

DUV/LSI-11

COMB TIMING & INTER TEST
MD-11-DZDUU-A

EP-DZDUU-A-DL-A

APR 1977

COPYRIGHT © 1977

digital

FICHE 1 OF 1

MADE IN USA

The microfiche card displays a grid of 48 frames, arranged in 8 rows and 6 columns. Each frame contains a small, high-contrast image of technical data, likely related to timing and testing for the DUV/LSI-11 chip. The frames contain various diagrams, including waveforms, logic symbols, and data tables, which are typical of a timing and inter-test manual. The text within the frames is too small to be legible, but the overall layout suggests a comprehensive set of test procedures and timing diagrams.

11

801

EOF1DZD88RS80411
DZDUU1.M11

08000800
02-FEB-77 08:18

MA2832327(1006)

03-FEB-77 08:00

PAGE91DR1DZDUUASEQ

00010000

770323

.REM *

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

PRODUCT CODE: MAINDEC-11-DZDUU-A-D

PRODUCT NAME: DUV11 OFFLINE COMBINED TIMING & INTERRUPT TESTS

RELEASE DATE: FEB. 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 COMBINED TIMING AND INTERRUPT TESTS VERIFY THAT THE TRANSMITTER AND RECEIVER CAN CORRECTLY TALK TO EACH OTHER. INTERRUPT LOGIC AND PRIORITY LEVEL /VECTOR ADDRESSES ARE ALSO VERIFIED

* .REM *

2. REQUIREMENTS

* .REM *

PDP-11/03 COMPUTER (LSI)

DUV11 SYNCHRONOUS/ISOCRONOUS OPTION

ONE CONSOLE TELETYPE OR EQUIVALENT

2.2

STORAGE

THE PROGRAM LOADS INTO 4K OF MEMORY WITH BOOTSTRAP

3.

LOADING PROCEDURE

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

STARTING ADDRESS
FOR ABSOLUTE LOADER

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

4.

STARTING PROCEDURE

4.1

CONTROL SWITCH SETTINGS

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

NOTE: RUNNING UNDER APT-11, THERE IS A USER SWITCH REGISTER CALLED "SUSWR". IN ORDER TO BE FLEXIBLE ON THE AVAILABILITY OF THE H315 CONNECTOR, ONE BIT PASSES STATUS TO APT-11. BIT 0 IN SUSWR REFLECTS THIS STATUS, A 0 = CONNECTOR

PRESENT, A 1 = CONNECTOR NOT AVAILABLE.

- 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)
SW02=1
- NOTE1: IN GENERAL SW01 WILL BE USED WHEN SW02=1 IS USED
 NOTE2: WITHOUT SW01=1 "LOCK ON TEST" WILL DEFAULT TO TEST 1

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

4.3 PROGRAM AND/OR OPERATOR ACTION

4.3.1 INITIAL PROGRAM START

- 4.3.1.1 LOAD PROGRAM INTO MEMORY WITH ABSOLUTE LOADER
- 4.3.1.2 SET SWITCH REGISTER (LOC. 176) TO ZERO.
- 4.3.1.3 TYPE 200G.
- 4.3.1.4 PROGRAM WILL START.

4.3.1.5 THE PROGRAM WILL TYPE "DUV11 DZDUU-A TAPE E" (ONCE ONLY)

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

4.3.2 PROGRAM RESTART WITH ALL SWITCHES DOWN

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

4.3.3 PROGRAM RESTART WITH SW00=1

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

*
 * .REM *
 *
 * .REM *

4.3.3.2 TYPE 200G.

4.3.3.3 PROGRAM WILL START.

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

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

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

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

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

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

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

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

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

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

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

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

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

4.3.3.11.1 IF AN "OUT OF RANGE" ADDRESS IS TYPED
IE. MORE THAN 16 (10) DEVICES AWAY (UPWARDS).....THE
PROGRAM WILL TYPE "OUT OF RANGE:RETYPE LAST DEVICE RXCSR ADDRESS-"
AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

4.3.3.20 THE PROGRAM WILL TYPE "ARE YOU RUNNING IN MAINT.
MODE EXTERNAL ? ANDDO 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 SW02 =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 SW02=1 IN SWITCH REG. (LOC. 176)

4.3.5.2 TYPE 200G.

4.3.5.3 PROGRAM WILL START.

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

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

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

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

5. OPERATING PROCEDURE

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

SW15 =1 HALT ON ERROR
SW14 =1 LOOP ON CURRENT TEST
SW13 =1 INHIBIT ERROR TYPEOUT
SW11 =1 INHIBIT ITERATIONS
SW10 =1 ESCAPE TO NEXT TEST ON ERROR
SW09 =1 LOOP ON ERROR
SW02 =1 LOCK ON TEST
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 16XXX YYYYY ZZZZZ

WHERE 16XXX IS THE ADDRESS OF THE FAILING DEVICE REGISTER

WHERE YYYYY IS THE EXPECTED CONTENTS OF THAT REGISTER

WHERE ZZZZZZ IS THE ACTUAL CONTENTS OF THAT REGISTER

6.1.3 PC +2 = RECEIVER ERROR PC
REGISTER EXPECTED ACTUAL
16XXXX YYYYYY ZZZZZZ

WHERE 16XXXX IS THE ADDRESS OF THE FAILING RECEIVER (RXDBUF) REGISTER

WHERE YYYYYY IS THE EXPECTED DATA CONTENTS OF THAT REGISTER

WHERE ZZZZZZ IS THE ACTUAL DATA CONTENTS OF THAT REGISTER

6.1.4 PC +2 = TRANSMITTER ERROR PC
REGISTER EXPECTED ACTUAL
16XXXX YYYYYY ZZZZZZ

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

WHERE YYYYYY IS THE EXPECTED CONTENTS OF THAT REGISTER

WHERE ZZZZZZ IS THE ACTUAL CONTENTS OF THAT REGISTER

6.1.5 ERROR DESCRIPTIONS
SEE LISTINGS FOR DETAILS OF ERRORS

6.2 ERROR RECOVERY

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

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

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

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

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

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

END OF PASS TAPE Y

16XXXX = DEVICE

WHERE Y IS THE TAPE LOADED

WHERE 16XXXX IS THE DEVICE'S BASE REGISTER ADDRESS

TO INHIBIT THIS TYPEOUT - TURN TELETYPE OFF

7. RESTRICTIONS

7.1 MULTIPLE DEVICES

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

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

7.2 DISQUALIFYING DEVICES WHEN RUNNING MULTIPLE DEVICES

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

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

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

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

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

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

7.3 CABLE DELAYS

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

7.3.1 TO PROVIDE SUFFICIENT DELAY FOR CLOCK SIGNAL OVER THE CABLE, LOCATION "HOLD:" MUST BE MODIFIED TO ACCOMODATE FOR FASTER MACHINES. PRESENTLY "HOLD:" =20 IS SUFFICIENT TIME ON AN 11/03 MACHINE.

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

7.4 TO USE THE "XOR" TESTER THE BRANCH AROUND THE "XOR" CODE MUST BE PATCHED TO A "NOP". (SEE LISTINGS FOR DETAILS)

8. DEFAULT PARAMETERS:
 1ST DEVICE: RECEIVER CONTROL REGISTER ADDRESS- RXCSR: 160010
 VECTOR ADDRESS- DURIV: 770
 ARE YOU RUNNING MULTIPLE DEVICES ?- NO MULTD: 0
 LAST DEVICE: RECEIVER CONTROL REGISTER ADDRESS- LASTADD: 0
 # OF SYNC CHARS SELECTED - 2 SYNCNO: 377
 IS SEC XMIT SWITCH E55-2 ON?- YES SEXMIT: 377
 IS SEC REC SWITCH E55-3 ON?- YES SEREC: 377
 IS OPT CLR ENABLE SWITCH E55-1 ON?- YES OPTCLR: 377
 DO YOU HAVE THE EXTERNAL MODEM BYPASS JUMPER
 CONNECTOR ON (H315)- YES JMRBY: 377

9. PROGRAM DESCRIPTION

9.1 THIS PROGRAM PERFORMS THE OFFLINE COMBINED (TRANSMITTER & RECEIVER)
 TIMING & INTERRUPT TESTING OF THE DEVICE
 SEE LISTING FOR DETAILS

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

11. LISTINGS

*
 .REM *
 *
 .REM *
 *

LO1

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 12
DZDUU2.M11 02-FEB-77 08:20

000001 STN=1

000001

STN=1

MO1

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 14
 DZDUUA.M1: 13-OCT-76 08:28 APT COMMUNICATIONS ROUTINE

557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
5
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612

001100

000011
000012
000015
000200
177776

177774
177772
177570
177570

000000
000001
000002
000003
000004
000005
000006
000007
000006
000007

000000
000040
000100
000140
000200
000240
000300
000340

100000
040000

```

.ENABLE ABS

;DUV11 DZDUU-A TAPE E
;COPYRIGHT 1977, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754

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

.SBTTL BASIC DEFINITIONS

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

;#MISCELLANEOUS DEFINITIONS
MT= 11 ;;CODE FOR HORIZONTAL TAB
LF= 12 ;;CODE FOR LINE FEED
CR= 15 ;;CODE FOR CARRIAGE RETURN
CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 177776 ;;PROCESSOR STATUS WORD
.EQUIV PS,PSW
STKLMT= 177774 ;;STACK LIMIT REGISTER
PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
DSWR= 177570 ;;HARDWARE SWITCH REGISTER
DDISP= 177570 ;;HARDWARE DISPLAY REGISTER

;#GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0 ;;GENERAL REGISTER
R1= %1 ;;GENERAL REGISTER
R2= %2 ;;GENERAL REGISTER
R3= %3 ;;GENERAL REGISTER
R4= %4 ;;GENERAL REGISTER
R5= %5 ;;GENERAL REGISTER
R6= %6 ;;GENERAL REGISTER
R7= %7 ;;GENERAL REGISTER
SP= %6 ;;STACK POINTER
PC= %7 ;;PROGRAM COUNTER

;#PRIORITY LEVEL DEFINITIONS
PR0= 0 ;;PRIORITY LEVEL 0
PR1= 40 ;;PRIORITY LEVEL 1
PR2= 100 ;;PRIORITY LEVEL 2
PR3= 140 ;;PRIORITY LEVEL 3
PR4= 200 ;;PRIORITY LEVEL 4
PR5= 240 ;;PRIORITY LEVEL 5
PR6= 300 ;;PRIORITY LEVEL 6
PR7= 340 ;;PRIORITY LEVEL 7

;#"SWITCH REGISTER" SWITCH DEFINITIONS
SW15= 100000
SW14= 40000
  
```

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

:#DATA BIT DEFINITIONS (BIT00 TO BIT15)

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

:#BASIC "CPU" TRAP VECTOR ADDRESSES

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

669	000014	TBITVEC=14
670	000014	TRTVEC= 14
671	000014	BPTVEC= 14
672	000020	IOTVEC= 20
673	000024	PWRVEC= 24
674	000030	EMTVEC= 30
675	000034	TRAPVEC=34
676	000060	TKVEC= 60
677	000064	TPVEC= 64
678	000240	PIRQVEC=240

```
:"T" BIT  
:TRACE TRAP  
:BREAKPOINT TRAP (BPT)  
:INPUT/OUTPUT TRAP (IOT) **SCOPE**  
:POWER FAIL  
:EMULATOR TRAP (EMT) **ERROR**  
:"TRAP" TRAP  
:TTY KEYBOARD VECTOR  
:TTY PRINTER VECTOR  
:PROGRAM INTERRUPT REQUEST VECTOR
```

```

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

```



```

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

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 19
 DZDUUA.M11 13-OCT-76 08:28 BASIC DEFINITIONS

```

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

```

F02

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 20
DZDUUA.M11 13-OCT-76 08:28 BASIC DEFINITIONS

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

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 21
DZDUUA.M11 13-OCT-76 08:28 COMMON TAGS

814
815
816
817
818
819
820 001400
821 001400
822 001400 000000
823 001402 000
824 001403 000
825 001404 000000
826 001406 000000
827 001410 000000
828 001412 000000
829 001414 000
830 001415 001
831 001416 000000
832 001420 000000
833 001422 000000
834 001424 000000
835 001426 000000
836 001430 000000
837 001432 000000
838 001434 000
839 001435 000
840 001436 000000
841 001440 177570
842 001442 177570
843 001444 177560
844 001446 177562
845 001450 177564
846 001452 177566
847 001454 000
848 001455 002
849 001456 012
850 001457 000
851 001460 000000
852
853 001462 000000
854 001464 000000
855 001466 000000
856 001470 000000
857 001472 000000
858 001474 000000
859 001476 000000
860 001500 000000
861 001502 000000
862 001504 000000
863 001506 000000
864 001510 000000
865 001512 000000
866 001514 000000
867 001516 177607 000377
868 001522 077
869 001523 015

.SBTTL COMMON TAGS

*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
*USED IN THE PROGRAM.

SCMTAG: .=. ; ; START OF COMMON TAGS
\$STNM: .WORD 0 ; ; CONTAINS THE TEST NUMBER
\$ERFLG: .BYTE 00 ; ; CONTAINS ERROR FLAG
\$ICNT: .WORD 00 ; ; CONTAINS SUBTEST ITERATION COUNT
\$LPADR: .WORD 00 ; ; CONTAINS SC & LOOP ADDRESS
\$LPERR: .WORD 00 ; ; CONTAINS SCOPE RETURN FOR ERRORS
\$ERTTL: .WORD 00 ; ; CONTAINS TOTAL ERRORS DETECTED
\$ITEMB: .BYTE 00 ; ; CONTAINS ITEM CONTROL BYTE
\$ERMAX: .BYTE 1 ; ; CONTAINS MAX. ERRORS PER TEST
\$ERRPC: .WORD 00 ; ; CONTAINS PC OF LAST ERROR INSTRUCTION
\$GDADR: .WORD 00 ; ; CONTAINS ADDRESS OF 'GOOD' DATA
\$BDADR: .WORD 00 ; ; CONTAINS ADDRESS OF 'BAD' DATA
\$GDAT: .WORD 00 ; ; CONTAINS 'GOOD' DATA
\$BDAT: .WORD 00 ; ; CONTAINS 'BAD' DATA
 ; ; RESERVED--NOT TO BE USED
\$AUTOB: .BYTE 00 ; ; AUTOMATIC MODE INDICATOR
\$INTAG: .BYTE 00 ; ; INTERRUPT MODE INDICATOR
\$SWR: .WORD DSWR ; ; ADDRESS OF SWITCH REGISTER
\$DISPLAY: .WORD DDISP ; ; ADDRESS OF DISPLAY REGISTER
\$TKS: 177560 ; ; TTY KBD STATUS
\$TKB: 177562 ; ; TTY KBD BUFFER
\$TPS: 177564 ; ; TTY PRINTER STATUS REG. ADDRESS
\$TPB: 177566 ; ; TTY PRINTER BUFFER REG. ADDRESS
\$NULL: .BYTE 0 ; ; CONTAINS NULL CHARACTER FOR FILLS
\$FILLS: .BYTE 2 ; ; CONTAINS # OF FILLER CHARACTERS REQUIRED
\$FILLC: .BYTE 12 ; ; INSERT FILL CHARS. AFTER A "LINE FEED"
\$STPFLG: .BYTE 0 ; ; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
\$REGAD: .WORD 0 ; ; CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED
\$REGO: .WORD 0 ; ; CONTAINS ((\$REGAD)+0)
\$REG1: .WORD 00 ; ; CONTAINS ((\$REGAD)+2)
\$REG2: .WORD 00 ; ; CONTAINS ((\$REGAD)+4)
\$REG3: .WORD 00 ; ; CONTAINS ((\$REGAD)+6)
\$REG4: .WORD 00 ; ; CONTAINS ((\$REGAD)+10)
\$REG5: .WORD 00 ; ; CONTAINS ((\$REGAD)+12)
\$TMP0: .WORD 00 ; ; USER DEFINED
\$TMP1: .WORD 00 ; ; USER DEFINED
\$TMP2: .WORD 00 ; ; USER DEFINED
\$TMP3: .WORD 00 ; ; USER DEFINED
\$TMP4: .WORD 00 ; ; USER DEFINED
\$TMP5: .WORD 00 ; ; USER DEFINED
\$TIMES: 0 ; ; MAX. NUMBER OF ITERATIONS
\$ESCAPE: 0 ; ; ESCAPE ON ERROR ADDRESS
\$BELL: .ASCIZ <207><377><377> ; ; CODE FOR BELL
\$QUES: .ASCII /?/ ; ; QUESTION MARK
\$CRLF: .ASCII <15> ; ; CARRIAGE RETURN

H02

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 22
DZDUUA.M11 13-OCT-76 08:28 COMMON TAGS

870 001524 000012
871
872
873
874
875
876 001526
877 001526 000000
878 001530 000000
879 001532 000000
880 001534 000000
881 001536 000000
882 001540 000000
883 001542 000000
884 001544 000000
885 001546
886 001546 000
887 001547 000
888 001550 000000
889 001552 000000
890 001554 000000
891
892
893
894
895
896
897 001556 000
898 001557 000
899
900
901
902
903 001560 000000
904
905 001562 000
906 001563 000
907 001564 000000
908 001566 000
909 001567 000
910 001570 000000
911 001572 000
912 001573 000
913 001574 000000
914 001576 000000
915 001600 000000
916 001602 000000
917 001604 000000
918 001606 000000
919 001610 000000
920 001612 000000
921 001614 000000
922 001616 000000
923 001620 000000
924 001622 000000
925 001624 000000

\$LF: .ASCIZ <12> ;:LINE FEED
:*****
:SBTTL APT MAILBOX-ETABLE
:*****
:EVEN
\$MAIL: ;: APT MAILBOX
\$MSGTY: .WORD AMSGTY ;: MESSAGE TYPE CODE
\$FATAL: .WORD AFATAL ;: FATAL ERROR NUMBER
\$TESTN: .WORD ATESTN ;: TEST NUMBER
\$PASS: .WORD APASS ;: PASS COUNT
\$DEVCT: .WORD ADEVCT ;: DEVICE COUNT
\$UNIT: .WORD AUNIT ;: I/O UNIT NUMBER
\$MSGAD: .WORD AMSGAD ;: MESSAGE ADDRESS
\$MSGLG: .WORD AMGLG ;: MESSAGE LENGTH
\$ETABLE: ;: APT ENVIRONMENT TABLE
\$ENV: .BYTE AENV ;: ENVIRONMENT BYTE
\$ENVM: .BYTE AENVM ;: ENVIRONMENT MODE BITS
\$SWREG: .WORD ASWREG ;: APT SWITCH REGISTER
\$USWR: .WORD AUSWR ;: USER SWITCHES
\$CPUOP: .WORD ACPUOP ;: CPU TYPE, OPTIONS
BITS 15-11=CPU TYPE
11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
11/70=06, PDQ=07, Q=10
BIT 10=REAL TIME CLOCK
BIT 9=FLOATING POINT PROCESSOR
BIT 8=MEMORY MANAGEMENT
\$MAMS1: .BYTE AMAMS1 ;: HIGH ADDRESS, M.S. BYTE
\$MTYP1: .BYTE AMTYP1 ;: MEM. TYPE, BLK#1
MEM. TYPE BYTE -- (HIGH BYTE)
900 NSEC CORE=001
300 NSEC BIPOLAR=002
500 NSEC MOS=003
\$MADR1: .WORD AMADR1 ;: HIGH ADDRESS, BLK#1
MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
\$MAMS2: .BYTE AMAMS2 ;: HIGH ADDRESS, M.S. BYTE
\$MTYP2: .BYTE AMTYP2 ;: MEM. TYPE, BLK#2
\$MADR2: .WORD AMADR2 ;: MEM. LAST ADDRESS, BLK#2
\$MAMS3: .BYTE AMAMS3 ;: HIGH ADDRESS, M.S. BYTE
\$MTYP3: .BYTE AMTYP3 ;: MEM. TYPE, BLK#3
\$MADR3: .WORD AMADR3 ;: MEM. LAST ADDRESS, BLK#3
\$MAMS4: .BYTE AMAMS4 ;: HIGH ADDRESS, M.S. BYTE
\$MTYP4: .BYTE AMTYP4 ;: MEM. TYPE, BLK#4
\$MADR4: .WORD AMADR4 ;: MEM. LAST ADDRESS, BLK#4
\$VECT1: .WORD AVECT1 ;: INTERRUPT VECTOR#1, BUS PRIORITY#1
\$VECT2: .WORD AVECT2 ;: INTERRUPT VECTOR#2, BUS PRIORITY#2
\$BASE: .WORD ABASE ;: BASE ADDRESS OF EQUIPMENT UNDER TEST
\$DEVN: .WORD ADEVN ;: DEVICE MAP
\$CDW1: .WORD ACDW1 ;: CONTROLLER DESCRIPTION WORD#1
\$CDW2: .WORD ACDW2 ;: CONTROLLER DESCRIPTION WORD#2
\$DDW0: .WORD ADDW0 ;: DEVICE DESCRIPTOR WORD#0
\$DDW1: .WORD ADDW1 ;: DEVICE DESCRIPTOR WORD#1
\$DDW2: .WORD ADDW2 ;: DEVICE DESCRIPTOR WORD#2
\$DDW3: .WORD ADDW3 ;: DEVICE DESCRIPTOR WORD#3
\$DDW4: .WORD ADDW4 ;: DEVICE DESCRIPTOR WORD#4
\$DDW5: .WORD ADDW5 ;: DEVICE DESCRIPTOR WORD#5


```

942
943
944
945           ; INSTRUCTION DEFINITIONS
946
947           005746   PUSH1SP=5746   ; DECREMENT PROCESSOR STACK 1 WORD =TST -(SP)
948           005726   POP1SP=5726   ; INCREMENT PROCESSOR STACK 1 WORD =TST (SP)+
949           010046   PUSHRO=10046  ; SAVE RO ON STACK =MOV RO, -(SP)
950           012600   POPRO=12600   ; RESTORE RO FROM STACK =MOV (SP)+, RO
951           024646   PUSH2SP=24646 ; DECREMENT STACK TWICE =CMP -(SP), -(SP)
952           022626   POP2SP=22626  ; INCREMENT STACK TWICE =CMP (SP)+, (SP)+
953           ; REGISTER DEFINITIONS
954           ; RXCSR BIT DEFINITIONS
955           100000   DSC=BIT15   ; DATA SET CHANGE
956           040000   RING=BIT14   ; RING
957           020000   CTS=BIT13   ; CLR TO SEND
958           010000   CARDET=BIT12  ; CARRIER DETECT
959           004000   REACT=BIT11  ; REC ACTIVE
960           002000   SRO=BIT10   ; SEC REC DATA
961           001000   DSR=BIT9    ; DATA SET RDY
962           000400   STPSYN=BIT8  ; STRIP SYNC
963           000200   RXDONE=BIT7  ; REC DONE
964           000100   RINTEN=BIT6  ; REC INTR ENABLE
965           000040   DSINTE=BIT5  ; DSC INTR ENABLE
966           000020   SYNSCH=BIT4  ; SYNC SEARCH
967           000010   STD=BIT3    ; SEC XMIT DATA
968           000004   RTS=BIT2    ; REQ TO SEND
969           000002   DTR=BIT1    ; DATA TERM RDY
970           000001   VOID=BIT0
971           ; RXDBUF BIT DEFINITIONS
972           100000   RXERR=BIT15  ; REC ERROR
973           040000   OVERRUN=BIT14 ; OVERRUN
974           020000   FRMERR=BIT13  ; FRAME ERROR
975           010000   PARER=BIT12  ; PARITY ERROR
976           ; PARCSR BIT DEFINITIONS
977           001000   PAREN=BIT9    ; PARITY ENABLE
978           000400   EVPAR=BIT8   ; EVEN PARITY SENSE
979           ; PARCSR WRD DEFINITIONS
980           030000   SYNINT=30000  ; SYNC EXTERNAL MODE
981           020000   SYNEXT=20000  ; SYNC INTERNAL MODE
982           000000   ISYMOD=0     ; ISOC MODE
983           000000   FIVE=0       ; WORD LENGTH 5 BITS
984           002000   SIX=2000     ; WORD LENGTH 6 BITS
985           004000   SEVEN=4000   ; WORD LENGTH 7 BITS
986           006000   EIGHT=6000  ; WORD LENGTH 8 BITS
987           000000   NOPAR=0     ; NO PARITY
988           001000   ODDPAR=1000  ; ODD PARITY
989           001400   EVEPAR=1400  ; EVEN PARITY
990           ; TXCSR BIT DEFINITIONS
991           100000   DNA=BIT15    ; DATA NOT AVAILABLE
992           040000   MTDATA=BIT14  ; MAINT DATA
993           020000   CLK=BIT13    ; CLK
994           002000   BITW=BIT10   ; BIT WINDOW
995           000400   MRESET=BIT8   ; MASTER RESET
996           000200   TXDONE=BIT7  ; XMIT DONE
997           000100   TXINTE=BIT6  ; XMIT INTR ENABLE
    
```

K02

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 25
DZDUUA.M11 13-OCT-76 08:28 APT MAILBOX-ETABLE

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

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

001652

001652 001762
001654 002067
001656 002116
001660 002132
001662 002022
001664 002067
001666 002116
001670 002132
001672 002043
001674 002067
001676 002116
001700 002132
001702 001746
001704 000000
001706 002126
001710 002132

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

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

;DEFAULT DU ADDRESSES
RXCSR: 160010
HRXCSR: 160011
RXDBUF: 160012
HRXDBUF: 160013
PARCSR: 160012
HPARCSR: 160013
TXCSR: 160014
HTXCSR: 160015
TXDBUF: 160016
HTXDBUF: 160017
;DEFAULT DU VECTORS
DURIV: 770 ;REC INTR VECTOR
DURIS: 772 ;REC INTR STATUS
DUTIV: 774 ;XMIT INTR VECTOR
DUTIS: 776 ;XMIT INTR STATUS

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

M02

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 27
 DZDUUA.M11 13-OCT-76 08:28 ERROR POINTER TABLE

1063	002012	044507	052123	051105	
1064	002020	000123			
1065	002022	020040	042522	042503	EM2: .ASCIZ / RECEIVER ERROR/
1066	002030	053111	051105	042440	
1067	002036	051122	051117	000	
1068	002043	040	052040	040522	EM3: .ASCIZ / TRANSMITTER ERROR/
1069	002050	051516	044515	052124	
1070	002056	051105	042440	051122	
1071	002064	051117	000		
1072					;DATA HEADERS FOR ERROR MESSAGES
1073	002067	105	051122	041520	DH1: .ASCIZ /ERRPC WANTED ACTUAL/
1074	002074	020040	040527	052116	
1075	002102	042105	020040	041501	
1076	002110	052524	046101	000	
1077		002116			.EVEN
1078					;DATA TABLES FOR ERROR MESSAGES
1079	002116	001416	001130	001132	DT1: .WORD \$ERRPC,HLDO,HLDI,0
1080	002124	000000			
1081					
1082	002126	001416	000000		DT4: .WORD \$ERRPC,0
1083					
1084	002132	000	000	000	DF1: .BYTE 0,0,0,0
1085	002135	000			
1086					.EVEN
1087					.SBTTL ACT11 HOOKS
1088					
1089					::*****
1090					;HOOKS REQUIRED BY ACT11
1091		002136			\$SVPC= ;SAVE PC
1092		000046			.=46
1093	000046	012702			\$ENDAD ;;1)SET LOC.46 TO ADDRESS OF \$ENDAD IN .SEOP
1094		000052			.=52
1095	000052	000000			.WORD 0 ;;2)SET LOC.52 TO ZERO
1096		002136			.\$SVPC ;; RESTORE PC
1097					.SBTTL APT PARAMETER BLOCK
1098					
1099					::*****
1100					;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1101					::*****
1102		002136			.\$X= ;;SAVE CURRENT LOCATION
1103		000024			.=24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1104	000024	000200			200 ;;FOR APT START UP
1105		000044			.=44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
1106	000044	002136			\$APTHDR ;;POINT TO APT HEADER BLOCK
1107		002136			.\$X ;;RESET LOCATION COUNTER
1108					::*****
1109					;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1110					;INTERFACE SPEC.
1111					
1112	002136				\$APTHD:
1113	002136	000000			\$:BITS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1114	002140	001526			\$MBAOR: .WORD \$MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1115	002142	000010			\$STMT: .WORD 10 ;;RUN TIM OF LONGEST TEST
1116	002144	000010			\$PASTM: .WORD 10 ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1117	002146	000000			\$UNITM: .WORD ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1118	002150	000052			.WORD \$ETEND-\$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)

```

1119
1120
1121          : PROGRAM INITIALIZATION
1122          : LOCK OUT INTERRUPTS
1123          : SET UP PROCESSOR STACK
1124          : SET UP POWER FAIL VECTOR
1125          : CLEAR PROGRAM CONTROL FLAGS AND COUNTS
1126          : TYPE TITLE MESSAGE
1127
1128 002152    .START:
1129          .SBTTL INITIALIZE THE COMMON TAGS
1130          ;; CLEAR THE COMMON TAGS (%SCMTAG) AREA
1131 002152 012706 001400    MOV    %SCMTAG,%R6    ;; FIRST LOCATION TO BE CLEARED
1132 002156 005026          CLR    (R6)+          ;; CLEAR MEMORY LOCATION
1133 002160 022706 001440    CMP    %SWR,%R6    ;; DONE?
1134 002164 001374          BNE    -6            ;; LOOP BACK IF NO
1135 002166 012706 001100    MOV    %%STACK,SP    ;; SETUP THE STACK POINTER
1136          ;; INITIALIZE A FEW VECTORS
1137 002172 012737 016332 000020    MOV    %%SCOPE,%IOTVEC    ;; IOT VECTOR FOR SCOPE ROUTINE
1138 002700 012737 000340 000022    MOV    %340,%IOTVEC+2    ;; LEVEL 7
1139 002206 012737 014222 000030    MOV    %%ERROR,%EMTVEC    ;; EMT VECTOR FOR ERROR ROUTINE
1140 002214 012737 000340 000032    MOV    %340,%EMTVEC+2    ;; LEVEL 7
1141 002222 012737 016650 000034    MOV    %%TRAP,%TRAPVEC    ;; TRAP VECTOR FOR TRAP CALLS
1142 002230 012737 000340 000036    MOV    %340,%TRAPVEC+2    ;; LEVEL 7
1143 002236 012737 015024 000024    MOV    %%SPWRON,%PWRVEC    ;; POWER FAILURE VECTOR
1144 002244 012737 000340 000026    MOV    %340,%PWRVEC+2    ;; LEVEL 7
1145 002252 005067 177234          CLR    $TIMES          ;; INITIALIZE NUMBER OF ITERATIONS
1146 002256 005067 177232          CLR    $ESCAPE        ;; CLEAR THE ESCAPE ON ERROR ADDRESS
1147 002262 112767 000001 177125    MOVB  %1,$ERMAX        ;; ALLOW ONE ERROR PER TEST
1148 002270 012767 002270 177110    MOV    %.,$LPAOR        ;; INITIALIZE THE LOOP ADDRESS FOR SCOPE
1149 002276 012767 002276 177104    MOV    %.,$LPERR        ;; SETUP THE ERROR LOOP ADDRESS
1150          ;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1151          ;; EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1152 002304 013746 000004          MOV    %ERRVEC,-(SP)    ;; SAVE ERROR VECTOR
1153 002310 012737 002344 000004    MOV    %1,$ERRVEC      ;; SET UP ERROR VECTOR
1154 002316 012767 177570 177114    MOV    %DSWR,$SWR      ;; SETUP FOR A HARDWARE SWICH REGISTER
1155 002324 012767 177570 177110    MOV    %DDISP,$DISPLAY  ;; AND A HARDWARE DISPLAY REGISTER
1156 002332 022777 177777 177100    CMP    %-1,$SWR        ;; TRY TO REFERENCE HARDWARE SWR
1157 002340 001012          BNE    66$            ;; BRANCH IF NO TIMEOUT TRAP OCCURRED
1158          ;; AND THE HARDWARE SWR IS NOT = -1
1159 002342 000403          BR    65$            ;; BRANCH IF NO TIMEOUT
1160 002344 012716 002352 64$:    MOV    %65$,(SP)      ;; SET UP FOR TRAP RETURN
1161 002350 000002          RTI
1162 002352 012767 000176 177060 65$:    MOV    %%SWREG,$SWR    ;; POINT TO SOFTWARE SWR
1163 002360 012767 000174 177054    MOV    %%DISPREG,$DISPLAY
1164 002366 012637 000004 66$:    MOV    (SP)+,%ERRVEC    ;; RESTORE ERROR VECTOR
1165
1166 002372 005067 177136          CLR    $PASS          ;; CLEAR PASS COUNT
1167 002376 132767 000200 177143    BITB  %APTSIZE,$ENVM    ;; TEST USER SIZE UNDER APT
1168 002404 001403          BEQ    67$            ;; YES, USE NON-APT SWITCH
1169 002406 012767 001550 177024    MOV    %%SSWREG,$SWR   ;; NO, USE APT SWITCH REGISTER
1170 002414
1171 002414 012706 001100          MOV    %%STACK,SP      ;; SET STACK
1172 002420 106427 000340          MTPS  %340            ;; LOCK INTERRUPTS
1173 002424 012737 015024 000024    MOV    %PFAIL,%24      ;; SET UP POWER FAIL VECTOR
1174 002432 105067 176535          CLRB  $STFLG          ;; CLEAR START FLAG

```

```

1175 002436 005067 176450 CLR PASCNT ;CLEAR PASS COUNT
1176 002442 105067 176735 CLRB SERFLG ;CLEAR ERROR FLAG
1177 002446 005067 176740 CLR SERTTL ;CLEAR ERROR COUNT
1178 002452 005067 176740 CLR SERRPC ;CLEAR LAST ERROR POINTER
1179 002456 012767 000001 176716 MOV #1,STSTNM ;SET UP FOR TEST 1
1180 002464 012767 002152 176412 MOV #.START,RETURN ;SET JP FOR POWER FAIL BEFORE
1181 ;TESTING STARTS
1182 002472 013746 000006 MOV #6,-(SP)
1183 002476 013746 000004 MOV #4,-(SP)
1184 002502 012737 002516 000004 MOV #15,#4
1185 002510 005777 176724 TST #SWR
1186 002514 000407 BR 25
1187 002516 012767 000176 176714 15: MOV #SWREG,SWR
1188 002524 012767 000174 176710 MOV #DISPREG,DISPLAY
1189 002532 022626 CMP (SP)+,(SP)+
1190 002534 012637 000004 25: MOV (SP)+,#4
1191 002540 012637 000006 MOV (SP)+,#6
1192 002544 022767 000176 176666 CMP #SWREG,SWR
1193 002552 001007 BNE 35
1194 002554 005737 000042 TST #42 ;CHECK FOR CHAIN
1195 002560 001402 BEQ 335
1196 002562 000167 000522 JMP .BEGIN
1197 002566 004767 010216 335: JSR PC,CNTLU
1198 002572 105767 176374 35: TSTB INIFLG ;HAS INITIALIZATION BEEN PERFORMED
1199 002576 001004 BNE ONCE
1200 002600 104401 015164 TYPE #TITLE ;TYPE TITLE MESSAGE
1201 002604 105167 176362 COMB INIFLG ;IF NOT SET FLAG AND DO
1202 002610 105767 176732 ONCE: TSTB #ENV ;APT CONTROL?
1203 002614 001410 BEQ 115 ;BR IF NO
1204 002616 032767 000001 176726 BIT #1,SUSWR ;EXTERNAL JUMPER ON?
1205 002624 001002 BNE 125 ;NO
1206 002626 105067 176321 CLRB JMRBY ;CLEAR FLAG
1207 002632 000167 000452 125: JMP .BEGIN ;GO DO IT
1208 002636 032777 000001 176574 115: BIT #SW00,#SWR ;RESELECT VECTOR & CONTROL REG?
1209 002644 001002 BNE 15
1210 002646 000167 000436 JMP .BEGIN
1211 002652 012700 000300 15: MOV #300,R0 ;RESTORE VECTOR AREA TO TRAPCATCHER
1212 002656 012701 000302 MOV #302,R1 ;START AT LOCATION 300
1213 002662 012702 000004 MOV #4,R2
1214 002666 010110 25: MOV R1,(R0)
1215 002670 005011 CLR (R1)
1216 002672 060200 ADD R2,R0
1217 002674 060201 ADD R2,R1
1218 002676 022701 001000 CMP #1000,R1 ;END AT LOCATION 776
1219 002702 002771 BLT 25
1220 002704 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1221 002706 015232 MREGAD ;MESSAGE
1222 002710 104410 PARAM ;CONVERT STRING
1223 002712 160000 160000 ;LOW LIMIT
1224 002714 167776 167776 ;HIGH LIMIT
1225 002716 017144 DUBASE ;STORE AT THIS LOCATION
1226 002720 001 .BYTE 1 ;MASK
1227 002721 001 .BYTE 1 ;HOW MANY TIMES + 2
1228 002722 016767 014216 176226 MOV DUBASE,KEPPAD ;SAVE
1229 002730 004767 014056 JSR PC,DUADDR
1230 002734 016767 176216 176212 MOV KEPPAD,BASEADD ;RESTORE FOR ROTATION
    
```

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 30
 DZDUUA.M11 13-OCT-76 08:28 INITIALIZE THE COMMON TAGS

1231	002742	104406				IMSTR	: OUTPUT MESSAGE & GET INPUT STRING
1232	002744	015217				MVECTO	: MESSAGE
1233	002746	104410				PARAM	: CONVERT STRING
1234	002750	000300				300	: LOW LIMIT
1235	002752	000776				776	: HIGH LIMIT
1236	002754	001736				DURIV	: STORE AT THIS LOCATION
1237	002756	001			.BYTE	1	: MASK
1238	002757	004			.BYTE	4	: HOW MANY TIMES + 2
1239	002760	016767	176752	176176		MOV	DURIV,KEEPIV ;SAVE
1240	002766	016767	176744	176166		MOV	DURIV,BASEIV ;SET UP FOR ROTATION
1241	002774	104406				IMSTR	: OUTPUT MESSAGE & GET INPUT STRING
1242	002776	015262				MMULT	: MESSAGE
1243	003000	104414				SETFLG	: SET FLAG BASED UPON INPUT STRING
1244	003002	001152				MULTD	: THIS FLAG
1245	003004	105767	176142			TSTB	MULTD ;ARE THERE MULTIPLE DEVICES
1246							: ON THE SYSTEM ?
1247	003010	100406				BMI	B88 ;YES,ASK NEXT QUESTION
1248	003012	005067	176150			CLR	ACTREG
1249	003016	005067	176146			CLR	ROTADD
1250	003022	000167	000140			JMP	OUTMUL ;JUMP AROUND NEXT QUESTION
1251	003026				B88:		
1252	003026	104406				IMSTR	: OUTPUT MESSAGE & GET INPUT STRING
1253	003030	015311				MLASTD	: MESSAGE
1254	003032	104410				PARAM	: CONVERT STRING
1255	003034	160000				160000	: LOW LIMIT
1256	003036	167776				167776	: HIGH LIMIT
1257	003040	001160				LASTADD	: STORE AT THIS LOCATION
1258	003042	001			.BYTE	1	: MASK
1259	003043	001			.BYTE	1	: HOW MANY TIMES + 2
1260							: THE FOLLOWING ROUTINE SETS UP ACTREG FOR THE FIRST TIME
1261	003044	012767	000001	176116	1\$:	MOV	#1,ROTADD ;SET UP POINTER
1262	003052	005067	176110			CLR	ACTREG ;CLR ACTIVE REGISTER
1263	003056	056767	176106	176102	2\$:	BIS	ROTADD,ACTREG ;MAKE THIS DEVICE ACTIVE
1264	003064	000241				CLC	
1265	003066	006167	176076			ROL	ROTADD ;SET UP POINTER
1266	003072	103421				BCS	3\$;ARE YOU OUT OF RANGE ?
1267	003074	062767	000010	176052		ADD	#10,BASEADD ;SET UP BASE ADDRESS
1268	003102	026767	176052	176044		CMP	LASTADD,BASEADD ;IS THIS THE LAST DEVICE ?
1269	003110	101362				BHI	2\$;NO DO IT AGAIN
1270	003112	056767	176052	176046		BIS	ROTADD,ACTREG ;THIS ASSUMES THAT THERE ARE AT
1271							: LEAST TWO DEVICES WHEN YOU ANSWER YES TO
1272							: MULTIPLE DEVICE QUESTION
1273	003120	012767	000001	176042	4\$:	MOV	#1,ROTADD ;SET UP FOR LATER USE IN END OF PASS ROUTINE
1274	003126	016767	176024	176020		MOV	KEEPADD,BASEADD ;DITTO
1275	003134	000414				BR	OUTMUL ;CONTINUE QUESTIONS
1276	003136	016767	176014	176010	3\$:	MOV	KEEPADD,BASEADD ;RESTORE
1277	003144	104406				IMSTR	: OUTPUT MESSAGE & GET INPUT STRING
1278	003146	015405				MRANGE	: MESSAGE
1279	003150	104410				PARAM	: CONVERT STRING
1280	003152	160000				160000	: LOW LIMIT
1281	003154	167776				167776	: HIGH LIMIT
1282	003156	001160				LASTADD	: STORE AT THIS LOCATION
1283	003160	001			.BYTE	1	: MASK
1284	003161	001			.BYTE	1	: HOW MANY TIMES + 2
1285	003162	000167	177656			JMP	1\$;DO IT AGAIN
1286	003166	012767	000340	013612	OUTMUL:	MOV	#340,DUPRT

```

1287 003174 004767 013536 JSR PC,DULEV
1288 ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
1289 ;BUFFER TO THE CHARACTERS "1" AND "2"
1290 ;IF THE CHARACTER IS "1" CLEAR THE FLAG
1291 ;IF THE CHARACTER IS "2" SET THE FLAG
1292 003200 AAA:
1293 003200 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1294 003202 015623 MSYNC ;MESSAGE
1295 003204 122767 00006: 012752 3$: CMPB #'1,INBUF ;IS IT "1" ?
1296 003212 001003 BNE 1$
1297 003214 105067 175726 CLRB SYNCNO ;000
1298 003220 000412 BR 4$
1299 003222 122767 000062 012734 1$: CMPB #'2,INBUF ;IS IT "2" ?
1300 003230 001004 BNE 2$
1301 003232 112767 177777 175706 MOVB #-1,SYNCNO ;377
1302 003240 000402 BR 4$
1303 003242 104407 2$: INSTR ;RETRY
1304 003244 000757 BR 3$
1305 003246 000240 4$: NOP
1306 003250 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1307 003252 015671 MWIRE6 ;MESSAGE
1308 003254 104414 SETFLG ;SET FLAG BASED UPON INPUT STRING
1309 003256 001147 SEXMIT ;THIS FLAG
1310 003260 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1311 003262 015742 MWIRE5 ;MESSAGE
1312 003264 104414 SETFLG ;SET FLAG BASED UPON INPUT STRING
1313 003266 001150 SEREC ;THIS FLAG
1314 003270 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1315 003272 016012 MWIRE4 ;MESSAGE
1316 003274 104414 SETFLG ;SET FLAG BASED UPON INPUT STRING
1317 003276 001151 OPTCLR ;THIS FLAG
1318 003300 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1319 003302 016071 MEXTJ ;MESSAGE
1320 003304 104414 SETFLG ;SET FLAG BASED UPON INPUT STRING
1321 003306 001153 JMRBY ;THIS FLAG
1322
1323 ;TEST START AND RESTART
1324
1325 003310 012706 001100 .BEGIN: MOV #STACK,SP ;SET UP STACK
1326 003314 106427 000340 MTPS #340 ;LOCK OUT INTERRUPTS
1327 003320 032777 000002 176112 BIT #SW01,DSWR ;IF SW01=1, GET STARTING PC
1328 003326 001406 BEQ 3$
1329 003330 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1330 003332 015555 MTSTPC ;MESSAGE
1331 003334 104410 PARAM ;CONVERT STRING
1332 003336 003362 TST1 ;LOW LIMIT
1333 ;HIGH LIMIT
1334 ;STORE AT THIS LOCATION
1335 003340 001 .BYTE 1 ;MASK
1336 003341 001 .BYTE 1 ;HOW MANY TIMES + 2
1337 003342 000403 BR 4$
1338 003344 012767 003362 175532 3$: MOV #TST1,RETURN ;START AT TEST 1
1339 003352 104401 015551 4$: TYPE R ;TYPE R
1340 003356 000177 175522 JMP #RETURN ;START TESTING
1341
1342 ;;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION

```

INITIALIZE THE COMMON TAGS

1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398

```

003362 000004
003364 052777 000400 176334
003372 012777 000000 176322
003400 052777 000400 176320

003406 012777 004020 176312

003414 012777 005426 176300
003422 016703 176300
003426 112777 000125 176276
003434 012767 001252 176036
003442 012767 000012 175452

003450 052777 020000 176250
003456 042777 020000 176242
003464 005000
003466 006067 176006
003472 103002
003474 052700 002000

003500
003500 052777 020000 176220
003506 042777 020000 176212
003514 017701 176206
003520 042701 075777
003524 020001
003526 001401
003530 104003

003532 005367 175364
003536 001352

003540 052777 020000 176160
003546 012700 000000
003552 017701 176150
003556 042701 077777
003562 020001
003564 001401
003566 104003
    
```

```

; OF THE TRANSMITTER SECTION.
; IT ALSO CHECKS DNA TIMING
; MODE: ISYMOD
; LENGTH: SEVEN PLUS PARITY
; PARITY: EVEPAR
; CHARACTER: 125
; *****
†ST1: SCOPE
      BIS      #MRESET,@TXCSR ; MASTER RESET
      MOV      #ISYMOD,@PARCSR ; SET THE MODE
      BIS      #MRESET,@TXCSR ; MASTER RESET

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

; SET MODE # OF BITS, PARITY SENSE, & LOAD SYNC REG
      MOV      #ISYMOD!SEVEN!EVEPAR!26,@PARCSR
      MOV      TXCSR,R3 ; SET UP FOR ERROR MSG
      MOV      #125,@TXDBUF ; LOAD DATA CHAR
      MOV      #1252,$TMP1 ; TO BE SHIFTED CHAR
      MOV      #10,$SHIFT ; # OF SHIFTS

; POKE CLK TO GET INTO SYNCHRONIZATION
      BIS      #CLK,@TXCSR ; POKE CLK UP
      BIC      #CLK,@TXCSR ; POKE CLK DOWN
1$:   CLR      R0
      ROR      $TMP1 ; FORCE CARRY
      BCC     2$ ; BR IF CARRY CLR
      BIS      #BITW,R0 ; EQUIV OF BITW
2$:   BIS      #CLK,@TXCSR ; POKE CLK UP
      BIC      #CLK,@TXCSR ; POKE CLK DOWN
      MOV      @TXCSR,R1 ; ACTUAL
      BIC      #075777,R1 ; SAVE BITW & DNA
      CMP     R0,R1 ; COMPARE EXP VS ACT
      BEQ     +4
      ERROR   3 ; BIT WINDOW DID NOT MATCH ACTUAL DATA
; BIT ... ALSO CHECK DNA
; # OF SHIFTS
; DO IT AGAIN ?
      DEC     $SHIFT
      BNE     1$

; NOW POKE CLK TO SEE DNA
      BIS      #CLK,@TXCSR ; POKE CLK
      MOV      #0,R0 ; EXPECTED
      MOV      @TXCSR,R1 ; ACTUAL
      BIC      #77777,R1 ; SAVE DNA ONLY
      CMP     R0,R1 ; COMPARE EXP VS ACT
      BEQ     +4
      ERROR   3 ; DNA SHOULD BE SET
; IF DNA DID NOT SET
; CHECK WORD LENGTH SELECT LOGIC

; THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
; OF THE TRANSMITTER SECTION.
; IT ALSO CHECKS DNA TIMING
; MODE: ISYMOD
    
```

F03

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 33
 DZDUUA.M11 13-OCT-76 08:28 INITIALIZE THE COMMON TAGS

```

1399          ;:LENGTH:SEVEN PLUS PARITY
1400          ;:PARITY:ODDPAR
1401          ;:CHARACTER:125
1402          ;:*****
1403          ;:*****
1404 003570 000004          †ST2: SCOPE
1405 003572 052777 000400 176126      BIS      #MRESET,@TXCSR ;MASTER RESET
1406 003600 012777 000000 176114      MOV      #ISYMOD,@PARCSR ;SET THE MODE
1407 003606 052777 000400 176112      BIS      #MRESET,@TXCSR ;MASTER RESET
1408
1409          ;SET MAINTENANCE MODE & SEND
1410          ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1411 003614 012777 004020 176104      MOV      #MINT!SEND,@TXCSR
1412
1413          ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
1414 003622 012777 005026 176072      MOV      #ISYMOD!SEVEN!ODDPAR!26,@PARCSR
1415 003630 016703 176072              MOV      TXCSR,R3 ;SET UP FOR ERROR MSG
1416 003634 112777 000125 176070      MOV      #125,@TXDBUF ;LOAD DATA CHAR
1417 003642 012767 001652 175630      MOV      #1652,$TMP1 ;TO BE SHIFTED CHAR
1418 003650 012767 000012 175244      MOV      #10,SHIFT ;# OF SHIFTS
1419          ;POKE CLK TO GET INTO SYNCRONIZATION
1420 003656 052777 020000 176042      BIS      #CLK,@TXCSR ;POKE CLK UP
1421 003664 042777 020000 176034      BIC      #CLK,@TXCSR ;POKE CLK DOWN
1422 003672 005000
1423 003674 006067 175600          1$: CLR      RO
1424 003700 103002              ROR      $TMP1 ;FORCE CARRY
1425 003702 052700 002000          BCC      2$ ;BR IF CARRY CLR
1426 003706
1427 003706 052777 020000 176012      BIS      #CLK,@TXCSR ;POKE CLK UP
1428 003714 042777 020000 176004      BIC      #CLK,@TXCSR ;POKE CLK DOWN
1429 003722 017701 176000          MOV      @TXCSR,R1 ;ACTUAL
1430 003726 042701 075777          BIC      #075777,R1 ;SAVE BITW & DNA
1431 003732 020001          CMP      RO,R1 ;COMPARE EXP VS ACT
1432 003734 001401          BEQ      +4
1433 003736 104003          ERROR   3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1434          ;BIT...ALSO CHECK DNA
1435 003740 005367 175156          DEC      SHIFT ;# OF SHIFTS
1436 003744 001352          BNE      1$ ;DO IT AGAIN ?
1437          ;NOW POKE CLK TO SEE DNA
1438 003746 052777 020000 175752      BIS      #CLK,@TXCSR ;POKE CLK
1439 003754 012700 000000          MOV      #0,RO ;EXPECTED
1440 003760 017701 175742          MOV      @TXCSR,R1 ;ACTUAL
1441 003764 042701 077777          BIC      #77777,R1 ;SAVE DNA ONLY
1442 003770 020001          CMP      RO,R1 ;COMPARE EXP VS ACT
1443 003772 001401          BEQ      +4
1444 003774 104003          ERROR   3 ;DNA SHOULD BE SET
1445          ;IF DNA DID NOT SET
1446          ;CHECK WORD LENGTH SELECT LOGIC
1447
1448          ;:THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
1449          ;:OF THE TRANSMITTER SECTION.
1450          ;:IT ALSO CHECKS DNA TIMING
1451          ;:MODE:SYNINT
1452          ;:LENGTH:EIGHT PLUS PARITY
1453          ;:PARITY:EVEPAR
1454          ;:CHARACTER:125

```



```

1455
1456
1457 003776 000004
1458 004000 052777 000400 175720
1459 004006 012777 030000 175706
1460 004014 052777 000400 175704
1461
1462
1463
1464 004022 012777 004020 175676
1465
1466
1467 004030 012777 037426 175664
1468 004036 016703 175664
1469 004042 112777 000125 175662
1470 004050 012767 000125 175422
1471 004056 012767 000011 175036
1472
1473 004064 052777 020000 175634
1474 004072 042777 020000 175626
1475 004100 005000
1476 004102 006067 175372
1477 004106 103002
1478 004110 052700 002000
1479 004114
1480 004114 052777 020000 175604
1481 004122 042777 020000 175576
1482 004130 017701 175572
1483 004134 042701 075777
1484 004140 020001
1485 004142 001401
1486 004144 104003
1487
1488 004146 005367 174750
1489 004152 001352
1490
1491 004154 052777 020000 175544
1492 004162 012700 100000
1493 004166 017701 175534
1494 004172 042701 077777
1495 004176 020001
1496 004200 001401
1497 004202 104003
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510 004204 000004

;*****
;TST3: SCOPE
;BIS #MRESET,@TXCSR ;MASTER RESET
;MOV #SYNINT,@PARCSR ;SET THE MODE
;BIS #MRESET,@TXCSR ;MASTER RESET

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

;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
;MOV #SYNINT!EIGHT!EVEPAR!26,@PARCSR
;MOV TXCSR,R3 ;SET UP FOR ERROR MSG
;MOVB #125,@TXDBUF ;LOAD DATA CHAR
;MOV #125,$TMP1 ;TO BE SHIFTED CHAR
;MOV #9,$SHIFT ;# OF SHIFTS

;POKE CLK TO GET INTO SYNCHRONIZATION
;BIS #CLK,@TXCSR ;POKE CLK UP
;BIC #CLK,@TXCSR ;POKE CLK DOWN
1$: CLR R0
;ROR $TMP1 ;FORCE CARRY
;BCC 2$ ;BR IF CARRY CLR
;BIS #BITW,R0 ;EQUIV OF BITW
2$: BIS #CLK,@TXCSR ;POKE CLK UP
;BIC #CLK,@TXCSR ;POKE CLK DOWN
;MOV @TXCSR,R1 ;ACTUAL
;BIC #075777,R1 ;SAVE BITW & DNA
;CMP R0,R1 ;COMPARE EXP VS ACT
;BEQ +4
;ERROR 3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
; ;BIT...ALSO CHECK DNA
; ;# OF SHIFTS
;DEC SHIFT ;DO IT AGAIN ?
;BNE 1$

;NOW POKE CLK TO SEE DNA
;BIS #CLK,@TXCSR ;POKE CLK
;MOV #100000,R0 ;EXPECTED
;MOV @TXCSR,R1 ;ACTUAL
;BIC #77777,R1 ;SAVE DNA ONLY
;CMP R0,R1 ;COMPARE EXP VS ACT
;BEQ +4
;ERROR 3 ;DNA SHOULD BE SET
; ;IF DNA DID NOT SET
; ;CHECK WORD LENGTH SELECT LOGIC

; ;THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
; ;OF THE TRANSMITTER SECTION.
; ;IT ALSO CHECKS DNA TIMING
; ;MODE:SYNINT
; ;LENGTH:EIGHT PLUS PARITY
; ;PARITY:00DPAR
; ;CHARACTER:125

;*****
;TST4: SCOPE

```

H03

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 35
 DZDUUA.M11 13-OCT-76 08:28 INITIALIZE THE COMMON TAGS

```

1511 004206 052777 000400 175512      BIS      #MRESET,@TXCSR ;MASTER RESET
1512 004214 012777 030000 175500      MOV      #SYNINT,@PARCSR ;SET THE MODE
1513 004222 052777 000400 175476      BIS      #MRESET,@TXCSR ;MASTER RESET
1514
1515 ;SET MAINTENANCE MODE & SEND
1516 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1517 004230 012777 004020 175470      MOV      #MINT!SEND,@TXCSR
1518
1519 ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
1520 004236 012777 037026 175456      MOV      #SYNINT!EIGHT!000PAR!26,@PARCSR
1521 004244 016703 175456      MOV      TXCSR,R3 ;SET UP FOR ERROR MSG
1522 004250 112777 000125 175454      MOV      #125,@TXDBUF ;LOAD DATA CHAR
1523 004256 012767 000525 175214      MOV      #525,$TMP1 ;TO BE SHIFTED CHAR
1524 004264 012767 000011 174630      MOV      #9,$SHIFT ;# OF SHIFTS
1525
1526 ;POKE CLK TO GET INTO SYNCHRONIZATION
1527 004272 052777 020000 175426      BIS      #CLK,@TXCSR ;POKE CLK UP
1528 004300 042777 020000 175420      BIC      #CLK,@TXCSR ;POKE CLK DOWN
1529 004310 006067 175164      CLR      R0
1530 004314 103002 ;FORCE CARRY
1531 004316 052700 002000      ROR      $TMP1 ;BR IF CARRY CLR
1532 004322 ;EQUIV OF BITW
1533 004322 052777 020000 175376      BIS      #BITW,R0
1534 004330 042777 020000 175370      BIC      #CLK,@TXCSR ;POKE CLK DOWN
1535 004336 017701 175364      MOV      @TXCSR,R1 ;ACTUAL
1536 004342 042701 075777      BIC      #075777,R1 ;SAVE BITW & DNA
1537 004346 020001 ;COMPARE EXP VS ACT
1538 004350 001401      CMP      R0,R1
1539 004352 104003      BEQ      .+4
1540 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1541 ;BIT...ALSO CHECK DNA
1542 004354 005367 174542      DEC      SHIFT ;# OF SHIFTS
1543 004360 001352      BNE      $S ;DO IT AGAIN ?
1544 ;NOW POKE CLK TO SEE DNA
1545 004362 052777 020000 175336      BIS      #CLK,@TXCSR ;POKE CLK
1546 004370 012700 100000      MOV      #100000,R0 ;EXPECTED
1547 004374 017701 175326      MOV      @TXCSR,R1 ;ACTUAL
1548 004400 042701 077777      BIC      #77777,R1 ;SAVE DNA ONLY
1549 004404 020001 ;COMPARE EXP VS ACT
1550 004406 001401      CMP      R0,R1
1551 004410 104003      BEQ      .+4
1552 ;DNA SHOULD BE SET
1553 ;IF DNA DID NOT SET
1554 ;CHECK WORD LENGTH SELECT LOGIC
1555
1556 ;: THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
1557 ;: OF THE TRANSMITTER SECTION.
1558 ;: IT ALSO CHECKS DNA TIMING
1559 ;: MODE: ISYMOD
1560 ;: LENGTH:EIGHT PLUS PARITY
1561 ;: PARITY:EVEPAR
1562 ;: CHARACTER:125
1563 ;:*****
1564 ;†ST5: SCOPE
1565 004412 000004
1566 004414 052777 000400 175304      BIS      #MRESET,@TXCSR ;MASTER RESET
1567 004422 012777 000000 175272      MOV      #ISYMOD,@PARCSR ;SET THE MODE
1568 004430 052777 000400 175270      BIS      #MRESET,@TXCSR ;MASTER RESET

```

INITIALIZE THE COMMON TAGS

```

1567
1568 ;SET MAINTENANCE MODE & SEND
1569 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1570 004436 012777 004020 175262 MOV #MINT!SEND,@TXCSR
1571
1572 ;SET MODE, # OF BITS,PARITY SENSE & LOAD SYNC REG
1573 004444 012777 007426 175250 MOV #ISYMOD!EIGHT!EVEPAR!26,@PARCSR
1574 004452 016703 175250 MOV TXCSR,R3 ;SET UP FOR ERROR MSG
1575 004456 112777 000125 175246 MOV #125,@TXDBUF ;LOAD DATA CHAR
1576 004464 012767 002252 175006 MOV #2252,STMP1 ;TO BE SHIFTED CHAR
1577 004472 012767 000013 174422 MOV #11,SHIFT ;# OF SHIFTS
1578
1579 004500 052777 020000 175220 ;POKE CLK TO GET INTO SYNCHRONIZATION
1580 004506 042777 020000 175212 BIS #CLK,@TXCSR ;POKE CLK UP
1581 004514 005000 CLR R0 ;POKE CLK DOWN
1582 004516 006067 174756 ROR STMP1 ;FORCE CARRY
1583 004522 103002 BCC 2$ ;BR IF CARRY CLR
1584 004524 052700 002000 BIS #BITW,R0 ;EQUIV OF BITW
1585
1586 004530 052777 020000 175170 BIS #CLK,@TXCSR ;POKE CLK UP
1587 004536 042777 020000 175162 BIC #CLK,@TXCSR ;POKE CLK DOWN
1588 004544 017701 175156 MOV @TXCSR,R1 ;ACTUAL
1589 004550 042701 075777 BIC #075777,R1 ;SAVE BITW & DNA
1590 004554 020001 CMP R0,R1 ;COMPARE EXP VS ACT
1591 004556 001401 BEQ +4
1592 004560 104003 ERROR 3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1593 ;BIT...ALSO CHECK DNA
1594 004562 005367 174334 DEC SHIFT ;# OF SHIFTS
1595 004566 001352 BNE 1$ ;DO IT AGAIN ?
1596
1597 004570 052777 020000 175130 ;NOW POKE CLK TO SEE DNA
1598 004576 012700 000000 BIS #CLK,@TXCSR ;POKE CLK
1599 004602 017701 175120 MOV #0,R0 ;EXPECTED
1600 004606 042701 077777 MOV @TXCSR,R1 ;ACTUAL
1601 004612 020001 BIC #77777,R1 ;SAVE DNA ONLY
1602 004614 001401 CMP R0,R1 ;COMPARE EXP VS ACT
1603 004616 104003 BEQ +4
1604 ERROR 3 ;DNA SHOULD BE SET
1605 ;IF DNA DID NOT SET
1606 ;CHECK WORD LENGTH SELECT LOGIC
1607
1608 ;; THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
1609 ;; OF THE TRANSMITTER SECTION.
1610 ;; IT ALSO CHECKS DNA TIMING
1611 ;; MODE:ISYMOD
1612 ;; LENGTH:EIGHT PLUS PARITY
1613 ;; PARITY:ODDPAR
1614 ;; CHARACTER:125
1615
1616 004620 000004 *****
1617 004622 052777 000400 175076 †ST6: SCOPE BIS #MRESET,@TXCSR ;MASTER RESET
1618 004630 012777 000000 175064 MOV #ISYMOD,@PARCSR ;SET THE MODE
1619 004636 052777 000400 175062 BIS #MRESET,@TXCSR ;MASTER RESET
1620
1621 ;SET MAINTENANCE MODE & SEND
1622 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)

```

J03

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 37
 DZDUUA.M11 13-OCT-76 08:28 INITIALIZE THE COMMON TAGS

```

1623 004644 012777 004020 175054      MOV      #MINT!SEND,@TXCSR
1624
1625                                     ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
1626 004652 012777 007026 175042      MOV      #ISYMOD!EIGHT!000PAR!26,@PARCSR
1627 004660 016703 175042      MOV      TXCSR,R3 ;SET UP FOR ERROR MSG
1628 004664 112777 000125 175040      MOV      #125,@TXDBUF ;LOAD DATA CHAR
1629 004672 012767 003252 174600      MOV      #3252,STMP1 ;TO BE SHIFTED CHAR
1630 004700 012767 000013 174214      MOV      #11,SHIFT ;# OF SHIFTS
1631                                     ;POKE CLK TO GET INTO SYNCHRONIZATION
1632 004706 052777 020000 175012      BIS      #CLK,@TXCSR ;POKE CLK UP
1633 004714 042777 020000 175004      BIC      #CLK,@TXCSR ;POKE CLK DOWN
1634 004722 005000
1635 004724 006067 174550      1$:     CLR      RO
1636 004730 103002      ROR      STMP1 ;FORCE CARRY
1637 004732 052700 002000      BCC     2$:     ;BR IF CARRY CLR
1638 004736      BIS      #BITW,RO ;EQUIV OF BITW
1639 004736 052777 020000 174762      2$:     BIS      #CLK,@TXCSR ;POKE CLK UP
1640 004744 042777 020000 174754      BIC      #CLK,@TXCSR ;POKE CLK DOWN
1641 004752 017701 174750      MOV      @TXCSR,R1 ;ACTUAL
1642 004756 042701 075777      BIC      #075777,R1 ;SAVE BITW & DNA
1643 004762 020001      CMP      RO,R1 ;COMPARE EXP VS ACT
1644 004764 001401      BEQ     .+4
1645 004766 104003      ERROR   3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1646                                     ;BIT... ALSO CHECK DNA
1647 004770 005367 174126      DEC     SHIFT ;# OF SHIFTS
1648 004774 001352      BNE     1$ ;DO IT AGAIN ?
1649                                     ;NOW POKE CLK TO SEE DNA
1650 004776 052777 020000 174722      BIS      #CLK,@TXCSR ;POKE CLK
1651 005004 012700 000000      MOV      #0,RO ;EXPECTED
1652 005010 017701 174712      MOV      @TXCSR,R1 ;ACTUAL
1653 005014 042701 077777      BIC      #77777,R1 ;SAVE DNA ONLY
1654 005020 020001      CMP      RO,R1 ;COMPARE EXP VS ACT
1655 005022 001401      BEQ     .+4
1656 005024 104003      ERROR   3 ;DNA SHOULD BE SET
1657                                     ;IF DNA DID NOT SET
1658                                     ;CHECK WORD LENGTH SELECT LOGIC
1659
1660
1661                                     ;: THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
1662                                     ;: OF THE TRANSMITTER SECTION.
1663                                     ;: IT ALSO CHECKS DNA TIMING
1664                                     ;: MODE:SYNINT
1665                                     ;: LENGTH:EIGHT PLUS PARITY
1666                                     ;: PARITY:EVEPAR
1667                                     ;: CHARACTER:252
1668
1669                                     ;:*****
1670 005026 000004      †ST7:  SCOPE
1671 005030 052777 000400 174670      BIS      #MRESET,@TXCSR ;MASTER RESET
1672 005036 012777 030000 174656      MOV      #SYNINT,@PARCSR ;SET THE MODE
1673 005044 052777 000400 174654      BIS      #MRESET,@TXCSR ;MASTER RESET
1674
1675                                     ;SET MAINTENANCE MODE & SEND
1676                                     ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1677 005052 012777 004020 174646      MOV      #MINT!SEND,@TXCSR
1678

```

K03

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 38
 DZDUUA.M11 13-OCT-76 08:28

INITIALIZE THE COMMON TAGS

```

1679 ;SET MODE # OF BITS, PARITY SENSE, & LOAD SYNC REG
1680 005060 012777 037426 174634 MOV #SYNINT!EIGHT!EVEPAR!26,@PARCSR
1681 005066 016703 174634 MOV TXCSR,R3 ;SET UP FOR ERROR MSG
1682 005072 112777 000252 174632 MOVB #252,@TXDBUF ;LOAD DATA CHAR
1683 005100 012767 000252 174372 MOV #25,@STMP1 ;TO BE SHIFTED CHAR
1684 005106 012767 000011 174006 MOV #9,SHIFT ;OF SHIFTS
1685 ;POKE CLK TO GET INTO SYNCHRONIZATION
1686 005114 052777 020000 174604 BIS #CLK,@TXCSR ;POKE CLK UP
1687 005122 042777 020000 174576 BIC #CLK,@TXC ;POKE CLK DOWN
1688 005130 005000 1S: CLR RO
1689 005132 006067 174342 ROR STMP1 ;FORCE CARRY
1690 005136 103002 BCC 2S ;BR IF CARRY CLR
1691 005140 052700 002000 BIS #BITW,RO ;EQUIV OF BITW
1692 005144 2S:
1693 005144 052777 020000 174554 BIS #CLK,@TXCSR ;POKE CLK UP
1694 005152 042777 020000 174546 BIC #CLK,@TXCSR ;POKE CLK DOWN
1695 005160 017701 174542 MOV @TXCSR,R1 ;ACTUAL
1696 005164 042701 075777 BIC #075777,R1 ;SAVE BITW & DNA
1697 005170 020001 CMP RO,R1 ;COMPARE EXP VS ACT
1698 005172 001401 BEQ +4
1699 005174 104003 ERROR 3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1700 ;BIT ALSO CHECK DNA
1701 005176 005367 173720 DEC SHIFT ;# OF SHIFTS
1702 005202 001352 BNE 1S ;DO IT AGAIN ?
1703 ;NOW POKE CLK TO SEE DNA
1704 005204 052777 020000 174514 BIS #CLK,@TXCSR ;POKE CLK
1705 005212 012700 100000 MOV #100000,RO ;EXPECTED
1706 005216 017701 174504 MOV @TXCSR,R1 ;ACTUAL
1707 005222 042701 077777 BIC #77777,R1 ;SAVE DNA ONLY
1708 005226 020001 CMP RO,R1 ;COMPARE EXP VS ACT
1709 005230 001401 BEQ +4
1710 005232 104003 ERROR 3 ;DNA SHOULD BE SET
1711 ;IF DNA DID NOT SET
1712 ;CHECK WORD LENGTH SELECT LOGIC
1713
1714 ;: THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
1715 ;: OF THE TRANSMITTER SECTION.
1716 ;: IT ALSO CHECKS DNA TIMING
1717 ;: MODE:SYNINT
1718 ;: LENGTH:EIGHT PLUS PARITY
1719 ;: PARITY:ODDPAR
1720 ;: CHARACTER:252
1721
1722 ;:*****
1723 005234 000004 †ST10: SCOPE
1724 005236 052777 000400 174462 BIS #MRESET,@TXCSR ;MASTER RESET
1725 005244 012777 030000 174450 MOV #SYNINT,@PARCSR ;SET THE MODE
1726 005252 052777 000400 174446 BIS #MRESET,@TXCSR ;MASTER RESET
1727
1728 ;SET MAINTENANCE MODE & SEND
1729 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1730 005260 012777 004020 174440 MOV #MINT!SEND,@TXCSR
1731
1732 ;SET MODE # OF BITS, PARITY SENSE, & LOAD SYNC REG
1733 005266 012777 037026 174426 MOV #SYNINT!EIGHT!ODDPAR!26,@PARCSR
1734 005274 016703 174426 MOV TXCSR,R3 ;SET UP FOR ERROR MSG
  
```

INITIALIZE THE COMMON TAGS

```

1735 005300 112777 000252 174424      MOVB    #252,@TXDBUF    ;LOAD DATA CHAR
1736 005306 012767 000652 174164      MOV     #652,$TMP1     ;TO BE SHIFTED CHAR
1737 005314 012767 000011 173600      MOV     #9,$SHIFT      ;# OF SHIFTS
1738                                     ;POKE CLK TO GET INTO SYNCHRONIZATION
1739 005322 052777 020000 174376      BIS     #CLK,@TXCSR    ;POKE CLK UP
1740 005330 042777 020000 174370      BIC     #CLK,@TXCSR    ;POKE CLK DOWN
1741 005336 005000                                     15:    CLR     R0
1742 005340 006067 174134      ROR     $TMP1          ;FORCE CARRY
1743 005344 103002                                     BCC     25             ;BR IF CARRY CLR
1744 005346 052700 002000      BIS     #BITW,R0       ;EQUIV OF BITW
1745 005352                                     25:
1746 005352 052777 020000 174346      BIS     #CLK,@TXCSR    ;POKE CLK UP
1747 005360 042777 020000 174340      BIC     #CLK,@TXCSR    ;POKE CLK DOWN
1748 005366 017701 174334      MOV     @TXCSR,R1      ;ACTUAL
1749 005372 042701 075777      BIC     #075777,R1     ;SAVE BITW & DNA
1750 005376 020001      CMP     R0,R1          ;COMPARE EXP VS ACT
1751 005400 001401      BEQ     +4
1752 005402 104003      ERROR   3              ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1753                                     ;BIT...ALSO CHECK DNA
1754 005404 005367 173512      DEC     SHIFT          ;# OF SHIFTS
1755 005410 001352      BNE     15             ;DO IT AGAIN ?
1756                                     ;NOW POKE CLK TO SEE DNA
1757 005412 052777 020000 174306      BIS     #CLK,@TXCSR    ;POKE CLK
1758 005420 012700 100000      MOV     #100000,R0     ;EXPECTED
1759 005424 017701 174276      MOV     @TXCSR,R1      ;ACTUAL
1760 005430 042701 077777      BIC     #77777,R1      ;SAVE DNA ONLY
1761 005434 020001      CMP     R0,R1          ;COMPARE EXP VS ACT
1762 005436 001401      BEQ     +4
1763 005440 104003      ERROR   3              ;DNA SHOULD BE SET
1764                                     ;IF DNA DID NOT SET
1765                                     ;CHECK WORD LENGTH SELECT LOGIC
1766
1767                                     ;; THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
1768                                     ;; OF THE TRANSMITTER SECTION.
1769                                     ;; IT ALSO CHECKS DNA TIMING
1770                                     ;; MODE:SYNINT
1771                                     ;; LENGTH:EIGHT PLUS PARITY
1772                                     ;; PARITY:EVEPAR
1773                                     ;; CHARACTER:0
1774
1775                                     ;*****
1776 005442 000004      $ST11: SCOPE
1777 005444 052777 000400 174254      BIS     #MRESET,@TXCSR ;MASTER RESET
1778 005452 012777 030000 174242      MOV     #SYNINT,@PARCSR ;SET THE MODE
1779 005460 052777 000400 174240      BIS     #MRESET,@TXCSR ;MASTER RESET
1780
1781                                     ;SET MAINTENANCE MODE & SEND
1782                                     ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1783 005466 012777 004020 174232      MOV     #MINT!SEND,@TXCSR
1784
1785                                     ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
1786 005474 012777 037426 174220      MOV     #SYNINT!EIGHT!EVEPAR!26,@PARCSR
1787 005502 016703 174220      MOV     TXCSR,R3        ;SET UP FOR ERROR MSG
1788 005506 112777 000000 174216      MOVB    #0,@TXDBUF     ;LOAD DATA CHAR
1789 005514 012767 000000 173756      MOV     #0,$TMP1       ;TO BE SHIFTED CHAR
1790 005522 012767 000011 173372      MOV     #9,$SHIFT      ;# OF SHIFTS
    
```

INITIALIZE THE COMMON TAGS

```

1791 ;POKE CLK TO GET INTO SYNCHRONIZATION
1792 005530 052777 020000 174170 BIS #CLK,@TXCSR ;POKE CLK UP
1793 005536 042777 020000 174162 BIC #CLK,@TXCSR ;POKE CLK DOWN
1794 005544 005000 1$: CLR RO
1795 005546 006067 173726 ROR STMP1 ;FORCE CARRY
1796 005552 103002 BCC 2$ ;BR IF CARRY CLR
1797 005554 052700 002000 BIS #BITW,RO ;EQUIV OF BITW
1798 005560 2$: BIS #CLK,@TXCSR ;POKE CLK UP
1799 005560 052777 020000 174140 BIC #CLK,@TXCSR ;POKE CLK DOWN
1800 005566 042777 020000 174132 MOV @TXCSR,R1 ;ACTUAL
1801 005574 017701 174126 BIC #075777,R1 ;SAVE BITW & DNA
1802 005600 042701 075777 CMP RO,R1 ;COMPARE EXP VS ACT
1803 005604 020001 BEQ +4
1804 005606 001401 ERROR 3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1805 005610 104003 ;BIT... ALSO CHECK DNA
1806 ; # OF SHIFTS
1807 005612 005367 173304 DEC SHIFT
1808 005616 001352 BNE 1$ ;DO IT AGAIN ?
1809 ;NOW POKE CLK TO SEE DNA
1810 005620 052777 020000 174100 BIS #CLK,@TXCSR ;POKE CLK
1811 005626 012700 100000 MOV #100000,RO ;EXPECTED
1812 005632 017701 174070 MOV @TXCSR,R1 ;ACTUAL
1813 005636 042701 077777 BIC #77777,R1 ;SAVE DNA ONLY
1814 005642 020001 CMP RO,R1 ;COMPARE EXP VS ACT
1815 005644 001401 BEQ +4
1816 005646 104003 ERROR 3 ;DNA SHOULD BE SET
1817 ;IF DNA DID NOT SET
1818 ;CHECK WORD LENGTH SELECT LOGIC
1819
1820 ;: THIS TEST VERIFYS CHARACTER P'JS PARITY GENERATION
1821 ;: OF THE TRANSMITTER SECTION.
1822 ;: IT ALSO CHECKS DNA TIMING
1823 ;: MODE:SYNINT
1824 ;: LENGTH:EIGHT PLUS PARITY
1825 ;: PARITY:ODDPAR
1826 ;: CHARACTER:0
1827
1828 ;*****
1829 005650 000004 †ST12: SCOPE
1830 005652 052777 000400 174046 BIS #MRESET,@TXCSR ;MASTER RESET
1831 005660 012777 030000 174034 MOV #SYNINT,@PARCSR ;SET THE MODE
1832 005666 052777 000400 174032 BIS #MRESET,@TXCSR ;MASTER RESET
1833
1834 ;SET MAINTENANCE MODE & SEND
1835 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1836 005674 012777 004020 174024 MOV #MINT!SEND,@TXCSR
1837
1838 ;SET MODE # OF BITS PARITY SENSE & LOAD SYNC REG
1839 005702 012777 037026 174012 MOV #SYNINT!EIGHT!ODDPAR!26,@PARCSR
1840 005710 016703 174012 MOV TXCSR,J ;SET UP FOR ERROR MSG
1841 005714 112777 000000 174010 MOVB #0,@TXOBUF ;LOAD DATA CHAR
1842 005722 012767 000400 173550 MOV #400,STMP1 ;TO BE SHIFTED CHAR
1843 005730 012767 000011 173164 MOV #9,SHIFT ;# OF SHIFTS
1844 ;POKE CLK TO GET INTO SYNCHRONIZATION
1845 005736 052777 020000 173762 BIS #CLK,@TXCSR ;POKE CLK UP
1846 005744 042777 020000 173754 BIC #CLK,@TXCSR ;POKE CLK DOWN
    
```

N03

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 41
DZDUUA.M11 13-OCT-76 08:28 INITIALIZE THE COMMON TAGS

1847 005752 005000
1848 005754 006067 173520
1849 005760 103002
1850 005762 052700 002000
1851 005766
1852 005766 052777 020000 173732
1853 005774 042777 020000 173724
1854 006002 017701 173720
1855 006006 042701 075777
1856 006012 020001
1857 006014 001401
1858 006016 104003
1859
1860 006020 003367 173076
1861 006024 001352
1862
1863 006026 052777 020000 173672
1864 006034 012700 100000
1865 006040 017701 173662
1866 006044 042701 077777
1867 006050 020001
1868 006052 001401
1869 006054 104003
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882 006056 000004
1883 006060 052777 000400 173640
1884 006066 012777 030000 173626
1885 006074 052777 000400 173624
1886
1887
1888
1889 006102 012777 004020 173616
1890
1891
1892 006110 012777 037426 173604
1893 006116 016703 173604
1894 006122 112777 000377 173602
1895 006130 012767 000377 173342
1896 006136 012767 000011 172756
1897
1898 006144 052777 020000 173554
1899 006152 042777 020000 173546
1900 006160 005000
1901 006162 006067 173312
1902 006166 103002

```
1S: CLR R0
ROR $TMP1 ;FORCE CARRY
BCC 2S ;BR IF CARRY CLR
BIS #BITW,R0 ;EQUIV OF BITW
2S: BIS #CLK,@TXCSR ;POKE CLK UP
BIC #CLK,@TXCSR ;POKE CLK DOWN
MOV @TXCSR,R1 ;ACTUAL
BIC #075777,R1 ;SAVE BITW & DNA
CMP R0,R1 ;COMPARE EXP VS ACT
BEQ +4
ERROR 3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
;BIT...ALSO CHECK DNA
DEC SHIFT ;# OF SHIFTS
BNE 1S ;DO IT AGAIN ?
;NOW POKE CLK TO SEE DNA
BIS #CLK,@TXCSR ;POKE CLK
MOV #100000,R0 ;EXPECTED
MOV @TXCSR,R1 ;ACTUAL
BIC #77777,R1 ;SAVE DNA ONLY
CMP R0,R1 ;COMPARE EXP VS ACT
BEQ +4
ERROR 3 ;DNA SHOULD BE SET
;IF DNA DID NOT SET
;CHECK WORD LENGTH SELECT LOGIC
```

```
;; THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
;; OF THE TRANSMITTER SECTION.
;; IT ALSO CHECKS DNA TIMING
;; MODE:SYNINT
;; LENGTH:EIGHT PLUS PARITY
;; PARITY:EVEPAR
;; CHARACTER:377
```

```
*****
†ST13: SCOPE
BIS #MRESET,@TXCSR ;MASTER RESET
MOV #SYNINT,@PARCSR ;SET THE MODE
BIS #MRESET,@TXCSR ;MASTER RESET
```

```
;SET MAINTENANCE MODE & SEND
;NOTE:BIT WINDOWSCLK ARE CLEARED (MTDATA=0)
MOV #MINT!SEND,@TXCSR
```

```
;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
MOV #SYNINT!EIGHT!EVEPAR!26,@PARCSR
MOV TXCSR,R3 ;SET UP FOR ERROR MSG
MOVB #377,@TXDBUF ;LOAD DATA CHAR
MOV #377,$TMP1 ;TO BE SHIFTED CHAR
MOV #9,SHIFT ;# OF SHIFTS
```

```
;POKE CLK TO GET INTO SYNCHRONIZATION
BIS #CLK,@TXCSR ;POKE CLK UP
BIC #CLK,@TXCSR ;POKE CLK DOWN
```

```
1S: CLR R0
ROR $TMP1 ;FORCE CARRY
BCC 2S ;BR IF CARRY CLR
```



```

1903 006170 052700 002000          BIS    #BITW,RO          ;EQUIV OF BITW
1904 006174                                     25:
1905 006174 052777 020000 173524    BIS    #CLK,@TXCSR      ;POKE CLK UP
1906 006202 042777 020000 173516    BIC    #CLK,@TXCSR      ;POKE CLK DOWN
1907 006210 017701 173512          MOV    @TXCSR,R1        ;ACTUAL
1908 006214 042701 075777          BIC    #075777,R1       ;SAVE BITW & DNA
1909 006220 020001          CMP    RO,R1           ;COMPARE EXP VS ACT
1910 006222 001401          BEQ    +4
1911 006224 104003          ERROR  3              ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1912                                     ;BIT... ALSO CHECK DNA
1913 006226 005367 172670          DEC    SHIFT           ;# OF SHIFTS
1914 006232 001352          BNE    15              ;DO IT AGAIN ?
1915                                     ;NOW POKE CLK TO SEE DNA
1916 006234 052777 020000 173464    BIS    #CLK,@TXCSR      ;POKE CLK
1917 006242 012700 100000          MOV    #100000,RO      ;EXPECTED
1918 006246 017701 173454          MOV    @TXCSR,R1        ;ACTUAL
1919 006252 042701 077777          BIC    #77777,R1       ;SAVE DNA ONLY
1920 006256 020001          CMP    RO,R1           ;COMPARE EXP VS ACT
1921 006260 001401          BEQ    +4
1922 006262 104003          ERROR  3              ;DNA SHOULD BE SET
1923                                     ;IF DNA DID NOT SET
1924                                     ;CHECK WORD LENGTH SELECT LOGIC
1925
1926                                     ;; THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
1927                                     ;; OF THE TRANSMITTER SECTION.
1928                                     ;; IT ALSO CHECKS DNA TIMING
1929                                     ;; MODE:SYNINT
1930                                     ;; LENGTH:EIGHT PLUS PARITY
1931                                     ;; PARITY:000PAR
1932                                     ;; CHARACTER:377
1933
1934                                     ;*****
1935 006264 000004          *ST14: SCOPE
1936 006266 052777 000400 173432    BIS    #MRESET,@TXCSR  ;MASTER RESET
1937 006274 012777 030000 173420    MOV    #SYNINT,@PARCSR ;SET THE MODE
1938 006302 052777 000400 173416    BIS    #MRESET,@TXCSR  ;MASTER RESET
1939
1940                                     ;SET MAINTENANCE MODE & SEND
1941                                     ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1942 006310 012777 004020 173410    MOV    #MINT!SEND,@TXCSR
1943
1944                                     ;SET MODE,# OF BITS,PARITY SENSE & LOAD SYNC REG
1945 006316 012777 037026 173376    MOV    #SYNINT!EIGHT!000PAR!26,@PARCSR
1946 006324 016703 173376          MOV    TXCSR,R3        ;SET UP FOR ERROR MSG
1947 006330 112777 000377 173374    MOVB   #377,@TXDBUF    ;LOAD DATA CHAR
1948 006336 012767 000777 173134    MOV    #777,$TMP1      ;TO BE SHIFTED CHAR
1949 006344 012767 000011 172550    MOV    #9,SHIFT        ;# OF SHIFTS
1950
1951                                     ;POKE CLK TO GET INTO SYNCHRONIZATION
1952 006352 052777 020000 173346    BIS    #CLK,@TXCSR      ;POKE CLK UP
1953 006360 042777 020000 173340    BIC    #CLK,@TXCSR      ;POKE CLK DOWN
1954 006366 005000          15:  CLR    RO
1955 006370 006067 173104          ROR    $TMP1           ;FORCE CARRY
1956 006374 103002          BCC    25              ;BR IF CARRY CLR
1957 006376 052700 002000          BIS    #BITW,RO        ;EQUIV OF BITW
1958 006402 052777 020000 173316    25:  BIS    #CLK,@TXCSR      ;POKE CLK UP

```

```

1959 006410 042777 020000 173310      BIC    #CLK,@TXCSR      ;POKE CLK DOWN
1960 006416 017701 173304              MOV    @TXCSR,R1        ;ACTUAL
1961 006422 042701 075777              BIC    #075777,R1      ;SAVE BITW & DNA
1962 006426 020001              CMP    R0,R1           ;COMPARE EXP VS ACT
1963 006430 0C1401              BEQ    +4
1964 006432 104003              ERROR  3               ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1965                                ;BIT...ALSO CHECK DNA
1966 006434 005367 172462              DEC    SHIFT           ;# OF SHIFTS
1967 006440 001352              BNE    1$             ;DO IT AGAIN ?
1968                                ;NOW POKE CLK TO SEE DNA
1969 006442 052777 020000 173256      BIS    #CLK,@TXCSR     ;POKE CLK
1970 006450 012700 100000              MOV    #100000,R0      ;EXPECTED
1971 006454 017701 173246              MOV    @TXCSR,R1       ;ACTUAL
1972 006460 042701 077777              BIC    #77777,R1      ;SAVE DNA ONLY
1973 006464 020001              CMP    R0,R1           ;COMPARE EXP VS ACT
1974 006466 001401              BEQ    +4
1975 006470 104003              ERROR  3               ;DNA SHOULD BE SET
1976                                ;IF DNA DID NOT SET
1977                                ;CHECK WORD LENGTH SELECT LOGIC

```

```

1981 :: THIS TEST VERIFYS THAT BY SENDING ONLY ONE SYNC
1982 :: CHARACTER (TWO SELECTED BY STRAPPING ) RECACT =0
1983 :: THEN SEND ONE ORDINARY CHARACTER (TO BREAK UP THE SEQUENCE)
1984 :: RECACT =0.....IT WILL TAKE TWO MORE SYNC CHARS
1985 :: BEFORE RECACT =1
1986 :: NOTE: THIS TEST WILL ONLY WORK WHEN TWO SYNC CHARS
1987 :: HAS BEEN BEEN SELECTED...OTHERWISE JUMP AROUND THIS TEST
1988 :: MODE:SYNC INTERNAL (SYNINT)
1989 :: PARITY: NOPAR
1990 :: LENGTH: EIGHT

```

```

1991 :: THIS TEST CHECKS ONLY THE RECEIVER SECTION

```

```

1992 *****

```

```

1995 TST15: SCOPE
1996 TSTB  SYNCNO ;TEST FOR # OF SYNC CHARS REQUIRED
1997 BPL   2$     ;IF NOT TWO GET OUT OF TEST
1998 BIS   #MRESET,@TXCSR ;MASTER RESET
1999 MOV   #SYNINT,@PARCSR ;SET THE MODE
2000 BIS   #MRESET,@TXCSR ;MASTER RESET

```

```

2001 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2002 MOV   #MCDATA!CLK!MINT!BREAK,@TXCSR

```

```

2003 ;SET MODE ,# OF BITS,PARITY SENSE &LOAD SYNC REG
2004 MOV   #SYNINT!EIGHT!NOPAR!26,@PARCSR
2005 BIS   #SYNSCH,@TXCSR ;SET SYNC SEARCH
2006 ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....

```

```

2007 BIC   #CLK,@TXCSR ;POKE CLK DOWN
2008 BIS   #CLK,@TXCSR ;POKE CLK UP

```

```

2009 ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2010 BIC   #CLK,@TXCSR ;POKE CLK DOWN
2011 BIS   #CLK,@TXCSR ;POKE CLK UP
2012 MOV   #8.,SHIFT ;# OF SHIFTS

```

```

2013 006570 052777 020000 173130
2014 006576 012767 000010 172316

```

INITIALIZE THE COMMON TAGS

```

2015 006604 012767 000026 172666      MOV      #26,STMP1      ; SYNC CHAR TO BE SHIFTED IN
2016 006612 004767 010330          JSR      PC,RPOKE      ; SHIFT IN THIS SYNC CHAR
2017 006616 032777 004000 173066      BIT      #REACT,@RXCSR ; REACT = 0 ?
2018 006624 001401          BEQ      .+4
2019 006626 104004          ERROR   4              ; REACT SHOULD BE 0
2020 006630 012767 000010 172264      MOV      #8.,SHIFT     ; # OF SHIFTS
2021 006636 012767 000025 172634      MOV      #25,STMP1    ; ANY CHARACTER
2022 006644 004767 010276          JSR      PC,RPOKE      ; SHIFT IN THIS CHARACTER
2023                                     ; YOU HAVE JUST LOST SYNCHRONIZATION.
2024                                     ; POKE THE CLK TWICE TO GET INTO SYNCHRONIZATION
2025                                     ; POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2026 006650 042777 020000 173050      BIC      #CLK,@TXCSR   ; POKE CLK DOWN
2027 006656 052777 020000 173042      BIS      #CLK,@TXCSR   ; POKE CLK UP
2028                                     ; POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2029 006664 042777 020000 173034      BIC      #CLK,@TXCSR   ; POKE CLK DOWN
2030 006672 052777 020000 173026      BIS      #CLK,@TXCSR   ; POKE CLK UP
2031 006700 012767 000002 172216      MOV      #2,COUNT     ; # OF SYNC CHARS
2032 006706 032777 004000 172776 1$: BIT      #REACT,@RXCSR ; REACT = 0 ?
2033 006714 001401          BEQ      .+4
2034 006716 104004          ERROR   4              ; REACT SHOULD BE 0
2035 006720 012767 000010 172174      MOV      #8.,SHIFT     ; # OF SHIFTS
2036 006726 012767 000026 172544      MOV      #26,STMP1    ; SYNC CHAR
2037 006734 004767 010206          JSR      PC,RPOKE      ; SHIFT IN THIS SYNC CHAR
2038 006740 005367 172160          DEC     COUNT
2039 006744 001360          BNE     1$             ; IS COUNT = 0 ? NO GO AGAIN
2040 006746 032777 004000 172736      BIT      #REACT,@RXCSR ; REACT = 1 ?
2041 006754 001001          BNE     .+4
2042 006756 104004          ERROR   4              ; REACT SHOULD BE ASSERTED
2043
2044 2$:
2045  ; THIS TEST VERIFYS MODE SELECT.....
2046  ; SYNC EXTERNAL VS. SYNC INTERNAL
2047  ;
2048  ; BASICALLY THE TEST CHECKS THAT THE RECEIVED
2049  ; DATA FREEZES IN SYNC INTERNAL
2050  ; IN SYNC EXTERNAL THIS DATA IS TRANSPARENT
2051  ; THIS TEST ONLY APPLIES TO THE RECEIVER SECTION
2052  ; LENGTH: EIGHT
2053  ; NOTE:SEARCH SYNC IS NOT SET
2054  ; *****
2054 006760 000004          ST16: SCOPE
2055 006762 052777 000400 172736      BIS      #MRESET,@TXCSR ; MASTER RESET
2056 006770 012777 030000 172724      MOV      #SYNINT,@PARCSR ; SET THE MODE
2057 006776 052777 000400 172722      BIS      #MRESET,@TXCSR ; MASTER RESET
2058
2059  ; SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2060 007004 012777 064001 172714      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
2061
2062  ; SET MODE, # OF BITS,PARITY SENSE &LOAD SYNC REG
2063 007012 012777 036026 172702      MOV      #SYNINT!EIGHT!NOPAR!26,@PARCSR
2064  ; POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2065 007020 042777 020000 172700      BIC      #CLK,@TXCSR   ; POKE CLK DOWN
2066 007026 052777 020000 172672      BIS      #CLK,@TXCSR   ; POKE CLK UP
2067  ; POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2068 007034 042777 020000 172664      BIC      #CLK,@TXCSR   ; POKE CLK DOWN
2069 007042 052777 020000 172656      BIS      #CLK,@TXCSR   ; POKE CLK UP
2070 007050 012767 000010 172044      MOV      #8.,SHIFT     ; # OF SHIFTS

```

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 45
 DZDUUA.M11 13-OCT-76 08:28 INITIALIZE THE COMMON TAGS

```

2071 007056 012767 000125 172414      MOV      #125,$TMP1      ;DATA CHARACTER
2072 007064 016703 172626      MOV      RXDBUF,R3      ;FOR ERROR MESSAGE
2073 007070 042777 040000 172630 1$:  BIC      #MTDATA,@TXCSR ;CLEAR MAINT DATA
2074 007076 000241      CLC
2075 007100 006067 172374      ROR      $TMP1      ;FORCE CARRY
2076 007104 103003      BCC      2$
2077 007106 052777 040000 172612      BIS      #MTDATA,@TXCSR ;SET MTDATA
2078 007114 042777 020000 172604 2$:  BIC      #CLK,@TXCSR    ;POKE CLK
2079 007122 052777 020000 172576      BIS      #CLK,@TXCSR
2080 007130 012700 000377      MOV      #377,R0      ;EXPECTED
2081 007134 017701 172556      MOV      @RXDBUF,R1    ;ACTUAL
2082 007140 020001      CMP      R0,R1
2083 007142 001401      BEQ      .+4
2084 007144 104002      ERROR    2
2085      ;DATA CHARACTER SHOULD COMPARE.....
2086 007146 005367 171750      DEC      SHIFT        ;IS IT THE LAST SHIFT?
2087 007152 001346      BNE      1$          ;NO ... SHIFT SOME MORE
2088 007154 052777 000400 172544      BIS      #MRESET,@TXCSR ;MASTER RESET
2089 007162 012777 020000 172532      MOV      #SYNEXT,@PARCSR ;SET THE MODE
2090 007170 052777 000400 172530      BIS      #MRESET,@TXCSR ;MASTER RESET
2091      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2092      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
2093 007176 012777 064001 172522
2094
2095      ;SET MODE # OF BITS,PARITY SENSE,&LOAD SYNC REG
2096 007204 012777 026026 172510      MOV      #SYNEXT!EIGHT!NOPAR!26,@PARCSR
2097      ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2098 007212 042777 020000 172506      BIC      #CLK,@TXCSR    ;POKE CLK DOWN
2099 007220 052777 020000 172500      BIS      #CLK,@TXCSR    ;POKE CLK UP
2100 007226 012767 000010 171666      MOV      #8,SHIFT      ;# OF SHIFTS
2101 007234 012767 000125 172236      MOV      #125,$TMP1    ;DATA CHARACTER
2102 007242 005067 172234      CLR      $TMP2
2103 007246 105167 172230      COMB    $TMP2      ;MAKE LOW BYTE ALL 1'S
2104      ;TO MATCH RXDBUF'S CONTENTS AFTER A MASTER RESET
2105 007252 042777 040000 172446 5$:  BIC      #MTDATA,@TXCSR ;CLR MAINT DATA
2106 007260 000241      CLC
2107 007262 006067 172212      ROR      $TMP1      ;FORCE CARRY
2108 007266 103003      BCC      6$
2109 007270 052777 040000 172430      BIS      #MTDATA,@TXCSR
2110 007276 106067 172200 6$:  RORB    $TMP2      ;PICK UP CARRY BIT
2111 007302 042777 020000 172416      BIC      #CLK,@TXCSR
2112 007310 052777 020000 172410      BIS      #CLK,@TXCSR
2113 007316 016700 172160      MOV      $TMP2,R0      ;EXPECTED
2114 007322 017701 172370      MOV      @RXDBUF,R1    ;ACTUAL
2115 007326 020001      CMP      R0,R1
2116 007330 001401      BEQ      .+4
2117 007332 104002      ERROR    2      ;DATA CHARACTER SHOULD COMPARE...
2118      ;THE DATA CHARACTER SHOULD BE SEEN AS IT
2119      ;SHIFTS ACROSS THE RECEIVER DATA OUTPUT
2120 007334 005367 171562      DEC      SHIFT
2121 007340 001344      BNE      5$
2122
2123      ;;THIS TEST VERIFYS TX DONE FUNCTION, DONE = 1
2124      ;;MODE: SYNC INTERNAL
2125      ;;PARITY: NO PARITY (NOPAR)
2126      ;;LENGTH: EIGHT

```

```

2127
2128
2129 007342 000004
2130
2131 007344 052777 000400 172354      BIS      #MRESET,@TXCSR ;MASTER RESET
2132 007352 012777 030000 172342      MOV      #SYNINT,@PARCSR ;SET THE MODE
2133 007360 052777 000400 172340      BIS      #MRESET,@TXCSR ;MASTER RESET
2134
2135 ;SET MAINTENANCE MODE & SEND
2136 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
2137 007366 012777 004020 172332      MOV      #MINT!SEND,@TXCSR
2138
2139 ;SET MODE # OF BITS,PARITY SENSE, & LOAD SYNC REG
2140 007374 012777 036026 172320      MOV      #SYNINT!EIGHT!NOPAR!26,@PARCSR
2141
2142 007402 105777 172320      TSTB    @TXCSR          ;TXDONE?
2143 007406 100401      BMI     .+4
2144 007410 104004      ERROR   4              ;TXDONE SHOULD BE SET
2145 007412 112777 000021 172312      MOVB    #21,@TXDBUF    ;LOAD ANY CHAR
2146 007420 052777 020000 172300      BIS     #CLK,@TXCSR    ;POKE CLK UP
2147 007426 042777 020000 172272      BIC     #CLK,@TXCSR    ;POKE CLK DOWN
2148 007434 105777 172266      TSTB    @TXCSR
2149 007440 100001      BPL     .+4
2150 007442 104004      ERROR   4              ;TXDONE SHOULD BE CLR
2151
2152 007444 052777 020000 172254      BIS     #CLK,@TXCSR    ;POKE CLK UP
2153 007452 042777 020000 172246      BIC     #CLK,@TXCSR    ;POKE CLK DOWN
2154 007460 105777 172242      TSTB    @TXCSR
2155 007464 100401      BMI     .+4
2156 007466 104004      ERROR   4              ;TXDONE SHOULD BE SET
2157
2158
2159
2160 ;; THIS TEST PROVES THAT RXERR FREEZES THE "RECEIVER
2161 ;; RESET" WHILE IN STRIP SYNC MODE
2162 ;; THIS TEST FIRST PROVES THAT AUTOMATIC RESETS OCCUR
2163 ;; WHEN STRIP SYNC IS SET AND SYNC CHARACTERS ARE SENT
2164 ;; BUT IF AN ERROR SHOULD OCCUR...THIS AUTOMATIC RESET
2165 ;; IS DISCOMBOBULATED
2166 ;; IE: FORCE OVERRUN (OVERRUN) WHILE STRIP SYNC IS SET
2167 ;; BY TRANSMITTING A DATA CHARACTER THEN TRANSMIT A SYNC CHARACTER
2168 ;; AND DON'T READ THAT DATA CHARACTER. NOTE: NORMALLY THE LOGIC
2169 ;; RESETS THE RXDONE & ERROR FLAGS PROVIDING THAT ONLY SYNC CHARACTERS ARE
2170 ;; STRIPPED
2171 ;; MODE: SYNC EXTERNAL (SYNEXT)
2172 ;; LENGTH: EIGHT
2173 ;; NOTE: THIS TEST USES BOTH RECEIVER AND TRANSMITTER LOGIC
2174 ;;
2175
2176
2177 007470 000004
2178
2179 007472 052777 000400 172226      BIS     #MRESET,@TXCSR ;MASTER RESET
2180 007500 012777 020000 172214      MOV     #SYNEXT,@PARCSR ;SET THE MODE
2181 007506 052777 000400 172212      BIS     #MRESET,@TXCSR ;MASTER RESET
2182

```

INITIALIZE THE COMMON TAGS

```

2183 ;SET MAINTENANCE MODE & SEND
2184 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
2185 007514 J12777 004020 172204 MOV #MINT!SEND,@TXCSR
2186
2187 ;SET MODE, # OF BITS, PARITY SENSE, & LOAD SYNC REG
2188 007522 012777 026026 172172 MOV #SYNEXT!EIGHT!NOPAR!26,@PARCSR
2189 007530 112777 000026 172174 MOVB #26,@TXDBUF ;LOAD SYNC CHAR
2190 007536 052777 000420 172146 BIS #SYNSCH!STPSYN,@RXCSR ;SET SYNC SEARCH & STRIP SYNC
2191 007544 016703 172146 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2192 007550 012767 000003 171346 MOV #3,COUNT ;# OF TIMES SYNC WILL BE SENT
2193 007556 052777 020000 172142 BIS #CLK,@TXCSR ;POKE CLK UP
2194 007564 042777 020000 172134 BIC #CLK,@TXCSR ;POKE CLK DOWN
2195 007572 012767 000010 171322 25: MOV #8.,SHIFT ;# OF SHIFTS
2196 007600 15:
2197 007600 052777 020000 172120 BIS #CLK,@TXCSR ;POKE CLK UP
2198 007606 042777 020000 172112 BIC #CLK,@TXCSR ;POKE CLK DOWN
2199 007614 005367 171302 DEC SHIFT
2200 007620 001367 BNE 15
2201 007622 105777 172064 TSTB @RXCSR ;RXDONE?
2202 007626 100001 BPL .+4
2203 007630 104004 ERROR 4 ;RXDONE SHOULD NOT ASSERT
2204 007632 005367 171266 DEC COUNT
2205 007636 001355 BNE 25
2206 007640 012700 000026 MOV #26,R0 ;EXPECTED
2207 007644 017701 172046 MOV @RXDBUF,R1 ;ACTUAL
2208 007650 020001 CMP R0,R1
2209 007652 001401 BEQ .+4
2210 007654 104002 ERROR 2 ;NOTE THAT OVERRUN SHOULD NOT OCCUR, ALSO
2211 ;SECOND & 3RD SYNC CHARACTER CAME FROM
2212 ;SYNC HOLDING REGISTER
2213 007656 012767 000003 171240 MOV #3,COUNT ;# OF TIMES
2214 007664 112777 000025 172040 MOVB #25,@TXDBUF ;LOAD ANY CHAR...HOWEVER...
2215 ;ONE MORE SYNC CHAR WILL BE SENT BEFORE
2216 ;THE "25" CHAR IS SENT (THE DMA BIT IS
2217 ;ALREADY UP)
2218
2219 007672 012767 000010 171222 45: MOV #8.,SHIFT ;# OF SHIFTS
2220
2221 007700 35:
2222 007700 052777 020000 172020 BIS #CLK,@TXCSR ;POKE CLK UP
2223 007706 042777 020000 172012 BIC #CLK,@TXCSR ;POKE CLK DOWN
2224 007714 005367 171202 DEC SHIFT
2225 007720 001367 BNE 35
2226 007722 005367 171176 DEC COUNT
2227 007726 001361 BNE 45
2228 007730 105777 171756 TSTB @RXCSR ;RXDONE = 1 ?
2229 007734 100401 BMI .+4
2230 007736 104004 ERROR 4 ;RXDONE SHOULD BE SET
2231 007740 012700 140026 MOV #RXERR!OVRUN!26,R0 ;EXPECTED
2232 007744 017701 171746 MOV @RXDBUF,R1 ;ACTUAL
2233 007750 020001 CMP R0,R1
2234 007752 001401 BEQ .+4
2235 007754 104002 ERROR 2 ;NOTE THAT OVRUN SHOULD OCCUR,
2236 ;ALSO SECOND SYNC CHARACTER CAME
2237 ;FROM SYNC HOLDING REGISTER
2238 ;SUMMARY: THE OVRUN STOPPED
    
```

H04

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 48
DZDUUA.M11 13-OCT-76 08:28 INITIALIZE THE COMMON TAGS

;THE AUTOMATIC RESETTING OF
;RXDONE & ERROR FLAGS.....CHECK THIS

;; THIS TEST VERIFYS THAT EITHER SEQUENCE OF
;; LOADING TXDBUF AND SETTING SEND
;; DOES CAUSE TRANSMISSION
;; MODE: SYNC EXT
;; LENGTH: EIGHT

2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294

007756 000004
007760 052777 000400 171740
007766 012777 020000 171726
007774 052777 000400 171724
010002 012777 004020 171716
010010 012777 026026 171704
010016 016703 171704
010022 042777 000020 171676
010030 012767 000025 171442
010036 112777 000025 171666
010044 052777 000020 171654
010052 052777 020000 171646
010060 042777 020000 171640
010066 012767 000010 171026
010074 005000
010076 006067 171376
010102 103002
010104 052700 002000
010110
010110 052777 020000 171610
010116 042777 020000 171602
010124 017701 171576
010130 042701 075777
010134 020001
010136 001401
010140 104003
010142 005367 170754
010146 001352

†ST21: SCOPE
BIS #MRESET,@TXCSR ;MASTER RESET
MOV #SYNEXT,@PARCSR ;SET THE MODE
BIS #MRESET,@TXCSR ;MASTER RESET
;SET MAINTENANCE MODE & SEND
;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
MOV #MINT!SEND,@TXCSR
;SET MODE, # OF BITS,PARITY SENSE, & LOAD SYNC REG
MOV #SYNEXT!EIGHT!NOPAR!26,@PARCSR
MOV TXCSR,R3 ;SET UP FOR ERROR MESSAGE
BIC #SEND,@TXCSR ;DROP SEND
MOV #25,\$TMP1
MOVB #25,@TXDBUF ;LOAD CHARACTER
BIS #SEND,@TXCSR ;SET SEND
;GET INTO SYNCRONIZATION
BIS #CLK,@TXCSR ;POKE CLK UP
BIC #CLK,@TXCSR ;POKE CLK DOWN
MOV #8,\$SHIFT ;# OF SHIFTS
1\$: CLR R0
ROR \$TMP1
BCC 2\$
BIS #BITW,R0 ;EQUIV OF BIT WINDOW
2\$: BIS #CLK,@TXCSR ;POKE CLK UP
BIC #CLK,@TXCSR ;POKE CLK DOWN
MOV @TXCSR,R1 ;ACTUAL
BIC #075777,R1 ;SAVE BIT WINDOW & DNA
CMP R0,R1
BEQ .+4
ERROR 3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA BIT
;ALSO CHECK DNA
DEC SHIFT
1\$ BNE 1\$

;; THIS TEST VERIFYS THAT DROPPING OF SEND IN THE
;; MIDDLE OF TRANSMITTING A CHARACTER DOES INDEED
;; FINISH TRANSMITTING THAT CHARACTER
;; MODE: SYNC EXT
;; LENGTH: EIGHT

INITIALIZE THE COMMON TAGS

```

2295
2296
2297 010150 000004
2298
2299 010152 052777 000400 171546
2300 010160 012777 020000 171534
2301 010166 052777 000400 171532
2302
2303
2304
2305 010174 012777 004020 171524
2306
2307
2308 010202 012777 026026 171512
2309 010210 016703 171512
2310 010214 112777 000252 171510
2311 010222 012767 000252 171250
2312 010230 012767 000010 170664
2313
2314 010236 052777 020000 171462
2315 010244 042777 020000 171454
2316
2317 010252 005000
2318 010254 006067 171220
2319 010260 103002
2320 010262 052700 002000
2321
2322 010266
2323 010266 052777 020000 171432
2324 010274 042777 020000 171424
2325 010302 017701 171420
2326 010306 042701 075777
2327 010312 020001
2328 010314 001401
2329 010316 104003
2330
2331 010320 005367 170576
2332 010324 022767 000003 170570
2333 010332 001003
2334 010334 042777 000020 171364
2335 010342 005767 170554
2336 010346 001341
2337
2338
2339
2340
2341
2342
2343
2344
2345 010350 000004
2346 010352 052777 000400 171346
2347 010360 012777 030000 171334
2348 010366 052777 000400 171332
2349
2350

```

```

*****
TST2: SCOPE
BIS #MRESET,@TXCSR ;MASTER RESET
MOV #SYNEXT,@PARCSR ;SET THE MODE
BIS #MRESET,@TXCSR ;MASTER RESET

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

;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
MOV #SYNEXT!EIGHT!NOPAR!26,@PARCSR
MOV TXCSR,R3 ;SETUP FOR ERROR MESSAGE
MOVB #25,@TXDBUF ;LOAD DATA CHAR.
MOV #25,$TMP1 ;SHIFTED CHAR
MOV #8,SHIFT ;# OF SHIFTS
;GET INTO SYNCRONIZATION
BIS #CLK,@TXCSR ;POKE CLK UP
BIC #CLK,@TXCSR ;POKE CLK DOWN

15: CLR R0
ROR $TMP1 ;FORCE CARRY
BCC 25
BIS #BITW,R0 ;EQUIV OF BIT WINDOW

25: BIS #CLK,@TXCSR ;POKE CLK UP
BIC #CLK,@TXCSR ;POKE CLK DOWN
MOV @TXCSR,R1 ;ACTUAL
BIC #075777,R1 ;SAVE ONLY BIT WINDOW & DNA
CMP R0,R1
BEQ +4
ERROR 3 ;BIT WINDOW DID NOT MATCH
;ACTUAL DATA BIT

DEC SHIFT
CMP #3,SHIFT
BNE 35
BIC #SEND,@TXCSR ;DROP SEND

35: TST SHIFT
BNE 15 ;DO IT AGAIN?

; THIS TEST VERIFYS THAT RXDONE ASSERTS WHEN STRIP SYNC IS SET
; MODE: SYNC INTERNAL
; LENGTH: EIGHT
*****
TST3: SCOPE
BIS #MRESET,@TXCSR ;MASTER RESET
MOV #SYNINT,@PARCSR ;SET THE MODE
BIS #MRESET,@TXCSR ;MASTER RESET

;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE

```



```

2351 010374 012777 064001 171324      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
2352                                     ;SET MODE , # OF BITS,PARITY SENSE &LOAD SYNC REG
2353                                     MOV      #SYNINT!EIGHT!NOPAR!26,@PARCSR
2354 010402 012777 036026 171312
2355
2356 010410 052777 000420 171274      BIS      #SYNSCH!STPSYN,@RXCSR      ;SET SYNC SEARCH &
2357                                     ;STRIP SYNC
2358                                     ;POKE CLK TO GET RECEIVER INTO SYNCHRONIZATION.....
2359 010416 042777 020000 171302      BIC      #CLK,@TXCSR      ;POKE CLK
2360 010424 052777 020000 171274      BIS      #CLK,@TXCSR
2361                                     ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2362 010432 042777 020000 171266      BIC      #CLK,@TXCSR      ;POKE CLK DOWN
2363 010440 052777 020000 171260      BIS      #CLK,@TXCSR      ;POKE CLK UP
2364 010446 016703 171244      MOV      RXDBUF,R3      ;FOR ERROR MESSAGE
2365 010452 012767 000002 170444      MOV      #2,COUNT      ;# OF TIMES OF SYNC CHARS.
2366                                     ;TEST TO SEE HOW MANY SYNC CHARS NEEDED
2367 010460 105767 170462      TSTB    SYNCNO
2368 010464 100402      BMI     3$      ;WILL IT BE ONE OR TWO ?
2369 010466 005367 170432      DEC     COUNT      ;MAKE IT ONE LESS
2370 010472 012767 000010 170422 3$:  MOV     #8,SHIFT      ;#OF SHIFTS
2371 010500 012767 000026 170772      MOV     #26,STMP1      ;SYNC CHAR
2372 010506 004767 006434      JSR     PC,RPOKE
2373 010512 005367 170406      DEC     COUNT      ;IS IT THE LAST SYNC CHAR ?
2374 010516 001365      BNE     3$      ;GO AGAIN AND SHIFT IN ANOTHER SYNC CHAR
2375 010520 032777 004000 171164      BIT     #REACT,@RXCSR      ;REACT=1?
2376 010526 001001      BNE     .+4
2377 010530 104004      ERROR   4      ;REACT SHOULD BE ASSERTED
2378 010532 105777 171154      TSTB    @RXCSR      ;RXDONE=0?
2379 010536 100001      BPL     .+4
2380 010540 104004      ERROR   4      ;RXDONE SHOULD NOT BE ASSERTED
2381 010542 012767 000010 170352      MOV     #8,SHIFT      ;#OF SHIFTS
2382 010550 012767 000025 170722      MOV     #25,STMP1      ;ANY CHARACTER
2383 010556 004767 006364      JSR     PC,RPOKE
2384 010562 105777 171124      TSTB    @RXCSR      ;RXDONE=1?
2385 010566 100401      BMI     .+4
2386 010570 104004      ERROR   4      ;RXDONE SHOULD NOW BE ASSERTED
2387 010572 012700 000025      MOV     #25,R0      ;EXPECTED
2388 010576 017701 171114      MOV     @RXDBUF,R1      ;ACTUAL
2389 010602 020001      CMP     R0,R1
2390 010604 001401      BEQ     .+4
2391 010606 104002      ERROR   2      ;CHARACTERS SHOULD BE MATCHED
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406

```

```

; THIS TEST VERIFYS THAT BY DROPPING SYNSCH
; IN THE MIDDLE OF A CHARACTER, SYNC CHARACTER SEQUENCE
; IS NEEDED BEFORE REACT, RXDONE ASSERT AGAIN.
; ALSO NOTE: SINCE REACT IS DEPENDENT ON MATCH DETECT,
; AND IF SYNSCH IS DROPPED IN THE MIDDLE OF
; A SYNC CHARACTER AND THEN RAISED AGAIN ;RXDONE SHOULD
; NOT ASSERT UNTIL NEW SYNC CHARACTER SEQUENCE..... HOWEVER
; ONE "PAD" CHARACTER MUST PRECEED THIS JUGGLING OF SEARCH SYNC (SYNSCH)
; MODE: SYNC INTERNAL (SYNINT)
; LENGTH: EIGHT

```

07
08
09
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60

010610 000004
010612 052777 000400 171106
010620 012777 030000 171074
010626 052777 000400 171072

010634 012777 064001 171064

010642 012777 036026 171052

010650 052777 000020 171034
010656 042777 020000 171042
010664 052777 020000 171034

010672 042777 020000 171026
010700 052777 020000 171020
010706 012767 000002 170210
010714 016703 170776
010720 012767 000010 170174
010726 012767 000026 170544
010734 004767 006206
010740 005367 170160
010744 001403

010746 105767 170174
010752 100762

010754 032777 004000 170730
010762 001001
010764 104004
010766 105777 170720
010772 100001
010774 104004

010776 012767 000010 170116
011004 012767 000025 170466
011012 004767 006130
011016 105777 170670
011022 100401
011024 104004
011026 017701 170664
011032 012700 000025
011036 020001
011040 001401
011042 104002

```
*****  
↑ST24: SCOPE  
BIS #MRESET,@TXCSR ;MASTER RESET  
MOV #SYNINT,@PARCSR ;SET THE MODE  
BIS #MRESET,@TXCSR ;MASTER RESET  
;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE  
MOV @MNTDATA!CLK!MINT!BREAK,@TXCSR  
;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG  
MOV #SYNINT!EIGHT!NOPAR!26,@PARCSR  
BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH  
;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....  
BIC #CLK,@TXCSR ;POKE CLK DOWN  
BIS #CLK,@TXCSR ;POKE CLK UP  
;POKE CLK TO GET LOGIC INTO SYNCRONIZATION  
BIC #CLK,@TXCSR ;POKE CLK DOWN  
BIS #CLK,@TXCSR ;POKE CLK UP  
MOV #2,COUNT ;# OF TIMES  
MOV RXDBUF,R3 ;FOR ERROR MESSAGE  
15: MOV #8,SHIFT ;# OF SHIFTS  
MOV #26,$TMP1 ;SYNC CHAR.  
JSR PC,RPOKE  
DEC COUNT  
BEQ 25  
;TEST TO SEE HOW MANY SYNC CHARACTERS NEEDED  
TSTB SYNCNO  
BMI 15  
25: BIT #REACT,@RXCSR ;REACT=1?  
BNE .+4  
ERROR 4 ;REACT SHOULD BE SET  
TSTB @RXCSR ;RXDONE = 0?  
BPL .+4  
ERROR 4 ;RXDONE SHOULD = 0  
;THE FOLLOWING ALLOWS THE RECEIVER "CHIP" TO RECOGNIZE THE DROPPING OF SYNSCH  
;...BASICALLY IT SIMULATES THE "PAD" CHARACTER REQUIREMENT AT THE END  
;OF DATA TRANSFER.  
;ONE "PAD" CHARACTER IS REQUIRED TO FLUSH OUT SYNC CHARACTER  
;AND ...YOU HAVE TO DROP SYNSCH ON THE NEXT OR FOLLOWING CHARACTER  
;FOR A "CLEAN" RESTART  
MOV #8,SHIFT ;# OF SHIFTS  
MOV #25,$TMP1 ;"PAD" CHARACTER  
JSR PC,RPOKE ;SHIFT IN "PAD" CHARACTER  
TSTB @RXCSR ;RXDONE = 1?  
BMI .+4  
ERROR 4 ;RXDONE SHOULD = 1  
MOV @RXDBUF,R1 ;STORE ACTUAL & CLEAR RXDONE  
MOV #25,R0 ;EXPECTED  
CMP R0,R1 ;COMPARE EXPECTED DATA VS ACTUAL DATA  
BEQ .+4  
ERROR 2 ;DATA SHOULD COMPARE WITHOUT ERROR FLAGS  
;THE FOLLOWING IS THE SYNC CHARACTER AFTER THE "PAD" CHARACTER  
;IN WHICH SEARCH SYNC (SYNSCH) IS JUGGLED TO CREATE AN ABORT SITUATION
```

```

;AT THE END OF A DATA TRANSFER SEQUENCE
2463      011044 012767 000004 170050      MOV      #4,SHIFT      ;# OF SHIFTS
2464      011052 012767 000026 170420      MOV      #26,$TMP1     ;SYNC CHAR.
2465      011060 004767 006062          JSR      PC,RPOKE
2466      011064 032777 004000 170620      BIT      #REACT,@RXCSR ;REACT=1?
2467      011072 001001          BNE     .+4
2468      011074 104004          ERROR   4              ;REACT SHOULD STILL BE SET
2469      011076 105777 170610      TSTB    @RXCSR ;RXDONE = 0 ?
2470      011102 100001          BPL     .+4
2471      011104 104004          ERROR   4              ;RXDONE SHOULD = 0 FROM PREVIOUS READING OF RXDBUF
2472      011106 042777 000020 170576      BIC      #SYNSCH,@RXCSR ;DROP SEARCH SYNC
2473      011114 032777 004000 170570      BIT      #REACT,@RXCSR ;REACT=0?
2474      011122 001401          BEQ     .+4
2475      011124 104004          ERROR   4              ;REACT SHOULD NOT BE SET
2476      ;NOW SHIFT TWO BITS TO ALLOW SEARCH SYNC = 0 TO TAKE
2477      ;EFFECT IN THE LOGIC(THIS ALLOWS THE RECEIVER CHIP TO SEE
2478      ;THE DROPPING OF SEARCH SYNC)
2479      011126 012767 000002 167766      MOV      #2,SHIFT      ;# OF SHIFTS
2480      011134 004767 006006          JSR      PC,RPOKE
2481      011140 052777 000020 170544      BIS      #SYNSCH,@RXCSR ;SET SEARCH SYNC
2482      011146 032777 004000 170536      BIT      #REACT,@RXCSR
2483      011154 001401          BEQ     .+4
2484      011156 104004          ERROR   4              ;REACT = 0 ?
2485      011160 105777 170526      TSTB    @RXCSR ;RXDONE = 0 ?
2486      011164 100001          BPL     .+4
2487      011166 104004          ERROR   4              ;RXDONE = 0 ?
2488      011170 012767 000002 167724      MOV      #2,SHIFT      ;# OF SHIFTS TO FINISH UP THE SYNC CHARACTER
2489      011176 004767 005744          JSR      PC,RPOKE
2490      011202 032777 004000 170502      BIT      #REACT,@RXCSR ;REACT=0?
2491      011210 001401          BEQ     .+4
2492      011212 104004          ERROR   4              ;REACT SHOULD NOT BE SET
2493      011214 105777 170472      TSTB    @RXCSR ;RXDONE=0?
2494      011220 100001          BPL     .+4
2495      011222 104004          ERROR   4              ;RXDONE SHOULD NOT BE ASSERTED
2496      011224 012700 000025          MOV      #25,R0        ;EXPECTED
2497      011230 017701 170462          MOV      @RXDBUF,R1    ;ACTUAL
2498      011234 020001          CMP      R0,R1        ;COMPARE EXPECTED VS ACTUAL
2499      011236 001401          BEQ     .+4
2500      011240 104002          ERROR   2              ;CHARACTERS SHOULD BE MATCHED.
2501      ;REMEMBER THAT THE " 25 " FROZE WHEN SYNSCH WAS DROPPED
2502      ;THEREFORE . . . 25 WILL BE READ AND NOT 26
2503      ;THE FOLLOWING VERIFYS THAT THE RE SYNCHRONIZATION COMES UP "CLEAN"
2504      011242 012767 000002 167654      MOV      #2,COUNT      ;# OF TIMES OF SYNC CHARS.
2505      ;
2506      ;TEST TO SEE HOW MANY SYNC CHARS NEEDED
2507      TSTB    SYNCNO ;HOW MANY SYNCs ARE YOU STRAPPED FOR ?
2508      BMI     3$      ;WILL IT BE ONE OR TWO ?
2509      DEC     COUNT  ;IT WAS ONLY ONE NEEDED
2510      3$:      MOV      #8,SHIFT ;#OF SHIFTS
2511      MOV      #26,$TMP1 ;SYNC CHAR
2512      JSR      PC,RPOKE
2513      DEC     COUNT  ;IS IT THE LAST SYNC CHAR ?
2514      BNE     3$      ;GO AGAIN AND SHIFT IN ANOTHER SYNC CHAR
2515      BIT      #REACT,@RXCSR ;REACT=1?
2516      BEQ     .+4
2517      ERROR   4              ;REACT SHOULD BE ASSERTED
2518

```

M04

```

2519 011322 105777 170364      TSTB   @RXCSR           ;RXDONE=0?
2520 011326 100001                BPL     .+4
2521 011330 104004                ERROR   4                ;RXDONE SHOULD NOT BE ASSERTED
2522 011332 012767 000010 167562  MOV    #8,SHIFT        ;#OF SHIFTS
2523 011340 012767 000025 170132  MOV    #25,$TMP1      ;ANY CHARACTER
2524 011346 004767 005574                JSR    PC,@POKE
2525 011352 105777 170334      TSTB   @RXCSR           ;RXDONE=1?
2526 011356 100401                BMI     .+4
2527 011360 104004                ERROR   4                ;RXDONE SHOULD NOW BE ASSERTED
2528 011362 012700 000025                MOV    #25,R0        ;EXPECTED
2529 011366 017701 170324      MOV    @RXDBUF,R1     ;ACTUAL
2530 011372 020001                CMP    R0,R1
2531 011374 001401                BEQ    .+4
2532 011376 104002                ERROR   2                ;CHARACTERS SHOULD BE MATCHED
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543 011400 000004
2544 011402 105767 167545      TSTB   JMRBY
2545 011406 100155                BPL     1$             ;GET OUT OF THIS TEST IF "NO"
2546 011410 016703 170302      MOV    @RXDBUF,R3     ;FOR ERROR MESSAGE
2547 011414 052777 000400 170304  BIS    @MRESET,@TXCSR ;MASTER RESET
2548 011422 012777 020000 170272  MOV    @SYNEXT,@PARCSR ;SET THE MODE
2549 011430 052777 000400 170270  BIS    @MRESET,@TXCSR ;MASTER RESET
2550
2551 ;SET MAINTENANCE MODE & SEND
2552 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
2553 011436 012777 010020 170262  MOV    @MEXT!SEND,@TXCSR
2554
2555 ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
2556 011444 012777 026026 170250  MOV    @SYNEXT!EIGHT!NOPAR!26,@PARCSR
2557 011452 052777 000020 170232  BIS    @SYNSCH,@RXCSR ;SET SEARCH SYNC
2558 011460 112777 000025 170244  MOV    #25,@TXDBUF   ;ANY CHARACTER
2559 ;POKE CLK FOR SYNCHRONIZATION
2560 011466 052777 020000 170232  BIS    @CLK,@TXCSR   ;POKE CLK UP
2561 ;WAIT FOR CABLE & DRIVER DELAYS
2562 011474 016702 167420      MOV    HOLD,R2       ;WAIT THIS AMT
2563 011500 005302                DEC    R2             ;WAIT
2564 011502 001376                BNE    .-2
2565 ;EXIT...
2566 011504 042777 020000 170214  BIC    @CLK,@TXCSR   ;POKE CLK DOWN
2567 ;WAIT FOR CABLE & DRIVER DELAYS
2568 011512 016702 167402      MOV    HOLD,R2       ;WAIT THIS AMT
2569 011516 005302                DEC    R2             ;WAIT
2570 011520 001376                BNE    .-2
2571 ;EXIT...
2572 011522 012767 000010 167372  MOV    #8,SHIFT      ;# OF SHIFTS
2573 011530
2574 011530 052777 020000 170170  BIS    @CLK,@TXCSR   ;POKE CLK UP

```

```

:: THIS TEST VERIFYS THAT HDX MODE DISQUALIFIES THE
:: RECEIVER WHEN SEND IS ASSERTED
:: MODE: SYNC EXT
:: LENGTH: EIGHT
:: NOTE: THIS TEST WORKS ONLY IN MAINT. EXTERNAL MODE
:: THIS TEST USES BOTH RECEIVER & TRANSMITTER LOGIC
:: *****

```

```

†ST25: SCOPE
TSTB   JMRBY
BPL     1$             ;GET OUT OF THIS TEST IF "NO"
MOV    @RXDBUF,R3     ;FOR ERROR MESSAGE
BIS    @MRESET,@TXCSR ;MASTER RESET
MOV    @SYNEXT,@PARCSR ;SET THE MODE
BIS    @MRESET,@TXCSR ;MASTER RESET

```

```

;SET MAINTENANCE MODE & SEND
;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
MOV    @MEXT!SEND,@TXCSR

```

```

;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
MOV    @SYNEXT!EIGHT!NOPAR!26,@PARCSR
BIS    @SYNSCH,@RXCSR ;SET SEARCH SYNC
MOV    #25,@TXDBUF   ;ANY CHARACTER
;POKE CLK FOR SYNCHRONIZATION
BIS    @CLK,@TXCSR   ;POKE CLK UP
;WAIT FOR CABLE & DRIVER DELAYS
MOV    HOLD,R2       ;WAIT THIS AMT
DEC    R2             ;WAIT
BNE    .-2
;EXIT...
BIC    @CLK,@TXCSR   ;POKE CLK DOWN
;WAIT FOR CABLE & DRIVER DELAYS
MOV    HOLD,R2       ;WAIT THIS AMT
DEC    R2             ;WAIT
BNE    .-2
;EXIT...

```

```

25: MOV    #8,SHIFT      ;# OF SHIFTS
BIS    @CLK,@TXCSR   ;POKE CLK UP

```

```

2575      ;WAIT FOR CABLE & DRIVER DELAYS
2576 011536 016702 167356  MOV    HOLD,R2 ;WAIT THIS AMT
2577 011542 005302  DEC    R2      ;WAIT
2578 011544 001376  BNE    .-2
2579      ;EXIT...
2580 011546 042777 020000 170152  BIC    #CLK,@TXCSR ;POKE CLK DOWN
2581      ;WAIT FOR CABLE & DRIVER DELAYS
2582 011554 016702 167340  MOV    HOLD,R2 ;WAIT THIS AMT
2583 011560 005302  DEC    R2      ;WAIT
2584 011562 001376  BNE    .-2
2585      ;EXIT...
2586 011564 005367 167332  DEC    SHIFT ;# OF SHIFTS
2587 011570 022767 000003 167324  CMP    #3,SHIFT ;IS IT TIME TO LOAD NEXT CHAR ?
2588 011576 001003  BNE    3$
2589 011600 112777 000024 170124  MOV    #24,@TXDBUF ;LOAD NEXT CHAR.
2590 011606 005767 167310 3$:  TST   SHIFT
2591 011612 001346  BNE    2$
2592 011614 105777 170072  TSTB  @RXCSR ;RXDONE=1?
2593 011620 100401  BMI   .+4
2594 011622 104004  ERROR 4 ;RXDONE SHOULD BE SET
2595 011624 012700 000025  MOV    #25,R0 ;EXPECTED
2596 011630 017701 170062  MOV    @RXDBUF,R1 ;ACTUAL
2597 011634 020001  CMP    R0,R1
2598 011636 001401  BEQ   .+4
2599 011640 104002  ERROR 2 ;CHARACTERS SHOULD COMPARE
2600 011642 052777 000010 170056  BIS    #HDXFN,@TXCSR ;SET HALF DUPLEX HDX
2601 011650 012767 000010 167244  MOV    #8.,SHIFT ;# OF SHIFT
2602      4$:
2603 011656 052777 020000 170042  BIS    #CLK,@TXCSR ;POKE CLK UP
2604      ;WAIT FOR CABLE & DRIVER DELAYS
2605 011664 016702 167230  MOV    HOLD,R2 ;WAIT THIS AMT
2606 011670 005302  DEC    R2      ;WAIT
2607 011672 001376  BNE    .-2
2608      ;EXIT...
2609 011674 042777 020000 170024  BIC    #CLK,@TXCSR ;POKE CLK DOWN
2610      ;WAIT FOR CABLE & DRIVER DELAYS
2611 011702 016702 167212  MOV    HOLD,R2 ;WAIT THIS AMT
2612 011706 005302  DEC    R2      ;WAIT
2613 011710 001376  BNE    .-2
2614      ;EXIT...
2615 011712 005367 167204  DEC    SHIFT
2616 011716 001357  BNE    4$
2617 011720 105777 167766  TSTB  @RXCSR ;RXDONE=0?
2618 011724 100001  BPL   .+4
2619 011726 104004  ERROR 4 ;RXDONE SHOULD NOT BE ASSERTED
2620      ;CHECK OUT HDX LOGIC
2621 011730 017701 167762  MOV    @RXDBUF,R1 ;ACTUAL
2622 011734 020001  CMP    R0,R1
2623 011736 001401  BEQ   .+4
2624 011740 104002  ERROR 2 ;CHARACTERS SHOULD COMPARE
2625      ;NOTE THAT CHARACTER 25 WILL BE FROZEN
2626      ;IN THE RXDBUF EVEN THOUGH CHARACTER 24 WAS
2627      ;SENT TO THE RECEIVER
2628
2629 011742 1$:
2630

```

```

2631
2632
2633
2634
2635
2636
2637
2638
2639
2640 011742 000004
2641 011744 052777 000400 167754
2642 011752 012777 020000 167742
2643 011760 052777 000400 167740
2644
2645
2646
2647 011766 012777 004020 167732
2648
2649
2650 011774 012777 026026 167720
2651 012002 052777 000020 167702
2652 012010 016703 167702
2653 012014 012767 000002 167102
2654 012022 112777 000025 167702
2655
2656 012030 052777 020000 167670
2657 012036 042777 020000 167662
2658 012044 012700 000025
2659 012050 012767 000010 167044 1$:
2660
2661 012056
2662 012056 052777 020000 167642 2$:
2663 012064 042777 020000 167634
2664 012072 005367 167024
2665 012076 022767 000003 167016
2666 012104 001003
2667 012106 112777 000024 167616 3$:
2668 012114 005767 167002
2669 012120 001356
2670 012122 105777 167564
2671 012126 100401
2672 012130 104004
2673 012132 017701 167560
2674 012136 020001
2675 012140 001401
2676 012142 104003
2677 012144 052777 000001 167554
2678 012152 012700 020000
2679 012156 005367 166742
2680 012162 001332
2681
2682
2683
2684
2685
2686
    ;; THIS TEST VERIFYS THAT BREAK FORCES A SPACE CONDITION
    ;; ON THE LINE WHILE TRANSMITTING
    ;; THIS TEST USES BOTH THE RECEIVER AND TRANSMITTLR LOGIC
    ;; MODE: SYNC EXT (SYNEXT)
    ;; LENGTH: EIGHT
    ;;
    ;; *****
    TST26: SCOPE
    BIS #MRESET,@TXCSR ;MASTER RESET
    MOV #SYNEXT,@PARCSR ;SET THE MODE
    BIS #MRESET,@TXCSR ;MASTER RESET

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

    ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
    MOV #SYNEXT!EIGHT!NOPAR!26,@PARCSR
    BIS #SYNSCH,@RXCSR ;SET SEARCH SYNC
    MOV RXDBUF,R3 ;FOR ERROR MESSAGE
    MOV #2,COUNT ;# OF TIMES
    MOVB #25,@TXDBUF ;ANY CHARACTER
    ;POKE CLK FOR SYNCRONIZATION
    BIS #CLK,@TXCSR ;POKE CLK UP
    BIC #CLK,@TXCSR ;POKE CLK DOWN
    MOV #25,R0 ;EXPECTED
    1$: MOV #8.,SHIFT ;# OF SHIFTS

    2$:
    BIS #CLK,@TXCSR ;POKE CLK UP
    BIC #CLK,@TXCSR ;POKE CLK DOWN
    DEC SHIFT
    CMP #3,SHIFT
    BNE 3$
    MOVB #24,@TXDBUF ;LOAD NEXT CHAR
    3$: TST SHIFT
    BNE 2$
    TSTB @RXCSR ;RXDONE=1?
    BMI .+4
    ERROR 4
    MOV @RXDBUF,R1 ;ACTUAL
    CMP R0,R1
    BEQ .+4
    ERROR 3
    BIS #BREAK,@TXCSR ;SET BREAK
    MOV #0,R0 ;EXPECTED
    DEC COUNT
    BNE 1$

    ;; THIS TEST VERIFYS THAT DSC CAUSES AN INTERRUPT
    ;; THIS TEST ONLY WORKS IN MAINT EXTERNAL MODE
    ;; INTERRUPT VECTOR: DURIV
    ;; *****
    
```

```

2687 012164 000004          TST27: SCOPE
2688 012166 105767 166761  TSTB      JMRBY          ; IN MAINT EXTERNAL?
2689 012172 100073          BPL       3$          ; IF ANSWER NO JUMP AROUND TEST
2690 012174 052777 000400 167524  BIS      @MRESET,@TXCSR ; MASTER RESET
2691 012202 105767 166743  TSTB      OPTCLR      ; IS THE OPTIONAL CLR JUMPER IN ?
2692 012206 100405          BMI       5$          ; YES
2693 012210 012777 000000 167474  MOV      @0,@RXCSR    ; CLR THE UNRESETTABLE BITS
2694 012216 005777 167470  TST      @RXCSR      ; GET RID OF DSC BY READING RXCSR
2695 012222 012777 012244 167506  5$: MOV    @4$,@OURIV   ; SET UP TRAPCATCHER
2696 012230 016777 004552 167502  MOV      DUPRT,@OURIS ;
2697 012236 106427 000000  MTPS     @0          ; ALLOW INTERRUPTS
2698 012242 000422          BR        1$          ; JUMP AROUND INTERRUPT SVC ROUTINE
2699                                     ; THE FOLLOWING IS THE INTERRUPT SVC ROUTINE
2700 012244 106427 000340  4$: MTPS   @340       ; DON'T ALLOW ANYMORE INTERRUPTS
2701 012250 005777 167436  TST      @RXCSR      ; DSC=1?
2702 012254 100401          BMI       .+4
2703 012256 104004          ERROR    4          ; FALSE INTERRUPT
2704 012260 042777 000040 167424  BIC      @DSINTE,@RXCSR ; CLEAR INTERRUPT ENABLE
2705 012266 012716 012356  MOV      @2$,(SP)     ; SET UP RETURN LOCATION
2706 012272 016777 167442 167436  MOV      @OURIS,@OURIV ; RESTORE TRAPCATCHER
2707 012300 012777 000000 167432  MOV      @0,@OURIS   ;
2708 012306 000002          RTI
2709
2710 012310 052777 000040 167374  1$: BIS    @DSINTE,@RXCSR ; SET INTERRUPT ENABLE
2711 012316 052777 000002 167366  BIS     @DTR,@RXCSR   ; TRY TO CAUSE INTERRUPT
2712 012324 005000          CLR      R0
2713 012326 005200          INC     R0          ; WAIT FOR INTERRUPT
2714 012330 001376          BNE     .-2
2715 012332 016777 167402 167376  MOV     @OURIS,@OURIV ; RESTORE TRAPCATCHER
2716 012340 012777 000000 167372  MOV     @0,@OURIS   ;
2717
2718 012346 042777 000040 167336  BIC     @DSINTE,@RXCSR ; CLEAR INTERRUPT ENABLE
2719 012354 104004          ERROR    4          ; INTERRUPT FAILED TO OCCUR
2720 012356 106427 000340  2$: MTPS   @340
2721 012362
2722
2723
2724                                     ; END OF PASS
2725                                     ; TYPE NAME OF TEST
2726                                     ; UPDATE PASS COUNT
2727                                     ; CHECK FOR EXIT TO ACT-11
2728                                     ; RESTART TEST
2729
2730 012362 000004          .EOP: SCOPE
2731 012364 004767 000344  JSR     PC,CKSWR
2732 012370 104401          TYPE
2733 012372 015524          MEPASS
2734 012374 104413 012626  CONVRT  ,OUTCRY
2735 012400 104401 015343  TYPE    ,DEVICE
2736 012404 105767 166542  TSTB   MULTD        ; ARE YOU RUNNING MULTIPLE DEVICES ?
2737 012410 001511          BEQ     CCC          ; NO, JUMP AROUND
2738 012412 005767 166550  TST    ACTREG       ; ARE ANY DEVICES ACTIVE ?
2739 012416 001007          BNE     RUNIT       ; YES
2740 012420 104401 015355  TYPE   MCOW         ; NO
2741 012424 016700 166536  MOV    ACTREG,R0    ; DISPLAY ACTREG
2742 012430 000000          HALT   ; SELECT SOMETHING TO RUN @ ACTREG:
    
```

INITIALIZE THE COMMON TAGS

```

2743          :SELECT SWITCHES & HIT CONTINUE (PUT SW00 =1)
2744 012432 000167 167514      JMP      .START .START OVER AGAIN.....YOU DESELECTED EVERYTHING
2745 012436 062767 000010 166510 RUNIT:  ADD      #10,BASEADD ;NEXT BLOCK (ADDRESSES)
2746 012444 062767 000010 166510 ZERO:   ADD      #10,BASEIV  ;NEXT BLOCK (VECTORS)
2747 012452 000241          CLC
2748 012454 006167 166510      ROL      ROTADD ;UP DATE ROTATING POINTER
2749 012460 103410          BCS      2$      ;IS IT THE LAST DEVICE
2750          :TO BE TESTED IN THIS PASS ?
2751 012462 036767 166502 166476      BIT      ROTADD,ACTREG ;TEST THIS DEVICE FOR ACTIVE STATUS
2752 012470 001762          BEQ      RUNIT ;IF NOT ACTIVE, TRY NEXT ADDRESS
2753 012472 004767 000034          JSR      PC,REPLAY ;CALCULATE NEW PARAMETERS
2754 012476 000167 000210          JMP      RESTRT ;YES IT WAS ACTIVE,TEST THIS DEVICE
2755 012502 012767 000001 166460 2$:   MOV      #1,ROTADD ;OK!,NOW SET UP ROTATING
2756          :POINTER FOR NEXT MULTIPLE PASS
2757 012510 016767 166442 166436      MOV      KEEPADD,BASEADD ;RESTORE BASE ADDRESS
2758 012516 016767 166442 166436      MOV      KEEPIV,BASEIV ;RESTORE BASE INTERRUPT VECTORS
2759 012524 004767 000002          JSR      PC,REPLAY ;CALC NEW PARAMETERS
2760 012530 000441          BR       CCC ;JUMP AROUND REPLAY
2761 012532 016767 166416 004404 REPLAY: MOV      BASEADD,DUBASE ;SET UP FOR NEW ADDRESSES
2762 012540 004767 004246          JSR      PC,DUADD ;CREATE NEW ADDRESSES
2763 012544 016767 166412 167164      MOV      BASEIV,DURIV ;CREATE DURIV
2764 012552 062767 000002 166402      ADD      #2,BASEIV
2765 012560 016767 166376 167152      MOV      BASEIV,DURIS ;CREATE DURIS
2766 012566 062767 000002 166366      ADD      #2,BASEIV
2767 012574 016767 166362 167140      MOV      BASEIV,DUTIV ;CREATE DUTIV
2768 012602 062767 000002 166352      ADD      #2,BASEIV
2769 012610 016767 166346 167126      MOV      BASEIV,DUTIS ;CREATE DUTIS
2770 012616 016767 167114 166336      MOV      DURIV,BASEIV ;RESTORE
2771 012624 000207          RTS      PC
2772
2773 012626 000001          OUTCRY: 1
2774 012630 006          002      .BYTE 6,2
2775 012632 001712          RXCSR
2776
2777          CCC:
2778 012634 005067 166542          CLR      $STNM ;CLEAR TEST NUMBER
2779 012640 005067 166552          CLR      $ERRPC ;CLEAR LAST ERROR PC
2780 012644 005067 166533          CLR      $ERFLG ;CLEAR ERROR FLAG
2781 012650 005267 166236          INC      PASCNT ;UPDATE PASS COUNT
2782 012654 016767 166232 166220      MOV      PASCNT,LIGHTS ;DISPLAY PASS COUNT
2783 012662 016767 166224 166644      MOV      PASCNT,$PASS ;PASS COUNT TO APT
2784 012670 013701 000042          MOV      #42,R1 ;CHECK FOR ACT-11 OR DDP
2785 012674 001406          BEQ      RESTRT ;IF NO CONTINUE TESTING
2786 012676 000005          RESET
2787 012700 000005          RESET
2788 012702 004711          SENDAD: JSR      PC,(R1)
2789 012704 000240          NOP
2790 012706 000240          NOP
2791 012710 000240          NOP
2792 012712 106427 000340          RESTRT: MTPS   #340 ;PREVENT INTERRUPTS (PRIO: 7)
2793 012716 004767 000012          JSR      PC,CKSWR
2794 012722 012767 003364 166456      MOV      #TST1+2,$LPADR ;SET LAST ADDRESS POINTER
2795 012730 000167 170426          JMP      TST1
2796
2797          ;CHECK SWITCH REGISTER ROUTINE.
2798          ;CHECKS TO ALLOW FOR <+G> TO ALLOW

```



```

2799
2800
2801 012734 005737 000042
2802 012740 001040
2803 012742 022767 000176 166470
2804 012750 001034
2805 012752 105777 166466
2806 012756 100031
2807 012760 017767 166462 000422
2808 012766 042767 177600 000414
2809 012774 122767 000007 000406
2810 013002 001017
2811 013004 104401 016131
2812 013010 005137 013050
2813 013014 104401 016141
2814 013020 104413
2815 013022 013052
2816 013024 104406 016152
2817 013030 104410
2818 013032 000000
2819 013034 177777
2820 013036 000176
2821 013040 000 001
2822 013042 005037 013050
2823 013046 000207
2824 013050 000000
2825 013052 000001
2826 013054 006 002
2827 013056 000176
2828
2829 013060 000005
2830
2831
2832
2833 013062 004767 177646
2834 013066 032777 001000 166344
2835 013074 001402
2836 013076 016716 166006
2837 013102 000002
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854

```

```

;THE CHANGING OF LOCATION 176
CKSWR: TST 0042
        BNE OUT
        CMP #SWREG,SWR ;SOFTWARE SWR PRESENT?
        BNE OUT ;NO--LEAVE
        TSTB #STKS ;CHECK TTY READY
        BPL OUT ;NO--LEAVE
        MOV #STKB,MSG ;GET CHARACTER
        BIC #177600,.MSG ;STRIP JUNK
        CMPB #7,MSG ;IS IT <IG> ?
        BNE OUT ;NO
CNTLU: COM #RDSW
        TYPE ,MMSWR
        CONVRT
        SWREGL
        INSTR,MMNEW
        PARAM
        0
        177777
        SWREG
        .BYTE 0,1
        OUT: CLR #RDSW
        RTS PC
        RDSW: .WORD 0
        SWREGL: 1
        .BYTE 6,2
        SWREG
        5
;CHECK FOR FREEZE ON CURRENT DATA
.SCOPI: JSR PC,CKSWR
        BIT #SW09,#SWR
        BEQ IS
        MOV LOCK,(SP)
IS: RTI
.SBTTL TYPE ROUTINE
;*****
;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
;NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
;
;CALL:
;1) USING A TRAP INSTRUCTION
; TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
;OR
; TYPE
; MESADR
;

```

2855	013104	105767	166347		\$TYPE:	TSTB	\$TPFLG	;; IS THERE A TERMINAL?
2856	013110	100002				BPL	1\$	BR IF YES
2857	013112	000000				HALT		HALT HERE IF NO TERMINAL
2858	013114	000430				BR	3\$	LEAVE
2859	013116	010046			1\$:	MOV	RO, -(SP)	SAVE RO
2860	013120	017600	000002			MOV	22(SP), RO	GET ADDRESS OF ASCIZ STRING
2861	013124	122767	000001	166414		CMPB	#APTENV, \$ENV	RUNNING IN APT MODE
2862	013132	001011				BNE	62\$	NO GO CHECK FOR APT CONSOLE
2863	013134	132767	000100	166405		BITB	#APTSPOOL, \$ENVM	SPOOL MESSAGE TO APT
2864	013142	001405				BEQ	62\$	NO GO CHECK FOR CONSOLE
2865	013144	010067	000004			MOV	RO, 61\$	SETUP MESSAGE ADDRESS FOR APT
2866	013150	004767	000006			JSR	PC, \$ATY3	SPOOL MESSAGE TO APT
2867	013154	000000			61\$:	.WORD	0	MESSAGE ADDRESS
2868	013156	132767	000040	166363	62\$:	BITB	#APTCSUP, \$ENVM	APT CONSOLE SUPPRESSED
2869	013164	001003				BNE	60\$	YES, SKIP TYPE OUT
2870	013166	112046			2\$:	MOVB	(RO)+, -(SP)	PUSH CHARACTER TO BE TYPED ONTO STACK
2871	013170	001005				BNE	4\$	BR IF IT ISN'T THE TERMINATOR
2872	013172	005726				TST	(SP)+	IF TERMINATOR POP IT OFF THE STACK
2873	013174	012600			60\$:	MOV	(SP)+, RO	RESTORE RO
2874	013176	062716	000002		3\$:	ADD	#2, (SP)	ADJUST RETURN PC
2875	013202	000002				RTI		RETURN
2876	013204	122716	000011		4\$:	CMPB	#HT, (SP)	BRANCH IF <HT>
2877	013210	001430				BEQ	8\$	
2878	013212	122716	000200			CMPB	#CRLF, (SP)	;; BRANCH IF NOT <CRLF>
2879	013216	001006				BNE	5\$	
2880	013220	005726				TST	(SP)+	;; POP <CR><LF> EQUIV
2881	013222	104401				TYPE		;; TYPE A CR AND LF
2882	013224	001523				\$CRLF		
2883	013226	105067	000130			CLRB	\$CHARCNT	;; CLEAR CHARACTER COUNT
2884	013232	000755				BR	2\$;; GET NEXT CHARACTER
2885	013234	004767	000056		5\$:	JSR	PC, \$TYPEC	;; GO TYPE THIS CHARACTER
2886	013240	126726	166212		6\$:	CMPB	\$FILLC, (SP)+	;; IS IT TIME FOR FILLER CHARS.?
2887	013244	001350				BNE	2\$;; IF NO GO GET NEXT CHAR.
2888	013246	016746	166202			MOV	\$NULL, -(SP)	;; GET # OF FILLER CHARS. NEEDED
2889								AND THE NULL CHAR.
2890	013252	105366	000001		7\$:	DECB	1(SP)	;; DOES A NULL NEED TO BE TYPED?
2891	013256	002770				BLT	6\$;; BR IF NO--GO POP THE NULL OFF OF STACK
2892	013260	004767	000032			JSR	PC, \$TYPEC	;; GO TYPE A NULL
2893	013264	105367	000072			DECB	\$CHARCNT	;; DO NOT COUNT AS A COUNT
2894	013270	000770				BR	7\$;; LOOP
2895								
2896								
2897								
2898	013272	112716	000040		8\$:	MOVB	#' (SP)	;; REPLACE TAB WITH SPACE
2899	013276	004767	000014		9\$:	JSR	PC, \$TYPEC	;; TYPE A SPACE
2900	013302	132767	000007	000052		BITB	#7, \$CHARCNT	;; BRANCH IF NOT AT
2901	013310	001372				BNE	9\$;; TAB STOP
2902	013312	005726				TST	(SP)+	;; POP SPACE OFF STACK
2903	013314	000724				BR	2\$;; GET NEXT CHARACTER
2904	013316	105777	166126		\$TYPEC:	TSTB	2\$TPS	;; WAIT UNTIL PRINTER IS READY
2905	013322	100375				BPL	\$TYPEC	
2906	013324	116677	000002	166120		MOVB	2(SP), 2\$TPB	;; LOAD CHAR TO BE TYPED INTO DATA REG.
2907	013332	122766	000015	000002		CMPB	#CR, 2(SP)	;; IS CHARACTER A CARRIAGE RETURN?
2908	013340	001003				BNE	1\$;; BRANCH IF NO
2909	013342	105067	000014			CLRB	\$CHARCNT	;; YES--CLEAR CHARACTER COUNT
2910	013346	000406				BR	\$TYPEX	;; EXIT

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 60
 DZDUUA.M11 13-OCT-76 08:28 TYPE ROUTINE

```

2911 013350 122766 000012 000002 1S:  CMPB  #LF,2(SP)      ; IS CHARACTER A LINE FEED?
2912 013356 001402          BEQ  $TYPEX      ; BRANCH IF YES
2913 013360 105227          INCB  (PC)+      ; COUNT THE CHARACTER
2914 013362 000000          $CHARCNT:..WORD 0      ; CHARACTER COUNT STORAGE
2915 013364 000207          $TYPEX: RTS     PC
    
```

;ASCII STRING INPUT ROUTINE

```

2920 013366 017667 000000 000014 .INSTR: MOV  2(SP),MSG      ; PICK UP MESSAGE
2921 013374 062716 000002          ADD  #2,(SP)      ; JUMP AROUND MESSAGE FOR RTI
2922 013400 105767 166142          TSTB $ENV        ; APT CONTROL
2923 013404 001036          BNE  INSTR2      ; YES NO TYPE
2924 013406 104401          .INST1: TYPE
2925 013410 000000          .MSG:  0
2926 013412 012704 016164          MOV  #INBUF,R4   ; GET STARTING LOC OF INBUF
2927 013416 012703 000007          MOV  #7,R3       ; MAX # OF CHARS
2928 013422 105777 166016 1S:  TSTB 2$TKS      ; TTY FLAG
2929 013426 100375          BPL  1$
2930 013430 117714 166012          MOVB 2$TKB,(R4)  ; TAKE CHAR
2931 013434 142714 000200          BICB #200,(R4)   ; STRIP
2932 013440 121427 000025          CMPB (R4),#25    ; IS IT <↑G>
2933 013444 001760          BEQ  .INST1
2934 013446 122427 000015          CMPB (R4)+,#15   ; CHECK FOR CR
2935 013452 001413          BEQ  INSTR2
2936 013454 105777 165770 2S:  TSTB 2$TPS      ; TEST FLAG
2937 013460 100375          BPL  2$
2938 013462 117777 165760 165762          MOVB 2$TKB,2$TPB ; ECHO CHARACTER
2939 013470 005303          DEC  R3          ; DID YOU TYPE TOO MANY CHARS ?
2940 013472 001353          BNE  1$
2941 013474 104401          .INSTE: TYPE
2942 013476 015451          MQM  ;?
2943 013500 000742          BR   .INST1      ; RETRY
2944 013502 000002          INSTR2: RTI
    
```

;CONVERT ASCII STRING TO OCTAL

```

2948 013504 011605          .PARAM: MOV  (SP),R5 ; PUT CONTENTS OF SP INTO R5
2949 013506 012567 000162          MOV  (R5)+,LOLIM ; PUT LOW LIMIT INTO LOLIM
2950 013512 012567 000160          MOV  (R5)+,HILIM ; PUT HIGH LIMIT INTO HILIM
2951 013516 012567 000156          MOV  (R5)+,DEVADR ; PUT STORE LOC INTO DEVADR
2952 013522 112567 000154          MOVB (R5)+,LOBITS ; PUT MASK INTO LOBITS
2953 0135 6 112567 000151          MOVB (R5)+,ADRCNT ; PUT COUNT INTO ADRCNT
2954 013 3 7 010516          MOV  R5,(SP) ; RESTORE RETURN ADDR ON STACK FOR RTI
2955 013 3 4 005005          PARAM1: CLR  R5
2956 013536 012704 016164          MOV  #INBUF,R4
2957 013542 122714 000015          CMPB #15,(R4)   ; CR ?
2958 013546 001420          BEQ  PARERR      ; YOU TYPED CR TOO SOON !
2959 013550 121427 000060 1S:  CMPB (R4),#60    ; LOW LIMIT ASCII 0
2960 013554 002415          BLT  PARERR
2961 013556 121427 000067          CMPB (R4),#67    ; HIGH LIMIT ASCII 7
2962 013562 003012          BGT  PARERR
2963 013564 142714 000060          BICB #60,(R4)   ; CONVERT TO OCTAL
2964 013570 152405          BISB (R4)+,R5    ; STORE AWAY ITS AN OK CHAR
2965 013572 122714 000015          CMPB #15,(R4)   ; CR ?
2966 013576 001414          BEQ  LIMITS      ; NOW CHECK FOR HIGH & LOW LIMIT CONDS
    
```

H05

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 61
 DZDUUA.M11 13-OCT-76 08:28 TYPE ROUTINE

2967	013600	006305		ASL	R5	;ALLOCATE ROOM FOR NEXT CHAR
2968	013602	006305		ASL	R5	
2969	013604	006305		ASL	R5	
2970	013606	000760		BR	1\$	
2971	013610	122714	000015	PARERR: CMPB	#15, (R4)	;CR?
2972	013614	001003		BNE	120\$	
2973	013616	005737	013050	TST	#RDSW	;CK SWR USED
2974	013622	001023		BNE	PARTI	
2975	013624	104407		120\$: INSTER	:RETRY	
2976	013626	000742		BR	PARAM1	
2977						
2978						;TEST TO SEE IF NUMBER IS WITHIN LIMITS
2979						
2980	013630	020567	000042	LIMITS: CMP	R5, HILIM	
2981	013634	101365		BHI	PARERR	;THE # IS TOO HIGH
2982	013636	020567	000032	CMP	R5, LOLIM	
2983	013642	103762		BLO	PARERR	;THE # IS TOO LOW
2984	013644	136705	000032	BITB	LOBITS, R5	;TEST BY MASKINGTHE #
2985	013650	001357		BNE	PARERR	
2986						
2987						;STORE NUMBER AT SPECIFIED ADDRESS
2988						
2989	013652	016704	000022	1\$: MOV	DEVAOR, R4	;GET STARTING ADDR OF
2990	013656	010524		MOV	R5, (R4)+	;STORE AT THIS ADDR
2991	013660	062705	000002	ADD	#2, R5	
2992	013664	105367	000013	DECB	ADRCNT	;HOW MANY TIMES + 2 ?
2993	013670	001372		BNE	1\$	
2994	013672	000002		PARTI: RTI		
2995	013674	000000		LOLIM: 0		
2996	013676	000000		HILIM: 0		
2997	013700	000000		DEVAOR: 0		
2998	013702	000000		LOBITS: 0		
2999		013703		ADRCNT=LOBITS+1		
3000						
3001						;SAVE PC OF TEST THAT FAILED AND R0-R5
3002						
3003	013704	016667	000004	165214 .SAV05: MOV	4(SP), SAVPC	
3004						
3005						;SAVE R0-R5
3006						
3007	013712	010567	165556	SV05: MOV	R5, \$REG5	
3008	013716	010467	165550	MOV	R4, \$REG4	
3009	013722	010367	165542	MOV	R3, \$REG3	
3010	013726	010267	165534	MOV	R2, \$REG2	
3011	013732	010167	165526	MOV	R1, \$REG1	
3012	013736	010067	165520	MOV	R0, \$REG0	
3013	013742	000002		RTI		
3014						
3015						;RESTORE R0-R5
3016						
3017	013744	016700	165512	.RES05: MOV	\$REG0, R0	
3018	013750	016701	165510	MOV	\$REG1, R1	
3019	013754	016702	165506	MOV	\$REG2, R2	
3020	013760	016703	165504	MOV	\$REG3, R3	
3021	013764	016704	165502	MOV	\$REG4, R4	
3022	013770	016705	165500	MOV	\$REG5, R5	

```

3023 013774 000002          RTI
3024
3025                          ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
3026
3027 013776 104401          .CONVR: TYPE
3028 014000 015455          MCRLF      ;CR LF
3029 014002 017601 000000  MOV      2(SP),R1      ;PICK UP DATA POINTER
3030 014006 062716 000002  ADD      #2,(SP)      ;SET UP SP FOR RTI
3031 014012 012167 000130  MOV      (R1)+,WRDCNT ;PICK UP # OF WORDS FROM TABLE
3032 014016 112167 000126 1$:      MOV      (R1)+,CHRCNT ;PICK UP # OF CHARS FROM TABLE
3033 014022 112167 000123  MOV      (R1)+,SPACNT ;PICK UP # OF SPACES FROM TABLE
3034 014026 013167 000120  MOV      2(R1)+,BINWRD ;PICK UP ADDRESS OF MSG
3035                          ;FROM TABLE
3036 014032 016704 000114 2$:      MOV      BINWRD,R4      ;SAVE
3037 014036 116705 000106  MOV      CHRCNT,R5      ;SAVE
3038 014042 012700 016226  MOV      #TEMP,R0      ;STARTING ADDRESS OF TEMP BLOCK
3039 014046 010403 3$:      MOV      R4,R3          ;SAVE
3040 014050 042703 177770  BIC      #177770,R3     ;CLR OUT UPPER BITS .. SAVE CHAR
3041 014054 062703 000260  ADD      #260,R3      ;CONVERT TO ASCII
3042 014060 110320  MOV      R3,(R0)+      ;STORE AWAY
3043 014062 006204  ASR      R4            ;SHIFT FOR NEXT #
3044 014064 006204  ASR      R4            ;DITTO
3045 014066 006204  ASR      R4            ;DITTO
3046 014070 005305  DEC      R5            ;DEC CHAR COUNT
3047 014072 001365  BNE      3$           ;DO IT AGAIN ?
3048 014074 012703 016270  MOV      #MDATA,R3     ;STARTING ADDRESS OF MDATA BLOCK
3049 014100 114023 4$:      MOV      -(R0),(R3)+    ;REVERSE THE ORDER OF NUMBERS
3050 014102 105367 000042  DECB    CHRCNT        ;DEC CHAR COUNT
3051 014106 001374  BNE      4$           ;DO IT AGAIN ?
3052 014110 105767 000035  TSTB    SPACNT        ;HOW MANY SPACES ?
3053 014114 001405  BEQ      6$           ;TYPE # IF BR = 0
3054 014116 112723 000240 5$:      MOV      #240,(R3)+    ;"SPACE" IN ASCII
3055 014122 105367 000023  DECB    SPACNT        ;DEC # OF SPACE COUNT
3056 014126 001373  BNE      5$           ;DO IT AGAIN ?
3057 014130 105013 6$:      CLRB    (R3)         ;INSERT "0" FOR TTY OUTPUT ROUTINE
3058 014132 104401  TYPE
3059 014134 016270  MDATA      ;THIS MESSAGE
3060 014136 005367 000004  DEC      WRDCNT        ;HOW MANY #'S ?
3061 014142 001325  BNE      1$           ;DO THIS ROUTINE AGAIN IF NOT EQUAL TO 0
3062 014144 000002  RTI          ;RETURN TO PROGRAM
3063 014146 000000  WRDCNT: 0
3064 014150 000000  CHRCNT: 0
3065 014151 014151  SPACNT=CHRCNT+1
3066 014152 000000  BINWRD: 0
3067
3068                          ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
3069                          ;BUFFER TO THE CHARACTERS "N" AND "Y"
3070                          ;IF THE CHARACTER IS "N" CLEAR THE FLAG
3071                          ;IF THE CHARACTER IS "Y" SET THE FLAG
3072
3073 014154 017605 000000  .SETFLG: MOV      2(SP),R5
3074 014160 122767 000116 001776  CMPB    #'N,INBUF      ;IS IT "N" ?
3075 014166 001002  BNE      1$
3076 014170 105015  CLRB    (R5)          ;000
3077 014172 000406  BR      2$
3078 014174 122767 000131 001762 1$:      CMPB    #'Y,INBUF      ;IS IT "Y" ?
    
```

```

3079 014202 001005          BNE      3$
3080 014204 112715 177777    MOVB     #-1,(R5)          ;377
3081 014210 062716 000002    2$:     ADD      #2,(SP)
3082 014214 000002          RTI
3083 014216 104407          3$:     INSTER ;RETRY
3084 014220 000755          BR       .SETFLG
3085          .SBTTL  ERROR HANDLER ROUTINE
3086
3087          ;*****
3088          ;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3089          ;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3090          ;AND GO TO SAVIT ON ERROR
3091          ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3092          ;*SW15=1      HALT ON ERROR
3093          ;*SW13=1      INHIBIT ERROR TYPEOUTS
3094          ;*SW10=1      BELL ON ERROR
3095          ;*SW09=1      LOOP ON ERROR
3096          ;*CALL
3097          ;*      ERROR      N          ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3098
3099          $ERROR:
3100 014222 105267 165155    7$:     INCB     $ERFLG          ;; SET THE ERROR FLAG
3101 014226 001775          BEQ      7$                  ;; DON'T LET THE FLAG GO TO ZERO
3102 014230 016777 165146 165204    MOV      $STNM,$DISPLAY      ;; DISPLAY TEST NUMBER AND ERROR FLAG
3103 014236 032777 002000 165174    BIT      #BIT10,$SWR          ;; BELL ON ERROR?
3104 014244 001402          BEQ      1$                  ;; NO - SKIP
3105 014246 104401 001516          TYPE     $BELL              ;; RING BELL
3106 014252 005267 165134          1$:     INC      $ERTTL          ;; COUNT THE NUMBER OF ERRORS
3107 014256 011667 165134          MOV      (SP),$ERRPC          ;; GET ADDRESS OF ERROR INSTRUCTION
3108 014262 162767 000002 165126    SUB      #2,$ERRPC
3109 014270 117767 165122 165116    MOVB     2,$ERRPC,$ITEMB      ;; STRIP AND SAVE THE ERROR ITEM CODE
3110 014276 032777 020000 165134    BIT      #BIT13,$SWR          ;; SKIP TYPEOUT IF SET
3111 014304 001004          BNE      20$                 ;; SKIP TYPEOUTS
3112 014306 004767 000072          JSR     PC,SAVIT              ;; GO TO USER ERROR ROUTINE
3113 014312 104401 001523          TYPE     ,SCLF
3114 014316
3115 014316 122767 000001 165222    20$:    CMPB     #APTENV,$ENV          ;; RUNNING IN APT MODE
3116 014324 001007          BNE      2$                  ;; NO SKIP APT ERROR REPORT
3117 014326 116767 165062 000004    MOVB     $ITEMB,21$           ;; SET ITEM NUMBER AS ERROR NUMBER
3118 014334 004767 000016          JSR     PC,$ATY4              ;; REPORT FATAL ERROR TO APT
3119 014340 000
3120 014341 000
3121 014342 000777          21$:    .BYTE     0
3122 014344 005777 165070          .BYTE     0
3123 014350 100001          22$:    BR       22$                ;; APT ERROR LOOP
3124 014352 000000          2$:     TST      2$SWR          ;; HALT ON ERROR
3125 014354 032777 001000 165056    BPL      3$                  ;; SKIP IF CONTINUE
3126 014362 001402          HALT     ;                   ;; HALT ON ERROR!
3127 014364 016716 165020          3$:     BIT      #BIT09,$SWR      ;; LOOP ON ERROR SWITCH SET?
3128 014370 005767 165120          BEQ     4$                  ;; BR IF NO
3129 014374 001402          MOV      $LPERR,(SP)          ;; FUDGE RETURN FOR LOOPING
3130 014376 016716 165112          4$:     TST      $ESCAPE          ;; CHECK FOR AN ESCAPE ADDRESS
3131 014402          BEQ     5$                  ;; BR IF NONE
3132 014402 000002          MOV      $ESCAPE,(SP)        ;; FUDGE RETURN ADDRESS FOR ESCAPE
3133 014404 010067 164520          5$:
3134 014410 010167 164516          RTI
          SAVIT: MOV      R0,HLD0          ;; RETURN
          MOV      R1,HLD1
    
```

3135 014414 010267 164514
 3136 014420 010367 164512
 3137 014424 010467 164510
 3138 014430 010567 164506
 3139 014434 016767 164742 164502

MOV R2,HL02
 MOV R3,HL03
 MOV R4,HL04
 MOV R5,HL05
 MOV \$STNM,HL06

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

3140
 3141
 3142
 3143
 3144
 3145
 3146
 3147
 3148 014442
 3149 014442 104401 001523
 3150 014446 010046
 3151 014450 005000
 3152 014452 153700 001414
 3153 014456 001004
 3154
 3155 014460 016746 164732
 3156
 3157 014464 104402
 3158 014466 000426
 3159 014470 005300
 3160 014472 006300
 3161 014474 006300
 3162 014476 006300
 3163 014500 062700 001652
 3164 014504 012067 000004
 3165 014510 001404
 3166 014512 104401
 3167 014514 000000
 3168 014516 104401 001523
 3169 014522 012067 000004
 3170 014526 001404
 3171 014530 104401
 3172 014532 000000
 3173 014534 104401 001523
 3174 014540 011000
 3175 014542 001004
 3176 014544 012600
 3177 014546 104401 001523
 3178 014552 000207
 3179 014554
 3180 014554 013046
 3181 014556 104402
 3182 014560 005710
 3183 014562 001170
 3184 014564 001170 014572
 3185 014570 000171
 3186 014572 020040 000
 3187 014576
 3188
 3189
 3190

 *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.

\$ERRTYP:

	TYPE	\$CRLF	;; "CARRIAGE RETURN" & "LINE FEED"
	MOV	RO,-(SP)	;; SAVE RO
	CLR	RO	;; PICKUP THE ITEM INDEX
	BISB	2*\$ITEMB,RO	
	BNE	1\$;; IF ITEM NUMBER IS ZERO, JUST
			;; TYPE THE PC OF THE ERROR
	MOV	\$ERRPC,-(SP)	;; SAVE \$ERRPC FOR TYPEOUT
			;; ERROR ADDRESS
	TYPOC		;; GO TYPE--OCTAL ASCII(ALL DIGITS)
	BR	6\$;; GET OUT
1\$:	DEC	RO	;; ADJUST THE INDEX SO THAT IT WILL
	ASL	RO	;; WORK FOR THE ERROR TABLE
	ASL	RO	
	ASL	RO	
	ADD	#\$ERRTB,RO	;; FORM TABLE POINTER
	MOV	(RO)+,2\$;; PICKUP "ERROR MESSAGE" POINTER
	BEQ	3\$;; SKIP TYPEOUT IF NO POINTER
	TYPE		;; TYPE THE "ERROR MESSAGE"
2\$:	.WORD	0	;; "ERROR MESSAGE" POINTER GOES HERE
	TYPE	\$CRLF	;; "CARRIAGE RETURN" & "LINE FEED"
3\$:	MOV	(RO)+,4\$;; PICKUP "DATA HEADER" POINTER
	BEQ	5\$;; SKIP TYPEOUT IF 0
	TYPE		;; TYPE THE "DATA HEADER"
4\$:	.WORD	0	;; "DATA HEADER" POINTER GOES HERE
	TYPE	\$CRLF	;; "CARRIAGE RETURN" & "LINE FEED"
5\$:	MOV	(RO),RO	;; PICKUP "DATA TABLE" POINTER
	BNE	7\$;; GO TYPE THE DATA
6\$:	MOV	(SP)+,RO	;; RESTORE RO
	TYPE	\$CRLF	;; "CARRIAGE RETURN" & "LINE FEED"
	RTS	PC	;; RETURN
7\$:			
	MOV	2(RO)+,-(SP)	;; SAVE 2(RO)+ FOR TYPEOUT
	TYPOC		;; GO TYPE--OCTAL ASCII(ALL DIGITS)
	TST	(RO)	;; IS THERE ANOTHER NUMBER?
	BEQ	6\$;; BR IF NO
	TYPE	8\$;; TYPE TWO(2) SPACES
	BR	7\$;; LOOP
8\$:	.ASCIZ	/ /	;; TWO(2) SPACES
	.EVEN		

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

;;*****

L05

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 65
 DZDUUA.M11 13-OCT-76 08:28 BINARY TO OCTAL (ASCII) AND TYPE

```

3191      ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3192      ;*OCTAL (ASCII) NUMBER AND TYPE IT.
3193      ;*STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3194      ;*CALL:
3195      ;*      MOV      NUM,-(SP)      ;:NUMBER TO BE TYPED
3196      ;*      TYPOS      ;:CALL FOR TYPEOUT
3197      ;*      .BYTE  N      ;:N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3198      ;*      .BYTE  M      ;:M=1 OR 0
3199      ;*                               ;:1=TYPE LEADING ZEROS
3200      ;*                               ;:0=SUPPRESS LEADING ZEROS
3201      ;*
3202      ;*STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3203      ;*STYPOS OR STYPOC
3204      ;*CALL:
3205      ;*      MOV      NUM,-(SP)      ;:NUMBER TO BE TYPED
3206      ;*      TYPON      ;:CALL FOR TYPEOUT
3207      ;*
3208      ;*STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3209      ;*CALL:
3210      ;*      MOV      NUM,-(SP)      ;:NUMBER TO BE TYPED
3211      ;*      TYPOC      ;:CALL FOR TYPEOUT
3212
3213 014576 017646 000000      STYPOS: MOV      2(SP),-(SP)      ;:PICKUP THE MODE
3214 014602 116667 000001 000211  MOVB     1(SP),SOFILL ;:LOAD ZERO FILL SWITCH
3215 014610 112667 000207      MOVB     (SP)+,SOMODE+1 ;:NUMBER OF DIGITS TO TYPE
3216 014614 062716 000002      ADD      #2,(SP)      ;:ADJUST RETURN ADDRESS
3217 014620 000406      BR      STYPON
3218 014622 112767 000001 000171  STYPOC: MOVB     #1,SOFILL ;:SET THE ZERO FILL SWITCH
3219 014630 112767 000006 000165  MOVB     #6,SOMODE+1 ;:SET FOR SIX(6) DIGITS
3220 014636 112767 000005 000154  STYPON: MOVB     #5,SOCNT ;:SET THE ITERATION COUNT
3221 014644 010346      MOV      R3,-(SP)      ;:SAVE R3
3222 014646 010446      MOV      R4,-(SP)      ;:SAVE R4
3223 014650 010546      MOV      R5,-(SP)      ;:SAVE R5
3224 014652 116704 000145      MOVB     SOMODE+1,R4 ;:GET THE NUMBER OF DIGITS TO TYPE
3225 014656 005404      NEG      R4
3226 014660 062704 000006      AND     #6,R4      ;:SUBTRACT IT FOR MAX. ALLOWED
3227 014664 110467 000132      MOVB     R4,SOMODE ;:SAVE IT FOR USE
3228 014670 116704 000125      MOVB     SOFILL,R4 ;:GET THE ZERO FILL SWITCH
3229 014674 016605 000012      MOV      12(SP),R5 ;:PICKUP THE INPUT NUMBER
3230 014700 005003      CLR      R3      ;:CLEAR THE OUTPUT WORD
3231 014702 006105      1$:    ROL      R5      ;:ROTATE MSB INTO "C"
3232 014704 000404      BR      3$      ;:GO DO MSB
3233 014706 006105      2$:    ROL      R5      ;:FORM THIS DIGIT
3234 014710 006105      ROL      R5
3235 014712 006105      ROL      R5
3236 014714 010503      MOV      R5,R3
3237 014716 006103      3$:    ROL      R3      ;:GET LSB OF THIS DIGIT
3238 014720 105367 000076      DECB     SOMODE ;:TYPE THIS DIGIT?
3239 014724 100016      BPL     7$      ;:BR IF NO
3240 014726 042703 177770      BIC     #177770,R3 ;:GET RID OF JUNK
3241 014732 001002      BNE     4$      ;:TEST FOR 0
3242 014734 005704      TST     R4      ;:SUPPRESS THIS 0?
3243 014736 001403      BEQ     5$      ;:BR IF YES
3244 014740 005204      4$:    INC      R4      ;:DON'T SUPPRESS ANYMORE 0'S
3245 014742 052703 000060      BIS     #'0,R3 ;:MAKE THIS DIGIT ASCII
3246 014746 052703 000040      5$:    BIS     #' ,R3 ;:MAKE ASCII IF NOT ALREADY

```


M05

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 66
 DZDUUA.M11 13-OCT-76 08:28 BINARY TO OCTAL (ASCII) AND TYPE

3247	014752	110367	000040		MOV B	R3, B5		; SAVE FOR TYPING
3248	014756	104401	015016		TYPE	B5		; GO TYPE THIS DIGIT
3249	014762	105367	000032	75:	DECB	\$OCNT		; COUNT BY 1
3250	014766	003347			BGT	25		; BR IF MORE TO DO
3251	014770	002402			BLT	65		; BR IF DONE
3252	014772	005204			INC	R4		; INSURE LAST DIGIT ISN'T A BLANK
3253	014774	000744			BR	25		; GO DO THE LAST DIGIT
3254	014776	012605		65:	MOV	(SP)+, R5		; RESTORE R5
3255	015000	012604			MOV	(SP)+, R4		; RESTORE R4
3256	015002	012603			MOV	(SP)+, R3		; RESTORE R3
3257	015004	016666	000002 000004		MOV	2(SP), 4(SP)		; SET THE STACK FOR RETURNING
3258	015012	012616			MOV	(SP)+, (SP)		
3259	015014	000002			RTI			; RETURN
3260	015016	000		85:	.BYTE	0		; STORAGE FOR ASCII DIGIT
3261	015017	000			.BYTE	0		; TERMINATOR FOR TYPE ROUTINE
3262	015020	000		\$OCNT:	.BYTE	0		; OCTAL DIGIT COUNTER
3263	015021	000		\$DFILL:	.BYTE	0		; ZERO FILL SWITCH
3264	015022	000000		\$OMODE:	.WORD	0		; NUMBER OF DIGITS TO TYPE
3265								; ENTER HERE ON POWER FAILURE
3266								
3267								
3268	015024			\$PWRDN:				
3269	015024	010046		.PFAIL:	MOV	R0, -(SP)		; SAVE R0-R5 ON PROCESSOR STACK
3270	015026	010146			MOV	R1, -(SP)		
3271	015030	010246			MOV	R2, -(SP)		
3272	015032	010346			MOV	R3, -(SP)		
3273	015034	010446			MOV	R4, -(SP)		
3274	015036	010546			MOV	R5, -(SP)		
3275	015040	016746	162760		MOV	24, -(SP)		
3276	015044	010667	164046		MOV	SP, SAVSP		; SAVE STACK POINTER
3277	015050	012767	015062 162746		MOV	#RESTART, 24		; SET UP FOR POWER UP TRAP
3278	015056	000000			HALT			; HALT ON POWER DOWN NORMAL
3279	015060	000777			BR	.		
3280								
3. 81								; PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
3282								
3283	015062	016706	164030	RESTAR:	MJV	SAVSP, SP		; RESTORE STACK POINTER
3284	015066	012605			MOV	(SP)+, R5		; RESTORE R0-R5
3285	015070	012604			MOV	(SP)+, R4		
3286	015072	012603			MOV	(SP)+, R3		
3287	015074	012602			MOV	(SP)+, R2		
3288	015076	012601			MOV	(SP)+, R1		
3289	015100	012600			MOV	(SP)+, R0		
3290	015102	012767	015024 162714		MOV	#.PFAIL, 24		; SET UP FOR POWER FAILURE
3291	015110	106427	000340		MTPS	#340		
3292	015114	012706	001100		MOV	#STACK, SP		
3293	015120	005067	001102		CLR	TEMP		
3294	015124	005267	001076		INC	TEMP		
3295	015130	001375			BNE	.-4		
3296	015132	104413			CONVRT			
3297	015134	015156			PFTAB			
3298	015136	104401			TYPE			
3299	015140	015460			MPFAIL			
3300	015142	005067	164235		CLR	\$ERFLG		
3301	015146	005067	164244		CLR	\$ERRPC		
3302	015152	000177	163726		JMP	@RETURN		

N05

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 67
 DZDUUA.M11 13-OCT-76 08:28 BINARY TO OCTAL (ASCII) AND TYPE

3303	015156	000001			PFTAB: 1
3304	015160	006	002		.BYTE 6,2
3305	015162	000207			RETURN
3306	015164	005015	042012	053125	MTITLE: .ASCIZ <15><12><12>/DUV11 DZDUU-A TAPE E /<15><12>
3307	015172	030461	042040	042132	
3308	015200	052525	040455	052040	
3309	015206	050101	020105	020105	
3310	015214	005015	000		
3311	015217	015	053012	041505	MVECTO: .ASCIZ <15><12>/VEC ADD- /
3312	015224	040440	042104	000055	
3313	015232	005015	051461	020124	MREGAD: .ASCIZ <15><12>/1ST DEV: REC CSR ADD- /
3314	015240	042504	035126	051040	
3315	015246	041505	041440	051123	
3316	015254	040440	042104	000055	
3317	015262	005015	052515	052114	MMULT: .ASCIZ <15><12>/MULT DEV ? (Y OR N)- /
3318	015270	042040	053105	037440	
3319	015276	024040	020131	051117	
3320	015304	047040	026451	000	
3321	015311	015	046012	051501	MLASTD: .ASCIZ <15><12>/LAST DEV: REC CSR ADDR- /
3322	015316	020124	042504	035126	
3323	015324	051040	041505	041440	
3324	015332	051123	040440	042104	
3325	015340	026522	000		
3326	015343	075	042504	044526	DEVICE: .ASCIZ /=DEVICE /
3327	015350	042503	020040	000	
3328	015355	015	051412	046105	MCOM: .ASCIZ <15><12>/SELECT TO RUN DACTREG /
3329	015362	041505	020124	047524	
3330	015370	051040	047125	040040	
3331	015376	041501	051124	043505	
3332	015404	000			
3333	015405	015	047412	043126	MRANGE: .ASCIZ <15><12>/OVFLO:RETYPE LAST DEV RXCSR ADDS- /
3334	015412	047514	051072	052105	
3335	015420	050131	020105	040514	
3336	015426	052123	042040	053105	
3337	015434	051040	041530	051123	
3338	015442	040440	042104	026523	
3339	015450	000			
3340	015451	040	037440	000	MGM .ASCIZ / ? /
3341	015455	015	000012		MCR .ASCIZ <15><12>
3342	015460	043120	044501	026114	MPF .ASCIZ /PFAIL, RESTART AT TEST IN PROGRESS /
3343	015465	020040	042522	052123	
3344	015474	051101	020124	052101	
3345	015502	052040	051505	020124	
3346	015510	047111	050040	047522	
3347	015516	051107	051505	000123	
3348	015524	005015	047105	020104	MEPASS: .ASCIZ <15><12>/END OF PASS TAPE E /
3349	015532	043117	050040	051501	
3350	015540	020123	040524	042520	
3351	015546	042440	000		
3352	015551	015	051012	000	MR: .ASCIZ <15><12>/R /
3353	015555	015	052012	051505	MTSTPC: .ASCIZ <15><12>/TEST PC- /
3354	015562	020124	041520	000055	
3355	015570	005015	047514	045503	MLOCK: .ASCIZ <15><12>/LOCK ON TEST? (Y OR N)- /
3356	015576	047440	020116	052040	
3357	015604	051505	037524	024040	
3358	015612	020131	051117	047040	

```

3359 015620 026451 000
3360 015623 015 021412 047440 MSYNC: .ASCIZ <15><12>/# OF SYNC CHARS SELECTED ( 1 OR 2)-/
3361 015630 020106 054523 041516
3362 015636 041440 040510 051522
3363 015644 051440 046105 041505
3364 015652 042524 020104 020050
3365 015660 020061 051117 031040
3366 015666 026451 000
3367 015671 015 044412 020123 MWIRE6: .ASCIZ <15><12>/IS SEC XMIT SWITCH E55-2 IN? (Y OR N)-/
3368 015676 042523 020103 046530
3369 015704 052111 051440 044527
3370 015712 041524 020110 032505
3371 015720 026465 020062 047111
3372 015726 020077 054450 047440
3373 015734 020122 024516 000055
3374 015742 005015 051511 051440 MWIRES: .ASCIZ <15><12>/IS SEC REC SWITCH E55-3 IN? (Y OR N)-/
3375 015750 041505 051040 041505
3376 015756 051440 044527 041524
3377 015764 020110 032505 026465
3378 015772 020063 047111 020077
3379 016000 054450 047440 020122
3380 016006 024516 000055
3381 016012 005015 051511 047440 MWIRE4: .ASCIZ <15><12>/IS OPT CLR ENABLE SWITCH E55-1 IN? (Y OR N)-/
3382 016020 052120 041440 051114
3383 016026 042440 040516 046102
3384 016034 020105 053523 052111
3385 016042 044103 042440 032465
3386 016050 030455 044440 037516
3387 016056 024040 020131 051117
3388 016064 047040 026451 000
3389 016071 015 005012 031510 MEXTJ: .ASCIZ <15><12><12>/H315 CONNECTOR ON ?(Y OR N)-/
3390 016076 032461 041440 047117
3391 016104 042516 052103 051117
3392 016112 047440 020116 024077
3393 016120 020131 051117 047040
3394 016126 026451 000
3395 016131 015 020012 043536 MCNTG: .ASCIZ <15><12>/ +G /
3396 016136 020040 000
3397 016141 040 053523 036522 MMSWR: .ASCIZ / SWR= /
3398 016146 020040 000040
3399 016152 020040 047040 053505 MNNEW: .ASCIZ / NEW= /
3400 016160 020075 000040
3401 .EVEN
3402
3403 ;BUFFERS FOR INPUT-OUTPUT
3404
3405 016164 000000 INBUF: 0
3406 016226 .=. +40
3407 016226 000000 TEMP: 0
3408 016270 .=. +40
3409 016270 000000 MDATA: 0
3410 016332 .=. +40
3411 .SBTTL SCOPE HANDLER ROUTINE
3412
3413 ;*****
3414 ; THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT

```

```

3415 ; *AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG. (DISPLAY<7:0>)
3416 ; *AND LOAD THE ERROR FLAG (SERFLG) INTO DISPLAY<15:08>
3417 ; *THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3418 ; *SW14=1 LOOP ON TEST
3419 ; *SW11=1 INHIBIT ITERATIONS
3420 ; *SW09=1 LOOP ON ERROR
3421 ; *SW08=1 LOOP ON TEST IN SWR<7:0>
3422 ; *CALL
3423 ; * SCOPE ; ;SCOPE=IOT
3424
3425 016332 $SCOPE:
3426
3427 ;SCOPE LOOP AND INTERATION HANDLER
3428
3429 .SCOPE:
3430 JSR PC,CKSWR
3431 CLR SERPC ;CLEAR LAST ERROR PC
3432 CMP #TST1+2,(SP) ;IS SCOPE AT BEGINING OF TEST 1?
3433 BEQ $XTSTR ;YES NO LOOP.
3434
3435 016350 TTST: BR 1$ ;GO TO 1$ (IF LOCK SW02=1)
3436 016352 TSTB #STKS ;KEYBOARD DONE?
3437 016356 BPL $OVER ;BR IF NO
3438 016360 MOV #STKB,-2(SP) ;CLEAR DONE BIT
3439 016366 032777 040000 163044 1$: BIT #BIT14,#SWR ;LOOP ON PRESENT TEST?
3440 016374 001114 BNE $OVER ;YES IF SW14=1
3441 ;#####START OF CODE FOR THE XOR TESTER#####
3442 016376 000416 $XTSTR: BR 6$ ;IF RUNNING ON THE "XOR" TESTER CHANGE
3443 ;THIS INSTRUCTION TO A "NOP" (NOP=240)
3444 016400 013746 000004 MOV #ERRVEC,-(SP) ;SAVE THE CONTENTS OF THE ERROR VECTOR
3445 016404 012737 016424 000004 MOV #5$,#ERRVEC ;SET FOR TIMEOUT
3446 016412 005737 177060 TST #177060 ;TIME OUT ON XOR?
3447 016416 012637 000004 MOV (SP)+,#ERRVEC ;RESTORE THE ERROR VECTOR
3448 016422 000463 BR $SVLAD ;GO TO THE NEXT TEST
3449 016424 022626 5$: CMP (SP)+,(SP)+ ;CLEAR THE STACK AFTER A TIME OUT
3450 016426 012637 000004 MOV (SP)+,#ERRVEC ;RESTORE THE ERROR VECTOR
3451 016432 000423 BR 7$ ;LOOP ON THE PRESENT TEST
3452 016434 6$: ;#####END OF CODE FOR THE XOR TESTER#####
3453 016434 032777 000400 162776 BIT #BIT08,#SWR ;LOOP ON SPEC. TEST?
3454 016442 001404 BEQ 2$ ;BR IF NO
3455 016444 127767 162770 162730 CMPB #SWR,$TSTNM ;ON THE RIGHT TEST? SWR<7:0>
3456 016452 071465 BEQ $OVER ;BR IF YES
3457 016454 105767 162723 2$: TSTB SERFLG ;HAS AN ERROR OCCURRED?
3458 016460 001421 BEQ 3$ ;BR IF NO
3459 016462 126767 162727 162713 CMPB $ERMAX,SERFLG ;MAX. ERRORS FOR THIS TEST OCCURRED?
3460 016470 101015 BHI 3$ ;BR IF NO
3461 016472 032777 001000 162740 BIT #BIT09,#SWR ;LOOP ON ERROR?
3462 016500 001404 BEQ 4$ ;BR IF NO
3463 016502 016767 162702 162676 7$: MOV $LPERR,$LPADR ;SET LOOP ADDRESS TO LAST SCOPE
3464 016510 000446 BR $OVER
3465 016512 105067 162665 4$: CLRB SERFLG ;ZERO THE ERROR FLAG
3466 016516 005067 162770 CLR $TIMES ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
3467 016522 000415 BR 1$ ;ESCAPE TO THE NEXT TEST
3468 016524 032777 004000 162706 3$: BIT #BIT11,#SWR ;INHIBIT ITERATIONS?
3469 016532 001011 BNE 1$ ;BR IF YES
3470 016534 005767 162774 TST $PASS ;IF FIRST PASS OF PROGRAM
    
```

```

3471 016540 001406          BEQ     IS          ;; INHIBIT ITERATIONS
3472 016542 005267 162636    INC     $ICNT      ;; INCREMENT ITERATION COUNT
3473 016546 026767 162740 162630  CMP     $TIMES,$ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
3474 016554 002024          BGE     $OVER     ;; BR IF MORE ITERATION REQUIRED
3475 016556 012767 000001 162620  IS:    MOV     #1,$ICNT  ;; REINITIALIZE THE ITERATION COUNTER
3476 016564 016767 000056 162720  MOV     $MXCNT,$TIMES ;; SET NUMBER OF ITERATIONS TO DO
3477 016572 105267 162604          $SVLAD: INCB    $STNM   ;; COUNT TEST NUMBERS
3478 016576 116767 162600 162726  MOVB   $STNM,$STEN  ;; SET TEST NUMBER IN APT MAILBOX
3479 016604 011667 162576          MOV     (SP),$LPADR ;; SAVE SCOPE LOOP ADDRESS
3480 016610 011667 162574          MOV     (SP),$LPERR ;; SAVE ERROR LOOP ADDRESS
3481 016614 005067 162674          CLR     $ESCAPE   ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
3482 016620 112767 000001 162567  MOVB   #1,$ERMAX   ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3483 016626 016777 162550 162606  $OVER: MOV     $STNM,$DISPLAY ;; DISPLAY TEST NUMBER
3484 016634 016716 162546          MOV     $LPADR,(SP) ;; FUDGE RETURN ADDRESS
3485 016640 000002          45:    RTI
3486 016642 001407          BRW:   1407
3487 016644 000432          BRX:   432
3488 016646 000005          $MXCNT: 5          ;; MAX. NUMBER OF ITERATIONS
3489          .SBTTL TRAP DECODER
3490
3491          ;; *****
3492          ;; *THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3493          ;; *AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3494          ;; *OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3495          ;; *GO TO THAT ROUTINE.
3496
3497 016650 010046          $TRAP: MOV     RO,-(SP)  ;; SAVE RO
3498 016652 016600 000002    MOV     2(SP),RO    ;; GET TRAP ADDRESS
3499 016656 005740          TST     -(RO)       ;; BACKUP BY 2
3500 016660 111000          MOVB   (RO),RO     ;; GET RIGHT BYTE OF TRAP
3501 016662 006300          ASL    RO          ;; POSITION FOR INDEXING
3502 016664 016000 016704    MOV     $TRPAD(RO),RO ;; INDEX TO TABLE
3503 016670 000200          RTS     RO         ;; GO TO ROUTINE
3504
3505          ;; THIS IS USE TO HANDLE THE "GETPRI" MACRO
3506
3507
3508 016672 011646          $TRAP2: MOV    (SP),-(SP) ;; MOVE THE PC DOWN
3509 016674 016666 000004 000002  MOV    4(SP),2(SP)  ;; MOVE THE PSW DOWN
3510 016702 000002          RTI              ;; RESTORE THE PSW
3511
3512          .SBTTL TRAP TABLE
3513
3514          ;; *THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3515          ;; *BY THE "TRAP" INSTRUCTION.
3516
3517          ;
3518          ROUTINE
3519          -----
3520 016704 016672          $TRPAD: .WORD   $TRAP2          TRAP+1(104401) TTY TYPEOUT ROUTINE
3521 016706 013104          $TYPE   ;; CALL=TYPE          TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3522 016710 014622          $TYPOC  ;; CALL=TYPOC         TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
3523 016712 014576          $TYPOS  ;; CALL=TYPOS         TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
3524 016714 014636          $TYPON  ;; CALL=TYPON
3525
3526 016716 013062          .SCOPI ;; CALL=SCOPI        TRAP+5(104405)
    
```

3527 016720 013366
3528 016722 013474
3529 016724 013504
3530 016726 013704
3531 016730 013744
3532 016732 013776
3533 016734 014154

.INSTR :CALL=INSTR TRAP+6(104406)
.INSTER :CALL=INSTER TRAP+7(104407)
.PARAM :CALL=PARAM TRAP+10(104410)
.SAVOS :CALL=SAVOS TRAP+11(104411)
.RESOS :CALL=R_SOS TRAP+12(104412)
.CONVRT :CALL=CONVRT TRAP+13(104413)
.SETFLG :CALL=SETFLG TRAP+14(104414)

;UTILITIES

3539 016736 006367 000044
3540 016742 006367 000040
3541 016746 006367 000034
3542 016752 006367 000030
3543 016756 006367 000024
3544 016762 016767 000020 000020
3545 016770 162767 000001 000012
3546 016776 042767 000037 000004
3547 017004 000207
3548 017006 000240
3549 017010 000200

; THIS UTILITY CALCULATES PRIORITY LEVEL
DULEV: ASL DUPRT ;SHIFT LEFT
ASL DUPRT
ASL DUPRT
ASL DUPRT
ASL DUPRT
MOV DUPRT,LESS1 ;MOVE THIS TO LESS1
SUB #1,LESS1 ;CREATE LESS1
BIC #37,LESS1 ;CLEAR TNZVC
RTS PC
DUPRT: PR5
LESS1: PR4 ;LEVEL TO ALLOW INTERRUPTS

3551
3552 017012 016767 000126 162672
3553 017020 005267 000120
3554 017024 016767 000114 162662
3555 017032 005267 000106
3556 017036 016767 000102 162652
3557 017044 016767 000074 162650
3558 017052 005267 000066
3559 017056 016767 000062 162634
3560 017064 016767 000054 162632
3561 017072 005267 000046
3562 017076 016767 000042 162622
3563 017104 005267 000034
3564 017110 016767 000030 162612
3565 017116 005267 000022
3566 017122 016767 000016 162602
3567 017130 005267 000010
3568 017134 016767 000004 162572
3569 017142 000207
3570 017144 000000

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

3571
3572
3573
3574
3575 017146 042777 040000 162552
3576 017154 005067 162322
3577 017160 006067 162314
3578 017164 006067 162312
3579 017170 006267 162306
3580 017174 042767 100000 162300
3581 017202 056777 162274 162516
3582 017210 042777 020000 162510

; THIS UTILITY POKES THE MAINT DATA BASED UPON THE
; INFORMATION CONTAINED IN STMP1 AND IT IS
; SHIFTED IN BY THE CONTENTS OF SHIFT
RPOKE: BIC #MTDATA,HTXCSR
CLR STMP2
ROR STMP1 ;FORCE CARRY
ROR STMP2 ;PICK UP CARRY IN BIT 15
ASR STMP2 ;SHIFT INTO BIT 14
BIC #BIT15,STMP2 ;CLR BIT 15
BIS STMP2,HTXCSR ;POKE MAINT DATA
BIC #CLK,HTXCSR ;POKE CLK

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 72
 DZDUUA.M11 13-OCT-76 08:28 TRAP TABLE

```

3583 017216 052777 020000 162502      BIS      #CLK, TXCSR      ;
3584 017224 005367 161672      DEC      SHIFT
3585 017230 001346      BNE      RPOKE
3586 017232 000207      RTS      PC
3587
3588
3589 017234 016767 162240 162240 0008:  ; THIS ROUTINE CALCULATES ODD PARITY FOR AN 8 BIT CHAR
3590 017242 005067 162236      MOV      $TMP1, $TMP2 ; SAVE TMP1
3591 017246 012727 000010      CLR      $TMP3
3592 017252 000000      MOV      #8., (PC)+
3593 017254 006067 162222      4$:    0
3594 017260 005567 162220      1$:    ROR      $TMP2
3595 017264 005367 177762      ADC      $TMP3
3596 017270 001371      DEC      4$
3597 017272 006067 162206      BNE      1$
3598 017276 103404      ROR      $TMP3
3599 017300 052767 000400 162172      BCS      2$
3600 017306 000403      BIS      #BIT8, $TMP1 ; SET ODD PARITY
3601 017310 042767 000400 162162 2$:    BR      3$
3602      : $TMP1 NOW HAS ODD PARITY CHARACTER
3603 017316 000207      3$:    BIC      #BIT8, $TMP1 ; CLR EVEN PARITY
3604      : $TMP1 NOW HAS ODD PARITY CHARACTER
3605      RTS      PC
3606
3607      ; THIS ROUTINE CALCULATES EVEN PARITY FOR AN 8 BIT CHARACTER
3608 017320 016767 162154 162154  EVEN8:  MOV      $TMP1, $TMP2 ; SAVE TMP1
3609 017326 005067 162152      CLR      $TMP3
3610 017332 012727 000010      MOV      #8., (PC)+
3611 017336 000000      4$:    0
3612 017340 006067 162136      1$:    ROR      $TMP2
3613 017344 005567 162134      ADC      $TMP3
3614 017350 005367 177762      DEC      4$
3615 017354 001371      BNE      1$
3616 017356 006067 162122      ROR      $TMP3
3617 017362 103004      BCC      2$
3618 017364 052767 000400 162106      BIS      #BIT8, $TMP1 ; SET EVEN PARITY
3619 017372 000403      BR      3$
3620 017374 042767 000400 162076 2$:    BIC      #BIT8, $TMP1 ; CLR ODD PARITY
3621 017402 000207      : $TMP1 NOW HAS EVEN PARITY CHARACTER
3622 017404 062716 000002      3$:    RTS      PC
3623 017410 000002      TRPREG: ADD     #2, (SP) ; ALLOW IT TO "CRUNCH" INTO HLT BACK
3624      RTI      ; IN MAIN PART OF THE PROGRAM
      .END

```

AAA	003200	1292#							
ABASE =	000000	875	916						
ACDW1 =	000000	875	918						
ACDW2 =	000000	875	919						
ACPUOP=	000000	875	890						
ACTREG	001166	734#	1248#	1262*	1263*	1270*	2738	2741	2751
ADDW0 =	000000	875	920						
ADDW1 =	000000	875	921						
ADDW10=	000000	875	930						
ADDW11=	000000	875	931						
ADDW12=	000000	875	932						
ADDW13=	000000	875	933						
ADDW14=	000000	875	934						
ADDW15=	000000	875	935						
ADDW2 =	000000	875	922						
ADDW3 =	000000	875	923						
ADDW4 =	000000	875	924						
ADDW5 =	000000	875	925						
ADDW6 =	000000	875	926						
ADDW7 =	000000	875	927						
ADDW8 =	000000	875	928						
ADDW9 =	000000	875	929						
ADEVCT=	000000	875	881						
ADEVN =	000000	875	917						
ADRCNT=	013703	2953*	2992*	2999#					
RENV =	000000	875	886						
RENVN =	000000	875	887						
AFATAL=	000000	875	878						
AMADR1=	000000	875	903						
AMADR2=	000000	875	907						
AMADR3=	000000	875	910						
AMADR4=	000000	875	913						
AMAMS1=	000000	875	897						
AMAMS2=	000000	875	905						
AMAMS3=	000000	875	908						
AMAMS4=	000000	875	911						
AMSGAD=	000000	875	883						
AMSGLG=	000000	875	884						
AMSGTY=	000000	875	877						
AMTYP1=	000000	875	898						
AMTYP2=	000000	875	906						
AMTYP3=	000000	875	909						
AMTYP4=	000000	875	912						
APASS =	000000	875	880						
APRIOR=	000000	875							
APTC SU=	000040	535#	2868						
APTEMV=	000001	535#	2861	3115					
APTSIZ=	000200	535#	1167						
APTSP0=	000100	535#	2863						
ASWREG=	000000	875	888						
ATESTN=	000000	875	879						
AUNIT =	000000	875	882						
AUSWR =	000000	875	889						
AVECT1=	000000	875	914						
AVECT2=	000000	875	915						
BASEAD	001154	729#	1230#	1267*	1268	1274*	1276*	2745*	2757* 2761

MO6

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 80
 DZDUUA.M11 13-OCT-76 08:28 CROSS REFERENCE TABLE -- USER SYMBOLS

SM2 = 000004	634#													
SM3 = 000010	633#													
SM4 = 000020	632#													
SM5 = 000040	631#													
SM6 = 000100	630#													
SM7 = 000200	629#													
SM8 = 000400	628#													
SM9 = 001000	627#													
SYMCNO 001146	720#	1297*	1301*	1996	2367	2435	2508							
SYNEXT= 020000	788#	981#	2089	2096	2180	2188	2252	2260	2300	2308	2548	2556	2642	
	2650													
SYNINT= 030000	787#	980#	1459	1467	1512	1520	1672	1680	1725	1733	1778	1786	1831	
	1839	1884	1892	1937	1945	1999	2006	2056	2063	2132	2140	2347	2354	
	2411	2418												
SYNSCH= 000020	773#	966#	2007	2190	2356	2420	2473	2482	2557	2651				
SYSTST= 014000	813#	1006#												
TBITVE= 000014	669#													
TEMP 016226	3038	3293*	3294*	3407#										
TKVEC = 000060	676#													
TPVEC = 000064	677#													
TRAPVE= 000034	675#	1141*	1142*											
TRPREG 017404	3621#													
TRTVEC= 000014	670#													
TST1 003362	1332	1338	1351#	2794	2795	3432								
TST10 005234	1723#													
TST11 005442	1776#													
TST12 005650	1829#													
TST13 006056	1882#													
TST14 006264	1935#													
TST15 006472	1995#													
TST16 006760	2054#													
TST17 007342	2129#													
TST2 003570	1404#													
TST20 007470	2177#													
TST21 007756	2249#													
TST22 010150	2297#													
TST23 010350	2345#													
TST24 010610	2408#													
TST25 011400	2543#													
TST26 011742	2640#													
TST27 012164	2687#													
TST3 003776	1457#													
TST4 004204	1510#													
TST5 004412	1563#													
TST6 004620	1616#													
TST7 005026	1670#													
TTST 016350	3435#													
TXCSR 001726	1047#	1352*	1354*	1358*	1362	1367*	1368*	1374*	1375*	1376	1385*	1387	1405*	
	1407#	1411*	1415	1420*	1421*	1427*	1428*	1429	1438*	1440	1453*	1460*	1464*	
	1468	1473*	1474*	1480*	1481*	1482	1491*	1493	1511*	1513*	1517*	1521	1526*	
	1527*	1533*	1534*	1535	1544*	1546	1564*	1566*	1570*	1574	1579*	1580*	1586*	
	1587*	1588	1597*	1599	1617*	1619*	1623*	1627	1632*	1633*	1639*	1640*	1641	
	1650*	1652	1671*	1673*	1677*	1681	1686*	1687*	1693*	1694*	1695	1704*	1706	
	1724*	1726*	1730*	1734	1739*	1740*	1746*	1747*	1748	1757*	1759	1777*	1779*	
	1783*	1787	1792*	1793*	1799*	1800*	1801	1810*	1812	1830*	1832*	1836*	1840	
	1845*	1846*	1852*	1853*	1854	1863*	1865	1883*	1885*	1889*	1893	1898*	1899*	

G07

DZDUU-A MACY11 27(1006) 03-FEB-77 08:00 PAGE 88
DZDUUA.M11 13-OCT-76 08:28 CROSS REFERENCE TABLE -- MACRO NAMES

.SCMTA	5248	814
.SEOP	5248	
.SERRO	5248	3085
.SERRT	5248	3141
.SPOHE	5248	
.SSCOP	5248	3411
.STRAP	5248	3489
.STYPE	5248	2838
.STYPO	5248	3188

. ABS. 017412 000

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DZDUUA, DZDUUA, SEQ/SOL/CRF+DZDUU1/EQ:RUNE, DZDUU2, DZDUUA
RUN-TIME: 17 12 1 SECONDS
RUN-TIME RATIO: 213/31=6.7
CORE USED: 31K (62 PAGES)

EOF1DZDUUASEQ 00010000 770323 PDP10 411