

DUV11

DUV11 OFLNE LGC TSTS
CZDUQCO

AH-8706C-MC
FICHE 1 OF 1

FEB 1981
COPYRIGHT © 77-80
MADE IN USA



.REM

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

PRODUCT CODE: AC-8704C-MC

PRODUCT NAME: CZDUQCO DUV11 OFLNE LGC TSTS

PRODUCT DATE: AUGUST , 1980

MAINTAINER: DIAGNOSTIC ENGINEERING

*
.REM

COPYRIGHT (C) 1977,1980 BY DIGITAL EQUIPMENT CORPORATION

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.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	

*

GENERAL DESCRIPTION

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

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

2. REQUIREMENTS

PDP-11/03 COMPUTER (LSI)

DUV11 SYNCHRONOUS/ISOCRONOUS OPTION

ONE CONSOLE TELETYPE OR EQUIVALENT

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

3. LOADING PROCEDURE

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

STARTING ADDRESS FOR ABSOLUTE LOADER

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

4. STARTING PROCEDURE

- 4.1 CONTROL SWITCH SETTINGS

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

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

.REM *

* .REM *

* .REM *

- SW14=1

4.2

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

- #### 4.3.1 INITIAL PROGRAM START

- 1

-REM

1

-REM

4.3.2 PROGRAM RESTART WITH ALL SWITCHES DOWN

- #### 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 @ 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 XM11 SWITCH E55-2 ON? (Y OR N)-" AND WAIT FOR AN INPUT FROM THE TELETYPE KEYBOARD

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

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

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

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

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

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

- 4.3.3.19 TYPE IN THE APPROPRIATE ANSWER YES OR NO FOLLOWED

BY A <CARRIAGE RETURN>. (NOTE: ALL MULTIPLE DEVICES MUST BE THE SAME)

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

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

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

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

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

4.3.4 PROGRAM RESTART WITH SW01=1
NOTE: THIS WILL ONLY WORK WHEN A SINGLE DEVICE IS SELECTED
,,,IT WILL NOT WORK IF MULTIPLE DEVICES ARE SELECTED

IF MULTIPLE DEVICES WERE PREVIOUSLY SELECTED,LOAD 0C0200,
AND SELECT SW00=1 AND ANSWER "NO" TO THE MULTIPLE DEVICE QUESTION
SEE 4.3.3

4.3.4.1 SET SW01=1 IN SWITCH REG (LOC. 176)

4.3.4.2 TYPE 200G.

4.3.4.3 PROGRAM WILL START.

4.3.4.4 THE PROGRAM WILL TYPE "TEST PC-" AND WAIT FOR AN INPUT FROM
THE TELETYPE KEYBOARD

4.3.4.5 TYPE IN THE ADDRESS OF THE TEST AT WHICH THE PROGRAM IS TO
BE STARTED FOLLOWED BY A <CARRIAGE RETURN>

4.3.4.6 THE PROGRAM WILL TYPE 'R' TO INDICATE THAT IT HAS STARTED
TESTING AT THE SELECTED TEST

NOTE: CARE MUST BE TAKEN WHEN THIS FEATURE IS USED
SINCE THERE IS NO PROTECTION AGAINST SELECTING AN ADDRESS
THAT IS IN THE MIDDLE OF A TEST

4.3.5 PROGRAM RESTART WITH SW14 =1
NOTE: THIS WILL ONLY WORK WHEN A SINGLE DEVICE IS SELECTED
SEE NOTE IN 4.3.4 FOR MORE DETAILS

4.3.5.1 SET SW14=1 IN SWITCH REG. (LOC. 176)

4.3.5.2 TYPE 200G.

4.3.5.3 PROGRAM WILL START.

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

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

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

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

5. OPERATING PROCEDURE

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

SW15 =1 HALT ON ERROR
SW14 =1 LOOP ON CURRENT TEST
SW13 =1 INHIBIT ERROR TYPEOUT
SW11 =1 INHIBIT ITERATIONS
SW10 =1 ESCAPE TO NEXT TEST ON ERROR
SW09 =1 LOOP ON ERROR
SW01 =1 RESTART PROGRAM AT SELECTED TEST
SW00 =1 RESELECT VECTOR AND CONTROL REGISTER ADDRESSES
&PARAMETERS AFTER A PROGRAM RESTART
TO INHIBIT 'END OF PASS' TYPEOUT - TURN TELETYPE OFF

6. ERRORS

6.1 ERROR HALTS (UNDER LSI ALL HALT ERRORS RETURN CONTROL TO O.D.T.) THERE ARE FOUR DISTINCT ERROR TYPEOUTS

6.1.1 PC+2 = ERROR PC WHERE PC +2 IS THE ADDRESS OF THE CALL TO THE ERROR HANDLER +2

REFER TO THE ABOVE 'HLT' IN DIAGNOSTIC FOR ERROR DESCRIPTION

CHECK ADDRESS @ RXCSR: TO LOCATE THE DEVICE PRESENTLY UNDER
TEST WHEN RUNNING MULTIPLE DEVICES

6.1.2 PC +2 = REGISTER ERROR PC REGISTER EXPECTED ACTUAL 1XXXXX YYYYYY ZZZZZZ

WHERE 1XXXXX 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
1XXXXX YYYYYY ZZZZZZ

WHERE 1XXXXX 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
1XXXXX YYYYYY ZZZZZZ

WHERE 1XXXXX IS THE ADDRESS OF THE FAILING TRANSMITTER (TXCST) 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
1XXXXX = DEVICE

WHERE Y IS THE TAPE LOADED

WHERE 1XXXXX IS THE DEVICE'S BASE REGISTER ADDRESS

TO INHIBIT THIS TIMEOUT - 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.

7.2.2.2ORSET 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 TIMEOUT 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 LOGIC BIT BANGING
OF THE DEVICE
SEE LISTING FOR DETAILS

* .REM *

* .REM *

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

11. CHANGE HISTORY

NOTE: HISTORY BEGINS WITH REV. C0

REV. C0: 1) ALLOW OPERATOR TO SPECIFY ADDRESS > 170000
2) INCREASE DELAY VALUE 'HOLD' TO COMPENSATE
FOR 11/23 EXECUTION TIME (FROM 3777 TO 6200)

12. LISTINGS

*

539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593

.SBTTL APT COMMUNICATIONS ROUTINE

```
$ATY1:  MOVB  #1,$FFLG      ;;TO REPORT FATAL ERROR
$ATY3:  MOVB  #1,$MFLG      ;;TO TYPE A MESSAGE
      BR      $ATYC
$ATY4:  MOVB  #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
$ATYC:
      MOV     R0,-(SP)       ;;PUSH R0 ON STACK
      MOV     R1,-(SP)       ;;PUSH R1 ON STACK
      TSTB    $MFLG         ;;SHOULD TYPE A MESSAGE?
      BEQ     5$            ;;IF NOT: BR
      CMPB    #APTENV,$ENV   ;;OPERATING UNDER APT?
      BNE     3$            ;;IF NOT: BR
      BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
      BEQ     3$            ;;IF NOT: BR
      MOV     @4(SP),R0      ;;GET MESSAGE ADDR.
      ADD     #2,4(SP)       ;;BUMP RETURN ADDR.
      TST     $MSGTYPE       ;;SEE IF DONE W/ LAST XMISSION?
      BNE     1$            ;;IF NOT: WAIT
      MOV     R0,$MSGAD      ;;PUT ADDR IN MAILBOX
      TSTB    (R0)+         ;;FIND END OF MESSAGE
      BNE     2$
      SUB     $MSGAD,R0      ;;SUB START OF MESSAGE
      ASR     R0             ;;GET MESSAGE LNTH IN WORDS
      MOV     R0,$MSGGLT     ;;PUT LENGTH IN MAILBOX
      MOV     #4,$MSGTYPE    ;;TELL APT TO TAKE MSG.
      BR      5$
      MOV     @4(SP),4$      ;;PUT MSG ADDR IN JSR LINKAGE
      ADD     #2,4(SP)       ;;BUMP RETURN ADDRESS
      MOV     177776,-(SP)   ;;PUSH 177776 ON STACK
      JSR     PC,$TYPE       ;;CALL TYPE MACRO
      .WORD   0
      4$:
      5$:
      10$:  TSTB    $FFLG     ;;SHOULD REPORT FATAL ERROR?
      BEQ     12$           ;;IF NOT: BR
      TST     $ENV          ;;RUNNING UNDER APT?
      BEQ     12$           ;;IF NOT: BR
      TST     $MSGTYPE      ;;FINISHED LAST MESSAGE?
      BNE     11$           ;;IF NOT: WAIT
      MOV     @4(SP),$FATAL  ;;GET ERROR #
      ADD     #2,4(SP)       ;;BUMP RETURN ADDR.
      INC     $MSGTYPE      ;;TELL APT TO TAKE ERROR
      CLRB    $FFLG         ;;CLEAR FATAL FLAG
      11$:
      12$:  CLRB    $FFLG
```


CZDUQ-CO
CZDUQ4.M11

MACY11 30A(1052)
09-SEP-80 09:28

09-SEP-80 10:57 PAGE 14
APT COMMUNICATIONS ROUTINE

M 1

SEQ 0012

594	000224	105067	000013	CLRB	\$LFLG	::CLEAR LOG FLAG
595	000230	105067	000006	CLRB	\$MFLG	::CLEAR MESSAGE FLAG
596	000234	012601		MOV	(SP)+,R1	::POP STACK INTO R1
597	000236	012600		MOV	(SP)+,R0	::POP STACK INTO R0
598	000240	000207		RTS	PC	::RETURN
599	000242	000		\$MFLG:	.BYTE 0	::MESSG. FLAG
600	000243	000		\$LFLG:	.BYTE 0	::LOG FLAG
601	000244	000		\$FFLG:	.BYTE 0	::FATAL FLAG
602		000246			.EVEN	
603		000200		APTSIZE=	200	
604		000001		APTENV=	001	
605		000100		APTSPool=	100	
606		000040		APTCsup=	040	
607		000001		\$TN=	1	
608						
609						
610						
611						
612						
613						
614						

CZDUQ-CO
CZDUQ4.M11

MACY11 30A(1052)
09-SEP-80 09:28

09-SEP-80 10:57 PAGE 15
APT COMMUNICATIONS ROUTINE

615
616
617
618
619
620
621
622
623
624
625
626
627
628

629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684

.ENABLE ABS

:CZDUQ-CO DUV11 TAPE A

:COPYRIGHT 1977,1980, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754

:STARTING PROCEDURE

:TYPE 200G

:PROGRAM WILL TYPE 'CZDUQ-CO DUV11 TAPE A '

:PROGRAM WILL TYPE 'R' TO INDICATE THAT TESTING HAS STARTED

:AT THE END OF A PASS, PROGRAM WILL TYPE 'END OF PASS TAPE A'

:AND THEN RESUME TESTING

.SBTTL BASIC DEFINITIONS

:*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***

STACK= 1100

.EQUIV EMT,ERROR

;;BASIC DEFINITION OF ERROR CALL

.EQUIV IOT,SCOPE

;;BASIC DEFINITION OF SCOPE CALL

:*MISCELLANEOUS DEFINITIONS

HT= 11

;;CODE FOR HORIZONTAL TAB

LF= 12

;;CODE FOR LINE FEED

CR= 15

;;CODE FOR CARRIAGE RETURN

CRLF= 200

;;CODE FOR CARRIAGE RETURN-LINE FEED

PS= 177776

;;PROCESSOR STATUS WORD

.EQUIV PS,PSW

STKLMT= 177774

;;STACK LIMIT REGISTER

PIRQ= 177772

;;PROGRAM INTERRUPT REQUEST REGISTER

DSWR= 177570

;;HARDWARE SWITCH REGISTER

DDISP= 177570

;;HARDWARE DISPLAY REGISTER

:*GENERAL PURPOSE REGISTER DEFINITIONS

R0= X0

;;GENERAL REGISTER

R1= X1

;;GENERAL REGISTER

R2= X2

;;GENERAL REGISTER

R3= X3

;;GENERAL REGISTER

R4= X4

;;GENERAL REGISTER

R5= X5

;;GENERAL REGISTER

R6= X6

;;GENERAL REGISTER

R7= X7

;;GENERAL REGISTER

SP= X6

;;STACK POINTER

PC= X7

;;PROGRAM COUNTER

:*PRIORITY LEVEL DEFINITIONS

PR0= 0

;;PRIORITY LEVEL 0

PR1= 40

;;PRIORITY LEVEL 1

PR2= 100

;;PRIORITY LEVEL 2

PR3= 140

;;PRIORITY LEVEL 3

PR4= 200

;;PRIORITY LEVEL 4

PR5= 240

;;PRIORITY LEVEL 5

PR6= 300

;;PRIORITY LEVEL 6

PR7= 340

;;PRIORITY LEVEL 7

:*'SWITCH REGISTER' SWITCH DEFINITIONS

SW15= 100000

SW14= 40000

685	020000	SW13=	20000
686	010000	SW12=	10000
687	004000	SW11=	4000
688	002000	SW10=	2000
689	001000	SW09=	1000
690	000400	SW08=	400
691	000200	SW07=	200
692	000100	SW06=	100
693	000040	SW05=	40
694	000020	SW04=	20
695	000010	SW03=	10
696	000004	SW02=	4
697	000002	SW01=	2
698	000001	SW00=	1
699		.EQUIV	SW09,SW9
700		.EQUIV	SW08,SW8
701		.EQUIV	SW07,SW7
702		.EQUIV	SW06,SW6
703		.EQUIV	SW05,SW5
704		.EQUIV	SW04,SW4
705		.EQUIV	SW03,SW3
706		.EQUIV	SW02,SW2
707		.EQUIV	SW01,SW1
708		.EQUIV	SW00,SW0

709
710 ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

711	100000	BIT15=	100000
712	040000	BIT14=	40000
713	020000	BIT13=	20000
714	010000	BIT12=	10000
715	004000	BIT11=	4000
716	002000	BIT10=	2000
717	001000	BIT09=	1000
718	000400	BIT08=	400
719	000200	BIT07=	200
720	000100	BIT06=	100
721	000040	BIT05=	40
722	000020	BIT04=	20
723	000010	BIT03=	10
724	000004	BIT02=	4
725	000002	BIT01=	2
726	000001	BIT00=	1
727		.EQUIV	BIT09,BIT9
728		.EQUIV	BIT08,BIT8
729		.EQUIV	BIT07,BIT7
730		.EQUIV	BIT06,BIT6
731		.EQUIV	BIT05,BIT5
732		.EQUIV	BIT04,BIT4
733		.EQUIV	BIT03,BIT3
734		.EQUIV	BIT02,BIT2
735		.EQUIV	BIT01,BIT1
736		.EQUIV	BIT00,BIT0

737
738 ;*BASIC "CPU" TRAP VECTOR ADDRESSES

739	000004	ERRVEC= 4	::TIME OUT AND OTHER ERRORS
740	000010	RESVEC= 10	::RESERVED AND ILLEGAL INSTRUCTIONS

CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 19
BASIC DEFINITIONS

C 2

SEQ 0015

741	000014	TBITVEC=14	::"T" BIT
742	000014	TRTVEC= 14	::TRACE TRAP
743	000014	BPTVEC= 14	::BREAKPOINT TRAP (BPT)
744	000020	IOTVEC= 20	::INPUT/OUTPUT TRAP (IOT) **SCOPE**
745	000024	PWRVEC= 24	::POWER FAIL
746	000030	EMTVEC= 30	::EMULATOR TRAP (EMT) **ERROR**
747	000034	TRAPVEC=34	::"TRAP" TRAP
748	000060	TKVEC= 60	::TTY KEYBOARD VECTOR
749	000064	TPVEC= 64	::TTY PRINTER VECTOR
750	000240	PIRQVEC=240	::PROGRAM INTERRUPT REQUEST VECTOR


```
751                                     ;STANDARD INTERRUPT VECTORS
752
753
754                                     . =174
755 000174 000000  DISPREG:0
756 000176 000000  SWREG:0
757 000200 000200  . =200
758 000200 000167 001746  JMP      .START      ;GO TO START OF PROGRAM
759
760
761
762                                     . =1100
763 001100 000000  .WORD 0
764 001102 177570  LIGHTS:177570
765
766
767                                     ;PROGRAM CONTROL PARAMETERS
768
769
770 001104 000000  RETURN: 0
771 001106 000000  NEXT: 0      ;ADDRESS OF NEXT TEST TO BE EXECUTED
772 001110 000000  LOCK: 0      ;ADDRESS FOR LOCK ON CURRENT DATA
773 001112 000000  PASCNT: 0    ;ADDRESS CONTAINING PASS COUNT
774 001114 000000  ERRCNT: 0    ;ERROR COUNT
775 001116 000000  SAVSP: 0     ;STACK POINTER STORAGE
776
777                                     ;PROGRAM VARIABLES
778
779 001120 000020  HOLD: 20      ;TEMPORARY STORAGE=DELAY TIME FOR CABLES
780 001122 000000  SHIFT: 0     ;TEMPORARY STORAGE= # OF SHIFTS PER CHAR
781 001124 000000  COUNT: 0     ;TEMPORARY STORAGE= # OF TIMES A CHAR WILL BE SENT
782 001126 000000  SAVPC: 0     ;PROGRAM COUNTER STORAGE
783 001130 000000  HLD0: 0
784 001132 000000  HLD1: 0
785 001134 000000  HLD2: 0
786 001136 000000  HLD3: 0
787 001140 000000  HLD4: 0
788 001142 000000  HLD5: 0
789 001144 000000  HLD6: 0
790
```

```
791                                     ;PROGRAM CONVERSATIONAL PARAMETERS
792 001146      377      SYNCNO: .BYTE 377      ;# OF SYNC CHARS REQ'D FOR SYNC'ZATION
793 001147      377      SEXMIT: .BYTE 377      ;SEC XMIT JUMPER "IN"
794 001150      377      SEREC: .BYTE 377      ;SEC REC JUMPER "IN"
795 001151      377      OPTCLR: .BYTE 377      ;OPTIONAL JUMPER CLR "IN"
796 001152      000      MULTD: .BYTE 0        ;NO MULTIPLE DEVICE FLAG
797 001153      377      JMRBY: .BYTE 377      ;EXTERNAL MODEM BYPASS JUMPER "IN"
798                                     .EVEN
799
800                                     ;PROGRAM MULTIPLE DEVICE PARAMETERS
801 001154      000000     BASEADD:      0        ;PROG CONTROLLED 1ST DEVICE ADDR
802 001156      000000     KEEPADD:      0        ;SAVED 1ST DEVICE ADDR
803 001160      000000     LASTADD:      0        ;LAST DEVICE RXCSR ADDR
804 001162      000000     BASEIV:      0        ;PROG CONTROLLED IV
805 001164      000000     KEEPIV:      0        ;SAVED INTR VECTOR
806 001166      000000     ACTREG:      0        ;ACTIVE REGISTER,,,MODIFY THIS
807                                     ;LOCATION TO DISQUALIFY OR QUALIFY
808                                     ;DEVICES (1= RUN,,,0= DON'T RUN)
809 001170      000000     ROTADD:      0        ;ROTATING POINTER FOR ACTREG..POINTS
810                                     ;TO DEVICE PRESENTLY UNDER TEST WHEN RUNNING MULTIPLE DEVICES
811
812                                     ;PROGRAM CONTROL FLAGS
813
814 001172      000      INIFLG: .BYTE 0        ;PROGRAM INITIALIZATION FLAG
815 001173      000      STFLG: .BYTE 0        ;TEST START FLAG
816 001174      000      LOKFLG: .BYTE 0       ;LOCK ON CURRENT TEST FLAG
817                                     .EVEN
818                                     .=1400
819
820
```



```
821
822
823
824           ;INSTRUCTION DEFINITIONS
825
826           005746   PUSH1SP=5746       ;DECREMENT PROCESSOR STACK 1 WORD =TST -(SP)
827           005726   POP1SP=5726        ;INCREMENT PROCESSOR STACK 1 WORD =TST (SP)+
828           010046   PUSHRO=10046        ;SAVE R0 ON STACK =MOV R0,-(SP)
829           012600   POPRO=12600         ;RESTORE R0 FROM STACK =MOV (SP)+,R0
830           024646   PUSH2SP=24646      ;DECREMENT STACK TWICE =CMP -(SP),-(SP)
831           022626   POP2SP=22626       ;INCREMENT STACK TWICE =CMP (SP)+,(SP)+
832
833           ;REGISTER DEFINITIONS
834           100000    ;RXCSR BIT DEFINITIONS
835           040000    DSC=BIT15          ;DATA SET CHANGE
836           020000    RING=BIT14         ;RING
837           010000    CTS=BIT13          ;CLR TO SEND
838           004000    CARDET=BIT12       ;CARRIER DETECT
839           002000    REACT=BIT11        ;REC ACTIVE
840           001000    SRD=BIT10          ;SEC REC DATA
841           000400    DSR=BIT9           ;DATA SET RDY
842           000200    STPSYN=BIT8        ;STRIP SYNC
843           000100    RXDONE=BIT7       ;REC DONE
844           000040    RINTEN=BIT6       ;REC INTR ENABLE
845           000020    DSINTE=BIT5       ;DSC INTR ENABLE
846           000010    SYNSCH=BIT4       ;SYNC SEARCH
847           000004    STD=BIT3          ;SEC XMIT DATA
848           000002    RTS=BIT2          ;REQ TO SEND
849           000001    DTR=BIT1          ;DATA TERM RDY
850           000000    VOID=BIT0
851           100000    ;RXDBUF BIT DEFINITIONS
852           040000    RXERR=BIT15        ;REC ERROR
853           020000    OVRRUN=BIT14       ;OVERRUN
854           010000    FRMERR=BIT13      ;FRAME ERROR
855           001000    PARER=BIT12       ;PARITY ERROR
856           000400    ;PARCSR BIT DEFINITIONS
857           000100    PAREN=BIT9         ;PARITY ENABLE
858           000040    EVPAR=BIT8        ;EVEN PARITY SENSE
859           030000    ;PARCSR WRD DEFINITIONS
860           020000    SYNINT=30000       ;SYNC EXTERNAL MODE
861           000000    SYNEXT=20000      ;SYNC INTERNAL MODE
862           000000    ISYMOD=0          ;ISOC MODE
863           000000    FIVE=0            ;WORD LENGTH 5 BITS
864           002000    SIX=2000          ;WORD LENGTH 6 BITS
865           004000    SEVEN=4000        ;WORD LENGTH 7 BITS
866           006000    EIGHT=6000       ;WORD LENGTH 8 BITS
867           000000    NOPAR=0           ;NO PARITY
868           001000    ODDPAR=1000       ;ODD PARITY
869           001400    EVEPAR=1400       ;EVEN PARITY
870           100000    ;TXCSR BIT DEFINITIONS
871           040000    DNA=BIT15          ;DATA NOT AVAILABLE
872           020000    MTDATA=BIT14      ;MAINT DATA
873           002000    CLK=BIT13         ;CLK
874           000400    BITW=BIT10        ;BIT WINDOW
875           000200    MRESET=BIT8       ;MASTER RESET
876           000100    TXDONE=BIT7      ;XMIT DONE
877           000010    TXINTE=BIT6      ;XMIT INTR ENABLE
```

CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 23
BASIC DEFINITIONS

G 2

SEQ 0019

877	000040	DNAINTE=BIT5	:DNA INTR ENAB
878	000020	SEND=BIT4	:SEND
879	000010	HDXEN=BIT3	:HDX/FDX
880	000001	BREAK=BIT0	:BREAK
881		:TXCSR WRD DEFINITIONS	
882	000000	USER=0	:USER MODE
883	004000	MINT=4000	:MAINT INT MODE
884	010000	MEXT=10000	:MAINT EXT MODE
885	014000	SYSTST=14000	:SYSTEM TEST MODE


```
886 .SBTTL COMMON TAGS
887
888 *****
889 *THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
890 *USED IN THE PROGRAM.
891
892 001400
893 001400 001400
894 001400 000000
895 001402 000
896 001403 000
897 001404 000000
898 001406 000000
899 001410 000000
900 001412 000000
901 001414 000
902 001415 001
903 001416 000000
904 001420 000000
905 001422 000000
906 001424 000000
907 001426 000000
908 001430 000000
909 001432 000000
910 001434 000
911 001435 000
912 001436 000000
913 001440 177570
914 001442 177570
915 001444 177560
916 001446 177562
917 001450 177564
918 001452 177566
919 001454 000
920 001455 002
921 001456 012
922 001457 000
923 001460 000000
924
925 001462 000000
926 001464 000000
927 001466 000000
928 001470 000000
929 001472 000000
930 001474 000000
931 001476 000000
932 001500 000000
933 001502 000000
934 001504 000000
935 001506 000000
936 001510 000000
937 001512 000000
938 001514 000000
939 001516 177607 000377
940 001522 077
941 001523 015
```

```
SCMTAG: .=.
          .WORD 0
$STSNM: .BYTE 0
$ERFLG: .BYTE 0
$ICNT: .WORD 0
$LPADR: .WORD 0
$LPERR: .WORD 0
$ERTTL: .WORD 0
$ITEMB: .BYTE 0
$ERMAX: .BYTE 1
$ERRPC: .WORD 0
$GDADR: .WORD 0
$BDADR: .WORD 0
$GDDAT: .WORD 0
$BDDAT: .WORD 0
          .WORD 0
          .WORD 0
$AUTOB: .BYTE 0
$INTAG: .BYTE 0
          .WORD 0
$SWR: .WORD DSWR
$DISPLAY: .WORD DDISP
$TKS: 177560
$TKB: 177562
$TPS: 177564
$TPB: 177566
$NULL: .BYTE 0
$FILLS: .BYTE 2
$FILLC: .BYTE 12
$TPFLG: .BYTE 0
$REGAD: .WORD 0
          .WORD 0
$REG1: .WORD 0
$REG2: .WORD 0
$REG3: .WORD 0
$REG4: .WORD 0
$REG5: .WORD 0
$TMP0: .WORD 0
$TMP1: .WORD 0
$TMP2: .WORD 0
$TMP3: .WORD 0
$TMP4: .WORD 0
$TMP5: .WORD 0
$TIMES: 0
$ESCAPE: 0
$BELL: .ASCII <207><377><377>
$QUES: .ASCII /?/
$CRLF: .ASCII <15>
```

```
:::START OF COMMON TAGS
:::CONTAINS THE TEST NUMBER
:::CONTAINS ERROR FLAG
:::CONTAINS SUBTEST ITERATION COUNT
:::CONTAINS SCOPE LOOP ADDRESS
:::CONTAINS SCOPE RETURN FOR ERRORS
:::CONTAINS TOTAL ERRORS DETECTED
:::CONTAINS ITEM CONTROL BYTE
:::CONTAINS MAX. ERRORS PER TEST
:::CONTAINS PC OF LAST ERROR INSTRUCTION
:::CONTAINS ADDRESS OF 'GOOD' DATA
:::CONTAINS ADDRESS OF 'BAD' DATA
:::CONTAINS 'GOOD' DATA
:::CONTAINS 'BAD' DATA
:::RESERVED--NOT TO BE USED
:::AUTOMATIC MODE INDICATOR
:::INTERRUPT MODE INDICATOR
:::ADDRESS OF SWITCH REGISTER
:::ADDRESS OF DISPLAY REGISTER
:::TTY KBD STATUS
:::TTY KBD BUFFER
:::TTY PRINTER STATUS REG. ADDRESS
:::TTY PRINTER BUFFER REG. ADDRESS
:::CONTAINS NULL CHARACTER FOR FILLS
:::CONTAINS # OF FILLER CHARACTERS REQUIRED
:::INSERT FILL CHARS. AFTER A 'LINE FEED'
:::'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
:::CONTAINS THE ADDRESS FROM
:::WHICH ($REG0) WAS OBTAINED
:::CONTAINS (($REGAD)+0)
:::CONTAINS (($REGAD)+2)
:::CONTAINS (($REGAD)+4)
:::CONTAINS (($REGAD)+6)
:::CONTAINS (($REGAD)+10)
:::CONTAINS (($REGAD)+12)
:::USER DEFINED
:::USER DEFINED
:::USER DEFINED
:::USER DEFINED
:::USER DEFINED
:::USER DEFINED
:::MAX. NUMBER OF ITERATIONS
:::ESCAPE ON ERROR ADDRESS
:::CODE FOR BELL
:::QUESTION MARK
:::CARRIAGE RETURN
```

```
942 001524 000012      SLF: .ASCIZ <12>          ;;LINE FEED
943                      ;*****
944                      ;SBTTL  APT MAILBOX-ETABLE
945                      ;*****
946                      ;EVEN
947                      $MAIL:          ;;APT MAILBOX
948 001526              $MSGTY: .WORD   AMSGTY  ;;MESSAGE TYPE CODE
949 001526 000000      $FATAL: .WORD   AFATAL  ;;FATAL ERROR NUMBER
950 001530 000000      $TESTN: .WORD   ATESTN  ;;TEST NUMBER
951 001532 000000      $PASS:  .WORD   APASS   ;;PASS COUNT
952 001534 000000      $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
953 001536 000000      $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
954 001540 000000      $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
955 001542 000000      $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
956 001544 000000      $ETABLE:        ;;APT ENVIRONMENT TABLE
957 001546              $ENV:  .BYTE   AENV    ;;ENVIRONMENT BYTE
958 001546          000  $ENVN: .BYTE   AENVN  ;;ENVIRONMENT MODE BITS
959 001547          000  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
960 001550 000000      $USWR:  .WORD   AUSWR   ;;USER SWITCHES
961 001552 000000      $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
962 001554 000000      ;*              BITS 15-11=CPU TYPE
963                      ;*              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
964                      ;*              11/70=06,PDQ=07,Q=10
965                      ;*              BIT 10=REAL TIME CLOCK
966                      ;*              BIT 9=FLOATING POINT PROCESSOR
967                      ;*              BIT 8=MEMORY MANAGEMENT
968                      $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
969 001556          000  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
970 001557          000  ;*              MEM.TYPE BYTE  -- (HIGH BYTE)
971                      ;*              900 NSEC CORE=001
972                      ;*              300 NSEC BIPOLAR=002
973                      ;*              500 NSEC MOS=003
974                      $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
975 001560 000000      ;*              MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF "TYPE" ABOVE
976                      $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
977 001562          000  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
978 001563          000  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
979 001564 000000      $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
980 001566          000  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
981 001567          000  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
982 001570 000000      $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
983 001572          000  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
984 001573          000  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
985 001574 000000      $SVCT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
986 001576 000000      $SVCT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
987 001600 000000      $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
988 001602 000000      $DEVN:  .WORD   ADEVN   ;;DEVICE MAP
989 001604 000000      $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
990 001606 000000      $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
991 001610 000000      $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
992 001612 000000      $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
993 001614 000000      $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
994 001616 000000      $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
995 001620 000000      $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
996 001622 000000      $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
997 001624 000000
```


CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 26
APT MAILBOX-ETABLE

J 2

SEQ 0022

998 001626 000000
999 001630 000000
1000 001632 000000
1001 001634 000000
1002 001636 000000
1003 001640 000000
1004 001642 000000
1005 001644 000000
1006 001646 000000
1007 001650 000000
1008
1009
1010 001652
1011
1012
1013

\$DDW6: .WORD ADDW6 ;;DEVICE DESCRIPTOR WORD#6
\$DDW7: .WORD ADDW7 ;;DEVICE DESCRIPTOR WORD#7
\$DDW8: .WORD ADDW8 ;;DEVICE DESCRIPTOR WORD#8
\$DDW9: .WORD ADDW9 ;;DEVICE DESCRIPTOR WORD#9
\$DDW10: .WORD ADDW10 ;;DEVICE DESCRIPTOR WORD#10
\$DDW11: .WORD ADDW11 ;;DEVICE DESCRIPTOR WORD#11
\$DDW12: .WORD ADDW12 ;;DEVICE DESCRIPTOR WORD#12
\$DDW13: .WORD ADDW13 ;;DEVICE DESCRIPTOR WORD#13
\$DDW14: .WORD ADDW14 ;;DEVICE DESCRIPTOR WORD#14
\$DDW15: .WORD ADDW15 ;;DEVICE DESCRIPTOR WORD#15

SETEND:

1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069005746
005726
010046
012600
024646
022626

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

100000
040000
020000
010000

001000
000400

030000
020000
000000
000000
002000
004000
006000
000000
001000
001400

100000
040000
020000
002000
000400
000200
000100

; INSTRUCTION DEFINITIONS

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

; REGISTER DEFINITIONS

; RXCSR BIT DEFINITIONS

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

; RXDBUF BIT DEFINITIONS

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

; PARCSR BIT DEFINITIONS

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

; PARCSR WRD DEFINITIONS

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

; TXCSR BIT DEFINITIONS

DNA=BIT15 ; DATA NOT 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

CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052) 09-SEP-80 10:57 PAGE 28
20-AUG-80 09:19 APT MAILBOX-ETABLE

L 2

SEQ 0024

1070	000040	DNAINT=BIT5	;DNA INTR ENAB
1071	000020	SEND=BIT4	;SEND
1072	000010	HDXEN=BIT3	;HDX/FDX
1073	000001	BREAK=BIT0	;BREAK
1074		;TXCSR WRD DEFINITIONS	
1075	000000	USER=0	;USER MODE
1076	004000	MINT=4000	;MAINT INT MODE
1077	010000	MEXT=10000	;MAINT EXT MODE
1078	014000	SYSTST=14000	;SYSTEM TEST MODE

```
1079      .SBTTL  ERROR POINTER TABLE
1080
1081      : *THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
1082      : *THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1083      : *LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
1084      : *NOTE1:      IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
1085      : *NOTE2:      EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1086
1087      : *      EM      ::POINTS TO THE ERROR MESSAGE
1088      : *      DH      ::POINTS TO THE DATA HEADER
1089      : *      DT      ::POINTS TO THE DATA
1090      : *      DF      ::POINTS TO THE DATA FORMAT
1091
1092
1093      001652      $ERRTB:
1094                  :ERROR TABLE
1095      001652      001762      EM1      ;ERROR 1      REGISTER ERROR
1096      001654      002067      DH1
1097      001656      002116      DT1
1098      001660      002132      DF1
1099      001662      002022      EM2      ;ERROR 2      RECEIVER ERROR
1100      001664      002067      DH1
1101      001666      002116      DT1
1102      001670      002132      DF1
1103      001672      002043      EM3      ;ERROR 3      TRANSMITTER ERROR
1104      001674      002067      DH1
1105      001676      002116      DT1
1106      001700      002132      DF1
1107      001702      001746      EM4      ;ERROR 4      BIT ERROR (GENERAL)
1108      001704      000000      0
1109      001706      002126      DT4
1110      001710      002132      DF1
1111
1112                  :DEFAULT DU ADDRESSES
1113      001712      160010      RXCSR: 160010
1114      001714      160011      HRXCSR: 160011
1115      001716      160012      RXDBUF: 160012
1116      001720      160013      HRXDBUF: 160013
1117      001722      160012      PARCSR: 160012
1118      001724      160013      HPARCSR: 160013
1119      001726      160014      TXCSR: 160014
1120      001730      160015      HTXCSR: 160015
1121      001732      160016      TXDBUF: 160016
1122      001734      160017      HTXDBUF: 160017
1123
1124                  :DEFAULT DU VECTORS
1125      001736      000770      DURIV: 770      ;REC INTR VECTOR
1126      001740      000772      DURIS: 772      ;REC INTR STATUS
1127      001742      000774      DUTIV: 774      ;XMIT INTR VECTOR
1128      001744      000776      DUTIS: 776      ;XMIT INTR STATUS
1129
1129      001746      020040      051105      047522      :ERROR MESSAGES
1130      001754      020122      041520      000040      EM4: .ASCIZ / ERROR PC /
1131      001762      020040      047503      050115      EM1: .ASCIZ / COMPARISON ERROR ON REGISTERS/
1132      001770      051101      051511      047117
1133      001776      042440      051122      051117
1134      002004      047440      020116      042522
```



```
1135 002012 044507 052123 051105
1136 002020 000123
1137 002022 020040 042522 042503 EM2: .ASCIZ / RECEIVER ERROR/
1138 002030 053111 051105 042440
1139 002036 051122 051117 000
1140 002043 040 052040 040522 EM3: .ASCIZ / TRANSMITTER ERROR/
1141 002050 051516 044515 052124
1142 002056 051105 042440 051122
1143 002064 051117 000
1144 ;DATA HEADERS FOR ERROR MESSAGES
1145 002067 105 051122 041520 DH1: .ASCIZ /ERRPC WANTED ACTUAL/
1146 002074 020040 040527 052116
1147 002102 042105 020040 041501
1148 002110 052524 046101 000
1149 002116 .EVEN
1150 ;DATA TABLES FOR ERROR MESSAGES
1151 002116 001416 001130 001132 DT1: .WORD $ERRPC,HLD0,HLD1,0
1152 002124 000000
1153
1154 002126 001416 000000 DT4: .WORD $ERRPC,0
1155
1156 002132 000 000 000 DF1: .BYTE 0,0,0,0
1157 002135 000
1158 .EVEN
1159 .SBTTL ACT11 HOOKS
1160
1161 ;*****
1162 ;HOOKS REQUIRED BY ACT11
1163 002136 $SVPC= . ;SAVE PC
1164 000046 .=46
1165 012644 $ENDAD ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1166 000052 .=52
1167 000000 .WORD 0 ;;2)SET LOC.52 TO ZERO
1168 002136 .=$SVPC ;; RESTORE PC
1169 .SBTTL APT PARAMETER BLOCK
1170
1171 ;*****
1172 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1173 ;*****
1174 002136 .$X= . ;;SAVE CURRENT LOCATION
1175 000024 .=24 ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1176 000020 200 ;;FOR APT START UP
1177 000044 .=44 ;;POINT TO APT INDIRECT ADDRESS PNTR.
1178 002136 $APTHDR ;;POINT TO APT HEADER BLOCK
1179 002136 .=$X ;;RESET LOCATION COUNTER
1180 ;*****
1181 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1182 ;INTERFACE SPEC.
1183
1184 002136 $APTHD:
1185 002136 000000 $HIBTS: .WORD 0 ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1186 002140 001526 $MBADR: .WORD $MAIL ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1187 002142 000010 $TSTM: .WORD 10 ;;RUN TIM OF LONGEST TEST
1188 002144 000010 $PASTM: .WORD 10 ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1189 002146 000000 $UNITM: .WORD ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1190 002150 000052 .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200 002152
1201
1202
1203 002152 012706 001400
1204 002156 005026
1205 002160 022706 001440
1206 002164 001374
1207 002166 012706 001100
1208
1209 002172 012737 016272 000020
1210 002200 012737 000340 000022
1211 002206 012737 014160 000030
1212 002214 012737 000340 000032
1213 002222 012737 016626 000034
1214 002230 012737 000340 000036
1215 002236 012737 014762 000024
1216 002244 012737 000340 000026
1217 002252 005067 177234
1218 002256 005067 177232
1219 002262 112767 000001 177125
1220 002270 012767 002270 177110
1221 002276 012767 002276 177104
1222
1223
1224 002304 013746 000004
1225 002310 012737 002344 000004
1226 002316 012767 177570 177114
1227 002324 012767 177570 177110
1228 002332 022777 177777 177100
1229 002340 001012
1230
1231 002342 000403
1232 002344 012716 002352
1233 002350 000002
1234 002352 012767 000176 177060
1235 002360 012767 000174 177054
1236 002366 012637 000004
1237
1238 002372 005067 177136
1239 002376 132767 000200 177143
1240 002404 001403
1241 002406 012767 001550 177024
1242 002414
1243 002414 012706 001100
1244 002420 106427 000340
1245 002424 012737 014762 000024
1246 002432 105067 176535

;PROGRAM INITIALIZATION
;LOCK OUT INTERRUPTS
;SET UP PROCESSOR STACK
;SET UP POWER FAIL VECTOR
;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
;TYPE TITLE MESSAGE

.START:
.SBTTL INITIALIZE THE COMMON TAGS
;;CLEAR THE COMMON TAGS ($CMTAG) AREA
MOV $CMTAG,R6 ;;FIRST LOCATION TO BE CLEARED
CLR (R6)+ ;;CLEAR MEMORY LOCATION
CMP $SWR,R6 ;;DONE?
BNE -6 ;;LOOP BACK IF NO
MOV $STACK,SP ;;SETUP THE STACK POINTER
;;INITIALIZE A FEW VECTORS
MOV $SCOPE,$IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
MOV $340,$IOTVEC+2 ;;LEVEL 7
MOV $ERROR,$EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
MOV $340,$EMTVEC+2 ;;LEVEL 7
MOV $STRAP,$TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
MOV $340,$TRAPVEC+2 ;;LEVEL 7
MOV $SPWRDN,$PWRVEC ;;POWER FAILURE VECTOR
MOV $340,$PWRVEC+2 ;;LEVEL 7
CLR $TIMES ;;INITIALIZE NUMBER OF ITERATIONS
CLR $ESCAPE ;;CLEAR THE ESCAPE ON ERROR ADDRESS
MOVB #1,$SERMAX ;;ALLOW ONE ERROR PER TEST
MOV #,$SLPADR ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
MOV #,$LPERR ;;SETUP THE ERROR LOOP ADDRESS
;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
MOV $ERRVEC,-(SP) ;;SAVE ERROR VECTOR
MOV $64,$ERRVEC ;;SET UP ERROR VECTOR
MOV $DSWR,$SWR ;;SETUP FOR A HARDWARE SWICH REGISTER
MOV $DDISP,$DISPLAY ;;AND A HARDWARE DISPLAY REGISTER
CMP #-1,$SWR ;;TRY TO REFERENCE HARDWARE SWR
BNE 66$ ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
;;AND THE HARDWARE SWR IS NOT = -1
BR 65$ ;;BRANCH IF NO TIMEOUT
MOV $65$,(SP) ;;SET UP FOR TRAP RETURN
RTI
MOV $SWREG,$SWR ;;POINT TO SOFTWARE SWR
MOV $DISPREG,$DISPLAY
MOV (SP)+,$ERRVEC ;;RESTORE ERROR VECTOR
CLR $PASS ;;CLEAR PASS COUNT
BITB $APTSIZE,$ENVN ;;TEST USER SIZE UNDER APT
BEQ 67$ ;;YES,USE NON-APT SWITCH
MOV $SWREG,$SWR ;;NO,USE APT SWITCH REGISTER
MOV $STACK,SP ;;SET STACK
MTPS $340 ;;LOCK INTERRUPTS
MOV #,$PFAIL,$24 ;;SET UP POWER FAIL VECTOR
CLRB $STFLG ;;CLEAR START FLAG
```


CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 32
INITIALIZE THE COMMON TAGS

C 3

SEQ 0028

1247	002436	005067	176450		CLR	PASCNT		;CLEAR PASS COUNT
1248	002442	105067	176735		CLRB	\$ERFLG		;CLEAR ERROR FLAG
1249	002446	005067	176740		CLR	\$ERTTL		;CLEAR ERROR COUNT
1250	002452	005067	176740		CLR	\$ERRPC		;CLEAR LAST ERROR POINTER
1251	002456	012767	000001	176716	MOV	#1,\$STSNM		;SET UP FOR TEST 1
1252	002464	012767	002152	176412	MOV	#.START,RETURN		;SET UP FOR POWER FAIL BEFORE
1253								;TESTING STARTS
1254	002472	013746	000006		MOV	@#6,-(SP)		
1255	002476	013746	000004		MOV	@#4,-(SP)		
1256	002502	012737	002516	000004	MOV	#1\$,@#4		
1257	002510	005777	176724		TST	@SWR		
1258	002514	000407			BR	2\$		
1259	002516	012767	000176	176714	MOV	#SWREG,SWR		
1260	002524	012767	000174	176710	MOV	#DISPREG,DISPLAY		
1261	002532	022626			CMP	(SP)+,(SP)+		
1262	002534	012637	000004		MOV	(SP)+,@#4		
1263	002540	012637	000006		MOV	(SP)+,@#6		
1264	002544	022767	000176	176666	CMP	#SWREG,SWR		
1265	002552	001007			BNE	3\$		
1266	002554	005737	000042		TST	@#42		;CHECK FOR CHAIN
1267	002560	001402			BEQ	33\$		
1268	002562	000167	000522		JMP	.BEGIN		
1269	002566	004767	010154		JSR	PC,CNTLU		
1270	002572	105767	176374		TSTB	INIFLG		;HAS INITIALIZATION BEEN PERFORMED
1271	002576	001004			BNE	ONCE		
1272	002600	104401	015122		TYPE	,MTITLE		;TYPE TITLE MESSAGE
1273	002604	105167	176362		COMB	INIFLG		;IF NOT SET FLAG AND DO
1274	002610	105767	176732		TSTB	\$ENV		;APT CONTROL?
1275	002614	001410			BEQ	11\$		;BR IF NO
1276	002616	032767	000001	176726	BIT	#1,\$USWR		;EXTENAL JUMPER ON?
1277	002624	001002			BNE	12\$		;NO
1278	002626	105067	176321		CLRB	JMRBY		;CLEAR FLAG
1279	002632	000167	000452		JMP	.BEGIN		;GO DO IT
1280	002636	032777	000001	176574	BIT	#SW00,@SWR		;RESELECT VECTOR & CONTROL REG?
1281	002644	001002			BNE	1\$		
1282	002646	000167	000436		JMP	.BEGIN		
1283	002652	012700	000300		MOV	#300,R0		;RESTORE VECTOR AREA TO TRAPCATCHER
1284	002656	012701	000302		MOV	#302,R1		;START AT LOCATION 300
1285	002662	012702	000004		MOV	#4,R2		
1286	002666	010110			MOV	R1,(R0)		
1287	002670	005011			CLR	(R1)		
1288	002672	060200			ADD	R2,R0		
1289	002674	060201			ADD	R2,R1		
1290	002676	022701	001000		CMP	#1000,R1		;END AT LOCATION 776
1291	002702	002771			BLT	2\$		
1292	002704	104406			INSTR			;OUTPUT MESSAGE & GET INPUT STRING
1293	002706	015171			MREGAD			;MESSAGE
1294	002710	104410			PARAM			;CONVERT STRING
1295	002712	160000			160000			;LOW LIMIT
1296	002714	177776			177776			;HIGH LIMIT
1297	002716	017122			DUBASE			;STORE AT THIS LOCATION
1298	002720	001			1			;MASK
1299	002721	001			1			;HOW MANY TIMES + 2
1300	002722	016767	014174	176226	MOV	DUBASE,KEEPADD		;SAVE
1301	002730	004767	014034		JSR	PC,DUADDR		
1302	002734	016767	176216	176212	MOV	KEEPADD,BASEADD		;RESTORE FOR ROTATION

1303	002742	104406				INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1304	002744	015156				MVECTO	:MESSAGE
1305	002746	104410				PARAM	:CONVERT STRING
1306	002750	000300				300	:LOW LIMIT
1307	002752	000776				776	:HIGH LIMIT
1308	002754	001736				DURIV	:STORE AT THIS LOCATION
1309	002756	001			.BYTE	1	:MASK
1310	002757	004			.BYTE	4	:HOW MANY TIMES + 2
1311	002760	016767	176752	176176		MOV	DURIV,KEEPV ;SAVE
1312	002766	016767	176744	176166		MOV	DURIV,BASEV ;SET UP FOR ROTATION
1313	002774	104406				INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1314	002776	015221				MMULT	:MESSAGE
1315	003000	104414				SETFLG	:SET FLAG BASED UPON INPUT STRING
1316	003002	001152				MULTD	:THIS FLAG
1317	003004	105767	176142			TSTB	MULTD ;ARE THERE MULTIPLE DEVICES
1318							:ON THE SYSTEM ?
1319	003010	100406				BMI	BBB ;YES,ASK NEXT QUESTION
1320	003012	005067	176150			CLR	ACTREG
1321	003016	005067	176146			CLR	ROTADD
1322	003022	000167	000140			JMP	OUTMUL ;JUMP AROUND NEXT QUESTION
1323	003026				BBB:		
1324	003026	104406				INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1325	003030	015250				MLASTD	:MESSAGE
1326	003032	104410				PARAM	:CONVERT STRING
1327	003034	160000				160000	:LOW LIMIT
1328	003036	177776				177776	:HIGH LIMIT
1329	003040	001160				LASTADD	:STORE AT THIS LOCATION
1330	003042	001			.BYTE	1	:MASK
1331	003043	001			.BYTE	1	:HOW MANY TIMES + 2
1332							:THE FOLLOWING ROUTINE SETS UP ACTREG FOR THE FIRST TIME
1333	003044	012767	000001	176116	1\$:	MOV	#1,ROTADD ;SET UP POINTER
1334	003052	005067	176110			CLR	ACTREG ;CLR ACTIVE REGISTER
1335	003056	056767	176106	176102	2\$:	BIS	ROTADD,ACTREG ;MAKE THIS DEVICE ACTIVE
1336	003064	000241				CLC	
1337	003066	006167	176076			ROL	ROTADD ;SET UP POINTER
1338	003072	103421				BCS	3\$;ARE YOU OUT OF RANGE ?
1339	003074	062767	000010	176052		ADD	#10,BASEADD ;SET UP BASE ADDRESS
1340	003102	026767	176052	176044		CMP	LASTADD,BASEADD ;IS THIS THE LAST DEVICE ?
1341	003110	101362				BHI	2\$;NO DO IT AGAIN
1342	003112	056767	176052	176046		BIS	ROTADD,ACTREG ;THIS ASSUMES THAT THERE ARE AT
1343							:LEAST TWO DEVICES WHEN YOU ANSWER YES TO
1344							:MULTIPLE DEVICE QUESTION
1345	003120	012767	000001	176042	4\$:	MOV	#1,ROTADD ;SET UP FOR LATER USE IN END OF PASS ROUTINE
1346	003126	016767	176024	176020		MOV	KEEPADD,BASEADD ;DITTO
1347	003134	000414				BR	OUTMUL ;CONTINUE QUESTIONS
1348	003136	016767	176014	176010	3\$:	MOV	KEEPADD,BASEADD ;RESTORE
1349	003144	104406				INSTR	:OUTPUT MESSAGE & GET INPUT STRING
1350	003146	015344				MRANGE	:MESSAGE
1351	003150	104410				PARAM	:CONVERT STRING
1352	003152	160000				160000	:LOW LIMIT
1353	003154	177776				177776	:HIGH LIMIT
1354	003156	001160				LASTADD	:STORE AT THIS LOCATION
1355	003160	001			.BYTE	1	:MASK
1356	003161	001			.BYTE	1	:HOW MANY TIMES + 2
1357	003162	000167	177656			JMP	1\$;DO IT AGAIN
1358	003166	012767	000340	013570	OUTMUL:	MOV	#340,DUPRT


```
1359 003174 004767 013514 JSR PC,DULEV
1360 ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
1361 ;BUFFER TO THE CHARACTERS '1' AND '2'.
1362 ;IF THE CHARACTER IS '1' CLEAR THE FLAG
1363 ;IF THE CHARACTER IS '2' SET THE FLAG
1364 003200 AAA: INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1365 003200 104406 MSYNC ;MESSAGE
1366 003202 015562 CMPB #'1,INBUF ;IS IT '1' ?
1367 003204 122767 000061 012712 3$: BNE 1$
1368 003212 001003 CLRB SYNCNO ;000
1369 003214 105067 175726 BR 4$
1370 003220 000412 BR 4$
1371 003222 122767 000062 012674 1$: CMPB #'2,INBUF ;IS IT '2' ?
1372 003230 001004 BNE 2$
1373 003232 112767 177777 175706 MOVB #-1,SYNCNO ;377
1374 003240 000402 BR 4$
1375 003242 104407 2$: INSTER ;RETRY
1376 003244 000757 BR 3$
1377 003246 000240 4$: NOP
1378 003250 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1379 003252 015630 MWIRE6 ;MESSAGE
1380 003254 104414 SETFLG ;SET FLAG BASED UPON INPUT STRING
1381 003256 001147 SEXMIT ;THIS FLAG
1382 003260 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1383 003262 015701 MWIRE5 ;MESSAGE
1384 003264 104414 SETFLG ;SET FLAG BASED UPON INPUT STRING
1385 003266 001150 SEREC ;THIS FLAG
1386 003270 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1387 003272 015751 MWIRE4 ;MESSAGE
1388 003274 104414 SETFLG ;SET FLAG BASED UPON INPUT STRING
1389 003276 001151 OPTCLR ;THIS FLAG
1390 003300 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1391 003302 016030 MEXTJ ;MESSAGE
1392 003304 104414 SETFLG ;SET FLAG BASED UPON INPUT STRING
1393 003306 001153 JMRBY ;THIS FLAG
1394
1395 ;TEST START AND RESTART
1396
1397 003310 012706 001100 .BEGIN: MOV #STACK,SP ;SET UP STACK
1398 003314 106427 000340 MTPS #340 ;LOCK OUT INTERRUPTS
1399 003320 032777 000002 176112 BIT #SW01,@SWR ;IF SW01=1, GET STARTING PC
1400 003326 001413 BEQ 3$
1401 003330 104406 INSTR ;OUTPUT MESSAGE & GET INPUT STRING
1402 003332 015514 MTSTPC ;MESSAGE
1403 003334 104410 PARAM ;CONVERT STRING
1404 003336 003374 TST1 ;LOW LIMIT
1405 003340 017500 17500 ;HIGH LIMIT
1406 003342 001402 $TSTNM ;STORE AT THIS LOCATION
1407 003344 001 .BYTE 1 ;MASK
1408 003345 001 .BYTE 1 ;HOW MANY TIMES + 2
1409 003346 016767 176030 175530 MOV $TSTNM,RETURN
1410 003354 000403 BR 4$
1411 003356 012767 003374 175520 3$: MOV #TST1,RETURN ;START AT TEST 1
1412 003364 104401 015510 4$: TYPE ,MR ;TYPE R
1413 003370 000177 175510 JMP @RETURN ;START TESTING
1414
```

```
1415
1416
1417
1418
1419 003374 000004
1420 003376 012737 017362 000004
1421 003404 016737 174730 000006
1422 003412 105277 176274
1423 003416 000401
1424 003420 104004
1425 003422 105277 176266
1426 003426 000401
1427 003430 104004
1428 003432 012737 000006 000004
1429 003440 012737 000000 000006
1430
1431 003446 012767 006200 175444
1432
1433
1434
1435 003454 000004
1436 003456 012737 017362 000004
1437 003464 016737 174650 000006
1438 003472 105277 176220
1439 003476 000401
1440 003500 104004
1441 003502 105277 176212
1442 003506 000401
1443 003510 104004
1444 003512 012737 000006 000004
1445 003520 012737 000000 000006
1446
1447
1448
1449
1450 003526 000004
1451 003530 012737 017362 000004
1452 003536 016737 174576 000006
1453 003544 105277 176152
1454 003550 000401
1455 003552 104004
1456 003554 105277 176144
1457 003560 000401
1458 003562 104004
1459 003564 012737 000006 000004
1460 003572 012737 000000 000006
1461
1462
1463
1464
1465 003600 000004
1466 003602 012737 017362 000004
1467 003610 016737 174524 000006
1468 003616 105277 176104
1469 003622 000401
1470 003624 104004

;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
;;
*****
TST1: SCOPE
MOV #TRPREG,a#4 ;SETUP TRAPCATCHER
MOV PR7,a#6 ;
INCB @RXCSR ;TEST THIS REG
BR .+4 ;IF OK JMP AROUND ERROR
ERROR 4 ;CHECK DEVICE REG ADDRESSES
INCB @HRXCSR ;TEST UPPER BYTE THIS REGISTER
BR .+4 ;IF OK JMP AROUND ERROR
ERROR 4 ;CHECK DEVICE REG ADDRESSES
MOV #6,a#4 ;RESTORE TRAPCATCHER
MOV #0,a#6 ;
MOV #6200,HOLD ;SET LONGER DELAY FOR TEST. ;;REV. CO
;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
;;
*****
TST2: SCOPE
MOV #TRPREG,a#4 ;SETUP TRAPCATCHER
MOV PR7,a#6 ;
INCB @RXDBUF ;TEST THIS REG
BR .+4 ;IF OK JMP AROUND ERROR
ERROR 4 ;CHECK DEVICE REG ADDRESSES
INCB @HRXDBUF ;TEST UPPER BYTE THIS REGISTER
BR .+4 ;IF OK JMP AROUND ERROR
ERROR 4 ;CHECK DEVICE REG ADDRESSES
MOV #6,a#4 ;RESTORE TRAPCATCHER
MOV #0,a#6 ;
;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
;;
*****
TST3: SCOPE
MOV #TRPREG,a#4 ;SETUP TRAPCATCHER
MOV PR7,a#6 ;
INCB @PARCSR ;TEST THIS REG
BR .+4 ;IF OK JMP AROUND ERROR
ERROR 4 ;CHECK DEVICE REG ADDRESSES
INCB @HPARCSR ;TEST UPPER BYTE THIS REGISTER
BR .+4 ;IF OK JMP AROUND ERROR
ERROR 4 ;CHECK DEVICE REG ADDRESSES
MOV #6,a#4 ;RESTORE TRAPCATCHER
MOV #0,a#6 ;
;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
;;
*****
TST4: SCOPE
MOV #TRPREG,a#4 ;SETUP TRAPCATCHER
MOV PR7,a#6 ;
INCB @TXCSR ;TEST THIS REG
BR .+4 ;IF OK JMP AROUND ERROR
ERROR 4 ;CHECK DEVICE REG ADDRESSES
```



```
1471 003626 105277 176076      INCB    @HTXCSR ;TEST UPPER BYTE THIS REGISTER
1472 003632 000401              BR        .+4          ;IF OK JMP AROUND ERROR
1473 003634 104004              ERROR    4              ;CHECK DEVICE REG ADDRESSES
1474 003636 012737 000006 000004  MOV     #6,@#4          ;RESTORE TRAPCATCHER
1475 003644 012737 000000 000006  MOV     #0,@#6          ;
1476
1477                               ;;THIS TEST PROVES EXISTANCE OF DEVICE REGISTERS
1478                               ;;
1479                               ;*****
1480 003652 000004              TST5:  SCOPE
1481 003654 012737 017362 000004  MOV     #TRPREG,@#4      ;SETUP TRAPCATCHER
1482 003662 016737 174452 000006  MOV     PR7,@#6 ;
1483 003670 105277 176036      INCB    @TXDBUF          ;TEST THIS REG
1484 003674 000401              BR        .+4          ;IF OK JMP AROUND ERROR
1485 003676 104004              ERROR    4              ;CHECK DEVICE REG ADDRESSES
1486 003700 105277 176030      INCB    @HTXDBUF         ;TEST UPPER BYTE THIS REGISTER
1487 003704 000401              BR        .+4          ;IF OK JMP AROUND ERROR
1488 003706 104004              ERROR    4              ;CHECK DEVICE REG ADDRESSES
1489 003710 012737 000006 000004  MOV     #6,@#4          ;RESTORE TRAPCATCHER
1490 003716 012737 000000 000006  MOV     #0,@#6          ;
1491
1492                               ;;BUS DRIVER TEST
1493                               ;;
1494                               ;*****
1495 003724 000004              TST6:  SCOPE
1496 003726 022777 000000 175776  CMP     #0,@TXDBUF
1497 003734 001401              BEQ     .+4
1498 003736 104004              ERROR    4              ;READING TXDBUF SHOULD BE ALL ZERO'S
1499                               ;THIS TEST PERFORMS MASTER RESET TESTING &
1500                               ;TESTING OF READ/WRITE BIT DTR
1501                               ;
1502                               ;*****
1503 003740 000004              TST7:  SCOPE
1504 003742 052777 000002 175742  BIS     #DTR,@RXCSR      ;SET THIS BIT
1505 003750 032777 000002 175734  BIT     #DTR,@RXCSR      ;TEST THIS BIT
1506 003756 001001              BNE     .+4              ;BR IF '1'
1507 003760 104004              ERROR    4              ;THIS BIT SHOULD BE SET
1508 003762 042777 000002 175722  BIC     #DTR,@RXCSR      ;CLR THIS BIT
1509 003770 032777 000002 175714  BIT     #DTR,@RXCSR      ;TEST THIS BIT
1510 003776 001401              BEQ     .+4              ;BR IF '0'
1511 004000 104004              ERROR    4              ;THIS BIT SHOULD BE CLR
1512                               ;NOW SET THIS BIT
1513 004002 052777 000002 175702  BIS     #DTR,@RXCSR
1514 004010 052777 000400 175710  BIS     #MRESET,@TXCSR ;MASTER RESET
1515                               ;CHECK EXISTANCE OF OPTIONAL CLEAR JUMPER
1516                               ;
1517 004016 105767 175127      TSTB    OPTCLR          ;TEST FLAG
1518 004022 100006      BPL        1$              ;OPTIONAL CLR JUMPER IS NOT IN
1519 004024 032777 000002 175660  BIT     #DTR,@RXCSR      ;TEST THIS BIT
1520 004032 001401              BEQ     .+4              ;BR IF '0'
1521 004034 104004              ERROR    4              ;CHECK OUT MASTER RESET LOGIC
1522 004036 000405      BR        2$              ;JMP AROUND
1523 004040 032777 000002 175644 1$:  BIT     #DTR,@RXCSR      ;TEST THIS BIT
1524 004046 001001              BNE     .+4              ;BR IF '1'
1525 004050 104004              ERROR    4              ;CHECK OUT OPTIONAL CLR JUMPER
1526 004052 000240      2$:  NOP
```

```
1527
1528
1529
1530
1531
1532 004054 000004
1533 004056 052777 000004 175626
1534 004064 032777 000004 175620
1535 004072 001001
1536 004074 104004
1537 004076 042777 000004 175606
1538 004104 032777 000004 175600
1539 004112 001401
1540 004114 104004
1541
1542 004116 052777 000004 175566
1543 004124 052777 000400 175574
1544
1545
1546 004132 105767 175013
1547 004136 100006
1548 004140 032777 000004 175544
1549 004146 001401
1550 004150 104004
1551 004152 000405
1552 004154 032777 000004 175530 1$:
1553 004162 001001
1554 004164 104004
1555 004166 000240 2$:
1556
1557
1558
1559
1560
1561 004170 016702 174724
1562 004174 005302
1563 004176 001376
1564
1565
1566
1567
1568
1569
1570 004200 000004
1571 004202 052777 000010 175502
1572 004210 032777 000010 175474
1573 004216 001001
1574 004220 104004
1575 004222 042777 000010 175462
1576 004230 032777 000010 175454
1577 004236 001401
1578 004240 104004
1579
1580 004242 052777 000010 175442
1581 004250 052777 000400 175450
1582

::THIS TEST PERFORMS MASTER RESET TESTING &
::TESTING OF READ/WRITE BIT RTS
::
::*****
TST10: SCOPE
BIS #RTS,@RXCSR ;SET THIS BIT
BIT #RTS,@RXCSR ;TEST THIS BIT
BNE .+4 ;BR IF "1"
ERROR 4 ;THIS BIT SHOULD BE SET
BIC #RTS,@RXCSR ;CLR THIS BIT
BIT #RTS,@RXCSR ;TEST THIS BIT
BEQ .+4 ;BR IF "0"
ERROR 4 ;THIS BIT SHOULD BE CLR
;NOW SET THIS BIT
BIS #RTS,@RXCSR
BIS #MRESET,@TXCSR ;MASTER RESET
::CHECK EXISTANCE OF OPTIONAL CLEAR JUMPER
::
TSTB OPTCLR ;TEST FLAG
BPL 1$ ;OPTIONAL CLR JUMPER IS NOT IN
BIT #RTS,@RXCSR ;TEST THIS BIT
BEQ .+4 ;BR IF "0"
ERROR 4 ;CHECK OUT MASTER RESET LOGIC
BR 2$ ;JMP AROUND
BIT #RTS,@RXCSR ;TEST THIS BIT
BNE .+4 ;BR IF "1"
ERROR 4 ;CHECK OUT OPTIONAL CLR JUMPER
NOP

:WAIT FOR CABLE DELAYS
:*****
:MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
:*****
MOV HOLD,R2 ;SET DELAY TIME
DEC R2
BNE .-2 ;WAIT THIS TIME
;OK NOW FALL THRU AND CONTINUE TESTING.....
;EXIT STAGE LEFT....CHINNG!
::THIS TEST PERFORMS MASTER RESET TESTING &
::TESTING OF READ/WRITE BIT STD
::
::*****
TST11: SCOPE
BIS #STD,@RXCSR ;SET THIS BIT
BIT #STD,@RXCSR ;TEST THIS BIT
BNE .+4 ;BR IF "1"
ERROR 4 ;THIS BIT SHOULD BE SET
BIC #STD,@RXCSR ;CLR THIS BIT
BIT #STD,@RXCSR ;TEST THIS BIT
BEQ .+4 ;BR IF "0"
ERROR 4 ;THIS BIT SHOULD BE CLR
;NOW SET THIS BIT
BIS #STD,@RXCSR
BIS #MRESET,@TXCSR ;MASTER RESET
::CHECK EXISTANCE OF OPTIONAL CLEAR JUMPER
```



```
1583
1584 004256 105767 174667      ;:
1585 004262 100006      ;:TSTB OPTCLR ;TEST FLAG
1586 004264 032777 000010 175420 ;BPL 1$ ;OPTIONAL CLR JUMPER IS NOT IN
1587 004272 001401      ;BIT #STD,@RXCSR ;TEST THIS BIT
1588 004274 104004      ;BEQ .+4 ;BR IF "0"
1589 004276 000405      ;ERROR 4 ;CHECK OUT MASTER RESET LOGIC
1590 004300 032777 000010 175404 1$: ;BR 2$ ;JMP AROUND
1591 004306 001001      ;BIT #STD,@RXCSR ;TEST THIS BIT
1592 004310 104004      ;BNE .+4 ;BR IF "1"
1593 004312 000240      ;ERROR 4 ;CHECK OUT OPTIONAL CLR JUMPER
1594
1595      ;:THIS TEST PERFORMS MASTER RESET TESTING &
1596      ;:TESTING OF READ/WRITE BIT SYNSCH
1597      ;:
1598      ;:*****
1599 004314 000004      TST12: SCOPE
1600 004316 052777 000020 175366 ;BIS #SYNSCH,@RXCSR ;SET THIS BIT
1601 004324 032777 000020 175360 ;BIT #SYNSCH,@RXCSR ;TEST THIS BIT
1602 004332 001001      ;BNE .+4 ;BR IF "1"
1603 004334 104004      ;ERROR 4 ;THIS BIT SHOULD BE SET
1604 004336 042777 000020 175346 ;BIC #SYNSCH,@RXCSR ;CLR THIS BIT
1605 004344 032777 000020 175340 ;BIT #SYNSCH,@RXCSR ;TEST THIS BIT
1606 004352 001401      ;BEQ .+4 ;BR IF "0"
1607 004354 104004      ;ERROR 4 ;THIS BIT SHOULD BE CLR
1608      ;NOW SET THIS BIT
1609 004356 052777 000020 175326 ;BIS #SYNSCH,@RXCSR
1610 004364 052777 000400 175334 ;BIS #MRESET,@TXCSR ;MASTER RESET
1611 004372 032777 000020 175312 ;BIT #SYNSCH,@RXCSR ;TEST THIS BIT
1612 004400 001401      ;BEQ .+4 ;BR IF "0"
1613 004402 104004      ;ERROR 4 ;CHECK OUT MASTER RESET LOGIC
1614
1615      ;:THIS TEST PERFORMS MASTER RESET TESTING &
1616      ;:TESTING OF READ/WRITE BIT DSINTE
1617      ;:
1618      ;:*****
1619 004404 000004      TST13: SCOPE
1620 004406 052777 000040 175276 ;BIS #DSINTE,@RXCSR ;SET THIS BIT
1621 004414 032777 000040 175270 ;BIT #DSINTE,@RXCSR ;TEST THIS BIT
1622 004422 001001      ;BNE .+4 ;BR IF "1"
1623 004424 104004      ;ERROR 4 ;THIS BIT SHOULD BE SET
1624 004426 042777 000040 175256 ;BIC #DSINTE,@RXCSR ;CLR THIS BIT
1625 004434 032777 000040 175250 ;BIT #DSINTE,@RXCSR ;TEST THIS BIT
1626 004442 001401      ;BEQ .+4 ;BR IF "0"
1627 004444 104004      ;ERROR 4 ;THIS BIT SHOULD BE CLR
1628      ;NOW SET THIS BIT
1629 004446 052777 000040 175236 ;BIS #DSINTE,@RXCSR
1630 004454 052777 000400 175244 ;BIS #MRESET,@TXCSR ;MASTER RESET
1631 004462 032777 000040 175222 ;BIT #DSINTE,@RXCSR ;TEST THIS BIT
1632 004470 001401      ;BEQ .+4 ;BR IF "0"
1633 004472 104004      ;ERROR 4 ;CHECK OUT MASTER RESET LOGIC
1634
1635      ;:THIS TEST PERFORMS MASTER RESET TESTING &
1636      ;:TESTING OF READ/WRITE BIT RINTEN
1637      ;:
1638      ;:*****
```

```
1639 004474 000004
1640 004476 052777 000100 175206
1641 004504 032777 000100 175200
1642 004512 001001
1643 004514 104004
1644 004516 042777 000100 175166
1645 004524 032777 000100 175160
1646 004532 001401
1647 004534 104004
1648
1649 004536 052777 000100 175146
1650 004544 052777 000400 175154
1651 004552 032777 000100 175132
1652 004560 001401
1653 004562 104004
1654
1655
1656
1657
1658
1659 004564 000004
1660 004566 052777 000400 175116
1661 004574 032777 000400 175110
1662 004602 001001
1663 004604 104004
1664 004606 042777 000400 175076
1665 004614 032777 000400 175070
1666 004622 001401
1667 004624 104004
1668
1669 004626 052777 000400 175056
1670 004634 052777 000400 175064
1671 004642 032777 000400 175042
1672 004650 001401
1673 004652 104004
1674
1675
1676
1677
1678
1679 004654 000004
1680 004656 052777 000001 175042
1681 004664 032777 000001 175034
1682 004672 001001
1683 004674 104004
1684 004676 042777 000001 175022
1685 004704 032777 000001 175014
1686 004712 001401
1687 004714 104004
1688
1689 004716 052777 000001 175002
1690 004724 052777 000400 174774
1691 004732 032777 000001 174766
1692 004740 001401
1693 004742 104004
1694
```

```
TST14: SCOPE
BIS #RINTEN,@RXCSR ;SET THIS BIT
BIT #RINTEN,@RXCSR ;TEST THIS BIT
BNE .+4 ;BR IF "1"
ERROR 4 ;THIS BIT SHOULD BE SET
BIC #RINTEN,@RXCSR ;CLR THIS BIT
BIT #RINTEN,@RXCSR ;TEST THIS BIT
BEQ .+4 ;BR IF "0"
ERROR 4 ;THIS BIT SHOULD BE CLR
;NOW SET THIS BIT
BIS #RINTEN,@RXCSR ;MASTER RESET
BIS #MRESET,@TXCSR ;TEST THIS BIT
BIT #RINTEN,@RXCSR ;BR IF "0"
BEQ .+4 ;CHECK OUT MASTER RESET LOGIC
ERROR 4

;;THIS TEST PERFORMS MASTER RESET TESTING &
;;TESTING OF READ/WRITE BIT STPSYN
;;
*****
TST15: SCOPE
BIS #STPSYN,@RXCSR ;SET THIS BIT
BIT #STPSYN,@RXCSR ;TEST THIS BIT
BNE .+4 ;BR IF "1"
ERROR 4 ;THIS BIT SHOULD BE SET
BIC #STPSYN,@RXCSR ;CLR THIS BIT
BIT #STPSYN,@RXCSR ;TEST THIS BIT
BEQ .+4 ;BR IF "0"
ERROR 4 ;THIS BIT SHOULD BE CLR
;NOW SET THIS BIT
BIS #STPSYN,@RXCSR ;MASTER RESET
BIS #MRESET,@TXCSR ;TEST THIS BIT
BIT #STPSYN,@RXCSR ;BR IF "0"
BEQ .+4 ;CHECK OUT MASTER RESET LOGIC
ERROR 4

;;THIS TEST PERFORMS MASTER RESET TESTING &
;;TESTING OF READ/WRITE BIT BREAK
;;
*****
TST16: SCOPE
BIS #BREAK,@TXCSR ;SET THIS BIT
BIT #BREAK,@TXCSR ;TEST THIS BIT
BNE .+4 ;BR IF "1"
ERROR 4 ;THIS BIT SHOULD BE SET
BIC #BREAK,@TXCSR ;CLR THIS BIT
BIT #BREAK,@TXCSR ;TEST THIS BIT
BEQ .+4 ;BR IF "0"
ERROR 4 ;THIS BIT SHOULD BE CLR
;NOW SET THIS BIT
BIS #BREAK,@TXCSR ;MASTER RESET
BIS #MRESET,@TXCSR ;TEST THIS BIT
BIT #BREAK,@TXCSR ;BR IF "0"
BEQ .+4 ;CHECK OUT MASTER RESET LOGIC
ERROR 4
```



```
1695      ::THIS TEST PERFORMS MASTER RESET TESTING &
1696      ::TESTING OF READ/WRITE BIT HDXEN
1697      ::
1698      ::*****
1699      TST17: SCOPE
1700      BIS      #HDXEN,@TXCSR      ;SET THIS BIT
1701      BIT      #HDXEN,@TXCSR      ;TEST THIS BIT
1702      BNE      .+4      ;BR IF '1'
1703      ERROR    4      ;THIS BIT SHOULD BE SET
1704      BIC      #HDXEN,@TXCSR      ;CLR THIS BIT
1705      BIT      #HDXEN,@TXCSR      ;TEST THIS BIT
1706      BEQ      .+4      ;BR IF '0'
1707      ERROR    4      ;THIS BIT SHOULD BE CLR
1708      ;NOW SET THIS BIT
1709      BIS      #HDXEN,@TXCSR
1710      BIS      #MRESET,@TXCSR      ;MASTER RESET
1711      BIT      #HDXEN,@TXCSR      ;TEST THIS BIT
1712      BEQ      .+4      ;BR IF '0'
1713      ERROR    4      ;CHECK OUT MASTER RESET LOGIC
1714
1715      ::THIS TEST PERFORMS MASTER RESET TESTING &
1716      ::TESTING OF READ/WRITE BIT SEND
1717      ::
1718      ::*****
1719      TST20: SCOPE
1720      BIS      #SEND,@TXCSR      ;SET THIS BIT
1721      BIT      #SEND,@TXCSR      ;TEST THIS BIT
1722      BNE      .+4      ;BR IF '1'
1723      ERROR    4      ;THIS BIT SHOULD BE SET
1724      BIC      #SEND,@TXCSR      ;CLR THIS BIT
1725      BIT      #SEND,@TXCSR      ;TEST THIS BIT
1726      BEQ      .+4      ;BR IF '0'
1727      ERROR    4      ;THIS BIT SHOULD BE CLR
1728      ;NOW SET THIS BIT
1729      BIS      #SEND,@TXCSR
1730      BIS      #MRESET,@TXCSR      ;MASTER RESET
1731      BIT      #SEND,@TXCSR      ;TEST THIS BIT
1732      BEQ      .+4      ;BR IF '0'
1733      ERROR    4      ;CHECK OUT MASTER RESET LOGIC
1734
1735      ::THIS TEST PERFORMS MASTER RESET TESTING &
1736      ::TESTING OF READ/WRITE BIT DNAINTE
1737      ::
1738      ::*****
1739      TST21: SCOPE
1740      BIS      #DNAINTE,@TXCSR      ;SET THIS BIT
1741      BIT      #DNAINTE,@TXCSR      ;TEST THIS BIT
1742      BNE      .+4      ;BR IF '1'
1743      ERROR    4      ;THIS BIT SHOULD BE SET
1744      BIC      #DNAINTE,@TXCSR      ;CLR THIS BIT
1745      BIT      #DNAINTE,@TXCSR      ;TEST THIS BIT
1746      BEQ      .+4      ;BR IF '0'
1747      ERROR    4      ;THIS BIT SHOULD BE CLR
1748      ;NOW SET THIS BIT
1749      BIS      #DNAINTE,@TXCSR
1750      BIS      #MRESET,@TXCSR      ;MASTER RESET
```

```
1751 005202 032777 000040 174516 BIT #DNAINTE,@TXCSR ;TEST THIS BIT
1752 005210 001401 BEQ .+4 ;BR IF "0"
1753 005212 104004 ERROR 4 ;CHECK OUT MASTER RESET LOGIC
1754
1755 ;:THIS TEST PERFORMS MASTER RESET TESTING &
1756 ;:TESTING OF READ/WRITE BIT TXINTE
1757 ;:
1758 ;:*****
1759 005214 000004 TST22: SCOPE
1760 005216 052777 000100 174502 BIS #TXINTE,@TXCSR ;SET THIS BIT
1761 005224 032777 000100 174474 BIT #TXINTE,@TXCSR ;TEST THIS BIT
1762 005232 001001 BNE .+4 ;BR IF "1"
1763 005234 104004 ERROR 4 ;THIS BIT SHOULD BE SET
1764 005236 042777 000100 174462 BIC #TXINTE,@TXCSR ;CLR THIS BIT
1765 005244 032777 000100 174454 BIT #TXINTE,@TXCSR ;TEST THIS BIT
1766 005252 001401 BEQ .+4 ;BR IF "0"
1767 005254 104004 ERROR 4 ;THIS BIT SHOULD BE CLR
1768 ;NOW SET THIS BIT
1769 005256 052777 000100 174442 BIS #TXINTE,@TXCSR
1770 005264 052777 000400 174434 BIS #MRESET,@TXCSR ;MASTER RESET
1771 005272 032777 000100 174426 BIT #TXINTE,@TXCSR ;TEST THIS BIT
1772 005300 001401 BEQ .+4 ;BR IF "0"
1773 005302 104004 ERROR 4 ;CHECK OUT MASTER RESET LOGIC
1774
1775 ;:TEST MAINT MODE BIT 0
1776 ;:
1777 ;:THIS TEST PERFORMS MASTER RESET TESTING &
1778 ;:TESTING OF READ/WRITE BIT BIT11
1779 ;:
1780 ;:*****
1781 005304 000004 TST23: SCOPE
1782 005306 052777 004000 174412 BIS #BIT11,@TXCSR ;SET THIS BIT
1783 005314 032777 004000 174404 BIT #BIT11,@TXCSR ;TEST THIS BIT
1784 005322 001001 BNE .+4 ;BR IF "1"
1785 005324 104004 ERROR 4 ;THIS BIT SHOULD BE SET
1786 005326 042777 004000 174372 BIC #BIT11,@TXCSR ;CLR THIS BIT
1787 005334 032777 004000 174364 BIT #BIT11,@TXCSR ;TEST THIS BIT
1788 005342 001401 BEQ .+4 ;BR IF "0"
1789 005344 104004 ERROR 4 ;THIS BIT SHOULD BE CLR
1790 ;NOW SET THIS BIT
1791 005346 052777 004000 174352 BIS #BIT11,@TXCSR
1792 005354 052777 000400 174344 BIS #MRESET,@TXCSR ;MASTER RESET
1793 005362 032777 004000 174336 BIT #BIT11,@TXCSR ;TEST THIS BIT
1794 005370 001401 BEQ .+4 ;BR IF "0"
1795 005372 104004 ERROR 4 ;CHECK OUT MASTER RESET LOGIC
1796
1797 ;:TEST MAINT MODE BIT 1
1798 ;:
1799 ;:THIS TEST PERFORMS MASTER RESET TESTING &
1800 ;:TESTING OF READ/WRITE BIT BIT12
1801 ;:
1802 ;:*****
1803 005374 000004 TST24: SCOPE
1804 005376 052777 010000 174322 BIS #BIT12,@TXCSR ;SET THIS BIT
1805 005404 032777 010000 174314 BIT #BIT12,@TXCSR ;TEST THIS BIT
1806 005412 001001 BNE .+4 ;BR IF "1"
```



```
1807 005414 104004 ERROR 4 ;THIS BIT SHOULD BE SET
1808 005416 042777 010000 174302 BIC #BIT12,@TXCSR ;CLR THIS BIT
1809 005424 032777 010000 174274 BIT #BIT12,@TXCSR ;TEST THIS BIT
1810 005432 001401 BEQ +4 ;BR IF '0'
1811 005434 104004 ERROR 4 ;THIS BIT SHOULD BE CLR
1812 ;NOW SET THIS BIT
1813 005436 052777 010000 174262 BIS #BIT12,@TXCSR
1814 005444 052777 000400 174254 BIS #MRESET,@TXCSR ;MASTER RESET
1815 005452 032777 010000 174246 BIT #BIT12,@TXCSR ;TEST THIS BIT
1816 005460 001401 BEQ +4 ;BR IF '0'
1817 005462 104004 ERROR 4 ;CHECK OUT MASTER RESET LOGIC
1818
1819 ::THIS TEST PERFORMS MASTER RESET TESTING &
1820 ::TESTING OF READ/WRITE BIT CLK
1821 ::
1822 *****
1823 TST25: SCOPE
1824 005464 000004 BIS #CLK,@TXCSR ;SET THIS BIT
1825 005466 052777 020000 174232 BIT #CLK,@TXCSR ;TEST THIS BIT
1826 005474 032777 020000 174224 BNE +4 ;BR IF '1'
1827 005502 001001 ERROR 4 ;THIS BIT SHOULD BE SET
1828 005504 104004 BIC #CLK,@TXCSR ;CLR THIS BIT
1829 005506 042777 020000 174212 BIT #CLK,@TXCSR ;TEST THIS BIT
1830 005514 032777 020000 174204 BEQ +4 ;BR IF '0'
1831 005522 001401 ERROR 4 ;THIS BIT SHOULD BE CLR
1832 005524 104004 ;NOW SET THIS BIT
1833 005526 052777 020000 174172 BIS #CLK,@TXCSR
1834 005534 052777 000400 174164 BIS #MRESET,@TXCSR ;MASTER RESET
1835 005542 032777 020000 174156 BIT #CLK,@TXCSR ;TEST THIS BIT
1836 005550 001401 BEQ +4 ;BR IF '0'
1837 005552 104004 ERROR 4 ;CHECK OUT MASTER RESET LOGIC
1838
1839 ::THIS TEST PERFORMS MASTER RESET TESTING &
1840 ::TESTING OF READ/WRITE BIT MTDATA
1841 ::
1842 *****
1843 TST26: SCOPE
1844 005554 000004 BIS #MTDATA,@TXCSR ;SET THIS BIT
1845 005556 052777 040000 174142 BIT #MTDATA,@TXCSR ;TEST THIS BIT
1846 005564 032777 040000 174134 BNE +4 ;BR IF '1'
1847 005572 001001 ERROR 4 ;THIS BIT SHOULD BE SET
1848 005574 104004 BIC #MTDATA,@TXCSR ;CLR THIS BIT
1849 005576 042777 040000 174122 BIT #MTDATA,@TXCSR ;TEST THIS BIT
1850 005604 032777 040000 174114 BEQ +4 ;BR IF '0'
1851 005612 001401 ERROR 4 ;THIS BIT SHOULD BE CLR
1852 005614 104004 ;NOW SET THIS BIT
1853 005616 052777 040000 174102 BIS #MTDATA,@TXCSR
1854 005624 052777 000400 174074 BIS #MRESET,@TXCSR ;MASTER RESET
1855 005632 032777 040000 174066 BIT #MTDATA,@TXCSR ;TEST THIS BIT
1856 005640 001401 BEQ +4 ;BR IF '0'
1857 005642 104004 ERROR 4 ;CHECK OUT MASTER RESET LOGIC
1858
1859 ::THIS TEST VERIFYS THAT INIT (RESET) CLEARS BITS IN THE
1860 ::RXCSR & TXCSR
1861 ::
1862 *****
```

```
1863 005644 000004 TST27: SCOPE
1864 005646 012777 177777 174036 MOV #177777,@RXCSR ;SET ALL POSSIBLE BITS
1865 005654 012777 177777 174044 MOV #177777,@TXCSR ;DITTO
1866 005662 000005 RESET
1867 005664 106427 000340 MTPS #340 ;RESTORE NON INTERRUPT STATUS
1868 005670 017701 174016 MOV @RXCSR,R1 ;SAVE
1869 005674 017702 174026 MOV @TXCSR,R2 ;SAVE
1870 005700 105767 173245 TSTB OPTCLR ;IS THE OPTIONAL CLR JUMPER ON ?
1871 005704 100402 BMI 1$ ;YES
1872 005706 042701 000016 BIC #16,R1 ;CLR THE NON RESETABLE BITS
1873 005712 042701 073000 1$: BIC #073000,R1 ;CLR ALL NON-CLEARABLE BITS
1874 005716 005701 TST R1 ;ARE THEY ALL 0 ?
1875 005720 001401 BEQ .+4
1876 005722 104004 ERROR 4 ;ALL SPECIFIED BITS SHOULD BE CLEAR
1877 005724 042702 002200 BIC #002200,R2 ;CLEAR ALL NON-CLEARABLE BITS
1878 005730 005702 TST R2 ;ARE THEY ALL 0 ?
1879 005732 001401 BEQ .+4
1880 005734 104004 ERROR 4 ;ALL SPECIFIED BITS SHOULD BE CLEAR
1881 ;WAIT FOR CABLE DELAYS
1882 *****
1883 ;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
1884 *****
1885 005736 016702 173156 MOV HOLD,R2 ;SET DELAY TIME
1886 005742 005302 DEC R2
1887 005744 001376 BNE .-2 ;WAIT THIS TIME
1888 ;OK NOW FALL THRU AND CONTINUE TESTING.....
1889 ;EXIT STAGE LEFT....CHINNG!
1890
1891 ;;THIS TEST PERFORMS MASTER RESET TESTING &
1892 ;;TESTING OF WRITE ONLY BIT MRESET
1893 ;;
1894 *****
1895 005746 000004 TST30: SCOPE
1896 005750 052777 000400 173750 BIS #MRESET,@TXCSR ;TRY TO SET THIS BIT
1897 005756 032777 000400 173742 BIT #MRESET,@TXCSR ;TEST THIS BIT
1898 005764 001401 BEQ .+4 ;BR IF '0'
1899 005766 104004 ERROR 4 ;THIS BIT SHOULD NOT BE SET
1900 005770 052777 000400 173730 BIS #MRESET,@TXCSR ;MASTER RESET
1901 005776 032777 000400 173722 BIT #MRESET,@TXCSR ;TEST THIS BIT
1902 006004 001401 BEQ .+4 ;BR IF '0'
1903 006006 104004 ERROR 4 ;THIS BIT SHOULD NOT BE SET
1904 ;CHECK MASTER RESET LOGIC
1905
1906 ;;THIS TEST VERIFYS THAT THE RXCSR & TXCSR CAN BE BYTE ADDRESSED (DATOB)
1907 ;;
1908 *****
1909 006010 000004 TST31: SCOPE
1910 006012 052777 000400 173706 BIS #MRESET,@TXCSR ;MASTER RESET
1911 006020 105767 173125 TSTB OPTCLR ;IS THE OPTIONAL CLR JUMPER ON ?
1912 006024 100405 BMI 1$ ;YES
1913 006026 012777 000000 173656 MOV #0,@RXCSR ;CLR OUT NON RESETABLE BITS
1914 006034 005777 173652 TST @RXCSR ;CLR OUT DSC BY READING RXCSR
1915 006040 152777 000001 173646 1$: BISB #BIT0,@RXCSR ;SET STRIP SYNC UPPER BYTE
1916 006046 017701 173640 MOV @RXCSR,R1 ;SAVE RXCSR
1917 006052 022701 000400 CMP #400,R1 ;TEST RXCSR
1918 006056 001401 BEQ .+4
```



```
1919 006060 104004          ERROR 4 ;ONLY STRIP SYNC SHOULD BE SET
1920 006062 105077 173624    CLRB @RXCSR ;CLR LOWER BYTE
1921 006066 017701 173620    MOV @RXCSR,R1 ;SAVE RXCSR
1922 006072 022701 000400    CMP #400,R1 ;TEST RXCSR
1923 006076 001401          BEQ .+4
1924 006100 104004          ERROR 4 ;ONLY STRIP SYNC SHOULD BE SET
1925 006102 052777 000400 173616 BIS #MRESET,@TXCSR ;MASTER RESET
1926 006110 152777 000040 173612 BISB #BIT5,@MTXCSR ;SET MAINT CLK UPPER BYTE
1927 006116 017701 173604    MCV @TXCSR,R1 ;SAVE TXCSR
1928 006122 042701 002000    BIC #BITW,R1 ;CLR BIT WINDOW (DEPENDENT
1929                                ;ON H315 CONNECTOR EXISTANCE)
1930 006126 022701 020200    CMP #20200,R1 ;TEST TXCSR
1931 006132 001401          BEQ .+4
1932 006134 104004          ERROR 4 ;ONLY MAINT CLK BIT & TXDONE SHOULD BE SET
1933 006136 105077 173564    CLRB @TXCSR ;CLR LOWER BYTE
1934 006142 017701 173560    MOV @TXCSR,R1 ;SAVE TXCSR
1935 006146 042701 002000    BIC #BITW,R1 ;CLR BIT WINDOW (DITTO)
1936 006152 022701 020200    CMP #20200,R1 ;TEST TXCSR
1937 006156 001401          BEQ .+4
1938 006160 104004          ERROR 4 ;ONLY MAINT CLK BIT & TXDONE SHOULD BE SET
1939                                ;:THIS TEST PERFORMS MASTER RESET TESTING &
1940                                ;:TESTING OF READ ONLY BIT BITW
1941                                ;:MAINT INTERNAL
1942                                ;:
1943                                ;:*****
1944 006162 000004          TST32: SCOPE
1945 006164 012777 044001 173534 MOV #MINT!MTDATA!BREAK,@TXCSR ;SET MAINT INT.,BREAK,
1946                                ;&MTDATA
1947 006172 032777 002000 173526 BIT #BITW,@TXCSR ;TEST BITW
1948 006200 001001          BNE .+4
1949 006202 104004          ERROR 4 ;BIT WINDOW SHOULD BE SET
1950 006204 042777 040000 173514 BIC #MTDATA,@TXCSR
1951 006212 013702 001132    MOV @#1132,R2
1952 006216 005302          1$: DEC R2
1953 006220 001376          BNE 1$
1954 006222 032777 002000 173476 BIT #BITW,@TXCSR
1955 006230 001401          BEQ .+4
1956 006232 104004          ERROR 4 ;BIT SHOULD BE CLR
1957                                ;:NOW SET THE MTDATA
1958 006234 052777 040000 173464 BIS #MTDATA,@TXCSR
1959 006242 052777 000400 173456 BIS #MRESET,@TXCSR ;MASTER RESET
1960 006250 052777 004001 173450 BIS #MINT!BREAK,@TXCSR
1961 006256 013702 001132    MOV @#1132,R2
1962 006262 005302          2$: DEC R2
1963 006264 001376          BNE 2$
1964 006266 032777 002000 173432 BIT #BITW,@TXCSR
1965 006274 001401          BEQ .+4
1966 006276 104004          ERROR 4 ;BITW SHOULD BE CLR BY MASTER RESET
1967                                ;:THIS TEST PERFORMS MASTER RESET TESTING &
1968                                ;:TESTING OF READ ONLY BIT BITW
1969                                ;:MAINT EXTERNAL
1970                                ;:
1971                                ;:*****
1972 006300 000004          TST33: SCOPE
1973                                ;:TEST TO SEE IF EXTERNAL MODEM BYPASS CONNECTOR
1974                                ;:IS ON (H315)....IF "NO" JUMP AROUND TEST
```

```
1975 006302 105767 172645 TSTB JMRBY
1976 006306 100036 BPL 1$ ;IT IS NOT ON
1977 006310 012777 050001 173410 MOV #MEXT!MTDATA!BREAK,@TXCSR ;SET MAINT EXT.,BREAK,
1978 ;BMTDATA
1979 006316 032777 002000 173402 BIT #BITW,@TXCSR ;TEST BITW
1980 006324 001001 BNE .+4
1981 006326 104004 ERROR 4 ;BIT WINDOW SHOULD BE SET
1982 006330 042777 040000 173370 BIC #MTDATA,@TXCSR
1983 006336 032777 002000 173362 BIT #BITW,@TXCSR
1984 006344 001401 BEQ .+4
1985 006346 104004 ERROR 4 ;BIT SHOULD BE CLR
1986 ;NOW SET THE MTDATA
1987 006350 052777 040000 173350 BIS #MTDATA,@TXCSR
1988 006356 052777 000400 173342 BIS #MRESET,@TXCSR ;MASTER RESET
1989 006364 052777 010001 173334 BIS #MEXT!BREAK,@TXCSR
1990 006372 032777 002000 173326 BIT #BITW,@TXCSR
1991 006400 001401 BEQ .+4
1992 006402 104004 ERROR 4 ;BITW SHOULD BE CLR BY MASTER RESET
1993 006404 1$:
1994
1995
1996 ::THIS TEST PERFORMS MASTER RESET TESTING &
1997 ::TESTING OF READ ONLY BIT RXDONE
1998 ::
1999 ::*****
2000 TST34: SCOPE
2001 006406 052777 000400 173312 BIS #MRESET,@TXCSR ;MASTER RESET
2002 006414 032777 000200 173270 BIT #RXDONE,@RXCSR ;TEST THIS BIT
2003 006422 001401 BEQ .+4 ;BR IF "0"
2004 006424 104004 ERROR 4 ;CHECK MASTER RESET LOGIC
2005 ;OR SHORT ON THIS BIT
2006
2007 ::THIS TEST PERFORMS MASTER RESET TESTING &
2008 ::TESTING OF READ ONLY BIT RECACT
2009 ::
2010 ::*****
2011 TST35: SCOPE
2012 006426 000004
2013 006430 052777 000400 173270 BIS #MRESET,@TXCSR ;MASTER RESET
2014 006436 032777 004000 173246 BIT #RECACT,@RXCSR ;TEST THIS BIT
2015 006444 001401 BEQ .+4 ;BR IF "0"
2016 006446 104004 ERROR 4 ;CHECK MASTER RESET LOGIC
2017 ;OR SHORT ON THIS BIT
2018
2019 ::THIS TEST PERFORMS MASTER RESET TESTING &
2020 ::TESTING OF READ ONLY BIT DSC
2021 ::
2022 ::*****
2023 TST36: SCOPE
2024 006450 000004
2025 006452 052777 000400 173246 BIS #MRESET,@TXCSR ;MASTER RESET
2026 006460 032777 100000 173224 BIT #DSC,@RXCSR ;TEST THIS BIT
2027 006466 001401 BEQ .+4 ;BR IF "0"
2028 006470 104004 ERROR 4 ;CHECK MASTER RESET LOGIC
2029 ;OR SHORT ON THIS BIT
2030
2031 ::THIS TEST PERFORMS MASTER RESET TESTING &
2032 ::TESTING OF READ ONLY BIT TXDONE
```



```
2031
2032
2033 006472 000004
2034 006474 052777 000400 173224
2035 006502 032777 000200 173216
2036 006510 001001
2037 006512 104004
2038
2039
2040
2041
2042
2043 006514 000004
2044 006516 052777 000400 173202
2045 006524 032777 100000 173174
2046 006532 001401
2047 006534 104004
2048
2049
2050
2051
2052
2053
2054 006536 000004
2055 006540 052777 000400 173160
2056 006546 016703 173144
2057 006552 012700 000377
2058 006556 017701 173134
2059 006562 120C01
2060 006564 001401
2061 006566 104002
2062
2063
2064
2065
2066 006570 000004
2067 006572 052777 000400 173126
2068 006600 032777 010000 173110
2069 006606 001401
2070 006610 104004
2071
2072
2073
2074
2075
2076
2077 006612 000004
2078 006614 052777 000400 173104
2079 006622 032777 020000 173066
2080 006630 001401
2081 006632 104004
2082
2083
2084
2085
2086

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

```
2087
2088 006634 000004
2089 006636 052777 000400 173062
2090 006644 032777 040000 173044
2091 006652 001401
2092 006654 104004
2093
2094
2095
2096
2097
2098
2099 006656 000004
2100 006660 052777 000400 173040
2101 006666 032777 100000 173022
2102 006674 001401
2103 006676 104004
2104
2105
2106
2107
2108
2109 006700 000004
2110 006702 012777 177777 173002
2111 006710 052777 000400 173010
2112 006716 016703 172770
2113 006722 017701 172764
2114 006726 105767 172217
2115 006732 100010
2116 006734 042701 173000
2117
2118 006740 012700 000000
2119 006744 020001
2120 006746 001401
2121 006750 104001
2122 006752 000407
2123 006754 042701 173000
2124
2125 006760 012700 000016
2126 006764 020001
2127 006766 001401
2128 006770 104001
2129
2130
2131 006772
2132
2133
2134
2135
2136 006772 016702 172122
2137 006776 005302
2138 007000 001376
2139
2140
2141
2142

*****
TST44: SCOPE
BIS #MRESET,@TXCSR ;MASTER RESET
BIT #OVRRUN,@RXDBUF ;TEST THIS BIT
BEQ .+4 ;BR IF '0'
ERROR 4 ;CHECK MASTER RESET LOGIC
;OR SHORT ON THIS BIT

;;THIS TEST PERFORMS MASTER RESET TESTING &
;;TESTING OF READ ONLY BIT RXERR
;;
*****
TST45: SCOPE
BIS #MRESET,@TXCSR ;MASTER RESET
BIT #RXERR,@RXDBUF ;TEST THIS BIT
BEQ .+4 ;BR IF '0'
ERROR 4 ;CHECK MASTER RESET LOGIC
;OR SHORT ON THIS BIT

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



```
2143      ;;THIS TEST VERIFYS THAT THE DEVICE REGISTER TXCSR
2144      ;;IS CLEARED BY MASTER RESET
2145      ;;
2146      ;*****
2147      TST47: SCOPE
2148      MOV     #177777,@TXCSR ;SET ALL POSSIBLE BITS
2149      BIS     #MRESET,@TXCSR ;MASTER RESET
2150      MOV     TXCSR,R3       ;FOR ERROR MESSAGE
2151      MOV     @TXCSR,R1      ;SAVE ACTUAL
2152      MOV     #200,R0        ;EXPECTED
2153      CMP     R0,R1          ;EXPECTED VS ACTUAL
2154      BEQ     .+4
2155      ERROR   1              ;ONLY TXDONE SHOULD BE SET
2156
2157      ;;THIS TEST VERIFYS THAT THE DEVICE REGISTER RXDBUF
2158      ;;IS CLEARED BY MASTER RESET
2159      ;;
2160      ;*****
2161      TST50: SCOPE
2162      BIS     #MRESET,@TXCSR ;MASTER RESET
2163      MOV     RXDBUF,R3       ;FOR ERROR MESSAGE
2164      MOV     @RXDBUF,R1      ;SAVE
2165      MOV     #377,R0        ;EXPECTED
2166      CMP     R0,R1          ;EXPECTED VS ACTUAL
2167      BEQ     .+4
2168      ERROR   2              ;ONLY REC DATA BITS SHOULD BE SET
2169      ;;THIS TEST VERIFYS BITS RING,CTS,CARDET,SRD,DSR
2170      ;;ALSO DSC IS GENERATED WHEN ANY OF THESE BITS ARE SET
2171      ;;OR CLEARED.....IT ALSO CHECKS THE MODEM BYPASS
2172      ;;JUMPER AND THAT THESE BITS CAN BE READ
2173      ;;NOTE: THE MODEM BYPASS JUMPER MUST BE ON (H315)
2174      ;;
2175      ;*****
2176      TST51: SCOPE
2177      CLR     @RXCSR ;TO GET RID OF STD ,RTS,DTR IF OPTCLR JUMPER #4 IS NOT ON
2178      BIS     #MRESET,@TXCSR ;MASTER RESET
2179      ;TEST THAT A "YES" ANSWER WAS GIVEN TO QUESTION IN
2180      ;THE MONITOR OR BY DEFAULT
2181      ;THIS TEST WILL BE BYPASSED IF THE EXTERNAL BYPASS
2182      ;JUMPER IS NOT INSTALLED
2183      TSTB    JMRBY
2184      BMI     .+6 ;THE ANSWER WAS YES.....
2185      ;PERFORM THIS TEST
2186      JMP     OUT1 ;JUMP AROUND THIS TEST IF THE ANSWER
2187      ;WAS NO
2188      MOV     RXCSR,R3       ;SET UP FOR ERROR MESSAGE
2189      MOV     @RXCSR,R1      ;ACTUAL
2190      CLR     R0             ;EXPECTED
2191      TST     R1             ;IS IT = 0 ?
2192      BEQ     .+4
2193      ERROR   1              ;RXCSR SHOULD BE CLR
2194      BIS     #DTR,@RXCSR    ;SET DTR
2195      ;WAIT FOR CABLE DELAYS
2196      ;*****
2197      ;MODIFY "HOLD:" ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2198      ;*****
```

2199	007150	016702	171744	MOV	HOLD,R2 ;SET DELAY TIME
2200	007154	005302		DEC	R2
2201	007156	001376		BNE	.-2 ;WAIT THIS TIME
2202					;OK NOW FALL THRU AND CONTINUE TESTING.....
2203					;EXIT STAGE LEFT....CHINNG!
2204	007160	017701	172526	MOV	@RXCSR,R1 ;ACTUAL
2205	007164	012700	130002	MOV	#130002,R0 ;DSC,CTS,CARDET,DTR
2206	007170	020001		CMP	R0,R1 ;EXPECTED VS ACTUAL
2207	007172	001401		BEQ	+.4
2208	007174	104001		ERROR	1 ;CHECK BYPASS CONNECTOR
2209	007176	017701	172510	MOV	@RXCSR,R1 ;ACTUAL
2210	007202	012700	030002	MOV	#30002,R0 ;CTS,CARDET,DTR
2211	007206	020001		CMP	R0,R1 ;EXPECTED VS ACTUAL
2212	007210	001401		BEQ	+.4
2213	007212	104001		ERROR	1 ;PREVIOUS READING OF RXCSR SHOULD
2214					;HAVE CLEARED DSC
2215	007214	052777	000004 172470	BIS	#RTS,@RXCSR
2216					;WAIT FOR CABLE DELAYS
2217					*****
2218					;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2219					*****
2220	007222	016702	171672	MOV	HOLD,R2 ;SET DELAY TIME
2221	007226	005302		DEC	R2
2222	007230	001376		BNE	.-2 ;WAIT THIS TIME
2223					;OK NOW FALL THRU AND CONTINUE TESTING.....
2224					;EXIT STAGE LEFT....CHINNG!
2225	007232	017701	172454	MOV	@RXCSR,R1
2226	007236	012700	170006	MOV	#170006,R0 ;DSC,RING,CTS,CARDET,RTS,DTR
2227	007242	020001		CMP	R0,R1 ;EXPECTED VS ACTUAL
2228	007244	001401		BEQ	+.4
2229	007246	104001		ERROR	1 ;CHECK BYPASS CONNECTOR
2230	007250	017701	172436	MOV	@RXCSR,R1
2231	007254	012700	070006	MOV	#70006,R0 ;RING,CTS,CARDET,RTS,DTR
2232	007260	020001		CMP	R0,R1 ;EXPECTED VS ACTUAL
2233	007262	001401		BEQ	+.4
2234	007264	104001		ERROR	1 ;PREVIOUS READING OF RXCSR SHOULD
2235					;HAVE CLEARED DSC
2236	007266	105767	171655	TSTB	SEXMIT ;IS SEC XMIT JUMPER IN ?
2237	007272	100112		BPL	OUT2 ;NO
2238	007274	105767	171650	TSTB	SEREC ;IS SEC REC JUMPER IN ?
2239	007300	100163		BPL	OUT3 ;NO
2240	007302	052777	000010 172402	BIS	#STD,@RXCSR
2241					;WAIT FOR CABLE DELAYS
2242					*****
2243					;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2244					*****
2245	007310	016702	171604	MOV	HOLD,R2 ;SET DELAY TIME
2246	007314	005302		DEC	R2
2247	007316	001376		BNE	.-2 ;WAIT THIS TIME
2248					;OK NOW FALL THRU AND CONTINUE TESTING.....
2249					;EXIT STAGE LEFT....CHINNG!
2250	007320	017701	172366	MOV	@RXCSR,R1
2251	007324	012700	173016	MOV	#173016,R0 ;DSC,RING,CTS,CARDET,SRD,DSR
2252					;STD,RTS,DTR
2253	007330	020001		CMP	R0,R1 ;EXPECTED VS ACTUAL
2254	007332	001401		BEQ	+.4

MASK1:

2255	007334	104001		ERROR	1	;CHECK BYPASS CONNECTOR
2256	007336	017701	172350	MOV	@RXCSR,R1	
2257	007342	012700	073016	MASK2: MOV	#73016,R0	;RING,CTS,CARDET,SRD,DSR,STD
2258						;RTS,DTR
2259	007346	020001		CMP	R0,R1	;EXPECTED VS ACTUAL
2260	007350	001401		BEQ	.+4	
2261	007352	104001		ERROR	1	;PREVIOUS READING OF RXCSR SHOULD
2262						;HAVE CLEARED DSC
2263	007354	042777	000002 172330	BIC	#DTR,@RXCSR	
2264						;WAIT FOR CABLE DELAYS
2265						*****
2266						;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2267						*****
2268	007362	016702	171532	MOV	HOLD,R2	;SET DELAY TIME
2269	007366	005302		DEC	R2	
2270	007370	001376		BNE	.-2	;WAIT THIS TIME
2271						;OK NOW FALL THRU AND CONTINUE TESTING.....
2272						;EXIT STAGE LEFT....CHINNG!
2273	007372	017701	172314	MOV	@RXCSR,R1	
2274	007376	012700	143014	MOV	#143014,R0	;DSC,RING,SRD,DSR,STD,RTS
2275	007402	020001		CMP	R0,R1	;EXPECTED VS ACTUAL
2276	007404	001401		BEQ	.+4	
2277	007406	104001		ERROR	1	;DSC SHOULD BE SET
2278	007410	042777	000004 172274	BIC	#RTS,@RXCSR	
2279						;WAIT FOR CABLE DELAYS
2280						*****
2281						;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2282						*****
2283	007416	016702	171476	MOV	HOLD,R2	;SET DELAY TIME
2284	007422	005302		DEC	R2	
2285	007424	001376		BNE	.-2	;WAIT THIS TIME
2286						;OK NOW FALL THRU AND CONTINUE TESTING.....
2287						;EXIT STAGE LEFT....CHINNG!
2288	007426	017701	172260	MOV	@RXCSR,R1	
2289	007432	012700	103010	MASK3: MOV	#103010,R0	;DSC,SRD,DSR,STD
2290	007436	020001		CMP	R0,R1	;EXPECTED VS ACTUAL
2291	007440	001401		BEQ	.+4	
2292	007442	104001		ERROR	1	;DSC SHOULD BE SET
2293	007444	042777	000010 172240	BIC	#STD,@RXCSR	
2294						;WAIT FOR CABLE DELAYS
2295						*****
2296						;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2297						*****
2298	007452	016702	171442	MOV	HOLD,R2	;SET DELAY TIME
2299	007456	005302		DEC	R2	
2300	007460	001376		BNE	.-2	;WAIT THIS TIME
2301						;OK NOW FALL THRU AND CONTINUE TESTING.....
2302						;EXIT STAGE LEFT....CHINNG!
2303	007462	017701	172224	MOV	@RXCSR,R1	
2304	007466	012700	100000	MOV	#100000,R0	;DSC
2305	007472	020001		CMP	R0,R1	;EXPECTED VS ACTUAL
2306	007474	001401		BEQ	.+4	
2307	007476	104001		ERROR	1	;DSC SHOULD BE SET
2308	007500	017701	172206	MOV	@RXCSR,R1	
2309	007504	005000		CLR	R0	;NONE
2310	007506	005701		TST	R1	

```
2311 007510 001401      BEQ      .+4
2312 007512 104001      ERROR    1      ;DSC SHOULD BE CLEARED FROM PREVIOUS
2313                      ;READING OF RXCSR
2314 007514 000167 000254      JMP      OUT1      ;JUMP AROUND
2315                      ;THE FOLLOWING ROUTINE HANDLES THE SITUATION WHERE SEC XMIT
2316                      ;AND SEC REC JUMPERS ARE NOT ON
2317 007520 052777 000010 172164 OUT2: BIS      #STD,@RXCSR
2318                      ;WAIT FOR CABLE DELAYS
2319                      ;*****
2320                      ;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2321                      ;*****
2322 007526 016702 171366      MOV      HOLD,R2 ;SET DELAY TIME
2323 007532 005302          DEC      R2
2324 007534 001376          BNE      .-2      ;WAIT THIS TIME
2325                      ;OK NOW FALL THRU AND CONTINUE TESTING.....
2326                      ;EXIT STAGE LEFT....CHINNG!
2327 007536 017701 172150      MOV      @RXCSR,R1      ;ACTUAL
2328 007542 012700 070016      MOV      #70016,R0      ;EXPECTED: RING ,CTS,CARDET,STD,RTS,DTR
2329 007546 020001          CMP      R0,R1      ;EXPECTED VS ACTUAL
2330 007550 001401          BEQ      .+4
2331 007552 104001          ERROR    1      ;CHECK SEC XMIT & SEC REC JUMPERS
2332 007554 042777 000004 172130 BIC      #RTS,@RXCSR
2333                      ;WAIT FOR CABLE DELAYS
2334                      ;*****
2335                      ;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2336                      ;*****
2337 007562 016702 171332      MOV      HOLD,R2 ;SET DELAY TIME
2338 007566 005302          DEC      R2
2339 007570 001376          BNE      .-2      ;WAIT THIS TIME
2340                      ;OK NOW FALL THRU AND CONTINUE TESTING.....
2341                      ;EXIT STAGE LEFT....CHINNG!
2342 007572 017701 172114      MOV      @RXCSR,R1      ;ACTUAL
2343 007576 012700 130012      MOV      #130012,R0      ;DSC,CTS,CARDET,DTR,STD
2344                      ;NOTE THAT DSC STILL ASSERTS EVEN THO THE SEC XMIT JUMPER # 6 IS NOT ON
2345 007602 020001          CMP      R0,R1      ;EXPECTED VS ACTUAL
2346 007604 001401          BEQ      .+4
2347 007606 104001          ERROR    1      ;CHECK BYPASS CONNECTOR
2348 007610 042777 000002 172074 BIC      #DTR,@RXCSR
2349                      ;WAIT FOR CABLE DELAYS
2350                      ;*****
2351                      ;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2352                      ;*****
2353 007616 016702 171276      MOV      HOLD,R2 ;SET DELAY TIME
2354 007622 005302          DEC      R2
2355 007624 001376          BNE      .-2      ;WAIT THIS TIME
2356                      ;OK NOW FALL THRU AND CONTINUE TESTING.....
2357                      ;EXIT STAGE LEFT....CHINNG!
2358 007626 017701 172060      MOV      @RXCSR,R1      ;ACTUAL
2359 007632 012700 100010      MOV      #100010,R0      ;DSC,STD
2360 007636 020001          CMP      R0,R1      ;EXPECTED VS ACTUAL
2361 007640 001401          BEQ      .+4
2362 007642 104001          ERROR    1      ;ONLY DSC & STD SHOULD BE SET
2363 007644 000167 000124      JMP      OUT1      ;JUMP AROUND
2364 007650 052777 000010 172034 OUT3: BIS      #STD,@RXCSR
2365                      ;WAIT FOR CABLE DELAYS
2366                      ;*****
```



```
2367 ;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2368 ;*****
2369 007656 016702 171236 MOV HOLD,R2 ;SET DELAY TIME
2370 007662 005302 DEC R2
2371 007664 001376 BNE -2 ;WAIT THIS TIME
2372 ;OK NOW FALL THRU AND CONTINUE TESTING.....
2373 ;EXIT STAGE LEFT....CHINNG!
2374 007666 017701 172020 MOV @RXCSR,R1 ;ACTUAL
2375 007672 012700 171016 MOV #171016,R0 ;EXPECTED: DSC,RING,CTS,CARDET,DSR,STD,RTS,DTR
2376 007676 020001 CMP R0,R1 ;EXPECTED VS ACTUAL
2377 007700 001401 BEQ +4
2378 007702 104001 ERROR 1 ;CHECK SEC REC JUMPER
2379 007704 042777 000004 172000 BIC #RTS,@RXCSR
2380 ;WAIT FOR CABLE DELAYS
2381 ;*****
2382 ;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2383 ;*****
2384 007712 016702 171202 MOV HOLD,R2 ;SET DELAY TIME
2385 007716 005302 DEC R2
2386 007720 001376 BNE -2 ;WAIT THIS TIME
2387 ;OK NOW FALL THRU AND CONTINUE TESTING.....
2388 ;EXIT STAGE LEFT....CHINNG!
2389 007722 017701 171764 MOV @RXCSR,R1 ;ACTUAL
2390 007726 012700 131012 MOV #131012,R0 ;EXPECTED: DSC,CTS,CARDET,DSR,STD,DTR
2391 007732 020001 CMP R0,R1 ;EXPECTED VS ACTUAL
2392 007734 001401 BEQ +4
2393 007736 104001 ERROR 1 ;CHECK H315 CONNECTOR
2394 007740 042777 000002 171744 BIC #DTR,@RXCSR
2395 ;WAIT FOR CABLE DELAYS
2396 ;*****
2397 ;MODIFY 'HOLD:' ACCORDINGLY FOR FASTER OR SLOWER MACHINE
2398 ;*****
2399 007746 016702 171146 MOV HOLD,R2 ;SET DELAY TIME
2400 007752 005302 DEC R2
2401 007754 001376 BNE -2 ;WAIT THIS TIME
2402 ;OK NOW FALL THRU AND CONTINUE TESTING.....
2403 ;EXIT STAGE LEFT....CHINNG!
2404 007756 017701 171730 MOV @RXCSR,R1 ;ACTUAL
2405 007762 012700 101010 MOV #101010,R0 ;EXPECTED: DSC,DSR,STD
2406 007766 020001 CMP R0,R1 ;EXPECTED VS ACTUAL
2407 007770 001401 BEQ +4
2408 007772 104001 ERROR 1 ;CHECK H315 CONNECTOR
2409 007774 OUT1:
2410
2411 ;:THIS TEST VERIFYS THAT REACT (REC ACTIVE) ASSERTS
2412 ;:IMMED. WHEN SYNC EXTERNAL MODE IS SELECTED
2413 ;:AND SYNC SEARCH IS SET
2414 ;:
2415 ;:*****
2416 007774 000004 tst52: SCOPE
2417 007776 052777 000400 171722 BIS #MRESET,@TXCSR ;MASTER RESET
2418 010004 012777 020000 171710 MOV #SYNEXT,@PARCSR ;SET THE MODE
2419 010012 052777 000400 171706 BIS #MRESET,@TXCSR ;MASTER RESET
2420
2421 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2422 010020 012777 064001 171700 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
```

```
2423
2424
2425 010026 012777 026026 171666 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2426 010034 032777 004000 171650 MOV #SYNXT!EIGHT!NOPAR!26,@PARCSR
2427 010042 001401 BEQ .+4
2428 010044 104004 ERROR 4 ;RECACT SHOULD NOT BE SET
2429 010046 052777 000020 171636 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
2430 010054 032777 004000 171630 BIT #RECACT,@RXCSR
2431 010062 001001 BNE .+4
2432 010064 104004 ERROR 4 ;RECACT DID NOT ASSERT
2433 010066 042777 000020 171616 BIC #SYNSCH,@RXCSR ;DROP SEARCH SYNC
2434 010074 032777 004000 171610 BIT #RECACT,@RXCSR ;IS IT =0?
2435 010102 001401 BEQ .+4
2436 010104 104004 ERROR 4 ;RECACT SHOULD BE 0
2437
2438 ::THIS TEST VERIFYS THAT RECACT (REC ACTIVE) ASSERTS
2439 ::IMMED. WHEN ISOCRONOUS MODE IS SELECTED
2440 ::AND SYNC SEARCH IS SET
2441 ::
2442 ::*****
2443 010106 000004 ;TST53: SCOPE
2444 010110 052777 000400 171610 BIS #MRESET,@TXCSR ;MASTER RESET
2445 010116 012777 000000 171576 MOV #ISYMOD,@PARCSR ;SET THE MODE
2446 010124 052777 000400 171574 BIS #MRESET,@TXCSR ;MASTER RESET
2447
2448 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2449 010132 012777 064001 171566 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2450
2451 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2452 010140 012777 006026 171554 MOV #ISYMOD!EIGHT!NOPAR!26,@PARCSR
2453 010146 032777 004000 171536 BIT #RECACT,@RXCSR
2454 010154 001401 BEQ .+4
2455 010156 104004 ERROR 4 ;RECACT SHOULD NOT BE SET
2456 010160 052777 000020 171524 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
2457 010166 032777 004000 171516 BIT #RECACT,@RXCSR
2458 010174 001001 BNE .+4
2459 010176 104004 ERROR 4 ;RECACT DID NOT ASSERT
2460 010200 042777 000020 171504 BIC #SYNSCH,@RXCSR ;DROP SEARCH SYNC
2461 010206 032777 004000 171476 BIT #RECACT,@RXCSR ;IS IT =0?
2462 010214 001401 BEQ .+4
2463 010216 104004 ERROR 4 ;RECACT SHOULD BE 0
2464
2465 ::VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
2466 ::IN TWO * SYNC CHARS THRU MAINT DATA BIT
2467 ::WATCH THE RECACT BIT
2468 ::ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
2469 ::*: DEPENDENT ON MONITOR.....
2470 ::IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
2471 ::TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
2472 ::ON THE SECOND CHARACTER
2473 ::ALSO CHECK THIS CHARACTER IN RXDBUF
2474 ::AND CHECK OPERATION OF SYNSCH
2475 ::MODE: SYNC INTERNAL
2476 ::LENGTH:FIVE
2477 ::
2478 ::*****
```



```
2479 010220 000004 TST54: SCOPE
2480 010222 052777 000400 171476 BIS #MRESET,@TXCSR ;MASTER RESET
2481 010230 012777 030000 171464 MOV #SYNINT,@PARCSR ;SET THE MODE
2482 010236 052777 000400 171462 BIS #MRESET,@TXCSR ;MASTER RESET
2483
2484 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2485 010244 012777 064001 171454 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2486
2487 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2488 010252 012777 030026 171442 MOV #SYNINT!FIVE!NOPAR!26,@PARCSR
2489 010260 016703 171432 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2490 010264 052777 000020 171420 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
2491 ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
2492 010272 042777 020000 171426 BIC #CLK,@TXCSR ;POKE CLK DOWN
2493 010300 052777 020000 171420 BIS #CLK,@TXCSR ;POKE CLK UP
2494 ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2495 010306 042777 020000 171412 BIC #CLK,@TXCSR ;POKE CLK DOWN
2496 010314 052777 020000 171404 BIS #CLK,@TXCSR ;POKE CLK UP
2497 010322 012767 000002 170574 MOV #2,COUNT
2498 010330 012767 000005 170564 1$: MOV #5,SHIFT ;# OF SHIFTS
2499 010336 012767 000026 171134 MOV #26,$TMP1 ;SYNC CHARACTER
2500 010344 004767 006554 JSR PC,RPOKE
2501 010350 005367 170550 DEC COUNT
2502 010354 001403 BEQ 2$
2503 ;TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
2504 010356 105767 170564 TSTB SYNCNO
2505 010362 100762 BMI 1$ ;TWO SYNC CHARS
2506 010364 105777 171322 2$: TSTB @RXCSR ;CHECK REC DONE BIT
2507 010370 100001 BPL .+4
2508 010372 104004 ERROR 4 ;RXDONE SHOULD NOT BE ASSERTED
2509 010374 032777 004000 171310 BIT #REACT,@RXCSR
2510 010402 001001 BNE .+4
2511 010404 104004 ERROR 4 ;REACT SHOULD BE ASSERTED
2512 010406 012767 000005 170506 MOV #5,SHIFT
2513 010414 012767 000021 171056 MOV #21,$TMP1 ;ANY CHARACTER
2514 010422 004767 006476 JSR PC,RPOKE
2515 010426 105777 171260 TSTB @RXCSR ;CHECK RXDONE
2516 010432 100401 BMI .+4
2517 010434 104004 ERROR 4 ;RXDONE SHOULD BE ASSERTED
2518 010436 032777 004000 171246 BIT #REACT,@RXCSR
2519 010444 001001 BNE .+4
2520 010446 104004 ERROR 4 ;REACT SHOULD STILL BE ASSERTED
2521 010450 042777 000020 171234 BIC #SYNSCH,@RXCSR ;CLR SYNC SEARCH
2522 010456 032777 004000 171226 BIT #REACT,@RXCSR ;IT SHOULD DROP IMMEDIATELY
2523 010464 001401 BEQ .+4
2524 010466 104004 ERROR 4 ;REACT SHOULD BE CLR
2525 010470 105777 171216 TSTB @RXCSR ;RXDONE
2526 010474 100401 BMI .+4
2527 010476 104004 ERROR 4 ;RXDONE SHOULD STILL BE ASSERTED
2528 010500 012700 000021 MOV #21,R0 ;EXPECTED DATA
2529 010504 017701 171206 MOV @RXDBUF,R1 ;ACTUAL DATA
2530 010510 020001 CMP R0,R1 ;COMPARE EXP VS ACT
2531 010512 001401 BEQ .+4
2532 010514 104002 ERROR 2 ;DATA CHARS SHOULD COMPARE
2533 010516 105777 171170 TSTB @RXCSR ;CHECK RXDONE
2534 010522 100001 BPL .+4
```

```
2535 010524 104004          ERROR 4          :RXDONE SHOULD BE CLR FROM
2536                               :PREVIOUS READING OF RXDBUF
2537
2538                               ::VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
2539                               ::IN TWO * SYNC CHARS THRU MAINT DATA BIT
2540                               ::WATCH THE RECACT BIT
2541                               ::ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
2542                               ::*: DEPENDENT ON MONITOR.....
2543                               ::IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
2544                               ::TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
2545                               ::ON THE SECOND CHARACTER
2546                               ::ALSO CHECK THIS CHARACTER IN RXDBUF
2547                               ::AND CHECK OPERATION OF SYNSCH
2548                               ::MODE: SYNC INTERNAL
2549                               ::LENGTH:SIX
2550                               ::
2551                               ::*****
2552 010526 000004          TST55: SCOPE
2553 010530 052777 000400 171170      BIS      #MRESET,@TXCSR ;MASTER RESET
2554 010536 012777 030000 171156      MOV      #SYNINT,@PARCSR ;SET THE MODE
2555 010544 052777 000400 171154      BIS      #MRESET,@TXCSR ;MASTER RESET
2556
2557                               ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2558 010552 012777 064001 171146      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
2559
2560                               ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2561 010560 012777 032026 171134      MOV      #SYNINT!SIX!NOPAR!26,@PARCSR
2562 010566 016703 171124              MOV      RXDBUF,R3          ;SET UP FOR ERROR MESSAGE
2563 010572 052777 000020 171112      BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
2564                               ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
2565 010600 042777 020000 171120      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2566 010606 052777 020000 171112      BIS      #CLK,@TXCSR ;POKE CLK UP
2567                               ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2568 010614 042777 020000 171104      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2569 010622 052777 020000 171076      BIS      #CLK,@TXCSR ;POKE CLK UP
2570 010630 012767 000002 170266      MOV      #2,COUNT
2571 010636 012767 000006 170256      1$: MOV      #6,SHIFT          ;# OF SHIFTS
2572 010644 012767 000026 170626      MOV      #26,$TMP1        ;SYNC CHARACTER
2573 010652 004767 006246              JSR      PC,RPOKE
2574 010656 005367 170242              DEC      COUNT
2575 010662 001403              BEQ      2$
2576                               ;TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
2577 010664 105767 170256              TSTB     SYNCNO
2578 010670 100762              BMI      1$          ;TWO SYNC CHARS
2579 010672 105777 171014              2$: TSTB     @RXCSR ;CHECK REC DONE BIT
2580 010676 100001              BPL      .+4
2581 010700 104004          ERROR 4          :RXDONE SHOULD NOT BE ASSERTED
2582 010702 032777 004000 171002      BIT      #RECACT,@RXCSR
2583 010710 001001              BNE      .+4
2584 010712 104004          ERROR 4          :RECACT SHOULD BE ASSERTED
2585 010714 012767 000006 170200      MOV      #6,SHIFT
2586 010722 012767 000021 170550      MOV      #21,$TMP1        ;ANY CHARACTER
2587 010730 004767 006170              JSR      PC,RPOKE
2588 010734 105777 170752              TSTB     @RXCSR ;CHECK RXDONE
2589 010740 100401              BMI      .+4
2590 010742 104004          ERROR 4          :RXDONE SHOULD BE ASSERTED
```



```
2591 010744 032777 004000 170740 BIT #RECACT,@RXCSR
2592 010752 001001 BNE .+4
2593 010754 104004 ERROR 4 ;RECACT SHOULD STILL BE ASSERTED
2594 010756 042777 000020 170726 BIC #SYNSCH,@RXCSR ;CLR SYNC SEARCH
2595 010764 032777 004000 170720 BIT #RECACT,@RXCSR ;IT SHOULD DROP IMMEDIATELY
2596 010772 001401 BEQ .+4
2597 010774 104004 ERROR 4 ;RECACT SHOULD BE CLR
2598 010776 105777 170710 TSTB @RXCSR ;RXDONE
2599 011002 100401 BMI .+4
2600 011004 104004 ERROR 4 ;RXDONE SHOULD STILL BE ASSERTED
2601 011006 012700 000021 MOV #21,R0 ;EXPECTED DATA
2602 011012 017701 170700 MOV @RXDBUF,R1 ;ACTUAL DATA
2603 011016 020001 CMP R0,R1 ;COMPARE EXP VS ACT
2604 011020 001401 BEQ .+4
2605 011022 104002 ERROR 2 ;DATA CHARS SHOULD COMPARE
2606 011024 105777 170662 TSTB @RXCSR ;CHECK RXDONE
2607 011030 100001 BPL .+4
2608 011032 104004 ERROR 4 ;RXDONE SHOULD BE CLR FROM
2609 ;PREVIOUS READING OF RXDBUF
2610
2611 ::VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
2612 ::IN TWO * SYNC CHARS THRU MAINT DATA BIT
2613 ::WATCH THE RECACT BIT
2614 ::ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
2615 ::*: DEPENDENT ON MONITOR.....
2616 ::IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
2617 ::TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
2618 ::ON THE SECOND CHARACTER
2619 ::ALSO CHECK THIS CHARACTER IN RXDBUF
2620 ::AND CHECK OPERATION OF SYNSCH
2621 ::MODE: SYNC INTERNAL
2622 ::LENGTH:SEVEN
2623 ::
2624 ::*****
2625 011034 000004 TST56: SCOPE
2626 011036 052777 000400 170662 BIS #MRESET,@TXCSR ;MASTER RESET
2627 011044 012777 030000 170650 MOV #SYNINT,@PARCSR ;SET THE MODE
2628 011052 052777 000400 170646 BIS #MRESET,@TXCSR ;MASTER RESET
2629
2630 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2631 011060 012777 064001 170640 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2632
2633 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2634 011066 012777 034026 170626 MOV #SYNINT!SEVEN!NOPAR!26,@PARCSR
2635 011074 016703 170616 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2636 011100 052777 000020 170604 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
2637 ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
2638 011106 042777 020000 170612 BIC #CLK,@TXCSR ;POKE CLK DOWN
2639 011114 052777 020000 170604 BIS #CLK,@TXCSR ;POKE CLK UP
2640 ;POKE CLK TO GET LOGIC INTO SYNCHRONIZATION
2641 011122 042777 020000 170576 BIC #CLK,@TXCSR ;POKE CLK DOWN
2642 011130 052777 020000 170570 BIS #CLK,@TXCSR ;POKE CLK UP
2643 011136 012767 000002 167760 MOV #2,COUNT
2644 011144 012767 000007 167750 1$: MOV #7,SHIFT ;# OF SHIFTS
2645 011152 012767 000026 170320 MOV #26,$TMP1 ;SYNC CHARACTER
2646 011160 004767 005740 JSR PC,RPOKE
```

```
2647 011164 005367 167734      DEC      COUNT
2648 011170 001403              BEQ      2$
2649                                ;TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
2650 011172 105767 167750      TSTB     SYNCNO
2651 011176 100762              BMI      1$      ;TWO SYNC CHARS
2652 011200 105777 170506      2$:      TSTB     @RXCSR      ;CHECK REC DONE BIT
2653 011204 100001              BPL      .+4
2654 011206 104004              ERROR    4      ;RXDONE SHOULD NOT BE ASSERTED
2655 011210 032777 004000 170474  BIT      #RECACT,@RXCSR
2656 011216 001001              BNE      .+4
2657 011220 104004              ERROR    4      ;RECACT SHOULD BE ASSERTED
2658 011222 012767 000007 167672  MOV      #7,SHIFT
2659 011230 012767 000021 170242  MOV      #21,$TMP1      ;ANY CHARACTER
2660 011236 004767 005662      JSR      PC,RPOKE
2661 011242 105777 170444      TSTB     @RXCSR      ;CHECK RXDONE
2662 011246 100401              BMI      .+4
2663 011250 104004              ERROR    4      ;RXDONE SHOULD BE ASSERTED
2664 011252 032777 004000 170432  BIT      #RECACT,@RXCSR
2665 011260 001001              BNE      .+4
2666 011262 104004              ERROR    4      ;RECACT SHOULD STILL BE ASSERTED
2667 011264 042777 000020 170420  BIC      #SYNSCH,@RXCSR      ;CLR SYNC SEARCH
2668 011272 032777 004000 170412  BIT      #RECACT,@RXCSR      ;IT SHOULD DROP IMMEDIATELY
2669 011300 001401              BEQ      .+4
2670 011302 104004              ERROR    4      ;RECACT SHOULD BE CLR
2671 011304 105777 170402      TSTB     @RXCSR      ;RXDONE
2672 011310 100401              BMI      .+4
2673 011312 104004              ERROR    4      ;RXDONE SHOULD STILL BE ASSERTED
2674 011314 012700 000021      MOV      #21,R0      ;EXPECTED DATA
2675 011320 017701 170372      MOV      @RXDBUF,R1      ;ACTUAL DATA
2676 011324 020001              CMP      R0,R1      ;COMPARE EXP VS ACT
2677 011326 001401              BEQ      .+4
2678 011330 104002              ERROR    2      ;DATA CHARS SHOULD COMPARE
2679 011332 105777 170354      TSTB     @RXCSR      ;CHECK RXDONE
2680 011336 100001              BPL      .+4
2681 011340 104004              ERROR    4      ;RXDONE SHOULD BE CLR FROM
2682                                ;PREVIOUS READING OF RXDBUF
2683
2684                                ;;VERIFY THE MATCH DETECT & DATA RDY FLAGS BY PUMPING
2685                                ;;IN TWO * SYNC CHARS THRU MAINT DATA BIT
2686                                ;;WATCH THE RECACT BIT
2687                                ;;ON THE THIRD * CHARACTER IT SHOULD SET RXDONE
2688                                ;;*: DEPENDENT ON MONITOR.....
2689                                ;;IF ONE SYNC STRAP IS SELECTED THEN IT WILL ONLY
2690                                ;;TAKE ONE SYNC CHARACTER FOR RXDONE TO ASSERT
2691                                ;;ON THE SECOND CHARACTER
2692                                ;;ALSO CHECK THIS CHARACTER IN RXDBUF
2693                                ;;AND CHECK OPERATION OF SYNSCH
2694                                ;;MODE: SYNC INTERNAL
2695                                ;;LENGTH:EIGHT
2696                                ;;
2697                                ;*****
2698 011342 000004      TST57:  SCOPE
2699 011344 052777 000400 170354  BIS      #MRESET,@TXCSR      ;MASTER RESET
2700 011352 012777 030000 170342  MOV      #SYNINT,@PARCSR      ;SET THE MODE
2701 011360 052777 000400 170340  BIS      #MRESET,@TXCSR      ;MASTER RESET
2702
```



```
2703 ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2704 011366 012777 064001 170332 MOV #MTDATA!CLK!MINT!BREAK,@TXCSR
2705
2706 ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2707 011374 012777 036026 170320 MOV #SYNINT!EIGHT!NOPAR!26,@PARCSR
2708 011402 016703 170310 MOV RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2709 011406 052777 000020 170276 BIS #SYNSCH,@RXCSR ;SET SYNC SEARCH
2710 ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
2711 011414 042777 020000 170304 BIC #CLK,@TXCSR ;POKE CLK DOWN
2712 011422 052777 020000 170276 BIS #CLK,@TXCSR ;POKE CLK UP
2713 ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2714 011430 042777 020000 170270 BIC #CLK,@TXCSR ;POKE CLK DOWN
2715 011436 052777 020000 170262 BIS #CLK,@TXCSR ;POKE CLK UP
2716 011444 012767 000002 167452 MOV #2,COUNT
2717 011452 012767 000010 167442 1$: MOV #8,,SHIFT ;# OF SHIFTS
2718 011460 012767 000026 170012 MOV #26,$TMP1 ;SYNC CHARACTER
2719 011466 004767 005432 JSR PC,RPOKE
2720 011472 005367 167426 DEC COUNT
2721 011476 001403 BEQ 2$
2722 ;TEST SYNCNO TO SEE HOW MANY SYNC CHARS WERE SELECTED
2723 011500 105767 167442 TSTB SYNCNO
2724 011504 100762 BMI 1$ ;TWO SYNC CHARS
2725 011506 105777 170200 2$: TSTB @RXCSR ;CHECK REC DONE BIT
2726 011512 100001 BPL .+4
2727 011514 104004 ERROR 4 ;RXDONE SHOULD NOT BE ASSERTED
2728 011516 032777 004000 170166 BIT #REACT,@RXCSR
2729 011524 001001 BNE .+4
2730 011526 104004 ERROR 4 ;REACT SHOULD BE ASSERTED
2731 011530 012767 000010 167364 MOV #8,,SHIFT
2732 011536 012767 000021 167734 MOV #21,$TMP1 ;ANY CHARACTER
2733 011544 004767 005354 JSR PC,RPOKE
2734 011550 105777 170136 TSTB @RXCSR ;CHECK RXDONE
2735 011554 100401 BMI .+4
2736 011556 104004 ERROR 4 ;RXDONE SHOULD BE ASSERTED
2737 011560 032777 004000 170124 BIT #REACT,@RXCSR
2738 011566 001001 BNE .+4
2739 011570 104004 ERROR 4 ;REACT SHOULD STILL BE ASSERTED
2740 011572 042777 000020 170112 BIC #SYNSCH,@RXCSR ;CLR SYNC SEARCH
2741 011600 032777 004000 170104 BIT #REACT,@RXCSR ;IT SHOULD DROP IMMEDIATELY
2742 011606 001401 BEQ .+4
2743 011610 104004 ERROR 4 ;REACT SHOULD BE CLR
2744 011612 105777 170074 TSTB @RXCSR ;RXDONE
2745 011616 100401 BMI .+4
2746 011620 104004 ERROR 4 ;RXDONE SHOULD STILL BE ASSERTED
2747 011622 012700 000021 MOV #21,R0 ;EXPECTED DATA
2748 011626 017701 170064 MOV @RXDBUF,R1 ;ACTUAL DATA
2749 011632 020001 CMP R0,R1 ;COMPARE EXP VS ACT
2750 011634 001401 BEQ .+4
2751 011636 104002 ERROR 2 ;DATA CHARS SHOULD COMPARE
2752 011640 105777 170046 TSTB @RXCSR ;CHECK RXDONE
2753 011644 100001 BPL .+4
2754 011646 104004 ERROR 4 ;RXDONE SHOULD BE CLR FROM
2755 ;PREVIOUS READING OF RXDBUF
2756
2757
2758 ;:THIS TEST VERIFYS WORD LENGTH SELECT OF THE
;:RECEIVER SECTION,IT USES THE ERROR FLAGS
```

```
2759      ;;TO DETERMINE THAT IT WAS SELECTED CORRECTLY
2760      ;;(OVRUN,RXERR)
2761      ;;MODE:ISYMOD
2762      ;;LENGTH:FIVE
2763      ;;CHAR:25
2764      ;;
2765      ;*****
2766 011650 000004      TST60: SCOPE
2767 011652 052777 000400 170046      BIS      #MRESET,@TXCSR ;MASTER RESET
2768 011660 012777 000000 170034      MOV      #ISYMOD,@PARCSR ;SET THE MODE
2769 011666 052777 000400 170032      BIS      #MRESET,@TXCSR ;MASTER RESET
2770
2771      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2772 011674 012777 064001 170024      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
2773
2774      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2775 011702 012777 000000 170012      MOV      #ISYMOD!FIVE!NOPAR!0,@PARCSR
2776 011710 052777 000020 167774      BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
2777      ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
2778 011716 042777 020000 170002      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2779 011724 052777 020000 167774      BIS      #CLK,@TXCSR ;POKE CLK UP
2780      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2781 011732 042777 020000 167766      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2782 011740 052777 020000 167760      BIS      #CLK,@TXCSR ;POKE CLK UP
2783 011746 016703 167744      MOV      RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2784 011752 012700 000025      MOV      #25,R0 ;EXPECTED
2785 011756 012767 000007 167136      MOV      #7,SHIFT ;# OF SHIFTS
2786 011764 012767 000152 167506      MOV      #152,$TMP1 ;DATA CHAR
2787 011772 004767 005126      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
2788 011776 105777 167710      TSTB     @RXCSR ;RXDONE ?
2789 012002 100401      BMI      +4
2790 012004 104004      ERROR     4 ;RXDONE SHOULD BE SET
2791 012006 017701 167704      MOV      @RXDBUF,R1 ;ACTUAL
2792 012012 020001      CMP      R0,R1 ;COMPARE EXPECTED VS. ACTUAL
2793 012014 001401      BEQ      +4
2794 012016 104002      ERROR     2 ;RECEIVED DATA DID NOT MATCH
2795      ;EXPECTED DATA - CHECK MAINT DATA
2796      ;OR RECEIVER LOGIC
2797 012020 012767 000007 167074      MOV      #7,SHIFT ;# OF SHIFTS
2798 012026 012767 000152 167444      MOV      #152,$TMP1 ;DATA CHAR
2799 012034 004767 005064      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
2800      ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2801 012040 012767 000007 167054      MOV      #7,SHIFT ;# OF SHIFTS
2802 012046 012767 000152 167424      MOV      #152,$TMP1 ;DATA CHAR
2803 012054 004767 005044      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
2804 012060 012700 140025      MOV      #140000!25,R0 ;EXPECTED DATA PLUS
2805      ;RXERR & OVRUN
2806 012064 017701 167626      MOV      @RXDBUF,R1 ;ACTUAL
2807 012070 020001      CMP      R0,R1 ;COMPARE EXP VS. ACT
2808 012072 001401      BEQ      +4
2809 012074 104002      ERROR     2 ;SPECIFICALLY LOOK AT RXERR &
2810      ;OVRUN BITS...THEY BOTH SHOULD BE SET
2811
2812      ;;THIS TEST VERIFYS WORD LENGTH SELECT OF THE
2813      ;;RECEIVER SECTION,IT USES THE ERROR FLAGS
2814      ;;TO DETERMINE THAT IT WAS SELECTED CORRECTLY
```



```
2815      ;;(OVRUN,RXERR)
2816      ;;MODE:ISYMOD
2817      ;;LENGTH:FIVE
2818      ;;CHAR:12
2819      ;;
2820      ;*****
2821      TST61: SCOPE
2822      BIS      #MRESET,@TXCSR ;MASTER RESET
2823      MOV      #ISYMOD,@PARCSR ;SET THE MODE
2824      BIS      #MRESET,@TXCSR ;MASTER RESET
2825
2826      ;SET MAINT DATA,CLK,BREAK,&MAINTENANCE MODE
2827      MOV      #MTDATA!CLK!MINT!BREAK,@TXCSR
2828
2829      ;SET MODE ,# OF BITS,PARITY SENSE,&LOAD SYNC REG
2830      MOV      #ISYMOD!FIVE!NOPAR!0,@PARCSR
2831      BIS      #SYNSCH,@RXCSR ;SET SYNC SEARCH
2832      ;POKE CLK TO GET RECEIVER INTO SYNCROIZATION....
2833      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2834      BIS      #CLK,@TXCSR ;POKE CLK UP
2835      ;POKE CLK TO GET LOGIC INTO SYNCRONIZATION
2836      BIC      #CLK,@TXCSR ;POKE CLK DOWN
2837      BIS      #CLK,@TXCSR ;POKE CLK UP
2838      MOV      RXDBUF,R3 ;SET UP FOR ERROR MESSAGE
2839      MOV      #12,R0 ;EXPECTED
2840      MOV      #7,SHIFT ;# OF SHIFTS
2841      MOV      #124,$TMP1 ;DATA CHAR
2842      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
2843      TSTB     @RXCSR ;RXDONE ?
2844      BMI      .+4
2845      ERROR    4 ;RXDONE SHOULD BE SET
2846      MOV      @RXDBUF,R1 ;ACTUAL
2847      CMP      R0,R1 ;COMPARE EXPECTED VS. ACTUAL
2848      BEQ      .+4
2849      ERROR    2 ;RECEIVED DATA DID NOT MATCH
2850      ;EXPECTED DATA - CHECK MAINT DATA
2851      ;OR RECEIVER LOGIC
2852      MOV      #7,SHIFT ;# OF SHIFTS
2853      MOV      #124,$TMP1 ;DATA CHAR
2854      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
2855      ;NOW SHIFT IN A SECOND CHARACTER WITHOUT READING RXDBUF
2856      MOV      #7,SHIFT ;# OF SHIFTS
2857      MOV      #124,$TMP1 ;DATA CHAR
2858      JSR      PC,RPOKE ;SHIFT IN THIS CHAR
2859      MOV      #140000!12,R0 ;EXPECTED DATA PLUS
2860      ;RXERR & OVRUN
2861      MOV      @RXDBUF,R1 ;ACTUAL
2862      CMP      R0,R1 ;COMPARE EXP VS. ACT
2863      BEQ      .+4
2864      ERROR    2 ;SPECIFICALLY LOOK AT RXERR &
2865      ;OVRUN BITS...THEY BOTH SHOULD BE SET
2866
2867
2868
2869      ;END OF PASS
2870      ;TYPE NAME OF TEST
```

```
2871 ;UPDATE PASS COUNT
2872 ;CHECK FOR EXIT TO ACT-11
2873 ;RESTART TEST
2874
2875 012324 000004 .EOP: SCOPE
2876 012326 004767 000340 JSR PC,CKSWR
2877 012332 104401 TYPE ;TYPE NAME OF TEST
2878 012334 015463 MEPASS
2879 012336 104413 012570 CONVRT ,OUTCRY
2880 012342 104401 015302 TYPE ,DEVICE
2881 012346 105767 166600 TSTB MULTD ;ARE YOU RUNNING MULTIPLE DEVICES ?
2882 012352 001511 BEQ CCC ;NO,JUMP AROUND
2883 012354 005767 166606 TST ACTREG ;ARE ANY DEVICES ACTIVE ?
2884 012360 001007 BNE RUNIT ;YES
2885 012362 104401 015314 TYPE ,MCOW ;NO
2886 012366 016700 166574 MOV ACTREG,R0 ;DISPLAY ACTREG
2887 012372 000000 HALT ;SELECT SOMETHING TO RUN @ ACTREG:
2888 ;SELECT SWITCHES & HIT CONTINUE (PUT SW00 =1)
2889 012374 000167 167552 JMP .START ;START OVER AGAIN.....YOU DESELECTED EVERYTHING
2890 012400 062767 000010 166546 RUNIT: ADD #10,BASEADD ;NEXT BLOCK (ADDRESSES)
2891 012406 062767 000010 166546 ZERO: ADD #10,BASEIV ;NEXT BLOCK (VECTORS)
2892 012414 000241 CLC
2893 012416 006167 166546 ROL ROTADD ;UP DATE ROTATING POINTER
2894 012422 103410 BCS 2$ ;IS IT THE LAST DEVICE
2895 ;TO BE TESTED IN THIS PASS ?
2896 012424 036767 166540 166534 BIT ROTADD,ACTREG ;TEST THIS DEVICE FOR ACTIVE STATUS
2897 012432 001762 BEQ RUNIT ;IF NOT ACTIVE, TRY NEXT ADDRESS
2898 012434 004767 000034 JSR PC,REPLAY ;CALCULATE NEW PARAMETERS
2899 012440 000167 000210 JMP RESTR ;YES IT WAS ACTIVE,TEST THIS DEVICE
2900 012444 012767 000001 166516 2$: MOV #1,ROTADD ;OK!,NOW SET UP ROTATING
2901 ;POINTER FOR NEXT MULTIPLE PASS
2902 012452 016767 166500 166474 MOV KEEPADD,BASEADD ;RESTORE BASE ADDRESS
2903 012460 016767 166500 166474 MOV KEEPIV,BASEIV ;RESTORE BASE INTERRUPT VECTORS
2904 012466 004767 000002 JSR PC,REPLAY ;CALC NEW PARAMETERS
2905 012472 000441 BR CCC ;JUMP AROUND REPLAY
2906 012474 016767 166454 004420 REPLAY: MOV BASEADD,DUBASE ;SET UP FOR NEW ADDRESSES
2907 012502 004767 004262 JSR PC,DUADDR ;CREATE NEW ADDRESSES
2908 012506 016767 166450 167222 MOV BASEIV,DURIV ;CREATE DURIV
2909 012514 062767 000002 166440 ADD #2,BASEIV
2910 012522 016767 166434 167210 MOV BASEIV,DURIS ;CREATE DURIS
2911 012530 062767 000002 166424 ADD #2,BASEIV
2912 012536 016767 166420 167176 MOV BASEIV,DUTIV ;CREATE DUTIV
2913 012544 062767 000002 166410 ADD #2,BASEIV
2914 012552 016767 166404 167164 MOV BASEIV,DUTIS ;CREATE DUTIS
2915 012560 016767 167152 166374 MOV DURIV,BASEIV ;RESTORE
2916 012566 000207 RTS PC
2917
2918 012570 000001 OUTCRY: 1
2919 012572 006 002 .BYTE 6,2
2920 012574 001712 RXCSR
2921
2922 012576 CCC:
2923 012576 005067 166600 CLR $TSTNM ;CLEAR TEST NUMBER
2924 012602 005067 166610 CLR $ERRPC ;CLEAR LAST ERROR PC
2925 012606 005067 166571 CLR $ERFLG ;CLEAR ERROR FLAG
2926 012612 005267 166274 INC PASCNT ;UPDATE PASS COUNT
```



```
2927 012616 016767 166270 166256      MOV      PASCNT,LIGHTS      ;DISPLAY PASS COUNT
2928 012624 016767 166262 166702      MOV      PASCNT,$PASS      ;PASS COUNT TO APT
2929 012632 013701 000042              MOV      @#42,R1          ;CHECK FOR ACT-11 OR DDP
2930 012636 001406              BEQ      RESTRT          ;IF NO CONTINUE TESTING
2931 012640 000005              RESET
2932 012642 000005              RESET
2933 012644 004711      SENDAD: JSR      PC,(R1)
2934 012646 000240              NOP
2935 012650 000240              NOP
2936 012652 000240              NOP
2937 012654              RESTRT:
2938 012654 012767 003376 166524      MOV      #TST1+2,$LPADR ;LOAD LAST ADDR
2939 012662 004767 000004              JSR      PC,CKSWR
2940 012666 000167 170416              JMP      .BEGIN
2941
2942              ;CHECK SWITCH REGISTER ROUTINE.
2943              ;CHECKS TO ALLOW FOR <^G> TO ALLOW
2944              ;THE CHANGING OF LOCATION 176
2945
2946 012672 005737 000042      CKSWR: TST      @#42
2947 012676 001040              BNE      OUT
2948 012700 022767 000176 166532      CMP      #SWREG,SWR      ;SOFTWARE SWR PRESENT?
2949 012706 001034              BNE      OUT          ;NO--LEAVE
2950 012710 105777 166530      TSTB     @TKS          ;CHECK TTY READY
2951 012714 100031              BPL      OUT          ;NO--LEAVE
2952 012716 017767 166524 000422      MOV      @TKB,.MSG      ;GET CHARACTER
2953 012724 042767 177600 000414      BIC      #177600,.MSG   ;STRIP JUNK
2954 012732 122767 000007 000406      CMPB     #7,.MSG        ;IS IT <^G> ?
2955 012740 001017              BNE      OUT          ;NO
2956 012742 104401 016070      CNTLU: TYPE     ,MCNTG
2957 012746 005137 013006      COM      @#RDSW
2958 012752 104401 016100      TYPE     ,MMSWR
2959 012756 104413      CONVRT
2960 012760 013010      SWREGL
2961 012762 104406 016111      INSTR,MMNEW
2962 012766 104410      PARAM
2963 012770 000000      0
2964 012772 177777      177777
2965 012774 000176      SWREG
2966 012776      000      001      .BYTE 0,1
2967 013000 005037 013006      OUT:   CLR      @#RDSW
2968 013004 000207              RTS      PC
2969 013006 000000      RDSW:   .WORD 0
2970 013010 000001      SWREGL: 1
2971 013012      006      002      .BYTE 6,2
2972 013014 000176      SWREG
2973
2974 013016 000005      5
2975
2976              ;CHECK FOR FREEZE ON CURRENT DATA
2977
2978 013020 004767 177646      .SCOP1: JSR      PC,CKSWR
2979 013024 032777 001000 166406      BIT      #SW09,@SWR
2980 013032 001402              BEQ      1$
2981 013034 016716 166050      MOV      LOCK,(SP)
2982 013040 000002      1$:   RTI
```

```
2983 .SBTTL TYPE ROUTINE
2984
2985 *****
2986 *ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2987 *THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2988 *NOTE1: $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2989 *NOTE2: $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2990 *NOTE3: $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2991 *
2992 *CALL:
2993 *1) USING A TRAP INSTRUCTION
2994 * TYPE ,MESADR ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2995 *OR
2996 * TYPE
2997 * MESADR
2998 *
2999
3000 013042 105767 166411 $TYPE: TSTB $IPFLG ;;IS THERE A TERMINAL?
3001 013046 100002 BPL 1$ ;;BR IF YES
3002 013050 000000 HALT ;;HALT HERE IF NO TERMINAL
3003 013052 000430 BR 3$ ;;LEAVE
3004 013054 010046 1$: MOV RO,-(SP) ;;SAVE RO
3005 013056 017600 000002 MOV @2(SP),RO ;;GET ADDRESS OF ASCIZ STRING
3006 013062 122767 000001 166456 CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
3007 013070 001011 BNE 62$ ;;NO,GO CHECK FOR APT CONSOLE
3008 013072 132767 000100 166447 BITB #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
3009 013100 001405 BEQ 62$ ;;NO,GO CHECK FOR CONSOLE
3010 013102 010067 000004 MOV RO,61$ ;;SETUP MESSAGE ADDRESS FOR APT
3011 013106 004767 164674 JSR PC,$ATY3 ;;SPOOL MESSAGE TO APT
3012 013112 000000 61$: .WORD 0 ;;MESSAGE ADDRESS
3013 013114 132767 000040 166425 62$: BITB #APTCSPUP,$ENVM ;;APT CONSOLE SUPPRESSED
3014 013122 001003 BNE 60$ ;;YES,SKIP TYPE OUT
3015 013124 112046 2$: MOVB (RO)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
3016 013126 001005 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
3017 013130 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
3018 013132 012600 60$: MOV (SP)+,RO ;;RESTORE RO
3019 013134 062716 000002 3$: ADD #2,(SP) ;;ADJUST RETURN PC
3020 013140 000002 RTI ;;RETURN
3021 013142 122716 000011 4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
3022 013146 001430 BEQ 8$
3023 013150 122716 000200 CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
3024 013154 001006 BNE 5$
3025 013156 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
3026 013160 104401 TYPE ;;TYPE A CR AND LF
3027 013162 001523 $CRLF
3028 013164 105067 000130 CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
3029 013170 000755 BR 2$ ;;GET NEXT CHARACTER
3030 013172 004767 000056 5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
3031 013176 126726 166254 6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
3032 013202 001350 BNE 2$ ;;IF NO GO GET NEXT CHAR.
3033 013204 016746 166244 MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
3034 013210 105366 000001 7$: DECB 1(SP) ;;AND THE NULL CHAR.
3035 013214 002770 6$: BLT 6$ ;;DOES A NULL NEED TO BE TYPED?
3036 013216 004767 000032 JSR PC,$TYPEC ;;BR IF NO--GO POP THE NULL OFF OF STACK
3037 013222 105367 000072 DECB $CHARCNT ;;GO TYPE A NULL
3038 ;;DO NOT COUNT AS A COUNT
```



```
3039 013226 000770          BR      7$          ;;LOOP
3040
3041          ;HORIZONTAL TAB PROCESSOR
3042
3043 013230 112716 000040      8$:   MOVB    #' (SP)          ;;REPLACE TAB WITH SPACE
3044 013234 004767 000014      9$:   JSR     PC,$TYPEC          ;;TYPE A SPACE
3045 013240 132767 000007 000052  BITB    #7,$CHARCNT          ;;BRANCH IF NOT AT
3046 013246 001372          BNE     9$          ;;TAB STOP
3047 013250 005726          TST     (SP)+          ;;POP SPACE OFF STACK
3048 013252 000724          BR      2$          ;;GET NEXT CHARACTER
3049 013254 105777 166170      $TYPEC: TSTB    @STPS          ;;WAIT UNTIL PRINTER IS READY
3050 013260 100375          BPL     $TYPEC
3051 013262 116677 000002 166162  MOVB    2(SP),@STPB          ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3052 013270 122766 000015 000002  CMPB    #CR,2(SP)          ;;IS CHARACTER A CARRIAGE RETURN?
3053 013276 001003          BNE     1$          ;;BRANCH IF NO
3054 013300 105067 000014          CLRB    $CHARCNT          ;;YES--CLEAR CHARACTER COUNT
3055 013304 000406          BR      $TYPEX          ;;EXIT
3056 013306 122766 000012 000002  1$:   CMPB    #LF,2(SP)          ;;IS CHARACTER A LINE FEED?
3057 013314 001402          BEQ     $TYPEX          ;;BRANCH IF YES
3058 013316 105227          INCB    (PC)+          ;;COUNT THE CHARACTER
3059 013320 000000          $CHARCNT: .WORD 0          ;;CHARACTER COUNT STORAGE
3060 013322 000207          $TYPEX: RTS      PC
3061
3062
3063          ;ASCII STRING INPUT ROUTINE
3064
3065 013324 017667 000000 000014  .INSTR: MOV    @ (SP),.MSG          ;PICK UP MESSAGE
3066 013332 062716 000002          ADD     #2,(SP)          ;JUMP AROUND MESSAGE FOR RTI
3067 013336 105767 166204          TSTB    $ENV          ;APT CONTROL
3068 013342 001036          BNE     INSTR2          ;YES NO TYPE
3069 013344 104401          .INST1: TYPE
3070 013346 000000          .MSG: 0
3071 013350 012704 016124          MOV     #INBUF,R4          ;GET STARTING LOC OF INBUF
3072 013354 012703 000007          MOV     #7,R3          ;MAX # OF CHARS
3073 013360 105777 166060      1$:   TSTB    @STKS          ;TTY FLAG
3074 013364 100375          BPL     1$
3075 013366 117714 166054          MOVB    @STKB,(R4)          ;TAKE CHAR
3076 013372 142714 000200          BICB    #200,(R4)          ;STRIP
3077 013376 121427 000025          CMPB    (R4),#25          ;IS IT <^G>
3078 013402 001760          BEQ     .INST1
3079 013404 122427 000015          CMPB    (R4)+,#15          ;CHECK FOR CR
3080 013410 001413          BEQ     INSTR2
3081 013412 105777 166032      2$:   TSTB    @STPS          ;TEST FLAG
3082 013416 100375          BPL     2$
3083 013420 117777 166022 166024  MOVB    @STKB,@STPB          ;ECHO CHARACTER
3084 013426 005303          DEC     R3          ;DID YOU TYPE TOO MANY CHARS ?
3085 013430 001353          BNE     1$
3086 013432 104401          .INSTE: TYPE
3087 013434 015410          MQM     ;?
3088 013436 000742          BR      .INST1          ;RETRY
3089 013440 000002          INSTR2: RTI
3090
3091          ;CONVERT ASCII STRING TO OCTAL
3092
3093 013442 011605          .PARAM: MOV    (SP),R5          ;PUT CONTENTS OF SP INTO R5
3094 013444 012567 000162          MOV    (R5)+,LOLIM          ;PUT LOW LIMIT INTO LOLIM
```

```
3095 013450 012567 000160      MOV      (R5)+,HILIM      ;PUT HIGH LIMIT INTO HILIM
3096 013454 012567 000156      MOV      (R5)+,DEVADR      ;PUT STORE LOC INTO DEVADR
3097 013460 112567 000154      MOV      (R5)+,LOBITS      ;PUT MASK INTO LOBITS
3098 013464 112567 000151      MOV      (R5)+,ADRCNT      ;PUT COUNT INTO ADRCNT
3099 013470 010516              MOV      R5,(SP) ;RESTORE RETURN ADDR ON STACK FOR RTI
3100 013472 005005              PARAM1: CLR      R5
3101 013474 012704 016124      MOV      #INBUF,R4
3102 013500 122714 000015      CMPB     #15,(R4)          ;CR ?
3103 013504 001420              BEQ      PARERR ;YOU TYPED CR TOO SOON !
3104 013506 121427 000060      1$:    CMPB     (R4),#60          ;LOW LIMIT ASCII 0
3105 013512 002415              BLT      PARERR
3106 013514 121427 000067      CMPB     (R4),#67          ;HIGH LIMIT ASCII 7
3107 013520 003012              BGT      PARERR
3108 013522 142714 000060      BICB     #60,(R4)          ;CONVERT TO OCTAL
3109 013526 152405              BISB     (R4)+,R5          ;STORE AWAY ITS AN OK CHAR
3110 013530 122714 000015      CMPB     #15,(R4)          ;CR ?
3111 013534 001414              BEQ      LIMITS ;NOW CHECK FOR HIGH & LOW LIMIT CONDS
3112 013536 006305              ASL      R5          ;ALLOCATE ROOM FOR NEXT CHAR
3113 013540 006305              ASL      R5
3114 013542 006305              ASL      R5
3115 013544 000760              BR       1$
3116 013546 122714 000015      PARERR: CMPB     #15,(R4)          ;CR?
3117 013552 001003              BNE      120$
3118 013554 005737 013006      TST      @#RDSW          ;CK SWR USED
3119 013560 001023              BNE      PARTI
3120 013562 104407              120$:  INSTER ;RETRY
3121 013564 000742              BR       PARAM1
3122
3123                          ;TEST TO SEE IF NUMBER IS WITHIN LIMITS
3124
3125 013566 020567 000042      LIMITS: CMP      R5,HILIM
3126 013572 101365              BHI      PARERR ;THE # IS TOO HIGH
3127 013574 020567 000032      CMP      R5,LOLIM
3128 013600 103762              BLO      PARERR ;THE # IS TOO LOW
3129 013602 136705 000032      BITB     LOBITS,R5          ;TEST BY MASKINGTHE #
3130 013606 001357              BNE      PARERR
3131
3132                          ;STORE NUMBER AT SPECIFIED ADDRESS
3133
3134 013610 016704 000022      1$:    MOV      DEVADR,R4          ;GET STARTING ADDR OF
3135 013614 010524              MOV      R5,(R4)+          ;STORE AT THIS ADDR
3136 013616 062705 000002      ADD      #2,R5
3137 013622 105367 000013      DECB     ADRCNT ;HOW MANY TIMES + 2 ?
3138 013626 001372              BNE      1$
3139 013630 000002              PARTI: RTI
3140 013632 000000              LOLIM: 0
3141 013634 000000              HILIM: 0
3142 013636 000000              DEVADR: 0
3143 013640 000000              LOBITS: 0
3144                          ADRCNT=LOBITS+1
3145
3146                          ;SAVE PC OF TEST THAT FAILED AND R0-R5
3147
3148 013642 016667 000004 165256 .SAV05: MOV      4(SP),SAVPC
3149
3150                          ;SAVE R0-R5
```



```
3151
3152 013650 010567 165620      SV05:  MOV    R5,$REG5
3153 013654 010467 165612      MOV    R4,$REG4
3154 013660 010367 165604      MOV    R3,$REG3
3155 013664 010267 165576      MOV    R2,$REG2
3156 013670 010167 165570      MOV    R1,$REG1
3157 013674 010067 165562      MOV    R0,$REG0
3158 013700 000002              RTI
3159
3160                          ;RESTORE R0-R5
3161
3162 013702 016700 165554      .RES05: MOV    $REG0,R0
3163 013706 016701 165552      MOV    $REG1,R1
3164 013712 016702 165550      MOV    $REG2,R2
3165 013716 016703 165546      MOV    $REG3,R3
3166 013722 016704 165544      MOV    $REG4,R4
3167 013726 016705 165542      MOV    $REG5,R5
3168 013732 000002              RTI
3169
3170                          ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
3171
3172 013734 104401      .CONVR: TYPE
3173 013736 015414      MCRLF      ;CR LF
3174 013740 017601 000000      MOV    @ (SP),R1      ;PICK UP DATA POINTER
3175 013744 062716 000002      ADD    #2,(SP) ;SET UP SP FOR RTI
3176 013750 012167 000130      MOV    (R1)+,WRDCNT      ;PICK UP # OF WORDS FROM TABLE
3177 013754 112167 000126      1$:  MOV    (R1)+,CHRCNT      ;PICK UP # OF CHARS FROM TABLE
3178 013760 112167 000123      MOV    (R1)+,SPACNT      ;PICK UP # OF SPACES FROM TABLE
3179 013764 013167 000120      MOV    @ (R1)+,BINWRD      ;PICK UP ADDRESS OF MSG
3180                                  ;FROM TABLE
3181 013770 016704 000114      2$:  MOV    BINWRD,R4      ;SAVE
3182 013774 116705 000106      MOV    CHRCNT,R5      ;SAVE
3183 014000 012700 016166      MOV    #TEMP,R0      ;STARTING ADDRESS OF TEMP BLOCK
3184 014004 010403 3$:  MOV    R4,R3      ;SAVE
3185 014006 042703 177770      BIC    #177770,R3      ;CLR OUT UPPER BITS .. SAVE CHAR
3186 014012 062703 000260      ADD    #260,R3 ;CONVERT TO ASCII
3187 014016 110320      MOV    R3,(R0)+      ;STORE AWAY
3188 014020 006204      ASR    R4      ;SHIFT FOR NEXT #
3189 014022 006204      ASR    R4      ;DITTO
3190 014024 006204      ASR    R4      ;DITTO
3191 014026 005305      DEC    R5      ;DEC CHAR COUNT
3192 014030 001365      BNE    3$      ;DO IT AGAIN ?
3193 014032 012703 016230      MOV    #MDATA,R3      ;STARTING ADDRESS OF MDATA BLOCK
3194 014036 114023 4$:  MOV    -(R0),(R3)+      ;REVERSE THE ORDER OF NUMBERS
3195 014040 105367 000042      DECB   CHRCNT      ;DEC CHAR COUNT
3196 014044 001374      BNE    4$      ;DO IT AGAIN ?
3197 014046 105767 000035      TSTB   SPACNT      ;HOW MANY SPACES ?
3198 014052 001405      BEQ    6$      ;TYPE # IF BR =0
3199 014054 112723 000240      5$:  MOV    #240,(R3)+      ;"SPACE" IN ASCII
3200 014060 105367 000023      DECB   SPACNT      ;DEC # OF SPACE COUNT
3201 014064 001373      BNE    5$      ;DO IT AGAIN ?
3202 014066 105013 6$:  CLRB    (R3)      ;INSERT '0' FOR TTY OUTPUT ROUTINE
3203 014070 104401      TYPE
3204 014072 016230      MDATA      ;THIS MESSAGE
3205 014074 005367 000004      DEC    WRDCNT      ;HOW MANY #'S ?
3206 014100 001325      BNE    1$      ;DO THIS ROUTINE AGAIN IF NOT EQUAL TO 0
```

```
3207 014102 000002          RTI      ;RETURN TO PROGRAM
3208 014104 000000          WRDCNT: 0
3209 014106 000000          CHRCNT: 0
3210      014107          SPACNT=CHRCNT+1
3211 014110 000000          BINWRD: 0
3212
3213      ;COMPARE THE FIRST CHARACTER IN THE TELETYPE INPUT
3214      ;BUFFER TO THE CHARACTERS 'N' AND 'Y'.
3215      ;IF THE CHARACTER IS 'N' CLEAR THE FLAG
3216      ;IF THE CHARACTER IS 'Y' SET THE FLAG
3217
3218 014112 017605 000000      .SETFLG:MOV    @ (SP),R5
3219 014116 122767 000116 002000      CMPB    #'N',INBUF      ;IS IT 'N' ?
3220 014124 001002          SNE      1$
3221 014126 105015          CLRB    (R5)      ;000
3222 014130 000406          BR      2$
3223 014132 122767 000131 001764 1$:    CMPB    #'Y',INBUF      ;IS IT 'Y' ?
3224 014140 001005          BNE      3$
3225 014142 112715 177777      MOV      #-1,(R5)      ;377
3226 014146 062716 000002      2$:    ADD      #2,(SP)
3227 014152 000002          RTI
3228 014154 104407          3$:    INSTER    ;RETRY
3229 014156 000755          BR      .SETFLG
3230      .SBTTL    ERROR HANDLER ROUTINE
3231
3232      ;*****
3233      ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3234      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3235      ;*AND GO TO SAVIT ON ERROR
3236      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3237      ;*SW15=1      HALT ON ERROR
3238      ;*SW13=1      INHIBIT ERROR TYPEOUTS
3239      ;*SW10=1      BELL ON ERROR
3240      ;*SW09=1      LOOP ON ERROR
3241      ;*CALL
3242      ;*      ERROR    N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3243
3244 014160          $ERROR:
3245 014160 105267 165217      7$:    INCB    $ERFLG      ;;SET THE ERROR FLAG
3246 014164 001775          BEQ      7$      ;;DON'T LET THE FLAG GO TO ZERO
3247 014166 016777 165210 165246      MOV      $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
3248 014174 032777 002000 165236      BIT      #BIT10,@SWR      ;;BELL ON ERROR?
3249 014202 001402          BEQ      1$      ;;NO - SKIP
3250 014204 104401 001516          TYPE    , $BELL      ;;RING BELL
3251 014210 005267 165176      1$:    INC      $ERTTL      ;;COUNT THE NUMBER OF ERRORS
3252 014214 011667 165176      MOV      (SP), $ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
3253 014220 162767 000002 165170      SUB      #2,$ERRPC
3254 014226 117767 165164 165160      MOV      @ $ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE
3255 014234 032777 020000 165176      BIT      #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
3256 014242 001004          BNE      20$      ;;SKIP TYPEOUTS
3257 014244 004767 000072      JSR      PC,SAVIT      ;;GO TO USER ERROR ROUTINE
3258 014250 104401 001523          TYPE    , $CRLF
3259 014254
3260 014254 122767 000001 165264      20$:    CMPB    #APTENV,$ENV      ;;RUNNING IN APT MODE
3261 014262 001007          BNE      2$      ;;NO, SKIP APT ERROR REPORT
3262 014264 116767 165124 000004      MOV      $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
```



```
3263 014272 004767 163520      JSR    PC,$ATY4      ;;REPORT FATAL ERROR TO APT
3264 014276      000      21$:  .BYTE    0
3265 014277      000      .BYTE    0
3266 014300 000777      22$:  BR      22$      ;;APT ERROR LOOP
3267 014302 005777 165132      2$:  TST    @SWR      ;;HALT ON ERROR
3268 014306 100001      BPL      3$      ;;SKIP IF CONTINUE
3269 014310 000000      HALT      ;;HALT ON ERROR!
3270 014312 032777 001000 165120 3$:  BIT    #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
3271 014320 001402      BEQ      4$      ;;BR IF NO
3272 014322 016716 165062      MOV    $LPERR,(SP)      ;;FUDGE RETURN FOR LOOPING
3273 014326 005767 165162      4$:  TST    $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
3274 014332 001402      BEQ      5$      ;;BR IF NONE
3275 014334 016716 165154      MOV    $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
3276 014340      5$:
3277 014340 000002      RTI      ;;RETURN
3278 014342 010067 164562      SAVIT: MOV    R0,HLD0
3279 014346 010167 164560      MOV    R1,HLD1
3280 014352 010267 164556      MOV    R2,HLD2
3281 014356 010367 164554      MOV    R3,HLD3
3282 014362 010467 164552      MOV    R4,HLD4
3283 014366 010567 164550      MOV    R5,HLD5
3284 014372 016767 165004 164544      MOV    $TSTNM,HLD6
```

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

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

```
3293 014400      $ERRTYP:
3294 014400 104401 001523      TYPE    , $CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
3295 014404 010046      MOV    R0,-(SP)      ;;SAVE R0
3296 014406 005000      CLR      R0      ;;PICKUP THE ITEM INDEX
3297 014410 153700 001414      BISB    @#$ITEMB,R0
3298 014414 001004      BNE      1$      ;;IF ITEM NUMBER IS ZERO, JUST
3299      MOV    $ERRPC,-(SP)      ;;TYPE THE PC OF THE ERROR
3300 014416 016746 164774      ;;SAVE $ERRPC FOR TYPEOUT
3301      ERROR ADDRESS
3302 014422 104402      TYP0C      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3303 014424 000426      BR      6$      ;;GET OUT
3304 014426 005300      1$:  DEC      R0      ;;ADJUST THE INDEX SO THAT IT WILL
3305 014430 006300      ASL      R0      ;;WORK FOR THE ERROR TABLE
3306 014432 006300      ASL      R0
3307 014434 006300      ASL      R0
3308 014436 062700 001652      ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
3309 014442 012067 000004      MOV     (R0)+,2$      ;;PICKUP "ERROR MESSAGE" POINTER
3310 014446 001404      BEQ      3$      ;;SKIP TYPEOUT IF NO POINTER
3311 014450 104401      TYPE      ;;TYPE THE "ERROR MESSAGE"
3312 014452 000000      2$:  .WORD    0      ;;"ERROR MESSAGE" POINTER GOES HERE
3313 014454 104401 001523      TYPE    , $CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
3314 014460 012067 000004      3$:  MOV     (R0)+,4$      ;;PICKUP "DATA HEADER" POINTER
3315 014464 001404      BEQ      5$      ;;SKIP TYPEOUT IF 0
3316 014466 104401      TYPE      ;;TYPE THE "DATA HEADER"
3317 014470 000000      4$:  .WORD    0      ;;"DATA HEADER" POINTER GOES HERE
3318 014472 104401 001523      TYPE    , $CRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
```

```
3319 014476 011000      5$:  MOV      (R0),R0      ;;PICKUP 'DATA TABLE' POINTER
3320 014500 001004      BNE      7$      ;;GO TYPE THE DATA
3321 014502 012600      6$:  MOV      (SP)+,R0      ;;RESTORE R0
3322 014504 104401 001523  TYPE      ,SCRLF      ;;'CARRIAGE RETURN' & 'LINE FEED'
3323 014510 000207      RTS      PC      ;;RETURN
3324 014512
3325 014512 013046      7$:  MOV      @ (R0)+,-(SP)      ;;SAVE @ (R0)+ FOR TYPEOUT
3326 014514 104402      TYPOC      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3327 014516 005710      TST      (R0)      ;;IS THERE ANOTHER NUMBER?
3328 014520 001770      BEQ      6$      ;;BR IF NO
3329 014522 104401 014530  TYPE      ,8$      ;;TYPE TWO(2) SPACES
3330 014526 000771      BR      7$      ;;LOOP
3331 014530 020040 000      8$:  .ASCIZ  / /      ;;TWO(2) SPACES
3332 014534      .EVEN
3333      .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3334
3335      ;*****
3336      ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3337      ;OCTAL (ASCII) NUMBER AND TYPE IT.
3338      ;$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3339      ;CALL:
3340      ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3341      ;      TYPOS      ;;CALL FOR TYPEOUT
3342      ;      .BYTE      N      ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
3343      ;      .BYTE      M      ;;M=1 OR 0
3344      ;      ;;1=TYPE LEADING ZEROS
3345      ;      ;;0=SUPPRESS LEADING ZEROS
3346      ;
3347      ;$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
3348      ;$TYPOS OR $TYPOC
3349      ;CALL:
3350      ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3351      ;      TYPON      ;;CALL FOR TYPEOUT
3352      ;
3353      ;$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
3354      ;CALL:
3355      ;      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
3356      ;      TYPOC      ;;CALL FOR TYPEOUT
3357      ;
3358 014534 017646 000000 000211 $TYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
3359 014540 116667 000001      MOV      1(SP),SOFILL      ;;LOAD ZERO FILL SWITCH
3360 014546 112667 000207      MOV      (SP)+,SOMODE+1      ;;NUMBER OF DIGITS TO TYPE
3361 014552 062716 000002      ADD      #2,(SP)      ;;ADJUST RETURN ADDRESS
3362 014556 000406      BR      $TYPON
3363 014560 112767 000001 000171 $TYPOC: MOV      #1,SOFILL      ;;SET THE ZERO FILL SWITCH
3364 014566 112767 000006 000165      MOV      #6,SOMODE+1      ;;SET FOR SIX(6) DIGITS
3365 014574 112767 000005 000154 $TYPON: MOV      #5,SOCNT      ;;SET THE ITERATION COUNT
3366 014602 010346      MOV      R3,-(SP)      ;;SAVE R3
3367 014604 010446      MOV      R4,-(SP)      ;;SAVE R4
3368 014606 010546      MOV      R5,-(SP)      ;;SAVE R5
3369 014610 116704 000145      MOV      SOMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
3370 014614 005404      NEG      R4
3371 014616 062704 000006      ADD      #6,R4      ;;SUBTRACT IT FOR MAX. ALLOWED
3372 014622 110467 000132      MOV      R4,SOMODE      ;;SAVE IT FOR USE
3373 014626 116704 000125      MOV      SOFILL,R4      ;;GET THE ZERO FILL SWITCH
3374 014632 016605 000012      MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
```



```
3375 014636 005003          CLR      R3          ;;CLEAR THE OUTPUT WORD
3376 014640 006105          1$:    ROL      R5          ;;ROTATE MSB INTO 'C'
3377 014642 000404          BR        3$          ;;GO DO MSB
3378 014644 006105          2$:    ROL      R5          ;;FORM THIS DIGIT
3379 014646 006105          ROL      R5
3380 014650 006105          ROL      R5
3381 014652 010503          MOV      R5,R3
3382 014654 006103          3$:    ROL      R3          ;;GET LSB OF THIS DIGIT
3383 014656 105367 000076    DECB     $OMODE        ;;TYPE THIS DIGIT?
3384 014662 100016          BPL      7$          ;;BR IF NO
3385 014664 042703 177770    BIC      #177770,R3      ;;GET RID OF JUNK
3386 014670 001002          BNE      4$          ;;TEST FOR 0
3387 014672 005704          TST      R4          ;;SUPPRESS THIS 0?
3388 014674 001403          BEQ      5$          ;;BR IF YES
3389 014676 005204          4$:    INC      R4          ;;DON'T SUPPRESS ANYMORE 0'S
3390 014700 052703 000060    BIS      #'0,R3        ;;MAKE THIS DIGIT ASCII
3391 014704 052703 000040    5$:    BIS      #'1,R3        ;;MAKE ASCII IF NOT ALREADY
3392 014710 110367 000040    MOV      R3,8$        ;;SAVE FOR TYPING
3393 014714 104401 014754    TYPE     ,8$          ;;GO TYPE THIS DIGIT
3394 014720 105367 000032    7$:    DECB     $OCNT        ;;COUNT BY 1
3395 014724 003347          BGT      2$          ;;BR IF MORE TO DO
3396 014726 002402          BLT      6$          ;;BR IF DONE
3397 014730 005204          INC      R4          ;;INSURE LAST DIGIT ISN'T A BLANK
3398 014732 000744          BR        2$          ;;GO DO THE LAST DIGIT
3399 014734 012605          6$:    MOV      (SP)+,R5      ;;RESTORE R5
3400 014736 012604          MOV      (SP)+,R4      ;;RESTORE R4
3401 014740 012603          MOV      (SP)+,R3      ;;RESTORE R3
3402 014742 016666 000002 000004  MOV     2(SP),4(SP)  ;;SET THE STACK FOR RETURNING
3403 014750 012616          MOV      (SP)+,(SP)
3404 014752 000002          RTI          ;;RETURN
3405 014754 000          8$:    .BYTE    0          ;;STORAGE FOR ASCII DIGIT
3406 014755 000          .BYTE    0          ;;TERMINATOR FOR TYPE ROUTINE
3407 014756 000          $OCNT:  .BYTE    0          ;;OCTAL DIGIT COUNTER
3408 014757 000          $OFILL: .BYTE    0          ;;ZERO FILL SWITCH
3409 014760 000000          $OMODE: .WORD    0          ;;NUMBER OF DIGITS TO TYPE
3410          .ENTER HERE ON POWER FAILURE
3411
3412
3413 014762          $PWRDN:
3414 014762 010046          .PFAIL: MOV     R0,-(SP)          ;SAVE R0-R5 ON PROCESSOR STACK
3415 014764 010146          MOV     R1,-(SP)
3416 014766 010246          MOV     R2,-(SP)
3417 014770 010346          MOV     R3,-(SP)
3418 014772 010446          MOV     R4,-(SP)
3419 014774 010546          MOV     R5,-(SP)
3420 014776 016746 163022    MOV     24,-(SP)
3421 015002 010667 164110    MOV     SP,SAVSP          ;SAVE STACK POINTER
3422 015006 012767 015020 163010  MOV     #RESTART,24      ;SET UP FOR POWER UP TRAP
3423 015014 000000          HALT          ;HALT ON POWER DOWN NORMAL
3424 015016 000777          BR        .
3425
3426          ;PROCESSOR WILL TRAP HERE WHEN POWER IS RESTORED
3427
3428 015020 016706 164072    RESTAR: MOV     SAVSP,SP          ;RESTORE STACK POINTER
3429 015024 012605          MOV     (SP)+,R5          ;RESTORE R0-R5
3430 015026 012604          MOV     (SP)+,R4
```

3431	015030	012603			MOV	(SP)+,R3	
3432	015032	012602			MOV	(SP)+,R2	
3433	015034	012601			MOV	(SP)+,R1	
3434	015036	012600			MOV	(SP)+,R0	
3435	015040	012767	014762	162756	MOV	#.PFAIL,24	;SET UP FOR POWER FAILURE
3436	015046	106427	000340		MTPS	#340	
3437	015052	012706	001100		MOV	#STACK,SP	
3438	015056	005067	001104		CLR	TEMP	
3439	015062	005267	001100		INC	TEMP	
3440	015066	001375			BNE	.-4	
3441	015070	104413			CONVRT		
3442	015072	015114			PFTAB		
3443	015074	104401			TYPE		
3444	015076	015417			MPFAIL		
3445	015100	005067	164277		CLR	\$ERFLG	
3446	015104	005067	164306		CLR	\$ERRPC	
3447	015110	000177	163770		JMP	@RETIJRN	
3448	015114	000001			PFTAB:	1	
3449	015116	006	002		.BYTE	6,2	
3450	015120	000207			RETURN		
3451	015122	005015	041412	042132	MTITLE:	.ASCIIZ	<15><12><12>/CZDUQ-CO DUV11 TAPE A /<15><12>
3452	015130	050525	041455	020060			
3453	015136	052504	030526	020061			
3454	015144	040524	042520	040440			
3455	015152	006440	000012				
3456	015156	005015	042526	020103	MVECTO:	.ASCIIZ	<15><12>/VEC ADD-/
3457	015164	042101	026504	000			
3458	015171	015	030412	052123	MREGAD:	.ASCIIZ	<15><12>/1ST DEV: REC CSR ADD-/
3459	015176	042040	053105	020072			
3460	015204	042522	020103	051503			
3461	015212	020122	042101	026504			
3462	015220	000					
3463	015221	015	046412	046125	MMULT:	.ASCIIZ	<15><12>/MULT DEV ? (Y OR N)-/
3464	015226	020124	042504	020126			
3465	015234	020077	054450	047440			
3466	015242	020122	024516	000055			
3467	015250	005015	040514	052123	MLASTD:	.ASCIIZ	<15><12>/LAST DEV: REC CSR ADDR-/
3468	015256	042040	053105	020072			
3469	015264	042522	020103	051503			
3470	015272	020122	042101	051104			
3471	015300	000055					
3472	015302	042075	053105	041511	DEVICE:	.ASCIIZ	/=DEVICE /
3473	015310	020105	000040				
3474	015314	005015	042523	042514	MCOW:	.ASCIIZ	<15><12>/SELECT TO RUN @ACTREG/
3475	015322	052103	052040	020117			
3476	015330	052522	020116	040500			
3477	015336	052103	042522	000107			
3478	015344	005015	053117	046106	MRANGE:	.ASCIIZ	<15><12>/OVFLO:RETYPE LAST DEV RXCSR ADDS-/
3479	015352	035117	042522	054524			
3480	015360	042520	046040	051501			
3481	015366	020124	042504	020126			
3482	015374	054122	051503	020122			
3483	015402	042101	051504	000055			
3484	015410	020040	000077		MQM:	.ASCIIZ	/ ?/
3485	015414	005015	000		MCRLF:	.ASCIIZ	<15><12>
3486	015417	120	040506	046111	MPFAIL:	.ASCIIZ	/PFail, RESTART AT TEST IN PROGRESS/

CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 72
BINARY TO OCTAL (ASCII) AND TYPE

D 6

SEQ 0068

3487	015424	020054	051040	051505	
3488	015432	040524	052122	040440	
3489	015440	020124	042524	052123	
3490	015446	044440	020116	051120	
3491	015454	043517	042522	051523	
3492	015462	000			
3493	015463	015	042412	042116	MEPASS: .ASCIIZ <15><12>/END OF PASS TAPE A/
3494	015470	047440	020106	040520	
3495	015476	051523	052040	050101	
3496	015504	020105	000101		
3497	015510	005015	000122		MR: .ASCIIZ <15><12>/R/
3498	015514	005015	042524	052123	MTSTPC: .ASCIIZ <15><12>/TEST PC-/
3499	015522	050040	026503	000	
3500	015527	015	046012	041517	MLOCK: .ASCIIZ <15><12>/LOCK ON TEST? (Y OR N)-/
3501	015534	020113	047117	020040	
3502	015542	042524	052123	020077	
3503	015550	054450	047440	020122	
3504	015556	024516	000055		
3505	015562	005015	020043	043117	MSYNC: .ASCIIZ <15><12>/# OF SYNC CHARS SELECTED (1 OR 2)-/
3506	015570	051440	047131	020103	
3507	015576	044103	051101	020123	
3508	015604	042523	042514	052103	
3509	015612	042105	024040	030440	
3510	015620	047440	020122	024462	
3511	015626	000055			
3512	015630	005015	051511	051440	MWIRE6: .ASCIIZ <15><12>/IS SEC XMIT SWITCH E55-2 IN? (Y OR N)-/
3513	015636	041505	054040	044515	
3514	015644	020124	053523	052111	
3515	015652	044103	042440	032465	
3516	015660	031055	044440	037516	
3517	015666	024040	020131	051117	
3518	015674	047040	026451	000	
3519	015701	015	044412	020123	MWIRE5: .ASCIIZ <15><12>/IS SEC REC SWITCH E55-3 IN? (Y OR N)-/
3520	015706	042523	020103	042522	
3521	015714	020103	053523	052111	
3522	015722	044103	042440	032465	
3523	015730	031455	044440	037516	
3524	015736	024040	020131	051117	
3525	015744	047040	026451	000	
3526	015751	015	044412	020123	MWIRE4: .ASCIIZ <15><12>/IS OPT CLR ENABLE SWITCH E55-1 IN? (Y OR N)-/
3527	015756	050117	020124	046103	
3528	015764	020122	047105	041101	
3529	015772	042514	051440	044527	
3530	016000	041524	020110	032505	
3531	016006	026465	020061	047111	
3532	016014	020077	054450	047440	
3533	016022	020122	024516	000055	
3534	016030	005015	044012	030463	MEXTJ: .ASCIIZ <15><12><12>/H315 CONNECTOR ON ?(Y OR N)-/
3535	016036	020065	047503	047116	
3536	016044	041505	047524	020122	
3537	016052	047117	037440	054450	
3538	016060	047440	020122	024516	
3539	016066	000055			
3540	016070	005015	057040	020107	MCNTG: .ASCIIZ <15><12>/ ^G /
3541	016076	000040			
3542	016100	051440	051127	020075	MMSWR: .ASCIIZ / SWR= /

```
3543 016106 020040 000
3544 016111 040 020040 042516 MMNEW: .ASCIZ / NEW= /
3545 016116 036527 020040 000
3546 016124 .EVEN
3547
3548 ;BUFFERS FOR INPUT-OUTPUT
3549
3550 016124 000000 INBUF: 0
3551 016166 016166 .=.+40
3552 016166 000000 TEMP: 0
3553 016230 016230 .=.+40
3554 016230 000000 MDATA: 0
3555 016272 016272 .=.+40
3556 .SBTTL SCOPE HANDLER ROUTINE
3557
3558 ;*****
3559 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
3560 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISP.LAY<7:0>)
3561 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
3562 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3563 ;*SW14=1 LOOP ON TEST
3564 ;*SW11=1 INHIBIT ITERATIONS
3565 ;*SW09=1 LOOP ON ERROR
3566 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
3567 ;*CALL
3568 ;* SCOPE ;:SCOPE=IOT
3569
3570 016272 $SCOPE:
3571
3572 ;SCOPE LOOP AND INTERATION HANDLER
3573
3574 016272 .SCOPE:
3575 016272 004767 174374 JSR PC,CKSWR
3576 016276 005067 163114 CLR $ERRPC ;CLEAR LAST ERROR PC
3577 016302 022716 003376 CMP #TST1+2,(SP) ;IS SCOPE AT BEGINING OF TEST 1?
3578 016306 001422 BEQ $XTSTR ;YES NO LOOP.
3579
3580 016310 032777 040000 163122 TTST: BIT #BIT14,$SWR ;THIS CODE IS FOR TESTING FOR BIT 14
3581 016316 001412 BEQ 1$ ;ON LSI WHICH SYSMAC CANNOT HANDLE
3582 016320 016767 163056 163060 MOV $TSTNM,$LPADR
3583 016326 000406 BR 1$
3584 016330 105777 163110 TSTB @TKS ;KEYBOARD DONE?
3585 016334 100123 BPL $OVER ;BR IF NO
3586 016336 017766 163104 177776 MOV @TKB,-2(SP) ;CLEAR DONE BIT
3587 016344 032777 040000 163066 1$: BIT #BIT14,$SWR ;:LOOP ON PRESENT TEST?
3588 016352 001114 BNE $OVER ;:YES IF SW14=1
3589 ;*****START OF CODE FOR THE XOR TESTER*****
3590 016354 000416 $XTSTR: BR 6$ ;:IF RUNNING ON THE "XOR" TESTER CHANGE
3591 ;:THIS INSTRUCTION TO A "NOP" (NOP=240)
3592 016356 013746 000004 MOV @ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
3593 016362 012737 016402 000004 MOV #5,$@ERRVEC ;:SET FOR TIMEOUT
3594 016370 005737 177060 TST @177060 ;:TIME OUT ON XOR?
3595 016374 012637 000004 MOV (SP)+,$@ERRVEC ;:RESTORE THE ERROR VECTOR
3596 016400 000463 BR $SVLAD ;:GO TO THE NEXT TEST
3597 016402 022626 5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
3598 016404 012637 000004 MOV (SP)+,$@ERRVEC ;:RESTORE THE ERROR VECTOR
```



```
3599 016410 000423          BR      7$          ;;LOOP ON THE PRESENT TEST
3600 016412          6$::#####END OF CODE FOR THE XOR TESTER#####
3601 016412 032777 000400 163020 BIT      #BIT08,@SWR      ;;LOOP ON SPEC. TEST?
3602 016420 001404          BEQ      2$          ;;BR IF NO
3603 016422 127767 163012 162752 CMPB    @SWR,$STNM      ;;ON THE RIGHT TEST? SWR<7:0>
3604 016430 001465          BEQ      $OVER      ;;BR IF YES
3605 016432 105767 162745 2$:      TSTB    $ERFLG      ;;HAS AN ERROR OCCURRED?
3606 016436 001421          BEQ      3$          ;;BR IF NO
3607 016440 126767 162751 162735 CMPB    $ERMAX,$ERFLG  ;;MAX. ERRORS FOR THIS TEST OCCURRED?
3608 016446 101015          BHI      3$          ;;BR IF NO
3609 016450 032777 001000 162762 BIT      #BIT09,@SWR      ;;LOOP ON ERROR?
3610 016456 001404          BEQ      4$          ;;BR IF NO
3611 016460 016767 162724 162720 7$:   MOV    $LPERR,$LPADR  ;;SET LOOP ADDRESS TO LAST SCOPE
3612 016466 000446          BR      $OVER
3613 016470 105067 162707 4$:      CLRB    $ERFLG      ;;ZERO THE ERROR FLAG
3614 016474 005067 163012          CLR     $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
3615 016500 000415          BR      1$          ;;ESCAPE TO THE NEXT TEST
3616 016502 032777 004000 162730 3$:   BIT    #BIT11,@SWR      ;;INHIBIT ITERATIONS?
3617 016510 001011          BNE      1$          ;;BR IF YES
3618 016512 005767 163016          TST     $PASS      ;;IF FIRST PASS OF PROGRAM
3619 016516 001406          BEQ      1$          ;;INHIBIT ITERATIONS
3620 016520 005267 162660          INC     $ICNT      ;;INCREMENT ITERATION COUNT
3621 016524 026767 162762 162652 CMP     $TIMES,$ICNT  ;;CHECK THE NUMBER OF ITERATIONS MADE
3622 016532 002024          BGE      $OVER      ;;BR IF MORE ITERATION REQUIRED
3623 016534 012767 000001 162642 1$:   MOV    #1,$ICNT      ;;REINITIALIZE THE ITERATION COUNTER
3624 016542 016767 000056 162742          MOV    $MXCNT,$TIMES  ;;SET NUMBER OF ITERATIONS TO DO
3625 016550 105267 162626          $SVLAD: INCB    $STNM      ;;COUNT TEST NUMBERS
3626 016554 116767 162622 162750          MOVB   $STNM,$TESTN  ;;SET TEST NUMBER IN APT MAILBOX
3627 016562 011667 162620          MOV    (SP),$LPADR      ;;SAVE SCOPE LOOP ADDRESS
3628 016566 011667 162616          MOV    (SP),$LPERR      ;;SAVE ERROR LOOP ADDRESS
3629 016572 005067 162716          CLR     $ESCAPE      ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
3630 016576 112767 000001 162611          MOVB   #1,$ERMAX      ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3631 016604 016777 162572 162630 $OVER: MOV    $STNM,@DISPLAY  ;;DISPLAY TEST NUMBER
3632 016612 016716 162570          MOV    $LPADR,(SP)      ;;FUDGE RETURN ADDRESS
3633 016616 000002          4$:      RTI
3634 016620 001407          BRW:      1407
3635 016622 000432          BRX:      432
3636 016624 000005          $MXCNT: 5          ;;MAX. NUMBER OF ITERATIONS
3637          .SBTTL TRAP DECODER
3638
3639          ;*****
3640          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
3641          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
3642          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
3643          ;*GO TO THAT ROUTINE.
3644
3645 016626 010046          $TRAP: MOV    R0,-(SP)      ;;SAVE R0
3646 016630 016600 000002          MOV    2(SP),R0      ;;GET TRAP ADDRESS
3647 016634 005740          TST     -(R0)      ;;BACKUP BY 2
3648 016636 111000          MOVB   (R0),R0      ;;GET RIGHT BYTE OF TRAP
3649 016640 006300          ASL     R0          ;;POSITION FOR INDEXING
3650 016642 016000 016662          MOV    $TRPAD(R0),R0  ;;INDEX TO TABLE
3651 016646 000200          RTS     R0          ;;GO TO ROUTINE
3652
3653          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
3654
```

```
3655
3656 016650 011646
3657 016652 016666 000004 000002 $TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
3658 016660 000002 MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
3659 RTI ;;RESTORE THE PSW
3660
3661 .SBTTL TRAP TABLE
3662
3663 ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
3664 ;*BY THE "TRAP" INSTRUCTION.
3665
3666 ; ROUTINE
3667 ;-----
3667 $TRPAD: .WORD $TRAP2
3668 $TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
3669 $TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
3670 $TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
3671 $TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
3672
3673
3674 .SCOP1 ;;CALL=SCOP1 TRAP+5(104405)
3675 .INSTR ;;CALL=INSTR TRAP+6(104406)
3676 .INSTER ;;CALL=INSTER TRAP+7(104407)
3677 .PARAM ;;CALL=PARAM TRAP+10(104410)
3678 .SAV05 ;;CALL=SAV05 TRAP+11(104411)
3679 .RES05 ;;CALL=RES05 TRAP+12(104412)
3680 .CONVRT ;;CALL=CONVRT TRAP+13(104413)
3681 .SETFLG ;;CALL=SETFLG TRAP+14(104414)
3682
3683 ;*****
3684 ;UTILITIES
3685 ;*****
3686
3687 ;THIS UTILITY CALCULATES PRIORITY LEVEL
3687 DULEV: ASL DUPRT ;SHIFT LEFT
3688 ASL DUPRT ;
3689 ASL DUPRT ;
3690 ASL DUPRT ;
3691 ASL DUPRT ;
3692 MOV DUPRT,LESS1 ;MOVE THIS TO LESS1
3693 SUB #1,LESS1 ;CREATE LESS1
3694 BIC #37,LESS1 ;CLEAR TNZVC
3695 RTS PC
3696
3697 DUPRT: PR5
3697 LESS1: PR4 ;LEVEL TO ALLOW INTERRUPTS
3698
3699 ;NEW DU ADDRESSES
3700 DUADDR: MOV DUBASE,RXCSR ;XXX0
3701 INC DUBASE
3702 MOV DUBASE,HRXCSR ;XXX1
3703 INC DUBASE
3704 MOV DUBASE,RXDBUF ;XXX2
3705 MOV DUBASE,PARCSR ;XXX2
3706 INC DUBASE
3707 MOV DUBASE,HRXDBUF ;XXX3
3708 MOV DUBASE,HPARCSR ;XXX3
3709 INC DUBASE
3710 MOV DUBASE,TXCSR ;XXX4
```



```
3711 017062 005267 000034      INC      DUBASE
3712 017066 016767 000030 162634  MOV      DUBASE,HTXCSR      ;XXX5
3713 017074 005267 000022      INC      DUBASE
3714 017100 016767 000016 162624  MOV      DUBASE,TXDBUF      ;XXX6
3715 017106 005267 000010      INC      DUBASE
3716 017112 016767 000004 162614  MOV      DUBASE,HTXDBUF      ;XXX7
3717 017120 000207      RTS      PC
3718 017122 000000      DUBASE: 0
3719
3720      ;THIS UTILITY POKES THE MAINT DATA BASED UPON THE
3721      ;INFORMATION CONTAINED IN $TMP1 AND IT IS
3722      ;SHIFTED IN BY THE CONTENTS OF SHIFT
3723 017124 042777 040000 162574  RPOKE: BIC      #MTDATA,@TXCSR
3724 017132 005067 162344      CLR      $TMP2
3725 017136 006067 162336      ROR      $TMP1      ;FORCE CARRY
3726 017142 006067 162334      ROR      $TMP2      ;PICK UP CARRY IN BIT 15
3727 017146 006267 162330      ASR      $TMP2      ;SHIFT INTO BIT 14
3728 017152 042767 100000 162322  BIC      #BIT15,$TMP2      ;CLR BIT 15
3729 017160 056777 162316 162540  BIS      $TMP2,@TXCSR      ;POKE MAINT DATA
3730 017166 042777 020000 162532  BIC      #CLK,@TXCSR      ;POKE CLK
3731 017174 052777 020000 162524  BIS      #CLK,@TXCSR      ;
3732 017202 005367 161714      DEC      SHIFT
3733 017206 001346      BNE      RPOKE
3734 017210 000207      RTS      PC
3735
3736      ;THIS ROUTINE CALCULATES ODD PARITY FOR AN 8 BIT CHAR
3737 017212 016767 162262 162262  ODD8: MOV      $TMP1,$TMP2      ;SAVE TEMP1
3738 017220 005067 162260      CLR      $TMP3
3739 017224 012727 000010      MOV      #8.,(PC)+
3740 017230 000000      4$: 0
3741 017232 006067 162244      1$: ROR      $TMP2
3742 017236 005567 162242      ADC      $TMP3
3743 017242 005367 177762      DEC      4$
3744 017246 001371      BNE      1$
3745 017250 006067 162230      ROR      $TMP3
3746 017254 103404      BCS      2$
3747 017256 052767 000400 162214  BIS      #BIT8,$TMP1      ;SET ODD PARITY
3748 017264 000403      BR      3$
3749 017266 042767 000400 162204  2$: BIC      #BIT8,$TMP1      ;CLR EVEN PARITY
3750      ;$TMP1 NOW HAS ODD PARITY CHARACTER
3751 017274 000207      3$: RTS      PC
3752
3753      ;THIS ROUTINE CALCULATES EVEN PARITY FOR AN 8 BIT CHARACTER
3754 017276 016767 162176 162176  EVEN8: MOV      $TMP1,$TMP2      ;SAVE TEMP1
3755 017304 005067 162174      CLR      $TMP3
3756 017310 012727 000010      MOV      #8.,(PC)+
3757 017314 000000      4$: 0
3758 017316 006067 162160      1$: ROR      $TMP2
3759 017322 005567 162156      ADC      $TMP3
3760 017326 005367 177762      DEC      4$
3761 017332 001371      BNE      1$
3762 017334 006067 162144      ROR      $TMP3
3763 017340 103004      BCC      2$
3764 017342 052767 000400 162130  BIS      #BIT8,$TMP1      ;SET EVEN PARITY
3765 017350 000403      BR      3$
3766 017352 042767 000400 162120  2$: BIC      #BIT8,$TMP1      ;CLR ODD PARITY
```

CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 77
TRAP TABLE

I 6

SEQ 0073

```
3767                                     ;$TMP1 NOW HAS EVEN PARITY CHARACTER
3768 017360 000207                      3$: RTS      PC
3769 017362 062716 000002              TRPREG: ADD    #2,(SP) ;ALLOW IT TO 'CRUNCH' INTO ERROR BACK
3770                                     ;IN MAIN PART OF THE PROGRAM
3771 017366 000002                      RTI
3772 000001                          .END
```


CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 79

J 6

CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0074

AAA	003200	1364#							
ABASE =	000000	947	988						
ACDW1 =	000000	947	990						
ACDW2 =	000000	947	991						
ACPUOP=	000000	947	962						
ACTREG	001166	806#	1320*	1334*	1335*	1342*	2883	2886	2896
ADDW0 =	000000	947	992						
ADDW1 =	000000	947	993						
ADDW10=	000000	947	1002						
ADDW11=	000000	947	1003						
ADDW12=	000000	947	1004						
ADDW13=	000000	947	1005						
ADDW14=	000000	947	1006						
ADDW15=	000000	947	1007						
ADDW2 =	000000	947	994						
ADDW3 =	000000	947	995						
ADDW4 =	000000	947	996						
ADDW5 =	000000	947	997						
ADDW6 =	000000	947	998						
ADDW7 =	000000	947	999						
ADDW8 =	000000	947	1000						
ADDW9 =	000000	947	1001						
ADEVCT=	000000	947	953						
ADEVN =	000000	947	989						
ADRCNT=	013641	3098*	3137*	3144#					
AENV =	000000	947	958						
AENVN =	000000	947	959						
AFATAL=	000000	947	950						
AMADR1=	000000	947	975						
AMADR2=	000000	947	979						
AMADR3=	000000	947	982						
AMADR4=	000000	947	985						
AMAMS1=	000000	947	969						
AMAMS2=	000000	947	977						
AMAMS3=	000000	947	980						
AMAMS4=	000000	947	983						
AMSGAD=	000000	947	955						
AMSGLG=	000000	947	956						
AMSGTY=	000000	947	949						
AMTYP1=	000000	947	970						
AMTYP2=	000000	947	978						
AMTYP3=	000000	947	981						
AMTYP4=	000000	947	984						
APASS =	000000	947	952						
APRIOR=	000000	947							
APTCSU=	000040	606#	3013						
APTENV=	000001	562	604#	3006		3260			
APTSIZ=	000200	603#	1239						
APTSPO=	000100	564	605#	3008					
ASWREG=	000000	947	960						
ATESTN=	000000	947	951						
AUNIT =	000000	947	954						
AUSWR =	000000	947	961						
AVECT1=	000000	947	986						
AVECT2=	000000	947	987						
BASEAD	001154	801#	1302*	1339*	1340	1346*	1348*	2890*	2902* 2906

BASE IV	001162	804#	1312*	2891*	2903*	2908	2909*	2910	2911*	2912	2913*	2914	2915*
BBB	003026	1319	1323#										
BINWRD	014110	3179*	3181	3211#									
BITW =	002000	873#	1066#	1928	1935	1947	1954	1964	1979	1983	1990		
BIT0 =	000001	736#	849	880	1042	1073	1915						
BIT00 =	000001	726#	736										
BIT01 =	000002	725#	735										
BIT02 =	000004	724#	734										
BIT03 =	000010	723#	733										
BIT04 =	000020	722#	732										
BIT05 =	000040	721#	731										
BIT06 =	000100	720#	730										
BIT07 =	000200	719#	729										
BIT08 =	000400	718#	728	3601									
BIT09 =	001000	717#	727	3270	3609								
BIT1 =	000002	735#	848	1041									
BIT10 =	002000	716#	839	873	1032	1066	3248						
BIT11 =	004000	715#	838	1031	1782	1783	1786	1787	1791	1793	3616		
BIT12 =	010000	714#	837	854	1030	1047	1804	1805	1808	1809	1813	1815	
BIT13 =	020000	713#	836	853	872	1029	1046	1065	3255				
BIT14 =	040000	712#	835	852	871	1028	1045	1064	3580	3587			
BIT15 =	100000	711#	834	851	870	1027	1044	1063	3728				
BIT2 =	000004	734#	847	1040									
BIT3 =	000010	733#	846	879	1039	1072							
BIT4 =	000020	732#	845	878	1038	1071							
BIT5 =	000040	731#	844	877	1037	1070	1926						
BIT6 =	000100	730#	843	876	1036	1069							
BIT7 =	000200	729#	842	875	1035	1068							
BIT8 =	000400	728#	841	857	874	1034	1050	1067	3747	3749	3764	3766	
BIT9 =	001000	727#	840	856	1033	1049							
BPTVEC=	000014	743#											
BREAK =	000001	880#	1073#	1680	1681	1684	1685	1689	1691	1945	1960	1977	1989 2422
		2449	2485	2558	2631	2704	2772	2827					
BRW	016620	3634#											
BRX	016622	3635#											
CARDET=	010000	837#	1030#										
CCC	012576	2882	2905	2922#									
CHRCNT	014106	3177*	3182	3195*	3209#	3210							
CKSWR	012672	2876	2939	2946#	2978	3575							
CLK =	020000	872#	1065#	1824	1825	1828	1829	1833	1835	2422	2449	2485	2492 2493
		2495	2496	2558	2565	2566	2568	2569	2631	2638	2639	2641	2642 2704
		2711	2712	2714	2715	2772	2778	2779	2781	2782	2827	2833	2834 2836
		2837	3730	3731									
CNTLU	012746	1269	2957#										
CONVRT=	104413	2879	2959	3441	3680#								
COUNT	001124	781#	2497*	2501*	2570*	2574*	2643*	2647*	2716*	2720*			
CR =	000015	651#	3052	3062									
CRLF =	000200	652#	3023	3062									
CTS =	020000	836#	1029#										
DDISP =	177570	658#	914	1227									
DEVADR	013636	3096*	3134	3142#									
DEVICE	015302	2880	3472#										
DF1	002132	1098	1102	1106	1110	1156#							
DH1	002067	1096	1100	1104	1145#								
DISPLA	001442	914#	1227*	1235*	1260*	3247*	3631*						
DISPRE	000174	755#	1235	1260									

CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 81

L 6

CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0076

DNA = 100000	870#	1063#	2045											
DNAINT= 000040	877#	1070#	1740	1741	1744	1745	1749	1751						
DSC = 100000	834#	1027#	2024											
DSINTE= 000040	844#	1037#	1620	1621	1624	1625	1629	1631						
DSR = 001000	840#	1033#												
DSWR = 177570	657#	913	1226											
DTR = 000002	848#	1041#	1504	1505	1508	1509	1513	1519	1523	2194	2263	2348	2394	
DT1 002116	1097	1101	1105	1151#										
DT4 002126	1109	1154#												
DUADDR 016770	1301	2907	3700#											
DUBASE 017122	1297	1300	2906*	3700	3701*	3702	3703*	3704	3705	3706*	3707	3708	3709*	
	3710	3711*	3712	3713*	3714	3715*	3716	3718#						
DULEV 016714	1359	3687#												
DUPRT 016764	1358*	3687*	3688*	3689*	3690*	3691*	3692	3696#						
DURIS 001740	1125#	2910*												
DURIV 001736	1124#	1308	1311	1312	2908*	2915								
DUTIS 001744	1127#	2914*												
DUTIV 001742	1126#	2912*												
EIGHT = 006000	865#	1058#	2425	2452	2707									
EMTVEC= 000030	746#	1211*	1212*											
EM1 001762	1095	1131#												
EM2 002022	1099	1137#												
EM3 002043	1103	1140#												
EM4 001746	1107	1129#												
ERRCNT 001114	774#													
ERRVEC= 000004	739#	1224	1225*	1236*	3592	3593*	3595*	3598*						
EVENB 017276	3754#													
EVEPAR= 001400	868#	1061#												
EVPAR = 000400	857#	1050#												
FIVE = 000000	862#	1055#	2488	2775	2830									
FRMERR= 020000	853#	1046#	2079											
GNS = ***** U	3668	3669	3670	3671	3674	3675	3676	3677	3678	3679	3680	3681		
HDXEN = 000010	879#	1072#	1700	1701	1704	1705	1709	1711						
HILIM 013634	3095*	3125	3141#											
HLD0 001130	783#	1151	3278*											
HLD1 001132	784#	1151	3279*											
HLD2 001134	785#	3280*												
HLD3 001136	786#	3281*												
HLD4 001140	787#	3282*												
HLD5 001142	788#	3283*												
HLD6 001144	789#	3284*												
HOLD 001120	779#	1431*	1561	1885	2136	2199	2220	2245	2268	2283	2298	2322	2337	
	2353	2369	2384	2399										
HPARCS 001724	1118#	1456*	3708*											
HRXCSR 001714	1114#	1425*	1915*	3702*										
HRXDBU 001720	1116#	1441*	3707*											
HT = 000011	649#	3021	3062											
HTXCSR 001730	1120#	1471*	1926*	3712*										
HTXDBU 001734	1122#	1486*	3716*											
INBUF 016124	1367	1371	3071	3101	3219	3223	3550#							
INIFLG 001172	814#	1270	1273*											
INSTER= 104407	1375	3120	3228	3676#										
INSTR = 104406	1292	1303	1313	1324	1349	1365	1378	1382	1386	1390	1401	2961	3675#	
INSTR2 013440	3068	3080	3089#											
IOTVEC= 000020	744#	1209*	1210*											
ISYMOD= 000000	861#	1054#	2445	2452	2768	2775	2823	2830						

JMRBY	001153	797#	1278*	1393	1975	2183														
KEEPAD	001156	802#	1300*	1302	1346	1348	2902													
KEEPIV	001164	805#	1311*	2903																
LASTAD	001160	803#	1329	1340	1354															
LESS1	016766	3692*	3693*	3694*	3697#															
LF =	000012	650#	3056	3062																
LIGHTS	001102	764#	2927*																	
LIMITS	013566	3111	3125#																	
LOBITS	013640	3097*	3129	3143#	3144															
LOCK	001110	772#	2981																	
LOKFLG	001174	816#																		
LOLIM	013632	3094*	3127	3140#																
MASK1	007324	2251#																		
MASK2	007342	2257#																		
MASK3	007432	2289#																		
MCNTG	016070	2956	3540#																	
MCOW	015314	2885	3474#																	
MCRLF	015414	3173	3485#																	
MDATA	016230	3193	3204	3554#																
MEPASS	015463	2878	3493#																	
MEXT =	010000	884#	1077#	1977	1989															
MEXTJ	016030	1391	3534#																	
MINT =	004000	883#	1076#	1945	1960	2422	2449	2485	2558	2631	2704	2772	2827							
MLASTD	015250	1325	3467#																	
MLOCK	015527	3500#																		
MMNEW	016111	2961	3544#																	
MMSWR	016100	2958	3542#																	
MMULT	015221	1314	3463#																	
MPFAIL	015417	3444	3486#																	
MQM	015410	3087	3484#																	
MR	015510	1412	3497#																	
MRANGE	015344	1350	3478#																	
MREGAD	015171	1293	3458#																	
MRESET=	000400	874#	1067#	1514	1543	1581	1610	1630	1650	1670	1690	1710	1730	1750						
		1770	1792	1814	1834	1854	1896	1897	1900	1901	1910	1925	1959	1988						
		2001	2012	2023	2034	2044	2055	2067	2078	2089	2100	2111	2149	2162						
		2178	2417	2419	2444	2446	2480	2482	2553	2555	2626	2628	2699	2701						
		2767	2769	2822	2824															
MSYNC	015562	1366	3505#																	
MTDATA=	040000	871#	1064#	1844	1845	1848	1849	1853	1855	1945	1950	1958	1977	1982						
		1987	2422	2449	2485	2558	2631	2704	2772	2827	3723									
		1272	3451#																	
MTITLE	015122	1402	3498#																	
MTSTPC	015514	796#	1316	1317	2881															
MULTD	001152	1304	3456#																	
MVECTO	015156	1387	3526#																	
MWIRE4	015751	1383	3519#																	
MWIRE5	015701	1379	3512#																	
MWIRE6	015630	771#																		
NEXT	001106	866#	1059#	2425	2452	2488	2561	2634	2707	2775	2830									
NOPAR =	000000	867#	1060#																	
ODDPAR=	001000	3737#																		
ODDB	017212	1271	1274#																	
ONCE	002610	795#	1389	1517	1546	1584	1870	1911	2114											
OPTCLR	001151	2947	2949	2951	2955	2967#														
OUT	013000	2879	2918#																	
OUTCRY	012570																			

OUTMUL	003166	1322	1347	1358#															
OUT1	007774	2186	2314	2363	2409#														
OUT2	007520	2237	2317#																
OUT3	007650	2239	2364#																
OVR RUN=	040000	852#	1045#	2090															
PARAM =	104410	1294	1305	1326	1351	1403	2962	3677#											
PARAM1	013472	3100#	3121																
PARCSR	001722	1117#	1453*	2418*	2425*	2445*	2452*	2481*	2488*	2554*	2561*	2627*	2634*	2700*					
		2707*	2768*	2775*	2823*	2830*	3705*												
PAREN =	001000	856#	1049#																
PARER =	010000	854#	1047#	2068															
PARERR	013546	3103	3105	3107	3116#	3126	3128	3130											
PARTI	013630	3119	3139#																
PASCNT	001112	773#	1247*	2926*	2927	2928													
PFTAB	015114	3442	3448#																
PIRQ =	177772	656#																	
PIRQVE=	000240	750#																	
POPRO =	012600	829#	1022#																
POP1SP=	005726	827#	1020#																
POP2SP=	022626	831#	1024#																
PR0 =	000000	673#																	
PR1 =	000040	674#																	
PR2 =	000100	675#																	
PR3 =	000140	676#																	
PR4 =	000200	677#	3697																
PR5 =	000240	678#	3696																
PR6 =	000300	679#																	
PR7 =	000340	680#	1421	1437	1452	1467	1482												
PS =	177776	653#	654																
PSW =	177776	654#																	
PUSHRO=	010046	828#	1021#																
PUSH1S=	005746	826#	1019#																
PUSH2S=	024646	830#	1023#																
PWRVEC=	000024	745#	1215*	1216*															
RDSW	013006	2957*	2967*	2969#	3118														
REACT=	004000	838#	1031#	2013	2426	2430	2434	2453	2457	2461	2509	2518	2522	2582					
		2591	2595	2655	2664	2668	2728	2737	2741										
REPLAY	012474	2898	2904	2906#															
RESTAR	015020	3422	3428#																
RESTR	012654	2899	2930	2937#															
RESVEC=	000010	740#																	
RES05 =	104412	3679#																	
RETURN	001104	770#	1252*	1409*	1411*	1413	3447												
RING =	040000	835#	1028#																
RINTEN=	000100	843#	1036#	1640	1641	1644	1645	1649	1651										
ROTADD	001170	809#	1321*	1333*	1335	1337*	1342	1345*	2893*	2896	2900*								
RPOKE	017124	2500	2514	2573	2587	2646	2660	2719	2733	2787	2799	2803	2842	2854					
		2858	3723#	3733															
		847#	1040#	1533	1534	1537	1538	1542	1548	1552	2215	2278	2332	2379					
RTS =	000004	1	54	143	516														
RUNA =	000000																		
RUNB =	***** U	21	59	146	521														
RUNC =	***** U	21	59	146	521														
RUND =	***** U	21	59	146	521														
RUNE =	***** U	21	59	146	521														
RUNF =	***** U	21	59	146	521														
RUNIT	012400	2884	2890#	2897															

CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 84

B 7

CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0079

RXCSR 001712

1113#	1422*	1504*	1505	1508*	1509	1513*	1519	1523	1533*	1534	1537*	1538
1542*	1548	1552	1571*	1572	1575*	1576	1580*	1586	1590	1600*	1601	1604*
1605	1609*	1611	1620*	1621	1624*	1625	1629*	1631	1640*	1641	1644*	1645
1649*	1651	1660*	1661	1664*	1665	1669*	1671	1864*	1868	1913*	1914	1916
1920*	1921	2002	2013	2024	2110*	2112	2113	2177*	2188	2189	2194*	2204
2209	2215*	2225	2230	2240*	2250	2256	2263*	2273	2278*	2288	2293*	2303
2308	2317*	2327	2332*	2342	2348*	2358	2364*	2374	2379*	2389	2394*	2404
2426	2429*	2430	2433*	2434	2453	2456*	2457	2460*	2461	2490*	2506	2509
2515	2518	2521*	2522	2525	2533	2563*	2579	2582	2588	2591	2594*	2595
2598	2606	2636*	2652	2655	2661	2664	2667*	2668	2671	2679	2709*	2725
2728	2734	2737	2740*	2741	2744	2752	2776*	2788	2831*	2843	2920	3700*
1115#	1438*	2056	2058	2068	2079	2090	2101	2163	2164	2489	2529	2562
2602	2635	2675	2708	2748	2783	2791	2806	2838	2846	2861	3704*	
842#	1035#	2002										
851#	1044#	2101										
3257	3278#											
782#	3148*											
775#	3421*	3428										
3678#												
3674#												
878#	1071#	1720	1721	1724	1725	1729	1731					
794#	1385	2238										
1315	1380	1384	1388	1392	3681#							
864#	1057#	2634										
793#	1381	2236										
780#	2498*	2512*	2571*	2585*	2644*	2658*	2717*	2731*	2785*	2797*	2801*	2840*
2852*	2856*	3732*										
863#	1056#	2561										
3178*	3197	3200*	3210#									
839#	1032#											
644#	1207	1243	1397	3437								
846#	1039#	1571	1572	1575	1576	1580	1586	1590	2240	2293	2317	2364
815#	1246*											
655#												
841#	1034#	1660	1661	1664	1665	1669	1671					
3152#												
913#	1205	1226*	1228	1234*	1241*	1257	1259*	1264	1280	1399	2948	2979
3248	3255	3267	3270	3580	3587	3601	3603	3609	3616			
756#	1234	1259	1264	2948	2965	2972						
2960	2970#											
708#												
698#	708	1280										
697#	707	1399										
696#	706											
695#	705											
694#	704											
693#	703											
692#	702											
691#	701											
690#	700											
689#	699	2979										
707#												
688#												
687#												
686#												
685#												

RXDBUF 001716

RXDONE= 000200

RXERR = 100000

SAVIT 014342

SAVPC 001126

SAVSP 001116

SAVOS = 104411

SCOP1 = 104405

SEND = 000020

SEREC 001150

SETFLG= 104414

SEVEN = 004000

SEXMIT 001147

SHIFT 001122

SIX = 002000

SPACNT= 014107

SRD = 002000

STACK = 001100

STD = 000010

STFLG 001173

STKLMT= 177774

STPSYN= 000400

SVOS 013650

SWR 001440

SWREG 000176

SWREGL 013010

SW0 = 000001

SW00 = 000001

SW01 = 000002

SW02 = 000004

SW03 = 000010

SW04 = 000020

SW05 = 000040

SW06 = 000100

SW07 = 000200

SW08 = 000400

SW09 = 001000

SW1 = 000002

SW10 = 002000

SW11 = 004000

SW12 = 010000

SW13 = 020000

SW14	=	040000	684#																	
SW15	=	100000	683#																	
SW2	=	000004	706#																	
SW3	=	000010	705#																	
SW4	=	000020	704#																	
SW5	=	000040	703#																	
SW6	=	000100	702#																	
SW7	=	000200	701#																	
SW8	=	000400	700#																	
SW9	=	001000	699#																	
SYNCNO		001146	792#	1369*	1373*	2504	2577	2650	2723											
SYNEXT=		020000	860#	1053#	2418	2425														
SYNINT=		030000	859#	1052#	2481	2488	2554	2561	2627	2634	2700	2707								
SYNSCH=		000020	845#	1038#	1600	1601	1604	1605	1609	1611	2429	2433	2456	2460	2490					
			2521	2563	2594	2636	2667	2709	2740	2776	2831									
SYSTST=		014000	885#	1078#																
TBITVE=		000014	741#																	
TEMP		016166	3183	3438*	3439*	3552#														
TKVEC =		000060	748#																	
TPVEC =		000064	749#																	
TRAPVE=		000034	747#	1213*	1214*															
TRPREG		017362	1420	1436	1451	1466	1481	3769#												
TRTVEC=		000014	742#																	
TST1		003374	1404	1411	1419#	2938	3577													
TST10		004054	1532#																	
TST11		004200	1570#																	
TST12		004314	1599#																	
TST13		004404	1619#																	
TST14		004474	1639#																	
TST15		004564	1659#																	
TST16		004654	1679#																	
TST17		004744	1699#																	
TST2		003454	1435#																	
TST20		005034	1719#																	
TST21		005124	1739#																	
TST22		005214	1759#																	
TST23		005304	1781#																	

2099冊
2109冊
2147冊
1480冊
2161冊
2176冊
2416冊
2443冊
2479冊
2552冊
2625冊
2698冊
1495冊
2766冊
2821冊
1503冊
3580冊

1468*	1514*	1543*	1581*	1610*	1630*	1650*	1670*	1680*	1681	1684*	1685
1690*	1691	1700*	1701	1704*	1705	1709*	1710*	1711	1720*	1721	1724*
1729*	1730*	1731	1740*	1741	1744*	1745	1749*	1750*	1751	1760*	1761
1765	1769*	1770*	1771	1782*	1783	1786*	1787	1791*	1792*	1793	1804*
1808*	1809	1813*	1814*	1815	1824*	1825	1828*	1829	1833*	1834*	1835
1845	1848*	1849	1853*	1854*	1855	1865*	1869	1896*	1897	1900*	1901
1925*	1927	1933*	1934	1945*	1947	1950*	1954	1958*	1959*	1960*	1964
1979	1982*	1983	1987*	1988*	1989*	1990	2001*	2012*	2023*	2034*	2035
2045	2055*	2067*	2078*	2089*	2100*	2111*	2148*	2149*	2150	2151	2162*
2417*	2419*	2422*	2444*	2446*	2449*	2480*	2482*	2485*	2492*	2493*	2495*
2553*	2555*	2558*	2565*	2566*	2568*	2569*	2626*	2628*	2631*	2638*	2639*
2642*	2699*	2701*	2704*	2711*	2712*	2714*	2715*	2767*	2769*	2772*	2778*
2781*	2782*	2822*	2824*	2827*	2833*	2834*	2836*	2837*	3710*	3723*	3729*

```
TXDBUF 001732
TXDONE= 000200
TXINTE= 000100
TYPE   = 104401
```

1121#
875#
876#
1272
3258
3302
3671#
3670#
882#
849#
3176*
2891#
1178
584
555
553#
554#
556#
910#
988#
905#
907#
939#
990#
991#

[illegible]

[illegible]

[illegible]

CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 89

G 7

CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0084

		3607	3614	3615	3616	3628	3631	3636						
\$SWREG	001550	960#	1241											
\$SWRMK=	000000	3566	3567	3603										
\$TESTN	001532	951#	3626*											
\$TIMES	001512	937#	1217*	3614*	3621	3624*	3636							
\$TKB	001446	916#	2952	3075	3083	3586								
\$TKS	001444	915#	2950	3073	3584									
\$TMP0	001476	931#												
\$TMP1	001500	932#	2499*	2513*	2572*	2586*	2645*	2659*	2718*	2732*	2786*	2798*	2802*	2841*
		2853*	2857*	3725*	3737	3747*	3749*	3754	3764*	3766*				
\$TMP2	001502	933#	3724*	3726*	3727*	3728*	3729	3737*	3741*	3754*	3758*			
\$TMP3	001504	934#	3738*	3742*	3745*	3755*	3759*	3762*						
\$TMP4	001506	935#												
\$TMP5	001510	936#												
\$TN =	000062	607#	1418	1420#	1434	1436#	1449	1451#	1464	1466#	1479	1481#	1494	1496#
		1502	1504#	1531	1533#	1569	1571#	1598	1600#	1618	1620#	1638	1640#	1658
		1660#	1678	1680#	1698	1700#	1718	1720#	1738	1740#	1758	1760#	1780	1782#
		1802	1804#	1822	1824#	1842	1844#	1862	1864#	1894	1896#	1908	1910#	1943
		1945#	1971	1973#	1999	2001#	2010	2012#	2021	2023#	2032	2034#	2042	2044#
		2053	2055#	2065	2067#	2076	2078#	2087	2089#	2098	2100#	2108	2110#	2146
		2148#	2160	2162#	2175	2177#	2415	2417#	2442	2444#	2478	2480#	2551	2553#
		2624	2626#	2697	2699#	2765	2767#	2820	2822#					
\$TPB	001452	918#	3051*	3062	3083*									
\$TPFLG	001457	922#	3000	3062										
\$TPS	001450	917#	3049	3062	3081									
\$TRAP	016626	1213	3645#											
\$TRAP2	016650	3656#	3667											
\$TRP =	000015	3660#	3669#	3670#	3671#	3672#	3674	3675#	3676#	3677#	3678#	3679#	3680#	3681#
		3682#												
\$TRPAD	016662	3650	3667#											
\$TSTM	002142	1187#												
\$TSTNM	001402	895#	1251*	1406	1409	2923*	3247	3278	3284	3561	3582	3603	3625*	3626
		3631	3637											
\$STYPBN=	***** U	3672												
\$STYPDS=	***** U	3672												
\$STYPE	013042	581	3000#	3660	3668									
\$STYPEC	013254	3030	3037	3044	3049#	3050								
\$STYPEX	013322	3055	3057	3060#										
\$STYPOC	014560	3363#	3669											
\$STYPON	014574	3362	3365#	3671										
\$STYPOS	014534	3358#	3670											
\$SUNIT	001540	954#												
\$SUNITM	002146	1189#												
\$SUSWR	001552	961#	1276											
\$VECT1	001576	986#												
\$VECT2	001600	987#												
\$XTSTR	016354	3578	3590#											
\$OFILL	014757	3359*	3363*	3373	3408#									
\$40CAT=	***** U	3257	3587											
.	= 017370	602#	641#	754#	757#	762#	817#	818#	892#	943	1149#	1163	1164#	1166#
		1168#	1174	1175#	1177#	1179#	1206	1220	1221	1423	1426	1439	1442	1454
		1457	1469	1472	1484	1487	1497	1506	1510	1520	1524	1535	1539	1549
		1553	1563	1573	1577	1587	1591	1602	1606	1612	1622	1626	1632	1642
		1646	1652	1662	1666	1672	1682	1686	1692	1702	1706	1712	1722	1726
		1732	1742	1746	1752	1762	1766	1772	1784	1788	1794	1806	1810	1816
		1826	1830	1836	1846	1850	1856	1875	1879	1887	1898	1902	1918	1923

SEQ 0085

[illegible]

COMMON ENDCOM ERROR	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#	1# 751# 1# 751#
645#	1424	1427	1440	1443	1455	1458	1470	1473	1485	1488	1498	1507	1511	1521	
1525	1536	1540	1550	1554	1574	1578	1588	1592	1603	1607	1613	1623	1627	1633	
1643	1647	1653	1663	1667	1673	1683	1687	1693	1703	1707	1713	1723	1727	1733	
1743	1747	1753	1763	1767	1773	1785	1789	1795	1807	1811	1817	1827	1831	1837	
1847	1851	1857	1876	1880	1899	1903	1919	1924	1932	1938	1949	1956	1966	1981	
1985	1992	2004	2015	2026	2037	2047	2061	2070	2081	2092	2103	2121	2128	2155	
2168	2193	2208	2213	2229	2234	2255	2261	2277	2292	2307	2312	2331	2347	2362	
2378	2393	2408	2428	2432	2436	2455	2459	2463	2508	2511	2517	2520	2524	2527	
2532	2535	2581	2584	2590	2593	2597	2600	2605	2608	2654	2657	2663	2666	2670	
2673	2678	2681	2727	2730	2736	2739	2743	2746	2751	2754	2790	2794	2809	2845	
2849	2864														
ESCAPE	1#	751#													
GETPRI	1#	751#													
GETSWR	1#	751#													
MULT	1#	751#													
NEWTST	1#	751#	1418	1434	1449	1464	1479	1494	1502	1531	1569	1598	1618	1638	1658
1678	1698	1718	1738	1758	1780	1802	1822	1842	1862	1894	1908	1943	1971	1999	
2010	2021	2032	2042	2053	2065	2076	2087	2098	2108	2146	2160	2175	2415	2442	
2478	2551	2624	2697	2765	2820										
POP	1#	596	597	751#											
PRGEND	629#	2868													
PRGFRT	608#	630													
PUSH	1#	557	559	580	751#										
PUSSYF	626#														
REPORT	1#	539#	751#												
RSETUP	613#	2417	2444	2480	2553	2626	2699	2767	2822						
SC	548#	3571													
SCOPE	646#	1419	1435	1450	1465	1480	1495	1503	1532	1570	1599	1619	1639	1659	1679

\$BINAR	625#														
\$BUFFE	607#	3547													
\$CABLE	627#	1557	1881	2132	2195	2216	2241	2264	2279	2294	2318	2333	2349	2365	2380
	2395														
\$CATCH	542#	641													
\$CK	549#	2941													
\$CLRVE	548#	1274													
\$CONVR	607#	3169													
\$DNA	623#														
\$EOP	549#	2868													
\$GETFL	548#	1313	1378	1382	1386	1390									
\$GETPA	548#	1292	1303	1323	1349	1401									
\$GETSY	548#	1360													
\$HEADE	539#	630													
\$INSTR	607#	3062													
\$ISOB	621#														
\$MATCH	618#														
\$MRR	612#	1996	2007	2018	2039	2062	2073	2084	2095						
\$MRRW	610#	1499	1528	1566	1595	1615	1635	1655	1675	1695	1715	1735	1755	1777	1799
	1819	1839													
\$MRW	611#	1891													
\$MSG	607#	3451													
\$NEED	628#	3278													
\$PARAM	607#	3090													
\$PFAIL	607#	3410													
\$POKE	616#														
\$POKER	617#	2494	2567	2640	2713	2780	2835								
\$RCNET	628#														
\$RECAC	616#	2465	2538	2611	2684										
\$REG	607#	3145													
\$RESET	609#	1514	1543	1581	1610	1630	1650	1670	1690	1710	1730	1750	1770	1792	1814
	1834	1854	1900	1910	1925	1959	1988	2001	2012	2023	2034	2044	2055	2067	2078
	2089	2100	2111	2149	2162	2178	2417	2419	2444	2446	2480	2482	2553	2555	2626
	2628	2699	2701	2767	2769	2822	2824								
\$RXACT	615#	2411	2438												
\$SCOPE	548#	3556													
\$SCOP1	549#	2975													
\$SETFL	548#	3212													
\$SETVE	543#	751													
\$START	548#	1192													
\$STRIP	622#														
\$SYMB0	539#	819	1012												
\$SYNCR	618#	2490	2563	2636	2709	2776	2831								
\$TRPAR	624#														
\$TSTNO	548#	1418	1434	1449	1464	1479	1494	1502	1531	1569	1598	1618	1638	1658	1678
	1698	1718	1738	1758	1780	1802	1822	1842	1862	1894	1908	1943	1971	1999	2010
	2021	2032	2042	2053	2065	2076	2087	2098	2108	2146	2160	2175	2415	2442	2478
	2551	2624	2697	2765	2820										
\$UNIBU	608#	1416	1432	1447	1462	1477									
\$VARIA	546#	761													
\$WORDF	619#														
\$WORDO	618#	2757	2812												
\$WORDP	620#														
\$SCMRE	886#	925	926	927	928	929	930								
\$SCMTM	886#	931	932	933	934	935	936								
\$SESCA	1#	751#													

SSNEWT	1#	751#	1418	1434	1449	1464	1479	1494	1502	1531	1569	1598	1618	1638	1658
	1678	1698	1718	1738	1758	1780	1802	1822	1842	1862	1894	1908	1943	1971	1999
	2010	2021	2032	2042	2053	2065	2076	2087	2098	2108	2146	2160	2175	2415	2442
	2478	2551	2624	2697	2765	2820									
SSSET	3660#	3669	3670	3671	3674	3675	3676	3677	3678	3679	3680	3681			
SSSETM	1238#														
SSSKIP	1#	751#													
.EQUAT	1#	539#	641												
.HEADE	1#	539#													
.KT11	1#														
.SETUP	1#	539#	1191												
.SWRHI	1#														
.SACT1	1#	539#	1159												
.SAPT8	1#	539#	944#												
.SAPTH	1#	539#	1169												
.SAPTY	1#	539#	550												
.SASTA	1#														
.SCATC	1#	539#													
.SCMTA	1#	539#	886												
.SDB2D	1#														
.SDB2O	1#														
.SDIV	1#														
.SEOP	1#	539#													
.SERRO	1#	539#	3230												
.SERRT	1#	539#	3286												
.SMULT	1#														
.SPOWE	1#	539#													
.SRAND	1#														
.SRDDE	1#														
.SRDOC	1#														
.SREAD	1#														
.SR2AZ	1#														
.SSAVE	1#														
.SSB2D	1#														
.SSB2O	1#														
.SSCOP	1#	539#	3556												
.SSIZE	1#														
.SSUPR	1#														
.STRAP	1#	539#	3637												
.STYPB	1#														
.STYPD	1#														
.STYPE	1#	539#	2983												
.STYPO	1#	539#	3333												
.S4OCA	1#														
.1170	1#														

. ABS. 017370 000

ERRORS DETECTED: 0

CZDUQC,CZDUQC/CRF/SOL/NL:TOC=CZDUQC.SML,CZDUQ3/EQ:RUNA,CZDUQ4,CZDUQC.M11
RUN-TIME: 50 63 4 SECONDS
RUN-TIME RATIO: 229/119=1.9
CORE USED: 43K (85 PAGES)

CZDUQ-CO
CZDUQC.M11

MACY11 30A(1052)
20-AUG-80 09:19

09-SEP-80 10:57 PAGE 95
CROSS REFERENCE TABLE -- MACRO NAMES

L 7

SEQ 0089