF TEST 1?
3507 016304 001422 BEQ $XTSTR , YES NO LOOP
3508
3509 016306 032777 040000 163124 TTST BIT #BIT14, @SWR , THIS CODE IS FOR TESTING FOR BIT 14
3510 016314 001412 BEQ 1$ , ON LSI WHICH SYSMAC CANNOT HANDLE
3511 016316 016767 163060 163062 MOV $TSTNM, $LPADR
3512 016324 000406 BR 1$
3513 016326 105777 163112 TSTB @STKS , KEYBOARD DONE?
3514 016332 100123 BPL $OVER , BR IF NO
3515 016334 017766 163106 177776 MOV @STKB, -2(SP) , CLEAR DONE BIT
3516 016342 032777 040000 163070 1$ BIT #BIT14, @SWR , LOOP ON PRESENT TEST?
3517 016350 001114 BNE $OVER , YES IF SW14=1
3518 , *****START OF CODE FOR THE XOR TESTER*****
3519 016352 000416 $XTSTR BR 6$ , IF RUNNING ON THE "XOR" TESTER CHANGE
3520 , THIS INSTRUCTION TO A "NOP" (NOP=240)
3521 016354 013746 000004 MOV @#ERRVEC, -(SP) , SAVE THE CONTENTS OF THE ERROR VECTOR
3522 016360 012737 016400 000004 MOV #5$, @#ERRVEC , SET FOR TIMEOUT
3523 016366 005737 177060 TST @#177060 , TIME OUT ON XOR?
3524 016372 012637 000004 MOV (SP)+, @#ERRVEC , RESTORE THE ERROR VECTOR
3525 016376 000463 BR $SVLAD , GO TO THE NEXT TEST
3526 016400 022626 5$ CMP (SP)+, (SP)+ , CLEAR THE STACK AFTER A TIME OUT
3527 016402 012637 000004 MOV (SP)+, @#ERRVEC , RESTORE THE ERROR VECTOR

```

```

3528 016406 000423          BR      7$          ,, LOOP ON THE PRESENT TEST
3529 016410                6$ ,#####END OF CODE FOR THE XOR TESTER#####
3530 016410 032777 000400 163022 BIT      #BIT08,@SWR    ,, LOOP ON SPEC. TEST?
3531 016416 001404          BEQ      2$          ,, BR IF NO
3532 016420 127767 163014 162754 CMPB    @SWR,$STSTM    ,, ON THE RIGHT TEST? SWR<? 0>
3533 016426 001465          BEQ      $OVER       ,, BR IF YES
3534 016430 105767 162747          2$ TSTB    $ERFLG      ,, HAS AN ERROR OCCURRED?
3535 016434 001421          BEQ      3$          ,, BR IF NO
3536 016436 126767 162753 162737 CMPB    $ERMAX,$ERFLG ,, MAX ERRORS FOR THIS TEST OCCURRED?
3537 016444 101015          BHI      3$          ,, BR IF NO
3538 016446 032777 001000 162764 BIT      #BIT09,@SWR    ,, LOOP ON ERROR?
3539 016454 001404          BEQ      4$          ,, BR IF NO
3540 016456 016767 162726 162722 7$ MOV     $LPERR,$LPADR ,, SET LOOP ADDRESS TO LAST SCOPE
3541 016464 000446          BR      $OVER       ,,
3542 016466 105067 162711          4$ CLRB    $ERFLG      ,, ZERO THE ERROR FLAG
3543 016472 005067 163014          CLR     $TIMES      ,, CLEAR THE NUMBER OF ITERATIONS TO MAKE
3544 016476 000415          BR      1$          ,, ESCAPE TO THE NEXT TEST
3545 016500 032777 004000 162732 3$ BIT      #BIT11,@SWR   ,, INHIBIT ITERATIONS?
3546 016506 001011          BNE     1$          ,, BR IF YES
3547 016510 005767 163020          TST     $PASS      ,, IF FIRST PASS OF PROGRAM
3548 016514 001406          BEQ     1$          ,, INHIBIT ITERATIONS
3549 016516 005267 162662          INC     $ICNT      ,, INCREMENT ITERATION COUNT
3550 016522 026767 162764 162654 CMP     $TIMES,$ICNT ,, CHECK THE NUMBER OF ITERATIONS MADE
3551 016530 002024          BGE     $OVER       ,, BR IF MORE ITERATION REQUIRED
3552 016532 012767 000001 162644 1$ MOV     #1,$ICNT    ,, REINITIALIZE THE ITERATION COUNTER
3553 016540 016767 000056 162744          MOV     $MXCNT,$TIMES ,, SET NUMBER OF ITERATIONS TO DO
3554 016546 105267 162630          $$SVLAD INCB    $STSTM    ,, COUNT TEST NUMBERS
3555 016552 116767 162624 162752 MOVB    $STSTM,$TESTN ,, SET TEST NUMBER IN APT MAILBOX
3556 016560 011667 162622          MOV     (SP),$LPADR ,, SAVE SCOPE LOOP ADDRESS
3557 016564 011667 162620          MOV     (SP),$LPERR ,, SAVE ERROR LOOP ADDRESS
3558 016570 005067 162720          CLR     $ESCAPE    ,, CLEAR THE ESCAPE FROM ERROR ADDRESS
3559 016574 112767 000001 162613 MOVB    #1,$ERMAX    ,, ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3560 016602 016777 162574 162632 $OVER MOV     $STSTM,@DISPLAY ,, DISPLAY TEST NUMBER
3561 016610 016716 162572          MOV     $LPADR,(SP) ,, FUDGE RETURN ADDRESS
3562 016614 000002          4$ RTI
3563 016616 001407          BRW     1407
3564 016620 000432          BRX     432
3565 016622 000005          $MXCNT 5          ,, MAX NUMBER OF ITERATIONS
3566          SBTTL TRAP DECODER
3567
3568          ,, *****
3569          ,*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3570          ,*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3571          ,*OF THE DESIRED ROUTINE THEN USING THE ADDRESS OBTAINED IT WILL
3572          ,*GO TO THAT ROUTINE.
3573
3574 016624 010046          STRAP MOV     R0,-(SP)    ,, SAVE R0
3575 016626 016600 000002          MOV     2(SP),R0    ,, GET TRAP ADDRESS
3576 016632 005740          TST     -(R0)      ,, BACKUP BY 2
3577 016634 111000          MOVB   (R0),R0     ,, GET RIGHT BYTE OF TRAP
3578 016636 006300          ASL    R0          ,, POSITION FOR INDEXING
3579 016640 016000 016660          MOV     $TRPAD(R0),R0 ,, INDEX TO TABLE
3580 016644 000200          RTS    R0          ,, GO TO ROUTINE
3581
3582
3583          ,, THIS IS USE TO HANDLE THE "GETPRI" MACRO
  
```

```

3584
3585 016646 011646 $TRAP2 MOV (SP),-(SP) ,MOVE THE PC DOWN
3586 016650 016666 000004 000002 MOV 4(SP),2(SP) ,MOVE THE PSW DOWN
3587 016656 000002 RTI ,RESTORE THE PSW
3588
3589 .SBTTL TRAP TABLE
3590
3591 ,*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3592 ,*BY THE "TRAP" INSTRUCTION
3593
3594 , ROUTINE
3595 ,-----
3596 016660 016646 $TRPAD WORD $TRAP2
3597 016662 013042 $TYPE ,CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
3598 016664 014560 $TYPOC ,CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3599 016666 014534 $TYPOS ,CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
3600 016670 014574 $TYPON ,CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
3601
3602
3603 016672 013020 SCOP1 ,CALL=SCOP1 TRAP+5(104405)
3604 016674 013324 INSTR ,CALL=INSTR TRAP+6(104406)
3605 016676 013432 INSTER ,CALL=INSTER TRAP+7(104407)
3606 016700 013442 PARAM ,CALL=PARAM TRAP+10(104410)
3607 016702 013642 SAVOS ,CALL=SAVOS TRAP+11(104411)
3608 016704 013702 RESOS ,CALL=RESOS TRAP+12(104412)
3609 016706 013734 CONVRT ,CALL=CONVRT TRAP+13(104413)
3610 016710 014112 SETFLG ,CALL=SETFLG TRAP+14(104414)
3611 ,*****
3612 ,UTILITIES
3613 ,*****
3614
3615 ,THIS UTILITY CALCULATES PRIORITY LEVEL
3616 016712 006367 000044 DULEV ASL DUPRT ,SHIFT LEFT
3617 016716 006367 000040 ASL DUPRT ,
3618 016722 006367 000034 ASL DUPRT ,
3619 016726 006367 000030 ASL DUPRT ,
3620 016732 006367 000024 ASL DUPRT ,
3621 016736 016767 000020 000020 MOV DUPRT,LESS1 ,MOVE THIS TO LESS1
3622 016744 162767 000001 000012 SUB #1,LESS1 ,CREATE LESS1
3623 016752 042767 000037 000004 BIC #37,LESS1 ,CLEAR TNZVC
3624 016760 000207 RTS PC
3625 016762 000240 DUPRT PR5
3626 016764 000200 LESS1 PR4 ,LEVEL TO ALLOW INTERRUPTS
3627
3628 .NEW DU ADDRESSES
3629 016766 016767 000126 162716 DUADDR MOV DUBASE,RXCSR ,XXX0
3630 016774 005267 000120 INC DJBASE
3631 017000 016767 000114 162706 MOV DUBASE,HRXCSR ,XXX1
3632 017006 005267 000106 INC DUBASE
3633 017012 016767 000102 162676 MOV DUBASE,RXDBUF ,XXX2
3634 017020 016767 000074 162674 MOV DUBASE,PARCSR ,XXX2
3635 017026 005267 000066 INC DUBASE
3636 017032 016767 000062 162660 MOV DUBASE,HRXDBUF ,XXX3
3637 017040 016767 000054 162656 MOV DJBASE,HPARCSR ,XXX3
3638 017046 005267 000046 INC DUBASE
3639 017052 016767 000042 162646 MOV DUBASE,TXCSR ,XXX4

```

3640	017060	005267	000034			INC	DUBASE	
3641	017064	016767	000030	162636		MOV	DUBASE, HTXCSR	, XXX5
3642	017072	005267	000022			INC	DUBASE	
3643	017076	016767	000016	162626		MOV	DUBASE, TXDBUF	; XXX6
3644	017104	005267	000010			INC	DUBASE	
3645	017110	016767	000004	162616		MOV	DUBASE, HTXDBUF	, XXX7
3646	017116	000207				RTS	PC	
3647	017120	000000			DUBASE	0		
3648								
3649								
3650								
3651								
3652	017122	042777	040000	162576	RPOKE	BIC	#MTDATA, @TXCSR	
3653	017130	005067	162346			CLR	\$TMP2	
3654	017134	006067	162340			ROR	\$TMP1	, FORCE CARRY
3655	017140	006067	162336			ROR	\$TMP2	, PICK UP CARRY IN BIT 15
3656	017144	006267	162332			ASR	\$TMP2	, SHIFT INTO BIT 14
3657	017150	042767	100000	162324		BIC	#BIT15, \$TMP2	, CLR BIT 15
3658	017156	056777	162320	162542		BIS	\$TMP2, @TXCSR	, POKE MAINT DATA
3659	017164	042777	020000	162534		BIC	#CLK, @TXCSR	, POKE CLK
3660	017172	052777	020000	162526		B S	#CLK, @TXCSR	
3661	017200	005367	161716			DEC	SHIFT	
3662	017204	001346				BNE	RPOKE	
3663	017206	000207				RTS	PC	
3664								
3665								
3666	017210	016767	162264	162264	ODD8	MOV	\$TMP1, \$TMP2	, SAVE TEMP1
3667	017216	005067	162262			CLR	\$TMP3	
3668	017222	012727	000010			MOV	#8, (PC)+	
3669	017226	000000			4\$	0		
3670	017230	006067	162246		1\$	ROR	\$TMP2	
3671	017234	005567	162244			ADC	\$TMP3	
3672	017240	005367	177762			DEC	4\$	
3673	017244	001371				BNE	1\$	
3674	017246	006067	162232			ROR	\$TMP3	
3675	017252	103404				BCS	2\$	
3676	017254	052767	000400	162216		BIS	#BIT8, \$TMP1	, SET ODD PARITY
3677	017262	000403				BR	3\$	
3678	017264	042767	000400	162206	2\$	BIC	#BIT8, \$TMP1	, CLR EVEN PARITY
3679								
3680	017272	000207			3\$	RTS	PC	
3681								
3682								
3683	017274	016767	162200	162200	EVEN8	MOV	\$TMP1, \$TMP2	, SAVE TEMP1
3684	017302	005067	162176			CLR	\$TMP3	
3685	017306	012727	000010			MOV	#8, (PC)+	
3686	017312	000000			4\$	0		
3687	017314	006067	162162		1\$	ROR	\$TMP2	
3688	017320	005567	162160			ADC	\$TMP3	
3689	017324	005367	177762			DEC	4\$	
3690	017330	001371				BNE	1\$	
3691	017332	006067	162146			ROR	\$TMP3	
3692	017336	103004				BCC	2\$	
3693	017340	052767	000400	162132		BIS	#BIT8, \$TMP1	, SET EVEN PARITY
3694	017346	000403				BR	3\$	
3695	017350	042767	000400	162122	2\$	BIC	#BIT8, \$TMP1	, CLR ODD PARITY



DZDUQ-B MACY11 30(1046) 21-SEP-77 09 04 PAGE 74  
UZDU6B M11 31-MAY-77 09 47 TRAP TABLE

SEQ 0072

3696						, STMP1 NOW HAS EVEN PARITY CHARACTER
3697	017356	000207		3\$	RTS	PC
3698	017360	062716	000002	TRPREG	ADD	#2, (SP) , ALLOW T TO "CRUNCH" INTO ERROR BACK
3699						, IN MAIN PART OF THE PROGRAM
3700	017364	000002			RTI	
3701		000001		END		

AAA	003200	1293*							
ABASE =	000000	876	917						
ACDW1 =	000000	876	919						
ACDW2 =	000000	876	920						
ACPUOP=	000000	876	891						
ACTREG	001166	735*	1249*	1263*	1264*	1271*	2812	2815	2825
ADDW0 =	000000	876	921						
ADDW1 =	000000	876	922						
ADDW10=	000000	876	931						
ADDW11=	000000	876	932						
ADDW12=	000000	876	933						
ADDW13=	000000	876	934						
ADDW14=	000000	876	935						
ADDW15=	000000	876	936						
ADDW2 =	000000	876	923						
ADDW3 =	000000	876	924						
ADDW4 =	000000	876	925						
ADDW5 =	000000	876	926						
ADDW6 =	000000	876	927						
ADDW7 =	000000	876	928						
ADDW8 =	000000	876	929						
ADDW9 =	000000	876	930						
ADEVCT=	000000	876	882						
ADEVN =	000000	876	918						
ADRCNT=	013641	3027*	3066*	3073*					
ARENV =	000000	876	887						
ARENVM =	000000	876	888						
AFATAL=	000000	876	879						
AMADR1=	000000	876	904						
AMADR2=	000000	876	908						
AMADR3=	000000	876	911						
AMADR4=	000000	876	914						
AMAMS1=	000000	876	898						
AMAMS2=	000000	876	906						
AMAMS3=	000000	876	909						
AMAMS4=	000000	876	912						
AMSGAD=	000000	876	884						
AMSGLG=	000000	876	885						
AMSGTY=	000000	876	878						
AMTYP1=	000000	876	899						
AMTYP2=	000000	876	907						
AMTYP3=	000000	876	910						
AMTYP4=	000000	876	913						
APASS =	000000	876	881						
APRIOR=	000000	876							
APTCSU=	000040	536*	2942						
APTENV=	000001	536*	2935	3189					
APTSIZ=	000200	536*	1168						
APTSP0=	000100	536*	2937						
ASWREG=	000000	876	889						
ATESTN=	000000	876	880						
AUNIT =	000000	876	883						
AUSWR =	000000	876	890						
AVECT1=	000000	876	915						
AVECT2=	000000	876	916						
BASEAD	001154	730*	1231*	1268*	1269	1275*	1277*	2819*	2831* 2835



CROSS REFERENCE TABLE -- USER SYMBOLS

DNA = 100000	799#	992#	1974											
DNAINT= 000040	806#	999#	1669	1670	1673	1674	1678	1680						
DSC = 100000	763#	956#	1953											
DSINTE= 000040	773#	966#	1549	1550	1553	1554	1558	1560						
DSR = 001000	769#	962#												
DSWR = 177570	586#	842	1155											
DTR = 000002	777#	970#	1433	1434	1437	1438	1442	1448	1452	2123	2192	2277	2323	
DT1 002116	1026	1030	1034	1080#										
DT4 002126	1038	1083#												
DUADDR 016766	1230	2836	3629#											
DUBASE 017120	1226	1229	2835*	3629	3630*	3631	3632*	3633	3634	3635*	3636	3637	3638*	
	3639	3640*	3641	3642*	3643	3644*	3645	3647#						
DULEV 016712	1288	3616#												
DUPRT 016762	1287*	3616*	3617*	3618*	3619*	3620*	3621	3625#						
DURIS 001740	1054#	2839*												
DURIV 001736	1053#	1237	1240	1241	2837*	2844								
DUTIS 001744	1056#	2843*												
DUTIV 001742	1055#	2841*												
EIGHT = 006000	794#	987#	2354	2381	2636									
EMTVEC= 000030	675#	1140*	1141*											
EM1 001762	1024	1060#												
EM2 002022	1028	1066#												
EM3 002043	1032	1069#												
EM4 001746	1036	1058#												
ERRCNT 001114	703#													
ERRVEC= 000004	668#	1153	1154*	1165*	3521	3522*	3524*	3527*						
EVEN8 017274	3683#													
EVEPAR= 001400	797#	990#												
EVPAR = 000400	786#	979#												
FIVE = 000000	791#	984#	2417	2704	2759									
FRMERR= 020000	782#	975#	2008											
GNS = ***** U	3597	3598	3599	3600	3603	3604	3605	3606	3607	3608	3609	3610		
HOXEN = 000010	808#	1001#	1629	1630	1633	1634	1638	1640						
HILIM 013634	3024*	3054	3070#											
HLD0 001130	712#	1080	3207*											
HLD1 001132	713#	1080	3208*											
HLD2 001134	714#	3209*												
HLD3 001136	715#	3210*												
HLD4 001140	716#	3211*												
HLD5 001142	717#	3212*												
HLD6 001144	718#	3213*												
HOLD 001120	708#	1360*	1490	1814	2065	2128	2149	2174	2197	2212	2227	2251	2266	
	2282	2298	2313	2328										
HPARCS 001724	1047#	1385*	3637*											
HRXCSR 001714	1043#	1354*	1844*	3631*										
HRXDBU 001720	1045#	1370*	3636*											
HT = 000011	578#	2950	2991											
HTXCSR 001730	1049#	1400*	1855*	3641*										
HTXDBU 001734	1051#	1415*	3645*											
INBUF 016122	1296	1300	3000	3030	3148	3152	3479#							
INIFLG 001172	743#	1199	1202*											
INSTER= 104407	1304	3049	3157	3605#										
INSTR = 104406	1221	1232	1242	1253	1278	1294	1307	1311	1315	1319	1330	2890	3604#	
INSTR2 013440	2997	3009	3018#											
IOTVEC= 000020	673#	1138*	1139*											
ISYMOD= 000000	790#	983#	2374	2381	2697	2704	2752	2759						

















CROSS REFERENCE TABLE -- USER SYMBOLS

	3536	3543	3544	3545	3557	3560	3565							
SSWREG 001550	889#	1170												
SSWRMK= 000000	3495	3496	3532											
STESTN 001532	880#	3555*												
STIMES 001512	866#	1146*	3543*	3550	3553*	3565								
STKB 001446	845#	2881	3004	3012	3515									
STKS 001444	844#	2879	3002	3513										
STMP0 001476	860#													
STMP1 001500	861#	2428*	2442*	2501*	2515*	2574*	2588*	2647*	2661*	2715*	2727*	2731*	2770*	
	2782*	2786*	3654*	3666	3676*	3678*	3683	3693*	3695*					
STMP2 001502	862#	3653*	3655*	3656*	3657*	3658	3666*	3670*	3683*	3687*				
STMP3 001504	863#	3667*	3671*	3674*	3684*	3688*	3691*							
STMP4 001506	864#													
STMP5 001510	865#													
STN = 000062	536#	1347	1349#	1363	1365#	1378	1380#	1393	1395#	1408	1410#	1423	1425#	
	1431	1433#	1460	1462#	1498	1500#	1527	1529#	1547	1549#	1567	1569#	1587	
	1589#	1607	1609#	1627	1629#	1647	1649#	1667	1669#	1687	1689#	1709	1711#	
	1731	1733#	1751	1753#	1771	1773#	1791	1793#	1823	1825#	1837	1839#	1872	
	1874#	1900	1902#	1928	1930#	1939	1941#	1950	1952#	1961	1963#	1971	1973#	
	1982	1984#	1994	1996#	2005	2007#	2016	2018#	2027	2029#	2037	2039#	2075	
	2077#	2089	2091#	2104	2106#	2344	2346#	2371	2373#	2407	2409#	2480	2482#	
	2553	2555#	2626	2628#	2694	2696#	2749	2751#						
STPB 001452	847#	2980*	2991	3012*										
STPFLG 001457	851#	2929	2991											
STPS 001450	846#	2978	2991	3010										
STRAP 016624	1142	3574#												
STRAP2 016646	3585#	3596												
STRP = 000015	3589#	3598#	3599#	3600#	3601#	3603	3604#	3605#	3606#	3607#	3608#	3609#	3610#	
	3611#													
STRPAD 016660	3579	3596#												
STSTM 002142	1116#													
STSTNM 001402	824#	1180*	1335	1338	2852*	3176	3207	3213	3490	3511	3532	3554*	3555	
	3560	3566												
STYPBN= ***** U	3601													
STYPDS= ***** U	3601													
STYPE 013042	536	2929#	3589	3597										
STYPEC 013254	2959	2966	2973	2978#	2979									
STYPEX 013322	2984	2986	2989#											
STYPOC 014560	3292#	3598												
STYPON 014574	3291	3294#	3600											
STYPOS 014534	3287#	3599												
SUNIT 001540	883#													
SUNITM 002146	1118#													
SUSHR 001552	790#	1205												
SVECT1 001576	915#													
SVECT2 001600	916#													
SXTSTR 016352	3507	3519#												
SOFILL 014757	3288*	3292*	3302	3337#										
S4OCAT= ***** U	3186	3516												
= 017366	536#	570#	683#	686#	691#	746#	747#	821#	872	1078#	1092	1093#	1095#	
	1097#	1103	1104#	1106#	1108#	1135	1149	1150	1352	1355	1368	1371	1383	
	1386	1398	1401	1413	1416	1426	1435	1439	1449	1453	1464	1468	1478	
	1482	1492	1502	1506	1516	1520	1531	1535	1541	1551	1555	1561	1571	
	1575	1581	1591	1595	1601	1611	1615	1621	1631	1635	1641	1651	1655	
	1661	1671	1675	1681	1691	1695	1701	1713	1717	1723	1735	1739	1745	
	1755	1759	1765	1775	1779	1785	1804	1808	1816	1827	1831	1847	1852	







CROSS REFERENCE TABLE -- MACRO NAMES

SSNEWT	680#	1347	1363	1378	1393	1408	1423	1431	1460	1498	1527	1547	1567	1587	1607
	1627	1647	1667	1687	1709	1731	1751	1771	1791	1823	1837	1872	1900	1928	1939
	1950	1961	1971	1982	1994	2005	2016	2027	2037	2075	2089	2104	2344	2371	2407
	2480	2553	2626	2694	2749										
SSSET	3589#	3598	3599	3600	3603	3604	3605	3606	3607	3608	3609	3610			
SSSETH	1167#														
SSSKIP	680#														
EQUAT	525#	570													
HEADE	525#														
SETUP	525#	1120													
SACT1	525#	1088													
SAPT8	525#	873#													
SAPTH	525#	1098													
SAPTY	525#	536													
SCATC	525#														
SCMTA	525#	815													
SEOP	525#														
SERRO	525#	3159													
SERRT	525#	3215													
SPOWE	525#														
SSCOP	525#	3485													
STRAP	525#	3566													
STYPE	525#	2912													
STYPO	525#	3262													

ABS 017366 000

ERRORS DETECTED 0

DZDUQB, DZDUQB/SOL/CRF=DZDUQ1/EQ RUNA, DZDUQ2, DZDUQB  
RUN-TIME 23 14 1 SECONDS  
RUN-TIME RATIO 307/39=7 8  
CORE USED 30K (59 PAGES)