

DL11-C,D,E

OFF-LINE TEST
MD-11-DZDLC-A

EP DZDLC-A-DL-A
COPYRIGHT 1976
FICHE 1 OF 1

NOV 1976
digital
MADE IN USA

This microfiche card contains a grid of frames, each displaying test data for the MD-11-DZDLC-A system. The data is organized into columns and rows, with some frames containing headers such as 'TEST', 'TIME', and 'STATUS'. The frames are arranged in a grid that is approximately 12 columns wide and 15 rows high. The data appears to be a series of test results or status reports, with some frames showing numerical values and others showing text-based information. The overall layout is a structured table of test data.

DL11-C,D,E
MD-11-DZDLC-A
NOV 1976
FICHE 1 OF 1

1.0 PROGRAM PURPOSE (ABSTRACT)

THIS PROGRAM HAS THE ABILITY TO TEST THE DL11 (ASYNCHRONOUS MODEM INTERFACE), OFF LINE. MODELS ABLE TO BE TESTED ARE C, D, AND E ONLY. THE USE OF A MODEM IS NOT REQUIRED FOR TESTING; HOWEVER, A SPECIAL CABLE CONNECTOR BC50C AND A SPECIAL MODEM TEST CONNECTOR H315A IS REQUIRED. THIS PROGRAM IS CAPABLE OF THE FOLLOWING:

- A. VERIFICATION OF MAINTENANCE BIT
- B. VERIFICATION THAT TRANSMITTER CAN CAUSE AN INTERRUPT
- C. VERIFICATION THAT RECEIVER 'DONE' CAN CAUSE AN INTERRUPT
- D. CHECKS THAT 'REQ TO SEND' ASSERTS 'RING'
- E. CHECKS THAT 'SEC XMIT' ASSERTS 'SEC REC' AND 'DATA SET INT'
- F. CHECKS THAT 'DTR' CAN ASSERT 'CLR TO SEND' AND 'CAR DET'
- G. VERIFIES THAT 'DATA SET I.E.' CAN CAUSE A RECP INTR
- H. CHECKS THE 'BREAK' FEATURE
- I. PERFORMS NULL-DEL-NUL PATTERN
- J. PERFORMS BINARY UP COUNT PATTERN
- K. PERFORMS BINARY DOWN COUNT PATTERN
- L. RUNS A WORSE CASE PATTERN

INCLUDED IN THE PROGRAM ARE SPECIAL USER ROUTINES - PRG #2, PRG #3, PRG #4, AND PRG #5 (WHICH WILL BE DESCRIBED FURTHER INTO THIS DOCUMENT).

NOTE WELL TWO(2) POINTS:

1. THIS PROGRAM IS CAPABLE OF TESTING SIXTEEN(16) DL11'S AND ASSUMES CONTIGUOUS ADDRESSING FROM 1ST DEVICE TO LAST.
 - A. IF MULTIPLE DEVICES ARE NOT BEING TESTED, THIS NOT REQUIRING A PASS THRU THE PROGRAM ONCE PER DEVICE, THEN THE PROGRAM WILL DEFAULT TO TESTING THE 1ST POSSIBLE DL11-E DEVICE I.E., RCSR ADDRESS = 775610, AND TEST THIS DEVICE ONLY.
 - B. IF MULTIPLE DEVICE TESTING IS NOT BEING CONDUCTED, AND THE DEVICE EXISTING IS NOT THE DEFAULT DL11-E, THEN THE USER ON STARTING THE PROGRAM WILL HAVE TO SET SW<D>=1 TO ENTER THE QUESTION & ANSWER MODE.
2. THIS PROGRAM HAS PROVISION FOR CHARACTER LENGTH I.E. IT ASSUMES DATA IS 9 BITS, BUT ALSO HAS THE ABILITY TO HANDLE 5, 6, OR 7 BITS OF DATA AS WELL.

2.0 SYSTEM REQUIREMENTS

A. HARDWARE REQUIREMENTS

001
002
003
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029
030
031
032
033
034
035
036
037
038
039
040
041
042
043
044
045
046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100

001
002
003
004
005
006
007
008
009
010
011
012
013
014
015
016
017
018
019
020
021
022
023
024
025
026
027
028
029
030
031
032
033
034
035
036
037
038
039
040
041
042
043
044
045
046
047
048
049
050
051
052
053
054
055
056
057
058
059
060
061
062
063
064
065
066
067
068
069
070
071
072
073
074
075
076
077
078
079
080
081
082
083
084
085
086
087
088
089
090
091
092
093
094
095
096
097
098
099
100

E01

MAINDEC-11-D2DLC-A
D2DLC.A.P11

MACY11 27(732) 20-SEP-76 09:19 PAGE 4

154
155

PDP-11 FAMILY PROCESSOR WITH 8K OF MEMORY
M7800 DL11 ASYNCHRONOUS LINE INTERFACE MODULE

BC05C SPECIAL CABLE CONNECTOR
H315A SPECIAL MODEM TEST CONNECTOR

B. SOFTWARE REQUIREMENTS

THIS PROGRAM WAS SPECIFICALLY DESIGNED FOR THE 11/40 FRONT END C. 1080 CONSOLE PROCESSOR SYSTEM. IN THIS ENVIRONMENT IT WOULD BE LOADED BY THE TCDP (DECTAPE) DIAGNOSTIC MONITOR. HOWEVER, ANY 11/40 USER WITH 8K OF MEMORY CAN RUN THIS PROGRAM TO TEST ONE(1) OR MULTIPLE DL11'S.

THE PROGRAM HAS THE PROPER INTERFACE CODE TO ALLOW RUNNING UNDER THE AUTOMATED MANUFACTURING TEST LINE SYSTEM - ACT11.

3.0 RELATED DOCUMENTS AND STANDARDS

- A. PROGRAMMING PRACTICES - DOCUMENT NO. 175-003-009-00
- B. PDP11/40 PROCESSOR HANDBOOK
- C. DL11 ASYNCHRONOUS LINE INTERFACE MANUAL
DOCUMENT NO. DEC-11-HDLAA
- D. PDP-11 MAINDEC SYSMAC UTILITY PACKAGE
-
MAINDEC-11-DZQAC-A1
- E. APPLICABLE CIRCUIT SCHEMATIC
M7800

4.0 DIAGNOSTIC HIERARCHY PREREQUISITES

BEFORE RUNNING THIS PROGRAM, THE FOLLOWING TWO(2) DIAGNOSTIC PROGRAMS SHOULD BE RUN FOR VERIFICATION OF FUNCTIONALITY OF THE 11-INSTRUCTION SET AND MEMORY:

- 1. MAINDEC-11-DBQEA AND.
- 2. MAINDEC-11-DZQMC

5.0 LOADING AND STARTING PROCEDURE

LOAD PROGRAM IN MEMORY USING ABS LOADER
LOAD ADDRESS 200.

NOTE

IN THE CASE OF A 1080 SYSTEM ENVIRONMENT
LOAD THE PROGRAM USING THE TCDP
(DECTAPE) DIAGNOSTIC MONITOR.

PRESS START.

156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200

GO1

MAINDEC-11-DZDLC-A
DZDLCR.P11

MACY11 27(732) 20-SEP-76 09:19 PAGE 6

212
213

A. THERE ARE ALSO THREE(3) OPTIONAL START ADDRESSES FOR THE

MAINDEC-11-DZDLC-A
DZDLCR.P11

MACY11 27(732) 20-SEP-76 09:19 PAGE 8

270
271

8.0 EXECUTION TIMES

5-2-76

K01

MAINDEC-11-DZDLC-A
DZDLC.A.P11

MACY11 27(732) 20-SEP-76 09:19 PAGE 10

328

TO ALLOW THE INFORMATION TRANSFER.

MO1

MAINDEC-11-DZDLC-A
DZDLCR.F11

MACY11 27(732) 20-SEP-76 09:19 PAGE 12

395
395

OF CURRENT TRANSMISSION OR
AN ERROR CONDITION.

635	001224	000000	STMP11: .WORD	0	:: USER DEFINED
636	001226	000000	STMP12: .WORD	0	:: USER DEFINED
637	001230	000000	STMP13: .WORD	0	:: USER DEFINED
638	001232	000000	STMP14: .WORD	0	:: USER DEFINED
639	001234	000000	STMP15: .WORD	0	:: USER DEFINED
640	001236	000000	STMP16: .WORD	0	:: USER DEFINED
641	001240	000000	STMP17: .WORD	0	:: USER DEFINED
642	001242	000000	STIMES: 0		:: MAX. NUMBER OF ITERATIONS
643	001244	000000	\$ESCAPE: 0		:: ESCAPE ON ERROR ADDRESS
644	001246	177607 000377	\$BELL: .ASCIZ	<207><377><377>	:: CODE FOR BELL
645	001252	077	\$QUES: .ASCII	/?	:: QUESTION MARK
646	001253	015	\$CRLF: .ASCII	<15>	:: CARRIAGE RETURN
647	001254	000012	\$LF: .ASCIZ	<12>	:: LINE FEED
648			:*****		
649			:THE FOLLOWING TAG(S) ARE USER SUPPLIED BY CALLING THE MACRO		
650			: 'MORETAGS' AS ONE OF THE ARGUMENTS TO THE SYSMAC ROUTINE .SCMTAG		
651					
652					
653	001256	000000	TABFLG: .WORD	0	: AN INDICATOR TO SHOW THAT THE
654					: INFORMATION FOR MULTIPLE DEVICE
655					: TESTING HAS ALREADY TRANSPIRED
656					: & 'MAINDEC' NAME HAS BEEN PRINTED
657	001260	000000	DLBASE: .WORD	0	: STORAGE & WORKING LOCATION FOR A DEVICE
658					: RECEIVER STATUS REGISTER ADDRESS
659	001262	000000	KEEPPAD: .WORD	0	: STORAGE LOCATION FOR THE 1ST
660					: DEVICE RCSR FROM WHICH
661					: 'BASEADD' IS RESTORED AT THE
662					: END OF A COMPLETE PROGRAM PASS.
663	001264	000000	BASEADD: .WORD	0	: STORAGE LOCATION WHICH HOLDS
664					: THE RCSR ADDRESS OF THE 'NEXT'
665					: DEVICE DURING MULTIPLE TESTING
666	001266	000000	KEEPIV: .WORD	0	: STORAGE LOCATION FOR THE 1ST
667					: DEVICE RECEIVER VECTOR FROM
668					: WHICH 'BASEIV' IS RESTORED AT THE
669					: END OF A COMPLETE PROGRAM PASS
670	001270	000000	BASEIV: .WORD	0	: STORAGE LOCATION WHICH HOLDS
671					: THE VECTOR ADDRESS OF THE 'NEXT'
672					: DEVICE DURING MULTIPLE TESTING
673	001272	000000	MULTD: .WORD	0	: FLAG TO INDICATE TO 'END OF PASS'
674					: ROUTINE THAT MULTIPLE DEVICE
675					: TESTING IS BEING CONDUCTED
676					: 0=NO. 1=YES
677	001274	000000	ACTREG: .WORD	0	: THIS IS THE DEVICE ACTIVE REGISTER
678					: A BIT IS SET (STARTING AT
679					: BIT) FOR EACH CONTIGUOUS DEVICE
680					: (A MAX. OF 16) THAT IS TO UNDERGO
681					: TESTING. THIS LOCATION IS
682					: AUTOMATICALLY FILLED BASED ON
683					: USER RESPONSE TO PROGRAM QUESTIONS
684	001276	000000	ROTADD: .WORD	0	: A ROTATING POINTER TO SIGNAL
685					: THE LAST DEVICE TESTED (IF
686					: MULTIPLE DEVICE TESTING WAS BEING
687					: DONE) IF LESS THAN A FULL COMPLE-
688					: MENT OF DEVICES (16) WAS SELECTED
689	001280	000000	LASTADD: .WORD	0	: STORAGE LOCATION FOR THE
690					: RCSR ADDRESS OF THE LAST DEVICE

691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710

001302 000000

DLPRI: .WORD 0

001304 000000

LESS1: .WORD 0

001306 177740

STLMSK: 177740

:TESTED (IF MULTIPLE DEVICE
:TESTING WAS SELECTED BY USER)
:STORAGE LOCATION FOR THE DEVICE
:INTERRUPT PRIORITY LEVEL
:THE PRIORITY LEVEL THE CPU
:MUST BE AT TO ALLOW DEVICE INTERRUPTS.
:THIS WILL BE 1 LEVEL LESS THAN
:THE DEVICE LEVEL (BASED ON &
:CALCULATED FROM USER RESPONSE TO
:DEVICE PRIORITY LEVEL QUESTION)
:THIS MASK IS USED BY THE 'STALL'
:ROUTINE WHICH WAITS A RANDOM NO.
:OF MILLISECONDS. ITS USE PREVENTS
:A STALL > 37 MSEC. THIS LOCATION
:HOWEVER, CAN BE PATCHED BY THE
:USER TO ALLOW LARGER 'STALLS'.

:END OF USER SUPPLIED TAGS.

709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764

.SBTTL ERROR POINTER TABLE

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

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

\$ERRTB:

:ERROR TABLE ITEM FOR ERROR MESSAGE 1

EM1 : "DL11 REGISTER REFERENCE CAUSED TIMEOUT"
DH1 : " (PC) (PS) (SP) TEST DEVADR REGADR "
DT1 : (R7) (PSW) (R6) (R0) (R1) (R2)
0 :PRINT ALL OCTAL

:ERROR TABLE ITEM FOR ERROR MESSAGE 2

EM2 : " DL11 REGISTER ERROR "
DH2 : " (PC) (PS) (SP) TEST DEVADR REGADR WAS S/B "
DT2 : " (R7) (PSW) (R6) (R0) (R1) (R2) (R3) (R4) "
0 :PRINT ALL OCTAL

:ERROR TABLE ITEM FOR ERROR MESSAGE 3

EM3 : " DL11 DATA COMPARE ERROR "
DH3 : " (PC) (PS) (SP) TEST WASADR SHBADR WAS S/B "
DT3 : " (R7) (PSW) (R6) (R0) (R1) (R2) (R3) (R4) "
0 :PRINT ALL OCTAL

:ERROR TABLE ITEM FOR ERROR MESSAGE 4

EM4 : " UNEXPECTED TRAP TO VECTOR AT LOCATION XXX "
DH4 : " (PC) (PS) (SP) TEST "
DT4 : " (R7) (PSW) (R6) (R0) "
0 :PRINT ALL OCTAL

:ERROR TABLE ITEM FOR ERROR MESSAGE 5

EM5 : " DL11 SOFT ERROR (PARITY, FRAMING, OR OVERRUN)
DH5 : " (PC) (PS) (SP) TEST DEVADR REGADR (REG) "
DT5 : " (R7) (PSW) (R6) (R0) (R1) (R2) (R3) "
0

:ERROR TABLE ITEM FOR ERROR MESSAGE 6

EM6 : "DL11 REGISTER REFERENCE CAUSED TIMEOUT"
DH6 : " (PC) (PS) (SP) REGADR"
DT6 : \$ERRPC, \$TMPO, \$REG6, \$REG2

001310

001310 015132
001312 015201
001314 015260
001316 000000

001320 015276
001322 015322
001324 015420
001326 000000

001330 015442
001332 015472
001334 015570
001336 000000

001340 015612
001342 015664
001344 015722
001346 000000

001350 015734
001352 016011
001354 016100
001356 000000

001360 015132
001362 016120
001364 016160

```

765 001366 000000          0          ;PRINT ALL OCTAL
766                                     ;ERROR TABLE ITEM FOR ERROR MESSAGE 7
767
768
769 001370 015734          EMS          ;" DL11 SOFT ERROR (PARITY, FRAMING, OR OVERRUN) "
770 001372 016172          DH7          ;" (PC) DEVADR REGADR (REG)"
771 001374 016232          DT7          ;$ERRPC,$REG1,$REG2,$REG3
772 001376 000000          0          ;PRINT ALL OCTAL
773
774                                     ;ERROR TABLE ITEM FOR ERROR MESSAGE 10
775
776 001400 015442          EM3          ;" DL11 DATA COMPARE ERROR "
777 001402 016244          DH10         ;" (PC) DEVADR REGADR (REG) S/B"
778 001404 016312          DT10         ;$ERRPC,$REG1,$REG2,$REG3,$REG4
779 001406 000000          0          ;PRINT ALL OCTAL
780
781 ::*****
782 :DL11 DEFINITIONS
783 ::*****
784
785 001410 175610          DLRCR: 175610          ;CONTAINS ADDRESS OF RCVR CSR
786 001412 175612          DLRDBR: 175612          ;CONTAINS ADDRESS OF RCVR DBR
787 001414 175614          DLXCSR: 175614          ;CONTAINS ADDRESS OF XMIT CSR
788 001416 175616          DLXDBR: 175616          ;CONTAINS ADDRESS OF XMIT DBR
789 001420 000300          DLVECT: 300          ;CONTAINS VECTOR ADDRESS OF CURRENT DL11
790 001422 000000          XFLGO: 0          ;FLAG FOR HARD XMIT ERRORS
791 001424 000000          RFLGO: 0          ;FLAG FOR HARD RCVR ERRORS
792 001426 000000          RFLG1: 0          ;FLAG FOR SOFT RCVR ERRORS
793 001430 000000          RTRY: 0          ;COUNTS NO. OF RETRIES ON SOFT ERRORS
794 001432 000000          OPTR: 0          ;CONTAINS POINTER TO OUTPUT BUFFER
795 001434 000000          IPTR: 0          ;CONTAINS POINTER TO INPUT BUFFER
796 001436 000000          LDOUT: 0          ;CONTAINS POINTER TO LOAD BUFFER ROUTINE
797 001440 000000          TIMR1: 0          ;TIMERS FOR 256. BYTE BLOCK TRANSFERS
798 001442 000000          TIMR2: 0
799 001444 000000          INTFLG: 0          ;SOFTWARE INTR. FLAG
800
801
802
803
804 001446 000240          BEGIN: NOP          ;PROGRAM WILL START HERE
805 .SBTTL INITIALIZE THE COMMON TAGS
806 ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
807 001450 012706 001100          MOV #CMTAG,R6          ;:FIRST LOCATION TO BE CLEARED
808 001454 005026          CLR (R6)+          ;:CLEAR MEMORY LOCATION
809 001456 022706 001140          CMP #SWR,R6 ;;DONE?
810 001462 001374          BNE -6          ;:LOOP BACK IF NO
811 001464 012706 001100          MOV #STACK,SP          ;:SETUP THE STACK POINTER
812 ;;INITIALIZE A FEW VECTORS
813 001470 012737 010462 000020          MOV #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
814 001476 012737 000340 000022          MOV #340,@IOTVEC+2 ;:LEVEL 7
815 001504 012737 010732 000030          MOV #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
816 001512 012737 000340 000032          MOV #340,@EMTVEC+2 ;:LEVEL 7
817 001520 012737 013052 000034          MOV #TRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
818 001526 012737 000340 000036          MOV #340,@TRAPVEC+2 ;:LEVEL 7
819 001534 012737 013132 000024          MOV #PWRDN,@PWRVEC ;:POWER FAILURE VECTOR
820 001542 012737 000340 000026          MOV #340,@PWRVEC+2 ;:LEVEL 7

```

```

821 001550 005067 177466 CLR $TIMES ;: INITIALIZE NUMBER OF ITERATIONS
822 001554 005067 177464 CLR $ESCAPE ;: CLEAR THE ESCAPE ON ERROR ADDRESS
823 001560 112767 000001 177327 MOVB #1,$ERMAX ;: ALLOW ONE ERROR PER TEST
824 001566 012767 001566 177312 MOV #,$LPADR ;: INITIALIZE THE LOOP ADDRESS FOR SCOPE
825 001574 012767 001574 177305 MOV #,$LPERR ;: SETUP THE ERROR LOOP ADDRESS
826 ;: SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
827 ;: EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
828 001602 013746 000004 MOV @#ERRVEC, -(SP) ;: SAVE ERROR VECTOR
829 001606 012737 001642 000004 MOV #64,$@#ERRVEC ;: SET UP ERROR VECTOR
830 001614 012767 177570 177316 MOV #DSWR, SWR ;: SETUP FOR A HARDWARE SWICH REGISTER
831 001622 012767 177570 177312 MOV #DDISP, DISPLAY ;: AND A HARDWARE DISPLAY REGISTER
832 001630 022777 177777 177302 CMP #-1, @SWR ;: TRY TO REFERENCE HARDWARE SWR
833 001636 001012 BNE 66$ ;: BRANCH IF NO TIMEOUT TRAP OCCURRED
834 ;: AND THE HARDWARE SWR IS NOT = -1
835 001640 000403 BR 65$ ;: BRANCH IF NO TIMEOUT
836 001642 012716 001650 64$: MOV #65$, (SP) ;: SET UP FOR TRAP RETURN
837 001646 000002 RTI
838 001650 012767 000176 177262 65$: MOV #SWREG, SWR ;: POINT TO SOFTWARE SWR
839 001656 012767 000174 177256 MOV #DISPREG, DISPLAY
840 001664 012637 000004 66$: MOV (SP)+, @#ERRVEC ;: RESTORE ERROR VECTOR
841
842 001670 005067 177376 CLR MULTD ;: CLEAR MULTIPLE DEVICE
843 ;: TESTING FLAG
844 001674 005067 177356 CLR TABFLG ;: CLEAR TABLE CREATION FLAG
845 001700 012767 000010 177326 MOV #8., $TMP15 ;: SET CHARACTER LENGTH DESIGNATOR
846 ;: FOR 8 BITS --- THIS IS THE DEFAULT
847 ;: LENGTH ASSUMED BY THE PROGRAM
848 ;: UNLESS THE USER CHANGES IT THRU
849 ;: THE QUESTION AND ANSWER CYCLE
850 ;: INITIATED BY SETTING SW<0> TO A 1
851 001706 012767 000200 177366 MOV #200, DLPRI ;: SET STANDARD PRIORITY LEVEL
852 ;: FOR DEVICE
853 001714 032767 000400 177216 BIT #SWB, SWR ;: IS THE 'LOOP ON TEST' SWITCH SET?
854 001722 001411 BEG 1$ ;: BRANCH IF NOT
855
856 ;: IF THE 'LOOP ON TEST' SWITCH WAS SET WE WILL TAKE THE NEXT BRANCH
857 ;: INSTRUCTION THUS BYPASSING TABLE CREATION
858
859 ;: IF THE USER DESIRED TO LOOP ON A TEST OF OTHER THAN THE DEFAULT DEVICE
860 ;: THEN HE SHOULD HAVE PREVIOUSLY FILLED THE FOLLOWING PROGRAM LOCATIONS
861 ;: WITH THE DESIRED DEVICE REGISTER VALUES:
862
863 *****
864 UNDER ;DL11 DEFINITIONS ABOVE
865 *****
866
867 DLRCR: PATCH THE ADDRESS OF THE RCVR CSR
868 DLRDBR: PATCH THE ADDRESS OF THE RCVR DBR
869 DLXCSR: PATCH THE ADDRESS OF THE XMIT CSR
870 DLXDBR: PATCH THE ADDRESS OF THE XMIT DBR
871 DLVECT: PATCH THE VECTOR ADDRESS OF THIS DL11
872
873 001724 104401 016635 TYPE, STMES ;: PRINT OUT 'MAINDEC' NAME
874 001730 104401 020735 TYPE, FAILSA ;: TYPE FAILSAFE MESSAGE
875 001734 104000 ERROR +0 ;: TYPE OUT THE PC VALUE
876 001736 104401 021313 TYPE, PCMSG ;: FOLLOWED BY =PC

```

```

877 001742 000000          HALT                ;WAIT FOR USER TO RESPOND
878 001744 000443          BR          ONCE          ;GO TO TEST DEVICE PATCHED IN BY USER
879 001746
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906 002026 105767 177224
907 002032 001010
908
909 002034 104401 016635
910
911 002040 105167 177212
912
913 002044 032767 000001 177066
914
915
916
917 002052 001012
918 002054 005077 177334
919 002060 005077 177324
920 002064 005777 177322
921 002070 005777 177316
922 002074 000167 000670
923
924
925
926
927
928
929
930
931 002100 104401 017217
932

```

IS:

```

:ENSURE THAT IF MULTIPLE DEVICE TESTING WAS BEING DONE
:AND THE USER 'HALTED' THE PROGRAM BEFORE ALL DEVICES
:WERE COMPLETED AND WENT BACK TO 'LOAD ADDRESS 200'
:TO RESTART THE PROGRAM THAT AS A BARE MINIMUM
:HE CAN RUN THE DEFAULT DEVICE (1ST RECEIVER
:STATUS REGISTER ADDRESS 175610)
:NOTE: IF THIS IS NOT SUITABLE THE USER WILL
:       HAVE TO SET SWO=1 (OR UP) IN ORDER TO
:       RECREATE THE TABLE HE DESTROYED FROM
:       ABOVE
890 001746 012767 175610 177304  MOV    #175610,DLBASE ;1ST POSSIBLE RECEIVER CSR
891 001754 004767 006074          JSR    PC,DLADDR     ;FORM DL ADDRESSES FOR
892                                ;1ST POSSIBLE DEVICE
893 001760 012767 000300 177432  MOV    #300,DLVECT  ;1ST POSSIBLE INTERRUPT VECTOR
894 001766 005067 177264          CLR    TABFLG       ;CLEAR TABLE CREATION FLAG
895
896 001772 012706 001100          RESTRT: MOV   #STACK,SP ;SET UP STACK POINTER
897 001776 012737 015020 000004  MOV   #BUSERR, @#ERRVEC ;SET UP BUS ERROR VECTOR
898 002004 012737 009340 000006  MOV   #340, @#ERRVEC+2
899 002012 012737 015044 000010  MOV   #RSVERR, @#RESVEC ;SET UP RSVD INSTR. VECTOR
900 002020 012737 000340 000002  MOV   #340, @#RESVEC+2
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932

```

:THIS NEXT SECTION WILL CHECK TO SEE IF MULTIPLE DEVICE TESTING
:WILL TAKE PLACE I.E. -
: A) HAS FREE RUNNING DEVICE TABLE ALREADY BEEN CREATED, AND/OR
: B) IF IT HAS, DOES USER WISH TO CHANGE IT, OR DO WE TEST DEFAULT DEVICE?
: TSTB TABFLG ;HAS TABLE CREATION BEEN PERFORMED?
: BNE ONCE ;BRANCH IF YES TO SKIP 'MAINDEC
: ;TITLE' MESSAGE
: TYPE ,STMES ;OTHERWISE, PRINT OUT 'MAINDEC'
: ;NAME
: COMB TABFLG ;IF TABLE CREATION HAS NOT BEEN
: ;PERFORMED, THEN SET FLAG, AND DO SO
: BIT #SWO,SWR ;THE PROGRAM HAS OBVIOUSLY BEEN
: ;RESTARTED - DOES USER WISH TO
: ;RESELECT VECTOR AND CONTROL REGISTER
: ;ADDRESSES I.E. - CREATE A NEW TABLE?
: ;BRANCH IF YES
: ONCE: CLR @DLXCSR ;CLEAR OUT BOTH CSR'S
: CLR @DLRCSR
: TST @DLRDBR ;FLUSH RCVR "DONE" BIT
: TST @DLRDBR
: JMP TST1 ;OTHERWISE, GO WITH EXISTING
: ;TABLE OR NOT USE ANY TABLE AT
: ;ALL WHICHEVER THE CASE MAY BE
: ;DEFAULT CASE IS 1ST POSSIBLE
: ;DEVICE)
: IF WE COME THIS PATH THE USER HAS DECIDED 1 OF 2 ALTERNATIVES:
: A) TO RUN MULTIPLE DEVICES
: B) TO CREATE A NEW TABLE TO RUN FROM, OR
: C) TO CHANGE THE CHARACTER LENGTH
: GO: TYPE, LENGTH ;ASK USER FOR THE CHARACTER LENGTH
: ;FOR WHICH HIS DEVICE IS SET


```

933 002104 104411          RUDEC          ;ACCEPT THE ANSWER TYPED BY USER
934          ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
935 002106 012600          MOV          (SP)+,RO      ;GET THE ANSWER TYPED
936 002110 020027 000010  CMP          RO,#8.        ;IS THE NUMBER TOO HIGH?
937 002114 101114          BHI          RETRY        ;IF YES - GO TO RETRY SITUATION
938 002116 020027 000005  CMP          RO,#5.        ;IS THE NUMBER TOO LOW?
939 002122 103511          BLO          RETRY        ;IF YES - GO TO RETRY SITUATION
940 002124 010067 177104  MOV          RO,$TMP15    ;THE VALUE TYPED IS OK
941          ;STORE FOR FUTURE USE
942 002130 104401 017301  TYPE,        DEFAULT    ;ASK USER IF HE WISHES TO TEST OTHER
943          ;THAN THE DEFAULT DEVICE
944 002134 104410          RDOCT          ;ACCEPT THE ANSWER TYPED BY USER
945 002136 005726          TST          (SP)+        ;LOOK AT THE ANSWER
946 002140 001002          BNE          1$          ;BRANCH IF REPLY WAS YES
947 002142 000137 002724  JMP          @#FLUSH      ;OTHERWISE, SKIP REST OF INTERROGATION
948 002146 012700 000300  1$:          MOV          #300,RO ;START RESTORATION OF TRAPCATCHER
949 002152 012701 000302  MOV          #302,R1      ;AREA FROM LOCATIONS 300 TO 776
950 002156 012702 000004  MOV          #4,R2        ;SO THAT WE CREATE THE MULTIPLE
951 002162 010110 2$:          MOV          R1,(RO)    ;DEVICES TABLE WITH A CLEAN SLATE
952 002164 005011          CLR          (R1)
953 002166 060200          ADD          R2,RO
954 002170 150201          ADD          R2,R1
955 002172 022701 001000  CMP          #1000,R1
956 002176 002771          BLT          2$
957          ;THE TRAPCATCHER VECTOR AREA FROM 300 - 776 SHOULD NOW BE RESTORED.
958          ;PROCEED TO FIND OUT THE 1ST DEVICE RECEIVER CONTROL REGISTER
959          ;ADDRESS
960 002200 104401 017406  FIRSTD: TYPE ,MFIRSTD    ;ASK USER FOR THE RECEIVER CONTROL
961          ;REGISTER ADDRESS OF HIS FIRST
962          ;DEVICE
963 002204 104410          RDOCT          ;ACCEPT THE ANSWER TYPED BY USER
964          ;AND STORE ON TOP OF STACK
965          ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
966 002206 012600          MOV          (SP)+,RO      ;GET THE ANSWER TYPED
967 002210 020027 176170  CMP          RO,#176170   ;IS THE NUMBER TOO HIGH?
968 002214 101060          BHI          RETRY        ;IF YES-GO TO RETRY SITUATION
969 002216 020027 175610  CMP          RO,#175610   ;IS THE NUMBER TOO LOW?
970 002222 103455          BLO          RETRY        ;IF YES - GO TO RETRY SITUATION
971 002224 132700 000001  BITB         #BIT0,RO    ;NUMBER IS IN RANGE BUT IS IT
972          ;ON AN EVEN BOUNDARY?
973 002230 001052          BNE          RETRY        ;IF NO - GO TO RETRY SITUATION
974          ;CHECK TO SEE IF USER RESPONSE WAS TRULY A RCVR STATUS REGISTER
975 002232 032700 000007  BIT          #7,RO        ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
976          ;USER RESPONSE EQUAL TO A ZERO?
977 002236 001047          BNE          RETRY        ;BRANCH IF NOT
978 002240 010067 177014  MOV          RO,DLBASE    ;THE 1ST ADDRESS VALUE TYPED IS OK
979          ;STORE FOR FUTURE USE
980          ;NOW WE ARE READY TO FIND OUT THE DEVICE INTERRUPT VECTOR
981 002244 016767 177010 177010  MOV          DLBASE,KEEPADD ;GET 1ST ADDRESS VALUE
982 002252 004767 005576          JSR          PC,DLADDR    ;GO FORM DL ADDRESSES FOR
983          ;1ST DEVICE SELECTED
984 002256 016767 177000 177000  MOV          KEEPADD,BASEADD ;RESTORE 1ST DEVICE ADDRESS
985 002264 104401 017474  VECT:       TYPE ,MVECT   ;ASK USER FOR A VECTOR ADDRESS
986 002270 104410          RDOCT          ;ACCEPT THE ANSWER TYPED BY USER
987          ;AND STORE ON TOP OF STACK
988          ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS

```

999	002272	012600			MOV	(SP)+,RO		;GET THE ANSWER TYPED
990	002274	020027	000776		CMP	RO,#776		;IS THE NUMBER TOO HIGH?
991	002300	101032			BHI	RETRY1		;IF YES - GO TO RETRY SITUATION
992	002302	020027	000300		CMP	RO,#300		;IS THE NUMBER TOO LOW?
993	002306	103427			BLO	RETRY1		;IF YES - GO TO RETRY SITUATION
994	002310	132700	000001		BITB	#BIT0,RO		;NUMBER IS IN RANGE BUT IS IT
995								;ON AN EVEN BOUNDARY?
996	002314	001024			BNE	RETRY1		;IF NO - GO TO RETRY SITUATION
997								;CHECK TO SEE IF THE USER RESPONSE WAS TRULY A RCVR VECTOR ADDRESS
998	002316	032700	000007		BIT	#7,RO		;WAS THE LEAST SIGNIFICANT DIGIT OF THE
999								;USER RESPONSE EQUAL TO A ZERO?
1000	002322	001021			BNE	RETRY1		;BRANCH IF NOT
1001	002324	010067	177070		MOV	RO,DLVECT		;THE VECTOR VALUE TYPED IS OK
1002								;STORE FOR FUTURE USE
1003	002330	016767	177064	176730	MOV	DLVECT,KEEP1V		;GET THE FIRST VECTOR VALUE
1004	002336	016767	177056	176724	MOV	DLVECT,BASE1V		;SAVE FIRST VECTOR VALUE
1005	002344	000414			BR	HOWMANY		;GO TO SEE IF USER WANTS MORE
1006								;THAN 1 DEVICE
1007	002346	104401	001252		RETRY:	TYPE, \$QUES		;TYPE '?' INDICATING USER TYPED
1008								;SOMETHING WRONG FOR CHARACTER LENGTH
1009	002352	000167	177522		JMP	GO		;GO BACK TO REISSUE QUESTION
1010	002356	104401	001252		RETRY0:	TYPE . \$QUES		;TYPE '?' INDICATING USER TYPE
1011								;SOMETHING WRONG FOR 1ST ADDRESS
1012	002362	000167	177612		JMP	FIRST0		;GO BACK TO REISSUE QUESTION
1013	002366	104401	001252		RETRY1:	TYPE . \$QUES		;TYPE '?' INDICATING USER TYPED
1014								;SOMETHING WRONG FOR VECTOR
1015	002372	000167	177666		JMP	VECT		;GO BACK TO REISSUE QUESTION
1016	002376	104401	017552		HOWMANY:	TYPE .MULDEV		;ASK USER IF HE WISHES TO RUN
1017								;MULTIPLE DEVICES
1018	002402	104410			RDOCT			;ACCEPT THE ANSWER TYPED BY USER
1019								;AND STORE ON TOP OF STACK
1020	002404	012600			MOV	(SP)+,RO		;GET THE ANSWER TYPED
1021	002406	005700			TST	RO		;WAS THE ANSWER YES?
1022	002410	001003			BNE	1\$;BRANCH IF IT WAS
1023	002412	005067	176654		CLR	MULTD		;OTHERWISE, INITIALIZE FLAG TO
1024								;INDICATE NON-MULTIPLE DEVICES
1025	002416	000402			BR	2\$;SKIP NEXT INSTRUCTION
1026	002420	105167	176646		1\$:	COMB MULTD		;INITIALIZE FLAG TO INDICATE
1027								;RUNNING OF MULTIPLE DEVICES
1028	002424				2\$:			
1029	002424	105767	176642		TSTB	MULTD		;ARE THERE MULTIPLE DEVICES ON
1030								;THE SYSTEM?
1031	002430	100406			BMI	LASTD ;IF SO,		GO TO ASK NEXT QUESTION
1032	002432	005067	176636		CLR	ACTREG		;CLEAR DEVICE ACTIVE FLAG TO
1033								;INDICATE NO RUNNING OF MULTIPLE
1034								;DEVICES
1035	002436	005067	176634		CLR	ROTADD		;CLEAR DEVICE ADDRESS POINTER IN
1036								;USE WHEN RUNNING MULTIPLE DEVICES
1037	002442	000167	000160		JMP	CONQUES		;SKIP ASKING NEXT QUESTION
1038	002446							
1039					LASTD:			
1040								;WE WILL NOW BEGIN TO SET UP THE DEVICE ACTIVE REGISTER FOR RUNNING
1041	002446	104401	017637					;MULTIPLE DL11 DEVICES
1042					TYPE	,MLASTD		;ASK USER FOR THE RECEIVER
1043								;CONTROL REGISTER ADDRESS OF
1044	002452	104410			RDOCT			;HIS LAST DEVICE
								;ACCEPT THE ANSWER TYPED BY

```

1045                                     ;USER AND STORE ON TOP OF STACK
1046                                     ;CHECK TO SEE IF THE USER RESPONSE WAS WITHIN LIMITS
1047 002454 012600 1$: MOV (SP)+,RO ;GET THE ANSWER TYPED
1048 002456 020027 176170 CMP RO,#176170 ;IS THE NUMBER TOO HIGH?
1049 002462 101132 BHI RETRY2 ;IF YES - GO TO RETRY SITUATION
1050 002464 020027 175610 CMP RO,#175610 ;IS THE NUMBER TOO LOW?
1051 002470 103527 SLO RETRY2 ;IF YES - GO TO RETRY SITUATION
1052 002472 132700 000001 BITB #BIT0,RO ;NUMBER IS IN RANGE BUT IS IT
1053                                     ;ON AN EVEN BOUNDARY?
1054 002476 001124 BNE RETRY2 ;IF NOT - GO TO RETRY SITUATION
1055                                     ;CHECK TO SEE IF USER RESPONSE WAS TRULY A RCVR STATUS REGISTER
1056 002500 032700 000007 BIT #7,RO ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
1057                                     ;USER RESPONSE EQUAL TO A ZERO?
1058 002504 001121 BNE RETRY2 ;BRANCH IF NOT
1059 002506 010067 176566 MOV RO, LASTADD ;THE LAST ADDRESS VALUE TYPED IS OK
1060                                     ;STORE FOR FUTURE USE
1061                                     ;NOW WE BEGIN TO ACTUALLY INITIALIZE THE DEVICE ACTIVE REGISTER
1062                                     ;FROM WHICH THE PROGRAM WILL CYCLE UNTIL ALL DEVICES HAVE BEEN TESTED
1063 002512 012767 000001 176556 MOV #1, ROTADD ;SET UP POINTER FOR 'ACTREG'
1064 002520 005067 176550 CLR ACTREG ;CLEAR DEVICE ACTIVE REGISTER
1065 002524 056767 176546 176542 2$: BIS ROTADD, ACTREG ;MAKE 1ST DEVICE ACTIVE
1066 002532 000241 CLC ;CLEAR CARRY BIT FOR POINTER
1067                                     ;ROTATION
1068 002534 006167 176536 ROL ROTADD ;ARE WE PAST 16 LINE RANGE?
1069 002540 103422 BCS 3$ ;BRANCH IF YES
1070 002542 062767 000010 176514 ADD #10, BASEADD ;STEP UP BASE ADDRESS
1071 002550 026767 176524 176506 CMP LASTADD, BASEADD ;IS THIS THE LAST DEVICE?
1072 002556 101362 BHI 2$ ;BRANCH IF NOT
1073                                     ;NOTE: IF THIS PATH IS TAKEN IT IS ASSUMED THAT AT LEAST 2 DEVICES
1074                                     ;EXIST AND THAT ALL ADDRESSING IS CONTIGUOUS
1075 002560 056767 176512 176506 BIS ROTADD, ACTREG ;INDICATE NEXT DEVICE ACTIVE
1076 002566 012767 000001 176502 MOV #1, ROTADD ;RESET POINTER FOR 'ACTREG' FOR
1077                                     ;LATER USE IN END OF PASS ROUTINE
1078 002574 016767 176462 176462 MOV KEEPADD, BASEADD ;RESET 1ST DEVICE RECEIVER
1079                                     ;CONTROLLER REGISTER ADDRESS FOR
1080                                     ;LATER USE IN END OF PASS ROUTINE
1081 002602 000167 000020 JMP CONQUES ;GO TO CONTINUE QUESTIONING OF USER
1082 002605 3$:
1083                                     ;IF WE TAKE THIS PATH IT APPEARS THAT THERE ARE NOT AT LEAST
1084                                     ;TWO DEVICES PRESENT - IN RESPONSE TO USER TYPING 'YES' TO MULTIPLE
1085                                     ;DEVICES QUESTION
1086 002606 016767 176450 176450 MOV KEEPADD, BASEADD ;RESET 1ST DEVICE RECEIVER
1087                                     ;CONTROLLER REGISTER ADDRESS
1088 002614 104401 017735 TYPE ,MRANGE ;INFORM USER TO CHECK AND RETYPE
1089                                     ;THE LAST DEVICE RCSR ADDRESS
1090 002620 104410 RDOCT ;ACCEPT THE ANSWER TYPED BY USER
1091                                     ;AND STORE ON TOP OF STACK
1092 002622 000167 177526 JMP 1$
1093 002626 CONQUES:
1094                                     ;IF WE HAVE REACHED THIS PORTION WE KNOW:
1095                                     ;A) THE 'RXCSR' ADDRESS OF THE 1ST DEVICE
1096                                     ;B) THE 'RXCSR' ADDRESS OF THE LAST DEVICE, SAND
1097                                     ;C) THE INTERRUPT VECTOR OF THE 1ST DEVICE
1098                                     ;NOW LET'S FIND THE PRIORITY LEVEL
1099 002626 104401 020021 TYPE ,PLEVEL ;ASK USER FOR PRIORITY LEVEL
1100 002632 104410 RDOCT ;ACCEPT ANSWER TYPED BY USER AND

```


D03

MACY 11-0220-4
YES +4+ "READ" BIT IS ONLY BIT SET IN XCSR

MACY 11-0220-4 20-SEP-76 09:19 PAGE 29
YES +4+ "READ" BIT IS ONLY BIT SET IN XCSR

1250 003354 020412
1251 003356 021403
1252 003360 024767
1253 003364 104002
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500

```

                                CMP      R4,R2)          ;[XCSR]=000200 ??
                                BEQ      TST7          ;<BR IF YES>
                                JSR      PC,SUER2       ;GO SETUP ERROR INFO
                                ERROR+2             ;[XCSR] INCORRECT ON START JP
*****
;*TEST 7 TEST THAT "MAINT" BIT CAN BE SET AND CLEARED
*****
TST7:  SCOPE
                                MOV      #204,R4        ;RESULT IN XCSR S/B = 000204
                                MOV      CLXCSR,R2     ;REGADR = XCSR ADR
                                BIS      #BIT2,(R2)    ;SET THE "MAINT" BIT
                                CMP      R4,(R2) ;RESULT IN XCSR OK ??
                                BEQ      IS           ;<BR IF YES>
                                JSR      PC,SUER2       ;GO SET UP ERROR INFO
                                ERROR+2             ;MAINT. BIT FAILED TO SET PROPERLY
IS:   MOV      #200,R4        ;RESULT IN XCSR S/B = 000200
                                BIC      #BIT2,(R2)    ;NOW CLEAR THE "MAINT" BIT
                                CMP      R4,(R2) ;RESULT IN XCSR OK ??
                                BEQ      TST10        ;<BR IF YES>
                                JSR      PC,SUER2       ;GO SET UP ERROR INFO
                                ERROR+2             ;MAINT BIT FAILED TO CLEAR PROPERLY
*****
;*TEST 10 TEST THAT XMIT I.E. CAN CAUSE AN INTR
*****
TST10: SCOPE
                                CLR      INTFLG        ;INIT SOFTWARE INTR FLAG
                                MOV      OLVECT,R5     ;GET VECTOR ADDRESS
                                MOV      #254,R5      ;GO TO 45 ON INTR
                                MOV      CLPRI,6(R5)   ;PRIORITY LEVEL 4
                                CLR      R5           ;INIT INTR. TIMER
                                MOV      #200,R4        ;RESULT IN XCSR S/B = 000200
                                MOV      CLXCSR,R2     ;REGADR = XCSR ADR
                                BIS      #100,(R2)    ;SET INTR. ENABLE BIT C6
                                TST      INTFLG        ;DID INTR OCCUR YET ??
                                BNE      IS           ;BR IF IT DID
                                DEC      R5           ;COUNT THE TIMER
                                BNE      IS           ;BR IF NO TIMEOUT
                                MOV      #300,R4        ;RESULT IN XCSR S/B = 000300
                                JSR      PC,SUER2       ;GO SETUP ERROR INFO
                                MOV      #45,$ESCAPE   ;RETURN TO 45 AFTER ERROR PRINT
                                ERROR+2             ;INTR. FAILED
45:   BR      TST11        ;<GO TO NEXT TEST>
25:   COM      INTFLG        ;SET THE SOFTWARE FLAG
                                BIC      #100,(R2)    ;TURN OFF I.E. BIT
                                RTI                    ;RETURN CONTROL TO INTR. ROUTINE
35:   CMP      R4,(R2) ;RESULT IN XCSR OK ??
                                BEQ      TST11        ;<BR IF YES>
                                JSR      PC,SUER2       ;GO SET UP ERROR INFO
                                ERROR+2             ;XMIT INTR. NOT SERVICED PROPERLY
*****
;*TEST 11 TEST THAT RCVR I.E. BIT CAN BE SET AND CLEARED
*****
TST11: SCOPE
                                MOV      #100,R4        ;RESULT IN RCSR S/B = 000100
                                MOV      CLRCSR,R2     ;REGADR = RCSR ADR

```

E03

MAINDEC-11-DZDLC-A
DZDLC.A.P11 T11

MACY11 27(732) 20-SEP-76 09:19 PAGE 30
TEST THAT RCVR I.E. BIT CAN BE SET AND CLEARED

```

1269 003476 052712 000100      BIS      #BIT6,(R2)      ;SET I.E. BIT
1270 003502 020412      CMP      R4,(R2)      ;DID IT SET PROPERLY ??
1271 003504 001403      BEQ     1$           ;<BR IF YES>
1272 003506 004767 010324      JSR     PC,SUER2     ;GO SET UP ERROR INFO.
1273 003512 104002      ERROR+2            ;RCVR I.E. BIT FAILED TO SET PROPERLY
1274 003514 005004      1$:      CLR      R4          ;RESULT IN RCSR S/B = 0000C0
1275 003516 042712 000100      BIC     #BIT6,(R2)   ;CLEAR THE I.E. BIT
1276 003522 020412      CMP      R4,(R2)      ;DID IT CLEAR PROPERLY ??
1277 003524 001403      BEQ     TST12        ;<BR IF YES>
1278 003526 004767 010304      JSR     PC,SUER2     ;GO SET UP ERROR INFO
1279 003532 104002      ERROR+2            ;RCVR I.E. BIT FAILED TO CLEAR PROPERLY
1280
1281
1282
1283
1284 003534 000004      *TEST 12          TEST THAT RCVR "DONE" CAN GENERATE AN INTR.
1285 003536 016705 175656      *****
1286 003542 012725 003702      *ST12:          SCOPE
1287 003546 016715 175530      MOV     DLVECT,R5    ;GET THE VECTOR ADDRESS
1288 003552 005067 175666      MOV     #3$(R5)+    ;GO TO 3$ ON RCVR INTR.
1289 003556 005005      MOV     DLPRI,(R5)  ;AT LEVEL 4
1290 003560 105067 175420      CLR     INTFLG      ;INIT THE SOFTWARE FLAG
1291 003564 016702 175620      CLR     R5          ;INIT INTR. TIMER
1292 003570 005012      CLR     $TMP1       ;INIT WHERE DATA WILL BE STORED
1293 003572 052712 000100      MOV     DLRCSR,R2   ;REGADR = RCSR ADR
1294 003576 052762 000004 000004      CLR     (R2)        ;INIT THE PCSR TO 0000C0
1295 003604 112762 000252 000006      BIS     #BIT6,(R2)  ;ENABLE RCVR INTERRUPTS
1296 003612 005767 175626      BIS     #BIT2,4(R2) ;NOW TURN ON MAINT MODE
1297 003616 001044      MOV     #252,6(R2)  ;LOAD XMIT BUFFER REG.
1298 003620 005305      1$:      TST     INTFLG      ;DID RCVR INTR. YET ??
1299 003622 001373      BNE     4$          ;BR IF IT DID
1300 003624 013767 177776 175350      DEC     R5          ;COUNT THE TIMER
1301 003632 042762 000004 000004      BNE     1$          ;BR IF NO TIMEOUT
1302 003640 042712 000100      MOV     @PSW,$TMP1  ;SAVE ERROR PSW
1303 003644 010667 175326      BIC     #BIT2,4(R2) ;DISABLE MAINT MODE
1304 003650 010201      BIC     #100,(R2)   ;DISABLE RCVR INTR.
1305 003652 011203      MOV     SP,$REG6    ;SAVE THE ERROR SP
1306 003654 012704 000200      MOV     R2,R1       ;DEVADR = RCSR ADR
1307 003664 012767 003674 175352      MOV     (R2),R3     ;GET THE WAS DATA
1308 003672 104002      MOV     #200,R4     ;[RCSR] S/B = 000200
1309 003674 005762 000002      JSP     PC,SUERR1   ;GO SET UP ERROR INFO.
1310
1311
1312 003700 000440      2$:      MOV     #2$, $ESCAPE ;RETURN TO 2$ AFTER ERROR ALWAYS
1313 003702 042762 000004 000004      ERROR+2          ;RCVR INTERRUPT FAILED
1314 003710 116267 000002 175266      TST     2(R2)       ;REFERENCE RCVR DATA BUFFER
1315 003716 042712 000100      ;TO CLEAR RCSR IN CASE RCVR
1316 003722 005167 175516      ;INTERRUPTS COULD NOT BE ENABLED
1317 003726 000002      ;<GO TO NEXT TEST>
1318 003730 005004      3$:      BR      TST13        ;DISABLE THE MAINT MODE
1319 003732 005712      BIC     #BIT2,4(R2) ;GET THE RECEIVED DATA
1320 003734 001403      MOV     2(R2),$TMP1 ;TURN OFF RCVR INTR. ENAB
1321 003736 004767 010074      BIC     #BIT6,(R2)  ;SET THE SOFTWARE FLAG
1322 003742 104002      COM     INTFLG      ;RETURN TO MAINLINE
1323 003744 016701 175442      RTI                    ;[RCSR] S/B=0
1324 003750 016702 175442      4$:      CLR     R4          ;IS IT ALL ZEROES ??
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721
2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736
2737
2738
2739
2740
2741
2742
2743
2744
2745
2746
2747
2748
2749
2750
2751
2752
2753
2754
2755
2756
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768
2769
2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824
2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933
2934
2935
2936
2937
2938
2939
2940
2941
2942
2943
2944
2945
2946
2947
2948
2949
2950
2951
2952
2953
2954
2955
2956
2957
2958
2959
2960
2961
2962
2963
2964
2965
2966
2967
2968
2969
2970
2971
2972
2973
2974
2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985
2986
2987
2988
2989
2990
2991
2992
2993
2994
2995
2996
2997
2998
2999
3000
3001
3002
3003
3004
3005
3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023
3024
3025
3026
3027
3028
3029
3030
3031
3032
3033
3034
3035
3036
3037
3038
3039
3040
3041
3042
3043
3044
3045
3046
3047
3048
3049
3050
3051
3052
3053
3054
3055
3056
3057
3058
3059
3060
3061
3062
3063
3064
3065
3066
3067
3068
3069
3070
3071
3072
3073
3074
3075
3076
3077
3078
3079
3080
3081
3082
3083
3084
3085
3086
3087
3088
3089
3090
3091
3092
3093
3094
3095
3096
3097
3098
3099
3100
3101
3102
3103
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119
3120
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138
3139
3140
3141
3142
3143
3144
3145
3146
3147
3148
3149
3150
3151
3152
3153
3154
3155
3156
3157
3158
3159
3160
3161
3162
3163
3164
3165
3166
3167
3168
3169
3170
3171
3172
3173
3174
3175
3176
3177
3178
3179
3180
3181
3182
3183
3184
3185
3186
3187
3188
3189
3190
3191
3192
3193
3194
3195
3196
3197
3198
3199
3200
3201
3202
3203
3204
3205
3206
3207
3208
3209
3210
3211
3212
3213
3214
3215
3216
3217
3218
3219
3220
3221
3222
3223
3224
3225
3226
3227
3228
3229
3230
3231
3232
3233
3234
3235
3236
3237
3238
3239
3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253
3254
3255
3256
3257
3258
3259
3260
3261
3262
3263
3264
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288
3289
3290
3291
3292
3293
3294
3295
3296
3297
3298
3299
3300
3301
3302
3303
3304
3305
3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
33
```

F03

MAINDEC-11-DZDLC-A
DZDLCR.P11 T12

MACY11 27(732) 20-SEP-76 09:19 PAGE 31
TEST THAT RCVR "DONE" CAN GENERATE AN INTR.

```

1325 003754 016703 175224      MOV      $TMP1,R3      ;GET THE WAS DATA
1325 003760 042703 177400      BIC      #177400,R3    ;REMOVE JUNK FROM HI BYTE
1327 003764 012704 000252      MOV      #252,R4      ;DATA S/B = 252
1329 003770 020403          CMP      R4,R3        ;WAS = S/B ??
1329 003772 001403          BEQ     TST13        ;<BR IF YES>
1330 003774 004767 010064      JSR     PC,SUERR1    ;GO SET UP THE ERROR INFO
1331 004000 104003          ERROR+3             ;DATA COMPARE ERROR
1332
1333      ;*****
1333      ;*TEST 13      TEST THAT "REQ TO SEND" ASSERTS "RING"
1334      ;*****
1335 004002 000004      TST13: SCOPE
1336 004004 032767 010000 175126      BIT     #SW12,SWR     ;ARE WE TESTING /C OR /D MODEL?
1337 004012 001047          BNE     TST14        ;<BRANCH IF YES>
1338 004014 012704 140004      MOV     #140004,R4    ;RESULT IN RCSR S/B = 140004
1339 004020 016702 175364      MOV     DLRCR,R2     ;REGADR = RCSR ADR
1340 004024 005012          CLR     (R2)         ;INIT THE RCSR TO 000000
1341 004026 052712 000004      BIS     #BIT2,(R2)   ;SET "REQ TO SEND"
1342 004032 032777 100000 175350      BIT     #BIT15,DLRCR ;DID "RING" SET "DATA SET INT" ?
1343 004040 001003          BNE     1$          ;<BR IF YES>
1344 004042 004767 007770      JSR     PC,SLER2    ;GO SET UP ERROR INFO.
1345 004046 104002          ERROR+2             ;"RING" TRANSITION FAILED TO SET "DATA SET INT"
1346
1347 004050 012704 040004 1$:      MOV     #40004,R4    ;NOTE: "BIT #BIT15,(R2)" RESETS BIT15
1348 004054 020412          CMP     R4,(R2)     ;RESULT IN RCSR S/B = 40004
1349 004056 001403          BEQ     2$          ;BOTH "RING" AND "REQ TO SEND" ASSERTED ?
1350 004060 004767 007752      JSR     PC,SUER2    ;GO SET UP ERROR INFO.
1351 004064 104002          ERROR+2             ;"RING" OR "REQ TO SEND" FAILED TO SET
1352 004066 005004 2$:      CLR     R4          ;RESULT IN RCSR S/B = 000000
1353 004070 042712 000004      BIC     #BIT2,(R2)   ;CLEAR "REQ TO SEND"
1354 004074 032777 100000 175306      BIT     #BIT15,DLRCR ;DID "DATA SET INT" GET SET ??
1355 004102 001403          BEQ     3$          ;<BR IF NOT>
1356 004104 004767 007726      JSR     PC,SUER2    ;GO SET UP ERROR INFO
1357 004110 104002          ERROR+2             ;CLEARING "RING" SET "DATA SET INT"
1358 004112 020412 3$:      CMP     R4,(R2)     ;RCSR CONTAIN ALL ZEROES ??
1359 004114 001406          BEQ     TST14        ;<BR IF YES>
1360 004116 004767 007714      JSR     PC,SUER2    ;GO SET UP ERROR INFO.
1361 004122 016767 000002 175050      MOV     .+6,$REG7    ;SAVE THE ERROR PC
1362 004130 104002          ERROR+2             ;CLEARING "REQ TO SEND" FAILED TO CLEAR "RING"
1363
1364      ;*****
1364      ;*TEST 14      TEST THAT "SEC XMIT" ASSERTS "SEC REC" AND "DATA SET INT"
1365      ;*****
1366 004132 000004      TST14: SCOPE
1367 004134 032767 010000 174776      BIT     #SW12,SWR     ;ARE WE TESTING /C OR /D MODEL?
1368 004142 001046          BNE     TST15        ;<BRANCH IF YES>
1369 004144 016702 175240      MOV     DLRCR,R2     ;REGADR = RCSR ADR
1370 004150 005012          CLR     (R2)         ;INIT RCSR TO 000000
1371 004152 012704 102010      MOV     #102010,R4   ;CONTENTS OF RCSR S/B = 102010
1372 004156 052712 000010      BIS     #BIT3,(R2)   ;SET "SEC XMIT" BIT
1373 004162 032777 100000 175220      BIT     #BIT15,DLRCR ;DID "DATA SET INT" SET ??
1374 004170 001003          BNE     1$          ;<BR IF YES>
1375 004172 004767 007640      JSR     PC,SUER2    ;GO SET UP ERROR INFO
1376 004176 104002          ERROR+2             ;"DATA SET INT" FAILED TO SET-NOTE THAT
1377
1378 004200 012704 002010 1$:      MOV     #2010,R4    ;"BIT #BIT15,(R2)" RESETS BIT15
1379 004204 020412          CMP     R4,(R2)     ;RESULT IN RCSR S/B = 2010
1380 004206 001403          BEQ     2$          ;ARE "SEC XMIT" AND "SEC REC" BOTH SET ?
1380

```



```

1391 004210 004767 007622 JSR PC,SUER2 ;GO SET UP ERROR INFO
1392 004214 104002 ERROR+2 ;"SEC XMIT" OR "SEC REC" FAILED TO SET
1393 ;OR "DATA SET INT" FAILED TO BE CLEARED
1394 ;WHEN REFERENCING RCSR
1395 004216 012704 100000 2$: MOV #BIT15,R4 ;RESULT IN RCSR S/B = 100000
1396 004222 042712 000010 BIC #BIT3,(R2) ;CLEAR "SEC XMIT" BIT
1397 004226 032777 100000 175154 BIT #BIT15,DLCRCSR ;DID CLEARING IT SET "DATA SET INT"?
1398 004234 001003 BNE 3$ ;<BR IF YES>
1399 004236 004767 007574 JSR PC,SUER2 ;GO SET UP ERROR INFO
1400 004242 104002 ERROR+2 ;CLEARING "SEC XMIT" FAILED TO SET "DATA
1401 ;SET INT. (NOTE THAT REFERENCING RCSR
1402 ;CLEAR "DATA SET INT"
1403 004244 005004 3$: CLR R4 ;RESULT IN RCSR S/B = 000000
1404 004246 020412 CMP R4,(R2) ;"SEC XMIT" AND "SEC REC" CLEAR ?
1405 004250 001403 BEQ TST15 ;<BR IF YES>
1406 004252 004767 007560 JSR PC,SUER2 ;GO SETUP ERROR INFO
1407 004256 104002 ERROR+2 ;"SEC XMIT" OR "SEC REC" FAILED TO CLEAR
1408 ;OR REFERENCING RCSR FAILED TO CLEAR "DATA SET INT"
1409 ;*****
1410 ;*TEST 15 TEST THAT "DTR" CAN ASSERT "CLR TO SEND" AND "CAR DET"
1411 ;*****
1412 TST15: SCOPE
1413 BIT #SW12,SWR ;ARE WE TESTING /C OR /D MODEL?
1414 BNE TST16 ;<BRANCH IF YES>
1415 MOV DLRCR,R2 ;REGADR = RCSR ADR
1416 CLR (R2) ;INIT RCSR TO 000000
1417 MOV #130002,R4 ;RESULT IN RCSR S/B = 130002
1418 BIS #BIT1,(R2) ;SET "DTR" BIT
1419 BIT #BIT15,DLCRCSR ;DID "DATA SET INT" SET ??
1420 BNE 1$ ;<BR IF YES>
1421 JSR PC,SUER2 ;GO SET UP ERROR INFO
1422 ERROR+2 ;"DATA SET INT" FAILED TO SET -
1423 ;NOTE: THE REFERENCE TO RCSR ABOVE WILL
1424 ;WILL UNCONDITIONALLY CLEAR RCSR BIT 15.
1425 004326 012704 030002 1$: MOV #30002,R4 ;RESULT IN RCSR S/B = 30002
1426 004332 020412 CMP R4,(R2) ;"DTR" "CLR TO SEND", AND "CAR DET" ALLSET
1427 004334 001403 BEQ 2$ ;<BR IF ALL SET>
1428 004336 004767 007474 JSR PC,SUER2 ;GO SET UP ERROR INFO
1429 004342 104002 ERROR+2 ;"DTR" "CLR TO SEND" OR "CAR DET" FAILED
1430 ;TO SET OR "DATA SET INT" FAILED TO CLEAR
1431 004344 012704 100000 2$: MOV #BIT15,R4 ;RESULT IN RCSR S/B = 100000
1432 004350 042712 000002 BIC #BIT1,(R2) ;NOW CLEAR "DTR"
1433 004354 032777 100000 175026 BIT #BIT15,DLCRCSR ;DID "DATA SET INT" SET ??
1434 004362 001003 BNE 3$ ;<BR IF YES>
1435 004364 004767 007446 JSR PC,SUER2 ;GO SETUP ERROR INFO
1436 004370 104002 ERROR+2 ;"DATA SET INT" FAILED TO SET WHEN "DTR"
1437 ;WENT TO A ZERO.
1438 004372 005004 3$: CLR R4 ;RESULT IN RCSR S/B = 000000
1439 004374 020412 CMP R4,(R2) ;DID ALL BITS CLEAR??
1440 004376 001403 BEQ TST16 ;<BR IF YES>
1441 004400 004767 007432 JSR PC,SUER2 ;GO SET UP ERROR INFO
1442 004404 104002 ERROR+2 ;"DTR" "CLR TO SEND" OR "CAR DET" FAILED
1443 ;TO CLEAR PROPERLY
1444 ;*****
1445 ;*TEST 16 TEST THAT "DATA SET INT ENAB" CAN SET AND CLEAR
1446 ;*****

```

H03

MAINDEC-11-DZDLC-A
DZDLC.A.P11 T16

MACY11 27(732) 20-SEP-76 09:19 PAGE 33
TEST THAT "DATA SET INT ENAB" CAN SET AND CLEAR

```
1437 004406 000004 TST16: SCOPE
1438 004410 032767 010000 174522 BIT #SW12,SWR ;ARE WE TESTING /C OR /D MODEL?
1439 004416 001023 BNE TST17 ;<BRANCH IF YES>
1440 004420 016702 174764 MOV DLRCR,R2 ;REGADR = RCSR ADR
1441 004424 012704 000040 MOV #40,R4 ;RESULT IN RCSR S/B = 000040
1442 004430 052712 000040 BIS #BIT5,(R2) ;SET THE "DATA SET I.E." BIT
1443 004434 020412 CMP R4,(R2) ;DID IT SET OK ??
1444 004436 001403 BEQ 1$ ;<BR IF YES>
1445 004440 004767 007372 JSR PC,SUER2 ;GO SET UP ERROR INFO
1446 004444 104002 ERROR+2 ;"DATA SET I. E." FAILED TO SET
1447 004446 005004 1$: CLR R4 ;MAKE S/B DATA = 000000
1448 004450 042712 000040 BIC #BIT5,(R2) ;NOW CLEAR THE "DATA SET I.E." BIT
1449 004454 020412 CMP R4,(R2) ;DID IT CLEAR OK ??
1450 004456 001403 BEQ 1CT17 ;<BR IF YES>
1451 004460 004767 007352 JSR PC,SUER2 ;GO SET UP ERROR INFO.
1452 004464 104002 ERROR+2 ;"DATA SET I.E." FAILED TO CLEAR
1453
1454 ;*****
1455 ;*TEST 17 TEST THE "DATA SET I.E." CAN CAUSE A RCVR INTR
1456 ;*****
1456 004466 000004 TST17: SCOPE
1457 004470 032767 010000 174442 BIT #SW12,SWR ;ARE WE TESTING A /C OR /D MODEL?
1458 004476 001054 BNE TST20 ;<BRANCH IF YES>
1459 004500 016705 174714 MOV DLVECT,R5 ;GET THE VECTOR ADDR
1460 004504 012725 004572 MOV #3$, (R5)+ ;GO TO 3$ ON RCVR INTR.
1461 004510 016715 174566 MOV DLPRI,(R5) ;AT LEVEL 4
1462 004514 005005 CLR R5 ;INIT INTR. TIMER
1463 004516 005067 174722 CLR INTFLG ;INIT SOFTWARE FLAG
1464 004522 005004 CLR R4 ;RESULT IN RCSR S/B = 0 AFTER INTR.
1465 004524 016702 174660 MOV DLRCR,R2 ;REGADR = RCSR ADR
1466 004530 052712 000040 BIS #BIT5,(R2) ;SET THE "DATA SET I.E." BIT
1467 004534 052712 000002 BIS #BIT1,(R2) ;NOW SET "DTR" TO GEN INTR.
1468 004540 005767 174700 1$: TST INTFLG ;DID INTR OCCUR YET ??
1469 004544 001016 BNE 4$ ;BR IF YES
1470 004546 005305 DEC R5 ;COUNT THE TIMER
1471 004550 0013 BNE 1$ ;BR IF NO TIMEOUT
1472 004552 0047 007260 JSR PC,SUER2 ;GO SET UP ERROR INFO
1473 004556 005012 CLR (R2) ;TURN IT ALL OFF
1474 004560 012767 004570 174456 MOV #2$, $ESCAPE ;COME BACK TO 2$ IN ALL CASES
1475 004566 104002 ERROR+2 ;"DATA SET" INTR FAILED TO OCCUR
1476 004570 2$: BR TST20 ;<GO TO NEXT TEST>
1477 004570 000417 BR TST20 ;<GO TO NEXT TEST>
1478 004572 005012 3$: CLR (R2) ;ZERO THE RCSR
1479 004574 005167 174644 COM INTFLG ;SET THE SOFTWARE FLAG
1480 004600 000002 RTI ;RETURN TO SENDER
1481 004602 032712 100000 4$: BIT #BIT15,(R2) ;DID "DATA SET INT" GET SET BY INTR. SERVICE ??
1482 004606 001003 BNE 5$ ;<BR IF YES>
1483 004610 004767 007222 JSR PC,SUER2 ;GO SET UP ERROR INFO
1484 004614 104002 ERROR+2 ;DATA SET INTR. NOT SERVICED PROPERLY
1485 004616 020412 5$: CMP R4,(R2) ;ALL BITS IN RCSR CLEAR ??
1486 004620 001403 BEQ TST20 ;<BR IF YES>
1487 004622 004767 007210 JSR PC,SUER2 ;GO SET UP ERROR INFO
1488 004626 104002 ERROR+2 ;INTR. SERVICE FAILED TO CLEAR RCSR
1489
1490 ;*****
1491 ;*TEST 20 TEST THAT THE "BREAK" BIT CAN BE SET AND CLEARED
1492 ;*****
1492 004630 000004 TST20: SCOPE
```

MAINDEC-11-DZOLC-A
DZOLCA.P11 T20

MACY11 27(732) 20-SEP-76 09:19 PAGE 34
TEST THAT THE "BREAK" BIT CAN BE SET AND CLEARED

```

1493 004632 032767 010000 174300      BIT      #SW12.SWR      ;ARE WE TESTING /C OR /D MODEL?
1494 004640 001024      BNE      TST21      ;:<BRANCH IF YES>
1495 004642 012704 000201      MOV      #201,R4    ;RESULT S/B = 201 IN XCSR
1496 004646 016702 174542      MOV      DLXCSR,R2  ;SET UP REGADR
1497 004652 052712 000001      BIS      #BIT0,(R2) ;SET THE "BREAK" BIT
1498 004656 020412      CMP      R4,(R2)    ;DID IT SET PROPERLY ??
1499 004660 001403      BEQ      1$        ;:<BR IF YES>
1500 004662 004767 007150      JSR      PC,SUER2   ;GO SET UP ERROR INFO.
1501 004666 104002      ERROR+2 ;"BREAK" BIT FAILED TO SET PROPERLY
1502 004670 012704 000200      1$: MOV      #200,R4    ;RESULT S/B = 200 IN XCSR
1503 004674 042712 000001      BIC      #BIT0,(R2) ;CLEAR THE "BREAK" BIT
1504 004700 020412      CMP      R4,(R2)    ;DID IT CLEAR PROPERLY ??
1505 004702 001403      BEQ      TST21     ;:<BR IF YES>
1506 004704 004767 007126      JSR      PC,SUER2   ;GO SET UP ERROR INFO
1507 004710 104002      ERROR+2 ;"BREAK" FAILED TO CLEAR PROPERLY
1508
1509
1510
1511
1512
1513 004712 000004      *TEST 21 TEST THAT A "RESET" CLEARS THE "BREAK" BIT
1514 004714 012704 000200      *TEST 21 TEST THAT A "RESET" CLEARS THE "BREAK" BIT
1515 004720 016702 174470      *TEST 21 TEST THAT A "RESET" CLEARS THE "BREAK" BIT
1516 004724 052712 000001      *TEST 21 TEST THAT A "RESET" CLEARS THE "BREAK" BIT
1517 004730 000005      TST21: SCOPE
1518 004732 020412      MOV      #200,R4    ;RESULT S/B = 200
1519 004734 001403      MOV      DLXCSR,R2 ;SET UP REGADR
1520 004736 004767 007074      BIS      #BIT0,(R2) ;SET THE "BREAK" BIT
1521 004742 104002      RESET     ;CLEAR IT WITH A "RESET"
1522      CMP      R4,(R2)  ;DID IT CLEAR ??
1523      BEQ      TST22     ;:<BR IF YES>
1524      JSR      PC,SUER2 ;GO SET UP ERROR INFO.
1525      ERROR+2 ;RESET INSTR. FAILED TO CLEAR "BREAK"

```

```

1523
1524
1525
1526 004744 000004
1527 004746 012767 000071 174266
1528 004754 004767 000012
1529 004760 005067 174444
1530 004764 012767 014346 174444 1$:
1531 004772 004767 007222
1532 004776 005767 174420 2$:
1533 005002 001040
1534 005004 005767 174414
1535 005010 001053
1536 005012 005767 174410
1537 005016 001065
1538 005020 022767 022322 174406
1539 005026 001003
1540 005030 004767 007456
1541 005034 000500
1542 005036 005367 174376 3$:
1543 005042 001355
1544 005044 005367 174372
1545 005050 001352
1546 005052 042777 000100 174330
1547 005060 042777 000104 174326
1548 005066 104401 016326
1549 005072 012767 005102 174144
1550 005100 104000
1551 005102 4$:
1552 005102 000455
1553 005104 016701 174300 5$:
1554 005110 016702 174300
1555 005114 011203
1556 005116 012704 000204
1557 005122 004767 006736
1558 005126 012767 005136 174110
1559 005134 104002
1560 005136 6$:
1561 005136 000437
1562 005140 016701 174244 7$:
1563 005144 010102
1564 005146 011203
1565 005150 012704 000200
1566 005154 004767 006704
1567 005160 012767 005170 174056
1568 005166 104002
1569 005170 8$:
1570 005170 000422
1571 005172 016701 174212 9$:
1572 005176 016702 174210
1573 005202 016703 173776
1574 005206 004767 006652
1575 005212 012767 005222 174024
1576 005220 004005
1577 005222 005267 174202 10$:
1578 005226 022767 000003 174174

```

```

*****
*TEST 22 TEST TO TURN AROUND NULL-DEL-NULL PATTERN
*****
TST2: SCOPE
MOV #1,$TIMES ; 1 ITERATION
JSR PC,SUVEC ; GO SET UP VECTORS
CLR RTRY ; INITIALIZE RETRY FLAG
MOV #LDOUT1,LDOUT ; SET POINTER TO LOAD ROUTINE
JSR PC,PRIME ; GO SET UP BUFFERS AND DEVICE
TST XFLGO ; ANY HARD XMIT ERRORS ??
BNE 5$ ; BR IF YES
TST RFLGO ; ANY HARD RECEIVER ERROR ??
BNE 7$ ; BR IF YES
TST RFLG1 ; ANY SOFT RECEIVER ERRORS ??
BNE 9$ ; BR IF YES
CMP #BUFENC,IPTR ; RECEIVED 256. BYTES ??
BNE 3$ ; BR IF NOT
JSR PC,CHKDAT ; GO CHECK THE DATA BUFFERS
BR TST23 ; <GO TO NEXT TEST>
DEC TIMR1 ; DEC TIMEOUT COUNTER 1
BNE 2$ ; BR IF NO TIMEOUT
DEC TIMR2 ; DEC TIMEOUT COUNTER 2
BNE 2$ ; BR IF NO TIMEOUT
BIC #100,$DLRCSR ; TURN OFF THE INTRAS.
BIC #104,$DLXCSR
TYPE ,XMSG1 ; GO TYPE TIMEOUT MESSAGE
MOV #4$,$ESCAPE ; GO TO 4$ AFTER ERROR PRINT
ERROR ; PRINT ERROR PC
4$:
BR TST23 ; <GO TO NEXT TEST>
5$:
MOV DLRCSR,R1 ; PUT DEVADR IN R1
MOV DLXCSR,R2 ; PUT REGADR IN R2
MOV (R2),R3 ; GET THE WAS DATA
MOV #204,R4 ; PUT S/B DATA IN R4
JSR PC,SUERR1 ; GO SET UP ERROR INFO
MOV #6$,$ESCAPE ; GO TO 6$ AFTER PRINTING ERROR
ERROR+2 ; TRANSMITTER FALSE INTERRUPT
6$:
BR TST23 ; <GO TO NEXT TEST>
7$:
MOV DLRCSR,R1 ; SAVE THE DEVADR
MOV R1,R2 ; SAVE THE REGADR
MOV (R2),R3 ; GET THE WAS DATA
MOV #200,R4 ; RESULT S/B = 200
JSR PC,SUERR1 ; GO SET UP ERROR INFO
MOV #8$,$ESCAPE ; GO TO 8$ AFTER ERROR PRINT
ERROR+2 ; RECEIVER FALSE INTERRUPT
8$:
BR TST23 ; <GO TO NEXT TEST>
9$:
MOV DLRCSR,R1 ; SAVE THE DEVADR
MOV DLROBR,R2 ; SAVE REGADR
MOV $TMP1,R3 ; GET CONTENTS OF ERROR ROBR
JSR PC,SUERR1 ; GO SETUP ERROR INFO
MOV #10$,$ESCAPE ; GO TO 10$ AFTER ERROR PRINT
ERROR+5 ; REPORT SOFT ERROR (PARITY,FRAMING, OR OVERRUN)
10$:
INC RTRY ; COUNT ONE TRY
CMP #3,RTRY ; TRIED THREE TIMES

```

K03

MAINDEC-11-DZDLC-A
DZDLC.A.P11 T22

MACY11 27(732) 20-SEP-76 09:19 PAGE 36
TEST TO TURN AROUND NULL-DEL-NULL PATTERN

1579 005234 001253

BNE 1\$

:BR IF NCT

```

1580      ;:*****
1581      ;*TEST 23      TEST TO TURN AROUND BINARY UP COUNT PATTERN
1582      ;:*****
1583      TST23:  SCOPE
1584      MOV      #1,$TIMES      ;:DO 1 ITERATION
1585      JSR      PC,SUVEC      ;:GO SET UP VECTORS
1586      CLR      RTRY          ;:INITIALIZE RETRY FLAG
1587      MOV      #LDOUT2,LDOUT ;:SET POINTER TO LOAD ROUTINE
1588      JSR      PC,PRIME      ;:GO SET UP BUFFERS AND DEVICE
1589      TST      XFLGO         ;:ANY HARD XMIT ERRORS ??
1590      BNE      5$           ;:BR IF YES
1591      TST      RFLGO         ;:ANY HARD RECEIVER ERROR ??
1592      BNE      7$           ;:BR IF YES
1593      TST      RFLG1         ;:ANY SOFT RECEIVER ERRORS ??
1594      BNE      9$           ;:BR IF YES
1595      CMP      #BUFEND,IPTR  ;:RECEIVED 256. BYTES ??
1596      BNE      3$           ;:BR IF NOT
1597      JSR      PC,CHKDAT     ;:GO CHECK THE DATA BUFFERS
1598      BR       TST24        ;:<GO TO NEXT TEST>
1599      DEC      TIMR1         ;:DEC TIMEOUT COUNTER 1
1600      BNE      2$           ;:BR IF NO TIMEOUT
1601      DEC      TIMR2         ;:DEC TIMEOUT COUNTER 2
1602      BNE      2$           ;:BR IF NO TIMEOUT
1603      BIC      #100,$DLRCSR  ;:TURN OFF THE INTRs.
1604      BIC      #104,$DLXCSR
1605      TYPE      ,XMSG2      ;:GO TYPE TIMEOUT MESSAGE
1606      MOV      #4$,$ESCAPE  ;:GO TO 4$ AFTER ERROR PRINT
1607      ERROR    ;:PRINT ERROR PC
1608      4$:
1609      BR       TST24        ;:<GO TO NEXT TEST>
1610      MOV      DLRCSR,R1     ;:PUT DEVADR IN R1
1611      MOV      DLXCSR,R2     ;:PUT REGADR IN R2
1612      MOV      (R2),R3      ;:GET THE WAS DATA
1613      MOV      #204,R4      ;:PUT S/B DATA IN R4
1614      JSR      PC,SUERR1     ;:GO SET UP ERROR INFO
1615      MOV      #6$,$ESCAPE  ;:GO TO 6$ AFTER PRINTING ERROR
1616      ERROR+2 ;:TRANSMITTER FALSE INTERRUPT
1617      6$:
1618      BR       TST24        ;:<GO TO NEXT TEST>
1619      MOV      DLRCSR,R1     ;:SAVE THE DEVADR
1620      MOV      R1,R2        ;:SAVE THE REGADR
1621      MOV      (R2),R3      ;:GET THE WAS DATA
1622      MOV      #200,R4      ;:RESULT S/B = 200
1623      JSR      PC,SUERR1     ;:GO SET UP ERROR INFO
1624      MOV      #8$,$ESCAPE  ;:GO TO 8$ AFTER ERROR PRINT
1625      ERROR+2 ;:RECEIVER FALSE INTERRUPT
1626      8$:
1627      BR       TST24        ;:<GO TO NEXT TEST>
1628      MOV      DLRCSR,R1     ;:SAVE THE DEVADR
1629      MOV      DLRDBR,R2     ;:SAVE REGADR
1630      MOV      $TMP1,R3      ;:GET CONTENTS OF ERROR RDBR
1631      JSR      PC,SUERR1     ;:GO SETUP ERROR INFO
1632      MOV      #10$,$ESCAPE  ;:GO TO 10$ AFTER ERROR PRINT
1633      ERROR+5 ;:REPORT SOFT ERROR (PARITY,FRAMING, OR OVERRUN)
1634      INC      RTRY          ;:COUNT ONE TRY
1635      CMP      #3,RTRY       ;:TRIED THREE TIMES

```

M03

MAINDEC-11-DZDLC-A
DZDLC.A.F11 T23

MACY11 27(732) 20-SEP-76 09:19 PAGE 39
TEST TO TURN AROUND BINARY UP COUNT PATTERN

1636 005526 301253

BNE 1\$

;BR IF NOT

```

1637
1638
1639
1640 005530 000004
1641 005532 012767 000001 173502
1642 005540 004767 006426
1643 005544 005067 173660
1644 005550 012767 014410 173660 1$:
1645 005556 004767 006436
1646 005562 005767 173634 2$:
1647 005566 001040
1648 005570 005767 173630
1649 005574 001053
1650 005576 005767 173624
1651 005602 001065
1652 005604 022767 022322 173622
1653 005612 001003
1654 005614 004767 006672
1655 005620 000500
1656 005622 005367 173612 3$:
1657 005626 001355
1658 005630 005367 173606
1659 005634 001352
1660 005636 042777 000100 173544
1661 005644 042777 000104 173542
1662 005652 104401 016466
1663 005656 012767 005666 173360
1664 005664 104000
1665 005666 4$:
1666 005666 000455
1667 005670 016701 173514 5$:
1668 005674 016702 173514
1669 005700 011203
1670 005702 012704 000204
1671 005706 004767 006152
1672 005712 012767 005722 173324
1673 005720 104002
1674 005722 6$:
1675 005722 000437
1676 005724 016701 173460 7$:
1677 005730 010102
1678 005732 011203
1679 005734 012704 000200
1680 005740 004767 006120
1681 005744 012767 005754 173272
1682 005752 104002
1683 005754 8$:
1684 005754 000422
1685 005756 016701 173426 9$:
1686 005762 016702 173424
1687 005766 016703 173212
1688 005772 004767 006066
1689 005776 012767 006006 173240
1690 006004 104005
1691 006006 005267 173416 10$:
1692 006012 022767 000003 173410

```

```

*****
:TEST 24 TEST TO TURN AROUND BINARY DOWN COUNT PATTERN
*****
TST24: SCOPE
MOV #1,$TIMES ;DO 1 ITERATION
JSR PC,SUVEC ;GO SET UP VECTORS
CLR RTRY ;INITIALIZE RETRY FLAG
MOV #LDOUT3,LDOUT ;SET POINTER TO LOAD ROUTINE
JSR PC,PRIME ;GO SET UP BUFFERS AND DEVICE
TST XFLGO ;ANY HARD XMIT ERRORS ??
BNE 5$ ;BR IF YES
TST RFLGO ;ANY HARD RECEIVER ERROR ??
BNE 7$ ;BR IF YES
TST RFLG1 ;ANY SOF RECEIVER ERRORS ??
BNE 9$ ;BR IF YES
CMP #BUFEND,IPTR ;RECEIVED 256. BYTES ??
BNE 3$ ;BR IF NOT
JSR PC,CHKDAT ;GO CHECK THE DATA BUFFERS
BR TST25 ;<GO TO NEXT TEST>
DEC TIMR1 ;DEC TIMEOUT COUNTER 1
BNE 2$ ;BR IF NO TIMEOUT
DEC TIMR2 ;DEC TIMEOUT COUNTER 2
BNE 2$ ;BR IF NO TIMEOUT
BIC #100,$DLRCSR ;TURN OFF THE INTRS.
BIC #104,$DLXCSR
TYPE ,XMSG3 ;GO TYPE TIMEOUT MESSAGE
MOV #4$, $ESCAPE ;GO TO 4$ AFTER ERROR PRINT
ERROR ;PRINT ERROR PC
4$: BR TST25 ;<GO TO NEXT TEST>
5$: MOV DLRCSR,R1 ;PUT DEVADR IN R1
MOV DLXCSR,R2 ;PUT REGADR IN R2
MOV (R2),R3 ;GET THE WAS DATA
MOV #204,R4 ;PUT S/B DATA IN R4
JSR PC,SUERR1 ;GO SET UP ERROR INFO
MOV #6$, $ESCAPE ;GO TO 6$ AFTER PRINTING ERROR
ERROR+2 ;TRANSMITTER FALSE INTERRUPT
6$: BR TST25 ;<GO TO NEXT TEST>
7$: MOV DLRCSR,R1 ;SAVE THE DEVADR
MOV R1,R2 ;SAVE THE REGADR
MOV (R2),R3 ;GET THE WAS DATA
MOV #200,R4 ;RESULT S/B = 200
JSR PC,SUERR1 ;GO SET UP ERROR INFO
MOV #8$, $ESCAPE ;GO TO 8$ AFTER ERROR PRINT
ERROR+2 ;RECEIVER FALSE INTERRUPT
8$: BR TST25 ;<GO TO NEXT TEST>
9$: MOV DLRCSR,R1 ;SAVE THE DEVADR
MOV DLRDBR,R2 ;SAVE REGADR
MOV $TMP1,R3 ;GET CONTENTS OF ERROR RDBR
JSR PC,SUERR1 ;GO SETUP ERROR INFO
MOV #10$, $ESCAPE ;GO TO 10$ AFTER ERROR PRINT
ERROR+5 ;REPORT SOFT ERROR (PARITY,FRAMING, OR OVERFLOW)
10$: INC RTRY ;COUNT ONE TRY
CMP #3,RTRY ;TRIED THREE TIMES

```


Vertical text on the left margin, possibly a list of addresses or labels, including 'TST25', '15:', '25:', '35:', '45:', '55:', '65:', '75:', '85:', '95:', '105:'.

Hexadecimal address column: 000000, 000001, 173210, 006134, 173366, 014444, 006144, 173342, 005104, 173336, 005105, 173332, 005107, 022322, 173330, 005400, 022012, 005336, 173316, 005337, 173312, 000100, 173250, 000104, 173246, 016551, 006162, 173264, 001742, 001670, 173216, 011203, 000204, 004767, 005654, 006220, 173226, 104002, 001704, 001670, 173160, 011010, 011203, 012704, 000200, 004767, 005627, 012767, 006254, 172772, 104002, 000167, 001650, 001670, 173124, 016702, 173122, 016703, 172700, 004767, 005664, 006310, 172736, 104005, 006306, 005267, 173114, 022767, 000003, 173106, 001247, 001600.

```
*****
TEST 25 TEST TO TURN AROUND WORST CASE PATTERN
*****
TST25: SEOP
MOV #1, $TIMES
JSR PC, SUVEC
RTR
LDOUT4, LDOUT
LDOUT4, LDOUT
JSR PC, PRIME
TST XFLG0
BNE $S
TST RFLG0
BNE $S
TST RFLG1
BNE $S
BGFEND, IPTR
JSR PC, CHKDAT
JMP SEOP
DEC TIMR1
BNE $S
DEC TIMR2
BNE $S
BIO #100, DLXCSR
BIO #104, DLXCSR
MOV #45, $ESCAPE
ERROR
JMP SEOP
MOV DLXCSR, R1
MOV DLXCSR, R2
MOV (R2), R3
MOV #204, R4
JSR PC, SUERR1
MOV #65, $ESCAPE
ERROR+2
JMP SEOP
MOV DLXCSR, R1
MOV R1, R2
MOV (R2), R3
MOV #200, R4
JSR PC, SUERR1
MOV #85, $ESCAPE
ERROR+2
JMP SEOP
MOV DLXCSR, R1
MOV DLXCSR, R2
MOV (R2), R3
JSR PC, SUERR1
MOV #105, $ESCAPE
ERROR+5
INC RTRY
CMP #3, RTRY
BNE $S
JMP SEOP
```

```
:: DO 1 ITERATION
: GO SET UP VECTORS
: INITIALIZE RETRY FLAG
: SET POINTER TO LOAD ROUTINE
: GO SET UP BUFFERS AND DEVICE
: ANY HARD XMIT ERRORS ??
: BR IF YES
: ANY HARD RECEIVED ERROR ??
: BR IF YES
: ANY SOFT RECEIVER ERRORS ??
: BR IF YES
: RECEIVED 256. BYTES ??
: BR IF NOT
: GO CHECK THE DATA BUFFERS
: GO TO NEXT TEST
: DEC TIMEOUT COUNTER 1
: BR IF NO TIMEOUT
: DEC TIMEOUT COUNTER 2
: BR IF NO TIMEOUT
: TURN OFF THE INTRs.
: GO TYPE TIMEOUT MESSAGE
: GO TO 45 AFTER ERROR PRINT
: PRINT ERROR PC
: GO TO NEXT TEST
: PUT DEVAOR IN R1
: PUT REGADR IN R2
: GET THE WAS DATA
: PUT S/B DATA IN R4
: GO SET UP ERROR INFO
: GO TO 65 AFTER PRINTING ERROR
: TRANSMITTER FALSE INTERRUPT
: GO TO NEXT TEST
: SAVE THE DEVAOR
: SAVE THE REGADR
: GET THE WAS DATA
: RESULT S/B = 200
: GO SET UP ERROR INFO
: GO TO 85 AFTER ERROR PRINT
: RECEIVER FALSE INTERRUPT
: GO TO NEXT TEST
: SAVE THE DEVAOR
: SAVE REGADR
: GET CONTENTS OF ERROR RCSR
: GO SETUP ERROR INFO
: GO TO 105 AFTER ERROR PRINT
: REPORT SOFT ERROR (PARITY, FRAMING, OR OVERRUN)
: COUNT ONE TRY
: TRIED THREE TIMES
: BR IF NOT
: GO TO END OF PASS ROUTINE
```

1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805

006330	012706	001100	
006334	012707	013052	000034
006342	012707	000340	000036
006350	012707	010732	000037
006356	012707	000340	000032
006364	104401	016664	
006370	104401	020306	
006374	104410		
006376	012602		
006400	020227	176176	
006404	020227		
006406	020227	175616	
006412	102462		
006414	102702	000001	
006420	001057		
006422	010203		
006424	142703	000370	
006430	122703	000006	
006434	001057		
006436	010267	172540	
006442	016746	171336	
006446	012767	006460	171330
006454	005712		
006456	000412		
006460	004767	005460	
006464	012767	006474	172552
006472	104006		
006474	022626		
006476	012667	171302	
006502	000426		
006504	012667	171274	

```

: THIS IS PROGRAM #2
: THE FOLLOWING USER UTILITY PROGRAM WILL ALLOW:
: A) SELECTION OF A TRANSMITTED DATA BUFFER
: B) SELECTION OF A CHARACTER FOR CONTINUOUS TRANSFER
: C) SELECTION OF AN EXPIRATION TIME IN MILLISECONDS
:   BETWEEN EACH TRANSMITTER DATA BUFFER CHARACTER TRANSFER
: D) A TIGHT SCOPE LOOP LOCK ON A SPECIFIC CHARACTER
:
PRG2:  MOV     #STACK,SP           :INITIALIZE THE STACK POINTER
      MOV     #STRAP,#TRAPVEC     :TRAP VECTOR FOR TRAP CALLS
      MOV     #340,#TRAPVEC+2    :LEVEL 7
      MOV     #ERROR,#EMTVEC     :EMT VECTOR FOR ERROR ROUTINE
      MOV     #340,#EMTVEC+2    :LEVEL 7
      TYPE    .PRG2M             :INDICATE THAT USER SELECTED
                                   :PROGRAM #2
PRG2A: TYPE    .LINTAD           :ASK USER FOR THE TRANSMITTER
                                   :DATA BUFFER ADDRESS OF THE DEVICE
                                   :HE WISHES TO TEST
      RDOCT                      :ACCEPT THE ANSWER TYPED BY USER
                                   :AND STORE ON TOP OF STACK
:CHECK TO SEE IF THE USER RESPONSE WAS WITHIN LIMITS
      MOV     (SP)+,R2           :GET THE ANSWER TYPED
      CMP     R2,#176176        :IS THE NUMBER TOO HIGH?
      BHI     REDC1             :IF YES - GO TO RETRY SITUATION
      CMP     R2,#175616        :IS THE NUMBER TOO LOW?
      BLO     REDC1             :IF YES - GO TO RETRY SITUATION
      BITB   #BIT0,R2          :NUMBER IS IN RANGE BUT IS IT
                                   :ON AN EVEN BOUNDARY?
      BNE     REDC1             :IF NOT GO TO RETRY SITUATION
:CHECK TO SEE IF USER RESPONSE WAS TRULY A XMIT BUFFER REGISTER
      MOV     R2,R3             :GET THE USER RESPONSE
      BICB   #370,R3           :MASK OFF LOWER BYTE EXCEPT FOR
                                   :LEAST SIGNIFICANT DIGIT
      CMPB   #6,R3             :WAS THE LEAST SIGNIFICANT DIGIT OF THE
                                   :USER RESPONSE EQUAL TO A SIX?
      BNE     REDC1             :BRANCH IF NOT
      MOV     R2,$TMPD         :THE TRANSMITTER ADDRESS
                                   :TYPED IS OK - STORE FOR
                                   :FUTURE USE
:NOW CHECK TO MAKE SURE THE DEVICE IS PRESENT
      MOV     ERRVEC,-(SP)      :SAVE THE TIMEOUT VECTOR
      MOV     #2$ ,ERRVEC      :SET UP TIMEOUT SERVICE ADDRESS
      TST    (R2)              :IF PRESENT WE WILL EXECUTE THE
                                   :NEXT INSTRUCTION - IF NOT
                                   :WE GO TO 2$:
      BR     4$                :BRANCH IF PRESENT
2$:   JSR    PC,SUERT2         :GO SET UP FOR ERROR INFORMATION
      MOV    #3$, $ESCAPE     :POINT OF RETURN AFTER ERROR REPORT
      ERROR +6                :XDRR REFERENCE CAUSED "TIMEOUT"
3$:   CMP    (SP)+,(SP)+      :CLEAN STACK FROM TIMEOUT
      MOV    (SP)+,ERRVEC     :RESTORE TIMEOUT VECTOR
      BR     REDC1           :GO TO RETRY SITUATION
4$:   MOV    (SP)+,ERRVEC     :DEVICE REGISTER IS PRESENT!
                                   :RESTORE TIMEOUT VECTOR
:WE ARE NOW READY FOR THE CHARACTER TO BE TRANSMITTED, AND THE

```

```

:006      :DELAY TIME (IN MILLISECONDS) THAT IS TO TRANSPIRE BETWEEN
:007      :SUCCESSIVE CHARACTER TRANSFERS
:008      PRG2B: TYPE .SELCAR      :ASK USER FOR THE CHARACTER HE
:009      006510 104401 020367      :WISHES TO TRANSFER
:010      :ACCEPT THE ANSWER TYPED BY
:011      006514 104410      RDOCT      :USER AND STORE ON TOP OF STACK
:012      006516 012667 172462      MOV (SP)+,$TMP1      :GET THE ANSWER TYPED
:013      :NOTE: THE USER RESPONSE FOR THE CHARACTER WAS TO BE THE
:014      :      CCTAL ASCII EQUIVALENT OF THE CHARACTER E.G. A=101
:015      006522 104401 020475      TYPE .SELDLY      :ASK THE USER FOR THE DELAY
:016      :IN MSEC (OCTAL NO.) BETWEEN
:017      006526 104410      RDOCT      :CHARACTER TRANSFERS
:018      006530 012667 172452      MOV (SP)+,$TMP2      :ACCEPT THE ANSWER TYPED BY
:019      006534 116767 172446 000012 15: MOV9 $TMP2,25      :USER AND STORE ON TOP OF STACK
:020      :GET THE ANSWER TYPED
:021      006542 116777 172436 172432 MOV8 $TMP1,$TMP0      :SET THE DELAY COUNT ARGUMENT
:022      :FOR TIMER ROUTINE
:023      006550 024767 024750      JSR PC,DELAY      :LOAD THE TRANSMITTER DATA
:024      :BUFFER WITH THE CHARACTER
:025      :GO OFF TO WAIT THE SPECIFIED

```

```

1826                                     ;NO. OF MSEC. BEFORE ISSUING
1827                                     ;ANOTHER CHARACTER
1828 005554 000000 2$: .WORD 0 ;THIS IS WHERE THE DELAY COUNT RESIDES
1829 006556 000766 BR 1$ ;GO BACK TO ISSUE ANOTHER CHARACTER
1830 006560 104401 001252 RED01: TYPE $QUES ;TYPE A QUESTION MARK(?)
1831 006564 000167 177500 JMP PRG2A ;REITERATE THE XDBR QUESTION TO USER
1832                                     ;THIS IS PROGRAM #3
1833                                     ;THE FOLLOWING USER UTILITY PROGRAM WILL ALLOW:
1834                                     A) SELECTION OF A RECEIVER DATA BUFFER
1835                                     B) SELECTION OF A CHARACTER FOR CONTINUOUS TRANSFER
1836                                     C) SELECTION OF AN EXPIRATION TIME IN MILLISECONDS
1837                                     BETWEEN EACH RECEIVER DATA BUFFER CHARACTER TRANSFER
1838                                     D) A TIGHT SCOPE LOOP LOCK ON A SPECIFIC CHARACTER
1839
1840
1841 006570 012706 001100 PRG3: MOV #STACK,SP ;INITIALIZE THE STACK POINTER
1842 006574 012737 013052 000034 MOV #STRAP,2#TRAPVEC ;TRAP VECTOR FOR TRAP CALLS
1843 006602 012737 000340 000036 MOV #340,2#TRAPVEC+2 ;LEVEL 7
1844 006610 012737 010732 000030 MOV #SEPROR,2#EMTVEC ;EMT VECTOR FOR ERROR ROUTINE
1845 006616 012737 000340 000032 MOV #340,2#EMTVEC+2 ;LEVEL 7
1846 006624 104401 016730 TYPE .PRG3M ;INDICATE THAT USER SELECTED
1847                                     PROGRAM #3
1848 006630 104401 020570 PRG3A: TYPE .LINRAD ;ASK USER FOR THE RECEIVER DATA
1849                                     BUFFER ADDRESS OF THE DEVICE
1850                                     HE WISHES TO TEST
1851 006634 104410 RDOCT ;ACCEPT THE ANSWER TYPED BY
1852                                     USER AND STORE ON TOP OF STACK
1853 ;CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
1854 006636 012602 MOV (SP)+,R2 ;GET THE ANSWER TYPED
1855 006640 020227 176172 CMP R2,#176172 ;IS THE NUMBER TOO HIGH?
1856 006644 101071 BHI RED02 ;IF YES - GO TO RETRY SITUATION
1857 006646 020227 175612 CMP R2,#175612 ;IS THE NUMBER TOO LOW?
1858 006652 103466 BLO RED02 ;IF YES - GO TO RETRY SITUATION
1859 006654 132702 000001 B.TE #BIT0,R2 ;NUMBER IS IN RANGE BUT IS IT
1860                                     ON AN EVEN BOUNDARY?
1861 006660 001063 BNE RED02 ;IF NOT - GO TO RETRY SITUATION
1862 ;CHECK TO SEE IF USER RESPONSE WAS TRULY A RCVR DBR ADDRESS
1863 006662 010203 MOV R2,R3 ;GET THE USER RESPONSE
1864 006664 142703 000370 BICB #370,R3 ;MASK OFF LOWER BYTE EXCEPT FOR
1865                                     LEAST SIGNIFICANT DIGIT
1866 006670 122703 000002 CMPB #2,R3 ;WAS THE LEAST SIGNIFICANT DIGIT OF THE
1867                                     USER RESPONSE EQUAL TO A TWO?
1868 006674 001055 BNE RED02 ;BRANCH IF NOT
1869 006676 010267 172300 MOV R2,$TMP0 ;THE RECEIVER ADDRESS TYPED IS
1870                                     OK - STORE FOR FUTURE USE
1871 ;NOW CHECK TO MAKE SURE THE DEVICE IS PRESENT
1872 006702 016746 171076 MOV ERRVEC,-(SP) ;SAVE THE TIMEOUT VECTOR
1873 006706 012767 006720 171070 MOV #2$,ERRVEC ;SET UP TIMEOUT SERVICE ADDRESS
1874 006714 005712 TST (R2) ;IF PRESENT WE WILL EXECUTE THE
1875                                     NEXT INSTRUCTION - IF NOT WE
1876                                     GO TO 2$:
1877 006716 000412 BR 4$ ;BRANCH IF PRESENT
1878 006720 004767 005220 2$: JSR PC,SUERT2 ;GO SET UP FOR ERROR INFORMATION
1879 006724 012767 005734 172312 MOV #3$, $ESCAPE ;POINT OF RETURN AFTER ERROR REPORT
1880 006732 104006 ERROR +6 ;RDBR REFERENCE CAUSED TIMEOUT
1881 006734 022626 3$: CMP (SP)+,(SP)+ ;CLEAN STACK FROM TIMEOUT

```


H04

MAINDEC-11-DZOLC-A
DZOLC.A.P11 T25

MAY 11 27 (732) 20-SEP-76 09:19 PAGE 46
TEST TO TURN AROUND WORST CASE PATTERN

```

:938          :CHECK TO SEE IF THE USER RESPONSE WAS WITHIN LIMITS
1939 007106 012602      MOV      (SP)+,R2      :GET THE ANSWER TYPED
1940 007117 020227 176176  CMP      R2,#176176   :IS THE NUMBER TOO HIGH?
1941 007114 101136      BHI      RED03        :IF YES - GO TO RETRY SITUATION
1942 007116 020227 175616  CMP      R2,#175616   :IS THE NUMBER TOO LOW?
1943 007122 103533      BLC      RED03        :IF YES - GO TO RETRY SITUATION
1944 007124 132702 000001  BITB     #BIT0,R2     :NUMBER IS IN RANGE BUT IS IT
1945          :ON AN EVEN BOUNDARY?
1946 007130 001130      BNE      RED03        :IF NO - GO TO RETRY SITUATION
1947          :CHECK TO SEE IF USER RESPONSE WAS TRULY A XMIT BUFFER REGISTER
1948 007132 010203      MOV      R2,R3        :GET THE USER RESPONSE
1949 007134 142703 000370  BICB     #370,R3      :MASK OFF LOWER BYTE EXCEPT FOR
1950          :LEAST SIGNIFICANT DIGIT
1951 007140 122703 000006  CMPB     #6,R3        :WAS THE LEAST SIGNIFICANT DIGIT OF THE
1952          :USER RESPONSE EQUAL TO A SIX?
1953 007144 001122      BNE      RED03        :BRANCH IF NOT
1954 007146 010267 172030  MOV      R0,$TMP0     :THE TRANSMITTER ADDRESS TYPED
1955          :IS OK - STORE FOR FUTURE USE
1956          :NOW CHECK TO MAKE SURE THE DEVICE IS PRESENT
1957 007152 016746 170626  MOV      ERRVEC,-(SP)  :SAVE THE TIMEOUT VECTOR
1958 007156 012767 007170 170620  MOV      #25,ERRVEC   :SET UP TIMEOUT SERVICE ADDRESS
1959 007164 005712      TST      (R2)         :IF PRESENT WE WILL EXECUTE THE
1960          :NEXT INSTRUCTION - IF NOT WE
1961          :GO TO 25:
1962 007166 000412      BR       4$           :BRANCH IF PRESENT
1963 007170 004767 004750 25:     JSR      PC,$JERT2   :GO SET UP FOR ERROR INFORMATION
1964 007174 012767 007204 172042  MOV      #3$, $ESCAPE :POINT OF RETURN AFTER ERROR REPORT
1965 007202 104006      ERROR    +6          :XDBR REFERENCE CAUSED TIMEOUT
1966 007204 022626 35:     CMP      (SP)+,(SP)+ :CLEAN STACK FROM TIMEOUT
1967 007206 012667 170572  MOV      (SP)+,ERRVEC :RESTORE TIMEOUT VECTOR
1968 007212 000477      BR      RED03        :GO TO RETRY SITUATION
1969 007214 012667 170564 45:     MOV      (SP)+,ERRVEC :DEVICE REGISTER IS PRESENT!
1970          :RESTORE TIMEOUT VECTOR
1971 007220 104401 020646      TYPE     ,RSTALL    :ASK THE USER IF HE DESIRES SOME
1972          :RANDOM NO. OF MSEC. WAIT TIME
1973          :BEFORE CHECKING FOR XCSR DONE
1974          :FLAG
1975 007224 104410      RDOCT          :ACCEPT THE ANSWER TYPED BY USER
1976          :AND STORE ON TOP OF STACK
1977 007226 012667 171754  MOV      (SP)+,$TMP2  :GET THE ANSWER TYPED
1978          :WE ARE NOW READY FOR THE CHARACTER TO BE TRANSMITTED
1979 007232 104401 020367 PRG4B: TYPE     ,SELCAR :ASK USER FOR THE CHARACTER HE
1980          :WISHES TO TRANSFER
1981 007236 104410      RDOCT          :ACCEPT THE ANSWER TYPED BY USER
1982          :AND STORE ON TOP OF STACK
1983 007240 012667 171740  MOV      (SP)+,$TMP1  :GET THE ANSWER TYPED
1984          :NOTE: THE USER RESPONSE FOR THE CHARACTER WAS TO BE THE OCTAL
1985          :ASCII EQUIVALENT OF THE CHARACTER E.G. C=103
1986          :
1987 007244 104401 017217 PRG4C: TYPE     ,LENGTH ;ASK USER FOR THE CHARACTER LENGTH
1988          :FOR WHICH HIS DEVICE IS SET
1989 007250 104411      RDECT          :ACCEPT THE ANSWER TYPED BY USER
1990          :CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
1991 007252 012600      MOV      (SP)+,R0    :GET THE ANSWER TYPED
1992 007254 020027 000010  CMP      R0,#8        :IS THE NUMBER TOO HIGH?
1993 007260 101060      BHI      RED03A      :IF YES - GO TO RETRY SITUATION

```

MAINDEC-11-DZDLC-A
DZDLC.A.P11 T25

MACY11 27(732) 20-SEP-76 09:19 PAGE 47
TEST TO TURN AROUND WORST CASE PATTERN

```

1994 007262 020027 000005          CMP      RO,#5          ;IS THE NUMBER TOO LOW?
1995 007266 103455          BLO     RED03A         ;IF YES - GO TO RETRY SITUATION
1996 007270 010067 171740          MOV     RO,$TMP15      ;THE VALUE TYPED IS OK
1997                                     ;STORE FOR FUTURE USE
1998 007274 016767 171702 171706          MOV     $TMP0,$TMP3   ;GET THE XDBR ADDRESS
1999 007302 162767 000002 171700          SUB     #2,$TMP3      ;FORM THE XCSR ADDRESS
2000 007310 005767 171672          TST     $TMP2         ;DO WE RANDOM STALL?
2001 007314 001402          BEQ     2$           ;BRANCH IF IT WASN'T DESIRED
2002 007316 004767 004246          JSR     PC,$STALL     ;GO STALL RANDOM VALUE OF MSEC.
2003 007322 004767 004352          JSR     PC,$TIMETX    ;GO WAIT FOR TRANSMITTER DONE
2004                                     ;BIT TO SET
2005 007326 104401 017104          TYPE   ,XDB          ;TYPE TRANSMITTER DONE BIT MESSAGE
2006 007332 104000          ERROR  +0           ;XCSR DONE BIT NEVER SET
2007 007334 052777 000004 171646          BIS     #BIT2,$TMP3   ;SET THE MAINTENANCE BIT IN THE
2008                                     ;TRANSMITTER CONTROL STATUS REGISTER
2009 007342 016777 171636 171632          MOV     $TMP1,$TMP0   ;LOAD TRANSMITTER DATA BUFFER
2010                                     ;WITH SELECTED CHARACTER
2011 007350 004767 004306          JSR     PC,$TIMERX    ;GO WAIT FOR RECEIVER DONE BIT
2012                                     ;TO SET
2013 007354 104401 017153          TYPE   ,RDB          ;TYPE RECEIVER DONE BIT MESSAGE
2014 007360 104000          ERROR  +0           ;RCSR DONE BIT NEVER SET
2015 007362 016767 171622 171622          MOV     $TMP3,$TMP4   ;GET THE TRANSMITTER CONTROL
2016                                     ;STATUS REGISTER ADDRESS
2017 007370 162767 000002 171614          SUB     #2,$TMP4      ;FORM THE RECEIVER DATA BUFFER
2018                                     ;ADDRESS
2019 007376 017767 171610 171610          MOV     $TMP4,$TMP5   ;STORE THE CHARACTER FROM THE
2020                                     ;RECEIVER BUFFER + REST OF CONTENTS
2021 007404 004767 004326          JSR     PC,$DATCHK    ;GO TO COMPARE EXPECTED & RECEIVED
2022                                     ;DATA
2023 007410 000737          BR     1$           ;GO BACK TO ISSUE ANOTHER CHARACTER
2024 007412 104401 001252          RED03: TYPE   ,SQJES  ;TYPE A QUESTION MARK(?)
2025 007416 000167 177456          JMP     PRG4A        ;REITERATE THE XDBR QUESTION TO USER
2026 007422 104401 001252          RED03A: TYPE  ,SQUES  ;TYPE '?' INDICATING USER TYPED
2027                                     ;SOMETHING WRONG FOR CHARACTER LENGTH
2028 007426 000167 177612          JMP     PRG4C        ;GO BACK TO REISSUE QUESTION
2029
2030                                     ;THIS IS PROGRAM #5
2031                                     ;THE FOLLOWING USER UTILITY PROGRAM WILL ALLOW USER PARAMETERS
2032                                     ;FOR RUNNING A BINARY COUNT IN MAINTENANCE MODE
2033
2034
2035 007432 012706 001100          PRG5:  MOV     #STACK,SP ;INITIALIZE THE STACK POINTER
2036 007436 012737 013052 000034          MOV     #STRAP,$TRAPVEC ;TRAP VECTOR FOR TRAP CALLS
2037 007444 012707 000340 000036          MOV     #340,$TRAPVEC+2 ;LEVEL 7
2038 007452 012737 010732 00003C          MOV     #ERROR,$EMTVEC  ;EMT VECTOR FOR ERROR ROUTINE
2039 007460 012737 000340 000032          MOV     #340,$EMTVEC+2  ;LEVEL 7
2040 007466 104401 017040          TYPE   ,PROG5M        ;INDICATE THAT USER SELECTED
2041                                     ;PROGRAM #5
2042 007472 104401 020306          PRG5A: TYPE   ,LINTAD  ;ASK USER FOR THE TRANSMITTER DATA
2043                                     ;BUFFER ADDRESS OF THE DEVICE
2044                                     ;HE WISHES TO TEST
2045 007476 104410          RDOCT                ;ACCEPT THE ANSWER TYPED BY USER
2046                                     ;AND STORE ON TOP OF STACK
2047                                     ;CHECK TO SEE IF THE USER RESPONSE WAS WITHIN LIMITS
2048 007500 012602          MOV     (SP)+,R2      ;GET THE ANSWER TYPED
2049 007502 020227 176176          CMP     R2,#176176    ;IS THE NUMBER TOO HIGH?

```



```

2050 007506 101152          BHI    RED04          ; IF YES - GO TO RETRY SITUATION
2051 007510 020227 175616    CMP    R2,#175616    ; IS THE NUMBER TOO LOW?
2052 007514 103547          BLO    RED04          ; IF YES - GO TO RETRY SITUATION
2053 007516 132702 000001    BITB   #BIT0,R2     ; NUMBER IS IN RANGE BUT IS IT
2054                                     ; ON AN EVEN BOUNDARY?
2055 007522 001144          BNE    RED04          ; IF NOT - GO TO RETRY SITUATION
2056                                     ; CHECK TO SEE IF USER RESPONSE WAS TRULY A XMIT BUFFER REGISTER
2057 007524 010203          MOV    R2,R3         ; GET THE USER RESPONSE
2058 007526 142703 000370    BICB   #370,R3 ; MASK OFF LOWER BYTE EXCEPT FOR
2059                                     ; LEAST SIGNIFICANT DIGIT
2060 007532 122703 000006    CMPB   #6,R3        ; WAS THE LEAST SIGNIFICANT DIGIT OF THE
2061                                     ; USER RESPONSE EQUAL TO A SIX?
2062 007536 001136          BNE    RED04          ; BRANCH IF NOT
2063 007540 010267 171436    MOV    R2,$TMP0     ; THE TRANSMITTER ADDRESS TYPED
2064                                     ; IS OK - STORE FOR FUTURE USE
2065                                     ; NOW CHECK TO MAKE SURE THE DEVICE IS PRESENT
2066 007544 016746 170234    MOV    ERRVEC, -(SP) ; SAVE THE TIMEOUT VECTOR
2067 007550 012767 007562 170226  MOV    #2,$ERRVEC   ; SET UP TIMEOUT SERVICE ADDRESS
2068 007556 005712          TST    (R2)          ; IF PRESENT WE WILL EXECUTE THE
2069                                     ; NEXT INSTRUCTION - IF NOT WE
2070                                     ; GO TO 2$:
2071 007560 000412          BR     4$            ; BRANCH IF PRESENT
2072 007562 004767 004354 2$: JSR    PC,$JERT2    ; GO SETUP FOR ERROR INFORMATION
2073 007566 012767 007576 171450  MOV    #3,$$ESCAPE ; POINT OF RETURN AFTER ERROR REPORT
2074 007574 104006          ERROR  +6           ; XDR REFERENCE CAUSED TIMEOUT
2075 007576 022526 3$: CMP    (SP)+,(SP)+ ; CLEAN STACK FROM TIMEOUT
2076 007600 012667 170200    MOV    (SP)+,$ERRVEC ; RESTORE TIMEOUT VECTOR
2077 007604 000513          BR     RED04         ; GO TO RETRY SITUATION
2078 007606 012667 170172 4$: MOV    (SP)+,$ERRVEC ; DEVICE REGISTER IS PRESENT!
2079                                     ; ASK THE USER IF HE DESIRES SOME
2080                                     ; RANDOM NO. OF MSEC. WAIT TIME
2081                                     ; BEFORE CHECKING XCSR DONE FLAG
2082 007612 104401 017217    PRG5C: TYPE, LENGTH ; ASK USER FOR THE CHARACTER LENGTH
2083                                     ; FOR WHICH HIS DEVICE IS SET
2084 007616 104411          RDEC                                     ; ACCEPT THE ANSWER TYPED BY USER
2085                                     ; CHECK TO SEE IF USER RESPONSE WAS WITHIN LIMITS
2086 007620 012600          MOV    (SP)+,R0     ; GET THE ANSWER TYPED
2087 007622 020027 000010    CMP    R0,#8        ; IS THE NUMBER TOO HIGH?
2088 007626 101106          BHI    RED04A       ; IF YES - GO TO RETRY SITUATION
2089 007630 020027 000005    CMP    R0,#5        ; IS THE NUMBER TOO LOW?
2090 007634 103503          BLO    RED04A       ; IF YES - GO TO RETRY SITUATION
2091 007636 010067 171372    MOV    R0,$TMP15   ; THE VALUE TYPED IS OK
2092                                     ; STORE FOR FUTURE USE
2093 007642 104401 020646    TYPE   .RSTALL     ; RANDOM NO. OF MSEC. WAIT TIME
2094 007646 104410          RDOCT                                     ; ACCEPT THE ANSWER TYPED BY USER
2095                                     ; AND STORE ON TOP OF STACK
2096 007650 012667 171332    MOV    (SP)+,$TMP2 ; GET THE ANSWER TYPED
2097                                     ; WE ARE NOW READY TO INITIALIZE THE BINARY COUNT AND GET
2098                                     ; THE BINARY CHARACTER
2099                                     ;
2100 007654 012767 177777 171342  MOV    #-1,$TMP11  ; SET LEAD IN VARIABLE TO -1
2101 007662 016767 171336 171336  PRG5B: MOV    $TMP11,$TMP12 ; STORE PREVIOUS BINARY CHARACTER
2102 007670 005267 171332          INC    $TMP12       ; FLIP BINARY CHARACTER AGAIN
2103 007674 042767 177400 171324  BIC    #177400,$TMP12 ; MASK TO 2 BITS
2104 007702 016767 171320 171314  MOV    $TMP12,$TMP11 ; STORE BINARY CHARACTER
2105 007710 016767 171312 171266  MOV    $TMP12,$TMP1 ; STORE BINARY CHARACTER

```



```

2162
2163
2164
2165
2166
2167
2168
2169
2170
2171 01013C
2172
2173
2174
2175
2176 010130 000004
2177 010132 105767 171134
2178 010136 001501
2179
2180 010140 005767 171130
2181 010144 001011
2182 010146 104401 020074
2183
2184
2185
2186 010152 000000
2187 010154 005067 171112
2188 010160 005067 171072
2189 010164 000167 171502
2190
2191
2192 010170 062767 000010 171066 1$: ADD #10,BASEADD
2193
2194
2195 010176 062767 000010 171064
2196
2197
2198 010204 000241
2199 010206 006167 171054
2200
2201 010212 103431
2202
2203
2204 010214 036767 171056 171052
2205
2206 010222 001762
2207
2208 010224 016767 171034 171026
2209
2210 010232 016767 171032 171160
2211 010240 000240
2212 010242 004767 177606
2213
2214 010246 005067 170630
2215
2216 010252 005077 171136
2217 010256 005077 171126

```

```

.SBTTL END OF PASS ROUTINE

:*****
:*INCREMENT THE PASS NUMBER ($PASS)
:*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
:*IF THERES A MONITOR GO TO IT
:*IF THERE ISN'T JUMP TO RESTRT

$EOP:
:THIS NEXT SECTION UP TO THE NEXT LINE OF ASTERISKS WAS
:SUPPLIED BY THE MACRO 'EOPBEG'. THE MACRO NAME APPEARS IN
:THE SOURCE PROGRAM AS ONE OF THE ARGUMENTS TO THE .$EOP
:SYSMAC UTILITY ROUTINE CALL.

SCOPE
TSTB MULTD ;ARE WE RUNNING MULTIPLE DEVICES?
BEQ 3$ ;BRANCH IF NOT FOR NORMAL
;'END PASS # XX' TYPEOUT
TST ACTREG ;ARE ANY DEVICES ACTIVE?
BNE 1$ ;BRANCH IF YES
TYPE ,FOULUP ;INDICATE SOMETHING WRONG!
;MULTIPLE DEVICES ARE BEING
;RUN SUPPOSEDLY, BUT NONE ARE
;SHOWN ACTIVE
;WAIT FOR A USER RESPONSE
CLR MULTD ;CLEAR MULTIPLE DEVICE FLAG
CLR TABFLG ;CLEAR TABLE CREATION FLAG
JMP RESTRT ;GET READY TO START ALL OVER
;AGAIN - ALL DEVICES WERE
;DESELECTED SOMEHOW!
;FORM A NEW BASE
;ADDRESS FOR START OF NEXT BLOCK
;OF REGISTERS FOR NEXT DEVICE
;FORM A NEW BASE ADDRESS FOR
;START OF NEXT BLOCK OF CTORS
;FOR NEXT DEVICE
CLR LAST DEVICE INDICATOR
ROL ROTADD ;UPDATE NEXT POSSIBLE DEVICE ACTIVE
;POINTER
BCS 2$ ;BRANCH IF THIS WAS THE
;LAST DEVICE TO BE TESTED ON
;THIS PASS
BIT ROTADD,ACTREG ;IS THIS DEVICE TRULY ACTIVE
;(AS PER USER)
BEQ 1$ ;BRANCH IF NOT TO SEE IF NEXT
;ONE POSSIBLE IS
MOV BASEADD,DLBASE ;FORM THE RECEIVER STATUS REGISTER
;ADDRESS OF NEXT DEVICE
MOV BASEIV,DLVECT ;GET NEXT DEVICE RCVR VECTOR
NOP
JSR PC,DLADDR ;GO FORM DL ADDRESSES FOR NEXT
;DEVICE SELECTED
CLR $TSTNM ;INITIALIZE TEST NO. FOR A PROGRAM
;PASS OVER THE NEXT DEVICE ACTIVE
CLR $DLXCSR ;CLEAR OUT BOTH CSR'S
CLR $DLRCSR

```

```

2218 010262 005777 171124          TST    QDLRDBR          ;FLUSH RCVR "DONE" BIT
2219 010266 005777 171120          TST    QDLRDBR
2220 010272 000167 172472          JMP    TST1            ;START TESTING THIS DEVICE
2221 010275
2222
2223
2224
2225 010276 012767 000001 170772    MOV    #1,ROTADD      ;SET UP ROTATING POINTER FOR NEXT
2226
2227 010304 016767 170752 170752    MOV    KEEPADD,BASEADD ;RESTORE BASE ADDRESS
2228 010312 016767 170750 170750    MOV    KEEPIV,BASEIV  ;RESTORE BASE INTERRUPT VECTOR
2229 010320 016767 170740 170732    MOV    BASEADD,DLBASE ;RESTORE 1ST DEVICE BASE ADDRESS
2230 010326 016767 170736 171064    MOV    BASEIV,DLVECT  ;RESTORE 1ST DEVICE VECTOR ADDRESS
2231 010334 000240
2232 010335 004767 177512          JSR    PC,DLADDR      ;FORM ADDRESSES FOR 1ST DEVICE
2233 010342
2234
2235 010342 005067 170534          CLR    $STNM          ;;ZERO THE TEST NUMBER
2236 010346 005067 170670          CLR    $TIMES        ;;ZERO THE NUMBER OF ITERATIONS
2237 010352 005267 170522          INC    $PASS         ;;INCREMENT THE PASS NUMBER
2238 010356 042757 100000 170514    BIC    #10000,$PASS  ;;DON'T ALLOW A NEG. NUMBER
2239 010364 C 377          DEC    (PC)+         ;;LOOP?
2240 010366 000001          SEOPCT: .WORD 1
2241 010370 003022          BGT    $DOAGN        ;;YES
2242 010372 012737          MOV    (PC)+,Q(PC)+  ;;RESTORE COUNTER
2243 010374 000001          SENDCT: .WORD 1
2244 010376 010366          SEOPCT
2245 010400 104401 010445          TYPE   $SENDMG       ;;TYPE "END PASS #"
2246 010404 016746 170470          MOV    $PASS,-(SP)   ;;SAVE $PASS FOR TYPEOUT
2247 010410 104405          TYPDS  ;;GO TYPE--DECIMAL ASCII WITH SIGN
2248 010412 104401 010442          TYPE   $ENULL        ;;TYPE A NULL CHARACTER
2249 010416 013700 000042          $GET42: MOV    Q#42,R0  ;;GET MONITOR ADDRESS
2250 010422 001405          BEQ    $DOAGN        ;;BRANCH IF NO MONITOR
2251 010424 000005          RESET  ;;CLEAR THE WORLD
2252 010426 004710          SENDAD: JSR    PC,(R0) ;;GO TO MONITOR
2253 010430 000240          NOP    ;;SAVE ROOM
2254 010432 000240          NOP    ;;FOR
2255 010434 000240          NOP    ;;ACT11
2256 010436
2257 010436 000137          $DOAGN: JMP    Q(PC)+        ;;RETURN
2258 010440 001772          $RTNAD: .WORD  RESTR
2259 010442 377 377 300          $ENULL: .BYTE  -1,-1,0 ;;NULL CHARACTER STRING
2260 010445 015 042412 04E.16          $ENDMG: .ASCIZ  <15><12>/END PASS #/
2261 010452 050040 051501 020123
2262 010460 000043

```

.SBTTL SCOPE HANDLER ROUTINE

```

2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273

```

```

;*****
;THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCRMENT
;AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
;THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;*SW14=1      LOOP ON TEST
;*SW11=1      INHIBIT ITERATIONS
;*SW09=1      LOOP ON ERROR
;*SW08=1      LOOP ON TEST IN SWR<7:0>

```

```

2274          ;*CALL
2275          ;*      SCOPE          ;;SCOPE=IOT
2276
2277          $SCOPE:
2278          1$: BIT      #BIT14, $SWR      ;;LOOP ON PRESENT TEST?
2279          010462 032777 040000 170450 BNE      $OVER          ;;YES IF SW14=1
2280          010470 001111
2281          010472 000416          ;*****START OF CODE FOR THE XOR TESTER*****
2282          $XTSTR: BR      6$          ;;IF RUNNING ON THE "XCR" TESTER CHANGE
2283          ;*****END OF CODE FOR THE XOR TESTER*****
2284          010474 013746 000004          MOV      @#ERRVEC, -(SP)      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
2285          010500 012737 010520 000004          MOV      #5$, @#ERRVEC      ;;SET FOR TIMEOUT
2286          010506 005737 177060          TST      @#177060          ;;TIME OUT ON XOR?
2287          010512 012637 000004          MOV      (SP)+, @#ERRVEC      ;;RESTORE THE ERROR VECTOR
2288          010516 000463          BR      $SVLAD          ;;GO TO THE NEXT TEST
2289          010520 022626          5$: CMP      (SP)+, (SP)+      ;;CLEAR THE STACK AFTER A TIME OUT
2290          010522 012637 000004          MOV      (SP)+, @#ERRVEC      ;;RESTORE THE ERROR VECTOR
2291          010526 000423          BR      7$          ;;LOOP ON THE PRESENT TEST
2292          010530          6$: ;*****END OF CODE FOR THE XOR TESTER*****
2293          010530 032777 000400 170402          BIT      #BIT08, $SWR      ;;LOOP ON SPEC. TEST?
2294          010536 001404          BEQ      2$          ;;BR IF NO
2295          010540 127767 170374 170334          CMPB     $SWR, $STNM      ;;ON THE RIGHT TEST? SWR<7:0>
2296          010546 001462          BEQ      $OVER          ;;BR IF YES
2297          010550 105767 170327          2$: TSTB     $ERFLG          ;;HAS AN ERROR OCCURRED?
2298          010554 001421          BEQ      3$          ;;BR IF NO
2299          010556 126767 170333 170317          CMPB     $ERMAX, $ERFLG      ;;MAX. ERRORS FOR THIS TEST OCCURRED?
2300          010564 101015          BHI      3$          ;;BR IF NO
2301          010566 032777 001000 170344          BIT      #BIT09, $SWR      ;;LOOP ON ERROR?
2302          010574 001404          BEQ      4$          ;;BR IF NO
2303          010576 016767 170306 170302          7$: MOV      $LPERR, $LPADR      ;;SET LOOP ADDRESS TO LAST SCOPE
2304          010604 000443          BR      $OVER
2305          010606 105067 170271          4$: CLRB     $ERFLG          ;;ZERO THE ERROR FLAG
2306          010612 005067 170424          CLR      $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
2307          010616 000415          BR      1$          ;;ESCAPE TO THE NEXT TEST
2308          010620 032777 004000 170312          3$: BIT      #BIT11, $SWR      ;;INHIBIT ITERATIONS?
2309          010626 001011          BNE      1$          ;;BR IF YES
2310          010630 005767 170244          TST      $PASS          ;;IF FIRST PASS OF PROGRAM
2311          010634 001406          BEQ      1$          ;;INHIBIT ITERATIONS
2312          010636 005267 170242          INC      $ICNT          ;;INCREMENT ITERATION COUNT
2313          010642 026767 170374 170234          CMP      $TIMES, $ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
2314          010650 002021          BGE      $OVER          ;;BR IF MORE ITERATION REQUIRED
2315          010652 012767 000001 170224          1$: MOV      #1, $ICNT          ;;REINITIALIZE THE ITERATION COUNTER
2316          010660 016767 000044 170354          MOV      $MXCNT, $TIMES      ;;SET NUMBER OF ITERATIONS TO DO
2317          010666 105267 170210          $SVLAD: INCB     $STNM          ;;COUNT TEST NUMBERS
2318          010672 011667 170210          MOV      (SP), $LPADR      ;;SAVE SCOPE LOOP ADDRESS
2319          010676 011667 170206          MOV      (SP), $LPERR      ;;SAVE ERROR LOOP ADDRESS
2320          010702 005067 170336          CLR      $ESCAPE          ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
2321          010706 112767 000001 170201          MOV      #1, $ERMAX          ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
2322          010714 016777 170162 170220          $OVER: MOV      $STNM, @DISPLAY      ;;DISPLAY TEST NUMBER
2323          010722 016716 170160          MOV      $LPADR, (SP)      ;;FUDGE RETURN ADDRESS
2324          010730 000002          RTI          ;;FIXES PS
2325          010730 000100          $MXCNT: 100          ;;MAX. NUMBER OF ITERATIONS
2326          .SBTTL  ERROR HANDLER ROUTINE
2327
2328          ;*****
2329          ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.

```



```

2554 011650 012616          MOV      (SP)+,(SP)
2555 011652 000002          RTI              ;;RETURN TO USER
2556 011654 023420          $DTBL: 10000.
2557 011656 001750          1000.
2558 011660 000144          100.
2559 011662 000012          10.
2560 011654 000004          $DBLK: .BLKW 4
2561
2562          .SBTTL TTY INPUT ROUTINE
2563
2564          ;*****
2565          .ENABL LSB
2566
2567          .DSABL LSB
2568
2569
2570          ;*****
2571          *THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2572          *CALL:
2573          *
2574          *   RDCHR              ;; INPUT A SINGLE CHARACTER FROM THE TTY
2575          *   RETURN HERE    ;; CHARACTER IS ON THE STACK
2576          *                   ;; WITH PARITY BIT STRIPPED OFF
2577          *
2578
2578 011674 011646          $RDCHR: MOV      (SP),-(SP)      ;; PUSH DOWN THE PC
2579 011676 016666 000004 000002  MOV      4(SP),2(SP)        ;; SAVE THE PS
2580 011704 105777 167234 1$:  TSTB    0$TKS           ;; WAIT FOR
2581 011710 100375 1$      BPL      1$              ;; A CHARACTER
2582 011712 117766 167230 000004  MOVB    0$TKB,4(SP)        ;; READ THE TTY
2583 011720 042766 177600 000004  BIC     #1C(177),4(SP)    ;; GET RID OF JUNK IF ANY
2584 011726 026627 000004 000023  CMP     4(SP),#23        ;; IS IT A CONTROL-S?
2585 011734 001013 3$      BNE     3$              ;; BRANCH IF NO
2586 011736 105777 167202 2$:  TSTB    0$TKS           ;; WAIT FOR A CHARACTER
2587 011742 100375 2$      BPL      2$              ;; LOOP UNTIL ITS THERE
2588 011744 117746 167176  MOVB    0$TKB,-(SP)        ;; GET CHARACTER
2589 011750 042716 177600  BIC     #1C17,(SP)        ;; MAKE IT 7-BIT ASCII
2590 011754 022627 000021  CMP     (SP)+,#21        ;; IS IT A CONTROL-Q?
2591 011760 001366 2$      BNE     2$              ;; IF NOT DISCARD IT
2592 011762 000750 1$      BR      1$              ;; YES, RESUME
2593 011764 026627 000004 000140 3$:  CMP     4(SP),#140       ;; IS IT UPPER CASE?
2594 011772 002407 4$      BLT     4$              ;; BRANCH IF YES
2595 011774 026627 000004 000175  CMP     4(SP),#175       ;; IS IT A SPECIAL CHAR?
2596 012002 003003 4$      BGT     4$              ;; BRANCH IF YES
2597 012004 042766 000040 000004  BIC     #40,4(SP)        ;; MAKE IT UPPER CASE
2598 012012 000002 4$      RTI              ;; GO BACK TO USER
2599
2600          ;*****
2601          *THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2602          *CALL:
2603          *
2604          *   RDLIN              ;; INPUT A STRING FROM THE TTY
2605          *   RETURN HERE    ;; ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2606          *                   ;; TERMINATOR WILL BE A BYTE OF ALL 0'S
2607
2606 012014 010346          $RDLIN: MOV      R3,-(SP)      ;; SAVE R3
2607 012016 005046          CLR     -(SP)            ;; CLEAR THE RUBOUT KEY
2608 012020 012703 012250 1$:  MOV     #1TTYIN,R3        ;; GET ADDRESS
2609 012024 022703 012260 2$:  CMP     #1TTYIN+8.,R3     ;; BUFFER FULL?
  
```

```

2610 012030 101456      BLOS      4$      ;; BR IF YES
2611 012032 104406      R0CHR     ;; GO READ ONE CHARACTER FROM THE TTY
2612 012034      13      MOVVB     (SP)+,(R3)  ;; GET CHARACTER
2613 012036      13 000177 10$:  CMPB     #177,(R3)  ;; IS IT A RUBOUT
2614 012042      1022    BNE      5$      ;; BR IF NO
2615 012044 005716    TST      (SP)     ;; IS THIS THE FIRST RUBOUT
2616 012046 001007    SNE      6$      ;; BR IF NO
2617 012050 112767 000134 000170  MOVVB     #' \,9$   ;; TYPE A BACK SLASH
2618 012056 104401 012246    TYPE     ,9$
2619 012062 012716 177777    MOV      #-1,(SP)  ;; SET THE RUBOUT KEY
2620 012066 005303 6$:  DEC      R3      ;; BACKUP BY ONE
2621 012070 020327 012250    CMP      R3,#$TTYIN  ;; STACK EMPTY?
2622 012074 103434    BLO      4$      ;; BR IF YES
2623 012076 111367 000144  MOVVB     (R3),9$  ;; SETUP TO TYPEOUT THE DELETED CHAR.
2624 012102 104401 012246    TYPE     ,9$      ;; GO TYPE
2625 012106 000746    BR      2$      ;; GO READ ANOTHER CHAR.
2626 012110 005716 5$:  TST      (SP)     ;; RUBOUT KEY SET?
2627 012112 001406    BEQ      7$      ;; BR IF NO
2628 012114 112767 000134 000124  MOVVB     #' \,9$   ;; TYPE A BACK SLASH
2629 012122 104401 012246    TYPE     ,9$
2630 012126 005016    CLR      (SP)     ;; CLEAR THE RUBOUT KEY
2631 012130 122713 000025 7$:  CMPB     #25,(R3)  ;; IS CHARACTER A CTRL U?
2632 012134 001003    BNE      8$      ;; BR IF NO
2633 012136 104401 012260    TYPE     ,SCNTLU   ;; TYPE A CONTROL "U"
2634 012142 000726    BR      1$      ;; GO START OVER
2635 012144 122713 000022 8$:  CMPB     #22,(R3)  ;; IS CHARACTER A "R"?
2636 012150 001011    BNE      3$      ;; BRANCH IF NO
2637 012152 105013    CLRB     (R3)     ;; CLEAR THE CHARACTER
2638 012154 104401 001253    TYPE     ,SCLF    ;; TYPE A "CR" & "LF"
2639 012160 104401 012250    TYPE     ,TTYIN   ;; TYPE THE INPUT STRING
2640 012164 000717    BR      2$      ;; GO PICKUP ANOTHER CHARACTER
2641 012166 104401 001252 4$:  TYPE     ,SQUES   ;; TYPE A '?'
2642 012172 000712    BR      1$      ;; CLEAR THE BUFFER AND LOOP
2643 012174 111367 000046 3$:  MOVVB     (R3),9$  ;; ECHO THE CHARACTER
2644 012200 104401 012246    TYPE     ,9$
2645 012204 122723 000015    CMPB     #15,(R3)+  ;; CHECK FOR RETURN
2646 012210 001305    BNE      2$      ;; LOOP IF NOT RETURN
2647 012212 105063 177777    CLRB     -1(R3)   ;; CLEAR RETURN (THE 15)
2648 012216 104401 001254    TYPE     ,SLF     ;; TYPE A LINE FEED
2649 012222 005726    TST      (SP)+    ;; CLEAN RUBOUT KEY FROM THE STACK
2650 012224 012603    MOV      (SP)+,R3  ;; RESTORE R3
2651 012226 011646    MOV      (SP),-(SP)  ;; ADJUST THE STACK AND PLT ADDRESS OF THE
2652 012230 016666 000004 000002  MOV      4(SP),2(SP)  ;; FIRST ASCII CHARACTER ON IT
2653 012236 012766 012250 000004  MOV      #TTYIN,4(SP)
2654 012244 000002    RTI
2655 012246      000 9$:  .BYTE    0      ;; RETURN
2656 012247      000    .BYTE    0      ;; STORAGE FOR ASCII CHAR. TO TYPE
2657 012250 000010  STTYIN: .BLKB    8.    ;; TERMINATOR
2658 012260 052536 005015      000 SCNTLU: .ASCIZ  /TU<15><12>  ;; RESERVE 8 BYTES FOR TTY INPUT
2659 012265      136 006507 000012 SCNTLG: .ASCIZ  /TG<15><12>  ;; CONTROL "J"
2660 012272 005015 053523 020122 SMSWR:  .ASCIZ  <15><12>SWR = /  ;; CONTROL "G"
2661 012300 020075      000
2662 012303      040 047040 053505 SMNEW:  .ASCIZ  / NEW = /
2663 012303 036440 000040
2664
2665 .SBTTL READ AN OCTAL NUMBER FROM THE TTY

```

H05

2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697
2698
2699
2700
2701
2702
2703
2704
2705
2706
2707
2708
2709
2710
2711
2712
2713
2714
2715
2716
2717
2718
2719
2720
2721

012314 011646
012315 016666 000004 000002
012324 010046
012326 010146
012330 010246
012332 104407
012334 012600
012336 010057 000100
012342 005001
012344 005002
012346 112046
012350 001420
012352 122716 000060
012356 003026
012360 122716 000067
012364 002423
012366 006301
012370 006102
012372 006301
012374 006102
012376 006301
012400 006102
012402 042716 177770
012406 062601
012410 000756
012412 005726
012414 010166 000012
012420 010267 000026
012424 012602
012426 012601
012430 012600
012432 000002
012434 005726
012436 105010
012440 104401
012442 000000
012444 104401 001252
012450 000730
012452 000000

```
*****  
*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND  
*CHANGE IT TO BINARY.  
*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL  
*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED  
*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST  
*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.  
*CALL:  
*      RDOCT          ;; READ AN OCTAL NUMBER  
*      RETURN HERE   ;; LOW ORDER BITS ARE ON TOP OF THE STACK  
*                   ;; HIGH ORDER BITS ARE IN $HI OCT  
  
$RDOCT: MOV      (SP), -(SP)      ;; PROVIDE SPACE FOR THE  
        MOV      4(SP), 2(SP)    ;; INPUT NUMBER  
        MOV      R0, -(SP)       ;; PUSH R0 ON STACK  
        MOV      R1, -(SP)       ;; PUSH R1 ON STACK  
        MOV      R2, -(SP)       ;; PUSH R2 ON STACK  
  
1$:    RDLIN          ;; READ AN ASCII LINE  
        MOV      (SP)+, R0       ;; GET ADDRESS OF 1ST CHARACTER  
        MOV      R0, 5$         ;; AND SAVE IT  
        CLR      R1             ;; CLEAR DATA WORD  
        CLR      R2  
  
2$:    MOVB         (R0)+, -(SP)  ;; PICKUP THIS CHARACTER  
        BEQ      3$            ;; IF ZERO GET OUT  
        CMPB     #'0, (SP)      ;; MAKE SURE THIS CHARACTER  
        BGT      4$            ;; IS AN OCTAL DIGIT  
        CMPB     #'7, (SP)  
        BLT      4$  
        ASL     R1             ;; *2  
        ROL     R2  
        ASL     R1             ;; *4  
        ROL     R2  
        ASL     R1             ;; *8  
        ROL     R2  
  
3$:    BIC         #'107, (SP)    ;; STRIP THE ASCII JUNK  
        ADD     (SP)+, R1       ;; ADD IN THIS DIGIT  
        BR      2$            ;; LOOP  
        TST     (SP)+          ;; CLEAN TERMINATOR FROM STACK  
        MOV     R1, 12(SP)      ;; SAVE THE RESULT  
        MOV     R2, $HI OCT  
        MOV     (SP)+, R2      ;; POP STACK INTO R2  
        MOV     (SP)+, R1      ;; POP STACK INTO R1  
        MOV     (SP)+, R0      ;; POP STACK INTO R0  
        RTI                    ;; RETURN  
  
4$:    TST         (SP)+          ;; CLEAN PARTIAL FROM STACK  
        CLRB    (R0)           ;; SET A TERMINATOR  
        TYPE    TYPE           ;; TYPE UP THRU THE BAD CHAR.  
  
5$:    .WORD     0              ;; "?" "CR" & "LF"  
        TYPE    $QUES          ;; TRY AGAIN  
        BR      1$            ;; HIGH ORDER BITS GO HERE  
  
$HI OCT: .WORD    0  
  
.SBTTL  TYPE ROUTINE  
  
;*****
```

```

2722
2723
2724
2725
2726
2727
2728
2729
2730
2731
2732
2733
2734
2735
2736 012454 105767 166477 $TYPE: TSTB $TFPLG ;; IS THERE A TERMINAL?
2737 012460 100002 BPL 1$ ;; BR IF YES
2738 012462 000000 HALT ;; HALT HERE IF NO TERMINAL
2739 012464 000407 BR 3$ ;; LEAVE
2740 012466 010046 1$: MOV RO,-(SP) ;; SAVE RO
2741 012470 017600 000032 2$: MOV 22(SP),RO ;; GET ADDRESS OF ASCIZ STRING
2742 012474 112046 2$: MOVB (RO)+,-(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
2743 012476 001005 BNE 4$ ;; BR IF IT ISN'T THE TERMINATOR
2744 012500 005726 TST (SP)+ ;; IF TERMINATOR POP IT OFF THE STACK
2745 012502 012600 60$: MOV (SP)+,RO ;; RESTORE RO
2746 012504 062716 3$: ADD #2,(SP) ;; ADJUST RETURN PC
2747 012506 000002 RTI ;; RETURN
2748 012512 122716 4$: CMPB #HT,(SP) ;; BRANCH IF <HT>
2749 012516 001430 BEQ 8$
2750 012520 122716 000200 CMPB #CRLF,(SP) ;; BRANCH IF NOT <CRLF>
2751 012524 001006 BNE 5$
2752 012526 005726 TST (SP)+ ;; POP <CR><LF> EQUIV
2753 012530 104401 TYPE ;; TYPE A CR AND LF
2754 012532 001253 $CRLF
2755 012534 105067 000130 CLRB $CHARCNT ;; CLEAR CHARACTER COUNT
2756 012540 000755 BR 2$ ;; GET NEXT CHARACTER
2757 012542 004767 000056 5$: JSR PC,$TYPEC ;; GO TYPE THIS CHARACTER
2758 012546 126726 166404 6$: CMPB $FILLC,(SP)+ ;; IS IT TIME FOR FILLER CHARS.?
2759 012552 001350 BNE 3$ ;; IF NO GO GET NEXT CHAR.
2760 012554 016746 166374 MOV $NULL,-(SP) ;; GET # OF FILLER CHARS. NEEDED
2761 AND THE NULL CHAR.
2762 012560 105366 000031 7$: DECB 1(SP) ;; DOES A NULL NEED TO BE TYPED?
2763 012564 002770 BLT 6$ ;; BR IF NO--GO POP THE NULL OFF OF STACK
2764 012566 004767 000032 JSR PC,$TYPEC ;; GO TYPE A NULL
2765 012572 105367 000072 DECB $CHARCNT ;; DO NOT COUNT AS A COUNT
2766 012576 000770 BR 7$ ;; LOOP
2767
2768 ;HORIZONTAL TAB PROCESSOR
2769
2770 012600 112716 000040 8$: MOVB #' ,(SP) ;; REPLACE TAB WITH SPACE
2771 012604 004767 000014 9$: JSR PC,$TYPEC ;; TYPE A SPACE
2772 012610 132767 000007 000052 BITB #7,$CHARCNT ;; BRANCH IF NOT AT
2773 012616 001372 BNE 9$ ;; TAB STC?
2774 012620 005726 TST (SP)+ ;; POP SPACE OFF STACK
2775 012622 000724 BR 2$ ;; GET NEXT CHARACTER
2776 012624 105777 166320 $TYPEC: TSTB $STPS ;; WAIT UNTIL PRINTER IS READY
2777 012630 100375 BPL $TYPEC
  
```

```

2778 012632 116677 000002 166312      MOVB 2(SP),2$TPB  ;; LOAD CHAR TO BE TYPED INTO DATA REG.
2779 012640 122766 000015 000002      CMPB #CR,2(SP)   ;; IS CHARACTER A CARRIAGE RETURN?
2780 012646 001003                BNE 1$          ;; BRANCH IF NO
2791 012650 105067 000014                CLRB $CHARCNT   ;; YES--CLEAR CHARACTER COUNT
2782 012654 000406                BR $TYPEX       ;; EXIT
2783 012656 122766 000012 000002 1$:  CMPB #LF,2(SP)  ;; IS CHARACTER A LINE FEED?
2784 012664 001402                BEQ $TYPEX      ;; BRANCH IF YES
2795 012666 105227                INCB (PC)+      ;; COUNT THE CHARACTER
2786 012670 000000      $CHARCNT: .WORD 0 ;; CHARACTER COUNT STORAGE
2797 012672 000207      $TYPEX: RTS    PC

```

.SBTTL READ A DECIMAL NUMBER FROM THE TTY

```

*****
*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
*POSITIVE 32767 TO NEGATIVE 32768.

```

```

*CALL:
*      RODEC                ;; READ A DECIMAL NUMBER
*      RETURN HERE         ;; NUMBER IS ON TOP OF THE STACK

```

```

2804 012674 011646 000004 000002  $RODEC: MOV (SP),-(SP)  ;; PROVIDE SPACE FOR
2805 012676 016666                MOV 4(SP),2(SP)  ;; THE INPUT NUMBER
2806 012704 010046                MOV R0,-(SP)     ;; PUSH R0 ON STACK
2807 012706 010146                MOV R1,-(SP)     ;; PUSH R1 ON STACK
2808 012710 010246                MOV R2,-(SP)     ;; PUSH R2 ON STACK
2809 012712 104407 1$:  RDLIN  ;; READ AN ASCII LINE
2810 012714 012600                MOV (SP)+,R0    ;; ADDRESS OF 1ST CHAR.
2811 012716 010067 000120                MOV R0,6$      ;; SAVE IN CASE OF BAD INPUT
2812 012722 005046                CLR -(SP)      ;; CLEAR DATA WORD
2813 012724 005002                CLR R2         ;; SIGN SET POSITIVE
2814 012726 122710 000055                CMPB #'-(R0)   ;; SEE IF A MINUS SIGN WAS TYPED
2815 012732 001001                BNE 2$        ;; BR IF NO MINUS SIGN
2816 012734 112002                MOVB (R0)+,R2  ;; SAVE FOR LATER USE
2817 012736 112001 2$:  MOVB (R0)+,R1   ;; PICKUP THIS CHARACTER
2818 012740 001424                BEQ 3$        ;; GET OUT IF ZERO
2819 012742 122701 000050                CMPB #'0,R1   ;; MAKE SURE THIS CHARACTER
2820 012746 003032                BGT 5$        ;; IS A DIGIT BETWEEN 0 & 9
2821 012750 122701 000071                CMPB #'9,R1
2822 012754 002427                BLT 5$
2823 012756 032716 170000                BIT #'C7777,(SP) ;; DON'T LET NUMBER GET TO BIG
2824 012762 001024                BNE 5$        ;; BR IF NUMBER WOULD OVERFLOW
2825 012764 006316                ASL (SP)      ;; *2
2826 012766 011646                MOV (SP),-(SP) ;; SAVE FOR LATER
2827 012770 006316                ASL (SP)      ;; *4
2828 012772 006316                ASL (SP)      ;; *8
2829 012774 062616                ADD (SP)+,(SP) ;; *10
2830 012776 102416                BVS 5$        ;; OVERFLOW ISN'T ALLOWED
2831 013000 162701 000060                SUB #'0,R1    ;; STRIP AWAY THE ASCII JUNK
2832 013004 060116                ADD R1,(SP)   ;; ADD IN THIS DIGIT
2833 013006 102412                BVS 5$        ;; OVERFLOW ISN'T ALLOWED

```

```

2834 013010 000752          BR      2$          ;; LOOP
2835 013012 005702      3$:  TST      R2          ;; CHECK IF NUMBER IS NEG
2836 013014 001401          BEQ      4$          ;; BR IF NO
2837 013016 005416          NEG      (SP)          ;; YES--NEGATE THE NUMBER
2838 013020 012666 000012      4$:  MOV      (SP)+,12(SP)  ;; SAVE THE RESULT
2839 013024 012602          MOV      (SP)+,R2      ;; POP STACK INTO R2
2840 013026 012601          MOV      (SP)+,R1      ;; POP STACK INTO R1
2841 013030 012600          MOV      (SP)+,R0      ;; POP STACK INTO R0
2842 013032 000002          RTI          ;; RETURN
2843
2844 013034 005726      5$:  TST      (SP)+          ;; CLEAN PARTIAL NUMBER FROM STACK
2845 013036 105010          CLAB     (R0)          ;; SET A TERMINATOR
2846 013040 104001          TYPE          ;; TYPE THE INPUT UP TO BAD CHAR.
2847 013042 000000      6$:  .WORD     0          ;; POINTER GOES HERE
2848 013044 104401 001252          TYPE     $QUES       ;; "?" "CR" &"LF"
2849 013050 000720          BR      1$          ;; TRY AGAIN
2850
2851          .SBTTL  TRAP DECODER
2852
2853          ;;*****
2854          ;;THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2855          ;;AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2856          ;;OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2857          ;;GO TO THAT ROUTINE.
2858
2859 013052 010046      $TRAP:  MOV      RO,-(SP)          ;; SAVE RO
2860 013054 016600 000002          MOV      2(SP),R0      ;; GET TRAP ADDRESS
2861 013060 005740          TST      -(R0)          ;; BACKUP BY 2
2862 013062 111000          MOVB    (R0),R0        ;; GET RIGHT BYTE OF TRAP
2863 013064 006500          ASL     R0              ;; POSITION FOR INDEXING
2864 013066 016000 013106          MOV      $TRPAD(R0),R0  ;; INDEX TO TABLE
2865 013072 000200          RTS      R0           ;; GO TO ROUTINE
2866
2867
2868          ;;THIS IS USE TO HANDLE THE "GETPR!" MACRO
2869
2870 013074 011646      $TRAP2:  MOV      (SP),-(SP)          ;; MOVE THE PC DOWN
2871 013076 016666 000004 000002          MOV      4(SP),2(SP)  ;; MOVE THE PSW DOWN
2872 013104 000002          RTI          ;; RESTORE THE PSW
2873
2874          .SBTTL  TRAP TABLE
2875
2876          ;;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2877          ;;BY THE "TRAP" INSTRUCTION.
2878
2879          ;          ROUTINE
2880          ;          -----
2881 013106 013074      $TRPAD:  .WORD     $TRAP2          TRAP+1(104401)  TTY TYPEOUT ROUTINE
2882 013110 012454          $TYPE          ;;CALL=TYPE          TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2883 013112 011246          $TYPOC         ;;CALL=TYPOC         TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2884 013114 011222          $TYPOS         ;;CALL=TYPOS         TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2885 013116 011262          $TYPON         ;;CALL=TYPON         TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
2886 013120 011450          $TYPDS         ;;CALL=TYPDS
2887
2888
2889 013122 011674          $RDCHR        ;;CALL=RDCHR        TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE

```

2890	013124	012014			\$RDLIN	::CALL=RDLIN	TRAP+7(104407)	TTY TYPEIN STRING ROUTINE
2891	013126	012314			\$RDOCT	::CALL=RDOCT	TRAP+10(104410)	READ AN OCTAL NUMBER FROM TTY
2892	013130	012674			\$RDECC	::CALL=RDECC	TRAP+11(104411)	READ A DECIMAL NUMBER FROM TTY

.SBTTL POWER DOWN AND UP ROUTINES

```

*****
:POWER DOWN ROUTINE
$PWRDN: MOV    $SILLUP, @#PWRVEC    ;; SET FOR FAST UP
        MOV    #340, @#PWRVEC+2    ;; PRIO:7
        MOV    RO, -(SP)           ;; PUSH RO ON STACK
        MOV    R1, -(SP)           ;; PUSH R1 ON STACK
        MOV    R2, -(SP)           ;; PUSH R2 ON STACK
        MOV    R3, -(SP)           ;; PUSH R3 ON STACK
        MOV    R4, -(SP)           ;; PUSH R4 ON STACK
        MOV    R5, -(SP)           ;; PUSH R5 ON STACK
        MOV    @SWR, -(SP)         ;; PUSH @SWR ON STACK
        MOV    SP, $SAVR6         ;; SAVE SP
        MOV    #PWRUP, @#PWRVEC    ;; SET UP VECTOR
        HALT
        BR     -2                  ;; HANG UP

```

```

*****
:POWER UP ROUTINE
$PWRUP: MOV    $SILLUP, @#PWRVEC    ;; SET FOR FAST DOWN
        MOV    $SAVR6, SP         ;; GET SP
        CLR    $SAVR6            ;; WAIT LOOP FOR THE TTY
1$:     INC    $SAVR6             ;; WAIT FOR THE INC
        BNE    1$                 ;; OF WORD
        MOV    (SP)+, @SWR        ;; POP STACK INTO @SWR
        MOV    (SP)+, R5         ;; POP STACK INTO R5
        MOV    (SP)+, R4         ;; POP STACK INTO R4
        MOV    (SP)+, R3         ;; POP STACK INTO R3
        MOV    (SP)+, R2         ;; POP STACK INTO R2
        MOV    (SP)+, R1         ;; POP STACK INTO R1
        MOV    (SP)+, RO         ;; POP STACK INTO RO
        MOV    #PWRDN, @#PWRVEC    ;; SET UP THE POWER DOWN VECTOR
        MOV    #340, @#PWRVEC+2    ;; PRIO:7
        TYPE   $POWER             ;; REPORT THE POWER FAILURE
$PWRMG: .WORD  $POWER            ;; POWER FAIL MESSAGE POINTER
        MOV    (PC)+, (SP)        ;; RESTART AT RESTR
$PWRPD: .WORD  RESTR             ;; RESTART ADDRESS
        RTI
$SILLUP: HALT                    ;; THE POWER UP SEQUENCE WAS STARTED
        BR     -2                  ;; BEFORE THE POWER DOWN WAS COMPLETE
$SAVR6: 0                          ;; PUT THE SP HERE
$POWER:  .ASCIZ <15><12>"POWER"

```

.EVEN

```

*****
:TRANSMIT INTERRUPT SERVICE ROUTINE FOR 256. BYTE BLOCK TRANSFERS
*****

```

2944	013314	105777	166074		XINT:	TSTB	@DLXCSR	;"READY" SET "
2945	013320	100416				BMI	1\$;;BR IF YES


```

2946 013322 013767 177776 165652      MOV      Q#PSW,$TMPD      ;SAVE THE ERROR PSW
2947 013330 010667 165642      MOV      SP,$REG6        ;SAVE THE ERROR STACK POINTER
2948 013334 005167 166062      COM      XFLGO           ;SET XMIT SOFTWARE ERROR FLAG
2949 013340 042777 000100 166042      BIC      #100,QDLRCSR    ;TURN OFF THE INTERRUPT ENABLES
2950 013346 042777 000100 166040      BIC      #100,QDLXCSR
2951 013354 000411                BR      2$              ;GO TO EXIT
2952 013356 022767 021722 166046 1$:      CMP      #DLBUFI,OPTR    ;XMITTED 256. BYTES YET ??
2953 013364 001405                BEQ      2$              ;BR IF YES
2954 013366 117777 166040 166022      MOVB    QOPTR,QDLXDBR    ;OUTPUT A BYTE
2955 013374 005267 166032                INC      OPTR            ;UPDATE BUFFER POINTER
2956 013400 000002                2$:      RTI                  ;RETURN TO MAINLINE TEST
2957
2958 ;*****
2959 ;RECEIVER INTERRUPT SERVICE ROUTINE FOR 256. BYTE BLOCK TRANSFERS
2960 ;*****
2961
2962 013402 105777 166002      RINT:   TSTB    QDLRCSR ;"DONE" SET ??
2963 013406 100410                BMI     1$              ;BR IF YES
2964 013410 013767 177776 165564      MOV      Q#PSW,$TMPD    ;SAVE THE ERROR PSW
2965 013416 010667 165554      MOV      SP,$REG6        ;SAVR THE ERROR STACK POINTER
2966 013422 005167 165776      COM      RFLGO           ;SET HARD RCVR ERROR FLAG
2967 013426 000415                BR      2$              ;GO EXIT
2968 013430 005777 165756      1$:     TST      QDLRDBR    ;ANY SOFT ERRORS ??
2969 013434 100021                BPL     3$              ;BR IF NOT
2970 013436 013767 177776 165536      MOV      Q#PSW,$TMPD    ;SAVE THE ERROR PSW
2971 013444 010667 165526      MOV      SP,$REG6        ;SAVE THE ERROR STACK POINTER
2972 013450 017767 165736 165526      MOV      QDLRDBR,$TMP1   ;SAVE THE ERROR REGISTER IN TMP1
2973 013456 005167 165744      COM      RFLG1           ;SET THE SOFT ERROR FLAG
2974 013462 042777 000100 165724 2$:     BIC      #100,QDLXCSR    ;TURN OFF THE INTR. ENABLES
2975 013470 042777 000100 165712      BIC      #100,QDLRCSR
2976 013476 000411                BR      4$              ;GO TO EXIT
2977 013500 022767 022322 165726 3$:     CMP      #BUFEND,IPTR    ;RECEIVED 256. BYTES YET ??
2978 013506 001405                BEQ     4$              ;BR IF YES
2979 013510 117777 165676 165716      MOVB    QDLRDBR,QIPTR    ;INPUT A BYTE FROM THE DL11
2980 013516 005267 165712                INC     IPTR            ;UPDATE BUFFER POINTER
2981 013522 000002                4$:     RTI                  ;RETURN TO MAINLINE TEST
2982
2983 ;THE FOLLOWING ROUTINE IS USED BY THE USER UTILITY PROGRAMS TO WAIT
2984 ;A SPECIFIED NO. OF MILLISECONDS BETWEEN CHARACTER TRANSFERS
2985
2986 013524 017667 000000 000034 DELAY:  MOV      Q(R6),DELCNT ;GET THE NO. OF MSEC. DELAY COUNT
2987                                ;TYPED IN BY USER
2988 013532 062716 000002                ADD     #2,(R6)         ;SET UP THIS ROUTINE'S EXIT ADDRESS
2989 013536 005767 000024                TST    DELCNT           ;IS THE DELAY COUNT ZERO?
2990 013542 001410                BEQ     3$              ;BRANCH IF YES
2991 013544 012746 000226      1$:     MOV      #226,-(SP)    ;PUSH A 1 MSEC. COUNT TO STACK
2992 013550 005316                2$:     DEC     (SP)         ;DECREMENT THE 1 MSEC. COUNT BY 1
2993 013552 001376                BNE     2$              ;BRANCH IF 1 MSEC. NOT EATEN
2994                                ;AWAY YET
2995 013554 005726                TST    (SP)+           ;RESET STACK AFTER 1 MSEC. TIME UP
2996 013556 005367 000004                DEC     DELCNT          ;DECREMENT THE TOT'L NO. OF
2997                                ;MSEC. COUNT
2998 013562 001370                BNE     1$              ;BRANCH IF WE HAVE MORE MSEC.
2999                                ;TO WAIT
3000 013564 000207                3$:     RTS      PC        ;GO BACK TO REISSUE A CHARACTER
3001 013566 000000 DELCNT: .WORD 0          ;THE NO. OF MSEC. NEEDED TO

```

```

3002                                     ; TRANSPIRE RESIDES HERE
3003                                     ; THE FOLLOWING ROUTINE IS USED BY USER PROGRAM #4 AND WILL ALLOW
3004                                     ; A RANDOM NUMBER OF MILLISECONDS BEFORE TRANSMISSION OF CHARACTER
3005
3006 013570 016700 000062          STALL: MOV     NUMONE,RO      ; GET THE LOW LIMIT
3007 013574 006100                ROL     RO              ; MULTIPLY BY 4
3008 013576 006100                ROL     RO
3009 013600 066700 000054          ADD     NUMTWO,RO      ; ADD IN THE HIGH LIMIT
3010 013604 010067 000046          MOV     RO,NUMONE     ; STORE THIS AS NEW LOW LIMIT
3011 013610 006100                ROL     RO              ; MULTIPLY NEW LOW LIMIT BY 4
3012 013612 006100                ROL     RO
3013 013614 066700 000040          ADD     NUMTWO,RO      ; ADD IN THE HIGH LIMIT
3014 013620 006100                ROL     RO              ; MULTIPLY BY 4 AGAIN
3015 013622 006100                ROL     RO
3016 013624 010067 000030          MOV     RO,NUMTWO     ; STORE THIS AS NEW HIGH LIMIT
3017 013630 016700 000022          MOV     NUMCNE,RO     ; SAVE THE RANDOMLY GENERATED NO.
3018 013634 046700 165446          BIC     STLMASK,RO    ; STRIP ALL BUT 1ST 5 BITS SO AS
3019                                     ; NOT TO ALLOW THE STALL TO BE TOO
3020                                     ; LARGE
3021 013640 001405                BEQ     2$              ; BRANCH IF RESULT WAS ZERO
3022 013642 010067 000004          MOV     RO,1$         ; SET STALL TIME FOR DELAY ROUTINE
3023 013646 004767 177652          JSR     PC,DELAY      ; GO OFF TO STALL
3024 013652 000000          1$: .WORD 0           ; THIS IS WHERE STALL TIME RESIDES
3025 013654 000207          2$: RTS     PC        ; RETURN TO ISSUE CHARACTER
3026 013656 001233          NUMONE: 1233         ; LOW LIMIT FOR RANDOM NO.
3027 013660 007622          NUMTWO: 7622        ; HIGH LIMIT FOR RANDOM NO.
3028                                     ; THE FOLLOWING ROUTINE CHECKS THE 'DONE' BIT FOR BOTH THE RECEIVER
3029                                     ; AND TRANSMITTER. THIS ROUTINE IS USED BY PROGRAM #4
3030
3031 013662 016767 165322 000044  TIMERX: MOV     $TMP3,DUT    ; GET THE TRANSMITTER CONTROL
3032                                     ; STATUS REGISTER ADDRESS
3033 013670 162767 000004 000036          SUB     #4,DUT        ; FORM THE RECEIVER CONTROL
3034                                     ; STATUS REGISTER ADDRESS
3035 013676 000403                BR      TCONT         ; GO TO TIME OUT THE RECEIVERS'
3036                                     ; DONE BIT
3037 013700 016767 165304 000026  TIMETX: MOV     $TMP3,DUT    ; GET THE TRANSMITTER CONTROL
3038                                     ; STATUS REGISTER ADDRESS
3039 013706 005067 165304          TCONT: CLR     $TMP6   ; INITIALIZE A TIME COUNT
3040 013712 005267 165300          1$: INC     $TMP6     ; INCREMENT THE TIME COUNT
3041 013716 001405                BEQ     2$              ; BRANCH IF TIME COUNTER OVERFLOWED
3042                                     ; INDICATING DONE BIT NEVER SET
3043                                     ; WITH PLENTY OF TIME ELAPSED
3044 013720 105777 000010                TSTB   @DUT           ; SEE IF DONE BIT IS SET YET
3045 013724 100372                BPL     1$             ; WAIT SOME MORE IF IT ISN'T
3046 013726 062716 000006          ADD     #6,@R6        ; DONE BIT IS SET - SET UP EXIT
3047                                     ; RETURN TO SKIP ERROR REPORT
3048 013732 000207          2$: RTS     PC        ; RETURN TO PROGRAM #4
3049 013734 000000          DUT: .WORD 0         ; THIS IS WHERE THE RCSR OR XCSR
3050                                     ; ADDRESS RESIDES
3051                                     ; THIS ROUTINE IS USED BY PROGRAMS #4 & 5, AND WILL CHECK FOR CORRECT
3052                                     ; EXPECTED AND RECEIVED DATA, IN ADDITION TO ANY ERROR BITS
3053
3054 013736 016767 165252 165252  DATCHK: MOV     $TMP5,$TMP6 ; GET THE CONTENTS OF THE RECEIVER
3055                                     ; BUFFER
3056 013744 016767 165242 165214          MOV     $TMP4,$REG2  ; STORE THE ADDRESS OF THE RECEIVER
3057                                     ; DATA BUFFER

```



```

165135 SLERR2: MOV    20PSW,STMPD    :SAVE THE (PSW)
                MOV    DLRCR,R1    :PUT DEVAOR IN R1
                MOV    R2,R3    :PUT WAS INFO IN R3
                MOV    SP,SREG6    :SAVE THE (SP)
                ADD    R2,SREG6    :CORRECT FOR CALLING JSR
SLERR1: MOVB   $1,STNM,R0    :PUT TEST NO. IN R0
                MOV    R0,SREG0    :SAVE (R0) THRU (R4)
                MOV    R1,SREG1
                MOV    R2,SREG2
                MOV    R3,SREG3
                MOV    R4,SREG4
                RTS    PC    :RETURN TO CALLING TEST

165056 SLERR1: MOV    20PSW,STMPD    :SAVE THE (PSW)
                MOVB   $1,STNM,R0    :PUT TEST NO. IN R0
                MOV    DLRCR,R1    :PUT DEVAOR IN R1
                MOV    R0,SREG0    :SAVE (R0)
                MOV    R1,SREG1    :SAVE (R1)
165032 SLERR2: MOV    20PSW,STMPD    :SAVE THE (PSW)
                MOV    SP,SREG6    :SAVE THE (SP)
165012      ADD    R2,SREG6    :CORRECT FOR CALLING JSR
                MOV    R2,SREG2    :SAVE (R2)
                RTS    PC    :RETURN

:SUBROUTINE TO SETUP VECTORS FOR 256. BYTE BLOCK TRANSFER TESTS
SLVEC:  MOV    DLVECT,R5    :SET FIRST VECTOR ADDRESS
                MOV    RINT,(R5)+  :SET UP RCVR VECTOR
                MOV    DLPRI,(R5)+
                MOV    XINT,(R5)+  :SET UP XMIT VECTOR
                MOV    DLPRI,(R5)
                RTS    PC    :RETURN TO CALLER

:SUBROUTINE TO PRIME DATA BUFFERS AND DEVICE FOR 256. BYTE TRANSFER
PRIME:  CLR    2DLXCSR    :CLEAR XMIT AND RCVR CSR'S
                CLR    2DLRCR
                CLR    XFLG0    :INITIALIZE ERROR FLAGS
                CLR    RFLG0
                CLR    RFLG1
165160      MOV    2DLBUFO,OPTR    :SET UP OUTPUT POINTER
165154      MOV    2DLBUFI,IPTR    :SET UP INPLT POINTER
                JSR    PC,CLDLBF    :GO CLEAR THE BUFFERS
                JSR    PC,CLDCOUT    :GO SET UP THE PATTERN
                CLR    TMR1    :INIT "IMEOU" COUNTERS
165140      MOV    30,TMR2
                TST    2DLR09R    :FLUSH "DCNE" BIT IN RCVR CSR
                TST    2DLR0BR
165070      BIS    100,2DLRCR    :ENABLE RCVR INTR.
165066      BIS    104,2DLXCSR    :ENABLE XMIT INTR. AND MAINT MODE
                RTS    PC

```

:THIS ROUTINE IS CALLED TO CLEAR THE INPUT AND OUTPUT BUFFERS

```

014330 012705 021322 0LDLBF: MOV #DLBUF0,R5 :R5 POINTS TO BEGINNING OF BUFFER AREA
014332 005025 1S: CLR (R5)+ :CLEAR A WORD
014334 022705 022322 :CMP #BLFEND,R5 :DONE ALL WORDS ??
014336 001375 :BNE 1S :BR IF NOT
014338 000207 :RTS PC :RETURN TO CALLER

:THIS ROUTINE IS CALLED TO SET UP THE NULL-DEL-NUL PATTERN

014344 012705 021322 0LDOUT1: MOV #DLBUF0,R5 :R5 POINTS TO OUTPUT BUFFER
014346 105025 1S: CLRB (R5)+ :MOVE A NULL CHAR
014348 112725 000377 :MOVB #377,(R5)+ :MOV A DEL CHAR
014350 022705 021722 :CMP #DLBUF1,R5 :ALL DONE ??
014352 001375 :BNE 1S :BR IF NOT
014354 000207 :RTS PC :RETURN TO CALLER

:THIS ROUTINE IS USED TO LOAD AN ASCENDING BINARY COUNT PATTERN

014370 005005 021322 0LDOUT2: CLR R5 :START WITH 000
014372 110565 1S: MOVB R5,DLBUF0(R5) :LOAD ONE BYTE
014374 005205 :INC R5 :INCREMENT BYTE
014376 022705 000400 :CMP #400,R5 :DONE 000 THRU 377 ??
014378 001375 :BNE 1S :BR IF NOT
014380 000207 :RTS PC :RETURN TO CALLER

:THIS ROUTINE IS USED TO LOAD A DESCENDING BINARY COUNT PATTERN

014410 112767 000377 164602 0LDOUT3: MOVB #377,$STMP7 :START WITH A 377 BYTE
014412 012705 021322 1S: MOV #DLBUF0,R5 :R5 POINTS TO OUTPUT BUFFER
014414 116725 164572 :MOVB $STMP7,(R5)+ :LOAD ONE BYTE
014416 022705 021722 :CMP #DLBUF1,R5 :ALL DONE ??
014418 001403 :BEQ 2S :BR IF YES
014420 105367 114560 :DECB $STMP7 :GENERATE NEXT BYTE
014422 000770 :BR 1S :GO MOVE IT
014424 000207 :RTS PC :RETURN TO CALLER

:THIS ROUTINE LOADS A COMPLEMENTING WORST CASE PATTERN

014444 012705 021322 0LDOUT4: MOV #DLBUF0,R5 :R5 POINTS TO OUTPUT BUFFER
014446 005067 164544 :CLR $STMP7 :INIT. BYTE GENERATOR
014448 116725 164540 1S: MOVB $STMP7,(R5)+ :MOVE A BYTE
014450 105167 164534 :COMB $STMP7 :COMPLEMENT IT
014452 116725 164530 :MOVB $STMP7,(R5)+ :NOW LOAD THE 1'S COMPLEMENT
014454 105267 164525 :INCB $STMP7+1 :INCREMENT THE BYTE
014456 116767 164521 164516 :MOVB $STMP7+1,$STMP7 :SET UP TO LOAD NEXT TWO
014458 022705 021722 :CMP #DLBUF1,R5 :ALL DONE ??
014460 001362 :BNE 1S :BR IF NOT
014462 000207 :RTS PC :RETURN TO CALLER

:THIS ROUTINE CHECKS FOR DATA COMPARE ERRORS IN 256. BYTE BLOCK TRANSFERS

014510 042777 000104 164674 0CHKDAT: BIC #104,$DLXCSR :DISABLE BOTH XMIT AND RCVR INTR. ENAB.
014512 042777 000100 164662 :BIC #100,$DLRCSR
014514 012702 021322 :MOV #DLBUF0,R2 :R2 POINTS TO S/B DATA IN OUTPUT BUFFER
014516 004767 000070 :JSR PC,MASKING :GO TO MASK OFF BITS AS A FUNCTION OF
    
```

```

3194
3195 014536 012701 021722          MOV    #DLBUF1,R1      ; CHARACTER LENGTH'S 5, 6, 7, OR 8 BITS,
3196 014542 122221          1$:  CMPB   (R2)+,(R1)+ ; R1 POINTS TO WAS DATA IN RCVR. BUFFER
3197 014544 001004          BNE    3$             ; DID S/B = WAS ??
3198 014546 022701 022322          2$:  CMP    #BUFEND,R1 ; BR IF NOT
3199 014550 001373          BNE    1$             ; CHECK'D ALL BYTES ??
3200 014554 000207          RTS    PC             ; BR IF NOT
3201 014556 013767 177776 164416 3$:  MOV    #PSW,$TMP0    ; RETURN TO CALLER
3202 014564 010667 164406          MOV    SP,$REG6      ; SAVE THE [PSW]
3203 014570 114204          MOVB   -(R2),R4      ; SAVE THE [SP]
3204 014572 042704 177400          BIC    #177400,R4    ; GET THE S/B DATA
3205 014576 114103          MOVB   -(R1),R3      ; CLEAR JUNK FROM HI BYTE
3206 014600 042703 177400          BIC    #177400,R3    ; GET THE WAS DATA
3207 014604 004767 177254          JSR    PC,$UERR1     ; CLEAR JUNK FROM HI BYTE
3208 014610 012767 014620 164426          MOV    #4,$$ESCAPE  ; GO SET UP ERROR INFO.
3209 014616 104003          ERROR+3              ; RETURN TO 4$ AFTER ERROR PRINT
3210 014620 005202          4$:  INC    R2           ; DATA COMPARE ERROR
3211 014622 005201          INC    R1           ; REPOSITION BUFFER POINTERS
3212 014624 000750          BR     2$           ; GO CHECK NEXT BYTE

```

```

; THIS ROUTINE IS USED BY THE PATTERN TESTS
; IT WILL MASK OFF THE CHARACTER SENT OUT BY THE XMITTER
; BEFORE THE COMPARISON OF DATA OF WHAT WAS RECEIVED AND WHAT WAS TRANSMITTED
; IS DONE. THE MASKING IS DONE AS A FUNCTION OF CHARACTER LENGTH WHICH
; CAN BE EITHER 5, 6, 7, OR 8 BITS .

```

```

3220 014626 005005          MASKING:  CLR    R5          ; INITIALIZE TABLE OFFSET
3221
3222 014630 022767 000010 164376          CMP    #8,$TMP15     ; FOR PICKING UP MASK WORD
3223 014636 001427          BEQ    3$             ; IS THE CHARACTER LENGTH 8 BITS?
3224 014640 062705 000002          ADD    #2,R5         ; BRANCH IF IT IS
3225
3226 014644 022767 000007 164352          CMP    #7,$TMP15     ; SET UP FOR NEXT MASK WORD
3227 014652 001410          BEQ    1$             ; IT COULD BE THIS ONE
3228 014654 062705 000002          ADD    #2,R5         ; IS THE CHARACTER LENGTH 7 BITS?
3229
3230 014660 022767 000006 164346          CMP    #6,$TMP15     ; BRANCH IF IT IS
3231 014666 001402          BEQ    1$             ; SET UP FOR NEXT MASK WORD
3232 014670 062705 000002          ADD    #2,R5         ; IT COULD BE THIS ONE
3233
3234 014674 016505 014720          1$:  MOV    CHARL(R5),R5 ; IS THE CHARACTER LENGTH 6 BITS?
3235 014700 005105          COM    R5             ; BRANCH IF IT IS
3236 014702 140522          2$:  BICB   R5,(R2)+     ; FORM THE BITS THAT ARE TO BE MASKED
3237 014704 022702 021722          CMP    #DLBUF1,R2   ; MASK A BYTE
3238
3239 014710 001374          BNE    2$             ; ARE WE AT THE END OF THE XMITTER
3240 014712 012702 021322          MOV    #DLBUF0,R2   ; OUTPUT BUFFER
3241 014716 000207          3$:  RTS    PC             ; BRANCH IF NO TO MAS, NEXT BYTE
3242
; TABLE OF MASK WORDS
; RESTORE R2 BEFORE RETURNING
; RETURN TO MAINLINE CODE
CHARL: .WORD 377 ; 8. BITS IN LENGTH
        .WORD 177 ; 7. BITS IN LENGTH
        .WORD 77  ; 6. BITS IN LENGTH
        .WORD 37  ; 5. BITS IN LENGTH

```

```

; THIS ROUTINE IS USED BY PROGRAMS #4 & ;
; IT WILL MASK OFF THE CHARACTER SENT OUT BY THE TRANSMITTER

```

```

3250 ;BEFORE THE COMPARISON OF DATA OF WHAT WAS RECEIVED AND WHAT WAS
3251 ;TRANSMITTED IS DONE. THE MASKING IS DONE AS A FUNCTION OF CHARACTER
3252 ;LENGTH WHICH CAN BE EITHER 5, 6, 7, OR 8 BITS.
3253
3254 014730 016767 164250 164274 UPMASK: MOV $TMP1,$TMP14 ;PICK UP THE CHARACTER THAT WAS
3255 ;SENT OUT FROM THE XMITTER
3256 014736 005005 CLR R5 ;INITIALIZE TABLE OFFSET
3257 ;FOR PICKING UP MASK WORD
3258 014740 022767 000010 164266 CMP #8,$TMP15 ;IS THE CHARACTER LENGTH 8 BITS?
3259 014746 001423 BEQ 2$ ;BRANCH IF IT IS
3260 014750 062705 000002 ADD #2,R5 ;SET UP FOR NEXT MASK WORD
3261 ;IT COULD BE THIS ONE
3262 014754 022767 000007 164252 CMP #7,$TMP15 ;IS THE CHARACTER LENGTH 7 BITS?
3263 014762 001410 BEQ 1$ ;BRANCH IF IT IS
3264 014764 062705 000002 ADD #2,R5 ;SET UP FOR NEXT MASK WORD
3265 ;IT COULD BE THIS ONE
3266 014770 022767 000006 164236 CMP #6,$TMP15 ;IS THE CHARACTER LENGTH 6 BITS?
3267 014776 001402 BEQ 1$ ;BRANCH IF IT IS
3268 015000 062705 000002 ADD #2,R5 ;SET UP FOR NEXT MASK WORD
3269 ;IT MUST BE THIS ONE!!!!
3270 015004 016505 014720 1$: MOV CHARL(R5),R5 ;PICK UP THE MASK WORD
3271 015010 005105 COM R5 ;FORM THE BITS THAT ARE TO BE MASKED
3272 015012 140567 164214 BICB R5,$TMP14 ;MASK THE LOW BYTE
3273 015016 000207 RTS PC ;RETURN TO MAINLINE CODE
3274
3275 ;ROUTINE TO SERVICE BUS ERROR TRAPS
3276
3277 015020 112767 000060 000632 BUSERR: MOVB #60,EM4+46 ;SET UP ERROR MESSAGE
3278 015026 112767 000060 000625 MOVB #60,EM4+47
3279 015034 112767 000064 000620 MOVB #64,EM4+50
3280 015042 000412 BR TRPCOM ;GO SET UP AND REPORT BUS ERROR
3281
3282 ;ROUTINE TO SERVICE RSVD INSTRUCTION TRAPS
3283
3284 015044 112767 000060 000606 RSVERR: MOVB #60,EM4+46 ;SET UP ERROR MESSAGE
3285 015052 112767 000061 000601 MOVB #61,EM4+47
3286 015060 112767 000060 000574 MOVB #60,EM4+50
3287 015066 000400 BR TRPCOM ;GO SET UP AND REPORT RSVD INSTR. ERROR
3288
3289 ;ROUTINE TO SET UP AND REPORT BUS ERROR AND RSVD INSTR ERRORS
3290
3291 015070 010667 164102 TRPCOM: MOV SP,$REG6 ;SAVE THE TRAP SP
3292 015074 116700 164002 MOVB $STNM,R0 ;PUT TEST NO. IN R0
3293 015100 010067 164056 MOV R0,$REG0 ;SAVE TEST #
3294 015104 016667 000002 164070 MOV 2(SP),$TMP0 ;SAVE THE ERROR PSW
3295 015112 012767 015126 164124 MOV #1,$ESCAPE ;GO TO 1$ AFTER ERROR PRINT
3296 015120 011667 164054 MOV (SP),$REG7 ;SAVE THE ERROR PC
3297 015124 104004 ERROR+4 ;REPORTED TRAP ERROR
3298 015126 000137 001772 1$: JMP @RESTR ;ATTEMPT TO RESTART THE PROGRAM
3299 ;AND TRY AGAIN
3300
3301
3302 ;*****
3303 ;ERROR MESSAGE INFORMATION
3304 ;*****
3305

```

3306
3307
3308
3309
3310
3311
3312
3313
3314
3315
3316
3317
3318
3319
3320
3321
3322
3323
3324
3325
3326
3327
3328
3329
3330
3331
3332
3333
3334
3335
3336
3337
3338
3339
3340
3341
3342
3343
3344
3345
3346
3347
3348
3349
3350
3351
3352
3353
3354
3355
3356
3357
3358
3359
3360
3361

015132 046104 030461 051040
015140 043505 051511 042524
015146 020122 042522 042506
015154 042522 041516 020105
015162 040503 051525 042105
015170 052040 046511 047505
015176 052125 000
015201 040 050050 024503
015206 020040 020040 050050
015214 024523 020040 020040
015222 051450 024520 020040
015230 020040 042524 052123
015236 020040 042040 053105
015244 042101 020122 051040
015252 043505 042101 000122

015260 001116 001202 001176
015266 001162 001164 001166
015274 000000

015276 046104 030461 051040
015304 043505 051511 042524
015312 020122 051105 047522
015320 000122
015322 024040 041520 020051
015330 020040 024040 051520
015336 020051 020040 024040
015344 050123 020051 020040
015352 052040 051505 020124
015360 020040 042504 040526
015366 051104 020040 042522
015374 040507 051104 020040
015402 053440 051501 020040
015410 020040 051440 041057
015416 000
015420 015420
015426 001116 001202 001176
015434 001162 001164 001166
001170 001172 000000

015442 046104 030461 042040
015450 052101 020101 047503
015456 050115 051101 020105
015464 051105 047522 000122
015472 024040 041520 020051
015500 020040 024040 051520
015506 020051 020040 024040
015514 050123 020051 020040
015522 052040 051505 020124
015530 020040 040527 040523

: INFORMATION FOR ERROR MESSAGE 1

EM1: .ASCIZ 'DL11 REGISTER REFERENCE CAUSED TIMEOUT'

DH1: .ASCIZ ' (PC) (PS) (SP) TEST DEVADR REGADR'

.EVEN
DT1: .WORD \$ERRPC,\$TMPD,\$REG6,\$REG0,\$REG1,\$REG2,0

: INFORMATION FOR ERROR MESSAGE 2

EM2: .ASCIZ 'DL11 REGISTER ERROR'

DH2: .ASCIZ ' (PC) (PS) (SP) TEST DEVADR REGADR WAS S B'

.EVEN
DT2: .WORD \$ERRPC,\$TMPD,\$REG6,\$REG0,\$REG1,\$REG2,\$REG3,\$REG4,0

: INFORMATION FOR MESSAGE 3

EM3: .ASCIZ 'DL11 DATA COMPARE ERROR'

DH3: .ASCIZ ' (PC) (PS) (SP) TEST WASADR SHADR WAS S B'

3362	015536	051104	020040	044123
3363	015544	040502	051104	020040
3364	015552	020040	040527	020123
3365	015560	020040	020040	027523
3366	015566	000102		
3367				
3368	015570	001116	001202	001176
3369	015576	001162	001164	001166
3370	015604	001170	001172	000000

```
.EVEN
DT3: .WORD $ERRPC,$TMPD,$REG6,$REG0,$REG1,$REG2,$REG3,0
```

: INFORMATION FOR MESSAGE 4

3374	015612	047125	054105	042520
3375	015620	052103	042105	052040
3376	015626	040522	020120	047524
3377	015634	053040	041505	047524
3378	015642	020122	052101	046040
3379	015650	041517	052101	047511
3380	015656	020116	020040	000040
3381	015664	024040	041520	020051
3382	015672	020040	024040	051520
3383	015700	020051	020040	024040
3384	015706	050123	020051	020040
3385	015714	052040	051505	000124

```
EM4: .ASCIZ 'UNEXPECTED TRAP TO VECTOR AT LOCATION '
```

```
DM4: .ASCIZ ' (PC) (PS) (SP) TEST'
```

3387	015722	001200	001202	001176
3388	015730	001162	000000	

```
.EVEN
DT4: .WORD $REG7,$TMPD,$REG6,$REG0,0
```

: ERROR INFORMATION FOR ERROR MESSAGE 5

3392	015734	046104	030461	051440
3393	015742	043117	020124	051105
3394	015750	047522	020122	050050
3395	015756	051101	052111	026131
3396	015764	051106	046501	047111
3397	015772	026107	047440	020122
3398	016000	053117	051105	052522
3399	016006	024516	000	
3400	016011	040	050050	024503
3401	016016	020040	020040	050050
3402	016024	024523	020040	020040
3403	016032	051450	024520	020040
3404	016040	020040	042524	052123
3405	016046	020040	042040	053105
3406	016054	042101	020122	051040
3407	016062	043505	042101	020122
3408	016070	020040	051050	043505
3409	016076	000051		

```
EM5: .ASCIZ 'DL11 SOFT ERROR (PARITY, FRAMING, OR OVERRUN)'
```

```
DM5: .ASCIZ ' (PC) (PS) (SP) TEST DEVADR REGADR (REG)'
```

3411	016100	001116	001202	001176
3412	016106	001162	001164	001166
3413	016114	001170	000000	

```
.EVEN
DT5: .WORD $ERRPC,$TMPD,$REG6,$REG0,$REG1,$REG2,$REG3,0
```

: INFORMATION FOR ERROR MESSAGE 6

3417	016120	024040	041520	020051
------	--------	--------	--------	--------

```
DM6: .ASCIZ ' (PC) (PS) (SP) REGADR'
```

MAINDEC-11-DZDLC-A MACY11 27(732) 20-SEP-76 09:19 PAGE 73
 DZDLCR.P11 POWER DOWN AND UP ROUTINES

3418	016126	020040	024040	051520	
3419	016134	020051	020040	024040	
3420	016142	050123	020051	020040	
3421	016150	042522	040507	051104	
3422	016156	000			
3423		016160			
3424	016160	001116	001204	001176	.EVEN DT6: .WORD SERRPC,STMP1,\$REG6,\$REG2,0
3425	016166	001166	000000		
3426					
3427					
3428					; INFORMATION FOR ERROR MESSAGE 7
3429	016172	024040	041520	020051	DH7: .ASCIZ '(PC) DEVADR REGADR (REG)'
3430	016200	020040	042504	040526	
3431	016206	051104	020040	042522	
3432	016214	040507	051104	020040	
3433	016222	024040	042522	024507	
3434	016230	000			
3435		016232			.EVEN
3436	016232	001116	001164	001166	DT7: .WORD SERRPC,\$REG1,\$REG2,\$REG3,0
3437	016240	001170	000000		
3438					
3439					
3440					; INFORMATION FOR ERROR MESSAGE 10
3441	016244	024040	041520	020051	DH10: .ASCIZ '(PC) DEVADR REGADR (REG) S/B'
3442	016252	020040	042504	040526	
3443	016260	051104	020040	042522	
3444	016266	040507	051104	020040	
3445	016274	024040	042522	024507	
3446	016302	020040	020040	027523	
3447	016310	000102			
3448					.EVEN
3449	016312	001116	001164	001166	DT10: .WORD SERRPC,\$REG1,\$REG2,\$REG3,\$REG4,0
3450	016320	001170	001172	000000	
3451					; MISCELLANEOUS MESSAGES
3452					
3453	016326	052516	046114	042055	XMSG1: .ASCIZ 'NULL-DEL-NULL SEQUENCE TIMEOUT AT FOLLOWING PC'
3454	016334	046105	047055	046125	
3455	016342	020114	042523	052521	
3456	016350	047105	042503	052040	
3457	016356	046511	047505	052125	
3458	016364	040440	020124	047506	
3459	016372	046114	053517	047111	
3460	016400	020107	041520	000	
3461	016405	102	047111	051101	XMSG2: .ASCIZ 'BINARY UP COUNT SEQUENCE TIMEOUT AT FOLLOWING PC'
3462	016412	020131	050125	041440	
3463	016420	052517	052116	051440	
3464	016426	050505	042525	041516	
3465	016434	020105	044524	042515	
3466	016442	052517	020124	052101	
3467	016450	043040	046117	047514	
3468	016456	044527	043516	050040	
3469	016464	000103			
3470	016466	044502	040516	054522	XMSG3: .ASCIZ 'BINARY DOWN COUNT SEQUENCE TIMEOUT AT FOLLOWING PC'
3471	016474	042040	053517	020116	
3472	016502	047503	047125	020124	
3473	016510	042523	052521	047105	

3474	016516	042503	052040	046511	
3475	016524	047505	052125	040440	
3476	016532	020124	047506	046114	
3477	016540	053517	047111	020107	
3478	016546	041520	000		
3479	016551	127	051117	052123	XMSG4: .ASCIZ 'WORST CASE PATTERN SEQUENCE TIMEOUT AT FOLLOWING PC'
3480	016556	041440	051501	020105	
3481	016564	040520	052124	051105	
3482	016572	020116	042523	052521	
3483	016607	047105	042503	052040	
3484	016606	046511	047505	052125	
3485	016614	040440	020124	047506	
3486	016622	046114	053517	047111	
3487	016630	020107	041520	000	
3488					
3489	016635	015	046412	044501	STMES: .ASCIZ <15><12>'MAINDEC-11-DZDLC-A'<15><12>
3490	016642	042116	041505	030455	
3491	016650	026461	055104	046104	
3492	016656	026503	006501	000012	
3493					
3494	016664	005015	047531	020125	PROGEM: .ASCIZ <15><12>'YOU HAVE SELECTED PROGRAM NO. 2'<15><12>
3495	016672	040510	042526	051440	
3496	016700	046105	041505	042524	
3497	016706	020104	051120	043517	
3498	016714	040522	020115	047516	
3499	016722	020056	006462	000012	
3500	016730	005015	047531	020125	PROG3M: .ASCIZ <15><12>'YOU HAVE SELECTED PROGRAM NO. 3'<15><12>
3501	016736	040510	042526	051440	
3502	016744	046105	041505	042524	
3503	016752	020104	051120	043517	
3504	016760	040522	020115	047516	
3505	016766	020056	006463	000012	
3506	016774	005015	047531	020125	PROG4M: .ASCIZ <15><12>'YOU HAVE SELECTED PROGRAM NO. 4'<15><12>
3507	017002	040510	042526	051440	
3508	017010	046105	041505	042524	
3509	017016	020104	051120	043517	
3510	017024	040522	020115	047516	
3511	017032	020056	006464	000012	
3512	017040	005015	047531	020125	PROG5M: .ASCIZ <15><12>'YOU HAVE SELECTED PROGRAM NO. 5'<15><12>
3513	017046	040510	042526	051440	
3514	017054	046105	041505	042524	
3515	017062	020104	051120	043517	
3516	017070	040522	020115	047516	
3517	017076	020056	006465	000012	
3518	017104	005015	051124	047101	XDB: .ASCIZ <15><12>'TRANSMITTER DONE BIT NEVER SET PC= '
3519	017112	046523	052111	042524	
3520	017120	020122	047504	042516	
3521	017126	041040	052111	047040	
3522	017134	053105	051105	051440	
3523	017142	052105	020040	041520	
3524	017150	020075	000		
3525	017153	015	051012	041505	ROB: .ASCIZ <15><12>'RECEIVER DONE BIT NEVER SET PC= '
3526	017160	044505	042526	020122	
3527	017166	047504	042516	041040	
3528	017174	052111	047040	053105	
3529	017202	051105	051440	052105	

3530	017210	020040	041520	020075
3531	017216	000		
3532				
3533				
3534	017217	015	053412	040510
3535	017224	020124	051511	052040
3536	017232	042510	041440	040510
3537	017240	040522	052103	051105
3538	017246	046040	047105	052107
3539	017254	020110	032450	033054
3540	017262	033454	047440	020122
3541	017270	020070	044502	051524
3542	017276	037451	000	
3543	017301	015	042012	020117
3544	017305	047531	020125	044527
3545	017314	044123	052040	020117
3546	017322	042524	052123	047440
3547	017330	044124	051105	052040
3548	017336	040510	020116	044124
3549	017344	105		
3550	017345	015	042012	043105
3551	017352	052501	052114	042040
3552	017360	053105	041511	020105
3553	017366	030450	030057	036440
3554	017374	054440	051505	047057
3555	017402	024517	000077	
3556	017406	005015	044127	052111
3557	017414	044440	020123	044124
3558	017422	020105	051461	020124
3559	017430	042522	042503	053111
3560	017436	051105	051440	040524
3561	017444	052524	020123	042522
3562	017452	044507	052123	051105
3563	017460	040440	042104	042522
3564	017466	051523	020077	000040
3565	017474	005015	044127	052101
3566	017502	044440	020123	044124
3567	017510	020105	051461	020124
3568	017516	042522	042503	053111
3569	017524	051105	020123	042526
3570	017532	052103	051117	040440
3571	017540	042104	042522	051523
3572	017546	020077	000040	
3573	017552	005015	047504	054440
3574	017560	052517	053440	047101
3575	017566	020124	047524	052040
3576	017574	051505	020124	052515
3577	017602	052114	050111	042514
3578	017610	042040	053105	041511
3579	017616	051505	030440	030057
3580	017624	054475	051505	047057
3581	017632	037517	020040	000
3582	017637	015	053412	040510
3583	017644	020124	051511	052040
3584	017652	042510	051440	040524
3585	017660	052524	020123	042522

:MESSAGES SEEKING USER RESPONSE

LENGTH: .ASCIZ <15><12>'WHAT IS THE CHARACTER LENGTH (5,6,7 OR 9 BITS)''

DEFAULT: .ASCII <15><12>'DO YOU WISH TO TEST OTHER THAN THE'

.ASCIZ <15><12>'DEFAULT DEVICE (1/0 = YES/NO)''

MFIRSTD: .ASCIZ <15><12>'WHAT IS THE 1ST RECEIVER STATUS REGISTER ADDRESS? '

MVECT: .ASCIZ <15><12>'WHAT IS THE 1ST RECEIVER'S VECTOR ADDRESS? '

MULDEV: .ASCIZ <15><12>'DO YOU WANT TO TEST MULTIPLE DEVICES 1/0=YES/NO? '

MLASTD: .ASCIZ <15><12>'WHAT IS THE STATUS REGISTER ADDRESS OF THE LAST RECEIVER? '

3586	017666	044507	052123	051105	
3587	017674	040440	042104	042522	
3588	017702	051523	047440	020106	
3589	017710	044124	020105	040514	
3590	017716	052123	051040	041505	
3591	017724	044505	042526	037522	
3592	017732	020040	000		
3593	017735	015	051412	046517	MRANGE: .ASCIZ <15><12>'SOMETHING WRONG-ANSWER THE LAST QUESTION AGAIN! '
3594	017742	052105	044510	043516	
3595	017750	053440	047522	043516	
3596	017756	040455	051516	042527	
3597	017764	020122	044124	020105	
3598	017772	040514	052123	050440	
3599	020000	042525	052123	047511	
3600	020006	020116	043501	044501	
3601	020014	020516	020040	000	
3602	020021	015	053412	040510	PLEVEL: .ASCIZ <15><12>'WHAT IS YOUR INTERRUPT PRIORITY LEVEL? '
3603	020026	020124	051511	054440	
3604	020034	052517	020122	047111	
3605	020042	042524	051122	050125	
3606	020050	020124	051120	047511	
3607	020056	044522	054524	046040	
3608	020064	053105	046105	020077	
3609	020072	000040			
3610	020074	005015	051120	043517	FOULUP: .ASCII <15><12>'PROGRAM DEVICE ACTIVE LOCATION SHOWS NO DEVICE ACTIVE'
3611	020102	040522	020115	047304	
3612	020110	044526	042503	040440	
3613	020116	052103	053111	020105	
3614	020124	047514	040503	044524	
3615	020132	047117	051440	047510	
3616	020140	051527	047040	020117	
3617	020146	042504	044526	042503	
3618	020154	040440	052103	053111	
3619	020162	105			
3620	020163	015	051412	052105	.ASCII <15><12>'SET SWITCH 0 TO A ONE (1) AND'
3621	020170	051440	044527	041524	
3622	020176	020110	020060	047524	
3623	020204	040440	047440	042516	
3624	020212	024040	024461	040440	
3625	020220	042116			
3626	020222	005015	044510	020124	.ASCIZ <15><12>'HIT CONTINUE TO GO BACK TO DEVICE SECTION AGAIN'
3627	020230	047503	052116	047111	
3628	020236	042525	052040	020117	
3629	020244	047507	041040	041501	
3630	020252	020113	047524	042040	
3631	020260	053105	041511	020105	
3632	020266	042523	042514	052103	
3633	020274	047511	020116	043501	
3634	020302	044501	000116		
3635	020306	005015	044127	052101	LINTAD: .ASCIZ <15><12>'WHAT IS THE TRANSMITTER DATA BUFFER ADDRESS? '
3636	020314	044440	020123	044124	
3637	020322	020105	051124	047101	
3638	020330	046523	052111	042524	
3639	020336	020122	040504	040524	
3640	020344	041040	043125	042506	
3641	020352	020122	042101	051104	

3642	020360	051505	037523	020040	
3643	020366	000			
3644	020367	015	053412	040510	SELCAR: .ASCIZ <15><12>'WHAT IS THE CHARACTER TO BE TRANSMITTED (OCTAL ASCII E.G. A=101
3645	020374	020124	051511	052040	
3646	020402	042510	041440	040510	
3647	020410	040522	052103	051105	
3648	020416	052040	020117	042502	
3649	020424	052040	040522	051516	
3650	020432	044515	052124	042105	
3651	020440	024040	041517	040524	
3652	020446	020114	051501	044503	
3653	020454	020111	027105	027107	
3654	020462	040440	030475	030460	
3655	020470	037451	020040	000	
3656	020475	015	053412	040510	SELDLY: .ASCIZ <15><12>'WHAT IS THE DESIRED MSEC. DELAY (OCTAL E.G. 10=8(10))? '
3657	020502	020124	051511	052040	
3658	020510	042510	042040	051505	
3659	020516	051111	042105	046440	
3660	020524	042523	027103	042040	
3661	020532	046105	054501	024040	
3662	020540	041517	040524	020114	
3663	020546	027105	027107	030440	
3664	020554	036460	024070	030061	
3665	020562	024451	020077	000040	
3666	020570	005015	044127	052101	LINRAD: .ASCIZ <15><12>'WHAT IS THE RECEIVER DATA BUFFER ADDRESS? '
3667	020576	044440	020123	044124	
3668	020604	020105	042522	042503	
3669	020612	053111	051105	042040	
3670	020620	052101	020101	052502	
3671	020626	043106	051105	040440	
3672	020634	042104	042522	051523	
3673	020642	020077	000040		
3674	020646	005015	051511	040440	RSTALL: .ASCIZ <15><12>'IS A RANDOM WAIT TIME (MSEC.) DESIRED 1/0=YES/NO? '
3675	020654	051040	047101	047504	
3676	020662	020115	040527	052111	
3677	020670	052040	046511	020105	
3678	020676	046450	042523	027103	
3679	020704	020051	042504	044523	
3680	020712	042522	020104	030440	
3681	020720	030057	054475	051505	
3682	020726	047057	037517	020040	
3683	020734	000			
3684	020735	015	054412	052517	FAILSA: .ASCII <15><12>'YOU HAVE SWRB SET INDICATING LOOP ON TEST'
3685	020742	044040	053101	020105	
3686	020750	053523	034122	051440	
3687	020756	052105	044440	042116	
3688	020764	041511	052101	047111	
3689	020772	020107	047514	050117	
3690	021000	047440	020116	042524	
3691	021006	052123			
3692	021010	005015	040510	042526	.ASCII <15><12>'HAVE YOU MODIFIED THE PROPER LOCATIONS FOR THE'
3693	021016	054440	052517	046440	
3694	021024	042117	043111	042511	
3695	021032	020104	044124	020105	
3696	021040	051120	050117	051105	
3697	021046	046040	041517	052101	

```

3698 021054 047511 051516 043040
3699 021062 051117 052040 042510
3700 021070 005015 042504 044526
3701 021076 042503 052040 040510
3702 021104 020124 047531 020125
3703 021112 040527 052116 052040
3704 021120 020117 042524 052123
3705 021126 077
3706 021127 015 044412 020106
3707 021134 047523 026440 050040
3708 021142 042522 051523 052040
3709 021150 042510 041440 047117
3710 021156 044524 052516 020105
3711 021164 053523 052111 044103
3712 021172 005015 043111 047040
3713 021200 052117 026440 046440
3714 021206 042117 043111 020131
3715 021214 044124 020105 051120
3716 021222 050117 051105 046040
3717 021230 041517 052101 047511
3718 021236 051516 020054 044124
3719 021244 047105
3720 021246 005015 042522 052123
3721 021254 051101 020124 044124
3722 021262 020105 051120 043517
3723 021270 040522 020115 052101
3724 021276 040440 042104 042522
3725 021304 051523 031040 030060
3726 021312 000
3727
3728 021313 040 020075 041520
3729 021320 000040
3730
3731
3732
3733
3734 021322 000400
3735
3736
3737 021722 000400
3738
3739
3740
3741 022322 000000
3742
3743 000001
  
```

```

.ASCII <15><12>'DEVICE THAT YOU WANT TO TEST?'
.ASCII <15><12>'IF SO - PRESS THE CONTINUE SWITCH'
.ASCII <15><12>'IF NOT - MODIFY THE PROPER LOCATIONS, THEN'
.ASCIIZ <15><12>'RESTART THE PROGRAM AT ADDRESS 200'
  
```

```

PCMSG: .ASCII : = PC'
        .ASCIIZ : ;
  
```

```

.EVEN
;512. WORDS RESERVED FOR TWO 256. BYTE INPUT/OUTPUT DATA BUFFERS
DLBUFO: .BLKB 256. ;RSVD FOR OUTPUT BUFFER
;THIS IS THE DATA BEING SENT OUT
;BY THE TRANSMITTER
DLBUFI: .BLKB 256. ;RSVD FOR INPUT BUFFER
;THIS IS THE DATA THAT WAS PICKED
;UP BY THE RECEIVER (I.E. DATA
;SENT BY THE TRANSMITTER - HOPEFULLY)
;TAG MARKS END OF BUFFERS
BUFEND: 0
.END
  
```


SGETH2	010416	2249#																			
SGTSWR=	*****	2998																			
SHO	= 000001	427	429																		
SHOCT	012452	2706*	2717#																		
SICNT	001104	590#	2311*	2312	2314*	2324															
SILLUP	013276	2899	2914	2933#																	
SINTAG	001135	604#																			
SITEMB	001114	594#	2350*	2367	2379																
SLF	001254	647#	2367#	2548	2658	2718	2789	2850													
SLPADR	001106	591#	824*	2302*	2317*	2322	2324														
SLPERR	001110	592#	825*	2302	2318*	2324	2361														
SMAIL =	*****	847	2317	2356	2742																
SMNEW	012303	2662#																			
SMSWR	012272	2660#																			
SMXCNT	010730	2315	2324*																		
SNUL	001154	612#	2760	2789																	
SNWTST=	000001	1137#	1152#	1167#	1182#	1197#	1207#	1217#	1234#	1263#	1280#	1332#	1363#	1399#							
		1434#	1453#	1489#	1510#	1523#	1580#	1637#	1694#												
SOCNT	011444	2448*	2477*	2490#																	
SOMODE	011446	2443*	2447*	2452	2455*	2466*	2492#														
SOVER	010714	2279	2295	2303	2313	2321#															
SPASS	001100	587#	2237*	2238*	2246	2259	2309	2325													
SPOWER	013304	2929	2936#																		
SPWRAD	013272	2931#																			
SPWRDN	013132	819	2898#	2926																	
SPWRMG	013266	2929#																			
SPWRJP	013204	2908	2914#																		
SQUES	001252	645#	1007	1010	1013	1129	1132	1830	1915	2024	2026	2133	2135	2367							
		2641	2658	2715	2718	2789	2949	2850													
SROCHR	011674	2578#	2889																		
SRODEC	012674	2804#	2892																		
SROLIN	012014	2606#	2890																		
SROOCT	012314	2679#	2891																		
SROSZ =	000010	2599#																			
SREGAD	001160	616#																			
SREG0	011162	618#	3088*	3098*	3293*	3024	3346	3368	3387	3411											
SREG1	011164	619#	3058*	3060*	3089*	3099*	3324	3346	3368	3411	3436	3449									
SREG2	011166	620#	3056*	3058	3090*	3103*	3224	3346	3368	3411	3424	3436	3449								
SREG3	001170	621#	3062*	3091*	3346	3368	3411	3436	3449												
SREG4	001172	622#	3071*	3092*	3346	3368	3449														
SREG5	001174	623#																			
SREG6	001176	624#	1302*	2947*	2965*	2971*	3085*	3086*	3101*	3102*	3202*	3291*	3324	3346							
		3368	3287	3411	3424																
SREG7	001200	625#	1361*	3296*	3387																
SRTNAD	010440	2258#																			
SRA =	*****	2893																			
SSAVRE =	*****	2893																			
SSAVRE	013302	2907*	2915	2916*	2917*	2935#															
SSCOPE	010462	813	2277#																		
SSE*UP=	000017	803#	812	813	817	817	819	821	822	824	2235	2278	2341	2359							
		2366	2567	2664																	
SET*F =	177777	803#																			
SEVLAC	010666	2287	2316#																		
SSWR =	167400	410#	427	435	436	437	438	439	440	441	442	642	643	644							
		821	822	824	825	1111	1156	1171	1186	1201	1211	1221	1230	1231							
		1284	1336	1367	1403	1438	1457	1493	1514	1527	1594	1641	1643	1644							

.SWRHI	412#	431
.SWRLO	412#	443#
.SCHTC	412#	447
.SCMTA	411#	579
.SEOP	412#	2163
.SERRJ	411#	2326
.SEART	413#	2368
.SPOWE	413#	2894
.SRDDE	414#	2790
.SRDOC	414#	2665
.SREAD	414#	2562
.SSCOP	413#	2263
.STRAP	413#	2851
.STYFD	413#	2494
.STYFE	413#	2719
.STYFO	413#	2416

444

