DA11F

BUS WINDOW EXERCISER

MD-11-DZDAB-A

EP-DZDAB-A-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN U.S.A

```
                                .REM    %
```

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZDAB-A-D
PRODUCT NAME:	BUS WINDOW EXERCISER
DATE CREATED:	1-MAY-73
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	JOE STUBBLEBINE

CONTENTS

1. ABSTRACT

THIS PROGRAM EXERCISES THE BUS WINDOW USING AN I/O DEVICE OR EXECUTING CODE THROUGHOUT THE OTHER PROCESSOR'S MEMORY. IT IS ASYNCRONOUS AND 3/4 DUPLEX IN ITS OPERATION. IF THE WINDOW IS ABOVE 32K, KT11C/D WILL BE USED TO REACH THE WINDOW.

2. REQUIREMENTS

2.1 EQUIPMENT

TWO PDP11 STANDARD COMPUTERS WITH A MINIMUM OF 8K OF MEMORY
KT11C/D FOR MEMORY EXPANSION (OPTIONAL)

2.2 STORAGE

PROGRAM STORAGE - THE ROUTINES USE MEMORY 0 - 17776

2.3 PRELIMINARY PROGRAMS

MEMORY DIAGNOSTICS
DEVICE DIAGNOSTICS
MAINDEC-11-DZDAA-A

3. LOADING PROCEDURE

1) LOAD THE ABS LOADER IF NOT IN MEMORY
2) LOAD PROGRAM WITH ABS LOADER
3) LOAD OTHER PROCESSOR
   A) IF WINDOW IS NOT UP, USE ABS LOADER ON OTHER PROCESSOR
      OR
   B) IF THE WINDOW IS UP:
      1) LOAD ADDRESS 510
      2) START
      3) LOAD ADDRESS *500 IN OTHER MACHINE (* = WINDOW ADDRESS)
      4) START
      5) BOTH PROCESSORS WILL SELF START IF SWITCH 7 IS CLEAR.

4. STARTING PROCEDURE

1) THERE ARE TWO WAYS OF STARTING THE PROGRAM. FIRST OF ALL THE PROGRAM CAN BE LOADED INTO EACH PROCESSOR BY USING

```
   141                                  THE ABS LOADER.  THE PROGRAMS CAN THEN BE STARTED AT  THE
   142                                  STARTING  ADDRESS  SPECIFIED  IN SECTION 4.3 .  THE OTHER
```

OPTION IS TO LOAD THE PROGRAM INTO ONE PROCESSOR USING THE ABS LOADER AND THE HAVE THE WINDOW TRANSFER THE PROGRAM TO THE OTHER PROCESSOR AS DESCRIBED IN SECTION 3 . TO USE THIS METHOD THE WINDOW MUST BE WORKING, IT MUST HAVE A SIZE OF AT LEAST 4K, AND THE WINDOW CANNOT BE LOCATED AT 32K OR ABOVE.

2) IF THE WINDOW IS USED TO LOAD THE PROGRAM INTO PROCESSOR "B" AND SWITCH 7 IS CLEAR, THE PROGRAMS WILL SELF START. PROCESSOR "A" WILL AUTOMATICALLY BECOME THE MASTER.

3) IF THE PROGRAM DOES NOT AUTOMATICALLY SELF START DUE TO LOADING PROGRAM MANUALLY INTO BOTH PROCESSORS OR SWITCH 7 BEING SET WHEN LOADING THRU THE WINDOW, THEN IT BECOMES NECESSARY TO START EACH PROCESSOR AT A DIFFERENT STARTING ADDRESS. ONE PROCESSOR MUST BE STARTED AT SA=200 AND THE OTHER PROCESSOR MUST BE STARTED FROM SA 210. BY DOING THIS A MASTER SLAVE CONFIGURATION IS ESTABLISHED TO PREVENT RACE CONDITIONS WHEN ASKING FOR THE WINDOW. AFTER THE INITIAL START, WHICH MODIFIES SOME CODE, ALL RESTARTS MUST BE DONE FROM SA 2 OR BOTH PROCESSORS.

4.1 CONTROL SWITCH SETTINGS

SEE ! 1 (ALL DOWN FOR WORST CASE TESTING)

4.2 STARTING ADDRESS

SEE 4.0.

4.3 PROGRAM AND/OR OPERATOR ACTION

1) START AT SA 200 FOR ONE PROCESSOR AND SA 210 FOR THE OTHER PROCESSOR OR SELF START AS IN SEC. 4.0
2) TYPE DEVICE (RF11, RK11, RP11, RC11, TC11, TM11, OR EXECUTE) AND RETURN.
3) TEST WILL START (↑C WILL RETURN TO STEP 2)
4) NOTE: ALL PROGRAM RESTARTS ARE FROM SA200 FOR BOTH PROCESSORS.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

AT STARTING ADDRESS, ALL SWITCHES DOWN IS WORST CASE TESTING. THE BELL WILL RING UPON COMPLETION OF A PASS.

```
   199
```

5.1.1 SWITCH SETTINGS ARE:

SW(15) = 1 ..... HALT ON ERROR
SW(14) = 1 ..... HANG ON CURRENT SEGMENT
SW(13) = 1 ..... INHIBIT PRINTOUT
SW(10) = 1 ..... INHIBIT BELL ON PASS COMPLETE
SW(08) = 1 ..... TYPE SEGMENT UNDER TEST
SW(07) = 1 ..... INHIBIT AUTOMATIC SELF START WHEN LOADING
PROGRAM THRU THE WINDOW

5.2 SUBROUTINE ABSTRACTS

5.2.1 TRAPCATCHER

A ".+2" - "HALT" SEQUENCE IS REPEATED FROM 4 - 776 TO CATCH ANY UNEXPECTED TRAPS. THUS ANY UNEXPECTED TRAPS OR INTERUPTS WILL HALT AT THE VECTOR + 2.

5.2.2 TRAP HANDLER

MOST OF THE SUBROUTINE CALLS ARE DONE VIA A TRAP+N CALL. TO FIND THE SUBROUTINE BEING CALLED, LOOK IN THE COMMENT SECTION OF THE TRAP DEFINATION TABLE FOR THE NAME OF THE TRAP, THEN SCAN TO THE LEFT MARGIN FOR THE STARTING ADDRESS OF THE SUBROUTINE.

6. ERRORS

6.1 ERROR PRINTOUT

THE FORMAT IS AS FOLLOWS:

RF11 ERROR: CS = 100305 ER = 001000

RF11 = DEVICE UNDER TEST
CS = CONTROL AND STATUS REGISTER
ER = ERROR REGISTER

DATA ERROR AT 401220 TRUE = 177777 RECEIVED = 177757

401120 = ADDRESS OF BAD DATA (18 BIT)
TRUE = DATA SENT
RECEIVED = DATA FOUND

WINDOW ERROR AT 146012 WCSRA = 141022 WDARA = 000000

146012 = ADDRESS TRAPPED FROM

```
256                                                    WCSRA =     WINDOW CONTROL AND STATUS REG.
257                                                    WDARA =     WINDOW DISPLACEMENT ADDRESS REGISTER
```

6.2 ERROR RECOVERY

RESTART AT 200 FOR BOTH PROCESSORS

7. RESTRICTIONS

IN ORDER TO ESTABLISH A MASTER/SLAVE CONFIGURATION, THE PROGRAMMER MUST INITIALLY BE STARTED AT TWO DIFFERENT STARTING ADDRESSES. IF THE WINDOW IS USED TO LOAD THE PROGRAM AND SWITCH 7 IS CLEAR, THE MASTER/SLAVE CONFIGURATION WILL BE SETUP. IF THE PROGRAMS ARE MANUALY STARTED REFER TO SECTION 4.0 FOR STARTING INSTRUCTIONS. ALL RESTARTS ARE FROM SA 200 FOR BOTH PROCESSORS.

8. MISCELLANEOUS

WHEN FINISHED RUNNING THE PROGRAM THE LOADER MAY BE RESTORED BY LOADING SA 562 AND HITTING START. THE PROCESSOR WILL RESTORE THE LOADER AND HALT.

8.1 EXECUTION TIME

THE EXECUTATION TIME IS DEPENDANT ON THE AMOUNT OF MEMORY AND THE DEVICE USED. THE BELL SHOULD RING WITHIN 5 MINUTES (USING TC11 IN 124K.)

8.2 STACK POINTER

STACK IS INITALLY SET TO 500

8.3 POWER FAIL

N/A

9. PROGRAM DESCRIPTION

THE PROGRAM SIZES CORE AND TEST SEQUENTIALLY ALL CORE OR ANY SECTION OF CORE IN CHUNKS EQUAL TO THE WONDOW SIZE USING THE DEVICE SPECIFIED. SO A SEGMENT OF CORE IS EQUAL TO THE WINDOW SIZE. THE DEVICE ON SIDE A USES THE MEMORY ON BUS B AND THE DEVICE ON SIDE B USES MEMORY ON BUS A. WORST CASE NOISE PATTERNS ARE USED FOR EACH MEMORY TYPE ON EACH PASS. IF A DEVICE IS NOT AVAILABLE, THE EXECUTE COMMAND CAN BE USED TO EXECUTE CODE THROUGHOUT THE OTHER PROCESSOR'S MEMORY. EACH MACHINE HAS A WINDOW INTERUPT HANDLER FOR OPENING THE WINDOW UPON REQUEST AND CODE TO REQUEST THE

```
                                        WINDOW IN A PARTICULAR MODE.  BECAUSE  BOTH  PROCESSORS  CAN
                                        NOT  READ AT THE SAME TIME, THE REQUESTING ROUTINE WAITS FOR
```

```
                                MAINDEC-11-DZDAB-A      BUS WINDOW EXERCISER                    PAGE 7
                                DESCRIPTION


                                          THE  OTHER  PROCESSOR  TO  DROP  THE  READ  REQUEST   BEFORE
                                          REQUESTING READ PRIVLAGES HIMSELF.
                                        %
                                .TITLE  MAINDEC-11-DZDAB-A      BUS WINDOW EXERCISER
                                ;COPYRIGHT 1972,DIGITAL EQUIPMENT CORP., MAYNARD MASS.
                                ;PROGRAM BY BOB BRAIN



                                ;     SWITCH                    USE
                                ;     ------                    ---------------------
100000                          SW15=   100000          ;HALT ON ERROR
040000                          SW14=   40000           ;HANG ON CURRENT SEGMENT
020000                          SW13=   20000           ;INHIBIT ERROR TYPEOUTS
010000                          SW12=   10000           ;
004000                          SW11=   4000            ;
002000                          SW10=   2000            ;0 - BELL ON PASS COMPLETE
                                                        ;1 - BELL ON ERROR
001000                          SW9=    1000            ;INHIBIT USE OF MEMORY EXPANSION DEVICE
000400                          SW8=    400             ;TRACE BANK UNDER TEST
000200                          SW7=    200             ;HALT ON LOAD COMPLETE


                                ;       BIT             WCSRA

                                ;       15              A ERROR
                                ;       14              B TIME OUT
                                ;       13              B ACLO
                                ;       12              B NEW DATA
                                ;       11              B DATA 3
                                ;       10              B DATA 2
                                ;        9              B DATA 1
                                ;        8              A TRANS ENABLE
                                ;        7              B TRANS ENABLE
                                ;        6              A I/E
                                ;        5              A DATA 3
                                ;        4              A DATA 2
                                ;        3              A DATA 1
                                ;        2              B WRITE ENABLE
                                ;        1              A WRITE ENABLE
                                ;        0              A NEW DATA


                                ;FUNCTIONS (IN THE COMMUNICATION BITS)

                                ;       FUNCTION        MEANING

000001                          CLEAR=      01          ;NOP - CLOSE WINDOW
000011                          READ=       11          ;READ ONLY
000021                          WRITE=      21          ;WRITE ONLY
000031                          R.W=        31          ;READ/WRITE
000041                          BANK=       41          ;BANK SELECT
```

```
372                             ;       51 & 61         ;FUTURE USE
373                             ;         71            ;GET READY FOR FUNCTION
```

```
                                        ;LOADING PROCEDURE:
                                        ;
                                        ;1) LOAD THE ABS LOADER IF NOT IN MEMORY
                                        ;2) LOAD PROGRAM WITH ABS LOADER
                                        ;3) LOAD ADDRESS 510
                                        ;4) START
                                        ;5) LOAD ADDRESS *500 IN OTHER MACHINE (* = WINDOW ADDRESS)
                                        ;6) START
                                        ;
  104400                                SCOPE=  TRAP
  104000                                HLT=    EMT
  000004                                TYPE=   IOT
  177776                                PS=     177776
  177570                                SWR=    177570
  177570                                DISPLAY=SWR
  000007                                BELL=   7
  000000                                R0=     %0
  000001                                R1=     %1
  000002                                R2=     %2
  000003                                R3=     %3
  000004                                R4=     %4
  000005                                R5=     %5
  000005                                TTY=    %5
  000006                                SP=     %6
  000007                                PC=     %7
  000000                                N=      0
                                        ;
  000000                                .=      0                       ;TRAP CATCHER FROM 0 - 776
                                        ;
  000200                                .=      200
                                        ;
000200  000137  001026                  JMP     @#BEGIN                 ;JUMP TO BEGINING ADDRESS OF PROGRAM
                                        ;
                                        ;STARTING ADDRESS OF MASTER
                                        ;
  000210                                .=      210
000210  012737  000240  006660          MOV     #240,@#DOF2
000216  012737  000240  006562          MOV     #240,@#DOF2+2
000224  012737  000240  007106          MOV     #240,@#DOF5
000232  000137  001026                  JMP     @#BEGIN
                                        ;
  000500                                .=      500
                                        ;
000500  000167  000010                  JMP     GETCOR                  ;GET THE PROGRAM
000504  000137  000562                  JMP     @#LOADR                 ;RESTORE THE LOADER
000510  000137  011026                  JMP     @#OPEN                  ;OPEN THE WINDOW
                                        ;
                                        ;THIS ROUTINE IS USED BY THE OTHER PROCESSOR TO LOAD THE PROGRAM
                                        ;INTO ITSELF. IT IS RUN AT LOAD TIME ONLY BY LOADING THE STARTING
                                        ;ADDRESS OF THE WINDOW PLUS 500 INTO THE SWITCHES. THEN DO A LOAD
                                        ;ADDRESS AND HIT START. THE OTHER PROCESSOR WILL THEN EXECUTE THE
                                        ;FOLLOWING CODE.
                                        ;
000514  010701                  GETCOR: MOV     PC,R1                   ;GET WINDOW ADDRESS
000516  000005                          RESET                           ;CLEAR THE WORLD
000520  162701  000516                  SUB     #GETCOR+2,R1            ;RELOCATE
```

```
000524  005000                          CLR     R0              ;SETUP FOR TRANSFER
000526  012120                  1$:     MOV     (1)+,(0)+       ;GET IT
000530  022700  017500                  CMP     #17500,R0       ;END?
000534  001374                          BNE     1$              ;NO - LOOP
000536  032737  000200  177570          BIT     #SW7,@#SWR      ;CHECK FOR HALT ON LOAD COMPLETE
000544  001401                          BEQ     .+4             ;CONTINUE
000546  000000                          HALT                    ;WAIT FOR CONTINUE
000550  012767  000240  010306          MOV     #NOP,BR.        ;MAKE IT EXECUTE A JMP @#BEGIN
000556  000137  001025                  JMP     @#BEGIN         ;START THE TEST

                        ;IF YOU WISH TO RESTORE THE ABSOLUTE LOADER, LOAD THE ADDRESS OF
                        ;THE FOLLOWING ROUTINE AND HIT START. THE PROCESSROR WILL HALT WHEN
                        ;COMPLETE.

000562  004767  010714          LOADR:  JSR     PC,LOORES       ;RESTORE THE LOADER
000566  000000                          HALT
```

```
                    001000                          .=      1000
                                                                                    #
001000  000000                                 ICNT:   0               ;LH = ITERATION COUNT ;RH = TEST NO.
001002  000000                                 ERRORS: 0               ;ERROR COUNT
001004  000000  000000                         PCNT:   0,0             ;2 WORD PASS COUNT
001010  000000                                 LAD:    0               ;LOOP ADDRESS FOR SCOPE
001012  000000                                 HLTADR: 0               ;LAST HLT INSTRUCTION ADD. EXECUTED
001014  000000                                 MX:     0               ;SEGMENT UNDER TEST
001016  000000                                 MIN:    0               ;FIRST BANK
001020  000000                                 LIMIT:  0               ;LAST BANK
001022  000000                                 OLIM:   0               ;OTHER MACHINE LIMIT
001024  000000                                 OPTION: 0               ;OPTION - GT = KT11

001026  005037  000006                 BEGIN:  CLR     @#6             ;HALT
001032  005067  010126                         CLR     CFLAG           ;CLEAR CONTROL "C" FLAG
001036  012706  000500                         MOV     #500,SP         ;SET STACK TO  ** 500 **
001042  012737  011572  000020                 MOV     #IOT$,@#20      ;LOAD IOT VECTOR
001050  012737  000340  000022                 MOV     #340,@#22       ;LOCK UP
001056  012737  012346  000024                 MOV     #PDOWN$,@#24    ;LOAD PF VECTOR
001064  012737  000340  000026                 MOV     #340,@#26       ;LOCK UP
001072  012737  006252  000034                 MOV     #TRAP$,@#34     ;SET FOR TRAP
001100  012737  000340  000036                 MOV     #340,@#36       ;LOCK UP WORLD
001106  012777  000340  010264                 MOV     #340,@RCTRAP+2  ;BR5
001114  012777  000340  010252                 MOV     #340,@RFTRAP+2  ;BR5
001122  012777  000340  010254                 MOV     #340,@RPTRAP+2  ;BR5
001130  012777  000340  010252                 MOV     #340,@RKTRAP+2  ;BR5
001136  012777  000340  010250                 MOV     #340,@TCTRAP+2  ;BR6
001144  012777  000340  010246                 MOV     #340,@TMTRAP+2  ;BR5
001152  012737  007614  000250                 MOV     #MMERR,@#250    ;SET FOR MEMORY MANAGEMENT ERROR
001160  012737  000340  000252                 MOV     #340,@#252      ;BR7
001166  004767  006324                         JSR     PC,.MAMK        ;ENABLE TRAP ON MEMORY PARITY ERRORS
001172  012767  011122  176660                 MOV     #READR,60       ;KEYBOARD VECTOR
001200  005067  177612                         CLR     MIN             ;RESET MINIMUM BANK
001204  012767  000000  177602                 MOV     #0,MX           ;INIT
001212  005067  006226                         CLR     MEMORY          ;WHICH MEMORY BEING CHECKED
001216  005067  010052                         CLR     NOTRD
001222  005067  177576                         CLR     OPTION          ;TYPE THE OPTION
001226  022767  177777  010036                 CMP     #-1,ONCE        ;IS THIS FIRST TIME THRU?
001234  001402                                 BEQ     1$              ;BRANCH IF YES
001236  000167  001322                         JMP     REDO
001242  032737  001000  177570  1$:     BIT     #SW9,@#SWR      ;INHIBIT USE OF MEMORY EXTENTION DEVICE
001250  001007                                 BNE     SIZEIT          ;SKIP
001252  012737  001270  000004                 MOV     #SIZEIT,@#4     ;SET FOR TIMEOUT
001260  005767  176306                         TST     SR0             ;CHECK FOR KT11
001264  105167  177534                         COMB    OPTION          ;OPTION IS KT11
```

```
                                                                ;RESET FOR MEM
001270  012737  000006  000004  SIZEIT: MOV     #6,@#4          ;CLEAR MEMORY MANAGEMENT FLAG
001276  005067  007766                  CLR     MMM             ;WHICH OPTION AM I
001302  005767  177516                  TST     OPTION          ;MUST HAVE MEMORY MANAGEMENT
001306  001016                          BNE     DOSEG

                                        ;THIS ROUTINE IS USED TO SIZE CORE IF MEMORY MANAGEMENT IS NOT
                                        ;AVAILABLE.


001310  012737  001342  000004  DOCORE: MOV     #2$,@#4         ;SET FOR MEM
001316  012701  017776                  MOV     #17776,R1       ;SET UP ADDRESS
001322  005000                          CLR     R0              ;SET UP BANK COUNT
001324  062701  020000          1$:     ADD     #20000,R1       ;MOVE TO NEXT BANK
001330  005200                          INC     R0              ;INC THE BANK COUNT
001332  005711                          TST     (1)             ;TIMEOUT?
001334  022701  177776                  CMP     #177776,R1      ;END?
001340  001371                          BNE     1$              ;LOOP IF NOT AT THE END
001342  000432                  2$:     BR      TYPEIT          ;TYPE IT

                                        ;THIS ROUTINE IS USED TO SIZE CORE IF MEMORY MANAGEMENT IS
                                        ;AVAILABLE.


001344  104446                  DOSEG:  MAPIT                   ;SETUP MEMORY MANAGEMENT REGISTERS
001346  005267  176220                  INC     SR0             ;TURN ON MEMORY MANAGEMENT
001352  012767  000200  170764          MOV     #200,KIPAR2     ;INIT TO BANK 1
001360  012737  001410  000004          MOV     #2$,@#4         ;SET TIMEOUT ADDRESS FOR CORE CALCULATIONS
001366  005737  057776          1$:     TST     @#57776         ;TRAP ON NON EX MEM
001372  062767  000200  170744          ADD     #200,KIPAR2     ;GO TO NEXT BANK
001400  022767  007600  170736          CMP     #7600,KIPAR2    ;LAST ONE?
001406  003367                          BGT     1$              ;TRY NEXT
001410  016700  170730          2$:     MOV     KIPAR2,R0       ;SAVE PAR2 IN R0
001414  006300                          ASL     R0              ; *ASH #-7,R0*
001416  000300                          SWAB    R0              ;KLUDGY ISN'T IT
001420  042700  177740                  BIC     #177740,R0      ;CLEAR JUNK
001424  005067  176142                  CLR     SR0             ;TURN IT OFF

001430  005300                  TYPEIT: DEC     R0              ;DROP BACK
001432  012737  000006  000004          MOV     #6,@#4          ;SET FOR MEM
001440  012706  000500                  MOV     #500,SP         ;CLEAR STACK **500**
001444  005267  007622                  INC     ONCE            ;TYPE THE OPTION ONLY ONCE
001450  004767  007752                  JSR     7,LODGET        ;GET THE LOADER ONLY THE FIRST TIME
001454  000004  001460                  TYPE    ,.+2            ;.ASCIZ <15><12><12>"MAINDEC-11-DZDAB-A"<15><12><12>"BAN
001524  010005                          MOV     R0,TTY          ;TYPE R0 IN OCTAL
001526  004767  010400                  JSR     PC,PRINTS       ;AND SUPRESS LEADING ZERO'S
001532  000004  001536                  TYPE    ,.+2            ;.ASCIZ " EXIST"
001546  005767  177252                  TST     OPTION          ;WHICH OPTION?
001552  001410                          BEQ     NOKT            ;GET OUT IF NONE
001554  000004  001560                  TYPE    ,.+2            ;.ASCIZ " WITH KT11"
```

```
                                NOKT:   MOV     R0,LIMIT        ;SAVE SIZE
                                        TYPE    ,.+2            ;.ASCIZ <15><12><12>
                        000004  3$:     MOV     #7$,@#4         ;SET FOR TRAPPING
                                        TST     @WCSRA          ;IS THE ADDRESS LEAGAL
                                        BR      FNDCSR          ;YES
                                7$:     CMP     (6)+,(6)+       ;CLEAR STACK
                        007402          MOV     #164000,WCSRA   ;SET WCSRA FOR SCANNING
                                        MOV     #8.,R1          ;SET COUNT
                        000004          MOV     #5$,@#4         ;SET FOR TIMEOUT
                                6$:     TST     @WCSRA          ;IS IT THERE?
                                        BR      FNDCSR          ;YES - GET OUT
                                5$:     CMP     (6)+,(6)+       ;CLEAR STACK
                        007352          ADD     #20,WCSRA       ;BUMP ADDRESS
                                        DEC     R1              ;DEC COUNT
                                        BNE     6$              ;LOOP UNTIL FOUND
                                        TYPE    ,.+2            ;.ASCIZ "WCSRA NOT FOUND"<15><12><12>
                                        HALT                    ;NO WCSRA
                                        BR      3$              ;TRY IT AGAIN

;THIS ROUTINE WILL DETERMINE WHERE THE WINDOW'S REGISTERS ARE AND
;THEN PROCEDE TO TYPE OUT ITS FUNCTIONAL PARAMETERS. IT ALSO DETERMINES
;WETHER OR NOT MEMORY MANAGEMENT IS REQUIRED. IF SO, IT SETS UP THE
;NECESSARY MEMORY MANAGEMENT REGISTERS.

FNDCSR:
                                        TYPE    ,.+2            ;.ASCIZ "WCSRA = "
                                        MOV     @WCSRA,TTY      ;TYPE WCSRA IN OCTAL
                                        JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
                                        TYPE    ,.+2            ;.ASCIZ "  "
                                        BR      32$             ;SKIP TRAP AREA
                                31$:    HALT                    ;WINDOW ADDRESS TIMED OUT
                                        CMP     (6)+,(6)+       ;CLEAR STACK
                        000004  32$:    MOV     #31$,@#4        ;SET 4 FOR TIMEOUT
                                        MOV     WCSRA,R0        ;GET THE FIRST ADD
                                        MOV     #WDBRA,R1       ;GET TABLE ADDRESS
                                60$:    TST     (0)+            ;ADD 2
                                        MOV     R0,(1)+         ;GET THE DATA
                                        CMP     #WVECA,R1       ;END?
                                        BNE     50$             ;LOOP UNTIL DONE
                        000004  21$:    MOV     #6,@#4          ;RESET 4
                                        MOV     #-1,@WRARA      ;FIND THE WINDOW SIZE
                                        MOV     @WRARA,R0       ;SAVE THE SIZE
                                        MOV     R0,SIZE         ;SAVE THE SIZE FOR POSTERITY
                                        NEG     R0              ;BUMP IT TO 1 BIT ONLY
                                        SWAB    R0              ;GET UPPER BITS
                                        ASR     R0              ;MOVE IT INTO POSITION
                                        CLR     R1              ;INIT FOR OFFSET
                                FND1:   TST     R0              ;IS IT 0
                                        BEQ     43$             ;YES - GET OUT
                                        ASR     R0              ;COUNT BITS
                                        ADD     #2,R1           ;KEEP COUNT
                                        BR      FND1            ;LOOP
                        007132  43$:    MOV     WIND(1),SIZ     ;GET THE SIZE
                                        MOV     R1,OFFSET       ;SAVE OFFSET
```

```
  599  002102  000004  011230                    TYPE    ,SIZ            ;TYPE THE WINDOW SIZE
  600  002106  000004  002112                    TYPE    .+2             ;.ASCIZ "K WINDOW AT "
  601  002130  017700  007114                    MOV     @WADRAX,R0      ;GET WINDOW ADDRESS
  602  002134  005067  176654                    CLR     MX              ;GET READY TO TYPE
  603  002140  012701  000005                    MOV     #5,R1           ;INIT COUNT
  604  002144  006300                    52$:    ASL     R0              ;MOVE IT LEFT
  605  002146  006167  176642                    ROL     MX              ;PICK UP DROPPED BIT
  606  002152  005301                            DEC     R1              ;COUNT IT
  607  002154  001373                            BNE     52$             ;LOOP UNTIL 0
  608  002156  017700  007066                    MOV     @WADRAX,R0      ;GET AGAIN
  609  002162  006300                            ASL     R0              ;GET FOR
  610  002164  006300                            ASL     R0              ;TYPING
  611  002166  010067  007070                    MOV     R0,WADRA        ;SAVE THE WINDOW ADDRESS
  612  002172  010005                            MOV     R0,TTY          ;TYPE R0 WITH MX AS 18 BIT ADDRESS
  613  002174  004767  007756                    JSR     PC,PRINTA       ;GO TO ADDRESS PRINTER
  614  002200  005067  007060                    CLR     EMBITS          ;CLEAR EXTENDED MEMORY BITS
  615  002204  017700  007040                    MOV     @WADRAX,R0      ;GET WINDOW ADDRESS
  616  002210  006300                            ASL     R0
  617  002212  103003                            BCC     1$              ;BRANCH IF BIT 17 CLEAR
  618  002214  052767  000040  007042            BIS     #40,EMBITS      ;SET MEMORY EXTENDED BIT 5
  619  002222  006300                    1$:     ASL     R0
  620  002224  103003                            BCC     2$              ;BRANCH IF BIT 16 CLEAR
  621  002226  052767  000020  007030            BIS     #20,EMBITS      ;BRANCH IF BIT 16 CLEAR
  622  002234  010067  007162            2$:     MOV     R0,IOWAC
  623  002240  012737  002402  000004            MOV     #FN02,@#4       ;SETUP TIMEOUT TRAP
  624  002246  005737  177572                    TST     @#SR0           ;TEST FOR MM
  625  002252  012767  177777  007010            MOV     #-1,MMN         ;SET MEMORY MANAGEMENT FLAG
  626  002260  005067  170054                    CLR     KIPAR0          ;MAP TO 0
  627  002264  012767  077406  170006            MOV     #77406,KIPDR0   ;READ/WRITE 4K
  628  002272  012702  000005                    MOV     #5,R2           ;SET THE COUNT
  629  002276  012701  172304                    MOV     #KIPDR2,R1      ;GET THE ADDRESS
  630  002302  012721  077406            47$:    MOV     #77406,(1)+     ;SET THE DPR
  631  002306  005302                            DEC     R2              ;CHECK COUNT
  632  002310  001374                            BNE     47$             ;LOOP
  633  002312  012767  007600  170036            MOV     #7600,KIPAR7    ;MAP TO  BANK 37
  634  002320  012767  077406  167770            MOV     #77406,KIPDR7   ;READ/WRITE 4K
  635  002326  017700  006716                    MOV     @WADRAX,R0      ;GET ADDRESS
  636  002332  006000                            ROR     R0              ;MAKE
  637  002334  006000                            ROR     R0              ;IT
  638  002336  006000                            ROR     R0              ;PAGE
  639  002340  006000                            ROR     R0              ;ADDRESS
  640  002342  042700  170000                    BIC     #170000,R0      ;CLEAR JUNK
  641  002346  010067  167772                    MOV     R0,KIPAR2       ;SET IT
  642  002352  012702  000004                    MOV     #4,R2           ;SET THE COUNT
  643  002356  012701  172346                    MOV     #KIPAR3,R1      ;GET ADDRESS
  644  002362  062700  000200            57$:    ADD     #200,R0         ;GO TO NEXT
  645  002366  010021                            MOV     R0,(1)+         ;SET IN THE ADDRESS
  646  002370  005302                            DEC     R2              ;COUNT
  647  002372  001373                            BNE     57$             ;LOOP
  648  002374  012767  [illegible]  000660       MOV     #40000,WADRA    ;FUDGE ADDRESS
  649  002402  012[illegible]  000004    FN02:   MOV     #6,@#4          ;RESET 4
  650  002410          006636                    MOV     @WVARA,R0       ;GET VECTOR ADDRESS
  651  [illegible]     002420                    TYPE    .+2             ;.ASCIZ "    VECTOR = "
  [illegible]  022700  001000                    CMP     #1000,R0        ;CHECK FOR UNDER 1000
       002442  002403                            BLT     25$             ;GO TO ERROR ROUTINE
       002444  022700  000136                    CMP     #136,R0         ;CHECK FOR OVER 136
```

```
   655  002450  002402                          BLT     26$             ;SKIP IF IN RANGE
   656  002452  000000                  25$:    HALT                    ;VECTOR IS NOT BETWEEN 140-776
   657  002454  000411                          BR      FND3            ;GET OUT IF CONTINUED
   658  002456  010067  006574          26$:    MOV     R0,WVECA        ;SAVE THE VECTOR ADDRESS
   659  002462  005720                          TST     (0)+            ;ADD 2
   660  002464  010067  006570                  MOV     R0,WVECA+2      ;SET UP STATUS
   661  002470  017705  006556                  MOV     @WVARA,TTY      ;TYPE @WVARA IN OCTAL
   662  002474  004767  007432                  JSR     PC,PRINTS       ;AND SUPRESS LEADING ZERO'S
   663  002500  012767  170000  006716  FND3:   MOV     #-4096.,WRDCNT  ;SETUP MAXIMUM WORDCOUNT
   664  002506  032767  000400  006552          BIT     #400,SIZE       ;.5K WINDOW?
        002514  001404                          BEQ     1$              ;BRANCH IF NO
        002516  012767  177000  006700          MOV     #-512.,WRDCNT   ;LIMIT WORDCOUNT TO 512
        002524  000417                          BR      REDO
   668  002526  032767  001000  006532  1$:     BIT     #1000,SIZE      ;1K WINDOW?
   669  002534  001404                          BEQ     2$              ;BRANCH IF NO
   670  002536  012767  176000  006660          MOV     #-1024.,WRDCNT  ;LIMIT WORDCOUNT TO 1000
   671  002544  000407                          BR      REDO
   672  002546  012767  002000  006512  2$:     BIT     #2000,SIZE      ;2K WINDOW?
   673  002554  001403                          BEQ     REDO            ;BRANCH IF NO
   674  002556  012767  174000  006640          MOV     #-2048.,WRDCNT  ;LIMIT WORDCOUNT TO 2K
   675  002564  012777  010404  006464  REDO:   MOV     #WINDOW,@WVECA  ;SET WINDOW VECTOR
   676  002572  005037  177776                  CLR     @#PS            ;LOWER PROCESSOR STATUS
   677  002576  012777  000340  006454          MOV     #340,@WVECA+2   ;HALT ADDRESS
   678  002604  042777  100000  006424          BIC     #100000,@WCSRA  ;KILL ERROR
   679  002612  052777  000100  006416          BIS     #100,@WCSRA     ;I/E
   680  002620  012737  000340  000006          MOV     #340,@#6        ;BR7
   681  002626  005767  006436                  TST     MMN             ;MEMORY MANAGEMENT NEEDED?
   682  002632  001403                          BEQ     1$              ;BRANCH IF NO
   683  002634  012737  000001  177572          MOV     #1,@#SR0        ;TURN ON MEMORY MANAGEMENT
   684  002642  032767  004000  006416  1$:     BIT     #4000,SIZE      ;IS WINDOW OVER 4K?
   685  002650  001006                          BNE     TTYIN           ;BRANCH IF NO
   686  002652  042767  020000  006402          BIC     #20000,WADR0
   687  002660  042767  020000  006534          BIC     #20000,IOWAD
```

```
                                        ;THE FOLLOWING CODE WILL DETERMINE WHICH TEST THE OPERATOR SELECTED
                                        ;AND TRANSFER CONTROL TO IT.  IF THE COMMAND CANNOT BE INTERPRETED,
                                        ;IT WILL TYPE A LIST OF AVAILABLE COMMANDS.

002666  012737  010326  000004  TTYIN:  MOV     #TRAP4,@#4      ;SET UP TRAP IF NO DEVICE
002674  000004  002700                  TYPE    ,.+2            ;.ASCIZ <15><12><12>"*"
002706  004767  007034                  JSR     PC,READ$        ;INPUT DATA FROM TTY
002712  004767  006610                  JSR     PC,CONTC        ;CHECK FOR RESTART
002716  012737  000340  177776          MOV     #340,@#PS       ;LOCK UP THE WORLD
002724  104450  000001                  FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
002730  005077  006304                  CLR     @WDBRA          ;BANK 0
002734  104450  000041                  FUNCTN  ,BANK           ;REQUEST WINDOW IN BANK MODE
002740  104450  000011                  FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
002744  016700  006312                  MOV     WHORA,R0        ;GET WINDOW ADDRESS
002750  016067  001020  176044          MOV     LIMIT(0),OLIM   ;GET THE SIZE
002756  032767  004000  006302          BIT     #4000,SIZE      ;4K WINDOW?
002764  001026                          BNE     2$              ;BRANCH IF 4K OR LESS
002766  032767  010000  006272          BIT     #10000,SIZE     ;8K WINDOW?
002774  001403                          BEQ     1$              ;BRANCH IF NO
002776  006267  176020                  ASR     OLIM            ;CONVERT BANKS TO WINDOW SEGMENTS
003002  000417                          BR      2$
003004  032767  020000  006254  1$:     BIT     #20000,SIZE     ;16K WINDOW?
003012  001405                          BEQ     3$              ;BRANCH IF NO
003014  006267  176002                  ASR     OLIM            ;CONVERT BANKS TO WINDOW SEGMENTS
003020  006267  175776                  ASR     OLIM
003024  000406                          BR      2$
003026  006267  175770          3$:     ASR     OLIM            ;CONVERT BANKS TO WINDOW SEGMENTS
003032  006267  175764                  ASR     OLIM
003036  006267  175760                  ASR     OLIM
003042                          2$:
003042  104450  000001                  FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
003046  012737  003466  000004          MOV     #NODEV,@#4      ;SET UP TRAP IF NO DEVICE
003054  012706  000500          RTPF:   MOV     #500,SP         ;RESET THE STACK  ** 500 **
003060  005037  177776                  CLR     @#PS            ;DROP PRIORITY
003064  052737  000100  177560          BIS     #100,@#177560   ;IE THE TTY
003072  042777  100000  006136          BIC     #100000,@WCSRA  ;KILL ERROR
003100  052777  000100  006130          BIS     #100,@WCSRA     ;I/E
003106  022767  041522  006736          CMP     #"RC,INPUT      ;CHECK FOR RC11
003114  001010                          BNE     T1              ;TRY NEXT
003116  012777  004164  006252          MOV     #TRP.RC,@RCTRAP ;SET RC VECTOR
003124  052777  000100  006156          BIS     #100,@RCCS      ;TURN ON INTERUPT
003132  000167  005154                  JMP     WAIT            ;GO TO WAIT
003136  022767  050122  006706  T1:     CMP     #"RP,INPUT      ;CHECK FOR RP11
003144  001010                          BNE     T2              ;TRY NEXT
003146  012777  004624  006226          MOV     #TRP.RP,@RPTRAP ;SET RP VECTOR
003154  052777  000100  006140          BIS     #100,@RPCS      ;TURN ON INTERUPT
003162  000167  005124                  JMP     WAIT            ;GO TO WAIT
003166  022767  043122  006656  T2:     CMP     #"RF,INPUT      ;CHECK FOR RF11
003174  001010                          BNE     T3              ;TRY NEXT
003176  012777  003676  006166          MOV     #TRP.RF,@RFTRAP ;SET RF VECTOR
003204  052777  000100  006064          BIS     #100,@RFCS      ;TURN ON INTERUPT
003212  000167  005074                  JMP     WAIT            ;GO TO WAIT
003216  022767  045522  006626  T3:     CMP     #"RK,INPUT      ;CHECK FOR RK11
003224  001010                          BNE     T4              ;TRY NEXT
003226  012777  005004  006152          MOV     #TRP.RK,@RKTRAP ;SET RK VECTOR
003234  052777  000100  006074          BIS     #100,@RKCS      ;TURN ON INTERUPT
```

```
  744  003242  000167  005044                      JMP     WAIT            ;GO TO WAIT
  745  003246  022767  046524  006576  T4:        CMP     #"TM,INPUT      ;CHECK FOR TM11
  746  003254  001010                              BNE     T5              ;TRY NEXT
  747  003256  012777  005772  006132              MOV     #TRP.TM,@TMTRAP ;SET TM VECTOR
  748  003264  052777  000100  006070              BIS     #100,@TMCS      ;TURN ON INTERUPT
  749  003272  000167  005014                      JMP     WAIT            ;GO TO WAIT
  750  003276  022767  041524  006546  T5:        CMP     #"TC,INPUT      ;CHECK FOR TC11
  751  003304  001013                              BNE     T6              ;TRY NEXT
  752  003306  012777  005314  006076              MOV     #TRP.TC,@TCTRAP ;SET TC VECTOR
  753  003314  012767  000010  002436              MOV     #10,TC.TA       ;INITIALIZE STARTING BLOCK
  754  003322  052777  000100  006020              BIS     #100,@TCCS      ;TURN ON INTERUPT
  755  003330  000167  004756                      JMP     WAIT            ;GO TO WAIT
  756  003334  022767  054105  006510  T6:        CMP     #"EX,INPUT      ;CHECK FOR EXECUTE CODE
  757  003342  001005                              BNE     T8              ;GO TO NEXT
  758  003344  012737  010326  000004              MOV     #TRAP4,@#4      ;RESET 4
  759  003352  000167  000162                      JMP     EXCODE          ;EXECUTE THE CODE
  760  003356                           T8:
  761  003356  000004  003362                      TYPE    .+2             ;.ASCIZ <15><12><12>"PC11, RP11, RF11, RK11, TM11, TC11,
  762  003456  005037  177560                      CLR     @#177560        ;ID THE TTY
  763  003462  000167  177200                      JMP     TTYIN           ;LOOP AND TRY AGAIN
```

```
                                ;THIS ROUTINE IS ENTERED IF THE OPERATOR SELECTS A DEVICE WHICH
                                ;DOES NOT EXIST.
                                
                                
003466  005037  177776  NODEV:  CLR     @#PS            ;LOWER PROCESSOR STATUS
003472  000004  003476          TYPE    ,.+2            ;.ASCIZ <15><12><12>
003502  000004  012052          TYPE    ,INPUT          ;TYPE DEVICE NAME
003506  000004  003512          TYPE    ,.+2            ;.ASCIZ " DOES NOT EXIST"<15><12>
003534  000137  002564          JMP     @#REDO          ;RETURN TO TTY INPUT MODE
```

```
                                                ;THE FOLLOWING CODE PERFORMS THE EXECUTE COMMAND. IT WILL MOVE
                                                ;THE LITTLE PROGRAM STARTING AT "CODE" THRU THE WINDOW TO THE OTHER
                                                ;PROCESSOR AND THEN EXECUTE THE CODE IN THE OTHER PROCESSOR'S MEMORY.
                                                ;IT WILL EXECUTE THE CODE IN ALL AVAILABLE BANKS OF THE OTHER PROCESSOR.


003540  012737  000340  177776  EXCODE: MOV     #340,@#PS               ;LOCK UP THE WORLD
003546  104444                          NEXT                            ;GET THE NEXT BANK
003550  104450  000031                  FUNCTN  ,R.W                    ;REQUEST WINDOW IN R.W MODE
003554  016700  005502                  MOV     WADRA,R0                ;GET ADDRESS OF WINDOW
003560  012701  003626                  MOV     #CODE,R1                ;GET ADDRESS OF CODE
003564  012120                  1$:     MOV     (1)+,(0)+               ;MOVE THE WORLD
003566  022701  003672                  CMP     #ENDC+2,R1              ;END?
003572  001374                          BNE     1$                      ;LOOP UNTIL DONE
003574  016701  005462                  MOV     WADRA,R1                ;GET ADDRESS OF WINDOW
003600  062701  000043                  ADD     #ENDC-CODE+1,R1         ;GET ENDC ADDRESS AND ODD
003604  004777  005452                  JSR     PC,@WADRA               ;EXECUTE THE CODE
003610  104450  000001                  FUNCTN  ,CLEAR                  ;REQUEST WINDOW IN CLEAR MODE
003614  005037  177776                  CLR     @#PS                    ;DROP THE PRIO TO 0
003620  004767  005702                  JSR     PC,CONTC                ;CHECK FOR RESTART
003624  000745                          BR      EXCODE                  ;LOOP FOR EVER

                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

003626  105421                  CODE:   NEGB    (1)+                    ;THIS
003630  106041                          RORB    -(1)                    ;CODE
003632  105421                          NEGB    (1)+                    ;IS
003634  106141                          ROLB    -(1)                    ;WORST
003636  106121                          ROLB    (1)+                    ;CASE
003640  105441                          NEGB    -(1)                    ;BUS
003642  106021                          RORB    (1)+                    ;NOISE
003644  105441                          NEGB    -(1)                    ;TO
003646  105421                          NEGB    (1)+                    ;CAUSE
003650  106041                          RORB    -(1)                    ;HAVIC
003652  105421                          NEGB    (1)+                    ;ON
003654  106141                          ROLB    -(1)                    ;THE
003656  106121                          ROLB    (1)+                    ;U
003660  105441                          NEGB    -(1)                    ;NI
003662  106021                          RORB    (1)+                    ;BUS
003664  105441                          NEGB    -(1)                    ;!!!!!!
003666  000207                          RTS     PC                      ;RETURN

003670  000000                  ENDC:   0                               ;DATA BUFFER AREA

                                ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
                                        ;ENTERED WHEN INTERRUPTED BY A READ COMMAND  IT WILL THEN
                                        ;ISSUE A WRITE COMMAND.
                                        
003672  104410                  W.RF:   CHK.RF                  ;CHECK FOR ERRORS
003674  104430                          CKDATA                  ;CHECK CORE
                                        
                                        ;ENTERED HERE INITIALLY BY THE COMMAND INTERPRETER.
                                        
003676  005077  005374          TRP.RF: CLR     @RFCS           ;CLEAR INT ENB
003702  004767  005620                  JSR     PC,CONTC        ;CHECK FOR RESTART
003706  104404                          CORE                    ;LOAD CORE WITH DATA
003710  012777  003746  005454          MOV     #C.RF,@RFTRAP   ;'CHECK' TRAP ADDRESS
003716  016777  005502  005356          MOV     WRDCNT,@RFWC    ;WORD COUNT
003724  005077  005356                  CLR     @RFDA           ;DSK ADDRESS
003730  104450  000011                  FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
003734  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!3 AND GO
003736  011304  000103  011276          @RFCA,100!3,RFCS
003744  000002                          RTI
                                        
                                        ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
                                        ;ISSUE A WRITECHECK COMMAND.
                                        
003746  104410                  C.RF:   CHK.RF                  ;CHECK FOR ERRORS
003750  012777  004006  005414          MOV     #R.RF,@RFTRAP   ;'READ' TRAP ADDRESS
003756  016777  005442  005316          MOV     WRDCNT,@RFWC    ;WORD COUNT
003764  005077  005316                  CLR     @RFDA           ;DSK ADDRESS
003770  104450  000011                  FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
003774  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!7 AND GO
003776  011304  000107  011276          @RFCA,100!7,RFCS
004004  000002                          RTI
                                        
                                        ;ENTERED WHEN INTERRUPTED BY A WRITECHECK COMMAND. IT WILL
                                        ;THEN ISSUE A READ COMMAND.
                                        
004006  104410                  R.RF:   CHK.RF                  ;CHECK FOR ERRORS
004010  104406                          COMCOR                  ;COMPLIMENT CORE
004012  012777  003672  005352          MOV     #W.RF,@RFTRAP   ;'WRITE' TRAP ADDRESS
004020  016777  005400  005254          MOV     WRDCNT,@RFWC    ;WORD COUNT
004026  005077  005254                  CLR     @RFDA           ;DSK ADDRESS
004032  104450  000021                  FUNCTN  ,WRITE          ;REQUEST WINDOW IN WRITE MODE
004036  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!5 AND GO
004040  011304  000105  011276          @RFCA,100!5,RFCS
004046  000002                          RTI
                                        
                                        ;CHECK THE STATUS OF THE PREVIOUS OPERATION.
                                        
004050  005067  005114          CH.RF:  CLR     EX              ;CLEAR ERROR FLAG
004054  104450  000001                  FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
004060  005777  005212                  TST     @RFCS           ;ANY ERRORS?
004064  100034                          BPL     1$              ;BRANCH IF NO ERRORS
004066  005267  005076                  INC     EX              ;SET ERROR FLAG
004072  032737  020000  177570          BIT     #SW13,@#SWR     ;INHIBIT ERROR TYPEOUT
004100  001026                          BNE     1$              ;INHIBIT TYPEOUTS
004102  017705  005170                  MOV     @RFCS,TTY       ;SET TTY FOR TYPING
004106  000004  004112                  TYPE    ,.+2            ;.ASCIZ <15><12>"RF11 ERROR:"
004130  000004  011172                  TYPE    ,M3             ;TYPE CS
```

```
  875  004134  004767  005762                    JSR     PC,PRINTR       ;TYPE RFCS IN OCTAL
  876  004140  000004  011202                    TYPE    ,M4             ;TYPE ER
  877  004144  017705  005130                    MOV     @RFER,TTY       ;TYPE @RFER IN OCTAL
  878  004150  004767  005746                    JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
  879  004154  104434                            STOP                    ;HALT ON ERROR
  880  004156  000002                    1$:     RTI                     ;RETURN
```

```
                                        ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
                                        ;ISSUE A WRITE COMMAND.

004160  104412                  W.RC:   CHK.RC                  ;CHECK FOR ERRORS
004162  104430                          CKDATA                  ;CHECK CORE

                                        ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETE.

004164  005077  005120          TRP.RC: CLR     @RCCS           ;CLEAR INT ENB
004170  004767  005334                  JSR     PC,CONTC        ;CHECK FOR RESTART
004174  104404                          CORE                    ;LOAD CORE WITH DATA
004176  012777  004234  005172          MOV     #C.RC,@RCTRAP   ;'CHECK' TRAP ADDRESS
004204  016777  000214  005102          MOV     WRDCNT,@RCWC    ;WORD COUNT
004212  005077  005102                  CLR     @RCDA           ;DSK ADDRESS
004216  104450  000011                  FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
004222  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!3 AND GO
004224  011316  000103  011310          RCCA,100!3,RCCS
004232  000002                          RTI

                                        ;ENTERED  WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
                                        ;ISSUE A WRITECHECK COMMAND.

004234  104412                  C.RC:   CHK.RC                  ;CHECK FOR ERRORS
004236  012777  004274  005132          MOV     #R.RC,@RCTRAP   ;'READ' TRAP ADDRESS
004244  016777  005154  005042          MOV     WRDCNT,@RCWC    ;WORD COUNT
004252  005077  005042                  CLR     @RCDA           ;DSK ADDRESS
004256  104450  000011                  FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
004262  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!7 AND GO
004264  011316  000107  011310          RCCA,100!7,RCCS
004272  000002                          RTI

                                        ;ENTERED WHEN INTERRUPTED BY A WRITECHECK COMMAND. IT WILL THEN
                                        ;ISSUE A READ COMMAND.

004274  104412                  R.RC:   CHK.RC                  ;CHECK FOR ERRORS
004276  104406                          COMCOR                  ;COMPLIMENT CORE
004300  012777  004160  005070          MOV     #W.RC,@RCTRAP   ;'WRITE' TRAP ADDRESS
004306  016777  005112  005000          MOV     WRDCNT,@RCWC    ;WORD COUNT
004314  005077  005000                  CLR     @RCDA           ;DSK ADDRESS
004320  104450  000021                  FUNCTN  ,WRITE          ;REQUEST WINDOW IN WRITE MODE
004324  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!5 AND GO
004326  011316  000105  011310          RCCA,100!5,RCCS
004334  000002                          RTI

                                        ;CHECK THE STATUS OF THE PREVIOUS OPERATION.

004336  005067  004626          CH.RC:  CLR     EX              ;CLEAR ERROR FLAG
004342  104450  000001                  FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
004346  105777  004736                  TST     @RCCS           ;ANY ERRORS?
004352  100034                          BPL     1$              ;BRANCH IF NO ERRORS
004354  005267  004610                  INC     EX              ;SET ERROR FLAG
004360  012737  020000  177570          BIT     #SW13,@#SWR     ;INHIBIT ERROR TYPEOUT
004366  001026                          BNE     1$              ;INHIBIT TYPEOUTS
004370  013705  004714                  MOV     @RCCS,TTY       ;SET TTY FOR TYPING
004374  104004  004400                  TYPE    ,.+2            ;.ASCIZ <15><12>"RC11 ERROR:"
004416  104004  011172                  TYPE    ,M3             ;TYPE CS
```

```
937  004422  004767  005474                    JSR     PC,PRINTR       ;TYPE RCCS IN OCTAL
938  004426  000004  011202                    TYPE    ,M4             ;TYPE ER
939  004432  017705  004654                    MOV     @RCER,TTY       ;TYPE @RCER IN OCTAL
940  004436  004767  005460                    JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
941  004442  104434                            STOP                    ;HALT ON ERROR
942  004444  000002                    1$:     RTI                     ;RETURN
```

```
                                        ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
                                        ;ISSUE A WRITE COMMAND.

004446  104414                  W.RP:   CHK.RP                  ;CHECK FOR ERRORS
004450  104430                          CKDATA                  ;CHECK CORE
004452  005077  004644          TP.RP:  CLR     @RPCS           ;CLEAR INT ENB
004456  004767  005044                  JSR     PC,CONTC        ;CHECK FOR RESTART
004462  104404                          CORE                    ;LOAD CORE WITH DATA
004464  012777  004522  004710          MOV     #C.RP,@RPTRAP   ;'CHECK' TRAP ADDRESS
004472  016777  004726  004624          MOV     WRDCNT,@RPWC    ;WORD COUNT
004500  005077  004624                  CLR     @RPDA           ;DSK ADDRESS
004504  104450  000011                  FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
004510  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!3 AND GO
004512  011326  000103  011322          RPCA,100!3,RPCS
004520  000002                          RTI

                                        ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
                                        ;ISSUE A WRITECHECK COMMAND.

004522  104414                  C.RP:   CHK.RP                  ;CHECK FOR ERRORS
004524  012777  004562  004450          MOV     #R.RP,@RPTRAP   ;'READ' TRAP ADDRESS
004532  016777  004666  004564          MOV     WRDCNT,@RPWC    ;WORD COUNT
004540  005077  004564                  CLR     @RPDA           ;DSK ADDRESS
004544  104450  000011                  FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
004550  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!7 AND GO
004552  011326  000107  011322          RPCA,100!7,RPCS
004560  000002                          RTI

                                        ;ENTERED WHEN INTERRUPTED BY A WRITECHECK COMMAND. IT WILL THEN
                                        ;ISSUE A READ COMMAND.

004562  104414                  R.RP:   CHK.RP                  ;CHECK FOR ERRORS
004564  104406                          COMCOR                  ;COMPLIMENT CORE
004566  012777  004446  004606          MOV     #W.RP,@RPTRAP   ;'WRITE' TRAP ADDRESS
004574  016777  004624  004522          MOV     WRDCNT,@RPWC    ;WORD COUNT
004602  005077  004522                  CLR     @RPDA           ;DSK ADDRESS
004606  104450  000021                  FUNCTN  ,WRITE          ;REQUEST WINDOW IN WRITE MODE
004612  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!5 AND GO
004614  011326  000105  011322          RPCA,100!5,RPCS
004622  000002                          RTI

                                        ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETER.

004624  012777  000011  004470  TRP.RP: MOV     #11,@RPCS       ;INITIALIZE
004632  105777  004464                  TSTB    @RPCS           ;ROUTINE
004636  100375                          BPL     .-4             ;FOR
004640  005777  004470                  TST     @RPDS           ;THE
004644  100375                          BPL     .-4             ;RP11
004646  005077  004450                  CLR     @RPCS           ;DISK
004652  000677                          BR      TP.RP           ;PACK
```

```
  993                                   ;CHECK THE STATUS OF THE PREVIOUS OPERATION.
  994
  995  004654  005067  004310          CH.RP:  CLR     EX              ;CLEAR ERROR FLAG
  996  004660  104450  000001                  FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
  997  004664  005777  004432                  TST     @RPCS           ;ANY ERRORS?
  998  004670  100002                          BPL     1$              ;BRANCH IF NO ERRORS
  999  004672  005267  004272                  INC     EX              ;SET ERROR FLAG
 1000  004676  032737  020000  177570          BIT     #SW13,@#SWR     ;INHIBIT ERROR TYPEOUT
 1001  004704  001034                          BNE     1$              ;INHIBIT TYPEOUTS
 1002  004706  017705  004410                  MOV     @RPCS,TTY       ;SET TTY FOR TYPING
 1003  004712  000004  004716                  TYPE    ,..+2           ;.ASCIZ <15><12>"RP11 ERROR:"
 1004  004734  000004  011172                  TYPE    ,M3             ;TYPE CS
 1005  004740  004767  005156                  JSR     PC,PRINTR       ;TYPE RPCS IN OCTAL
 1006  004744  000004  011202                  TYPE    ,M4             ;TYPE ER
 1007  004750  017705  004356                  MOV     @RPER,TTY       ;TYPE @RPER IN OCTAL
 1008  004754  004767  005142                  JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
 1009  004760  104434                          STOP                    ;HALT ON ERROR
 1010  004762  012777  000001  004332          MOV     #1,@RPCS        ;LOAD A CLEAR
 1011  004770  105777  004326                  TSTB    @RPCS           ;WAIT FOR
 1012  004774  100375                          BPL     .-4             ;DONE
 1013  004776  000002                  1$:     RTI                     ;RETURN
```

```
                                        ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
                                        ;ISSUE A WRITE COMMAND.

005000  104416                  W.RK:   CHK.RK                  ;CHECK FOR ERRORS
005002  104430                          CKDATA                  ;CHECK CORE

                                        ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETER.

005004  005077  004326          TRP.RK: CLR     @RKCS           ;CLEAR INT ENB
005010  004767  004512                  JSR     PC,CONTC        ;CHECK FOR RESTART
005014  104404                          CORE                    ;LOAD CORE WITH DATA
005016  012777  005054  004362          MOV     #C.RK,@RKTRAP   ;'CHECK' TRAP ADDRESS
005024  016777  004374  004310          MOV     WRDCNT,@RKWC    ;WORD COUNT
005032  005077  004310                  CLR     @RKDA           ;DSK ADDRESS
005036  104450  000011                  FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
005042  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!3 AND GO
005044  011344  000103  011336          RKCA,100!3,RKCS
005052  000002                          RTI

                                        ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL
                                        ;THEN ISSUE A WRITE CHECK COMMAND.

005054  104416                  C.RK:   CHK.RK                  ;CHECK FOR ERRORS
005056  012777  005114  004322          MOV     #R.RK,@RKTRAP   ;'READ' TRAP ADDRESS
005064  016777  004334  004250          MOV     WRDCNT,@RKWC    ;WORD COUNT
005072  005077  004250                  CLR     @RKDA           ;DSK ADDRESS
005076  104450  000011                  FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
005102  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!7 AND GO
005104  011344  000107  011336          RKCA,100!7,RKCS
005112  000002                          RTI

                                        ;ENTERED WHEN INTERRUPTED BY A WRITECHECK COMMAND. IT WILL THEN
                                        ;ISSUE A READ COMMAND.

005114  104416                  R.RK:   CHK.RK                  ;CHECK FOR ERRORS
005116  104406                          COMCOR                  ;COMPLIMENT CORE
005120  012777  005000  004260          MOV     #W.RK,@RKTRAP   ;'WRITE' TRAP ADDRESS
005126  016777  004272  004206          MOV     WRDCNT,@RKWC    ;WORD COUNT
005134  005077  004206                  CLR     @RKDA           ;DSK ADDRESS
005140  104450  000021                  FUNCTN  ,WRITE          ;REQUEST WINDOW IN WRITE MODE
005144  104440                          .START                  ;LOAD CURRENT ADDRESS & IE!5 AND GO
005146  011344  000105  011336          RKCA,100!5,RKCS
005154  000002                          RTI

                                        ;CHECK THE STATUS OF THE PREVIOUS OPERATION.

005156  005067  004006          CH.RK:  CLR     EX              ;CLEAR ERROR FLAG
005162  104450  000001                  FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
005166  005777  004144                  TST     @RKCS           ;ANY ERRORS?
005172  100042                          BPL     1$              ;BRANCH IF NO ERRORS
005174  005267  003770                  INC     EX              ;SET ERROR FLAG
005200  032737  020000  177570          BIT     #SW13,@#SWR     ;INHIBIT ERROR TYPEOUT
005206  001034                          BNE     1$              ;INHIBIT TYPEOUTS
005210  017705  004122                  MOV     @RKCS,TTY       ;SET TTY FOR TYPING
005214  000004  005220                  TYPE    ,.+2            ;.ASCIZ <15><12>"RK11 ERROR:"
005236  000004  011172                  TYPE    ,M3             ;TYPE CS
```

F03

```
MAINDEC-11-DZDAB-A      BUS WINDOW EXERCISER    MACY11 27(732)  05-OCT-76  11:05  PAGE 32
DZDABA.P11              ERROR ROUTINE FOR RK11

 1070  015242  004767  004654                    JSR     PC,PRINTR       ;TYPE RKCS IN OCTAL
 1071  015246  000004  011202                    TYPE    ,M4             ;TYPE ER
 1072  015252  017705  004062                    MOV     @RKER,TTY       ;TYPE @RKER IN OCTAL
 1073  015256  004767  004640                    JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
 1074  015262  104434                            STOP                    ;HALT ON ERROR
 1075  015264  012777  000001  004044            MOV     #1,@RKCS        ;LOAD A CLEAR
 1076  015272  105777  004040                    TSTB    @RKCS           ;WAIT FOR
 1077  015276  100375                            BPL     .-4             ;DONE
 1078  015300  000002                    1$:     RTI                     ;RETURN
```

```
1079                                        ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
1080                                        ;ISSUE A WRITE COMMAND.
1081
1082  005302  104420                        W.TC:   CHK.TC                  ;CHECK FOR ERRORS
1083  005304  104430                                CKDATA                  ;CHECK CORE
1084  005306  012777  000001  004034                MOV     #1,@TCCS        ;STOP ALL TRANSPORTS
1085
1086                                        ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETER.
1087
1088  005314  004767  004206                TRP.TC: JSR     PC,CONTC        ;CHECK FOR RESTART
1089  005320  062767  000001  000432                ADD     #1,TC.TA        ;UPDATE BLOCK
1090  005326  022767  000400  000424                CMP     #400,TC.TA
1091  005334  003003                               BGT     1$
1092  005336  012767  000010  000414                MOV     #10,TC.TA
1093  005344  104404                        1$:     CORE                    ;LOAD WORST CASE NOISE
1094  005346  104450  000011                        FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
1095  005352  104424  000010                        SEARCH  ,10             ;SEARCH FOR BLOCK
1096  005356  012777  005404  004026                MOV     #R.TC,@TCTRAP   ;TRAP ADDRESS
1097  005364  016777  004034  003762                MOV     WRDCNT,@TCWC    ;WORD COUNT
1098  005372  104440                               .START
1099  005374  011356  000115  011350                TCCA,115,TCCS
1100  005402  000002                               RTI
1101
1102                                        ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
1103                                        ;ISSUE A READ COMMAND.
1104
1105  005404  104420                        R.TC:   CHK.TC                  ;CHECK FOR ERROR
1106  005406  104406                                COMCOR                  ;COMPLIMENT CORE
1107  005410  104450  000021                        FUNCTN  ,WRITE          ;REQUEST WINDOW IN WRITE MODE
1108  005414  104424  000010                        SEARCH  ,10             ;SEARCH FOR BLOCK
1109  005420  012777  005302  003764                MOV     #W.TC,@TCTRAP   ;TRAP ADDRESS
1110  005426  016777  003772  003720                MOV     WRDCNT,@TCWC    ;WORD COUNT
1111  005434  104440                               .START
1112  005436  011356  000105  011350                TCCA,105,TCCS
1113  005444  000002                               RTI
1114
1115                                        ;CHECK THE STATUS OF THE PREVIOUS OPERATION.
1116
1117  005446  005067  003516                CH.TC:  CLR     EX              ;CLEAR ERROR FLAG
1118  005452  104450  000001                        FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
1119  005456  005777  003666                        TST     @TCCS           ;ANY ERRORS?
1120  005462  100042                               BPL     1$              ;BRANCH IF NO ERRORS
1121  005464  005267  003500                        INC     EX              ;SET ERROR FLAG
1122  005470  032737  020000  177570                BIT     #SW13,@#SWR     ;INHIBIT ERROR TYPEOUT
1123  005476  001034                               BNE     1$              ;INHIBIT TYPEOUTS
1124  005500  017705  003644                        MOV     @TCCS,TTY       ;SET TTY FOR TYPING
1125  005504  000004  005510                        TYPE    ,.+2            ;.ASCIZ <15><12>"TC11 ERROR:"
1126  005526  000004  011172                        TYPE    ,M3             ;TYPE CS
1127  005532  004767  004364                        JSR     PC,PRINTR       ;TYPE TCCS IN OCTAL
1128  005536  000004  011202                        TYPE    ,M4             ;TYPE ER
1129  005542  017705  003604                        MOV     @TCER,TTY       ;TYPE @TCER IN OCTAL
1130  005546  004767  004350                        JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
1131  005552  104434                               STOP                    ;HALT ON ERROR
1132  005554  012777  000001  003566                MOV     #1,@TCCS        ;LOAD A CLEAR
1133  005562  105777  003562                        TSTB    @TCCS           ;WAIT FOR
1134  005566  100375                               BPL     .-4             ;DONE
```

```
 1135  005570  000002                  1$:     RTI                     ;RETURN
```

```
1136                                        ;THE FOLLOWING ROUTINE WILL SEARCH FOR THE DESIGNATED
1137                                        ;DEC-TAPE BLOCK. THE DESIRED BLOCK IS STORED IN "TC.TA".
1138
1139  005572  062716  000002          S.TC:   ADD     #2,(6)
1140  005576  012777  005636  003606          MOV     #SC1,@TCTRAP    ;TRAP ADDRESS
1141  005604  016767  000150  000144          MOV     TC.TA,WANT
1142  005612  162767  000002  000136          SUB     #2,WANT         ;2 PAST BLOCK
1143  005620  012667  000140                  MOV     (6)+,RETURN     ;SAVE RETURN ADDRESS
1144  005624  005726                          TST     (6)+
1145  005626  012777  004103  003514  RETI1:  MOV     #4103,@TCCS     ;SEARCH REVERSE
1146  005634  000002                          RTI
1147
1148  005636  005777  003506          SC1:    TST     @TCCS           ;CHECK FOR ERROR
1149  005642  100010                          BPL     SC1A            ;NO ERRORS
1150  005644  005777  003502                  TST     @TCER           ;CHECK FOR END OF TAPE
1151  005650  100404                          BMI     1$
1152  005652  104436                          ERR3
1153  005654  005767  003304                  TST     CFLAG           ;IS ^C TYPED?
1154  005660  001034                          BNE     SCQ1            ;BRANCH IF TYPED
1155  005662  000405                  1$:     BR      SCF1
1156
1157  005664  026777  000066  003466  SC1A:   CMP     WANT,@TCDB      ;ON BLOCK?
1158  005672  003001                          BGT     SCF1            ;GO REVERSE
1159  005674  000754                          BR      RETI1           ;CONTINUE AS YOU WERE
1160
1161  005676  012777  005714  003506  SCF1:   MOV     #SCF2,@TCTRAP   ;TRAP ADDRESS
1162  005704  012777  000103  003436  RETR1:  MOV     #103,@TCCS      ;SEARCH FOWARD
1163  005712  000002                          RTI
1164
1165  005714  005777  003430          SCF2:   TST     @TCCS           ;ERROR?
1166  005720  100010                          BPL     SCF1A           ;NO ERRORS
1167  005722  005777  003424                  TST     @TCER           ;END OF TAPE?
1168  005726  100404                          BMI     1$
1169  005730  104432                          ERR2                    ;NO
1170  005732  005767  003226                  TST     CFLAG           ;IS ^C TYPED?
1171  005736  001005                          BNE     SCQ1            ;BRANCH IF SET
1172  005740  000732                  1$:     BR      RETI1           ;SWITCH DIRECTIONS
1173
1174  005742  026777  000012  003410  SCF1A:  CMP     TC.TA,@TCDB     ;RIGHT BLOCK?
1175  005750  001355                          BNE     RETR1           ;NO
1176  005752  000177  000006          SCQ1:   JMP     @RETURN
1177
1178  005756  000000                  WANT:   0
1179  005760  000000                  TC.TA:  0
1180  005762  000000                  TC.WC:  0
1181  005764  000000                  RETURN: 0
```

```
1182                                        ;ENTERED WHEN INTERRUPTED BY A READ COMMAND. IT WILL THEN
1183                                        ;ISSUE A WRITE COMMAND.
1184
1185  005766  104422                        W.TM:   CHK.TM                  ;CHECK FOR ERROR
1186  005770  104430                                CKDATA
1187
1188                                        ;ENTERED HERE INITIALLY FROM THE COMMAND INTERPRETER.
1189
1190  005772  005077  003364                TRP.TM: CLR     @TMCS           ;CLEAR INT ENB
1191  005776  004767  003524                        JSR     PC,CONTC        ;CHECK FOR RESTART
1192  006002  104404                                CORE                    ;SET UP CORE
1193  006004  104426                                REWIND
1194  006006  012777  006074  003402                MOV     #R.TM,@TMTRAP   ;TRAP ADDRESS
1195  006014  016777  003404  003344                MOV     WRDCNT,@TMBC    ;BYTE COUNT
1196  006022  006377  003340                       ASL     @TMBC           ;MAKE BYTE COUNT
1197  006026  104450  000011                       FUNCTN  ,READ           ;REQUEST WINDOW IN READ MODE
1198  006032  104440                                .START                  ;LOAD CURRENT ADDRESS & IE!60005 AND GO
1199  006034  011370  060105  011362                TMCA,100!60005,TMCS
1200  006042  000002                                RTI
1201
1202                                        ;REWIND THE TAPE
1203
1204  006044  012777  006070  003344        RW.TM:  MOV     #X.TM,@TMTRAP   ;TRAP ADDRESS
1205  006052  012667  000062                        MOV     (6)+,TMRET      ;SAVE ADDRESS
1206  006056  005726                                TST     (6)+            ;RESTORE STACK
1207  006060  012777  060117  003274                MOV     #60117,@TMCS    ;REWIND
1208  006066  000002                                RTI
1209
1210  006070  000177  000044                X.TM:   JMP     @TMRET
1211
1212                                        ;ENTERED WHEN INTERRUPTED BY A WRITE COMMAND. IT WILL THEN
1213                                        ;ISSUE A READ COMMAND.
1214
1215  006074  104422                        R.TM:   CHK.TM                  ;ANY ERRORS?
1216  006076  104406                                COMCOR                  ;COMPLIMENT CORE
1217  006100  104426                                REWIND
1218  006102  012777  005766  003306                MOV     #W.TM,@TMTRAP   ;TRAP ADDRESS
1219  006110  016777  003310  003250                MOV     WRDCNT,@TMBC    ;BYTE COUNT
1220  006116  006377  003244                       ASL     @TMBC           ;MAKE BYTE COUNT
1221  006122  104450  000021                       FUNCTN  ,WRITE          ;REQUEST WINDOW IN WRITE MODE
1222  006126  104440                                .START                  ;LOAD CURRENT ADDRESS & IE!60003 AND GO
1223  006130  011370  060103  011362                TMCA,100!60003,TMCS
1224  006136  000002                                RTI
1225
1226  006140  000000                        TMRET:  0
1227
1228                                        ;COME HERE TO CHECK THE STATUS OF THE PREVIOUS OPERATION.
1229
1230  006142  005067  003022                CH.TM:  CLR     EX              ;CLEAR ERROR FLAG
1231  006146  104450  000001                        FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
1232  006152  005777  003204                        TST     @TMCS           ;ANY ERRORS?
1233  006156  100034                                BPL     1$              ;BRANCH IF NO ERRORS
1234  006160  005267  003004                       INC     EX              ;SET ERROR FLAG
1235  006164  032737  020000  177570                BIT     #SW13,@#SWR     ;INHIBIT ERROR TYPEOUT
1236  006172  001026                                BNE     1$              ;INHIBIT TYPEOUTS
1237  006174  017705  003162                       MOV     @TMCS,TTY       ;SET TTY FOR TYPING
```

```
1238  006200  000004  006204                    TYPE    ,.+2            ;.ASCIZ <15><12>"TM11 ERROR:"
1239  006222  000004  011172                    TYPE    ,M3             ;TYPE CS
1240  006226  004767  003670                    JSR     PC,PRINTR       ;TYPE TMCS IN OCTAL
1241  006232  000004  011202                    TYPE    ,M4             ;TYPE ER
1242  006236  017705  003122                    MOV     @TMER,TTY       ;TYPE @TMER IN OCTAL
1243  006242  004767  003654                    JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
1244  006246  104434                            STOP                    ;HALT ON ERROR
1245  006250  000002                    1$:     RTI                     ;RETURN
```

```
                                        ;       $TRAP           ;TRAP HANDLER
                                        ;
                                        ;THIS ROUTINE DECODES A TRAP CALL AND JUMPS TO THE APROPRATE
                                        ;SUBROUTINE. THE CALL IS A "TRAP+N" WHERE N IS A MULTIPLE OF 2.
                                        ;THE "SET" MACRO WILL CREATE THE TABLE NEEDED. IT HAS TO
                                        ;FOLLOW THIS MACRO.
                                        ;
006252  011605                  TRAP$:  MOV     (6),R5          ;GET TRAP INSTRUCTION
006254  162705  000002                  SUB     #2,R5           ;BACK UP BY 2
006260  111505                          MOVB    (5),R5          ;GET THE RIGHT BYTE OF TRAP
006262  062705  006304                  ADD     #ADRTAB,R5      ;INDEX TO TABLE
006266  022705  006354                  CMP     #ENDTAB,R5      ;CHECK FOR END OF TABLE
006272  100401                          BMI     .+4             ;OUT OF BOUNDS
006274  000175  000000                  JMP     @(5)            ;GO TO ROUTINE
006300  000000                          HALT                    ;TRAP OUT OF BOUNDS
006302  000776                          BR      .-2             ;HANG UP
                                        ;
006304                          ADRTAB:
006304  012506                          PATCH1          ;DUM1   =  TRAP+0       (104400)
006306  012526                          PATCH2          ;DUM2   =  TRAP+2       (104402)
006310  007446                          WRADR           ;CORE   =  TRAP+4       (104404)
006312  007724                          WRCADR          ;COMCOR =  TRAP+6       (104406)
006314  004050                          CH.RF           ;CHK.RF =  TRAP+10      (104410)
006316  004336                          CH.RC           ;CHK.RC =  TRAP+12      (104412)
006320  004654                          CH.RP           ;CHK.RP =  TRAP+14      (104414)
006322  005156                          CH.RK           ;CHK.RK =  TRAP+16      (104416)
006324  005446                          CH.TC           ;CHK.TC =  TRAP+20      (104420)
006326  006142                          CH.TM           ;CHK.TM =  TRAP+22      (104422)
006330  005572                          S.TC            ;SEARCH =  TRAP+24      (104424)
006332  006044                          RW.TM           ;REWIND =  TRAP+26      (104426)
006334  007140                          CDATA           ;CKDATA =  TRAP+30      (104430)
006336  005446                          CH.TC           ;ERR2   =  TRAP+32      (104432)
006340  007336                          POTS            ;STOP   =  TRAP+34      (104434)
006342  007754                          ER3             ;ERR3   =  TRAP+36      (104436)
006344  007354                          .BEGIN          ;.START =  TRAP+40      (104440)
006346  010122                          SETBAK          ;SETBAK =  TRAP+42      (104442)
006350  006356                          NXT             ;NEXT   =  TRAP+44      (104444)
006352  010244                          MAP             ;MAPIT  =  TRAP+46      (104446)
006354  006602                          DOFUN           ;FUNCTN =  TRAP+50      (104450)
006354                  ENDTAB=.-2
```

```
1286                                        ;THIS ROUTINE WILL DETERMINE WHICH WINDOW SEGMENT WILL BE TESTED
1287                                        ;NEXT. LOCATION "OLIM" CONTAINS THE NUMBER OF WINDOW SEGMENTS
1288                                        ;AVAILABLE IN THE OTHER PROCESSOR. WHEN EXITED, LOCATION "MX"
1289                                        ;WILL CONTAIN THE NUMBER OF THE SEGMENT TO BE TESTED.
1290
1291  006356  032737  040000  177570  NXT:  BIT   #SW14,@#SWR    ;LOOP ON BANK?
1292  006364  001104                        BNE   2$            ;SKIP IF LOOP
1293  006366  026767  172422  172426        CMP   MX,OLIM       ;LAST PAGE?
1294  006374  002453                        BLT   1$            ;SKIP IF NOT LAST PAGE
1295  006376  032737  002000  177570        BIT   #SW10,@#SWR   ;INHIBIT BELL
1296  006404  001002                        BNE   5$            ;YES!
1297  006406  000004  000007              TYPE  ,BELL           ;RING THE BELL
1298  006412  016767  172400  172374  5$:  MOV   MIN,MX        ;RESET MX
1299  006420  032767  004000  002640        BIT   #4000,SIZE    ;4K WINDOW OR LESS?
1300  006426  001402                        BEQ   8$            ;BRANCH IF GREATER
1301  006430  005267  172360              INC   MX
1302  006434  016737  172354  177570  8$:  MOV   MX,@#DISPLAY  ;DISPLAY IT
1303  006442  032737  000400  177570        BIT   #SW8,@#SWR    ;TYPE BANK NUMBER??
1304  006450  001410                        BEQ   6$            ;NO!
1305  006452  000004  006456              TYPE  ,.+2            ;.ASCIZ <15><12>" "
1306  006462  016705  172326              MOV   MX,TTY          ;TYPE MX IN OCTAL
1307  006466  004767  003440              JSR   PC,PRINTS       ;AND SUPRESS LEADING ZERO'S
1308  006472  005767  000746        6$:   TST   MEMORY          ;CHECK STATE OF FLAG
1309  006476  100407                        BMI   4$            ;SWAB IT
1310  006500  001403                        BEQ   3$            ;COM IT
1311  006502  005067  000736              CLR   MEMORY          ;INIT IT
1312  006506  000433                        BR    2$            ;CONTINUE
1313  006510  105167  000731        3$:   COMB  MEMORY+1        ;MAKE IT 177400
1314  006514  000430                        BR    2$            ;SKIP
1315  006516  000367  000722        4$:   SWAB  MEMORY          ;MAKE IT 377
1316  006522  000425                        BR    2$
1317  006524  005267  172264        1$:   INC   MX              ;GO TO NEXT BANK
1318  006530  016737  172260  177570       MOV   MX,@#DISPLAY  ;DISPLAY THE BANK IN USE
1319  006536  032737  000400  177570       BIT   #SW8,@#SWR    ;TYPE BANK NUMBER??
1320  006544  001414                        BEQ   2$            ;NO!
1321  006546  000004  006552              TYPE  ,.+2            ;.ASCIZ " "
1322  006554  016705  172234              MOV   MX,TTY          ;SET FOR TYPING
1323  006560  032705  000007              BIT   #7,TTY          ;MULTIPLE OF 10?
1324  006564  001402                        BEQ   7$            ;YES
1325  006566  042705  177770              BIC   #177770,TTY     ;CLEAR OUT UPPER BITS
1326  006572  004767  003334        7$:   JSR   PC,PRINTS       ;TYPE MX AND SUPPRESS LEADING ZERCES
1327  006576  104442              2$:   SETBANK                 ;SET MXC
1328  006600  000002                    RTI
```

```
1329                                    ;THIS ROUTINE WILL REQIEST THE OTHER PROCESSOR TO SETUP THE WINDOW
1330                                    ;IN RESPONSE TO A CODE IN THE COMMUNICATION BITS OF THE CSR. THE
1331                                    ;RESPONSE OF THE OTHER WINDOW IS THEN VERIFIED; AND IF IT IS WRONG
1332                                    ;AN ERROR IS REPORTED. OPERATIONS THROUGH THE WINDOW ARE RESTRICTED
1333                                    ;SO THAT EACH PROCESSOR CAN'T HAVE READ PRIVELGES AT THE SAME TIME.
1334                                    ;ONE OF THE PROCESSORS IS CONSIDERED A SLAVE TO THE OTHER. THIS IS
1335                                    ;NECESSARY TO AVOID RACE CONDITIONS WHEN REQUESTING THE WINDOW.
1336                                    ;WHEN THE SLAVE PROCESSOR IS IN THIS ROUTINE IT WILL LOWER ITS
1337                                    ;PRIORITY LEVEL TO 0 TO GIVE THE OTHER PROCESSOR THE RIGHT OF WAY.
1338                                    ;WHEN THE MASTER PROCESSOR IS IN THIS ROUTINE, HIS PRIORITY LEVEL
1339                                    ;IS KEPT AT 7. THE STARTING PROCEDURE DETERMINES WHICH PROCESSOR
1340                                    ;IS THE MASTER.
1341
1342  006602  005037  177776        DOFUN:  CLR     @#PS            ;DROP PRIORITY
1343  006606  005767  002462                TST     NOTRD           ;IS THE READ FLAG SET?
1344  006612  001410                        BEQ     3$              ;DO IT IF NO READ
1345  006614  032776  000010  000000        BIT     #10,@(6)        ;IS IT A READ REQUEST?
1346  006622  001404                        BEQ     3$              ;NO - DO IT
1347  006624  032777  000400  002404  2$:   BIT     #400,@WCSRA     ;R/W?
1348  006632  001374                        BNE     2$              ;HANG UNTIL DONE
1349  006634  012737  000340  177776  3$:   MOV     #340,@#PS       ;LOCK UP THE WORLD
1350  006642  152777  000071  002366        BISB    #71,@WCSRA      ;SEND FUNCTION GET READY
1351  006650  032777  000001  002360  DOF1:  BIT     #1,@WCSRA       ;WAIT FOR NEW DATA TO GO AWAY
1352  006656  001403                        BEQ     DOF7            ;BRANCH IF GONE
1353  006660  005037  177776        DOF2:   CLR     @#PS            ;LOWER STATUS IF SLAVE
1354  006664  000771                        BR      DOF1            ;WAIT
1355  006666  142777  000170  002342  DOF7:  BICB    #70,@WCSRA      ;CLEAR THE FUNCTION BITS
1356  006674  157677  000100  002334        BISB    @(6),@WCSRA     ;SET NEW FUNCTION BITS
1357  006702  032777  000001  002326  1$:   BIT     #1,@WCSRA       ;WAIT FOR NEW DATA
1358  006710  001374                        BNE     1$              ;HANG
1359  006712  017604  000000                MOV     @(6),R4         ;GET THE FUNCTION BITS
1360  006716  010467  000214                MOV     R4,SAVTI?       ;REMEMBER OLD FUNCTION
1361  006722  006204                        ASR     R4              ;SHIFT
1362  006724  006204                        ASR     R4              ;TWICE
1363  006726  017705  002314                MOV     @WCSRA,R5       ;GET WCSRA
1364  006732  042705  1775?3                BIC     #17?573,R5      ;CLEAR JUNK
1365  006736  026405  007116                CMP     CSRTAB(R4),R5   ;RIGHT RESPONCE?
1366  006742  001452                        BEQ     DOF3            ;SKIP IF OK
1367  006744  012737  000340  177776        MOV     #340,@#PS
1368  006752  010401                        MOV     R4,R1
1369  006754  010500                        MOV     R5,R0
1370  006756  000004  006762                TYPE    ,.+2            ;.ASCIZ <15><12><12>"FUNCTION IS WRONG:  SENT = "
1371  007022  006301                        ASL     R1              ;GET
1372  007024  000261                        SEC                     ;IT
1373  007026  006101                        ROL     R1              ;BACK
1374  007030  010105                        MOV     R1,TTY          ;TYPE R1 IN OCTAL
1375  007032  004767  003074                JSR     PC,PRINTS       ;AND SUPRESS LEADING ZERO'S
1376  007036  000004  007042                TYPE    ,.+2            ;.ASCIZ "  WCSRA = "
1377  007056  017705  002154                MOV     @WCSRA,TTY      ;TYPE @WCSRA IN OCTAL
1378  007062  004767  003034                JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
1379  007066  000000                        HALT                    ;FUNCTION NOT RIGHT!
1380  007070  005767  002200        DOF3:   TST     NOTRD           ;IS THE MASTER READING?
1381  007074  001405                        BEQ     DOF4            ;BRANCH IF NO
1382  007076  032776  000010  000000        BIT     #10,@(6)        ;IS SLAVE TRYING TO READ?
1383  007104  001401                        BEQ     DOF4            ;BRANCH IF NO
1384  007106  000770                DOF5:   BR      DOF3            ;WAIT IF SLAVE
```

```
1385  007110  062716  000002          DOF4:   AUD     #2,(6)          ;UPDATE THE RETURN ADDRESS
1386  007114  000002                          RTI                     ;RETURN
1387
1388  007116  000000                  CSRTAB: 0                       ;CLEAR
1389  007120  000200                          200                     ;READ
1390  007122  000204                          204                     ;WRITE
1391  007124  000204                          204                     ;READ/WRITE
1392  007126  000000  000000  000000          0,0,0                   ;N/A
1393  007134  000000                          0                       ;GET READY
1394
1395  007136  000000                  SAVTMP: 0
```

```
1396                                    ;THIS ROUTINE WILL COMPARE THE DATA READ FROM A DEVICE AGAINST THE
1397                                    ;KNOWN PATTERN. ANY ERRORS ARE REPORTED AND THE ROUTINE WILL CONTINUE
1398                                    ;TO SCAN THE BUFFER.
1399
1400  007140  005767  002024      CDATA:  TST     EX              ;CHECK DATA
1401  007144  001401                      BEQ     .+4             ;SKIP IF NO ERROR
1402  007146  000002                      RTI                     ;RETURN IF ERROR EXISTS
1403  007150  104450  000031              FUNCTN  ,R.W            ;REQUEST WINDOW IN R.W MODE
1404  007154  005067  000260              CLR     DATA            ;SET UP DATA
1405  007160  005067  000256              CLR     BIT             ;CLEAR FLAG
1406  007164  016702  002072              MOV     WADRA,R2        ;SET UP BEGINNING OF BUFFER
1407  007170  016705  002230              MOV     WRDCNT,TTY      ;SET UP COUNT
1408  007174  004767  000032      1$:     JSR     PC,GDATA        ;GET THE DATA
1409  007200  026712  000234              CMP     DATA,(2)        ;SAME?
1410  007204  001401                      BEQ     .+4             ;SKIP IF OK
1411  007206  104436                      ERR3                    ;DATA IS WRONG - TYPE IT
1412  007210  005722                      TST     (2)+            ;ADD 2
1413  007212  005767  001746              TST     CFLAG           ;IS ^C TYPED?
1414  007216  001002                      BNE     2$              ;BRANCH IF TYPED
1415  007220  005205                      INC     TTY             ;DEC THE COUNT
1416  007222  001364                      BNE     1$              ;LOOP IF NOT END
1417  007224                      2$:
1418  007224  104450  000001              FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
1419  007230  000002                      RTI                     ;RETURN
1420
1421                                    ;THIS ROUTINE WILL GENERATE THE REQUESTED PATTERN.
1422
1423  007232  005767  000206      GDATA:  TST     MEMORY          ;CHECK FLAG
1424  007236  100430                      BMI     T3XOR9          ;MM11G
1425  007240  003015                      BGT     TBXOR13         ;MM11F
1426
1427  007242  010203              T1XOR8: MOV     R2,R3           ;MM11E MEMORY
1428  007244  000303                      SWAB    R3
1429  007246  006103                      ROL     R3
1430  007250  060203                      ADD     R2,R3
1431  007252  006003                      ROR     R3
1432  007254  066703  000162      ALLOR:  ADD     BIT,R3
1433  007260  006003                      ROR     R3
1434  007262  103404                      BCS     1$
1435  007264  005167  000152              COM     BIT
1436  007270  005167  000144              COM     DATA
1437  007274  000207              1$:     RTS     PC
1438  007276  010203              TBXOR13:MOV     R2,R3           ;MM11F MEMORY
1439  007300  010201                      MOV     R2,R1
1440  007302  000303                      SWAB    R3
1441  007304  006101                      ROL     R1
1442  007306  006101                      ROL     R1
1443  007310  006101                      ROL     R1
1444  007312  006101                      ROL     R1
1445  007314  060103              T3X9:   ADD     R1,R3
1446  007316  000756                      BR      ALLOR
1447  007320  010203              T3XOR9: MOV     R2,R3           ;MM11G MEMORY
1448  007322  006003                      ROR     R3
1449  007324  010301                      MOV     R3,R1
1450  007326  000301                      SWAB    R1
1451  007330  006003                      ROR     R3
```

```
  1452  007332  006003                          ROR     R3
  1453  007334  000767                          BR      T3X9
```

```
                                        ;CHECK TO SEE IF HALT ON ERROR HAS BEEN SELECTED BY SWITCH 15.
                                        ;IF NOT CONTINUE TESTING.
                                        
007336  005037  177776        POTS:   CLR     @#PS            ;DROP PRIORITY
007342  005737  177570                TST     @#SWR           ;HALT ON ERROR?
007346  100001                        BPL     .+4             ;NO - SKIP
007350  000000                        HALT                    ;WAIT FOR CONTINUE
007352  000002                        RTI                     ;RETURN

                                        ;ISSUE THE START COMMAND TO THE DEVICE UNDER TEST. THE WORDCOUNT
                                        ;REGISTER, THE BUS ADDRESS REGISTER, AND THE COMMAND REGISTER ARE
                                        ;LOADED.  THE "MEX" BITS OF THE COMMAND REGISTER ARE ALSO LOADED
                                        ;AT THIS TIME FROM LOCATION "EMBITS".

007354  017646  000000        .BEGIN: MOV     @(6),-(6)       ;GET ADDRESS
007360  017646  000000                MOV     @(6),-(6)       ;GET ADDRESS
007364  016776  002032  000000        MOV     IOWAD,@(6)      ;LOAD MEMORY ADDRESS
007372  022626                        CMP     (6)+,(6)+       ;RESTORE STACK
007374  062716  000002                ADD     #2,(6)          ;INCREMENT STACK
007400  016705  001660                MOV     EMBITS,TTY      ;CLEAR JUNK
007404  057605  000000                BIS     @(6),TTY        ;GET REST OF COMMAND
007410  062716  000002                ADD     #2,(6)          ;INCREMENT STACK
007414  017646  000000                MOV     @(6),-(6)       ;GET ADDRESS
007420  017646  000000                MOV     @(6),-(6)       ;GET ADDRESS
007424  010576  000000                MOV     TTY,@(6)        ;LOAD MEMORY ADDRESS
007430  022626                        CMP     (6)+,(6)+       ;RESTORE STACK
007432  062716  000002                ADD     #2,(6)          ;INCREMENT STACK
007436  000002                        RTI

                                        ;DATA FOR MEMORY
007440  000000          DATA:   0       ;FLAG
007442  000000          BIT:    0       ;WHAT TYPE OF MEMORY
007444  000000          MEMORY: 0

                                        ;REQUEST THE NEXT MEMORY SEGMENT FOR TESTING AND LOAD THE
                                        ;APPROPRIATE DATA PATTERN.

007446  104444          WRADR:  NEXT                    ;GET NEXT BANK
007450  104450  000031          FUNCTN  ,R.W            ;REQUEST WINDOW IN R.W MODE
007454  005067  177760          CLR     DATA            ;CLEAR
007460  005067  177756          CLR     BIT             ;FLAGS
007464  016705  001734          MOV     WRDCNT,TTY      ;SET THE COUNTER
007470  016702  001566          MOV     WRADR,R2        ;SET THE STARTING ADDRESS
007474  004767  177532  1$:     JSR     PC,GDATA        ;GET WORST CASE NOISE
007500  016722  177734          MOV     DATA,(2)+       ;PUT INTO MEMORY
007504  005205                  INC     TTY             ;DEC THE COUNTER
007506  001372                  BNE     1$              ;NO - LOOP
007510  104450  000001          FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
007514  000002                  RTI                     ;RETURN

        172100          PARCSR= 172100
        000114          PARVEC= 114

007516  012737  007610  000114  .MAMK:  MOV     #.PARSRV,@#PARVEC       ;SET PARITY INTERRUPT VECTOR
007524  012737  000340  000116          MOV     #340,@#PARVEC+2         ;SET PRIORITY LEVEL TO 7
007532  013746  000004                  MOV     @#4,-(SP)               ;SAVE CURRENT ERROR VECTOR
```

```
1510  007536  013746  000006          MOV     @#6,-(SP)               ;SAVE PRIORITY LEVEL
1511  007542  012737  000006  000004  MOV     #6,@#4
1512  007550  012737  000002  000006  MOV     #RTI,@#6
1513  007556  012700  172100          MOV     #PARCSR,R0              ;GET FIRST CSR ADDR
1514  007562  012702  000001          MOV     #1,R2
1515  007566  012720  000001  1$:     MOV     #1,(R0)+                ;SET ACTION ENABLE IF AVAILABLE
1516  007572  006302                  ASL     R2                      ;SHIFT AVAILABILITY INDICATOR
1517  007574  103374                  BCC     1$
1518  007576  012637  000006          MOV     (SP)+,@#6               ;RESTORE ERROR VECTOR PRIORITY
1519  007602  012637  000004          MOV     (SP)+,@#4               ;AND INTERRUPT VECTOR
1520  007606  000207                  RTS     PC
1521
1522  007610  000000          .PARSRV:HALT                            ;MEMORY PARITY ERROR TRAP
1523  007612  000002                  RTI
1524
1525                          ;COME HERE IF A MEMORY MANAGEMENT VIOLATION OCCURS.
1526
1527  007614                  MMERR:
1528  007614  000004  007620          TYPE    .+2                     ;.ASCIZ <15><12><12>"MEMORY MANAGEMENT ERROR AT "
1529  007660  016705  167712          MOV     SR0+4,TTY               ;TYPE SR0+4 IN OCTAL
1530  007664  004767  002232          JSR     PC,PRINTR               ;TYPE LEADING ZERO'S
1531  007670  000004  007674          TYPE    .+2                     ;.ASCIZ "    SR0 = "
1532  007710  016705  167656          MOV     SR0,TTY                 ;TYPE SR0 IN OCTAL
1533  007714  004767  002202          JSR     PC,PRINTR               ;TYPE LEADING ZERO'S
1534  007720  000000                  HALT                            ;MM ERROR
1535  007722  000002                  RTI                             ;RETURN
```

```
1536                                        ;COMPLEMENT THE DATA BUFFER.
1537
1538  007724  016702  001332        WRCADR: MOV     WADRA,R2        ;SET UP BEGINNING OF BUFFER
1539  007730  104450  000031                FUNCTN  ,R.W            ;REQUEST WINDOW IN R.W MODE
1540  007734  016705  001464                MOV     WRDCNT,TTY      ;SET COUNTER
1541  007740  005122                1$:     COM     (2)+            ;COMPLIMENT IT
1542  007742  005205                        INC     TTY             ;COUNT IT
1543  007744  001375                        BNE     1$              ;LOOP TIL END OF CORE
1544  007746  104450  000001                FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
1545  007752  000002                        RTI                     ;RETURN
1546
1547                                        ;REPORT A DATA COMPARE ERROR.
1548
1549  007754  032737  020000  177570 ER3:    BIT     #SW13,@#SWR     ;INHIBIT TYPEOUTS
1550  007762  001401                        BEQ     .+4             ;SKIP IF TYPOUTS WANTED
1551  007764  000002                        RTI                     ;RETURN
1552  007766  022767  041524  002056        CMP     #"TC,INPUT      ;IS IT A TC11
1553  007774  001003                        BNE     .+10            ;SKIP IF NOT
1554  007776  012777  000001  001344        MOV     #1,@TCCS        ;STOP THE TC11
1555  010004  000004  010010                TYPE    .+2             ;.ASCIZ <15><12>"DATA ERROR AT "
1556  010032  010205                        MOV     R2,TTY          ;TYPE R2 WITH MX AS 18 BIT ADDRESS
1557  010034  004767  002116                JSR     PC,PRINTA       ;GO TO ADDRESS PRINTER
1558  010040  000004  010044                TYPE    .+2             ;.ASCIZ "  TRUE = "
1559  010056  016705  177356                MOV     DATA,TTY        ;TYPE DATA IN OCTAL
1560  010062  004767  002034                JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
1561  010066  000004  010072                TYPE    .+2             ;.ASCIZ "  RECEIVED = "
1562  010110  011205                        MOV     (2),TTY         ;TYPE (2) IN OCTAL
1563  010112  004767  002004                JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
1564  010116  104434                        STOP                    ;HALT ON ERROR
1565  010120  000002                        RTI                     ;RETURN
1566
1567                                        ;THIS ROUTINE WILL TAKE THE CONTENTS OF LOCATION "MX" WHICH WAS
1568                                        ;SETUP BY SUBROUTINE "NXT" AND DETERMINE WHAT THE CONTENTS
1569                                        ;OF THE OTHER PROCESSOR'S RELOCATION REGISTER SHOULD BE. THIS INFORMATION
1570                                        ;IS PASSED TO THE OTHER PROCESSOR THROUGH THE WINDOW'S DATA BUFFER.
1571                                        ;IF DATA IS BEING TRANSFERRED TO SEGMENT 0 OF THE OTHER PROCESSOR
1572                                        ;THE STARTING ADDRESS OF THE WINDOW IS OFFSET BY 4K TO AVOID
1573                                        ;DESTROYING THE OTHER PROGRAM.
1574
1575  010122                        SET4K:
1576  010122  104450  000001                FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
1577  010126  032767  004000  001132        BIT     #4000,SIZE      ;4K WINDOW OR LESS?
1578  010134  001006                        BNE     3$              ;BRANCH IF YES
1579  010136  042767  020000  001116        BIC     #20000,WADRA
1580  010144  042767  020000  001250        BIC     #20000,IOWAD
1581  010152  016705  170636        3$:     MOV     MX,TTY          ;GET THE MX
1582  010156  016700  001104                MOV     SIZE,R0         ;GET THE SIZE OF THE WINDOW
1583  010162  032767  004000  001076        BIT     #4000,SIZE      ;4K OR LESS?
1584  010170  001402                        BEQ     5$              ;BRANCH IF GREATER
1585  010172  012700  174000                MOV     #174000,R0      ;MAKE WINDOW LOOK LIKE 4K
1586  010176  005400                5$:     NEG     R0              ;MAKE IT A BIT
1587  010200  006000                1$:     ROR     R0              ;SHIFT IT
1588  010202  103402                        BCS     2$              ;GET OUT IF SET
1589  010204  006305                        ASL     TTY             ;MOVE INTO POSITION
1590  010206  000774                        BR      1$              ;LOOP
1591  010210  010577  001024        2$:     MOV     TTY,@WDBRA      ;PUT INTO DATA BUFFER
```

```
1592  010214  005767  170574                    TST     MX
1593  010220  001006                            BNE     4$
1594  010222  052767  020000  001032            BIS     #20000,WADRA
1595  010230  052767  020000  001164            BIS     #20000,IOWAD
1596  010236                          4$:
1597  010236  104450  000041                    FUNCTN  ,BANK           ;REQUEST WINDOW IN BANK MODE
1598  010242  000002                            RTI                     ;RETURN
1599
1600
1601  010244  012767  000000  162066  MAP:      MOV     #0,KIPAR0       ;MAP BANK 0 INTO SELF
1602  010252  012767  077406  162020            MOV     #77406,KIPDR0   ;4K - R/W
1603  010260  012767  077406  162016            MOV     #77406,KIPDR2   ;4K - R/W
1604  010266  012767  077406  162012            MOV     #77406,KIPDR3   ;4K - R/W
1605  010274  012767  007600  162054            MOV     #7600,KIPAR7    ;MAP BANK 7 INTO 37
1606  010302  012767  077406  162006            MOV     #77406,KIPDR7   ;4K - R/W
1607  010310  000002                            RTI                     ;RETURN
```

```
                                        ;ALL OPERATIONS TO A DEVICE ARE UNDER INTERRUPT CONTROL AND WHEN A
                                        ;DEVICE IS BUSY PERFORMING AN OPERATION, CONTROL COMES HERE TO WAIT
                                        ;UNTIL AN INTERRUPT OCCURS.

010312  012737  010326  000004  WAIT:   MOV     #TRAP4,@#4      ;SET TRAP VECTOR
010320  005037  177776                  CLR     @#PS            ;BRO
010324  000777                          BR      .               ;WAIT

                                        ;COME HERE FOR A MEMORY TIMEOUT.

010326  005777  000704          TRAP4:  TST     @WCSRA          ;CHECK FOR ERROR
010332  100424                          BMI     WINDOW          ;SKIP IF WINDOW ERROR
010334  000004  010340                  TYPE    .+2             ;.ASCIZ <15><12><12>"TRAP TO 4 FROM "
010364  011600                          MOV     (6),R0          ;GET RETURN ADDRESS
010366  162700  000002                  SUB     #2,R0           ;MAKE IT REAL ADDRESS
010372  010005                          MOV     R0,TTY          ;TYPE R0 IN OCTAL
010374  004767  001522                  JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
010400  000000                          HALT                    ;TRAPPED TO 4
010402  000002                          RTI                     ;RETURN

                                        ;THIS IS THE WINDOW INTERRUPT HANDLER. IF THE INTERRUPT IS
                                        ;CAUSED BY THE SETTING OF NEW DATA, CONTROL IS TRANSFERRED TO "NOERR".
                                        ;IF IT IS AN ERROR, IT IS REPORTEDTO THE OPERATOR.

010404  017700  000626          WINDOW: MOV     @WCSRA,R0       ;SAVE WCSRA
010410  100054                          BPL     NOERR           ;SKIP IF NO ERROR FLAG
010412  000004  010416                  TYPE    .+2             ;.ASCIZ <15><12><12>"WINDOW ERROR AT "
010442  011600                          MOV     (6),R0          ;GET THE RETURN ADDRESS
010444  162700  000002                  SUB     #2,R0           ;MAKE IT REAL ADDRESS
010450  010005                          MOV     R0,TTY          ;TYPE R0 IN OCTAL
010452  004767  001444                  JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
010456  000004  010462                  TYPE    .+2             ;.ASCIZ "   WCSRA = "
010476  017705  000534                  MOV     @WCSRA,TTY      ;TYPE @WCSRA IN OCTAL
010502  004767  001414                  JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
010506  000004  010512                  TYPE    .+2             ;.ASCIZ "   WDARA = "
010526  017705  000512                  MOV     @WDARA,TTY      ;TYPE @WDARA IN OCTAL
010532  004767  001364                  JSR     PC,PRINTR       ;TYPE LEADING ZERO'S
010536  000000                          HALT                    ;WINDOW ERROR!
010540  000002                          RTI                     ;RETURN

                                        ;CONTROL COMES HERE WHEN THE OTHER PROCESSOR SETS NEW DATA. THE
                                        ;DATA BITS ARE EXAMINED TO DETERMINE WHICH FUNCTION IS BEING REQUESTED
                                        ;AND CONTROL IS TRANSFERRED TO THE PROPER ROUTINE.

010542  000300          NOERR:  SWAB    R0              ;GET THE
010544  042700  177761          BIC     #177761,R0      ;DATA BITS
010550  000170  010554          JMP     @WINTAB(0)      ;GO TO ROUTINE


010554  010576          WINTAB: W.NOP                   ;NOP - CLOSE WINDOW
010556  010626                  W.READ                  ;READ ONLY
010560  010660                  W.WRIT                  ;WRITE ONLY
010562  010660                  W.R.W                   ;READ/WRITE
010564  010712                  W.BANK                  ;BANK SELECT
010566  010574                  W.FUTR                  ;FUTURE USE
010570  010574                  W.FUTR                  ;FUTURE USE
```

```
  1664  010572  010744                          W.ZAP                   ;GET READY
```

```
1665                                        ;RESERVED FOR FUTURE USE.
1666
1667  010574  000000                        W.FUTR: HALT                    ;RESERVED FOR FUTURE USE
1668
1669                                        ;WINDOW REQUESTED TO BE CLOSED.
1670
1671  010576  042777  000402  000432        W.NOP:  BIC     #402,@WCSRA     ;CLOSE WINDOW
1672  010604  152777  000100  000424                BISB    #100,@WCSRA     ;I/E
1673  010612  005067  000456                        CLR     NOTRD           ;NOT A READ
1674  010616  042777  017000  000412                BIC     #17000,@WCSRA   ;CLEAR NEW DTAT
1675  010624  000002                                RTI                     ;RETURN
1676
1677                                        ;WINDOW REQUESTED IN THE READ MODE.
1678
1679  010626  052777  000400  000402        W.READ: BIS     #400,@WCSRA     ;READ ONLY
1680  010634  152777  000100  000374                BISB    #100,@WCSRA     ;I/E
1681  010642  012767  177777  000424                MOV     #-1,NOTRD       ;SET READ FLAG
1682  010650  042777  017000  000360                BIC     #17000,@WCSRA   ;CLEAR NEW DATA
1683  010656  000002                                RTI                     ;RETURN
1684
1685                                        ;WINDOW REQUESTED IN THE READ/WRITE OR WRITE MODE.
1686
1687  010660                                W.R.W:
1688  010660  012767  177777  000406        W.WRIT: MOV     #-1,NOTRD       ;NOT A READ
1689  010666  052777  000402  000342        1$:     BIS     #402,@WCSRA     ;READ/WRITE
1690  010674  152777  000100  000334                BISB    #100,@WCSRA     ;I/E
1691  010702  042777  017000  000326                BIC     #17000,@WCSRA   ;CLEAR NEW DATA
1692  010710  000002                                RTI                     ;RETURN
1693
1694                                        ;LOAD THE CONTENTS OF THE DATA BUFFER INTO THE RELOCATION
1695                                        ;REGISTER.
1696
1697  010712  017777  000324  000326        W.BANK: MOV     @WDBPB,@WRARA   ;BANK SELECT
1698  010720  042777  000402  000310                BIC     #402,@WCSRA     ;CLEAR NEW DATA
1699  010726  152777  000100  000302                BISB    #100,@WCSRA     ;I/E
1700  010734  042777  017000  000274                BIC     #17000,@WCSRA   ;CLEAR NEW DATA
1701  010742  000002                                RTI                     ;RETURN
1702
1703                                        ;WARNING TO THIS PROCESSOR TO GET READY TO GET
1704                                        ;THE REQUESTED FUNCTION.
1705
1706  010744  042777  010502  000264        W.ZAP:  BIC     #10502,@WCSRA
1707  010752  032777  010000  000256        1$:     BIT     #10000,@WCSRA   ;WAIT FOR NEW DATA
1708  010760  001774                                BEQ     1$              ;TO GO AWAY
1709  010762  000167  177416                        JMP     WINDOW          ;NOW GET FUNCTION
1710
1711
1712                                        ;DETERMINE WHERE THE WINDOW'S CSR REGISTER IS LOCATED.
1713
1714  010766  012700  164000                REGSCN: MOV     #164000,R0      ;SET WCSRA FOR SCANNING
1715  010772  012702  000010                        MOV     #8.,R2          ;SET COUNT
1716  010776  012737  011010  000004                MOV     #5$,@#4         ;SET FOR TIMEOUT
1717  011004  005710                        6$:     TST     (0)             ;IS IT THERE?
1718  011006  000207                                RTS     PC              ;RETURN
1719  011010  022626                        5$:     CMP     (6)+,(6)+       ;CLEAR STACK
1720  011012  062700  000020                        ADD     #20,R0          ;BUMP ADDRESS
```

```
 1721  011016  005302                          DEC     R2              ;DEC COUNT
 1722  011020  001371                          BNE     6$              ;LOOP UNTIL FOUND
 1723  011022  000000                          HALT                    ;NO WCSRA
 1724  011024  000760                          BR      REGSCN          ;TRY IT AGAIN
 1725
 1726                                  ;OPEN THE WINDOW TO ALLOW THE OTHER PROCESSOR TO LOAD THE PROGRAM
 1727                                  ;INTO ITSELF. THIS PROCESSOR WILL LOOP AT LOCATION "BR." WHILE THE
 1728                                  ;OTHER PROCESSOR GETS THE PROGRAM. THE OTHER PROCESSOR WILL REPLACE THE
 1729                                  ;THE BRANCH TO SELF INSTRUCTION WITH A NOP WHEN THE PROGRAM IS LOADED.
 1730
 1731  011026  000005                  OPEN:   RESET                   ;ZAP THE WORLD
 1732  011030  012706  000500                  MOV     #500,SP         ;INITIALIZE THE STACK
 1733  011034  012767  177777  000230        MOV     #-1,ONCE        ;RESET ONCE
 1734  011042  012767  000777  000014        MOV     #777,BR.        ;RESTORE THE BR.
 1735  011050  004767  177712                TSR     PC,REGSCN       ;GET THE REGISTER
 1736  011054  010067  000040                MOV     R0,BR.1         ;SAVE WCSRA
 1737  011060  012710  000402                MOV     #402,(0)        ;OPEN THE WINDOW
 1738  011064  000777                  BR.:    BR      .               ;WAIT
 1739  011066  005077  000026                CLR     @BR.1           ;CLEAR WCSRA
 1740  011072  012737  000240  006660        MOV     #240,@#DOF2
 1741  011100  012737  000240  006662        MOV     #240,@#DOF2+2
 1742  011106  012737  000240  007106        MOV     #240,@#DOF5
 1743  011114  000137  001026                JMP     @#BEGIN         ;SELF START
 1744
 1745  011120  000000                  BR.1:   0
```

```
1746  011122  116767  166434  000722  READR:  MOVB    @#7562,INPUT    ;READ THE CHARACTER
1747  011130  142767  000200  000714          BICB    #200,INPUT      ;CLEAR JUNK
1748  011136  122767  000003  000706  1$:     CMPB    #'C-100,INPUT   ;IS IT A ^C?
1749  011144  001401                          BEQ     .+4             ;SKIP IF ^C
1750  011146  000002                          RTI                     ;RETURN IF NOT
1751  011150  010667  000010                  MOV     SP,CFLAG        ;SET CONTROL "C" FLAG
1752  011154  042737  000100  177560          BIC     #100,@#177560   ;DISABLE TTY INTERRUPT
1753  011162  000002                          RTI
1754
1755  011164  000000                  CFLAG:  0
1756
1757
1758  011166  000000                  CHK:    0
1759  011170  000000                  EX:     0
1760  011172  020040  051503  036440  M3:     .ASCIZ  "  CS = "
1761  011200  000040
1762  011202  020040  051105  036440  M4:     .ASCIZ  "  ER = "
1763  011210  000040
1764                                  .EVEN
1765  011212  032456  030440  031040  WIND:   .ASCII  ".5 1 2 4 81632"
1766  011220  032040  034040  033061
1767  011226  031063
1768  011230  000000  000000          SIZ:    0,0
1769  011234  000000                  OFFSET: 0
1770
1771          177572                  SR0=    177572                  ;KT11 - MEMORY MANAGEMENT
1772          172340                  KIPAR0= 172340
1773          172300                  KIPDR0= 172300
1774          172344                  KIPAR2= 172344
1775          172304                  KIPDR2= 172304
1776          172346                  KIPAR3= 172346
1777          172306                  KIPDR3= 172306
1778          172356                  KIPAR7= 172356
1779          172316                  KIPDR7= 172316
1780
1781  011236  164000                  WCSRA:  164000                  ;CONTROL & STATUS REG. (R/W)  -  WINDOW A
1782  011240  164002                  WDBRA:  164002                  ;OUTPUT DATA BUFFER (R/W)
1783  011242  164004                  WDBRB:  164004                  ;INPUT DATA BUFFER (R)
1784  011244  164006                  WDARA:  164006                  ;DISPLACEMENT ADDRESS REG. (R)
1785  011246  164010                  WRARA:  164010                  ;RELOCATION ADDRESS REGISTER (R/W)
1786  011250  164012                  WADRAX: 164012                  ;WINDOW ADDRESS REGISTER (R)
1787  011252  164014                  WVARA:  164014                  ;VECTOR ADDRESS REGISTER (R)
1788  011254  000000                          0
1789
1790  011256  000300  000302          WVECA:  300,302                 ;VECTOR ADDRESS
1791
1792  011262  020000                  WADRA:  20000                   ;WINDOW ADDRESS
1793  011264  000000                  EMBITS: 0                       ;EXTENDED MEMORY BITS
1794
1795  011266  000000                  SIZE:   0                       ;WINDOW SIZE
1796  011270  000000                  MMN:    0                       ;MEMORY MANAGEMENT REQUIRED FLAG
1797  011272  177777                  ONCE:   -1                      ;ONCE ONLY FLAG
1798  011274  000000                  NOTRD:  0                       ;READ FLAG - 0=NO READ
```

```
 1799  011276  177460                RFCS:   177460                ;RF11  DISK  (256K)
 1800  011300  177470                RFER:   177470
 1801  011302  177462                RFWC:   177462
 1802  011304  177464                RFCA:   177464
 1803  011306  177466                RFDA:   177466
 1804  011310  177446                RCCS:   177446                ;RC11  DISK  (64K)
 1805  011312  177444                RCER:   177444
 1806  011314  177450                RCWC:   177450
 1807  011316  177452                RCCA:   177452
 1808  011320  177442                RCDA:   177442
 1809  011322  176714                RPCS:   176714                ;RP11  DISK  (PACK)
 1810  011324  176716                RPWC:   176716
 1811  011326  176720                RPCA:   176720
 1812  011330  176724                RPDA:   176724
 1813  011332  176712                RPER:   176712
 1814  011334  176710                RPDS:   176710
 1815  011336  177404                RKCS:   177404                ;RK11  DISK  (CARTRIDGE)
 1816  011340  177402                RKER:   177402
 1817  011342  177406                RKWC:   177406
 1818  011344  177410                RKCA:   177410
 1819  011346  177412                RKDA:   177412
 1820  011350  177342                TCCS:   177342                ;TC11  DECTAPE
 1821  011352  177340                TCER:   177340
 1822  011354  177344                TCWC:   177344
 1823  011356  177346                TCCA:   177346
 1824  011360  177350                TCDB:   177350
 1825  011362  172522                TMCS:   172522                ;TM11 MAGTAPE
 1826  011364  172520                TMER:   172520
 1827  011366  172524                TMBC:   172524
 1828  011370  172526                TMCA:   172526
 1829
 1830  011372  000204  000206        RFTRAP: 204,206
 1831  011376  000210  000212        RCTRAP: 210,212
 1832  011402  000254  000256        RPTRAP: 254,256
 1833  011406  000220  000222        RKTRAP: 220,222
 1834  011412  000214  000216        TCTRAP: 214,216
 1835  011416  000224  000226        TMTRAP: 224,226
 1836
 1837  011422  000000                IOWAD:  0
 1838  011424  177400                WRDCNT. -400                  ;WORD COUNT
```

```
                                        ;       $LOADR          GET AND RESTORE THE LOADER
                                        ;
                                        ;THESE ROUTINES FIRST FIND THE LOADER (TOP OF HIGHEST
                                        ;4K BANK IN 28K) AND SAVE IT AT TAG "ENDP:". ENTRY FOR
                                        ;THIS ROUTINE IS "JSR PC,LODGET".
                                        ;
                                        ;"JSR PC,LODRES" WILL RESTORE THE LOADER IN ITS ORIGONAL
                                        ;LOCATION. ".LOD:" CONTAINS THE ADDRESS OF THE LOADER.
                                        ;
011426  010001          LODGET: MOV     R0,R1           ;GET THE BANK
011430  022701  000006          CMP     #6,R1           ;IS IT > 6?
011434  100002                  BPL     1$              ;SKIP IF LES THAN 6
011436  012701  000006          MOV     #6,R1           ;IF > 7 MAKE 7
011442  006001          1$:     ROR     R1              ;GET THE
011444  006001                  ROR     R1              ;UPPER
011446  006001                  ROR     R1              ;THREE
011450  006001                  ROR     R1              ;BITS
011452  042701  017777          BIC     #17777,R1       ;CLEAR JUNK
011456  012702  017200          MOV     #ENDP,R2        ;GET SAVE ADDRESS
011462  060201                  ADD     R2,R1           ;MAKE OTHER ONE
011464  010167  000034          MOV     R1,.LOD         ;SAVE ADDRESS
011470  012122          2$:     MOV     (1)+,(2)+       ;MOVE WORD
011472  022702  020000          CMP     #20000,R2       ;END?
011476  001374                  BNE     2$              ;NO!
011500  000207                  RTS     PC              ;RETURN
                                        ;
011502  012702  017200  LODRES: MOV     #ENDP,R2        ;GET END OF PROGRAM
011506  016701  000012          MOV     .LOD,R1         ;GET SAVE ADDRESS
011512  012221          1$:     MOV     (2)+,(1)+       ;RESTORE WORD
011514  022702  020000          CMP     #20000,R2       ;END?
011520  001374                  BNE     1$              ;LOOP
011522  000207          3$:     RTS     PC              ;RETURN
                                        ;
011524  000000          .LOD:   0                       ;STARTING ADDRESS OF LOADER
                                        ;
                                        ;THIS ROUTINE TESTS THE FLAG TO SEE IF CONTROL "C" HAS BEEN TYPED.
                                        ;IF TYPED THE PROGRAM RESTARTS, OTHERWISE IT CONTINUES.
                                        ;
011526  005767  177432  CONTC:  TST     CFLAG           ;CONTROL "C" TYPED?
011532  001001                  BNE     1$              ;BRANCH IF YES
011534  000207                  RTS     PC              ;CONTINUE
011536  012706  000500  1$:     MOV     #500,SP         ;RESTORE THE STACK
011542  000004  011546          TYPE    .+2             ;.ASCIZ "^C"<177>
011552  005067  177406          CLR     CFLAG           ;CLEAR RESTART FLAG
011556  104450  000001          FUNCTN  ,CLEAR          ;REQUEST WINDOW IN CLEAR MODE
011562  005037  177776          CLR     @#PS
011566  000137  002564          JMP     @#REDO          ;GO RESTART
```

```
                                        ;       $TYPE           MESSAGE TYPEOUT ROUTINE
                                        ;
                                        ;THIS ROUTINE IS USE TO TYPE ASCII MESSAGES ON THE TTY. THE
                                        ;CALL CAN BE IN ONE OF 3 FORMS: 1) "TYPE ,ADR" - TYPES THE
                                        ;MESSAGE STARTING IN LOCATION "ADR:", 2) "TYPE ,CHAR" - TYPES
                                        ;THE ASCII "CHAR", AND 3) "PRINT <<15><12>"MESSAGE"> - TYPES
                                        ;THE MESSAGE WHICH IS INLINE ASCII.
011572  010467  000146          IOT$:   MOV     R4,IOTI$        ;SAVE R4
011576  010546                          MOV     TTY,-(6)        ;SAVE TTY RS=TTY
011600  017605  000002                  MOV     @2(6),TTY       ;GET ADDRESS TO BE TYPED
011604  032705  177400                  BIT     #177400,TTY     ;IS IT A TYPEM?
011610  001004                          BNE     1$              ;NO
011612  010567  000124                  MOV     TTY,.TYPE       ;GET THE CHARACTER
011616  012705  011742                  MOV     #.TYPE,TTY      ;FUDGE THE ADDRESS
011622  105715                  1$:     TSTB    (TTY)           ;TERMINATOR?
011624  001423                          BEQ     2$              ;GET OUT IF SO
011626  122715  000012                  CMPB    #12,(TTY)       ;IS THE CHAR A LINE FEED?
011632  001012                          BNE     4$              ;BRANCH IF NO
011634  116704  000101                  MOVB    FILCHR+1,R4     ;GET FILLER COUNT
011640  116737  000074  177566  5$:     MOVB    FILCHR,@#177566 ;OUTPUT FILLER CHAR
011646  105737  177564                  TSTB    @#177564        ;WAIT FOR DONE
011652  100375                          BPL     .-4
011654  005304                          DEC     R4              ;DECREMENT FILLER COUNT
011656  001370                          BNE     5$
011660  112537  177566          4$:     MOVB    (TTY)+,@#177566 ;LOAD AND TYPE THE CHARACTER
011664  105737  177564                  TSTB    @#177564        ;IS THE PRINTER READY
011670  100375                          BPL     .-4             ;WAIT UNTIL IT IS
011672  000753                          BR      1$              ;GET THE NEXT CHARACTER
011674  017646  000002          2$:     MOV     @2(6),-(6)      ;GET ADDRESS TO BE TYPED
011700  062766  000002  000004          ADD     #2,4(6)         ;ADD 2 TO THE ADDRESS
011706  022666  000002                  CMP     (6)+,2(6)       ;IS IT .+2?
011712  001006                          BNE     3$              ;NO
011714  062705  000002                  ADD     #2,TTY          ;ADD 2 TO THE ADDRESS
011720  042705  000001                  BIC     #1,TTY          ;BACK UP TO AN EVEN BYTE
011724  010566  000002                  MOV     TTY,2(6)        ;RESTORE ADDRESS
011730  012605                  3$:     MOV     (6)+,TTY        ;RESTORE TTY
011732  016704  000006                  MOV     IOTI$,R4        ;RESTORE R4
011736  000002                          RTI                     ;RETURN
011740  001000                  FILCHR: 1000                    ;FILCHR=0 (CHAR)  FILCHR+1=2 (COUNT)
011742  000000                  .TYPE:  0                       ;CHARACTER TYPE LOCATION
011744  000000                  IOTI$:  0
                                        ;
011746  010346                  READ$:  MOV     R3,-(6)         ;SAVE R3
011750  012703  012052          1$:     MOV     #INPUT,R3       ;GET ADDRESS
011754  022703  012072                  CMP     #INPUT+20,R3    ;BUFFER FULL?
011760  001412                          BEQ     4$              ;YES - TYPE "?"
011762  105737  177560          2$:     TSTB    @#177560        ;WAIT FOR
011766  100375                          BPL     .-4             ;A CHARACTER
011770  113713  177562                  MOVB    @#177562,(3)    ;GET CHARACTER
011774  142713  000200                  BICB    #200,(3)        ;GET RID OF JUNK
012000  122713  000177                  CMPB    #177,(3)        ;IS IT A RUBOUT
012004  001005                          BNE     3$              ;SKIP IF NOT
012006                          4$:
012006  000004  012012                  TYPE    ,.+2            ;.ASCIZ "?"<15><12>
012016  000754                          BR      1$              ;ZAP THE BUFFER AND LOOP
```

```
 1943  012020  111367  177716          3$:     MOVB    (3),.TYPE       ;SET UP FOR TYPING
 1944  012024  000004  011742                  TYPE    ,.TYPE          ;ECHO IT
 1945  012030  122723  000015                  CMPB    #15,(3)+        ;CHECK FOR RETURN
 1946  012034  001352                          BNE     2$              ;LOOP IF NOT RETURN
 1947  012036  105063  177777                  CLRB    -1(3)           ;ZAP RETURN (THE 15)
 1948  012042  000004  000012                  TYPE    ,I2             ;TYPE A LINE FEED
 1949  012046  012603                          MOV     (6)+,R3         ;RESTORE R3
 1950  012050  000207                          RTS     PC              ;RETURN
 1951  012052  000020              INPUT:.BLKW     20                  ;TTY INPUT AREA
```

```
1952                                    ;       $OCTAL          OCTAL TYPEOUT ROUTINE
1953
1954                                    ;THIS ROUTINE IS USED TO TYPE AN OCTAL NUMBER ON THE TTY. IT WILL TYPE
1955                                    ;ALL 6 CHARACTERS, SUPPRESS LEADING ZEROES, TYPE AN 18 BIT ADDRESS, OR TYPE
1956                                    ;THE 16 BITS. IT IS CALLED VIA THE DUMP, SDUMP, DUMP18, OR BITYPE MACRO'S.
1957
1958  012112  012767  170101  000202  BITYPS: MOV     #170101,.PR     ;SET BIT FLAG ANS 16. CHARACTER COUNT
1959  012120  000411                  BR      .PTIT           ;NOW TYPE IT IN BIT FORM
1960  012122  112767  000001  000172  PRINTR: MOVB    #1,.PR          ;SET ZERO FILL SWITCH
1961  012130  000402                  BR      .+6             ;SKIP
1962  012132  005067  000164          PRINTS: CLR     .PR             ;SUPRESS LEADING ZERO'S
1963  012136  112767  177772  000157          MOVB    #-6,.PR+1       ;SET COUNT
1964  012144  010446                  .PTIT:  MOV     R4,-(6)         ;SAVE R4
1965  012146  012704  012324                  MOV     #.PR+2,R4       ;SET POINTER TO FIRST ASCII CHAR.
1966  012152  105014                          CLRB    (4)             ;CLEAR FIRST BYTE
1967  012154  000432                  BR      .PRF            ;ROTATE FIRST BIT
1968  012156  010446                  PRINTA: MOV     R4,-(6)         ;SAVE R4
1969  012160  012704  012324                  MOV     #.PR+2,R4       ;SET UP POINTER TO OUTPUT AREA
1970  012164  116714  166624                  MOVB    MX,(4)          ;MX CONTAINS UPPER 5 BITS
1971  012170  006305                          ASL     TTY             ;GET RID
1972  012172  006305                          ASL     TTY             ;OF 3
1973  012174  006305                          ASL     TTY             ;JUNK BITS
1974  012176  106214                          ASRB    (4)             ;GET BIT13
1975  012200  006005                          ROR     TTY             ;PACK IT
1976  012202  106214                          ASRB    (4)             ;GET BIT14
1977  012204  006005                          ROR     TTY             ;PACK IT
1978  012206  152724  000060                  BISB    #'0,(4)+        ;MAKE IT ASCII
1979  012212  012767  175401  000102          MOV     #175401,.PR     ;-5,1 - 5 BYTES AND FILL
1980  012220  105014                  .PRL:   CLRB    (4)             ;CLEAR BYTE OF CHARACTER
1981  012222  032767  000100  000072          BIT     #100,.PR        ;BIT TYPING MODE?
1982  012230  001004                          BNE     .PRF            ;YES - SKIP 2 ROTATES
1983  012232  006105                          ROL     TTY             ;ROTATE BIT INTO C
1984  012234  106114                          ROLB    (4)             ;PACK IT
1985  012236  006105                          ROL     TTY             ;ROTATE BIT INTO C
1986  012240  106114                          ROLB    (4)             ;PACK IT
1987  012242  006105                  .PRF:   ROL     TTY             ;ROTATE BIT INTO C
1988  012244  106114                          ROLB    (4)             ;PACK IT
1989  012246  105714                          TSTB    (4)             ;IS IT ZERO?
1990  012250  001402                          BEQ     .+6             ;SKIP INC
1991  012252  105267  000044                  INCB    .PR             ;SET FILL SWITCH
1992  012256  105767  000040                  TSTB    .PR             ;CHECK FILL SWITCH
1993  012262  001402                          BEQ     .+6             ;SKIP BITSET
1994  012264  152724  000060                  BISB    #'0,(4)+        ;MAKE INTO ASCII CHAR
1995  012270  105267  000027                  INCB    .PR+1           ;INC COUNT
1996  012274  001351                          BNE     .PRL            ;REPEAT
1997  012276  022704  012324                  CMP     #.PR+2,R4       ;EMPTY BUFFER?
1998  012302  001002                          BNE     .+6             ;SKIP IF NOT
1999  012304  112724  000060                  MOVB    #'0,(4)+        ;LOAD 1 ZERO
2000  012310  105014                          CLRB    (4)             ;NULL TERMINATOR
2001  012312  000004  012324                  TYPE    .PR+2           ;TYPE IT
2002  012316  012604                          MOV     (6)+,R4         ;RESTORE R4
2003  012320  000207                          RTS     PC              ;RETURN
2004  012322  000012                  .PR:    .BLKW   12              ;COUNT, SWITCH, AND OUTPUT BUFFER
```

```
2005  012346  012777  012474  000126  PDOWNS: MOV     #ILLUP,@PUVEC$  ;SET FOR FAST UP
2006  012354  012777  000340  000122          MOV     #340,@PUVEC$+2  ;PRIO:7
2007  012362  010046                          MOV     R0,-(6)         ;PUSH R0 ON STACK
2008  012364  010146                          MOV     R1,-(6)         ;PUSH R1 ON STACK
2009  012366  010246                          MOV     R2,-(6)         ;PUSH R2 ON STACK
2010  012370  010346                          MOV     R3,-(6)         ;PUSH R3 ON STACK
2011  012372  010446                          MOV     R4,-(6)         ;PUSH R4 ON STACK
2012  012374  010546                          MOV     R5,-(6)         ;PUSH R5 ON STACK
2013  012376  010667  000076                  MOV     SP,SAVR6        ;SAVE SP
2014  012402  012777  012412  000072          MOV     #PUP$,@PUVEC$   ;SET UP VECTOR
2015  012410  000000                          HALT                    ;WAIT FOR PF
2016
2017  012412  016706  000062          PUP$:   MOV     SAVR6,SP        ;GET SP
2018  012416  005001                          CLR     R1              ;WAIT LOOP FOR THE TTY
2019  012420  005201                  1$:     INC     R1              ;WAIT FOR THE INC
2020  012422  001376                          BNE     1$              ;OF  WORD
2021  012424  012605                          MOV     (6)+,R5         ;POP STACK INTO R5
2022  012426  012604                          MOV     (6)+,R4         ;POP STACK INTO R4
2023  012430  012603                          MOV     (6)+,R3         ;POP STACK INTO R3
2024  012432  012602                          MOV     (6)+,R2         ;POP STACK INTO R2
2025  012434  012601                          MOV     (6)+,R1         ;POP STACK INTO R1
2026  012436  012600                          MOV     (6)+,R0         ;POP STACK INTO R0
2027  012440  012737  012346  000024          MOV     #PDOWNS,@#24    ;SET UP THE POWER DOWN VECTOR
2028  012446  012737  000340  000026          MOV     #340,@#26       ;PRIO:7
2029  012454  000004  012460                  TYPE    ..+2            ;.ASCIZ <15><12>"POWER"
2030  012470  000167  170360                  JMP     RTPF            ;JMP TO USER ADDRESS
2031
2032  012474  000000                  ILLUP:  HALT                    ;THE POWER UP SEQUENCE WAS STARTED
2033  012476  011062                          BR.-2                   ; BEFORE THE POWER DOWN WAS COMPLETE
2034
2035  012500  000000                  SAVR6:  0                       ;PUT THE SP HERE
2036  012502  000024  000026          PUVEC$: 24,26                   ;POWER UP VECTOR
2037  012506  000010                  PATCH1: .BLKW   10              ;PATCH AREA #1
2038  012526  000010                  PATCH2: .BLKW   10              ;PATCH AREA #2
2039
2040          017200                  .=      17200
204.
2042  017200  000300                  ENDP:   .BLKB   300
2043
2044          000001                  .END
```

```
ADRTAB  006304          1256    1263#
ALLOR   007254          1432#   1446
BANK  = 000041           371#    699    1597
BEGIN   001026           405     413     438     461#   1743
BELL  = 000007           389#   1297
BIT     007442          1405*   1432    1435*   1484#   1494#
BITYPS  012112          1958#
BR.     011064           437*   1734*   1738#   2033
BR.1    011120          1736*   1739*   1745#
CDATA   007140          1276    1400#
CFLAG   011164           462*   1153    1170    1413    1751*   1755#   1878    1883*
CHK     011166          1758#
CHK.RC= 104412           884     903     915    1269#
CHK.RF= 104410           822     841     853    1268#
CHK.RK= 104416          1017    1036    1048    1271#
CHK.RP= 104414           946     962     974    1270#
CHK.TC= 104420          1082    1105    1272#
CHK.TM= 104422          1185    1215    1273#
CH.RC   004336           927#   1269
CH.RF   004050           865#   1268
CH.RK   005156          1060#   1271
CH.RP   004654           995#   1270
CH.TC   005446          1117#   1272    1277
CH.TM   006142          1230#   1273
CKDATA= 104430           823     885     947    1018    1083    1186    1276#
CLEAR = 000001           367#    697     718     791     866     928     996    1061    1118    1231    1418    1501    1544
                        1576    1884
CODE    003626           784     789     798#
COMCOR= 104406           854     916     975    1049    1106    1216    1267#
CONTC   011526           695     793     828     890     949    1023    1088    1191    1878#
CORE  = 104404           829     891     950    1024    1093    1192    1266#
CSRTAB  007116          1365    1388#
C.RC    004234           892     903#
C.RF    003746           830     841#
C.RK    005054          1025    1036#
C.RP    004522           951     962#
DATA    007440          1404*   1409    1436*   1483#   1493*   1498    1559
DISPLA= 177570           388#   1302*   1318*
DOCORE  001310           503#
DOFUN   006602          1284    1342#
DOF1    006650          1351#   1354
DOF2    006660           410*    411*   1353#   1740*   1741*
DOF3    007070          1366    1380#   1384
DOF4    007110          1381    1383    1385#
DOF5    007106           412*   1384#   1742*
DOF7    006666          1352    1355#
DOIEG   001344           496     517#
DUM1  = 104400          1264#
DUM2  = 104402          1265#
EMBITS  011264           614*    618*    621*   1473    1793#
ENDC    003670           786     789     816#
ENDP    017200          1857    1865    2042#
ENDTAB= 006354          1257    1285#
ERRORS  001002           450#
ERR2  = 104432          1169    1277#
ERR3  = 104436          1152    1279#   1411
```

DZDABA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
ER3     007754          1279    1549#
EX      011170           865*    869*    927*    931*    995*    999*   1060*   1064*   1117*   1121*   1230*   1234*   1400
                        1759#
EXCODE  003540           759     780#    794
FILCHR  011740          1906    1907    1926#
FNDCSR  001724           547     553     569#
FND1    002054           592#    596
FND2    002402           623     649#
FND3    002500           657     663#
FUNCTN= 104450           697     699     700     718     782     791     833     845     858     866     895     907     920
                         928     954     966     979     996    1028    1040    1053    1061    1094    1107    1118    1197
                        1221    1231    1284#   1403    1418    1492    1501    1539    1544    1576    1597    1884
GDATA   007232          1408    1423#   1497
GETCOR  000514           417     427#    429
HLT   = 104000           384#
HLTADR  001012           453#
ICNT    001000           449#
ILLUP   012474          2005    2032#
INPUT   012052           725     730     735     740     745     750     756     771    1552    1746*   1747*   1748    1931
                        1932    1951#
IOT$    011572           464    1895#
IOT1$   011744          1895*   1924    1928#
IOWRD   011422           622*    687*   1470    1580*   1595*   1837#
KIPAR0= 172340           626*   1601*   1772#
KIPAR2= 172344           519*    522*    523     525     641*   1774#
KIPAR3= 172346           643    1776#
KIPAR7= 172356           633*   1605*   1778#
KIPDR0= 172300           627*   1602*   1773#
KIPDR2= 172304           629    1603*   1775#
KIPDR3= 172306          1604*   1777#
KIPDR7= 172316           634*   1606*   1779#
LAD     001010           452#
LIMIT   001020           456#    543*    702
LOADR   000562           418     445#
LOOGET  011426           535    1848#
LOORES  011502           445    1865#
MAP     010244          1283    1601#
MAPIT = 104446           517    1283#
MEMORY  007444           482*   1308    1311*   1313*   1315*   1423    1485#
MIN     001016           455#    480*   1298
MMERR   007614           476    1527#
MMN     011270           494*    625*    681    1796#
MX      001014           454#    481*    602*    605*   1293    1298*   1301*   1302    1306    1317*   1318    1322    1581
                        1592    1970    2005
M3      011172           874     936    1004    1069    1126    1239    1760#
M4      011202           876     938    1006    1071    1128    1241    1762#
N     = 000052           399#   1264    1265#   1266#   1267#   1268#   1269#   1270#   1271#   1272#   1273#   1274#   1275#
                        1276#   1277#   1278#   1279#   1280#   1281#   1282#   1283#   1284#   1285#
NEXT  = 104444           781    1282#   1491
NODEV   003466           719     769#
NOERR   010542          1633    1652#
NOKT    001574           541     543#
NOTRD   011274           483*    543     1380    1673*   1681*   1688*   1798#
NXT     006356          1282    1291#
OFFSET  011234           598*   1769#
OLIM    001022           457#    702*    707*    711*    712*    714*    715*    716*   1293
```

DZDABA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
ONCE    011272            485    534*  1733*  1797#
OPEN    011026            419   1731#
OPTION  001024            458#   484*   492*   495    540
PARCSR= 172100           1504#  1513
PARVEC= 000114           1505#  1507*  1508*
PATCH1  012506           1264   2037#
PATCH2  012526           1265   2038#
PC     =X000007           398#   427    445*   478*   538*   572*   613*   662*   694*   695*   790*   793*   814*
                          828*   875*   878*   890*   937*   940*   949*  1005*  1008*  1023*  1070*  1073*  1088*
                         1127*  1130*  1191*  1240*  1243*  1307*  1326*  1375*  1378*  1408*  1437*  1497*  1520*
                         1530*  1533*  1557*  1560*  1563*  1624*  1638*  1641*  1644*  1718*  1735*  1863*  1870*
                         1880*  1950*  2003*
PCNT    001004            451#
PDOWNS  012346            466   2005#  2027
POTS    007336           1278   1457#
PRINTA  012156            613   1557   1968#
PRINTR  012122            572    875    878    937    940   1005   1008   1070   1073   1127   1130   1240   1243
                         1378   1530   1533   1560   1563   1624   1638   1641   1644   1960#
PRINTS  012132            538    662   1307   1326   1375   1962#
PS     = 177776           386#   676*   696*   721*   769*   780*   792*  1342*  1349*  1353*  1367*  1457*  1613*
                         1885*
PUPS    012412           2014   2017#
PUVECS  012502           2005*  2006*  2014*  2036#
RCCA    011316            897    909    922   1807#
RCCS    011310            728#   889*   897    909    922    929    934   1804#
RCDA    011320            894*   906*   919*  1808#
RCER    011312            939   1805#
RCTRAP  011376            470*   727*   892*   904*   917*  1831#
RCWC    011314            893*   905*   918*  1806#
READ   = 000011            368#   700    833    845    895    907    954    966   1028   1040   1094   1197
READOR  011122            479   1746#
READOS  011746            694   1930#
REDO    002564            487    667    671    673    675#   773   1886
REGSCN  010766           1714#  1724   1735
RETI1   005626           1145#  1159   1172
RETR1   005704           1162#  1175
RETURN  005764           1143*  1176   1181#
REWIND= 104426           1193   1217   1275#
RFCA    011304            835    847    860   1802#
RFCS    011276            738#   827*   835    847    860    867    872   1799#
RFDA    011306            832*   844*   857*  1803#
RFER    011300            877   1800#
RFTRAP  011372            471*   737*   830*   842*   855*  1830#
RFWC    011302            831*   843*   856*  1801#
RKCA    011344           1030   1042   1055   1818#
RKCS    011336            743*  1022*  1030   1042   1055   1062   1067   1075*  1076   1815#
RKDA    011346           1027*  1039*  1052*  1819#
RKER    011340           1072   1816#
RKTRAP  011406            473*   742*  1025*  1037*  1050*  1833#
RKWC    011342           1026*  1038*  1051*  1817#
RPCA    011326            956    968    981   1811#
RPCS    011322            733*   948*   956    968    981    986*   987    991*   997   1002  1010*  1011   1809#
RPDA    011330            953*   965*   978*  1812#
RPOS    011334            989   1814#
RPER    011332           1007   1813#
RPTRAP  011402            472*   732*   951*   963*   976*  1832#
```

DZDABA.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
RPWC    011324          952*   964*   977*  1810#
RTPF    003054          720#  2030
RW.TM   006044         1204#  1275
R.RC    004274          904    915#
R.RF    004006          842    853#
R.RK    005114         1037   1048#
R.RP    004562          963    974#
R.TC    005404         1096   1105#
R.TM    006074         1194   1215#
R.W   = 000031          370#   782   1403   1492   1539
R0    =%000000          390#   430#   432    505#   507#   525#   526#   527#   528#   531#   537    543    578#
                        581    586#   587    588#   589#   590#   592    594#   601#   604#   608#   609#   610#
                        611    612    615#   616#   619#   622    635#   636#   637#   638#   639#   640#   641
                        644#   645    650#   652    654    658    660    701#   783#  1369#  1513#  1515#  1582#
                       1585#  1586#  1587#  1621#  1622#  1623   1632#  1635#  1636#  1637   1652#  1653#  1714#
                       1720#  1736   1848   2007   2026#
R1    =%000001          391#   427#   429#   504#   506#   509    550#   556#   579#   582    591#   595#   598
                        603#   606#   629#   643#   784#   786    788#   789#  1368#  1371#  1373#  1374   1439#
                       1441#  1442#  1443#  1444#  1445   1449#  1450#  1848#  1849   1851#  1852#  1853#  1854#
                       1855#  1856#  1858#  1859   1866#  2008   2018#  2019#  2025#
R2    =%000002          392#   628#   631#   642#   646#  1406#  1427   1430   1438   1439   1447   1496#  1514#
                       1516#  1538#  1556   1715#  1721#  1857#  1858   1861   1865#  1868   2009   2024#
R3    =%000003          393#  1427#  1428#  1429#  1430#  1431#  1432#  1433#  1438#  1440#  1445#  1447#  1448#
                       1449   1451#  1452#  1930   1931#  1932   1949#  2010   2023#
R4    =%000004          394#  1359#  1360   1361#  1362#  1365   1368   1895   1906#  1910#  1924#  1964   1965#
                       1968   1969#  1997   2002#  2011   2022#
R5    =%000005          395#  1253#  1254#  1255#  1256#  1257   1363#  1364#  1365   1369   2012   2021#
SAVR6   012500         2013#  2017   2035#
SAVTMP  007136         1360#  1395#
SCF1    005676         1155   1158   1161#
SCF1A   005742         1166   1174#
SCF2    005714         1161   1165#
SCOPE = 104400          383#
SCQ1    005752         1154   1171   1176#
SC1     005636         1140   1148#
SC1A    005664         1149   1157#
SEARCH= 104424         1095   1108   1274#
SETBAK  010122         1281    575#
SETBAN= 104442         1281#  1327
SIZ     011230          597#   599   1768#
SIZE    011266          587#   664    668    672    684    703    705    709   1299   1577   1582   1583   1795#
SIZEIT  001270          489    490    493#
SP    =%000006          397#   463#   533#   720#  1509#  1510#  1518   1519  1732#  1751   1881#  2013   2017#
SR0   = 177572          491    518#   529#   624    683#  1529   1532   1771#
STOP  = 104434          879    941   1009   1074   1131   1244   1278#  1564
SWR   = 177570          387#   388    434    488    870    932   1000   1065  1122   1235   1291   1295   1303
                       1319   1458   1549
SW10  = 002000          336#  1295
SW11  = 004000          335#
SW12  = 010000          334#
SW13  = 020000          332#   870    932   1000   1065   1122   1235   1549
SW14  = 040000          332#  1291
SW15  = 100000          331#
SW7   = 000200          340#   434
SW8   = 000400          339#  1303   1319
SW9   = 001000          338#   488
```

```
S.TC    005572  1139*  1274
TCCA    011356  1099   1112   1823#
TCCS    011350   754*  1084*  1099   1112   1119   1124   1132*  1133   1145*  1148   1162*  1165   1554*
                1820#
TCDB    011360  1157   1174   1824#
TCER    011352  1129   1150   1167   1821#
TCTRAP  011412   474*   752*  1096*  1109*  1140*  1161*  1834#
TCWC    011354  1097*  1110*  1822#
TC.TA   005760   753*  1089*  1090   1092*  1141   1174   1179#
TC.WC   005762  1180#
TMBC    011366  1195*  1196*  1219*  1220*  1827#
TMCA    011370  1199   1223   1828#
TMCS    011362   748*  1190*  1199   1207*  1223   1232   1237   1825#
TMER    011364  1242   1826#
TMRET   006140  1205*  1210   1226#
TMTRAP  011416   475*   747*  1194*  1204*  1218*  1835#
TP.RP   004452   948#   992
TRAPS   006252   468   1253#
TRAP4   010326   692    758    1612   1618#
TRP.RC  004164   727    889#
TRP.RF  003676   737    827#
TRP.RK  005004   742   1022#
TRP.RP  004624   732    986#
TRP.TC  005314   752   1088#
TRP.TM  005772   747   1190#
TTY    =000005   396#   537*   571*   612*   661*   872*   877*   934*   939*  1002*  1007*  1067*  1072*
                1124*  1129*  1237*  1242*  1306*  1322*  1323   1325*  1374*  1377*  1407*  1415*  1473*
                1474*  1478   1495*  1499*  1529*  1532*  1540*  1542*  1556*  1559*  1562*  1581*  1589*
                1591   1623*  1637*  1640*  1643*  1896   1897*  1898   1900   1901*  1902   1904   1912
                1920*  1921*  1922   1923*  1971*  1972*  1973*  1975*  1977*  1983*  1985*  198?*
TTYIN   002666   685   692#   763
TYPE   = 000004   385#   536    539    542    544    558    570    573    599    600    651    693    761
                 770    771    772    873    874    876    935    936    938   1003   1004   1006   1068
                1069   1071   1125   1126   1128   1238   1239   1241   1297   1305   1321   1370   1376
                1528   1531   1555   1558   1561   1620   1634   1639   1642   1882   1941   1944   1948
                2001   2029
TYPEIT  001430   511    531#
T1      003136   726    730#
T1XOR8  007242  1427#
T2      003166   731    735#
T3      003216   736    740#
T3XOR9  007320  1424   1447#
T3X9    007314  1445#  1453
T4      003246   741    745#
T5      003276   746    750#
T6      003334   751    756#
T8      003356   757    760#
T8XOR1  007276  1425   1438#
WADRA   011262   611*   648*   686*   701    783    788    790   1406   1496   1538   1579*  1594*  1792#
WADRAX  011250   601    608    615    635   1786#
WAIT    010312   729    734    739    744    749    755   1612#
WANT    005756  1141*  1142*  1157   1178#
WCSRA   011236   546    549*   552    555*   571    578    678*   679*   723*   724*  1347   1350*  1351
                1355*  1356*  1357   1363   1377   1618   1632   1640   1671*  1672*  1674*  1679*  1680*
                1682*  1689*  1690*  1691*  1698*  1699*  1700*  1706*  1707   1781#
WDARA   011244  1643   1784#
```

```
WDBRA   011240            579     698*   1591*   1782*
WDBRB   011242           1697    1783*
WIND    011212            597    1765*
WINDOW  010404            675    1619    1632*   1709
WINTAB  010554           1654    1657*
WRADR   007446           1266    1491*
WRARA   011246            585*    586    1697*   1785*
WRCADR  007724           1267    1538*
WRDCNT  011424            663*    666*    670*    674*    831     843     856     893     905     918     952     964     977
                         1026    1038    1051    1097    1110    1195    1219    1407    1495    1540    1838*
WRITE = 000021            369*    858     920     979    1053    1107    1221
WVARA   011252            650     661    1787*
WVECA   011256            582     658*    660*    675*    677*   1790*
W.BANK  010712           1661    1697*
W.FUTR  010574           1662    1663    1667*
W.NOP   010576           1657    1671*
W.RC    004160            884*    917
W.READ  010626           1658    1679*
W.RF    003672            822*    855
W.RK    005000           1017*   1050
W.RP    004446            946*    976
W.R.W   010660           1660    1687*
W.TC    005302           1082*   1109
W.TM    005766           1195*   1218
W.WRIT  010660           1659    1688*
W.ZAP   010744           1664    1706*
X.TM    006070           1204    1210*
.       = 017500          401*    402     403*    409*    415*    435     447*    536     537*    539     540*    542     543*
                          544     558     559*    570     571*    573     600     601*    651     652*    693     694*    761
                          762*    770     772     873     935     988     990    1003    1012    1068    1077    1125    1134
                         1238    1258    1261    1285    1305    1321    1370    1371*   1376    1377*   1401    1410    1459
                         1528    1529*   1531    1532*   1550    1553    1555    1556*   1558    1561    1614    1620    1621*
                         1634    1639    1642    1738    1749    1882    1909    1914    1935    1941    1951*   1961    1990
                         1993    1998    2004*   2005    2029    2037*   2038*   2040*   2042*
.BEGIN  007354           1280    1468*
.BIT  = 122000            323*    331     332     333     334     335     336     337     338     339
.LOD    011524           1859*   1866    1872*
.MAMK   007516            478    1507*
.PARSR  007610           1507    1522*
.PR     012322           1956*   1960*   1962*   1963*   1965    1969    1979*   1981    1991*   1992    1995*   1997    2001
                         2004*
.PRF    012242           1967    1982    1987*
.PRL    012220           1980*   1996
.PTIT   012144           1959    1964*
.START= 104440            834     846     859     896     908     921     955     967     980    1029    1041    1054    1098
                         1111    1198    1222    1280*
.TYPE   011742           1900*   1901    1927*   1943*   1944
```

```
BITYPE      1#    399#
BLKBLK      1#
BUFHDR      1#
DECML       1#    399#
DEVCHK    447#    865    927    995   1060   1117   1230
DUMP        1#    399#   571    877    939   1007   1072   1129   1242   1377   1529   1532   1559   1562   1623
          1637   1640   1643
DUMP18      1#    323#   612   1556
FILBLK      1#
FUNCT     447#    697    699    700    717    782    791    833    845    858    866    895    907    920    928
           954    966    979    996   1028   1040   1053   1061   1094   1107   1118   1197   1221   1231   1403
          1417   1492   1501   1539   1544   1575   1596   1884
GO        447#    833    845    858    895    907    920    954    966    979   1028   1040   1053   1197   1221
LNKBLK      1#
ODT11       1#
ODT11X      1#
POP         1#    399#  2021
PRINT       1#    399#   536    539    542    544    558    569    573    600    651    693    760    770    772
           873    435   1003   1068   1125   1238   1305   1321   1370   1376   1527   1531   1555   1558   1561
          1620   1634   1639   1642   1882   1940   2029
PUSH        1#    399#  2007
SCOPE.      1#
SCOP        1#
SDUMP       1#    399#   537    661   1306   1374
SET       1246#  1264   1265   1266   1267   1268   1269   1270   1271   1272   1273   1274   1275   1276   1277
          1278   1279   1280   1281   1282   1283   1284
TRACE       1#    323#
TPNBLK      1#
TYPEM       1#
WRITE       1#
$CATCH      1#    323#   401
$CMTAG      1#    323#   447
$DECML      1#
$END        1#    323#
$EQUAT      1#    323#   383
$HLT        1#    323#
$KRAT       1#
$LOADR      1#    323#  1839
$OCTAL      1#    323#  1952
$PRINTER    1#    323#  2005
$RAND       1#
$RAND4      1#
$READ       1#    323#  1930
$SCOPE      1#    323#
$SET      1246#  1264   1265   1266   1267   1268   1269   1270   1271   1272   1273   1274   1275   1276   1277
          1278   1279   1280   1281   1282   1283   1284
$SETUP      1#    323#
$SRAT       1#
$SWDOC      1#    323#   328
$TRAP       1#    323#  1246
$TYPE       1#    323#  1887
$URAT       1#
.SCOP       1#
.SCOPE      1#
```

```
ADD     506   522   555   595   644   789  1089  1139  1256  1385  1430  1432  1445  1472  1475
       1480  1720  1858  1917  1920
ASL     526   604   609   610   616   619  1196  1220  1371  1516  1589  1971  1972  1973
ASR     590   594   707   711   712   714   715   716  1361  1362
ASRB   1974  1976
BCC     617   620  1517
BCS    1434  1588
BEQ     435   486   541   593   665   669   673   682   706   710  1300  1304  1310  1320  1324
       1344  1346  1352  1366  1381  1383  1401  1410  1550  1584  1708  1749  1903  1933  1990
       1993
BGT     524  1091  1158  1425
BIC     528   640   678   686   687   723  1325  1364  1579  1580  1653  1671  1674  1682  1691
       1698  1700  1706  1752  1856  1921
BICB   1355  1747  1937
BIS     618   621   679   722   724   728   733   738   743   748   754  1474  1594  1595  1679
       1689
BISB   1350  1356  1672  1680  1690  1699  1978  1994
BIT     434   488   664   668   672   684   703   705   709   870   932  1000  1065  1122  1235
       1291  1295  1299  1303  1319  1323  1345  1347  1351  1357  1382  1549  1577  1583  1707
       1898  1981
BLT     653   655  1294
BMI    1151  1168  1258  1309  1424  1619
BNE     433   489   496   510   557   583   607   632   647   685   704   726   731   736   741
        746   751   757   787   871   933  1001  1066  1123  1154  1171  1175  1236  1292  1296
       1348  1358  1414  1416  1500  1543  1553  1578  1593  1722  1862  1869  1879  1899  1905
       1911  1919  1939  1946  1982  1996  1998  2020
BPL     868   930   988   990   998  1012  1063  1077  1120  1134  1149  1166  1233  1459  1633
       1850  1909  1914  1935
BR      511   547   553   560   574   596   657   667   671   708   713   794   992  1155  1159
       1172  1261  1312  1314  1316  1354  1384  1446  1453  1590  1614  1724  1738  1915  1942
       1959  1961  1967
CLR     430   461   462   480   482   483   484   494   505   529   591   602   614   626   676
        698   721   762   769   792   827   832   844   857   865   889   894   906   919   927
        948   953   965   978   991   995  1022  1027  1039  1052  1060  1117  1190  1230  1311
       1342  1353  1404  1405  1457  1493  1494  1613  1673  1739  1883  1885  1962  2018
CLRB   1947  1966  1980  2000
CMP     432   485   509   523   548   554   576   582   652   654   725   730   735   740   745
        750   756   786  1090  1157  1174  1257  1293  1365  1409  1471  1479  1552  1719  1849
       1861  1868  1918  1932  1997
CMPB   1748  1904  1938  1945
COM    1435  1436  1541
COMB    492  1313
DEC     531   556   606   631   646  1721  1910
EMT     384
HALT    402   436   446   559   575   656  1260  1379  1460  1522  1534  1625  1645  1667  1723
       2015  2032
INC     507   518   534   869   931   999  1064  1121  1234  1301  1317  1415  1499  1542  2019
INCB   1991  1995
IOT     385
JMP     405   413   417   418   419   438   487   729   734   739   744   749   755   759   763
        773  1176  1210  1259  1654  1709  1743  1886  2030
JSR     445   478   535   538   572   613   662   694   695   790   793   828   875   878   890
        937   940   949  1005  1008  1023  1070  1073  1088  1127  1130  1191  1240  1243  1307
       1326  1375  1378  1408  1497  1530  1533  1557  1560  1563  1624  1638  1641  1644  1735
MOV     410   411   412   427   431   437   463   464   465   466   467   468   469   470   471
        472   473   474   475   476   477   479   481   490   493   503   504   519   520   525
```

	532	533	537	543	545	549	550	551	571	577	578	579	581	584	585
	586	587	597	598	601	603	608	611	612	615	622	623	625	627	628
	629	630	633	634	635	641	642	643	645	648	649	650	658	660	661
	663	666	670	674	675	677	680	683	692	696	701	702	719	720	727
	732	737	742	747	752	753	758	780	783	784	785	788	830	831	842
	843	855	856	872	877	892	893	904	905	917	918	934	939	951	952
	963	964	976	977	986	1002	1007	1010	1025	1026	1037	1038	1050	1051	1067
	1072	1075	1084	1092	1096	1097	1109	1110	1124	1129	1132	1140	1141	1143	1145
	1161	1162	1194	1195	1204	1205	1207	1218	1219	1237	1242	1253	1298	1302	1306
	1318	1322	1349	1359	1360	1363	1367	1368	1369	1374	1377	1406	1407	1427	1438
	1439	1447	1449	1468	1469	1470	1473	1476	1477	1478	1485	1496	1498	1507	1508
	1509	1510	1511	1512	1513	1514	1515	1518	1519	1529	1532	1538	1540	1554	1556
	1559	1562	1581	1582	1585	1591	1601	1602	1603	1604	1605	1606	1612	1621	1623
	1632	1635	1637	1640	1643	1681	1688	1697	1714	1715	1716	1732	1733	1734	1736
	1737	1740	1741	1742	1751	1748	1851	1857	1859	1860	1865	1866	1867	1881	1895
	1896	1897	1900	1901	1916	1922	1923	1924	1930	1931	1949	1958	1964	1965	1967
	1969	1979	2002	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2017	2021
	2022	2023	2024	2025	2026	2027	2028								
MOVB	1255	1746	1906	1907	1912	1936	1943	1960	1963	1970	1999				
NEG	583	1586													
NEGB	798	800	803	805	806	808	811	813							
NOP	437														
RESET	428	1731													
ROL	605	1373	1429	1441	1442	1443	1444	1983	1985	1987					
ROLB	801	802	809	810	1384	1996	1988								
ROR	636	637	638	639	1431	1433	1448	1451	1452	1587	1852	1853	1854	1855	1975
	1977														
RORB	799	804	807	812											
RTI	635	848	861	880	898	910	923	942	957	969	982	1013	1031	1043	1056
	1078	1100	1113	1135	1146	1163	1200	1208	1224	1245	1328	1386	1402	1419	1461
	1481	1502	1512	1523	1535	1545	1551	1565	1598	1607	1626	1646	1675	1683	1692
	1701	1750	1753	1925											
RTS	814	1437	1520	1718	1863	1870	1880	1950	2003						
SEC	1372														
SUB	429	1142	1254	1622	1636										
SUBB	527	589	1315	1428	1440	1450	1652								
TRAP	383	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277
	1278	1279	1280	1281	1282	1283	1284								
TST	491	495	508	521	540	546	552	580	592	624	659	681	867	929	989
	997	1062	1119	1144	1148	1150	1153	1165	1167	1170	1206	1232	1308	1343	1380
	1400	1412	1413	1423	1458	1592	1618	1717	1878						
TSTB	987	1011	1075	1133	1902	1908	1913	1934	1989	1992					
.ASCII	1765														
.ASCIZ	537	540	543	545	559	571	574	601	652	694	762	771	773	874	936
	1004	1069	1126	1239	1306	1322	1371	1377	1529	1532	1556	1559	1562	1621	1635
	1640	1643	1760	1762	1883	1942	2030								
.BLKB	2042														
.BLKW	1951	2004	2037	2038											
.ENABL	1	323													
.END	2044														
.ENDC	399	873	880	935	942	1003	1013	1068	1078	1125	1135	1238	1245	1849	1866
	1870	1902	1926	1933	1952	1980	2005	2013	2021	2030	2031				
.EVEN	537	540	543	545	559	571	574	601	652	694	762	771	773	874	936
	1004	1069	1126	1239	1306	1322	1371	1377	1529	1532	1556	1559	1562	1621	1635
	1640	1643	1764	1883	1942	2030									
.IF	399	873	880	935	942	1003	1010	1068	1075	1125	1132	1238	1245	1848	1865

DZDABA.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
        1870  1898  1916  1932  1951  1952  1968  2005  2013  2021  2029  2030
.IFF    1848  1865  1926  1933  1952  2030
.IIF     331   332   333   334   335   336   337   338   339  1951  2005
.IRP    2007  2021
.LIST      1   323   399   402   447   537   540   543   545   559   571   574   601   652   688
         694   762   771   773   774   819   865   874   881   927   936   943   995  1004  1014
        1060  1069  1079  1117  1126  1182  1230  1239  1246  1264  1265  1266  1267  1268  1269
        1270  1271  1272  1273  1274  1275  1276  1277  1278  1279  1280  1281  1282  1283  1284
        1285  1296  1306  1322  1371  1377  1529  1532  1556  1559  1562  1608  1621  1635  1640
        1643  1799  1839  1883  1887  1930  1942  1952  2005  2030
.MACRO     1   447  1246
.MCALL   323   399
.NLIST     1   323   399   402   447   537   540   543   545   559   571   574   601   652   688
         694   762   771   773   774   819   865   874   881   927   936   943   995  1004  1014
        1060  1069  1079  1117  1126  1182  1230  1239  1246  1264  1265  1266  1267  1268  1269
        1270  1271  1272  1273  1274  1275  1276  1277  1278  1279  1280  1281  1282  1283  1284
        1285  1296  1306  1322  1371  1377  1529  1532  1556  1559  1562  1608  1621  1635  1640
        1643  1799  1839  1883  1887  1930  1942  1952  2005  2030
.PAGE    993  1246  1887  1952  2005
.REM       1
.REPT    402
.SBTTL   447   688   774   819   865   881   927   943   995  1014  1060  1079  1117  1182  1230
        1246  1286  1608  1799  1839  1887  1930  1952  2005
.TITLE   323
```

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

*,DZDABA.SEQ/SOL/CRF/PAGNUM/NL:TOC=STEVE.SML,DZDABA.P11
RUN-TIME: 22 28 3 SECONDS
RUN-TIME RATIO: 369/54=6.7
CORE USED: 26K (51 PAGES)

D06

Spooler runtime 0 Seconds, 36 KCS, 210 disk reads, 4 disk writes, 67 pages