

# DUV/LSI-11

DUV11 OFFLINE TRANS TEST  
MD-11-DZDUT-A

EP-DZDUT-A-DL-A  
COPYRIGHT © 1977  
FICHE 1 OF 1

APR 1977  
**digital**  
MADE IN USA

This microfiche card contains a grid of 48 frames of test data, arranged in 8 rows and 6 columns. Each frame displays a different set of test results, likely from a memory dump or diagnostic routine. The data is presented in a structured, tabular format with various columns of hexadecimal or binary values, and some frames include headers or labels. The frames are separated by dark borders, and the overall layout is typical of a microfiche card used for data storage and retrieval.

11



B01

EOF1DZDABBS80411  
DZDUT1.M11

02-FEB-77 08:17

00000000

MA2032327(1006) 03-PBB1074107:54 PAGE 0DR1DZDUTASEQ

00010000

770323

.REM \*

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

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

PRODUCT NAME: DUV11 OFFLINE TRANSMITTER 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 TRANSMITTER TESTS VERIFY THAT THE TRANSMITTER SECTION PROVIDES THE CORRECT ERROR FLAGS, AND THAT IT TRANSMITS CHARACTERS THRU THE BIT WINDOW AT THE CORRECT NUMBER OF BITS PER CHARACTER.

\* .REM \*

- 2. REQUIREMENTS

\* .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 DZDUT-A TAPE D" (ONCE ONLY)

\*  
 .REM \*  
 \*  
 .REM \*

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

4.3.2 PROGRAM RESTART WITH ALL SWITCHES DOWN

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

4.3.3 PROGRAM RESTART WITH SW00=1

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



4.3.3.2 TYPE 200G.

4.3.3.3 PROGRAM WILL START.

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

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

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

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

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

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

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

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

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

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

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

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

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

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



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

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

IF A DEVICE ADDRESS LOWER THAN 1ST DEVICE ADDRESS IS TYPED.....  
...SCHOOLS OUT.....THERE IS NO PROTECTION FOR THIS.  
THE PROGRAM WILL DEFAULT TO TWO DEVICES ACTIVE (UPWARDS FROM 1ST DEVICE ADDRESS).THE SAME APPLIES TO IDENTICAL ADDRESSES TYPED FOR FIRST AND LAST DEVICE.  
OBSERVE LOCATION 2 ACTREG: SEE SECTION 7.2

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

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

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

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

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

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

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

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

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

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

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



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

4.3.3.20 THE PROGRAM WILL TYPE "ARE YOU RUNNING IN MAINT.  
MODE EXTERNAL ? AND .....DO YOU HAVE THE EXTERNAL MODEM  
BYPASS JUMPER CONNECTOR ON ? (Y OR N)-" AND WAIT FOR AN  
INPUT FROM THE TELETYPE KEYBOARD

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

IF AN INCORRECT ANSWER IS TYPED ,THE PROGRAM WILL TYPE "?"  
AND WILL REPEAT THE MESSAGE OF 4.3.3.20

4.3.3.22 THE PROGRAM WILL TYPE "R" TO INDICATE THAT IT  
HAS STARTED AND WILL COMMENCE TESTING AT TEST 1

4.3.4 PROGRAM RESTART WITH SWD1=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 SWD0=1 AND ANSWER "NO" TO THE MULTIPLE DEVICE QUESTION  
SEE 4.3.3

4.3.4.1 SET SWD1=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 SWD2 =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 SWD2=1 IN SWITCH REG. (LOC. 176)

4.3.5.2 TYPE 200G.

4.3.5.3 PROGRAM WILL START.



# H01

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 7  
DZDUT1.M11 02-FEB-77 08:17

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
16XXXX	YYYYYY	ZZZZZZ

WHERE 16XXXX IS THE ADDRESS OF THE FAILING DEVICE REGISTER

WHERE YYYYYY IS THE EXPECTED CONTENTS OF THAT REGISTER



WHERE ZZZZZZ IS THE ACTUAL CONTENTS OF THAT REGISTER

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

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

WHERE YYYYYY IS THE EXPECTED DATA CONTENTS OF THAT REGISTER

WHERE ZZZZZZ IS THE ACTUAL DATA CONTENTS OF THAT REGISTER

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

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

WHERE YYYYYY IS THE EXPECTED CONTENTS OF THAT REGISTER

WHERE ZZZZZZ IS THE ACTUAL CONTENTS OF THAT REGISTER

6.1.5 ERROR DESCRIPTIONS  
SEE LISTINGS FOR DETAILS OF ERRORS

6.2 ERROR RECOVERY

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

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

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

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

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

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

END OF PASS TAPE Y



16XXXX = DEVICE

WHERE Y IS THE TAPE LOADED

WHERE 16XXXX IS THE DEVICE'S BASE REGISTER ADDRESS

TO INHIBIT THIS TYPEOUT - TURN TELETYPE OFF

## 7. RESTRICTIONS

### 7.1 MULTIPLE DEVICES

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

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

### 7.2 DISQUALIFYING DEVICES WHEN RUNNING MULTIPLE DEVICES

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

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

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

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

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

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

### 7.3 CABLE DELAYS

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

#### 7.3.1 TO PROVIDE SUFFICIENT DELAY FOR CLOCK SIGNAL OVER THE CABLE, LOCATION "HOLD:" MUST BE MODIFIED TO ACCOMODATE FOR FASTER MACHINES. PRESENTLY "HOLD:" =20 IS SUFFICIENT TIME ON AN 11/03 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 TRANSMITTER SECTION TESTING  
 OF THE DEVICE  
 SEE LISTING FOR DETAILS

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

11. LISTINGS

\*  
 .REM \*  
 \*  
 .REM \*  
 \*



L01

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 12  
DZDUT2.M11 02-FEB-77 08:20

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

000001

STN=1

# MO1

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 14  
 DZDUTA.M11 13-OCT-76 08:39 APT COMMUNICATIONS ROUTINE

```

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



613 020000  
 614 010000  
 615 004000  
 616 002000  
 617 001000  
 618 000400  
 619 000200  
 620 000100  
 621 000040  
 622 000020  
 623 000010  
 624 000004  
 625 000002  
 626 000001  
 627  
 628  
 629  
 630  
 631  
 632  
 633  
 634  
 635  
 636  
 637  
 638  
 639 100000  
 640 040000  
 641 020000  
 642 010000  
 643 004000  
 644 002000  
 645 001000  
 646 000400  
 647 000200  
 648 000100  
 649 000040  
 650 000020  
 651 000010  
 652 000004  
 653 000002  
 654 000001  
 655  
 656  
 657  
 658  
 659  
 660  
 661  
 662  
 663  
 664  
 665  
 666  
 667 000004  
 668 000010

SW13= 20000  
 SW12= 10000  
 SW11= 4000  
 SW10= 2000  
 SW09= 1000  
 SW08= 400  
 SW07= 200  
 SW06= 100  
 SW05= 40  
 SW04= 20  
 SW03= 10  
 SW02= 4  
 SW01= 2  
 SW00= 1  
 .EQUIV SW09,SW9  
 .EQUIV SW08,SW8  
 .EQUIV SW07,SW7  
 .EQUIV SW06,SW6  
 .EQUIV SW05,SW5  
 .EQUIV SW04,SW4  
 .EQUIV SW03,SW3  
 .EQUIV SW02,SW2  
 .EQUIV SW01,SW1  
 .EQUIV SW00,SW0

.;#DATA BIT DEFINITIONS (BIT00 TO BIT15)

BIT15= 100000  
 BIT14= 40000  
 BIT13= 20000  
 BIT12= 10000  
 BIT11= 4000  
 BIT10= 2000  
 BIT09= 1000  
 BIT08= 400  
 BIT07= 200  
 BIT06= 100  
 BIT05= 40  
 BIT04= 20  
 BIT03= 10  
 BIT02= 4  
 BIT01= 2  
 BIT00= 1  
 .EQUIV BIT09,BIT9  
 .EQUIV BIT08,BIT8  
 .EQUIV BIT07,BIT7  
 .EQUIV BIT06,BIT6  
 .EQUIV BIT05,BIT5  
 .EQUIV BIT04,BIT4  
 .EQUIV BIT03,BIT3  
 .EQUIV BIT02,BIT2  
 .EQUIV BIT01,BIT1  
 .EQUIV BIT00,BIT0

.;#BASIC "CPU" TRAP VECTOR ADDRESSES

ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS  
 RESVEC= 10 ;:RESERVED AND ILLEGAL INSTRUCTIONS

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 16  
DZDUTA.M11 13-OCT-76 08:39 BASIC DEFINITIONS

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



```

;STANDARD INTERRUPT VECTORS
679
680
681
682      000174      000174      .=174
683      000174      000000      DISPREG:0
684      000176      000000      SWREG:0
685      000200      000200      .=200
686      000200      000167      001746      JMP      .START      ;GO TO START OF PROGRAM
687
688
689
690      001100      001100      .=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
720 001146 377 SYNCNO: .BYTE 377 ;PROGRAM CONVERSATIONAL PARAMETERS
721 001147 377 SEXMIT: .BYTE 377 ;# OF SYNC CHARS REQ'D FOR SYNC'ZATION
722 001150 377 SEREC: .BYTE 377 ;SEC XMIT JUMPER "IN"
723 001151 377 OPTCLR: .BYTE 377 ;SEC REC JUMPER "IN"
724 001152 000 MULTD: .BYTE 0 ;OPTIONAL JUMPER CLR "IN"
725 001153 377 JMRBY: .BYTE 377 ;NO MULTIPLE DEVICE FLAG
726
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
746 001400
747
748
  
```



```

749
750
751
752           ; INSTRUCTION DEFINITIONS
753
754           005746   PUSH1SP=5746   ; DECREMENT PROCESSOR STACK 1 WORD =TST -(SP)
755           005726   POP1SP=5726   ; INCREMENT PROCESSOR STACK 1 WORD =TST (SP)+
756           010046   PUSHRO=10046  ; SAVE RO ON STACK =MOV RO, -(SP)
757           012600   POPRO=12600   ; RESTORE RO FROM STACK =MOV (SP)+, RO
758           024646   PUSH2SP=24646 ; DECREMENT STACK TWICE =CMP -(SP), -(SP)
759           022626   POP2SP=22626  ; INCREMENT STACK TWICE =CMP (SP)+, (SP)+
760           ; REGISTER DEFINITIONS
761           ; RXCSR BIT DEFINITIONS
762           100000   DSC=BIT15   ; DATA SET CHANGE
763           040000   RING=BIT14   ; RING
764           020000   CTS=BIT13   ; CLR TO SEND
765           010000   CARDET=BIT12  ; CARRIER DETECT
766           004000   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           ; 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           ; PARCSR BIT DEFINITIONS
784           001000   PAREN=BIT9    ; PARITY ENABLE
785           000400   EVPAR=BIT8    ; EVEN PARITY SENSE
786           ; PARCSR WRD DEFINITIONS
787           030000   SYNINT=30000  ; SYNC EXTERNAL MODE
788           020000   SYNEXT=20000  ; SYNC INTERNAL MODE
789           000000   ISYMOD=0      ; ISOC MODE
790           000000   FIVE=0       ; WORD LENGTH 5 BITS
791           002000   SIX=2000     ; WORD LENGTH 6 BITS
792           004000   SEVEN=4000   ; WORD LENGTH 7 BITS
793           006000   EIGHT=6000   ; WORD LENGTH 8 BITS
794           000000   NOPAR=0      ; NO PARITY
795           001000   ODDPAR=1000  ; ODD PARITY
796           001400   EVEPAR=1400  ; EVEN PARITY
797           ; TXCSR BIT DEFINITIONS
798           100000   DNA=BIT15     ; DATA NOT AVAILABLE
799           040000   MTDATA=BIT14  ; MAINT DATA
800           020000   CLK=BIT13    ; CLK
801           002000   BITW=BIT10   ; BIT WINDOW
802           000400   MRESET=BIT8   ; MASTER RESET
803           000200   TXDONE=BIT7  ; XMIT DONE
804           000100   TXINTE=BIT6  ; XMIT INTR ENABLE

```

F02

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 20  
DZDUTA.M11 13-OCT-76 08:39 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



```

814      .SBTTL COMMON TAGS
815
816      ;*****
817      ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
818      ;*USED IN THE PROGRAM.
819
820      001400
821      001400 000000
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
  
```

SCMTAG: . =.				;; START OF COMMON TAGS
STSTNM: .WORD	0			;; CONTAINS THE TEST NUMBER
SERFLG: .BYTE	00			;; CONTAINS ERROR FLAG
SICNT: .WORD	00			;; CONTAINS SUBTEST ITERATION COUNT
SLPADR: .WORD	00			;; CONTAINS SCOPE LOOP ADDRESS
SLPERR: .WORD	00			;; CONTAINS SCOPE RETURN FOR ERRORS
SERTTL: .WORD	00			;; CONTAINS TOTAL ERRORS DETECTED
SITEMB: .BYTE	00			;; CONTAINS ITEM CONTROL BYTE
SERMAX: .BYTE	1			;; CONTAINS MAX. ERRORS PER TEST
SERRPC: .WORD	00			;; CONTAINS PC OF LAST ERROR INSTRUCTION
SGDADR: .WORD	00			;; CONTAINS ADDRESS OF 'GOOD' DATA
SBDADR: .WORD	00			;; CONTAINS ADDRESS OF 'BAD' DATA
SGDDAT: .WORD	00			;; CONTAINS 'GOOD' DATA
SBDAT: .WORD	00			;; CONTAINS 'BAD' DATA
	00			;; RESERVED--NOT TO BE USED
	00			
SAUTOB: .BYTE	00			;; AUTOMATIC MODE INDICATOR
SINTAG: .BYTE	00			;; INTERRUPT MODE INDICATOR
	0			
SWR: .WORD	DSWR			;; ADDRESS OF SWITCH REGISTER
DISPLAY: .WORD	DDISP			;; ADDRESS OF DISPLAY REGISTER
STKS: 177560				;; TTY KBD STATUS
STKB: 177562				;; TTY KBD BUFFER
STPS: 177564				;; TTY PRINTER STATUS REG. ADDRESS
STPB: 177566				;; TTY PRINTER BUFFER REG. ADDRESS
SNULL: .BYTE	0			;; CONTAINS NULL CHARACTER FOR FILLS
SFILLS: .BYTE	2			;; CONTAINS # OF FILLER CHARACTERS REQUIRED
SFILLC: .BYTE	12			;; INSERT FILL CHARS. AFTER A "LINE FEED"
STPFLG: .BYTE	0			;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
SREGAD: .WORD	0			;; CONTAINS THE ADDRESS FROM WHICH (\$REGO) WAS OBTAINED
SREGO: .WORD	0			;; CONTAINS ((\$REGAD)+0)
SREG1: .WORD	00			;; CONTAINS ((\$REGAD)+2)
SREG2: .WORD	00			;; CONTAINS ((\$REGAD)+4)
SREG3: .WORD	00			;; CONTAINS ((\$REGAD)+6)
SREG4: .WORD	00			;; CONTAINS ((\$REGAD)+10)
SREG5: .WORD	00			;; CONTAINS ((\$REGAD)+12)
STMP0: .WORD	00			;; USER DEFINED
STMP1: .WORD	00			;; USER DEFINED
STMP2: .WORD	00			;; USER DEFINED
STMP3: .WORD	00			;; USER DEFINED
STMP4: .WORD	00			;; USER DEFINED
STMPS: .WORD	0			;; USER DEFINED
STIMES: 0				;; MAX. NUMBER OF ITERATIONS
SESCAPE: 0				;; ESCAPE ON ERROR ADDRESS
SBELL: .ASCIZ	<207><377><377>			;; CODE FOR BELL
SQUES: .ASCII	!/?			;; QUESTION MARK
SCRLF: .ASCII	<15>			;; CARRIAGE RETURN

H02

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 22  
DZDUTA.M11 13-OCT-76 08:39 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

SLF: .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 AMSGLG ; 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  
\$SVECT1: .WORD AVECT1 ; INTERRUPT VECTOR#1, BUS PRIORITY#1  
\$SVECT2: .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  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997

005746  
005726  
010046  
012600  
024646  
022626  
  
100000  
040000  
020000  
010000  
004000  
002000  
001000  
000400  
000200  
000100  
000040  
000020  
000010  
000004  
000002  
000001  
  
100000  
040000  
020000  
010000  
  
001000  
000400  
  
030000  
020000  
000000  
000000  
002000  
004000  
006000  
000000  
001000  
001400  
  
100000  
040000  
020000  
002000  
000400  
000200  
000100

; INSTRUCTION DEFINITIONS

```

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

; REGISTER DEFINITIONS

; RXCSR BIT DEFINITIONS

```

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

; RXDBUF BIT DEFINITIONS

```

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

; PARCSR BIT DEFINITIONS

```

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

; PARCSR WRD DEFINITIONS

```

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

; TXCSR BIT DEFINITIONS

```

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



K02

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 25  
DZDUTA.M11 13-OCT-76 08:39 APT MAILBOX-ETABLE

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

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

001652  
  
001652 001762  
001654 002067  
001656 002116  
001660 002132  
001662 002022  
001664 002067  
001666 002116  
001670 002132  
001672 002043  
001674 002067  
001676 002116  
001700 002132  
001702 001746  
001704 000000  
001706 002126  
001710 002132  
  
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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.  
;\*NOTE1: IF SITEMB 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/



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,HL01,0
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			;;SSVPC=.
1092		000046			;;SAVE PC
1093	000046	012656			;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
1094		000052			;;2)SET LOC.52 TO ZERO
1095	000052	000000			;;RESTORE PC
1096		002136			;;
1097					.SBTTL APT PARAMETER BLOCK
1098					
1099					::*****
1100					::SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1101					::*****
1102		002136			;;SAVE CURRENT LOCATION
1103		000024			;;SET POWER FAIL TO POINT TO START OF PROGRAM
1104	000024	000200			;;FOR APT START UP
1105		000044			;;POINT TO APT INDIRECT ADDRESS PNTR.
1106	000044	002136			;;POINT TO APT HEADER BLOCK
1107		002136			;;RESET LOCATION COUNTER
1108					::*****
1109					::SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1110					::INTERFACE SPEC.
1111					
1112	002136				\$APTHD: .WORD 0
1113	002136	000000			;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1114	002140	001526			\$MADR: .WORD \$MAIL
1115	002142	000010			;;ADDRESS OF APT MAILBOX (BITS 0-15)
1116	002144	000010			\$STMT: .WORD 10
1117	002146	000000			;;RUN TIME OF LONGEST TEST
1118	002150	000052			\$PASTM: .WORD 10
					;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
					\$UNITM: .WORD
					;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
					.WORD SETEND-\$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 ($CMTAG) AREA
1131 002152   012706 001400   MOV     #SCMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1132 002156   005026          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1133 002160   022706 001440   CMP     #SWR,R6      ;;DONE?
1134 002164   001374          BNE     -6             ;;LOOP BACK IF NO
1135 002166   012706 001100   MOV     #STACK,SP    ;;SETUP THE STACK POINTER
1136          ;;INITIALIZE A FEW VECTORS
1137 002172   012737 016306 000020   MOV     #SCOPE,%IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1138 002200   012737 000340 000022   MOV     #340,%IOTVEC+2 ;;LEVEL 7
1139 002206   012737 014176 000030   MOV     #ERROR,%EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1140 002214   012737 000340 000032   MOV     #340,%EMTVEC+2 ;;LEVEL 7
1141 002222   012737 016624 000034   MOV     #TRAP,%TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1142 002230   012737 000340 000036   MOV     #340,%TRAPVEC+2 ;;LEVEL 7
1143 002236   012737 015000 000024   MOV     #PWRON,%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     #,$SLPADR     ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1149 002276   012767 002276 177104   MOV     #,$SLPERR     ;;SETUP THE ERROR LOOP ADDRESS
1150          ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1151          ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1152 002304   013746 000004          MOV     %ERRVEC,-(SP) ;;SAVE ERROR VECTOR
1153 002310   012737 002344 000004   MOV     #64,$ERRVEC  ;;SET UP ERROR VECTOR
1154 002316   012767 177570 177114   MOV     #DSWR,$SWR    ;;SETUP FOR A HARDWARE SWICH REGISTER
1155 002324   012767 177570 177110   MOV     #DDISP,$DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
1156 002332   022777 177777 177100   CMP     #-1,$SWR     ;;TRY TO REFERENCE HARDWARE SWR
1157 002340   001012          BNE     66$          ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1158          ;;AND THE HARDWARE SWR IS NOT = -1
1159 002342   000403          BR     65$          ;;BRANCH IF NO TIMEOUT
1160 002344   012716 002352 64$:   MOV     #65$,(SP)   ;;SET UP FOR TRAP RETURN
1161 002350   000002          RTI
1162 002352   012767 000176 177060 65$:   MOV     #SWREG,$SWR  ;;POINT TO SOFTWARE SWR
1163 002360   012767 000174 177054   MOV     #DISPREG,$DISPLAY
1164 002366   012637 000004 66$:   MOV     (SP)+,%ERRVEC ;;RESTORE ERROR VECTOR
1165
1166 002372   005067 177136          CLR     $PASS        ;;CLEAR PASS COUNT
1167 002376   132767 000200 177143   BITB   #APTSIZE,$ENVH ;;TEST USER SIZE UNDER APT
1168 002404   001403          BEQ    67$          ;;YES,USE NON-APT SWITCH
1169 002406   012767 001550 177024   MOV     #SSWREG,$SWR ;;NO,USE APT SWITCH REGISTER
1170 002414          67$:
1171 002414   012706 001100          MOV     #STACK,SP   ;;SET STACK
1172 002420   106427 000340          MTPS   #340         ;;LOCK INTERRUPTS
1173 002424   012737 015000 000024   MOV     #PFAIL,%24   ;;SET UP POWER FAIL VECTOR
1174 002432   105067 176535          CLRB   $STFLG       ;;CLEAR START FLAG
    
```



DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 29  
 DZDUTA.M11 13-OCT-76 08:39 INITIALIZE THE COMMON TAGS

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, \$STSTM		; SET UP FOR TEST 1
1180	002464	012767	002152	176412	MOV	#.START, RETURN		; SET UP FOR POWER FAIL BEFORE
1181								; TESTING STARTS
1182	002472	013746	000006		MOV	#6, -(SP)		
1183	002476	013746	000004		MOV	#4, -(SP)		
1184	002502	012737	002516	000004	MOV	#15, #4		
1185	002510	005777	176724		TST	\$SWR		
1186	002514	000407			BR	\$S		
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	\$S		
1194	002554	005737	000042		TST	#42		; CHECK FOR CHAIN
1195	002560	001402			BEQ	\$S		
1196	002562	000167	000522		JMP	.BEGIN		
1197	002566	004767	010172		33:	JSR	PC, CNTLU	
1198	002572	105767	176374		35:	TSTB	INIFLG	; HAS INITIALIZATION BEEN PERFORMED
1199	002576	001004			BNE	ONCE		
1200	002600	104401	015140		TYPE	MTITLE		; TYPE TITLE MESSAGE
1201	002604	105167	176362		COMB	INIFLG		; IF NOT SET FLAG AND DO
1202	002610	105767	176732		ONCE:	TSTB	SENV	; APT CONTROL?
1203	002614	001410			BEQ	\$S		; BR IF NO
1204	002616	032767	000001	176726	BIT	#1, SUSWR		; EXTENAL JUMPER ON?
1205	002624	001002			BNE	\$S		; 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	\$S		
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	\$S		
1220	002704	104406			INSTR			; OUTPUT MESSAGE & GET INPUT STRING
1221	002706	015206			MREGAD			; MESSAGE
1222	002710	104410			PARAM			; CONVERT STRING
1223	002712	160000			160000			; LOW LIMIT
1224	002714	167776			167776			; HIGH LIMIT
1225	002716	017120			DUBASE			; STORE AT THIS LOCATION
1226	002720	001			.BYTE	1		; MASK
1227	002721	001			.BYTE	1		; HOW MANY TIMES + 2
1228	002722	016767	014172	176226	MOV	DUBASE, KEEPADD		; SAVE
1229	002730	004767	014032		JSR	PC, DUADR		
1230	002734	016767	176216	176212	MOV	KEEPADD, BASEADD		; RESTORE FOR ROTATION



1231	002742	104406					INSTR	: OUTPUT MESSAGE & GET INPUT STRING
1232	002744	015173					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,KEEPV : SAVE
1240	002766	016767	176744	176166			MOV	DURIV,BASEV : SET UP FOR ROTATION
1241	002774	104406					INSTR	: OUTPUT MESSAGE & GET INPUT STRING
1242	002776	015236					MMULT	: MESSAGE
1243	003000	104414					SETFLG	: SET FLAG BASED UPON INPUT STRING
1244	003002	001152					MULTD	: THIS FLAG
1245	003004	105767	176142				TSTB	MULTD : ARE THERE MULTIPLE DEVICES
1246								: ON THE SYSTEM ?
1247	003010	100406					BMI	BBB : YES,ASK NEXT QUESTION
1248	003012	005067	176150				CLR	ACTREG
1249	003016	005067	176146				CLR	ROTADD
1250	003022	000167	000140				JMP	OUTMUL : JUMP AROUND NEXT QUESTION
1251	003026					BBB:		
1252	003026	104406					INSTR	: OUTPUT MESSAGE & GET INPUT STRING
1253	003030	015265					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	KEEPPAD,BASEADD : DITTO
1275	003134	000414					BR	OUTMUL : CONTINUE QUESTIONS
1276	003136	016767	176014	176010	3\$:		MOV	KEEPPAD,BASEADD : RESTORE
1277	003144	104406					INSTR	: OUTPUT MESSAGE & GET INPUT STRING
1278	003146	015361					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	013566		OUTMUL:	MOV	#340,DUPRT

```

1287 003174 004767 013512 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 015577 MSYNC ;MESSAGE
1295 003204 122767 000061 012726 3S: CMPB #'1,INBUF ;IS IT "1" ?
1296 003212 001003 BNE 1S
1297 003214 105067 175726 CLRB SYNCNO ;000
1298 003220 000412 BR 4S
1299 003222 122767 000062 012710 1S: CMPB #'2,INBUF ;IS IT "2" ?
1300 003230 001004 BNE 2S
1301 003232 112767 177777 175706 MOVB #-1,SYNCNO ;377
1302 003240 000402 BR 4S
1303 003242 104407 2S: INSTER ;RETRY
1304 003244 000757 BR 3S
1305 003246 000240 4S: NOP
1306 003250 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1307 003252 015645 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 015716 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 015766 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 016045 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,JSWR ;IF SW01=1, GET STARTING PC
1328 003326 001406 BEQ 3S
1329 003330 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1330 003332 015531 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 4S
1338 003344 012767 003362 175532 3S: MOV #TST1,RETURN ;START AT TEST 1
1339 003352 104401 015525 4S: TYPE MR ;TYPE R
1340 003356 000177 175522 JMP @RETURN ;START TESTING
1341
1342 ;;THIS TEST CHECKS THE STRIP SYNC FUNCTION
    
```



```

1343                                     ;: OF THE RECEIVER LOGIC
1344                                     ;: MODE: SYNINT
1345                                     ;: LENGTH: EIGHT
1346                                     ;: NOTE: RXDONE SHOULD NEVER ASSERT
1347                                     ;: CHAR: 26 (SYNC)
1348
1349                                     ;: *****
1350 003362 000004                               †ST1: SCOPE
1351 003364 052777 000400 176334             BIS      #MRESET,@TXCSR ; MASTER RESET
1352 003372 012777 030000 176322             MOV      #SYNINT,@PARCSR ; SET THE MODE
1353 003400 052777 000400 176320             BIS      #MRESET,@TXCSR ; MASTER RESET
1354
1355                                     ; SET MAINT DATA, CLK, BREAK, & MAINTENANCE MODE
1356 003406 012777 064001 176312             MOV      #MNTDATA!CLK!MINT!BREAK,@TXCSR
1357
1358                                     ; SET MODE # OF BITS, PARITY SENSE, & LOAD SYNC REG
1359 003414 012777 036026 176300             MOV      #SYNINT!EIGHT!NOPAR!26,@PARCSR
1360 003422 052777 000020 176262             BIS      #SYNSCH,@RXCSR ; SET SYNC SEARCH
1361                                     ; POKE CLK TO GET RECEIVER INTO SYNCRIZATION....
1362 003430 042777 020000 176270             BIC      #CLK,@TXCSR ; POKE CLK DOWN
1363 003436 052777 020000 176262             BIS      #CLK,@TXCSR ; POKE CLK UP
1364                                     ; POKE CLK TO GET LOGIC INTO SYNCRIZATION
1365 003444 042777 020000 176254             BIC      #CLK,@TXCSR ; POKE CLK DOWN
1366 003452 052777 020000 176246             BIS      #CLK,@TXCSR ; POKE CLK UP
1367 003460 052777 000400 176224             BIS      #STPSYN,@RXCSR ; SET STRIP SYNC
1368 003466 012767 000003 175430             MOV      #3,COUNT ; # OF SYNC CHARS
1369 003474 012767 000026 175776             1S: MOV   #26,$TMP1 ; CHAR TO BE SHIFTED
1370 003502 012767 000010 175412             MOV      #8,$SHIFT ; # OF SHIFTS
1371 003510 004767 013406                               JSR      PC,@POKE ; SHIFT IN THIS CHAR
1372 003514 105777 176172             TSTB    @RXCSR ; RXDONE ?
1373 003520 100001                               BPL      +4
1374 003522 104004                               ERROR   4 ; RXDONE SHOULD NOT BE ASSERTED
1375 003524 005367 175374             DEC      COUNT ; # OF SYNC CHARS
1376 003530 001361                               BNE     1S
1377
1378                                     ;: THIS TEST PROVES THAT RXERR FREEZES THE "RECEIVER RESET"
1379                                     ;: WHILE IN STRIP SYNC MODE
1380                                     ;: THIS TEST FIRST PROVES THAT AUTOMATIC RESETS OCCUR WHEN
1381                                     ;: STRIP SYNC IS SET & SYNC CHARACTERS ARE SENT
1382                                     ;: BUT IF AN ERROR SHOULD OCCUR....THIS AUTOMATIC RESET
1383                                     ;: IS DISCOMBOBULATED
1384                                     ;: IE, FORCE PARITY ERROR WHILE STRIP SYNC IS SET
1385                                     ;: NOTE: NORMALLY THE LOGIC RESETS THE RXDONE & ERROR FLAGS
1386                                     ;: PROVIDING THAT ONLY GOOD SYNC CHARACTERS ARE SENT....
1387                                     ;: BUT, IF AN RXERR OCCURS RXDONE PLUS RXERR ARE ASSERTED
1388                                     ;: MODE: ISOC (ISYMOD)
1389                                     ;: LENGTH: EIGHT
1390                                     ;: PARITY: EVEPAR
1391                                     ;: CHARACTER EXPECTED: 26
1392                                     ;: CHARACTER SENT: SYNC CHARACTER
1393                                     ;: NOTE: THIS TEST USES ONLY THE RECEIVER LOGIC
1394
1395                                     ;: *****
1396 003532 000004                               †ST2: SCOPE
1397 003534 052777 000400 176164             BIS      #MRESET,@TXCSR ; MASTER RESET
1398 003542 012777 000000 176152             MOV      #ISYMOD,@PARCSR ; SET THE MODE
    
```



# F03

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 33  
 DZDUTA.M11 13-OCT-76 08:39 INITIALIZE THE COMMON TAGS

```

1399 003550 052777 000400 176150      BIS      #MRESET,@TXCSR ;MASTER RESET
1400
1401                                     ;SET MAINT DATA,CLK BREAK,&MAINTENANCE MODE
1402 003556 012777 064001 176142      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
1403
1404                                     ;SET MODE , # OF BITS,PARITY SENSE,&LOAD SYNC REG
1405 003564 012777 007426 176130      MOV      #ISYMOD!EIGHT!EVEPAR!26,@PARCSR
1406 003572 016703 176120      MOV      @RXDBUF,R3 ;SET UP FOR ERROR MSG
1407 003576 012767 000003 175320      MOV      #3,COUNT ;# OF TIMES SYNC CHAR WILL BE SENT
1408 003604 052777 000020 176100      BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
1409                                     ;POKE CLK TO GET RECEIVER INTO SYNCRIZATION....
1410 003612 042777 020000 176106      BIC      @CLK,@TXCSR ;POKE CLK DOWN
1411 003620 052777 020000 176100      BIS      @CLK,@TXCSR ;POKE CLK UP
1412                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1413 003626 042777 020000 176072      BIC      @CLK,@TXCSR ;POKE CLK DOWN
1414 003634 052777 020000 176064      BIS      @CLK,@TXCSR ;POKE CLK UP
1415 003642 052777 000400 176042      BIS      #STPSYN,@RXCSR ;SET STRIP SYNC
1416 003650 012767 000013 175244      2$:     MOV      #11,SHIFT ;# OF SHIFTS
1417 003656 012767 003054 175614      MOV      #3054,STMP1 ;SYNC CHAR + START&STOP+ PARITY
1418 003664 004767 013232      1$:     JSR      PC,RPOKE ;SHIFT IN THIS CHARACTER
1419 003670 105777 176016      TSTB    @RXCSR ;RXDONE = 0 ?
1420 003674 100001      BPL     .+4
1421 003676 104004      ERROR   4 ;RXDONE SHOULD NOT BE SET
1422 003700 005367 175220      DEC     COUNT ;# OF SYNC CHARS
1423 003704 001361      BNE     2$ ;GO AGAIN ?
1424 003706 012700 000026      MOV     #26,R0 ;EXPECTED
1425 003712 017701 176000      MOV     @RXDBUF,R1 ;ACTUAL
1426                                     ;NOTE THAT THIS IS THE FIRST TIME
1427                                     ;RXDBUF IS READ.....THERE SHOULD BE
1428                                     ;NO OVER RUN ERROR 4$
1429 003716 020001      CMP     R0,R1 ;COMPARE EXPECTED VS ACTUAL
1430 003720 001401      BEQ    .+4
1431 003722 104002      ERROR   2 ;DATA CHARS SHOULD COMPARE
1432                                     ;THERE SHOULD BE NO RXERR'S
1433 003724 012767 000004 175172      MOV     #4,COUNT ;# OF TIMES
1434 003732 012700 110026      MOV     @RXERR!PARER!26,R0 ;EXPECTED
1435 003736 012767 002054 175534      MOV     #2054,STMP1 ;BAD SYNC CHAR (WRONG PARITY)
1436 003744 012767 000013 175150      3$:     MOV     #11,SHIFT ;# OF SHIFTS
1437 003752 004767 013144      JSR     PC,RPOKE ;SHIFT IN THIS CHAR
1438 003756 105777 175730      TSTB    @RXCSR ;RXDONE = 1?
1439 003762 100401      BMI    .+4
1440 003764 104004      ERROR   4 ;RXDONE SHOULD BE SET
1441 003766 017701 175724      MOV     @RXDBUF,R1 ;ACTUAL DATA
1442 003772 020001      CMP     R0,R1 ;COMPARE EXP VS ACT
1443 003774 001401      BEQ    .+4
1444 003776 104000      ERROR   ;DID THE RESPECTIVE ERROR 4 STOP THE
1445                                     ;AUTOMATIC RESSETTING OF RXDONE & ERROR FLAGS
1446                                     ;CHECK THIS.....
1447 004000 005367 175120      DEC     COUNT ;# OF SYNC CHARS
1448 004004 001445      BEQ    5$ ;FINISHED ? GET OUT OF TEST
1449 004006 022767 000003 175110      CMP     #3,COUNT ;# OF SYNC CHARS
1450 004014 001423      BEQ    6$ ;CHECK FRAME ERROR ?
1451 004016 022767 000002 175100      CMP     #2,COUNT ;# OF SYNC CHARS
1452 004024 001426      BEQ    7$ ;CHECK FRAME ERROR & BAD PARITY ?
1453                                     ;NOPE THEN IT (COUNT) MUST BE = 1 THEREFORE....
1454 004026 012767 000013 175066      MOV     #11.,SHIFT ;# OF SHIFTS
  
```

```

1455 004034 012767 000054 175436      MOV      #54,STMP1      ;FRAME & PARITY ERROR
1456 004042 004767 013054      JSR      PC,RPOKE      ;SHIFT IN THIS CHAR
1457                                     ;NOW DON'T READ THE RXDBUF TO CREATE OVER RUN
1458 004046 012767 000054 175424      MOV      #54,STMP1      ;FRAME & PARITY ERROR
1459 004054 012700 170026      MOV      @RXERR!OVRRUN!FRMERR!PARER!26,RO      ;EXPECTED
1460 004060 000167 177660      JMP      3$            ;DO IT AGAIN
1461 004064 012767 001054 175406 6$:      MOV      #1054,STMP1    ;BAD STOP BIT FOR FRAME ERROR
1462 004072 012700 120026      MOV      @RXERR!FRMERR!26,RO      ;EXPECTED
1463 004076 000167 177642      JMP      3$            ;DO IT AGAIN
1464 004102 012767 000054 175370 7$:      MOV      #54,STMP1      ;BAD STOP BIT & PARITY
1465 004110 012700 130026      MOV      @RXERR!FRMERR!PARER!26,RO      ;EXPECTED
1466 004114 000167 177624      JMP      3$            ;DO IT AGAIN
1467 004120
1468                                     5$:
1469                                     ;; THIS TEST PROVES THAT RXERR FREEZES THE "RECEIVER RESET"
1470                                     ;; WHILE IN STRIP SYNC MODE
1471                                     ;; THIS TEST FIRST PROVES THAT AUTOMATIC RESETS OCCUR WHEN
1472                                     ;; STRIP SYNC IS SET & SYNC CHARACTERS ARE SENT
1473                                     ;; BUT IF AN ERROR SHOULD OCCUR....THIS AUTOMATIC RESET
1474                                     ;; IS DISCOMBOBULATED
1475                                     ;; IE. FORCE PARITY ERROR WHILE STRIP SYNC IS SET
1476                                     ;; NOTE: NORMALLY THE LOGIC RESETS THE RXDONE & ERROR FLAGS
1477                                     ;; PROVIDING THAT ONLY GOOD SYNC CHARACTERS ARE SENT....
1478                                     ;; BUT, IF AN RXERR OCCURS RXDONE PLUS RXERR ARE ASSERTED
1479                                     ;; MODE: ISOC (ISYMOD)
1480                                     ;; LENGTH: SEVEN
1481                                     ;; PARITY: EVEPAR
1482                                     ;; CHARACTER EXPECTED: 226
1483                                     ;; NOTE THAT THE PARITY BIT SHOULD SHOW
1484                                     ;; UP IN THE DATA IE. BIT SEVEN FOR
1485                                     ;; SEVEN LEVEL CODE
1486                                     ;; CHARACTER SENT: SYNC CHARACTER
1487                                     ;; NOTE: THIS TEST USES ONLY THE RECEIVER LOGIC
1488                                     ;; *****
1489 004120 000004      TST3:   SCOPE
1490 004122 052777 000400 175576      BIS      @MRESET,@TXCSR      ;MASTER RESET
1491 004130 012777 000000 175564      MOV      @ISYMOD,@PARCSR     ;SET THE MODE
1492 004136 052777 000400 175562      BIS      @MRESET,@TXCSR      ;MASTER RESET
1493
1494                                     ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
1495 004144 012777 064001 175554      MOV      @MTDATA!CLK!MINT!BREAK,@TXCSR
1496
1497                                     ;SET MODE , # OF BITS,PARITY SENSE &LOAD SYNC REG
1498 004152 012777 005626 175542      MOV      @ISYMOD!SEVEN!EVEPAR!226,@PARCSR
1499 004160 016703 175532      MOV      RXDBUF,R3          ;SET UP FOR ERROR MSG
1500 004164 012767 000003 174732      MOV      #3,COUNT          ;# OF TIMES SYNC CHAR WILL BE SENT
1501 004172 052777 000020 175512      BIS      @SYNSCH,@RXCSR     ;SET SYNC SEARCH
1502                                     ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
1503 004200 042777 020000 175520      BIC      @CLK,@TXCSR        ;POKE CLK DOWN
1504 004206 052777 020000 175512      BIS      @CLK,@TXCSR        ;POKE CLK UP
1505                                     ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1506 004214 042777 020000 175504      BIC      @CLK,@TXCSR        ;POKE CLK DOWN
1507 004222 052777 020000 175476      BIS      @CLK,@TXCSR        ;POKE CLK UP
1508 004230 052777 000400 175454      BIS      @STPSYN,@RXCSR     ;SET STRIP SYNC
1509 004236 012767 000012 174656 2$:      MOV      #10,SHIFT         ;# OF SHIFTS
1510 004244 012767 001454 175226      MOV      #1454,STMP1       ;SYNC CHAR + START&STOP+ PARITY

```



```

1511 004252 004767 012644      1S:   JSR   PC,RPOKE      ;SHIFT IN THIS CHARACTER
1512 004256 105777 175430      TSTB  @RXCSR      ;RXDONE = 0 ?
1513 004262 100001          BPL   .+4
1514 004264 104004          ERROR 4           ;RXDONE SHOULD NOT BE SET
1515 004266 005367 174632      DEC   COUNT      ;# OF SYNC CHARS
1516 004272 001361          BNE   2$         ;GO AGAIN ?
1517 004274 012700 000226      MOV   @226,RO    ;EXPECTED
1518 004300 017701 175412      MOV   @RXDBUF,R1 ;ACTUAL
1519                                ;NOTE THAT THIS IS THE FIRST TIME
1520                                ;RXDBUF IS READ.....THERE SHOULD BE
1521                                ;NO OVER RUN ERROR      4$
1522 004304 020001          CMP   RO,R1      ;COMPARE EXPECTED VS ACTUAL
1523 004306 001401          BEQ   .+4
1524 004310 104002          ERROR 2           ;DATA CHARS SHOULD COMPARE
1525                                ;THERE SHOULD BE NO RXERR'S
1526 004312 012767 000004 174604      MOV   #4,COUNT   ;# OF TIMES
1527 004320 012700 110026      MOV   @RXERR!PARER!26,RO ;EXPECTED
1528 004324 012767 001054 175146      MOV   #1054,$TMP1 ;BAD SYNC CHAR (WRONG PARITY)
1529 004332 012767 000012 174562 3S:   MOV   #10,$SHIFT ;# OF SHIFTS
1530 004340 004767 012556      JSR   PC,RPOKE   ;SHIFT IN THIS CHAR
1531 004344 105777 175342      TSTB  @RXCSR      ;RXDONE = 1?
1532 004350 100401          BMI   .+4
1533 004352 104004          ERROR 4           ;RXDONE SHOULD BE SET
1534 004354 017701 175336      MOV   @RXDBUF,R1 ;ACTUAL DATA
1535 004360 020001          CMP   RO,R1      ;COMPARE EXP VS ACT
1536 004362 001401          BEQ   .+4
1537 004364 104000          ERROR 4           ;DID THE RESPECTIVE ERROR      4 STOP THE
1538                                ;AUTOMATIC RESSETTING OF RXDONE & ERROR FLAGS
1539                                ;.....CHECK THIS.....
1540                                ;NOTE THAT THE PARITY BIT SHOULD
1541                                ;SHOW UP IN THE DATA
1542                                ;IE. BIT SEVEN FOR SEVEN LEVEL CODE
1543 004366 005367 174532      DEC   COUNT      ;# OF SYNC CHARS
1544 004372 001445          BEQ   5$         ;FINISHED ? GET OUT OF TEST
1545 004374 022767 000003 174522      CMP   #3,COUNT   ;# OF SYNC CHARS
1546 004402 001423          BEQ   6$         ;CHECK FRAME ERROR ?
1547 004404 022767 000002 174512      CMP   #2,COUNT   ;# OF SYNC CHARS
1548 004412 001426          BEQ   7$         ;CHECK FRAME ERROR & BAD PARITY ?
1549                                ;NOPE THEN IT (COUNT) MUST BE = 1 THEREFORE....
1550 004414 012767 000012 174500      MOV   #10,$SHIFT ;# OF SHIFTS
1551 004422 012767 000054 175050      MOV   #54,$TMP1  ;FRAME & PARITY ERROR
1552 004430 004767 012466      JSR   PC,RPOKE   ;SHIFT IN THIS CHAR
1553                                ;NOW DON'T READ THE RXDBUF TO CREATE OVER RUN
1554 004434 012767 000054 175036      MOV   #54,$TMP1  ;FRAME & PARITY ERROR
1555 004442 012700 170026      MOV   @RXERR!OVRUN!FRMERR!PARER!26,RO ;EXPECTED
1556 004446 000167 177660          JMP   3$         ;DO IT AGAIN
1557 004452 012767 000454 175020 6S:   MOV   #454,$TMP1 ;BAD STOP BIT FOR FRAME ERROR
1558 004460 012700 120226      MOV   @RXERR!FRMERR!226,RO ;EXPECTED
1559 004464 000167 177642          JMP   3$         ;DO IT AGAIN
1560 004470 012767 000054 175002 7S:   MOV   #54,$TMP1  ;BAD STOP BIT & PARITY
1561 004476 012700 130026      MOV   @RXERR!FRMERR!PARER!26,RO ;EXPECTED
1562 004502 000167 177624          JMP   3$         ;DO IT AGAIN
1563
1564 5S:
1565 ;;THIS TEST PROVES THAT RXERR FREEZES THE "RECEIVER RESET"
1566 ;;WHILE IN STRIP SYNC MODE
1566 ;;THIS TEST FIRST PROVES THAT AUTOMATIC RESETS OCCUR WHEN

```



```

1567      ;; STRIP SYNC IS SET & SYNC CHARACTERS ARE SENT
1568      ;;      BUT IF AN ERROR SHOULD OCCUR....THIS AUTOMATIC RESET
1569      ;; IS DISCOMBOLATED
1570      ;; IE. FORCE PARITY ERROR WHILE STRIP SYNC IS SET
1571      ;; NOTE: NORMALLY THE LOGIC RESETS THE RXDONE & ERROR FLAGS
1572      ;; PROVIDING THAT ONLY GOOD SYNC CHARACTERS ARE SENT
1573      ;; BUT, IF AN RXERR OCCURS  RXDONE PLUS RXERR ARE ASSERTED
1574      ;; MODE: ISOC (ISYMOD)
1575      ;; LENGTH: SIX
1576      ;; PARITY: EVEPAR
1577      ;; CHARACTER EXPECTED: 126
1578      ;;      NOTE THAT THE PARITY BIT SHOULD SHOW
1579      ;;      UP IN THE DATA  IE. BIT SIX FOR
1580      ;;      SIX LEVEL CODE
1581      ;; CHARACTER SENT: SYNC CHARACTER
1582      ;; NOTE: THIS TEST USES ONLY THE RECEIVER LOGIC
1583
1584      *****
1585      ST4:  SCOPE
1586      BIS    #MRESET,@TXCSR ;MASTER RESET
1587      MOV    #ISYMOD,@PARCSR ;SET THE MODE
1588      BIS    #MRESET,@TXCSR ;MASTER RESET
1589
1590      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
1591      MOV    @MNTDATA:CLK!MINT!BREAK,@TXCSR
1592
1593      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
1594      MOV    #ISYMOD!SIX!EVEPAR!126,@PARCSR
1595      MOV    @RXDBUF,R3      ;SET UP FOR ERROR MSG
1596      MOV    #3,COUNT       ;# OF TIMES SYNC CHAR WILL BE SENT
1597      BIS    #SYNSCH,@RXCSR ;SET SYNC SEARCH
1598      ;POKE CLK TO GET RECEIVER INTO SYNCRIZATION....
1599      BIC    @CLK,@TXCSR    ;POKE CLK DOWN
1600      BIS    @CLK,@TXCSR    ;POKE CLK UP
1601      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
1602      BIC    @CLK,@TXCSR    ;POKE CLK DOWN
1603      BIS    @CLK,@TXCSR    ;POKE CLK UP
1604      BIS    #STPSYN,@RXCSR ;SET STRIP SYNC
1605      MOV    #9,SHIFT       ;# OF SHIFTS
1606      MOV    #654,STMP1     ;SYNC CHAR + START&STOP+ PARITY
1607      JSR    PC,RPOKE       ;SHIFT IN THIS CHARACTER
1608      TSTB  @RXCSR ;RXDONE = 0 ?
1609      BPL   .+4
1610      ERROR 4 ;RXDONE SHOULD NOT BE SET
1611      DEC   COUNT ;# OF SYNC CHARS
1612      BNE  2$ ;GO AGAIN ?
1613      MOV  #126,R0 ;EXPECTED
1614      MOV  @RXDBUF,R1 ;ACTUAL
1615      ;NOTE THAT THIS IS THE FIRST TIME
1616      ;RXDBUF IS READ.....THERE SHOULD BE
1617      ;NO OVER RUN ERROR 45
1618      CMP  R0,R1 ;COMPARE EXPECTED VS ACTUAL
1619      BEQ  .+4
1620      ERROR 2 ;DATA CHARS SHOULD COMPARE
1621      ;THERE SHOULD BE NO RXERR'S
1622      MOV  #4,COUNT ;# OF TIMES
    
```

```

1623 004706 012700 110026      MOV      #RXERR!PARER!26,RO      ;EXPECTED
1624 004712 012767 000454 174560  MOV      #454,STMP1             ;BAD SYNC CHAR (WRONG PARITY)
1625 004720 012767 000011 174174 3$:  MOV      #9,SHIFT              ;# OF SHIFTS
1626 004726 004767 012170      JSR      PC,RPOKE              ;SHIFT IN THIS CHAR
1627 004732 105777 174754      TSTB    #RXCSR ;RXDONE = 1?
1628 004736 100401      BMI     .+4
1629 004740 104004      ERROR   4 ;RXDONE SHOULD BE SET
1630 004742 017701 174750      MOV      #RXDBUF,R1           ;ACTUAL DATA
1631 004746 020001      CMP     RO,R1 ;COMPARE EXP VS ACT
1632 004750 001401      BEQ     .+4
1633 004752 104000      ERROR   ;DID THE RESPECTIVE ERROR 4 STOP THE
1634                                     ;AUTOMATIC RESSETTING OF RXDONE & ERROR FLAGS
1635                                     ;CHECK THIS
1636                                     ;NOTE THAT THE PARITY BIT SHOULD
1637                                     ;SHOW UP IN THE DATA
1638                                     ;IE. BIT SIX FOR SIX LEVEL CODE
1639 004754 005367 174144      DEC     COUNT ;# OF SYNC CHARS
1640 004760 001445      BEQ     5$ ;FINISHED ? GET OUT OF TEST
1641 004762 022767 000003 174134  CMP     #3,COUNT ;# OF SYNC CHARS
1642 004770 001423      BEQ     6$ ;CHECK FRAME ERROR ?
1643 004772 022767 000002 174124  CMP     #2,COUNT ;# OF SYNC CHARS
1644 005000 001426      BEQ     7$ ;CHECK FRAME ERROR & BAD PARITY ?
1645                                     ;NOPE THEN IT (COUNT) MUST BE = 1 THEREFORE....
1646 005002 012767 000011 174112  MOV      #9,SHIFT ;# OF SHIFTS
1647 005010 012767 000054 174462  MOV      #54,STMP1 ;FRAME & PARITY ERROR
1648 005016 004767 012100      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
1649                                     ;NOW DON'T READ THE RXDBUF TO CREATE OVER RUN
1650 005022 012767 000054 174450  MOV      #54,STMP1 ;FRAME & PARITY ERROR
1651 005030 012700 170026      MOV      #RXERR!OVRUN!FRMERR!PARER!26,RO ;EXPECTED
1652 005034 000167 177660      JMP     3$ ;DO IT AGAIN
1653 005040 012767 000254 174432 6$:  MOV      #254,STMP1 ;BAD STOP BIT FOR FRAME ERROR
1654 005046 012700 120126      MOV      #RXERR!FRMERR!126,RO ;EXPECTED
1655 005052 000167 177642      JMP     3$ ;DO IT AGAIN
1656 005056 012767 000054 174414 7$:  MOV      #54,STMP1 ;BAD STOP BIT & PARITY
1657 005064 012700 130026      MOV      #RXERR!FRMERR!PARER!26,RO ;EXPECTED
1658 005070 000167 177624      JMP     3$ ;DO IT AGAIN
1659
1660 5$:
1661 ;; THIS TEST PROVES THAT RXERR FREEZES THE "RECEIVER RESET"
1662 ;; WHILE IN STRIP SYNC MODE
1663 ;; THIS TEST FIRST PROVES THAT AUTOMATIC RESETS OCCUR WHEN
1664 ;; STRIP SYNC IS SET & SYNC CHARACTERS ARE SENT
1665 ;; BUT IF AN ERROR SHOULD OCCUR....THIS AUTOMATIC RESET
1666 ;; IS DISCOMBOBULATED
1667 ;; IE. FORCE PARITY ERROR WHILE STRIP SYNC IS SET
1668 ;; NOTE: NORMALLY THE LOGIC RESETS THE RXDONE & ERROR FLAGS
1669 ;; PROVIDING THAT ONLY GOOD SYNC CHARACTERS ARE SENT
1670 ;; BUT IF AN RXERR OCCURS RXDONE PLUS RXERR ARE ASSERTED
1671 ;; MODE: ISOC (ISYMOD)
1672 ;; LENGTH: FIVE
1673 ;; PARITY: EVEPAR
1674 ;; CHARACTER EXPECTED:66
1675                                     ;NOTE THAT THE PARITY BIT SHOULD SHOW
1676                                     ;UP IN THE DATA IE. BIT FIVE FOR
1677                                     ;FIVE LEVEL CODE
1678 ;; CHARACTER SENT: SYNC CHARACTER
1679 ;; NOTE: THIS TEST USES ONLY THE RECEIVER LOGIC
    
```



# K03

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 38  
 DZDUTA.M11 13-OCT-76 08:39 INITIALIZE THE COMMON TAGS

```

1679
1680
1681 005074 000004
1682 005076 052777 000400 174622
1683 005104 012777 000000 174610
1684 005112 052777 000400 174606
1685
1686
1687 005120 012777 064001 174600
1688
1689
1690 005126 012777 001466 174566
1691 005134 016703 174556
1692 005140 012767 000003 173756
1693 005146 052777 000020 174536
1694
1695 005154 042777 020000 174544
1696 005162 052777 020000 174536
1697
1698 005170 042777 020000 174530
1699 005176 052777 020000 174522
1700 005204 052777 000400 174500
1701 005212 012767 000010 173702
1702 005220 012767 000354 174252
1703 005226 004767 011670
1704 005232 105777 174454
1705 005236 100001
1706 005240 104004
1707 005242 005367 173656
1708 005246 001361
1709 005250 012700 000066
1710 005254 017701 174436
1711
1712
1713
1714 005260 020001
1715 005262 001401
1716 005264 104002
1717
1718 005266 012767 000004 173630
1719 005274 012700 110026
1720 005300 012767 000254 174172
1721 005306 012767 000010 173606
1722 005314 004767 011602
1723 005320 105777 174366
1724 005324 100401
1725 005326 104004
1726 005330 017701 174362
1727 005334 020001
1728 005336 001401
1729 005340 104000
1730
1731
1732
1733
1734

;*****
;STS: SCOPE
;      BIS      @MRESET,@TXCSR ;MASTER RESET
;      MOV      @ISYMOD,@PARCSR ;SET THE MODE
;      BIS      @MRESET,@TXCSR ;MASTER RESET
;SET MAINT DATA,CLK BREAK,&MAINTENANCE MODE
;      MOV      @MTDATA!CLK!MINT!BREAK,@TXCSR
;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
;      MOV      @ISYMOD!FIVE!EVEPAR!66,@PARCSR
;      MOV      @RXDBUF,R3 ;SET UP FOR ERROR MSG
;      MOV      #3,COUNT ;# OF TIMES SYNC CHAR WILL BE SENT
;      BIS      @SYNSCH,@RXCSR ;SET SYNC SEARCH
;      ;POKE CLK TO GET RECEIVER INTO SYNCRIZATION....
;      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
;      BIS      @STPSYN,@RXCSR ;SET STRIP SYNC
2S:  MOV      #8,SHIFT ;# OF SHIFTS
;      MOV      @354,STMP1 ;SYNC CHAR + START&STOP+ PARITY
1S:  JSR      PC,@POKE ;SHIFT IN THIS CHARACTER
;      TSTB     @RXCSR ;RXDONE = 0 ?
;      BPL      .+4
;      ERROR    4 ;RXDONE SHOULD NOT BE SET
;      DEC      COUNT ;# OF SYNC CHARS
;      BNE     2S ;GO AGAIN ?
;      MOV      @66,R0 ;EXPECTED
;      MOV      @RXDBUF,R1 ;ACTUAL
;      ;NOTE THAT THIS IS THE FIRST TIME
;      ;RXDBUF IS READ.....THERE SHOULD BE
;      ;NO OVER RUN ERROR 4S
;      CMP      R0,R1 ;COMPARE EXPECTED VS ACTUAL
;      BEQ     .+4
;      ERROR    2 ;DATA CHARS SHOULD COMPARE
;      ;THERE SHOULD BE NO RXERR'S
;      MOV      #4,COUNT ;# OF TIMES
;      MOV      @RXERR!PARER!26,R0 ;EXPECTED
;      MOV      @254,STMP1 ;BAD SYNC CHAR (WRONG PARITY)
3S:  MOV      #8,SHIFT ;# OF SHIFTS
;      JSR      PC,@POKE ;SHIFT IN THIS CHAR
;      TSTB     @RXCSR ;RXDONE = 1?
;      BMI     .+4
;      ERROR    4 ;RXDONE SHOULD BE SET
;      MOV      @RXDBUF,R1 ;ACTUAL DATA
;      CMP      R0,R1 ;COMPARE EXP VS ACT
;      BEQ     .+4
;      ERROR    ;DID THE RESPECTIVE ERROR 4 STOP THE
;      ;AUTOMATIC RESSETTING OF RXDONE & ERROR FLAGS
;      ;.....CHECK THIS.....
;      ;NOTE THAT THE PARITY BIT SHOULD
;      ;SHOW UP IN THE DATA
;      ;IE. BIT FIVE FOR FIVE LEVEL CODE

```



```

1735 005342 005367 173556      DEC    COUNT    ; # OF SYNC CHARS
1736 005346 001445              BEQ    55       ; FINISHED ? GET OUT OF TEST
1737 005350 022767 000003 173546  CMP    #3,COUNT ; # OF SYNC CHARS
1738 005356 001423              BEQ    65       ; CHECK FRAME ERROR ?
1739 005360 022767 000002 173536  CMP    #2,COUNT ; # OF SYNC CHARS
1740 005366 001426              BEQ    75       ; CHECK FRAME ERROR & BAD PARITY ?
1741                               ; NOPE THEN IT (COUNT) MUST BE = 1 THEREFORE....
1742 005370 012767 000010 173524  MOV    #8,SHIFT ; # OF SHIFTS
1743 005376 012767 000054 174074  MOV    #54,STMP1 ; FRAME & PARITY ERROR
1744 005404 004767 011512      JSR    PC,RPOKE ; SHIFT IN THIS CHAR
1745                               ; NOW DON'T READ THE RXDBUF TO CREATE OVER RUN
1746 005410 012767 000054 174062  MOV    #54,STMP1 ; FRAME & PARITY ERROR
1747 005416 012700 170026      MOV    #RXERR!OVRUN!FRMERR!PARER!26,RO ; EXPECTED
1748 005422 000167 177660      JMP    35       ; DO IT AGAIN
1749 005426 012767 000154 174044 65:   MOV    #154,STMP1 ; BAD STOP BIT FOR FRAME ERROR
1750 005434 012700 120066      MOV    #RXERR!FRMERR!66,RO ; EXPECTED
1751 005440 000167 177642      JMP    35       ; DO IT AGAIN
1752 005444 012767 000054 174026 75:   MOV    #54,STMP1 ; BAD STOP BIT & PARITY
1753 005452 012700 130026      MOV    #RXERR!FRMERR!PARER!26,RO ; EXPECTED
1754 005456 000167 177624      JMP    35       ; DO IT AGAIN
1755                               55:
1756                               ; THIS TEST VERIFYS WORD LENGTH SELECT OF
1757                               ; THE TRANSMITTER SECTION, IT USES THE DNA FLAG
1758                               ; AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
1759                               ; CORRECTLY
1760                               ; NOTE: DNA COMES UP ON THE FIRST RISING BIT
1761                               ; EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
1762                               ; LOADED INTO TXDBUF
1763                               ; MODE:SYNINT
1764                               ; PARITY:NO PARITY
1765                               ; LENGTH:FIVE
1766                               ; *****
1767                               ; *****
1768 005462 000004              tst6:  SCOPE
1769 005464 052777 000400 174234  BIS    #MRESET,@TXCSR ; MASTER RESET
1770 005472 012777 030000 174222  MOV    #SYNINT,@PARCSR ; SET THE MODE
1771 005500 052777 000400 174220  BIS    #MRESET,@TXCSR ; MASTER RESET
1772
1773                               ; SET MAINTENANCE MODE & SEND
1774                               ; NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1775 005506 012777 004020 174212  MOV    #MINT!SEND,@TXCSR
1776
1777                               ; SET MODE, # OF BITS, PARITY SENSE, & LOAD SYNC REG
1778 005514 012777 030026 174200  MOV    #SYNINT!FIVE!NOPAR!26,@PARCSR
1779 005522 016703 174200      MOV    TXCSR,R3 ; SET UP FOR ERROR MSG
1780 005526 112777 000021 174176  MOVB  #21,@TXDBUF ; LOAD CHAR
1781 005534 012767 000021 173736  MOV    #21,STMP1 ; SHIFTED CHAR
1782 005542 012767 000005 173352  MOV    #5,SHIFT ; # OF SHIFTS
1783                               ; POKE CLK TO GET INTO SYNCHRONIZATION
1784 005550 052777 020000 174150  BIS    #CLK,@TXCSR ; POKE CLK UP
1785 005556 042777 020000 174142  BIC    #CLK,@TXCSR ; POKE CLK DOWN
1786 005564 005000              15:   CLR    RO
1787 005566 006067 173706      ROR    STMP1 ; FORCE CARRY
1788 005572 103002              BCC    25
1789 005574 052700 002000      BIS    #BITW,RO ; EQUIV OF BIT WINDOW
1790 005600              25:

```

# M03

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 40  
 DZDUTA.M11 13-OCT-76 08:39 INITIALIZE THE COMMON TAGS

```

1791 005600 052777 020000 174120 BIS #CLK,@TXCSR ;POKE CLK UP
1792 005606 042777 020000 174112 BIC #CLK,@TXCSR ;POKE CLK DOWN
1793 005614 017701 174106 MOV @TXCSR,R1 ;ACTUAL
1794 005620 042701 075777 BIC #075777,R1 ;SAVE BITW & DNA
1795 005624 020001 CMP R0,R1 ;COMPARE EXP VS ACT
1796 005626 001401 BEQ +4
1797 005630 104003 ERROR 3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1798 ;BIT... ALSO CHECK DNA
1799 005632 005367 173264 DEC SHIFT ;# OF SHIFTS
1800 005636 001352 BNE IS ;DO IT AGAIN ?
1801 ;NOW POKE CLK TO SEE DNA
1802 005640 052777 020000 174060 BIS #CLK,@TXCSR ;POKE CLK
1803 005646 012700 100000 MOV #100000,R0 ;EXPECTED
1804 005652 017701 174050 MOV @TXCSR,R1 ;ACTUAL
1805 005656 042701 077777 BIC #77777,R1 ;SAVE DNA ONLY
1806 005662 020001 CMP R0,R1 ;COMPARE EXPECTED VS ACTUAL
1807 005664 001401 BEQ +4
1808 005666 104003 ERROR 3 ;DNA SHOULD BE SET
1809 ;IF DNA DID NOT SET,CHECK WORD LENGTH
1810 ;SELECT LOGIC OF THE TRANSMITTER
1811 005670 005777 174032 TST @TXCSR ;DNA ?
1812 005674 100001 BPL +4
1813 005676 104004 ERROR 4 ;DNA SHOULD NOT BE SET
1814 ;IT SHOULD HAVE BEEN CLEARED FROM
1815 ;PREVIOUS READ
1816
1817 ;: THIS TEST VERIFYS WORD LENGTH SELECT OF
1818 ;: THE TRANSMITTER SECTION, IT USES THE DNA FLAG
1819 ;: AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
1820 ;: CORRECTLY
1821 ;: NOTE: DNA COMES UP ON THE FIRST RISING BIT
1822 ;: EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
1823 ;: LOADED INTO TXDBUF
1824 ;: MODE:SYNINT
1825 ;: PARITY:NO PARITY
1826 ;: LENGTH:SIX
1827
1828 ;:*****
1829 005700 000004 TST7: SCOPE
1830 005702 052777 000400 174016 BIS #MRESET,@TXCSR ;MASTER RESET
1831 005710 012777 030000 174004 MOV #SYNINT,@PARCSR ;SET THE MODE
1832 005716 052777 000400 174002 BIS #MRESET,@TXCSR ;MASTER RESET
1833
1834 ;SET MAINTENANCE MODE & SEND
1835 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1836 005724 012777 004020 173774 MOV #MINT!SEND,@TXCSR
1837
1838 ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
1839 005732 012777 032026 173762 MOV #SYNINT!SIX!NOPAR!26,@PARCSR
1840 005740 016703 173762 MOV TXCSR,R3 ;SET UP FOR ERROR MSG
1841 005744 112777 000021 173760 MOVB #21,@TXDBUF ;LOAD CHAR
1842 005752 012767 000021 173520 MOV #21,$TMP1 ;SHIFTED CHAR
1843 005760 012767 000006 173134 MOV #6,$SHIFT ;# OF SHIFTS
1844 ;POKE CLK TO GET INTO SYNCRONIZATION
1845 005766 052777 020000 173732 BIS #CLK,@TXCSR ;POKE CLK UP
1846 005774 042777 020000 173724 BIC #CLK,@TXCSR ;POKE CLK DOWN
  
```



# N03

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 41  
 DZDUTA.M11 13-OCT-76 08:39 INITIALIZE THE COMMON TAGS

```

1847 006002 005000          15: CLR      RO
1848 006004 006067 173470 ROR      $TMP1 ;FORCE CARRY
1849 006010 103002          BCC      25
1850 006012 052700 002000 BIS      #BITW,RO ;EQUIV OF BIT WINDOW
1851 006016          25: BIS      #CLK,@TXCSR ;POKE CLK UP
1852 006016 052777 020000 173702 BIC      #CLK,@TXCSR ;POKE CLK DOWN
1853 006024 042777 020000 173674 MOV      @TXCSR,R1 ;ACTUAL
1854 006032 017701 173670 BIC      #075777,R1 ;SAVE BITW & DNA
1855 006036 042701 075777 CMP      RO,R1 ;COMPARE EXP VS ACT
1856 006042 020001 BEQ      +4
1857 006044 001401 ERROR    3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1858 006046 104003 ;BIT ALSO CHECK DNA
1859 ;# OF SHIFTS
1860 006050 005367 173046 DEC      SHIFT ;DO IT AGAIN ?
1861 006054 001352 ;NOW POKE CLK TO SEE DNA
1862 BNE      15
1863 006056 052777 020000 173642 BIS      #CLK,@TXCSR ;POKE CLK
1864 006064 012700 100000 MOV      #100000,RO ;EXPECTED
1865 006070 017701 173632 MOV      @TXCSR,R1 ;ACTUAL
1866 006074 042701 077777 BIC      #77777,R1 ;SAVE DNA ONLY
1867 006100 020001 CMP      RO,R1 ;COMPARE EXPECTED VS ACTUAL
1868 006102 001401 BEQ      +4
1869 006104 104003 ERROR    3 ;DNA SHOULD BE SET
1870 ;IF DNA DID NOT SET,CHECK WORD LENGTH
1871 ;SELECT LOGIC OF THE TRANSMITTER
1872 006106 005777 173614 TST      @TXCSR ;DNA ?
1873 006112 100001 BPL      +4
1874 006114 104004 ERROR    4 ;DNA SHOULD NOT BE SET
1875 ;IT SHOULD HAVE BEEN CLEARED FROM
1876 ;PREVIOUS READ
1877
1878 ;: THIS TEST VERIFYS WORD LENGTH SELECT OF
1879 ;: THE TRANSMITTER SECTION, IT USES THE DNA FLAG
1880 ;: AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
1881 ;: CORRECTLY
1882 ;: NOTE: DNA COMES UP ON THE FIRST RISING BIT
1883 ;: EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
1884 ;: LOADED INTO TXDBUF
1885 ;: MODE:SYNINT
1886 ;: PARITY:NO PARITY
1887 ;: LENGTH:SEVEN
1888 ;:
1889 ;:*****
1890 006116 000004          †ST10: SCOPE
1891 006120 052777 000400 173600 BIS      #MRESET,@TXCSR ;MASTER RESET
1892 006126 012777 030000 173566 MOV      #SYNINT,@PARCSR ;SET THE MODE
1893 006134 052777 000400 173564 BIS      #MRESET,@TXCSR ;MASTER RESET
1894
1895 ;SET MAINTENANCE MODE & SEND
1896 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1897 006142 012777 004020 173556 MOV      #MINT!SEND,@TXCSR
1898
1899 ;SET MODE,# OF BITS,PARITY SENSE & LOAD SYNC REG
1900 006150 012777 034026 173544 MOV      #SYNINT!SEVEN!NOPAR!26,@PARCSR
1901 006156 016703 173544 MOV      TXCSR,R3 ;SET UP FOR ERROR MSG
1902 006162 112777 000021 173542 MOVB    #21,@TXDBUF ;LOAD CHAR
  
```

```

1903 006170 012767 000021 173302      MOV      #21,STMP1      ;SHIFTED CHAR
1904 006176 012767 000007 172716      MOV      #7,SHIFT      ;# OF SHIFTS
1905                                     ;POKE CLK TO GET INTO SYNCHRONIZATION
1906 006204 052777 020000 173514      BIS      @CLK,@TXCSR    ;POKE CLK UP
1907 006212 042777 020000 173506      BIC      @CLK,@TXCSR    ;POKE CLK DOWN
1908 006220 005000                                     1S:     CLR      R0
1909 006222 006067 173252                                     ROR      STMP1      ;FORCE CARRY
1910 006226 103002                                     BCC      2S
1911 006230 052700 002000                                     BIS      @BITW,R0      ;EQUIV OF BIT WINDOW
1912 006234                                     2S:
1913 006234 052777 020000 173464      BIS      @CLK,@TXCSR    ;POKE CLK UP
1914 006242 042777 020000 173456      BIC      @CLK,@TXCSR    ;POKE CLK DOWN
1915 006250 017701 173452      MOV      @TXCSR,R1      ;ACTUAL
1916 006254 042701 075777      BIC      @075777,R1     ;SAVE BITW & DNA
1917 006260 020001      CMP      R0,R1      ;COMPARE EXP VS ACT
1918 006262 001401      BEQ      +4
1919 006264 104003      ERROR    3      ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1920                                     ;BIT... ALSO CHECK DNA
1921 006266 005367 172630      DEC      SHIFT      ;# OF SHIFTS
1922 006272 001352      BNE      1S      ;DO IT AGAIN ?
1923                                     ;NOW POKE CLK TO SEE DNA
1924 006274 052777 020000 173424      BIS      @CLK,@TXCSR    ;POKE CLK
1925 006302 012700 100000      MOV      #100000,R0     ;EXPECTED
1926 006306 017701 173414      MOV      @TXCSR,R1      ;ACTUAL
1927 006312 042701 077777      BIC      @77777,R1     ;SAVE DNA ONLY
1928 006316 020001      CMP      R0,R1      ;COMPARE EXPECTED VS ACTUAL
1929 006320 001401      BEQ      +4
1930 006322 104003      ERROR    3      ;DNA SHOULD BE SET
1931                                     ;IF DNA DID NOT SET ,CHECK WORD LENGTH
1932                                     ;SELECT LOGIC OF THE TRANSMITTER
1933 006324 005777 173376      TST      @TXCSR      ;DNA ?
1934 006330 100001      BPL      +4
1935 006332 104004      ERROR    4      ;DNA SHOULD NOT BE SET
1936                                     ;IT SHOULD HAVE BEEN CLEARED FROM
1937                                     ;PREVIOUS READ
1938
1939                                     ;: THIS TEST VERIFYS WORD LENGTH SELECT OF
1940                                     ;: THE TRANSMITTER SECTION, IT USES THE DNA FLAG
1941                                     ;: AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
1942                                     ;: CORRECTLY
1943                                     ;: NOTE: DNA COMES UP ON THE FIRST RISING BIT
1944                                     ;: EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
1945                                     ;: LOADED INTO TXDBUF
1946                                     ;: MODE:SYNINT
1947                                     ;: PARITY:NO PARITY
1948                                     ;: LENGTH:EIGHT
1949
1950                                     ;:*****
1951 006334 000004      †ST11: SCOPE
1952 006336 052777 000400 173362      BIS      @MRESET,@TXCSR ;MASTER RESET
1953 006344 012777 030000 173350      MOV      @SYNINT,@PARCSR ;SET THE MODE
1954 006352 052777 000400 173346      BIS      @MRESET,@TXCSR ;MASTER RESET
1955
1956                                     ;SET MAINTENANCE MODE & SEND
1957                                     ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
1958 006360 012777 004020 173340      MOV      @MINT!SEND,@TXCSR
  
```



```

1959
1960 ;SET MODE, # OF BITS, PARITY SENSE, & LOAD SYNC REG
1961 006366 012777 036026 173326 MOV #SYNINT!EIGHT!NOPAR!26,@PARCSR
1962 006374 016703 173326 MOV TXCSR,R3 ;SET UP FOR ERROR MSG
1963 006400 112777 000021 173324 MOV #21,@TXDBUF ;LOAD CHAR
1964 006406 012767 000021 173064 MOV #21,$TMP1 ;SHIFTED CHAR
1965 006414 012767 000010 172500 MOV #8,SHIFT ;# OF SHIFTS
1966 ;POKE CLK TO GET INTO SYNCHRONIZATION
1967 006422 052777 020000 173276 BIS #CLK,@TXCSR ;POKE CLK UP
1968 006430 042777 020000 173270 BIC #CLK,@TXCSR ;POKE CLK DOWN
1969 006436 005000 CLR R0
1970 006440 006067 173034 ROR $TMP1 ;FORCE CARRY
1971 006444 103002 BCC 2$
1972 006446 052700 002000 BIS #BITW,R0 ;EQUIV OF BIT WINDOW
1973 006452 2$:
1974 006452 052777 020000 173246 BIS #CLK,@TXCSR ;POKE CLK UP
1975 006460 042777 020000 173240 BIC #CLK,@TXCSR ;POKE CLK DOWN
1976 006466 017701 173234 MOV @TXCSR,R1 ;ACTUAL
1977 006472 042701 075777 BIC #075777,R1 ;SAVE BITW & DNA
1978 006476 020001 CMP R0,R1 ;COMPARE EXP VS ACT
1979 006500 001401 BEQ +4
1980 006502 104003 ERROR 3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
1981 ;BIT... ALSO CHECK DNA
1982 006504 005367 172412 DEC SHIFT ;# OF SHIFTS
1983 006510 001352 BNE 1$ ;DO IT AGAIN ?
1984 ;NOW POKE CLK TO SEE DNA
1985 006512 052777 020000 173206 BIS #CLK,@TXCSR ;POKE CLK
1986 006520 012700 100000 MOV #100000,R0 ;EXPECTED
1987 006524 017701 173176 MOV @TXCSR,R1 ;ACTUAL
1988 006530 042701 077777 BIC #77777,R1 ;SAVE DNA ONLY
1989 006534 020001 CMP R0,R1 ;COMPARE EXPECTED VS ACTUAL
1990 006536 001401 BEQ +4
1991 006540 104003 ERROR 3 ;DNA SHOULD BE SET
1992 ;IF DNA DID NOT SET, CHECK WORD LENGTH
1993 ;SELECT LOGIC OF THE TRANSMITTER
1994 006542 005777 173160 TST @TXCSR ;DNA ?
1995 006546 103001 BPL +4
1996 006550 104004 ERROR 4 ;DNA SHOULD NOT BE SET
1997 ;IT SHOULD HAVE BEEN CLEARED FROM
1998 ;PREVIOUS READ
1999
2000 ;: THIS TEST VERIFYS WORD LENGTH SELECT OF
2001 ;: THE TRANSMITTER SECTION, IT USES THE DNA FLAG
2002 ;: AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
2003 ;: CORRECTLY
2004 ;: NOTE: DNA COMES UP ON THE FIRST RISING BIT
2005 ;: EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
2006 ;: LOADED INTO TXDBUF
2007 ;: MODE:SYNEXT
2008 ;: PARITY:NO PARITY
2009 ;: LENGTH:FIVE
2010
2011 ;:*****
2012 006552 000004 TST12: SCOPE
2013 006554 052777 000400 173144 BIS #MRESET,@TXCSR ;MASTER RESET
2014 006562 012777 020000 173132 MOV #SYNEXT,@PARCSR ;SET THE MODE

```

```

2015 006570 052777 000400 173130      BIS      #MRESET,@TXCSR ;MASTER RESET
2016
2017 ;SET MAINTENANCE MODE & SEND
2018 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
2019 006576 012777 004020 173122      MOV      #MINT!SEND,@TXCSR
2020
2021 ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
2022 006604 012777 020026 173110      MOV      #SYNEXT!FIVE!NOPAR!26,@PARCSR
2023 006612 016703 173110      MOV      TXCSR,R3 ;SET UP FOR ERROR MSG
2024 006616 112777 000021 173106      MOVB    #21,@TXDBUF ;LOAD CHAR
2025 006624 012767 000021 172646      MOV      #21,STMP1 ;SHIFTED CHAR
2026 006632 012767 000005 172262      MOV      #5,SHIFT ;# OF SHIFTS
2027 ;POKE CLK TO GET INTO SYNCRONIZATION
2028 006640 052777 020000 173060      BIS      #CLK,@TXCSR ;POKE CLK UP
2029 006646 042777 020000 173052      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2030 006654 005000
2031 006656 006067 172616      1$:     CLR      RO
2032 006662 103002
2033 006664 052700 002000      ROR      STMP1 ;FORCE CARRY
2034 006670
2035 006670 052777 020000 173030      BCC     2$
2036 006676 042777 020000 173022      BIS      #BITW,RO ;EQUIV OF BIT WINDOW
2037 006704 017701 173016      2$:     BIS      #CLK,@TXCSR ;POKE CLK UP
2038 006710 042701 075777      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2039 006714 020001
2040 006716 001401
2041 006720 104003
2042
2043 006722 005367 172174      MOV     @TXCSR,R1 ;ACTUAL
2044 006726 001352
2045 ;NOW POKE CLK TO SEE DNA
2046 006730 052777 020000 172770      BIC     #075777,R1 ;SAVE BITW & DNA
2047 006736 012700 100000      CMP     RO,R1 ;COMPARE EXP VS ACT
2048 006742 017701 172760      BEQ     +4
2049 006746 042701 077777      ERROR   3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
2050 006752 020001
2051 006754 001401
2052 006756 104003
2053
2054
2055 006760 005777 172742      DEC     SHIFT ;# OF SHIFTS
2056 006764 100001
2057 006766 104004      BNE     1$ ;DO IT AGAIN ?
2058
2059
2060
2061 ;POKE CLK
2062 006730 052777 020000 172770      BIS     #CLK,@TXCSR ;POKE CLK
2063 006736 012700 100000      MOV     #100000,RO ;EXPECTED
2064 006742 017701 172760      MOV     @TXCSR,R1 ;ACTUAL
2065 006746 042701 077777      BIC     #77777,R1 ;SAVE DNA ONLY
2066 006752 020001
2067 006754 001401
2068 006756 104003
2069
2070      CMP     RO,R1 ;COMPARE EXPECTED VS ACTUAL
2071      BEQ     +4
2072      ERROR   3 ;DNA SHOULD BE SET
2073 ;IF DNA DID NOT SET ,CHECK WORD LENGTH
2074 ;SELECT LOGIC OF THE TRANSMITTER
2075      TST     @TXCSR ;DNA ?
2076      BPL     +4
2077      ERROR   4 ;DNA SHOULD NOT BE SET
2078 ;IT SHOULD HAVE BEEN CLEARED FROM
2079 ;PREVIOUS READ
2080
2081 ;: THIS TEST VERIFYS WORD LENGTH SELECT OF
2082 ;: THE TRANSMITTER SECTION, IT USES THE DNA FLAG
2083 ;: AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
2084 ;: CORRECTLY
2085 ;: NOTE: DNA COMES UP ON THE FIRST RISING BIT
2086 ;: EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
2087 ;: LOADED INTO TXDBUF
2088 ;: MODE:SYNEXT
2089 ;: PARITY:NO PARITY
2090 ;: LENGTH:SIX

```



```

2071
2072
2073 006770 000004
2074 006772 052777 000400 172726
2075 007000 012777 020000 172714
2076 007006 052777 000400 172712
2077
2078
2079
2080 007014 012777 004020 172704
2081
2082
2083 007022 012777 022026 172672
2084 007030 016703 172672
2085 007034 112777 000021 172670
2086 007042 012767 000021 172430
2087 007050 012767 000006 172044
2088
2089 007056 052777 020000 172642
2090 007064 042777 020000 172634
2091 007072 005000
2092 007074 006067 172400
2093 007100 103002
2094 007102 052700 002000
2095 007106
2096 007106 052777 020000 172612
2097 007114 042777 020000 172604
2098 007122 017701 172600
2099 007126 042701 075777
2100 007132 020001
2101 007134 001401
2102 007136 104003
2103
2104 007140 005367 171756
2105 007144 001352
2106
2107 007146 052777 020000 172552
2108 007154 012700 100000
2109 007160 017701 172542
2110 007164 042701 077777
2111 007170 020001
2112 007172 001401
2113 007174 104003
2114
2115
2116 007176 005777 172524
2117 007202 100001
2118 007204 104004
2119
2120
2121
2122
2123
2124
2125
2126

;*****
;T13: 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!SIX!NOPAR!26,@PARCSR
;MOV TXCSR,R3 ;SET UP FOR ERROR MSG
;MOVB #21,@TXDBUF ;LOAD CHAR
;MOV #21,STMP1 ;SHIFTED CHAR
;MOV #6,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 STMP1 ;FORCE CARRY
;BCC 2S
;BIS #BITW,R0 ;EQUIV OF BIT WINDOW
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
; ;# OF SHIFTS
; ;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 EXPECTED VS ACTUAL
;BEQ .+4
;ERROR 3 ;DNA SHOULD BE SET
; ;IF DNA DID NOT SET,CHECK WORD LENGTH
; ;SELECT LOGIC OF THE TRANSMITTER
; ;DNA ?

;TST @TXCSR
;BPL .+4
;ERROR 4 ;DNA SHOULD NOT BE SET
; ;IT SHOULD HAVE BEEN CLEARED FROM
; ;PREVIOUS READ

; ;THIS TEST VERIFYS WORD LENGTH SELECT OF
; ;THE TRANSMITTER SECTION,IT USES THE DNA FLAG
; ;AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
; ;CORRECTLY
; ;NOTE: DNA COMES UP ON THE FIRST RISING BIT
  
```





INITIALIZE THE COMMON TAGS

```

2183                                     ;: THIS TEST VERIFYS WORD LENGTH SELECT OF
2184                                     ;: THE TRANSMITTER SECTION, IT USES THE DNA FLAG
2185                                     ;: AND BIT WINDOW TO DETERMINE THAT IT WAS SELECTED
2186                                     ;: CORRECTLY
2187                                     ;: NOTE: DNA COMES UP ON THE FIRST RISING BIT
2188                                     ;: EDGE OF THE NEXT CHARACTER IF NO NEW CHARACTER IS
2189                                     ;: LOADED INTO TXDBUF
2190                                     ;: MODE:SYNEXT
2191                                     ;: PARITY:NO PARITY
2192                                     ;: LENGTH:EIGHT
2193
2194                                     ;:*****
2195 007424 000004                               †ST15: SCOPE
2196 007426 052777 000400 172272             BIS      #MRESET,@TXCSR ;MASTER RESET
2197 007434 012777 020000 172260             MOV      #SYNEXT,@PARCSR ;SET THE MODE
2198 007442 052777 000400 172256             BIS      #MRESET,@TXCSR ;MASTER RESET
2199
2200                                     ;SET MAINTENANCE MODE & SEND
2201                                     ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
2202 007450 012777 004020 172250             MOV      #MINT!SEND,@TXCSR
2203
2204                                     ;SET MODE, # OF BITS, PARITY SENSE, & LOAD SYNC REG
2205 007456 012777 026026 172236             MOV      #SYNEXT!EIGHT!NOPAR!26,@PARCSR
2206 007464 016703 172236                     MOV      TXCSR,R3 ;SET UP FOR ERROR MSG
2207 007470 112777 000021 172234             MOV      #21,@TXDBUF ;LOAD CHAR
2208 007476 012767 000021 171774             MOV      #21,$TMP1 ;SHIFTED CHAR
2209 007504 012767 000010 171410             MOV      #8,$SHIFT ;# OF SHIFTS
2210                                     ;POKE CLK TO GET INTO SYNCHRONIZATION
2211 007512 052777 020000 172206             BIS      #CLK,@TXCSR ;POKE CLK UP
2212 007520 042777 020000 172200             BIC      #CLK,@TXCSR ;POKE CLK DOWN
2213 007526 005000                               1$: CLR      R0
2214 007530 006067 171744                     ROR      $TMP1 ;FORCE CARRY
2215 007534 103002                               BCC     2$
2216 007536 052700 002000                     BIS      #BITW,R0 ;EQUIV OF BIT WINDOW
2217 007542
2218 007542 052777 020000 172156             BIS      #CLK,@TXCSR ;POKE CLK UP
2219 007550 042777 020000 172150             BIC      #CLK,@TXCSR ;POKE CLK DOWN
2220 007556 017701 172144                     MOV      @TXCSR,R1 ;ACTUAL
2221 007562 042701 075777                     BIC      #075777,R1 ;SAVE BITW & DNA
2222 007566 020001                               CMP     R0,R1 ;COMPARE EXP VS ACT
2223 007570 001401                               BEQ     +4
2224 007572 104003                               ERROR   3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
2225                                     ;BIT.....ALSO CHECK DNA
2226 007574 005367 171322                     DEC     SHIFT ;# OF SHIFTS
2227 007600 001352                               BNE     1$ ;DO IT AGAIN ?
2228
2229 007602 052777 020000 172116             ;NOW POKE CLK TO SEE DNA
2230 007610 012700 100000                     BIS      #CLK,@TXCSR ;POKE CLK
2231 007614 017701 172106                     MOV      #100000,R0 ;EXPECTED
2232 007620 042701 077777                     MOV      @TXCSR,R1 ;ACTUAL
2233 007624 020001                               BIC      #77777,R1 ;SAVE DNA ONLY
2234 007626 001401                               CMP     R0,R1 ;COMPARE EXPECTED VS ACTUAL
2235 007630 104003                               BEQ     +4
2236                                     ;DNA SHOULD BE SET
2237                                     ;IF DNA DID NOT SET ,CHECK WORD LENGTH
2238 007632 005777 172070                     ERROR   3 ;SELECT LOGIC OF THE TRANSMITTER
2239                                     ;DNA ?
  
```

H04

```

2239 007636 100001          BPL      .+4
2240 007640 104004          ERROR    4          ;DNA SHOULD NOT BE SET
                ;IT SHOULD HAVE BEEN CLEARED FROM
                ;PREVIOUS READ
                ;: THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
                ;: OF THE TRANSMITTER SECTION.
                ;: IT ALSO CHECKS DNA TIMING
                ;: MODE:SYNINT
                ;: LENGTH:FIVE PLUS PARITY
                ;: PARITY:EVEPAR
                ;: CHARACTER:25
2252 *****
2253 007642 000004          ST16:  SCOPE
2254 007644 052777 000400 172054  BIS     #MRESET,@TXCSR ;MASTER RESET
2255 007652 012777 030000 172042  MOV     #SYNINT,@PARCSR ;SET THE MODE
2256 007660 052777 000400 172040  BIS     #MRESET,@TXCSR ;MASTER RESET
                ;SET MAINTENANCE MODE & SEND
                ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
2259 007666 012777 004020 172032  MOV     #MINT!SEND,@TXCSR
                ;SET MODE, # OF BITS, PARITY SENSE & LOAD SYNC REG
2263 007674 012777 031426 172020  MOV     #SYNINT!FIVE!EVEPAR!26,@PARCSR
2264 007702 016703 172020  MOV     TXCSR,R3          ;SET UP FOR ERROR MSG
2265 007706 112777 000025 172016  MOV     #25,@TXDBUF      ;LOAD DATA CHAR
2266 007714 012767 000065 171556  MOV     #65,STMP1        ;TO BE SHIFTED CHAR
2267 007722 012767 000006 171172  MOV     #6,SHIFT         ;# OF SHIFTS
                ;POKE CLK TO GET INTO SYNCHRONIZATION
2269 007730 052777 020000 171770  BIS     #CLK,@TXCSR      ;POKE CLK UP
2270 007736 042777 020000 171762  BIC     #CLK,@TXCSR      ;POKE CLK DOWN
2271 007744 005000          15:    CLR     RO
2272 007746 006067 171526  ROR     STMP1            ;FORCE CARRY
2273 007752 103002          BCC     25              ;BR IF CARRY CLR
2274 007754 052700 002000          BIS     #BITW,RO        ;EQUIV OF BITW
2275 007760          25:
2276 007760 052777 020000 171740  BIS     #CLK,@TXCSR      ;POKE CLK UP
2277 007766 042777 020000 171732  BIC     #CLK,@TXCSR      ;POKE CLK DOWN
2278 007774 017701 171726  MOV     @TXCSR,R1        ;ACTUAL
2279 010000 042701 075777          BIC     #075777,R1      ;SAVE BITW & DNA
2280 010004 020001          CMP     RO,R1           ;COMPARE EXP VS ACT
2281 010006 001401          BEQ     .+4
2282 010010 104003          ERROR    3          ;BIT WINDOW DID NOT MATCH ACTUAL DATA
                ;BIT... ALSO CHECK DNA
2283          DEC     SHIFT    ;# OF SHIFTS
2284 010012 005367 171104          BNE     15              ;DO IT AGAIN ?
                ;NOW POKE CLK TO SEE DNA
2286 010020 052777 020000 171700  BIS     #CLK,@TXCSR      ;POKE CLK
2287 010026 012700 100000          MOV     #100000,RO      ;EXPECTED
2288 010032 017701 171670          MOV     @TXCSR,R1      ;ACTUAL
2289 010036 042701 077777          BIC     #77777,R1      ;SAVE DNA ONLY
2290 010042 020001          CMP     RO,R1           ;COMPARE EXP VS ACT
2291 010044 001401          BEQ     .+4
2292 010046 104003          ERROR    3          ;DNA SHOULD BE SET
                ;IF DNA DID NOT SET

```



2295  
2296  
2297  
2298  
2299  
2300  
2301  
2302  
2303  
2304  
2305  
2306  
2307  
2308  
2309  
2310  
2311  
2312  
2313  
2314  
2315  
2316  
2317  
2318  
2319  
2320  
2321  
2322  
2323  
2324  
2325  
2326  
2327  
2328  
2329  
2330  
2331  
2332  
2333  
2334  
2335  
2336  
2337  
2338  
2339  
2340  
2341  
2342  
2343  
2344  
2345  
2346  
2347  
2348  
2349  
2350

```

;CHECK WORD LENGTH SELECT LOGIC
;: THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
;: OF THE TRANSMITTER SECTION.
;: IT ALSO CHECKS DNA TIMING
;: MODE:SYNINT
;: LENGTH:FIVE PLUS PARITY
;: PARITY:ODDPAR
;: CHARACTER:25
;:*****
†ST17: 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!FIVE!ODDPAR!25,@PARCSR
MOV TXCSR,R3 ;SET UP FOR ERROR MSG
MOV #25,@TXDBUF ;LOAD DATA CHAR
MOV #25,STMP1 ;TO BE SHIFTED CHAR
MOV #6,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 STMP1 ;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
BC# +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 #10000,R0 ;EXPECTED
MOV @TXCSR,R1 ;ACTUAL
BIC #777777,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
    
```

2351  
2352  
2353  
2354  
2355  
2356  
2357  
2358  
2359  
2360  
2361  
2362  
2363  
2364  
2365  
2366  
2367  
2368  
2369  
2370  
2371  
2372  
2373  
2374  
2375  
2376  
2377  
2378  
2379  
2380  
2381  
2382  
2383  
2384  
2385  
2386  
2387  
2388  
2389  
2390  
2391  
2392  
2393  
2394  
2395  
2396  
2397  
2398  
2399  
2400  
2401  
2402  
2403  
2404  
2405  
2406

```

    ;; OF THE TRANSMITTER SECTION.
    ;; IT ALSO CHECKS DNA TIMING
    ;; MODE: ISYMOD
    ;; LENGTH: FIVE PLUS PARITY
    ;; PARITY: EVEPAR
    ;; CHARACTER: 25
    ;; *****
    TST20: 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!FIVE!EVEPAR!25, @PARCSR
    MOV      TXCSR, R3 ; SET UP FOR ERROR MSG
    MOVB     #25, @TXDBUF ; LOAD DATA CHAR
    MOV      #352, $TMP1 ; TO BE SHIFTED CHAR
    MOV      #8, $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
    DEC      $SHIFT ; # OF SHIFTS
    BNE      1$, ; DO IT AGAIN ?

    ; 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
    
```



K04

007  
008  
009  
010  
011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053  
054  
055  
056  
057  
058  
059  
060  
061  
062

010464 000004  
010466 052777 000400 171232  
010474 012777 000000 171220  
010502 052777 000400 171216  
  
010510 012777 004020 171210  
  
010516 012777 001026 171176  
010524 016703 171176  
010530 112777 000025 171174  
010536 012767 000252 170734  
010544 012767 000010 170350  
  
010552 052777 020000 171146  
010560 042777 020000 171140  
010566 005000  
010570 006067 170704  
010574 103002  
010576 052700 002000  
010602  
010602 052777 020000 171116  
010610 042777 020000 171110  
010616 017701 171104  
010622 042701 075777  
010626 020001  
010630 001401  
010632 104003  
  
010634 005367 170262  
010640 001352  
  
010642 052777 020000 171056  
010650 012700 000000  
010654 017701 171046  
010660 042701 077777  
010664 020001  
010666 001401  
010670 104003

```

::LENGTH:FIVE PLUS PARITY
::PARITY:00DPAR
::CHARACTER:25
::*****
TST21: 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!FIVE!00DPAR!26,@PARCSR
MOV TXCSR,R3 ;SET UP FOR ERROR MSG
MOVB #25,@TXDBUF ;LOAD DATA CHAR
MOV #252,@STMP1 ;TO BE SHIFTED CHAR
MOV #8,@SHIFT ;# OF SHIFTS

;POKE CLK TO GET INTO SYNCHRONIZATION
BIS #CLK,@TXCSR ;POKE CLK UP
BIC #CLK,@TXCSR ;POKE CLK DOWN
15: CLR R0
ROR STMP1 ;FORCE CARRY
BCC 25 ;BR IF CARRY CLR
BIS #BITW,R0 ;EQUIV OF BITW
25: 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 15 ;DO IT AGAIN ?

;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:SYNINT
::LENGTH:SIX PLUS PARITY
::PARITY:EVEPAR
```

463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518

010672 000004  
010674 052777 000400 171024  
010702 012777 030000 171012  
010710 052777 000400 171010  
  
010716 012777 004020 171002  
  
010724 012777 033426 170770  
010732 016703 170770  
010736 112777 000025 170766  
010744 012767 000125 170526  
010752 012767 000007 170142  
  
010760 052777 020000 170740  
010766 042777 020000 170732  
010774 005000  
010776 006067 170476  
011002 103002  
011004 052700 002000  
011010  
011010 052777 020000 170710  
011016 042777 020000 170702  
011024 017701 170676  
011030 042701 075777  
011034 020001  
011036 001401  
011040 104003  
  
011042 005367 170054  
011046 001352  
  
011050 052777 020000 170650  
011056 012700 100000  
011062 017701 170640  
011066 042701 077777  
011072 020001  
011074 001401  
011076 104003

```
;; CHARACTER:25
;*****
;ST22: 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!SIX!EVEPAR!26,@PARCSR
      MOV      TXCSR,R3 ;SET UP FOR ERROR MSG
      MOV      #25,@TXDBUF ;LOAD DATA CHAR
      MOV      #125,STMP1 ;TO BE SHIFTED CHAR
      MOV      #7,SHIFT ;# OF SHIFTS

;POKE CLK TO GET INTO SYNCHRONIZATION
      BIS      #CLK,@TXCSR ;POKE CLK UP
      BIC      #CLK,@TXCSR ;POKE CLK DOWN
1$: CLR      RO
      ROR      STMP1 ;FORCE CARRY
      BCC      2$, ;BR IF CARRY CLR
      BIS      #BITW,RO ;EQUIV OF BITW
2$: BIS      #CLK,@TXCSR ;POKE CLK UP
      BIC      #CLK,@TXCSR ;POKE CLK DOWN
      MOV      @TXCSR,R1 ;ACTUAL
      BIC      #075777,R1 ;SAVE BITW & DNA
      CMP      RO,R1 ;COMPARE EXP VS ACT
      BEQ      +4
      ERROR    3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
      ;BIT... ALSO CHECK DNA
      DEC      SHIFT ;# OF SHIFTS
      BNE      1$, ;DO IT AGAIN ?

;NOW POKE CLK TO SEE DNA
      BIS      #CLK,@TXCSR ;POKE CLK
      MOV      #10000,RO ;EXPECTED
      MOV      @TXCSR,R1 ;ACTUAL
      BIC      #77777,R1 ;SAVE DNA ONLY
      CMP      RO,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:SIX PLUS PARITY
; PARITY:ODDPAR
; CHARACTER:25
;*****
```



```

2519 011100 000004          TST23: SCOPE
2520 011102 052777 000400 170616  BIS      #MRESET,@TXCSR ;MASTER RESET
2521 011110 012777 030000 170604  MOV      #SYNINT,@PARCSR ;SET THE MODE
2522 011116 052777 000400 170602  BIS      #MRESET,@TXCSR ;MASTER RESET
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
011124 012777 004020 170574  MOV      #MINT!SEND,@TXCSR
;SET MAINTENANCE MODE & SEND
;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
MOV      #SYNINT!SIX!ODD@PAR!26,@PARCSR
MOV      TXCSR,R3 ;SET UP FOR ERROR MSG
MOVB     #25,@TXDBUF ;LOAD DATA CHAR
MOV      #25,STMP1 ;TO BE SHIFTED CHAR
MOV      #7,SHIFT ;# OF SHIFTS
;POKE CLK TO GET INTO SYNCHRONIZATION
BIS      #CLK,@TXCSR ;POKE CLK UP
BIC      #CLK,@TXCSR ;POKE CLK DOWN
1$:
CLF      RO
ROR      STMP1 ;FORCE CARRY
BCC     2$, ;BR IF CARRY CLR
BIS      #BITW,RO ;EQUIV OF BITW
2$:
BIS      #CLK,@TXCSR ;POKE CLK UP
BIC      #CLK,@TXCSR ;POKE CLK DOWN
MOV      @TXCSR,R1 ;ACTUAL
BIC      #075777,R1 ;SAVE BITW & DNA
CMP      RO,R1 ;COMPARE EXP VS ACT
BEQ     .+4
ERROR    3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
;BIT...ALSO CHECK DNA
;# OF SHIFTS
DEC     SHIFT
BNE     1$ ;DO IT AGAIN ?
;NOW POKE CLK TO SEE DNA
BIS      #CLK,@TXCSR ;POKE CLK
MOV      #100000,RO ;EXPECTED
MOV      @TXCSR,R1 ;ACTUAL
BIC      #77777,R1 ;SAVE DNA ONLY
CMP      RO,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
;LENGTH:SIX PLUS PARITY
;PARITY:EVEPAR
;CHARACTER:25
;*****
†ST24: SCOPE
BIS      #MRESET,@TXCSR ;MASTER RESET
MOV      #ISYMOD,@PARCSR ;SET THE MODE
    
```

INITIALIZE THE COMMON TAGS

```

2575 011324 052777 000400 170374      BIS      #MRESET,@TXCSR ;MASTER RESET
2576
2577 ;SET MAINTENANCE MODE & SEND
2578 ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
2579 011332 012777 004020 170366      MOV      #MINT!SEND,@TXCSR
2580
2581 ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
2582 011340 012777 003426 170354      MOV      #ISYMOD!SIX!EVEPAR!26,@PARCSR
2583 011346 016703 170354      MOV      TXCSR,R3 ;SET UP FOR ERROR MSG
2584 011352 112777 000025 170352      MOV      #25,@TXDBUF ;LOAD DATA CHAR
2585 011360 012767 000652 170112      MOV      #652,$TMP1 ;TO BE SHIFTED CHAR
2586 011366 012767 000011 167526      MOV      #9,$SHIFT ;# OF SHIFTS
2587 ;POKE CLK TO GET INTO SYNCHRONIZATION
2588 011374 052777 020000 170324      BIS      #CLK,@TXCSR ;POKE CLK UP
2589 011402 042777 020000 170316      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2590 011410 005000
2591 011412 006067 170062      15:     CLR      R0
2592 011416 103002      ROR      $TMP1 ;FORCE CARRY
2593 011420 052700 002000      BCC      25 ;BR IF CARRY CLR
2594 011424
2595 011424 052777 020000 170274      25:     BIS      #CLK,@TXCSR ;POKE CLK UP
2596 011432 042777 020000 170266      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2597 011440 017701 170262      MOV      @TXCSR,R1 ;ACTUAL
2598 011444 042701 075777      BIC      #075777,R1 ;SAVE BITW & DNA
2599 011450 020001      CMP      R0,R1 ;COMPARE EXP VS ACT
2600 011452 001401      BEQ      +4
2601 011454 104003      ERROR   3 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
2602 ;BIT...ALSO CHECK DNA
2603 011456 005367 167440      DEC      $SHIFT ;# OF SHIFTS
2604 011462 001352      BNE      15 ;DO IT AGAIN ?
2605 ;NOW POKE CLK TO SEE DNA
2606 011464 052777 020000 170234      BIS      #CLK,@TXCSR ;POKE CLK
2607 011472 012700 000000      MOV      #0,R0 ;EXPECTED
2608 011476 017701 170224      MOV      @TXCSR,R1 ;ACTUAL
2609 011502 042701 077777      BIC      #77777,R1 ;SAVE DNA ONLY
2610 011506 020001      CMP      R0,R1 ;COMPARE EXP VS ACT
2611 011510 001401      BEQ      +4
2612 011512 104003      ERROR   3 ;DNA SHOULD BE SET
2613 ;IF DNA DID NOT SET
2614 ;CHECK WORD LENGTH SELECT LOGIC
2615
2616 ;: THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
2617 ;: OF THE TRANSMITTER SECTION.
2618 ;: IT ALSO CHECKS DNA TIMING
2619 ;: MODE:ISYMOD
2620 ;: LENGTH:SIX PLUS PARITY
2621 ;: PARITY:ODDPAR
2622 ;: CHARACTER:25
2623
2624 ;:*****
2625 011514 000004      †ST25: SCOPE
2626 011516 052777 000400 170202      BIS      #MRESET,@TXCSR ;MASTER RESET
2627 011524 012777 000000 170170      MOV      #ISYMOD,@PARCSR ;SET THE MODE
2628 011532 052777 000400 170166      BIS      #MRESET,@TXCSR ;MASTER RESET
2629
2630 ;SET MAINTENANCE MODE & SEND
  
```



```

2631                                     ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
2632 011540 012777 004020 170160      MOV      #MINT!SEND,@TXCSR
2633
2634                                     ;SET MODE, # OF BITS, PARITY SENSE, & LOAD SYNC REG
2635 011546 012777 003026 170146      MOV      #ISYNOD!SIX!ODDPAR!26,@PARCSR
2636 011554 016703 170146              MOV      TXCSR,R3                ;SET UP FOR ERROR MSG
2637 011560 112777 000025 170144      MOV      #25,@TXDBUF           ;LOAD DATA CHAR
2638 011566 012767 000452 167704      MOV      #452,$TMP1           ;TO BE SHIFTED CHAR
2639 011574 012767 000011 167320      MOV      #9,$SHIFT            ;# OF SHIFTS
2640                                     ;POKE CLK TO GET INTO SYNCHRONIZATION
2641 011602 052777 020000 170116      BIS      @CLK,@TXCSR           ;POKE CLK UP
2642 011610 042777 020000 170110      BIC      @CLK,@TXCSR           ;POKE CLK DOWN
2643 011616 005000                      CLR
2644 011620 006067 167654              ROR      $TMP1                ;FORCE CARRY
2645 011624 103002                      BCC      2$                   ;BR IF CARRY CLR
2646 011626 052700 002000              BIS      @BITW,R0              ;EQUIV OF BITW
2647 011632
2648 011632 052777 020000 170066      BIS      @CLK,@TXCSR           ;POKE CLK UP
2649 011640 042777 020000 170060      BIC      @CLK,@TXCSR           ;POKE CLK DOWN
2650 011646 017701 170054              MOV      @TXCSR,R1            ;ACTUAL
2651 011652 042701 075777              BIC      #075777,R1           ;SAVE BITW & DNA
2652 011656 020001                      CMP      R0,R1                ;COMPARE EXP VS ACT
2653 011660 001401                      BEQ      +4
2654 011662 104003                      ERROR    3                    ;BIT WINDOW DID NOT MATCH ACTUAL DATA
2655                                     ;BIT... ALSO CHECK DNA
2656 011664 005367 167232              DEC      $SHIFT                ;# OF SHIFTS
2657 011670 001352                      BNE     1$                    ;DO IT AGAIN ?
2658                                     ;NOW POKE CLK TO SEE DNA
2659 011672 052777 020000 170026      BIS      @CLK,@TXCSR           ;POKE CLK
2660 011700 012700 000000              MOV      #0,R0                ;EXPECTED
2661 011704 017701 170016              MOV      @TXCSR,R1            ;ACTUAL
2662 011710 042701 077777              BIC      #77777,R1            ;SAVE DNA ONLY
2663 011714 020001                      CMP      R0,R1                ;COMPARE EXP VS ACT
2664 011716 001401                      BEQ      +4
2665 011720 104003                      ERROR    3                    ;DNA SHOULD BE SET
2666                                     ;IF DNA DID NOT SET
2667                                     ;CHECK WORD LENGTH SELECT LOGIC
2668
2669
2670
2671                                     ;: THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
2672                                     ;: OF THE TRANSMITTER SECTION.
2673                                     ;: IT ALSO CHECKS DNA TIMING
2674                                     ;: MODE:SYNINT
2675                                     ;: LENGTH:SEVEN PLUS PARITY
2676                                     ;: PARITY:EVEPAR
2677                                     ;: CHARACTER:125
2678
2679                                     ;:*****
2680 011722 000004                      †ST26: SCOPE
2681 011724 052777 000400 167774      BIS      #MRESET,@TXCSR       ;MASTER RESET
2682 011732 012777 030000 167762      MOV      #SYNINT,@PARCSR      ;SET THE MODE
2683 011740 052777 000400 167760      BIS      #MRESET,@TXCSR       ;MASTER RESET
2684
2685                                     ;SET MAINTENANCE MODE & SEND
2686                                     ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
    
```

```

2687 011746 012777 004020 167752      MOV      #MINT!SEND,@TXCSR
2688
2689                                     ;SET MODE # OF BITS,PARITY SENSE & LOAD SYNC REG
2690 011754 012777 035426 167740      MOV      #SYNINT!SEVEN!EVEPAR!25,@PARCSR
2691 011762 016703 167740      MOV      TXCSR,R3      ;SET UP FOR ERROR MSG
2692 011766 112777 000125 167736      MOV      #125,@TXDBUF  ;LOAD DATA CHAR
2693 011774 012767 000125 167476      MOV      #125,STMP1    ;TO BE SHIFTED CHAR
2694 012002 012767 000010 167112      MOV      #8,SHIFT     ;# OF SHIFTS
2695
2696                                     ;POKE CLK TO GET INTO SYNCHRONIZATION
2697 012010 052777 020000 167710      BIS      #CLK,@TXCSR   ;POKE CLK UP
2698 012016 042777 020000 167702      BIC      #CLK,@TXCSR   ;POKE CLK DOWN
2699 012024 005000
2700 012026 006067 167446      1$:      CLR      R0
2701 012032 103002
2702 012034 052700 002000      ROR      STMP1        ;FORCE CARRY
2703 012040                                     BCC      2$,          ;BR IF CARRY CLR
2704 012040 052777 020000 167660      BIS      #CLK,@TXCSR   ;POKE CLK UP
2705 012046 042777 020000 167652      BIC      #CLK,@TXCSR   ;POKE CLK DOWN
2706 012054 017701 167646      MOV      @TXCSR,R1     ;ACTUAL
2707 012060 042701 075777      BIC      #075777,R1    ;SAVE BITW & DNA
2708 012064 020001      CMP      R0,R1        ;COMPARE EXP VS ACT
2709 012066 001401      BEQ      +4
2710 012070 104003      ERROR    3            ;BIT WINDOW DID NOT MATCH ACTUAL DATA
2711                                     ;BIT...ALSO CHECK DNA
2712 012072 005367 167024      DEC      SHIFT        ;# OF SHIFTS
2713 012076 001352      BNE      1$,          ;DO IT AGAIN ?
2714                                     ;NOW POKE CLK TO SEE DNA
2715 012100 052777 020000 167620      BIS      #CLK,@TXCSR   ;POKE CLK
2716 012106 012700 100000      MOV      #100000,R0    ;EXPECTED
2717 012112 017701 167610      MOV      @TXCSR,R1     ;ACTUAL
2718 012116 042701 077777      BIC      #77777,R1     ;SAVE DNA ONLY
2719 012122 020001      CMP      R0,R1        ;COMPARE EXP VS ACT
2720 012126 104003      BEQ      +4
2721      ERROR    3            ;DNA SHOULD BE SET
2722                                     ;IF DNA DID NOT SET
2723                                     ;CHECK WORD LENGTH SELECT LOGIC
2724
2725                                     ;: THIS TEST VERIFYS CHARACTER PLUS PARITY GENERATION
2726                                     ;: OF THE TRANSMITTER SECTION.
2727                                     ;: IT ALSO CHECKS DNA TIMING
2728                                     ;: MODE:SYNINT
2729                                     ;: LENGTH:SEVEN PLUS PARITY
2730                                     ;: PARITY:ODDPAR
2731                                     ;: CHARACTER:125
2732                                     ;:*****
2733 012130 000004      †ST27: SCOPE
2734 012132 052777 000400 167566      BIS      #MRESET,@TXCSR ;MASTER RESET
2735 012140 012777 030000 167554      MOV      #SYNINT,@PARCSR ;SET THE MODE
2736 012146 052777 000400 167552      BIS      #MRESET,@TXCSR ;MASTER RESET
2737
2738                                     ;SET MAINTENANCE MODE & SEND
2739                                     ;NOTE:BIT WINDOW&CLK ARE CLEARED (MTDATA=0)
2740 012154 012777 004020 167544      MOV      #MINT!SEND,@TXCSR
2741
2742                                     ;SET MODE,# OF BITS,PARITY SENSE,& LOAD SYNC REG
    
```



```

2743 012162 012777 035026 167532      MOV      #SYNINT!SEVEN!ODDPAR!26,@PARCSR
2744 012170 016703 167532              MOV      TXCSR,R3          ;SET UP FOR ERROR MSG
2745 012174 112777 000125 167530      MOVVB   #125,@TXDBUF      ;LOAD DATA CHAR
2746 012202 012767 000325 167270      MOV      #325,@STMP1      ;TO BE SHIFTED CHAR
2747 012210 012767 000010 166704      MOV      #8,SHIFT         ;# OF SHIFTS
2748                                ;POKE CLK TO GET INTO SYNCHRONIZATION
2749 012216 052777 020000 167502      BIS      @CLK,@TXCSR      ;POKE CLK UP
2750 012224 042777 020000 167474      BIC      @CLK,@TXCSR      ;POKE CLK DOWN
2751 012232 005000                    1S:     CLR      R0
2752 012234 006067 167240              ROR      @STMP1           ;FORCE CARRY
2753 012240 103002                    BCC     2S               ;BR IF CARRY CLR
2754 012242 052700 002000              BIS      @BITW,R0         ;EQUIV OF BITW
2755 012246                                2S:
2756 012246 052777 020000 167452      BIS      @CLK,@TXCSR      ;POKE CLK UP
2757 012254 042777 020000 167444      BIC      @CLK,@TXCSR      ;POKE CLK DOWN
2758 012262 017701 167440              MOV      @TXCSR,R1        ;ACTUAL
2759 012266 042701 075777              BIC      #075777,R1       ;SAVE BITW & DNA
2760 012272 020001                    CMP     R0,R1             ;COMPARE EXP VS ACT
2761 012274 001401                    BEQ     +4
2762 012276 104003                    ERROR   3                 ;BIT WINDOW DID NOT MATCH ACTUAL DATA
2763                                ;BIT...ALSO CHECK DNA
2764 012300 005367 166616              DEC     SHIFT             ;# OF SHIFTS
2765 012304 001352                    BNE     1S               ;DO IT AGAIN ?
2766                                ;NOW POKE CLK TO SEE DNA
2767 012306 052777 020000 167412      BIS      @CLK,@TXCSR      ;POKE CLK
2768 012314 012700 100000              MOV      #100000,R0       ;EXPECTED
2769 012320 017701 167402              MOV      @TXCSR,R1        ;ACTUAL
2770 012324 042701 077777              BIC      #77777,R1        ;SAVE DNA ONLY
2771 012330 020001                    CMP     R0,R1             ;COMPARE EXP VS ACT
2772 012332 001401                    BEQ     +4
2773 012334 104003                    ERROR   3                 ;DNA SHOULD BE SET
2774                                ;IF DNA DID NOT SET
2775                                ;CHECK WORD LENGTH SELECT LOGIC
2776
2777
2778
2779                                ;END OF PASS
2780                                ;TYPE NAME OF TEST
2781                                ;UPDATE PASS COUNT
2782                                ;CHECK FOR EXIT TO ACT-11
2783                                ;RESTART TEST
2784
2785 012336 000004                    .EOP:  SCOPE
2786 012340 004767 000344              JSR     PC,CKSWR
2787 012344 104401                                ;TYPE NAME OF TEST
2788 012346 015500                    MEPASS
2789 012350 104413 012602              CONVRT  ,OUTCRY
2790 012354 104401 015317              TYPE   DEVICE
2791 012360 105767 166566              TSTB   MULTD             ;ARE YOU RUNNING MULTIPLE DEVICES ?
2792 012364 001511                    BEQ     CCC               ;NO JUMP AROUND
2793 012366 005767 166574              TST    ACTREG            ;ARE ANY DEVICES ACTIVE ?
2794 012372 001007                    BNE     RUNIT             ;YES
2795 012374 104401 015331              TYPE   MCON              ;NO
2796 012400 016700 166562              MOV     @ACTREG,R0        ;DISPLAY ACTREG
2797 012404 000000                    HALT    ;SELECT SOMETHING TO RUN @ ACTREG:
2798                                ;SELECT SWITCHES & HIT CONTINUE (PUT SW00 =1)

```





```

2855
2856 012710 005737 000042          CKSWR: TST      2#42
2857 012714 001040                BNE      OUT
2858 012716 022767 000176 166514  CMP      #SWREG,SWR      ; SOFTWARE SWR PRESENT?
2859 012724 001034                BNE      OUT              ; NO--LEAVE
2860 012726 105777 166512          TSTB    2$TKS           ; CHECK TTY READY
2861 012732 100031                BPL      OUT              ; NO--LEAVE
2862 012734 017767 166506 000422  MOV     2$TKB,MSG       ; GET CHARACTER
2863 012742 042767 177600 000414  BIC     #177600,.MSG    ; STRIP JUNK
2864 012750 122767 000007 000406  CMPB   #7,MSG          ; IS IT (<↑G) ?
2865 012756 001017                BNE      OUT              ; NO
2866 012760 104401 016105          TYPE    MCNTG
2867 012764 005137 013024          CNTLU: COM    2#RDSW
2868 012770 104401 016115          TYPE    ,MMSWR
2869 012774 104413          CONVRT
2870 012776 013026          SWREGL
2871 013000 104406 016126          INSTR,MMNEW
2872 013004 104410          PARAM
2873 013006 000000          0
2874 013010 177777          177777
2875 013012 000176          SWREG
2876 013014 000000 001          .BYTE  0,1
2877 013016 005037 013024          OUT:   CLR      2#RDSW
2878 013022 000207          RTS      PC
2879 013024 000000          RDSW:   .WORD   0
2880 013026 000001          SWREGL: 1
2881 013030 000006 002          .BYTE  6,2
2882 013032 000176          SWREG
2883
2884 013034 000005          5
2885
2886          ;CHECK FOR FREEZE ON CURRENT DATA
2887
2888 013036 004767 177646          .SCOP1: JSR     PC,CKSWR
2889 013042 032777 001000 166370  BIT     #SW09,2$SWR
2890 013050 001402          BEQ     1$
2891 013052 016716 166032          MOV     LOCK,(SP)
2892 013056 000002          1$:    RTI
2893          .SBTTL  TYPE ROUTINE
2894
2895          ;*****
2896          ;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2897          ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2898          ;NOTE1:    $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2899          ;NOTE2:    $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2900          ;NOTE3:    $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2901          ;
2902          ;CALL:
2903          ;1) USING A TRAP INSTRUCTION
2904          ;    TYPE    ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2905          ;OR
2906          ;    TYPE    MESADR
2907          ;
2908          ;
2909
2910 013060 105767 166373          $TYPE: TSTB    2$PFLG      ;; IS THERE A TERMINAL?

```

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 60  
 DZDUTA.M11 13-OCT-76 08:39 TYPE ROUTINE

2911	013064	100002			BPL	1\$	:: BR IF YES
2912	013066	000000			HALT		:: HALT HERE IF NO TERMINAL
2913	013070	000430			BR	3\$	:: LEAVE
2914	013072	010046			MOV	RO, -(SP)	:: SAVE RO
2915	013074	017600	000002		MOV	22(SP), RO	:: GET ADDRESS OF ASCIZ STRING
2916	013100	122767	000001	166440	CMPB	#APTENV, \$ENV	:: RUNNING IN APT MODE
2917	013106	001011			BNE	62\$	:: NO, GO CHECK FOR APT CONSOLE
2918	013110	132767	000100	166431	BITB	#APTSPOOL, \$ENVM	:: SPOOL MESSAGE TO APT
2919	013116	001405			BEQ	62\$	:: NO, GO CHECK FOR CONSOLE
2920	013120	010067	000004		MOV	RO, 61\$	:: SETUP MESSAGE ADDRESS FOR APT
2921	013124	004767	000006		JSR	PC, \$ATY3	:: SPOOL MESSAGE TO APT
2922	013130	000000			.WORD	0	:: MESSAGE ADDRESS
2923	013132	132767	000040	166407	BITB	#APTCSUP, \$ENVM	:: APT CONSOLE SUPPRESSED
2924	013140	001003			BNE	60\$	:: YES, SKIP TYPE OUT
2925	013142	112046			MOV	(RO)+, -(SP)	:: PUSH CHARACTER TO BE TYPED ONTO STACK
2926	013144	001005			BNE	4\$	:: BR IF IT ISN'T THE TERMINATOR
2927	013146	005726			TST	(SP)+	:: IF TERMINATOR POP IT OFF THE STACK
2928	013150	012600			MOV	(SP)+, RO	:: RESTORE RO
2929	013152	062716	000002		ADD	#2, (SP)	:: ADJUST RETURN PC
2930	013156	000002			RTI		:: RETURN
2931	013160	122716	000011		CMPB	#HT, (SP)	:: BRANCH IF <HT>
2932	013164	001430			BEQ	8\$	
2933	013166	122716	000200		CMPB	#CRLF, (SP)	:: BRANCH IF NOT <CRLF>
2934	013172	001006			BNE	5\$	
2935	013174	005726			TST	(SP)+	:: POP <CR><LF> EQUIV
2936	013176	104401			TYPE		:: TYPE A CR AND LF
2937	013200	001523			\$CRLF		
2938	013202	105067	000130		CLRB	\$CHARCNT	:: CLEAR CHARACTER COUNT
2939	013206	000755			BR	2\$	:: GET NEXT CHARACTER
2940	013210	004767	000056		JSR	PC, \$TYPEC	:: GO TYPE THIS CHARACTER
2941	013214	126726	166236		CMPB	\$FILLC, (SP)+	:: IS IT TIME FOR FILLER CHARS.?
2942	013220	001350			BNE	2\$	:: IF NO GO GET NEXT CHAR.
2943	013222	016746	166226		MOV	\$NULL, -(SP)	:: GET # OF FILLER CHARS. NEEDED
2944							:: AND THE NULL CHAR.
2945	013226	105366	000001		DECB	1(SP)	:: DOES A NULL NEED TO BE TYPED?
2946	013232	002770			BLT	6\$	:: BR IF NO--GO POP THE NULL OFF OF STACK
2947	013234	004767	000032		JSR	PC, \$TYPEC	:: GO TYPE A NULL
2948	013240	105367	000072		DECB	\$CHARCNT	:: DO NOT COUNT AS A COUNT
2949	013244	000770			BR	7\$	:: LOOP
2950							
2951							
2952							
2953	013246	112716	000040		MOV	#' (SP)	:: REPLACE TAB WITH SPACE
2954	013252	004767	000014		JSR	PC, \$TYPEC	:: TYPE A SPACE
2955	013256	132767	000007	000052	BITB	#7, \$CHARCNT	:: BRANCH IF NOT AT
2956	013264	001372			BNE	9\$	:: TAB STOP
2957	013266	005726			TST	(SP)+	:: POP SPACE OFF STACK
2958	013270	000724			BR	2\$	:: GET NEXT CHARACTER
2959	013272	105777	166152		\$TYPEC:	\$STPS	:: WAIT UNTIL PRINTER IS READY
2960	013276	100375			BPL	\$TYPEC	
2961	013300	116677	000002	166144	MOV	2(SP), \$STPB	:: LOAD CHAR TO BE TYPED INTO DATA REG.
2962	013306	122766	000015	000002	CMPB	#CR, 2(SP)	:: IS CHARACTER A CARRIAGE RETURN?
2963	013314	001003			BNE	1\$	:: BRANCH IF NO
2964	013316	105067	000014		CLRB	\$CHARCNT	:: YES--CLEAR CHARACTER COUNT
2965	013322	000406			BR	\$TYPEX	:: EXIT
2966	013324	122766	000012	000002	CMPB	#LF, 2(SP)	:: IS CHARACTER A LINE FEED?



DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 61  
 DZDUTA.M11 13-OCT-76 08:39 TYPE ROUTINE

```

2967 013332 001402          BEQ      $TYPEX      ;; BRANCH IF YES
2968 013334 105227          INCB     (PC)+        ;; COUNT THE CHARACTER
2969 013336 000000          $CHARCNT: WORD 0      ;; CHARACTER COUNT STORAGE
2970 013340 000207          $TYPEX: RTS      PC
2971
2972
2973          ;ASCII STRING INPUT ROUTINE
2974
2975 013342 017667 000000 000014 .INSTR: MOV      2(SP),MSG ;PICK UP MESSAGE
2976 013350 062716 000002          ADD     #2,(SP)      ;JUMP AROUND MESSAGE FOR RTI
2977 013354 105767 166166          TSTB   $ENV         ;APT CONTROL
2978 013360 001036          BNE     INSTR2       ;YES NO TYPE
2979 013362 104401          .INST1: TYPE
2980 013364 000000          .MSG:  0
2981 013366 012704 016140          MOV     #INBUF,R4    ;GET STARTING LOC OF INBUF
2982 013372 012703 000007          MOV     #7,R3        ;MAX # OF CHARS
2983 013376 105777 166042          15:    TSTB   $STKS   ;TTY FLAG
2984 013402 100375          BPL     15
2985 013404 117714 166036          MOVB   $STKB,(R4)    ;TAKE CHAR
2986 013410 142714 000200          BICB   #200,(R4)    ;STRIP
2987 013414 121427 000025          CMPB   (R4),#25     ;IS IT <↑G>
2988 013420 001760          BEQ     .INST1
2989 013422 122427 000015          CMPB   (R4)+,#15    ;CHECK FOR CR
2990 013426 001413          BEQ     INSTR2
2991 013430 105777 166014          25:    TSTB   $STPS   ;TEST FLAG
2992 013434 100375          BPL     25
2993 013436 117777 166004 166006          MOVB   $STKB,$STPB  ;ECHO CHARACTER
2994 013444 005303          DEC     R3           ;DID YOU TYPE TOO MANY CHARS ?
2995 013446 001353          BNE     15
2996 013450 104401          .INSTE: TYPE
2997 013452 015425          MQM    ;?
2998 013454 000742          BR     .INST1 ;RETRY
2999 013456 000002          INSTR2: RTI
3000
3001          ;CONVERT ASCII STRING TO OCTAL
3002
3003 013460 011605          .PARAM: MOV     (SP),R5 ;PUT CONTENTS OF SP INTO R5
3004 013462 012567 000162          MOV     (R5)+,LOLIM  ;PUT LOW LIMIT INTO LOLIM
3005 013466 012567 000160          MOV     (R5)+,HILIM  ;PUT HIGH LIMIT INTO HILIM
3006 013472 012567 000156          MOV     (R5)+,DEVADR  ;PUT STORE LOC INTO DEVADR
3007 013476 112567 000154          MOVB   (R5)+,LOBITS  ;PUT MASK INTO LOBITS
3008 013502 112567 000151          MOVB   (R5)+,ADRCNT  ;PUT COUNT INTO ADRCNT
3009 013506 010516          MOV     R5,(SP) ;RESTORE RETURN ADDR ON STACK FOR RTI
3010 013510 005005          PARAM1: CLR    R5
3011 013512 012704 016140          MOV     #INBUF,R4
3012 013516 122714 000015          CMPB   #15,(R4)     ;CR ?
3013 013522 001420          BEQ     PARERR ;YOU TYPED CR TOO SOON !
3014 013524 121427 000060          15:    CMPB   (R4),#60 ;LOW LIMIT ASCII 0
3015 013530 002415          BLT    PARERR
3016 013532 121427 000067          CMPB   (R4),#67     ;HIGH LIMIT ASCII 7
3017 013536 003012          BGT    PARERR
3018 013540 142714 000060          BICB   #60,(R4)    ;CONVERT TO OCTAL
3019 013544 152405          BISB   (R4)+,R5    ;STORE AWAY ITS AN OK CHAR
3020 013546 122714 000015          CMPB   #15,(R4)   ;CR ?
3021 013552 001414          BEQ     LIMITS ;NOW CHECK FOR HIGH & LOW LIMIT CONDS
3022 013554 006305          ASL    R5 ;ALLOCATE ROOM FOR NEXT CHAR

```

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 62  
 DZDUTA.M11 13-OCT-76 08:39 TYPE ROUTINE

3023	013556	006305			ASL	R5	
3024	013560	006305			ASL	R5	
3025	013562	000760			BR	15	
3026	013564	122714	000015		PARERR: CMPB	#15 (R4)	;CR?
3027	013570	001003			BNE	120\$	
3028	013572	005737	013024		TST	2#RDSW	;CK SWR USED
3029	013576	001023			BNE	PARTI	
3030	013600	104407			120\$: INSTER	:RETRY	
3031	013602	000742			BR	PARAM1	
3032							
3033							
3034							
3035	013604	020567	000042		LIMITS: CMP	R5, HILIM	
3036	013610	101365			BHI	PARERR	;THE # IS TOO HIGH
3037	013612	020567	000032		CMP	R5, LOLIM	
3038	013616	103762			BLO	PARERR	;THE # IS TOO LOW
3039	013620	136705	000032		BITB	LOBITS, R5	;TEST BY MASKINGTHE #
3040	013624	001357			BNE	PARERR	
3041							
3042							
3043							
3044	013626	016704	000022				
3045	013632	010524			15: MOV	DEVADR, R4	;GET STARTING ADDR OF
3046	013634	062705	000002		MOV	R5, (R4)+	;STORE AT THIS ADDR
3047	013640	105367	000013		ADD	#2, R5	
3048	013644	001372			DECB	ADRCNT	;HOW MANY TIMES + 2 ?
3049	013646	000002			BNE	15	
3050	013650	000000			PARTI: RTI		
3051	013652	000000			LOLIM: 0		
3052	013654	000000			HILIM: 0		
3053	013656	000000			DEVADR: 0		
3054		013657			LOBITS: 0		
3055					ADRCNT=LOBITS+1		
3056							
3057							
3058	013660	016667	000004	165240	.SAV05: MOV	4(SP), SAVPC	
3059							
3060							
3061							
3062	013666	010567	165602		SV05: MOV	R5, \$REG5	
3063	013672	010467	165574		MOV	R4, \$REG4	
3064	013676	010367	165566		MOV	R3, \$REG3	
3065	013702	010267	165560		MOV	R2, \$REG2	
3066	013706	010167	165552		MOV	R1, \$REG1	
3067	013712	010067	165544		MOV	RO, \$REG0	
3068	013716	000002			RTI		
3069							
3070							
3071							
3072	013720	016700	165536		.RES05: MOV	\$REG0, RO	
3073	013724	016701	165534		MOV	\$REG1, R1	
3074	013730	016702	165532		MOV	\$REG2, R2	
3075	013734	016703	165530		MOV	\$REG3, R3	
3076	013740	016704	165526		MOV	\$REG4, R4	
3077	013744	016705	165524		MOV	\$REG5, R5	
3078	013750	000002			RTI		

;RESTORE RO-R5



```

3079
3080 ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
3081
3082 013752 104401 .CONVR: TYPE
3083 013754 015431 MCRLF ;CR LF
3084 013756 017601 000000 MOV 2(SP),R1 ;PICK UP DATA POINTER
3085 013762 062716 000002 ADD #2(SP);SET UP SP FOR RTI
3086 013766 012167 000130 MOV (R1)+,WRDCNT ;PICK UP # OF WORDS FROM TABLE
3087 013772 112167 000126 1$: MOVB (R1)+,CHRCNT ;PICK UP # OF CHARS FROM TABLE
3088 013776 112167 000123 MOVB (R1)+,SPACNT ;PICK UP # OF SPACES FROM TABLE
3089 014002 013167 000120 MOV 2(R1)+,BINWRD ;PICK UP ADDRESS OF MSG
3090 ;FROM TABLE
3091 014006 016704 000114 2$: MOV BINWRD,R4 ;SAVE
3092 014012 116705 000106 MOVB CHRCNT,R5 ;SAVE
3093 014016 012700 016202 MOV #TEMP,R0 ;STARTING ADDRESS OF TEMP BLOCK
3094 014022 010403 3$: MOV R4,R3 ;SAVE
3095 014024 042703 177770 BIC #177770,R3 ;CLR OUT UPPER BITS .. SAVE CHAR
3096 014030 062703 000260 ADD #260,R3 ;CONVERT TO ASCII
3097 014034 110320 MOVB R3,(R0)+ ;STORE AWAY
3098 014036 006204 ASR R4 ;SHIFT FOR NEXT #
3099 014040 006204 ASR R4 ;DITTO
3100 014042 006204 ASR R4 ;DITTO
3101 014044 005305 DEC R5 ;DEC CHAR COUNT
3102 014046 001365 BNE 3$ ;DO IT AGAIN ?
3103 014050 012703 016244 MOV #MDATA,R3 ;STARTING ADDRESS OF MDATA BLOCK
3104 014054 114023 4$: MOVB -(R0),(R3)+ ;REVERSE THE ORDER OF NUMBERS
3105 014056 105367 000042 DECB CHRCNT ;DEC CHAR COUNT
3106 014062 001374 BNE 4$ ;DO IT AGAIN ?
3107 014064 105767 000035 TSTB SPACNT ;HOW MANY SPACES ?
3108 014070 001405 BEQ 6$ ;TYPE # IF BR =0
3109 014072 112723 5$: MOVB #240,(R3)+ ;"SPACE" IN ASCII
3110 014076 105367 000023 DECB SPACNT ;DEC # OF SPACE COUNT
3111 014102 001373 BNE 5$ ;DO IT AGAIN ?
3112 014104 105013 6$: CLRB (R3) ;INSERT "0" FOR TTY OUTPUT ROUTINE
3113 014106 104401 TYPE
3114 014110 016244 MDATA ;THIS MESSAGE
3115 014112 005367 000004 DEC WRDCNT ;HOW MANY #'S ?
3116 014116 001325 BNE 1$ ;DO THIS ROUTINE AGAIN IF NOT EQUAL TO 0
3117 014120 000002 RTI ;RETURN TO PROGRAM
3118 014122 000000 WRDCNT: 0
3119 014124 000000 CHRCNT: 0
3120 014125 014125 SPACNT=CHRCNT+1
3121 014126 000000 BINWRD: 0
3122
3123 ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
3124 ;BUFFER TO THE CHARACTERS "N" AND "Y"
3125 ;IF THE CHARACTER IS "N" CLEAR THE FLAG
3126 ;IF THE CHARACTER IS "Y" SET THE FLAG
3127
3128 014130 017605 000000 .SETFLG: MOV 2(SP),R5
3129 014134 122767 000116 001776 CMPB #'N,INBUF ;IS IT "N" ?
3130 014142 001002 BNE 1$
3131 014144 105015 CLRB (R5) ;000
3132 014146 000406 BR 2$
3133 014150 122767 000131 001762 1$: CMPB #'Y,INBUF ;IS IT "Y" ?
3134 014156 001005 BNE 3$
    
```

```

3135 014160 112715 177777      MOVB    #-1,(R5)      ;377
3136 014164 062716 000002      2$:    ADD     #2,(SP)
3137 014170 000002      RTI
3138 014172 104407      3$:    INSTER ;RETRY
3139 014174 000755      BR     .SETFLG
3140      .SBTTL  ERROR HANDLER ROUTINE
3141
3142      ;*****
3143      ;THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3144      ;SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3145      ;AND GO TO SAVIT ON ERROR
3146      ;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3147      ;*SW15=1      HALT ON ERROR
3148      ;*SW13=1      INHIBIT ERROR TYPEOUTS
3149      ;*SW10=1      BELL ON ERROR
3150      ;*SM09=1      LOOP ON ERROR
3151      ;*CALL
3152      ;*      ERROR  N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3153
3154      SERROR:
3155 014176 105267 165201      7$:    INCB    $ERFLG      ;; SET THE ERROR FLAG
3156 014202 001775      BEQ     7$              ;; DON'T LET THE FLAG GO TO ZERO
3157 014204 016777 165172 165230      MOV     $STNM,$DISPLAY  ;; DISPLAY TEST NUMBER AND ERROR FLAG
3158 014212 032777 002000 165220      BIT     #BIT10,$SWR     ;; BELL ON ERROR?
3159 014220 001402      BEQ     1$              ;; NO - SKIP
3160 014222 104401 001516      TYPE    $BELL          ;; RING BELL
3161 014226 005267 165160      1$:    INC     $ERTTL     ;; COUNT THE NUMBER OF ERRORS
3162 014232 011667 165160      MOV     (SP),$ERRPC    ;; GET ADDRESS OF ERROR INSTRUCTION
3163 014236 162767 000002 165152      SUB     #2,$ERRPC
3164 014244 117767 165146 165142      MOVB   $ERRPC,$ITEMB  ;; STRIP AND SAVE THE ERROR ITEM CODE
3165 014252 032777 020000 165160      BIT     #BIT13,$SWR   ;; SKIP TYPEOUT IF SET
3166 014260 001004      BNE     20$            ;; SKIP TYPEOUTS
3167 014262 004767 000072      JSR    PC,SAVIT       ;; GO TO USER ERROR ROUTINE
3168 014266 104401 001523      TYPE    ,SCLF
3169 014272
3170 014272 122767 000001 165246      20$:   CMPB   #APTENV,$ENV  ;; RUNNING IN APT MODE
3171 014300 001007      BNE     2$              ;; NO SKIP APT ERROR REPORT
3172 014302 116767 165106 000004      MOVB   $ITEMB,21$     ;; SET ITEM NUMBER AS ERROR NUMBER
3173 014310 004767 000016      JSR    PC,$ATY4      ;; REPORT FATAL ERROR TO APT
3174 014314      .BYTE  0
3175 014315      .BYTE  0
3176 014316 000777      21$:   .BYTE  0
3177 014320 005777 165114      22$:   BR     22$
3178 014324 100001      2$:    TST    $SWR
3179 014326 000000      BPL    3$
3180 014330 032777 001000 165102      3$:    HALT
3181 014336 001402      BIT     #BIT09,$SWR   ;; LOOP ON ERROR SWITCH SET?
3182 014340 016716 165044      BEQ    4$              ;; BR IF NO
3183 014344 005767 165144      MOV     $LPERR,(SP)   ;; FUDGE RETURN FOR LOOPING
3184 014350 001402      4$:    TST    $ESCAPE    ;; CHECK FOR AN ESCAPE ADDRESS
3185 014352 016716 165136      BEQ    5$              ;; BR IF NONE
3186 014356      MOV     $ESCAPE,(SP) ;; FUDGE RETURN ADDRESS FOR ESCAPE
3187 014357 000002      5$:
3188 014360 010067 164544      RTI
3189 014364 010167 164542      SAVIT: MOV    R0,HLD0      ;; RETURN
3190 014370 010267 164540      MOV    R1,HLD1
      MOV    R2,HLD2
    
```



3191 014374 010367 164536  
 3192 014400 010467 164534  
 3193 014404 010567 164532  
 3194 014410 016767 164766 164526

MOV R3,HLDB3  
 MOV R4,HLDB4  
 MOV R5,HLDB5  
 MOV STSTNM,HLDB6

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

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

3203 014416  
 3204 014416 104401 001523  
 3205 014422 010046  
 3206 014424 005000  
 3207 014426 153700 001414  
 3208 014432 001004  
 3209  
 3210 014434 016746 164756  
 3211  
 3212 014440 104402  
 3213 014442 000426  
 3214 014444 005300  
 3215 014446 006300  
 3216 014450 006300  
 3217 014452 006300  
 3218 014454 062700 001652  
 3219 014460 012067 000004  
 3220 014464 001404  
 3221 014466 104401  
 3222 014470 000000  
 3223 014472 104401 001523  
 3224 014476 012067 000004  
 3225 014502 001404  
 3226 014504 104401  
 3227 014506 000000  
 3228 014510 104401 001523  
 3229 014514 011000  
 3230 014516 001004  
 3231 014520 012600  
 3232 014522 104401 001523  
 3233 014526 000207  
 3234 014530  
 3235 014530 013046  
 3236 014532 104402  
 3237 014534 005710  
 3238 014536 001770  
 3239 014540 104401 014546  
 3240 014544 000771  
 3241 014546 020040 000  
 3242 014552

SERRTYP:  
 TYPE SCRLF ;;"CARRIAGE RETURN" & "LINE FEED"  
 MOV RO,-(SP) ;;SAVE RO  
 CLR RO ;;PICKUP THE ITEM INDEX  
 BISB @#\$ITEMB,RO  
 BNE IS ;;IF ITEM NUMBER IS ZERO, JUST  
 ;;TYPE THE PC OF THE ERROR  
 MOV SERRPC,-(SP) ;;SAVE SERRPC FOR TYPEOUT  
 ;;ERROR ADDRESS  
 TYP0C ;;GO TYPE--OCTAL ASCII(ALL DIGITS)  
 BR 6\$ ;;GET OUT  
 IS: DEC RO ;;ADJUST THE INDEX SO THAT IT WILL  
 ;;WORK FOR THE ERROR TABLE  
 ASL RO  
 ASL RO  
 ASL RO  
 ADD #SERRTB,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 SCRLF ;;"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 SCRLF ;;"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 SCRLF ;;"CARRIAGE RETURN" & "LINE FEED"  
 7\$: RTS PC ;;RETURN  
 MOV @2(RO)+,-(SP) ;;SAVE @2(RO)+ FOR TYPEOUT  
 TYP0C ;;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

\*\*\*\*\*  
 \*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT

3243  
 3244  
 3245  
 3246

# M05

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 66  
 DZDUTA.M11 13-OCT-76 08:39 BINARY TO OCTAL (ASCII) AND TYPE

```

3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268 014552 017646 000000
3269 014556 116667 000001 000211
3270 014564 112667 000207
3271 014570 062716 000002
3272 014574 000406
3273 014576 112767 000001 000171
3274 014604 112767 000006 000165
3275 014612 112767 000005 000154
3276 014620 010346
3277 014622 010446
3278 014624 010546
3279 014626 116704 000145
3280 014632 005404
3281 014634 062704 000006
3282 014640 110467 000132
3283 014644 116704 000125
3284 014650 016605 000012
3285 014654 005003
3286 014656 006105
3287 014660 000404
3288 014662 006105
3289 014664 006105
3290 014666 006105
3291 014670 010503
3292 014672 006103
3293 014674 105367 000076
3294 014700 103016
3295 014702 042703 177770
3296 014706 001002
3297 014710 005704
3298 014712 001403
3299 014714 005204
3300 014716 052703 000060
3301 014722 052703 000040
3302 014726 110367 000040

```

```

;OCTAL (ASCII) NUMBER AND TYPE IT
;STYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
;CALL:
;   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
;   TYPOS   N              ;;CALL FOR TYPEOUT
;   .BYTE  N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
;   .BYTE  M              ;;M=1 OR 0
;                           ;;1=TYPE LEADING ZEROS
;                           ;;0=SUPPRESS LEADING ZEROS
;STYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
;STYPOS OR STYPOC
;CALL:
;   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
;   TYPON   N              ;;CALL FOR TYPEOUT
;STYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
;CALL:
;   MOV     NUM,-(SP)      ;;NUMBER TO BE TYPED
;   TYPOC   N              ;;CALL FOR TYPEOUT
;STYPOS: MOV     2(SP),-(SP)  ;;PICKUP THE MODE
;        MOV    1(SP),SOFILL  ;;LOAD ZERO FILL SWITCH
;        MOV    (SP)+,SOMODE+1 ;;NUMBER OF DIGITS TO TYPE
;        ADD    #2,(SP)      ;;ADJUST RETURN ADDRESS
;        BR    STYPON
;STYPOC: MOV    #1,SOFILL    ;;SET THE ZERO FILL SWITCH
;        MOV    #6,SOMODE+1  ;;SET FOR SIX(6) DIGITS
;STYPON: MOV    #5,SOCNT     ;;SET THE ITERATION COUNT
;        MOV    R3,-(SP)     ;;SAVE R3
;        MOV    R4,-(SP)     ;;SAVE R4
;        MOV    R5,-(SP)     ;;SAVE R5
;        MOV    SOMODE+1,R4  ;;GET THE NUMBER OF DIGITS TO TYPE
;        NEG    R4
;        ADD    #6,R4        ;;SUBTRACT IT FOR MAX. ALLOWED
;        MOV    R4,SOMODE    ;;SAVE IT FOR USE
;        MOV    SOFILL,R4    ;;GET THE ZERO FILL SWITCH
;        MOV    12(SP),R5    ;;PICKUP THE INPUT NUMBER
;        CLR    R3          ;;CLEAR THE OUTPUT WORD
;        ROL    R5          ;;ROTATE MSB INTO "C"
;        BR    3$          ;;GO DO MSB
;        ROL    R5          ;;FORM THIS DIGIT
;        ROL    R5
;        ROL    R5
;        MOV    R5,R3
;        ROL    R3          ;;GET LSB OF THIS DIGIT
;        DECB  SOMODE       ;;TYPE THIS DIGIT?
;        BPL   7$          ;;BR IF NO
;        BIC   #177770,R3  ;;GET RID OF JUNK
;        BNE  4$          ;;TEST FOR 0
;        TST  R4          ;;SUPPRESS THIS 0?
;        BEQ  5$          ;;BR IF YES
;        INC  R4          ;;DON'T SUPPRESS ANYMORE 0'S
;        BIS  #'0,R3      ;;MAKE THIS DIGIT ASCII
;        BIS  #' ,R3      ;;MAKE ASCII IF NOT ALREADY
;        MOV  R3,#$       ;;SAVE FOR TYPING

```



# N05

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 67  
 DZDUTA.M11 13-OCT-76 08:39 BINARY TO OCTAL (ASCII) AND TYPE

3303	014732	104401	014772		TYPE	BS	::GO TYPE THIS DIGIT
3304	014736	105367	000032	7S:	DECB	\$OCNT	::COUNT BY 1
3305	014742	003347			BGT	2S	::BR IF MORE TO DO
3306	014744	002402			BLT	6S	::BR IF DONE
3307	014746	005204			INC	R4	::INSURE LAST DIGIT ISN'T A BLANK
3308	014750	000744			BR	2S	::GO DO THE LAST DIGIT
3309	014752	012605		6S:	MOV	(SP)+,R5	::RESTORE R5
3310	014754	012604			MOV	(SP)+,R4	::RESTORE R4
3311	014756	012603			MOV	(SP)+,R3	::RESTORE R3
3312	014760	016666	000002 000004		MOV	2(SP),4(SP)	::SET THE STACK FOR RETURNING
3313	014766	012616			MOV	(SP)+,(SP)	
3314	014770	000002			RTI		::RETURN
3315	014772	000		8S:	.BYTE	0	::STORAGE FOR ASCII DIGIT
3316	014773	000			.BYTE	0	::TERMINATOR FOR TYPE ROUTINE
3317	014774	000		SOCNT:	.BYTE	0	::OCTAL DIGIT COUNTER
3318	014775	000		SOFILL:	.BYTE	0	::ZERO FILL SWITCH
3319	014776	000000		SOMODE:	.WORD	0	::NUMBER OF DIGITS TO TYPE
3320							::ENTER HERE ON POWER FAILURE
3321							
3322							
3323	015000			SPWRDN:			
3324	015000	010046		.PFAIL:	MOV	R0,-(SP)	::SAVE R0-R5 ON PROCESSOR STACK
3325	015002	010146			MOV	R1,-(SP)	
3326	015004	010246			MOV	R2,-(SP)	
3327	015006	010346			MOV	R3,-(SP)	
3328	015010	010446			MOV	R4,-(SP)	
3329	015012	010546			MOV	R5,-(SP)	
3330	015014	016746	163004		MOV	24,-(SP)	
3331	015020	010667	164072		MOV	SP,SAVSP	::SAVE STACK POINTER
3332	015024	012767	015036 162772		MOV	#RESTART,24	::SET UP FOR POWER UP TRAP
3333	015032	000000			HALT		::HALT ON POWER DOWN NORMAL
3334	015034	000777			BR		
3335							
3336							::PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
3337							
3338	015036	016706	164054	RESTAR:	MOV	SAVSP,SP	::RESTORE STACK POINTER
3339	015042	012605			MOV	(SP)+,R5	::RESTORE R0-R5
3340	015044	012604			MOV	(SP)+,R4	
3341	015046	012603			MOV	(SP)+,R3	
3342	015050	012602			MOV	(SP)+,R2	
3343	015052	012601			MOV	(SP)+,R1	
3344	015054	012600			MOV	(SP)+,R0	
3345	015056	012767	015000 162740		MOV	#PFAIL,24	::SET UP FOR POWER FAILURE
3346	015064	106427	000340		MTPS	#340	
3347	015070	012706	001100		MOV	#STACK,SP	
3348	015074	005067	001102		CLR	TEMP	
3349	015100	005267	001076		INC	TEMP	
3350	015104	001375			BNE	.-4	
3351	015106	104413			CONVRT		
3352	015110	015132			PFTAB		
3353	015112	104401			TYPE		
3354	015114	015434			MPFAIL		
3355	015116	005067	164261		CLR	SERFLG	
3356	015122	005067	164270		CLR	SERRPC	
3357	015126	000177	163752		JMP	#RETURN	
3358	015132	000001		PFTAB:	1		

3359	015134	006	002						
3360	015136	000207				.BYTE	6,2		
3361	015140	005015	042012	053125	MTITLE:	.ASCIZ	<15><12><12>/DUV11 DZDUT-A TAPE D /<15><12>		
3362	015146	030461	042040	042132					
3363	015154	052125	040455	052040					
3364	015162	050101	020105	020104					
3365	015170	005015	000						
3366	015173	015	053012	041505	MVECTO:	.ASCIZ	<15><12>/VEC ADD-/		
3367	015200	040440	042104	000055					
3368	015206	005015	051461	020124	MREGAD:	.ASCIZ	<15><12>/1ST DEV: REC CSR ADD-/		
3369	015214	042504	035126	051040					
3370	015222	041505	041440	051123					
3371	015230	040440	042104	000055					
3372	015236	005015	052515	052114	MMULT:	.ASCIZ	<15><12>/MULT DEV ? (Y OR N)-/		
3373	015244	042040	053105	037440					
3374	015252	024040	020131	051117					
3375	015260	047040	026451	000					
3376	015265	015	046012	051501	MLASTD:	.ASCIZ	<15><12>/LAST DEV: REC CSR ADDR-/		
3377	015272	020124	042504	035126					
3378	015300	051040	041505	041440					
3379	015306	051123	040440	042104					
3380	015314	026522	000						
3381	015317	075	042504	044526	DEVICE:	.ASCIZ	/=DEVICE /		
3382	015324	042503	020040	000					
3383	015331	015	051412	046105	MCOV:	.ASCIZ	<15><12>/SELECT TO RUN QACTREG/		
3384	015336	041505	020124	047524					
3385	015344	051040	047125	040040					
3386	015352	041501	051124	043505					
3387	015360	000							
3388	015361	015	047412	043126	MRANGE:	.ASCIZ	<15><12>/OVFLO:RETYPE LAST DEV RXCSR ADDS-/		
3389	015366	047514	051072	052105					
3390	015374	050131	020105	040514					
3391	015402	052123	042040	053105					
3392	015410	051040	041530	051123					
3393	015416	040440	042104	026523					
3394	015424	000							
3395	015425	040	037440	000	MOM:	.ASCIZ	/ ?/		
3396	015431	015	000012		MCRLF:	.ASCIZ	<15><12>		
3397	015434	043120	044501	026114	MPFAIL:	.ASCIZ	/PFAIL, RESTART AT TEST IN PROGRESS/		
3398	015442	020040	042522	052123					
3399	015450	051101	020124	052101					
3400	015456	052040	051505	020124					
3401	015464	047111	050040	047522					
3402	015472	051107	051505	000123					
3403	015500	005015	047105	020104	MEPASS:	.ASCIZ	<15><12>/END OF PASS TAPE D/		
3404	015506	043117	050040	051501					
3405	015514	020123	040524	042520					
3406	015522	042040	000						
3407	015525	015	051012	000	MR:	.ASCIZ	<15><12>/R/		
3408	015531	015	052012	051505	MTSTPC:	.ASCIZ	<15><12>/TEST PC-/		
3409	015536	020124	041520	000055					
3410	015544	005015	047514	045503	MLOCK:	.ASCIZ	<15><12>/LOCK ON TEST? (Y OR N)-/		
3411	015552	047440	020116	052040					
3412	015560	051505	037524	024040					
3413	015566	020131	051117	047040					
3414	015574	026451	000						



```

3415 015577 015 021412 047440 MSYNC: .ASCIZ <15><12>/# OF SYNC CHARS SELECTED ( 1 OR 2)-/
3416 015604 020106 054523 041516
3417 015612 041440 040510 051522
3418 015620 051440 046105 041505
3419 015628 042524 020104 020050
3420 015634 020061 051117 031040
3421 015642 026451 000
3422 015649 015 044412 020123 MWIRE6: .ASCIZ <15><12>/IS SEC XMIT SWITCH E55-2 IN? (Y OR N)-/
3423 015652 042523 020103 046530
3424 015660 052111 051440 044527
3425 015666 041524 020110 032505
3426 015674 026465 020062 047111
3427 015702 020077 054450 047440
3428 015710 020122 024516 000055
3429 015716 005015 051511 051440 MWIRE5: .ASCIZ <15><12>/IS SEC REC SWITCH E55-3 IN? (Y OR N)-/
3430 015724 041505 051040 041505
3431 015732 051440 044527 041524
3432 015740 020110 032505 026465
3433 015746 020063 047111 020077
3434 015754 054450 047440 020122
3435 015762 024516 000055
3436 015766 005015 051511 047440 MWIRE4: .ASCIZ <15><12>/IS OPT CLR ENABLE SWITCH E55-1 IN? (Y OR N)-/
3437 015774 052120 041440 051114
3438 016002 042440 040516 046102
3439 016010 020105 053523 052111
3440 016016 044103 042440 032465
3441 016024 030455 044440 037516
3442 016032 024040 020131 051117
3443 016040 047040 026451 000
3444 016045 015 005012 031510 MEXTJ: .ASCIZ <15><12><12>/H315 CONNECTOR ON ?(Y OR N)-/
3445 016052 032461 041440 047117
3446 016060 042516 052103 051117
3447 016066 047440 020116 024077
3448 016074 020131 051117 047040
3449 016102 026451 000
3450 016105 015 020012 043536 MCNTG: .ASCIZ <15><12>/ tG /
3451 016112 020040 000
3452 016115 040 053523 036522 MMSWR: .ASCIZ / SWR= /
3453 016122 020040 000040
3454 016126 020040 047040 053505 MMNEW: .ASCIZ / NEW= /
3455 016134 020075 000040
3456 .EVEN
3457
3458 ;BUFFERS FOR INPUT-OUTPUT
3459
3460 016140 000000 INBUF: 0
3461 016202 016202 .=. +40
3462 016202 000000 TEMP: 0
3463 016244 016244 .=. +40
3464 016244 000000 MDATA: 0
3465 016306 016306 .=. +40
3466 .SBTTL SCOPE HANDLER ROUTINE
3467
3468 ;: *****
3469 ;: *THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3470 ;: *AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)

```

```

3471 ;*AND LOAD THE ERROR FLAG (SERFLG) INTO DISPLAY<15:08>
3472 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3473 ;*SW14=1 LOOP ON TEST
3474 ;*SW11=1 INHIBIT ITERATIONS
3475 ;*SW09=1 LOOP ON ERROR
3476 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
3477 ;*CALL
3478 ;* SCOPE ;;SCOPE=IOT
3479
3480 016306 $$SCOPE:
3481
3482 ;SCOPE LOOP AND INTERATION HANDLER
3483
3484 .SCOPE:
3485 016306 004767 174376 JSR PC,CKSWR
3486 016312 005067 163100 CLR SERRPC ;CLEAR LAST ERROR PC
3487 016316 022716 003364 CMP #TST1+2,(SP) ;IS SCOPE AT BEGINING OF TEST 1?
3488 016322 001413 BEQ $XTSTR ;YES NO LOOP.
3489
3490 016324 000406 TTST: BR 1$ ;GO TO 1$ (IF LOCK SW02=1)
3491 016326 105777 163112 TSTB #STKS ;KEYBOARD DONE?
3492 016332 100123 BPL $OVER ;BR IF NO
3493 016334 017766 163106 177776 MOV #STKB,-2(SP) ;CLEAR DONE BIT
3494 016342 032777 040000 163070 1$: BIT #BIT14,#SWR ;LOOP ON PRESENT TEST?
3495 016350 001114 BNE $OVER ;YES IF SW14=1
3496
3497 016352 000416 ;*****START OF CODE FOR THE XOR TESTER*****
3498 $XTSTR: BR 6$ ;IF RUNNING ON THE "XOR" TESTER CHANGE
3499 016354 013746 000004 MOV #ERRVEC,-(SP) ;THIS INSTRUCTION TO A "NOP" (NOP=240)
3500 016360 012737 016400 000004 MOV #S$,#ERRVEC ;SAVE THE CONTENTS OF THE ERROR VECTOR
3501 016366 005737 177060 TST #177060 ;SET FOR TIMEOUT
3502 016372 012637 000004 MOV (SP)+,#ERRVEC ;TIME OUT ON XOR?
3503 016376 000463 BR $SVLAD ;RESTORE THE ERROR VECTOR
3504 016400 022626 5$: CMP (SP)+,(SP)+ ;GO TO THE NEXT TEST
3505 016402 012637 000004 MOV (SP)+,#ERRVEC ;CLEAR THE STACK AFTER A TIME OUT
3506 016406 000423 BR 7$ ;RESTORE THE ERROR VECTOR
3507 016410 6$: ;*****END OF CODE FOR THE XOR TESTER***** ;LOOP ON THE PRESENT TEST
3508 016410 032777 000400 163022 BIT #BIT08,#SWR ;LOOP ON SPEC. TEST?
3509 016416 001404 BEQ 2$ ;BR IF NO
3510 016420 127767 163014 162754 CMPB #SWR,$STSTNM ;ON THE RIGHT TEST? SWR<7:0>
3511 016426 001465 BEQ $OVER ;BR IF YES
3512 016430 105767 162747 2$: TSTB SERFLG ;HAS AN ERROR OCCURRED?
3513 016434 001421 BEQ 3$ ;BR IF NO
3514 016436 126767 162753 162737 CMPB $ERMAX,SERFLG ;MAX. ERRORS FOR THIS TEST OCCURRED?
3515 016444 101015 BHI 3$ ;BR IF NO
3516 016446 032777 001000 162764 BIT #BIT09,#SWR ;LOOP ON ERROR?
3517 016454 001404 BEQ 4$ ;BR IF NO
3518 016456 016767 162726 162722 7$: MOV $LPERR,$LPADR ;SET LOOP ADDRESS TO LAST SCOPE
3519 016464 000446 BR $OVER
3520 016466 105067 162711 4$: CLRB SERFLG ;ZERO THE ERROR FLAG
3521 016472 005067 163014 CLR $TIMES ;CLEAR THE NUMBER OF ITERATIONS TO MAKE
3522 016476 000415 BR 1$ ;ESCAPE TO THE NEXT TEST
3523 016500 032777 004000 162732 3$: BIT #BIT11,#SWR ;INHIBIT ITERATIONS?
3524 016506 001011 BNE 1$ ;BR IF YES
3525 016510 005767 163020 TST $PASS ;IF FIRST PASS OF PROGRAM
3526 016514 001406 BEQ 1$ ;INHIBIT ITERATIONS

```



DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 71  
 DZDUTA.M11 13-OCT-76 08:39 SCOPE HANDLER ROUTINE

```

3527 016516 005267 162662          INC      SICNT      ;; INCREMENT ITERATION COUNT
3528 016522 026767 162764 162654    CMP      $TIMES,$SICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
3529 016530 002024          BGE      $OVER      ;; BR IF MORE ITERATION REQUIRED
3530 016532 012767 000001 162644 1S:  MOV      #1,$SICNT  ;; REINITIALIZE THE ITERATION COUNTER
3531 016540 016767 000056 162744    MOV      $SMXCNT,$STIMES ;; SET NUMBER OF ITERATIONS TO DO
3532 016546 105267 162630          INCB     $STSTNM     ;; COUNT TEST NUMBERS
3533 016552 116767 162624 162752    MOVB    $STSTNM,$STSTN ;; SET TEST NUMBER IN APT MAILBOX
3534 016560 011667 162622          MOV      (SP),$LPADR  ;; SAVE SCOPE LOOP ADDRESS
3535 016564 011667 162620          MOV      (SP),$LPERR  ;; SAVE ERROR LOOP ADDRESS
3536 016570 005067 162720          CLR     $ESCAPE     ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
3537 016574 112767 000001 162613    MOVB    #1,$SERMAX   ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3538 016602 016777 162574 162632    $OVER:  MOV      $STSTNM,$DISPLAY ;; DISPLAY TEST NUMBER
3539 016610 016716 162572          MOV      $LPADR,(SP) ;; FUDGE RETURN ADDRESS
3540 016614 000002          4S:    RTI
3541 016616 001407          BRW:    1407
3542 016620 000432          BRX:    432
3543 016622 000005          $SMXCNT: 5          ;;MAX. NUMBER OF ITERATIONS
3544          .SBTTL TRAP DECODER
3545
3546          ;;*****
3547          ;;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3548          ;;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3549          ;;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3550          ;;*GO TO THAT ROUTINE.
3551
3552 016624 010046          $TRAP:  MOV      RO,-(SP)    ;; SAVE RO
3553 016626 016600 000002          MOV      2(SP),RO    ;; GET TRAP ADDRESS
3554 016632 005740          TST     -(RO)        ;; BACKUP BY 2
3555 016634 111000          MOVB   (RO),RO      ;; GET RIGHT BYTE OF TRAP
3556 016636 006300          ASL    RO            ;; POSITION FOR INDEXING
3557 016640 016000 016660          MOV      $TRPAD(RO),RO ;; INDEX TO TABLE
3558 016644 000200          RTS     RO           ;; GO TO ROUTINE
3559
3560          ;; THIS IS USE TO HANDLE THE "GETPRI" MACRO
3561
3562 016646 011646          $TRAP2: MOV      (SP),-(SP) ;; MOVE THE PC DOWN
3563 016650 016666 000004 000002    MOV      4(SP),2(SP) ;; MOVE THE PSW DOWN
3564 016656 000002          RTI          ;; RESTORE THE PSW
3565
3566          .SBTTL TRAP TABLE
3567
3568          ;;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3569          ;;*BY THE "TRAP" INSTRUCTION.
3570
3571          :
3572          : ROUTINE
3573          : -----
3574 016660 016646          $TRPAD: .WORD   $TRAP2      TRAP+1(104401)  TTY TYPEOUT ROUTINE
3575 016662 013060          .STYPE  ;;CALL=TYPE      TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3576 016664 014576          .STYPOC ;;CALL=TYPOC      TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
3577 016666 014552          .STYPOS ;;CALL=TYPOS      TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
3578 016670 014612          .STYPON ;;CALL=TYPON
3579
3580
3581 016672 013036          .SCOPI  ;;CALL=SCOPI    TRAP+5(104405)
3582 016674 013342          .INSTR  ;;CALL=INSTR    TRAP+6(104406)

```

```

3583 016676 013450 .INSTER ;; CALL=INSTER TRAP+7(104407)
3584 016700 013460 .PARAM ;; CALL=PARAM TRAP+10(104410)
3585 016702 013660 .SAVOS ;; CALL=SAVOS TRAP+11(104411)
3586 016704 013720 .RESOS ;; CALL=RESOS TRAP+12(104412)
3587 016706 013752 .CONVRT ;; CALL=CONVRT TRAP+13(104413)
3588 016710 014130 .SETFLG ;; CALL=SETFLG TRAP+14(104414)
3589 ;*****
3590 ;UTILITIES
3591 ;*****
3592
3593 ;THIS UTILITY CALCULATES PRIORITY LEVEL
3594 016712 006367 000044 DULEV: ASL DUPRT ;SHIFT LEFT
3595 016716 006367 000040 ASL DUPRT
3596 016722 006367 000034 ASL DUPRT
3597 016726 006367 000030 ASL DUPRT
3598 016732 006367 000024 ASL DUPRT
3599 016736 016767 000020 MOV DUPRT,LESS1 ;MOVE THIS TO LESS1
3600 016744 162767 000001 SUB #1,LESS1 ;CREATE LESS1
3601 016752 042767 000037 BIC #37,LESS1 ;CLEAR TNZVC
3602 016760 000207 RTS PC
3603 016762 000240 DUPRT: PR5
3604 016764 000200 LESS1: PR4 ;LEVEL TO ALLOW INTERRUPTS
3605
3606 ;NEW DU ADDRESSES
3607 016766 016767 000126 162716 DUADDR: MOV DUBASE,RXCSR ;XXX0
3608 016774 005267 000120 INC DUBASE
3609 017000 016767 000114 162706 MOV DUBASE,HRXCSR ;XXX1
3610 017006 005267 000106 INC DUBASE
3611 017012 016767 000102 162676 MOV DUBASE,RXDBUF ;XXX2
3612 017020 016767 000074 162674 MOV DUBASE,PARCSR ;XXX2
3613 017026 005267 000066 INC DUBASE
3614 017032 016767 000062 162660 MOV DUBASE,HRXDBUF ;XXX3
3615 017040 016767 000054 162656 MOV DUBASE,HPARCSR ;XXX3
3616 017046 005267 000046 INC DUBASE
3617 017052 016767 000042 162646 MOV DUBASE,TXCSR ;XXX4
3618 017060 005267 000034 INC DUBASE
3619 017064 016767 000030 162636 MOV DUBASE,HTXCSR ;XXX5
3620 017072 005267 000022 INC DUBASE
3621 017076 016767 000016 162626 MOV DUBASE,TXDBUF ;XXX6
3622 017104 005267 000010 INC DUBASE
3623 017110 016767 000004 162616 MOV DUBASE,HTXDBUF ;XXX7
3624 017116 000207 RTS PC
3625 017120 000000 DUBASE: 0
3626
3627 ;THIS UTILITY POKES THE MAINT DATA BASED UPON THE
3628 ;INFORMATION CONTAINED IN STMP1 AND IT IS
3629 ;SHIFTED IN BY THE CONTENTS OF SHIFT
3630 017122 042777 040000 162576 RPOKE: BIC #MTDATA,@TXCSR
3631 017130 005067 162346 CLR STMP2
3632 017134 006067 162340 ROR STMP1 ;FORCE CARRY
3633 017140 006067 162336 ROR STMP2 ;PICK UP CARRY IN BIT 15
3634 017144 006267 162332 ASR STMP2 ;SHIFT INTO BIT 14
3635 017150 042767 100000 162324 BIC #BIT15,STMP2 ;CLR BIT 15
3636 017156 056777 162320 162542 BIS STMP2,@TXCSR ;POKE MAINT DATA
3637 017164 042777 020000 162534 BIC #CLK,@TXCSR ;POKE CLK
3638 017172 052777 020000 162526 BIS #CLK,@TXCSR

```



DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 73  
 DZDUTA.M11 13-OCT-76 08:39 TRAP TABLE

```

3639 017200 005367 161716          DEC      SHIFT
3640 017204 001346          BNE      RPOKE
3641 017206 000207          RTS      PC
3642
3643          ;THIS ROUTINE CALCULATES ODD PARITY FOR AN 8 BIT CHAR
3644 017210 016767 162264 162264  ODD8:  MOV     STMP1,STMP2      ;SAVE TEMP1
3645 017216 005067 162262          CLR     STMP3
3646 017222 012727 000010          MOV     #8.,(PC)+
3647 017226 000000          4$:    D
3648 017230 006067 162246          1$:    ROR     STMP2
3649 017234 005567 162244          ADC     STMP3
3650 017240 005367 177762          DEC     4$
3651 017244 001371          BNE     1$
3652 017246 006067 162232          ROR     STMP3
3653 017252 103404          BCS     2$
3654 017254 052767 000400 162216  BIS     #BIT8,STMP1      ;SET ODD PARITY
3655 017262 000403          BR      3$
3656 017264 042767 000400 162206  2$:    BIC     #BIT8,STMP1      ;CLR EVEN PARITY
3657          ;STMP1 NOW HAS ODD PARITY CHARACTER
3658 017272 000207          3$:    RTS      PC
3659
3660          ;THIS ROUTINE CALCULATES EVEN PARITY FOR AN 8 BIT CHARACTER
3661 017274 016767 162200 162200  EVEN8: MOV     STMP1,STMP2      ;SAVE TEMP1
3662 017302 005067 162176          CLR     STMP3
3663 017306 012727 000010          MOV     #8.,(PC)+
3664 017312 000000          4$:    D
3665 017314 006067 162162          1$:    ROR     STMP2
3666 017320 005567 162160          ADC     STMP3
3667 017324 005367 177762          DEC     4$
3668 017330 001371          BNE     1$
3669 017332 006067 162146          ROR     STMP3
3670 017336 103004          BCC     2$
3671 017340 052767 000400 162132  BIS     #BIT8,STMP1      ;SET EVEN PARITY
3672 017346 000403          BR      3$
3673 017350 042767 000400 162122  2$:    BIC     #BIT8,STMP1      ;CLR ODD PARITY
3674          ;STMP1 NOW HAS EVEN PARITY CHARACTER
3675 017356 000207          3$:    RTS      PC
3676 017360 062716 000002  TRPREG: ADD     #2,(SP) ;ALLOW IT TO "CRUNCH" INTO HLT BACK
3677          ;IN MAIN PART OF THE PROGRAM
3678 017364 000002          RTI
3679          .END

```

# H06

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 75  
 DZDUTA.M11 13-OCT-76 08:39 CROSS REFERENCE TABLE -- USER SYMBOLS

AAA	003200	1292#								
ABASE =	000000	875	916							
ACDM1 =	000000	875	918							
ACDM2 =	000000	875	919							
ACPUOP =	000000	875	890							
ACTREG	001166	734#	1248#	1262*	1263*	1270*	2793	2796	2806	
ADDMD =	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 =	013657	3008#	3047#	3054#						
AENV =	000000	875	886							
AENVN =	000000	875	887							
AFATAL =	000000	875	878							
AMADR1 =	000000	875	903							
AMADR2 =	000000	875	907							
AMADR3 =	000000	875	910							
AMADR4 =	000000	875	913							
AMANS1 =	000000	875	897							
AMANS2 =	000000	875	905							
AMANS3 =	000000	875	908							
AMANS4 =	000000	875	911							
AMSGAO =	000000	875	883							
AMSGLC =	000000	875	884							
AMSGTY =	000000	875	877							
ANTYP1 =	000000	875	898							
ANTYP2 =	000000	875	906							
ANTYP3 =	000000	875	909							
ANTYP4 =	000000	875	912							
APASS =	000000	875	880							
APRIOR =	000000	875								
APTCsu =	000040	535#	2923							
APTENV =	000001	535#	2916	3170						
APTSIZ =	000200	535#	1167							
APTSPO =	000100	535#	2918							
ASWREG =	000000	875	888							
ATESTN =	000000	875	879							
AUNIT =	000000	875	882							
AUSMR =	000000	875	889							
AVECT1 =	000000	875	914							
AVECT2 =	000000	875	915							
BASEAD	001154	729#	1230#	1267*	1268	1274*	1276*	2800*	2812*	2816

















SW15 = 100000	611#													
SW2 = 000004	634#													
SW3 = 000010	633#													
SW4 = 000020	632#													
SW5 = 000040	631#													
SW6 = 000100	630#													
SW7 = 000200	629#													
SW8 = 000400	628#													
SW9 = 001000	627#													
SYNCNO 001146	720#	1297*	1301*											
SYNEXT= 020000	788#	981#	2014	2022	2075	2083	2136	2144	2197	2205				
SYNINT= 030000	787#	980#	1352	1359	1770	1778	1831	1839	1892	1900	1953	1961	2255	
	2263	2308	2316	2468	2476	2521	2529	2682	2690	2735	2743			
SYNSCH= 000020	773#	966#	1360	1408	1501	1597	1693							
SYSTST= 014000	813#	1006#												
TBITVE= 000014	669#													
TEMP 016202	3093	3348*	3349*	3462#										
TKVEC = 000060	676#													
TPVEC = 000064	677#													
TRAPVE= 000034	675#	1141*	1142*											
TRPREG 017360	3676#													
TRTVEC= 000014	670#													
TST1 003362	1332	1338	1350#	2849	2850	3487								
TST10 006116	1890#													
TST11 006334	1951#													
TST12 006552	2012#													
TST13 006770	2073#													
TST14 007206	2134#													
TST15 007424	2195#													
TST16 007642	2253#													
TST17 010050	2306#													
TST2 003532	1396#													
TST20 010256	2359#													
TST21 010464	2412#													
TST22 010672	2466#													
TST23 011100	2519#													
TST24 011306	2572#													
TST25 011514	2625#													
TST26 011722	2680#													
TST27 012130	2733#													
TST3 004120	1489#													
TST4 004506	1585#													
TST5 005074	1681#													
TST6 005462	1768#													
TST7 005700	1829#													
TTST 016324	3490#													
TXCSR 001726	1047#	1351*	1353*	1356*	1362*	1363*	1365*	1366*	1397*	1399*	1402*	1410*	1411*	
	1413#	1414#	1490#	1492#	1495#	1503#	1504#	1506#	1507#	1586#	1588#	1591#	1599#	
	1600#	1602#	1603#	1682#	1684#	1687#	1695#	1696#	1698#	1699#	1769#	1771#	1775#	
	1779	1784#	1785#	1791#	1792#	1793	1802#	1804	1811	1830#	1832#	1836#	1840	
	1845#	1846#	1852#	1853#	1854	1863#	1865	1872	1891#	1893#	1897#	1901	1906#	
	1907#	1913#	1914#	1915	1924#	1926	1933	1952#	1954#	1958#	1962	1967#	1968#	
	1974#	1975#	1976	1985#	1987	1994	2013#	2015#	2019#	2023	2028#	2029#	2035#	
	2036#	2037	2046#	2048	2055	2074#	2076#	2080#	2084	2089#	2090#	2096#	2097#	
	2098	2107#	2109	2116	2135#	2137#	2141#	2145	2150#	2151#	2157#	2158#	2159	
	2168#	2170	2177	2196#	2198#	2202#	2206	2211#	2212#	2218#	2219#	2220	2229#	









SNWTST=	000000	1349#	1395#	1488#	1584#	1680#	1767#	1828#	1889#	1950#	2011#	2072#	2133#	2194#
		2252#	2305#	2358#	2411#	2465#	2518#	2571#	2624#	2679#	2732#			
SOCNT	014774	3275*	3304*	3317#										
SOMODE	014776	3270#	3274*	3279	3282*	3293*	3319#							
SOVER	016602	3492	3495	3511	3519	3529	3538#							
SPASS	001534	880#	1166*	2838*	3525	3544								
SPASTH	002144	1116#												
SPHRON	015000	1143	3323#											
SOLES	001522	868#	2972	3188										
SROCHR=	*****	3581												
SRODEC=	*****	3581												
SROLIN=	*****	3581												
SROOCT=	*****	3581												
SREGAD	001460	851#												
SREG0	001462	853#	3067*	3072										
SREG1	001464	854#	3066*	3073										
SREG2	001466	855#	3065*	3074										
SREG3	001470	856#	3064*	3075										
SREG4	001472	857#	3063*	3076										
SREG5	001474	858#	3062*	3077										
SR2A =	*****	3581												
SSAVRE=	*****	3581												
SSCOPE	016306	1137	3480#											
SSETUP=	000017	1119#	1136	1137	1139	1141	1143	1145	1146	1148	3155	3180	3187	3481
SSTUP =	177777	1119#												
SSVLAD	016546	3503	3532#											
SSVPC =	002136	1091#	1096											
SSWR =	177400	524#	865	866	867	1145	1146	1148	1149	1351	1397	1490	1586	1682
		1769	1830	1891	1952	2013	2074	2135	2196	2254	2307	2360	2413	2467
		2520	2573	2626	2681	2734	3146	3147	3148	3149	3150	3158	3165	3177
		3180	3188	3472	3473	3474	3475	3476	3494	3506	3508	3509	3512	3513
		3514	3521	3522	3523	3535	3538	3543						
SSWREG	001550	888#	1169											
SSWRK=	000000	3476	3477	3510										
STESTN	001532	879#	3533*											
STIMES	001512	865#	1145*	3521*	3528	3531*	3543							
STKB	001446	844#	2862	2985	2993	3493								
STKS	001444	843#	2860	2983	3491									
STMP0	001476	859#												
STMP1	001500	860#	1369*	1417*	1435*	1455*	1458*	1461*	1464*	1510*	1528*	1551*	1554*	1557*
		1560#	1606*	1624*	1647*	1650*	1653*	1656*	1702*	1720*	1743*	1746*	1749*	1752*
		1781#	1787*	1842*	1848*	1903*	1909*	1964*	1970*	2025*	2031*	2086*	2092*	2147*
		2153#	2208*	2214*	2266*	2272*	2319*	2325*	2372*	2378*	2425*	2431*	2479*	2485*
		2532#	2538*	2585*	2591*	2638*	2644*	2693*	2699*	2746*	2752*	3632*	3644	3654*
		3656#	3661	3671*	3673*									
STMP2	001502	861#	3631*	3633*	3634*	3635*	3636	3644*	3648*	3661*	3665*			
STMP3	001504	862#	3645*	3649*	3652*	3662*	3666*	3669*						
STMP4	001506	863#												
STMP5	001510	864#												
STN =	000030	535#	1349	1351#	1395	1397#	1488	1490#	1584	1586#	1680	1682#	1767	1769#
		1828	1830#	1889	1891#	1950	1952#	2011	2013#	2072	2074#	2133	2135#	2194
		2196#	2252	2254#	2305	2307#	2358	2360#	2411	2413#	2465	2467#	2518	2520#
		2571	2573#	2624	2626#	2679	2681#	2732	2734#					
STPB	001452	846#	2961*	2972	2993*									
STPFLG	001457	850#	2910	2972										
STPS	001450	845#	2959	2972	2991									

U  
U  
U  
U  
U  
U











H07

DZDUT-A MACY11 27(1006) 03-FEB-77 07:54 PAGE 89  
DZDUTA.M11 13-OCT-76 08:39 CROSS REFERENCE TABLE -- MACRO NAMES

.SCHTA	524#	814
.SEOP	524#	
.SERRO	524#	3140
.SERRT	524#	3196
.SPOWE	524#	
.SSCOP	524#	3466
.STRAP	524#	3544
.STYPE	524#	2893
.STYPO	524#	3243

. ABS. 017366 000

ERRORS DETECTED: 0  
DEFAULT GLOBALS GENERATED: 0

DZDUTA, DZDUTA, SEQ/SOL/CRF+DZDUT1/EQ: RUND, DZDUT2, DZDUTA  
RUN-TIME: 18 12 1 SECONDS  
RUN-TIME RATIO: 141/31=4.4  
CORE USED: 31K (62 PAGES)