CXUACA.MAC 13-JAN-83 16:32

000000 .REPT 0

IDENTIFICATION

PRODUCT CODE: AC-T367A-MC

PRODUCT NAME: CXUACA0 DEC/X11 DIAG

PRODUCT DATE: JANUARY 1983

MAINTAINER: DISTRIBUTED SYSTEMS DIAGNOSTIC ENGINEERING

AUTHOR: CHERYL CONROY

.PAGE

1. ABSTRACT

THE DEUNA MODULE IS AN IOMODX THAT EXERCISES UP TO EIGHT DEUNA'S ON THE UNIBUS. THE DEUNA TRANSMITS PACKETS AT A STEADY RATE. RECEIVE PACKETS ARE STORED IN THE DEUNA UNTIL A LARGE (10 PACKETS) TRANSFER CAN BE INITIATED. THIS WILL RESULT IN A BURST MODE OF UNIBUS ACTIVITY FOLLOWED BY A RELATIVELY LOW LEVEL.

2. REQUIREMENTS

HARDWARE:

- PDP-11 CPU
- 28K WORDS OF MEMORY
- CONSOLE TERMINAL
- DEUNA BOARDSET (M7792, M7793)
- PLUS, ONE OF THE FOLLOWING:
  - LINK BOARD TO BULKHEAD CABLE CONNECTED AND BULKHEAD TO TRANSCEIVER TAP CABLE CONNECTED (NORMAL ONLINE CONFIGURATION)

  OR
  - LINK BOARD TO BULKHEAD CABLE CONNECTED AND BULKHEAD TO FIELD SERVICE EXTERNAL LOOPBACK CONNECTOR INSTALLED (OFFLINE CONFIGURATION)

3. PASS IDENTIFICATION

ONE PASS OF THE DEUNA MODULE CONSISTS OF 1000 ITERATIONS OF:

- PASS INITIALIZATION
  - HALT THE DEUNA
  - SET THE PORT CONTROL BLOCK
  - INITIALIZE THE TRANSMIT BUFFERS AND THE EXPECTED CRC'S
  - INITIALIZE RING BUFFER STRUCTURE
  - START THE DEUNA
- TRANSMISSION
- RECEPTION
- ANALYSIS
  - COMPARE THE BYTE COUNT
  - COMPARE THE EXPECTED CRC
  - COMPARE THE DATA PACKET ON CRC ERROR

.PAGE

4. EXECUTION TIME

ONE PASS RUNNING ALONE ON THE PDP-11/24 TAKES AN AVERAGE OF 1 MINUTE FOR ONE DEUNA.

5. CONFIGURATION REQUIREMENTS

DEFAULT PARAMETERS:

DEVADR: 170000, VECTOR: 700, BR1: 5, DEVCNT: 1

REQUIRED PARAMETERS:

NONE

6. DEVICE/OPTION SETUP

7. MODULE OPERATION

A. PROGRAM INITIALIZATION

B. PASS INITIALIZATION

HALT THE DEUNA

SET THE PORT CONTROL BLOCK

INITIALIZE THE TRANSMIT BUFFERS AND EXPECTED CRC'S

INITIALIZE THE RING BUFFER STRUCTURE

START THE DEUNA

C. TRANSMISSION

D. RECEPTION

E. ANALYSIS

COMPARE THE BYTE COUNT

COMPARE THE EXPECTED CRC

COMPARE THE DATA PACKET ON CRC ERROR

8. OPERATION OPTIONS

SR1 BIT 2 SET(1):
RUN IN INTERNAL LOOPBACK MODE.

SR1 BIT 2 CLEAR(0):
RUN IN EXTERNAL LOOPBACK MODE. THIS IS THE DEFAULT MODE.

.PAGE

9. NON-STANDARD PRINTOUTS

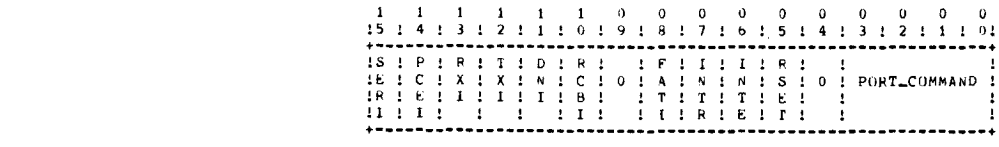
A. ALL PRINTOUTS HAVE THE STANDARD FORMAT DESCRIBED IN THE DEC/X11 DOCUMENT.

B. ERROR MESSAGES DUMP THE CONTENTS OF THE DEUNA DESCRIPTOR RINGS IN THE FOLLOWING ORDER.

CSRA	= PCSR0 ADDRESS	(1 WORD)
ACSR	= PCSR0 CONTENTS	(1 WORD)
ASTAT	= CONTENTS OF PCSR1	(1 WORD)
ERRTYP	= 0 (UNDEFINED)	(1 WORD)
TRANSMIT DESCRIPTOR RING		(5 WORDS)
RECEIVE DESCRIPTOR RING		(7 WORDS)

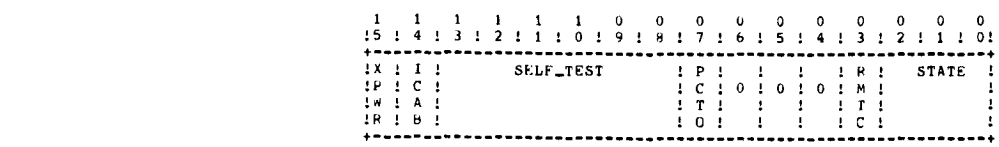
10. DEVICE REGISTERS

PORT CONTROL AND STATUS REGISTER 0 - PCSR0 -



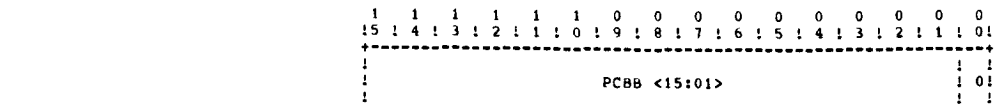
WORD	BITS	FIELD	DESCRIPTION
PCSR0	<15>	SERI	STATUS ERROR INTERRUPT
PCSR0	<14>	PCEI	PORT COMMAND ERROR INTERRUPT
PCSR0	<13>	RXI	RECEIVE RING INTERRUPT
PCSR0	<12>	TXI	TRANSMIT RING INTERRUPT
PCSR0	<11>	DNI	DONE INTERRUPT
PCSR0	<10>	RCBI	RECEIVE BUFFER UNAVAILABLE INTERRUPT
PCSR0	<09>	ZERO	
PCSR0	<08>	FATI	FATAL ERROR INTERRUPT
PCSR0	<07>	INTR	INTERRUPT SUMMARY
PCSR0	<06>	INTE	INTERRUPT ENABLE
PCSR0	<05>	RSET	UNA RESET
PCSR0	<04>	ZERO	
PCSR0	<03:00>	PORT_COMMAND	

PORT CONTROL AND STATUS REGISTER 1 - PCSR1 -



WORD	BITS	FIELD	DESCRIPTION
PCSR1	<15>	XPWR	TRANSCEIVER POWER FAILURE
PCSR1	<14>	ICAB	PORT/LINK CABLING FAILURE
PCSR1	<13:08>	SELF_TEST	SELF TEST ERROR CODE
PCSR1	<07>	PCTO	PORT COMMAND TIMEOUT
PCSR1	<06:04>	ZEROES	
PCSR1	<03>	RMTC	REMOTE CONSOLE RESERVED.
PCSR1	<02:00>	STATE	

PORT CONTROL AND STATUS REGISTER 2 - PCSR2 -



```
                        !                                                       ! !
                        +-------------------------------------------------------+

                        WORD        BITS      FIELD     DESCRIPTION
                        ----        ----      -----     -----------

                        PCSR2       <15:00>   PCBB      THE LOW ORDER 16 BITS OF THE
                                                        ADDRESS OF THE PORT CONTROL
                                                        BLOCK BASE.

                PORT CONTROL AND STATUS REGISTER 3 - PCSR3 -

                         1   1   1   1   1   1   0   0   0   0   0   0   0   0   0   0
                        !5 ! 4 ! 3 ! 2 ! 1 ! 0 ! 9 ! 8 ! 7 ! 6 ! 5 ! 4 ! 3 ! 2 ! 1 ! 0!
                        +-------------------------------------------------------------+
                        !                                                   ! PCBB !
                        !0   0   0   0   0   0   0   0   0   0   0   0   0 ! 17:16!
                        !                                                   !      !
                        !                                                   !      !
                        +-------------------------------------------------------+

                        WORD        BITS      FIELD     DESCRIPTION
                        ----        ----      -----     -----------

                        PCSR3       <15:02>   ZEROS

                        PCSR3       <01:00>   PCBB      THE HIGH ORDER 2 BITS OF THE
                                                        ADDRESS OF THE PORT CONTROL
                                                        BLOCK BASE.


                .ENDP
```



```
                        .DSABL  GBL
                        .ENABL  LC
                        .TITLE  UNA_DECX

                .MACRO  PRINT   MESSAGE,?A,?B
                        MSG     A
                        BR      B
                A:      .ASCII  '%'
                        .ASCIZ  /'MESSAGE'/
                        .EVEN
                B:
                .ENDM
                .MACRO  DEMAND  ACTION,MESSAGE,?A,?B
                        JSR     PC,SETCMD
                        .WORD   ACTION
                        BCC     B
                        MSG     A
                        HRDER   ERRPRT <>                                       ;CC
                        JMP     DROP
                A:      .ASCII  '%'
                        .ASCIZ  /'MESSAGE'/
                        .EVEN
                B:
                .ENDM
                .MACRO  WAIT    STATE,MESSAGE,?A,?B
                        JSR     PC,WSTATE
                        .WORD   STATE
                        BCC     B
                        MSG     A
                        HRDER   ERRPRT <>                                       ;CC
                        JMP     DROP
                A:      .ASCII  '%'
                        .ASCIZ  /'MESSAGE'/
                        .EVEN
                B:
                .ENDM
002000          BUFSIZ=         1024.
166670          POLYH=          166670          ;CRC FUNCTION POLYNOMIAL HIGH WORD
101440          POLYL=          101440          ;CRC FUNCTION POLYNOMIAL LOW WORD

000000          .PCSR0=         0               ;OFFSET TO DEUNA PCSR0
000002          .PCSR1=         2               ;OFFSET TO DEUNA PCSR1
000004          .PCSR2=         4               ;OFFSET TO DEUNA PCSR2
000006          .PCSR3=         6               ;OFFSET TO DEUNA PCSR3

                ;       PORT COMMANDS - PCSR0

000000          .NOP=           0               ;NOP
000001          .PCBB=          1               ;GET PORT CONTROL BLOCK
000002          .CMD=           2               ;FETCH & EXECUTE A COMMAND
000003          .STEST=         3               ;RUN SELF TEST
000004          .START=         4               ;ENABLE DEUNA TRANSMISSION & RECEPTION
000005          .BOOT=          5               ;ENTER PRIMARY LOAD MODE
000010          .PDMD=          10              ;POLLING DEMAND
000017          .STOP=          17              ;STOP DEUNA
```

```
                        ;       STATUS DEFINITION BITS - PCSR0

000040                  .REST=          BIT5                    ;RESET DEUNA
000100                  .INTE=          BIT6                    ;INTERRUPT ENABLE
000400                  .FATI=          BIT8                    ;FATAL ERROR INTERRUPT
002000                  .RCBI=          BIT10                   ;RECEIVER BUFFER UNAVAILABLE INTERRUPT
004000                  .DNI=           BIT11                   ;DONE INTERRUPT
010000                  .TXI=           BIT12                   ;TRANSMIT RING INTERRUPT
020000                  .RXI=           BIT13                   ;RECEIVE RING INTERRUPT
040000                  .PCEI=          BIT14                   ;PORT COMMAND ERROR INTERRUPT
100000                  .SERI=          BIT15                   ;STATUS ERROR INTERRUPT
142400                  .ERRI=          .SERI+.PCEI+.RCBI+.FATI ;ANY ERROR INTERRUPT

                        ;       NI STATUS CODES - FROM PCSR1

000007                  .STATE=         BIT2+BIT1+BIT0          ;DEUNA STATE MASK
000000                  .RESET=         0                       ;RESET
000001                  .PLOAD=         BIT0                    ;PRIMARY LOAD
000002                  .READY=         BIT1                    ;READY
000003                  .RUN=           BIT1+BIT0               ;RUNNING
000005                  .UHALT=         BIT2+BIT0               ;UNIBUS HALTED
000006                  .NHALT=         BIT2+BIT1               ;NI HALTED
000007                  .BHALT=         BIT2+BIT1+BIT0          ;NI & UNIBUS HALTED

                        ;       SOFTWARE  SWITCH REGISTER DEFINITION BITS

000001                  .DCHECK=        BIT0                    ;PERFORM BUFFER COMPARSIONS
000002                  .CHKALL=        BIT1                    ;FULL BUFFER DATA COMPARSIONS
000004                  .INTERN=        BIT2                    ;INTERNAL LOOPBACK

                        ;       MODE REGISTER BIT DEFINITIONS

000001                  .FDX=           BIT0                    ;FULL DUPLEX
000004                  .LOOP=          BIT2                    ;INTERNAL LOOPBACK

                        ;       RING DESCRIPTOR BLOCK

000005                  .TELEN=         5                       ;SIZE TRANSMIT DESCRIPTOR RING
000007                  .RELEN=         7                       ;SIZE RECEIVE DESCRIPTOR RING
000012                  .MXPKT=         10.                     ;MAXIMUM NUMBER OF PACKETS

000000                  .SLEN=          0                       ;OFFSET TO SEGMENT LENGTH
000002                  .SBUF=          2                       ;OFFSET TO SEGMENT BUFFER
000004                  .STAT=          4                       ;OFFSET TO STATUS WORD
000006                  .MLEN=          6                       ;OFFSET TO MESSAGE LENGTH WORD
000006                  .TDR=           6                       ;OFFSET TO TDR
000010                  .VA=            10                      ;OFFSET TO VIRTUAL ADDRESS
000012                  .CRCL=          12                      ;OFFSET TO CRC LOW WORD
000014                  .CRCH=          14                      ;OFFSET TO CRC HIGH WORD

000400                  .ENP=           BIT8                    ;END OF PACKET
001000                  .STP=           BIT9                    ;START OF PACKET
020000                  .MATCH=         BIT13                   ;REC EQUALS XMIT MATCH BIT

004000                  .CPC=           BIT11                   ;CRC FRAME CHECK ERROR
010000                  .OFLO=          BIT12                   ;MESSAGE OVERFLOW ERROR
```



```
020000                  .FRAM=          BIT13                   ;FRAME ERROR
040000                  .ERRS=          BIT14                   ;ERROR SUMMARY
100000                  .OWN=           BIT15                   ;OWNERSHIP BIT,  0=HOST , 1=DEUNA

020000                  .NCHN=          BIT13                   ;NO DATA CHAINING
040000                  .UBTO=          BIT14                   ;UNIBUS TIMEOUT
100000                  .BUFL=          BIT15                   ;BUFFER LENGTH FRAME ERROR
```

```
                                        .LIST   MEB
000000'                                 IOMODX <UNAA >,170000,700,5,0,1,1000.,,0,RECBUF,BUFSIZ,BUFSIZ
000000'                                 MODULE  150000,UNAA ,170000,700,5,0,1,1000.,,0,RECBUF,BUFSIZ,BUFSIZ
                                        .TITLE  UNA DEC/X11 SYSTEM EXERCISER MODULE
                                ;       DDXCOM  VERSION  6      23-MAY-78
000000'                                 BEGIN:
000000' 047125  040501      040 MODNAM: .ASCII  /UNAA /         ;MODULE NAME.
000005'    000                  XFLAG:  .BYTE   OPEN                ;USED TO KEEP TRACK OF WBUFF USAGE
000006' 170000                  ADDR:   .WORD   170000              ;1ST DEVICE ADDR.
000010' 000700                  VECTOR: .WORD   700                 ;1ST DEVICE VECTOR.
000012'    240                  BR1:    .BYTE   PRTY5           ;1ST BR LEVEL.
000013'    000                  BR2:    .BYTE   PRTY0           ;2ND BR LEVEL.
000014' 000002                  DVID1:  .WORD   1+1                 ;DEVICE INDICATOR 1.
000016' 000000                  SR1:    .WORD   OPEN                ;SWITCH REGISTER 1
000020' 000000                  SR2:    .WORD   OPEN                ;SWITCH REGISTER 2
000022' 000000                  SR3:    .WORD   OPEN                ;SWITCH REGISTER 3
000024' 000000                  SR4:    .WORD   OPEN                ;SWITCH REGISTER 4

000026' 150000                  STAT:   .WORD   150000              ;STATUS WORD.
000030' 000252'                 INIT:   .WORD   START               ;MODULE START ADDR.
000032' 000252'                 SPOINT: .WORD   MODSP               ;MODULE STACK POINTER.
000034' 000000                  PASCNT: .WORD   0                   ;PASS COUNTER.
000036' 001750                  ICONT:  .WORD   1000.               ;# OF ITERATIONS PER PASS=1000.
000040' 000000                  ICOUNT: .WORD   0                   ;LOC TO COUNT ITERATIONS
000042' 000000                  SOFCNT: .WORD   0                   ;LOC TO SAVE TOTAL SOFT ERRORS
000044' 000000                  HRDCNT: .WORD   0                   ;LOC TO SAVE TOTAL HARD ERRORS
000046' 000000                  SOFPAS: .WORD   0                   ;LOC TO SAVE SOFT ERRORS PER PASS
000050' 000000                  HRDPAS: .WORD   0                   ;LOC TO SAVE HARD ERRORS PER PASS
000052' 000000                  SYSCNT: .WORD   0                   ;# OF SYS ERRORS ACCUMULATED
000054' 000000                  RANNUM: .WORD   0                   ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000056'                         CONFIG:                             ;RESERVED FOR MONITOR USE
000056' 000000                  RES1:   .WORD   0                   ;RESERVED FOR MONITOR USE
000060' 000000                  RES2:   .WORD   0                   ;RESERVED FOR MONITOR USE
000062' 000000                  SVR0:   .WORD   OPEN                ;LOC TO SAVE R0.
000064' 000000                  SVR1:   .WORD   OPEN                ;LOC TO SAVE R1.
000066' 000000                  SVR2:   .WORD   OPEN                ;LOC TO SAVE R2.
000070' 000000                  SVR3:   .WORD   OPEN                ;LOC TO SAVE R3.
000072' 000000                  SVR4:   .WORD   OPEN                ;LOC TO SAVE R4.
000074' 000000                  SVR5:   .WORD   OPEN                ;LOC TO SAVE R5.
000076' 000000                  SVR6:   .WORD   OPEN                ;LOC TO SAVE R6.
000100' 000000                  CSRA:   .WORD   OPEN                ;ADDR OF CURRENT CSR.
000102'                         SRADR:                              ;ADDR OF GOOD DATA, OR
000102' 000000                  ACSR:   .WORD   OPEN                ;CONTENTS OF CSR.
000104'                         WASADR:                             ;ADDR OF BAD DATA, OR
000104' 000000                  ASTAT:  .WORD   OPEN                ;STATUS REG CONTENTS.
000106'                         ERRTYP:                             ;TYPE OF ERROR
000106' 000000                  ASR:    .WORD   OPEN                ;EXPECTED DATA.
000110' 000000                  AWAS:   .WORD   OPEN                ;ACTUAL DATA.
000112' 000274'                 RSTRT:  .WORD   RESTRT               ;RESTART ADDRESS AFTER END OF PASS
000114' 000000                  WDTU:   .WORD   OPEN                ;WORDS TO MEMORY PER ITERATION
000116' 000000                  WDFR:   .WORD   OPEN                ;WORDS FROM MEMORY PER ITERATION
000120' 000000                  INTR:   .WORD   OPEN                ;# OF INTERRUPTS PER ITERATION
000122' 000000                  IDNUM:  .WORD   0                   ;MODULE IDENTIFICATION NUMBER=0
           001                          .IF NB  <RECBUF>
000124' 006116'                 RBUFVA: .WORD   RECBUF              ;READ BUFFER VIRTUAL ADDRESS
000126' 000000                  RBUFPA: .WORD   OPEN                ;READ BUFFER PHYSICAL ADDRESS
```



```
000130' 000000                  RBUFEA: .WORD   OPEN                ;READ BUFFER EA BITS
000132' 002000                  RBUFSZ: .WORD   BUFSIZ              ;SIZE OF THE READ BUFFER
000134' 000000                  WBUFPA: .WORD   OPEN                ;WRITE BUFFER PHYSICAL ADDRESS
000136' 000000                  WBUFEA: .WORD   OPEN                ;WRITE BUFFER EA BITS
000140' 002000                  WBUFRQ: .WORD   BUFSIZ              ;WRITE BUFFER SIZE REQUESTED
000142' 000000                  WBUFSZ: .WORD   OPEN                ;WRITE BUFFER SIZE AVAILABLE
000144' 000000                  CDERCT: .WORD   OPEN                ;CDATA/DATCK ERROR COUNT
000146' 000000                  CDWDCT: .WORD   OPEN                ;CDATA/DATCK WORD COUNT
000150' 000000                  FREE:   .WORD   OPEN                ;RESERVED FOR FUTURE USE
        000000                          IMODX.=0
           000                          .ENDC
        000040                          .REPT   SPSIZ               ;MODULE STACK STARTS HERE.
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000152' 000000                          .WORD   0
000154' 000000                          .WORD   0
000156' 000000                          .WORD   0
000160' 000000                          .WORD   0
000162' 000000                          .WORD   0
000164' 000000                          .WORD   0
000166' 000000                          .WORD   0
000170' 000000                          .WORD   0
000172' 000000                          .WORD   0
000174' 000000                          .WORD   0
000176' 000000                          .WORD   0
000200' 000000                          .WORD   0
000202' 000000                          .WORD   0
000204' 000000                          .WORD   0
000206' 000000                          .WORD   0
000210' 000000                          .WORD   0
000212' 000000                          .WORD   0
000214' 000000                          .WORD   0
000216' 000000                          .WORD   0
000220' 000000                          .WORD   0
000222' 000000                          .WORD   0
000224' 000000                          .WORD   0
000226' 000000                          .WORD   0
000230' 000000                          .WORD   0
000232' 000000                          .WORD   0
000234' 000000                          .WORD   0
000236' 000000                          .WORD   0
000240' 000000                          .WORD   0
000242' 000000                          .WORD   0
000244' 000000                          .WORD   0
000246' 000000                          .WORD   0
000250' 000000                          .WORD   0
000252'                         MODSP:

000252'                                 EQUATS
                                ;EQUATE STATEMENTS
        000000                          OPEN=0
        177776                          PS=177776
        177776                          PSW=177776
```

```
                100000                  BIT15=100000
                040000                  BIT14=40000
                020000                  BIT13=20000
                010000                  BIT12=10000
                004000                  BIT11=4000
                002000                  BIT10=2000
                001000                  BIT9=1000
                000400                  BIT8=400
                000200                  BIT7=200
                000100                  BIT6=100
                000040                  BIT5=40
                000020                  BIT4=20
                000010                  BIT3=10
                000004                  BIT2=4
                000002                  BIT1=2
                000001                  BIT0=1
                001000                  ADDR22=1000
                000000                  R0=%0
                000001                  R1=%1
                000002                  R2=%2
                000003                  R3=%3
                000004                  R4=%4
                000005                  R5=%5
                000006                  R6=%6
                000007                  R7=%7
                000006                  SP=%6
                000007                  PC=%7
                000004                  PIRQ$=IOT
                000340                  PRTY7=340
                000300                  PRTY6=300
                000240                  PRTY5=240
                000200                  PRTY4=200
                000140                  PRTY3=140
                000100                  PRTY2=100
                000040                  PRTY1=40
                000000                  PRTY0=0
                000000                  PRTY=0
                005746                  PUSH=005746
                024646                  PUSH2=024646
                005726                  POPSP=005726
                022626                  POPSP2=022626
                000000                  TRPDFD=0
                000000                  NULL=0


                                ;DEFFERRED SERVICE:

        000252'                         DFSEVNT EXIT$
                104400                  EXIT$=TRAP+TRPDFD
                000001                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT MSG$
                104401                  MSG$=TRAP+TRPDFD
                000002                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT MSGS$
```



```
                104402                  MSGS$=TRAP+TRPDFD
                000003                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT MSGN$
                104403                  MSGN$=TRAP+TRPDFD
                000004                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT DATER$
                104404                  DATER$=TRAP+TRPDFD
                000005                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT HRDER$
                104405                  HRDER$=TRAP+TRPDFD
                000006                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT SOFER$
                104406                  SOFER$=TRAP+TRPDFD
                000007                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT BREAK$
                104407                  BREAK$=TRAP+TRPDFD
                000010                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT END$
                104410                  END$=TRAP+TRPDFD
                000011                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT DATCK$
                104411                  DATCK$=TRAP+TRPDFD
                000012                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT CDATA$
                104412                  CDATA$=TRAP+TRPDFD
                000013                  TRPDFD=TRPDFD+1
        000252'                         DFSEVNT ENDIT$
                104413                  ENDIT$=TRAP+TRPDFD
                000014                  TRPDFD=TRPDFD+1


                                ;DIRECT SERVICE:

        000252'                         DSEVNT  GWBUF$
                104414                  GWBUF$= TRAP + TRPDFD
                000015                  TRPDFD=TRPDFD+1
        000252'                         DSEVNT  GETPA$
                104415                  GETPA$= TRAP + TRPDFD
                000016                  TRPDFD=TRPDFD+1
        000252'                         DSEVNT  MAP22$
                104416                  MAP22$= TRAP + TRPDFD
                000017                  TRPDFD=TRPDFD+1
        000252'                         DSEVNT  RAND$
                104417                  RAND$= TRAP + TRPDFD
                000020                  TRPDFD=TRPDFD+1
        000252'                         DSEVNT  OTOA$
                104420                  OTOA$= TRAP + TRPDFD
                000021                  TRPDFD=TRPDFD+1
        000252'                         DSEVNT  BTOD$
                104421                  BTOD$= TRAP + TRPDFD
                000022                  TRPDFD=TRPDFD+1
```

```
000252' 016701  177532          START:  MOV     VECTOR,R1               ;GET INTERRUPT VECTOR ADDRESS
000256' 012721  002456'                 MOV     #INTSRV,(R1)+           ;SET THE INTERRPUT VECTOR
000262' 116700  177524                  MOVB    @R1,R0                  ;GET DEVICE PRIORITY
000266' 042700  177400                  BIC     #177400,R0              ;CLEAR THE UPPER BYTE
000272' 010011                          MOV     R0,(R1)                 ;AND SET IN VECTOR MEMORY

000274' 016705  177506          RESTRT: MOV     ADDR,R5                 ;GET ADDRESS OF DEUNA UNIBUS CSR'S
000300' 012765  000040  000000  1$:     MOV     #.REST,.PCSR0(R5)       ;WRITE UPPER BYTE TO CLEAR ALL
000306' 012702  001750                  MOV     #1000.,R2               ;DEVICE TIMEOUT WAIT COUNT
000312' 032765  000400  000000  2$:     BIT     #.FATI,.PCSR0(R5)       ;TEST FOR FATAL ERROR
000320' 001440                          BEQ     3$                      ;CONTINUE IF NOT SET
000322'                                 PRINT   <FATAL ERROR - RESETTING DEUNA>
000322'                                 MSG     64$
000322' 104401  000000' 000332'         MSGS,BEGIN,64$          ;ASCII MESSAGE CALL
000330' 000420                          BR      65$
000332'    045                  64$:    .ASCII  '%'
000333'    106  072141  066141          .ASCIZ  /FATAL ERROR - RESETTING DEUNA/
000340' 042440  071162  071157
000346' 026440  051040  071545
000354' 072145  064564  063556
000362' 042040  052505  040516
000370'    000
        000372'                         .EVEN
000372'                         65$:
000372' 005067  005446                  CLR     FLAGS                   ;CLEAR INTERRUPTS FLAGS WORD    ;CC
000376' 056567  000000  005440          BIS     .PCSR0(R5),FLAGS        ;GET CONTENTS OF PCSR0          ;CC
000404'                                 HRDER   ERRPRT <FATAL ERROR - RESETTING DEUNA>          ;CC
000404' 004767  004430                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
           001                          .IF B <ERRPRT>
                                        HRDERS,BEGIN,NULL       ;FATAL ERROR - RESETTING DEUNA
                                        .IFF
000410' 104405  000000' 005762'         HRDERS,BEGIN,ERRPRT     ;FATAL ERROR - RESETTING DEUNA
           000                          .ENDC
000416' 000167  001402                  JMP     DROP                    ;SEEMS TO BE DEAD
000422' 032765  004000  000000  3$:     BIT     #.DNI,.PCSR0(R5)        ;WAIT FOR DEVICE TO COME READY
000430' 001040                          BNE     4$                      ;CONTINUE PROCESSING WHEN SET
000432'                                 BREAK                           ;TEMP RETURN TO DECX
000432' 104407  000000'                 BREAKS,BEGIN            ;TEMPORARY RETURN TO MONITOR....
000436' 104407  000000'                 BREAKS,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
000442' 005302                          DEC     R2                      ;DECREMENT TIMEOUT COUNTER
000444' 001322                          BNE     2$                      ;AND LOOP TILL EXPIRED
000446'                                 PRINT   <NO DNI AFTER RESET>
000446'                                 MSG     66$
000446' 104401  000000' 000456'         MSGS,BEGIN,66$          ;ASCII MESSAGE CALL
000454' 000412                          BR      67$
000456'    045                  66$:    .ASCII  '%'
000457'    116  020157  047104          .ASCIZ  /NO DNI AFTER RESET/
000464' 020111  063141  062564
000472' 020162  062562  062563
000500' 000164
                                        .EVEN
000502'                         67$:
000502' 005067  005336                  CLR     FLAGS                   ;CLEAR INTERRUPTS FLAGS WORD    ;CC
000506' 056567  000000  005330          BIS     .PCSR0(R5),FLAGS        ;GET CONTENTS OF PCSR0          ;CC
000514'                                 HRDER   ERRPRT  <NO DNI AFTER RESET>
000514' 004767  004320                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
```



```
           001                          .IF B <ERRPRT>
                                        HRDERS,BEGIN,NULL       ;NO DNI AFTER RESET
                                        .IFF
000520' 104405  000000' 005762'         HRDERS,BEGIN,ERRPRT     ;NO DNI AFTER RESET
           000                          .ENDC
000526' 000167  001272                  JMP     DROP                    ;ALL DONE FOLKS
000532' 012765  004000  000000  4$:     MOV     #.DNI,.PCSR0(R5)        ;CLEAR THE REGISTER
000540' 012765  000100  000000          MOV     #.INTE,.PCSR0(R5)       ;ENABLE INTERRUPTS
000546' 005067  005272                  CLR     FLAGS                   ;CLEAR THE INTERRUPT FLAG WORD

000552'                                 DEMAND  .STOP,<UNABLE TO STOP THE DEUNA>
000552' 004767  002000                  JSR     PC,SETCMD
000556' 000017                          .WORD   .STOP
000560' 103027                          BCC     69$
000562'                                 MSG     68$
000562' 104401  000000' 000606'         MSGS,BEGIN,68$          ;ASCII MESSAGE CALL
000570'                                 HRDER   ERRPRT <>                               ;CC
000570' 004767  004244                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
           001                          .IF B <ERRPRT>
                                        HRDERS,BEGIN,NULL       ;
                                        .IFF
000574' 104405  000000' 005762'         HRDERS,BEGIN,ERRPRT     ;
           000                          .ENDC
000602' 000167  001216                  JMP     DROP
000606'    045                  68$:    .ASCII  '%'
000607'    125  060556  066142          .ASCIZ  /UNABLE TO STOP THE DEUNA/
000614' 020145  067564  071440
000622' 067564  020160  064164
000630' 020145  042504  047125
000636' 000101
                                        .EVEN
000640'                         69$:
000640'                                 WAIT    .READY,<UNABLE TO ENTER READY STATE>
000640' 004767  002012                  JSR     PC,@STATE
000644' 000002                          .WORD   .READY
000646' 103031                          BCC     71$
000650'                                 MSG     70$
000650' 104401  000000' 000674'         MSGS,BEGIN,70$          ;ASCII MESSAGE CALL
000656'                                 HRDER   ERRPRT <>                               ;CC
000656' 004767  004156                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
           001                          .IF B <ERRPRT>
                                        HRDERS,BEGIN,NULL       ;
                                        .IFF
000662' 104405  000000' 005762'         HRDERS,BEGIN,ERRPRT     ;
           000                          .ENDC
000670' 000167  001130                  JMP     DROP
000674'    045                  70$:    .ASCII  '%'
000675'    125  060556  066142          .ASCIZ  /UNABLE TO ENTER READY STATE/
000702' 020145  067564  062440
000710' 072156  071145  071040
000716' 060545  074544  071440
000724' 060564  062564     000
        000732'                         .EVEN
000732'                         71$:
```

```
000736' 004767  004036                  JSR     PC,CVTPHY               ;CONVERT TO PHYSICAL
000742' 010067  004254                  MOV     R0,RNGBLK               ;SET IN RING BLOCK HEADER
000746' 110167  004252                  MOVB    R1,RNGBLK+2             ;AND THE EXTENDED ADDRESS BITS
000752' 112767  000005  004245          MOVB    #.TELEN,RNGBLK+3        ;NUMBER OF WORDS IN EACH ENTRY
000760' 012700  005546'                 MOV     #RDRB,R0                ;VIRTUAL ADDRESS OF THE RECEIVE RING
000764' 004767  004010                  JSR     PC,CVTPHY               ;CONVERT TO PHYSICAL
000770' 010067  004234                  MOV     R0,RNGBLK+6             ;SET IN RING BLOCK HEADER
000774' 110167  004232                  MOVB    R1,RNGBLK+10            ;AND THE EXTENDED ADDRESS BITS
001000' 112767  000007  004225          MOVB    #.RELEN,RNGBLK+11       ;NUMBER OF WORDS IN EACH ENTRY
001006' 012700  005212'                 MOV     #PCB,R0                 ;GET VIRTUAL PORT CONTROL BASE ADDRESS
001012' 004767  003762                  JSR     PC,CVTPHY               ;CONVERT TO PHYSICAL
001016' 010065  000004                  MOV     R0,.PCSR2(R5)           ;SET THE PORT CONTROL BLOCK BASE ADDRESS
001022' 010165  000006                  MOV     R1,.PCSR3(R5)           ;SET THE EXTENDED ADDRESS BITS
001026'                                 DEMAND  .PCBB,<UNABLE TO SET THE PORT CONTROL BASE ADDRESS>
001026' 004767  001524                  JSR     PC,SETCMD
001032' 000001                          .WORD   .PCBB
001034' 103041                          BCC     73$
001036'                                 MSG     72$
001036' 104401  000000' 001062'         MSG$,BEGIN,72$          ;ASCII MESSAGE CALL
001044'                                 HRDER   ERRPRT <>                                       ;CC
001044' 004767  003770                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
           001                          .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;
                                        .IFF
001050' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;
           000                          .ENDC
001056' 000167  000742                  JMP     DROP
001062'    045                  72$:    .ASCII  '%'
001063'    125  060556  066142          .ASCIZ  /UNABLE TO SET THE PORT CONTROL BASE ADDRESS/
001070' 020145  067564  071440
001076' 072145  072040  062550
001104' 070040  071157  020164
001112' 067543  072156  067562
001120' 020154  060542  062563
001126' 060440  062144  062562
001134' 071563     000
           001140'                      .EVEN
001140'                         73$:

001140' 012767  000015  004044          MOV     #15,PCB                 ;WRITE MODE REGISTER
001146' 005067  004042                  CLR     PCB+2                   ;FULL DUPLEX, EXTERNAL LOOPBACK
001152' 032767  000004  176636          BIT     #.INTERN,SR1            ;INTERNAL LOOPBACK
001160' 001403                          BEQ     5$                      ;NO - KEEP DEFAULT EXTERNAL
001162' 052767  000004  004024          BIS     #.LOOP,PCB+2            ;SET THE INTERNAL LOOPBACK BIT
001170'                         5$:     DEMAND  .CMD,<UNABLE TO WRITE THE MODE REGISTER>
001170' 004767  001362                  JSR     PC,SETCMD
001174' 000002                          .WORD   .CMD
001176' 103034                          BCC     75$
001200'                                 MSG     74$
001200' 104401  000000' 001224'         MSG$,BEGIN,74$          ;ASCII MESSAGE CALL
001206'                                 HRDER   ERRPRT <>                                       ;CC
001206' 004767  003626                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
           001                          .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;
                                        .IFF
001212' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;
```



```
           000                          .ENDC
001220' 000167  000600                  JMP     DROP
001224'    045                  74$:    .ASCII  '%'
001225'    125  060556  066142          .ASCIZ  /UNABLE TO WRITE THE MODE REGISTER/
001232' 020145  067564  073440
001240' 064562  062564  072040
001246' 062550  066440  062157
001254' 020145  062562  064547
001262' 072163  071145     000
           001270'                      .EVEN
001270'                         75$:

001270' 012767  000004  003714          MOV     #4,PCB                  ;READ PHYSICAL ADDRESS
001276'                                 DEMAND  .CMD,<UNABLE TO READ THE PHYSICAL ADDRESS ROM>
001276' 004767  001254                  JSR     PC,SETCMD
001302' 000002                          .WORD   .CMD
001304' 103037                          BCC     77$
001306'                                 MSG     76$
001306' 104401  000000' 001332'         MSG$,BEGIN,76$          ;ASCII MESSAGE CALL
001314'                                 HRDER   ERRPRT <>                                       ;CC
001314' 004767  003520                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
           001                          .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;
                                        .IFF
001320' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;
           000                          .ENDC
001326' 000167  000472                  JMP     DROP
001332'    045                  76$:    .ASCII  '%'
001333'    125  060556  066142          .ASCIZ  /UNABLE TO READ THE PHYSICAL ADDRESS ROM/
001340' 020145  067564  071040
001346' 060545  020144  064164
001354' 020145  064160  071571
001362' 061551  066141  060440
001370' 062144  062562  071563
001376' 051040  046517     000
           001404'                      .EVEN
001404'                         77$:
001404' 012701  006052'                 MOV     #HEADER,R1              ;WHERE TO PUT IT
001410' 012702  000002                  MOV     #2,R2                   ;PACK IT TWICE
001414' 012700  005214'         6$:     MOV     #PCB+2,R0               ;START OF RECEIVED DATA
001420' 012021                          MOV     (R0)+,(R1)+             ;MOVE 6 BYTES
001422' 012021                          MOV     (R0)+,(R1)+             ;BYTES 3 & 4
001424' 012021                          MOV     (R0)+,(R1)+             ;BYTES 5 & 6
001426' 005302                          DEC     R2                      ;DECREMENT THE COUNTER
001430' 001371                          BNE     6$                      ;AND LOOP TILL DONE
001432' 012721  002540                  MOV     #2540,(R1)+             ;SET RECORD TYPE FIELD TO DIAG
001436' 005021                          CLR     (R1)+                   ;CLEAR THE MSG LENGTH
001440' 012721  003001                  MOV     #3001,(R1)+             ;SET SUBTYPE FIELD TO DEC/X
001444' 012701  006052'                 MOV     #HEADER,R1              ;START OF HEADER
001450' 012702  000022                  MOV     #18.,R2                 ;SIZE OF HEADER
001454' 012703  177777                  MOV     #-1,R3                  ;INITIAL CRC
001460' 010304                          MOV     R3,R4                   ;SAME IN R4
001462' 004767  003206                  JSR     PC,BLKCRC               ;DO THE CALCULATIONS
001466' 010367  004354                  MOV     R3,HDRCRC               ;SAVE THE CRC - HIGH WORD
001472' 010467  004352                  MOV     R4,HDRCRC+2             ;SAVE THE CRC - LOW WORD
```

```
CXUACA.MAC    13-JAN-83 16:32

001476' 004767  002552                  JSR     PC,TSETUP               ;SETUP THE TRANSMIT RING STRUCTURE
001502' 004767  003006                  JSR     PC,RSETUP               ;SETUP THE RECEIVER RING STRUCTURE

001506' 012700  005222'                 MOV     #RNGBLK,R0              ;GET THE VIRTUAL ADDRESS
001512' 004767  003262                  JSR     PC,CVTPHY               ;CONVERT TO PHYSICAL
001516' 010067  003472                  MOV     R0,PCB+2                ;SET THE RING BUFFER STRUCTURE
001522' 010167  003470                  MOV     R1,PCB+4                ; AND THE EXTENDED ADDRESS BITS
001526' 012767  000011  003456          MOV     #11,PCB                 ;WRITE RING FUNCTION CODE
001534'                                 DEMAND  .CMD,<UNABLE TO SET THE RING BUFFER STRUCTURE>
001534' 004767  001016                  JSR     PC,SETCMD
001540' 000002                          .WORD   .CMD
001542' 103037                          BCC     79$
001544'                                 MSG     78$
001544' 104401  000000' 001570'         MSG$,BEGIN,78$          ;ASCII MESSAGE CALL
001552'                                 HRDER   ERRPRT <>                                               ;CC
001552' 004767  003262                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
           001                          .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;
                                        .IFF
001556' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;
           000                          .ENDC
001564' 000167  000234                  JMP     DROP
001570'    045                  78$:    .ASCII  '%'
001571'    125  060556  066142          .ASCIZ  /UNABLE TO SET THE RING BUFFER STRUCTURE/
001576' 020145  067564  071440
001604' 072145  072040  062550
001612' 071040  067151  020147
001620' 072542  063146  071145
001626' 071440  071164  061565
001634' 072564  062562     000
           001642'
001642'                                 .EVEN
001642'                         79$:

001642'                                 DEMAND  .START,<UNABLE TO START THE DEUNA>
001642' 004767  000710                  JSR     PC,SETCMD
001646' 000004                          .WORD   .START
001650' 103030                          BCC     81$
001652'                                 MSG     80$
001652' 104401  000000' 001676'         MSG$,BEGIN,80$          ;ASCII MESSAGE CALL
001660'                                 HRDER   ERRPRT <>                                               ;CC
001660' 004767  003154                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
           001                          .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;
                                        .IFF
001664' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;
           000                          .ENDC
001672' 000167  000126                  JMP     DROP
001676'    045                  80$:    .ASCII  '%'
001677'    125  060556  066142          .ASCIZ  /UNABLE TO START THE DEUNA/
001704' 020145  067564  071440
001712' 060564  072162  072040
001720' 062550  042040  052505
001726' 040516     000
           001732'                      .EVEN
001732'                         81$:
001732'                                 WAIT    .RUN,<UNABLE TO ENTER RUN STATE>
```



```
CXUACA.MAC    13-JAN-83 16:32

001732' 004767  000720                  JSR     PC,#STATE
001736' 000003                          .WORD   .RUN
001740' 103030                          BCC     83$
001742'                                 MSG     82$
001742' 104401  000000' 001766'         MSG$,BEGIN,82$          ;ASCII MESSAGE CALL
001750'                                 HRDER   ERRPRT <>                                               ;CC
001750' 004767  003064                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
           001                          .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;
                                        .IFF
001754' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;
           000                          .ENDC
001762' 000167  000036                  JMP     DROP
001766'    045                  82$:    .ASCII  '%'
001767'    125  060556  066142          .ASCIZ  /UNABLE TO ENTER RUN STATE/
001774' 020145  067564  062440
002002' 072156  071145  071040
002010' 067165  071440  060564
002016' 062564     000
           002022'                      .EVEN
002022'                         83$:
002022' 000404                          BR      LOOP                    ;DO SOME TRANSMIT/RECEPTIONS

002024' 005065  000000          DROP:   CLR     .PCSR0(R5)              ;DISABLE INTERRUPTS
002030'                                 END                             ;THATS ALL FOLKS
002030' 104410  000000'                 END$,BEGIN              ;
```

```
                                ;--
                                ;
                                ;       EXECUTE A SERIES OF TRANSMITS AND RECEPTIONS
                                ;
                                ;
                                ;--
002034' 016702  003174          LOOP:   MOV     RPCNT,R2                ;GET NUMBER OF RECEIVER PACKETS
002040' 012703  005236'                 MOV     #TDRB,R3                ;ADDRESS OF TRANSMITTER DESC BLOCK
002044' 012704  005546'                 MOV     #RDRB,R4                ;GET ADDRESS OF RECEIVER DESC BLOCK
002050' 042763  176774  000004  1$:     BIC     #^C1003,.STA1(R3)       ;CLEAR ALL BUT THE EXTENDED ADDRESS BITS
002056' 052763  100000  000004          BIS     #.OWN,.STAT(R3)         ;SET OWNERSHIP TO DEUNA
002064' 062703  000012                  ADD     #.TELEN*2,R3            ;ADVANCE TO NEXT ENTRY
002070' 042763  177374  000004          BIC     #^C403,.STAT(R3)        ;CLEAR ALL BUT THE EXTENDED ADDRESS BITS
002076' 052763  100000  000004          BIS     #.OWN,.STAT(R3)         ;SET OWNERSHIP TO DEUNA
002104' 062703  000012                  ADD     #.TELEN*2,R3            ;ADVANCE TO NEXT ENTRY
002110' 042764  177774  000004          BIC     #^C3,.STAT(R4)          ;CLEAR ALL BUT THE EXTENDED ADDRESS BITS
002116' 052764  100000  000004          BIS     #.OWN,.STAT(R4)         ;ASSIGN OWNERSHIP TO THE DEUNA
002124' 005064  000006                  CLR     .MLEN(R4)               ;CLEAR THE RETURNED BYTE COUNT
002130' 062704  000016                  ADD     #.RELEN*2,R4            ;ADVANCE TO NEXT ENTRY
002134' 005302                          DEC     R2                      ;DECREMENT PACKET COUNTER
002136' 001344                          BNE     1$                      ;DO ENTIRE LIST
002140' 005067  003700                  CLR     FLAGS                   ;CLEAR INTERRUPT (CSR) FLAG WORD
002144'                                 DEMAND  .PDMD,<UNABLE TO ISSUE A POLL DEMAND>
002144' 004767  000406                  JSR     PC,SETCMD
002150' 000010                          .WORD   .PDMD
002152' 103032                          BCC     65$
002154'                                 MSG     64$
002154' 104401  000000' 002200'         MSG$,BEGIN,64$          ;ASCII MESSAGE CALL
002162'                                 HRDER   ERRPRT <>                                       ;CC
002162' 004767  002652                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;
                                        .IFF
002166' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;
            000                         .ENDC
002174' 000167  177624                  JMP     DROP
002200'    045                  64$:    .ASCII  '%'
002201'    125  060556  066142          .ASCIZ  /UNABLE TO ISSUE A POLL DEMAND/
002206' 020145  067564  064440
002214' 071563  062565  060440
002222' 070040  066157  020154
002230' 062544  060555  062156
002236'    000
            002240'                     .EVEN
002240'                         65$:

002240' 012702  007640                  MOV     #4000.,R2               ;TIMEOUT LOOP COUNT
002244' 016700  003574          2$:     MOV     FLAGS,R0                ;GET CURRENT FLAGS WORD
002250' 032700  142400                  BIT     #.ERRI,R0               ;LOOK FOR ERROR CONDITION
002254' 001040                          BNE     3$
002256' 042700  147777                  BIC     #^C<.TXI+.RXI>,R0       ;KEEP ONLY COMPLETION BITS
002262' 022700  030000                  CMP     #.TXI+.RXI,R0           ;LOOK FOR BOTH COMPLETION
002266' 001433                          BEQ     3$                      ;FINISHED WHEN BOTH SET
002270'                                 BREAK                           ;TEMP RETURN TO DECX
002270' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
002274' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
```



```
002300' 005302                          DEC     R2                      ;DECREMENT THE LOOP COUNT
002302' 001360                          BNE     2$                      ;LOOP WHILE NOT ZERO
002304'                                 PRINT   <RECEIVER TIMEOUT>
002304'                                 MSG     66$
002304' 104401  000000' 002314'         MSG$,BEGIN,66$          ;ASCII MESSAGE CALL
002312' 000411                          BR      67$
002314'    045                  66$:    .ASCII  '%'
002315'    122  061545  064545          .ASCIZ  /RECEIVER TIMEOUT/
002322' 062566  020162  064564
002330' 062555  072557  000164

                                        .EVEN
002336'                         67$:
002336'                                 HRDER   ERRPRT  <RECEIVER TIMEOUT>                        ;CC
002336' 004767  002476                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;RECEIVER TIMEOUT
                                        .IFF
002342' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;RECEIVER TIMEOUT
            000                         .ENDC
002350' 005065  000000                  CLR     .PCSR0(R5)              ;DISABLE INTERRUPTS             ;CC
002354' 000425                          BR      10$                     ;                               ;CC
002356' 005065  000000          3$:     CLR     .PCSR0(R5)              ;DISABLE INTERRUPTS
002362' 004767  000346                  JSR     PC,CSRCHK               ;CHECK THE CSR (FLAG) FOR ERRORS
002366' 103427                          BCS     11$                     ;DROP ON REAL TOUGH ONE
002370' 004767  001156          8$:     JSR     PC,CHKSTA               ;CHECK TRANSMIT/RECEIVE STATUS
002374' 103424                          BCS     11$                     ;AND EXIT ON ERROR
002376' 004767  001412                  JSR     PC,CHKBCT               ;CHECK THE BYTE COUNT
002402' 103421                          BCS     11$
002404' 004767  001512                  JSR     PC,CHKCRC               ;CHECK THE RETURNED CRC'S
002410' 103404                          BCS     9$
002412' 032767  000001  175376          BIT     #.DCHECK,SR1            ;PERFORM DATA COMPARSIONS
002420' 001403                          BEQ     10$
002422' 004767  000740          9$:     JSR     PC,CHKBUF               ;CHECK THE DATA BUFFERS
002426' 103407                          BCS     11$
002430'                         10$:    ENDIT                           ;END OF ITTERATION
002430' 104413  000000'                 ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
                                                                ;MONITOR SHALL TEST END OF PASS
002434' 012765  000100  000000          MOV     #.INTE,.PCSR0(R5)       ;ENABLE INTERRUPTS
002442' 000167  177366                  JMP     LOOP                    ;AND GO TO START OF LOOP
002446'                         11$:    ENDIT                           ;END OF ITERATION               ;CC
002446' 104413  000000'                 ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
                                                                ;MONITOR SHALL TEST END OF PASS
002452' 000167  175616                  JMP     RESTRT
```

```
                                ;--
                                ;
                                ;       SERVICE THE DEUNA INTERRUPTS
                                ;
                                ;       ON RETURN
                                ;               FLAGS = COPY OF PCSR0
                                ;
                                ;       NOTE:   THE RECEIVE FLAG IS SET ONLY WHEN "ALL" PACKETS
                                ;               HAVE BEEN RECEIVED.
                                ;
                                ;--
002456' 010046                  INTSRV: MOV     R0,-(SP)                ;SAVE R0
002460' 010146                          MOV     R1,-(SP)                ;SAVE R1
002462' 016701  175320                  MOV     ADDR,R1                 ;GET ADDRESS OF DEUNA
002466' 056167  000000  003350          BIS     .PCSR0(R1),FLAGS        ;GET CONTENTS OF PCSR0
002474' 116761  003345  000001          MOVB    FLAGS+1,.PCSR0+1(R1)    ;AND WRITE UPPER BITS TO CLEAR
002502' 032767  020000  003334          BIT     #.RXI,FLAGS             ;TEST FOR RECEIVER INTERRUPT
002510' 001417                          BEQ     3$                      ;NOT THIS ONE
002512' 012701  005546'                 MOV     #RQRB,R1                ;RECEIVER BUFFER RING
002516' 016700  002512                  MOV     RPCNT,R0                ;NUMBER OF PACKETS
002522' 005761  000004          1$:     TST     .STAT(R1)               ;TEST THE OWNERSHIP BIT
002526' 100405                          BMI     2$                      ;AND EXIT IF NEGATIVE
002530' 062701  000016                  ADD     #.RELEN*2,R1            ;ADVANCE TO NEXT ENTRY
002534' 005300                          DEC     R0                      ;DECREMENT PACKET COUNTER
002536' 001371                          BNE     1$                      ;AND LOOP TILL ALL TESTED
002540' 000403                          BR      3$
002542' 042767  020000  003274  2$:     BIC     #.RXI,FLAGS             ;CLEAR THE FLAG BIT
002550' 012601                  3$:     MOV     (SP)+,R1                ;RESTORE R1
002552' 012600                          MOV     (SP)+,R0                ;RESTORE R0
002554' 000002                          RTI                             ;OR JUST RETURN
```



```
                                ;--
                                ;
                                ;       SET THE COMMAND AND WAIT FOR COMPLETION
                                ;
                                ;       ON RETURN
                                ;               C = 0 - SUCCESS
                                ;               C = 1 - FAILURE - COMMAND TIMEOUT
                                ;
                                ;--
002556' 004767  002362          SETCMD: JSR     PC,SAVREG               ;SAVE ALL REGISTERS
002562' 017600  000016                  MOV     @16(SP),R0              ;GET THE COMMAND WORD
002566' 062766  000002  000016          ADD     #2,16(SP)               ;ADJUST THE RETURN ADDRESS
002574' 052700  000100                  BIS     #.INTE,R0               ;FORCE THE INTERRUPT ENABLE
002600' 010065  000000                  MOV     R0,.PCSR0(R5)           ;SET THE COMMAND FUNCTION CODE
002604' 012700  000144                  MOV     #100.,R0                ;OUTER TIMEOUT LOOP COUNT
002610' 012701  000062          1$:     MOV     #50.,R1                 ;INNER LOOP COUNT
002614' 032767  004000  003222  2$:     BIT     #.DNI,FLAGS             ;TEST FOR COMMAND COMPLETION
002622' 001011                          BNE     3$                      ;EXIT WHEN SET
002624' 005301                          DEC     R1                      ;TEST A FEW TIMES BEFORE BREAKING
002626' 001372                          BNE     2$                      ;QUICK LOOP ENTRY
002630'                                 BREAK                           ;TEMP EXIT TO MONITOR
002630' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
002634' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
002640' 005300                          DEC     R0                      ;DECREMENT TIMEOUT LOOP COUNT
002642' 001362                          BNE     1$                      ;LONG LOOP ENTRY
002644' 000261                          SEC                             ;SET THE ERROR INDICATOR
002646' 042767  004000  003170  3$:     BIC     #.DNI,FLAGS             ;CLEAR THE DONE BIT
002654' 000207                          RTS     PC                      ;AND RETURN

                                ;--
                                ;
                                ;       WAIT FOR DEUNA TO ENTER SPECIFIED STATE
                                ;
                                ;       ON RETURN
                                ;               C = 0 - SUCCESS
                                ;               C = 1 - FAILURE - COMMAND TIMEOUT
                                ;
                                ;--
002656' 004767  002262          WSTATE: JSR     PC,SAVREG               ;SAVE ALL REGISTERS
002662' 017602  000016                  MOV     @16(SP),R2              ;GET EXPECTED STATE
002666' 062766  000002  000016          ADD     #2,16(SP)               ;ADJUST THE RETURN ADDRESS
002674' 012701  000100                  MOV     #100,R1                 ;TIMEOUT VALUE
002700' 016500  000002          1$:     MOV     .PCSR1(R5),R0           ;GET STATUS WORD
002704' 042700  177770                  BIC     #^C.STATE,R0            ;KEEP ONLY THE STATE FIELD
002710' 020002                          CMP     R0,R2                   ;COMPARE WITH EXPECTED
002712' 001407                          BEQ     2$                      ;EXIT IF THE SAME
002714'                                 BREAK                           ;TEMP RETURN TO DECX
002714' 104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
002720' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
002724' 005301                          DEC     R1                      ;DECREMENT TIMEOUT COUNTER
002726' 001364                          BNE     1$                      ;AND LOOP BACK IF NOT ZERO
002730' 000261                          SEC                             ;SET THE ERROR FLAG
002732' 000207                  2$:     RTS     PC                      ;AND RETURN
```

```
                                ;--
                                ;
                                ;       CHECK THE CSR (FLAGS) FOR ERRORS AND
                                ;       NORMAL COMPLETION CODES
                                ;
                                ;--

002734' 004767  002204          CSRCHK: JSR     PC,SAVREG               ;SAVE ALL REGISTERS
002740' 016700  003100                  MOV     FLAGS,R0                ;GET INTERRUPT SUMMARY
002744' 032700  000400          1$:     BIT     #.FATI,R0               ;TEST FOR FATAL ERROR
002750' 001425                          BEQ     2$
002752'                                 PRINT   <FATAL ERROR INTERRUPT>
002752'                                 MSG     64$
002752' 104401  000000' 002762'         MSG$,BEGIN,64$          ;ASCII MESSAGE CALL
002760' 000414                          BR      65$
002762'     045                 64$:    .ASCII  '%'
002763'     106 072141  066141          .ASCIZ  /FATAL ERROR INTERRUPT/
002770' 042440  071162  071157
002776' 044440  072156  071145
003004' 072562  072160     000
        003012'
                                        .EVEN
003012'                         65$:
003012'                                 HRDER   ERRPRT  <FATAL ERROR INTERRUPT>
003012' 004767  002022                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;FATAL ERROR INTERRUPT
                                        .IFF
003016' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;FATAL ERROR INTERRUPT
            000                         .ENDC
003024' 032700  002000          2$:     BIT     #.RCBI,R0               ;RECEIVE BUFFER UNAVAILABLE
003030' 001435                          BEQ     3$
003032'                                 PRINT   <RECEIVE BUFFER NOT AVAILABLE INTERRUPT>
003032'                                 MSG     66$
003032' 104401  000000' 003042'         MSG$,BEGIN,66$          ;ASCII MESSAGE CALL
003040' 000424                          BR      67$
003042'     045                 66$:    .ASCII  '%'
003043'     122 061545  064545          .ASCIZ  /RECEIVE BUFFER NOT AVAILABLE INTERRUPT/
003050' 062566  041040  063165
003056' 062546  020162  067516
003064' 020164  073101  064541
003072' 060554  066142  020145
003100' 067111  062564  071162
003106' 070165  000164
                                        .EVEN
003112'                         67$:
003112'                                 HRDER   ERRPRT  <RECEIVE BUFFER NOT AVAILABLE INTERRUPT>
003112' 004767  001722                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;RECEIVE BUFFER NOT AVAILABLE INTERRUPT
                                        .IFF
003116' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;RECEIVE BUFFER NOT AVAILABLE INTERRUPT
            000                         .ENDC
003124' 032700  040000          3$:     BIT     #.PCEI,R0               ;PORT COMMAND ERROR
003130' 001430                          BEQ     4$
003132'                                 PRINT   <PORT COMMAND ERROR INTERRUPT>
003132'                                 MSG     68$
```



```
003132' 104401  000000' 003142'         MSG$,BEGIN,68$          ;ASCII MESSAGE CALL
003140' 000417                          BR      69$
003142'     045                 68$:    .ASCII  '%'
003143'     120 071157  020164          .ASCIZ  /PORT COMMAND ERROR INTERRUPT/
003150' 067503  066555  067141
003156' 020144  071105  067562
003164' 020162  067111  062564
003172' 071162  070165  000164
                                        .EVEN
003200'                         69$:
003200'                                 HRDER   ERRPRT  <PORT COMMAND ERROR INTERRUPT>
003200' 004767  001634                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;PORT COMMAND ERROR INTERRUPT
                                        .IFF
003204' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;PORT COMMAND ERROR INTERRUPT
            000                         .ENDC
003212' 032700  100000          4$:     BIT     #.SERI,R0               ;STATUS ERROR INTERRUPT
003216' 001425                          BEQ     5$
003220'                                 PRINT   <STATUS ERROR INTERRUPT>
003220'                                 MSG     70$
003220' 104401  000000' 003230'         MSG$,BEGIN,70$          ;ASCII MESSAGE CALL
003226' 000414                          BR      71$
003230'     045                 70$:    .ASCII  '%'
003231'     123 060564  072564          .ASCIZ  /STATUS ERROR INTERRUPT/
003236' 020163  071105  067562
003244' 020162  067111  062564
003252' 071162  070165  000164
                                        .EVEN
003260'                         71$:
003260'                                 HRDER   ERRPRT  <STATUS ERROR INTERRUPT>
003260' 004767  001554                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;STATUS ERROR INTERRUPT
                                        .IFF
003264' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;STATUS ERROR INTERRUPT
            000                         .ENDC
003272' 032767  010000  002544  5$:     BIT     #.TXI,FLAGS             ;TEST THAT THE TRANSMITTER FINISHED
003300' 001026                          BNE     6$
003302'                                 PRINT   <NO TRANSMIT INTERRUPT>
003302'                                 MSG     72$
003302' 104401  000000' 003312'         MSG$,BEGIN,72$          ;ASCII MESSAGE CALL
003310' 000414                          BR      73$
003312'     045                 72$:    .ASCII  '%'
003313'     116 020157  071124          .ASCIZ  /NO TRANSMIT INTERRUPT/
003320' 067141  066563  072151
003326' 044440  072156  071145
003334' 072562  072160     000
        003342'
                                        .EVEN
003342'                         73$:
003342'                                 HRDER   ERRPRT  <NO TRANSMIT INTERRUPT>
003342' 004767  001472                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        HRDER$,BEGIN,NULL       ;NO TRANSMIT INTERRUPT
                                        .IFF
003346' 104405  000000' 005762'         HRDER$,BEGIN,ERRPRT     ;NO TRANSMIT INTERRUPT
```

```
                000                     .ENDC
003354' 000402                  BR      7$
003356' 000241                  6$:     CLC                             ;CLEAR THE ERROR INDICATOR
003360' 000401                  BR      8$                              ;AND EXIT
003362' 000261                  7$:     SEC                             ;SET THE ERROR INDICATOR
003364' 000207                  8$:     RTS     PC                      ;AND RETURN
```



```
                                ;--
                                ;
                                ;       CHECK THE RECEIVED PACKETS DATA
                                ;
                                ;       ON RETURN
                                ;               C = 0 - SUCCESS
                                ;               C = 1 - FAILURE - COMMAND TIMEOUT
                                ;
                                ;--
003366' 004767  001552          CHKBUF: JSR     PC,SAVREG               ;SAVE ALL REGISTERS
003372' 012703  005546'                 MOV     #RDRH,R3                ;START OF RECEIVER RING DECS.
003376' 012704  005250'                 MOV     #TDRH+<.TELEN*2>,R4     ;START OF TRANSMITTER RING DESC.
003402' 016705  001626                  MOV     RPCNT,R5                ;RECEIVE PACKET COUNT
003406' 016400  000010          1$:     MOV     .VA(R4),R0              ;TRANSMIT BUFFER
003412' 016301  000010                  MOV     .VA(R3),R1              ;RECEIVE BUFFER
003416' 062701  000022                  ADD     #18.,R1                 ;BYPASS THE HEADER
003422' 016302  000006                  MOV     .MLEN(R3),R2            ;EXPECTED BYTE COUNT
003426' 162702  000026                  SUB     #22.,R2                 ;BACK OFF FOR CRC AND HEADER
003432' 006202                          ASR     R2                      ;CONVERT BYTE COUNT TO WORD COUNT
003434' 005046                          CLR     -(SP)                   ;CLEAR THE STATUS WORD
003436' 005067  174446                  CLR     AWAS                    ;CLEAR THE ACTUAL WORD
003442' 005067  174440                  CLR     ASB                     ;CLEAR THE SHOULD BE WORD
003446' 021011                  2$:     CMP     (R0),(R1)               ;COMPARE THE BYTES
003450' 001417                          BEQ     3$                      ;ADVANCE TO NEXT IF SAME
003452' 010067  174424                  MOV     R0,SBADR                ;SET ADDRESS OF SHOULD BE
003456' 011067  174424                  MOV     (R0),ASB                ;SET DATA OF SHOULD BE
003462' 010167  174416                  MOV     R1,WASADR               ;SET ADDRESS OF ACTUAL
003466' 011167  174416                  MOV     (R1),AWAS               ;SET DATA OF ACTUAL
003472'                                 DATERR                          ;REPORT THE ERROR
003472' 104404  000000'                 DATER$,BEGIN            ;DATA ERROR!!!
003476' 005216                          INC     (SP)                    ;SET THE ERROR FLAG
003500' 032767  000002  174310          BIT     #.CHKALL,SR1            ;TEST ENTIRE BUFFER
003506' 001406                          BEQ     4$                      ;EXIT IF CLEAR
003510' 062700  000002          3$:     ADD     #2,R0                   ;ADVANCE THE SHOULD BE POINTER
003514' 062701  000002                  ADD     #2,R1                   ;ADVANCE THE ACTUAL POINTER
003520' 005302                          DEC     R2                      ;DECREMENT THE BYTE COUNT
003522' 001351                          BNE     2$                      ;AND LOOP TILL DONE
003524' 005726                  4$:     TST     (SP)+                   ;TEST THE ERROR FLAG
003526' 001402                          BEQ     5$                      ;CONTINUE PROCESSING IF ZERO
003530' 000261                          SEC                             ;SET THE ERROR FLAG
003532' 000406                          BR      6$                      ;AND EXIT
003534' 062704  000024          5$:     ADD     #.TELEN*4,R4            ;ADVANCE TRANSMIT POINTER
003540' 062703  000016                  ADD     #.RELEN*2,R3            ;ADVANCE RECEIVER POINTER
003544' 005305                          DEC     R5                      ;DECREMENT THE PACKET COUNTER
003546' 001317                          BNE     1$                      ;AND LOOP TILL DONE
003550' 000207                  6$:     RTS     PC                      ;RETURN
```

```
                                ;--
                                ;
                                ;       CHECK TRANSMIT AND RECEIVE STATUS REGISTERS
                                ;
                                ;       ON RETURN
                                ;               C = 0 - SUCCESS
                                ;               C = 1 - FAILURE - COMMAND TIMEOUT
                                ;
                                ;--
003552' 004767  001366          CHKSTA: JSR     PC,SAVREG               ;SAVE ALL REGISTERS
003556' 012701  005250'                 MOV     #TDRB+<.TELEN*2>,R1     ;FIRST TRANSMIT POINTER
003562' 012702  005546'                 MOV     #RDRB,R2                ;FIRST RECEIVE POINTER
003566' 016703  001442                  MOV     RPCNT,R3                ;RECEIVE PACKET COUNT
003572' 016100  000004          1$:     MOV     .STAT(R1),R0            ;GET TRANSMIT STATUS
003576' 042700  016377                  BIC     #16377,R0               ;CLEAR THE LOWER BYTE
003602' 020027  020400                  CMP     R0,#.ENP+.MATCH         ;SHOULD BE JUST END OF PACKET
003606' 001432                          BEQ     2$                      ;CONTINUE TESTING IF EXACT
003610'                                 PRINT   <TRANSMIT PACKET STATUS ERROR>
003610'                                 MSG     64$
003610' 104401  000000' 003620'         MSGS,BEGIN,64$          ;ASCII MESSAGE CALL
003616' 000417                          BR      65$
003620'    045                  64$:    .ASCII  '%'
003621'    124  060562  071556          .ASCIZ  /TRANSMIT PACKET STATUS ERROR/
003626' 064555  020164  060520
003634' 065543  072145  051440
003642' 060564  072564  020163
003650' 071105  067562  000162
                                        .EVEN
003656'                         65$:
003656'                                 SOFER   ERRPRT <TRANSMIT PACKET STATUS ERROR>
003656' 004767  001156                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        SOFERS,BEGIN,NULL       ;TRANSMIT PACKET STATUS ERROR
                                        .IFF
003662' 104406  000000' 005762'         SOFERS,BEGIN,ERRPRT     ;TRANSMIT PACKET STATUS ERROR
            000                         .ENDC
003670' 000261                          SEC                             ;SET THE ERROR FLAG
003672' 000447                          BR      4$                      ;AND EXIT ROUTINE
003674' 016200  000004          2$:     MOV     .STAT(R2),R0            ;GET RECEIVE STATUS
003700' 042700  000377                  BIC     #377,R0                 ;CLEAR LOWER BYTE
003704' 020027  045400                  CMP     R0,#.ERRS+.CRC+.STP+.ENP;EXPECT A CRC ERROR
003710' 001432                          BEQ     3$                      ;CONTINUE PROCESSING IF EXACT MATCH
003712'                                 PRINT   <RECEIVE PACKET STATUS ERROR>
003712'                                 MSG     66$
003712' 104401  000000' 003722'         MSGS,BEGIN,66$          ;ASCII MESSAGE CALL
003720' 000417                          BR      67$
003722'    045                  66$:    .ASCII  '%'
003723'    122  061545  064545          .ASCIZ  /RECEIVE PACKET STATUS ERROR/
003730' 062566  050040  061541
003736' 062553  020164  072123
003744' 072141  071565  042440
003752' 071162  071157     000
        003760'                         .EVEN
003760'                         67$:
003760'                                 SOFER   ERRPRT <RECEIVER PACKET STATUS ERROR>
003760' 004767  001054                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
```



```
            001                         .IF B <ERRPRT>
                                        SOFERS,BEGIN,NULL       ;RECEIVER PACKET STATUS ERROR
                                        .IFF
003764' 104406  000000' 005762'         SOFERS,BEGIN,ERRPRT     ;RECEIVER PACKET STATUS ERROR
            000                         .ENDC
003772' 000261                          SEC                             ;SET THE ERROR FLAG
003774' 000406                          BR      4$                      ;AND EXIT ROUTINE
003776' 062701  000024          3$:     ADD     #.TELEN*4,R1            ;ADVANCE THE TRANSMIT POINTER
004002' 062702  000016                  ADD     #.RELEN*2,R2            ;ADVANCE THE RECEIVE POINTER
004006' 005303                          DEC     R3                      ;DECREMENT THE PACKET COUNT
004010' 001270                          BNE     1$                      ;LOOP TILL ALL DONE
004012' 000207                  4$:     RTS     PC                      ;AND RETURN
```

```
                                ;--
                                ;
                                ;       CHECK THE BYTE COUNTS
                                ;
                                ;       ON RETURN
                                ;               C = 0 - SUCCESS
                                ;               C = 1 - FAILURE - COMMAND TIMEOUT
                                ;
                                ;--
004014' 004767  001124          CHKBCT: JSR     PC,SAVREG               ;SAVE ALL REGISTERS
004020' 012701  005546'                 MOV     #RDRB,R1                ;START OF RECEIVER RING DECS.
004024' 016700  001204                  MOV     RPCNT,R0                ;PACKET COUNTER
004030' 026161  000000  000006  1$:     CMP     .SLEN(R1),.MLEN(R1)     ;COMPARE EXPECTED BYTE COUNT WITH RECEIVED
004036' 001424                          BEQ     2$                      ;OK IF THE SAME
004040'                                 PRINT   <BYTE COUNT ERROR>      ;DISPLAY THE MESSAGE
004040'                                 MSG     64$
004040' 104401  000000' 004050'         MSG$,BEGIN,64$          ;ASCII MESSAGE CALL
004046' 000411                          BR      65$
004050'     045                 64$:    .ASCII  '%'
004051'     102 072171  020145          .ASCIZ  /BYTE COUNT ERROR/
004056' 067543  067165  020164
004064' 071145  067562  000162
                                        .EVEN
004072'                         65$:
004072'                                 SOFER   ERRPRT <BYTE COUNT ERROR>
004072' 004767  000742                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        SOFER$,BEGIN,NULL       ;BYTE COUNT ERROR
                                        .IFF
004076' 104406  000000' 005762'         SOFER$,BEGIN,ERRPRT     ;BYTE COUNT ERROR
            000                         .ENDC
004104' 000261                          SEC                             ;SET THE ERROR FLAG
004106' 000404                          BR      3$                      ;AND TAKE THE ERROR EXIT
004110' 062701  000016          2$:     ADD     #.RELEN*2,R1            ;ADVANCE RECEIVER POINTER
004114' 005300                          DEC     R0                      ;DECREMENT THE PACKET COUNTER
004116' 001344                          BNE     1$                      ;LOOP TILL ALL TESTED
004120' 000207                  3$:     RTS     PC                      ;RETURN
```



```
                                ;--
                                ;
                                ;       CHECK THE RECEIVED PACKET CRC PATTERNS
                                ;
                                ;       ON RETURN
                                ;               C = 0 - SUCCESS
                                ;               C = 1 - FAILURE - COMMAND TIMEOUT
                                ;
                                ;--
004122' 004767  001016          CHKCRC: JSR     PC,SAVREG               ;SAVE ALL REGISTERS
004126' 012702  005546'                 MOV     #RDRB,R2                ;START OF RECEIVER RING DECS.
004132' 016701  001076                  MOV     RPCNT,R1                ;NUMBER OF RECEIVE PACKETS
004136' 016200  000010          1$:     MOV     .VA(R2),R0              ;GET STARTING ADDRESS OF PACKET
004142' 066200  000006                  ADD     .MLEN(R2),R0            ;ADD RECEIVED PACKET LENGTH
004146' 162700  000004                  SUB     #4,R0                   ;BACK OFF FOR CRC
004152' 026220  000012                  CMP     .CRCL(R2),(R0)+         ;COMPARE THE ACTUAL WITH EXPECTED
004156' 001003                          BNE     2$                      ;AND EXIT IF BAD
004160' 026220  000014                  CMP     .CRCH(R2),(R0)+         ;COMPARE THE ACTUAL WITH EXPECTED
004164' 001426                          BEQ     3$                      ;CONTINUE PROCESSING IF OK
004166'                         2$:     PRINT   <CRC COMPARSION ERROR>
004166'                                 MSG     64$
004166' 104401  000000' 004176'         MSG$,BEGIN,64$          ;ASCII MESSAGE CALL
004174' 000413                          BR      65$
004176'     045                 64$:    .ASCII  '%'
004177'     103 041522  061440          .ASCIZ  /CRC COMPARSION ERROR/
004204' 066557  060560  071562
004212' 067551  020156  071145
004220' 067562  000162
                                        .EVEN
004224'                         65$:
004224'                                 SOFER   ERRPRT <CRC COMPARISON ERROR>
004224' 004767  000610                  JSR     PC,ERRMSG       ;SET UP ERROR PRINTOUT
            001                         .IF B <ERRPRT>
                                        SOFER$,BEGIN,NULL       ;CRC COMPARISON ERROR
                                        .IFF
004230' 104406  000000' 005762'         SOFER$,BEGIN,ERRPRT     ;CRC COMPARISON ERROR
            000                         .ENDC
004236' 000261                          SEC                             ;SET THE ERROR FLAG
004240' 000404                          BR      4$                      ;AND EXIT ROUTINE
004242' 062702  000016          3$:     ADD     #.RELEN*2,R2            ;ADVANCE RECEIVER POINTER
004246' 005301                          DEC     R1                      ;DECREMENT PACKET COUNT
004250' 001332                          BNE     1$                      ;LOOP TILL ALL DONE
004252' 000207                  4$:     RTS     PC                      ;RETURN
```

```
                                ;--
                                ;
                                ;       TRANSMIT RING SET UP ROUTINE
                                ;
                                ;--
004254' 004767  000664          TSETUP: JSR     PC,SAVREG               ;SAVE ALL REGISTERS
004260' 005067  000742                  CLR     TPCNT                   ;NUMBER OF PACKETS GENERATED
004264' 012704  002000                  MOV     #BUFSIZ,R4              ;MAX NUMBER OF RECEIVER BYTES
004270' 012703  005236'                 MOV     #TDRB,R3                ;TRANSMIT DESCRIPTOR RING ENTRY
004274' 012763  000022  000000  1$:     MOV     #18.,.SLEN(R3)          ;SET THE SIZE OF THE HEADER
004302' 012700  006052'                 MOV     #HEADER,R0              ;GET VIRTUAL ADDRESS
004306' 010063  000010                  MOV     R0,.VA(R3)              ;SET THE VIRTUAL ADDRESS
004312' 004767  000462                  JSR     PC,CVTPHY               ;CONVERT TO PHYSICAL
004316' 010063  000002                  MOV     R0,.SBUF(R3)            ;AND THE STARTING ADDRESS
004322' 052701  001000                  BIS     #.STP,R1                ;SET THE START OF PACKET BIT
004326' 010163  000004                  MOV     R1,.STAT(R3)            ;SET THE EXTENDED ADDRESS BITS <16-17>
004332' 005063  000006                  CLR     .TDR(R3)                ;AND CLEAR THE TDR ENTRY
004336' 062703  000012                  ADD     #.TELEN*2,R3            ;POINT TO NEXT PACKET
004342'                                 RAND                            ;CALCULATE A PACKET SIZE
004342' 104417  000000'                 RAND$,BEGIN
004346' 016702  173502                  MOV     RANNUM,R2               ;GET THE RAW NUMBER
004352' 042702  177001                  BIC     #^C776,R2               ;KEEP LESS THAN 512. AND EVEN
004356' 062702  000144                  ADD     #100.,R2                ;FORCE A LOWER LIMIT
004362' 162704  000026                  SUB     #22.,R4                 ;SUBTRACT THE HEADER & CRC
004366' 020204                          CMP     R2,R4                   ;COMPARE WITH WHATS STILL AVAILABLE
004370' 002401                          BLT     2$                      ;AND CONTINUE IF LESS THAN
004372' 010402                          MOV     R4,R2                   ;FORCE TO WHATS AVAILABLE
004374' 010263  000000          2$:     MOV     R2,.SLEN(R3)            ;AND SET IN BYTE COUNT LOCATION
004400'                                 RAND                            ;GET A RANDOM NUMBER
004400' 104417  000000'                 RAND$,BEGIN
004404' 016700  173444                  MOV     RANNUM,R0               ;TO FORM THE STARTING ADDRESS
004410' 042700  177401                  BIC     #^C376,R0               ;KEEP THE NUMBER SMALL AND EVEN
004414' 062700  000252'                 ADD     #START,R0               ;FORM THE STARTING ADDRESS
004420' 010063  000010                  MOV     R0,.VA(R3)              ;SAVE THE VIRTUAL ADDRESS
004424' 004767  000350                  JSR     PC,CVTPHY               ;CONVERT TO A PHYSICAL ADDRESS
004430' 010063  000002                  MOV     R0,.SBUF(R3)            ;AND PACK IN MEMORY
004434' 052701  000400                  BIS     #.ENP,R1                ;SET THE END OF PACKET BIT
004440' 010163  000004                  MOV     R1,.STAT(R3)            ;AND PACK IN RING STRUCTURE
004444' 005063  000006                  CLR     .TDR(R3)                ;CLEAR THE TRANSMIT TDR
004450' 062703  000012                  ADD     #.TELEN*2,R3            ;POINT TO NEXT PACKET
004454' 005267  000546                  INC     TPCNT                   ;COUNT THE PACKETS
004460' 026727  000542  000012          CMP     TPCNT,#.MXPKT           ;COMPARE WITH MAX NUMBER OF ENTRIES
004466' 001404                          BEQ     3$                      ;AND EXIT IF EQUAL
004470' 160204                          SUB     R2,R4                   ;SUBTRACT LEGNTH OF THIS PACKET
004472' 020427  000310                  CMP     R4,#200.                ;ENOUGH TO DO ANOTHER
004476' 003276                          BGT     1$                      ;YUP - KEEP GOING
004500' 016767  000522  000526  3$:     MOV     TPCNT,RPCNT             ;COPY INTO RECEIVER PACKET COUNTER
004506' 006367  000514                  ASL     TPCNT                   ;MULTIPLY BY TWO
004512' 000207                          RTS     PC                      ;RETURN
```



```
                                ;--
                                ;
                                ;       RECEIVE RING SET UP ROUTINE
                                ;
                                ;--
004514' 004767  000424          RSETUP: JSR     PC,SAVREG               ;SAVE ALL REGISTERS
004520' 016703  000510                  MOV     RPCNT,R3                ;SET THE PACKET COUNT
004524' 012704  005250'                 MOV     #TDRB+<.TELEN*2>,R4     ;TRANSMIT DESCRIPTOR RING ENTRY
004530' 012705  005546'                 MOV     #RDRB,R5                ;RECEIVE DESCRIPTOR RING ENTRY
004534' 012700  006116'                 MOV     #RECBUF,R0              ;RECEIVE BUFFER
004540' 010065  000010          1$:     MOV     R0,.VA(R5)              ;SAVE VIRTUAL ADDRESS
004544' 016401  000010                  MOV     .VA(R4),R1              ;GET START OF TRANSMIT DATA BUFFER
004550' 016402  000000                  MOV     .SLEN(R4),R2            ;GET THE TRANSMIT BUFFER SIZE
004554' 010346                          MOV     R3,-(SP)                ;SAVE R3
004556' 010446                          MOV     R4,-(SP)                ;SAVE R4
004560' 016703  001262                  MOV     BLKCRC,R3               ;GET HEADER CRC VALUE - HIGH WORD
004564' 016704  001260                  MOV     BLKCRC+2,R4             ;GET HEADER CRC VALUE - LOW WORD
004570' 004767  000100                  JSR     PC,BLKCRC               ;PERFORM THE CALCULATION
004574' 005103                          COM     R3                      ;INVERT THE DATA PATTERN - HIGH WORD
004576' 005104                          COM     R4                      ;INVERT THE DATA PATTERN - LOW WORD
004600' 010465  000012                  MOV     R4,.CRCL(R5)            ;SAVE EXPECTED CRC - LOW WORD
004604' 010365  000014                  MOV     R3,.CRCH(R5)            ;SAVE EXPECTED CRC - HIGH WORD
004610' 012604                          MOV     (SP)+,R4                ;RESTORE R4
004612' 012603                          MOV     (SP)+,R3                ;RESTORE R3
004614' 062702  000026                  ADD     #22.,R2                 ;ACCOUNT FOR THE EXPECTED CRC & HEADER
004620' 010265  000000                  MOV     R2,.SLEN(R5)            ;AND SET IN RING BUFFER
004624' 016500  000010                  MOV     .VA(R5),R0              ;GET VIRTUAL ADDRESS
004630' 004767  000144                  JSR     PC,CVTPHY               ;CONVERT TO PHYSICAL ADDRESS
004634' 010065  000002                  MOV     R0,.SBUF(R5)            ;SET THE RECEIVE BUFFER ADDRESS
004640' 010165  000004                  MOV     R1,.STAT(R5)            ;SET THE EXTENDED ADDRESS BITS <16-17>
004644' 005065  000006                  CLR     .MLEN(R5)               ;CLEAR THE RECEIVE MESSAGE LENGTH
004650' 016500  000010                  MOV     .VA(R5),R0              ;GET STARTING VIRTUAL ADDRESS
004654' 060200                          ADD     R2,R0                   ;ADD SIZE OF BUFFER
004656' 062704  000024                  ADD     #.TELEN*4,R4            ;POINT TO NEXT TRANSMIT RING
004662' 062705  000016                  ADD     #.RELEN*2,R5            ;POINT TO NEXT RECEIVER RING
004666' 005303                          DEC     R3                      ;COUNT THE NUMBER TO DO
004670' 001323                          BNE     1$                      ;AND LOOP TILL DONE
004672' 000207                          RTS     PC                      ;RETURN
```

```
                                ;--
                                ;
                                ;       CRC CALCULATION ON A BLOCK OF DATA
                                ;
                                ;       ON CALL
                                ;               R1 = ADDRESS OF DATA BLOCK
                                ;               R2 = BYTE COUNT
                                ;               R3,R4 = INITIAL CRC
                                ;
                                ;       ON RETURN
                                ;               R3,R4 = CRC CODE
                                ;
                                ;--
004674' 010046                  BLKCRC: MOV     R0,-(SP)                ;SAVE R0
004676' 010146                          MOV     R1,-(SP)                ;SAVE R1
004700' 010246                          MOV     R2,-(SP)                ;SAVE R2
004702' 112100                  1$:     MOVB    (R1)+,R0                ;GET NEXT BYTE
004704' 004767  000012                  JSR     PC,GETCRC               ;CALCULATE THE CRC
004710' 077204                  2$:     SOB     R2,1$                   ;LOOP TILL DONE
004712' 012602                          MOV     (SP)+,R2                ;RESTORE R2
004714' 012601                          MOV     (SP)+,R1                ;RESTORE R1
004716' 012600                          MOV     (SP)+,R0                ;RESTORE R0
004720' 000207                          RTS     PC                      ;AND RETURN

                                ;--
                                ;
                                ;       BYTE WISE 32-BIT CRC CALCULATOR
                                ;
                                ;       ON CALL
                                ;               R0 = NEW BYTE TO ADD TO CRC
                                ;               R3,R4 = CURRENT PARTIAL CRC CODE
                                ;
                                ;       ON RETURN
                                ;               R3,R4 = UPDATED CRC
                                ;
                                ;--
004722' 010046                  GETCRC: MOV     R0,-(SP)                ;SAVE R0
004724' 010146                          MOV     R1,-(SP)                ;SAVE R1
004726' 010246                          MOV     R2,-(SP)                ;SAVE R2
004730' 042700  177400                  BIC     #^C377,R0               ;CLEAR HIGH BYTE
004734' 074004                          XOR     R0,R4                   ;MERGE NEW BYTE WITH OLD CRC
004736' 012701  166670                  MOV     #POLYH,R1               ;GET CRC POLYNOMIAL HIGH WORD
004742' 012702  101440                  MOV     #POLYL,R2               ;GET CRC POLYNOMIAL LOW WORD
004746' 012700  000010                  MOV     #8.,R0                  ;LOOP COUNT
004752' 000241                  1$:     CLC                             ;CLEAR THE CARRY
004754' 006003                          ROR     R3                      ;SHIFT RIGHT THE CRC
004756' 006004                          ROR     R4                      ;32 BITS WORTH
004760' 103002                          BCC     2$                      ;SKIP IF BIT 0 NOT SET
004762' 074103                          XOR     R1,R3                   ;EXCLUSIVE OR IN THE POLY
004764' 074204                          XOR     R2,R4                   ;BOTH HIGH AND LOW WORDS
004766' 077007                  2$:     SOB     R0,1$                   ;AND LOOP ON ALL 8 BITS
004770' 012602                          MOV     (SP)+,R2                ;RESTORE R2
004772' 012601                          MOV     (SP)+,R1                ;RESTORE R1
004774' 012600                          MOV     (SP)+,R0                ;RESTORE R0
004776' 000207                          RTS     PC                      ;AND RETURN
```



```
                                ;--
                                ;
                                ;       CONVERT 16 BIT VIRTUAL ADDRESS TO 18 BIT PHYSICAL ADDRESS
                                ;
                                ;       ON CALL
                                ;               R0 = 16 BIT VIRTUAL ADDRESS
                                ;       ON RETURN
                                ;               R0 = 16 LOWER BITS OF PHYSICAL ADDRESS
                                ;               R1 =  2 UPPER BITS OF PHYSICAL ADDRESS
                                ;
                                ;--
005000' 010067  173120          CVTPHY: MOV     R0,RBUFVA               ;SAVE IN VIRTUAL ADDRESS LOCATION
005004'                                 GETPA   RBUFVA                  ;CONVERT TO PHYSICAL
005004' 104415  000000' 000124'         GETPAS,BEGIN, RBUFVA            ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
005012' 016700  173110                  MOV     RBUFPA,R0               ;GET PHYSICAL ADDRESS
005016' 016701  173106                  MOV     RBUFEA,R1               ;GET EXTENDED ADDRESS
005022' 006201                          ASR     R1                      ;SHIFT INTO LOWER TWO BITS
005024' 006201                          ASR     R1                      ;  BIT 2
005026' 006201                          ASR     R1                      ;     BIT 1
005030' 006201                          ASR     R1                      ;        BIT 0
005032' 042701  177774                  BIC     #^C3,R1                 ;KEEP ONLY TWO BITS
005036' 000207                          RTS     PC                      ;AND RETURN

                                ;--
                                ;       SET UP EXTENDED ERROR PRINTOUT
                                ;
                                ;--
005040' 004767  000100          ERRMSG: JSR     PC,SAVREG               ;SAVE ALL OF THE REGISTERS      ;CC
005044' 016704  172736                  MOV     ADDR,R4                 ;GET ADDR OF PCSR0              ;CC
005050' 010467  173024                  MOV     R4,CSRA                 ;LOAD ADDRESS OF PCSR0          ;CC
005054' 016767  000764  173020          MOV     FLAGS,ACSR              ;LOAD CONTENTS OF PCSR0         ;CC
005062' 016467  000002  173014          MOV     .PCSR1(R4),ASTAT        ;LOAD CONTENTS OF PCSR1         ;CC
005070' 012767  000000  173010          MOV     #0.,ERRTYP              ;SET ERROR TYPE UNDEFINED       ;CC

005076' 012702  005236'                 MOV     #TDRB,R2                ;GET ADDRESS OF XMIT DESC       ;CC
005102' 012703  005762'                 MOV     #ERRPRT,R3              ;GET ADDRESS OF ERROR PRINTOUT  ;CC
005106' 012705  000005                  MOV     #.TELEN,R5              ;GET SIZE OF XMIT DESC RING     ;CC
005112' 012223                  1$:     MOV     (R2)+,(R3)+             ;CREATE TABLE                   ;CC
005114' 005305                          DEC     R5                      ;DECREMENT THE COUNTER          ;CC
005116' 001375                          BNE     1$                      ;LOOP UNTIL END OF XMIT DESC RING
005120' 012705  000007                  MOV     #.RELEN,R5              ;GET SIZE OF REC. DESC RING     ;CC
005124' 012702  005546'                 MOV     #RDRB,R2                ;GET ADDRESS OF RECEIVE DESC RING
005130' 012223                  2$:     MOV     (R2)+,(R3)+             ;ADD RECEIVE RING TO TABLE      ;CC
005132' 005305                          DEC     R5                      ;DECREMENT TABLE ENTRY COUNTER  ;CC
005134' 001375                          BNE     2$                      ;LOOP UNTIL END OF RECEIVE RING ;CC
005136' 012723  177777                  MOV     #177777,(R3)+           ;TABLE TERMINATOR               ;CC
005142' 000207                          RTS     PC                      ;AND RETURN                     ;CC
                                ;--
                                ;
                                ;       SAVE REGISTERS R0 - R5
                                ;
                                ;--
005144' 010146                  SAVREG: MOV     R1,-(SP)                ;SAVE R1
005146' 010246                          MOV     R2,-(SP)                ;SAVE R2
005150' 010346                          MOV     R3,-(SP)                ;SAVE R3
005152' 010446                          MOV     R4,-(SP)                ;SAVE R4
```

CXUACA.MAC 13-JAN-83 16:32

```
005154' 010546                  MOV     R5,-(SP)                ;SAVE R5
005156' 012746  005174'         MOV     #RSTREG,-(SP)           ;FIX A RESTORE REGISTERS ADDRESS
005162' 016646  000014          MOV     14(SP),-(SP)            ;GET THE RETURN ADDRESS
005166' 010066  177762          MOV     R0,-16(SP)              ;SAVE R0
005172' 000207                  RTS     PC                      ;AND RETURN

                        ;--
                        ;
                        ;       RESTORE REGISTERS R0 - R5
                        ;
                        ;--
005174' 012605          RSTREG: MOV     (SP)+,R5                ;RESTORE R5
005176' 012604                  MOV     (SP)+,R4                ;RESTORE R4
005200' 012603                  MOV     (SP)+,R3                ;RESTORE R3
005202' 012602                  MOV     (SP)+,R2                ;RESTORE R2
005204' 012601                  MOV     (SP)+,R1                ;RESTORE R1
005206' 012600                  MOV     (SP)+,R0                ;RESTORE R0
005210' 000207                  RTS     PC                      ;AND RETURN
```


CXUACA.MAC 13-JAN-83 16:32

```
005212' 000000  000000  000000  PCB:    .WORD   0,0,0,0                 ;PORT CONTROL BLOCK
005220' 000000

005222' 000000                  RNGBLK: .WORD   0                       ;ADDRESS OF TRANSMIT DESCRIPTOR RING BASE
005224'     000                         .BYTE   0                       ;EXTENDED ADDRESS BITS <16-17>
005225'     000                         .BYTE   0                       ;NUMBER OF WORDS IN TDRB
005226' 000000                  TPCNT:  .WORD   0                       ;NUMBER OF ENTRIES IN RING

005230' 000000                          .WORD   0                       ;ADDRESS OF RECEIVE DESCRIPTOR RING BASE
005232'     000                         .BYTE   0                       ;EXTENDED ADDRESS BITS <16-17>
005233'     000                         .BYTE   0                       ;NUMBER OF WORDS IN EACH ENTRY
005234' 000000                  RPCNT:  .WORD   0                       ;NUMBER OF ENTRIES IN RECEIVE RING

005236' 000144                  TDRB:   .BLKW   .TELEN*.MAXPKT*2        ;TRANSMIT DECRIPTOR RING ENTRIES
005546' 000106                  RDRB:   .BLKW   .RELEN*.MAXPKT          ;RECEIVE DESCRIPTOR RING ENTRIES

005762' 000031                  ERRPRT: .BLKW   25.                     ;ERROR PRINTOUT ENTRIES         ;CC

006044' 000000                  FLAGS:  .WORD   0                       ;INTERRUPT FLAG WORD
006046' 000000  000000          RDPCRC: .WORD   0,0
006052' 000022                  HEADER: .BLKW   18.
006116' 002000                  RECBUF: .BLKB   BUFSIZ
        000001                          .END
```

CXUACA.MAC 13-JAN-83 16:32 CROSS REFERENCE TABLE -- USER SYMBOLS

```
ACSR    000102R          418#  1568#
ADDR    000096R          384#   602   1014   1566
ADDR22= 001000           504#
ASB     000106R          422#  1223*  1227*
ASTAT   000104R          420#  1569#
AWAS    000110R          423#  1222*  1229*
BEGIN   000000R          381#   609    627    633    634    639    655    667    673    689    695    726    732
                         757    763    782    788    833    839    858    864    880    886    901    929    935
                         956    957    962    975    993    998   1052   1053   1078   1079   1097   1111   1117
                        1134   1140   1155   1161   1175   1181   1195   1231   1267   1282   1292   1307   1332
                        1345   1374   1388   1415   1425   1551
BIT0  = 000001           328    330    332    333    335    339    345    503#
BIT1  = 000002           328    331    332    334    335    340    502#
BIT10 = 002000           318    493#
BIT11 = 004000           319    367    492#
BIT12 = 010000           320    368    491#
BIT13 = 020000           321    365    369    373    490#
BIT14 = 040000           322    370    374    489#
BIT15 = 100000           323    371    375    488#
BIT2  = 000004           328    333    334    335    341    346    501#
BIT3  = 000010           500#
BIT4  = 000020           499#
BIT5  = 000040           315    498#
BIT6  = 000100           316    497#
BIT7  = 000200           496#
BIT8  = 000400           317    363    495#
BIT9  = 001000           364    494#
BLKCRC  004674R          816   1462   1496#
BREAK$= 104407           559#   633    634    956    957   1052   1053   1078   1079
BR1     000012R          386#   598
BR2     000013R          387#
BTOO$ = 104421           594#
BUFSIZ= 002000           293#   433    436   1403   1637
CDATA$= 104412           568#
CDERCT  000144R          438#
CDWDCT  000146R          439#
CHKBCT  004014R          984   1325#
CHKBUF  003366R          990   1211#
CHKCRC  004122R          986   1362#
CHKSTA  003552R          982   1257#
CONFIG  000056R          406#
CSRA    000100R          416#  1567#
CSRCHK  002734R          980   1091#
CVTPHY  005000R          708    713    718    824   1408   1430   1472   1549#
DATCK$= 104411           565#
DATER$= 104404           550#  1231
DROP    002024R          629    657    675    697    734    765    790    841    866    888    899#   937
DVIDI   000014R          388#
ENDIT$= 104413           571#   993    998
END$  = 104410           562#   901
ERRMSG  005040R          623    651    669    691    728    759    784    835    860    882    931    971   1107
                        1130   1151   1171   1191   1278   1303   1341   1384   1565#
ERRPRT  005762R          627    655    673    695    732    763    788    839    864    886    935    975   1111
                        1134   1155   1175   1195   1282   1307   1345   1388   1573   1632#
ERRTYP  000106R          421#  1570#
EXIT$ = 104400           538#
```


CXUACA.MAC 13-JAN-83 16:32 CROSS REFERENCE TABLE -- USER SYMBOLS

```
FLAGS   006044R          620#   621*   648*   649*   660*   923*   949   1015*  1016   1017   1027*  1047   1057*
                        1092   1177   1568   1634#
FREE    000150R          440#
GETCHC  004722R         1500   1519#
GETPAS= 104415           582#  1551
GWBUFS= 104414           579#
HDRCHC  006046R          817*   818*  1460   1461   1635#
HEADER  006052R          801    812   1406   1636#
HRDCNT  000044R          401#
HRDER$= 104405           553#   627    655    673    695    732    763    788    839    864    886    935    975
                        1111   1134   1155   1175   1195
HRDPAS  000050R          403#
ICONT   000036R          398#
ICOUNT  000040R          399#
IDNUM   000122R          428#
IMODX.= 000000           441#
INIT    000030R          395#
INTR    000120R          427#
INTSRV  002456R          597   1012#
LOOP    002034R          897    908#   996
MAP22$= 104416           585#
MODNAM  000000R          382#
MODSP   000252R          396    480#
MSG4$ = 104403           547#
MSGS$ = 104402           544#
MSG$  = 104401           541#   609    639    667    689    726    757    782    833    858    880    929    962
                        1097   1117   1140   1161   1181   1267   1292   1332   1374
NULL  = 000000           530#
OPEN  = 000000           383    389    390    391    392    409    410    411    412    413    414    415    416
                         418    420    422    423    425    426    427    431    432    434    435    437    438
                         439    440    484#
OTOA$ = 104420           591#
PASCNT  000034R          397#
PCB     005212R          717    747*   748*   751*   776*   803    825*   826*   827*  1616#
PIRQ$ = 000004           515#
POLYH = 166670           294#  1524
POLYL = 101440           295#  1525
POPSP = 005726           527#
POPSP2= 022626           528#
PRTY  = 000000           524#
PRTY0 = 000000           387    523#
PRTY1 = 000040           522#
PRTY2 = 000100           521#
PRTY3 = 000140           520#
PRTY4 = 000200           519#
PRTY5 = 000240           386    518#
PRTY6 = 000300           517#
PRTY7 = 000340           516#
PS    = 177776           485#
PSW   = 177776           486#
PUSH  = 005746           525#
PUSH2 = 024646           526#
RAND$ = 104417           588#  1415   1425
RANNUM  000054R          405#  1416   1426
RBUFEA  000130R          432#  1553
RBUFPA  000126R          431#  1552
```

CXUACA.MAC 13-JAN-83 16:32 CROSS REFERENCE TABLE -- USER SYMBOLS

```
RBUFSZ  000132R          433#
RBUFVA  000124R          430#  1549*  1551
RDRB    005546R          712    910   1019   1212   1259   1326   1363   1453   1579   1630#
RECBUF  006116R          430   1454   1637#
RESTRT  000274R          424    602#  1000
RES1    000056R          407#
RES2    000060R          408#
RNGBLK  005222R          709*   710*   711*   714*   715*   716*   823   1619#
RPCNT   005234R          908   1020   1214   1260   1327   1364   1442*  1451   1627#
RSETUP  004514R          821   1450#
RSTREG  005174R         1595   1605#
RSTRT   000112R          424#
SAVREG  005144R         1040   1069   1091   1211   1257   1325   1362   1401   1450   1565   1590#
SRADR   000102R          417#  1226*
SETCMD  002556R          663    722    753    778    829    854    925   1040#
SOFCNT  000042R          400#
SOFERS= 104406           556#  1282   1307   1345   1388
SOFPAS  000046R          402#
SPUINT  000032R          396#
SPSIZ = 000040             1#   443
SR1     000016R          389#   749    988   1233
SR2     000020R          390#
SR3     000022R          391#
SR4     000024R          392#
START   000252R          395    596#  1428
STAT    000026R          394#
SVR0    000062R          409#
SVR1    000064R          410#
SVR2    000066R          411#
SVR3    000070R          412#
SVR4    000072R          413#
SVR5    000074R          414#
SVR6    000076R          415#
SYSCNT  000052R          404#
TDRB    005236R          707    909   1213   1258   1404   1452   1572   1629#
TPCNT   005226R         1402#  1436*  1437   1442   1443*  1622#
TRPDFD= 000022           529#   538    539#   541    542#   544    545#   547    548#   550    551#   553    554#
                         556    557#   559    560#   562    563#   565    566#   568    569#   571    572#   579
                         580#   582    583#   585    586#   588    589#   591    592#   594    595#
TSETUP  004254R          420   1401#
VECTOR  000010R          385#   596
WASADR  000104R          419#  1228*
WBUFEA  000136R          435#
WBUFPA  000134R          434#
WBUFPQ  000140R          436#
WBUFSZ  000142R          437#
WDFR    000116R          426#
WOTQ    000114R          425#
WSTATE  002656R          685    876   1069#
XFLAG   000005R          383#
.       = 010116R        618#   704#   744#   773#   799#   850#   873#   895#   945#  1104#  1188#  1300#  1629#
                        1630#  1632#  1636#  1637#
.BHALT= 000007           335#
.BOOT = 000005           309#
.BUFL = 100000           375#
.CHKAL= 000002           340#  1233
```


CXUACA.MAC 13-JAN-83 16:32 CROSS REFERENCE TABLE -- USER SYMBOLS

```
.CMD  = 000002           306#   754    779    830
.CRC  = 004000           367#  1288
.CRCH = 000014           361#  1370   1466*
.CRCL = 000012           360#  1368   1465*
.DCHFC= 000001           339#   988
.DNI  = 004000           319#   630    658   1047   1057
.ENP  = 000400           363#  1263   1288   1432
.ERRI = 142400           324#   950
.ERRS = 040000           370#  1288
.FATI = 000400           317#   324    605   1093
.FDX  = 000001           345#
.FRAM = 020000           369#
.INTE = 000100           316#   659    995   1043
.INTER= 000004           341#   749
.LOOP = 000004           346#   751
.MATCH= 020000           365#  1263
.MLEN = 000006           357#   919*  1218   1328   1366   1475*
.MXPKT= 000012           352#  1437   1629   1630
.NCHN = 020000           373#
.NHALT= 000006           334#
.NOP  = 000000           304#
.OFLO = 010000           368#
.OWN  = 100000           371#   912    915    918
.PCBB = 000001           305#   723
.PCEI = 040000           322#   324   1136
.PCSR0= 000000           297#   603*   605    621    630    649    658*   659*   899*   977*   979*   995*  1015
                        1016*  1044*
.PCSR1= 000002           298#  1073   1569
.PCSR2= 000004           299#   719*
.PCSR3= 000006           300#   720*
.PDMD = 000010           310#   926
.PLOAD= 000001           330#
.RCBI = 002000           318#   324   1113
.READY= 000002           331#   686
.RELEN= 000007           351#   716    920   1023   1244   1312   1349   1392   1479   1578   1630
.RESET= 000000           329#
.REST = 000040           315#   603
.RUN  = 000003           332#   877
.RXI  = 020000           321#   952    953   1017   1027
.SBUF = 000002           355#  1409*  1431*  1473*
.SERI = 100000           323#   324   1157
.SLEN = 000000           354#  1328   1405*  1423*  1457   1470*
.START= 000004           308#   855
.STAT = 000004           356#   911*   912*   914*   915*   917*   918*  1021   1261   1286   1411*  1433*  1474*
.STATE= 000007           328#  1074
.STEST= 000003           307#
.STOP = 000017           311#   664
.STP  = 001000           364#  1288   1410
.TDR  = 000006           358#  1412*  1434*
.TELEN= 000005           350#   711    913    916   1213   1243   1258   1311   1413   1435   1452   1478   1574
                        1629
.TXI  = 010000           320#   952    953   1177
.UBTO = 040000           374#
.UHALT= 000005           333#
.VA   = 000010           359#  1215   1216   1365   1407*  1429*  1455*  1456   1471   1476
```

```
CXUACA.MAC    13-JAN-83 16:32          CROSS REFERENCE TABLE -- MACRO NAMES

ABKMOD       1#
ABREAK       1#     632     955    1051    1077
ATOD         1#
CKDATA       1#
DATACK       1#
DATERR       1#    1230
DEMAND     269#     662     721     752     777     828     853     924
DFSERVN      1#     537     540     543     546     549     552     555     558     561     564     567     570
DSERVNT      1#     578     581     584     587     590     593
END          1#     900
ENDIT        1#     992     997
ENDMOD       1#
EQUATS       1#     482
EXIT         1#
GETPA        1#    1550
GWBUFF       1#
HRDER        1#     622     650     668     690     727     758     783     834     859     881     930     970    1106    1129
            1150    1170    1190
IOMOD        1#
IOMODP       1#
IOMODR       1#
IOMODX       1#     377
MAP22        1#
MODULE       1#     378
MSG          1#     608     638     666     688     725     756     781     832     857     879     928     961    1096    1116
            1139    1160    1180    1266    1291    1331    1373
MSGN         1#
MSGS         1#
NBKMOD       1#
OTOA         1#
PIRQ         1#
PRINT      261#     607     637     960    1095    1115    1138    1159    1179    1265    1290    1330    1372
RAND         1#    1414    1424
SHKMOD       1#
SUPER        1#    1277    1302    1340    1383
WAIT       281#     684     875

. ABS.  000000    000
        010116    001

ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

CXUACA.OBJ,CXUACA.LST/CRF/SOL/ML:TOC=DDXCOM.P11,CXUACA.MAC
RUN-TIME: 2 3 .5 SECONDS
RUN-TIME RATIO: 88/7=12.6
```

DVID1: 1 BIT SET FOR EACH DH11 LINE. ITS POSITION SHOULD CORROSPOND WITH THE LINE #. E.G. IF DH11 LINE 6 IS USED 'DVID1' BIT 6 SHOULD BE SET.

6. DEVICE SETUP

A. THE USER MUST LOAD AND START TEST 21 OF MAINDEC-11-DBVTA IN THE VT20 PDP11/05 IN ORDER FOR THIS MODULE TO EXERCISE. CONSULT THE ABOVE DOCUMENT AND COMPLY WITH THE OPERATING INSTRUCTIONS FOR TEST 21 (SECTION 26). THIS DEC/X11 MODULE EXPECTS THE USER TO ENTER DATA ON EACH SELECTED TUBE AND SET EACH TUBE IN THE CONTINUOUS TRANSMIT MODE. THIS STEP IS TAKEN AFTER THE DEC/X11 EXERCISER HAS BEEN STARTED BY THE 'RUN' COMMAND. TYPICAL USER ACTION ON EACH SELECTED TUBE WILL BE AS FOLLOWS:

KEY	FUNCTION
CTRL E	CLEAR SCREEN
CTRL W	GENERATE WORST CASE CHARACTER PATTERN ON TOP OF SCREEN
CTRL T	CONTINUOUS TRANSMIT TO DEC/X11 MODULE (DEC/X11 MODULE WILL RECEIVE DATA AND TRANSMIT IT BACK TO BOTTOM OF SCREEN)

NOTE: IF THE CHARACTER PATTERN FAILS TO RETURN ON THE BOTTOM OF THE SCREEN AFTER ONE 'CTRL T', THEN RETRY AFTER 'END PASS' IS REPORTED FOR THIS DEC/X11 MODULE (DL11 RECEIVERS ARE TURNED OFF SECONDS BEFORE 'END PASS' MSG). IF DATA IS STILL NOT RETURNED FROM HOST COMPUTER (DEC/X11 SYSTEM) THEN VERIFY THE VT20 HOST COMPUTER BY RUNNING MAINDEC-11-DZVTE.

B. IF LINES WITH BAUD RATES OTHER THAT 9600 ARE TO BE USED, THEN THE VALUE OF THE CORRESPONDING WORD IN THE BAUD RATE TABLE (16 WORDS STARTING AT LOC 'LOBR') MUST BE MODIFIED. REFER TO THE PDP-11 PERIPHERALS AND INTERFACING HANDBOOK FOR THE EXACT VALUES NEEDED

8. OPERATOR OPTIONS

A. THE USER CAN MODIFY (VTA 14) "DVID1" TO SELECT OR DESELECT INDIVIDUAL VT20'S. THIS MODULE IS QUITE ABLE TO HANDLE VT20'S THAT DO NOT HAPPEN TO HAVE ADJACENT DH11 LINES.

B. THE USER CAN USE THE "MOD" COMMAND TO DUMP THE TABLES OR BUFFERS DESCRIBED IN 7.2 TO OBTAIN MORE DETAILED ERROR INFORMATION.

9. ERROR PRINTOUTS

9.1 ERROR FORMAT - RECEIVE

CSRA = CSR ADDRESS
CSRC = NRC WORD AS FOLLOWS:

THE # PRINTED OUT LABELED AS 'STATC' IS THE NEXT RECIEVED CHARACTER
BIT 15 = DATA PRESENT
BIT 14 = OVERRUN
BIT 13 = FRAMING
BIT 12 = PARITY
BIT 11-8 = LINE #
BIT 7-0 = DATA RECEIVED

WITH SOME ERRORS SUCH AS "NO DH11 LINES REMAIN SELECTED" THE CONTENTS OF THE DH11 REGISTERS ARE IRRELEVENT. IN SUCH CASES THEY ARE PRINTED ANYWAYS.

9.2 ERROR FORMAT - TRANSMIT

CSRA = CSR ADDRESS
CSRC = CSR CONTENTS AS FOLLOWS:

BIT 7 = XMITR READY
BIT 6 = XMITR INTERRUPT ENABLED

.REM -

IDENTIFICATION

PRODUCT CODE: AC-E956B-MC

PRODUCT NAME: CXVTBB0 DH11/VT20 MODULE

PRODUCT DATE: SEPTEMBER 1978

MAINTAINER: DEC/X11 SUPPORT GROUP

1. ABSTRACT

VTB IS AN IOMODX THAT EXERCISES UP TO FOUR VT20'S (DH11 LINES). IT IS INTENDED TO BE A DATA HANDLING ROUTINE USED IN CONJUNCTION WITH TEST 21 OF MAINDEC-11-DBVTA (PREVIOUSLY LOADED AND RUNNING IN THE VT20'S PDP11-05). DATA IS ENTERED AT EACH SELECTED TUBE AND SET INTO THE CONTINUOUS TRANSMIT MODE. THIS DATA IS THEN RECEIVED AND RETRANSMITTED BY THE VT20 HOST COMPUTER (THIS DEC/X11 MODULE). ALL LINES SELECTED FOR TEST CAN BE ACTIVATED AND RUN CONCURRENTLY. ALL TRANSMIT AND RECEIVE ERRORS ARE REPORTED ON THE CONSOLE TTY. NO DATA ERRORS ARE REPORTED BY THIS MODULE.

2. REQUIREMENTS

HARDWARE: AT LEAST ONE VT20 CONNECTED VIA A DH11

STORAGE:: VTB REQUIRES:
1. DECIMAL WORDS: 4002
2. OCTAL WORDS: 07642
3. OCTAL BYTES: 17504

3. PASS DEFINITION

ONE PASS OF THE VTAB MODULE CONSISTS OF CONTINUOUSLY RECEIVING AND TRANSMITTING THE DATA ENTERED ON ALL SELECTED LINES FOR THE PERIOD DEFINED BELOW.

4. EXECUTION TIME

EXECUTION TIME VARIES WITH THE NUMBER OF JOBS (MODULES) ACTIVE, THE BAUD RATE AND THE NUMBER OF TUBES BEING EXERCISED. HOWEVER, THIS MODULE RUNNING ALONE WILL TAKE NO MORE THAN 3 MINUTES WITH 16 TUBES AT 110 BAUD

5. CONFIGURATION PARAMETERS

DEFAULT PARAMETERS:

DVA:160020, VCT:350, BR1:5, BR2:0, DVC:1

LOBR-L17BR:IF ANY OF THE DH11 LINES IS NOT A 9600 BAUD LINE THE WORD ASSOCIATED WITH THAT LINE MUST BE MODIFIED BEFORE RUNNING

REQUIRED PARAMETERS:

DVC: NO OF VT20'S IF GREATER THAN 1

DVID1: 1 BIT SET FOR EACH DH11 LINE. ITS POSITION SHOULD CORRESPOND WITH THE LINE #. E.G. IF DH11 LINE 6 IS USED 'DVID1' BIT 6 SHOULD BE SET.

6. DEVICE SETUP

A. THE USER MUST LOAD AND START TEST 21 OF MAINDEC-11-DBVTA IN THE VT20 PDP11/05 IN ORDER FOR THIS MODULE TO EXERCISE. CONSULT THE ABOVE DOCUMENT AND COMPLY WITH THE OPERATING INSTRUCTIONS FOR TEST 21 (SECTION 26). THIS DEC/X11 MODULE EXPECTS THE USER TO ENTER DATA ON EACH SELECTED TUBE AND SET EACH TUBE IN THE CONTINUOUS TRANSMIT MODE. THIS STEP IS TAKEN AFTER THE DEC/X11 EXERCISER HAS BEEN STARTED BY THE 'RUN' COMMAND. TYPICAL USER ACTION ON EACH SELECTED TUBE WILL BE AS FOLLOWS:

KEY	FUNCTION
CTRL E	CLEAR SCREEN
CTRL W	GENERATE WORST CASE CHARACTER PATTERN ON TOP OF SCREEN
CTRL T	CONTINUOUS TRANSMIT TO DEC/X11 MODULE (DEC/X11 MODULE WILL RECEIVE DATA AND TRANSMIT IT BACK TO BOTTOM OF SCREEN)

NOTE: IF THE CHARACTER PATTERN FAILS TO RETURN ON THE BOTTOM OF THE SCREEN AFTER ONE 'CTRL T', THEN RETRY AFTER 'END PASS' IS REPORTED FOR THIS DEC/X11 MODULE (DL11 RECEIVERS ARE TURNED OFF SECONDS BEFORE 'END PASS' MSG). IF DATA IS STILL NOT RETURNED FROM HOST COMPUTER (DEC/X11 SYSTEM) THEN VERIFY THE VT20 HOST COMPUTER BY RUNNING MAINDEC-11-D2VTE.

B. IF LINES WITH BAUD RATES OTHER THAT 9600 ARE TO BE USED, THEN THE VALUE OF THE CORRESPONDING WORD IN THE BAUD RATE TABLE (16 WORDS STARTING AT LOC "LOBR") MUST BE MODIFIED. REFER TO THE PDP-11 PERIPHERALS AND INTERFACING HANDBOOK FOR THE EXACT VALUES NEEDED

8. OPERATOR OPTIONS

A. THE USER CAN MODIFY (VTA 14) "DVID1" TO SELECT OR DESELECT INDIVIDUAL VT20-S. THIS MODULE IS QUITE ABLE TO HANDLE VT20'S THAT DO NOT HAPPEN TO HAVE ADJACENT DH11 LINES.

B. THE USER CAN USE THE "MOD" COMMAND TO DUMP THE TABLES OR BUFFERS DESCRIBED IN 7.2 TO OBTAIN MORE DETAILED ERROR INFORMATION.

9. ERROR PRINTOUTS

9.1 ERROR FORMAT - RECEIVE

CSRA = CSR ADDRESS
CSRC = NRC WORD AS FOLLOWS:

THE # PRINTED OUT LABELED AS 'STATC' IS THE NEXT RECIEVED CHARACTER
BIT 15 = DATA PRESENT
BIT 14 = OVERRUN
BIT 13 = FRAMING
BIT 12 = PARITY
BIT 11-8 = LINE #
BIT 7-0 = DATA RECEIVED

WITH SOME ERRORS SUCH AS "NO DH11 LINES REMAIN SELECTED" THE CONTENTS OF THE DH11 REGISTERS ARE IRRELEVENT. IN SUCH CASES THEY ARE PRINTED ANYWAYS.

9.2 ERROR FORMAT - TRANSMIT

CSRA = CSR ADDRESS
CSRC = CSR CONTENTS AS FOLLOWS:

BIT 7 = XMITR READY
BIT 6 = XMITR INTERRUPT ENABLED

.LIST SEQ,BIN,TOC

```
                                        ;SOME DEFINITIONS
000000'                                 IOMODX  <VTBB >,160020,350,5,,,1,126    ;BR LEVEL 5, UP TO 16 DEVICES
000000'                                 MODULE  150000,VTBB ,160020,350,5,,,1,126
                                        .TITLE  VTBB DEC/X11 SYSTEM EXERCISER MODULE
                                ;       DDXCOM  VERSION  6      23-MAY-78
                                        .LIST   BIN
                                ;*****************************************************************************
000000'                                 BEGIN:
000000' 052126  041102     040  MODNAM: .ASCII  /VTBB / ;MODULE NAME.
000005'     000                 XFLAG:  .BYTE   OPEN                    ;USED TO KEEP TRACK OF WBUFF USAGE
000006' 160020                  ADDR:   160020+0                        ;1ST DEVICE ADDR.
000010' 000350                  VECTOR: 350+0                           ;1ST DEVICE VECTOR.
000012'     240                 BR1:    .BYTE   PRTY5+0                 ;1ST BR LEVEL.
000013'     000                 BR2:    .BYTE   PRTY+0                  ;2ND BR LEVEL.
000014' 000001                  DVID1:  +1                              ;DEVICE INDICATOR 1.
000016' 000000                  SR1:    OPEN                            ;SWITCH REGISTER 1
000020' 000000                  SR2:    OPEN                            ;SWITCH REGISTER 2
000022' 000000                  SR3:    OPEN                            ;SWITCH REGISTER 3
000024' 000000                  SR4:    OPEN                            ;SWITCH REGISTER 4
                                ;*****************************************************************************
000026' 150000                  STAT:   150000                          ;STATUS WORD.
000030' 000554'                 INIT:   START                           ;MODULE START ADDR.
000032' 000224'                 SPOINT: MODSP                           ;MODULE STACK POINTER.
000034' 000000                  PASCNT: 0                               ;PASS COUNTER.
000036' 000001                  ICONT:  1                               ;# OF ITERATIONS PER PASS=1
000040' 000000                  ICOUNT: 0                               ;LOC TO COUNT ITERATIONS
000042' 000000                  SOFCNT: 0                               ;LOC TO SAVE TOTAL SOFT ERRORS
000044' 000000                  HRDCNT: 0                               ;LOC TO SAVE TOTAL HARD ERRORS
000046' 000000                  SOFPAS: 0                               ;LOC TO SAVE SOFT ERRORS PER PASS
000050' 000000                  HRDPAS: 0                               ;LOC TO SAVE HARD ERRORS PER PASS
000052' 000000                  SYSCNT: 0                               ;# OF SYS ERRORS ACCUMULATED
000054' 000000                  RANNUM: 0                               ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000056'                         CONFIG:                                 ;RESERVED FOR MONITOR USE
000056' 000000                  RES1:   0                               ;RESERVED FOR MONITOR USE
000060' 000000                  RES2:   0                               ;RESERVED FOR MONITOR USE
000062' 000000                  SVR0:   OPEN                            ;LOC TO SAVE R0.
000064' 000000                  SVR1:   OPEN                            ;LOC TO SAVE R1.
000066' 000000                  SVR2:   OPEN                            ;LOC TO SAVE R2.
000070' 000000                  SVR3:   OPEN                            ;LOC TO SAVE R3.
000072' 000000                  SVR4:   OPEN                            ;LOC TO SAVE R4.
000074' 000000                  SVR5:   OPEN                            ;LOC TO SAVE R5.
000076' 000000                  SVR6:   OPEN                            ;LOC TO SAVE R6.
000100' 000000                  CSRA:   OPEN                            ;ADDR OF CURRENT CSR.
000102'                         SBADR:                                  ;ADDR OF GOOD DATA, OR
000102' 000000                  ACSR:   OPEN                            ;CONTENTS OF CSR.
000104'                         WASADR:                                 ;ADDR OF BAD DATA, OR
000104' 000000                  ASTAT:  OPEN                            ;STATUS REG CONTENTS.
000106'                         ERRTYP:                                 ;TYPE OF ERROR
000106' 000000                  ASB:    OPEN                            ;EXPECTED DATA.
000110' 000000                  AWAS:   OPEN                            ;ACTUAL DATA.
000112' 000652'                 RSTRT:  RESTRT                          ;RESTART ADDRESS AFTER END OF PASS
000114' 000000                  WDTO:   OPEN                            ;WCRDS TO MEMORY PER ITERATION
000116' 000000                  WDFR:   OPEN                            ;WCRDS FROM MEMORY PER ITERATION
000120' 000000                  INTR:   OPEN                            ;# OF INTERRUPTS PER ITERATION
000122' 000126                  IDNUM:  126                             ;MODULE IDENTIFICATION NUMBER=126
                 000040                 .REPT   SPSIZ                   ;MODULE STACK STARTS HERE.
```



```
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000224'                         MODSP:
                                ;*****************************************************************************
                                ;SOME POINTERS AND VARIABLES AND CONSTANTS UNIQUE TO THIS MODULE
000224' 000000                  SELCNT: 0                               ;PROGRAM SETS THIS LOC TO = # OF VT20'S SELECTED
000226' 000000                  SAVEIO: 000000                          ;PLACE TO STUFF IO WHILE SERVICING INTERRUPTS
000230' 000000                  WAITIM: 000000                          ;USED AS A COUNTER FOR THE INTERRUPT WAIT LOOP
000232' 000000                  RELOC:  000000                          ;# RELOCATION COUNT FOR INDEX REGISTERS IF NEEDED
000234' 000000                  SVT0:   000000
000236' 000000                  SVT1:   000000
000240' 000000                  XCNT:   000000                          ;COUNT FOR # OF XFERS TO DO EACH PASS
000242' 000000                  VIRTAD: 000000                          ;VIRTUAL ADDRESS
000244' 000000                  PHYSAD: 000000                          ;PHYSICAL ADDRESS
000246' 000000                  EXTAD:  000000                          ;EXTENDED ADDRESS BITS
000250' 000000                  DUNCNT: 000000                          ;# OF DEVICES FINISHED
000252' 000000                  WAITIM: 000000                          ;TIMER COUNT
000254' 000000                  TEMP:   000000                          ;TEMPORARY LOCATION
000256' 000000                  TEMP0:  0                               ;GENERAL PURPOSE TEMPORARY STORAGE
000260' 000000                  TEMP1:  0                               ;GENERAL PURPOSE TEMPORARY STORAGE
000262' 000000                  COUNT1: 000000                          ;SET BY BY USER AT CONFIGURATION TIME
                                                                        ;TO A 3 IF LINES OF 1200 BAUD OR LESS ARE USED
000264' 000000                  INTSW:  000000                          ;SWITCH=1 IF WE SHOULD RETURN FROM INTERRUPT
                                                                        ;SERVICE USING AN 'RTI' INSTRUCTION
000266' 000000                  FATERR: 000000                          ;SWITCH=1 TO SIGNAL A FATAL ERROR.
000270' 000000                  ERPT1:  000000                          ;POINTS TO LAST WORD SERVICED FROM ERROR QUEUE
000272' 000000                  ERPT2:  000000                          ;POINTS TO LAST WORD ENTERED IN ERROR QUEUE

000274' 002542'                 BPLIST: VTBF0                           ;POINTS TO ONE OF THE 16 BUFFERS
000276' 003350'                         VTBF1                           ;POINTS TO 1 OF THE 16 BUFFERS
000300' 004156'                         VTBF2                           ;POINTS TO 1 OF THE 16 BUFFERS
000302' 004764'                         VTBF3                           ;POINTS TO ONE OF THE 16 BUFFERS
000304' 005572'                         VTBF4           ;POINTS TO ONE OF THE 16 BUFFERS
000306' 006400'                         VTBF5           ;POINTS TO ONE OF THE 16 BUFFERS
000310' 007206'                         VTBF6           ;POINTS TO ONE OF THE 16 BUFFERS
000312' 010014'                         VTBF7           ;POINTS TO ONE OF THE 16 BUFFERS
000314' 010622'                         VTBF10          ;POINTS TO ONE OF THE 16 BUFFERS
000316' 011430'                         VTBF11          ;POINTS TO ONE OF THE 16 BUFFERS
000320' 012236'                         VTBF12          ;POINTS TO ONE OF THE 16 BUFFERS
000322' 013044'                         VTBF13          ;POINTS TO ONE OF THE 16 BUFFERS
000324' 013652'                         VTBF14          ;POINTS TO ONE OF THE 16 BUFFERS
000326' 014460'                         VTBF15          ;POINTS TO ONE OF THE 16 BUFFERS
000330' 015266'                         VTBF16          ;POINTS TO ONE OF THE 16 BUFFERS
000332' 016074'                         VTBF17          ;POINTS TO ONE OF THE 16 BUFFERS
                                ;EACH OF THE NEXT 8 WORDS IS SETUP BY THIS MODULE TO POINT TO A DH11 REGISTER
000334' 000000                  X00:    0               ;POINTS TO THE SYSTEM CONTROL REGISTER
000336' 000000                  X02:    0               ;POINTS TO THE NEXT CHARACTER RECEIVED REGISTER
000340' 000000                  X04:    0               ;POINTS TO THE LINE PARAMETER REGISTER
000342' 000000                  X06:    0               ;POINTS TO THE CURRENT ADDRESS REGISTER
000344' 000000                  X10:    0               ;POINTS TO THE BYTE COUNT REGISTER
000346' 000000                  X12:    0               ;POINTS TO THE BUFFER ACTIVE REGISTER
000350' 000000                  X14:    0               ;POINTS TO THE BREAK CONTROL REGISTER
000352' 000000                  X16:    0               ;POINTS TO THE SILO STATUS REGISTER
```

```
                                ;TABLE OF BUFFER POINTERS FOR DH11 LINES
000354' 000000                  L0SW:   0                       ;BUFFER POINTER FOR DH11 LINE 0
000356' 000000                  L1SW:   0                       ;BUFFER POINTER FOR DH11 LINE 1
000360' 000000                  L2SW:   0                       ;BUFFER POINTER FOR DH11 LINE 2
000362' 000000                  L3SW:   0                       ;BUFFER POINTER FOR DH11 LINE 3
000364' 000000                  L4SW:   0                       ;BUFFER POINTER FOR DH11 LINE 4
000366' 000000                  L5SW:   0                       ;BUFFER POINTER FOR DH11 LINE 5
000370' 000000                  L6SW:   0                       ;BUFFER POINTER FOR DH11 LINE 6
000372' 000000                  L7SW:   0                       ;BUFFER POINTER FOR DH11 LINE 7
000374' 000000                  L10SW:  0                       ;BUFFER POINTER FOR DH11 LINE 10
000376' 000000                  L11SW:  0                       ;BUFFER POINTER FOR DH11 LINE 11
000400' 000000                  L12SW:  0                       ;BUFFER POINTER FOR DH11 LINE 12
000402' 000000                  L13SW:  0                       ;BUFFER POINTER FOR DH11 LINE 13
000404' 000000                  L14SW:  0                       ;BUFFER POINTER FOR DH11 LINE 14
000406' 000000                  L15SW:  0                       ;BUFFER POINTER FOR DH11 LINE 15
000410' 000000                  L16SW:  0                       ;BUFFER POINTER FOR DH11 LINE 16
000412' 000000                  L17SW:  0                       ;BUFFER POINTER FOR DH11 LINE 17

                                ;TABLE OF BAUD RATES FOR DH11 LINES
000414' 033500                  L0BR:   033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 0
000416' 033500                  L1BR:   033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 1
000420' 033500                  L2BR:   033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 2
000422' 033500                  L3BR:   033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 3
000424' 033500                  L4BR:   033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 4
000426' 033500                  L5BR:   033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 5
000430' 033500                  L6BR:   033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 6
000432' 033500                  L7BR:   033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 7
000434' 033500                  L10BR:  033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 10
000436' 033500                  L11BR:  033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 11
000440' 033500                  L12BR:  033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 12
000442' 033500                  L13BR:  033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 13
000444' 033500                  L14BR:  033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 14
000446' 033500                  L15BR:  033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 15
000450' 033500                  L16BR:  033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 16
000452' 033500                  L17BR:  033500                  ;DEFAULT BAUD RATE (9600) FOR DH11 LINE 17


                                ;EACH OF THE FOLLOWING 16 WORDS CONTAINS THE ERROR AND STATUS INFORMATION
                                ;FOR 1 DH11 LINE. THE HIGH ORDER BYTE HOLDS THE # OF ERRORS IN THAT LINE
                                ;AND THE LOW ORDER BYTE CONTAINS THE LINES CURRENT STATUS
                                ;LOW BYTE
                                ;       MODE
                                ;   0       UNSELECTED OR DROPPED
                                ;   2       ACTIVE MODE
                                ;   4       DONE RECIEVING
000454' 000000                  L0EC:   0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 0
000456' 000000                  L1EC:   0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 1
000460' 000000                  L2EC:   0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 2
000462' 000000                  L3EC:   0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 3
000464' 000000                  L4EC:   0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 4
000466' 000000                  L5EC:   0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 5
000470' 000000                  L6EC:   0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 6
000472' 000000                  L7EC:   0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 7
000474' 000000                  L10EC:  0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 10
000476' 000000                  L11EC:  0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 11
000500' 000000                  L12EC:  0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 12
```



```
000502' 000000                  L13EC:  0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 13
000504' 000000                  L14EC:  0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 14
000506' 000000                  L15EC:  0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 15
000510' 000000                  L16EC:  0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 16
000512' 000000                  L17EC:  0                       ;ERROR COUNT/LINE STATUS FOR DH11 LINE 17


                                ;TABLE OF CHARACTER COUNT LOCATIONS FOR DH11 LINES
000514' 000000                  L0CC:   0                       ;CHARACTER COUNT FOR DH11 LINE 0
000516' 000000                  L1CC:   0                       ;CHARACTER COUNT FOR DH11 LINE 1
000520' 000000                  L2CC:   0                       ;CHARACTER COUNT FOR DH11 LINE 2
000522' 000000                  L3CC:   0                       ;CHARACTER COUNT FOR DH11 LINE 3
000524' 000000                  L4CC:   0                       ;CHARACTER COUNT FOR DH11 LINE 4
000526' 000000                  L5CC:   0                       ;CHARACTER COUNT FOR DH11 LINE 5
000530' 000000                  L6CC:   0                       ;CHARACTER COUNT FOR DH11 LINE 6
000532' 000000                  L7CC:   0                       ;CHARACTER COUNT FOR DH11 LINE 7
000534' 000000                  L10CC:  0                       ;CHARACTER COUNT FOR DH11 LINE 10
000536' 000000                  L11CC:  0                       ;CHARACTER COUNT FOR DH11 LINE 11
000540' 000000                  L12CC:  0                       ;CHARACTER COUNT FOR DH11 LINE 12
000542' 000000                  L13CC:  0                       ;CHARACTER COUNT FOR DH11 LINE 13
000544' 000000                  L14CC:  0                       ;CHARACTER COUNT FOR DH11 LINE 14
000546' 000000                  L15CC:  0                       ;CHARACTER COUNT FOR DH11 LINE 15
000550' 000000                  L16CC:  0                       ;CHARACTER COUNT FOR DH11 LINE 16
000552' 000000                  L17CC:  0                       ;CHARACTER COUNT FOR DH11 LINE 17


                                ;DETERMINE IF ANY DH'S ARE SELECTED, IF SO SETUP DEVICE REGISTER
                                ;ADDRESSES, INTERRUPT VECTORS, AND START THE
                                ;MODULE PROCESSING... IF NOT, DROP THE MODULE FROM THE RUN

000554' 012767  000004  177336  START:  MOV     #4,INTR         ;4 INTERRUPTS / ITERATION
000562' 012767  000020  177324          MOV     #20,WDTO        ;20 WORDS TO MEM/ITERATION
000570' 012767  000020  177320          MOV     #20,WDFR        ;20 WORDS FROM MEM/ITERATION
000576' 012777  004400  177202          MOV     #4004400,@ADDR  ;INIT THE DH11
000604' 012767  000004  177440          MOV     #4,WAITLM       ;MAKE A LONG 1ST PASS TO ALLOW FOR STARTING VT20S
000612' 016706  177214                  MOV     SPOINT,SP       ;SETUP THE MODULE STACK
000616' 005767  177172                  TST     DVID1           ;CHECK FOR SELLECTED VT20'S
000622' 001155                          BNE     MODROP          ;ANY SELLECTED?
                                ;WE END UP AT "DROP" IF NO VT20 IS SELECTED FOR TEST (DVID1=0)
                                ;WE WILL NOW DROP THIS MODULE
000624' 016767  177156  177502  DROP:   MOV     ADDR,X00        ;SETUP ADDRESS OF THE DH11 CONTROL/STATUS REG
000632' 012777  004000  177474          MOV     #4000,@X00      ;RC. PREVENT DH11 INTERRUPTS
000640' 104403  000000' 017424-         MSG$,BEGIN,ERRMS        ;ASCII MESSAGE CALL WITH COMMON HEADER
000646' 104410  000000'                 END$,BEGIN              ;DROP THIS MODULE


                                ;THIS IS THE RESTART ROUTINE. IT RESTARTS IN TRANSMIT MODE, ALL LINES THAT
                                ;FINISHED RECIEVING ON THE LAST PASS. ALL OTHER LINES IF IGNCRS
000652' 116767  177537  177360  RESTRT: MOVB    L0BR+1,XCNT     ;GET BAUD RATE INFO
000660' 042767  177774  177352          BIC     #177774,XCNT    ;A ROUGH GUESS
000666' 005267  177346                  INC     XCNT            ;AT HOW MANY XFERS WE SHOULD DO PER PASS
000672' 006167  177342                  ROL     XCNT            ;MULT XCNT BY 16
000676' 006167  177336                  ROL     XCNT            ;TO GET THE NUMBER
000702' 006167  177332                  ROL     XCNT            ;OF TRANSFERS TO DO
000706' 006167  177326                  ROL     XCNT            ;EACH PASS
000712' 005467  177322                  NEG     XCNT            ;MAKE IT NEGATIVE
000716' 005000                  NXTFER: CLR     R0              ;START WITH LINE 0
```

```
000720' 012767  000003  177324          MOV     #3,WAIT1M       ;REGULAR PASS TIMEOUT
000726' 005067  177324                  CLR     TEMP0           ;INIT WORD WE ARE GOING TO SEND TO X12
000732' 005067  177312                  CLR     DUNCNT          ;CLEAR OUT COUNT FOR # OF LINES
000736' 126027  000454' 000004  1$:     CMPB    LQEC(0),#4      ;IS LINE READY TO XMIT?
000744' 001054                          BNE     2$              ;IF NOT, GO SEE ABOUT NEXT LINE
000746' 012760  000003  000454'         MOV     #3,LQEC(0)      ;IF IT IS READY, SET IT TO ACTIVE MODE
000754' 000261                          SEC                     ;SET THE C BIT AND
000756' 006067  177274                  ROR     TEMP0           ;ROTATE IT INTO THE NEW "BAR" REGISTER
000762' 005367  177262                  DEC     DUNCNT          ;ADD 1 TO LINE COUNT
000766' 016067  177262                  MOV     R0,TEMP         ;GRAB THE LINE #
000772' 006067  177256                  ROR     TEMP            ;DIVIDE IT BY2 TO MAKE IT REAL
000776' 042767  177760  177250          BIC     #177760,TEMP    ;MAKE SURE ONLY THE LINE # IS SET
001004' 016777  177244  177322          MOV     TEMP,@X00       ;PUT IT INTO THE STATUS REGISTER
001012' 016067  000514' 177234          MOV     LOCC(0),TEMP    ;NOW GET THE CHARACTER COUNT
001020' 005467  177230                  NEG     TEMP            ;USE ITS 2'S COMPLIMENT
001024' 016777  177224  177312          MOV     TEMP,@X10       ;AND SETUP THE BYTE COUNT REGISTER WITH IT
001032' 016067  000554' 177202          MOV     LOSW(0),VIRTAD  ;GET VIRTUAL ADDRESS OF THE BUFFER
001040' 104415  000000' 000242'         GETPA$,BEGIN,VIRTAD     ;GET PHYSICAL ADDRESS FROM 16-BIT VIRTAD
001046' 042767  177177  177172          BIC     #177(7),EXTAD   ;CLEAR ALL BUT THE EXTEND BITS
001054' 056777  177166  177252          BIS     EXTAD,@X00      ;SET ADDRESS EXTEND BITS IN STATUS REG
001062' 016777  177156  177252          MOV     PHYSAD,@X06     ;SETUP CURRENT ADDRESS FOR THIS LINE
001070' 005060  000514'                 CLR     LOCC(0)         ;AND REINIT ITS CHARACTER COUNT
001074' 000403                          BR      3$              ;GO SEE IF WE HAVE CHECKED ALL LINES YET
001076' 000241                  2$:     CLC                     ;CLEAR C BIT TO
001100' 006067  177152                  ROR     TEMP0           ;MAKE SURE THAT THIS LINES "BAR" REGISTER BIT IS CLEAR
001104' 062700  000002          3$:     ADD     #2,R0           ;GO ON TO THE NEXT LINE
001110' 020027  000040                  CMP     R0,#40          ;HAVE WE DONE ALL LINES YET?
001114' 001310                          BNE     1$              ;IF NOT, GO DO ANOTHER
001116' 005767  177134                  TST     TEMP0           ;ALL DONE, ARE ANY LINES STILL ACTIVE?
001122' 001010                          BNE     4$              ;IF SO, GO START EM UP XMITING
001124' 012777  004000  177202          MOV     #4000,@X00      ;NO LINES REMAIN. STOP THE DH11
001132' 104403  000000' 017432'         MSG$,BEGIN,ERRNR        ;ASCII MESSAGE CALL WITH COMMON HEADER
001140' 104410  000000'                 END$,BEGIN              ;AND DROP THIS MODULE
001144' 016777  177106  177174  4$:     MOV     TEMP0,@X12      ;SET THE BAR REGISTER
001152' 000167  000004                  JMP     MAIN            ;GO WAIT FOR SOMTHING TO HAPPEN
001156' 004767  001060          NODROP: JSR     PC,SETUP        ;DO A BUNCH OF INITIALIZATION

                                ;THIS IS THE MAIN PROGRAM LOOP
                                ;NOTHING IS DONE HERE EXCEPT TO CONTINUOUSLY TEST TO SEE IF ENOUGH TIME HAS PASSED
                                ;FOR AN END OF PASS, CHECK FOR ERRORS, AND CALL THE
                                ;INTERRUPT SERVICE ROUTINE TO CLEAN OUT THE SILO IN CASE IT HAS STUFF
                                ;IN IT, BUT NOT ENOUGH STUFF TO CAUSE AN INTERRUPT

001162' 012777  000040  177162  MAIN:   MOV     #40,@X16        ;SET SILO TO INTERRUPT ON 40(OCTAL) CHARS
001166' 012706  176636                  MOV     SPOINT,SP       ;MAKE SURE THAT THE STACK POINTER IS RIGHT
001174' 012777  030100  177132          MOV     #030100,@X00    ;ENABLE INTERRUPTS
001202' 012746  000000          MLOOP:  MOV     #0,-(SP)        ;PUT FAKE PS ONTO STACK
001206' 012746  001216'                 MOV     #BRAKE,-(SP)    ;AND A FAKE PC
001212' 000167  000464                  JMP     INTSRV          ;PRETEND THAT WE JUST HAD AN INTERRUPT
001216'                         BRAKE:
                                BREAK$,BEGIN                    ;TEMPORARY RETURN TO MONITOR....
001220' 104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001226' 027767  177036  177036  TSTRE:  CMP     ERPT1,ERPT2     ;ANY RECEIVE ERRORS?
001234' 001430                          BEQ     1$              ;IF NOT GO CHECK FOR FATAL ERRORS
001236' 016701  177026                  MOV     ERPT1,R1        ;IF SO, MAKE THE ERROR QUEUE POINTER MORE ACCESSABLE
001242' 016767  177066  176630          MOV     X00,CSRA        ;SETUP THE CONTROL REG ADDRESS FOR PRINTING
```



```
001250' 012167  176626                  MOV     (R1)+,ACSR      ;SETUP THE CONTROL REG CONTENTS FOR PRINTING
001254' 012167  176624                  MOV     (R1)+,ASTAT     ;SETUP THE STATUS REG CONTENTS FOR PRINTING
001260' 104403  000000' 017464'         MSG$,BEGIN,ERRRE        ;ASCII MESSAGE CALL WITH COMMON HEADER
001266' 012767  000017  176612          MOV     #17,ERRTYP      ;RECEIVER ERROR
                                ;*****************************************************************
001274' 104405  000000' 000000          ERDER$,BEGIN,NULL       ;RECEIVE ERROR - NON FATAL
                                ;*****************************************************************
001302' 010167  176762                  MOV     R1,ERPT1        ;UPDATE THE ERROR QUEUE POINTER
001306' 026767  176756  176756          CMP     ERPT1,ERPT2     ;IS THE ERROR QUEUE EMPTY?
001314' 001006                          BNE     TSTFE           ;IF IT IS NOT EMPTY, IGNORE IT FOR A MOMENT
001316' 012767  016702' 176744  1$:     MOV     #ERQUEU,ERPT1   ;ERROR QUEUE IS EMPTY. REINITIALIZE THE
001324' 012767  016702' 176740          MOV     #ERQUEU,ERPT2   ;ERROR QUEUE POINTERS
001332' 005767  176730          TSTFE:  TST     FATERR          ;HAVE WE HAD ANY FATAL ERRORS?
001336' 001132                          BNE     ERRPT           ;IF SO, GO REPORT IT AND DROP THIS MODULE
001340' 005767  176704                  TST     DUMCNT          ;HAVE ALL ACTIVE LINES FINISHED RECIEVING?
001344' 001406                          BEQ     ALDONE          ;IF SO GO CLEAN UP AND END THIS PASS
001346' 005267  176656                  INC     WAIT1M          ;NO, TICK GOES THE WAIT TIMER
001352' 001513                          BNE     MLOOP           ;HAVE THESE DEVICES HAD ENOUGH TIME TO FINISH?
001354' 005367  176672                  DEC     WAIT1M          ;HAD ENOUGH TIME TO FINNISH
001360' 001310                          BNE     MLOOP           ;IF NOT.
001362' 042777  030100  176744  ALDONE: BIC     #30100,@X00     ;ENOUGH TIME! PREVENT FURTHER DH11 INTERRUPTS
001370' 005000                          CLR     R0              ;SETUP A LINE INDEX REGISTER TO 0
001372' 105760  000454'         1$:     TSTB    LQEC(0)         ;FIND OUT IF THE LINE IS SELECTED
001376' 001442                          BEQ     2$              ;IF NOT GO TEST FOR ERRORS ON IT
001400' 126027  000454' 000004          CMPB    LQEC(0),#4      ;IF IT IS CHECK THAT IT HAS FINISHED RECIEVING
001406' 001432                          BEQ     3$              ;IF IT DID AT LEAST 1 TRANSFER, DONT PRINT AN ERROR
001410' 012760  000000  000454'         MOV     #0,LQEC(0)      ;DIDNT FINISH. SET THE LINE TO DROPPED MODE
001416' 016767  176712  176454          MOV     X00,CSRA        ;SETUP FOR PRINTING THE ADRESS OF THE CONTROL/STATUS REG
001424' 016767  176704  176450          MOV     @X00,ACSR       ;SETUP FCR PRINTING THE CONTENTS OF THE CONTROL/STATUS R
001432' 016767  176620  176444          MOV     TEMP0,ASTAT     ;SETUP TO PRINT CONTENTS OF THE NEXT CHAR RECIEVED REC
001440' 010067  001070                  MOV     R0,NUMBA1       ;SAVE IT
001444' 006067  001064                  ROR     NUMBA1
                                ;*****************************************************************
                                        ;CONVERT NUMBA1 TO ASCII AND
001450' 104420  000000' 002534'         OTOA$,BEGIN,NUMBA1,MESNM0       ;STORE AT MESNM0
001456' 017405
                                ;*****************************************************************
001460' 104403  000000' 017440'         MSG$,BEGIN,ERRDR        ;ASCII MESSAGE CALL WITH COMMON HEADER
001466' 012767  000023  176412          MOV     #23,ERRTYP      ;FAILED TO INTERRUPT
                                ;*****************************************************************
001474' 104405  000000' 000000          ERDER$,BEGIN,NULL       ; A DH11 LINE HUNG
                                ;*****************************************************************
001502' 000434                          BR      3$              ;GO TRY THE NEXT LINE FOR ERRORS
001504' 105760  000455'         2$:     TSTB    LQEC+1(0)       ;ANY ERRORS ON THIS UNSELECTED LINE?
001510' 001431                          BEQ     3$              ;IF NOT TRY THE NEXT LINE
001512' 010046                          MOV     R0,-(SP)        ;PUT THE LINE # INDEX ONTO THE STACK
001514' 006016                          ROR     (SP)            ;DIVIDE BY 2 TO GET THE LINE #
001516' 005046                          CLR     -(SP)           ;CLEAR OUT THE FIRST WORD ON THE STACK
001520' 116027  000455'                 MOVB    LQEC+1(0),(SP)  ;SO THAT WE CAN PUT THE # OF ERRORS INTO LOW BYTE
001524' 011667  001006                  MOV     (SP),NUMBA2     ;SAVE
                                ;*****************************************************************
                                        ;CONVERT NUMBA2 TO ASCII AND
                                        ;STORE AT MESNM0
001530' 104420  000000' 002536'         OTOA$,BEGIN,NUMBA2,MESNM0
```

```
  525  001536' 017405
  526
  527  001540' 005726
  528  001542' 010046
  529  001544' 000241
  530  001546' 006016
  531  001550' 011667
  532
  533
  534
  535  001554' 104420
  536  001562' 017414
  537
  538  001564' 005726
  539  001566' 104403
  540  001574' 162700
  541  001600' 027700
  542  001604' 001272
  543  001606' 005267
  544  001612' 001402
  545  001614' 000167
  546  001620'
  547  001620' 104413
  548
  549
  550
  551
  552  001624' 016767
  553  001632' 017767
  554  001640' 017767
  555  001646' 012777
  556  001654' 104403
  557  001662' 012767
  558
  559  001670' 104405
  560
  561  001676' 104410
  562
  563
  564
  565
  566
  567
  568
  569
  570
  571
  572
  573
  574
  575
  576
  577
  578
  579
  580
```



```
  581
  582
  583
  584
  585
  586  001702' 032777
  587  001710' 001412
  588  001712' 012777
  589  001720' 017767
  590  001726' 012767
  591  001734' 000002
  592  001736'
  593
  594  001736' 000004
  595
  596  001744' 010067
  597  001750' 012767
  598  001756' 017767
  599  001764' 100077
  600  001766' 116767
  601  001774' 016700
  602  002000' 006100
  603  002002' 042700
  604  002006' 005760
  605  002012' 001011
  606  002014' 105260
  607  002020' 005367
  608  002024' 001354
  609  002026' 016700
  610  002032' 104400
  611  002036' 116770
  612  002044' 005260
  613  002050' 005760
  614  002054' 032767
  615  002062' 001425
  616  002064' 105260
  617  002070' 026727
  618  002076' 001003
  619  002100' 012767
  620  002106' 017777
  621  002114' 062767
  622  002122' 016777
  623  002130' 062767
  624  002136' 152767
  625  002144' 001417
  626  002146' 005367
  627  002152' 001301
  628  002154' 016700
  629  002160' 104400
  630  002164' 016700
  631  002170' 104400
  632
  633
  634  002174' 042777
  635  002202' 000002
  636
```



```
  749  017156' 042040  042111  047040  MESDH1: .ASCIZ  / DID NOT COMPLETE 1 TRANSFER IN TIME/
  750  017164' 052117  041440  046517
  751  017172' 046120  052105  020105
  752  017200' 020061  051124  047101
  753  017206' 043123  051105  044440
  754  017214' 020116  044524  042515
  755  017222'     000
  756  017223'     123  020117  042527  MESDH2: .ASCIZ  /SO WE ARE DROPING THE LINE/
  757  017230' 040440  042522  042040
  758  017236' 047522  044520  043516
  759  017244' 052040  042510  046040
  760  017252' 047111  000105
  761  017256' 042440  051122  051117  MESUS:  .ASCIZ  / ERRORS ON UNSELECTED LINE/
  762  017264' 020123  047117  052440
  763  017272' 051516  046105  041505
  764  017300' 042524  020104  044514
  765  017306' 042516     000
  766  017311'     040  044523  047514  MESSO:  .ASCIZ  / SILO OVERFLOW... ITS FATAL/
  767  017316' 047440  042526  043122
  768  017324' 047514  027127  027056
  769  017332' 044440  051524  043040
  770  017340' 052101  046101     000
  771  017345'     040  020101  042522  MESRE:  .ASCIZ  / A RECEIVER ERROR BIT WAS SET/
  772  017352' 042503  053117  051105
  773  017360' 042440  051122  051117
  774  017366' 041040  051111  053440
  775  017374' 051501  051440  052105
  776  017402'     000
  777  017403'     045      000  MESBE:  .ASCIZ  /%/
  778  017405'     060  030060  030060  MESNM0: .ASCIZ  /000000/
  779  017412' 000060
  780  017414' 030060  030060  030060  MESNM1: .ASCIZ  /000000/
  781  017422'     000
  782          017424'                 .EVEN                   ;IN CASE THERE IS AN ODD # OF BYTES IN THE A
  783
  784                                  ;TABLE OF ERROR MESSAGE POINTERS & TERMINATORS (COMMENTED!)
  785  017424' 017403'         ERRMS:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
  786  017426' 017433'                 MESES                   ;POINTS TO AN ASCIZ ERROR MESSAGE
  787  017430' 177777                 177777                  ;TERMINATOR. ONLY 1 MESSAGE THIS TIME
  788  017432' 017403'         ERRNR:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
  789  017434' 017012'                 MESNR                   ;POINTS TO "NO LINES REMAIN ACTIVE" MESSAGE
  790  017436' 177777                 177777                  ;TERMINATE MESSAGE
  791  017440' 017403'         ERRDH:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
  792  017442' 017147'                 MESDH                   ;POINTS TO 1ST PART OF "RUNG" ERROR MESSAGE
  793  017444' 017405'                 MESNM0                  ;POINTS TO LINE # ASCIZ
  794  017446' 017156'                 MESDH1                  ;POINTS TO PART OF MESSAGE
  795  017450' 017403'                 MESBE                   ;RETURN
  796  017452' 017223'                 MESDH2                  ;DROPING A LINE MESSAGE
  797  017454' 177777                 177777                  ;END OF MESSAGE
  798  017456' 017403'         ERRSO:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
  799  017460' 017311'                 MESSO                   ;POINTS TO ASCIZ TEXT OF FATAL ERROR MESSAGE
  800  017462' 177777                 177777                  ;THIS INDICATES THAT NO MORE MESSAGES FOLLOW
  801  017464' 017403'         ERRRE:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
  802  017466' 017345'                 MESRE                   ;POINTS TO ASCIZ TEXT OF THE RECEIVER ERROR
  803  017470' 177777                 177777                  ;TERMINATOR
  804  017472' 017403'         ERRUS:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
```



```
  805  017474' 017405'                 MESNM0          ;POINTS TO ASCIZ FOR # OF ERRORS
  806  017476' 017256'                 MESUS           ;POINTS TO TEXT PART OF MESSAGE
  807  017500' 017414'                 MESNM1          ;POINTS TO ASCIZ FOR LINE #
  808  017502' 177777                 177777          ;SIGNALS END OF MESSAGE TO DECX/11
  809
  810          000001                 ;END OF MESSAGE TABLE
                                      .END
```

```
                                ;THIS ROUTINE IS ENTERED WHEN A LINE FINISHES RECIEVING
002204' 112760  000004  000454' XMIT:   MOVB    #4,LOEC(0)      ;SET LINE TO "RECIEVE DONE" MODE
002212' 166060  000514' 000354'         SUB     LOCC(0),LOSW(0) ;RESET THE LINES BUFFER POINTER
002220' 005267  176024                  INC     DUNCNT          ;ALSO TALLY UP 1 MORE FINISHED DEVICE
002224' 005367  176034                  DEC     INTSW           ;20 CHARS PULLED YET?
002230' 001352                          BNE     YYYY            ;IF NOT GO BACK AND DO ANOTHER
002232' 016700  175624                  MOV     SVR0,R0         ;YES, DONE 4 ALREADY. RESTORE R0
002236' 104400  000000'                 EXITS,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.

                                ;THIS IS THE INITIALIZATION ROUTINE.
                                ;FIRST IT CLEARS OUT THE FATAL ERROR SWITCH
                                ;THEN IT SETS UP POINTERS TO ALL OF THE DH11 REGISTERS
                                ;THEN IT ACCESSES EACH DH11 REGISTER WITH A 'TST' INSTRUCTION TO
                                ;MAKE SURE IT IS REALLY THERE.
                                ;THEN IT ACCESSES EACH DH11 REGISTER USING A 'TST' INSTRUCTION SO THAT IF
                                ;THE REGISTER DOES NOT EXIST OR HAS MAJOR PROBLEMS, WE FIND OUT RIGHT AWAY
                                ;THEN IT SETS UP DH11 VECTORS
                                ;THEN IT INITIALIZES THE ERROR QUEUE POINTERS
                                ;THEN IT ASSIGNS BUFFER POINTERS TO ALL SELECTED LINES
002242' 005067  176020  SETUP:  CLR     FATERR          ;INIT FATAL ERROR SWITCH = NO FATAL ERRORS
002246' 016700  175534          MOV     ADDR,R0         ;MAKE THE DEVICE REGISTER ADDRESS EASY TO ACCESS
002252' 010067  176056          MOV     R0,X00          ;SETUP ADDRESS OF THE SYSTEM CONTROL REGISTER
002256' 005720                  TST     (R0)+           ;TRAP IF IT DOES NOT EXIST
002260' 010067  176052          MOV     R0,X02          ;SETUP ADDRESS OF NEXT CHARACTER RECEIVED REGISTER
002264' 005720                  TST     (R0)+           ;TRAP IF IT DOES NOT EXIST
002266' 010067  176046          MOV     R0,X04          ;SETUP ADDRESS OF LINE PARAMETER REGISTER
002272' 005720                  TST     (R0)+           ;TRAP IF IT DOES NOT EXIST
002274' 010067  176042          MOV     R0,X06          ;SETUP ADDRESS OF CURRENT ADDRESS REGISTER
002300' 005720                  TST     (R0)+           ;TRAP IF IT DOES NOT EXIST
002302' 010067  176036          MOV     R0,X10          ;SETUP ADDRESS OF BYTE COUNT REGISTER
002306' 005720                  TST     (R0)+           ;TRAP IF IT DOES NOT EXIST
002310' 010067  176032          MOV     R0,X12          ;SETUP ADDRESS OF BUFFER ACTIVE REGISTER
002314' 005720                  TST     (R0)+           ;TRAP IF IT DOES NOT EXIST
002316' 010067  176026          MOV     R0,X14          ;SETUP ADDRESS OF BREAK CONTROL REGISTER
002322' 005720                  TST     (R0)+           ;TRAP IF IT DOES NOT EXIST
002324' 010067  176022          MOV     R0,X16          ;SETUP THE ADDRESS OF THE SILO STATUS REGISTER
002330' 005720                  TST     (R0)+           ;TRAP IF IT DOES NOT EXIST
                                ;NOW SETUP DH11 INTERRUPT VECTORS. THERE ARE 2 OF THEM, ONE
                                ;FOR RECEIVER INTERRUPTS, AND 1 FOR TRANSMITER INTERRUPTS
002332' 016700  175452          MOV     VECTOR,R0       ;GET THE VECTOR ADDRESS
002336' 012720  001702'         MOV     #INTSER,(R0)+   ;POINT VECTOR TO RECEIVER INTERRUPT SERVICE
002342' 116720  175444          MOVB    BR1,(R0)+       ;SET PRIORITY LEVEL FOR THE INTERRUPT ROUTINE
002346' 105020                  CLRB    (R0)+           ;SET R0 TO AN EVEN ADDRESS
002350' 012720  002174'         MOV     #XINT,(R0)+     ;POINT VECTOR AT TRANSMITER SERVICE ROUTINE
002354' 116720  175432          MOVB    BR1,(R0)+       ;PRIORITY LEVEL FOR THE TRANSMITER INTERRUPT RCUTINE
002360' 105020                  CLRB    (R0)+           ;SET R0 TO AN EVEN ADDRESS
                                ;NOW SETUP ERROR QUEUE POINTERS
002362' 012767  016702' 175700          MOV     #ERQUEU,ERPT1   ;SET THE "WHERE WE'VE SERVICED TO" ERROR QUEUE
                                                                ;POINTER TO THE BEGINING OF THE QUEUE AREA
002370' 012767  016702' 175674          MOV     #ERQUEU,ERPT2   ;SET THE "WHERE WE'VE PUT STUFF" ERROR QUEUE
                                                                ;POINTER TO THE BEGINING OF THE QUEUE AREA
                                ;NOW ASSIGN BUFFER POINTERS TO SELECTED LINES (DONE ONLY ONCE AT THE BEGINING)
002376' 005067  175622          CLR     SELCNT          ;INIT DEVICE SELECT COUNT TO 0 BEFORE COUNTING VT20'S
002402' 005067  175642          CLR     DUNCNT          ;AND OF COURSE NONE HAVE FINISHED YET
002406' 016701  175402          MOV     DVID1,R1        ;MAKE BITMAP OF SELECTED VT20 LINES MORE ACCESSABLE
002412' 005000                  CLR     R0              ;INIT FOR 16 POSSIBLE LINES
```



```
002414' 005067  175636          CLR     TEMPQ           ;INIT THE LINE #
002420' 005060  000454'  1$:    CLR     LOEC(0)         ;ASSUME LINE IS UNSELECTED UNTIL PROVEN OTHERWISE
002424' 006001                  ROR     R1              ;PUT A BIT INTO THE C-BIT
002426' 103031                  BCC     2$              ;IS THE LINE TO BE USED ?
002430' 016067  000274' 000354'         MOV     BUFLIST(0),LOSW(0)      ;YES, ASSIGN A BUFFER POINTER TO THAT LINE
002436' 042777  000017  175670          BIC     #17,@#X00       ;CLEAR OUT LINE SELECTION BITS
002444' 005367  175600                  DEC     DUNCNT          ;ADD 1 TO COUNT OF ACTIVE DEVICES
002450' 000240                  NOP
002452' 156777  175600  175654          BISB    TEMPQ,@X00      ;SET LINE SELECTION BITS FOR THIS LINE
002460' 016077  000414' 175652          MOV     LOBR(0),@X04    ;SETUP BAUD RATES FOR THIS LINE
002466' 052777  000023  175644          BIS     #23,@#X04       ;PARITY-FULL DUPLEX-8 BIT
002474' 005060  000514'                 CLR     LOCC(0)         ;CLEAR OUT THIS LINES CHARACTER COUNT
002500' 012760  000003  000454'         MOV     #3,LOEC(0)      ;START OUT THE LINE IN RECIEVE MODE
002506' 005267  175512                  INC     SELCNT          ;UPDATE COUNT OF SELLECTED VT2C'S
002512' 062700  000002          2$:     ADD     #2,R0           ;GO ON TO THE NEXT LINE
002516' 005267  175534                  INC     TEMPQ           ;NEXT LINE
002522' 026727  175530  000017          CMP     TEMPQ,#17       ;CHECKED ALL LINES FOR SELECTION YET ?
002530' 001333                  BNE     1$              ;IF NOT, GO BACK AND CHECK THE NEXT ONE
002532' 000207                  RTS     PC              ;RETURN


002534' 000000          NUMBA1: .WORD   0
002536' 000000          NUMBA2: .WORD   0
002540' 000000          NUMBA3: .WORD   0

002542' 000303          VTBF0:  .BLKW   195.            ;BUFFER SPACE FOR 1ST SELLECTED DH11 LINE
003350' 000303          VTBF1:  .BLKW   195.            ;BUFFER SPACE FOR 2ND SELECTED DH11 LINE
004156' 000303          VTBF2:  .BLKW   195.            ;BUFFER SPACE FOR 3RD SELLECTED DH11 LINE
004764' 000303          VTBF3:  .BLKW   195.            ;BUFFER SPACE FOR 4TH SELLECTED DH11 LINE
005572' 000303          VTBF4:  .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
006400' 000303          VTBF5:  .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
007206' 000303          VTBF6:  .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
010014' 000303          VTBF7:  .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
010622' 000303          VTBF10: .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
011430' 000303          VTBF11: .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
012236' 000303          VTBF12: .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
013044' 000303          VTBF13: .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
013652' 000303          VTBF14: .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
014460' 000303          VTBF15: .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
015266' 000303          VTBF16: .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
016074' 000303          VTBF17: .BLKW   195.            ;BUFFER SPACE FOR A SELLECTED DH11 LINE
016702' 000074          ERQUEU: .BLKW   60.             ;RECEIVER ERROR QUEUE SPACE
017072' 000000          ERQOFL: .WORD   0               ;ERROR QUEUE OVERFLOW BOUNDARY


                                ;FOLLOWING IS THE TEXT FOR ALL ASCII ERROR MESSAGES
017074' 047516  046040  047111  MESNR:  .ASCIZ  /NO LINES REMAIN ACTIVE/
017102' 051505  051040  046505
017110' 044501  020116  041501
017116' 044524  042526  000
017123'     040 047516  053040  MESWS:  .ASCIZ  / NO VT20'S SELECTED/
017130' 031124  023460  020123
017136' 042524  042514  052103
017144' 042105  000
017147'     040 044514  042516  MESDH:  .ASCIZ  / LINE /
017154' 000040
```

```
   749  017156' 042040  042111  047040  MESDH1: .ASCIZ  / DID NOT COMPLETE 1 TRANSFER IN TIME/
   750  017164' 052117  041440  046517
   751  017172' 046120  052105  020105
   752  017200' 020061  051124  047101
   753  017206' 043123  051105  044440
   754  017214' 020116  044524  042515
   755  017222'     0  000
   756  017223'   123  020117  042527  MESDH2: .ASCIZ  /SO WE ARE DROPING THE LINE/
   757  017230' 040440  042522  042040
   758  017236' 047522  044520  043516
   759  017244' 052040  042510  046040
   760  017252' 047111  000105
   761  017256' 042440  051124  051117  MESUS:  .ASCIZ  / ERRORS ON UNSELECTED LINE/
   762  017264' 020123  047105  052440
   763  017272' 051516  046105  041505
   764  017300' 042524  020104  044514
   765  017306' 042516      0  000
   766  017311'   040  044523  047514  MESSO:  .ASCIZ  / SILO OVERFLOW... ITS FATAL/
   767  017316' 047440  042526  043122
   768  017324' 047514  051127  042456
   769  017332' 044440  051524  043040
   770  017340' 052101  046101      0  000
   771  017345'   040  020101  042522  MESRE:  .ASCIZ  / A RECEIVER ERROR BIT WAS SET/
   772  017352' 042503  053111  051105
   773  017360' 042440  051122  051117
   774  017366' 041040  052111  053440
   775  017374' 051501  051440  052105
   776  017402'     0  000
   777  017403'    45      0  000  MESBE:  .ASCIZ  /%/
   778  017405'    60  030060  030060  MESNM0: .ASCIZ  /000000/
   779  017412' 000060
   780  017414' 030060  030060  030060  MESNM1: .ASCIZ  /000000/
   781  017422'     0  000
   782          017424'                        .EVEN                   ;IN CASE THERE IS AN ODD # OF BYTES IN THE ABOVE MESSAGE
   783
   784                                 ;TABLE OF ERROR MESSAGE POINTERS & TERMINATORS (COMMENTED!)
   785  017424' 017403'                ERRMS:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
   786  017426' 017443'                        MESMS                   ;POINTS TO AN ASCIZ ERROR MESSAGE
   787  017430' 177777'                        177777                  ;TERMINATOR, ONLY 1 MESSAGE THIS TIME
   788  017432' 017403'                ERRNR:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
   789  017434' 017074'                        MESNR                   ;POINTS TO "NO LINES REMAIN ACTIVE" MESSAGE
   790  017436' 177777'                        177777                  ;TERMINATE MESSAGE
   791  017440' 017403'                ERRDH:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
   792  017442' 017147'                        MESDH                   ;POINTS TO 1ST PART OF "HUNG" ERROR MESSAGE ASCIZ
   793  017444' 017405'                        MESNM0                  ;POINTS TO LINE # ASCIZ
   794  017446' 017156'                        MESDH1                  ;POINTS TO PART OF MESSAGE
   795  017450' 017403'                        MESBE                   ;RETURN
   796  017452' 017233'                        MESDH2                  ;DROPING A LINE MESSAGE
   797  017454' 177777'                        177777                  ;END OF MESSAGE
   798  017456' 017403'                ERRSO:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
   799  017460' 017311'                        MESSO                   ;POINTS TO ASCIZ TEXT OF FATAL ERROR MESSAGE
   800  017462' 177777'                        177777                  ;THIS INDICATES THAT NO MORE MESSAGES FOLLOW
   801  017464' 017403'                ERRRE:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
   802  017466' 017345'                        MESRE                   ;POINTS TO ASCIZ TEXT OF THE RECEIVER ERROR MESSAGE
   803  017470' 177777'                        177777                  ;TERMINATOR
   804  017472' 017403'                ERRUS:  MESBE                   ;POINTS TO A CARRIAGE RETURN-LINE FEED
```



```
   805  017474' 017405'                        MESNM0          ;POINTS TO ASCIZ FOR # OF ERRORS
   806  017476' 017256'                        MESUS           ;POINTS TO TEXT PART OF MESSAGE
   807  017500' 017412'                        MESNM1          ;POINTS TO ASCIZ FOR LINE #
   808  017502' 177777                        177777          ;SIGNALS END OF MESSAGE TO DECX/11
   809
   810          000001                 ;END OF MESSAGE TABLE
                                       .END
```

```
ACSR     000102R          233#  469*  497*  553*
ADDR     000006R          199#  390*  397   657
ADDR22=  001000           251#
ALDONE   001362R          484   489#
ASB      000106R          237#
ASTAT    000104R          235#  470*  498*  554*
AWAS     000110R          238#
BEGIN    000000R          196#  399   400   439   444   445   463   464   471   474   504   507   510
                          524   535   539   547   556   559   561   594   610   629   631   644
BIT0   =  000001           251#
BIT1   =  000002           251#
BIT10  =  002000           251#
BIT11  =  004000           251#
BIT12  =  010000           251#
BIT13  =  020000           251#
BIT14  =  040000           251#
BIT15  =  100000           251#
BIT2   =  000004           251#
BIT3   =  000010           251#
BIT4   =  000020           251#
BIT5   =  000040           251#
BIT6   =  000100           251#
BIT7   =  000200           251#
BIT8   =  000400           251#
BIT9   =  001000           251#
BPLIST   000274R          275#  697
BRAKE    001216R          460   462#
BREAK$=  104407           251#  463   464
BR1      000012R          201#  678   681
BR2      000013R          202#
BTOD$  = 104421           251#
CDATA$=  104412           251#
CONFIG   000056R          221#
COUNT1   000262R          267#
CSRA     000100R          231#  468*  496*  552*
DATCK$=  104411           251#
DATER$=  104404           251#
DROP     000624R          397#
DUNCNT   000250R          262#  415*  421*  483   640*  690*  699*
DVID1    000014R          203#  393   691
ENDIT$=  104413           251#  547
END$   = 104410           251#  400   445   561
ERPT1    000270R          272#  465   467   476*  477   479*  684*
ERPT2    000272R          273#  465   477   480*  617   619*  620*  621*  622*  623*  686*
ERQUEL   017072R          617   736#
ERQUEU   016702R          479   480#  619   684   686   735#
ERRDH    017440R          507   791#
ERRHW    017432R          444   788#
ERRNS    017424R          399   785#
ERRPT    001624R          482   552#
ERRRE    017464R          471   801#
ERRSO    017456R          556   798#
ERRTYP   000106R          236#  472#  508*  557*
ERRUS    017472R          539   804#
EXIT$  = 104400           251#  610*  629   631   644
EXTAD    000246R          261#  431*  432
```



```
FATERR   000266R          271#  481   589*  656*
GETPA$=  104415           251#  430
GWBUF$=  104414           251#
HRDCNT   000044R          216#
HRDERS=  104405           251#  474   510   559
HRDPAS   000050R          218#
ICONT    000036R          213#
ICOUNT   000040R          214#
IDNUM    000122R          243#
INIT     000030R          210#
INTR     000120R          242#  387*
INTSRV   001702R          461   586#  677
INTSW    000264R          269#  590*  597*  607*  626*  641*
LOBR     000414R          321#  404   702
LOCC     000514R          365#  426   434*  613*  639   704*
LOEC     000454R          346#  416   418*  491   493   495*  514   519   604   606*  616*  638*  694*
                          709#
LOSW     000354R          302#  429   611*  612*  639*  697*
L1BR     000416R          322#
L1CC     000516R          366#
L1EC     000456R          347#
L1SW     000356R          303#
L10BR    000434R          329#
L10CC    000534R          373#
L10EC    000474R          354#
L10SW    000374R          310#
L11BR    000436R          330#
L11CC    000536R          374#
L11EC    000476R          355#
L11SW    000376R          311#
L12BR    000440R          331#
L12CC    000540R          375#
L12EC    000500R          356#
L12SW    000400R          312#
L13BR    000442R          332#
L13CC    000542R          376#
L13EC    000502R          357#
L13SW    000402R          313#
L14BR    000444R          333#
L14CC    000544R          377#
L14EC    000504R          358#
L14SW    000404R          314#
L15BR    000446R          334#
L15CC    000546R          376#
L15EC    000506R          359#
L15SW    000406R          315#
L16BR    000450R          335#
L16CC    000550R          379#
L16EC    000510R          360#
L16SW    000410R          316#
L17BR    000452R          336#
L17CC    000552R          380#
L17EC    000512R          361#
L17SW    000412R          317#
L2BR     000420R          323#
L2CC     000520R          367#
```

```
L2EC    000460R  348#
L2SW    000360R  304#
L3BR    000422R  324#
L3CC    000522R  368#
L3EC    000462R  349#
L3SW    000362R  305#
L4BR    000424R  325#
L4CC    000524R  369#
L4EC    000464R  350#
L4SW    000364R  306#
L5BR    000426R  326#
L5CC    000526R  370#
L5EC    000466R  351#
L5SW    000366R  307#
L6BR    000430R  327#
L6CC    000530R  371#
L6EC    000470R  352#
L6SW    000370R  308#
L7BR    000432R  328#
L7CC    000532R  372#
L7EC    000472R  353#
L7SW    000372R  309#
MAIN    001162R  447    456#
MAP22$= 104416   251#
MESBE   017403R  777#   785     788     791     795     798     801     804
MESDH   017147R  747#   782
MESDH1  017156R  749#   784
MESDH2  017223R  756#   796
MESHW   017405R  504     524     778#    793     805
MESHW1  017414R  535     780#    807
MESNR   017074R  739#   789
MESNS   017123R  743#   786
MESRE   017345R  771#   802
MESSO   017311R  766#   799
MESUS   017256R  761#   806
MLOOP   001202R  459#   486     488
MODNAM  000000R  197#
MODSP   000224R  211     249#
MSGW$ = 104403   251#   399     444     471     507     539     556
MSGS$ = 104402   251#
MSG$  = 104401   251#
MODROP  001156R  394#   448#
NOERR   002136R  615     624#
NULL  = 000000   251#   474#
NUMBA1  002534R  496#   500*    510     559
NUMBA2  002536R  520*    524     715#
NUMBA3  002540R  531*    535     717#
NXTPER  000716R  412#   545
OPE#  = 000000   199     204     205     206     207     224     225     226     227     228     229     230     231
                 235     236     237     238     240     241     242     251#
OTOA$ = 104420   251#   504     524     535
PASCNT  000034R  212#
PHYSAD  000244R  260#   433
PIRQ$ = 000004   251#   594
POPSP = 005726   251#
POPSP2= 022626   251#
```



```
PRTY  = 000000   202     251#
PRTY0 = 000000   251#
PRTY1 = 000040   251#
PRTY2 = 000100   251#
PRTY3 = 000140   251#
PRTY4 = 000200   251#
PRTY5 = 000240   201     251#
PRTY6 = 000300   251#
PRTY7 = 000340   251#
PS    = 177776   251#
PSW   = 177776   251#
PUSH    005746   251#
PUSH2   024646   251#
RAND$ = 104417   251#
RANNUM  000054R  220#
RCVMOD  002036R  605     611#
RELOC   000232R  255#
RESTRT  000652R  539     404#
RES1    000056R  222#
RES2    000060R  223#
RETRN   002164R  588     639#
RINT    001736R  587     592#
RSTRT   000112R  239#
SAVR0   000226R  253#
SBADR   000102R  232#
SELCNT  000224R  252#   689*    706*
SETUP   002242R  448     656#
SOFCNT  000042R  215#
SOFFAS= 104406   251#
SOFFAS  000040R  214#
SPOINT  000032R  211     392     457
SPSIZ = 000040   1       244
SR1     000016R  204#
SR2     000020R  205#
SR3     000022R  206#
SR4     000024R  207#
START   000548R  210     387#
STAT    000026R  209#
SVR0    000062R  224#   596*    609     628     630     643
SVR1    000064R  225#
SVR2    000066R  226#
SVR3    000070R  227#
SVR4    000072R  228#
SVR5    000074R  229#
SVR6    000076R  230#
SVT0    000234R  256#
SVT1    000236R  257#
SYSCNT  000052R  219#
TEMP    000254R  264#   422*    423*    434*    435     436*    437*    438
TEMP0   000256R  265#   432*    439*    441     446     498     596*    600     611     622     624     693*
                 701     708*    709
TEMP1   000260R  266#   600*    601     614
TRPDFD= 000022   251#
TSTFE   001332R  478     481#
TSTRE   001226R  465#
VECTOR  000010R  200#   676
```

```
VIRTAD  000242R          259#    429*    430
VTBF0   002542R          275     719#
VTBF1   003350R          276     720#
VTBF10  010622R          283     727#
VTBF11  011430R          284     728#
VTBF12  012236R          285     729#
VTBF13  013044R          286     730#
VTBF14  013652R          287     731#
VTBF15  014460R          288     732#
VTBF16  015266R          289     733#
VTBF17  016074R          290     734#
VTBF2   004156R          277     721#
VTBF3   004764R          278     722#
VTBF4   005572R          279     723#
VTBF5   006400R          280     724#
VTBF6   007206R          281     725#
VTBF7   010014R          282     726#
WAITIM  000230R          252#    485*
WAITIN  000252R          263#    391*    413*    487*
WASADR  000104R          234#
WDFR    000116R          241#    389*
WDTQ    000114R          240#    388*
XCNT    000240R          258#    404*    405*    406*    407*    408*    409*    410*    411*    543*
XFLAG   000005R          198#
XINT    002174R          634#    680
XMIT    002204R          636     638#
XXXX    001744R          593     596#
X00     000334R          292#    397*    398*    425*    432*    443*    458*    466*    489*    496     497     552     553
                         555*    586     588*    620     634*    658*    698*    701*
X02     000336R          293#    598     660*
X04     000340R          294#    662*    703*
X06     000342R          295#    433*    664*
X10     000344R          296#    426*    666*
X12     000346R          297#    446*    668*
X14     000350R          298#    670*
X16     000352R          299#    456*    554     672*
YYYY    001756R          598#    608     627     645
.       = 017504R        719#    720#    721#    722#    723#    724#    725#    726#    727#    728#    729#    730#    731#
                         732#    733#    734#    735#    782#

. ABS.  000000      000
        017504      001

ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

XVTBB0,XVTBB0/SQL/CRF:SYM=DDXCOM,XVTBB0
RUN-TIME:  1  2  .4 SECONDS
RUN-TIME RATIO:  20/5=4.0
CORE USED:  7K  (13 PAGES)
```

DEC/X11 SYSTEM EXERSIZER MACRO DEFINITION MODULE

IDENTIFICATION

PROGRAM CODE: AC-T375A-MC

PROGRAM TITLE: CXKMDA0 KMV11A/B DEC-X11 MOD

PRODUCT DATE: JAN 83

AUTHOR: G MICHELET

MAINTAINER: COMPUTER SPECIAL SYSTEMS,
DIGITAL EQUIPMENT CO.
Z.A.E LES GLAISINS
ANNECY
FRANCE

```
000000                          .REPT 0
                              .ENABL LC
                                  DEC/X11 MODULE SPECIFICATION
```

```
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
```

PROGRAM HISTORY:

VO.	AUTHOR	DATE	PURPOSE
01	C.P TAYLOR	1-SEP-1982	ORIGINAL
02	G MICHELET	27-SEP-1982	
02	G MICHELET	10-NOV-82	MODIFICATION FOR 11/23A COMPATIBILITY

1. ABSTRACT

TST IS AN IOMOD THAT EXERCISES THE KMV11-A OR KMV11-B, IN


```
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
```

INTERNAL DMA MODE. THE MODULE GENERATES AND CHECKS THE DATA SENT AND RECEIVED AS WELL AS MONITORING THE STATUS OF THE KMV11 CONTROLLER.

2. REQUIREMENTS

2.1 HARDWARE

KMV11-A OR KMV11-B PROGRAMMABLE COMMUNICATIONS CONTROLLER.

2.2 STORAGE

TST REQUIRES A MINIMUM OF <> WORDS OF STORAGE.

3. PASS DEFINITION

A PASS OF TST IS COMPLETE WHEN ALL OF THE KMV11 INTERNAL MEMORY HAS BEEN USED TO HOLD THE DMA DATA.

4. EXECUTION TIME

<>

5. CONFIGURATION REQUIREMENTS

5.1 DEFAULT PARAMETERS

DVADR:	177000
VECTOR:	<NOT USED>
BR1:	<NOT USED>
BR2:	<NOT USED>

5.2 REQUIRED PARAMETERS

NONE.

6. DEVICE/OPTION SETUP

THE KMV MODULE MUST BE SET UP WITH THE ENDLESS LOOPING OF SELF TEST DISABLED, AND WITH THE ABILITY TO RUN THE INTERNAL TESTS.

7. MODULE OPERATION

A. SET UP THE DEVICE ADDRESS AND ITS PHYSICAL 22 BIT ADDRESS OR TO ITS 18 BIT ADDRESS ACORDING TO THE MONITOR USED.
B. CLEAR ALL THE ADDRESSES AND CHECK THAT THEY CLEARED
C. SET THE INTERFACE INTO MAINTENANCE MODE
D. CLEAR OUT RECEIVER DATA AREA AND SET UP TRANSMIT DATA
E. LOAD THE DEVICE REGISTERS WITH THE BUS ADDRESSES AND WORD COUNTS FOR THE TRANSFER
F. SET THE TEST NUMBER TO ACTIVATE INTO THE CSR REGISTER
G. WAIT FOR THE TEST TO COMPLETE

```
                                        1.      CHECK THE RESULTS OF THE OPERATION
                                        J.      IF THERE WERE ANY ERRORS, REPORT THEM AND GO TO M
                                        K.      IF THERE WERE NO ERRORS, CHECK THE DATA SENT AGAINST
                                                THAT RECEIVED.
                                        L.      IF THERE WERE ERRORS, REPORT THE FIRST TEN ONLY, AND
                                                ABANDON THE CHECKING IF THERE ARE 10 OR MORE ERRORS
                                        M.      SELECT THE NEXT BLOCK OF KMV MEMORY TO USE
                                        N.      IF THERE IS MORE MEMORY TO TEST GOTO C.
                                        O.      IF ALL THE MEMORY HAS BEEN USED SIGNAL THE END OF PASS
                                                AND RESTART AT B

                                        8.          OPERATION OPTIONS

                                        NONE.

                                        9.          NON STANDARD PRINTOUT

                                        10.         DECX11 MONITOR
                                                TO RUN KMV11 DECX11 MODULE ON PDP11/23A ,MONITOR C MUST BE USE.
                                                TO RUN KMV11 DECX11 MODULE ON PDP11/23B ,MONITOR Q MUST BE USE.


                                        NONE.
                                        .ENDR
```



```
140 000000                              IUMOD   <KMDA >,177000,300,4,4,1,1,0
    000000                              MODULE  140000,KMDA ,177000,300,4,4,1,1,0,,,,,
                                        .TITLE  KMDA DEC/X11 SYSTEM EXERCISER MODULE
                                ;       DDXCOM  VERSION  6.4     28-JAN-82
                                        .LIST   BIN
                                ;*****************************************************************************
    000000                              BEGIN:
    000000      113     115     104     MODNAM: .ASCII  /KMDA / ;MODULE NAME.
    000003      101     040
    000005      000                     XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF #BUFF USAGE
    000006  177000                      ADDR:   177000+0                ;1ST DEVICE ADDR.
    000010  000300                      VECTOR: 300+0                   ;1ST DEVICE VECTOR.
    000012      200                     BR1:    .BYTE   PRTY4+0         ;1ST BR LEVEL.
    000013      200                     BR2:    .BYTE   PRTY4+0         ;2ND BR LEVEL.
    000014  000002                      DVID1:  1+1                     ;DEVICE INDICATOR 1.
    000016  000000                      SR1:    OPEN                    ;SWITCH REGISTER 1
    000020  000000                      SR2:    OPEN                    ;SWITCH REGISTER 2
    000022  000000                      SR3:    OPEN                    ;SWITCH REGISTER 3
    000024  000000                      SR4:    OPEN                    ;SWITCH REGISTER 4
                                        ;*****************************************************************************
    000026  140000                      STAT:   140000                  ;STATUS WORD.
    000030  000224'                     INIT:   START                   ;MODULE START ADDR.
    000032  000224'                     SPOINT: MODSP                   ;MODULE STACK POINTER.
    000034  000000                      PASCNT: 0                       ;PASS COUNTER.
    000036  000001                      ICONT:  1                       ;# OF ITERATIONS PER PASS=1
    000040  000000                      ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
    000042  000000                      SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
    000044  000000                      HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
    000046  000000                      SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
    000050  000000                      HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
    000052  000000                      SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
    000054  000000                      RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
    000056                              CONFIG:                         ;RESERVED FOR MONITOR USE
    000056  000000                      RES1:   0                       ;RESERVED FOR MONITOR USE
    000060  000000                      RES2:   0                       ;RESERVED FOR MONITOR USE
    000062  000000                      SVR0:   OPEN                    ;LOC TO SAVE R0.
    000064  000000                      SVR1:   OPEN                    ;LOC TO SAVE R1.
    000066  000000                      SVR2:   OPEN                    ;LOC TO SAVE R2.
    000070  000000                      SVR3:   OPEN                    ;LOC TO SAVE R3.
    000072  000000                      SVR4:   OPEN                    ;LOC TO SAVE R4.
    000074  000000                      SVR5:   OPEN                    ;LOC TO SAVE R5.
    000076  000000                      SVR6:   OPEN                    ;LOC TO SAVE R6.
    000100  000000                      CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
    000102                              SBADR:                          ;ADDR OF GOOD DATA, OR
    000102  000000                      ACSR:   OPEN                    ;CONTENTS OF CSR.
    000104                              WASADR:                         ;ADDR OF BAD DATA, OR
    000104  000000                      ASTAT:  OPEN                    ;STATUS REG CONTENTS.
    000106                              ERRTYP:                         ;TYPE OF ERROR
    000106  000000                      ASB:    OPEN                    ;EXPECTED DATA.
    000110  000000                      AWAS:   OPEN                    ;ACTUAL DATA.
    000112  000256'                     RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
    000114  000000                      WDTO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
    000116  000000                      WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
    000120  000000                      INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
    000122  000000                      IDNUM:  0                       ;MODULE IDENTIFICATION NUMBER=0
            000040                              .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                        .NLIST
```

DEC/X11 SYSTEM EXERSIZER MACRO DEFINITION MODULE

```
                                        .WORD   0
                                        .LIST
                                        .ENDR
    000224                      MODSP:
                                ;*************************************************************************
    141         054000                  MAINT0=54000
    142         044000                  MAINT1= 44000
    143 000224                  START:
    144 000224  012767  065000  001454          MOV     #65000,MAXCNT           ;RAM MEMORY MAX LENGTH
    145 000232  012702  001656'                 MOV     #KMVCSP,R2              ;POINT TO CSR ADDRESS TABLE
    146 000236  016700  177544                  MOV     ADDR,R0                 ; GET THE FIRST ADDRESS
    147 000242  012701  000010                  MOV     #8.,R1                  ;LOOP FOR ALL 8 ADDRESSES
    148 000246                  1$:
    149 000246  010022                          MOV     R0,(R2)+
    150 000250  005720                          TST     (R0)+                   ;CHECK FOR VALID ADDRESS
    151 000252  005301                          DEC     R1
    152 000254  001374                          BNE     1$
    153 000256                  RESTRT:
    154
    155
    156 000256  016767  177576  011454          MOV     RES2,@MON               ;SAVE CONFIGURATION WORD 2
    157
    158 000264  005067  001412                  CLR     FLAG                    ; RESET THE ERROR FLAG
    159 000270  012767  001740' 001416          MOV     #TTABLE,TVAD            ; GET VIRTUAL ADDRESS FOR TX
    160 000276  104415  000000' 001714'         GETPA$,BEGIN, TVAD              ;GET PHYSICAL ADDRESS FROM 16-BIT TVAD
    161
    162 000304  012767  005740' 001414          MOV     #RTABLE,RVAD            ; GET VIRTUAL ADDRESS FOR RX
    163 000312  104415  000000' 001726'         GETPA$,BEGIN, RVAD              ;GET PHYSICAL ADDRESS FROM 16-BIT RVAD
    164
    165 000320  032767  000010  011412          BIT     #10,@MON                ;TEST IF Q 22 BUS MONITOR
    166 000326  001021                          BNE     2$                      ;IF IT IS ,CONVERT TO 22 BITS
    167
    168
    169 000330  006267  001364                  ASR     TEA                     ;IF NOT PREPARE THE 2 EXTENED ADDR BIT
    170 000334  006267  001360                  ASR     TEA                     ;FOR TX TABLE
    171 000340  006267  001354                  ASR     TEA
    172 000344  006267  001350                  ASR     TEA
    173
    174
    175
    176 000350  006267  001356                  ASR     REA                     ;PREPARE THE 2 EXTENDED ADDR BIT
    177 000354  006267  001352                  ASR     REA                     ;FOR RX TABLE
    178 000360  006267  001346                  ASR     REA
    179 000364  006267  001342                  ASR     REA
    180 000370  000406                          BR      22$
    181
    182 000372                  2$:
        000372  104416  000000' 001716'         MAP22$, BEGIN,TPA               ; GET 22-BIT ADDR FROM 18-BIT ADDR
    183 000400  104416  000000' 001730'         MAP22$, BEGIN,RPA               ; GET 22-BIT ADDR FROM 18-BIT ADDR
    184
    185 000406  005067  001272          22$:    CLR     COUNT                   ;SELECT RAM STARTING ADDRESS FOR TX
    186 000412                  TWODMA:
    187 000412  004767  000472                  JSR     PC,CLRKMV               ;CLEAR REG
    188 000416  004767  000564                  JSR     PC,MAINM1               ;SET MAINT 1
    189
    190 000422  012702  002000                  MOV     #2000,R2
    191 000426  012701  005740'                 MOV     #RTABLE,R1
```


DEC/X11 SYSTEM EXERSIZER MACRO DEFINITION MODULE

```
    192 000432  005021          10$:    CLR     (R1)+                   ;CLEAR RX TABLE
    193 000434  005302                  DEC     R2
    194 000436  001375                  BNE     10$
    195
    196 000440  012702  002000          MOV     #2000,R2
    197 000444  012701  001740'         MOV     #TTABLE,R1
    198 000450                  1$:
    199 000450  104417  000000'         RAND$,BEGIN
    200 000454  016721  177374          MOV     RANNUM,(R1)+            ;WRITTE ONE TABLE LOCATION
    201 000460  005302                  DEC     R2                      ;ALL LOCATION DONE?
    202 000462  001372                  BNE     1$                      ;NO
    203
    204 000464  032767  000010  011246  BIT     #10,@MON                ;TEST IF Q22 MONITOR
    205 000472  001015                  BNE     60$                     ;IF IT IS ,LOAD EXT ADDRESS ON 22 BITS
    206
    207 000474  016777  001216  001166  MOV     TPA,@KMVP12             ;SEND TTABLE ADDRESS
    208 000502  016777  001222  001152  MOV     RPA,@KMVP04             ;SEND RTABLE ADDRESS
    209 000510  016777  001204  001150  MOV     TEA,@KMVP10             ;EXT ADDR FOR TX TABLE ON 18 BITS
    210 000516  016777  001210  001134  MOV     REA,@KMVP02             ;EXT ADDR FOR RX TABLE ON 18 BITS
    211 000524  000414                  BR      40$
    212
    213 000526  016777  001172  001132  60$:    MOV     TEA22,@KMVP10           ;EXT ADDR FOR TX TABLE ON 22 BITS
    214 000534  016777  001176  001116  MOV     REA22,@KMVP02           ;EXT ADDR FOR RX TABLE ON 22 BITS
    215 000542  016777  001154  001120  MOV     TPA22,@KMVP12           ;SEND TTABLE ADDRESS
    216 000550  016777  001160  001104  MOV     RPA22,@KMVP04           ;SEND RTABLE ADDRESS
    217
    218 000556  016777  001122  001110  40$:    MOV     COUNT,@KMVP16           ;LOAD STATING ADDRESS IN RAM
    219
    220 000564  012777  002000  001100  MOV     #2000,@KMVP14           ;SEND TABLE LENGTH
    221 000572  052777  000022  001056  BIS     #22,@KMVCSR             ; LOAD TEST NUMBER 22
    222 000600  104407  000000'         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
        000604  104407  000000'         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
    223 000610  104407  000000'         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
        000614  104407  000000'         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
    224 000620  104407  000000'         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
        000624  104407  000000'         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
    225 000630  104407  000000'         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
        000634  104407  000000'         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
    226 000640  004767  000370          JSR     PC,TSTERR               ;CHECK BSEL0;WHICH ERROR
    227 000644  000441                  BR      3$                      ;TEST OK
    228 000646  000402                  BR      6$                      ;TIME OUT
    229 000650  000407                  BR      7$                      ;NO KMV11 ANSWER
    230 000652  000414                  BR      20$                     ;DATA CMP ERROR
    231
    232 000654                  6$:
                                ;*************************************************************
        000654  104404  000000'         DATER$,BEGIN            ;DATA ERROR!!!
                                ;*************************************************************
                                ;*************************************************************
    233
        000660  104405  000000' 001636'  HRDER$,BEGIN,DUMP      ;TIMEOUT ERROR
                                ;*************************************************************
    234 000666  000473                  BR      15$
    235
    236 000670                  7$:
                                ;*************************************************************
        000670  104404  000000'         DATER$,BEGIN            ;DATA ERROR!!!
                                ;*************************************************************
```

```
                                        ;*****************************************************************
    000674  104405  000000' 001636'    HRDERS,BEGIN,DUMP       ;NO KMV11 ANSWER
                                        ;*****************************************************************
238 000702  000465                      BR      15$
239
240 000704  016767  177200  177172  20$:  MOV     AWAS,WASADR
241 000712  017767  000746  177162        MOV     @KMVP06,SBADR           ;READ ADD IN ERROR
242 000720  017767  000734  177160        MOV     @KMVP02,ASB             ;GOOD VALUE
243 000726  017767  000730  177154        MOV     @KMVP04,AWAS            ;READ WRONG VALUE
244                                     ;*****************************************************************
    000734  104404  000000'            DATERS,BEGIN            ;DATA ERROR!!!
                                        ;*****************************************************************
                                        ;*****************************************************************
245
    000740  104405  000000' 001636'    HRDERS,BEGIN,DUMP       ;DATA CMP ERROR IN RAM DURING TRANSFER
                                        ;*****************************************************************
246 000746  000443                      BR      15$
247
248 000750  012701  001740'      3$:   MOV     #ITABLE,R1              ;TX TABLE ADDRESS
249 000754  012704  005740'            MOV     #RTABLE,R4              ;RX     "     "
250 000760  012702  002000              MOV     #2000,R2                ;TABLE LENGHT
251 000764  016767  177120  177112      MOV     AWAS,WASADR
252 000772  012167  177110      4$:   MOV     (R1)+,ASB
253 000776  012467  177106            MOV     (R4)+,AWAS              ; GET THE DATA TO COMPARE
254 001002  026767  177100  177100      CMP     ASB,AWAS                ;CMP RX TABLE AND TX TABLE
255 001010  001420                      BEQ     5$                      ;OK TEST NEXT LOCATION
256 001012  010467  177064              MOV     R4,SBADR                ; ERROR, REPORT WHERE IN THE TABLE
257 001016  162767  005740' 177056      SUB     #RTABLE,SBADR           ; THE ERROR OCCURRED
258 001024  026727  000652  000012      CMP     FLAG,#10.               ; REPORT ONLY THE FIRST 10 ERRORS
259 001032  001411                      BEQ     15$
260                                     ;*****************************************************************
    001034  104404  000000'            DATERS,BEGIN            ;DATA ERROR!!!
                                        ;*****************************************************************
                                        ;*****************************************************************
261
    001040  104405  000000' 001636'    HRDERS,BEGIN,DUMP       ;REPORT 10 FIRST ERROR
                                        ;*****************************************************************
262 001046  005267  000630              INC     FLAG
263
264 001052  005302              5$:   DEC     R2                      ;ALL MEMORY TESTED?
265 001054  001346                      BNE     4$                      ;NO BRANCH
266
267 001056                      15$:
268 001056  062767  002000  000620      ADD     #2000,COUNT             ;USE OTHER PART OF RAM
269 001064  026767  000614  000614      CMP     COUNT,MAXCNT            ;IS ALL RAM USED?
270 001072  100002                      BPL     30$
271 001074  000167  177312              JMP     TWODMA
272
273 001100                      30$:
    001100  104413  000000'            ENDITS,BEGIN            ;SIGNAL END OF ITERATION.
                                                                ;MONITOR SHALL TEST END OF PASS
274 001104  000167  177146              JMP     RESTRT
```



```
276                                     .SBTTL  SUBROUTINES
277                             ;
278 001110  005077  000542      CLRKMV: CLR     @KMVCSR
279 001114  012777  054000  000534      MOV     #MAINT0,@KMVCSR
280 001122  104407  000000'            BREAKS,BEGIN            ;TEMPORARY RETURN TO MONITOR....
    001126  104407  000000'            BREAKS,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
281 001132  104407  000000'            BREAKS,BEGIN            ;TEMPORARY RETURN TO MONITOR....
    001136  104407  000000'            BREAKS,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
282 001142  012702  000010              MOV     #10,R2
283 001146  016701  000504              MOV     KMVCSR,R1               ;LOAD ADDRESS
284 001152  005021              1$:   CLR     (R1)+                   ;CLEAR
285 001154  104407  000000'            BREAKS,BEGIN            ;TEMPORARY RETURN TO MONITOR....
    001160  104407  000000'            BREAKS,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
286 001164  005302                      DEC     R2                      ;ALL DONE
287 001166  001371                      BNE     1$                      ;NO
288 001170  004567  000116              JSR     R5,CKALL                ;CHECK ALL REG = 0
289 001174  000403                      BR      2$                      ;OK BRANCH AT END
290                                     ;*****************************************************************
    001176  104405  000000' 000000      HRDERS,BEGIN,NULL       ;CSR'S REGISTERS CAN'T BE CLEARED
                                        ;*****************************************************************
291 001204  000207              2$:   RTS     PC
292
293 001206  012777  044000  000442  MAINM1: MOV     #MAINT1,@KMVCSR         ;LOAD ADDRESS
294 001214  004567  000260              JSR     R5,CKSEL0               ;CHECK SEL0=0 BUT MODE BIT =1
295 001220  004000                      .WORD   4000
296 001222  000403                      BR      1$                      ;OK BRANCH
297                                     ;*****************************************************************
    001224  104405  000000' 000000      HRDERS,BEGIN,NULL       ;CANNOT GO INTO MAINT MODE 1
                                        ;*****************************************************************
298 001232  000207              1$:   RTS     PC
299                             ;
300 001234                      TSTERR:
301 001234  004567  000322              JSR     R5,CBSEL0               ;LOOK IF BSEL0=0
302 001240  000000                      .WORD   0
303 001242  000411                      BR      1$                      ;TEST IS OK ,RTS PC
304
305 001244  122767  000200  176636      CMPB    #200,AWAS               ;LOOK IF BSEL0=200
306 001252  001406                      BEQ     2$                      ;TIMEOUT DURING TEST,RTS PC+2
307
308 001254  122767  000100  176626      CMPB    #100,AWAS               ;LOOK IF BSEL0=100
309 001262  001405                      BEQ     3$                      ;DATA CMP ERROR,RTS PC+6
310
311 001264  000407                      BR      4$                      ;NO KMV11 ANSWER ,RTS PC+4
312
313 001266  000207              1$:   RTS     PC                      ;TEST OK
314
315 001270  062716  000002      2$:   ADD     #2,(SP)
316 001274  000207                      RTS     PC                      ;TIMEOUT ERROR
317
318 001276  062716  000006      3$:   ADD     #6,(SP)
319 001302  000207                      RTS     PC                      ;DATA CMP ERROR
320
321
322 001304  062716  000004      4$:   ADD     #4,(SP)
323 001310  000207                      RTS     PC                      ;NO KMV11 ANSWER
324                             ;
325 001312                      CKALL:
```

SUBROUTINES

```
326 001312  016767  000340  176560          MOV     KMVCSR,CSRA                     ;READ SEL0
327 001320  017767  176554  176554          MOV     @CSRA,ACSR
328 001326  001061                          BNE     1$
329 001330  016767  000324  176542          MOV     KMVP02,CSRA                     ;READ SEL2
330 001336  017767  176536  176536          MOV     @CSRA,ACSR
331 001344  001052                          BNE     1$
332 001346  016767  000310  176524          MOV     KMVP04,CSRA                     ;READ SEL4
333 001354  017767  176520  176520          MOV     @CSRA,ACSR
334 001362  001043                          BNE     1$
335 001364  016767  000274  176506          MOV     KMVP06,CSRA                     ;READ SEL6
336 001372  017767  176502  176502          MOV     @CSRA,ACSR
337 001400  001034                          BNE     1$
338 001402  016767  000260  176470          MOV     KMVP10,CSRA                     ;READ SEL10
339 001410  017767  176464  176464          MOV     @CSRA,ACSR
340 001416  001025                          BNE     1$
341 001420  016767  000244  176452          MOV     KMVP12,CSRA                     ;READ SEL12
342 001426  017767  176446  176446          MOV     @CSRA,ACSR
343 001434  001016                          BNE     1$
344 001436  016767  000230  176434          MOV     KMVP14,CSRA                     ;READ SEL14
345 001444  017767  176430  176430          MOV     @CSRA,ACSR
346 001452  001007                          BNE     1$
347 001454  016767  000214  176416          MOV     KMVP16,CSRA                     ;READ SEL16
348 001462  017767  176412  176412          MOV     @CSRA,ACSR
349 001470  001402                          BEQ     2$
350 001472  062705  000002          1$:     ADD     #2,R5
351 001476  000205                  2$:     RTS     R5
352                                         ;
353 001500  012567  176400          CKSEL0: MOV     (R5)+,ASTAT     ;WRITE GOOD
354 001504  012767  177700  000176          MOV     #177700,DELCT1  ; SET TIMEOUT VALUE
355 001512                          3$:
    001512  104407  000000'                BREAKS,BEGIN            ;TEMPORARY RETURN TO MONITOR....
    001516  104407  000000'                BREAKS,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
356 001522  016767  000130  176350          MOV     KMVCSR,CSRA
357 001530  017767  000122  176344          MOV     @KMVCSR,ACSR    ;READ SEL 0
358 001536  026767  176340  176340          CMP     ACSR,ASTAT
359 001544  001405                          BEQ     2$
360 001546  005267  000136                  INC     DELCT1          ; NOT YET KEEP TIMING
361 001552  001357                          BNE     3$
362 001554  062705  000002          1$:     ADD     #2,R5
363 001560  000205                  2$:     RTS     R5
364                                         ;
365 001562  012567  176320          CBSEL0: MOV     (R5)+,ASB
366 001566  012767  170000  000114          MOV     #170000,DELCT1  ; SET TIMEOUT VALUE
367 001574                          3$:
    001574  104407  000000'                BREAKS,BEGIN            ;TEMPORARY RETURN TO MONITOR....
    001600  104407  000000'                BREAKS,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
368 001604  117767  000046  176276          MOVB    @KMVCSR,AMAS
369 001612  126767  176270  176270          CMPB    ASB,AMAS
370 001620  001405                          BEQ     2$
371 001622  005267  000062                  INC     DELCT1          ; NOT YET KEEP TIMING
372 001626  001362                          BNE     3$
373 001630  062705  000002          1$:     ADD     #2,R5
374 001634  000205                  2$:     RTS     R5
375                                         ;
```


DATA STORAGE

```
377                                         .SBTTL  DATA STORAGE
378 001636  001716'                DUMP:    .WORD   IPA
379 001640  001720'                         .WORD   IEA
380 001642  001722'                         .WORD   IPA22
381 001644  001724'                         .WORD   IEA22
382 001646  001730'                         .WORD   RPA
383 001650  001732'                         .WORD   REA
384 001652  001734'                         .WORD   RPA22
385 001654  001736'                         .WORD   REA22
386 001656  000000                  KMVCSR: .WORD   0
387 001660  000000                  KMVP02: .WORD   0
388 001662  000000                  KMVP04: .WORD   0
389 001664  000000                  KMVP06: .WORD   0
390 001666  000000                  KMVP10: .WORD   0
391 001670  000000                  KMVP12: .WORD   0
392 001672  000000                  KMVP14: .WORD   0
393 001674  000000                  KMVP16: .WORD   0
394 001676  001704'                         .WORD   COUNT
395 001700  177777                         .WORD   177777
396                                         ;
397 001702  000000                  FLAG:   .WORD   0
398 001704  000000                  COUNT:  .WORD   0
399 001706  000000                  MAXCNT: .WORD   0
400 001710  000000                  DELCT1: .WORD   0
401 001712  000000                  NUB:    .WORD   0
402                                         ;
403 001714  000000                  TVAD:   .WORD   0
404 001716  000000                  TPA:    .WORD   0
405 001720  000000                  TEA:    .WORD   0
406 001722  000000                  TPA22:  .WORD   0
407 001724  000000                  TEA22:  .WORD   0
408                                         ;
409 001726  000000                  RVAD:   .WORD   0
410 001730  000000                  RPA:    .WORD   0
411 001732  000000                  REA:    .WORD   0
412 001734  000000                  RPA22:  .WORD   0
413 001736  000000                  REA22:  .WORD   0
414                                         ;
415 001740                          TTABLE: .BLKW   2000
416 005740                          RTABLE: .BLKW   2000
417 011740  000000                  QMON:   .WORD   0
418                                         ;
419         000001                          .END
```

```
KMDA DEC/X11 SYSTEM EXERCISER M MACRO M1200  04-JAN-83 09:37  PAGE 5-1
SYMBOL TABLE

ACSR     000102R        CKSEL0  001500R        KMVP10  001666R        PRTY  = 000000        SOFERS= 104406
ADDR     000006R        CLKPRE= 000001         KMVP12  001670R        PRTY0 = 000000        SOFPAS  000046R
ADDR22= 001000          CLKSPS= 104422         KMVP14  001672R        PRTY1 = 000040        SPOINT  000032R
APIPRE= 000200          CLRKMV  001110R        KMVP16  001674R        PRTY2 = 000100        SPSIZ = 000040
ASB      000106R        CONFIG  000056R        KIPRES= 000400         PRTY3 = 000140        SR1     000016R
ASTAT    000104R        COUNT   001704R        KIXTND= 040000         PRTY4 = 000200        SR2     000020R
AUTO  = 000010          CSRA    000100R        MAINM1  001206R        PRTY5 = 000240        SR3     000022R
AWAS     000110R        DATCKS= 104411         MAINT0= 054000         PRTY6 = 000300        SR4     000024R
BEGIN    000000R        DATERS= 104404         MAINT1= 044000         PRTY7 = 000340        START   000224R
BIT0  = 000001          DELCT1  001710R        MAP22S= 104416         PS    = 177776        STAT    000026R
BIT1  = 000002          DUMP    001636R        MAXCNT  001706R        PSW   = 177776        SVR0    000062R
BIT10 = 002000          DVID1   000014R        MODNAM  000000R        PUSH  = 005746        SVR1    000064R
BIT11 = 004000          ECCMEM= 000100         MODSP   000224R        PUSH2 = 024646        SVR2    000066R
BIT12 = 010000          ENDITS= 104413         MSGNS = 104403         PWRFLG= 000002        SVR3    000070R
BIT13 = 020000          ENDS  = 104410         MSGSS = 104402         QMON    011740R       SVR4    000072R
BIT14 = 040000          ERRTYP  000106R        MSGS  = 104401         QMON22= 000010        SVR5    000074R
BIT15 = 100000          EXITS = 104400         NCPUUP= 000020         RANDS = 104417        SVR6    000076R
BIT2  = 000004          FLAG    001702R        NOAPTY= 000002         RANNUM  000054R       SYSCNT  000052R
BIT3  = 000010          GETPAS= 104415         NUB     001712R        REA     001732R       TEA     001720R
BIT4  = 000020          GWBUFS= 104414         NULL  = 000000         REA22   001736R       TEA22   001724R
BIT5  = 000040          HRDCNT  000044R        OPEN  = 000000         RESIRT  000256R       TPA     001716R
BIT6  = 000100          HRDERS= 104405         OTOAS = 104420         RES1    000056R       TPA22   001722R
BIT7  = 000200          HRDPAS  000050R        PARPRE= 002000         RES2    000060R       TRPDFD= 000023
BIT8  = 000400          ICONT   000036R        PASCNT  000034R        RH70  = 001000        TSTERR  001234R
BIT9  = 001000          ICOUNT  000040R        PDPF11= 000002         RPA     001730R       TTABLE  001740R
BREAKS= 104407          IDNUM   000122R        PDPLS1= 020000         RPA22   001734R       TVAD    001714R
BR1      000012R        INDPAR= 000040         PDP44 = 100000         RSTRT   000112R       TWODMA  000412R
BR2      000013R        INIT    000030R        PDP60 = 004000         RTABLE  005740R       USTACK= 000001
BTOOS = 104421          INTR    000120R        PDP70 = 010000         RVAD    001726R       VECTOR  000010R
CAPRES= 000004          KMVCSR  001650R        PIRQS = 000004         R6    =%000006        WASADR  000104R
CBSEL0  001562R         KMVP02  001660R        PDPSP = 005726         R7    =%000007        WDFR    000116R
CDATAS= 104412          KMVP04  001662R        PDPSP2= 022626         SBADR   000102R       WDTO    000114R
CKALL   001312R         KMVP06  001664R        PRHMSS= 000002         SOFCNT  000042R       XFLAG   000005R
CKHNGS= 000001

. ABS.  000000    000
        011742    001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  15281 WORDS  ( 60 PAGES)
DYNAMIC MEMORY:  16644 WORDS  ( 64 PAGES)
ELAPSED TIME:  00:00:16
```

DIAGNOSTIC ENGINEERING

digital **DECO☐ DEPO☒ SUBMISSION☐**

FOR RELEASE ENG. USE: ☐ NEW ☐ CHANGE ☐ DELETE

PRODUCT IDENTIFICATION

MD	LIBRARY	PRODUCT NUMBER	REV	PATCH	ECO TALLY	PRODUCT DATE (DD MMM YY)	STATUS	DISTRIBUTION	1ST COPY-RIGHT YEAR	LAST COPY-RIGHT YEAR
	ZZ	CXDMC	B	1	Ø1	24 APR 79	☐ OBSOLETE	X G ☐ R	1976	1979

TITLE: CXDMCB1 DMC-11 MODULE

AUTHOR: D. BUTENHOF | MAINTAINING GROUP: DEC/XTL SUPT GRP | MAINTAINER: D. BUTENHOF | SUBMITTING ENGINEER: D. BUTENHOF

PRODUCT COMPONENTS

CK	DESCRIPTION	PRODUCT NO.	REV	CK	DESCRIPTION	PRODUCT NO.	REV
	DOCUMENT				INDEX		
	LISTING				SOURCE MEDIA		
	OBJECT MEDIA				TEST MEDIA		
		AF-E953B-M1					

PRODUCTS OBSOLETED (other than previous version)

	LIBRARY	PRODUCT NUMBER	REV		LIBRARY	PRODUCT NUMBER	REV		LIBRARY	PRODUCT NUMBER	REV
MD				M D				M D			

PRODUCT CHARACTERISTICS

PROCESSORS PRODUCT OPERATES WITH (Enter all applicable 2-digit codes representing the Processor the product operates with. See separate instructions.)

OPERATIONAL CODES (Enter all applicable 2-digit codes that describe the product. See separate instructions.)

ACT/APT/XXDP INFORMATION FIELD	EXT	ACT SEQ NUMBER	ACT/XXDP COMPATIBLE?	APT COMPATIBLE?	1ST PASS RUN TIME	SUBSEQUENT PASS RUN TIME
			☐ Y ☐ N	☐ Y ☐ N	SECONDS	SECONDS

DECO/DEPO INFORMATION

PROBLEM REPORTS CLOSED:

SOFTWARE AFFECTED | MULTIMEDIA AFFECTED? ☒ YES ☐ NO

KIT NUMBERS							
	ZJ13Ø-RB						
	ZJ129-RZ, FR						

PROBLEM:

THE "DDCMP ERROR COUNTERS ARE NON-ZERO" SOFT ERROR IS MISLEADING SINCE IT REFERS TO A RECOVERABLE AND SOFTWARE-TRANSPARENT SITUATION. NEVERTHELESS, IF 4Ø SUCH OCCUR , THE MODULE WILL BE DROPPED BY THE MONITOR.

SOLUTION:

THE SOFT ERROR CALL AND MESSAGE ARE DELETED FROM THE MODULE BY THIS PATCH, AND THE HARD ERROR CALL IS EXTENDED FOR CLARIFICATION OF THE SITUATION, IF THE DMC-11 SHOULD BE UNABLE TO RECOVER.

DEPO PATCH AREA

CHANGE LOC	FROM	TO	CHANGE LOC	FROM	TO
1226	1Ø44Ø3	4Ø2	1726	51122	51Ø4Ø
1334	1Ø44Ø6	4Ø2	173Ø	51117	521Ø5
1716	224ØØ	2ØØ72	1732	2644Ø	44522
172Ø	47523	53117	1734	42Ø4Ø	515Ø5
1722	521Ø6	511Ø5	1736	415Ø4	Ø
1724	4244Ø	3344Ø			

TESTING ENGINEER	MANUFACTURING ENGINEER	SUPPORT ENGINEER	CHARGE DECO/DEPO TO DISCRETE PROJECT NUMBER
DATE: 24 APR 79	DATE: 10-MAY-79	DATE:	Ø98-Ø546Ø
MAINTAINER	FIELD SERVICE	WAIVERING MANAGER	COORDINATION NO. 313Ø
	DATE:	DATE:	

J. E. CASELLA

Y
IEAA DEC/X11 SYSTEM EXERCISER M MACRO M1113 29-JUL-82 08:03

```
      1         000000                          .REPT 0
      2
      3
      4
      5
      6
      7
      8                                         IDENTIFICATION
      9                                         --------------
     10
     11
     12
     13                               PRODUCT CODE:          AC-T241A-MC
     14
     15
     16                               PRODUCT NAME:          CXIEAA0 IEU/IEQ11- DEC/X MOD
     17
     18
     19                               DATE CREATED:          JULY 1982
     20
     21
     22                               MAINTAINER:            CSS MUNICH
     23
     24
     25                               AUTHOR:                PETER SEEBACH
     30
     31
     32
     33                               COPYRIGHT:             1982,1983 ,DIGITAL EQUIPMENT CORPORATION
     34
     35                                                      THE INFORMATION IN THIS DOCUMENT IS
     36                                                      SUBJECT TO CHANGE WITHOUT NOTICE AND
     37                                                      SHOULD NOT BE CONSTRUED AS A COMMITMENT
     38                                                      BY DIGITAL EQUIPMENT CORPORATION. DEC
     39                                                      ASSUMES NO RESPONSIBILITY FOR ANY ERRORS
     40                                                      THAT MAY APPEAR IN THIS DOCUMENT. DEC
     41                                                      ASSUMES NO RESPONSIBILITY FOR USE OF
     42                                                      SOFTWARE ON EQUIPMENT NOT SUPPLIED BY
     43                                                      DEC.
```

```
                                1.0     ABSTRACT
                                        --------

                                        IEQ/IEQ IS AN IOMODX DEC/X11 SOFTWARE MODULE WHICH
                                        EXERCISES THE IEU-11 OR THE IEQ-11.
                                        THE MAIN CHARACTERISTICS OF THIS MODULE ARE:

                                        1.   DOES DMA DATA TRANSFER FROM CHANNEL 1 TO CHANNEL 2.
                                        2.   DOSE DMA DATA TRANSFER FROM CHANNEL 2 TO CHANNEL 1.

                                        THE MODULE CAN SUPPORT UP TO 8 DEVICES BY SETTING DVC TO
                                        THE APPROPRIATE VALUE AT CONFIGURATION TIME.IF MORE THAN ONE
                                        DEVICE IS SPECIFIED ,THE ADDRESSES ARE INCREMENTED BY 20 OCTAL
                                        FROM THAT GIVEN AS DVA.

                                        ALL ERRORS DETECTED ARE REPORTED ON THE CONSOLE.


                                2.0     REQUIREMENTS
                                        ------------

                                        HARDWARE:

                                        A.      PDP11 OR LSI11 CAPABLE OF SUPPORTING DEC/X11.
                                        B.      UP TO 8 IEU-11 (FOR PDP11) OR IEQ-11 (FOR LSI11) INTERFACES.
                                        C.      A BC08 TEST CABLE IS REQUIRED BETWEEN THE TWO
                                                CHANNELS OF THE MODULE.
                                        D.      ANY USER IEC BUS DEVICES MUST BE DISCONNECTED.

                                        SOFTWARE:

                                        DEC/X11 MONITOR

                                        STORAGE:

                                        1K OF MEMORY


                                3.      PASS DEFINITION
                                        ---------------

                                        ONE PASS OF THE IEU/IEQ MODULE CONSISTS OF TEN CYCLES
                                        OF THE TEST.


                                4.      EXECUTION TIME
                                        --------------

                                        ONE PASS OF THE IEU/IEQ RUNNING ALONE ON A PDP-11/60
                                        TAKES APPROXIMATELY 10 SECONDS.
```



```
                                5.      CONFIGURATION REQUIREMENTS
                                        --------------------------

                                        DEFAULT PARAMETERS:

                                        DEVADR:160140, VECTOR:420, BR1:4, DEVCNT:1, ICONST:10, RBUFVA:BUFIN,
                                        RBUFSZ:256., #BUFFS:256.

                                        REQUIRED PARAMETERS:

                                        NONE


                                6.      DEVICE/OPTION SETUP
                                        -------------------

                                        MAKE CERTAIN THAT THE IEU-11 OR IEQ-11 ARE CONNECTED AND READY.
                                        THIS ENTAILS INSTALLING A BC08S-01 CABLE FROM J1 TO J2 .


                                7.      MODULE OPERATION
                                        ----------------

                                        TEST SEQUENCE:
                                        A.      TEST DATA TRANSFER FROM CHANNEL 1 TO 2

                                                1. SET CHANNEL 1 IN CONTROLLER ACTIVE STATE
                                                2. SELECT CHANNEL 1 AS TAKER (TACS)
                                                3. SELECT CHANNEL 2 AS LISTENER (SENDING MLA2)
                                                4. LOADING DMA SETUP
                                                5. START DMA BY LOADING GTS INTO ACR1
                                                6. INTERRUPT OCCERS WHEN BC OF  OF CHANNEL 1 IS SET
                                                7. DATA COMPARE

                                        B.      TEST DATA TRANSFER FROM CHANNEL 2 TO 1

                                                1. SELECT CHANNEL 1 AS LISTENER (LOAD LUN)
                                                2. SELECT CHANNEL 2 AS TAKER (LOAD TUN)
                                                3. LOAD DMA SET UP AND START DMA (SET DMA ENB)
                                                4. INTERRUPT OCCERS WHEN BC OF OF CHANNEL 2 IS SET
                                                5. DATA COMPARE


                                8.      OPERATION OPTIONS
                                        -----------------

                                        NONE


                                9.      NO STANDARD PRINTOUTS
                                        ---------------------
```

```
159                                     ALL ERROR PRINTOUTS HAVE THE STANDARD FORMATS DESCRIBED
160                                     IN THE DEC/X11 DOCUMENT.
161
162
163
164                             10.     STARTING PROCEDURE
165                                     ------------------
166
167                                     A.  LOAD AND START THE DEC/X11 PROGRAM ACCORDING TO
168                                         THE DEC/X11 USER'S MANUAL.
169
170                                     B.  ENSURE THAT THE DEVICE ADDRESS AND VECTOR ARE
171                                         CORRECT FOR THE IEU/IEQ INTERFACE.
172
173
174
175                             11.     LISTING
176                                     -------
177                                     .ENDR
```



```
179                                     .SBTTL  DDXCOM HEADER
180 000000                      IDMODX  <IEAA >,1e0140,420,4,0,0,10.,0,BUFIN,256.,256.
    000000                              MODULE  150000,IEAA ,160140,420,4,0,0,10.,0,BUFIN,256.,256.
                                        .TITLE  IEAA DEC/X11 SYSTEM EXERCISER MODULE
                                ;       DDXCOM  VERSION  6       23-MAY-78
                                        .LIST   BIN
                                ;*****************************************************************************
    000000                              BEGIN:
    000000      111     105     101     MODNAM: .ASCII  /IEAA / ;MODULE NAME.
    000003      101     040
    000005      000                     XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF BUFF USAGE
    000006  160140                      ADDR:   160140+0                ;1ST DEVICE ADDR.
    000010  000420                      VECTOR: 420+0                   ;1ST DEVICE VECTOR.
    000012      200                     BR1:    .BYTE   PRTY4+0         ;1ST BR LEVEL.
    000013      000                     BR2:    .BYTE   PRTY0+0         ;2ND BR LEVEL.
    000014  000001                      DVID1:  0+1                     ;DEVICE INDICATOR 1.
    000016  000000                      SR1:    OPEN                    ;SWITCH REGISTER 1
    000020  000000                      SR2:    OPEN                    ;SWITCH REGISTER 2
    000022  000000                      SR3:    OPEN                    ;SWITCH REGISTER 3
    000024  000000                      SR4:    OPEN                    ;SWITCH REGISTER 4
                                ;*****************************************************************************
    000026  150000                      STAT:   150000                  ;STATUS WORD.
    000030  000336'                     INIT:   START                   ;MODULE START ADDR.
    000032  000252'                     SPOINT: MODSP                   ;MODULE STACK POINTER.
    000034  000000                      PASCNT: 0                       ;PASS COUNTER.
    000036  000012                      ICONT:  10.                     ;# OF ITERATIONS PER PASS=10.
    000040  000000                      ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
    000042  000000                      SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
    000044  000000                      HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
    000046  000000                      SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
    000050  000000                      HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
    000052  000000                      SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
    000054  000000                      RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
    000056                              CONFIG:                         ;RESERVED FOR MONITOR USE
    000056  000000                      RES1:   0                       ;RESERVED FOR MONITOR USE
    000060  000000                      RES2:   0                       ;RESERVED FOR MONITOR USE
    000062  000000                      SVR0:   OPEN                    ;LOC TO SAVE R0.
    000064  000000                      SVR1:   OPEN                    ;LOC TO SAVE R1.
    000066  000000                      SVR2:   OPEN                    ;LOC TO SAVE R2.
    000070  000000                      SVR3:   OPEN                    ;LOC TO SAVE R3.
    000072  000000                      SVR4:   OPEN                    ;LOC TO SAVE R4.
    000074  000000                      SVR5:   OPEN                    ;LOC TO SAVE R5.
    000076  000000                      SVR6:   OPEN                    ;LOC TO SAVE R6.
    000100  000000                      CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
    000102                              SBADR:                          ;ADDR OF GOOD DATA, OR
    000102  000000                      ACSR:   OPEN                    ;CONTENTS OF CSR.
    000104                              WASADR:                         ;ADDR OF BAD DATA, OR
    000104  000000                      ASTAT:  OPEN                    ;STATUS REG CONTENTS.
    000106                              ERRTYP:                         ;TYPE OF ERROR
    000106  000000                      ASR:    OPEN                    ;EXPECTED DATA.
    000110  000000                      AWAS:   OPEN                    ;ACTUAL DATA.
    000112  000372'                     RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
    000114  000000                      WDTO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
    000116  000000                      WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
    000120  000000                      INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
    000122  000000                      IDNUM:  0                       ;MODULE IDENTIFICATION NUMBER=0
    000124  001544'                     RBUFVA: BUFIN                   ;READ BUFFER VIRTUAL ADDRESS
```

```
000126  000000                  RBUFPA: OPEN                    ;READ BUFFER PHYSICAL ADDRESS
000130  000000                  RBUFEA: OPEN                    ;READ BUFFER EA BITS
000132  000400                  RBUFSZ: 256.                    ;SIZE OF THE READ BUFFER
000134  000000                  WBUFPA: OPEN                    ;WRITE BUFFER PHYSICAL ADDRESS
000136  000000                  WBUFEA: OPEN                    ;WRITE BUFFER EA BITS
000140  000400                  WBUFRQ: 256.                    ;WRITE BUFFER SIZE REQUESTED
000142  000000                  WBUFSZ: OPEN                    ;WRITE BUFFER SIZE AVAILABLE
000144  000000                  CDERCT: OPEN                    ;CDATA/DATCK ERROR COUNT
000146  000000                  CDWDCT: OPEN                    ;CDATA/DATCK WORD COUNT
000150  000000                  FREE:   OPEN                    ;RESERVED FOR FUTURE USE
        000040                          .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000252                          MODSP:
                                ;*****************************************************************************
```



```
182                                     .SBTTL  CONSTANTS AND VARIABLES.
183 000252  000000          WCNT1:  .WORD 0         ;
184 000254  000000          WCNT2:  .WORD 0         ;
185 000256  000000          XMEM:   .WORD 0         ;
186 000260  000000          FUNC:   .WORD 0         ;
187 000262  000000          DVICE:  .WORD 0         ;
188 000264  000000          MSK:    .WORD 0         ;
189 000266  000000          INTFLA: .WORD 0         ;
190 000270  000000          INTFLB: .WORD 0         ;
191
192 000272  000000          IIRX:   .WORD 0         ;POINTER TO IEEE INTERRUPT REGISTER
193 000274  000000          ADR::   .WORD 0         ;POINTER TO ADDRESS REGISTER
194 000276  000000          IIRHX:: .WORD 0         ;POINTER TO HIGH BYTE OF THE INTERRUPT REGISTER
195 000300  000000          ISRX::  .WORD 0         ;POINTER TO IEEE STATUS REGISTER
196 000302  000000          ISRLX:  .WORD 0         ;POINTER TO IEEE STATUS REGISTER ,LOW BYTE
197 000304  000000          SPR::   .WORD 0         ;POINTER TO SERIAL POLL REGISTER
198 000306  000000          ACR::   .WORD 0         ;POINTER TO AUXILIARY COMMAND REGISTER
199 000310  000000          TCRX::  .WORD 0         ;POINTER TO IEEE COMMAND REGISTER
200 000312  000000          MASK0:: .WORD 0         ;POINTER TO MASK0 REGISTER
201 000314  000000          MASK1:: .WORD 0         ;POINTER TO MASK1 REGISTER
202 000316  000000          IDRX::  .WORD 0         ;POINTER TO DATA REGISTER
203 000320  000000          PPR::   .WORD 0         ;POINTER TO PARALLEL POLL REGISTER
204 000322  000000          DOR::   .WORD 0         ;POINTER TO THE DATA OUT REGISTER
205 000324  000000          DIR:    .WORD 0         ;POINTER TO DATA IN REGISTER
206 000326  000000          CSRX:   .WORD 0         ;POINTER TO CONTROL STATUS REGISTER
207 000330  000000          BARX:   .WORD 0         ;POINTER TO BUS ADDRESS REGISTER
208 000332  000000          BCRX:   .WORD 0         ;POINTER TO BYTE COUNT REGISTER
209 000334  000000          MCRX:   .WORD 0         ;POINTER TO MATCH COUNT REGISTER
```

```
                                                .SBTTL  START-RESTART-TEST
                                        START:
                        016767  177452  177716          MOV     DVID1,DVICE     ;SAVE DEVICE INDICATOR TO A WORK LOCATION
                        012767  000001  177712          MOV     #1,MSK          ;SET UP FOR FIRST DEVICE
                        016705  177430                  MOV     ADDR,R5         ;GET FIRST DEVICE ADDRESS
                        010567  177516          SFT:    MOV     R5,CSRA         ;LOAD CSRA FOR CKDATA ERROR PRINTOUT
                        036767  177676  177672          BIT     MSK,DVICE       ;THIS DEVICE SELECTED?
                        001013                          BNE     CHECK           ;IF YES, BRANCH
                        062767  000020  177500  RESTRT: ADD     #20,CSRA        ;GET NEXT DEVICE ADDRESS
                        016705  177474                  MOV     CSRA,R5         ;LOAD NEXT DEVICE ADDRESS
                        106367  177654                  ASLB    MSK             ;MOVE DEVICE MASK BIT
                        001362                          BNE     SET             ;DONT CHECK MORE THAN 8 DEVICES
                        000240                          NOP
                        000167  177716                  JMP     START           ;IF 8 DEVICES ARE CHECKED JUMP TO START
                        004767  000272          CHECK:  JSR     PC,SETUP        ;GET DEVICE ADDRESSES
                        005067  177636          AGIAN:  CLR     INTFLA          ;CLEAR INTERRUPT FLAG 1
                        005067  177634                  CLR     INTFLB          ;CLEAR INTERRUPT FLAG 2
                        004767  000442                  JSR     PC,CACS         ;SET IEX CHANNEL 1 INTO CACS
                                                ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                                ;       TEST DATA TRANSFER FROM CHANNEL 1 TO 2 WITH INTERRUPT
                                                ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                        112777  000212  177640          MOVB    #212,@ACR       ;---LOAD TON INTO ACR 1-----
                        112777  000041  177646          MOVB    #41,@DDR        ;---LOAD MLA2 INTO DDR 1----
                        004767  000572                  JSR     PC,LOOP         ;WAIT A LITTLE
                                                ;+++DMA SET UP FOR CHANNEL 1+++++++++++++++++++++++++++++++++++++++++++++++++
                        004767  000610                  JSR     PC,WRITE        ;DMA SET UP FOR WRITE CANNEL (TX)
                                                ;+++DMA SET UP FOR CHANNEL 2+++++++++++++++++++++++++++++++++++++++++++++++++
                        052777  000010  177634          BIS     #10,@CSRX       ;SELECT CHANNEL 2
                        004767  000666                  JSR     PC,READ         ;DMA SET UP FOR READ CHANNEL (RX)
                                                ;+++START DMA AND CHECK DATA ++++++++++++++++++++++++++++++++++++++++++++++++
                        042777  000010  177622          BIC     #10,@CSRX       ;SELECT CHANNEL 1
                        112777  000013  177574          MOVB    #13,@ACR        ;LOAD GTS (START DMA)
                        104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR, MODULE WAIT FOR INTERRUPT.
                                        INTSV1:
                                                ;---------------------------------------------------------------
    000516              000004  000000' 000524'          PIRQ$,BEGIN,ITR1                ; QUEUE UP TO CONTINUE AT ITR1 AND RTI
                                                ;---------------------------------------------------------------
                        005267  177536          ITR1:   INC     INTFLA          ;INTERRUPT HAS OCCURRED
                        000240                          NOP                     ;RETURN ADDRESS FROM INTERRUPT SERVICE ROUTINE
                        104412  000000' 000126'         CDATA$,BEGIN,RBUFPA     ; REQUEST FOR MONITOR TO CHECK DATA
    000540              000542'                         .+2                     ; IF ERROR, CONTINUE
                        000240                          NOP
                                                ;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                                ;       IEST DATA TRANSFER FROM CHANNEL 2 T OCHANNEL 1
                                                ;++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                        005067  177520                  CLR     INTFLB          ;
                        005067  177512                  CLR     INTFLA          ;
                        005077  177546                  CLR     @CSRX           ;CLEAR CSR ON CHANNEL 1
                        017767  177540  177470          MOV     @DIR,XMEM       ;READ DIR FOR CLEAR BD BIT AND ACCRD
                        112777  000211  177512          MOVB    #211,@ACR       ;LOAD TON INTO ACR 1
                        052777  000010  177524          BIS     #10,@CSRX       ;SELECT CHANNEL 2
                        012777  000010  177516          MOV     #10,@CSRX       ;CLEAR CSR ON CHANNEL 2
                        112777  000212  177470          MOVB    #212,@ACR       ;LOAD TON INTO ACR 2
                                                ;+++++-DMA SET UP FOR CHANNEL 1-----------------------------------------
                        042777  000010  177502          BIC     #10,@CSRX       ;SELECT CHANNEL 1
                        004767  000534                  JSR     PC,READ         ;LOAD DMA SET UP FOR READ CHANNEL (RX)
                        000240                          NOP
```



```
                                                ;+++++-DMA SET UP AND START FOR CHANNEL 2-------------------------------
                        052777  000010  177466          BIS     #10,@CSRX       ;SELECT CHANNEL 2
                        000240                          NOP
                        004767  000426                  JSR     PC,WRITE        ;LOAD DMA SET UP FOR WRITE CHANNEL (TX)
                        104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR, MODULE WAIT FOR INTERRUPT.
                                        INTSV2:
                                                ;---------------------------------------------------------------
    000652              000004  000000' 000660'          PIRQ$,BEGIN,ITR2                ; QUEUE UP TO CONTINUE AT ITR2 AND RTI
                                                ;---------------------------------------------------------------
                        005267  177404          ITR2:   INC     INTFLB          ;INTERRUPT HAS OCCURRED
                        000240                          NOP                     ;RETUN ADDRESS FROM INTERRUPT SERVICE ROUTINE
                        104412  000000' 000126'         CDATA$,BEGIN,RBUFPA     ; REQUEST FOR MONITOR TO CHECK DATA
    000674              000676'                         .+2                     ; IF ERROR, CONTINUE
                        000240                          NOP
                        017767  177420  177350          MOV     @DIR,XMEM       ;READ DIR
                        104413  000000'                 ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
                                                                                ;MONITOR SHALL TEST END OF PASS
                                                                                ;MONITOR SHALL TEST END OF PASS
                        000167  177506                  JMP     AGIAN           ;IF NOT EQUAL JUMP TO AGIAN
```

```
                                        .SBTTL  SUBROUTINES
                                ;++*************************************************************************
                                ;       SUBROUTINE TO SETUP THE DEVICE ADDRESSES,VECTOR AND BR.
                                ;***************************************************************************
000716  010567  177356          SETUP:  MOV     R5,ISRX         ;GENERATE REGISTER ADDRESSES
000722  010567  177364                  MOV     R5,MASK0        ;LOAD ADDRESS 0
000726  105725                          TSTB    (R5)+           ;TEST ADDRESS 0
000730  010567  177360                  MOV     R5,MASK1        ;LOAD ADDRESS 1
000734  005205                          INC     R5              ;
000736  010567  177330                  MOV     R5,IIRX         ;LOAD ADDRESS 2
000742  010567  177334                  MOV     R5,ISRLX        ;
000746  105725                          TSTB    (R5)+           ;TEST ADDRESS 2
000750  010567  177320                  MOV     R5,ADR          ;LOAD ADDRESS 3
000754  005205                          INC     R5              ;
000756  010567  177326                  MOV     R5,ICRX         ;LOAD ADDRESS 4
000762  010567  177316                  MOV     R5,SPR          ;
000766  105725                          TSTB    (R5)+           ;TEST ADDRESS 4
000770  010567  177312                  MOV     R5,ACR          ;LOAD ADDRESS 5
000774  005205                          INC     R5              ;
000776  010567  177322                  MOV     R5,DIR          ;LOAD ADDRESS 6
001002  010567  177312                  MOV     R5,PPR          ;
001006  105725                          TSTB    (R5)+           ;TEST ADDRESS 6
001010  010567  177306                  MOV     R5,DOR          ;LOAD ADDRESS 7
001014  005205                          INC     R5              ;
001016  010567  177304                  MOV     R5,CSRX         ;LOAD ADDRESS 10
001022  005725                          TST     (R5)+           ;TEST ADDRESS 10
001024  010567  177300                  MOV     R5,HARX         ;LOAD ADDRESS 12
001030  005725                          TST     (R5)+           ;TEST ADDRESS 12
001032  010567  177274                  MOV     R5,BCRX         ;LOAD ADDRESS 14
001036  005725                          TST     (R5)+           ;TEST ADDRESS 14
001040  010567  177270                  MOV     R5,MCRX         ;LOAD ADDRESS 16
001044  016705  176740                  MOV     VECTOR,R5       ;GET FIRST VECTOR ADDRESS
001050  012725  000516'                 MOV     #INTSV1,(R5)+   ;POINT VECTOR TO INTERRUPT SERVICE ROUTINE
001054  116715  176732                  MOVB    BR1,(R5)        ;SET PRIORITY FOR FIRST VECTOR
001060  016705  176724                  MOV     VECTOR,R5       ;GET SECOND VECTOR ADDRESS
001064  062705  000004                  ADD     #4,R5           ;...
001070  012725  000652'                 MOV     #INTSV2,(R5)+   ;POINT VECTOR TO INTERRUPT SERVICE ROUTINE
001074  116715  176712                  MOVB    BR1,(R5)        ;SET PRIORITY FOR SECOND VECTOR
001100  000207                          RTS     PC              ;RETURN TO USER
```



```
                                        ;++*************************************************************************
                                        ;       SUBROUTINE TO CLEAR,AND SETUP BOTH IEX CHANNELS
                                        ;++*************************************************************************
001102  005077  177220                  CACS:   CLR     @CSRX           ;SELECT CHANNEL 1
001106  112777  000200  177172                  MOVB    #200,@ACR       ;LOAD SWRST INTO ACR 1
001114  112777  000000  177164                  MOVB    #0,@ACR         ;LOAD NOT SWRST INTO ACR 1
001122  112777  000000  177162                  MOVB    #0,@MASK0       ;CLEAR MASK REGISTER 0
001130  112777  000000  177156                  MOVB    #0,@MASK1       ;CLEAR MASK REGISTER 1
001136  112777  000000  177130                  MOVB    #0,@ADR         ;LOAD DEVICE PRIM. ADDR.1 INTO ADR 1
001144  012777  000010  177154                  MOV     #10,@CSRX       ;SELECT CHANNEL 2
001152  112777  000200  177126                  MOVB    #200,@ACR       ;LOAD SWRST INTO ACR 2
001160  112777  000000  177120                  MOVB    #0,@ACR         ;LOAD NOT SWRST INTO ACR 2
001166  112777  000000  177116                  MOVB    #0,@MASK0       ;CLEAR MASK REGISTER 0
001174  112777  000000  177112                  MOVB    #0,@MASK1       ;CLEAR MASK REGISTER 1
001202  112777  000001  177064                  MOVB    #1,@ADR         ;LOAD DEVICE PRIM. ADDR.2 INTO ADR 2
001210  005077  177112                          CLR     @CSRX           ;SELECT CHANNEL 1
001214  112777  000217  177064                  MOVB    #217,@ACR       ;LOAD SIC INTO ACR 1
001222  012701  000100                          MOV     #100,R1         ;WAIT 100 US
001226                                  WAIT:
001226  104407  000000'                         BREAKS,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001232  104407  000000'                         BREAKS,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001236  005301                                  DEC     R1              ;...
001240  001372                                  BNE     WAIT            ;...
001242  112777  000017  177036                  MOVB    #17,@ACR        ;LOAD NOT SIC INTO ACR 1
001250  000207                                  RTS     PC              ;RETURN
```

```
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                        ;       SUBROUTINE LOOP WAIT (AT LEAST) 1 US.
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
001252  012701  000001          LOOP:   MOV     #1,R1           ;LOAD LOOP COUNTER
001256                          1$:
001256  104407  000000'                  BREAKS,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001262  104407  000000'                  BREAKS,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001266  005301                          DEC     R1              ;DECREMENT LOOP COUNTER
001270  001372                          BNE     1$              ;LOOP UNTIL DONE
001272  000207                          RTS     PC              ;RETURN
```



```
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                        ;       SUBROUTINE WRITE (DMA SET UP)
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
001274                          WRITE:
001274  104414  000000'                  GETBUFS, BEGIN          ;GET WRITE BUFFER INFORMATION
001300  016767  176636  176744          MOV     WBUFSZ,WCNT1    ;GET WRITE BUFFER SIZE TO A WORK LOCATION
001306  006367  176740                  ASL     WCNT1           ;CORECTION FOR CRDATA (COMPERED WORD)
001312  005467  176734                  NEG     WCNT1           ;GET 2'S COMPLEMENT
001316  016777  176730  177006          MOV     WCNT1,@WCRX     ;LOAD BYTE COUNT REGISTER OF SELECTED CHANNEL
001324  016777  176604  176776          MOV     WBUFPA,@BAPX    ;LOAD BUFFER ADDRESS
001332  016767  176600  176716          MOV     WBUFEA,XMEM     ;GET EXTENED MEMORY BITS
001340  012767  000105  176712          MOV     #105,FUNC       ;LOAD DMA FUNCTION
001346  056767  176704  176704          BIS     XMEM,FUNC       ;SET IO THE DMA FUNCTION EXTENDED MEMORY BITS
001354  056777  176700  176744          BIS     FUNC,@CSRX      ;LOAD DMA FUNCTION INTO CSR
001362  000207                          RTS     PC              ;RETURN


                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
                                        ;       SUBROUTINE READ (DMA SET UP)
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
001364                          READ:
001364  104415  000000' 000124'         GETPAS,BEGIN, RBUFVA            ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
001372  016767  176534  176654          MOV     RBUFSZ,WCNT2    ;SAVE WRITE BUFFER SIZE
001400  005267  176650                  INC     WCNT2           ;WCNT1+1 NO INTERRUPT ON RECEIVE CHANNEL
001404  006367  176644                  ASL     WCNT2           ;CORRECTION FOR CRDATA (WORD COMPARE)
001410  005467  176640                  NEG     WCNT2           ;GET 2'S COMPLEMENT
001414  016777  176634  176710          MOV     WCNT2,@WCRX     ;LOAD BYTE COUNT REGISTER
001422  016777  176500  176700          MOV     RBUFPA,@BAPX    ;LOAD BUFFER ADDRESS
001430  016767  176474  176620          MOV     RBUFEA,XMEM     ;LOAD EXTENED MEMORY BITS
001436  012767  000101  176614          MOV     #101,FUNC       ;LOAD DMA FUNCTION
001444  056767  176606  176606          BIS     XMEM,FUNC       ;SET EXTENDED MEMORY BITS
001452  000240                          NOP
001454  056777  176600  176644          BIS     FUNC,@CSRX      ;LOAD DMA FUNCTION INTO CSR
001462  000207                          RTS     PC              ;RETURN
```

```
IEAA DEC/X11 SYSTEM EXERCISER M MACRO M1113  29-JUL-82 08:03  PAGE 11
BUFFER AERA

    394                                     .SBTTL  BUFFER AERA
    395
    396
    397 001464                              .BLKW   30              ;PATCH AREA
    398
    399 001544                      BUFIN:  .BLKB   1024.           ;BUFFER IS 1024. BYTES = 512. WORDS
    400
    401
    402
    403         000001                      .END                    ;END OF PROGRAM
```



```
IEAA DEC/X11 SYSTEM EXERCISER M MACRO M1113  29-JUL-82 08:03  PAGE 11-1
SYMBOL TABLE

ACR     000306RG        BUFIN   001544R         IIRX    000272R         PRTX0 = 000000          SPR     000304RG
ACSR    000102R         CACS    001102R         IMODX.= 000000          PRTY1 = 000040          SPSIZ = 000040
ADDR    000006R         CDATAS= 104412          INIT    000030R         PRTY2 = 000100          SR1     000016R
ADDR22= 001000          CDERCT  000144R         INTFLA  000266R         PRTY3 = 000140          SR2     000020R
ADR     000274RG        CDWDCT  000146R         INTFLB  000270R         PRTY4 = 000200          SR3     000022R
AGIAN   000424R         CHECK   000420R         INTR    000120R         PRTY5 = 000240          SR4     000024R
ASR     000106R         CONFIG  000056R         INTSV1  000516R         PRTY6 = 000300          START   000336R
ASTAT   000104R         CSRA    000100R         INTSV2  000652R         PRTY7 = 000340          STAT    000026R
AWAS    000110R         CSRX    000326R         ISRLX   000302R         PS      = 177776        SVR0    000062R
BARX    000330R         DATCKS= 104411          ISRX    000300RG        PSW     = 177776        SVR1    000064R
BCRX    000332R         DATERS= 104404          ITR1    000524R         PUSH    = 005746        SVR2    000066R
BEGIN   000000R         DIR     000324R         ITR2    000660R         PUSH2   = 024646        SVR3    000070R
BIT0    = 000001        DUR     000322RG        LOOP    001252R         RANDS   = 104417        SVR4    000072R
BIT1    = 000002        DVICE   000262R         MAP22S= 104416          RANNUM  000054R         SVR5    000074R
BIT10   = 002000        DVID1   000014R         MASK0   000312RG        RBUFEA  000130R         SVR6    000076R
BIT11   = 004000        ENDITS= 104413          MASK1   000314RG        RBUFPA  000126R         SYSCNT  000052R
BIT12   = 010000        ENDS    = 104410        MCRX    000334R         RBUFSZ  000132R         TRPDFD= 000022
BIT13   = 020000        ERRTYP  000106R         MODNAM  000000R         RBUFVA  000124R         VECTOR  000010R
BIT14   = 040000        EXITS   = 104400        MODSP   000252R         READ    001364R         WAIT    001226R
BIT15   = 100000        FREE    000150R         MSGNS   = 104403        RESTRT  000372R         WASADR  000104R
BIT2    = 000004        FUNC    000260R         MSGSS   = 104402        RES1    000356R         WBUFEA  000136R
BIT3    = 000010        GETPAS= 104415          MSGS    = 104401        RES2    000360R         WBUFPA  000134R
BIT4    = 000020        GWBUFS= 104414          MSK     000264R         RSTRT   000112R         WBUFRQ  000140R
BIT5    = 000040        HRDCNT  000044R         NULL    = 000000        R6      =%000006        WBUFSZ  000142R
BIT6    = 000100        HRDERS= 104405          OPEN    = 000000        R7      =%000007        WCN11   000252R
BIT7    = 000200        HRDPAS  000050R         OPOAS   = 104420        SBADR   000102R         WCN12   000254R
BIT8    = 000400        ICONT   000036R         PASCNT  000034R         SET     000356R         WDEB    000116R
BIT9    = 001000        ICOUNT  000040R         PIRQS   = 000004        SETUP   000716R         WDTO    000114R
BREAKS= 104407          ICRX    000310RG        POPSP   = 005726        SOFCNT  000042R         WRITE   001274R
BR1     000012R         IDNUM   000122R         POPSP2= 022626          SOFERS= 104406          XFLAG   000005R
BR2     000013R         IDRX    000316RG        PPR     000320RG        SOFPAS  000046R         XMEM    000256R
BTODS = 104421          IIRHX   000276RG        PRTY    = 000000        SPOINT  000032R

. ABS.  000000      000
        003544      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  6962 WORDS  ( 28 PAGES)
DYNAMIC MEMORY:  8102 WORDS  ( 31 PAGES)
ELAPSED TIME:  00:00:14
```

TABLE OF CONTENTS

DEC/X11-1 SYSTEM EXERCISER MACRO DEFINITION MODULE

```
                        .REM    %
```

IDENTIFICATION

PRODUCT CODE: AC-T124A-MC

PRODUCT NAME: RM80 MODULE

PRODUCT DATE: JANUARY 1982

MAINTAINER: CX DIAGNOSTIC GROUP

AUTHOR: MIKE LEAVITT

1. ABSTRACT

CXRNAA0 IS AN IOMODX THAT EXERCISES RM80 DISK DRIVES ON AN RH70 CONTROLLER. IT EXERCISES THE DRIVES BY DOING SEEKS, WRITES, WRITE-CHECKS, READS, AND IN-CORE COMPARISONS. ALL ERRORS DETECTED ARE REPORTED ON THE CONSOLE TTY.

2. REQUIREMENTS

PDP-11 PROCESSOR
PROGRAM LOADING DEVICE
TERMINAL
RH70 CONTROLLER
1 TO 8 DISK DRIVES (RM80'S)

TEST MEDIA:

THE RM80 DISK PACK(HDA) MUST BE FORMATTED IN 16 BIT DATA FORMAT.

LOAD DEVICE:

ANY DEVICE THAT IS SUPPORTED BY THE XXDP SOFTWARE PACKAGE.

3. START-UP

ON THE INITIAL START, THE PROGRAM WILL CLEAR BIT0 OF 'SR1' AND TYPE THE FOLLOWING MESSAGES.

```
'IF YOU WISH TO EXERCISE THE WHOLE DISK, SET BIT0
 IN SWITCH REGISTER 1(SR1) EQUAL TO 1.'

'RNAA0  ! OPERATING ON FE CYLINDERS ONLY !'
```

THIS WILL OCCUR REGUARDLESS OF THE CONDITION OF SR1 (BIT0) AT CONFIGURE TIME.

IF THE OPERATOR WISHES TO EXERCISE THE DRIVE ON ANY ADDRESS OUTSIDE THE "FE" CYLINDERS, SR1 (BIT0) MUST BE MODIFIED AT LOCATION 16 OF CXRNAA0 MODULE (SEE SECTION 8). THIS CAN BE ACCOMPLISHED BY USING THE 'MOD' COMMAND SUPPLIED BY THE DECX RUN TIME SYSTEM. UNLESS THE PROGRAM IS RELOADED OR THE OPERATOR MODIFIES THE LOCATION AGAIN, THE CONTENTS OF SR1 WILL REMAIN THE SAME ON ALL SUBSEQUENT STARTS.

ON ALL SUBSEQUENT STARTS, THE CONDITION OF SR1 (BIT0) WILL TYPE TO TERMINAL IN THE FOLLOWING MANOR.

IF BIT0 OF SR1 IS EQUAL TO 0 (ZERO), THE FOLLOWING WARNING WILL BE TYPED.

```
'RNAA0  ! OPERATING ON FE CYLINDERS ONLY !'
```


IF BIT0 OF SR1 IS EQUAL TO 1 (ONE), THE FOLLOWING WARNING WILL BE TYPED.

```
'RNAA0  ! CUSTOMER DATA WILL BE OVERWRITTEN !'
```

4. PASS DEFINITION

ONE PASS OF THE CXRNAA0 MODULE CONSISTS OF 200 CYCLES OF THE FOLLOWING TEST SEQUENCE FROM STEP 1 TO STEP 9.

1. SELECT A STARTING BLOCK ADDRESS. (SEE SECTION 11.)
2. SELECT A DRIVE FOR TEST. (ROUND-ROBIN)
3. IF NO DRIVE IS SELECTED, DROP THE CXRNAA0 MODULE.
4. DO AN EXPLICIT SEEK TO A CYLINDER. IF ERRORS, REPORT AND RETRY UP TO RETRY LIMIT.
5. DO A WRITE DATA - IF ERRORS, REPORT AND RETRY UP TO RETRY LIMIT.
6. DO A WRITE CHECK DATA - IF ERRORS, REPORT AND RETRY UP TO RETRY LIMIT.
7. DO A READ DATA - IF ERRORS, REPORT AND RETRY UP TO RETRY LIMIT.
8. DO AN IN-CORE DATA COMPARE - IF ERRORS, REPORT.
9. IF COMPLETED CYCLE OF ALL DRIVES, GO TO STEP 1, ELSE STEP 2.

NOTE: THE RETRY LIMIT IS 3 TIMES. THE MODULE CLEARS THE ERROR, BEFORE EACH RETRY. IF THE MODULE FAILS TO CLEAR THE ERROR, THE MODULE WILL BE DROPPED.

5. EXECUTION TIME

ONE PASS OF CXRNAA0 RUNNING ALONE ON A PDP-11/70 TAKES APPROXIMATELY 20 SECONDS.

6. CONFIGURATION REQUIREMENTS

DEFAULT PARAMETERS:

DEVADR: 176700, VECTOR: 254, BR1: 5, DEVCNT: 1

REQUIRED PARAMETERS:

NONE

7. DEVICE/OPTION SETUP

```
                        MAKE CERTAIN THAT ALL DRIVES ARE POWERED UP, WRITE ENABLED, AND READY


    8.      OPERATION OPTIONS
            -----------------

            MEANING OF SR1:

                    BIT00 = 1 ALLOW ACCESS TO ANY CYLINDER ADDRESS
                    BIT00 = 0 ALLOW ACCESS TO FE CYLINDER ADDRESSES ONLY

                    BIT02 = 1 COUNT DATA LATE ERRORS, BUT DON'T TYPE THEM.
                    BIT02 = 0 COUNT AND TYPE DATA LATE ERRORS.

                    BIT04 = 1 PORT B SELECTED
                    BIT04 = 0 PORT A SELECTED

                    BIT10 = 1 SELECT RANDOM BLOCK ADDRESSING.
                    BIT10 = 0 SELECT SEQUENTIAL BLOCK ADDRESSING.

                    BIT12 = 1 DUAL PORT OPERATION.
                    BIT12 = 0 SINGLE PORT OPERATION.

                    BIT15 = 1 32 REGISTERS ON RH70
                    BIT15 = 0 22 REGISTERS ON RH70

            NOTE:   BIT04 HAS NO MEANING IN SINGLE PORT OPERATION.


    9.      NON-STANDARD PRINTOUTS
            ----------------------

            A.  MOST PRINTOUTS HAVE THE STANDARD FORMATS DESCRIBED IN
                THE DEC/X11 DOCUMENT.

            B.  ERROR MESSAGES DUMP THE CONTENTS OF THE 20 OR 22 REGISTERS.
                (DEPENDING ON WHICH RH CONTROLLER IS USED)

                IN THE FOLLOWING ORDER:

                RMCS1  RMWC   RMBA   RMDA   RMCS2  RMDS   RMER1  RMAS
                RMLA   RMDB   RMMR1  RMDT   RMSN   RMOF   RMDC   RMHR
                RMMR2  RMER2  RMEC1  RMEC2 *RMBAE *RMCS3

                * PRINTED ON ERROR REPORT IF USING AN RH70 CONTROLLER


    10.     DUAL PORT SETUP:
            ----------------
            TO RUN A DUAL PORT SYSTEM, SR1 HAS TO BE MODIFIED TO INDICATE TO
            THE MODULE, WHICH PORT THE MODULE IS LOCATED ON. (SEE SECTION 8)
            FOR SR1 OPTIONS.

            THE CONTROLLER SELECT SWITCH ON THE RM DRIVE MUST BE IN THE A/B
            POSITION.  THIS SWITCH IS ACTIVATED WHEN THE DRIVE IS CYCLED UP.
            IF THE SWITCH WAS NOT IN THIS POSITION WHEN DRIVE WAS POWERED UP,
```



```
            THE FOLLOWING STEPS MUST BE TAKEN.  PLACE THE DUAL PORT SELECT
            SWITCH IN THE A/B POSITION, CYCLE THE DRIVE DOWN TO THE LOAD
            STATE AND THEN CYCLE THE DRIVE UP AND WAIT FOR READY. THIS WILL
            PUT THE DRIVE IN THE DUAL PORT MODE.


    11.     BLOCK ADDRESS
            -------------

            EACH SECTOR ON THE RM80 PACK IS CALLED A BLOCK.

            AT THE BEGINNING OF EACH TEST CYCLE, THE CXRNAA0 MODULE SELECTS
            A STARTING BLOCK ADDRESS SEQUENTIALLY OR RANDOMLY FOR EACH
            DATA TRANSFER.
            WHEN BIT10 OF SR1 IS RESET(0), CXRNAA0 MODULE SELECTS THE BLOCK
            ADDRESS SEQUENTIALLY FROM BLOCK 0 TO THE LAST BLOCK.
            WHEN BIT10 OF SR1 IS SET(1), CXRNAA0 MODULE SELECTS THE BLOCK
            ADDRESS RANDOMLY BETWEEN BLOCK 0 AND THE LAST BLOCK.

            IN SINGLE PORT OPERATION, CXRNAA0 MODULE ACCESSES EVERY BLOCK
            ON TRACKS 0 THRU 13.

            IN DUAL PORT OPERATION, CXRNAA0 MODULE ON THE PORT A ACCESSES
            EVERY BLOCK ON TRACKS 0 THRU 6.

            IN DUAL PORT OPERATION, CXRNAA0 MODULE ON THE PORT B ACCESSES
            EVERY BLOCK ON TRACKS 7. THRU 13.

            NOTE:   THE MANUFACTURE/USER BAD SECTOR FILES AND  THE SKIP
                    SECTOR  FILE  WILL NEVER  BE ACCESSED IN ANY WAY BY
                    THIS MODULE.


    12.     BAD SECTOR TABLE
            ----------------

            32. LOCATIONS STARTING AT THE LABEL "BADSPT" ARE RESERVED
            FOR 16. BAD SECTOR ENTRIES BY THE USER. EACH BAD SECTOR IS
            SPECIFIED THROUGH ENTERING ITS CYLINDER ADDRESS TO THE
            FIRST WORD AND ITS TRACK AND SECTOR ADDRESS TO THE SECOND
            WORD; THE HIGH BYTE ON THE SECOND WORD IS THE TRACK ADDRESS
            WHILE THE LOW BYTE ON THE SECOND WORD IS THE SECTOR ADDRESS.
            WHEN THE MODULE DETECTS A DATA TRANSFER ERROR, ON ANY OF
            THESE SPOTS, THE MODULE WILL SKIP OVER THE BAD SECTOR AND
            START ANOTHER TEST CYCLE.

            THE MODULE ALSO DETECTS BAD SECTORS THROUGH THE "BSE" ERROR BIT
            (BIT15 OF RMER2). WHENEVER THE "BSE" BIT IS SET AFTER EXECUTING
            A DATA TRANSFER COMMAND, THE MODULE WILL IGNORE THE ERROR.

            THE MODULE ALSO DETECTS BAD SECTORS THROUGH THE "SSE" ERROR BIT
            (BIT05 OF RMER2). WHENEVER THE "SSE" BIT IS SET AFTER EXECUTING
            A DATA TRANSFER COMMAND, THE MODULE WILL IGNORE THE ERROR, SET
            "SSEI" (BIT09 OF RMOF), INCREMENT THE DISK ADDRESS BY ONE SECTOR
            AND EXECUTE THE DATA COMMAND AGAIN.
```

```
229
230
231
232
233
234                                     *
```



```
1                                       ;LAST REVISION 04-AUG-81
6                                               .TITLE  RNAA DEC/X11 SYSTEM EXERCISER MODULE
                                        ;       DDXCOM  VERSION  6       23-MAY-78
                                                .LIST   BIN
                                        ;*****************************************************************************
000000                                  BEGIN:
000000     122     116     101          MODNAM: .ASCII  /RNAA / ;MODULE NAME.
000005     000                          XFLAG:  .BYTE   0PEN            ;USED TO KEEP TRACK OF WBUFF USAGE
000006  176700                          ADDR:   176700+0                ;1ST DEVICE ADDR.
000010  000254                          VECTOR: 254+0                   ;1ST DEVICE VECTOR.
000012     240                          BR1:    .BYTE   PR1Y5+0         ;1ST BR LEVEL.
000013     000                          BR2:    .BYTE   PRTY0+0         ;2ND BR LEVEL.
000014  000001                          DVID1:  0+1                     ;DEVICE INDICATOR 1.
000016  000000                          SR1:    OPEN                    ;SWITCH REGISTER 1
000020  000000                          SR2:    OPEN                    ;SWITCH REGISTER 2
000022  000000                          SR3:    OPEN                    ;SWITCH REGISTER 3
000024  000000                          SR4:    OPEN                    ;SWITCH REGISTER 4
                                        ;*****************************************************************************
000026  150000                          STAT:   150000                  ;STATUS WORD.
000030  001662'                         INIT:   START                   ;MODULE START ADDR.
000032  000252'                         SPOINT: MODSP                   ;MODULE STACK POINTER.
000034  000000                          PASCNT: 0                       ;PASS COUNTER.
000036  000310                          ICONT:  200.                    ;# OF ITERATIONS PER PASS=200.
000040  000000                          ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
000042  000000                          SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
000044  000000                          HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
000046  000000                          SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
000050  000000                          HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
000052  000000                          SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
000054  000000                          RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000056                                  CONFIG:                         ;RESERVED FOR MONITOR USE
000056  000000                          RES1:   0                       ;RESERVED FOR MONITOR USE
000060  000000                          RES2:   0                       ;RESERVED FOR MONITOR USE
000062  000000                          SVR0:   OPEN                    ;LOC TO SAVE R0.
000064  000000                          SVR1:   OPEN                    ;LOC TO SAVE R1.
000066  000000                          SVR2:   OPEN                    ;LOC TO SAVE R2.
000070  000000                          SVR3:   OPEN                    ;LOC TO SAVE R3.
000072  000000                          SVR4:   OPEN                    ;LOC TO SAVE R4.
000074  000000                          SVR5:   OPEN                    ;LOC TO SAVE R5.
000076  000000                          SVR6:   OPEN                    ;LOC TO SAVE R6.
000100  000000                          CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
000102                                  SBADR:                          ;ADDR OF GOOD DATA, OR
000102  000000                          ACSR:   OPEN                    ;CONTENTS OF CSR.
000104                                  WASADR:                         ;ADDR OF BAD DATA, OR
000104  000000                          ASTAT:  OPEN                    ;STATUS REG CONTENTS.
000106                                  ERRTYP:                         ;TYPE OF ERROR
000106  000000                          ASB:    OPEN                    ;EXPECTED DATA.
000110  000000                          AWAS:   OPEN                    ;ACTUAL DATA.
000112  002164'                         RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
000114  000000                          WDTO:   OPEN                    ;#WORDS TO MEMORY PER ITERATION
000116  000000                          WDFR:   OPEN                    ;#WORDS FROM MEMORY PER ITERATION
000120  000000                          INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
000122  000000                          IDNUM:  0                       ;MODULE IDENTIFICATION NUMBER=0
000124  000424'                         RBUFVA: RBUF                    ;READ BUFFER VIRTUAL ADDRESS
000126  000000                          RBUFPA: OPEN                    ;READ BUFFER PHYSICAL ADDRESS
000130  000000                          RBUFEA: OPEN                    ;READ BUFFER EA BITS
000132  000400                          RBUFSZ: 256.                    ;SIZE OF THE READ BUFFER
```

```
000134  000000                  $BUFPA: OPEN                    ;$RITE BUFFER PHYSICAL ADDRESS
000136  000000                  $BUFEA: OPEN                    ;$RITE BUFFER EA BITS
000140  000400                  $BUFRQ: 256.                    ;$RITE BUFFER SIZE REQUESTED
000142  000000                  $BUFSZ: OPEN                    ;$RITE BUFFER SIZE AVAILABLE
000144  000000                  CDERCT: OPEN                    ;CDATA/DATCK ERROR COUNT
000146  000000                  CDWDCT: OPEN                    ;CDATA/DATCK $ORD COUNT
000150  000000                  FREE:   OPEN                    ;RESERVED FOR FUTURE USE
        000040                          .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
000252                          MODSP:
                                ;*****************************************************************************

                                ;USER DEFINED BAD SPOT TABLE
000252  177777                  BADSPI: .$ORD   177777
000254  177777                          .$ORD   177777
000256  177777                          .$ORD   177777
000260  177777                          .$ORD   177777
000262  177777                          .$ORD   177777
000264  177777                          .$ORD   177777
000266  177777                          .$ORD   177777
000270  177777                          .$ORD   177777
000272  177777                          .$ORD   177777
000274  177777                          .$ORD   177777
000276  177777                          .$ORD   177777
000300  177777                          .$ORD   177777
000302  177777                          .$ORD   177777
000304  177777                          .$ORD   177777
000306  177777                          .$ORD   177777
000310  177777                          .$ORD   177777
000312  177777                          .$ORD   177777
000314  177777                          .$ORD   177777
000316  177777                          .$ORD   177777
000320  177777                          .$ORD   177777
000322  177777                          .$ORD   177777
000324  177777                          .$ORD   177777
000326  177777                          .$ORD   177777
000330  177777                          .$ORD   177777
000332  177777                          .$ORD   177777
000334  177777                          .$ORD   177777
000336  177777                          .$ORD   177777
000340  177777                          .$ORD   177777
000342  177777                          .$ORD   177777
000344  177777                          .$ORD   177777
000346  177777                          .$ORD   177777
000350  177777                          .$ORD   177777

                                ;DO NOT CHANGE THE ORDER OF THE NEXT 4 LOCATIONS, THEY ARE
                                ;NEEDED FOR MAP22 ROUTINE.
000352  000000                  PA18:   0
000354  000000                  XMEM:   0
000356  000000                  PA22:   0
000360  000000                  EA22:   0

000362  000000                  DRPDRV: 0               ;DROP DRIVE = 1
000364  000000                  FLAG70: 0               ;BIT15 SET INDICATES THAT AN 11/70 PRESENT
000366  000000                  CYLIND: 0               ;CYLINDER ADDRESS FOR DRIVER
000370  000000                  DSKADR: 0               ;COMPOSITE TRACK AND SECTOR ADDRESS FOR DRIVER
000372  000000                  DVICE:  0               ;DEVICE NUMBER
```



```
000374  000000                  UNITNO: 0               ;# OF UNIT BEING TESTED
000376  000000                  BADSEC: 0               ;BAD SECTOR FLAG, USER BAD SECTOR DETECTED = -1,
                                                        ;BSE = 100000 AND SSE = 40
000400  000000                  DLTCNT: 0               ;DATA LATE ERROR COUNT
000402  000000                  FUNC:   0               ;FUNCTION TO BE PERFORMED
000404  000000                  ZERO:   0
000406  000000                  FERADR: 0
000410  000000                  CLK:    0               ;USED IN TIMEOUT LOOPS
000412  000000                  TRY:    0
000414  000000                  DRVTYP: 0               ;DRIVE TYPE FLAG, RM80=26
000416  000000                  RMOFI:  0               ;INPUT STORAGE FOR OFFSET REGISTER
000420  000000                  TRKMAP: 0               ;MAPPING FACTOR FOR TRACK ADDRESS
000422  000000                  CYLMAP: 0               ;MAPPING FACTOR FOR CYLINDER ADDRESS

000424                          RBUF:   .BLKW   256.

                                ;RH/RM SAVED REGISTER POINTERS

001424  001606'                 TABLE:  S+NCS1
001426  001610'                         S+NWC
001430  001612'                         S+NBA
001432  001614'                         S+NDA
001434  001616'                         S+NCS2
001436  001620'                         S+NDS
001440  001622'                         S+NER1
001442  001624'                         S+NAS
001444  001626'                         S+NLA
001446  001630'                         S+NDB
001450  001632'                         S+NMR1
001452  001634'                         S+NDT
001454  001636'                         S+NSN
001456  001640'                         S+NOF
001460  001642'                         S+NDC
001462  001644'                         S+NHR
001464  001646'                         S+NMR2
001466  001650'                         S+NER2
001470  001652'                         S+NEC1
001472  001654'                         S+NEC2
001474  001656'                         S+NBAE
001476  001660'                         S+NCS3
001500  177777                          177777

001502  000000                  RMCS1:  0               ;CONTROL STATUS REGISTER #1
001504  000000                  RMWC:   0               ;$ORD COUNT REGISTER
001506  000000                  RMBA:   0               ;BUS ADDRESS REGISTER
001510  000000                  RMDA:   0               ;DESIRED TRACK/SECTOR ADDRESS REGISTER
001512  000000                  RMCS2:  0               ;CONTROL STATUS REGISTER #2
001514  000000                  RMDS:   0               ;DRIVE STATUS REGISTER
001516  000000                  RMER1:  0               ;ERROR REGISTER #1
001520  000000                  RMAS:   0               ;ATTENTION SUMMARY REGISTER
001522  000000                  RMLA:   0               ;LOOK AHEAD REGISTER
001524  000000                  RMDB:   0               ;DATA BUFFER REGISTER
001526  000000                  RMMR1:  0               ;MAINTENANCE REGISTER #1
001530  000000                  RMDT:   0               ;DRIVE TYPE REGISTER
001532  000000                  RMSN:   0               ;SERIAL NUMBER REGISTER
001534  000000                  RMOF:   0               ;OFFSET REGISTER
001536  000000                  RMDC:   0               ;DESIRED CYLINDER ADDRESS REGISTER
```

```
 87 001540  000000                 RMHR:   0               ;HOLDING REGISTER
 88 001542  000000                 RMMR2:  0               ;MAINTENANCE REGISTER #2
 89 001544  000000                 RMER2:  0               ;ERROR REGISTER #2
 90 001546  000000                 RMEC1:  0               ;ECC POSITION REGISTER
 91 001550  000000                 RMEC2:  0               ;ECC PATTERN REGISTER
 92 001552  000000                 RMBAE:  0               ;BUS ADDRESS EXTENSION REGISTER (RH70)
 93 001554  000000                 RMCS3:  0               ;CONTROL STATUS REGISTER #3 (RH70)
 94
 95 001556  000406'                XFERAD: FERADR
 96 001560  177777                         177777
 97
 98                                 ;REGISTER ADDRESS INDEX VALUE
 99
100          000000                NCS1    = 0
101          000002                NWC     = 2
102          000004                NBA     = 4
103          000006                NDA     = 6
104          000010                NCS2    = 10
105          000012                NDS     = 12
106          000014                NER1    = 14
107          000016                NAS     = 16
108          000020                NLA     = 20
109          000022                NDB     = 22
110          000024                NMR1    = 24
111          000026                NDT     = 26
112          000030                NSN     = 30
113          000032                NOF     = 32
114          000034                NDC     = 34
115          000036                NHR     = 36
116          000040                NMR2    = 40
117          000042                NER2    = 42
118          000044                NEC1    = 44
119          000046                NEC2    = 46
120          000050                NBAE    = 50
121          000052                NCS3    = 52
122
123                                 ;DISK ADDRESS EQUATES FOR LIMITS TABLES
124
125          000000                FT      = 0             ;FIRST TRACK
126          000002                LT      = 2             ;LAST TRACK
127          000004                LS      = 4             ;LAST SECTOR
128          000006                LC      = 6             ;LAST CYLINDER
129          000010                BST     = 10            ;BAD SECTOR TRACK (DEC 144 FILE)
130          000012                SEC     = 12            ;CURRENT SECTOR
131          000014                TRK     = 14            ;CURRENT TRACK
132          000016                CYL     = 16            ;CURRENT CYLINDER
133          000020                FFE     = 20            ;FIRST FE CYLINDER
134          000022                LFE     = 22            ;LAST FE CYLINDER
135
136                                 ;TABLE OF ADDRESS LIMITS FOR RM80
137
138                                 ;NOTES: (1) WORD 1 (FIRST TRACK) AND WORD 2 (LAST TRACK) WILL BE
139                                 ;           INITIALIZED AT PROGRAM START.
140                                 ;
141                                 ;       (2) DO NOT ALTER THE BAD SECTOR TRACK OR LAST CYLINDER
142                                 ;           VALUES. IF EITHER VALUE IS CHANGED, THE BAD SECTOR
143                                 ;           FILE(DEC 144 FILE) MAY BE DESTROYED.
```



```
144
145 001562  000000                 TAB80:  .WORD   0       ;FIRST TRACK
146 001564  000000                         .WORD   0       ;LAST TRACK
147 001566  000036                         .WORD   30.     ;LAST SECTOR
148 001570  001056                         .WORD   558.    ;LAST CYLINDER
149 001572  000015                         .WORD   13.     ;BAD SECTOR TRACK (DEC 144 FILE)
150 001574  000000                         .WORD   0       ;CURRENT SECTOR ADDRESS
151 001576  000000                         .WORD   0       ;CURRENT TRACK ADDRESS
152 001600  000000                         .WORD   0       ;CURRENT CYLINDER ADDRESS
153 001602  001057                         .WORD   559.    ;FIRST FE CYLINDER
154 001604  001060                         .WORD   560.    ;LAST FE CYLINDER
155
156 001606                         S:      .BLKW   22.     ;STORE THE RH/RM REGISTER CONTENTS
157
158                                 ;CONTROL STATUS REGISTER #1
159
160          100000                SC      = 100000        ;SPECIAL CONDITION
161          040000                TRE     = 40000         ;TRANSFER ERROR
162          020000                MCPE    = 20000         ;CONTROL BUS PARITY ERROR
163          004000                DVA     = 4000          ;DRIVE AVAILABLE
164          000200                RDY     = 200           ;CONTROL READY
165          000001                GO      = 1             ;GO BIT
166
167                                 ;CONTROL STATUS REGISTER #2
168
169          000040                CLR     = 40            ;MASSBUS CLEAR COMMAND
170          000100                IR      = 100           ;SILO INPUT READY
171          000200                OR      = 200           ;SILO OUTPUT READY
172          000400                MDPE    = 400           ;MASS BUS PARITY
173          001000                MXF     = 1000          ;MISSED TRANSFER ERROR
174          002000                PGE     = 2000          ;PROGRAM ERROR (RH)
175          004000                NEM     = 4000          ;NON-EXISTENCE MEMORY
176          010000                NED     = 10000         ;NON-EXISTENCE DRIVE
177          020000                UPE     = 20000         ;UNIBUS PARITY ERROR
178          040000                WCE     = 40000         ;WRITE CHECK ERROR
179          100000                DTE     = 100000        ;DATA LATE ERROR
180
181                                 ;RM DRIVE STATUS REGISTER
182
183          000001                OM      = 1             ;OFFSET MODE
184          000100                VV      = 100           ;VOLUMNE VALID
185          000200                DRY     = 200           ;DRIVE READY
186          000400                DPR     = 400           ;DRIVE PRESENT
187          001000                PGM     = 1000          ;PROGRAMMABLE STATE
188          002000                LBT     = 2000          ;LAST BLOCK TRANSFERRED
189          004000                WRL     = 4000          ;WRITE LOCK
190          010000                MOL     = 10000         ;MEDIUM ONLINE
191          020000                PIP     = 20000         ;PROSITIONING OPERATION IN PROGRESS
192          040000                ERR     = 40000         ;DRIVE ERROR
193          100000                ATA     = 100000        ;DRIVE ATTENTION
194
195
196                                 ;RM ERROR REGISTER #1
197
198          000001                ILF     = 1             ;ILLEGAL FUNCTION
199          000002                ILR     = 2             ;ILLEGAL REGISTER
200          000004                RMR     = 4             ;REGISTER MODIFICATION REFUSE
```

```
201          000020                  FER     = 20            ;FORMAT ERROR
202          000040                  WCF     = 40            ;WRITE CLOCK FAIL
203          000100                  ECH     = 100           ;HARD ERROR ECC
204          000200                  HCE     = 200           ;HEADER COMPARE FAIL
205          000400                  HCRC    = 400           ;HEADER CRC ERROR
206          001000                  AOE     = 1000          ;ADDRESS OVERLOW ERROR
207          002000                  IAE     = 2000          ;INVALID ADDRESS ERROR
208          004000                  WLE     = 4000          ;WRITE LOCK ERROR
209          010000                  DTE     = 10000         ;DRIVE TIMING ERROR
210          020000                  OPI     = 20000         ;OPERATION INCOMPLETE
211          040000                  UNS     = 40000         ;DRIVE UNSAFE
212          100000                  DCK     = 100000        ;DATA CHECK ERROR
213
214                                  ;RM ERROR REGISTER #2
215
216          000010                  DPE     = 10            ;DATA PARITY DURING WRITE
217          000040                  SSE     = 40            ;SKIP SECTOR ERROR
218          000200                  DVC     = 200           ;DEVICE CHECK
219          002000                  LBC     = 2000          ;LOST BIT CLOCK
220          004000                  LSC     = 4000          ;LOST SYSTEM CLOCK
221          040000                  SKI     = 40000         ;SEEK INCOMPLETE
222          100000                  BSE     = 100000        ;BAD SECTOR ERROR
223
224                                  ;RM OFFSET REGISTER
225
226          010000                  FMT     = 10000         ;16 BIT DATA FORMAT
227          001000                  SSEI    = 1000          ;SKIP SECTOR ERROR INHIBIT
```



```
  1                                     .SBTTL  START OF PROGRAM
  2
  3 001662  012767  000400  176224  START:  MOV     #256.,WDTO      ;256 WORDS TO MEM/ITERATION
  4 001670  012767  000400  176220          MOV     #256.,WDFM      ;256 WORDS FROM MEM/ITERATION
  5 001676  012767  000012  176214          MOV     #10.,INTR       ;10 INTERRUPTS/ITERATION
  6 001704  016767  176104  176460          MOV     DV101,DVICE     ;GET DRIVES TO TEST
  7 001712  016706  176114                  MOV     SPOINT,R6       ;INITIATE STACK POINT
 20
 21                                     ;SETUP DRIVE PARAMETERS TABLE ACCORDING TO 'SR1' CONTENTS, ISSUE A
 22                                     ;PACK ACKNOWLEDGE AND RECALIBRATE COMMAND TO DRIVES
 23
 24 001716  012700  001562'        SETBL:  MOV     #TABBO,R0       ;GET POINTER TO TABLE
 25 001722  012760  000007  000000          MOV     #7.,FT(R0)      ;USE THE UPPER PORTION OF THE DISK
 26 002000  012760  000015  000002          MOV     #13.,LT(R0)     ;IF THE DUAL PORT AND PORT B ARE SELECTED
 27 002006  016701  176004                  MOV     SR1,R1          ;GET SWITCH REGISTER #1
 28 002012  042701  167757                  BIC     #^C<BIT12!BIT4>,R1
 29 002016  022701  010020                  CMP     #BIT12!BIT4,R1  ;DUAL PORT AND PORT B ?
 30 002022  001411                          BEQ     1$              ;BR IF YES
 31 002024  012760  000000  000000          MOV     #0,FT(R0)       ;OTHERWISE SET THE FIRST TRACK TO 0
 32 002032  032701  010000                  BIT     #BIT12,R1       ;DUAL PORT OPERATION ?
 33 002036  001403                          BEQ     1$              ;BR IF NO
 34 002040  012760  000006  000002          MOV     #6.,LT(R0)      ;SET UP THE UPPER TRACK LIMIT
 35
 36 002046  012760  177777  000012  1$:     MOV     #-1,SEC(R0)     ;INTIALIZE THE SECTOR ADDRESS
 37 002054  011060  000014                  MOV     (R0),TRK(R0)    ;INITIALIZE THE TRACK ADDRESS
 38 002060  012760  000000  000016          MOV     #0,CYL(R0)      ;INITIALIZE THE CYLINDER ADDRESS
 39 002066  032767  000001  175722          BIT     #BIT0,SR1       ;FE CYLINDER ONLY ?
 40 002074  001003                          BNE     2$              ;BR IF NO
 41 002076  016060  000020  000016          MOV     FFE(R0),CYL(R0) ;GET STARTING ADDRESS FOR FE CYLINDERS
 42 002104  005067  176270          2$:     CLR     DLTCNT          ;RESET THE DATA LATE ERROR COUNT
 43 002110  004767  003560                  JSR     PC,SETUP        ;SETUP REGISTER ADDRESSES AND PSEL BIT
 44 002114  012767  177777  176252          MOV     #-1,UNITNO      ;PRE-SET UNIT NUMBER
 45
 46 002122  004767  001502          3$:     JSR     PC,GETDRV       ;GO GET DRIVE
 47 002126  000415                          BR      4$              ;RETURN HERE WHEN ALL DRIVES ARE DONE
 48 002130  004567  003000                  JSR     R5,READY        ;SEE IF DRIVE IS READY FOR COMMANDS
 49 002134  000772                          BR      3$              ;RETURN HERE ON ERROR
 50 002136  012777  000023  177336          MOV     #23,@RMCS1      ;GO ISSUE A PACK ACKNOWLEDGE COMMAND TO
 51                                                                 ;SET "VV" BIT IN DRIVE STATUS.
 52 002144  012777  000007  177330          MOV     #7,@RMCS1       ;ISSUE RECALIBRATE COMMAND AND
 53 002152  004767  003234                  JSR     PC,WICRDY       ;WAIT FOR CONTROLLER READY
 54 002156  000240                          NOP                     ;RETURN HERE IF ERROR
 55 002160  000760                          BR      3$              ;RETURN--DO NEXT DRIVE
 56 002162  000404                  4$:     BR      RSTRT1          ;START THE PASS
 57
 58 002164  005767  175644          RESTRT: TST     PASCNT          ;SUPPORT DT03
 59 002170  001001                          BNE     RSTRT1
 60 002172  000633                          BR      START
 61
 62 002174                          RSTRT1:
    002174  104415  000000' 000124'        GETPAS,BEGIN, RBUFVA            ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
 63
 64 002202  012777  000040  177302  LOOP1:  MOV     #CLR,@RMCS2     ;ISSUE A CLEAR COMMAND
 65 002210  012767  177777  176156          MOV     #-1,UNITNO      ;PRE-SET UNIT NUMBER
 66 002216  004767  000752                  JSR     PC,GENADR       ;GENERATE BLOCK ADDRESS
 67
 68 002222  012767  010000  176166  LOOP2:  MOV     #FMT,RMOFI      ;SET 16 BIT FORMAT
```

```
 69 002230  004767  001374              JSR     PC,GETDRV       ;GO GET DRIVE
 70 002234  000413                      BR      2$              ;RETURNS HERE IF ALL DRIVES DONE
 71 002236  004567  002672              JSR     R5,READY        ;SEE IF DRIVE IS READY
 72 002242  000767                      BR      LOOP2           ;LOOP BACK IF NOT READY
 73
 74 002244  004767  001330              JSR     PC,LDADR        ;LOAD BLOCK ADDRESS
 75 002250  004767  000022              JSR     PC,CYCLE        ;COMMAND SEQUENCE ROUTINE
 76 002254  000240                      NOP                     ;ERROR EXIT FOR CYCLE ROUTINE
 77
 78 002256                          1$:
    002256  104413  000000'            ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
                                                                ;MONITOR SHALL TEST END OF PASS
 79 002262  000757                      BR      LOOP2           ;DO IT TO NEXT DRIVE
 80
 81 002264  005767  176102          2$: TST     DVICE           ;ARE ALL DEVICES DROPPED ?
 82 002270  001344                      BNE     LOOP1           ;BR IF NO
 83
 84 002272  104410  000000'            END$,BEGIN              ;
```



```
  1                                 .SBTTL  SUBROUTINES
  2
  3                                 ;CALL
  4                                 ;       JSR     PC,CYCLE        ;CALL ROUTINE
  5                                 ;       BR      ?               ;ERROR EXIT
  6                                 ;       RET                     ;NORMAL RETURN
  7
  8 002276  005067  176110          CYCLE:  CLR     TRY             ;CLEAR THE RETRY COUNT
  9 002302  104414  000000'                  GWBUF$, BEGIN           ;GET WRITE BUFFER INFORMATION
 10 002306  004767  002736                   JSR     PC,WAIT         ;WAIT FOR DRIVE
 11 002312  000437                           BR      1$              ;EXIT ON ERROR
 12 002314  004567  000334                   JSR     R5,SEEK         ;DO AN EXPLICIT SEEK
 13 002320  000434                           BR      1$              ;EXIT IF SEEK ERROR
 14 002322  004567  000106                   JSR     R5,WRITE        ;WRITE DATA COMMAND
 15 002326  000431                           BR      1$              ;FAILURE EXIT
 16 002330  004767  002714                   JSR     PC,WAIT         ;WAIT FOR DRIVE
 17 002334  000426                           BR      1$              ;EXIT ON ERROR
 18 002336  004567  000312                   JSR     R5,SEEK         ;DO AN EXPLICIT SEEK
 19 002342  000423                           BR      1$              ;EXIT IF SEEK ERROR
 20 002344  004567  000144                   JSR     R5,WRITCK       ;WRITE CHECK DATA COMMAND
 21 002350  000420                           BR      1$              ;FAILURE EXIT
 22 002352  004767  002672                   JSR     PC,WAIT         ;WAIT FOR DRIVE
 23 002356  000415                           BR      1$              ;EXIT ON ERROR
 24 002360  004567  000270                   JSR     R5,SEEK         ;DO AN EXPLICIT SEEK
 25 002364  000412                           BR      1$              ;EXIT IF SEEK ERROR
 26 002366  004567  000202                   JSR     R5,READ         ;READ DATA COMMAND
 27 002372  000407                           BR      1$              ;FAILURE EXIT
 28 002374  104412  000000' 000126'          CDATA$,BEGIN,RBUFPA     ; REQUEST FOR MONITOR TO CHECK DATA
    002402  002432'                          2$                      ; IF ERROR, RETURN AT TAG 2$
 29 002404  062716  000002                   ADD     #2,(SP)         ;ADJUST GOOD RETURN FOR CYCLE ROUTINE
 30 002410  000410                           BR      2$              ;EXIT
 31
 32 002412  005767  175744          1$:     TST     DRPDRV          ;DROP THE DRIVE ?
 33 002416  001405                          BEQ     2$              ;BR IF NO
 34 002420  004767  001252                   JSR     PC,DROP         ;DROP THE DRIVE
 35 002424  104403  000000' 007522'          MSG$,BEGIN,DRM ;ASCII MESSAGE CALL WITH COMMON HEADER
 36 002432  000207                      2$: RTS     PC              ;EXIT
 37
```

```
                                .SBTTL  COMMAND SUBROUTINES

                                ;CALL
                                ;
                                ;       JSR     R5,COMMAND      ;CALL COMMAND ROUTINE
                                ;       BR      ?               ;ERROR RETURN
                                ;       RET                     ;NORMAL RETURN


                                ;THE ROUTINE COMMAND NAMES:
                                ;WRITE  - WRITE DATA, USING WRITE BUFFER,
                                ;WRITCK - WRITE CHECK DATA, USING WRITE BUFFER,
                                ;READ   - READ DATA, USING READ BUFFER
                                ;SEEK   - SEEK COMMAND

002434  012767  000161  175740  WRITE:  MOV     #161,FUNC       ;LOAD WRITE FUNCTION
002442  012767  002434' 175736          MOV     #WRITE,FERADR   ;SAVE WHERE WE WERE
002450  016746  175466                  MOV     WBUFSZ,-(SP)    ;GET WRITE SIZE
002454  005416                          NEG     (SP)            ;NEGATE IT
002456  012677  177022                  MOV     (SP)+,@RMWC     ;LOAD WORD COUNT
002462  016777  175446  177016          MOV     WBUFPA,@RMBA    ;LOAD BUFFER ADDRESS
002470  016746  175442                  MOV     WBUFEA,-(SP)    ;GET EXTENDED MEMORY BITS
002474  006316                          ASL     (SP)            ;SHIFT 4 PLACES TO THE LEFT
002476  006316                          ASL     (SP)            ;TO LINE UP WITH RMCS1
002500  006316                          ASL     (SP)
002502  006316                          ASL     (SP)
002504  012667  175644                  MOV     (SP)+,XMEM      ;SAVE THE SHIFTED BITS
002510  000167  000160                  JMP     GO1             ;CONTINUE

002514  012767  000151  175660  WRITCK: MOV     #151,FUNC       ;LOAD WRITE-CHECK FUNCTION
002522  012767  002514' 175656          MOV     #WRITCK,FERADR  ;SAVE WHERE WE WERE
002530  016746  175406                  MOV     WBUFSZ,-(SP)    ;GET WRITE SIZE
002534  005416                          NEG     (SP)            ;NEGATE IT
002536  012677  176742                  MOV     (SP)+,@RMWC     ;LOAD WORD COUNT
002542  016777  175366  176736          MOV     WBUFPA,@RMBA    ;LOAD BUFFER ADDRESS
002550  016746  175362                  MOV     WBUFEA,-(SP)    ;GET EXTENDED MEMORY BITS
002554  006316                          ASL     (SP)            ;SHIFT 4 PLACES TO THE LEFT
002556  006316                          ASL     (SP)            ;TO LINE UP WITH RMCS1
002560  006316                          ASL     (SP)
002562  006316                          ASL     (SP)
002564  012667  175564                  MOV     (SP)+,XMEM      ;SAVE THE SHIFTED BITS
002570  000167  000100                  JMP     GO1             ;CONTINUE

002574  012767  000171  175600  READ:   MOV     #171,FUNC       ;LOAD READ FUNCTION
002602  012767  002574' 175576          MOV     #READ,FERADR    ;SAVE WHERE WE WERE
002610  016746  175316                  MOV     RBUFSZ,-(SP)    ;GET READ SIZE
002614  005416                          NEG     (SP)            ;NEGATE IT
002616  012677  176662                  MOV     (SP)+,@RMWC     ;LOAD WORD COUNT
002622  016777  175300  176656          MOV     RBUFPA,@RMBA    ;LOAD BUFFER ADDRESS
002630  016746  175274                  MOV     RBUFEA,-(SP)    ;GET EXTENDED MEMORY BITS
002634  006316                          ASL     (SP)            ;SHIFT 4 PLACES TO THE LEFT
002636  006316                          ASL     (SP)            ;TO LINE UP WITH RMCS1
002640  006316                          ASL     (SP)
002642  006316                          ASL     (SP)
002644  012667  175504                  MOV     (SP)+,XMEM      ;SAVE THE SHIFTED BITS
002650  000167  000020                  JMP     GO1             ;CONTINUE

002654  012767  000105  175520  SEEK:   MOV     #105,FUNC       ;SEEK COMMAND
```



```
002662  012767  002654' 175516          MOV     #SEEK,FERADR    ;CALLING ADDRESS
002670  000167  000104                  JMP     GO2             ;EXECUTE THE SEEK COMMAND
```

```
                                .SBTTL  COMMAND EXECUTE ROUTINE
                        
002674  005767  175464          GO1:    TST     FLAG70          ;11/70???
002700  100034                          BPL     1$              ;NO
002702  017767  176600  175442          MOV     @RMBA,PA18      ;GET 18 BIT ADDR
002710  006267  175440                  ASR     XMEM            ;SHIFT EA BITS TO POSITION 4,5
002714  006267  175434                  ASR     XMEM
002720  006267  175430                  ASR     XMEM
002724  006267  175424                  ASR     XMEM
002730  104416  000000' 000352'         MAP22$, BEGIN,PA18              ; GET 22-BIT ADDR FROM 18-BIT ADDR
002736  016777  175414  176542          MOV     PA22,@RMBA      ;LOAD BA REG
002744  016777  175410  176600          MOV     EA22,@RMBAE     ;LOAD BAE REG
002752  042767  000074  175400          BIC     #74,EA22        ;CLEAR UNWANTED BITS
002760  000367  175374                  SWAB    EA22            ;LOAD INTO BITS 8,9
002764  016767  175370  175362          MOV     EA22,XMEM       ;LOAD XMEM TO SET INTO FUNCTION CODE
002772  056767  175356  175402  1$:     BIS     XMEM,FUNC       ;LOAD EXTENDED MEMORY BITS
                        
003000  016777  175412  176526  GO2:    MOV     RMOF1,@RMOF     ;LOAD OFFSET REGISTER
003006  016777  175354  176522          MOV     CYLIND,@RMDC    ;LOAD THE CYLINDER ADDRESS
003014  016777  175350  176466          MOV     DSKADR,@RMDA    ;LOAD THE TRACK AND SECTOR ADDRESS
003022  016777  175354  176452          MOV     FUNC,@RMCS1     ;EXECUTE THE FUNCTION
003030  104400  000000'                  EXIT$S,BEGIN                    ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
```



```
                                .SBTTL  CHECK FOR ERRORS
                        
003034                          NTRUPT:
                                ;---------------------------------------------------------------
003034  000004  000000' 003042'         PIRQ$,BEGIN,1$          ; QUEUE UP TO CONTINUE AT 1$ AND RTI
                                ;---------------------------------------------------------------
                        
003042  004567  000672          1$:     JSR     R5,ERRORS       ;GO CHECK FOR ERRORS
003046  000403                          BR      2$              ;RETURN HERE ON ERRORS
003050  062705  000002                  ADD     #2,R5           ;NORMAL RETURN
003054  000441                          BR      4$              ;EXIT
                        
003056  005767  175300          2$:     TST     DRPDRV          ;DROP DRIVE ?
003062  001043                          BNE     6$              ;BR IF YES
003064  004567  002044                  JSR     R5,READY        ;SEE IF DRIVE IS READY FOR COMMAND AGAIN
003070  000440                          BR      6$              ;BR TO 6$ ON ERROR
003072  022767  177777  175276          CMP     #-1,BADSEC      ;USR DEFINED BAD SPOT ?
003100  001434                          BEQ     6$              ;BR IF YES
003102  032767  000040  175266          BIT     #SSE,BADSEC     ;SKIP SECTOR ERROR ?
003110  001406                          BEQ     3$              ;BR IF NO
003112  012767  011000  175276          MOV     #FMT!SSEI,RMOF1 ;SET 16 BIT FORMAT WITH SKIP SECTOR INHIBIT
003120  005267  175244                  INC     DSKADR          ;PERFORM COMMAND ON NEXT SECTOR
003124  000420                          BR      5$
003126  032767  100000  175242  3$:     BIT     #BSE,BADSEC     ;BAD SECTOR ERROR ?
003134  001016                          BNE     6$              ;BR IF YES
                        
003136  005267  175250                  INC     TRY             ;INCREMENT THE RETRY COUNT
003142  026727  175244  000003          CMP     TRY,#3          ;OVER 3 RETRIES ?
003150  103406                          BLO     5$              ;BR IF NOT
003152  104403  000000' 007376'         MSG$S,BEGIN,EXCED       ;ASCII MESSAGE CALL WITH COMMON HEADER
003160  005067  175226          4$:     CLR     TRY
003164  000402                          BR      6$              ;EXIT
                        
003166  162705  000004          5$:     SUB     #4,R5           ;ADJUST ADDRESS FOR RETRY
003172  000205                  6$:     RTS     R5              ;EXIT
```

```
                                        .SBTTL  GENERATE DISK ADDRESS

                                        ;THIS ROUTINE IS USED TO GENERATOR A DISK ADDRESS ON WHICH THE
                                        ;DRIVE WILL SEEK, WRITE, WRITE CHECK AND READ ON.
                                        ;
                                        ;CALL
                                        ;       JSR     PC,GENADR       ;CALL GENERATOR ROUTINE

003174  012700  001562'        GENADR: MOV     #TAB80,R0       ;GET RM80 LIMITER AND STORAGE TABLE
003200  032767  002000  174610  1$:     BIT     #BIT10,SR1      ;CHECK SR1 FOR RANDOM ACCESS MODE
003206  001463                          BEQ     6$              ;BR IF NO
003210  104417  000000'                 RAND$,BEGIN
003214  016760  174634  000012          MOV     RANNUM,SEC(R0)  ;GENERATE THE SECTOR ADDRESS
003222  042760  177400  000012          BIC     #177400,SEC(R0) ;CHOP OFF HIGH BYTE
003230  104417  000000'                 RAND$,BEGIN
003234  016746  174614                  MOV     RANNUM,-(SP)    ;GENERATE THE TRACK ADDRESS
003240  042716  177400                  BIC     #177400,(SP)    ;CHOP OFF HIGH BYTE
003244  104417  000000'                 RAND$,BEGIN
003250  016746  174600                  MOV     RANNUM,-(SP)    ;GENERATE THE CYLINDER ADDRESS
003254  042716  176000                  BIC     #176000,(SP)    ;CHOP OFF HIGH BITS

003260  004767  000236                  JSR     PC,GETMAP       ;CALL MAPPING ROUTINE
003264  166716  175132          2$:     SUB     CYLMAP,(SP)     ;OVER THE CYLINDER LIMIT ?
003270  002375                          BGE     2$              ;BR IF YES
003272  066716  175124                  ADD     CYLMAP,(SP)     ;ADD MAPPING FACTOR BACK
003276  032767  000001  174512          BIT     #BIT0,SR1       ;FE CYLINDER ONLY ?
003304  001002                          BNE     3$              ;BR IF NO
003306  066016  000020                  ADD     FFE(R0),(SP)    ;ADJUST TO STARTING CYLINDER
003312  012660  000016          3$:     MOV     (SP)+,CYL(R0)   ;SAVE CYLINDER ADDRESS

003316  166716  175076          4$:     SUB     TRKMAP,(SP)     ;OVER THE TRACK LIMIT ?
003322  002375                          BGE     4$              ;BR IF YES
003324  066716  175070                  ADD     TRKMAP,(SP)     ;ADD MAPPING FACTOR BACK
003330  066016  000000                  ADD     FT(R0),(SP)     ;ADJUST TO STARTING TRACK
003334  012660  000014                  MOV     (SP)+,TRK(R0)   ;SAVE TRACK ADDRESS

003340  026060  000004  000012  5$:     CMP     LS(R0),SEC(R0)  ;OVER THE SECTOR LIMIT ?
003346  103044                          BHIS    8$              ;BR IF NOT
003350  006260  000012                  ASR     SEC(R0)         ;REDUCE TO HALF
003354  000771                          BR      5$              ;CHECK AGAIN

                                        ;GENERATE DISK ADDRESS BY SEQUENTIAL ADDRESSING

003356  005260  000012          6$:     INC     SEC(R0)         ;SEQUENTIAL ACCESS TO ALL SECTORS
003362  026060  000004  000012          CMP     LS(R0),SEC(R0)  ;TIME TO ADJUST THE TRACK ADDRESS
003370  103033                          BHIS    8$              ;BR IF NOT
003372  005060  000012                  CLR     SEC(R0)         ;RESET THE SECTOR ADDRESS

003376  005260  000014                  INC     TRK(R0)         ;INCREMENT THE TRACK ADDRESS
003402  026060  000002  000014          CMP     LT(R0),TRK(R0)  ;OVER LIMIT ?
003410  103023                          BHIS    8$              ;BR IF NOT
003412  016060  000000  000014          MOV     FT(R0),TRK(R0)  ;RESET THE STARTING TRACK ADDRESS

003420  005260  000016                  INC     CYL(R0)         ;ADJUST THE CYLINDER ADDRESS
003424  026060  000022  000016          CMP     LFE(R0),CYL(R0) ;OVER LIMIT ?
003432  103012                          BHIS    8$              ;BR IF NOT
003434  032767  000001  174354          BIT     #BIT0,SR1       ;FE CYLINDER ONLY ?
```



```
003442  001403                          BEQ     7$              ;BR IF YES
003444  005060  000016                  CLR     CYL(R0)         ;ELSE, RESET CYLINDER ADDRESS
003450  000403                          BR      8$
003452  016060  000020  000016  7$:     MOV     FFE(R0),CYL(R0) ;GET FIRST FE CYLINDER

                                        ;CHECK FOR MFG/USR AND SKIP SECTOR FILES IN TRACK ADDRESS

003460  026060  000006  000016  8$:     CMP     LC(R0),CYL(R0)  ;IS IT THE LAST CYLINDER ?
003466  001004                          BNE     9$              ;BR IF NO
003470  026060  000010  000014          CMP     BST(R0),TRK(R0) ;IS IT THE BAD SECTOR TRACK (DEC 144) ?
003476  001640                          BEQ     1$              ;YES, THIS IS BAD SECTOR FILE
                                                                ;GET ANOTHER ADDRESS
003500  026060  000020  000016  9$:     CMP     FFE(R0),CYL(R0) ;FIRST FE CYLINDER ?
003506  001004                          BNE     10$             ;BR IF NO
003510  026027  000014  000001          CMP     TRK(R0),#1      ;IS THIS TRACK 0 OR 1 ON FFE CYLINDER ?
003516  003630                          BLE     1$              ;GET ANOTHER ADDRESS
003520  000207          10$:    RTS     PC

                                        ;THIS ROUTINE IS USED TO MAP OUT THE RANGE OF TRACK AND CYLINDER
                                        ;ADDRESSES USED IN THE RANDOM GENERATING ROUTINE.
                                        ;
                                        ;CALL
                                        ;       JSR     PC,GETMAP       ;CALL ROUTINE

003522  032767  000001  174266  GETMAP: BIT     #BIT0,SR1       ;FE CYLINDER ONLY ?
003530  001402                          BEQ     1$              ;BR IF YES
003532  005002                          CLR     R2              ;GET FIRST CYLINDER
003534  000402                          BR      2$
003536  016002  000020          1$:     MOV     FFE(R0),R2      ;GET FIRST FE CYLINDER
003542  016003  000022          2$:     MOV     LFE(R0),R3      ;GET LAST FE CYLINDER
003546  005302                          DEC     R2              ;START ONE LESS THAN FIRST CYLINDER
003550  160203                          SUB     R2,R3           ;CALCULATE MAPPING FACTOR AND
003552  010367  174644                  MOV     R3,CYLMAP       ;SAVE IT.
003556  016002  000000                  MOV     FT(R0),R2       ;GET FIRST TRACK AND
003562  005302                          DEC     R2              ;START ONE TRACK LESS.
003564  016003  000002                  MOV     LT(R0),R3       ;GET LAST TRACK
003570  160203                          SUB     R2,R3           ;CALCULATE MAPPING FACTOR AND
003572  010367  174622                  MOV     R3,TRKMAP       ;SAVE IT.
003576  000207                          RTS     PC

                                        ;THIS ROUTINE IS USED TO LOAD THE SECTOR, TRACK AND CYLINDER ADDRESSES
                                        ;FOR THE DEVICE UNDER TEST.
                                        ;
                                        ;CALL
                                        ;       JSR     PC,LOADR        ;CALL LOAD ADDRESS ROUITNE

003600  012700  001562'        LOADR:  MOV     #TAB80,R0       ;GET RM80 TABLE POINTER
003604  116067  000012  174556          MOVB    SEC(R0),DSKADR  ;LOAD SECTOR ADDRESS
003612  116067  000014  174551          MOVB    TRK(R0),DSKADR+1        ;LOAD TRACK ADDRESS
003620  016067  000016  174540          MOV     CYL(R0),CYLIND  ;LOAD CYLINDER ADDRESS
003626  000207                          RTS     PC              ;EXIT

                                        ;ROUTINE TO GET A DRIVE FOR TESTING, FROM LOCATION 'DVICE'
                                        ;
                                        ;CALL
                                        ;       JSR     RC,GETDRV       ;CALL ROUTINE
                                        ;       RET                     ;RETURN HERE DRIVE IS NOT SELECTED
```

```
                                        ;       RET                     ;NORMAL RETURN
                                        ;
003630  005267  174540                  GETDRV: INC     UNITNO          ;POINT TO NEXT DRIVE
003634  026727  174534  000010                  CMP     UNITNO,#8.      ;DONE LOOKING?
003642  002010                                  BGE     1$              ;EXIT IF ALL DRIVE SELECTED
003644  016700  174524                          MOV     UNITNO,R0       ;USE AS AN INDEX
003650  136067  003666' 174514                  BITB    BITTAB(R0),DVICE        ;TEST THIS DEVICE?
003656  001764                                  BEQ     GETDRV          ;LOOP BACK , IF NOT SELECTED
003660  062716  000002                          ADD     #2,(SP)         ;ADJUST FOR DRIVE AVAILABLE RETURN
003664  000207                          1$:     RTS     PC
                                        ;
                                        ;THIS IS ATTENTION BIT TABLE FOR EACH DRIVE (BYTES)
                                        ;
003666  001001                          BITTAB: 1001
003670  004004                                  4004
003672  020020                                  20020
003674  100100                                  100100
                                        ;
                                        ;ROUTINE TO DROP THE ERRORED DRIVE FROM THE TEST
                                        ;
                                        ;CALL
                                        ;       JSR     PC,DROP         ;CALL ROUTINE
                                        ;
003676  012701  000001                  DROP:   MOV     #1,R1           ;INITIALIZE DROP PICKER
003702  016700  174466                          MOV     UNITNO,R0       ;GET THE DRIVE NUMBER
003706  001403                                  BEQ     2$              ;IF DRIVE 0 GO DROP IT
003710  006301                          1$:     ASL     R1              ;POINT TO NEXT DRIVE
003712  005300                                  DEC     R0              ;IS THIS THE ONE ?
003714  001375                                  BNE     1$              ;NO, LOOK AGAIN
003716  040167  174450                  2$:     BIC     R1,DVICE        ;DROP THE DRIVE
                                                ;*************************************************************************
                                                ;CONVERT UNITNO TO ASCII AND
                                                ;STORE AT ADR1
003722  104420  000000' 000374'                 DTOAS,BEGIN,UNITNO,ADR1
                                                ;*************************************************************************
003732  005067  174424                          CLR     DRPDRV          ;CLEAR DROP DRIVE FLAG
003736  000207                                  RTS     PC              ;RETURN
```



```
                                        .SBTTL  ERROR HANDLER ROUTINE
                                        ;
                                        ;CALL
                                        ;       JSR     R5,ERRORS       ;CHECK IF ANY ERROR ON RH/RM
                                        ;       RET                     ;ERROR RET
                                        ;       RET2                    ;NORMAL RET
                                        ;
003740  005067  174432                  ERRORS: CLR     BADSEC          ;CLEAR BAD SPOT FLAG
003744  004767  001706                          JSR     PC,ERSUB1       ;SET UP FOR ERROR REPORT
003750  004767  001554                          JSR     PC,GET          ;READ AND STORE ALL REGISTERS
003754  012777  040011  175520                  MOV     #40011,@RMCS1   ;ISSUE DRIVE CLEAR COMMAND
003762  016700  175620                          MOV     S+NCS1,R0       ;GET CONTROL STATUS
003766  042700  177576                          BIC     #^C<RDY!GO>,R0
003772  022700  000200                          CMP     #RDY,R0         ;RDY = 1 AND GO = 0 ?
003776  001416                                  BEQ     1$              ;BR IF YES
004000  012767  000001  174354                  MOV     #1,DRPDRV       ;SET DROP DRIVE FLAG
004006  104403  000000' 007512'                 MSGN$,BEGIN,NOTRDY      ;ASCII MESSAGE CALL WITH COMMON HEADER
004014  012767  000003  174064                  MOV     #3,ERRTYP       ;SAVE THE ERROR TYPE
                                                ;*************************************************************************
004022  104405  000000' 001424'                 HRDERS,BEGIN,TABLE      ;CONTROLLER NOT READY
                                                ;*************************************************************************
004030  000167  001076                          JMP     32$
                                        ;
004034  016700  175560                  1$:     MOV     S+NDS,R0        ;GET DRIVE STATUS
004040  005100                                  COM     R0
004042  032700  010700                          BIT     #DRY!DPR!VV!MOL,R0      ;DRIVE READY ?
004046  001416                                  BEQ     2$              ;BR IF YES
004050  012767  000001  174304                  MOV     #1,DRPDRV       ;SET DROP DRIVE FLAG
004056  104403  000000' 007502'                 MSGN$,BEGIN,NUDRDY      ;ASCII MESSAGE CALL WITH COMMON HEADER
004064  012767  000006  174014                  MOV     #6,ERRTYP       ;SAVE THE ERROR TYPE
                                                ;*************************************************************************
004072  104405  000000' 001424'                 HRDERS,BEGIN,TABLE      ;DRIVE NOT READY
                                                ;*************************************************************************
004100  000167  001026                          JMP     32$
                                        ;
004104  016700  175476                  2$:     MOV     S+NCS1,R0       ;CHECK IF SEEK COMMAND IS EXECUTED
004110  042700  177701                          BIC     #177701,R0      ;LEFT ONLY FUNCTION BITS
004114  022700  000004                          CMP     #4,R0           ;A SEEK COMMAND ?
004120  001017                                  BNE     3$              ;BR IF NOT
004122  016700  175472                          MOV     S+NDS,R0        ;CHECK THE STATUS
004126  042700  057777                          BIC     #^C<ATA!PIP>,R0
004132  022700  100000                          CMP     #ATA,R0         ;ATA = 1 AND PIP = 0 ?
004136  001413                                  BEQ     4$              ;BR IF YES
004140  012777  000013  175334                  MOV     #13,@RMCS1      ;RELEASE THE DRIVE
004146  104403  000000' 007426'                 MSGN$,BEGIN,SEEKER      ;ASCII MESSAGE CALL WITH COMMON HEADER
004154  000167  000736                          JMP     31$             ;EXIT
                                        ;
004160  012777  000013  175314  3$:     MOV     #13,@RMCS1      ;RELEASE THE DRIVE
004166  032767  060000  175412  4$:     BIT     #MCPE!TRE,S+NCS1        ;ANY RH CONTROLLER ERROR DETECTED ?
004174  001010                                  BNE     5$              ;BR IS SO
004176  032767  040000  175414                  BIT     #ERR,S+NDS      ;ANY DRIVE ERROR ?
004204  001004                                  BNE     5$              ;BR IS SO
004206  062705  000002                          ADD     #2,R5           ;ELSE,ADJUST FOR GOOD RETURN
004212  000167  000714                          JMP     32$             ;EXIT
                                        ;
                                        ;CHECK TO SEE IF USER DEFINED ANY BAD SPOTS
                                        ;
```

```
                                ;TABLE ADDRESS
                                ;TOTAL BAD SECTOR COUNT
                                ;ON THE SAME CYLINDER ?
                                ;BR IF NOT
                                ;ON THE SAME TRACK ?
                                ;BR IF NOT
                                ;ON THE SAME  SECTOR ?
                                ;BR IF NOT
                                ;SET USR BAD SPOT FLAG
                                ;EXIT
                                ;ADJUST TABLE ADDRESS
                                ;DECREMENT COUNT
                                ;BR IF NOT END OF TABLE

                                ;CHECK FOR HARDWARE DETECTED BAD SPOTS

                                ;HEADER CRC ERROR ?
                                ;BR IF YES
                                ;IS IT ONLY A HEADER COMPARE ERROR ?
                                ;BR IF YES
                                ;ANY MEDIA DETECTED ERRORS ?
                                ;BR IF NO
                                ;YES--GET RMER2 FOR HSE OR SSE DETECT

                                ;TAKE ERROR EXIT, BUT NOT REPORT THE ERROR

                                ;CHECK FOR OTHER ERRORS

                                ;RH CONTROL BUS PARITY ERROR ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;MASS BUS DATA PARITY ERROR ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;RH CONTROL FAILS ?
                                ;BR IF NOT
                                ;ANY DRIVE ERROR ?
                                ;BR IF SO
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;DATA LATE ERROR ?
                                ;BR IF NOT
                                ;INCREMENT DATA LATE ERROR COUNT
                                ;ALLOW TO TYPE OUT DATA LATE ERROR ?
                                ;BR IF SO
                                ;EXIT

                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;SAVE THE ERROR TYPE
                                ;*****************************************************************
                                ;DATA LATE ERROR
                                ;*****************************************************************
                                ;EXIT
                                ;ANY DRIVE ERROR ?
```



```
                                ;BR IF ANY
                                ;WRITE CHECK ERROR ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;THEN RH TRANSFER ERROR ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;UNKNOWN ERROR
                                ;*****************************************************************
                                ;UNKNOWN ERROR
                                ;*****************************************************************
                                ;EXIT
                                ;RM DEVICE UNSAFE ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;HEADER INFORMATION ERROR ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;DATA FORMAT ERROR ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;ADDRESSING ERROR ?
                                ;BR IF NOT
                                ;LAST BLOCK TRANSFER SET ?
                                ;EXIT, IF SO
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;POSITIONER FAILURE ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;SEEK ERROR ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;LOSS OF SYSTEM CLOCK ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;WRITE LOCK ERROR ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;DRIVE TIMING ERROR ?
                                ;BR IF NOT
                                ;ASCII MESSAGE CALL WITH COMMON HEADER
                                ;EXIT
                                ;LOST BIT CLOCK ?
                                ;BR IF NOT
```

```
163 005056  104403  000000' 007446'          MSGN$,BEGIN,LOSBIT      ;ASCII MESSAGE CALL WITH COMMON HEADER
164 005064  000414                          BR      31$             ;EXIT
165 005066  032767  000040  174526  29$:     BIT     #WCF,S+NER1     ;WRITE CLOCK ERROR ?
166 005074  001404                          BEQ     30$             ;BR IF NOT
167 005076  104403  000000' 007452'          MSGN$,BEGIN,WICLCK      ;ASCII MESSAGE CALL WITH COMMON HEADER
168 005104  000404                          BR      31$             ;EXIT
169 005106                          30$:
    005106  104403  000000' 007456'          MSGN$,BEGIN,DRVERR      ;ASCII MESSAGE CALL WITH COMMON HEADER
170 005114  000400                          BR      31$             ;EXIT
171 005116  012767  000000  172762  31$:     MOV     #0,ERRTYP       ;DUMMY ERROR NUMBER
172                                         ;*****************************************************************
    005124  104406  000000' 001424'          SOFER$,BEGIN,TABLE      ;DUMP RH/RM REGISTERS
                                            ;*****************************************************************
173
174 005132  000205                  32$:    RTS     R5              ;EXIT
175
```



```
  1                                 ;THIS ROUTINE IS USED TO SEE IF THE DRIVE IS READY FOR TESTING BY
  2                                 ;CHECKING 'DVA' AND THE DRIVE TYPE. IF 'DVA' IS NOT SET OR THE
  3                                 ;DRIVE TYPE IS INCORRECT, THEN AN ERROR IS REPORTED AND THE DRIVE
  4                                 ;IS DROPPED.
  5                                 ;CALL
  6                                 ;       JSR     R5,READY        ;CALL READY ROUTINE
  7                                 ;       BR      ?               ;ERROR RETURN
  8                                 ;       RETURN                  ;NORMAL RETURN
  9
 10 005134  004767  000110          READY:  JSR     PC,WAIT         ;WAIT FOR "DVA" BIT TO SET
 11 005140  000422                          BR      1$              ;RETURN HERE ON ERROR
 12 005142  017700  174362                  MOV     @RMDT,R0        ;READ THE DRIVE TYPE
 13 005146  010067  173242                  MOV     R0,DRVTYP       ;GET DRIVE TYPE AND
 14 005152  042767  177740  173234          BIC     #^C37,DRVTYP    ;SAVE TYPE BITS
 15 005160  022700  024026                  CMP     #24026,R0       ;DUAL PORT RM80 ?
 16 005164  001426                          BEQ     2$              ;BR IS SO
 17 005166  022700  020026                  CMP     #20026,R0       ;SINGLE PORT RM80 ?
 18 005172  001423                          BEQ     2$              ;BR IF SO
 19 005174  004767  000456                  JSR     PC,ERSUB1       ;SETUP FOR ERROR REPORT
 20 005200  104403  000000' 007506'          MSGN$,BEGIN,DEVNOT      ;ASCII MESSAGE CALL WITH COMMON HEADER
 21
 22 005206  004767  000316          1$:     JSR     PC,GET          ;READ AND STORE ALL REGISTERS
 23 005212  012767  000006  172666          MOV     #6,ERRTYP       ;SAVE THE ERROR TYPE
 24                                         ;*****************************************************************
    005220  104405  000000' 001424'          HRDER$,BEGIN,TABLE      ;DEVICE NOT AVAILABLE OR NOT AN RM80
                                            ;*****************************************************************
 25 005226  004767  176444                  JSR     PC,DROP         ;DROP THE DRIVE
 26 005232  104403  000000' 007522'          MSGN$,BEGIN,DRM ;ASCII MESSAGE CALL WITH COMMON HEADER
 27 005240  000205                          RTS     R5              ;EXIT FOR ERROR RETURN
 28
 29 005242  062705  000002          2$:     ADD     #2,R5           ;DRIVE AVAILABLE, ON-LINE, AND NO ERRORS
 30 005246  000205                          RTS     R5              ;NORMAL RETURN
 31
 32                                 ;THIS ROUTINE WAITS FOR "DVA" TO SET IN THE RMCS1 REGISTER. IF THE "DVA"
 33                                 ;BIT IS SET, THE DRIVE IS CLEARED AND THE COMMAND SEQUENCE BEGINS. IF THE
 34                                 ;"DVA" BIT IS NOT SET, A DRIVE CLEAR COMMAND IS ISSUED TO THE DRIVE TO SET
 35                                 ;PORT REQUEST. THE PROGRAM THEN CHECKS "DVA" IN RMCS1. IF THE "DVA" BIT
 36                                 ;DOES NOT SET WITHIN A CERTAIN PERIOD OF TIME, AN ERROR IS REPORTED AND
 37                                 ;RETURN IS MADE BACK TO THE MAIN PROGRAM, WHERE THE DRIVE WILL BE DROPPED
 38                                 ;FROM THE TEST.
 39                                 ;
 40                                 ;CALL
 41                                 ;       JSR     PC,WAIT         ;CALL WAIT ROUTINE
 42                                 ;       BR      ???             ;ERROR RETURN
 43                                 ;       RETURN                  ;NORMAL RETURN
 44
 45 005250  032777  004000  174224  WAIT:   BIT     #DVA,@RMCS1     ;IS DRIVE AVAILABLE ?
 46 005256  001035                          BNE     2$              ;BR IF YES
 47 005260  012777  000011  174214          MOV     #11,@RMCS1      ;ISSUE DRIVE CLEAR TO REQUEST PORT
 48 005266  012767  017777  173114          MOV     #17777,CLK      ;CLOCK COUNT
 49 005274  032777  004000  174200  1$:     BIT     #DVA,@RMCS1     ;IS DRIVE AVAILABLE ?
 50 005302  001023                          BNE     2$              ;BR IF YES
 51 005304  104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
    005310  104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
 52 005314  005367  173070                  DEC     CLK             ;DECREMENT CLOCK COUNT
 53 005320  001365                          BNE     1$
 54 005322  004767  000330                  JSR     PC,ERSUB1       ;SETUP FOR ERROR REPORT
```

```
    55 005326  012767  000000  172552          MOV     #0,ERRTYP       ;UNKNOWN ERROR TYPE
    56 005334  104403  000000' 007516'        MSGN$,BEGIN,NODVA       ;ASCII MESSAGE CALL WITH COMMON HEADER
    57 005342  012767  000001  173012          MOV     #1,DRPDRV       ;SET DROP DRIVE FLAG
    58 005350  000207                          RTS     PC              ;ERROR RETURN, TIMEOUT ON "DVA" BIT
    59
    60 005352  012777  040011  174122  2$:     MOV     #40011,@RMCS1   ;ISSUE A DRIVE CLEAR COMMAND
    61 005360  032767  040000  174262          BIT     #SKI,S+NER2     ;WAS LAST ERROR AN "SKI" ?
    62 005366  001406                          BEQ     3$              ;BR IF NO
    63 005370  012777  000007  174104          MOV     #7,@RMCS1       ;ISSUE A RECALIBRATE COMMAND AND
    64 005376  004767  000010                  JSR     PC,WTCRDY       ;WAIT FOR CONTROLLER READY
    65 005402  000207                          RTS     PC              ;ERROR RETURN, TIMEOUT ON "CRDY" BIT
    66
    67 005404  062716  000002          3$:     ADD     #2,(SP)         ;ADJUST FOR GOOD RETURN
    68 005410  000207                          RTS     PC              ;RETURN
    69
    70
    71                                 ;THIS ROTUINE IS USED TO WAIT FOR "CRDY" TO SET AND FOR "GO" TO CLEAR.
    72                                 ;IF "CRDY" DOES SET AND "GO" DOES NOT CLEAR IN A CERTAIN PERIOD OF
    73                                 ;TIME, AN ERROR WILL BE REPORTED AND THE DRIVE WILL BE DROPPED FROM
    74                                 ;THE TEST.
    75                                 ;
    76                                 ;CALL   JSR     PC,WTCRDY       ;CALL CONTROLLER WAIT ROUTINE
    77
    78 005412  012767  017777  172770  WTCRDY: MOV     #17777,CLK      ;CLOCK COUNT
    79 005420  017700  174056          1$:     MOV     @RMCS1,R0       ;GET CONTROL STATUS
    80 005424  042700  177576                  BIC     #^C<RDY!GO>,R0
    81 005430  022700  000200                  CMP     #RDY,R0         ;RDY = 1 AND GO = 0 ?
    82 005434  001432                          BEQ     2$              ;BR IF YES
    83 005436  104407  000000'                BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
       005442  104407  000000'                BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
    84 005446  005367  172736                  DEC     CLK             ;DECREMENT CLOCK COUNT
    85 005452  001362                          BNE     1$
    86 005454  004767  000176                  JSR     PC,ERSUB1       ;SETUP FOR ERROR REPORT
    87 005460  104403  000000' 007512'        MSGN$,BEGIN,NOTRDY      ;ASCII MESSAGE CALL WITH COMMON HEADER
    88 005466  004767  000036                  JSR     PC,GET          ;READ AND STORE ALL REGISTERS
    89 005472  012767  000003  172406          MOV     #3,ERRTYP       ;SAVE THE ERROR TYPE
    90                                         ;*****************************************************************
       005500  104405  000000' 001424'        HRDER$,BEGIN,TABLE      ;CONTROLLER READY DID NOT SET
                                               ;*****************************************************************
    91 005506  004767  176164                  JSR     PC,DROP         ;DROP THE DRIVE
    92 005512  104403  000000' 007522'        MSGN$,BEGIN,DRM ;ASCII MESSAGE CALL WITH COMMON HEADER
    93 005520  000207                          RTS     PC              ;ERROR RETURN
    94
    95 005522  062716  000002          2$:     ADD     #2,(SP)         ;ADJUST FOR GOOD RETURN
    96 005526  000207                          RTS     PC              ;RETURN
    97
    98                                 ;THIS ROUTINE IS USED TO READ AND STORE ALL REGISTERS FROM THE RH/RM
    99                                 ;WHEN AN ERROR HAS BEEN DETECTED.
   100                                 ;
   101                                 ;CALL
   102                                 ;       JSR     PC,GET          ;CALL GET ROUTINE
   103
   104 005530  005000                 GET:    CLR     R0              ;START WITH RMCS1 REGISTER INDEX 0
   105 005532  016701  173744                  MOV     RMCS1,R1        ;RH CONTROLLER BUS ADDRESS
   106 005536  022701  001524'        1$:     CMP     #RMDB,R1        ;IS IT RMDB REGISTER ?
   107 005542  001006                          BNE     2$              ;BR IF NO
   108 005544  032711  000200                  BIT     #DR,(R1)        ;IS "DR" BIT SET ?
```



```
   109 005550  001003                          BNE     2$              ;BR IF YES
   110 005552  005060  001606'                 CLR     S(R0)           ;ELSE, CLEAR RMDB REGISTER
   111 005556  000402                          BR      3$
   112 005560  012160  001606'          2$:    MOV     (R1)+,S(R0)     ;READ ONE REGISTER
   113 005564  005720                   3$:    TST     (R0)+           ;ADJUST TABLE INDEX
   114 005566  022700  000050                  CMP     #50,R0          ;DONE READING ALL REGISTERS ?
   115 005572  001361                          BNE     1$              ;BR IF NO
   116 005574  012760  177777  001424'        MOV     #-1,TABLE(R0)   ;TERMINATE DECX REGISTER TABLE FOR AN RH11
   117 005602  005767  172556                  TST     FLAG70          ;CHECK FOR ADDITIONAL RH70 REGISTERS
   118 005606  100010                          BPL     4$              ;BR IF NONE
   119 005610  012760  001656' 001424'        MOV     #S+NBAE,TABLE(R0)       ;EXTEND THE DECX REGISTER TABLE
   120 005616  012160  001606'                 MOV     (R1)+,S(R0)     ;GET RMBAE REGISTER
   121 005622  005720                          TST     (R0)+           ;ADJUST INDEX TABLE
   122 005624  011160  001606'                 MOV     (R1),S(R0)      ;GET RMCS3 REGISTER
   123 005630  000207                   4$:    RTS     PC              ;RETURN
   124
   125                                 ;THESE TWO ROUTINES ARE USED TO SETUP FOR ERROR REPORTING
   126                                 ;CALL
   127                                 ;       JSR     PC,ERSUB?       ;CALL ROUTINE
   128
   129 005632  014167  172250          ERSUB2: MOV     -(R1),ASB       ;LOAD THE DATA
   130 005636  010167  172240                  MOV     R1,SBADR        ;LOAD ADDRESS OF DATA WRITTEN
   131 005642  014267  172242                  MOV     -(R2),AWAS      ;LOAD THE DATA
   132 005646  010267  172232                  MOV     R2,WASADR       ;LOAD ADDRESS OF DATA READ
   133 005652  005721                          TST     (R1)+           ;RESET REG. 1
   134 005654  005722                          TST     (R2)+           ;RESET REG. 2
   135
   136 005656  016767  173620  172214  ERSUB1: MOV     RMCS1,CSRA      ;LOAD ADR OF CURRENT CSR
   137 005664  017767  173612  172210          MOV     @RMCS1,ACSR     ;LOAD CONTENTS OF CURRENT CSR
   138 005672  000207                          RTS     PC              ;RETURN
   139
   140                                 ;THIS ROUTINE IS USED TO SETUP THE RH BASE ADDRESS OFFSET TO DEFINE THE
   141                                 ;REGISTER ADDRESSES AND TO DETERMINE IF AN 11/70 IS BEING USED.
   142                                 ;CALL
   143                                 ;       JSR     PC,SETUP        ;CALL ROUTINE
   144
   145 005674  005067  172464          SETUP:  CLR     FLAG70          ;CLEAR 11/70 FLAG
   146 005700  016700  172102                  MOV     ADDR,R0         ;GET DEVICE ADDRESS (DEFAULT = 176700)
   147 005704  012701  001502'                 MOV     #RMCS1,R1       ;GET STARTING REGISTER ADDRESS
   148 005710  010021                   1$:    MOV     R0,(R1)+        ;GENERATE REGISTER ADDRESS
   149 005712  062700  000002                  ADD     #2,R0
   150 005716  022701  001552'                 CMP     #RMBAE,R1       ;LAST REGISTER FOR RH11 ?
   151 005722  001372                          BNE     1$              ;BR IF NO
   152 005724  052777  002000  173550          BIS     #BIT10,@RMCS1   ;TRY SETTING "PSEL" BIT
   153 005732  032777  002000  173542          BIT     #BIT10,@RMCS1   ;IS CPU AN 11/70 ?
   154 005740  001015                          BNE     3$              ;BR IF NO
   155 005742  052767  100000  172414          BIS     #BIT15,FLAG70   ;SET 11/70 FLAG
   156 005750  032767  100000  172040          BIT     #BIT15,SR1      ;SPECIFY 32 REGISTER ON RH70 ?
   157 005756  001402                          BEQ     2$              ;BR IF NO
   158 005760  062700  000024                  ADD     #24,R0          ;YES, ADJUST THE RMBAE ADDRESS
   159 005764  010021                   2$:    MOV     R0,(R1)+        ;SETUP RMBAE REGISTER
   160 005766  062700  000002                  ADD     #2,R0           ;ADJUST INDEX
   161 005772  010011                          MOV     R0,(R1)         ;SETUP RMCS3 REGISTER
   162
   163 005774  016700  172010          3$:     MOV     VECTOR,R0       ;GET VECTOR ADDRESS
   164 006000  012720  003034'                 MOV     #NTRUPT,(R0)+   ;SET POINTER JUST IN CASE
   165 006004  116710  172002                  MOVB    BR1,(R0)        ;SET PRIORITY
```

```
                                        RTS     PC              ;RETURN

                                .SBTTL  ASCIZ MESSAGE

006012  045  040  122  MES1:   .ASCIZ  '% RH CONTROLLER TRANSFER ERROR'
006051  045  040  115  MES2:   .ASCIZ  '% MASSBUS PARITY ERROR MCPE=1'
006107  045  040  115  MES3:   .ASCIZ  '% MASSBUS DATA PARITY ERROR MDPE=1'
006152  040  104  122  MES4:   .ASCIZ  ' DRIVE '
006162  040  104  122  MES5:   .ASCIZ  ' DROPPED%'
006174  045  040  105  MES6:   .ASCIZ  '% EXCEEDED RETRY LIMIT%'
006224  045  040  104  MES10:  .ASCIZ  '% DEVICE UNSAFE'
006244  045  040  110  MES11:  .ASCIZ  '% HEADER INFORMATION ERROR'
006277  045  040  104  MES12:  .ASCIZ  '% DATA FORMAT BIT ERROR'
006327  045  040  104  MES14:  .ASCIZ  '% DRIVE ERROR'
006345  045  040  122  MES15:  .ASCIZ  '% RH CONTROLLER ERROR'
006373  045  040  101  MES16:  .ASCIZ  '% ADDRESSING ERROR'
006416  045  040  123  MES17:  .ASCIZ  '% SEEK ERROR'
006433  045  040  114  MES20:  .ASCIZ  '% LOST SYSTEM CLOCK'
006457  045  040  127  MES21:  .ASCIZ  '% WRITE LOCK ERROR'
006502  045  040  104  MES22:  .ASCIZ  '% DATA CHECK ERROR'
006525  045  040  114  MES23:  .ASCIZ  '% LOST BIT CLOCK'
006546  045  040  127  MES24:  .ASCIZ  '% WRITE CLOCK ERROR'
006572  045  040  127  MES25:  .ASCIZ  '% WRITE CHECK ERROR'
006616  045  040  104  MES26:  .ASCIZ  '% DRIVE TIMING ERROR'
006643  045  000       MES27:  .ASCIZ  '%'
006645  045  040  120  MES30:  .ASCIZ  '% POSITIONING ERROR'
006671  045  040  104  MES31:  .ASCIZ  '% DATA LATE ERROR'
006713  045  040  104  MES32:  .ASCIZ  '% DRIVE NOT READY'
006735  045  040  104  MES33:  .ASCIZ  '% DEVICE NOT AN RM80'
006762  045  040  103  MES34:  .ASCIZ  '% CONTROLLER NOT READY'
007011  045  040  104  MES35:  .ASCIZ  '% DRIVE NOT AVAILABLE'
007037  040  041  040  MES36:  .ASCIZ  ' ! OPERATING ON FE CYLINDERS ONLY !%'
007104  007  007  040  MES37:  .ASCII  <07><07>' ! CUSTOMER DATA WILL BE OVERWRITTEN !%'
007155  040  055  055          .ASCIZ  ' ----------------------------------%'<07><07>
007227  040  111  106  MES40:  .ASCII  ' IF YOU WISH TO EXERCISE THE WHOLE DISK, SET BIT0%'
007311  040  111  116          .ASCIZ  ' IN SWITCH REGISTER 1(SR1) EQUAL TO 1.%'

                                .EVEN
007362  006012'                 TRERR:  MES1
007364  177777                          177777
007366  006051'                 MCPERR: MES2
007370  177777                          177777
007372  006107'                 MDPERR: MES3
007374  177777                          177777
007376  006174'                 EXCED:  MES6
007400  177777                          177777
007402  006345'                 RHFAIL: MES15
007404  177777                          177777
007406  006224'                 UNSAFE: MES10
007410  177777                          177777
007412  006244'                 HEADER: MES11
007414  177777                          177777
007416  006277'                 FORMAT: MES12
007420  177777                          177777
007422  006373'                 ADDRES: MES16
007424  177777                          177777
007426  006416'                 SEEKER: MES17
```



```
007430  177777                          177777
007432  006433'                 LOSYSC: MES20
007434  177777                          177777
007436  006457'                 WTLOCK: MES21
007440  177777                          177777
007442  006502'                 DATERR: MES22
007444  177777                          177777
007446  006525'                 LOSBIT: MES23
007450  177777                          177777
007452  006546'                 WTCLCK: MES24
007454  177777                          177777
007456  006327'                 DRVERR: MES14
007460  177777                          177777
007462  006572'                 WRTCK:  MES25
007464  177777                          177777
007466  006616'                 TIMERR: MES26
007470  177777                          177777
007472  006645'                 POSIT:  MES30
007474  177777                          177777
007476  006671'                 DLTERR: MES31
007500  177777                          177777
007502  006713'                 NODRDY: MES32
007504  177777                          177777
007506  006735'                 DEVNOT: MES33
007510  177777                          177777
007512  006762'                 NOTRDY: MES34
007514  177777                          177777
007516  007011'                 NODVA:  MES35
007520  177777                          177777
007522  006152'                 DRM:    MES4
007524  007553'                         NUMB
007526  006162'                         MES5
007530  177777                          177777
007532  007037'                 FEONLY: MES36
007534  177777                          177777
007536  007104'                 OVRWRT: MES37
007540  177777                          177777
007542  007227'                 WARNING:MES40
007544  177777                          177777

007546                          ADR1:   .BLKB   5               ;STARTING OF THE ASCII NUMBER STRING
007553      000                 NUMB:   .BYTE   0               ;LEAST SIGNIFICANT ASCII DIGIT
007554      000                         .BYTE   0               ;TERMINATOR OF THE ASCII STRING

                                .EVEN


        000001                  .END
```

SYMBOL TABLE

```
ACSR    000102R   DROP    003676R   IR     = 000100   NBAE   = 000050   PUSH2 = 024646
ADDR    000006R   DRPDRV  000362R   LBC    = 002000   NCS1   = 000000   PWRFLG= 000002
ADDRES  007422R   DRVERR  007456R   LBT    = 002000   NCS2   = 000010   RAND$ = 104417
ADDR22= 001000    DRVTYP  000414R   LC     = 000006   NCS3   = 000052   RANNUM  000054R
ADR1    007546R   DRY     = 000200  LDADR   003600R   NDA    = 000006   RBUF    000424R
AUE   = 001000    DSKADR  000370R   LFE    = 000022   NDB    = 000022   RBUFEA  000130R
ASB     000106R   DTE     = 010000  LOOP1   002202R   NDC    = 000034   RBUFPA  000126R
ASTAT   000104R   DTL     = 100000  LOOP2   002222R   NDS    = 000012   RBUFSZ  000132R
ATA   = 100000    DVA     = 004000  LOS8IT  007446R   NDT    = 000026   RBUFVA  000124R
AWAS    000110R   DVC     = 000200  LOSYSC  007432R   NEC1   = 000044   RDY   = 000200
BADSEC  000376R   DVICE   000372R   LS     = 000004   NEC2   = 000046   READ    002574R
BADSPT  000252R   DVID1   000014R   LSC    = 004000   NED    = 010000   READY   005134R
BEGIN   000000R   EA22    000360R   LT     = 000002   NEM    = 004000   RESTRT  002164R
BITTAB  003666R   ECH     = 000100  MAP22$= 104416   NER1   = 000014   RES1    000056R
BIT0  = 000001    ENDITS= 104413    MCPE   = 020000   NER2   = 000042   RES2    000060R
BIT1  = 000002    ENDS    = 104410  MCPERR  007366R   NHR    = 000036   RHFAIL  007402R
BIT10 = 002000    ERR     = 040000  MDPE   = 000400   NLA    = 000020   RMAS    001520R
BIT11 = 004000    ERRORS  003740R   MDPERR  007372R   NMR1   = 000024   RMBA    001506R
BIT12 = 010000    ERRTYP  000106R   MES1    006012R   NMR2   = 000040   RMBAE   001552R
BIT13 = 020000    ERSUB1  005656R   MES10   006224R   NODRDY  007502R   RMCS1   001502R
BIT14 = 040000    ERSUB2  005632R   MES11   006244R   NODVA   007516R   RMCS2   001512R
BIT15 = 100000    EXCED   007376R   MES12   006277R   NOF    = 000002   RMCS3   001554R
BIT2  = 000004    EXITS = 104400    MES14   006327R   NOTRDY  007512R   RMDA    001510R
BIT3  = 000010    FEONLY  007532R   MES15   006345R   NSN    = 000030   RMDB    001524R
BIT4  = 000020    FER     = 000020  MES16   006373R   NTRUPT  003034R   RMDC    001536R
BIT5  = 000040    FERADR  000406R   MES17   006416R   NULL   = 000000   RMDS    001514R
BIT6  = 000100    FFE     = 000020  MES2    006051R   NUMB    007553R   RMDT    001530R
BIT7  = 000200    FLAG70  000364R   MES20   006433R   NWC    = 000002   RMEC1   001546R
BIT8  = 000400    FMT     = 010000  MES21   006457R   OM     = 000001   RMEC2   001550R
BIT9  = 001000    FORMAT  007416R   MES22   006502R   OPEN   = 000000   RMER1   001516R
BREAK$= 104407    FREE    000150R   MES23   006525R   OPI    = 020000   RMER2   001544R
BR1     000012R   FT      = 000000  MES24   006546R   OR     = 000200   RMHR    001540R
BR2     000013R   FUNC    000402R   MES25   006572R   OTOA$  = 104420   RMLA    001522R
BSE   = 100000    GENADR  003174R   MES26   006616R   OVRWRT  007536R   RMMR1   001526R
BST   = 000010    GET     005530R   MES27   006643R   PAR    = 000010   RMMR2   001542R
BTOD$ = 104421    GETDRV  003630R   MES3    006107R   PASCNT  000034R   RMOF    001534R
CDATA$= 104412    GETMAP  003522R   MES30   006645R   PA18    000352R   RMOFI   000416R
CDERCT  000144R   GETPA$= 104415    MES31   006671R   PA22    000356R   RMR   = 000004
CDWDCT  000146R   GO      = 000001  MES32   006713R   PGE    = 002000   RMSN    001532R
CLK     000410R   GO1     002674R   MES33   006735R   PGM    = 001000   RMWC    001504R
CLR   = 000040    GO2     003000R   MES34   006762R   PIP    = 020000   RSTRT   000112R
CONFIG  000056R   G=BUF$= 104414    MES35   007011R   PIR0$  = 000004   RSTRT1  002174R
CSRA    000100R   HCE     = 000200  MES36   007037R   POPSP  = 005726   R6      =%000006
CYCLE   002276R   HCNC    = 000400  MES37   007104R   POPSP2= 022626    R7      =%000007
CYL   = 000016    HEADER  007412R   MES4    006152R   POSIT   007472R   S       001606R
CYLIND  000366R   HRDCNT  000044R   MES40   007227R   PRTY   = 000000   SBADR   000102R
CYLMAP  000422R   HRDER$= 104405    MES5    006162R   PRTY0  = 000000   SC    = 100000
DATCK$= 104411    HRDPAS  000050R   MES6    006174R   PRTY1  = 000040   SEC   = 000012
DATERR  007442R   IAE     = 002000  MODNAM  000000R   PRTY2  = 000100   SEEK    002654R
DATER$= 104404    ICONT   000036R   MODSP   000252R   PRTY3  = 000140   SEEKER  007426R
DCK   = 100000    ICOUNT  000040R   MOL    = 010000   PRTY4  = 000200   SET8L   001766R
DEVNOT  007506R   IDNUM   000122R   MSGN$  = 104403   PRTY5  = 000240   SETUP   005674R
DLTCNT  000400R   ILF     = 000001  MSGS$  = 104402   PRTY6  = 000300   SKI   = 040000
DLTERR  007476R   ILR     = 000002  MSG$   = 104401   PRTY7  = 000340   SOFCNT  000042R
DPE   = 000010    IMODX.= 000000    MXF    = 001000   PS     = 177776   SOFER$= 104406
DPR   = 000400    INIT    000030R   NAS    = 000016   PSW    = 177776   SOFPAS  000046R
DRM     007522R   INTR    000120R   NBA    = 000004   PUSH   = 005746   SPOINT  000032R
```


SYMBOL TABLE

```
SPSIZ = 000040    SVR2    000066R   TRK    = 000014   WARNIN  007542R   WRITCK  002514R
SR1     000016R   SVR3    000070R   TRKMAP  000420R   WASADR  000104R   WRITE   002434R
SR2     000020R   SVR4    000072R   TRPDFD= 000022    WBUFEA  000136R   WRL   = 004000
SR3     000022R   SVR5    000074R   TRY     000412R   WBUFPA  000134R   WRTCK   007462R
SR4     000024R   SVR6    000076R   UNITNO  000374R   WBUFRQ  000140R   WTCLCK  007452R
SSE   = 000040    SYSCNT  000052R   UNS    = 040000   WBUFSZ  000142R   WTCRDY  005412R
SSEI  = 001000    TABLE   001424R   UNSAFE  007406R   WCE    = 040000   WTLOCK  007436R
START   001662R   TAH80   001562R   UPE    = 020000   WCF    = 000040   XFERAD  001556R
STAT    000026R   TIMERR  007466R   VECTOR  000010R   WDFR    000116R   XFLAG   000005R
SVR0    000062R   TRE     = 040000  VV     = 000100   WDTO    000114R   XMEM    000354R
SVR1    000064R   TRERR   007362R   WAIT    005250R   WLE    = 004000   ZERO    000404R
```

```
. ABS.  000000    000
        007556    001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  12800 WORDS  ( 50 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  70 PAGES
XRNAA0,XRNAA0/C=[20,0]DDXCOM,[20,16]CXRNAA.DOC,CXRNAA
```

```
IDNUM    4-6#
ILF      4-197#
ILR      4-198#
IMODX.   4-6#      6-9
INIT     4-6#
INTR     4-6#      5-5*
IR       4-170#
LBC      4-219#   11-161
LBT      4-188#   11-134
LC       4-128#   10-65
LDADR    5-74     10-104#
LFE      4-134#   10-55     10-87
LOOP1    5-64#     5-82
LOOP2    5-68#     5-72      5-79
LOSBIT  11-163    12-230#
LOSYSC  11-148    12-224#
LS       4-127#   10-37     10-45
LSC      4-220#   11-146
LT       4-126#    5-26*     5-34*   10-50    10-93
MAP22$   4-6#      8-10
MCPE     4-162#   11-45     11-82
MCPERR  11-84     12-206#
MDPE     4-172#   11-86
MDPERR  11-88     12-208#
MES1    12-170#   12-204
MES10   12-176#   12-214
MES11   12-177#   12-216
MES12   12-178#   12-218
MES14   12-179#   12-234
MES15   12-180#   12-212
MES16   12-181#   12-220
MES17   12-182#   12-222
MES2    12-171#   12-206
MES20   12-183#   12-224
MES21   12-184#   12-226
MES22   12-185#   12-228
MES23   12-186#   12-230
MES24   12-187#   12-232
MES25   12-188#   12-236
MES26   12-189#   12-238
MES27   12-190#
MES3    12-172#   12-208
MES30   12-191#   12-240
MES31   12-192#   12-242
MES32   12-193#   12-244
MES33   12-194#   12-246
MES34   12-195#   12-248
MES35   12-196#   12-250
MES36   12-197#   12-256
MES37   12-198#   12-258
MES4    12-173#   12-252
MES40   12-200#   12-260
MES5    12-174#   12-254
MES6    12-175#   12-210
MODNAM   4-6#
MODSP    4-6       4-6#
```



```
MOL      4-190#   11-24
MSG$     4-6#
MSGN$    4-6#      5-12      5-15      5-17      6-35      9-27     11-17     11-27     11-41     11-84     11-88     11-94     11-103    11-111
        11-115    11-122    11-126    11-130    11-136    11-140    11-144    11-148    11-151    11-155    11-159    11-163    11-167    11-169
        12-20     12-26     12-56     12-87     12-92
MSGS$    4-6#
MXF      4-173#   11-90
NAS      4-55      4-107#
NBA      4-50      4-102#
NBAE     4-68      4-120#   12-119
NCS1     4-48      4-100#   11-12     11-32     11-45     11-82     11-113
NCS2     4-52      4-104#   11-86     11-90     11-96     11-109
NCS3     4-69      4-121#
NDA      4-51      4-103#
NDB      4-57      4-109#
NDC      4-62      4-114#
NDS      4-53      4-105#   11-22     11-36     11-47     11-92     11-107    11-134
NDT      4-59      4-111#
NEC1     4-66      4-118#
NEC2     4-67      4-119#
NED      4-176#   11-90
NEM      4-175#   11-90
NER1     4-54      4-106#   11-70     11-72     11-120    11-124    11-128    11-132    11-138    11-149    11-153    11-157    11-165
NER2     4-65      4-117#   11-74     11-76     11-142    11-146    11-161    12-61
NHP      4-63      4-115#
NLA      4-56      4-108#
NMR1     4-58      4-110#
NMR2     4-64      4-116#
NODRDY  11-27     12-244#
NODVA   12-56     12-250#
NOF      4-61      4-113#
NOIRDY  11-17     12-87    12-248#
NSN      4-60      4-112#
NTRUPT   9-3#     12-164
NULL     4-6#
NUMB    12-253    12-264#
NWC      4-49      4-101#
OM       4-183#
OPEN     4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6
         4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6
         4-6#
OPI      4-210#   11-136
OR       4-171#   12-108
OTOA$    4-6#     10-145
OVP#RT   5-15     12-258#
PA18     4-20#     8-5*      8-10
PA22     4-22#     8-11
PAR      4-200#
PASCNT   4-6#      5-58
PGE      4-174#   11-90
PGM      4-187#
PIP      4-191#   11-37
PIRQ$    4-6#      9-3
POPSP    4-6#
POPSP2   4-6#
POSIT   11-140    12-240#
```

```
PRTY     4-6#
PRTY0    4-6      4-6#
PRTY1    4-6#
PRTY2    4-6#
PRTY3    4-6#
PRTY4    4-6#
PRTY5    4-6      4-6#
PRTY6    4-6#
PRTY7    4-6#
PS       4-6#
PSW      4-6#
PUSH     4-6#
PUSH2    4-6#
PWRFLG   4-6#
R6       4-6#     5-7*
R7       4-6#
RANDS    4-6#     10-12    10-15    10-18
RANNUM   4-6#     10-13    10-16    10-19
RBUF     4-6      4-44#
RBUFEA   4-6#     7-44
RBUFPA   4-6#     6-28     7-43
RBUFSZ   4-6#     7-40
RBUFVA   4-6#     5-62
RDY      4-164#   11-13    11-14    12-80    12-81
READ     6-26     7-38#    7-39
READY    5-48     5-71     9-12     12-10#
RES1     4-6#
RES2     4-6#
RESTRT   4-6      5-58#
RHFAIL   11-94    12-212#
RMAS     4-79#
RMBA     4-74#    7-25*    7-34*    7-43*    8-5      8-11*
RMBAE    4-92#    8-12*    12-150
RMCS1    4-72#    5-50*    5-52*    8-21*    11-11*   11-40*   11-44*   12-45    12-47*   12-49    12-60*   12-63*   12-79    12-105
         12-136   12-137   12-147   12-152*  12-153
RMCS2    4-76#    5-64*
RMCS3    4-93#
RMDA     4-75#    8-20*
RMDB     4-81#    12-106
RMDC     4-86#    8-19*
RMDS     4-77#
RMDT     4-83#    12-12
RMEC1    4-90#
RMEC2    4-91#
RMER1    4-78#
RMER2    4-89#
RMHR     4-87#
RMLA     4-80#
RMMR1    4-82#
RMMR2    4-88#
RMOF     4-85#    8-18#
RMOFI    4-40#    5-68*    8-18     9-18*
RMR      4-199#
RMSN     4-84#
RMWC     4-73#    7-24*    7-33*    7-42*
RSTRT    4-6#
```



```
RSTRT1   5-56     5-59     5-62#
S        4-48     4-49     4-50     4-51     4-52     4-53     4-54     4-55     4-56     4-57     4-58     4-59     4-60     4-61
         4-62     4-63     4-64     4-65     4-66     4-67     4-68     4-69     4-156#   11-12    11-22    11-32    11-36    11-45
         11-47    11-70    11-72    11-74    11-76    11-82    11-86    11-90    11-92    11-96    11-107   11-109   11-113   11-120
         11-124   11-128   11-132   11-134   11-138   11-142   11-146   11-149   11-153   11-157   11-161   11-165   12-61    12-110*
         12-112*  12-119   12-120*  12-122*
SBADR    4-6#     12-130*
SC       4-160#
SEC      4-130#   5-36*    10-13*   10-14*   10-37    10-39*   10-44*   10-45    10-47*   10-105
SEEK     6-12     6-18     6-24     7-76#    7-77
SEEKER   11-41    11-144   12-222#
SETBL    5-24#
SETUP    5-43     12-145#
SKI      4-221#   11-142   12-61
SOFCNT   4-6#
SOFERS   4-6#     11-105   11-172
SOFPAS   4-6#
SPUINT   4-6#     5-7
SPSIZ    1-23#    4-6
SR1      4-6#     5-11*    5-13     5-27     5-39     10-10    10-26    10-57    10-82    11-99    12-156
SR2      4-6#
SR3      4-6#
SR4      4-6#
SSE      4-217#   9-16     11-74
SSEI     4-227#   9-18
START    4-6      5-3#     5-60
STAT     4-6#
SVR0     4-6#
SVR1     4-6#
SVR2     4-6#
SVR3     4-6#
SVR4     4-6#
SVR5     4-6#
SVR6     4-6#
SYSCNF   4-6#
TAB80    4-145#   5-24     10-9     10-104
TABLE    4-48#    11-19    11-29    11-105   11-118   11-172   12-24    12-90    12-116*  12-119*
TIMERR   11-155   12-238#
TRE      4-161#   11-45    11-113
TRERR    11-115   12-204#
TRK      4-131#   5-37*    10-35*   10-49*   10-50    10-52*   10-67    10-72    10-106
TRKMAP   4-41#    10-31    10-33    10-95*
TRPDFD   4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6
         4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6
         4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6      4-6#     4-6#     4-6#     4-6#     4-6#     4-6#
         4-6#     4-6#     4-6#     4-6#     4-6#     4-6#     4-6#     4-6#     4-6#     4-6#     4-6#     4-6#     4-6#
TRY      4-38#    6-8*     9-24*    9-25     9-28*
UNITNO   4-30#    5-44*    5-65*    10-117*  10-118   10-120   10-139   10-145
UNS      4-211#   11-120
UNSAFE   11-122   12-214#
UPE      4-177#   11-90
VECTOR   4-6#     12-163
VV       4-184#   11-24
WAIT     6-10     6-16     6-22     12-10    12-45#
WARNIN   5-12     12-260#
WASADR   4-6#     12-132*
```

```
WBUFEA    4-6#      7-26      7-35
WBUFPA    4-6#      7-25      7-34
WBUFRQ    4-6#
WBUFSZ    4-6#      7-22      7-31
WCE       4-178#    11-109
WCF       4-202#    11-165
WDFR      4-6#      5-4*
WDTO      4-6#      5-3*
WLE       4-208#    11-149
WRITCK    6-20      7-29#     7-30
WRITE     6-14      7-20#     7-21
WRL       4-189#
WRTCK     11-111    12-236#
WTCLCK    11-167    12-232#
WTCRDY    5-53      12-64     12-78#
WTLOCK    11-151    12-226#
XFERAD    4-95#
XFLAG     4-6#
XMEM      4-21#     7-26*     7-35*     7-44*     8-6*      8-7*      8-8*      8-9*      8-15*     8-16
ZERO      4-35#
```



```
BKMOD     1-111#
BREAK     1-201#    12-51     12-83
BTOD      1-220#
CKDATA    1-256#    6-28
DATACK    1-265#
DATERR    1-154#
DFSEVN    1-288#    4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6       4-6
DSEVNT    1-298#    4-6       4-6       4-6       4-6       4-6       4-6
END       1-191#
ENDIT     1-182#    5-78
ENDMOD    1-187#    5-84
EQUATS    1-306#    4-6
EXIT      1-136#    8-22
GETPA     1-247#    5-62
GWBUFF    1-235#    6-9
HRDER     1-144#    11-19     11-29     11-118    12-24     12-90
IOMOD     1-107#
IOMODP    1-131#
IOMODR    1-127#
IOMODX    1-123#    4-6
LINEUP    6-38#     7-26      7-35      7-44
MAP22     1-251#    8-10
MODULE    1-24#     4-6
MSG       1-170#
MSGN      1-174#    5-12      5-15      5-17      6-35      9-27      11-17     11-27     11-41     11-84     11-88     11-94     11-103    11-111
          11-115    11-122    11-126    11-130    11-136    11-140    11-144    11-148    11-151    11-155    11-159    11-163    11-167    11-169
          12-20     12-26     12-56     12-87     12-92
MSGS      1-178#
NBKMOD    1-119#
OTOA      1-206#    10-145
PIRQ      1-195#    9-3
RAND      1-140#    10-12     10-15     10-18
SBKMOD    1-115#
```

.REM %

IDENTIFICATION

PRODUCT CODE: AC-T087A-MC

PRODUCT NAME: CXBTCA0 Q22 BUS EXER DEC/X11

PRODUCT DATE: DECEMBER 1981

MAINTAINER: DIAGNOSTIC ENGINEERING

HISTORY

REVISION A FIRST RELEASE OF DIAGNOSTIC

```
                                ;*
                                ;*************************************************************
                                ;
                                ;
                                ;       Q22BE DEC/X11 MODULE
                                ;
                                ;       ABSTRACT
                                ;       REQUIREMENTS  (HARDWARE AND SOFTWARE)
                                ;       PASS DEFINITIONS
                                ;       EXECUTION TIME
                                ;       CONFIGURATION REQUIREMENTS
                                ;       DEVICE/OPTION SETUPS
                                ;       TEST SEQUENCE
                                ;       OPERATION OPTIONS
                                ;       NON-STANDARD PRINTOUTS
                                ;
                                ;
                                ;       1. ABSTRACT

                                ;       BTCA IS AN IUMODX THAT EXERCISES AND TESTS THE Q22BE IN
                                ;       A SYSTEMS ENVIRONMENT. THE PURPOSE OF THE Q22BE IS TO
                                ;       CAUSE SYSTEM BUS ACTIVITY AT A HIGH LEVEL. THIS SOFTWARE
                                ;       MODULE WILL HAVE THE Q22BE PERFORM DMA TRANSFERS USING
                                ;       DATA-OUTS AND DATA-INS. WHEN PERFORMING DATA-OUT
                                ;       TRANSFERS THE DATA WILL BE CHECKED FOR VALIDITY. ALL
                                ;       AVAILABLE INTERRUPT LEVELS WILL BE EXERCISED. ANY ERRORS
                                ;       DETECTED WILL BE REPORTED ON THE SYSTEM CONSOLE.

                                ;
                                ;
                                ;       2. REQUIREMENTS
                                ;
                                ;       THIS MODULE WILL REQUIRE APPROXIMATELY 2K WORDS OF MEMORY.
                                ;
                                ;       THE SYSTEM MUST CONTAIN AT LEAST 16K OF MEMORY.
                                ;
                                ;       A TYPICAL SYSTEM COULD CONTAIN MORE THAN ONE Q22BE,
                                ;       ALTHOUGH ONLY ONE IS NECESSARY.
                                ;
                                ;
                                ;       3. PASS DEFINITION
                                ;
                                ;       ONE ITERATION CONSISTS OF SETTING UP AND PERFORMING
                                ;       DMA OPERATIONS. ONE PASS WILL CONSIST OF MULTIPLE ITERATIONS.
                                ;
                                ;
                                ;       4. EXECUTION TIME
                                ;
                                ;       ONE PASS WILL RUN IN APPROXIMATELY ONE MINUTE.
                                ;
                                ;
                                ;       5. CONFIGURATION REQUIREMENTS
                                ;
```



```
                                ;       REFERENCE THE Q22 BUS EXERCISER SPECIFICATION FOR THE
                                ;       DEVICE PARAMETERS. THE DEFAULT PARAMETERS WILL BE AS FOLLOWS:
                                ;
                                ;       ADRS  170000      VECTOR 510     DEVENT 1
                                ;
                                ;
                                ;
                                ;       6. DEVICE/OPTION SETUPS
                                ;
                                ;       APPROPRIATE MODULE LEVEL DIAGNOSTICS MUST BE RUN TO
                                ;       INITIALLY ENSURE THAT THE SYSTEM COMPONENTS ARE FUNCTIONAL.
                                ;
                                ;
                                ;       7. TEST SEQUENCE
                                ;
                                ;
                                ;
                                ;       8. OPERATION OPTIONS
                                ;
                                ;       THIS MODULE IS LOADED AND RUN AS SPECIFIED IN THE DEC/X11
                                ;       REFERENCE GUIDE.
                                ;
                                ;       SR1 MUST BE SET UP BY THE OPERATOR PRIOR TO RUNNING
                                ;       THIS MODULE.
                                ;
                                ;       BIT0    CLEAR    MODULE WILL NOT ATTEMPT ANY BLOCK MODE
                                ;                        TRANSFERS
                                ;       BIT0    SET      MODULE WILL ATTEMPT BOTH NON-BLOCK AND
                                ;                        BLOCK MODE TRANSFERS. (ALL BLOCK MODE
                                ;                        MEMORY)
                                ;       BIT1    CLEAR    MODULE WILL ATTEMPT ONLY 1K WORD XFRS
                                ;       BIT1    SET      MODULE WILL ATTEMPT ONE DATA-IN 16K WORD
                                ;                        TRANSFER IN ADDITION TO OTHER
                                ;                        TRANSFERS OF 1K WORDS
                                ;
                                ;
                                ;
                                ;       9. ERROR REPORTING
                                ;
                                ;       ALL ERRORS WILL BE REPORTED AS THEY OCCUR. ERROR REPORTS
                                ;       WILL CONFORM TO THE STANDARD DEC/X11 FORMAT.
                                ;
                                ;       Q22BE REGISTERS WILL BE REPORTED IN THE FOLLOWING
                                ;       SEQUENCE.
                                ;
                                ;       CSR1    CSR2    BA      WC      DATA    LATCNT  MVLCNT
                                ;
                                ;
                                ;*************************************************************

                                        ;THIS FILE IS THE BEGINNING OF A NEW SOFTWARE
                                        ;MODULE FOR THE Q22 BUS EXERCISER. IT WILL BE
                                        ;THE DEC/X11 Q22 BUS EXERCISER PROGRAM. ALL
                                        ;NRECESSARY SET UP FUNCTIONS WILL BE INCLUDED
                                        ;IN THIS SECTION OF THE MODULE.
```

```
                        IDMODX  <BTCA >,170000,510,7,0,0,10,www,RBUF,256,16000
                                MODULE  150000,BTCA ,170000,510,7,0,0,10,WWW,RBUF,256,16000
                                .TITLE  BTCA DEC/X11 SYSTEM EXERCISER MODULE
                        ;       DDXCOM  VERSION  6      23-MAY-78
                                .LIST   BIN
                        ;*****************************************************************************
000000'                 BEGIN:
000000' 052102  040503      040 MODNAM: .ASCII  /BTCA / ;MODULE NAME.
000005'     000         XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF RBUFF USAGE
000006' 170000          ADDR:   170000+0                ;1ST DEVICE ADDR.
000010' 000510          VECTOR: 510+0                   ;1ST DEVICE VECTOR.
000012'     340         BR1:    .BYTE   PRTY7+0         ;1ST BR LEVEL.
000013'     000         BR2:    .BYTE   PRTY0+0         ;2ND BR LEVEL.
000014' 000001          DVID1:  0+1                     ;DEVICE INDICATOR 1.
000016' 000000          SR1:    OPEN                    ;SWITCH REGISTER 1
000020' 000000          SR2:    OPEN                    ;SWITCH REGISTER 2
000022' 000000          SR3:    OPEN                    ;SWITCH REGISTER 3
000024' 000000          SR4:    OPEN                    ;SWITCH REGISTER 4
                        ;*****************************************************************************
000026' 150000          STAT:   150000                  ;STATUS WORD.
000030' 000536'         INIT:   START                   ;MODULE START ADDR.
000032' 000252'         SPOINT: MODSP                   ;MODULE STACK POINTER.
000034' 000000          PASCNT: 0                       ;PASS COUNTER.
000036' 000010          ICONT:  10                      ;# OF ITERATIONS PER PASS=10
000040' 000000          ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
000042' 000000          SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
000044' 000000          HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
000046' 000000          SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
000050' 000000          HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
000052' 000000          SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
000054' 000000          RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000056'                 CONFIG:                         ;RESERVED FOR MONITOR USE
000056' 000000          RES1:   0                       ;RESERVED FOR MONITOR USE
000060' 000000          RES2:   0                       ;RESERVED FOR MONITOR USE
000062' 000000          SVR0:   OPEN                    ;LOC TO SAVE R0.
000064' 000000          SVR1:   OPEN                    ;LOC TO SAVE R1.
000066' 000000          SVR2:   OPEN                    ;LOC TO SAVE R2.
000070' 000000          SVR3:   OPEN                    ;LOC TO SAVE R3.
000072' 000000          SVR4:   OPEN                    ;LOC TO SAVE R4.
000074' 000000          SVR5:   OPEN                    ;LOC TO SAVE R5.
000076' 000000          SVR6:   OPEN                    ;LOC TO SAVE R6.
000100' 000000          CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
000102'                 SBADR:                          ;ADDR OF GOOD DATA, OR
000102' 000000          ACSR:   OPEN                    ;CONTENTS OF CSR.
000104'                 WASADR:                         ;ADDR OF BAD DATA, OR
000104' 000000          ASTAT:  OPEN                    ;STATUS REG CONTENTS.
000106'                 ERRTYP:                         ;TYPE OF ERROR
000106' 000000          ASB:    OPEN                    ;EXPECTED DATA.
000110' 000000          AWAS:   OPEN                    ;ACTUAL DATA.
000112' 001054'         RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
000114' 000000          WDTO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
000116' 000000          WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
000120' 000000          INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
000122' 000000G         IDNUM:  WWW                     ;MODULE IDENTIFICATION NUMBER=WWW
000124' 004240'         RBUFVA: RBUF                    ;READ BUFFER VIRTUAL ADDRESS
000126' 000000          RBUFPA: OPEN                    ;READ BUFFER PHYSICAL ADDRESS
```



```
000130' 000000          RBUFEA: OPEN                    ;READ BUFFER EA BITS
000132' 000256          RBUFSZ: 256                     ;SIZE OF THE READ BUFFER
000134' 000000          WBUFPA: OPEN                    ;WRITE BUFFER PHYSICAL ADDRESS
000136' 000000          WBUFEA: OPEN                    ;WRITE BUFFER EA BITS
000140' 016000          WBUFRQ: 16000                   ;WRITE BUFFER SIZE REQUESTED
000142' 000000          WBUFSZ: OPEN                    ;WRITE BUFFER SIZE AVAILABLE
000144' 000000          CDERCT: OPEN                    ;CDATA/DATCK ERROR COUNT
000146' 000000          CDWDCT: OPEN                    ;CDATA/DATCK WORD COUNT
000150' 000000          FREE:   OPEN                    ;RESERVED FOR FUTURE USE
        000040                  .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                .NLIST
                                .WORD   0
                                .LIST
                                .ENDR
000252'                 MODSP:
                        ;*****************************************************************************
                        ;*************************************************************

                        .LIST MC
                        .NLIST ME
                        .MCALL  STRUCT
000252'                 STRUCT
        000001          $LSTIN=1
        000001          $LSTTAG=1
        000001          $LOCTAG=1
                        .TITLE  DEC/X11 VERSION OF Q22BE


                        ;*COPYRIGHT (C) 1980
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY EUGENE S. REDNER
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING SPMAC
                        ;*
                        .SBTTL  BASIC DEFINITIONS


                        ;****** ************************************************************

000252' 000000          SVINLV: .WORD   0               ;SAVE INTERRUPT LEVEL
000254' 000000          DV:     .WORD   0
000256' 000000          MORE:   .WORD   0
000260' 000011          Q22TB:  .BLKW   11              ;TABEL FOR ADRS OF Q22BE
000302' 000011          VECT:   .BLKW   11              ;DEVICE VECTOR TABLE
000324' 000000          DEVADR: .WORD   0               ;ADDRESS
000326' 000000          DEVECT: .WORD   0               ;VECTOR
000330'                 TABLE1:
000330' 000000          CSR1:   .WORD   0               ;CONTROL REGISTER ONE
000332' 000000          CSR2:   .WORD   0               ;TWO
000334' 000000          BA:     .WORD   0               ;ADDRESS REGISTER
000336' 000000          WC:     .WORD   0               ;WORD COUNTER
000340' 000000          DATA:   .WORD   0               ;DATA REGISTER
000342' 000000          LATCNT: .WORD   0               ;LATENCY COUNTER
000344' 000000          MVLCNT: .WORD   0               ;MAX VALUE OF ABOVE
000346' 000000          SIMGOA: .WORD   0               ;SIMULTANEOUS GO REGISTER
```

```
000350' 000000                  ADR:    .WORD   0
000352' 000000                  XMEM:   .WORD   0               ;22 BIT STUFF
000354' 000000                  PA22:   .WORD   0
000356' 000000                  EA22:   .WORD   0
000360' 000000                  SVADR:  .WORD   0               ;PLACE TO SAVE ADR
                                ;***********************************************
                                ;* DATA BIT DEFINITIONS

        100000                  BIT15=  100000
        040000                  BIT14=  40000
        020000                  BIT13=  20000
        010000                  BIT12=  10000
        004000                  BIT11=  4000
        002000                  BIT10=  2000
        001000                  BIT09=  1000
        000400                  BIT08=  400
        000200                  BIT07=  200
        000100                  BIT06=  100
        000040                  BIT05=  40
        000020                  BIT04=  20
        000010                  BIT03=  10
        000004                  BIT02=  4
        000002                  BIT01=  2
        000001                  BIT00=  1


                                ;*********************************************************

000362'                         VECT0:
000362'                         LET SVINT := SVINT SET.BY #BIT00
000362' 052767  000001  000132                                                  BIS     #BIT00,SVINT
000370'                         INLINE <JMP     INTSRV>
000370' 000167  003026                                                          JMP     INTSRV
                                                        ;THE DEVICE INTERRUPT VECTORS
                                                        ;WILL BE STORED IN THE WORD
                                                        ;KNOWN AS SVINT. AS EACH INTRP
                                                        ;HAPPENS IT WILL COME THRU THIS
                                                        ;AREA BEFORE GOING TO THE INTR
                                                        ;SERVICE ROUTINE.

000374'                         VECT1:
000374'                         LET SVINT := SVINT SET.BY #BIT01
000374' 052767  000002  000120                                                  BIS     #BIT01,SVINT
000402'                         INLINE <JMP     INTSRV>
000402' 000167  003014                                                          JMP     INTSRV

000406'                         VECT2:
000406'                         LET SVINT := SVINT SET.BY #BIT02
000406' 052767  000004  000106                                                  BIS     #BIT02,SVINT
000414'                         INLINE <JMP     INTSRV>
000414' 000167  003002                                                          JMP     INTSRV

000420'                         VECT3:
000420'                         LET SVINT := SVINT SET.BY #BIT03
000420' 052767  000010  000074                                                  BIS     #BIT03,SVINT
000426'                         INLINE <JMP     INTSRV>
```



```
000426' 000167  002770                                                          JMP     INTSRV

000432'                         VECT4:
000432'                         LET SVINT := SVINT SET.BY #BIT04
000432' 052767  000020  000062                                                  BIS     #BIT04,SVINT
000440'                         INLINE <JMP     INTSRV>
000440' 000167  002756                                                          JMP     INTSRV

000444'                         VECT5:
000444'                         LET SVINT := SVINT SET.BY #BIT05
000444' 052767  000040  000050                                                  BIS     #BIT05,SVINT
000452'                         INLINE <JMP     INTSRV>
000452' 000167  002744                                                          JMP     INTSRV

000456'                         VECT6:
000456'                         LET SVINT := SVINT SET.BY #BIT06
000456' 052767  000100  000036                                                  BIS     #BIT06,SVINT
000464'                         INLINE <JMP     INTSRV>
000464' 000167  002732                                                          JMP     INTSRV

000470'                         VECT7:
000470'                         LET SVINT := SVINT SET.BY #BIT07
000470' 052767  000200  000024                                                  BIS     #BIT07,SVINT
000476'                         INLINE <JMP     INTSRV>
000476' 000167  002720                                                          JMP     INTSRV

                                ;*********************************************************


                                ;***********************************************
                                ;*  MORE EQUATES


000502' 000000                  BLKMBT: .WORD   0               ;BLOCK MODE BIT
000504' 000000                  INTRP:  .WORD   0               ;NUM OF INTRS THAT SHOULD HAPPEN
000506' 000000                  MASK:   .WORD   0
000510' 170000                  FIRADR: .WORD   170000
000512' 000510                  FIRVEC: .WORD   510
000514' 000000                  LPCNTR: .WORD   0               ;LOOP CNTR LOCATION
000516' 000000                          .WORD   0
000520' 000000                  SIMGBT: .WORD   0               ;SIMULT GO BIT
000522' 000000                  SVINT:  .WORD   0
000524' 000000                  INTCNT: .WORD   0               ;PLACE TO SAVE INTRS
000526' 000000                  TMPSR2: .WORD   0               ;PLACE TO BUILD UP CSR2
000530' 000000                  TMPR0:  .WORD   0
000532' 000000                  SVWDCT: .WORD   0
000534' 000000                  RY:     .WORD   0               ;USED IN SUBROUTINE 'GO'


                                ;*************************************************************

000536'                         START:

                                ;*********************************************************
```

```
000536'                         LET RBUFVA := #RBUF
000536' 012767  004240' 177360                                          MOV     #RBUF,RBUFVA
000544'                         LET R0 := RBUFVA                ;THIS CODE IS NECESSARY SINCE
000544' 016700  177354                                                  MOV     RBUFVA,R0
000550'                         LET SVWDCT := RBUFSZ            ;THE Q22BE MUST TRANSFERS START AT ADDRS
000550' 016767  177356  177754                                          MOV     RBUFSZ,SVWDCT
000556'                         LET R0 := R0 CLR.BY #177700     ;ZERO OR FORTY WHEN IN BLOCK MODE
000556' 042700  177700                                                  BIC     #177700,R0
000562'                         WHILE R0 NE #40 AND R0 NE #0 DO
000562'                                                                 $50000:
000562' 020027  000040                                                  CMP     R0,#40
000566' 001414                                                          BEQ     $50001
000570' 005700                                                          TST     R0
000572' 001412                                                          BEQ     $50001
000574'                             LET R0 := R0 + #2           ;TRY NEXT LOCATION
000574' 062700  000002                                                  ADD     #2,R0
000600'                             LET RBUFVA := RBUFVA + #2   ;UPDATED STRT LOC
000600' 062767  000002  177316                                          ADD     #2,RBUFVA
000606'                             LET SVWDCT := SVWDCT - #1   ;DEC WORD COUNT
000606' 005367  177720                                                  DEC     SVWDCT
000612'                             LET R0 := R0 CLR.BY #BIT06  ;WATCH FOR OVFLOW
000612' 042700  000100                                                  BIC     #BIT06,R0
000616'                         ENDDO
000616' 000761                                                          BR      $50000
000620'                                                                 $50001:
000620'                         LET INTRP := #0                 ;ZERO THE INTR COUNTER
000620' 005067  177660                                                  CLR     INTRP
000624'                         LET DV := DVID1
000624' 016767  177164  177422                                          MOV     DVID1,DV
000632'                         LET DV := DV CLR.BY #177400
000632' 042767  177400  177414                                          BIC     #177400,DV

                                ;*****************************************************
                                ;       IF A DEVICE IS DROPPED THE PROGRAM RE-ENTRY
                                ;       POINT IS HERE. GET HERE FROM THE SUBROUTINE 'DRPDEV'
                                ;*****************************************************

000640'                         INLINE <REENTY:>                ;RE-ENTRY HERE
000640'                                                                 REENTY:
000640'                         LET MASK := DV                  ;SHOULD ONLY BE 10 DEVICES
000640' 016767  177410  177640                                          MOV     DV,MASK
000646'                         LET MORE := DV
000646' 016767  177402  177402                                          MOV     DV,MORE
000654'                         LET R0 := DV
000654' 016700  177374                                                  MOV     DV,R0
000660'                         IF R0 EQ #0 THEN
000660' 005700                                                          TST     R0
000662' 001005                                                          BNE     $50002

                                    .NLIST MC
                                    .LIST ME
000664' 104403  000000' 003774'         MSGN$,BEGIN,TEXT0       ;ASCII MESSAGE CALL WITH COMMON HEADER
000672' 104410  000000'                 END$,BEGIN              ;
                                    .LIST MC
                                    .NLIST ME
000676'                         ENDIF                           ;GO AWAY, THERE ARE NONE
```



```
000676'                                                                 $50002:

                                ;*************************************************************

000676'                         INLINE <CLC>                    ;CLR THE C BIT
000676' 000241                                                          CLC
000700'                         WHILE R0 NE #0 DO               ;R0 HAS DEVICES
000700'                                                                 $50003:
000700' 005700                                                          TST     R0
000702' 001405                                                          BEQ     $50004
000704'                             LET R0 := R0 SHIFT -1       ;ASR
000704' 006200                                                          ASR     R0
000706'                             IFCOND CS THEN              ;A DEVICE AT THIS LOC
000706' 103002                                                          BCC     $50005
000710'                                 LET INTRP := INTRP + #1 ;YES, ADD 1 TO EXPECTED
000710' 005267  177570                                                  INC     INTRP
000714'                             ENDIF                       ;INTERRUPTS
000714'                                                                 $50005:
000714'                         ENDDO                           ;GO BACK TO CHK MORE
000714' 000771                                                          BR      $50003
000716'                                                                 $50004:

                                ;* ****** ***********************************************
                                ;THE PRECEDING CODE WILL DETERMINE IF THERE ARE ANY
                                ;DEVICES SELECTED BY THE OPERATOR AND WHERE THEY ARE
                                ;IN THE ADDRESS FIELD
                                ;***** ****** ***********************************************

000716'                         LET R6 := #MODSP                ;INITIALIZE THE STACK POINTER
000716' 012706  000252'                                                 MOV     #MODSP,R6

                                ;************************************ ******************
                                ;THE FOLLOWING CODE WILL FIND WHAT THE ADDRESSES AND
                                ;VECTORS ARE OF THE DEVICES THAT ARE THERE.
                                ;*****************************************************

000722'                         LET R0 := #22                   ;CLR TABLES
000722' 012700  000022                                                  MOV     #22,R0
000726'                         LET R1 := #Q22TB                ;ADRS OF TABLE INTO R1
000726' 012701  000260'                                                 MOV     #Q22TB,R1
000732'                         WHILE R0 NE #0 DO
000732'                                                                 $50006:
000732' 005700                                                          TST     R0
000734' 001403                                                          BEQ     $50007
000736'                             LET (R1)+ := #0
000736' 005021                                                          CLR     (R1)+
000740'                             LET R0 := R0 - #1           ;DECREMENT COUNTER
000740' 005300                                                          DEC     R0
000742'                         ENDDO
000742' 000773                                                          BR      $50006
000744'                                                                 $50007:
```

```
000744'                              LET DEVADR := FIRADR           ;1 ST ADDRS INTO DEVADR
000744' 016767  177540  177352                                                     MOV     FIRADR,DEVADR
000752'                              LET DEVECT := FIRVEC           ;FIRST VECTOR IN DEVECT
000752' 016767  177534  177346                                                     MOV     FIRVEC,DEVECT
000760'                              LET R4 := #Q22TB
000760' 012704  000260'                                                            MOV     #Q22TB,R4
000764'                              LET R3 := #VECT                ;SET UP TABLE POINTERS
000764' 012703  000302'                                                            MOV     #VECT,R3

                             ;***********************************************************

000770'                              WHILE MORE NE #0 DO            ;MORE HAS DEVICES
000770'                                                                            $50010:
000770' 005767  177262                                                             TST     MORE
000774' 001427                                                                     BEQ     $50011
000776'                                  INLINE <CLC>               ;CLR THE C BIT
000776' 000241                                                                     CLC
001000'                                  LET MORE := MORE SHIFT -1      ;IS THERE A DEVICE
001000' 006267  177252                                                             ASR     MORE
001004'                                  IFCOND CS THEN             ;IS C SET ?
001004' 103003                                                                     BCC     $50012
001006'                                      CALL CLRREG            ;YES, GO CLR REGISTERS AND
001006' 004767  000716                                                             JSR     PC,CLRREG
                                                                    ;SET UP TABLES
001012'                                  ELSE                       ;NO,A DEVICE IS NOT HERE
001012' 000417                                                                     BR      $50013
001014'                                                                            $50012:
001014'                                      LET DEVADR := DEVADR + #20     ;GET NXT ADRS
001014' 062767  000020  177302                                                     ADD     #20,DEVADR
001022'                                      LET DEVECT := DEVECT + #4      ;NXT VECTOR
001022' 062767  000004  177276                                                     ADD     #4,DEVECT
001030'                                      IF DEVADR EQ #170200 THEN
001030' 026727  177270  170200                                                     CMP     DEVADR,#170200
001036' 001005                                                                     BNE     $50014

                                             .NLIST MC
                                             .LIST ME
001040' 104403  000000' 003770'              MSGN$,BEGIN,TEXT1      ;ASCII MESSAGE CALL WITH COMMON HEADER
001046' 104410  000000'                      ENDS,BEGIN             ;
                                             .LIST MC
                                             .NLIST ME

001052'                                      ENDIF
001052'                                                                            $50014:
001052'                                  ENDIF
001052'                                                                            $50013:
001052'                              ENDDO
001052' 000746                                                                     BR      $50010
001054'                                                                            $50011:

                             ;*****************************************************
                             ;THE PRECEEDING CODE FOUND THE ADRS/VECTOR PAIRS
                             ;AND ALSO PLACED THE DEVICE ADRS IN THE CORRECT
                             ;TABLE.
```



```
                             ;*******************************************************

001054'                      RESTRI:

                             ;*******************************************************

                             .NLIST MC
                             .LIST ME
001054' 104415  000000' 000124'       GETPAS,BEGIN, RBUFVA          ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
                             .LIST MC
                             .NLIST ME
001062'                      LET BLKMBT := #0                       ;BLOCK MODE BIT TO ZERO
001062' 005067  177414                                                             CLR     BLKMBT

001066'                      LET RY := @LATCNT                      ;CLR LAT CNTR BY READING IT
001066' 017767  177250  177440                                                     MOV     @LATCNT,RY

001074'                      LET INTCNT := #0                       ;INTR COUNTER WORD TO 0
001074' 005067  177424                                                             CLR     INTCNT
001100'                      LET R6 := #MODSP
001100' 012706  000252'                                                            MOV     #MODSP,R6
001104'                      LET SIMGBT := #0                       ;SIM GO BIT INIT TO ZERO
001104' 005067  177410                                                             CLR     SIMGBT
                             .NLIST MC
                             .LIST ME
001110' 104414  000000'               GWBUFS, BEGIN                 ;GET WRITE BUFFER INFORMATION
                             .LIST MC
                             .NLIST ME


                             ;***** ************************************************
                             ;WE WILL NOW LOOK TO SEE IF THERE ARE ANY Q22'S IN
                             ;THE APPROPRIATE TABLE.
                             ;************************************************* **********

001114'                      LET DEVADR := Q22TB                    ;FIRST ADRS IN THE TABLE
001114' 016767  177140  177202                                                     MOV     Q22TB,DEVADR
001122'                      LET R4 := #Q22TB                       ;FIRST LOC OF TABLE
001122' 012704  000260'                                                            MOV     #Q22TB,R4
001126'                      LET R3 := #VECT                        ;ADRS OF VECTOR TABLE
001126' 012703  000302'                                                            MOV     #VECT,R3

001132'                      INLINE <ESR13:>                        ;LOCATION
001132'                                                                            ESR13:
001132'                      IF DEVADR EQ #0 THEN                   ;IS CONTENTS OF LOC EQUAL
001132' 005767  177166                                                             TST     DEVADR
001136' 001002                                                                     BNE     $50015
001140'                          INLINE <JMP SIMGO>
001140' 000167  000352                                                             JMP     SIMGO
001144'                      ENDIF
001144'                                                                            $50015:
001144'                      LET DEVECT := (R3)                     ;FOUND THE RIGHT ONE
001144' 011367  177156                                                             MOV     (R3),DEVECT

                             ;******* ***********************************************
                             ;WE NOW HAVE THE Q22 ADRS AND VECTOR
```

```
                        ;*****************************************************

                        LET SVINLV := #7                ;START W/INTR LEVEL SEVEN
001150' 012767  000007  177074                                          MOV     #7,SVINLV
                        CALL    ADRSET                  ;GO SET ADRS FOR REGISTERS
001156' 004767  001122                                                  JSR     PC,ADRSET
                                                        ;AND PSW, GET READY TO DO
                                                        ;SOME DATA OUT XFRS
                        ;*****************************************************
                        ;               DATA OUT TRANSFERS
                        ;* ***************************************************

                        #WHILE SVINLV NE #3 DO
001162'                                                                 $50016:
001162' 026727  177064  000003                                          CMP     SVINLV,#3
001170' 001542                                                          BEQ     $50017

001172'
                            IF BLKMBT EQ #1 THEN        ;8 XFRS/CYCLE
001172' 026727  177304  000001                                          CMP     BLKMBT,#1
001200' 001004                                                          BNE     $50020
001202'
                                LET @CSR1 := #721       ;BLK MODE XFRS DATA OUT
001202' 012777  000721  177120                                          MOV     #721,@CSR1
                            ELSE                        ;BA IS THE DATA SRCE
001210' 000403                                                          BR      $50021
001212'                                                                 $50020:
001212'
                                LET @CSR1 := #661       ;NON BLK XFRS DATA OUT
001212' 012777  000661  177110                                          MOV     #661,@CSR1
001220'                     ENDIF                       ;1 XFR/CYCLE
001220'                                                                 $50021:

                                                        ;ADJUST THE WC IF NECSY
                            LET R0 := #0
001220' 005000                                                          CLR     R0
                            LET R0 := R0 - SVWDCT       ;COMPLIMENT THIS
001222' 166700  177304                                                  SUB     SVWDCT,R0
                            LET @WC := R0
001226' 010077  177104                                                  MOV     R0,@WC
                                                        ; THE BA IS THE SOURCE

                            INLINE <JSR PC,BIT22>
001232' 004767  001156                                                  JSR     PC,BIT22
                            INLINE <JSR PC,ROUT1>       ;22 BIT ADRS AND INTR LEVEL
001236' 004767  000730                                                  JSR     PC,ROUT1
                            IF BLKMBT EQ #1 THEN
001242' 026727  177234  000001                                          CMP     BLKMBT,#1
001250' 001003                                                          BNE     $50022
001252'
                                LET TMPSR2 := TMPSR2 SET.BY #BIT09
001252' 052767  001000  177246                                          BIS     #BIT09,TMPSR2
001260'                     ENDIF                       ;BLOCK MODE BIT IN TMPSR2
001260'                                                                 $50022:

                            CALL        GO              ;GO START THE XFR
001260' 004767  001526                                                  JSR     PC,GO
```



```
                            CALL        CHKDA           ;GO CHECK DATA
001264' 004767  001404                                                  JSR     PC,CHKDA


                        ;*****************************************************************
                        ;                       DATA IN TRANSFERS
                        ;*****************************************************************


                            IF BLKMBT EQ #1 THEN        ;16 XFRS/CYCLE
001270' 026727  177206  000001                                          CMP     BLKMBT,#1
001276' 001004                                                          BNE     $50023
001300'
                                LET @CSR1 := #547       ;BLK MODE XFRS
001300' 012777  000547  177022                                          MOV     #547,@CSR1
                            ELSE
001306' 000403                                                          BR      $50024
001310'                                                                 $50023:
001310'
                                LET @CSR1 := #407       ;NON BLK , HOG MODE XFRS
001310' 012777  000407  177012                                          MOV     #407,@CSR1
001316'                     ENDIF
001316'                                                                 $50024:


001316'                     IF #BIT01 SET.IN SR1 THEN   ;A 16K XFR REQUESTED
001316' 032767  000002  176472                                          BIT     #BIT01,SR1
001324' 001416                                                          BEQ     $50025
001326'
                                LET R0 := WBUFSZ        ;YES
001326' 016700  176610                                                  MOV     WBUFSZ,R0
001332'                         IF R0 EQ #100000 THEN
001332' 020027  100000                                                  CMP     R0,#100000
001336' 001003                                                          BNE     $50026
001340'
                                    LET R0 := #77777
001340' 012700  077777                                                  MOV     #77777,R0
001344'                         ELSE
001344' 000401                                                          BR      $50027
001346'                                                                 $50026:
001346'
                                    INLINE <NEG R0>
001346' 005400                                                          NEG     R0
001350'                         ENDIF
001350'                                                                 $50027:
001350'                         LET @WC := R0           ;LD WORD COUNTER
001350' 010077  176762                                                  MOV     R0,@WC
001354'                         LET R0 := @LATCNT       ;CLR LAT CNTR TO STOP ERR BIT
001354' 017700  176762                                                  MOV     @LATCNT,R0
001360'                     ELSE
001360' 000403                                                          BR      $50030
001362'                                                                 $50025:
001362'
                                LET @WC := #177000      ;1 K WORDS
001362' 012777  177000  176746                                          MOV     #177000,@WC
001370'                     ENDIF
001370'                                                                 $50030:


001370'                     INLINE <JSR PC,BIT23>       ;GET 22 BIT ADRS
```

```
001370' 004767  001150                                                                  JSR     PC,BIT23
001374'                       INLINE <JSR PC,ROUT1>        ;GET INTR LEVEL
001374' 004767  000572                                                                  JSR     PC,ROUT1

001400'                       IF BLKMBT EQ #1 THEN
001400' 026727  177076  000001                                                          CMP     BLKMBT,#1
001406' 001003                                                                          BNE     $50031
001410'                          LET TMPSR2 := TMPSR2 SET.BY #BIT09
001410' 052767  001000  177110                                                          BIS     #BIT09,TMPSR2
001416'                       ENDIF
001416'                                                                                 $50031:
001416'                       CALL        GO                          ;GO START THE XFR
001416' 004767  001370                                                                  JSR     PC,GO

001422'                       IF #BIT00 SETIN SR1 THEN
001422' 032767  000001  176366                                                          BIT     #BIT00,SR1
001430' 001415                                                                          BEQ     $50032
001432'                          IF BLKMBT NE #1 THEN          ;HAS OPERATOR SELECTED TO
001432' 026727  177044  000001                                                          CMP     BLKMBT,#1
001440' 001404                                                                          BEQ     $50033
                                                               ;TRY BLOCK MODE XFRS
001442'                             LET BLKMBT := #1           ;YES, SET MY BLOCK MD BIT
001442' 012767  000001  177032                                                          MOV     #1,BLKMBT
001450'                          ELSE
001450' 000404                                                                          BR      $50034
001452'                                                                                 $50033:
001452'                             LET BLKMBT := #0           ;MAKE SURE BLK MODE BIT
001452' 005067  177024                                                                  CLR     BLKMBT
001456'                             LET SVINLV := SVINLV - #1  ;IS CLR. SUBT 1 FROM INTR LVL
001456' 005367  176570                                                                  DEC     SVINLV
001462'                          ENDIF
001462'                                                                                 $50034:
001462'                       ELSE
001462' 000404                                                                          BR      $50035
001464'                                                                                 $50032:
001464'                          LET BLKMBT := #0              ;MAKE SURE BLK MODE BIT
001464' 005067  177012                                                                  CLR     BLKMBT
001470'                          LET SVINLV := SVINLV - #1     ;IS CLR. SUBT 1 FROM INTR LVL
001470' 005367  176556                                                                  DEC     SVINLV
001474'                       ENDIF
001474'                                                                                 $50035:
001474'                    ENDDO
001474' 000632                                                                          BR      $50016
001476'                                                                                 $50017:

001476'                    LET R4 := R4 + #2                   ;NXT LOC IN Q22TB
001476' 062704  000002                                                                  ADD     #2,R4
001502'                    LET DEVADR := (R4)
001502' 011467  176616                                                                  MOV     (R4),DEVADR
001506'                    LET R3 := R3 + #2                   ;NXT VECTOR IN TABLE
001506' 062703  000002                                                                  ADD     #2,R3
001512'                    INLINE <JMP     ESR13>              ;GO BACK FOR NXT ADRS
001512' 000167  177414                                                                  JMP     ESR13

                           ;*******************************************************
                           ;
```



```
                           ;      THIS SECTION OF THE PROGRAM WILL SET UP ALL
                           ;      DEVICES TO ATTEMPT A SIMULTANEOUS TRANSFER
                           ;      VIA THE SIMGOA REGISTERS.
                           ;
                           ;*******************************************************

001516'                    SIMGO:
001516'                    LET SVINLV := #7                    ;FIRST INTR LEVEL
001516' 012767  000007  176526                                                          MOV     #7,SVINLV
001524'                    INLINE <MTPS    #340>               ;RAISE CPU LVL TO 7
001524' 106427  000340                                                                  MTPS    #340
001530'                    LET SIMGBT := #1
001530' 012767  000001  176762                                                          MOV     #1,SIMGBT

001536'                    INLINE <ESR33:>                     ;RESTART THIS SECTION HERE
001536'                                                                                 ESR33:

001536'                    LET R3 := #VECT                     ;TABLE POINTERS. R5 IS USED TO ALTER
001536' 012703  000302'                                                                 MOV     #VECT,R3
001542'                    LET R4 := #Q22TB                    ;XFRS SO THAT 1 DEVICE WILL DO DATA INS
001542' 012704  000260'                                                                 MOV     #Q22TB,R4
001546'                    LET R5 := #1                        ;AND THE NXT DEVICE WILL DO DATA OUTS
001546' 012705  000001                                                                  MOV     #1,R5

001552'                    WHILE (R3) NE #0 DO                 ;IF R3 IS 0 THEN VECTOR TABLE
001552'                                                                                 $50036:
001552' 005713                                                                          TST     (R3)
001554' 001443                                                                          BEQ     $50037

001556'                       LET DEVECT := (R3)        ;FIRST VECTOR
001556' 011367  176544                                                                  MOV     (R3),DEVECT
001562'                       LET DEVADR := (R4)        ;FIRST ADRS
001562' 011467  176536                                                                  MOV     (R4),DEVADR

001566'                       CALL        ADRSET        ;GET REG ADRS, SET MASK
001566' 004767  000512                                                                  JSR     PC,ADRSET
                                                        ;ADJUST THE WC IF NECSY
001572'                       LET R0 := #0
001572' 005000                                                                          CLR     R0
001574'                       LET R0 := R0 - SVWDCT            ;COMPLIMENT THIS
001574' 166700  176732                                                                  SUB     SVWDCT,R0
001600'                       LET @WC := R0
001600' 010077  176532                                                                  MOV     R0,@WC
001604'                       INLINE <JSR PC,ROUT1>     ;INTR LEVELS
001604' 004767  000362                                                                  JSR     PC,ROUT1
001610'                       IF R5 EQ #1 THEN
001610' 020527  000001                                                                  CMP     R5,#1
001614' 001007                                                                          BNE     $50040
001616'                          LET @CSR1 := #767      ;DATA SRCE IS THE BA
001616' 012777  000767  176504                                                          MOV     #767,@CSR1
001624'                          INLINE <JSR     PC,BIT22>
001624' 004767  000564                                                                  JSR     PC,BIT22
001630'                          LET R5 := #0           ;DATA OUT, 4 XFRS/CYCLE
001630' 005005                                                                          CLR     R5
001632'                       ELSE
001632' 000407                                                                          BR      $50041
```

```
                                                                                $50040:
                        LET @CSR1 := #507               ;2 XFRS/CYCLE
001634' 012777  000507  176466                                                  MOV     #507,@CSR1
                        INLINE <JSR     PC,BIT23>
001642' 004767  000676                                                          JSR     PC,BIT23
                        LET R5 := #1            ;DATA IN
001646' 012705  000001                                                          MOV     #1,R5
                    ENDIF
                                                                                $50041:
                    LET R3 := R3 + #2
001652' 062703  000002                                                          ADD     #2,R3
                    LET R4 := R4 + #2
001656' 062704  000002                                                          ADD     #2,R4
                ENDDO
001662' 000733                                                                  BR      $50036
                                                                                $50037:
                ;*************************************************************
                ;ALL DEVICES ARE NOW SET TO GO
                ;*************************************************************

                LET INTCNT := #0                        ;SET INTR CNTR TO 0
001664' 005067  176634                                                          CLR     INTCNT

                LET SVINT := #0                         ;CLR SAVE INTERRUPTS
001670' 005067  176626                                                          CLR     SVINT
                INLINE <JSR     PC,GO>
001674' 004767  001112                                                          JSR     PC,GO

                LET SVINLV := SVINLV - #1               ;SUBT 1 FROM INTR LEVEL
001700' 005367  176346                                                          DEC     SVINLV
                IF SVINLV NE #3 THEN
001704' 026727  176342  000003                                                  CMP     SVINLV,#3
001712' 001402                                                                  BEQ     $50042
                    INLINE <JMP ESR33>          ;RE ITERATE
001714' 000167  177616                                                          JMP     ESR33
                ENDIF                           ;THIS SECTION
001720'                                                                         $50042:

                ;*** ******************************************
                ;WHERE TO GO FROM HERE
                ;********************** ***************************
001720'         PASS:

                .NLIST MC
                .LIST ME
001720' 104413  000000'         ENDITS,BEGIN            ;SIGNAL END OF ITERATION.
                                                        ;MONITOR SHALL TEST END OF PASS
                .LIST MC
                .NLIST ME


001724'         INLINE <JMP     RESTRT>
001724' 000167  177124                                                          JMP     RESTRT
```



```
                .SBTTL  SUB-ROUTINES CALLED OUT BY THE MAIN PROGRAM


                        ;THIS WILL BE THE END SECTION OF THE DEC/X11
                        ;SOFTWARE MODULE. ALL NORMAL ROUTINES
                        ;FOUND IN AN END SECTION OF A SOFTWARE MODULE
                        ;WILL BE INCLUDED HERE.
                ;***** ***************************************************
                ;       THIS ROUTINE WILL CHECK ALL DEVICES THAT ARE THERE
                ;       AND WILL CLEAR THEIR APPROPRIATE REGISTERS.
                ;       IN ADDITION THIS ROUTINE WILL SET UP THE CORRECT
                ;       PSW AND THE APPROPRIATE ADRS FOR THE INTERRUPT
                ;       SERVICE ROUTINE.
                ;************************************ ************************


                                                        ;R3-R4 HAVE BEEN SET UP ALREADY
                ROUTINE CLRREG
001730'                                                                         CLRREG:

                LET R0 := DEVADR
001730' 016700  176370                                                          MOV     DEVADR,R0
                LET CSR1 := R0                  ;START WITH THIS ADRS
001734' 010067  176370                                                          MOV     R0,CSR1
                LET CSR2 := R0 + #2
001740' 010067  176366                                                          MOV     R0,CSR2
001744' 062767  000002  176360                                                  ADD     #2,CSR2
                LET BA := R0 + #4
001752' 010067  176356                                                          MOV     R0,BA
001756' 062767  000004  176350                                                  ADD     #4,BA
                LET WC := R0 + #6
001764' 010067  176346                                                          MOV     R0,WC
001770' 062767  000006  176340                                                  ADD     #6,WC
                LET DATA := R0 + #10
001776' 010067  176336                                                          MOV     R0,DATA
002002' 062767  000010  176330                                                  ADD     #10,DATA
                LET LATCNT := R0 + #12
002010' 010067  176326                                                          MOV     R0,LATCNT
002014' 062767  000012  176320                                                  ADD     #12,LATCNT
                LET MVLCNT := R0 + #14
002022' 010067  176316                                                          MOV     R0,MVLCNT
002026' 062767  000014  176310                                                  ADD     #14,MVLCNT
                LET SIMGOA := R0 + #16
002034' 010067  176306                                                          MOV     R0,SIMGOA
002040' 062767  000016  176300                                                  ADD     #16,SIMGOA

                LET R0 := @LATCNT               ;CLR LAT COUNTER BY
002046' 017700  176270                                                          MOV     @LATCNT,R0
                LET @CSR1 := #0                 ;READING IT
002052' 005077  176252                                                          CLR     @CSR1
                LET @BA := #0
002056' 005077  176252                                                          CLR     @BA
                LET @WC := #0
002062' 005077  176250                                                          CLR     @WC
                LET @DATA := #0
002066' 005077  176246                                                          CLR     @DATA
```

```
                                        LET @CSR2 := #0
002072' 005077  176234                                                  CLR     @CSR2
                                        LET (R4)+ := DEVADR
002076' 016724  176222                                                  MOV     DEVADR,(R4)+

                                        LET (R3)+ := DEVECT
002102' 016723  176220                                                  MOV     DEVECT,(R3)+

                                        LET R0 := FIRVEC
002106' 016700  176400                                                  MOV     FIRVEC,R0
                                        LET R1 := #VECT0                ;VECTOR OF FIRST POSSIBLE ONE INTO R1
002112' 012701  000362'                                                 MOV     #VECT0,R1
                                        WHILE R0 NE DEVECT DO           ;IF NOT THIS ONE
002116'                                                                 $50045:
002116' 020067  176204                                                  CMP     R0,DEVECT
002122' 001405                                                          BEQ     $50046
002124'                                     LET R0 := R0 + #4           ;THEN ADD 4 FOR THE
002124' 062700  000004                                                  ADD     #4,R0
002130'                                     LET R1 := R1 + #12          ;NXT ONE
002130' 062701  000012                                                  ADD     #12,R1
002134'                                 ENDDO
002134' 000770                                                          BR      $50045
002136'                                                                 $50046:
002136'                                 LET (R0) := R1                  ;PROPER ADRS
002136' 010110                                                          MOV     R1,(R0)

002140'                                 LET DEVECT := DEVECT + #2       ;2 MORE FOR THE PSW
002140' 062767  000002  176160                                          ADD     #2,DEVECT
002146'                                 LET @DEVECT := #340
002146' 012777  000340  176152                                          MOV     #340,@DEVECT
002154'                                 LET DEVADR := DEVADR + #20
002154' 062767  000020  176142                                          ADD     #20,DEVADR
002162'                                 LET DEVECT := DEVECT + #2       ;NXT ADRS/VECTOR PAIR
002162' 062767  000002  176136                                          ADD     #2,DEVECT

002170'                                 ENDRTN
002170'                                                                 $50043:
002170'                                                                 $50044:
002170' 000207                                                          RTS     PC


                                    ;***************************************************************
                                    ;
                                    ;       ROUTINE TO MOVE INTERRUPT LEVEL,SVINLV, INTO CSR2
                                    ;
                                    ;***** ***************************************************************

                                    ROUT1:
                                            SELECT SVINLV OF 7 BASE 3
                                                                        MOV     SVINLV,-(SP)
                                                                        SUB     #3,(SP)
002202' 002435                                                          BLT     $50055
002204' 026727  176042  000007                                          CMP     SVINLV,#7
```



```
002212' 003031                                                          BGT     $50055
002214' 006316                                                          ASL     (SP)
002216' 062716  002224'                                                 ADD     #$50047,(SP)
002222' 013607                                                          MOV     @(SP)+,PC
002224'                                                                 $50047:
002224' 002266'                                                         .WORD   $50054
002226' 002266'                                                         .WORD   $50053
002230' 002256'                                                         .WORD   $50052
002232' 002246'                                                         .WORD   $50051
002234' 002236'                                                         .WORD   $50050

002236'                                     CASE 7
002236'                                                                 $50050:
002236'                                         LET TMPSR2 := #32
002236' 012767  000032  176262                                          MOV     #32,TMPSR2
002244'                                     CASE 6
002244' 000416                                                          BR      $50056
002246'                                                                 $50051:
002246'                                         LET TMPSR2 := #12
002246' 012767  000012  176252                                          MOV     #12,TMPSR2
002254'                                     CASE 5
002254' 000412                                                          BR      $50056
002256'                                                                 $50052:
002256'                                         LET TMPSR2 := #6
002256' 012767  000006  176242                                          MOV     #6,TMPSR2
002264'                                     DEFAULT
002264' 000406                                                          BR      $50056
002266'                                                                 $50054:
002266'                                                                 $50053:
002266'                                         LET TMPSR2 := #2
002266' 012767  000002  176232                                          MOV     #2,TMPSR2
002274'                                     ENDSELECT
002274' 000402                                                          BR      $50056
002276'                                                                 $50055:
002276' 062706  000002                                                  ADD     #2,SP
002302'                                                                 $50056:
002302'                                     INLINE <RTS     PC>          ;GO BACK
002302' 000207                                                          RTS     PC


                                    ;***************************************************
                                    ;
                                    ;       THIS ROUTINE WILL SET UP CORRECT REGISTER ADDRESSES
                                    ;       FOR AN KNOWN DEVICE. IT WILL ALSO SET THE CORRECT
                                    ;       BITS IN THE INTERRUPT MASK
                                    ;
                                    ;***************************************************


                                    ROUTINE ADRSET
002304'                                                                 ADRSET:

                                    LET MASK := #0                  ;INITIALLY CLR MASK
002304' 005067  176176                                                  CLR     MASK

002310'                             IF SIMGBT EQ #0 THEN            ;ONE DEV AT A TIME
```

```
                                                                                    TST     SIMGBT
                                                                                    BNE     $50061
                        LET R1 := FIRADR            ;FIRST POSSIBLE ADRES
                                                                                    MOV     FIRADR,R1
                        LET R0 := #BIT00            ;FIRST BIT
                                                                                    MOV     #BIT00,R0
                        WHILE DEVADR NE R1 DO
                                                                                    $50062:
                                                                                    CMP     DEVADR,R1
                                                                                    BEQ     $50063
                            INLINE <ASL     R0>     ;SHIFT 1 TO THE LEFT
                                                                                    ASL     R0
                            LET R1 := R1 + #20      ;NXT ADRS
                                                                                    ADD     #20,R1
                        ENDDO
                                                                                    BR      $50062
                                                                                    $50063:
                        LET MASK := MASK SET.BY R0
                                                                                    BIS     R0,MASK
                    ELSE
                                                                                    BR      $50064
                                                                                    $50061:
                        LET MASK := DV              ;ALL DEVICES AT ONCE
                                                                                    MOV     DV,MASK
                    ENDIF
                                                                                    $50064:

                    ;*****************************************************
                    ;THE ABOVE CODE SETS UP THE INTR MASK
                    ;*****************************************************

                    LET R0 := DEVADR                ;STARTING REG ADRS
                                                                                    MOV     DEVADR,R0
                    LET R1 := #10
                                                                                    MOV     #10,R1
                    LET R2 := #CSR1
                                                                                    MOV     #CSR1,R2
                    WHILE R1 NE #0 DO
                                                                                    $50065:
                                                                                    TST     R1
                                                                                    BEQ     $50066
                        LET (R2)+ := R0
                                                                                    MOV     R0,(R2)+
                        LET R1 := R1 - #1
                                                                                    DEC     R1
                        LET R0 := R0 + #2           ;NXT ADRS
                                                                                    ADD     #2,R0
                    ENDDO
                                                                                    BR      $50065
                                                                                    $50066:


                    ENDRTN
                                                                                    $50057:
                                                                                    $50060:
                                                                                    RTS     PC
```



```
                    ;*****************************************************
                    ;
                    ;       ROUTINE TO SET UP A 22 BIT ADDRESS FOR USE IN THE
                    ;       MAIN PROGRAM FOR THE READ BUFFER INSTS.
                    ;*****************************************************


                    BIT22:
                        IF #ADDR22 SETIN CONFIG THEN
                                                                                    BIT     #ADDR22,CONFIG
                                                                                    BEQ     $50067
                            LET ADR := RBUFPA       ;LOWER 16 BITS INTO ADR
                                                                                    MOV     RBUFPA,ADR
                            LET XMEM := RBUFEA      ;BITS 17 AND 18
                                                                                    MOV     RBUFEA,XMEM
                            .NLIST MC
                            .LIST ME
                        MAP22S, BEGIN,ADR           ; GET 22-BIT ADDR FROM 18-BIT ADDR
                            .LIST MC
                            .NLIST ME
                            LET @BA := PA22         ;LOAD 16 BITS INTO THE BA
                                                                                    MOV     PA22,@BA
                                                    ;MOVE EA BITS FOR CSR1
                            LET EA22 := EA22 SHIFT 2
                                                                                    ASL     EA22
                                                                                    ASL     EA22
                            INLINE <SWAB        EA22>   ;EA BIT 0 NOW IN BIT 10
                                                                                    SWAB    EA22
                            LET @CSR1 := @CSR1 SET.BY EA22
                                                                                    BIS     EA22,@CSR1
                                                    ;SET EXTENDED ADRS BITS IN CSR1
                        ELSE
                                                                                    BR      $50070
                                                                                    $50067:
                            LET @BA := RBUFPA       ;ONLY A 16 BIT ADRS
                                                                                    MOV     RBUFPA,@BA
                            IF #BIT04 SETIN RBUFEA THEN
                                                                                    BIT     #BIT04,RBUFEA
                                                                                    BEQ     $50071
                                LET @CSR1 := @CSR1 SET.BY #BIT10
                                                                                    BIS     #BIT10,@CSR1
                            ENDIF
                                                                                    $50071:
                            IF #BIT05 SETIN RBUFEA THEN
                                                                                    BIT     #BIT05,RBUFEA
                                                                                    BEQ     $50072
                                LET @CSR1 := @CSR1 SET.BY #BIT11
                                                                                    BIS     #BIT11,@CSR1
                            ENDIF
                                                                                    $50072:
                                                    ;EA BITS INTO CSR1
                        ENDIF
                                                                                    $50070:
```

```
1163  002542'                          INLINE <RTS     PC>             ;GO BACK WHERE YOU CAME FROM
1164  002542' 000207                                                    RTS     PC
1165
1166                                   ;*************************************************
1167                                   ;
1168                                   ;       THIS ROUTINE WILL SET A 22 BIT ADRS FOR THE
1169                                   ;       GET WRITE BUFFER INSTS.
1170                                   ;
1171                                   ;***  *************************************************
1172
1173  002544'                          BIT23:
1174
1175  002544'                          IF #ADDR22 SETIN CONFIG THEN
1176  002544' 032767  001000  175304                                    BIT     #ADDR22,CONFIG
1177  002552' 001426                                                    BEQ     $50073
1178  002554'                              LET ADR := #BUFPA            ;LOWER 16 OF 18
1179  002554' 016767  175354  175566                                    MOV     WBUFPA,ADR
1180  002562'                              LET XMEM := #BUFEA
1181  002562' 016767  175350  175562                                    MOV     WBUFEA,XMEM
1182
1183                                       .NLIST MC
1184                                       .LIST ME
1185  002570' 104416  000000' 000350'          MAP22$, BEGIN,ADR        ; GET 22-BIT ADDR FROM 18-BIT ADDR
1186                                       .LIST MC
1187                                       .NLIST ME
1188
1189  002576'                              LET #BA := PA22
1190  002576' 016777  175552  175530                                    MOV     PA22,@BA
1191  002604'                              LET EA22 := EA22 SHIFT 2
1192  002604' 006367  175546                                           ASL     EA22
1193  002610' 006367  175542                                           ASL     EA22
1194  002614'                              INLINE <SWAB        EA22>    ;SWAP BYTES FOR CSR1
1195  002614' 000367  175536                                           SWAB    EA22
1196
1197  002620'                              LET #CSR1 := #CSR1 SET.BY EA22
1198  002620' 056777  175532  175502                                    BIS     EA22,@CSR1
1199  002626'                          ELSE
1200  002626' 000421                                                    BR      $50074
1201  002630'                                                           $50073:
1202  002630'                              LET #BA := #BUFPA            ;16 BIT ADRS
1203  002630' 016777  175300  175476                                    MOV     WBUFPA,@BA
1204  002636'                              IF #BIT04 SETIN #BUFEA THEN
1205  002636' 032767  000020  175272                                    BIT     #BIT04,#BUFEA
1206  002644' 001403                                                    BEQ     $50075
1207  002646'                                  LET #CSR1 := #CSR1 SET.BY BIT10
1208  002646' 056777  002000  175454                                    BIS     BIT10,@CSR1
1209  002654'                              ENDIF
1210  002654'                                                           $50075:
1211  002654'                              IF #BIT05 SETIN #BUFEA THEN
1212  002654' 032767  000040  175254                                    BIT     #BIT05,#BUFEA
1213  002662' 001403                                                    BEQ     $50076
1214  002664'                                  LET #CSR1 := #CSR1 SET.BY #BIT11
1215  002664' 052777  004000  175436                                    BIS     #BIT11,@CSR1
1216  002672'                              ENDIF
1217  002672'                                                           $50076:
1218                                   ;EA BITS INTO CSR1
```



```
1219  002672'                          ENDIF
1220  002672'                                                           $50074:
1221
1222  002672'                          INLINE <RTS     PC>
1223  002672' 000207                                                    RTS     PC
1224
1225
1226                                   ;***** ** *** ***************************************************
1227                                   ;
1228                                   ;       DATA CHECK ROUTINE
1229                                   ;
1230                                   ;*********************************************************************
1231
1232
1233  002674'                          ROUTINE CHKDA
1234  002674'                                                           CHKDA:
1235
1236  002674'                          LET R0 := #BUFVA                 ;STARTING ADRS OF RD BUF
1237  002674' 016700  175224                                           MOV     #BUFVA,R0
1238                                                                   ;THIS ADDRESS WAS ADJUSTED SO
1239                                                                   ;THAT STARTING ADDRESS WOULD
1240                                                                   ;START AT EITHER 0 OR 40. THIS
1241                                                                   ;IS NECESSARY FOR BLOCK MODE
1242                                                                   ;TRANSFERS TO COMMENCE AT THE
1243                                                                   ;CORRECT BOUNDARY.
1244  002700'                          LET R1 := SWADCT
1245  002700' 016701  175626                                           MOV     SWADCT,R1
1246  002704'                          IF #ADDR22 SETIN CONFIG THEN     ;22 BIT MONITOR
1247  002704' 032767  001000  175144                                   BIT     #ADDR22,CONFIG
1248  002712' 001403                                                   BEQ     $50101
1249  002714'                              LET R2 := PA22               ;1 DATA WORD IS 2 ADRS
1250  002714' 016702  175434                                           MOV     PA22,R2
1251  002720'                          ELSE
1252  002720' 000402                                                   BR      $50102
1253  002722'                                                          $50101:
1254  002722'                              LET R2 := #BUFPA             ;18 BIT MONITOR
1255  002722' 016702  175200                                           MOV     #BUFPA,R2
1256  002726'                          ENDIF
1257  002726'                                                          $50102:
1258  002726'                          LET R2 := R2 + #2
1259  002726' 062702  000002                                           ADD     #2,R2
1260
1261  002732'                          #WHILE R1 NE #0 DO
1262  002732'                                                          $50103:
1263  002732' 005701                                                   TST     R1
1264  002734' 001425                                                   BEQ     $50104
1265  002736'                              IF (R0) NE R2 THEN           ;CONTENTS OF R0 = R2 ?
1266  002736' 021002                                                   CMP     (R0),R2
1267  002740' 001415                                                   BEQ     $50105
1268  002742'                                  LET ACSR := @CSR1
1269  002742' 017767  175362  175132                                   MOV     @CSR1,ACSR
1270  002750'                                  LET SBADR := R2
1271  002750' 010267  175126                                          MOV     R2,SBADR
1272  002754'                                  LET WASADR := #0
1273  002754' 005067  175124                                          CLR     WASADR
1274  002760'                                  LET ASB := (R2)
```

```
002760' 011267  175122                                                          MOV     (R2),ASB
002764'                          LET AWAS := (R0)
002764' 011067  175120                                                          MOV     (R0),AWAS

                                 .NLIST MC
                                 .LIST ME
                                 ;***************************************************************
002770' 104404  000000'          DATERS,BEGIN            ;DATA ERROR!!!
                                 ;***************************************************************
                                 .LIST MC
                                 .NLIST ME

002774'                        ENDIF
002774'                                                                         $50105:
002774'                        LET R0 := R0 + #2              ;INC TO NXT ADRS OF DATA
002774' 062700  000002                                                          ADD     #2,R0
003000'                        LET R1 := R1 - #1              ;DEC COUNTER
003000' 005301                                                                  DEC     R1
003002'                        LET R2 := R2 + #2              ;INC TO NXT DATA WORD
003002' 062702  000002                                                          ADD     #2,R2
003006'                    ENDDO
003006' 000751                                                                  BR      $50103
003010'                                                                         $50104:

003010'                    ENDRIN
003010'                                                                         $50077:
003010'                                                                         $50100:
003010' 000207                                                                  RTS     PC
                           ;**************************************************
                           ;       THIS IS THE GO ROUTINE. IT WILL SET THE
                           ;       GO BIT, START THE LOOP COUNTER AND RETURN
                           ;       CONTROL TO THE MONITOR VIA THE BREAK
                           ;       INSTRUCTION.
                           ;**************************************************

003012'                    ROUTINE GO
003012'                                                                         GO:

003012'                    LET LPCNTR := #2000                ;LOOP CNTR INIT VALUE
003012' 012767  002000  175474                                                  MOV     #2000,LPCNTR
003020'                    IF SIMGBT EQ #1 THEN               ;CHK SIMULT GO BIT
003020' 026727  175474  000001                                                  CMP     SIMGBT,#1
003026' 001114                                                                  BNE     $50110

003030'                        LET TMPR0 := R0                ;TEMP SAVE R0
003030' 010067  175474                                                          MOV     R0,TMPR0
003034'                        LET R0 := #Q22TB               ;ADRS OF TABLE
003034' 012700  000260'                                                         MOV     #Q22TB,R0
003040'                        LET RY := (R0)                 ;CONTENTS OF ADRS
003040' 011067  175470                                                          MOV     (R0),RY
003044'                        LET SIMGOA := RY + #16         ;LOWEST VALID VALUE IN SIMGOA
003044' 016767  175464  175274                                                  MOV     RY,SIMGOA
003052' 062767  000016  175266                                                  ADD     #16,SIMGOA
003060'                        LET CSR2 := SIMGOA - #14       ;LOWEST VALID CSR2
003060' 016767  175262  175244                                                  MOV     SIMGOA,CSR2
003066' 162767  000014  175236                                                  SUB     #14,CSR2
```



```
003074'                        REPEAT
003074'                                                                         $50111:
003074'                            LET @CSR2 := TMPSR2        ;INTR BITS  INTO CSR2
003074' 016777  175426  175230                                                  MOV     TMPSR2,@CSR2
003102'                            LET CSR2 := CSR2 + #16     ;THIS WILL LD ALL CSR2'S
003102' 062767  000016  175222                                                  ADD     #16,CSR2
003110'                            LET R0 := R0 + #2
003110' 062700  000002                                                          ADD     #2,R0
003114'                            WHILE CSR2 NE (R0) AND (R0) NE #0 DO
003114'                                                                         $50112:
003114' 026710  175212                                                          CMP     CSR2,(R0)
003120' 001406                                                                  BEQ     $50113
003122' 005710                                                                  TST     (R0)
003124' 001404                                                                  BEQ     $50113
003126'                                LET CSR2 := CSR2 + #20 ;TRY NXT ADDRESS
003126' 062767  000020  175176                                                  ADD     #20,CSR2
003134'                            ENDDO
003134' 000767                                                                  BR      $50112
003136'                                                                         $50113:
003136'                            LET CSR2 := CSR2 + #2      ;CORRECT CSR2 ADRS
003136' 062767  000002  175166                                                  ADD     #2,CSR2
003144'                        UNTIL (R0) EQ #0
003144' 005710                                                                  TST     (R0)
003146' 001352                                                                  BNE     $50111

003150'                        LET CSR2 := CSR2 - #20         ;RESTORE THIS TO REAL VALUE
003150' 162767  000020  175154                                                  SUB     #20,CSR2

003156'                        LET @SIMGOA := #1              ;START ALL DEVICES
003156' 012777  000001  175162                                                  MOV     #1,@SIMGOA
003164'                        INLINE <MTPS      #140>        ;DROP CPU INTR LVL
003164' 106427  000140                                                          MTPS    #140
003170'                        WHILE RY NE #0 DO              ;IF 0, ALL HAVE BEEN CHECKED
003170'                                                                         $50114:
003170' 005767  175340                                                          TST     RY
003174' 001426                                                                  BEQ     $50115
003176'                            LET RY := RY + #2          ;GET ADRS OF CSR2
003176' 062767  000002  175330                                                  ADD     #2,RY
003204'                            WHILE LPCNTR NE #0 AND #BIT07 NOTSETIN @RY DO
003204'                                                                         $50116:
003204' 005767  175304                                                          TST     LPCNTR
003210' 001413                                                                  BEQ     $50117
003212' 032777  000200  175314                                                  BIT     #BIT07,@RY
003220' 001007                                                                  BNE     $50117
003222'                                LET LPCNTR := LPCNTR - #1
003222' 005367  175266                                                          DEC     LPCNTR

                                       .NLIST MC
                                       .LIST ME
003226' 104407  000000'            BREAKS,BEGIN          ;TEMPORARY RETURN TO MONITOR....
003232' 104407  000000'            BREAKS,BEGIN          ;THEN CONTINUE AT NEXT INSTRUCTION.
                                       .LIST MC
                                       .NLIST ME
003236'                            ENDDO                      ;ALL DEVICES SHOULD BE DONE
003236' 000762                                                                  BR      $50116
```

```
003240'                                                                                   $50117:
003240'                                 LET R0 := R0 + #2                                 ;BY NOW
003240' 062700  000002                                                                    ADD     #2,R0
003244'                                 LET RY := (R0)                 ;NXT DEV IN TABLE
003244' 011067  175264                                                                    MOV     (R0),RY
003250'                               ENDDO
003250' 000747                                                                            BR      $50114
003252'                                                                                   $50115:
003252'                               LET R0 := TMPR0                  ;RESTORE CONTENTS OF R0
003252' 016700  175252                                                                    MOV     TMPR0,R0

003256'                           ELSE
003256' 000430                                                                            BR      $50120
003260'                                                                                   $50110:
003260'                               LET TMPSR2 := TMPSR2 SET.BY #BIT00  ;GET GO BIT READY
003260' 052767  000001  175240                                                            BIS     #BIT00,TMPSR2
003266'                               LET @CSR2 := TMPSR2              ;GO, THIS IS FOR 1 DEVICE AT A TIME
003266' 016777  175234  175036                                                            MOV     TMPSR2,@CSR2

003274'                               WHILE LPCNTR NE #0 DO
003274'                                                                                   $50121:
003274' 005767  175214                                                                    TST     LPCNTR
003300' 001417                                                                            BEQ     $50122
003302'                                   IF #BIT07 SETIN @CSR2 THEN
003302' 032777  000200  175022                                                            BIT     #BIT07,@CSR2
003310' 001403                                                                            BEQ     $50123
003312'                                       LET LPCNTR := #0         ;IF DONE IS SET, LPCNTR TO 0
003312' 005067  175176                                                                    CLR     LPCNTR
003316'                                   ELSE                         ;OTHERWISE COUNT DOWN
003316' 000403                                                                            BR      $50124
003320'                                                                                   $50123:
003320'                                       LET LPCNTR := LPCNTR - #2
003320' 162767  000002  175166                                                            SUB     #2,LPCNTR
003326'                                   ENDIF
003326'                                                                                   $50124:

                                          .NLIST MC
                                          .LIST ME
003326' 104407  000000'                   BREAK$,BEGIN          ;TEMPORARY RETURN TO MONITOR....
003332' 104407  000000'                   BREAK$,BEGIN          ;THEN CONTINUE AT NEXT INSTRUCTION.
                                          .LIST MC
                                          .NLIST ME

003336'                               ENDDO                            ;IF LPCNTR IS 0 GO ON
003336' 000756                                                                            BR      $50121
003340'                                                                                   $50122:
003340'                           ENDIF
003340'                                                                                   $50120:

003340'                           CALL     CHKINT                      ;CHK FOR CORRECT INTRS
003340' 004767  000006                                                                    JSR     PC,CHKINT

003344'                           CALL     CHKERR                      ;CHK COMPOSITE ERROR BITS
003344' 004767  000150                                                                    JSR     PC,CHKERR

003350'                           ENDRTN
```



```
003350'                                                                                   $50106:
003350'                                                                                   $50107:
003350' 000207                                                                            RTS     PC

                          ;*****************************************************
                          ;
                          ;       THIS ROUTINE WILL CHECK THAT THE PROPER
                          ;       INTERRUPTS HAVE OCCURRED.
                          ;
                          ;*****************************************************

003352'                   ROUTINE CHKINT
003352'                                                                                   CHKINT:
                                                      ;BIT/S IN MASK ARE DEVICES THAT
                                                      ;ARE EXPECTED TO INTERRUPT. SVINT IS
                                                      ;THE DEVICES THAT ACTUALLY DID AN
                                                      ;INTERRUPT. THEREFORE IF A BIT
                                                      ;IS SET IN MASK AND NOT IN SVINT, THAT
                                                      ;DEVICE DID NOT INTERRUPT.

003352'                   IF MASK NOTSETIN SVINT THEN
003352' 036767  175130  175142                                                            BIT     MASK,SVINT
003360' 001002                                                                            BNE     $50127
003362'                       INLINE <JSR PC,DRPDEV>
003362' 004767  000046                                                                    JSR     PC,DRPDEV
003366'                   ENDIF                              ;DROP THIS DEVICE
003366'                                                                                   $50127:

003366'                   IF SIMGBT EQ #1 THEN               ;ALL DEVICES WERE SET OFF
003366' 026727  175126  000001                                                            CMP     SIMGBT,#1
003374' 001007                                                                            BNE     $50130
003376'                       IF INTRP NE INTCNT THEN        ;DO INTR COUNTS AGREE
003376' 026767  175102  175120                                                            CMP     INTRP,INTCNT
003404' 001403                                                                            BEQ     $50131
                                  .NLIST MC                  ;INTRP IS NUM OF DEVICES N SYS
                                  .LIST ME                   ;INTCNT IS NUM OF TIMES THRU
003406' 104403  000000' 003760'   MSGN$,BEGIN,TEXT3          ;ASCII MESSAGE CALL WITH COMMON HEADER
                                  .LIST MC
                                  .NLIST ME
003414'                       ENDIF
003414'                                                                                   $50131:
003414'                   ENDIF
003414'                                                                                   $50130:
003414'                   LET SVINT := #0          ;CLR SAVE INTR
003414' 005067  175102                                                                    CLR     SVINT

003420'                   ENDRTN
003420'                                                                                   $50125:
003420'                                                                                   $50126:
003420' 000207                                                                            RTS     PC

                          ;*****************************************************
                          ;
                          ;       THIS IS THE INTERRUPT SERVICE ROUTINE
                          ;
                          ;*****************************************************
```

```
1499
1500  003422'                          INTSRV:
1501
1502  003422'                          LET INTCNT := INTCNT + #1              ;ADD 1 TO INTR CNTR
1503  003422' 005267  175076                                                    INC     INTCNT
1504
1505  003426'                          INLINE <NOP>
1506  003426' 000240                                                            NOP
1507  003430'                          INLINE <NOP>
1508  003430' 000240                                                            NOP
1509  003432'                          INLINE <RTI>
1510  003432' 000002                                                            RTI
1511                                   ;*****    ********************************************
1512                                   ;       THIS ROUTINE WILL DROP A DEVICE AND WILL THEN
1513                                   ;       BRANCH TO START AND CONTINUE.
1514                                   ;
1515                                   ;*********************************************************
1516
1517  003434'                          DRPDEV:                          ;FROM CHKINT
1518
1519  003434'                          LET SVADR := FIRADR              ;FIRST POSSIBLE ADRS
1520  003434' 016767  175050  174716                                            MOV     FIRADR,SVADR
1521  003442'                          LET MORE := #BIT00               ;START WITH BIT 0
1522  003442' 012767  000001  174606                                            MOV     #BIT00,MORE
1523
1524  003450'                          WHILE MORE NOTSETIN MASK DO      ;MASK IS GOOD, SVINT WAS BAD
1525  003450'                                                                    $50132:
1526  003450' 036767  174602  175030                                            BIT     MORE,MASK
1527  003456' 001006                                                            BNE     $50133
1528  003460'                              INLINE <ASL MORE>
1529  003460' 006367  174572                                                    ASL     MORE
1530  003464'                              LET SVADR := SVADR + #20     ;NXT ADRS
1531  003464' 062767  000020  174666                                            ADD     #20,SVADR
1532  003472'                          ENDDO
1533  003472' 000766                                                            BR      $50132
1534  003474'                                                                   $50133:
1535
1536                                   .NLIST MC
1537                                   .LIST ME
1538  003474' 104403  000000' 0037b4'          MSGNS,BEGIN,TEXT2        ;ASCII MESSAGE CALL WITH COMMON HEADER
1539                                   .LIST MC
1540                                   .NLIST ME
1541
1542  003502'                          LET DV := DV CLR.BY MORE         ;NEW VALUE FOR DV
1543  003502' 046767  174550  174544                                            BIC     MORE,DV
1544  003510'                          LET INTRP := INTRP - #1          ;SUBT 1 FROM NUM OF INTRS
1545  003510' 005367  174770                                                    DEC     INTRP
1546  003514'                          INLINE <JMP REENTY>              ;PROGRAM RE-ENTRY POINT NR START
1547  003514' 000167  175120                                                    JMP REENTY
1548
1549                                   ;*************************************************
1550                                   ;
1551                                   ;       THIS ROUTINE WILL  CHECK FOR ERRORS IN THE
1552                                   ;       COMPOSITE ERROR BIT IN CSR2 OF THE Q22BE'S
1553                                   ;
1554                                   ;*****************************************************************
```



```
1555
1556  003520'                          ROUTINE CHKERR
1557  003520'                                                                   CHKERR:
1558
1559  003520'                          LET CSRA := #0
1560  003520' 005067  174354                                                    CLR     CSRA
1561  003524'                          LET ACSR := #0
1562  003524' 005067  174352                                                    CLR     ACSR
1563  003530'                          LET TMPR0 := SIMGOA              ;SAVE THIS
1564  003530' 016767  174612  174772                                            MOV     SIMGOA,TMPR0
1565  003536'                          LET ERRTYP := #44
1566  003536' 012767  000044  174342                                            MOV     #44,ERRTYP
1567  003544'                          LET SIMGOA := #-1                ;INSERT TABLE TERMINATOR
1568  003544' 012767  177777  174574                                            MOV     #-1,SIMGOA
1569
1570  003552'                          IF SIMGBT EQ #1 THEN             ;ALL DEVICES AT ONCE ?
1571  003552' 026727  174742  000001                                            CMP     SIMGBT,#1
1572  003560' 001023                                                            BNE     $50136
1573  003562'                              LET R1 := #Q22TB             ;GET FIRST ADRS IN TABLE
1574  003562' 012701  000260'                                                   MOV     #Q22TB,R1
1575  003566'                              WHILE (R1) NE #0 DO
1576  003566'                                                                   $50137:
1577  003566' 005711                                                            TST     (R1)
1578  003570' 001416                                                            BEQ     $50140
1579  003572'                                  LET R0 := (R1)               ;CONTENTS OF
1580  003572' 011100                                                            MOV     (R1),R0
1581  003574'                                  LET R0 := R0 + #2            ;GET ADRS OF CSR2
1582  003574' 062700  000002                                                    ADD     #2,R0
1583  003600'                                  IF #BIT14 SETIN @R0 THEN     ;NON-EXIST MEM ERROR BIT
1584  003600' 032710  040000                                                    BIT     #BIT14,@R0
1585  003604' 001405                                                            BEQ     $50141
1586  003606'                                      INLINE <JSR     PC,FILTAB>
1587  003606' 004767  000044                                                    JSR     PC,FILTAB
1588
1589                                           .NLIST MC
1590                                           .LIST ME
1591                                   ;*****************************************************************
1592  003612' 104405  000000' 000330'         HRDER$,BEGIN,TABLE1      ;DUMP REG CONTENTS
1593                                   ;*****************************************************************
1594                                           .LIST MC
1595                                           .NLIST ME
1596
1597  003620'                                  ENDIF
1598  003620'                                                                   $50141:
1599  003620'                                  LET R1 := R1 + #2            ;GET NXT DEVICE ADRS
1600  003620' 062701  000002                                                    ADD     #2,R1
1601  003624'                              ENDDO
1602  003624' 000760                                                            BR      $50137
1603  003626'                                                                   $50140:
1604  003626'                          ELSE
1605  003626' 000407                                                            BR      $50142
1606  003630'                                                                   $50136:
1607  003630'                              IF #BIT14 SETIN @CSR2 THEN       ;NON-EXIST MEM ERROR BIT
1608  003630' 032777  040000  174474                                            BIT     #BIT14,@CSR2
1609  003636' 001403                                                            BEQ     $50143
1610
```

```
                                        .NLIST MC
                                        .LIST ME
                                        ;*****************************************************************
003640' 104405  000000' 000330'         HNDERS,HEGIN,TABLE1     ;DUMP REGISTER CONTENTS
                                        ;*****************************************************************
                                        .LIST MC
                                        .NLIST ME

003646'                                 ENDIF
003646'                                                                         $50143:
003646'                                 ENDIF
003646'                                                                         $50142:
003646'                                 LET SIMGOA := TMPR0             ;RESTORE ADRS FOR SIMGO REG
003646' 016767  174656  174472                                                  MOV     TMPR0,SIMGOA

003654'                                 ENDRTN
003654'                                                                         $50134:
003654'                                                                         $50135:
003654' 000207                                                                  RTS     PC

                                        ;*****************************************************8
                                        ;       THIS ROUTINE WILL SETUP THE ERROR TABLE TO
                                        ;       PRINT OUT THE CONTENTS OF THE DEVICE REGISTERS
                                        ;*****************************************************

003656'                                 FILTAB:

003656'                                 LET CSR2 := R0
003656' 010067  174450                                                          MOV     R0,CSR2
003662'                                 LET CSR1 := R0 - #2             ;SUBT TWO FOR CSR1
003662' 010067  174442                                                          MOV     R0,CSR1
003666' 162767  000002  174434                                                  SUB     #2,CSR1
003674'                                 LET BA := R0 + #4
003674' 010067  174434                                                          MOV     R0,BA
003700' 062767  000004  174426                                                  ADD     #4,BA
003706'                                 LET WC := R0 + #6
003706' 010067  174424                                                          MOV     R0,WC
003712' 062767  000006  174416                                                  ADD     #6,WC
003720'                                 LET DATA := R0 + #10
003720' 010067  174414                                                          MOV     R0,DATA
003724' 062767  000010  174406                                                  ADD     #10,DATA
003732'                                 LET LATCNT := R0 + #12
003732' 010067  174404                                                          MOV     R0,LATCNT
003736' 062767  000012  174376                                                  ADD     #12,LATCNT
003744'                                 LET MVLCNT := R0 + #14
003744' 010067  174374                                                          MOV     R0,MVLCNT
003750' 062767  000014  174366                                                  ADD     #14,MVLCNT

003756'                                 INLINE <RTS     PC>
003756' 000207                                                                  RTS     PC

                                        ;*************************************************
                                        ;PUT SOME MESSAGES HERE
                                        ;
                                        ;************ ***************************************
003760' 004176'                         TEXT3:  MES3
```



```
003762' 177777                          177777
003764' 004102'                 TEXT2:  MES2
003766' 177777                          177777
003770' 004024'                 TEXT1:  MES1
003772' 177777                          177777
003774' 004000'                 TEXT0:  MES0
003776' 177777                          177777
004000' 047045  020117  042504  MES0:   .ASCIZ  '%NO DEVICES TO RUN%'
004006' 044526  042503  020123
004014' 047524  051040  047125
004022' 000045
004024' 042045  053105  041511  MES1:   .ASCIZ  '%DEVICE ADRS IS TOO HIGH MAX VALUE IS 170160%'
004032' 020105  042101  051522
004040' 044440  020123  047524
004046' 020117  044510  044107
004054' 046440  054101  053040
004062' 046101  042525  044440
004070' 020123  033461  030460
004076' 030066  000045
004102' 040445  042040  053105  MES2:   .ASCIZ  '%A DEVICE,Q22BE,HAS BEEN DROPPED DUE TO AN INTERRUPT ERROR%'
004110' 041511  026105  031121
004116' 041062  026105  040510
004124' 020123  042502  047105
004132' 042040  047522  050120
004140' 042105  042040  042525
004146' 052040  020117  047101
004154' 044440  052116  051105
004162' 052522  052120  042440
004170' 051122  051117  000045
004176' 044445  041516  051117  MES3:   .ASCIZ  '%INCORRECT NUMBER OF INTERRUPTS%'
004204' 042522  052103  047040
004212' 046525  042502  020122
004220' 043117  044440  052116
004226' 051105  052522  052120
004234' 022523     000

                004240'                 .EVEN

004240' 000256                  RBUF:   .BLKW   256
                000001                  .END
```

```
ACSR    000102R          199#  1269*  1562*
ADDR    000006R          165#
ADDR22= 001000           228#  1118   1176   1247
ADR     000350R          268#  1121*  1126   1179*  1185
ADRSHT  002304R          610    807   1045#
ASB     000106R          203#  1275*
ASSEMB= 000010           234
ASTAT   000104R          201#
AWAS    000110R          204#  1277#
BA      000334R          262#   911*   912*   934*  1130*  1144*  1190*  1203*  1644*  1645*
BEGIN   000000R          162#   431    432    531    532    556    573    873   1126   1185   1282   1381   1382
                        1425   1426   1479   1538   1592   1614
BIT0  = 000001           228#
BIT00 = 000001           291#   298    731   1056   1402   1522
BIT01 = 000002           290#   310    686
BIT02 = 000004           289#   316
BIT03 = 000010           288#   322
BIT04 = 000020           287#   328   1146   1205
BIT05 = 000040           286#   334   1153   1212
BIT06 = 000100           285#   340    401
BIT07 = 000200           284#   346   1374   1411
BIT08 = 000400           283#
BIT09 = 001000           282#   652    724
BIT1  = 000002           228#
BIT10 = 002000           228#   281#  1149   1208
BIT11 = 004000           228#   280#  1156   1215
BIT12 = 010000           228#   279#
BIT13 = 020000           228#   278#
BIT14 = 040000           228#   277#  1584   1608
BIT15 = 100000           228#   276#
BIT2  = 000004           228#
BIT22   002414R          645    823   1116#
BIT23   002544R          716    832   1173#
BIT3  = 000010           228#
BIT4  = 000020           228#
BIT5  = 000040           228#
BIT6  = 000100           228#
BIT7  = 000200           228#
BIT8  = 000400           228#
BIT9  = 001000           228#
BLKMBT  000502R          357#   560*   623    649    672    721    734    738*   743*   752*
BREAK$= 104407           228#  1381   1382   1425   1426
BR1     000012R          167#
BR2     000013R          168#
BTOD$ = 104421           228#
CDATA$= 104412           228#
CDERCT  000144R          218#
CDWDCT  000146R          219#
CHKDA   002674R          661   1234#
CHKERR  003520R         1440   1557#
CHKINT  003352R         1437   1455#
CLRREG  001730R          516    900#
CONFIG  000056R          187#  1118   1176   1247
CSRA    000100R          197#  1560#
CSR1    000330R          260#   626*   631*   675*   680*   821*   830*   906*   932*  1087  1138*  1149*  1156*
                        1198*  1208*  1215*  1269   1641*  1642*
```



```
CSR2    000332R          261#   908*   909*   940*  1329*  1330*  1335*  1337*  1342  1347*  1352*  1358*  1404*
                        1411   1608   1639*
DATA    000340R          264#   917*   918*   938*  1650*  1651*
DATCK$= 104411           228#
DATER$= 104404           228#  1282
DEVADR  000324R          257#   494*   522*   526    584*   593    764*   804*   904   942   970*  1059   1083
DEVECT  000326R          258#   496*   524*   600*   802*   945    953    966*   968*  972*
DIAGMC= 000000           234
DRPDEV  003434R         1467   1517#
DV      000254R          253#   409*   411*   421    423    425   1074   1543*
DVID1   000014R          169#   409
EA22    000356R          271#  1133*  1134*  1136*  1138   1192*  1193*  1195*  1198
ENDIT$= 104413           228#   873
END$  = 104410           228#   432    532
ERFLG = 000400           516#   610#   658#   661#   728#   807#   991#   993   1437#  1440#
ERRTYP  000106R          202#  1566*
ESR13   001132R          591#   768
ESR33   001536R          787#   862
EXIT$ = 104400           228#
FILTAB  003656R         1587   1636#
FIRADR  000510R          360#   494   1054   1520
FIRVEC  000512R          361#   496    948
FREE    000150R          220#
GETPA$= 104415           228#   556
GO      003012R          658    728    854   1311#
GWBUF$= 104414           228#   573
HRDCNT  000044R          182#
HRDER$= 104405           228#  1592   1614
HRDPAS  000050R          184#
ICONT   000036R          179#
ICOUNT  000040R          180#
IDNUM   000122R          209#
IMODX.= 000000           221#   574
INIT    000030R          176#
INTCNT  000524R          366#   566*   849*  1475   1503*
INTR    000120R          208#
INTRP   000504R          358#   407*   450*  1475   1545*
INTSRV  003422R          300    312    318    324    330    336    342    348   1500#
LATCNT  000342R          265#   563    705    920*   921*   930   1653*  1654*
LPCNTR  000514R          362#  1314*  1372   1377*  1408   1414*  1419*
MAP22$= 104416           228#  1126   1185
MASK    000506R          359#   421*  1048*  1069*  1074*  1464   1526
MES0    004000R         1672   1674#
MES1    004024R         1670   1678#
MES2    004102R         1668   1686#
MES3    004176R         1666   1696#
MODNAM  000000R          163#
MODSP   000252R          177    226#   467    568
MORE    000256R          254#   423*   507    512*  1522*  1526   1529*  1543
MSGN$ = 104403           228#   431    531   1479   1538
MSGS$ = 104402           228#
MSG$  = 104401           228#
MVLCNT  000344R          266#   923*   924*  1656*  1657*
NULL  = 000000           228#
OPEN  = 000000           164    170    171    172    173    190    191    192   193   194   195   196   197
                         199    201    203    204    206    207    208    211   212   214   215   217   218
```

```
                        219     220     228#
OTOAS = 104420          228#
PASCNT  000034R         178#
PASS    001720R         869#
PA22    000354R         270#    1130    1190    1250
PIRQS = 000004          228#
POPSP = 005726          228#
POPSP2= 022626          228#
PRTY  = 000000          228#
PRTY0 = 000000          168     228#
PRTY1 = 000040          228#
PRTY2 = 000100          228#
PRTY3 = 000140          228#
PRTY4 = 000200          228#
PRTY5 = 000240          228#
PRTY6 = 000300          228#
PRTY7 = 000340          167     228#
PS    = 177776          228#
PSW   = 177776          228#
PUSH  = 005746          228#
PUSH2 = 024646          228#
Q22TB   000260R         255#    480     498     584     586     792     1322    1574
RAND$ = 104417          228#
RANNUM  000054R         186#
RBUF    004240R         210     381     1706#
RBUFEA  000130R         212#    1123    1146    1153
RBUFPA  000126R         211#    1121    1144    1255
RBUFSZ  000132R         213#    385
RBUFVA  000124R         210#    381*    383     397*    556     1237
REENTY  000640R         419#    1547
RESTRT  001054R         205     550#    880
RES1    000056R         188#
RES2    000060R         189#
ROUT1   002172R         647     718     816     989#
RSTRT   000112R         205#
RY      000534R         370#    563#    1324*   1326    1360    1369*   1374    1391*
SBADR   000102R         198#    1271#
SIMGBT  000520R         364#    570#    784#    1051    1316    1472    1571
SIMGU   001516R         596     778#
SIMGUA  000346R         267#    926#    927#    1326#   1327#   1329    1361#   1564    1568#   1624#
SOFCNT  000942R         181#
SOFEPS= 104406          228#
SOFPAS  000046R         183#
SPOINT  000032R         177#
SPSIZ = 000040            1#    221
SR1     000016R         170#    686     731
SR2     000020R         171#
SR3     000022R         172#
SR4     000024R         173#
START   000536R         176     375#
STAT    000026R         175#
SVADR   000360R         272#    1520#   1531#
SVINLV  000252R         252#    608#    619     745#    754#    780#    857#    859     991     994
SVINT   000522R         298#    310#    316#    322#    328#    334#    340#    346#    365#    852#    1464    1487#
SVR0    000062R         190#
SVR1    000064R         191#
```



```
SVR2    000066R         192#
SVR3    000070R         193#
SVR4    000072R         194#
SVR5    000074R         195#
SVR6    000076R         196#
SVWDCT  000532R         369#    385#    399#    639     812     1245
SYM   = 000067          991#
SYMD  = 000007          298#    310#    316#    322#    328#    334#    340#    346#    387#    395#    397#    399#    401#
                        411#    446#    450#    488#    512#    522#    524#    639#    652#    724#    745#    754#    762#
                        766#    812#    838#    840#    857#    908#    911#    914#    917#    920#    923#    926#    956#
                        958#    966#    970#    972#    1064#   1069#   1095#   1097#   1133#   1138#   1149#   1156#   1192#
                       1198#   1208#   1215#   1259#   1290#   1292#   1294#   1326#   1329#   1337#   1339#   1347#   1352#
                       1358#   1369#   1377#   1389#   1402#   1419#   1503#   1531#   1543#   1545#   1582#   1600#   1641#
                       1644#   1647#   1650#   1653#   1656#
SYMS  = 000007          298#    310#    316#    322#    328#    334#    340#    346#    387#    395#    397#    399#    401#
                        411#    446#    450#    488#    512#    522#    524#    639#    652#    724#    745#    754#    762#
                        766#    812#    838#    840#    857#    908#    911#    914#    917#    920#    923#    926#    956#
                        958#    966#    970#    972#    1064#   1069#   1095#   1097#   1133#   1138#   1149#   1156#   1192#
                       1198#   1208#   1215#   1259#   1290#   1292#   1294#   1326#   1329#   1337#   1339#   1347#   1352#
                       1358#   1369#   1377#   1389#   1402#   1419#   1503#   1531#   1543#   1545#   1582#   1600#   1641#
                       1644#   1647#   1650#   1653#   1656#
SYSCNT  000052R         185#
TABLE1  000330R         259#    1592    1614
TEXT0   003774R         431     1672#
TEXT1   003770R         531     1670#
TEXT2   003764R        1538     1668#
TEXT3   003760R        1479     1666#
TMPR0   000530R         368#    1320*   1396    1564*   1624
TMPSR2  000526R         367#    652*    724*    1009*   1014*   1019*   1025*   1335    1402*   1404
TRPDFD= 000022          228#
VECT    000302R         256#    500     588     790
VECTOR  000010R         166#
VECT0   000362R         296#    950
VECT1   000374R         308#
VECT2   000406R         314#
VECT3   000420R         320#
VECT4   000432R         326#
VECT5   000444R         332#
VECT6   000456R         338#
VECT7   000470R         344#
WASADR  000104R         200#    1273#
WBUFEA  000136R         215#    1181    1205    1212
WBUFPA  000134R         214#    1179    1203
WBUFRW  000140R         216#
WBUFSZ  000142R         217#    689
WC      000336R         263#    641*    703*    710*    814*    914*    915*    936*    1647*   1648*
WDFR    000116R         207#
WDTO    000114R         206#
WWW   = ****** GX       209
XFLAG   000005R         164#
XMEM    000352R         269#    1123*   1181*
$BGNLE= 177777          234#
$BRJMP= 177777          234#    391     393     403     428     444     448     454     484     490     508     514     519
                        527     541     594     620     624     628     650     673     677     687     692     696     707
                        722     732     735     740     749     758     799     819     827     842     860     954     960
                        993     995     1011    1016    1021    1052    1060    1066    1071    1091    1099    1119    1141
```

	1147	1154	1177	1200	1206	1213	1248	1252	1264	1267	1296	1317	1343
	1345	1349	1355	1367	1373	1375	1386	1393	1399	1409	1412	1416	1431
	1465	1473	1476	1527	1533	1572	1578	1585	1602	1605	1609		
$ERFLG= 000460	234#	298#	299	310#	311	316#	317	322#	323	328#	329	334#	335
	340#	341	346#	347	381#	382	383#	384	385#	386	387#	388	395#
	396	397#	398	399#	400	401#	402	407#	408	409#	410	411#	412
	421#	422	423#	424	425#	426	446#	447	450#	451	467#	468	478#
	479	480#	481	486#	487	488#	489	494#	495	496#	497	498#	499
	500#	501	512#	513	522#	523	524#	525	560#	561	563#	564	566#
	567	568#	569	570#	571	584#	585	586#	587	588#	589	600#	601
	608#	609	626#	627	631#	632	637#	638	639#	640	641#	642	652#
	653	675#	676	680#	681	689#	690	694#	695	703#	704	705#	706
	710#	711	724#	725	738#	739	743#	744	745#	746	752#	753	754#
	755	762#	763	764#	765	766#	767	780#	781	784#	785	790#	791
	792#	793	794#	795	802#	803	804#	805	810#	811	812#	813	814#
	815	821#	822	825#	826	830#	831	834#	835	838#	839	840#	841
	849#	850	852#	853	857#	858	904#	905	906#	907	908#	909#	910
	911#	912#	913	914#	915#	916	917#	918#	919	920#	921#	922	923#
	924#	925	926#	927#	928	930#	931	932#	933	934#	935	936#	937
	938#	939	940#	941	942#	943	945#	946	948#	949	950#	951	956#
	957	958#	959	963#	964	966#	967	968#	969	970#	971	972#	973
	1009#	1010	1014#	1015	1019#	1020	1025#	1026	1048#	1049	1054#	1055	1056#
	1057	1064#	1065	1069#	1070	1074#	1075	1083#	1084	1085#	1086	1087#	1088
	1093#	1094	1095#	1096	1097#	1098	1121#	1122	1123#	1124	1130#	1131	1133#
	1135	1138#	1139	1144#	1145	1149#	1150	1156#	1157	1179#	1180	1181#	1182
	1190#	1191	1192#	1194	1198#	1199	1203#	1204	1208#	1209	1215#	1216	1237#
	1238	1245#	1246	1250#	1251	1255#	1256	1259#	1260	1269#	1270	1271#	1272
	1273#	1274	1275#	1276	1277#	1278	1290#	1291	1292#	1293	1294#	1295	1314#
	1315	1320#	1321	1322#	1323	1324#	1325	1326#	1327#	1328	1329#	1330#	1331
	1335#	1336	1337#	1338	1339#	1340	1347#	1348	1352#	1353	1358#	1359	1361#
	1362	1369#	1370	1377#	1378	1389#	1390	1391#	1392	1396#	1397	1402#	1403
	1404#	1405	1414#	1415	1419#	1420	1487#	1488	1503#	1504	1520#	1521	1522#
	1523	1531#	1532	1543#	1544	1545#	1546	1560#	1561	1562#	1563	1564#	1565
	1566#	1567	1568#	1569	1574#	1575	1580#	1581	1582#	1583	1600#	1601	1624#
	1625	1639#	1640	1641#	1642#	1643	1644#	1645#	1646	1647#	1648#	1649	1650#
	1651#	1652	1653#	1654#	1655	1656#	1657#	1658					
$FSAND= 000310	234#	392	429	445	485	509	528	595	621	625	651	674	688
	693	723	733	736	800	820	861	955	1053	1061	1092	1120	1148
	1155	1178	1207	1214	1249	1265	1268	1318	1344	1368	1374	1410	1413
	1466	1474	1477	1528	1573	1579	1586	1610					
$FSBAD= 000401	234#	298	310	316	322	328	334	340	346	381	383	385	387
	394	395	397	399	401	407	409	411	421	423	425	429	445
	446	450	467	478	480	485	486	488	494	496	498	500	509
	512	522	524	528	560	563	566	568	570	584	586	588	595
	600	608	621	625	626	631	637	639	641	651	652	674	675
	680	688	689	693	694	703	705	710	723	724	733	736	738
	743	745	752	754	762	764	766	780	784	790	792	794	800
	802	804	810	812	814	820	821	825	830	834	838	840	849
	852	857	861	904	906	908	911	914	917	920	923	926	930
	932	934	936	938	940	942	945	948	950	955	956	958	963
	966	968	970	972	991	1009	1014	1019	1025	1048	1053	1054	1056
	1061	1064	1069	1074	1083	1085	1087	1092	1093	1095	1097	1120	1121
	1123	1130	1133	1138	1144	1148	1149	1155	1156	1178	1179	1181	1190
	1192	1198	1203	1207	1208	1214	1215	1237	1245	1249	1250	1255	1259
	1265	1268	1269	1271	1273	1275	1277	1290	1292	1294	1314	1318	1320
	1327	1324	1326	1329	1335	1337	1339	1346	1347	1352	1358	1361	1368

	1369	1376	1377	1389	1391	1396	1402	1404	1410	1413	1414	1419	1466
	1474	1477	1487	1503	1520	1522	1528	1531	1543	1545	1560	1562	1564
	1566	1568	1573	1574	1579	1580	1582	1586	1600	1610	1624	1639	1641
	1644	1647	1650	1653	1656								
$FSBLA= 000170	234#												
$FSCAS= 000150	234#												
$FSDEC= 000220	234#												
$FSDO = 000340	234#	394	445	485	509	621	800	955	1061	1092	1265	1346	1368
	1376	1410	1528	1579									
$FSFAL= 000405	234#	1005											
$FSGOO= 000400	234#	298	299	310	311	316	317	322	323	328	329	334	335
	340	341	346	347	381	382	383	384	385	386	387	388	389
	390	395	396	397	398	399	400	401	402	407	408	409	410
	411	412	421	422	423	424	425	426	427	442	443	446	447
	448	450	451	467	468	478	479	480	481	482	483	486	487
	488	489	494	495	496	497	498	499	500	501	506	507	512
	513	514	516	522	523	524	525	526	560	561	563	564	566
	567	568	569	570	571	584	585	586	587	588	589	593	600
	601	608	609	610	618	619	623	626	627	631	632	637	638
	639	640	641	642	649	652	653	658	661	672	675	676	680
	681	686	689	690	691	694	695	703	704	705	706	710	711
	721	724	725	728	731	734	738	739	743	744	745	746	752
	753	754	755	762	763	764	765	766	767	780	781	784	785
	790	791	792	793	794	795	797	798	802	803	804	805	807
	810	811	812	813	814	815	818	821	822	825	826	830	831
	834	835	838	839	840	841	849	850	852	853	857	858	859
	904	905	906	907	908	909	910	911	912	913	914	915	916
	917	918	919	920	921	922	923	924	925	926	927	928	930
	931	932	933	934	935	936	937	938	939	940	941	942	943
	945	946	948	949	950	951	952	953	956	957	958	959	963
	964	966	967	968	969	970	971	972	973	991	993	1009	1010
	1014	1015	1019	1020	1025	1026	1048	1049	1051	1054	1055	1056	1057
	1058	1059	1064	1065	1069	1070	1074	1075	1083	1084	1085	1086	1087
	1088	1089	1090	1093	1094	1095	1096	1097	1098	1118	1121	1122	1123
	1124	1130	1131	1133	1135	1138	1139	1144	1145	1146	1149	1150	1153
	1156	1157	1176	1179	1180	1181	1182	1190	1191	1192	1194	1198	1199
	1203	1204	1205	1208	1209	1212	1215	1216	1237	1238	1245	1246	1247
	1250	1251	1255	1256	1259	1260	1262	1263	1266	1269	1270	1271	1272
	1273	1274	1275	1276	1277	1278	1290	1291	1292	1293	1294	1295	1314
	1315	1316	1320	1321	1322	1323	1324	1325	1326	1327	1328	1329	1330
	1331	1335	1336	1337	1338	1339	1340	1341	1342	1347	1348	1352	1353
	1358	1359	1361	1362	1365	1366	1369	1370	1371	1372	1377	1378	1389
	1390	1391	1392	1396	1397	1402	1403	1404	1405	1407	1408	1411	1414
	1415	1419	1420	1437	1440	1464	1472	1475	1487	1488	1503	1504	1520
	1521	1522	1523	1525	1526	1531	1532	1543	1544	1545	1546	1560	1561
	1562	1563	1564	1565	1566	1567	1568	1569	1571	1574	1575	1576	1577
	1580	1581	1582	1583	1584	1600	1601	1608	1624	1625	1639	1640	1641
	1642	1643	1644	1645	1646	1647	1648	1649	1650	1651	1652	1653	1654
	1655	1656	1657	1658									
$FSIF = 000110	234#	390	427	436	448	452	514	519	526	537	539	593	598
	623	628	633	649	654	672	677	682	686	691	696	701	707
	712	721	726	731	734	740	747	749	756	818	827	836	859
	864	1051	1071	1076	1118	1141	1146	1151	1153	1158	1161	1176	1200
	1205	1210	1212	1217	1220	1247	1252	1257	1266	1288	1316	1342	1372
	1399	1411	1416	1421	1434	1464	1469	1472	1475	1483	1485	1571	1584
	1598	1605	1608	1620	1622								

```
$FSINC= 000210   234#
$FSLOO= 000200   234#
$FSNAM= 000160   234#
$FSNO = 000403   234#  298  299  310  311  316  317  322  323  328  329  334  335
                 340  341  346  347  361  382  383  384  385  386  387  388  390
                 392  395  396  397  398  399  400  401  402  407  408  409  410
                 411  412  421  422  423  424  425  426  427  443  446  447  449
                 450  451  467  468  478  479  480  481  483  486  487  488  489
                 494  495  496  497  498  499  500  501  507  512  513  515  522
                 523  524  525  526  528  560  561  563  564  566  567  568  569
                 570  571  584  585  586  587  588  589  593  600  601  608  609
                 619  621  623  625  626  627  631  632  637  638  639  640  641
                 642  649  651  652  653  672  674  675  676  680  681  686  689
                 690  691  693  694  695  703  704  705  706  710  711  721  723
                 724  725  731  734  736  738  739  743  744  745  746  752  753
                 754  755  762  763  764  765  766  767  780  781  784  785  790
                 791  792  793  794  795  798  802  803  804  805  810  811  812
                 813  814  815  818  820  821  822  825  826  830  831  834  835
                 838  839  840  841  849  850  852  853  857  858  859  861  904
                 905  906  907  908  910  911  913  914  916  917  919  920  922
                 923  925  926  928  930  931  932  933  934  935  936  937  938
                 939  940  941  942  943  945  946  948  949  950  951  953  955
                 956  957  958  959  963  964  966  967  968  969  970  971  972
                 973  992  996 1009 1010 1014 1015 1019 1020 1025 1026 1048 1049
                1051 1054 1055 1056 1057 1059 1061 1064 1065 1069 1070 1074 1075
                1083 1084 1085 1086 1087 1088 1090 1093 1094 1095 1096 1097 1098
                1118 1121 1122 1123 1124 1130 1131 1133 1135 1138 1139 1144 1145
                1146 1149 1150 1153 1156 1157 1176 1179 1180 1181 1182 1190 1191
                1192 1194 1198 1199 1203 1204 1205 1208 1209 1212 1215 1216 1237
                1238 1245 1246 1247 1250 1251 1255 1256 1259 1260 1263 1266 1268
                1269 1270 1271 1272 1273 1274 1275 1276 1277 1278 1290 1291 1292
                1293 1294 1295 1314 1315 1316 1318 1320 1321 1322 1323 1324 1325
                1326 1328 1329 1331 1335 1336 1337 1338 1339 1340 1342 1344 1347
                1348 1352 1353 1358 1359 1361 1362 1366 1369 1370 1372 1374 1377
                1378 1389 1390 1391 1392 1396 1397 1402 1403 1404 1405 1408 1411
                1414 1415 1419 1420 1464 1472 1474 1475 1477 1487 1488 1503 1504
                1520 1521 1522 1523 1526 1531 1532 1543 1544 1545 1546 1560 1561
                1562 1563 1564 1565 1566 1567 1568 1569 1571 1573 1574 1575 1577
                1580 1581 1582 1583 1584 1600 1601 1608 1624 1625 1639 1640 1641
                1643 1644 1646 1647 1649 1650 1652 1653 1655 1656 1658
$FSOR = 000320   234#  392  394  429  445  485  509  528  595  621  625  651  674
                 688  693  723  733  736  800  820  861  955 1053 1061 1092 1120
                1148 1155 1178 1207 1214 1249 1265 1268 1318 1344 1346 1368 1374
                1376 1410 1413 1466 1474 1477 1528 1573 1579 1586 1610
$FSRTI= 000350   234#
$FSRTN= 000300   234#  901  975 1046 1104 1235 1300 1312 1443 1456 1490 1558 1627
$FSSEL= 000140   234#  991 1007 1011 1016 1021 1027
$FSTHE= 000330   234#  429  528  595  625  651  674  688  693  723  733  736  820
                 861 1053 1120 1148 1155 1178 1207 1214 1249 1268 1318 1413 1466
                1474 1477 1573 1586 1610
$FSTPU= 000404   234# 1007 1011 1012 1016 1017
$FSUNI= 000130   234# 1333 1354
$FSWHI= 000120   234#  389  390  392  403  442  443  454  482  483  490  506  507
                 541  618  619  758  797  798  842  952  953  960 1058 1059 1066
                1089 1090 1099 1262 1263 1296 1341 1342 1344 1349 1365 1366 1371
                1372 1374 1386 1393 1407 1408 1431 1525 1526 1533 1576 1577 1602
```



```
$FSYES= 000402   234#  299  311  317  323  329  335  341  347  382  384  386  388
                 390  392  396  398  400  402  408  410  412  422  424  426  427
                 436  443  447  448  451  452  468  479  481  483  487  489  495
                 497  499  501  507  513  514  519  523  525  526  537  539  561
                 564  567  569  571  585  587  589  593  598  601  609  619  623
                 627  628  632  633  638  640  642  649  653  654  672  676  677
                 681  682  686  690  691  695  696  701  704  706  707  711  712
                 721  725  726  731  734  739  740  744  746  747  749  753  755
                 756  763  765  767  781  785  791  793  795  798  803  805  811
                 813  815  818  822  826  827  831  835  836  839  841  850  853
                 858  859  864  905  907  910  913  916  919  922  925  928  931
                 933  935  937  939  941  943  946  949  951  953  957  959  964
                 967  969  971  973 1010 1015 1020 1026 1030 1049 1051 1055 1057
                1059 1065 1070 1071 1075 1076 1084 1086 1088 1090 1094 1096 1098
                1118 1122 1124 1131 1135 1139 1141 1145 1146 1150 1151 1153 1157
                1158 1161 1176 1180 1182 1191 1194 1199 1200 1204 1205 1209 1210
                1212 1216 1217 1220 1238 1246 1247 1251 1252 1256 1257 1260 1263
                1266 1270 1272 1274 1276 1278 1288 1291 1293 1295 1315 1316 1321
                1323 1325 1328 1331 1336 1338 1340 1342 1344 1348 1353 1359 1362
                1366 1370 1372 1374 1378 1390 1392 1397 1399 1403 1405 1408 1411
                1415 1416 1420 1421 1434 1464 1469 1472 1475 1483 1485 1488 1504
                1521 1523 1526 1532 1544 1546 1561 1563 1565 1567 1569 1571 1575
                1577 1581 1583 1584 1598 1601 1605 1608 1620 1622 1625 1640 1643
                1646 1649 1652 1655 1658
$IFLEV= 177777   234#  427#  436#  448#  452#  514#  526#  537#  539#  593#  598#  623#  633#
                 649#  654#  672#  682#  686#  691#  701#  712#  721#  726#  731#  734#  747#
                 756#  818#  836#  859#  864# 1051# 1076# 1118# 1146# 1151# 1153# 1158# 1161#
                1176# 1205# 1210# 1212# 1217# 1220# 1247# 1257# 1266# 1288# 1316# 1411# 1421#
                1434# 1464# 1469# 1472# 1475# 1483# 1485# 1571# 1584# 1598# 1608# 1620# 1622#
$ISK0 = 000001    427#  436  448#  452  514#  539  593#  598  623#  633  649#  654  672#
                 682  686#  712  721#  726  731#  756  818#  836  859#  864 1051# 1076
                1118# 1161 1176# 1220 1247# 1257 1266# 1288 1316# 1434 1464# 1469 1472#
                1485 1571# 1622
$ISK1 = 000001    526#  537  691#  701  734#  747 1146# 1151 1153# 1158 1205# 1210 1212#
                1217 1411# 1421 1475# 1483 1584# 1598 1608# 1620
$LO   = 000002    992# 1000 1005 1007# 1012# 1017#
$LOCTA= 000001    234#  236#  389  390  391  392  393  394  403  404  405  428  429
                 436  437  442  443  444  445  448  449  452  453  454  455  456
                 482  483  484  485  490  491  492  506  507  508  509  514  515
                 519  520  521  527  528  537  538  539  540  541  542  543  594
                 595  598  599  618  619  620  621  624  625  628  629  630  633
                 634  650  651  654  655  673  674  677  678  679  682  683  687
                 688  692  693  696  697  698  701  702  707  708  709  712  713
                 722  723  726  727  732  733  735  736  740  741  742  747  748
                 749  750  751  756  757  758  759  760  797  798  799  800  819
                 820  827  828  829  830  837  842  843  844  860  861  864  865
                 900  952  953  954  955  960  961  962  975  976  977  993  994
                 995  996  997  999 1000 1001 1002 1003 1004 1007 1008 1011 1012
                1013 1016 1017 1018 1021 1022 1023 1024 1027 1028 1029 1030 1031
                1045 1052 1053 1058 1059 1060 1061 1066 1067 1068 1071 1072 1073
                1076 1077 1089 1090 1091 1092 1099 1100 1101 1104 1105 1106 1119
                1120 1141 1142 1143 1147 1148 1151 1152 1154 1155 1158 1159 1161
                1162 1177 1178 1200 1201 1202 1206 1207 1210 1211 1213 1214 1217
                1218 1220 1221 1234 1248 1249 1252 1253 1254 1257 1258 1262 1263
                1264 1265 1267 1268 1288 1289 1296 1297 1298 1300 1301 1302 1311
                1317 1318 1333 1334 1341 1342 1343 1344 1345 1346 1349 1350 1351
```

```
                1355  1356  1365  1366  1367  1368  1371  1372  1373  1374  1375  1376  1386
                1387  1388  1393  1394  1395  1399  1400  1401  1407  1408  1409  1410  1412
                1413  1416  1417  1418  1421  1422  1431  1432  1433  1434  1435  1443  1444
                1445  1455  1465  1466  1469  1470  1473  1474  1476  1477  1483  1484  1485
                1486  1490  1491  1492  1525  1526  1527  1528  1533  1534  1535  1557  1572
                1573  1576  1577  1578  1579  1585  1586  1598  1599  1602  1603  1604  1605
                1606  1607  1609  1610  1620  1621  1622  1623  1627  1628  1629
$LSKO = 000003  1005#  1007  1012  1017
$LSTIN= 000001   234#   298   300   310   312   316   318   322   324   328   330   334   336
                 340   342   346   348   381   383   385   387   390   391   392   393   395
                 397   399   401   403   407   409   411   419   421   423   425   427   428
                 440   443   444   446   448   450   454   467   478   480   483   484   486
                 488   490   494   496   498   500   507   508   510   512   514   516   519
                 522   524   526   527   541   560   563   566   568   570   584   586   588
                 591   593   594   596   600   608   610   619   620   623   624   626   628
                 631   637   639   641   645   647   649   650   652   658   661   672   673
                 675   677   680   686   687   689   691   692   694   696   699   703   705
                 707   710   716   718   721   722   724   728   731   732   734   735   738
                 740   743   745   749   752   754   758   762   764   766   768   780   782
                 784   787   790   792   794   798   799   802   804   807   810   812   814
                 816   818   819   821   823   825   827   830   832   834   838   840   842
                 849   852   854   857   859   860   862   880   904   906   908   909   911
                 912   914   915   917   918   920   921   923   924   926   927   930   932
                 934   936   938   940   942   945   948   950   953   954   956   958   960
                 963   966   968   970   972   977   991   992   993   994   995   996   997
                 998  1000  1001  1002  1003  1004  1009  1011  1014  1016  1019  1021  1025
                1027  1029  1032  1048  1051  1052  1054  1056  1059  1060  1062  1064  1066
                1069  1071  1074  1083  1085  1087  1090  1091  1093  1095  1097  1099  1106
                1118  1119  1121  1123  1130  1133  1134  1136  1138  1141  1144  1146  1147
                1149  1153  1154  1156  1164  1176  1177  1179  1181  1190  1192  1193  1195
                1198  1200  1203  1205  1206  1208  1212  1213  1215  1223  1237  1245  1247
                1248  1250  1252  1255  1259  1263  1264  1266  1267  1269  1271  1273  1275
                1277  1290  1292  1294  1296  1302  1314  1316  1317  1320  1322  1324  1326
                1327  1329  1330  1335  1337  1339  1342  1343  1344  1345  1347  1349  1352
                1354  1355  1358  1361  1363  1366  1367  1369  1372  1373  1374  1375  1377
                1386  1389  1391  1393  1396  1399  1402  1404  1408  1409  1411  1412  1414
                1416  1419  1431  1437  1440  1445  1464  1465  1467  1472  1473  1475  1476
                1487  1492  1503  1506  1508  1510  1520  1522  1526  1527  1529  1531  1533
                1543  1545  1547  1560  1562  1564  1566  1568  1571  1572  1574  1577  1578
                1580  1582  1584  1585  1587  1600  1602  1605  1608  1609  1624  1629  1639
                1641  1642  1644  1645  1647  1648  1650  1651  1653  1654  1656  1657  1660
$LSTTA= 000001   234#   235#   389   404   436   442   452   455   482   491   506   520   537
                 539   542   598   618   629   633   654   678   682   697   701   708   712
                 726   741   747   750   756   759   797   829   836   843   864   900   952
                 961   975   976   999  1007  1012  1017  1022  1023  1028  1030  1045  1058
                1067  1072  1076  1089  1100  1104  1105  1142  1151  1158  1161  1201  1210
                1217  1220  1234  1253  1257  1262  1288  1297  1300  1301  1311  1333  1341
                1350  1365  1371  1387  1394  1400  1407  1417  1421  1432  1434  1443  1444
                1455  1469  1483  1485  1490  1491  1525  1531  1557  1576  1598  1603  1606
                1620  1622  1627  1628
$NESTL= 177777   234#   389#   403#   427#   436#   442#   448#   452#   454#   482#   490#   506#   514#
                 519   526#   537#   539#   541#   593#   598#   618#   623#   628   633#   649#   654#
                 672#   677   682#   686#   691#   696   701#   707   712#   721#   726#   731#   734#
                 740   747#   749   756#   758#   797#   818#   827   836#   842#   859#   864#   901#
                 952#   960#   975#   991#  1007  1011  1016  1021  1027#  1046#  1051#  1058#  1066#
                1071  1076#  1089#  1099#  1104#  1118#  1141  1146#  1151#  1153#  1158#  1161#  1176#
```



```
                1200  1205#  1210#  1212#  1217#  1220#  1235#  1247#  1252  1257#  1262#  1266#  1288#
                1296#  1300#  1312#  1316#  1333#  1341#  1349#  1354#  1365#  1371#  1386#  1393#  1399
                1407#  1411#  1416  1421#  1431#  1434#  1443#  1456#  1464#  1469#  1472#  1475#  1483#
                1485#  1490#  1525#  1533#  1558#  1571#  1576#  1584#  1598#  1602#  1605  1608#  1620#
                1622#  1627#
$NSK0 = 000300   389#   403   427#   436   442#   454   482#   490   506#   541   593#   598   618#
                 758   797#   842   859#   864   901#   975   991#  1007  1011  1016  1021  1027
                1046#  1104  1118#  1141  1161  1176#  1200  1220  1235#  1300  1312#  1443  1456#
                1490  1525#  1533  1558#  1627
$NSK1 = 000110   448#   452   514#   519   539   623#   628   633   649#   654   672#   677   682
                 686#   707   712   721#   726   731#   749   756   818#   827   836   952#   960
                1051#  1071  1076  1089#  1099  1146#  1151  1153#  1158  1205#  1210  1212#  1217
                1247#  1252  1257  1262#  1296  1316#  1399  1434  1464#  1469  1472#  1485  1571#
                1605  1622
$NSK2 = 000110   526#   537   691#   696   701   734#   740   747  1058#  1066  1266#  1288  1333#
                1354  1365#  1393  1407#  1431  1475#  1483  1576#  1602  1608#  1620
$NSK3 = 000110  1341#  1349  1371#  1386  1411#  1416  1421  1584#  1598
$SAVE = 000403   234#   992#  1030#
$SAVE2= 050055   992#   993   995
$SAVLE= 177777   234#   403#   404#   454#   455#   490#   491#   541#   542#   758#   759#   842#   843#
                 960#   961#   992#  1027#  1030#  1066#  1067#  1099#  1100#  1296#  1297#  1349#  1350#
                1386#  1387#  1393#  1394#  1431#  1432#  1533#  1534#  1602#  1603#
$SELLE= 000000   234#  1005#  1007  1012  1017
$SSK0 = 050140   403#   404   454#   455   490#   491   541#   542   758#   759   842#   843   960#
                 961   992#  1030  1066#  1067  1099#  1100  1296#  1297  1349#  1350  1386#  1387
                1393#  1394  1431#  1432  1533#  1534  1602#  1603
$SSK1 = 050055   992#  1027
$TAGLE= 177777   234#   390#   394#   403#   429#   436#   443#   445#   449#   452#   454#   483#   485#
                 490#   507#   509#   515#   520#   521#   528#   537#   539#   541#   595#   598#   619#
                 621#   625#   629#   630#   633#   651#   654#   674#   678#   679#   682#   688#   693#
                 697#   698#   701#   706#   709#   712#   723#   726#   733#   736#   741#   742#   747#
                 750#   751#   756#   758#   798#   800#   820#   828#   829#   836#   842#   861#   864#
                 953#   955#   960#   992#  1000#  1007  1011  1012  1016  1017  1021  1022  1023
                1027#  1031#  1053#  1059#  1061#  1066#  1072#  1073#  1076#  1090#  1092#  1099#  1120#
                1142#  1143#  1148#  1151#  1155#  1158#  1161#  1178#  1201#  1202#  1207#  1210#  1214#
                1217#  1220#  1249#  1253#  1254#  1257#  1263#  1265#  1268#  1288#  1296#  1318#  1334#
                1342#  1346#  1349#  1354#  1366#  1368#  1372#  1376#  1386#  1393#  1400#  1401#  1408#
                1410#  1413#  1417#  1418#  1421#  1431#  1434#  1466#  1469#  1474#  1477#  1483#  1485#
                1526#  1528#  1533#  1573#  1577#  1579#  1586#  1598#  1602#  1606#  1607#  1610#  1620#
                1622#
$TAGNU= 050144   234#   389   390#   391   393   394#   428   429#   442   443#   444   445#   448
                 449#   482   483#   484   485#   506   507#   508   509#   514   515#   519   521#
                 527   528#   594   595#   618   619#   620   621#   624   625#   628   630#   650
                 651#   673   674#   677   679#   687   688#   692   693#   696   698#   707   709#
                 722   723#   732   733#   735   736#   740   742#   749   751#   797   798#   799
                 800#   819   820#   827   829#   860   861#   901#   952   953#   954   955#   992
                 997   999  1000#  1046#  1052  1053#  1058  1059#  1060  1061#  1071  1073#  1089
                1090#  1091  1092#  1119  1120#  1141  1143#  1147  1148#  1154  1155#  1177  1178#
                1200  1202#  1206  1207#  1213  1214#  1235#  1248  1249#  1252  1254#  1262  1263#
                1264  1265#  1267  1268#  1312#  1317  1318#  1333  1334#  1341  1342#  1343  1345
                1346#  1365  1366#  1367  1368#  1371  1372#  1373  1375  1376#  1399  1401#  1407
                1408#  1409  1410#  1412  1413#  1416  1418#  1456#  1465  1466#  1473  1474#  1476
                1477#  1525  1526#  1527  1528#  1558#  1572  1573#  1576  1577#  1578  1579#  1585
                1586#  1605  1607#  1609  1610#
$TEMP = 000402   234#   298#   299#   310#   311#   316#   317#   322#   323#   328#   329#   334#   335#
                 340#   341#   346#   347#   381#   382#   383#   384#   385#   386#   387#   388#   395#
```

```
                396#  397#  398#  399#  400#  401#  402#  403#  404#  407#  408#  409#  410#
                411#  412#  421#  422#  423#  424#  425#  426#  436#  446#  447#  450#  451#
                452#  454#  455#  467#  468#  478#  479#  480#  481#  486#  487#  488#  489#
                490#  491#  494#  495#  496#  497#  498#  499#  500#  501#  512#  513#  519#
                520#  522#  523#  524#  525#  537#  539#  541#  542#  560#  561#  563#  564#
                566#  567#  568#  569#  570#  571#  584#  585#  586#  587#  588#  589#  598#
                600#  601#  608#  609#  626#  627#  628#  629#  631#  632#  633#  637#  638#
                639#  640#  641#  642#  652#  653#  654#  675#  676#  677#  678#  680#  681#
                682#  689#  690#  694#  695#  696#  697#  701#  703#  704#  705#  706#  707#
                708#  710#  711#  712#  724#  725#  726#  738#  739#  740#  741#  743#  744#
                745#  746#  747#  749#  750#  752#  753#  754#  755#  756#  758#  759#  762#
                763#  764#  765#  766#  767#  780#  781#  784#  785#  790#  791#  792#  793#
                794#  795#  802#  803#  804#  805#  810#  811#  812#  813#  814#  815#  821#
                822#  825#  826#  827#  828#  830#  831#  834#  835#  836#  838#  839#  840#
                841#  842#  843#  849#  850#  852#  853#  857#  858#  864#  904#  905#  906#
                907#  908#  910#  911#  913#  914#  916#  917#  919#  920#  922#  923#  925#
                926#  928#  930#  931#  932#  933#  934#  935#  936#  937#  938#  939#  940#
                941#  942#  943#  945#  946#  948#  949#  950#  951#  956#  957#  958#  959#
                960#  961#  963#  964#  966#  967#  968#  969#  970#  971#  972#  973#  975#
                992#  993#  1000  1001#  1002#  1003#  1004#  1005#  1007#  1009#  1010#  1011#  1012#
               1014#  1015#  1016#  1017#  1019#  1020#  1021#  1022#  1023#  1024#  1025#  1026#  1027#
               1031#  1048#  1049#  1054#  1055#  1056#  1057#  1064#  1065#  1066#  1067#  1069#  1070#
               1071#  1072#  1074#  1075#  1076#  1083#  1084#  1085#  1086#  1087#  1088#  1093#  1094#
               1095#  1096#  1097#  1098#  1099#  1100#  1104#  1121#  1122#  1123#  1124#  1130#  1131#
               1133#  1135#  1138#  1139#  1141#  1142#  1144#  1145#  1149#  1150#  1151#  1156#  1157#
               1158#  1161#  1179#  1180#  1181#  1182#  1190#  1191#  1192#  1194#  1198#  1199#  1200#
               1201#  1203#  1204#  1208#  1209#  1210#  1215#  1216#  1217#  1220#  1237#  1238#  1245#
               1246#  1250#  1251#  1252#  1253#  1255#  1256#  1257#  1259#  1260#  1269#  1270#  1271#
               1272#  1273#  1274#  1275#  1276#  1277#  1278#  1288#  1290#  1291#  1292#  1293#  1294#
               1295#  1296#  1297#  1300#  1314#  1315#  1320#  1321#  1322#  1323#  1324#  1325#  1326#
               1328#  1329#  1331#  1335#  1336#  1337#  1338#  1339#  1340#  1347#  1348#  1349#  1350#
               1352#  1353#  1354#  1355  1358#  1359#  1361#  1362#  1369#  1370#  1377#  1378#  1386#
               1387#  1389#  1390#  1391#  1392#  1393#  1394#  1396#  1397#  1399#  1400#  1402#  1403#
               1404#  1405#  1414#  1415#  1416#  1417#  1419#  1420#  1421#  1431#  1432#  1434#  1443#
               1469#  1483#  1485#  1487#  1488#  1490#  1503#  1504#  1520#  1521#  1522#  1523#  1531#
               1532#  1533#  1534#  1543#  1544#  1545#  1546#  1560#  1561#  1562#  1563#  1564#  1565#
               1566#  1567#  1568#  1569#  1574#  1575#  1580#  1581#  1582#  1583#  1598#  1600#  1601#
               1602#  1603#  1605#  1606#  1620#  1622#  1624#  1625#  1627#  1639#  1640#  1641#  1643#
               1644#  1646#  1647#  1649#  1650#  1652#  1653#  1655#  1656#  1658#
$TSK0 = 050142   390#   403   429#   436   443#   454   483#   490   507#   541   595#   598   619#
                 758   798#   842   861#   864   953#   960   992#  1024  1031  1053#  1072  1073#
                1076  1090#  1099  1120#  1142  1143#  1161  1178#  1201  1202#  1220  1249#  1253
                1254#  1257  1263#  1296  1318#  1400  1401#  1434  1466#  1469  1474#  1485  1526#
                1533  1573#  1606  1607#  1622
$TSK1 = 050143   394#   403   445#   454   465#   490   509#   541   621#   758   800#   842   955#
                 960  1000#  1007  1008#  1024  1031  1059#  1066  1092#  1099  1148#  1151  1155#
                1158  1207#  1210  1214#  1217  1265#  1296  1334#  1354  1366#  1393  1408#  1431
                1477#  1483  1528#  1533  1577#  1602  1610#  1620
$TSK2 = 050140   449#   452   515#   520   521#   539   625#   629   630#   633   651#   654   674#
                 678   679#   682   688#   708   709#   712   723#   726   733#   750   751#   756
                 820#   828   829#   836  1000#  1012  1013#  1024  1031  1061#  1066  1268#  1288
                1342#  1349  1368#  1393  1410#  1431  1579#  1602
$TSK3 = 050141   528#   537   693#   697   698#   701   736#   741   742#   747  1000#  1017  1018#
                1024  1031  1346#  1349  1372#  1386  1413#  1417  1418#  1421  1586#  1598
$TSK4 = 050117  1000#  1023  1024#  1031  1376#  1386
$TSK5 = 177777  1000#  1022  1023#  1031
```



```
$TSK6 = 050056  1000#  1011   1016   1021   1027
$$ARGC= 000000   234#   901#  1046#  1235#  1312#  1456#  1558#
$$BYTE= 000403   234#   390#   392#   427#   443#   483#   507#   526#   593#   619#   623#   649#   672#
                 686#   691#   721#   731#   734#   798#   818#   859#   953#  1051#  1059#  1090#  1118#
                1146#  1153#  1176#  1205#  1212#  1247#  1263#  1266#  1316#  1342#  1344#  1366#  1372#
                1374#  1408#  1411#  1464#  1472#  1475#  1526#  1571#  1577#  1584#  1608#
$$CASE= 000404   234#  1005#  1007#  1011   1012#  1016   1017#
$$DST = 000000   234#
$$ELOC= 000402   234#   427#   436#   448#   452#   514#   519   526#   537#   539#   593#   598#   623#
                 628   633#   649#   654#   672#   677   682#   686#   691#   696   701#   707   712#
                 721#   726#   731#   734#   740   747#   749   756#   818#   827   836#   859#   864#
                1051#  1071   1076#  1118#  1141   1146#  1151#  1153#  1158#  1161#  1176#  1200  1205#
                1210#  1212#  1217#  1220#  1247#  1252   1257#  1266#  1288#  1316#  1399   1411#  1416
                1421#  1434#  1464#  1469#  1472#  1475#  1483#  1485#  1571#  1584#  1598#  1605   1608#
                1620#  1622#
$$ERFL= 000000   234#
$$FLAG= 000001   234#   389#   390#   392#   394#   427#   429#   436#   442#   443#   445#   448#   452#
                 482#   483#   485#   506#   507#   509#   514#   526#   528#   537#   539#   593#   595#
                 598#   618#   619#   621#   623#   625#   633#   649#   651#   654#   672#   674#   682#
                 686#   688#   691#   693#   701#   712#   721#   723#   726#   731#   733#   734#   736#
                 747#   756#   797#   798#   800#   818#   820#   836#   859#   861#   864#   952#   953#
                 955#  1051#  1053#  1058#  1059#  1061#  1076#  1089#  1090#  1092#  1118#  1120#  1146#
                1148#  1151#  1153#  1155#  1158#  1161#  1176#  1178#  1205#  1207#  1210#  1212#  1214#
                1217#  1220#  1247#  1249#  1257#  1262#  1263#  1265#  1266#  1268#  1288#  1316#  1318#
                1341#  1342#  1344#  1346#  1365#  1366#  1368#  1371#  1372#  1374#  1376#  1407#  1408#
                1410#  1411#  1413#  1421#  1434#  1464#  1466#  1469#  1472#  1474#  1475#  1477#  1483#
                1485#  1525#  1526#  1528#  1571#  1573#  1576#  1577#  1579#  1584#  1586#  1598#  1608#
                1610#  1620#  1622#
$$FRMB= 000000   516#   610#   658#   661#   728#   807#  1437#  1440#
$$FROM= 000000   234#   516#   610#   658#   661#   728#   807#  1437#  1440#
$$IN  = 000000   516#   610#   658#   661#   728#   807#  1437#  1440#
$$INH = 000402   234#   390#   392#   427#   443#   449#   483#   507#   515#   526#   528#   593#   619#
                 621#   623#   625#   649#   651#   672#   674#   686#   691#   693#   721#   723#   731#
                 734#   736#   798#   818#   820#   859#   861#   953#   955#   996#  1051#  1059#  1061#
                1090#  1118#  1146#  1153#  1176#  1205#  1212#  1247#  1263#  1266#  1268#  1316#  1318#
                1342#  1344#  1366#  1372#  1374#  1408#  1411#  1464#  1472#  1474#  1475#  1477#  1526#
                1571#  1573#  1577#  1584#  1608#
$$LOC = 003636R  234#   391#   392   393#   394   428#   429   441#   445   448#   449   484#   485
                 508#   509   514#   515   527#   528   594#   595   620#   621   624#   625   650#
                 651   673#   674   687#   688   692#   693   722#   723   732#   733   735#   736
                 799#   800   819#   820   860#   861   954#   955   993#   994   995#   996  1052#
                1053  1060#  1061  1091#  1092  1119#  1120  1147#  1148  1154#  1155  1177#  1178
                1206#  1207  1213#  1214  1248#  1249  1264#  1265  1267#  1268  1317#  1318  1343#
                1344  1345#  1346  1355#  1356  1367#  1368  1373#  1374  1375#  1376  1409#  1410
                1412#  1413  1465#  1466  1473#  1474  1476#  1477  1527#  1528  1572#  1573  1578#
                1579  1585#  1586  1609#  1610
$$LOCN= 000000   234#   901#   977   1046#  1106   1235#  1302   1312#  1445   1456#  1492   1558#  1629
$$OUT = 000000   516#   610#   658#   661#   728#   807#  1437#  1440#
$$REG = 177777   234#
$$RETU= 000000   234#   901#   976   1046#  1105   1235#  1301   1312#  1444   1456#  1491   1558#  1628
$$RTN1= 050134   234#   901#   975   1046#  1104   1235#  1300   1312#  1443   1456#  1490   1558#  1627
$$RTN2= 050135   234#   901#   976   1046#  1105   1235#  1301   1312#  1444   1456#  1491   1558#  1628
$$SRC = 000000   234#
$$TGSV= 050056   234#  1027#  1030   1031
$$TGS1= 050055   234#  1007#  1008   1012#  1013   1017#  1018   1022#  1023#  1024#  1027#  1028
$$TGS2= 050056   234#  1021#  1022   1023   1024
```

```
$$TO  = 000000          234#   516#   610#   658#   661#   728#   807#  1437#  1440#
$$TOB = 000000          516#   610#   658#   661#   728#   807#  1437#  1440#
$$TOTL= 000000          516#   517    610#   611    658#   659    661#   662    728#   729    807#   808    1437#
                       1438   1440#  1441
$$$TAG= 050000          234#
$50000  000562R         389#   403
$50001  000620R         391    393    404#
$50002  000676R         428    436#
$50003  000700R         442#   454
$50004  000716R         444    455#
$50005  000714R         448    452#
$50006  000732R         482#   490
$50007  000744R         484    491#
$50010  000770R         506#   541
$50011  001054R         508    542#
$50012  001014R         514    520#
$50013  001052R         519    539#
$50014  001052R         527    537#
$50015  001144R         594    598#
$50016  001162R         618#   758
$50017  001476R         620    759#
$50020  001212R         624    629#
$50021  001220R         628    633#
$50022  001260R         650    654#
$50023  001310R         673    678#
$50024  001316R         677    682#
$50025  001362R         687    708#
$50026  001346R         692    697#
$50027  001350R         696    701#
$50030  001370R         707    712#
$50031  001416R         722    726#
$50032  001464R         732    750#
$50033  001452R         735    741#
$50034  001462R         740    747#
$50035  001474R         749    756#
$50036  001552R         797#   842
$50037  001664R         799    843#
$50040  001634R         819    828#
$50041  001652R         827    836#
$50042  001720R         860    864#
$50043  002170R         975#
$50044  002170R         976#
$50045  002116R         952#   960
$50046  002136R         954    961#
$50047  002224R         997    999#
$50050  002236R        1004   1007#
$50051  002246R        1003   1012#
$50052  002256R        1002   1017#
$50053  002266R        1001   1023#
$50054  002266R        1000   1022#
$50055  002276R         993    995   1028#
$50056  002302R        1011   1016   1021   1027   1030#
$50057  002412R        1104#
$50060  002412R        1105#
$50061  002352R        1052   1072#
$50062  002326R        1058#  1066
```



```
$50063  002344R        1060   1067#
$50064  002360R        1071   1076#
$50065  002374R        1089#  1099
$50066  002412R        1091   1100#
$50067  002500R        1119   1142#
$50070  002542R        1141   1161#
$50071  002524R        1147   1151#
$50072  002542R        1154   1158#
$50073  002430R        1177   1201#
$50074  002672R        1200   1220#
$50075  002654R        1206   1210#
$50076  002672R        1213   1217#
$50077  003010R        1300#
$50100  003010R        1301#
$50101  002722R        1248   1253#
$50102  002726R        1252   1257#
$50103  002732R        1262#  1296
$50104  003010R        1264   1297#
$50105  002774R        1267   1288#
$50106  003350R        1443#
$50107  003350R        1444#
$50110  003260R        1317   1400#
$50111  003074R        1333#  1355
$50112  003114R        1341#  1349
$50113  003136R        1343   1345   1350#
$50114  003170R        1365#  1393
$50115  003252R        1367   1394#
$50116  003204R        1371#  1386
$50117  003240R        1373   1375   1387#
$50120  003340R        1399   1434#
$50121  003274R        1407#  1431
$50122  003340R        1409   1432#
$50123  003320R        1412   1417#
$50124  003326R        1416   1421#
$50125  003420R        1490#
$50126  003420R        1491#
$50127  003366R        1465   1469#
$50130  003414R        1473   1485#
$50131  003414R        1476   1483#
$50132  003450R        1525#  1533
$50133  003474R        1527   1534#
$50134  003554R        1627#
$50135  003654R        1628#
$50136  003630R        1572   1606#
$50137  003666R        1576#  1602
$50140  003626R        1578   1603#
$50141  003620R        1585   1598#
$50142  003646R        1605   1622#
$50143  003646R        1609   1620#
.      = 004774R         255#   256#   391    392    393    394    428    429    444    445    448    449    484
                         485    508    509    514    515    527    528    594    595    620    621    624    625
                         650    651    673    674    687    688    692    693    722    723    732    733    735
                         736    799    800    819    820    860    861    954    955    993    994    995    996
                        1052   1053   1060   1061   1091   1092   1119   1120   1147   1148   1154   1155   1177
                        1178   1206   1207   1213   1214   1248   1249   1264   1265   1267   1268   1317   1318
                        1343   1344   1345   1346   1355   1356   1367   1368   1373   1374   1375   1376   1409
```

```
                1410    1412    1413    1465    1466    1473    1474    1476    1477    1527    1528    1572    1573
                1578    1579    1585    1586    1609    1610    1703#   1706#
```



```
ALLOCA    234#
BEGIN     234#
BKMOD       1#
BREAK       1#    1381    1425
BTOD        1#
CALL      234#     515     609     657     660     727     806    1436    1439
CASE      234#    1006    1010    1015
CKDATA      1#
DATACK      1#
DATERR      1#    1281
DEALLO    234#     975#   1104#   1300#   1443#   1490#   1627#
DECLAR    234#
DECR      234#
DECRB     234#
DECRU     234#
DECRUB    234#
DEFAUL    234#    1020
DFSEVN      1#     228
DSEVNT      1#     228
ELSE      231#     518     627     676     695     706     739     748     826    1070    1140    1199    1251    1398    1415
          1604
END         1#     234#
ENDDEC    234#
ENDDO     234#     402     453     489     540     757     841     959    1065    1098    1295    1348    1385    1392    1430
          1532    1601
ENDIF     234#     435     451     536     538     597     632     653     681     700     711     725     746     755     835
           863    1075    1150    1157    1160    1209    1216    1219    1256    1287    1420    1433    1468    1482    1484
          1597    1619    1621
ENDINC    234#
ENDIT       1#     673
ENDLOO    234#
ENDMOD      1#     432     532
ENDROU    234#
ENDRTI    234#
ENDRTN    234#     974    1103    1299    1442    1489    1626
ENDSEC    234#    1026
EQUATS      1#     228
EXIF      234#
EXIFB     234#
EXIT        1#
GETPA       1#     556
GWBUFF      1#     573
HRDER       1#    1591    1613
IF        234#     426     525     592     622     648     671     685     690     720     730     733     817     858    1050
          1117    1145    1152    1175    1204    1211    1246    1265    1315    1410    1463    1471    1474    1570    1583
          1607
IFB       234#
IFCOND    234#     447     513
IF.ERR    234#
IF.NO.    234#
INCR      234#
INCRB     234#
INCRU     234#
INCRUB    234#
INLINE    234#     299     311     317     323     329     335     341     347     418     439     509     590     595     644
           646     698     715     717     767     781     786     815     822     831     853     861     879    1031    1061
```

```
        1135    1163    1194    1222    1362    1466    1505    1507    1509    1528    1546    1586    1659
IOMOD     1#
IOMODP    1#
IOMODR    1#
IOMODX    1#    156
IROUTI  234#
LEAVE   234#
LET     234#    297     309     315     321     327     333     339     345     380     382     384     386     394     396
         398    400     406     408     410     420     422     424     445     449     466     477     479     485     487
         493    495     497     499     511     521     523     559     562     565     567     569     583     585     587
         599    607     625     630     636     638     640     651     674     679     688     693     702     704     709
         723    737     742     744     751     753     761     763     765     779     783     789     791     793     801
         803    809     811     813     820     824     829     833     837     839     848     851     856     903     905
         907    910     913     916     919     922     925     929     931     933     935     937     939     941     944
         947    949     955     957     962     965     967     969     971    1008    1013    1018    1024    1047    1053
        1055   1063    1068    1073    1082    1084    1086    1092    1094    1096    1120    1122    1129    1132    1137
        1143   1148    1155    1178    1180    1189    1191    1197    1202    1207    1214    1236    1244    1249    1254
        1258   1268    1270    1272    1274    1276    1289    1291    1293    1313    1319    1321    1323    1325    1328
        1334   1336    1338    1346    1351    1357    1360    1368    1376    1388    1390    1395    1401    1403    1413
        1418   1486    1502    1519    1521    1530    1542    1544    1559    1561    1563    1565    1567    1573    1579
        1581   1599    1623    1638    1640    1643    1646    1649    1652    1655
LOCAL   234#
LOOP    234#
MAP22     1#   1126    1185
MODULE    1#    157
MSG       1#
MSGN      1#    431     531    1479    1538
MSGS      1#
NBKMOD    1#
OTOA      1#
PIRQ      1#
POP     234#
PUSH    234#
RAND      1#
REPEAT  234#   1332
RETURN  234#
ROUTIN  234#    899    1044    1233    1310    1454    1556
SAVR14  234#
SBKMOD    1#
SELECT  234#    990
SOFER     1#
STRUCT  232#    233
UNTIL   234#   1353
UNTILH  234#
WHILE   234#    388     441     481     505     617     796     951    1057    1088    1261    1340    1364    1370    1406
        1524   1575
WHILEH  234#
$ADDDN  389#   390#    392#    394#    403#    427#    429#    442#    443#    445#    448#    449#    454#    482#    483#
         485#   490#    506#    507#    509#    514#    515#    519#    521#    526#    528#    541#    593#    595#    618#
         619#   621#    623#    625#    628#    630#    649#    651#    672#    674#    677#    679#    686#    688#    691#
         693#   696#    698#    707#    709#    721#    723#    731#    733#    734#    736#    740#    742#    749#    751#
         758#   797#    798#    800#    818#    820#    827#    829#    842#    859#    861#    900#    901#    952#    953#
         955#   960#    991#    992#   1000#   1005#   1045#   1046#   1051#   1053#   1058#   1059#   1061#   1066#   1071#
        1073#  1089#   1090#   1092#   1099#   1118#   1120#   1141#   1143#   1146#   1148#   1153#   1155#   1176#   1178#
        1200#  1202#   1205#   1207#   1212#   1214#   1234#   1235#   1247#   1249#   1252#   1254#   1262#   1263#   1265#
        1266#  1268#   1296#   1311#   1312#   1318#   1318#   1333#   1334#   1341#   1342#   1344#   1346#   1349#   1365#
```



```
        1366#  1368#   1371#   1372#   1374#   1376#   1386#   1393#   1399#   1401#   1407#   1408#   1410#   1411#   1413#
        1416#  1418#   1431#   1455#   1456#   1464#   1466#   1472#   1474#   1475#   1477#   1525#   1526#   1528#   1533#
        1557#  1558#   1571#   1573#   1576#   1577#   1579#   1584#   1586#   1602#   1605#   1607#   1608#   1610#
$AND     390#   392#    427#    443#    483#    507#    526#    593#    619#    623#    649#    672#    686#    691#    721#
         731#   734#    798#    818#    859#    953#   1051#   1059#   1090#   1118#   1146#   1153#   1176#   1205#   1212#
        1247#  1263#   1266#   1316#   1342#   1344#   1366#   1372#   1374#   1408#   1411#   1464#   1472#   1475#   1526#
        1571#  1577#   1584#   1608#
$BRANC   391#   393#    403#    428#    444#    448#    454#    484#    490#    509#    514#    519#    527#    541#    594#
         620#   624#    628#    650#    673#    677#    687#    692#    696#    707#    722#    732#    735#    740#    749#
         758#   799#    819#    827#    842#    860#    954#    960#    993#    995#   1007#   1011#   1016#   1021#   1052#
        1060#  1066#   1071#   1091#   1099#   1119#   1141#   1147#   1154#   1177#   1200#   1206#   1213#   1248#   1252#
        1264#  1267#   1296#   1317#   1343#   1345#   1349#   1355#   1367#   1373#   1375#   1386#   1393#   1399#   1409#
        1412#  1416#   1431#   1465#   1473#   1476#   1527#   1533#   1572#   1578#   1585#   1602#   1605#   1609#
$CALL    516#   610#    658#    661#    728#    807#   1437#   1440#
$CALL1   516#   610#    658#    661#    728#    807#   1437#   1440#
$CALL2   516#   610#    658#    661#    728#    807#   1437#   1440#
$CALL3   516#   610#    658#    661#    728#    807#   1437#   1440#
$CHECK   389#   390     427#    442#    443     482#    483     506#    507     526#    593#    618#    619     623#    649#
         672#   686#    691#    721#    731#    734#    797#    798     818#    859#    952#    953     1051#   1058#   1059
        1089#  1090     1118#   1146#   1153#   1176#   1205#   1212#   1247#   1262#   1263    1266#   1316#   1341#   1342
        1365#  1366     1371#   1372    1407#   1408    1411#   1464#   1472#   1475#   1525#   1526    1571#   1576#   1577
        1584#  1608#
$CKBAS   991#
$CKCAL   516#   610#    658#    661#    728#    807#   1437#   1440#
$CKOP1   381#   383#    385#    407#    409#    421#    423#    425#    467#    478#    480#    486#    494#    496#    498#
         500#   560#    563#    566#    568#    570#    584#    586#    588#    600#    608#    626#    631#    637#    641#
         675#   680#    689#    694#    703#    705#    710#    738#    743#    752#    764#    780#    784#    790#    792#
         794#   802#    804#    810#    814#    821#    825#    830#    834#    849#    852#    904#    906#    930#    932#
         934#   936#    938#    940#    942#    945#    948#    950#    963#    968#   1009#   1014#   1019#   1025#   1048#
        1054#  1056#   1074#   1083#   1085#   1087#   1093#   1121#   1123#   1130#   1144#   1179#   1181#   1190#   1203#
        1237#  1245#   1250#   1255#   1269#   1271#   1273#   1275#   1277#   1314#   1320#   1322#   1324#   1335#   1361#
        1391#  1396#   1404#   1414#   1487#   1520#   1522#   1560#   1562#   1564#   1566#   1568#   1574#   1580#   1624#
        1639#
$CKOP2   298#   310#    316#    322#    328#    334#    340#    346#    387#    395#    397#    399#    401#    411#    446#
         450#   488#    512#    522#    524#    639#    652#    724#    745#    754#    762#    766#    812#    838#    840#
         857#   906#    911#    914#    917#    920#    923#    926#    956#    958#    966#    970#    972#   1064#   1069#
        1095#  1097#   1133#   1138#   1149#   1156#   1192#   1198#   1208#   1215#   1259#   1290#   1292#   1294#   1326#
        1329#  1337#   1339#   1347#   1352#   1358#   1369#   1377#   1389#   1402#   1419#   1503#   1531#   1543#   1545#
        1582#  1600#   1641#   1644#   1647#   1650#   1653#   1656#
$CKSEL   991#
$CMND    389#   390     392     427#    442#    443     482#    483     506#    507     526#    593#    618#    619     623#
         649#   672#    686#    691#    721#    731#    734#    797#    798     818#    859#    952#    953     1051#   1058#
        1059   1089#   1090     1118#   1146#   1153#   1176#   1205#   1212#   1247#   1262#   1263    1266#   1316#   1341#
        1342   1344    1365#   1366    1371#   1372    1374    1407#   1408    1411#   1464#   1472#   1475#   1525#   1526
        1571#  1576#   1577    1584#   1608#
$COMPA   390#   427#    443#    448#    483#    507#    514#    526#    593#    619#    623#    649#    672#    686#    691#
         721#   731#    734#    798#    818#    859#    953#   1051#   1059#   1090#   1118#   1146#   1153#   1176#   1205#
        1212#  1247#   1263#   1266#   1316#   1342#   1366#   1372#   1408#   1411#   1464#   1472#   1475#   1526#   1571#
        1577#  1584#   1608#
$COUNT   516#   610#    658#    661#    728#    807#   1437#   1440#
$DO      390#   392#    427#    443#    483#    507#    526#    593#    619#    623#    649#    672#    686#    691#    721#
         731#   734#    798#    818#    859#    953#   1051#   1059#   1090#   1118#   1146#   1153#   1176#   1205#   1212#
        1247#  1263#   1266#   1316#   1342#   1344#   1366#   1372#   1374#   1408#   1411#   1464#   1472#   1475#   1526#
        1571#  1577#   1584#   1608#
$ELSE    519#   626#    677#    696#    707#    740#    749#    827#   1071#   1141#   1200#   1252#   1399#   1416#   1605#
$ERRMS   298#   310#    316#    322#    328#    334#    340#    346#    381#    383#    385#    387#    389#    390#    391#
```

```
            392#  393#  395#  397#  399#  401#  403#  404#  407#  409#  411#  421#  423#  425#  427#
            428#  436#  442#  443#  444#  446#  448#  450#  452#  454#  455#  467#  478#  480#  482#
            483#  484#  486#  488#  490#  491#  494#  496#  498#  500#  506#  507#  508#  512#  514#
            519#  520#  522#  524#  526#  527#  537#  539#  541#  542#  560#  563#  566#  568#  570#
            584#  586#  588#  593#  594#  598#  600#  608#  618#  619#  620#  623#  624#  626#  628#
            629#  631#  633#  637#  639#  641#  649#  650#  652#  654#  672#  673#  675#  677#  678#
            680#  682#  686#  687#  689#  691#  692#  694#  696#  697#  701#  703#  705#  707#  708#
            710#  712#  721#  722#  724#  726#  731#  732#  734#  735#  738#  740#  741#  743#  745#
            747#  749#  750#  752#  754#  756#  758#  759#  762#  764#  766#  780#  784#  790#  792#
            794#  797#  798#  799#  802#  804#  810#  812#  814#  818#  819#  821#  825#  827#  828#
            830#  834#  836#  838#  840#  842#  843#  849#  852#  857#  859#  860#  864#  904#  906#
            908#  909#  911#  912#  914#  915#  917#  918#  920#  921#  923#  924#  926#  927#  930#
            932#  934#  936#  938#  940#  942#  945#  948#  950#  952#  953#  954#  956#  958#  960#
            961#  963#  966#  968#  970#  972#  975#  991#  992#  993#  994#  995# 1007# 1009# 1011#
           1012# 1014# 1016# 1017# 1019# 1021# 1022# 1023# 1024# 1025# 1027# 1030# 1031# 1048# 1051#
           1052# 1054# 1056# 1058# 1059# 1060# 1064# 1066# 1067# 1069# 1071# 1072# 1074# 1076# 1083#
           1085# 1087# 1089# 1090# 1091# 1093# 1095# 1097# 1099# 1100# 1104# 1118# 1119# 1121# 1123#
           1130# 1133# 1138# 1141# 1142# 1144# 1146# 1147# 1149# 1151# 1153# 1154# 1156# 1158# 1161#
           1176# 1177# 1179# 1181# 1190# 1192# 1198# 1200# 1201# 1203# 1205# 1206# 1208# 1210# 1212#
           1213# 1215# 1217# 1220# 1237# 1245# 1247# 1248# 1250# 1252# 1253# 1255# 1257# 1259# 1262#
           1263# 1264# 1266# 1267# 1269# 1271# 1273# 1275# 1277# 1288# 1290# 1292# 1294# 1296# 1297#
           1300# 1314# 1316# 1317# 1320# 1322# 1324# 1326# 1327# 1329# 1330# 1335# 1337# 1339# 1341#
           1342# 1343# 1344# 1345# 1347# 1349# 1350# 1352# 1354# 1355# 1358# 1361# 1365# 1366# 1367#
           1369# 1371# 1372# 1373# 1374# 1375# 1377# 1386# 1387# 1389# 1391# 1393# 1394# 1396# 1399#
           1400# 1402# 1404# 1407# 1408# 1409# 1411# 1412# 1414# 1416# 1417# 1419# 1421# 1431# 1432#
           1434# 1443# 1464# 1465# 1469# 1472# 1473# 1475# 1476# 1483# 1485# 1487# 1490# 1503# 1520#
           1522# 1525# 1526# 1527# 1531# 1533# 1534# 1543# 1545# 1560# 1562# 1564# 1566# 1568# 1571#
           1572# 1574# 1576# 1577# 1578# 1580# 1582# 1584# 1585# 1598# 1600# 1602# 1603# 1605# 1606#
           1608# 1609# 1620# 1622# 1624# 1627# 1639# 1641# 1642# 1644# 1645# 1647# 1648# 1650# 1651#
           1653# 1654# 1656# 1657#
$FRAME      516#  610#  658#  661#  728#  807# 1437# 1440#
$GENHR      391#  393#  403#  428#  444#  448#  454#  484#  490#  508#  514#  519#  527#  541#  594#
            620#  624#  628#  650#  673#  677#  687#  692#  696#  707#  722#  732#  735#  740#  749#
            758#  799#  819#  827#  842#  860#  954#  960#  993#  995# 1011# 1016# 1021# 1027# 1052#
           1060# 1066# 1071# 1091# 1099# 1119# 1141# 1147# 1154# 1177# 1200# 1206# 1213# 1248# 1252#
           1264# 1267# 1296# 1317# 1343# 1345# 1349# 1355# 1367# 1373# 1375# 1386# 1393# 1399# 1409#
           1412# 1416# 1431# 1465# 1473# 1476# 1527# 1533# 1572# 1578# 1585# 1602# 1605# 1609#
$GENTA      389#  390#  392#  403#  404  427#  436#  442#  443#  452#  454#  455  482#  483#  490#
             491  506#  507#  519#  520  526#  537#  539#  541#  542  593#  598#  618#  619#  623#
             628#  629  633#  649#  654#  672#  677#  678  682#  686#  691#  696#  697  701#  707#
             708  712#  721#  726#  731#  734#  740#  741  747#  749#  750  756#  758#  759  797#
             798#  818#  827#  828  836#  842#  843  859#  864#  952#  953#  960#  961  975#  976
             991#  999 1007# 1011# 1012 1016# 1017 1022# 1023# 1024# 1027# 1028 1030 1051# 1058#
            1059# 1066# 1067 1071# 1072 1076# 1089# 1090# 1099# 1100 1104# 1105 1118# 1141# 1142
            1146# 1151# 1153# 1158# 1161# 1176# 1200# 1201 1205# 1210# 1212# 1217# 1220# 1247# 1252#
            1253 1257# 1262# 1263# 1266# 1288# 1296# 1297 1300# 1301 1316# 1333# 1341# 1342# 1344#
            1349# 1350 1365# 1366# 1371# 1372# 1374# 1386# 1387 1393# 1394 1399# 1400 1407# 1408#
            1411# 1416# 1417 1421# 1431# 1432 1434# 1443# 1444 1464# 1469# 1472# 1475# 1483# 1485#
            1490# 1491 1525# 1526# 1533# 1534 1571# 1576# 1577# 1584# 1598# 1602# 1603 1605# 1606
            1608# 1620# 1622# 1627# 1628
$IF         427#  526#  593#  623#  649#  672#  686#  691#  721#  731#  734#  818#  859# 1051# 1118#
            1146# 1153# 1176# 1205# 1212# 1247# 1266# 1316# 1411# 1464# 1472# 1475# 1571# 1584# 1608#
$IFBRJ      403#  454#  490#  541#  758#  842#  960# 1066# 1099# 1296# 1349# 1386# 1393# 1431# 1533#
           1602#
$IFCOD      390#  392#  427#  443#  483#  507#  526#  593#  619#  623#  649#  672#  686#  691#  721#
            731#  734#  798#  818#  859#  953#  991#  994# 1051# 1059# 1090# 1118# 1146# 1153# 1176#
```



```
           1205# 1212# 1247# 1263# 1266# 1316# 1342# 1344# 1354# 1366# 1372# 1374# 1408# 1411# 1464#
           1472# 1475# 1526# 1571# 1577# 1584# 1608#
$IFCON      448#  514#
$IFDEF      390   391   392#  393   427#  428   443#  444   448   483#  484   507#  508   514   526#
             527   593#  594   619#  620   623#  624   649#  650   672#  673   686#  687   691#  692
             721#  722   731#  732   734#  735   798#  799   818#  819   859#  860   953#  954   991#
             993   994#  995  1051#  1052  1059#  1060  1090#  1091  1118#  1119  1146#  1147  1153#  1154
            1176#  1177  1205#  1206  1212#  1213  1247#  1248  1263#  1264  1266#  1267  1316#  1317  1342#
            1343   1344#  1345  1354#  1355  1366#  1367  1372#  1373  1374#  1375  1408#  1409  1411#  1412
            1464#  1465  1472#  1473  1475#  1476  1526#  1527  1571#  1572  1577#  1578  1584#  1585  1608#
            1609
$IFOPR      391#  393#  428#  444#  448#  484#  508#  514#  527#  594#  620#  624#  650#  673#  687#
            692#  722#  732#  735#  799#  819#  860#  954#  993#  995# 1052# 1060# 1091# 1119# 1147#
           1154# 1177# 1206# 1213# 1248# 1264# 1267# 1317# 1343# 1345# 1355# 1367# 1373# 1375# 1409#
           1412# 1465# 1473# 1476# 1527# 1572# 1578# 1585# 1609#
$INHRJ      391#  393#  428#  444#  448#  484#  508#  514#  527#  594#  620#  624#  650#  673#  687#
            692#  722#  732#  735#  799#  819#  860#  954#  993#  995# 1052# 1060# 1091# 1119# 1147#
           1154# 1177# 1206# 1213# 1248# 1264# 1267# 1317# 1343# 1345# 1355# 1367# 1373# 1375# 1409#
           1412# 1465# 1473# 1476# 1527# 1572# 1578# 1585# 1609#
$LET        298#  310#  316#  322#  328#  334#  340#  346#  361#  383#  385#  387#  395#  397#  399#
            401#  407#  409#  411#  421#  423#  425#  446#  450#  467#  476#  480#  486#  488#  494#
            496#  498#  500#  512#  522#  524#  560#  563#  566#  568#  570#  584#  586#  588#  600#
            608#  626#  631#  637#  639#  641#  652#  675#  680#  689#  694#  703#  705#  710#  724#
            738#  743#  745#  752#  754#  762#  764#  766#  780#  784#  790#  792#  794#  802#  804#
            810#  812#  814#  821#  825#  830#  836#  838#  840#  849#  852#  857#  904#  906#  908#
            911#  914#  917#  920#  923#  926#  930#  932#  934#  936#  938#  940#  942#  945#  948#
            950#  956#  958#  963#  966#  968#  970#  972# 1009# 1011# 1019# 1025# 1048# 1054# 1056#
           1064# 1069# 1074# 1083# 1085# 1087# 1093# 1095# 1097# 1121# 1123# 1130# 1133# 1138# 1144#
           1149# 1156# 1179# 1181# 1190# 1192# 1198# 1203# 1208# 1215# 1237# 1245# 1250# 1255# 1259#
           1269# 1271# 1273# 1275# 1277# 1290# 1292# 1294# 1314# 1320# 1322# 1324# 1326# 1329# 1335#
           1337# 1339# 1347# 1352# 1358# 1361# 1369# 1377# 1389# 1391# 1396# 1402# 1404# 1414# 1419#
           1487# 1503# 1520# 1522# 1531# 1543# 1545# 1560# 1562# 1564# 1566# 1568# 1574# 1580# 1582#
           1600# 1624# 1639# 1641# 1644# 1647# 1650# 1653# 1656#
$MCHIG      234#
$MCLOW      234#
$OPADD      298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
            450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
            857#  909#  912#  915#  918#  921#  924#  927#  956#  958#  966#  970#  972# 1064# 1069#
           1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1327#
           1330# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
           1582# 1600# 1642# 1645# 1648# 1651# 1654# 1657#
$OPAND      298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
            450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
            857#  909#  912#  915#  918#  921#  924#  927#  956#  958#  966#  970#  972# 1064# 1069#
           1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1327#
           1330# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
           1582# 1600# 1642# 1645# 1648# 1651# 1654# 1657#
$OPCD2      298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
            450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
            857#  908#  909  911#  912  914#  915  917#  918  920#  921  923#  924  926#  927
            956#  958#  966#  970#  972# 1064# 1069# 1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198#
           1208# 1215# 1259# 1290# 1292# 1294# 1326# 1327 1329# 1330 1337# 1339# 1347# 1352# 1358#
           1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545# 1582# 1600# 1641# 1642 1644# 1645
           1647# 1648  1650# 1651  1653# 1654  1656# 1657
$OPDEF      298#  300#  310#  312#  316#  318#  322#  324#  328#  330#  334#  336#  340#  342#  346#
            348#  381#  383#  385#  387#  390#  391#  392#  393#  395#  397#  399#  401#  403#  407#
```

```
          409#  411#  419#  421#  423#  425#  427#  428#  440#  443#  444#  446#  448#  450#  454#
          467#  478#  480#  483#  484#  486#  488#  490#  494#  496#  498#  500#  507#  508#  510#
          512#  514#  516#  519#  522#  524#  526#  527#  541#  560#  563#  566#  568#  570#  584#
          586#  588#  591#  593#  594#  596#  600#  608#  610#  619#  620#  623#  624#  626#  628#
          631#  637#  639#  641#  645#  647#  649#  650#  652#  658#  661#  672#  673#  675#  677#
          680#  686#  687#  689#  691#  692#  694#  696#  699#  703#  705#  707#  710#  716#  718#
          721#  722#  724#  728#  731#  732#  734#  735#  738#  740#  743#  745#  749#  752#  754#
          758#  762#  764#  766#  768#  780#  782#  784#  787#  790#  792#  794#  798#  799#  802#
          804#  807#  810#  812#  814#  816#  818#  819#  821#  823#  825#  827#  830#  832#  834#
          838#  840#  842#  849#  852#  854#  857#  859#  860#  862#  880#  904#  906#  908#  909#
          911#  912#  914#  915#  917#  918#  920#  921#  923#  924#  926#  927#  930#  932#  934#
          936#  938#  940#  942#  945#  948#  950#  953#  954#  956#  958#  960#  963#  966#  968#
          970#  972#  975#  977   991#  992#  993#  994#  995#  996   997   998  1000# 1001# 1002#
         1003# 1004# 1009# 1011# 1014# 1016# 1019# 1021# 1025# 1027# 1029  1032# 1048# 1051# 1052#
         1054# 1056# 1059# 1060# 1062# 1064# 1066# 1069# 1071# 1074# 1083# 1085# 1087# 1090# 1091#
         1093# 1095# 1097# 1099# 1104# 1106  1118# 1119# 1121# 1123# 1130# 1133# 1134  1136# 1138#
         1141# 1144# 1146# 1147# 1149# 1153# 1154# 1156# 1164# 1176# 1177# 1179# 1181# 1190# 1192#
         1193  1195# 1198# 1200# 1203# 1205# 1206# 1208# 1212# 1213# 1215# 1223# 1237# 1245# 1247#
         1248# 1250# 1252# 1255# 1259# 1263# 1264# 1266# 1267# 1269# 1271# 1273# 1275# 1277# 1290#
         1292# 1294# 1296# 1300# 1302  1314# 1316# 1317# 1320# 1322# 1324# 1326# 1327# 1329# 1330#
         1335# 1337# 1339# 1342# 1343# 1344# 1345# 1347# 1349# 1352# 1354# 1355# 1358# 1361# 1363#
         1366# 1367# 1369# 1372# 1373# 1374# 1375# 1377# 1386# 1389# 1391# 1393# 1396# 1399# 1402#
         1404# 1408# 1409# 1411# 1412# 1414# 1416# 1419# 1431# 1437# 1440# 1443# 1445  1464# 1465#
         1467# 1472# 1473# 1475# 1476# 1487# 1490# 1492  1503# 1506# 1508# 1510# 1520# 1522# 1526#
         1527# 1529# 1531# 1533# 1543# 1545# 1547# 1560# 1562# 1564# 1566# 1568# 1571# 1572# 1574#
         1577# 1578# 1580# 1582# 1584# 1585# 1587# 1600# 1602# 1605# 1608# 1609# 1624# 1627# 1629
         1639# 1641# 1642# 1644# 1645# 1647# 1648# 1650# 1651# 1653# 1654# 1656# 1657# 1660#
$OPEQU    298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
          450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
          857#  909#  912#  915#  918#  921#  924#  927#  956#  958#  966#  970#  972# 1064# 1069#
         1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1327#
         1330# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
         1582# 1600# 1642# 1645# 1648# 1651# 1654# 1657#
$OPEXO    298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
          450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
          857#  908#  911#  914#  917#  920#  923#  926#  956#  958#  966#  970#  972# 1064# 1069#
         1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1326#
         1329# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
         1582# 1600# 1641# 1644# 1647# 1650# 1653# 1656#
$OPIDB    298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
          450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
          857#  909#  912#  915#  918#  921#  924#  927#  956#  958#  966#  970#  972# 1064# 1069#
         1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1327#
         1330# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
         1582# 1600# 1642# 1645# 1648# 1651# 1654# 1657#
$OPNAN    298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
          450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
          857#  909#  912#  915#  918#  921#  924#  927#  956#  958#  966#  970#  972# 1064# 1069#
         1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1327#
         1330# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
         1582# 1600# 1642# 1645# 1648# 1651# 1654# 1657#
$OPNOR    298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
          450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
          857#  909#  912#  915#  918#  921#  924#  927#  956#  958#  966#  970#  972# 1064# 1069#
         1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1327#
         1330# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
```



```
         1582# 1600# 1642# 1645# 1648# 1651# 1654# 1657#
$OPNOT    298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
          450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
          857#  909#  912#  915#  918#  921#  924#  927#  956#  958#  966#  970#  972# 1064# 1069#
         1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1327#
         1330# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
         1582# 1600# 1642# 1645# 1648# 1651# 1654# 1657#
$OPOR     298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
          450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
          857#  909#  912#  915#  918#  921#  924#  927#  956#  958#  966#  970#  972# 1064# 1069#
         1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1327#
         1330# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
         1582# 1600# 1642# 1645# 1648# 1651# 1654# 1657#
$OPROT    298#  310#  316#  322#  328#  334#  340#  346#  387#  395#  397#  399#  401#  411#  446#
          450#  488#  512#  522#  524#  639#  652#  724#  745#  754#  762#  766#  812#  838#  840#
          857#  909#  912#  915#  918#  921#  924#  927#  956#  958#  966#  970#  972# 1064# 1069#
         1095# 1097# 1133# 1138# 1149# 1156# 1192# 1198# 1208# 1215# 1259# 1290# 1292# 1294# 1327#
         1330# 1337# 1339# 1347# 1352# 1358# 1369# 1377# 1389# 1402# 1419# 1503# 1531# 1543# 1545#
         1582# 1600# 1642# 1645# 1648# 1651# 1654# 1657#
$OPR0     298#  310#  316#  322#  328#  334#  340#  346#  381#  383#  385#  387#  395#  397#  399#
          401#  407#  409#  411#  421#  423#  425#  446#  450#  467#  478#  480#  486#  488#  494#
          496#  498#  500#  512#  522#  524#  560#  563#  566#  568#  570#  584#  586#  588#  600#
          608#  626#  631#  637#  639#  641#  652#  675#  680#  689#  694#  703#  705#  710#  724#
          738#  743#  745#  752#  754#  762#  764#  766#  780#  784#  790#  792#  794#  802#  804#
          810#  812#  814#  821#  825#  830#  834#  838#  840#  849#  852#  857#  904#  906#  908#
          911#  914#  917#  920#  923#  926#  930#  932#  934#  936#  938#  940#  942#  945#  948#
          950#  956#  958#  963#  966#  968#  970#  972# 1009# 1014# 1019# 1025# 1048# 1054# 1056#
         1064# 1069# 1074# 1083# 1085# 1087# 1093# 1095# 1097# 1121# 1123# 1130# 1133# 1138# 1144#
         1149# 1156# 1179# 1181# 1190# 1192# 1198# 1203# 1208# 1215# 1237# 1245# 1250# 1255# 1259#
         1269# 1271# 1273# 1275# 1277# 1290# 1292# 1294# 1314# 1320# 1322# 1324# 1326# 1329# 1335#
         1337# 1339# 1347# 1352# 1358# 1361# 1369# 1377# 1389# 1391# 1396# 1402# 1404# 1414# 1419#
         1487# 1503# 1520# 1522# 1531# 1543# 1545# 1560# 1562# 1564# 1566# 1568# 1574# 1580# 1582#
         1600# 1624# 1639# 1641# 1644# 1647# 1650# 1653# 1656#
$OPR1     298#  310#  316#  322#  328#  334#  340#  346#  381#  383#  385#  387#  395#  397#  399#
          401#  407#  409#  411#  421#  423#  425#  446#  450#  467#  478#  480#  486#  488#  494#
          496#  498#  500#  512#  522#  524#  560#  563#  566#  568#  570#  584#  586#  588#  600#
          608#  626#  631#  637#  639#  641#  652#  675#  680#  689#  694#  703#  705#  710#  724#
          738#  743#  745#  752#  754#  762#  764#  766#  780#  784#  790#  792#  794#  802#  804#
          810#  812#  814#  821#  825#  830#  834#  838#  840#  849#  852#  857#  904#  906#  908#
          911#  914#  917#  920#  923#  926#  930#  932#  934#  936#  938#  940#  942#  945#  948#
          950#  956#  958#  963#  966#  968#  970#  972# 1009# 1014# 1019# 1025# 1048# 1054# 1056#
         1064# 1069# 1074# 1083# 1085# 1087# 1093# 1095# 1097# 1121# 1123# 1130# 1133# 1138# 1144#
         1149# 1156# 1179# 1181# 1190# 1192# 1198# 1203# 1208# 1215# 1237# 1245# 1250# 1255# 1259#
         1269# 1271# 1273# 1275# 1277# 1290# 1292# 1294# 1314# 1320# 1322# 1324# 1326# 1329# 1335#
         1337# 1339# 1347# 1352# 1358# 1361# 1369# 1377# 1389# 1391# 1396# 1402# 1404# 1414# 1419#
         1487# 1503# 1520# 1522# 1531# 1543# 1545# 1560# 1562# 1564# 1566# 1568# 1574# 1580# 1582#
         1600# 1624# 1639# 1641# 1644# 1647# 1650# 1653# 1656#
$OPR2     298#  310#  316#  322#  328#  334#  340#  346#  381#  383#  385#  387#  395#  397#  399#
          401#  407#  409#  411#  421#  423#  425#  446#  450#  467#  478#  480#  486#  488#  494#
          496#  498#  500#  512#  522#  524#  560#  563#  566#  568#  570#  584#  586#  588#  600#
          608#  626#  631#  637#  639#  641#  652#  675#  680#  689#  694#  703#  705#  710#  724#
          738#  743#  745#  752#  754#  762#  764#  766#  780#  784#  790#  792#  794#  802#  804#
          810#  812#  814#  821#  825#  830#  834#  838#  840#  849#  852#  857#  904#  906#  908#
          911#  914#  917#  920#  923#  926#  930#  932#  934#  936#  938#  940#  942#  945#  948#
          950#  956#  958#  963#  966#  968#  970#  972# 1009# 1014# 1019# 1025# 1048# 1054# 1056#
         1064# 1069# 1074# 1083# 1085# 1087# 1093# 1095# 1097# 1121# 1123# 1130# 1133# 1138# 1144#
```

```
        1024#   1046#   1051#   1053#   1058#   1059#   1061#   1066#   1073#   1089#   1090#   1092#   1099#   1118#   1120#
        1143#   1146#   1148#   1153#   1155#   1176#   1178#   1202#   1205#   1207#   1212#   1214#   1235#   1247#   1249#
        1254#   1262#   1263#   1265#   1266#   1268#   1296#   1312#   1316#   1318#   1333#   1334#   1341#   1342#   1346#
        1349#   1365#   1366#   1368#   1371#   1372#   1376#   1386#   1393#   1401#   1407#   1408#   1410#   1411#   1413#
        1418#   1431#   1456#   1464#   1466#   1472#   1474#   1475#   1477#   1525#   1526#   1528#   1533#   1558#   1571#
        1573#   1576#   1577#   1579#   1584#   1586#   1602#   1607#   1608#   1610#
$$SETT   991#   1007#   1008    1011#   1013    1016#   1018    1022#   1023#   1024#   1027#

. ABS.  000000      000.
        004774      001


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  1

 CXBTCA,CXBTCA.PRT/SOL/CRF=SPMACJ/ML,DDXCOM.P11,CXBTCA.P11,CXBTCA.END
 RUN-TIME:  48  41  4 SECONDS
 RUN-TIME RATIO:  132/94=1.4
 CORE USED:  24K  (48 PAGES)
```

```
.ENABL LC

                                                .REM &
```

IDENTIFICATION

PRODUCT CODE: AC-S914C-MC

PRODUCT NAME: CXDUBC0 UDA DEC/X MOD

PRODUCT DATE: 01-JUL-82

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: MATT TEDONE

TABLE OF CONTENTS

1.0 ABSTRACT

THE EXERCISER WILL BE SIMILAR TO THAT OF OTHER DISK SUBSYSTEM EXERCISERS. WRITES WILL BE PERFORMED TO THE DISKS FOLLOWED BY READ AND COMPARE OF THE DATA READ. THE UDA WILL DO ALL ERROR RETRYING. ERRORS WILL BE REPORTED ON THE CONSOLE TTY.

ALL DESIRED DISK DRIVES ON THE UDA WILL BE EXERCISED SIMULTANEOUSLY. IF DISK ACCESSING IS NOT REQUIRED, THEN DATA WRITTEN WILL GO ONLY AS FAR AS THE CONTROLLER'S RAM MEMORY.

THIS DEC/X11 MODULE WILL BE RELEASED UNDER THE NAME CXDUBB0 BEFORE FCS OF THE UDA-50 AND THE RA80. A DEC/X11-PLUS MODULE UNDER THE NAME CSDUBB0 WILL BE RELEASED AFTER BEING MODIFIED BY THE DIAGNOSTIC SYSTEMS ENGINEERING GROUP TO RUN UNDER THE DEC/X11-PLUS MONITOR.

IF THE RESULTS OF THE EXERCISER REQUIRES MORE INFORMATION, TWO OTHER PDP-11 DIAGNOSTIC PROGRAMS ARE AVAILABLE. THEY ARE:

CZUDEA0 - UDA AND DISK DIAGNOSTIC
CZUDFA0 - UDA DISK FORMATTER.

2.0 REQUIREMENTS

HARDWARE FOR ALL CASES:
ONE DEC/X11 MODULE CONFIGURED FOR ONE UDA CONTROLLER.

HARDWARE FOR DISK ACCESSING:
ONE UDA CONTROLLER WITH AT LEAST ONE DRIVE IS THE MINIMUM AMOUNT OR ONE UDA CONTROLLER WITH FOUR DRIVES IS THE MAXIMUM AMOUNT.

HARDWARE FOR NO DISK ACCESSING:
ONE UDA CONTROLLER IS THE ONLY REQUIREMENT.

STORAGE: DUBB REQUIRES

DECIMAL WORDS -- 4096 MAX

3.0 START-UP

ON THE INITIAL START, THE PROGRAM WILL CLEAR BIT1 OF 'SR1' AND TYPE THE FOLLOWING MESSAGES.

```
DUBB0 PA:0060162      APC: 000674      PASS #00000

'IF YOU WISH TO DESTROY CUSTOMER DATA, SET BIT1 (NOT BIT0)
 IN SWITCH REGISTER 1(SR1) OF DUBB? EQUAL TO 1.'

DUBB0 PA:0060210      APC: 000722      PASS #00000
'! OPERATING WITH NO DISK ACCESSING !'
```

THIS WILL OCCUR REGUARDLESS OF THE CONDITION OF SR1 (BIT1) AT CONFIGURE TIME.

IF THE OPERATOR WISHES TO EXERCISE THE UNIT, SR1 (BIT1) MUST BE MODIFIED AT LOCATION 16 OF CXDUBB0 MODULE (SEE SECTION 9). THIS CAN BE ACCOMPLISHED BY USING THE 'MOD' COMMAND SUPPLIED BY THE DECX11 RUN TIME SYSTEM. UNLESS THE PROGRAM IS RELOADED OR THE OPERATOR MODIFIES THE LOCATION AGAIN, THE CONTENTS OF SR1 WILL REMAIN THE SAME ON ALL SUBSEQUENT STARTS.

ON ALL SUBSEQUENT STARTS, THE CONDITION OF SR1 (BIT1) WILL TYPE TO TERMINAL IN THE FOLLOWING MANOR.

IF BIT1 OF SR1 IS EQUAL TO 0 (ZERO), THE FOLLOWING WARNING WILL BE TYPED.

```
DUBB0 PA:0060210      APC: 000722      PASS #00000
'! OPERATING WITH NO DISK ACCESSING !'
```

IF BIT1 OF SR1 IS EQUAL TO 1 (ONE), THE FOLLOWING WARNING WILL BE TYPED.


```
DUBB0 PA:0060210      APC: 000722      PASS #00000
'! CUSTOMER DATA WILL BE DESTROYED !'
```

<<<< NOTE >>>>
WHEN THIS DEC/X11 MODULE RUNS IN DISKLESS MODE, ITS DATA RATE EXCEEDS ALL OTHER DEVICES. THIS MAY CAUSE ERRONEOUS DATA LATES FROM OTHER DEVICES.

4.0 PASS DEFINITION

ONE PASS OF THE DUBB MODULE CONSISTS OF 512 ITERATIONS OF THE BASIC TEST SEQUENCE (WRITE, READ, DATA-CHECK). THE TEST SEQUENCE WRITES A USER DEFINED NUMBER OF WORDS (DEFAULT IS 256) WORDS, READS 256 WORDS, AND DATA-COMPARE SAME.

5.0 EXECUTION TIME

THE DEFAULT EXECUTION TIME OF ONE PASS OF DUBB RUNNING ALONE ON A PDP-11/44 UNDER SEQUENTIAL DISK ACCESSING MODE WILL BE APPROXIMATELY 20 SECONDS. UNDER RANDOM ACCESSING MODE, THE TIME IS 40 SECONDS. FOR NO DISK ACCESSING, THE TIME IS FIVE SECONDS

6.0 CONFIGURATION REQUIREMENTS

DEFAULT PARAMETERS:

DEVADR: 172150, VECTOR: 154, SR1: 5, DEVCNT: 1, SR1: 0, SR2: 0

REQUIRED PARAMETERS:

ADDITIONAL UDA MODULE(S) CONFIGURED MUST HAVE A DIFFERENT UNIBUS ADDRESS(ES) AND VECTOR(S).

7.0 DEVICE/OPTION SETUP

FOR DISK MODE, MAKE CERTAIN THAT ALL UNITS ARE POWERED UP, WRITE ENABLED, CONNECTED TO A UDA VIA THE SDI AND READY.

FOR DISKLESS MODE, MAKE CERTAIN THE UDA IS POWERED UP.

8.0 MODULE OPERATION

TEST SEQUENCE DISK MODE:

A. SETUP DEVICE REGISTER ADDRESSES AND MODULE VARIABLES.
SET CONTROLLER CHARACTERISTIC.
B. RESET ALL UNITS ON-LINE AND DROP ALL THAT ARE NOT.
C. GET A UNIT ADDRESS.
D. GET A DISK ADDRESS AND A FRESH BLOCK OF DATA.
E. DO A WRITE -- IF ERRORS, REPORT.
F. DO A READ -- IF ERRORS, REPORT.
G. DO A DATA-CHECK -- IF ERRORS, REPORT AND CONTINUE.
H. MAKE UNIT AVAILABLE.
I. WAIT FOR AVAILABLE ATTENTION MESSAGE.
J. IF END OF PASS, REPORT AND GO TO D.
K. IF END OF TESTING UNIT, GO TO C; ELSE GO TO D.

BLOCKS DETERMINED DEFECTIVE WON'T BE REPLACED BY THE EXERCISER DURING THIS SEQUENCE. THE EXERCISER MAKES FULL USE OF THE UDA WHICH DOES REVECTORING ON ITS OWN.

TEST SEQUENCE DISKLESS MODE:

A. GET A FRESH BLOCK OF DATA.
B. DO A WRITE TO UDA RAM BUFFER -- IF ERRORS, REPORT.
C. DO A READ FROM UDA RAM BUFFER -- IF ERRORS,REPORT.
D. DO A DATA-CHECK -- IF ERRORS, REPORT AND CONTINUE.
E. IF END OF PASS, REPORT.
F. GO TO A.

9.0 OPERATION OPTIONS

ONE OR MORE SOFTWARE SWITCH REGISTERS CAN BE USED BY THE MODULE PROGRAM GENERAL PURPOSE SWITCHES. THESE WORDS ARE USED TO DEFINE OR SPECIFY A UNIQUE DEVICE OPTION OR TO POINT TO A SPECIFIC ROUTINE IN THE MODULE. ANY OPTION MUST BE SPECIFIED BY THE OPERATOR BEFORE THE MODULE IS RUN. SWITCH REGISTER 1 HAS THE FOLLOWING CHARACTERISTICS.

SR1 BIT 1 SET (1):		ALLOW DISK TRANSFERS.
<<<< NOTE >>>> IF SET, CUSTOMER DATA WILL BE DESTROYED!	RESET (0):	NO DISK TRANSFERS.
SR1 BIT 2 SET (1):		DO NOT REPORT ERRORS AS THEY OCCUR.
	RESET (0):	REPORT ERRORS AS THEY OCCUR.
SR1 BIT 3 SET (1):		DO NOT PRINT ERROR SUMMARY AT END OF PASS.
	RESET (0):	PRINT ERROR SUMMARY AT END OF PASS.
SR1 BIT 9 SET (1):		RUN DUAL PORT MODE (ONLY VALID IF SR1 BIT 1 IS SET)
	RESET (0):	DO NOT RUN DUAL PORT MODE
SR1 BIT 10 SET (1):		SELECT RANDOM BLOCK ADDRESSING.(ONLY VALID IF SR1 BIT 1 IS SET)

	RESET (0):	SELECT SEQUENTIAL BLOCK ADDRESSING.
SR1 BIT 11 SET (1):		BYPASS DATA COMPARE.
	RESET (0):	DO DATA COMAPARE.

SWITCH REGISTER 2 HAS THE FOLLOWING CHARACTERISTICS.

SR2 BITS 0 TO 5: BURST RATE.

A BURST RATE TO SPEED UP NPR TRANSFERS BY THE UDA CAN BE USED. THIS VALUE IS 6 BITS MAXIMUM AND SET UP IN SR2 AT CONFIGURE TIME.

<<<< NOTE >>>>
THE DVIDI MASK REFLECTS THE NUMBER OF UNITS CHOSEN FOR TESTING AND WHICH UNITS ON THE SYSTEM ARE TO BE TESTED. EXAMPLE: IF DVIDI CONTAINS A 1, ONLY THE FIRST UNIT FOUND ON THE SYSTEM WILL BE TESTED. A UNIT'S ORDER ON THE SYSTEM IS JUDGED BY ITS UNIT NUMBER. THE LOWEST UNIT NUMBER ZERO (0). UNIT 0 WOULD BE THE FIRST TESTED ON THE SYSTEM.

IF DVIDI CONTAINS A 10, THE FOURTH UNIT ON THE SYSTEM WILL BE TESTED. IF THE FIRST TWO UNITS ARE CHOSEN, DVIDI IS 3. FOUR CONSECUTIVE UNITS MEANS DVIDI IS 17. SIX UNITS, DVIDI IS 77.

IF THERE IS NOT A UNIT CORRESPONDING TO THE DVIDI BIT SETTING, THE BIT SET IN DVIDI GETS CLEARED. THE EXERCISER WILL READJUST THE MASK AND DROP THE NONEXISTENT UNITS IF MORE UNITS ARE CHOSEN THAN ACTUALLY ARE PRESENT. THE MODULE IS DROPPED IF ALL DVIDI BITS ARE CLEARED.

IF THE NUMBER OF UNITS CHOSEN IS LESS THEN THE ACTUAL NUMBER OF UNITS PRESENT, ONLY THE DESIRED UNITS WILL BE USED DURING THE EXERCISE.

<<<< ANOTHER NOTE >>>>
MAKE SURE ALL SUBUNITED DRIVES ARE ACCOUNTED FOR. DESTROYING CUSTOMER DATA IS NOT DESIRABLE.

<<<< ONE MORE NOTE >>>>
IF SR1 BIT 3 IS RESET, A SUMMARY STATUS IS PRINTED EVERY 15 PASSES. THIS STATUS IS FORMATTED AS FOLLOWS:

DUBB0 PA: 00060470 ACP: 001210 PASS #00000

SOFT ERROR COUNT #00000 *** HARD ERRO COUNT #00000
CHECK DATA ERROR COUNT #00000

10.0 PRINTOUTS

A. MOST PRINTOUTS HAVE THE STANDARD FORMATS DESCRIBED IN THE DEC/X11 DOCUMENT.

B. NON-STANDARD PRINTOUTS INCLUDE ERROR MESSAGES WHICH DUMP THE FOLLOWING:

1) SUMMARY STATUS
2) FLAGS AND ENDCODE
3) UNIT NUMBER
4) BYTE COUNT
5) HI 16-BIT LBN VALUE
6) LO 16-BIT LBN VALUE
7) EXTENDED ADDRESS
8) PHYSICAL ADDRESS

ALL VALUES EXCEPT FOR PASS, RUNTIME AND ERRCNT ARE PRINTED IN OCTAL. PASS, RUNTIME AND ERRCNT ARE PRINTED IN DECIMAL.
EXAMPLE:

```
DUBB0 PA: 0006411b APC: 004b30 PASS: 00000 ERRCNT: 00001
CSRA: 172150 CSRC: 000000 ASTAT: 000006 ERRTYP: 000006
RUNTIME: 000:00:22

DUBB0 PA: 00064052 APC: 004564 PASS: 00000

STATUS ENDCOD UNITNU BYTECO HI LBN LO LBN EXTADR PHYADR
000006 000242 000005 000000 000003 116321 000001 062100
```

STATUS - RESPONCE OF THE COMMAND SENT TO THE UDA. THIS IS CONTAINED IN THE LAST FIVE BITS OF THE WORD. HERE IS A LIST OF STATUS CODES.

0 - SUCCESS
1 - INVALID COMMAND
2 - COMMAND ABORTED
3 - UNIT OFFLINE
4 - UNIT AVAILABLE
5 - MEDIA ERROR
6 - WRITE PROTECTED
7 - COMPARE ERROR
10 - DATA ERROR
11 - HOST BUFFER ACCESS ERROR
12 - CONTROLLER ERROR
13 - DRIVE ERROR

ENDCOD - ENDING CODE OF THE COMMAND SENT. THIS SHOWS WHAT COMMAND WAS SENT TO THE UDA. HERE IS A LIST OF ALL POSSIBLE ENDCODES THIS MODULE USES.

100 - AVAILABLE ATTENTION MESSAGE (NOT A COMMAND BUT A MESSAGE SENT TO THE HOST FROM THE UDA)
200 - INVALID COMMAND
203 - GET UNIT STATUS
204 - SET CONTROLLER CHARACTERISTICS
210 - AVAILABLE

211 - ONLINE
230 - MAINTENANCE READ
231 - MAINTENANCE WRITE
241 - READ
242 - WRITE

UNITNU - UNIT NUMBER OF THE DRIVE THAT IS BEING ACCESSED. THIS IS NOT RELEVENT IF THE USER IS RUNNING DISKLESS MODE.

BYTECO - SIZE OF THE BUFFER IN BYTES.

HI LBN - HIGH LOGICAL BLOCK NUMBER (UPPER 16 BITS) WHICH TELLS THE USER WHERE ON THE DISK THE DATA IS GOING. THIS IS ONLY VALID FOR DISK MODE.

LO LBN - LOW LOGICAL BLOCK NUMBER (LOWER 16 BITS).

EXTADR - EXTENDED ADDRESS OF THE READ/WRITE BUFFER.

PHYADR - PHYSICAL ADDRESS OF THE READ/WRITE BUFFER.

C. IF THE UDA FAILED TO PASS ITS INTERNAL DIAGNOSTIC, ONE OF THE FOLLOWING MESSAGES WILL BE PRINTED.

IF THE DIAGNOSTIC FOUND A FAULT:

```
DUBB0 PA: 00062052 APC: 002564 PASS: 00000
UDA INIT ERROR, FOUND BY DIAGNOSTIC
UDASA = XXXXXX IN STEP YYYYY
ADDR = ZZZZZZ
```

IF A STEP BIT WAS NOT SET AS EXPECTED DURING THE INITIALIZATION SEQUENCE OF THE UDA:

```
DUBB0 PA: 00062152 APC: 002664 PASS: 00000
UDA INIT ERROR, STEP NOT SET
UDASA = XXXXXX IN STEP YYYYY
ADDR = ZZZZZZ
```

IF DATA PASSED BACK FROM THE UDA WAS NOT EQUAL TO THE EXPECTED VALUE:

```
DUBB0 PA: 00062252 APC: 002764 PASS: 00000
UDA INIT ERROR, EXPECTED DATA WAS INCORRECT
UDASA = XXXXXX IN STEP YYYYY
ADDR = ZZZZZZ
```

WHERE XXXXXX CAN HAVE ANY OF THE FOLLOWING VALUES AND MEANINGS:

```
    104000 - FATAL SEQUNCER ERROR
    104040 - D PROCESSOR ALU ERROR
    104041 - D PROC ROM PARITY ERROR
    105102 - D PROC WITH NO BOARD #2 OR RAM PARITY ERROR
    105105 - D PROC RAM BUFFER ERROR
    105152 - D PROC SDI ERROR
    105153 - D PROC WRITE MODE WRAP SERDES ERROR
```

```
        105154 - D PROC READ MODE SERDES, RSGER, AND ECC ERROR
        106040 - U PROC ALU ERROR
        106041 - U PROC CONTROL REGISTER ERROR
        106042 - U PROC DFAIL/ROM PARITY ERROR/BOARD #1 TEST COUNT IS WRONG
        106047 - U PROC CONSTANT ROM ERROR WITH D PROC RUNNING SDI TEST
        106055 - UNEXPECTANT TRAP FOUND, ABORTED DIAGNOSTIC
        106071 - U PROC ROM ERROR
        106072 - U PROC ROM PARITY ERROR
        106200 - STEP 1 DATA ERROR (MSB NOT SET)
        107143 - U PROC RAM PARITY ERROR
        107107 - U PROC RAM BUFFER ERROR
        107115 - BOARD #2 TEST COUNT WAS WRONG
        112300 - STEP 2 ERROR
        122240 - NPR ERROR
        122300 - STEP 3 ERROR
        142300 - STEP 4 ERROR

    WHERE YYYYY IS THE STEP IN WHICH THE ERROR WAS FOUND.

    WHERE ZZZZZZ IS THE ADDRESS OF THE UDA.

    IF THE MAXIMMUM NUMBER OF RETRIES HAS BEEN EXCEEDED, THE
    FOLLOWNING MESSAGE WILL BE PRINTED.

     DUBBO PA: 00061414 APC:002126 PASS #00000

     RETRY COUNT EXCEEDED, ABORT

    THIS MEANS THE UDA DID NOT SUCCESSFULLY COMPLETE THE
    INITIALIZATION IN FOUR PASSES. THE MODULE IS THEN DROPPED.


    D. IF THE UDA DID NOT SUCCESSFULLY CLEAR THE RING BUFFER IN
    THE HOST AREA, THE FOLLOWING MESSAGE WILL BE PRINTED.

     DUBBO PA: 00061414 APC:002126 PASS #00000

     RING AREA NOT CLEARED

    THIS IS A FATAL ERROR. IT MEANS THAT THE UDA DID NOT ACCESS
    HOST MEMORY THAT THE UDA WOULD USE TO COMMUNICATE WITH THE
    HOST. THE MODULE IS THEN DROPPED.


    E. IF THE UDASA DISPLAYS A NON-ZERO VALUE AFTER THE
    INITIALIZATION SEQUENCE IS DONE, THE FOLLOWNG MESSAGE WILL
    BE PRINTED.

     DUBBO PA: 00064252 APC: 004764 PASS: 00000
     UDASA IS NOT ZERO, = XXXXXX
     UDA IS GOING THROUGH INITIALIZATION

    WHERE XXXXXX CAN HAVE THE FOLLOWING VALUES AND MEANINGS.

        004400 - UDA HAS BEEN INITED BY EITHER A BUS INIT OR BY
                 WRITING INTO THE UDAIP.
        100001 - UNIBUS ENVELOPE/PACKET READ ERROR (PARITY OR TIMEOUT)
```



```
        100002 - UNIBUS ENVELOPE/PACKET WRITE ERROR (PARITY OR TIMEOUT)
        100003 - UDA ROM AND RAM PARITY ERROR
        100004 - UDA RAM PARITY ERROR
        100005 - UDA ROM PARITY ERROR
        100006 - UNIBUS RING READ ERROR
        100007 - UNIBUS RING WRITE ERROR
        100010 - UNIBUS INTERRUPT MASTER FAILURE
        100011 - HOST ACCESS TIMEOUT ERROR
        100012 - HOST EXCEEDED CREDIT LIMIT
        100013 - UDA SDI HARDWARE FATAL ERROR
        100014 - DM XFC FATAL ERROR
        100015 - HARDWARE TIMEOUT OF INSTRUCTION LOOP
        100016 - INVALID VIRTUAL CIRCUIT IDENTIFIER
        100017 - INTERRUPT WRITE ERROR ON UNIBUS


    E. IF A DRIVE IS DROPPED BY THE EXERCISER, ONE OF THE
    FOLLOWING MESSAGES WILL BE PRINTED.

        IF THE DRIVE HAD AN ERROR IT COULD NOT HANDLE PROPERLY
    AFTER AN ITERATION, THE FOLLOWING MESSAGE WILL BE PRINTED:

     DUBBO PA: 00063012 APC: 003524 PASS #00000

     DRIVE 00000 DROPPED.
     DEVICE ID BIT = 000001
     ERRORS CAUSED DRIVE TO BE DROPPED

        IF THE DRIVE WAS NOT FOUND BY THE EXERCISER, THE
    FOLLOWING MESSAGE WILL BE PRINTED:

     DUBBO PA: 00063012 APC: 003524 PASS #00000

     DRIVE 00000 DROPPED.
     DEVICE ID BIT = 000001
     UNIT WAS NOT FOUND BY THE EXERCISER

        IF THERE WERE MORE DEVICE COUNT BITS SET THAN THE ACTUAL
    NUMBER OF DRIVES FOUND, THE FOLLOWING MESSAGE WILL BE
    PRINTED:

     DUBBO PA: 00063012 APC: 003524 PASS #00000

     DRIVE 00000 DROPPED.
     DEVICE ID BIT = 000001
     DVIDI BIT SET HIGHER THAN ACTUAL # OF DRIVES FOUND

    SOLUTION: TRY A LESSER NUMBER OF UNITS IN DVIDI (LOC 14)


11.0  DUAL PORT OPERATION

    TO RUN A DUAL PORT OPERATION, SET BIT9 OF SR1. THE
    EXERCISER WILL CHECK THE UNIT TO SEE IF IT IS OFFLINE OR
    AVAILABLE.
```

THE UDA WILL RETAIN CONTROL OF A UNIT UNTIL THE MSCP AVAILABLE COMMAND IS ENTERED BY THE HOST. DURING THIS TIME, THE OTHER CONTROLLER IS NOT ALLOWED ACCESS TO THE UNIT THROUGH THE OTHER PORT BETWEEN THE WRITE AND READ. THE OTHER CONTROLLER SENSES WHEN THE UNIT BECOMES AVAILABLE AND TAKES IT. THE MSCP AVAILABLE COMMAND IS ONLY EXECUTED IF SA1 BIT 9 AND SA1 BIT 1 ARE SET. THIS ALLOWS DUAL PORTING AND DISK ACCESSING RESPECTIVELY.

DEC/X11 WILL ONLY DUAL PORT A DRIVE WITH ANOTHER DEC/X11 EXERCISER.

12.0 GLOSSARY

DUBB FOLLOWS THE MODULE NAME FORMAT DESCRIBED IN THE DEC/X11 PROGRAMMER'S GUIDE.

DD--	IDENTIFIES THE HARDWARE AND THUS THE MODULE.
--B-	DISTIGUISHES BETWEEN TWO OR MORE DIFFERENT MODULES FOR THE SAME GENERIC DEVICE. THE SEQUENCE A, B, C, ETC. MUST BE USED FOR EACH ADDITIONAL EXAMPLE.
---B	SPECIFIES THE MODULE REVISION.

IOMODX IS A TYPE OF MODULE IN AN EXTENDED INPUT/OUTPUT MODE. THESE MODULES ARE INTERRUPT DRIVEN AND ARE CAPABLE OF INPUT/OUTPUT OPERATION. SOME ADDED CAPABILITIES PROVIDED INCLUDE:

- USE OF MONITOR SUPPLIED WRITE BUFFERS,
- ABILITY TO CHANGE THE SIZE OF THE WRITE BUFFERS,
- ACCESS TO THE MONITOR'S CHECK DATA UTILITY,
- CONVERSION ROUTINES TO GET 18 BIT ADDRESSES FROM 16 BIT ADDRESSES.

13.0 BIBLIOGRAPHY

CXQUAA0 'DEC/X11 USER'S MANUAL' SEPT 1978

CXQAPD0 'DEC/X11 PROGRAMMERS'S GUIDE' SEPT 1978


```
2
3                               .DSABL LC
```

```
                                .SBTTL  MODULE HEADER BLOCK
                                        IDMODX <DUBC  >,172150,154,5,0,0,1000,104,RBUF,256.,256.
                                        MODULE  150000,DUBC  ,172150,154,5,0,0,1000,104,RBUF,256.,256.
                                        .TITLE  DUBC DEC/X11 SYSTEM EXERCISER MODULE
                                ;       DDXCOM  VERSION  6       23-MAY-78
                                        .LIST   MIN
                                ;*****************************************************************************
000000                          BEGIN:
000000  104     125     102     MODNAM: .ASCII  /DUBC / ;MODULE NAME.
000003  103     040
000005  000                     XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF RBUFF USAGE
000006  172150                  AUDR:   172150+0                ;1ST DEVICE ADDR.
000010  000154                  VECTOR: 154+0                   ;1ST DEVICE VECTOR.
000012  240                     BR1:    .BYTE   PRTY5+0         ;1ST RX LEVEL.
000013  000                     BR2:    .BYTE   PRTY0+0         ;2ND RX LEVEL.
000014  000001                  DVID1:  0+1                     ;DEVICE INDICATOR 1.
000016  000000                  SR1:    OPEN                    ;SWITCH REGISTER 1
000020  000000                  SR2:    OPEN                    ;SWITCH REGISTER 2
000022  000000                  SR3:    OPEN                    ;SWITCH REGISTER 3
000024  000000                  SR4:    OPEN                    ;SWITCH REGISTER 4
                                ;*****************************************************************************
000026  150000                  STAT:   150000                  ;STATUS WORD.
000030  000660'                 INIT:   START                   ;MODULE START ADDR.
000032  000252'                 SPOINT: MODSP                   ;MODULE STACK POINTER.
000034  000000                  PASCNT: 0                       ;PASS COUNTER.
000036  001000                  ICONT:  1000                    ;# OF ITERATIONS PER PASS=1000
000040  000000                  ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
000042  000000                  SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
000044  000000                  HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
000046  000000                  SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
000050  000000                  HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
000052  000000                  SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULAIED
000054  000000                  RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000056                          CONFIG:                         ;RESERVED FOR MONITOR USE
000056  000000                  RES1:   0                       ;RESERVED FOR MONITOR USE
000060  000000                  RES2:   0                       ;RESERVED FOR MONITOR USE
000062  000000                  SVR0:   OPEN                    ;LOC TO SAVE R0.
000064  000000                  SVR1:   OPEN                    ;LOC TO SAVE R1.
000066  000000                  SVR2:   OPEN                    ;LOC TO SAVE R2.
000070  000000                  SVR3:   OPEN                    ;LOC TO SAVE R3.
000072  000000                  SVR4:   OPEN                    ;LOC TO SAVE R4.
000074  000000                  SVR5:   OPEN                    ;LOC TO SAVE R5.
000076  000000                  SVR6:   OPEN                    ;LOC TO SAVE R6.
000100  000000                  CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
000102                          SRADR:                          ;ADDR OF GOOD DATA, OR
000102  000000                  ACSR:   OPEN                    ;CONTENTS OF CSR.
000104                          WASADR:                         ;ADDR OF BAD DATA, OR
000104  000000                  ASTAT:  OPEN                    ;STATUS REG CONTENTS.
000106                          ERRTYP:                         ;TYPE OF ERROR
000106  000000                  ASR:    OPEN                    ;EXPECTED DATA.
000110  000000                  AWAS:   OPEN                    ;ACTUAL DATA.
000112  001036'                 RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
000114  000000                  WDIO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
000116  000000                  WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
000120  000000                  INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
000122  000104                  IDNUM:  104                     ;MODULE IDENTIFICATION NUMBER=104
```



```
000124  007676'                 RBUFVA: RBUF                    ;READ BUFFER VIRTUAL ADDRESS
000126  000000                  RBUFPA: OPEN                    ;READ BUFFER PHYSICAL ADDRESS
000130  000000                  RBUFEA: OPEN                    ;READ BUFFER EA BITS
000132  000400                  RBUFSZ: 256.                    ;SIZE OF THE READ BUFFER
000134  000000                  WBUFPA: OPEN                    ;WRITE BUFFER PHYSICAL ADDRESS
000136  000000                  WBUFEA: OPEN                    ;WRITE BUFFER EA BITS
000140  000400                  WBUFRQ: 256.                    ;WRITE BUFFER SIZE REQUESTED
000142  000000                  WBUFSZ: OPEN                    ;WRITE BUFFER SIZE AVAILABLE
000144  000000                  CDERCT: OPEN                    ;CDATA/DATCK ERROR COUNT
000146  000000                  CDWDCT: OPEN                    ;CDATA/DATCK WORD COUNT
000150  000000                  FREE:   OPEN                    ;RESERVED FOR FUTURE USE
        000040                          .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000252                          MODSP:
                                ;*****************************************************************************
                                ;*****************************************************************************
                                .SBTTL  MODULE STORAGE AREA
                                ;       VERSION 1.0     FOR RELEASE
                                ;       VERSION 1.1     NO LONGER TEST AFTER STEP 4
                                ;                       NO LONGER WAIT FOR INTERRUPT AFTER SENDING MSCP AVAILABLE COMMAND
                                ;       VERSION 2.0     USE BIT 9 IN SR1 FOR DUAL PORTING. (DON'T SEND MSCP AVAILABLE
                                ;                       COMMAND IF WE WANT JUST SEQUENTIAL OR RANDOM ACCESS MODE --
                                ;                       IN OTHER WORDS, ONLY SEND ONLINE COMMAND ONCE DURING PASS UNLESS
                                ;                       DUAL PORT MODE)

        000002                  SR.XFR = BIT01  ;NO DISK TRANSFER 0 = NO DISK TRANSFER, 1 = DO DISK TRANSFER
        000004                  SR.REP = BIT02  ;REPORT ERROR AS THEY OCCUR 0 = REPORT, 1 = DON'T REPORT
        000010                  SR.SUM = BIT03  ;REPORT ERRORS ON END OF PASS 0 = REPORT, 1 = DON'T REPORT
        001000                  SR.DUA = BIT09  ;DUAL PORT 0 = NO DUAL PORT, 1 = DUAL PORT
        002000                  SR.SEQ = BIT10  ;RANDOM (NOT SEQUENTIAL) DISK ADDRESSING 0 = SEQUENTIAL, 1 = RANDOM
        004000                  SR.CMP = BIT11  ;NO DATA COMPARE 0 = DO DATA COMPARE, 1 = DON'T DO DATA COMPARE

                                ;UDAIP: .WORD   0        ;UDA POLLING REG
000252  000000                  UDASA:  .WORD   0        ;UDA STATUS REG

000254  000000                  CINTR:  .WORD   0               ;COMMAND INTERRUPT INDICATOR
000256  000000                  RINTR:  .WORD   0               ;RESPONCE INTERRUPT INDICATOR

000260                          RSPQNC: .BLKW   2.              ;MESSAGE RING
000264                          COMMND: .BLKW   2.              ;COMMAND RING

000270  000000                  CMDREF: .WORD   0               ;COMMAND REFERENCE NUMBER

000272  000000                  RSPLEN: .WORD   0               ;RESPONCE PACKET LENGTH
000274  000000                  RSPVIR: .WORD   0               ;RESPONCE PACKET VIRTUAL CIRCUIT
000276                          RSPACK: .BLKW   24.             ;RESPONCE PACKET

000356  000000                  CMPLEN: .WORD   0               ;COMMAND PACKET LENGTH
000360  000000                  CMPVIR: .WORD   0               ;COMMAND PACKET VIRTUAL CIRCUIT
000362                          CMPACK: .BLKW   24.             ;COMMAND PACKET

000442  002b64'                 VA:     .WORD   COMMND          ;GENERIC VIRTUAL ADDRESS FOR GETPA
000444  000000                  PA:     .WORD   OPEN            ;GENERIC PHYSICAL ADDRESS
000446  000000                  EA:     .WORD   OPEN            ;GENERIC EXTENDED ADDRESS
```

```
000450  000000             RBFFEA: .WORD   0        ;READ  BUFFER EXTENDED ADDRESS SAVE AREA
000452  000000             WBFFEA: .WORD   0        ;WRITE BUFFER EXTENDED ADDRESS SAVE AREA

000454  000000             NUM:    .WORD   0                ;ADDRESS USED IN UTUA
000456  000000             OLDPA:  .WORD   0                ;THE OLD PHYSICAL ADDRESS
000460  000000             OLDEA:  .WORD   0                ;THE OLD EXTENDED ADDRESS TO CHECK IF
                                                            ;       UDA WILL BE REINITED

        000017             PRTNUM = 15.                     ;PRINT MESSAGE EVERY 15TH TIME
000462  000017             PRNMSG: .WORD   PRTNUM           ;PRINT WORD SAVES THE VALUE TO CHECK FOR WHEN
                                                            ; THE NEXT TIME AN END OF PASS MESSAGE IS WRITTEN
        002260             TIMER = 1200.                    ;TIMER VALUE TO WAIT 2-3 SECONDS AFTER DAP COMMAND

000464  177777             EXPAV:  .WORD   177777           ;EXPECTING AN AVAILABLE ATTENTION MESSAGE = 0
                                                            ;NOT EXPECTING AN AVAILABLE ATTENTION MESSAGE = 177777

000466                     ADR1:   .BLKB   6
000474     000                     .BYTE   0
000475                     ADR2:   .BLKB   6
000503     000                     .BYTE   0
000504                     ADR3:   .BLKB   6
000512     000                     .BYTE   0
000513                     ADR4:   .BLKB   6
000521     000                     .BYTE   0
000522                     ADR5:   .BLKB   6
000530     000                     .BYTE   0
000531                     ADR6:   .BLKB   6
000537     000                     .BYTE   0
000540                     ADR7:   .BLKB   6
000546     000                     .BYTE   0
000547                     ADR8:   .BLKB   6
000555     000                     .BYTE   0
                                   .EVEN
```



```
                           .SBTTL MORE MODULE STORAGE
                           ;
                           ;DO NOT CHANGE THE ORDER OF THE NEXT 4 LOCATIONS
                           ;NEEDED FOR MAP 22 ROUTINE
000556  000000             PA18:   .WORD   0
000560  000000             XMEM:   .WORD   0
000562  000000             PA22:   .WORD   0
000564  000000             EA22:   .WORD   0

000566  000000             SECL:   .WORD   0                ;CURRENT SECTOR LO ORDER ADDRESS
000570  000000             SECH:   .WORD   0                ;CURRENT SECTOR HI ORDER ADDRESS

000572  000000             UNSZL:  .WORD   0                ;UNIT SIZE LO ORDER LIMIT FROM ONLINE CMND
000574  000000             UNSZH:  .WORD   0                ;UNIT SIZE HI ORDER LIMIT

000576  003300             LIMIT:  .WORD   3300             ;4K - 1200 = MOST WORDS MAITW CAN TAKE

000600  000001             DVICE:  .WORD   1                ;DEVICE TO TEST
000602  000000             UNITNO: .WORD   0                ;UNIT NUMBER
000604  000000             TRY:    .WORD   0                ;NUMBER OF TRIES
000606  000001             PORTID: .WORD   1                ;BIT POSITION SELECTS THE PORT
000610  000000             UNITFL: .WORD   0                ;SAVE UNIT FLAGS
000612  000000             WORK:   .WORD   0                ;TEMPORARY WORK AREA

        005670             TIMOUT = 3000.                   ;TIME OUT GADGE
        000004             RLIM = 4                         ;RETRY LIMIT

000614  000000  000001     TABLEW: .WORD   0,1              ;TABLE ENTRY UNITNO,PORTID
000620  177777  177777             .WORD   -1,-1            ;CURRENT LAST TABLE ENTRY
000624                             .BLKW   12.              ;REST OF TABLE
000654  177777  177777     TEND:   .WORD   -1,-1            ;END MARKER

                           ;S:     .WORD   0,0,0,0,0,0,0,0          ;FOR HARD AND SOFT ERRORS
                           ;       .WORD   177777

                           ;TABLE: .WORD   S                ;EACH ENTRY OF THE TABLE POINTS TO
                           ;       .WORD   S+2              ;THE CORRESPONDING ENTRY OF S.
                           ;       .WORD   S+4              ;THIS IS USED IN HRDER & SOFER
                           ;       .WORD   S+6
                           ;       .WORD   S+10
                           ;       .WORD   S+12
                           ;       .WORD   S+14
                           ;       .WORD   S+16
                           ;       .WORD   177777
```

```
                                .SBTTL  MODULE PRIVATE DATA

000001                          BIT00 = 1
000002                          BIT01 = 2
000004                          BIT02 = 4
000010                          BIT03 = 10
000020                          BIT04 = 20
000040                          BIT05 = 40
000100                          BIT06 = 100
000200                          BIT07 = 200
000400                          BIT08 = 400
001000                          BIT09 = 1000
002000                          BIT10 = 2000
004000                          BIT11 = 4000
010000                          BIT12 = 10000
020000                          BIT13 = 20000
040000                          BIT14 = 40000
100000                          BIT15 = 100000

                                ;
                                ;       ERROR BITS
                                ;
000000                          ERR.0 = 0       ;NOT DEFINED
000001                          ERR.1 = 1       ;DATA ERROR
000003                          ERR.3 = 3       ;CONTROLLER NOT READY
000006                          ERR.6 = 6       ;DRIVE NOT READY, OFF LINE OR NON EXESTENT
000032                          ERR.32 = 32     ;NPR ERROR
```



```
                                .SBTTL UDA BIT DEFINITIONS

                                ;UDASA REGISTER UNIVERSAL READ BITS

004000                          SA.S1=  004000                  ;STEP 1 STATUS BIT
010000                          SA.S2=  010000                  ;STEP 2 STATUS BIT
020000                          SA.S3=  020000                  ;STEP 3 STATUS BIT
040000                          SA.S4=  040000                  ;STEP 4 STATUS BIT
100000                          SA.ERR= 100000                  ;ERROR INDICATOR

                                ;UDASA REGISTER ERROR STATUS BITS

003777                          SA.ERC= 003777                  ;ERROR CODE

                                ;UDASA REGISTER STEP ONE READ BITS

003400                          SA.CTP= 003400                  ;CONTROLLER TYPE
000400                          SA.DIA= 000400                  ;DIAG BIT IN UDASA

                                ;UDASA REGISTER STEP ONE WRITE BITS

000177                          SA.VEC= 000177                  ;INTERRUPT VECTOR (DIVIDED BY 4)
000200                          SA.INT= 000200                  ;INTERRUPT ENABLE DURING INITIALIZATION
007400                          SA.RSP= 007400                  ;RESPONCE RING LENGTH
170000                          SA.CMD= 170000                  ;COMMAND RING LENGTH

                                ;UDASA REGISTER STEP TWO READ BITS

000177                          SA.VCE= 000177                  ;INTERRUPT VECTOR ECHO
000200                          SA.INE= 000200                  ;INTERRUPT ENABLE ECHO

                                ;UDASA REGISTER STEP TWO WRITE BITS

                                                                ;LOW ORDER MESSAGE RING BYTE ADDRESS
000001                          SA.PRG= 000001                  ;ENABLE VAX UNIBUS ADAPTER PURGE INTERRUPT

                                ;UDASA REGISTER STEP THREE READ BITS

000017                          SA.RSE= 000017                  ;RESPONCE RING LENGTH ECHO
000360                          SA.CME= 000360                  ;COMMAND RING LENGTH ECHO

                                ;UDASA REGISTER STEP THREE WRITE BITS

                                                                ;HIGH ORDER MESSAGE RING BYTE ADDRESS
040000                          SA.LFC= 040000                  ;LAST FAILURE CODE REQUEST

                                ;UDASA REGISTER STEP FOUR READ BITS

000377                          SA.MCV= 000377                  ;UDA MICROCODE VERSION

                                ;UDASA REGISTER STEP FOUR WRITE BITS

000001                          SA.GO=  BIT0                    ;GO BIT TO START UDA FIRMWARE
```

```
                                .SBTTL COMMAND/MESSAGE DESCRIPTOR BIT DEFINITIONS

100000                          RG.OWN= BIT15                   ;SET WHEN UDA OWNS RING
040000                          RG.FLG= BIT14                   ;FLAG BIT

                                ;OFFSETS INTO HOST COMMUNICATIONS AREA WITH ONE DESCRIPTOR TO EACH RING

000010                          HC.SIZ= 8.                      ;SIZE OF HOST COMM AREA IN BYTES
000060                          PKTSIZ= 48.                     ;SIZE OF PACKETS IN BYTES

000000                          HC.RES= 0.                      ;RESPONCE RING START
000002                          HC.RCT= 2.                      ;RESPONCE RING CONTROL WORD
000004                          HC.CMD= 4.                      ;COMMAND RING START
000006                          HC.CCT= 6.                      ;CONTROL RING CONTROL WORD
000276'                         HC.RPK= RSPACK                  ;START OF RESPONCE PACKET BUFFER
000356'                         HC.CPK= HC.RPK+PKTSIZ           ;START OF COMMAND PACKET BUFFER
```



```
                                .SBTTL COMMAND PACKET OPCODES

000001                          OP.ABO= 01                      ;ABORT COMMAND
000020                          OP.ACC= 20                      ;ACCESS COMMAND
000010                          OP.AVL= 10                      ;AVAILABLE COMMAND
000021                          OP.CCD= 21                      ;COMPARE CONTROLLER DATA COMMAND
000040                          OP.CMP= 40                      ;COMPARE HOST DATA COMMAND
000013                          OP.DAP= 13                      ;DETERMINE ACCESS PATHS COMMAND
000022                          OP.ERS= 22                      ;ERASE COMMAND
000023                          OP.FLU= 23                      ;FLUSH COMMAND
000002                          OP.GCS= 02                      ;GET COMMAND STATUS COMMAND
000003                          OP.GUS= 03                      ;GET UNIT STATUS COMMAND
000011                          OP.ONL= 11                      ;ONLINE COMMAND
000041                          OP.RD=  41                      ;READ COMMAND
000024                          OP.RPL= 24                      ;REPLACE COMMAND
000004                          OP.SCC= 04                      ;SET CONTROLLER CHARACTERISTICS COMMAND
000012                          OP.SUC= 12                      ;SET UNIT CHARACTERISTICS COMMAND
000042                          OP.WR=  42                      ;WRITE COMMAND
000030                          OP.MRD= 30                      ;MAINTENANCE READ COMMAND
000031                          OP.MWR= 31                      ;MAINTENANCE WRITE COMMAND
000200                          OP.END= 200                     ;END PACKET FLAG
000100                          OP.AVA= 100                     ;AVAILABLE ATTENTION MESSAGE
000101                          OP.ERL= 101                     ;ERROR LOG ATTENTION MESSAGE
000102                          OP.SHC= 102                     ;SHADOW COPY COMPLETE ATTENTION MESSAGE
000102                          OP.ACP= 102                     ;ACCESS PATH ATTENTION MESSAGE

                                ;NOTE: END PACKET OPCODES (ALSO CALLED ENDCODES) ARE FORMED BY ADDING THE END
                                ;PACKET FLAG TO THE COMMAND OPCODE. THE UNKNOWN COMMAND END PACKET CONTAINS
                                ;JUST THE END PACKET FLAG IN ITS OPCODE FIELD.
```

```
                                .SBTTL COMMAND MODIFIERS

040000          MD.CMP= 040000          ;COMPARE
100000          MD.EXP= 100000          ;EXPRESS REQUEST
010000          MD.ERR= 010000          ;FORCE ERROR
004000          MD.SCH= 004000          ;SUPPRESS CACHING (HIGH SPEED)
002000          MD.SCL= 002000          ;SUPPRESS CACHING (LOW SPEED)
001000          MD.SEC= 001000          ;SUPPRESS ERROR CORRECTION
000400          MD.SER= 000400          ;SUPPRESS ERROR RECOVERY
000200          MD.SSH= 000200          ;SUPPRESS SHADOWING
000100          MD.WBN= 000100          ;WRITE-BACK (NON-VOLATILE)
000040          MD.WBV= 000040          ;WRITE BACK (VOLATILE)
000001          MD.SPD= 000001          ;SPIN-DOWN
000001          MD.FEU= 000001          ;FLUSH ENTIRE UNIT
000002          MD.VOL= 000002          ;VOLATILE ONLY
000001          MD.NXU= 000001          ;NEXT UNIT

                                .SBTTL END PACKET FLAGS

000200          EF.BBR= 000200          ;BAD BLOCK REPORTED
000100          EF.BBU= 000100          ;BAD BLOCK UNREPORTED
000040          EF.LOG= 000040          ;ERROR LOG GENERATED
000020          EF.SEX= 000020          ;SERIOUS EXCEPTION


                                .SBTTL UNIT FLAGS

000001          UF.CMR= 000001          ;COMPARE READS
000002          UF.CMW= 000002          ;COMPARE WRITES
010000          UF.RPL= 010000          ;HOST INITIATED BAD BLOCK REPLACEMENT
040000          UF.INA= 040000          ;INACTIVE SHADOW SET UNIT
000200          UF.WMV= 000200          ;REMOVEABLE MEDIA
004000          UF.SCH= 004000          ;SUPPRESS CACHING (HIGH SPEED)
002000          UF.SCL= 002000          ;SUPPRESS CACHING (LOW SPEED)
000040          UF.WBN= 000040          ;WRITE-BACK (NON-VOLATILE)
020000          UF.WPH= 020000          ;WRITE PROTECT(HARDWARE)
010000          UF.WPS= 010000          ;WRITE PROTECT(SOFTWARE OR VOLUME)
000004          UF.576= 000004          ;576 BYTE SECTORS
```



```
                                .SBTTL CONTROLLER FLAGS

000200          CF.AVL= 000200          ;ENABLE AVAILABLE ATTENTION MESSAGES
000100          CF.MSC= 000100          ;ENABLE MISCELLANEOUS ERROR LOG MESSAGES
000040          CF.OTH= 000040          ;ENABLE OTHER HOST'S ERROR LOG MESSAGES
000020          CF.THS= 000020          ;ENABLE THIS HOST'S ERROR LOG MESSAGES
000002          CF.SHD= 000002          ;SHADOWING
000001          CF.576= 000001          ;576 BYTE SECTORS

                                .SBTTL COMMAND PACKET OFFSETS

                                ;       GENERIC COMMAND PACKET OFFSETS:
000000          P.CRF=   0.             ;COMMAND REFERENCE NUMBER
000004          P.UNIT=  4.             ;UNIT NUMBER
000010          P.OPCD=  8.             ;OPCODE
000012          P.MOD=  10.             ;MODIFIERS
000014          P.BCNT= 12.             ;BYTE COUNT
000020          P.BUFF= 16.             ;BUFFER DESCRIPTOR
000020          P.ADPA= 16.             ;BUFFER'S PHYSICAL ADDRESS (P.BUFF)
000022          P.ADEA= 18.             ;BUFFER'S EXTENDED ADDRESS (P.BUFF+2)
000034          P.LBN=  28.             ;LOGICAL BLOCK NUMBER
000040          P.SFTW= 32.             ;SOFTWARE WORDS

                                ;       ABORT AND GET COMMAND STATUS COMMAND PACKET OFFSETS:
000014          P.OTRF= 12.             ;OUTSTANDING REFERENCE NUMBER

                                ;       ONLINE AND SET UNIT CHARACTERISTICS COMMAND PACKET OFFSETS:
000016          P.UNFL= 14.             ;UNIT FLAGS
000020          P.HSTI= 16.             ;HOST IDENTIFIER
000024          P.UNTI= 20.             ;UNIT IDENTIFIER
000034          P.ELGF= 28.             ;ERROR LOG FLAGS
000040          P.SHUN= 32.             ;SHADOW UNIT
000042          P.CPSP= 34.             ;COPY SPEED

                                ;       REPLACE COMMAND PACKET OFFSETS:
000014          P.RBN=  12.             ;REPLACEMENT BLOCK NUMBER

                                ;       SET CONTROLLER CHARACTERISTICS COMMAND PACKET OFFSETS:
000014          P.VRSN= 12.             ;MSCP VERSION
000016          P.CNTF= 14.             ;CONTROLLER FLAGS
000020          P.HTMO= 16.             ;HOST TIMEOUT
000022          P.USEF= 18.             ;USE FRACTION
000024          P.TIME= 20.             ;QUAD-WORD TIME AND DATE

                                ;       MAINTENANCE READ AND MAINTENANCE WRITE COMMAND PACKET OFFSETS:
000034          P.RGID= 28.             ;REGION ID
000040          P.RGOF= 32.             ;REGION OFFSET
```

```
                                .SBTTL END PACKET OFFSETS

                                ;               GENERIC END PACKET OFFSETS:
000000                          P.CRF=   0.             ;COMMAND REFERENCE NUMBER
000004                          P.UNIT=  4.             ;UNIT NUMBER
000010                          P.OPCD=  8.             ;OPCODE (ALSO CALLED ENDCODE)
000011                          P.FLGS=  9.             ;END PACKET FLAGS
000012                          P.STS=  10.             ;MODIFIERS
000014                          P.BCNT= 12.             ;BYTE COUNT
000034                          P.FBBK= 28.             ;FIRST BAD BLOCK
000040                          P.SFTW= 32.             ;SOFTWARE WORDS

                                ;               GET COMMAND STATUS END PACKET OFFSETS:
000014                          P.OTRF= 12.             ;OUTSTANDING REFERENCE NUMBER
000020                          P.CMST= 16.             ;COMMAND STATUS

                                ;               GET UNIT STATUS END PACKET OFFSETS:
000014                          P.MLUN= 12.             ;MULTI-UNIT CODE
000016                          P.UNFL= 14.             ;UNIT FLAGS
000020                          P.HSTI= 16.             ;HOST IDENTIFIER
000024                          P.UNTI= 20.             ;UNIT IDENTIFIER
000040                          P.SHUN= 32.             ;SHADOW UNIT
000042                          P.SHST= 34.             ;SHADOW STATUS
000044                          P.TRCK= 36.             ;TRACK SIZE
000046                          P.GRP=  38.             ;GROUP SIZE
000050                          P.CYL=  40.             ;CYLINDER SIZE
000054                          P.RCTS= 44.             ;RCT TABLE SIZE
000056                          P.RBNS= 46.             ;RBN / TRACK
000057                          P.RCTC= 47.             ;RCT COPIES

                                ;               ONLINE AND SET UNIT CHARACTERISTICS
000014                          P.MLUN= 12.             ;MULTI-UNIT CODE
000016                          P.UNFL= 14.             ;UNIT FLAGS
000020                          P.HSTI= 16.             ;HOST IDENTIFIER
000024                          P.UNTI= 20.             ;UNIT IDENTIFIER
000040                          P.SHUN= 32.             ;SHADOW UNIT
000044                          P.UNSZ= 36.             ;UNIT SIZE
000050                          P.VSER= 40.             ;VOLUME SERIAL NUMBER

                                ;               SET CONTROLLER CHARACTERISTICS END PACKET OFFSETS:
000014                          P.VRSN= 12.             ;MSCP VERSION
000016                          P.CNTF= 14.             ;CONTROLLER FLAGS
000020                          P.CTMO= 16.             ;CONTROLLER TIMEOUT
000022                          P.CNCL= 18.             ;CONTROLLER COMMAND LIMIT
000024                          P.CNTI= 20.             ;CONTROLLER ID
000034                          P.MEDI= 28.             ;MEDIA TYPE
000042                          P.SHST= 34.             ;SHADOW STATUS

                                ;ERROR LOG ATTENTION MESSAGE PACKET OFFSETS

000000                          P.CRF=   0.             ;COMMAND REFERENCE NUMBER
000004                          P.UNIT=  4.             ;UNIT NUMBER
000006                          P.CNT=   6.             ;COUNT
000010                          P.OPCD=  8.             ;OPCODE
000011                          P.FLGS=  9.             ;ERROR LOG FLAGS
000012                          P.SZOF= 10.             ;SIZE OR OFFSET
000014                          P.LGDT= 12.             ;START OF ERROR LOG DATA
```



```
                                .SBTTL ERROR LOG FLAGS

000200                          EF.FRS= 000200          ;FIRST PACKET
000100                          EF.LST= 000100          ;LAST PACKET
000001                          EF.MIS= 000001          ;MESSAGE MISSING

                                ;ERROR LOG MESSAGE OFFSETS

000000                          L.EVNT=  0.             ;EVENT CODE
000002                          L.SLUT=  2.             ;SLOT NUMBER
000004                          L.CNTI=  4.             ;CONTROLLER IDENTIFIER
000014                          L.CNTI= 12.             ;CONTROLLER SOFTWARE REVISION
000015                          L.CHVR= 13.             ;CONTROLLER HARDWARE REVISION
000016                          L.UNTI= 14.             ;UNIT IDENTIFIER
000026                          L.USVR= 22.             ;UNIT SOFTWARE REVISION
000027                          L.UHVR= 23.             ;UNIT HARDWARE REVISION
000030                          L.ERLC= 24.             ;ERROR LOCATION
000034                          L.CYL=  28.             ;CYLINDER
000040                          L.GRP=  32.             ;GROUP
000041                          L.TRCK= 33.             ;TRACK
000042                          L.SCTR= 34.             ;SECTOR
000044                          L.VSER= 36.             ;VOLUME SERIAL NUMBER
000050                          L.DATA= 40.             ;EVENT DEPENDENT DATA

                                ;STATUS AND EVENT CUE DEFINITIONS

000037                          ST.MSK= 37              ;STATUS / EVENT CODE MASK
000040                          ST.SUB= 40              ;SUB-CODE MULTIPLIER
000000                          ST.SUC=  0              ;SUCCESS
000001                          ST.CMD=  1              ;INVALID COMMAND
000002                          ST.ABO=  2              ;COMMAND ABORTED
000003                          ST.OFL=  3              ;UNIT-OFFLINE
000004                          ST.AVL=  4              ;UNIT-AVAILABLE
000005                          ST.MFE=  5              ;MEDIA ERROR
000006                          ST.WPR=  6              ;WRITE PROTECTED
000007                          ST.CMP=  7              ;COMPARE ERROR
000010                          ST.DAT= 10              ;DATA ERROR
000011                          ST.HST= 11              ;HOST BUFFER ACCESS ERROR
000012                          ST.CNT= 12              ;CONTROLLER ERROR
000013                          ST.DRV= 13              ;DRIVE ERROR
000037                          ST.DIA= 37              ;MESSAGE FROM AN INTERNAL DIAGNOSTIC

                                ;
                                ;       SUBCODES FOR ST.OFL
                                ;
000040                          SC.NVL =  40            ;NO VOLUME MOUNTED
                                                        ; OR DRIVE DISAVLED VIA RUN/STOP SWITCH
000100                          SC.IOP = 100            ;UNIT INOPERATIVE
000400                          SC.DIS = 400            ;UNIT DISABLED BY FIELD SERVICE
                                                        ; OR INTERNAL DIAGNOSTIC
000200                          SC.DUP = 200            ;DUPLICATE UNIT NUMBER
                                ;
                                ;       SUBCODES FOR ST.DRV
                                ;
000040                          SC.STO =  40            ;SDI RESPONCE TIME OUT
000100                          SC.INV = 100            ;INVALID SDI RESPONCE
```

```
                                .SBTTL  MODULE CODE
                                ;*****************************************************************************
                                ;                                                                            *
                                ;       INIT VALUES                                                          *
                                ;       INIT UDA                                                             *
                                ;       XFER TO DISK?                                                        *
                                ;               F FOR J = 1,CYCLE LIMIT                                      *
                                ;                 MAITENANCE WRITE                                           *
                                ;                 MAITENANCE READ                                            *
                                ;                 CHECK DATA?                                                *
                                ;                   T CHECK                                                  *
                                ;               NEXT J                                                       *
                                ;               T FOR J = 1,CYCLE LIMIT                                      *
                                ;                 GET UNIT STATUS                                            *
                                ;                       IF DRIVE IS NOT AVAILABLE, WAIT UNTIL IT IS          *
                                ;                 DRIVE THERE?                                               *
                                ;                 F   DROP                                                   *
                                ;                     ALL DRIVES DROPPED?                                    *
                                ;                       T DROP MODULE                                        *
                                ;                       F ---                                                *
                                ;                 T ONLINE                                                   *
                                ;                 ONLINE?                                                    *
                                ;                       T PICK BLOCK - IF RANDOM, GET RAND # MOD X           *
                                ;                                      ELSE INCREMENT                        *
                                ;                                           IF LBN > LIMIT THEN LBN <= 0     *
                                ;                         WRITE                                              *
                                ;                         READ                                               *
                                ;                         CHECK DATA ?                                       *
                                ;                           T CHECK                                          *
                                ;                         AVAILABLE DRIVE(I)                                 *
                                ;                       F TRY TO BRING ONLINE AGAIN                          *
                                ;               NEXT J                                                       *
                                ;*****************************************************************************


                                ;*****************************************************************************
                                ;                                                                            *
                                ;       START CODE                                                           *
                                ;                                                                            *
                                ;       IF THE CODE IS RESTARTED, CLEAR THE OLD ADDRESSES SO THE             *
                                ;       THE UDA WILL GET REINITED.                                           *
                                ;                                                                            *
                                ;*****************************************************************************

000660                          START:
000660  005227  177777                  INC     #-1                     ;FIRST TIME THRU HERE?
000664  001006                          BNE     1$                      ;BR IF NO
000666  042767  000002  177122          BIC     #SR.XFR,SR1             ;DO NOT ALLOW DISK TRANSFERS
000674  104403  000000' 005354'         MSGN$,BEGIN,WARN1       ;ASCII MESSAGE CALL WITH COMMON HEADER
000702  032767  000002  177106  1$:     BIT     #SR.XFR,SR1             ;WILL CUSTOMER DATA BE OVERWRITTEN?
000710  001404                          BEQ     2$                      ;BR IF NO
000712  104403  000000' 005360'         MSGN$,BEGIN,WARN2       ;ASCII MESSAGE CALL WITH COMMON HEADER
000720  000403                          BR      3$                      ;
000722                          2$:
000722  104403  000000' 005364'         MSGN$,BEGIN,WARN3       ;ASCII MESSAGE CALL WITH COMMON HEADER
000730                          3$:
```



```
000730  005067  177210                  CLR     CUERCT                  ;CLEAR DATA CHECK ERROR COUNT
000734  012767  177777  177522          MOV     #177777,EXPAV           ;NOT EXPECTING AN INTERRUPT
000742  012767  000017  177512          MOV     #PRINUM,PRNMSG          ;INITIALIZE PRINT #ORD
000750  016767  177040  177622          MOV     UVID1,DVICE             ;DVICE HAS DESIRED BITS SET
000756  005067  177632                  CLR     TABLE#                  ;SET TABLE FOR UNIT 0
000762  012767  000001  177626          MOV     #1,TABLE#+2             ;SET TABLE FOR PORTID FOR UNIT 0
000770  005067  177274                  CLR     CMDREF                  ;COMMAND REF # = 0
000774  104417  000000'                 RAND$,BEGIN
001000  016767  177050  177560          MOV     MANNUM,SECL
001006  005067  177556                  CLR     SECH                    ; FOR RESTARTING (INITIAL SECTOR ADDR)
001012  016767  176770  177232          MOV     ADDR,UDASA              ;STORE IN SA REG
001020  062767  000002  177224          ADD     #2,UDASA                ;UDASA HAS PROPER ADDRESS
001026  005067  177424                  CLR     OLDPA                   ;OLD PHYSICAL ADDRESS CLEARED
001032  005067  177422                  CLR     OLDEA                   ;OLD EXTENDED ADDRESS CLEARED
                                ;                FOR RESTARTING. THIS WILL FORCE A UDA REINIT TO TAKE PLACE
```

```
                                ;*****************************************************************************
                                ;                                                                           *
                                ;       RESTART SEQUENCE                                                    *
                                ;                                                                           *
                                ;       CHECK THE ADDRESS OF THE RINGS TO SEE IF THEY WERE RELOCATED        *
                                ;       IF THEY WERE, REINIT THE UDA.                                       *
                                ;                                                                           *
                                ;       GET THE NEW ADDRESSES.  IF THE DISKLESS OPERATION IS DESIRED        *
                                ;       THEN DO THE MAINTENENCE WRITE AND READ.  ELSE DO THE WRITE          *
                                ;       AND READ WITH A DRIVE.                                              *
                                ;                                                                           *
                                ;*****************************************************************************
001036                          RESTR1:
001036  012767  000260' 177376          MOV     #RSPUNC,VA              ;DID THE RINGS RELOCATE?
001044  104415  000000' 000442'          GETPAS,BEGIN, VA                ;GET PHYSICAL ADDRESS FROM 16-BIT VA
001052  026767  177366  177376          CMP     PA,OLDPA                ;IS THE OLD PHYS ADDR = NEW ONE?
001060  001004                          BNE     RESTR2                  ;IF SO, REINIT
001062  026767  177360  177370          CMP     EA,OLDEA                ;IS THE OLD EXTN ADDR = NEW ONE?
001070  001412                          BEQ     RESTR1                  ;IF NOT, DON'T REINIT
001072  016767  177346  177356  RESTR2: MOV     PA,OLDPA                ;ELSE SET THE OLD RING ADDR
001100  016767  177342  177352          MOV     EA,OLDEA                ; AND THE OLD EXTENED ADDR
001106  004767  000332                  JSR     PC,INITUD               ;AND INIT THE UDA
001112  005067  177466                  CLR     TRY                     ;CLEAR RETRY COUNT
001116                          RESTR1:
001116  032767  000010  176672          BIT     #SR.SUM,SR1             ;DO WE WANT THE REPORT?
001124  001034                          BNE     1$                      ;IF NOT, SKIP THE REPORT
001126  026767  177330  176700          CMP     PRNMSG,PASCNT           ;DO WE PRINT?
001134  001030                          BNE     1$                      ;IF PASS COUNT IS NOT = PRINT WORD, SKIP
001136  062767  000017  177316          ADD     #PRTNUM,PRNMSG          ;PRINT WORD IS INCREMENT
                                        ;*******************************************************************
                                                                        ;CONVERT SUFCNT TO ASCII AND
                                                                        ;STORE AT ADR2
001144  104421  000000' 000042'          BTODS,BEGIN,SUFCNT,ADR2
001152  000475'
                                        ;*******************************************************************
001154  105067  177322                  CLRB    ADR2+5
                                        ;*******************************************************************
                                                                        ;CONVERT HRDCNT TO ASCII AND
                                                                        ;STORE AT ADR3
001160  104421  000000' 000044'          BTODS,BEGIN,HRDCNT,ADR3
001166  000504'
                                        ;*******************************************************************
001170  105067  177315                  CLRB    ADR3+5
                                        ;*******************************************************************
                                                                        ;CONVERT CDERCT TO ASCII AND
                                                                        ;STORE AT ADR1
001174  104421  000000' 000144'          BTODS,BEGIN,CDERCT,ADR1
001202  000466'
                                        ;*******************************************************************
001204  105067  177263                  CLRB    ADR1+5
001210  104403  000000' 005316'          MSGNS,BEGIN,ERRPAS      ;ASCII MESSAGE CALL WITH COMMON HEADER
001216  012777  004336' 176564  1$:      MOV     #NINUPT,@VECTOR         ;GET VECTOR ADDRESS
                                                                        ;SET POINTER
001224  104415  000000' 000124'          GETPAS,BEGIN, RBUFVA            ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
001232  016700  176672                  MOV     RBUFEA,R0               ;GET EA TO ADJUST
001236  004767  000566                  JSR     PC,ASR04                ;GO ADJUST IT
001242  010067  177202                  MOV     R0,RBFFEA               ;PUT ADJUSTED VALUE IN A SAVE AREA
```



```
001246  005067  177330                  CLR     UNITNO                  ;PRESET UNIT #
001252  032767  000002  176536          BIT     #SR.XFER,SR1            ;DISK XFER???
001260  001454                          BEQ     MAINC                   ;NO!  DO MAINTENENCE (DISKLESS) ROUTINES
                                ;*****************************************************************************
                                ;       DO THE DISK OPERATIONS                                              *
                                ;               CHECK TO SEE WHICH PORTS ARE AVAILABLE                      *
                                ;*****************************************************************************

001262  004767  001026                  JSR     PC,SETUP                ;FIND DRIVES/SET UP TABLE
001266  005767  177306                  TST     DVICE                   ;ELSE, TEST FOR ANY MORE DRIVES
001272  001002                          BNE     LOOP1                   ;IF TRUE, DO A CYCLE

001274  104410  000000'                  ENDS,BEGIN              ;

001300                          LOOP1:
001300  104414  000000'                  GWBUFS, BEGIN           ;GET WRITE BUFFER INFORMATION
001304  016700  176626                  MOV     WBUFEA,R0               ;GET WRITE BUFFER INFORMATION
001310  004767  000514                  JSR     PC,ASR04                ;ADJUST IT
001314  010067  177132                  MOV     R0,WBFFEA               ;STORE EA IN SAVE AREA
001320  012704  000614'                  MOV     #TABLEW,R4              ;R4 -> TABLE OF UNITNO AND PORTID
001324  012703  000001                  MOV     #1,R3                   ;R3 IS AN INDEX TO DVICE
001330                          LOOP2:
001330  030367  177244                  BIT     R3,DVICE                ;HAS THE DRIVE BEEN DROPPED
001334  001412                          BEQ     9$                      ;IF SO, SKIP THIS DRIVE
001336  016467  000002  177242          MOV     2(R4),PORTID            ;SET UP PORTID
001344  011467  177232                  MOV     (R4),UNITNO             ;SET UP UNITNO
                                ; *** DO A DISK CYCLE
001350  004767  001244                  JSR     PC,CYCLED               ;DO A CYCLE FOR DISK OPERATION
001354  103002                          BCC     9$                      ;IF SUCCESSFUL, CONTINUE
001356  004767  002022                  JSR     PC,DROP1                ;IF NOT, DROP DRIVE
001362                          9$:
001362  062704  000004                  ADD     #4,R4                   ;POINT TO NEXT ENTRY OF THE TABLE
001366  006303                          ASL     R3                      ;R3 POINTS TO NEXT BIT
001370  022704  000654'                  CMP     #TEND,R4                ; POINT BEYOND LAST ENTRY?
001374  001403                          BEQ     12$                     ; IF NOT, THEN TRY AGAIN.
001376  020367  177176                  CMP     R3,DVICE                ;IF R3 > DVICE THEN DONE WITH ITERATION
001402  003752                          BLE     LOOP2                   ;IF < OR =, LOOP
001404                          12$:
001404  104413  000000'                  ENDITS,BEGIN            ;SIGNAL END OF ITERATION.
                                                                        ;MONITOR SHALL TEST END OF PASS
001410  000733                          BR      LOOP1                   ;AND DO AGAIN


                                ;*****************************************************************************
                                ;       MAINTENENCE ROUTINE, DO THE DISKLESS CODE                           *
                                ;*****************************************************************************

001412                          MAINC:
001412  104414  000000'                  GWBUFS, BEGIN           ;GET WRITE BUFFER INFORMATION
001416  016700  176514                  MOV     WNUFEA,R0               ;GET EA TO ADJUST
001422  004767  000402                  JSR     PC,ASR04                ;ADJUST IT
001426  010067  177020                  MOV     R0,WBFFEA               ;STORE EA IN SAVE AREA
001432  004767  001460                  JSR     PC,CYCLED
001436  104413  000000'                  ENDITS,BEGIN            ;SIGNAL END OF ITERATION.
                                                                        ;MONITOR SHALL TEST END OF PASS
001442  000763                          BR      MAINC
```

```
                                ;*****************************************************************
                                ;                                                               *
                                ;       INITIALIZE THE UDA                                      *
                                ;                                                               *
                                ;       DO THE 4 STEPS FOR INITIALIZING THE UDA.                *
                                ;                                                               *
                                ;       STEP 1 - CHECK FOR ERROR, STEP 1                        *
                                ;                SEND VECTOR/4, INTERRUPT ENABLE, RING LEN'S = 0*
                                ;                                                               *
                                ;       STEP 2 - CHECK VECTOR ECHO, INTERRUPT ECHO,             *
                                ;                        ERROR, STEP 2                          *
                                ;                SEND PHYSICAL ADDRESS & PURGE = 0              *
                                ;                                                               *
                                ;       STEP 3 - CHECK RING LEN = 0, ERROR, STEP 3              *
                                ;                SEND EXTENDED ADDRESS BITS                     *
                                ;                                                               *
                                ;       STEP 4 - CHECK STEP 4                                   *
                                ;                SEND LFAIL = 0 , GO AND BURST                  *
                                ;                                                               *
                                ;*****************************************************************

001444  012767  000260' 176770  INITUD: MOV     #RSPUNC,VA              ;VA -> RSPUNC
001452  104415  000000' 000442'         GETPAS,BEGIN, VA                ;GET PHYSICAL ADDRESS FROM 16-BIT VA
001460  005004                         CLR     R4                      ;R4 IS USED IF AN ERROR IS DETECTED
001462  012702  000001                 MOV     #1,R2                   ;R2 = STEP INDICATOR REG FOR MSG'S
001466  005077  176314                 CLR     @ADDR                   ;WRITE TO UDAIP TO INIT UDA
001472  012701  002260                 MOV     #TIMER,R1               ;SET TIME OUT LIMIT
001476  017700  176550         1$:     MOV     @UDASA,R0               ;R0 HAS UDASA DATA
001502  032700  100000                 BIT     #<SA.ERR>,R0            ;CHECK FOR ERROR
001506  001007                         BNE     2$                      ;IF FOUND, GET OUT OF LOOP
001510  104407  000000'                 BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR.....
001514  104407  000000'                 BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001520  005301                         DEC     R1                      ;TIME OUT?
001522  001365                         BNE     1$                      ;IF NOT, LOOP
001524  000404                         BR      4$                      ;IF DONE, CONTINUE
001526  012703  004000         2$:     MOV     #SA.S1,R3               ;R3 = STEP 1 BIT
001532  000167  000406                 JMP     ERROR1                  ;IF HERE, ERROR
001536  022700  004400         4$:     CMP     #<SA.S1+SA.DIA>,R0      ;DID DATA COMPARE PROPERLY?
001542  001402                         BEQ     5$                      ;IF SO, CONTINUE
001544  000167  000370                 JMP     ERROR3                  ;REPORT ERROR
                                ;
                                ;       STEP 2
001550  016705  176234         5$:     MOV     VECTOR,R5               ;VECTOR GIVEN
001554  006205                         ASR     R5                      ;SET TO APPROPRIATE VALUE
001556  006205                         ASR     R5                      ; = VECTOR/4
001560  052705  100200                 BIS     #<SA.INT+BIT15>,R5      ;ACTIVATE INTERRUPTS & SET MSB FOR STEP 1
                                                                        ;LEN'S ARE 0
001564  010500                         MOV     R5,R0                   ;STORE R5 IN R0 FOR SUBROUTINE
001566  012703  004000                 MOV     #SA.S1,R3               ;R3 HAS STEP BIT FOR SUBROUTINE
001572  004767  000244                 JSR     PC,SNDSTP               ;SEND STEP DATA
001576  042705  100000                 BIC     #BIT15,R5               ;CLEAR MSB FOR COMPARE DATA
001602  042700  000200                 BIC     #BIT07,R0               ;WAS BIT07 ONLY BIT SET?, SHOULD BE
001606  001404                         BEQ     6$
001610  052700  010200                 BIS     #<SA.S2+BIT07>,R0       ;SET R0 TO REPORT THE ERROR
001614  000167  000320                 JMP     ERROR3                  ;REPORT ERROR
001620  016700  176620         6$:     MOV     EA,R0                   ;R0 GETS PHYSICAL ADDRESS
001624  004767  000212                 JSR     PC,SNDSTP               ;SEND STEP DATA
001630  042705  177400                 BIC     #177400,R5              ;HIGH BYTE CLEARED
```



```
001634  020500                         CMP     R5,R0                   ;CHECK ECHO DATA
001636  001402                         BEQ     7$                      ;IF OK, SKIP
001640  000167  000274                 JMP     ERROR3                  ;IF NOT, REPORT ERROR
001644                          7$:
                                ;
                                ;       STEP 3
001644  016700  176576                 MOV     EA,R0                   ;ADJUST THE EXTENDED ADDRESS BITS
001650  004767  000154                 JSR     PC,ASR04                ;SHIFT EXTENDED ADDRESS BITS FOR UDA
001654  004767  000162                 JSR     PC,SNDSTP               ;SEND STEP DATA
001660  012700  000254'                 MOV     #RSPUNC-4,R0            ;R0 -> RING ENVELOP
                                ;       STEP 4
001664  005720                 8$:     TST     (R0)+                   ;IS THE RING ENTRY = 0?
001666  001117                         BNE     ERROR5                  ;IF NOT, ERROR
001670  022700  000270'                 CMP     #CMDREF,R0              ;IS R0 POINT PAST THE RINGS?
001674  001373                         BNE     8$                      ;IF NOT, LOOP
001676  016700  176116                 MOV     SR2,R0                  ;R0 = BURST VALUE
001702  000241                         CLC                             ;CLEAR CARRY
001704  006300                         ASL     R0                      ;ALIGN BURST FOR STEP 4
001706  006300                         ASL     R0                      ; "
001710  052700  000001                 BIS     #SA.GO,R0               ;SET GO BIT
001714  010077  176332                 MOV     R0,@UDASA               ;SEND DATA TO UDA/DONE WITH THE INIT
001720  012767  000362' 176514         MOV     #CMPACK,VA              ;GET COMMAND PACKET PA AND EA
001726  104415  000000' 000442'         GETPAS,BEGIN, VA                ;GET PHYSICAL ADDRESS FROM 16-BIT VA
001734  016767  176504  176322         MOV     PA,COMMND               ;STORE ADDRESS IN THE RING
001742  016700  176500                 MOV     EA,R0                   ;SAVE IN R0
001746  004767  000056                 JSR     PC,ASR04                ;SHIFT EXTENDED ADDRESS BITS FOR UDA
001752  010067  176310                 MOV     R0,COMMND+2             ;MOVE ADJUSTED EA INTO RING

001756  012767  000276' 176456         MOV     #RSPACK,VA              ;GET RESPONCE PACKET PA AND EA
001764  104415  000000' 000442'         GETPAS,BEGIN, VA                ;GET PHYSICAL ADDRESS FROM 16-BIT VA
001772  016767  176446  176260         MOV     PA,RSPUNC               ;STORE ADDRESS IN THE RING
002000  016700  176442                 MOV     EA,R0                   ;SAVE IN R0
002004  004767  000020                 JSR     PC,ASR04                ;SHIFT EXTENDED ADDRESS BITS FOR UDA
002010  010067  176246                 MOV     R0,RSPUNC+2             ;MOVE ADJUSTED EA INTO RING
002014  012777  004336' 175766         MOV     #INTRUPT,@VECTOR        ;STORE INTERRUPT ADDRESS IN VECTOR
002022  005067  176556                 CLR     TRY                     ;CLEAR TRY SO DRIVE WILL
                                                                        ;GO BACK ONLINE IF NECESSARY
002026  000207                         RTS     PC

                                ;*****************************************************************
                                ;                                                               *
                                ;       ASR04                                                   *
                                ;               ARITHMETIC SHIFT RIGHT REG 0 FOUR TIMES         *
                                ;                                                               *
                                ;       EXTENDED ADDRESS BITS (16 & 17) ARE SET IN BIT POSITION 4 & 5 *
                                ;       RESPECTIVELY.  SHIFT RIGHT FOUR TIMES TO REPOSTION THE VALUE *
                                ;                                                               *
                                ;       INPUT   R0 = UNADJUSTED EXTENDED ADDRESS BITS           *
                                ;                                                               *
                                ;       OUTPUT  R0 = ADJUSTED EXTENDED ADDRESS BITS             *
                                ;                                                               *
                                ;*****************************************************************
002030                  ASR04:
002030  006200                         ASR     R0              ;SHIFT 16
002032  006200                         ASR     R0              ; SHIFT 4
002034  006200                         ASR     R0              ;  SHIFT 2
002036  006200                         ASR     R0              ;   SHIFT 1
002040  000207                         RTS     PC              ;RETURN
```

```
                                        ;*****************************************************************************
                                        ;
                                        ;       SEND STEP DATA
                                        ;
                                        ;       INPU1:  R0 HAS DATA TO BE SENT TO UDA FOR STEP
                                        ;               R3 HAS PREVIOUS STEP FLAG SET
                                        ;
                                        ;       OUTPUT: R0 HAS DATA SENT FROM UDA TO HOST FOR ECHO AND NEXT STEP
                                        ;               R3 HAS CURRENT STEP FLAG SET
                                        ;
                                        ;*****************************************************************************
002042  016701  175742          SNDSTP: MOV     VECTOR,R1               ;
002046  012721  002066'                 MOV     #INTA,(R1)+             ;SET UP INTERRUPT HANDLER ADDRESS
002052  116711  175734                  MOVB    BR1,(R1)                ;SET PRIORITY LEVEL
002056  010077  176170                  MOV     R0,@UDASA               ;SEND STEP1 WRITE FORMMATED DATA
                                        ;
002062  104400  000000'                 EXITS,BEGIN                     ;EXIT TO MONITOR, MODULE WAIT FOR INTERRUPT.
                                        ;
002066                          INTA:
                                        ;-----------------------------------------------------------------------------
002066  000004  000000' 002074'         PIRQS,BEGIN,3S          ; QUEUE UP TO CONTINUE AT 3S AND RT1
                                        ;-----------------------------------------------------------------------------
002074                          3S:
002074  017700  176152                  MOV     @UDASA,R0               ;GET STEP N FORMATTED DATA
002100  032700  100000                  BIT     #SA.ERR,R0              ;TEST FOR ERROR
002104  001017                          BNE     ERROR1                  ;IF NOT OK, REPORT
002106  005202                          INC     R2                      ;SET STEP REGISTER
002110  006303                          ASL     R3                      ;R3 HAS STEP BIT PROPERLY SET
002112  030300                          BIT     R3,R0                   ;WAS STEP N SET?
002114  001002                          BNE     4S                      ;IF SO, CONTINUE
002116  000167  000020                  JMP     ERROR2                  ;IF NOT CORRECT STEP, ERROR
002122  040300                  4S:     BIC     R3,R0                   ;CLEAR THE STEP BIT, FOR COMPARE
002124  000207                          RTS     PC                      ;RETURN
```



```
                                        ;*****************************************************************************
                                        ;                                                                            ;
                                        ;       ERROR 1                                                              ;
                                        ;       PRINT AN ERROR REPORTED BY THE UDA DIAGNOSTICS                       ;
                                        ;                                                                            ;
                                        ;       ERROR2                                                               ;
                                        ;       PRINT A THE VALUE OF THE UDASA WHEN THE STEP BIT WAS NOT SET         ;
                                        ;                                                                            ;
                                        ;       ERROR3                                                               ;
                                        ;       PRINT A THE VALUE OF THE UDASA WHEN THE ECHO WAS NOT SET             ;
                                        ;               CORRECTLY                                                    ;
                                        ;                                                                            ;
                                        ;                                                                            ;
                                        ;       INPUT   R0 => UDASA                                                  ;
                                        ;               R2 = STEP COUNT                                              ;
                                        ;                                                                            ;
                                        ;       OUTPUT  THE RETRY COUNT IS INCREMENTED                               ;
                                        ;               IF THE RETRY COUNT > RETRY LIMIT, END MODULE                 ;
                                        ;                                                                            ;
                                        ;       ERRORS                                                               ;
                                        ;       RING WASN'T ALL ZERO -> ERROR                                        ;
                                        ;       DROP UDBA0                                                           ;
                                        ;                                                                            ;
                                        ;*****************************************************************************
002126                          ERRORS:
002126  104403  000000' 005374'         MSGN$,BEGIN,ZERO        ;ASCII MESSAGE CALL WITH COMMON HEADER
002134  104410  000000'                 ENDS,BEGIN              ;
                                        ;
002140  005204                  ERROR3: INC     R4                      ;R4 = 3 FOR ERROR3
002142  005204                  ERROR2: INC     R4                      ;R4 = 2 FOR ERROR2
002144  005204                  ERROR1: INC     R4                      ;R4 = 1 FOR ERROR1
002146  010267  176302                  MOV     R2,NUM                  ;STORE STEP REG IN A NUMBER FOR CONVRT
                                        ;*************************************************************
                                                                        ;CONVERT NUM TO ASCII AND
002152  104420  000000' 000454'         OTOA$,BEGIN,NUM,ADR2            ;STORE AT ADR2
002160  000475'
                                        ;*************************************************************
002162  017767  176064  176264          MOV     @UDASA,NUM              ;STORE VALUE IN A NUMBER
                                        ;*************************************************************
                                                                        ;CONVERT NUM TO ASCII AND
002170  104420  000000' 000454'         OTOA$,BEGIN,NUM,ADR1            ;STORE AT ADR1
002176  000466'
                                        ;*************************************************************
002200  005304                          DEC     R4                      ;ERROR 1?
002202  001003                          BNE     1S                      ;IF NOT, CHECK IF IT IS THE NEXT ERROR
002204  104403  000000' 005204          MSGN$,BEGIN,INITE1      ;ASCII MESSAGE CALL WITH COMMON HEADER
002212                          1S:
002212  005304                          DEC     R4                      ;ERROR 2?
002214  001003                          BNE     2S                      ;IF NOT, CHECK IF IT IS THE NEXT ERROR
002216  104403  000000' 005230          MSGN$,BEGIN,INITE2      ;ASCII MESSAGE CALL WITH COMMON HEADER
002224                          2S:
002224  005304                          DEC     R4                      ;ERROR 3?
002226  001003                          BNE     3S                      ;IF NOT, CHECK IF IT IS THE NEXT ERROR
002230  104403  000000' 005236          MSGN$,BEGIN,INITE3      ;ASCII MESSAGE CALL WITH COMMON HEADER
```

```
 47 002236                              3$:
 48                                             ;*****************************************************************
                                                ;CONVERT ADDR TO ASCII AND
                                                ;STORE AT ADR3
    002236  104420  000000' 000006'            OTOA$,BEGIN,ADDR,ADR3
    002244  000504'
 49                                             ;*****************************************************************
    002246  104405  000000' 000000             ;*****************************************************************
                                                HRDER$,BEGIN,NULL       ;
                                                ;*****************************************************************
 50 002254  104403  000000' 005212'            MSGN$,BEGIN,INITER      ;ASCII MESSAGE CALL WITH COMMON HEADER
 51 002262  005267  176316                      INC     TRY             ;INCREMENT RETRY COUNT
 52 002266  022767  000004  176310              CMP     #RLIM,TRY       ;IS THE RETRY COUNT EXCEEDED?
 53 002274  001402                              BEQ     6$              ;IF SO, END IT
 54 002276  000167  176356                      JMP     START           ;IF NOT, TRY AGAIN
 55 002302                              6$:
    002302  104403  000000' 005370'            MSGN$,BEGIN,ABORT       ;ASCII MESSAGE CALL WITH COMMON HEADER
 56 002310  104410  000000'                     END$,BEGIN              ;
 57
```



```
  1                                             ;*************************************************************************
  2                                             ;
  3                                             ;       SET UP
  4                                             ;
  5                                             ;       GO FIND OUT WHAT DRIVES ARE OUT THERE.
  6                                             ;       A TABLE IS FILLED WITH UNIT NUMBERS(MAX IS 16)
  7                                             ;
  8                                             ;       THIS SHOULD ONLY BE DONE AT THE VERY BEGINNING OF RUNNING
  9                                             ;       THIS DECX MODULE; THEN NOT RUN AGAIN.
 10                                             ;
 11                                             ;       INPUT:  DVICE HAS APPROPRIATE BITS SET.  THE # OF BITS =
 12                                             ;               # OF DRIVES WANTED TO TEST.
 13                                             ;               POSITION OF BITS = WHICH DRIVE IN THE SYSTEM IS DESIRED.
 14                                             ;
 15                                             ;*************************************************************************
 16 002314                              SETUP:
 17                                             ; ***   SET CONTRL CHAR AND WAIT FOR THE ATTENTION MESSAGES
 18 002314  004767  001650                      JSR     PC,SCC                  ;SET CONTROLLER CHARACTERISTICS
 19 002320  005367  175744                      DEC     CMDREF                  ;ONLY SET UP AT BEGINNING OF MODULE
 20 002324  001110                              BNE     19$                     ; (USE DRIVES FOUND AT BEGINNING)
 21 002326  012703  000001                      MOV     #1,R3                   ;INITIAL PORTID VALUE
 22 002332  012704  000614'                     MOV     #TABLEW,R4              ;R4 -> TABLEW
 23 002336  011467  176240                      MOV     (R4),UNITNO             ;INITIAL UNITNO IN TABLEW
 24 002342  016714  176234              1$:     MOV     UNITNO,(R4)             ;UNIT NO SET IN TABLEW;READY TO TEST
 25 002346  010367  176234                      MOV     R3,PORTID               ;PORT ID SET
 26 002352  010364  000002                      MOV     R3,2(R4)                ;PORTID SET IN TABLEW
 27 002356  012764  177777  000004              MOV     #177777,4(R4)           ;INSERT NEW -1,-1 FOR LAST ENTRY
 28 002364  016464  000004  000006              MOV     4(R4),6(R4)             ;  OF THE TABLEW
 29 002372  012767  002400  176212              MOV     #2400,WORK              ;WORK = RETRY LIMIT
 30 002400  004767  001526              3$:     JSR     PC,GTSTAT               ;GET STATUS, GET NEXT UNIT NUMBER
 31 002404  103006                              BCC     7$                      ;OK, CONTINUE
 32 002406  005367  176200                      DEC     WORK                    ;ELSE IF OFFLINE, DECR COUNT
 33 002412  001372                              BNE     3$                      ;IF COUNT > 0, TRY AGAIN.
 34 002414  004767  000774              5$:     JSR     PC,DROP2                ;DROP THE DRIVE
 35 002420  000437                              BR      17$                     ;TRY NEXT UNIT
 36 002422  016767  175654  176152      7$:     MOV     P.UNIT+RSPACK,UNITNO    ;UNIT NUMBER FROM RESPONCE PACKET IN UNITNO
                                                ; *** CHECK FOR CASE WHERE THE MORE UNITS THEN DRIVES HAVE BEEN SPECIFIED.
 38                                             ; *** NEXT UNIT MODIFIER WILL GIVE A DUPLICATE UNIT NUMBER.
 39 002430  012702  000614'                     MOV     #TABLEW,R2              ;R2 -> TABLE TO FIND DUPLICATE
 40 002434  012705  000001                      MOV     #1,R5                   ;R5 IS TEMP PORTID
 41 002440  020227  000654'              9$:     CMP     R2,#TEND                ;REACHED THE BOTTOM?
 42 002444  001420                              BEQ     15$                     ;IF SO, EXIT
 43 002446  020305                              CMP     R3,R5                   ;REACHED THE LATEST ENTRY?
 44 002450  001416                              BEQ     15$                     ;IF SO, EXIT
 45 002452  026712  176124                      CMP     UNITNO,(R2)             ;DO WE HAVE A DUPLICATE UNIT NUMBER?
 46 002456  001404                              BEQ     13$                     ;IF SO, ERROR
 47 002460  062702  000004                      ADD     #4,R2                   ;IF NOT, POINT TO NEXT POINTER
 48 002464  006305                              ASL     R5
 49 002466  000764                              BR      9$                      ;AND CONTINUE
 50 002470  011467  176106              13$:    MOV     (R4),UNITNO             ;DROP DRIVE FROM TABLE
 51 002474  010367  176106                      MOV     R3,PORTID               ;
 52 002500  004767  000720                      JSR     PC,DROP3                ;AND DROP IT
 53 002504  000405                              BR      17$
 54 002506                              15$:
 55                                             ; ***
 56 002506  026714  176070                      CMP     UNITNO,(R4)             ;IS THE UNITNO CORRECT?
 57 002512  001402                              BEQ     17$                     ;IF SO, CHECK FOR NEXT UNIT
```

```
                                        MOV     UNITNO,(R4)             ;ELSE, CORRECT THE UNIT NUMBER IN TABLE
                                17$:
                                        ASL     R3                      ;NEXT PORTID SET
                                        CMP     DVICE,R3                ;DONE?
                                        BMI     19$                     ;IF R3 > DVICE, ALL DESIRED DRIVES ARE FOUND.
                                        INC     UNITNO                  ;NEXT UNITNO SET
                                        ADD     #4,R4                   ;POINT TO NEXT ENTRY TO TEST DRIVE
                                        CMP     #TEND,R4                ;POINT TO END? IF SO, TABLE FULL
                                        BHI     1$                      ;IF R4 NOT REACHED END, GO TEST
                                19$:
                                        RTS     PC

                                ;*****************************************************************************
                                ;
                                ;       TSTOFL
                                ;
                                ;               TEST TO SEE WHAT KIND OF AN OFFLINE CONDITION HAS OCCURED.
                                ;
                                ;*****************************************************************************
                                TSTOFL: CMP     #ST.OFL,R0              ;WAS THE DRIVE FOUND OFFLINE?
                                        BEQ     10$                     ;CHECK WHAT KIND OF OFFLINE
                                        CMP     #ST.DRV,R0              ;WAS IT A DRIVE ERROR? -> SDI?
                                        BNE     13$                     ;IF IT WAS NOT, ERROR (DROP DRIVE)
                                10$:    BIT     #<SC.NVL+SC.DIS+SC.DUP+SC.IOP>,P.STS+RSPACK     ;WERE ANY OF THESE BITS SET?
                                                                        ; = NO VOLUME MOUNTED, UNIT DIABLED BY FIELD SRVICE
                                                                        ;  OR DUPLICATE UNIT NUMBER OR UNIT INUPERATIVE
                                        BNE     12$                     ;IF SO, EXIT
                                        BIT     #^C<SC.NVL+SC.DIS+SC.DUP+SC.IOP+ST.MSK>,P.STS+RSPACK ; ANY OTHER DATA?
                                        BNE     13$                     ;IF SO, DROP
                                12$:    CLC                             ;CLEAR CARRY
                                        RTS     PC                      ;RETURN
                                13$:    SEC                             ;SET CARRY, DRIVE WAS FOUND TO BE OFFLINE
                                                                        ; OR ANOTHER ERROR
                                        JSR     PC,ERRORH               ;REPORT ERROR
                                        RTS     PC                      ;RETURN
```



```
                                ;*****************************************************************************
                                ;                                                                            ;
                                ;       CYCLE DISK                                                           ;
                                ;                                                                            ;
                                ;       DO THE DISK CYCLE                                                    ;
                                ;               DO GET STATUS COMMANDS TO ASSURE THAT THE DRIVE              ;
                                ;                       IS AVAILABLE (FOR DUAL PORTING)                      ;
                                ;               CHECK DRIVE TO BE ONLINE                                     ;
                                ;               IF TRUE                                                      ;
                                ;                       PICK THE BLOCK                                       ;
                                ;                       WRITE                                                ;
                                ;                       READ                                                 ;
                                ;                       DATA CHECK                                           ;
                                ;                       MAKE THE DRIVE AVAILABLE                             ;
                                ;               ELSE DROP DRIVE                                              ;
                                ;                                                                            ;
                                ;*****************************************************************************
                                CYCLED:
                                        BIT     #SR.DUA,SR1             ;DUAL PORT?
                                        BNE     2$                      ;IF NOT, CONTINUE
                                ; *** CHECK IF WE DO ONLINE FOR THE FIRST TIME.
                                        TST     TRY                     ;IF TRY HAS SET MSB, DON'T DO ONLINE
                                        BMI     10$                     ; DON'T DO ONLINE
                                        BR      10$                     ; ELSE DO ONLINE (1ST TIME THROUGH IN THIS PASS)
                                ; ***
                                ; *** DO GET STATUS COMMANDS TO ASSURE THE DRIVE IS AVAILABLE TO THE UDA
                                ; ***  FOR DUAL PORTING.
                                ; ***
                                2$:     MOV     #10,R1                  ;R1 = # OF GET STATUS TO DO
                                4$:     JSR     PC,GISTA1               ;IS THE DRIVE OFFLINE?
                                        BCC     6$                      ;IF ALL OK, DO THE CYCLE
                                        JSR     PC,TSTOFL               ;ELSE, CHECK IF OFFLINE
                                        BCS     24$                     ;IF IT ERRED, DROP THE DRIVE
                                ; *** HANDLE OFF LINE DRIVE, WAIT FOR AVAILABLE ATTENTION MESSAGE
                                        CLR     EXPAV                   ;EXPECT AN AVAILABLE ATTENTION MESSAGE
                                        BIS     #<RG.OWN+RG.FLG>,RSPUNC+2       ;SET RING FOR ATTN MESSAGE
                                        JSR     PC,INTERP               ;WAIT FOR MESSAGE
                                                                        ; 2ND ATTENTION MESSAGE
                                        BR      10$
                                6$:     DEC     R1                      ;DONE?
                                        BNE     4$                      ;IF NOT DONE, TRY AGAIN
                                10$:    JSR     PC,ONLINE               ;DO AND ONLINE COMMAND
                                        BCS     2$                      ;IF CARRY WAS SET, TRY AGAIN
                                14$:    MOV     P.UNSZ+2+RSPACK,UNSZH   ;IS THE UNIT SIZE HI ADDRESS
                                        MOV     P.UNSZ+RSPACK,UNSZL     ;GET UNIT SIZE/IS IT = 0?
                                        BNE     10$                     ;IF NOT ZERO, CONTINUE WITH ITERATION
                                        TST     UNSZH                   ;IS UNSZH ALSO 0?
                                        BEQ     CYCLED                  ;IF 0, TRY TO BRING ONLINE AGAIN
                                ; *** SET MSB OF TRY TO SHOW THAT INITIAL ONLINE IS DONE
                                        MOV     #100000,TRY

                                ;*****************************************************************************
                                ;                                                                            ;
                                ;       THE FOLLOWING SEGMENT SETS THE LIMIT FOR THE UNIT SIZE.              ;
                                ;       THE VALUE (UNIT SIZE - (WRITE BUFFER SIZE/NORMAL BLOCK SIZE))        ;
                                ;       IS THE LAST SECTOR POSSIBLE TO RIGHT TO.                             ;
                                ;                                                                            ;
```

```
                                        ;*************************************************************************
002744                                  16$:
002744  016700  175172                          MOV     WBUFSZ,R0               ;WBUFSZ IN R0 AS A LIMIT
002750  005001                                  CLR     R1                      ;R1 = # OF BLOCKS
002752  005201                          18$:    INC     R1                      ;INCREMENT THE # OF BLOCKS
002754  162700  000400                          SUB     #400,R0                 ;DECREMENT A BLOCK
002760  100374                                  BPL     18$                     ;BR IF > 0
002762  160167  175604                          SUB     R1,UNSZL                ;ADJUST THE UNIT SIZE
                                        ; ***   NOW PICK WHICH BLOCK TO WRITE TO
002766  004767  000156                          JSR     PC,PICKBK               ;ELSE SELECT A SECTOR TO TEST
002772  004767  001134                          JSR     PC,GTSTAT               ;DID WE NOT GET THE DRIVE ONLINE?
003000  103720                                  BCS     2$                      ;IF WE DID NOT, GO BACK TO TOP AND TRY AGAIN
003002  022700  000004                          CMP     #ST.AVL,R0              ;IS IT AVAILABLE?
003006  001715                                  BEQ     2$                      ;IF SO, GO BACK TO TOP AND TRY AGAIN
                                        ; ***   WRITE TO THE BLOCK SELECTED
003010  004767  000720                          JSR     PC,WRITE                ;WRITE THE DATA FOR USER DEFINED # OF WORDS
003014  103007                                  BCC     19$                     ;IF OK, CONTINUE
003016  032767  001000  174774                  BIT     #SR.DUA,SR1             ;ARE WE DOING DUAL PORT?
003024  001306                                  BNE     2$                      ;IF YES, RETRY
003026  004767  001720                          JSR     PC,ERRORH               ;ELSE, HARD ERROR
003032  000421                                  BR      22$                     ;AND EXIT; BCS 22$ ;IF ERROR, EXIT
                                        ; ***   READ IT BACK
003034  004767  000730                  19$:    JSR     PC,READ                 ;READ A BLOCK
003040  103416                                  BCS     22$                     ;IF ERROR, EXIT
003042  032767  004000  174750                  BIT     #SR.CMP,SR1             ;DO A DATA COMPARE?
003050  001004                                  BNE     20$                     ;IF NOT, SKIP THE COMPARE
                                        ; ***   COMPARE DATA
003052  104412  000000' 000126'                 CDATA$,BEGIN,RBUFPA     ; REQUEST FOR MONITOR TO CHECK DATA
003056  003060'                                 .+2                     ; IF ERROR, CONTINUE
003060  032767  001000  174730          20$:    BIT     #SR.DUA,SR1             ;DO WE DO AN AVAILABLE?
003066  001402                                  BEQ     22$                     ;IF NOT(BIT NOT SET) SKIP AVAILABLE
                                        ; ***   MAKE THE DRIVE AVAILABLE
003070  004767  001014                          JSR     PC,AVAILM               ;RELEASE THE DRIVE
003074  000241                          22$:    CLC                             ;EVERY THING WAS OK
                                                                                ;WASTE A LITTLE TIME SO OTHER
                                                                                ;  CONTROLLER CAN GRAB DRIVE
003076  000207                          24$:    RTS     PC                      ;RETURN

                                        ; *** SUBROUTINE TO WAIT FOR AN INTERRUPT
                                        ; *** RETURNS AFTER THE INTERRUPT OCCURS
                                        DOINTR:
003100                                          CLR     EXPAV                   ;EXPECT AN AVAILABLE ATTENTION MESSAGE
003100  005067  175360
003104  052767  140000  175150                  BIS     #<RG.OWN+RG.FLG>,RSPONC+2   ; SET OWN AND FLAG FOR RESPONCE RING
003112  000167  001214                          JMP     INTERP                  ;WAIT FOR ATTENTION MESSAGE & RETURN

                                        ;*************************************************************************
                                        ;                                                                       *
                                        ;       DISKLESS CYCLE                                                  *
                                        ;               DO A MAITENENCE WRITE                                   *
                                        ;               AND A MAITENENCE READ                                   *
                                        ;               AND CHECK THE DATA                                      *
                                        ;                                                                       *
                                        ;*************************************************************************
003116                                  CYCLEL:
003116  004767  000470                          JSR     PC,MAITW                ;DO A MAINTENENCE WRITE
003122  004767  000430                          JSR     PC,MAITR                ;DO A MAINTENENCE READ
003126  032767  004000  174662                  BIT     #SR.CMP,SR1             ;DO A DATA COMPARE?
```



```
003134  001004                                  BNE     21$                     ;IF NOT, SKIP THE COMPARE
003136  104412  000000' 000126'                 CDATA$,BEGIN,RBUFPA     ; REQUEST FOR MONITOR TO CHECK DATA
003144  003146'                                 .+2                     ; IF ERROR, CONTINUE
003146                                  21$:
003146  000207                                  RTS     PC
```

```
                                ;*****************************************************************************
                                ;                                                                            *
                                ;       PICK A BLOCK TO WRITE TO.                                            *
                                ;                                                                            *
                                ;       EITHER PICK THE NEXT SEQUNTIAL BLOCK (DEFAULT) OR TAKE ONE AT        *
                                ;       RANDOM.                                                              *
                                ;                                                                            *
                                ;       OUTPUT: FILL SECH & SECL (CURRENT SECTOR ADDR)                       *
                                ;                                                                            *
                                ;*****************************************************************************
003150                          PICKBK:
003150  032767  002000  174640          BIT     #SR,SEQ,SR1             ;CHECK SR1 FOR RANDOM ACCESS MODE
003156  001467                        BEQ     SEQACC                  ;BR IF SEQUENTIAL ACCESS
003160                          RANACC:
003160  104417  000000'                RAND$,BEGIN
003164  016746  174664                MOV     RANNUM,-(SP)            ;GENERATE THE SECTOR ADDRESS
003170  104417  000000'                RAND$,BEGIN
003174  016746  174654                MOV     RANNUM,-(SP)            ;GENERATE THE SECTOR ADDRESS
                                ;
                                ;       ADJUST HI ADDRESS FIRST
                                ;
003200  000241                        CLC                             ;CLEAR CARRY FOR ROTATE
003202  042716  100000                BIC     #100000,(SP)            ;CLEAR UPPER BIT MAKES SURE VALUE'S +
003206  012667  175356                MOV     (SP)+,SECH              ;STORE IN SECTOR HI ADDRESS
003212  005767  175356                TST     UNSZH                   ;IS THE MAX SIZE 0?
003216  001430                        BEQ     3$                      ;IF 0, GET LOW SECTOR ADDRESS
                                ;  ***  UNSZH > 0 IF CODE FALLS THROUGH HERE
003220  016700  175350                MOV     UNSZH,R0                ;R0 = MAX VALUE
003224  005100                        COM     R0                      ;R0 COMPLEMENT, NOW FIND MS ZERO
003226  012701  100000                MOV     #100000,R1              ;R1 IS INDEX INTO MAX VALUE
003232  030100                1$:     BIT     R1,R0                   ;HAVE 0 YET?
003234  001403                        BEQ     2$                      ;IF 1ST 0 REACHED, CLEAR REST OF THE BITS
003236  000241                        CLC                             ;CLEAR CARRY FOR ROR
003240  006001                        ROR     R1                      ;POINT TO NEXT BIT
003242  000773                        BR      1$                      ;BRANCH TO TEST AGAIN
003244  040100                2$:     BIC     R1,R0                   ;CLEAR REST OF THE BITS
003246  000241                        CLC                             ;CLEAR CARRY FOR ROR
003250  006001                        ROR     R1                      ;IF R1 ROTATES INTO CARRY, R1 = 0
003252  001374                        BNE     2$                      ;IF R1 NOT 0, MORE BITS TO CLEAR
003254  040067  175310                BIC     R0,SECH                 ;CLEAR UPPER BITS OF HIGH SECTOR VALUE
003260  026767  175304  175306        CMP     SECH,UNSZH              ;IF THE HIGH SECTOR VALUE > MAX VALUE?
003266  002420                        BLT     7$                      ;IF <, EXIT
003270  001405                        BEQ     4$                      ;IF =, TEST LOW ORDER VALUE
003272  006267  175272                ASR     SECH                    ;SECH = SECH/2 - CAN'T BE > MAX NOW
003276  000414                        BR      7$                      ;EXIT
                                ;
                                ;       GET LOW SECTOR ADDRESS
                                ;
003300  005067  175264          3$:   CLR     SECH                    ;CLEAR HI SECTOR SIZE
003304  005767  175262          4$:   TST     UNSZL                   ;IS THE HIGHEST POSSIBLE = 0?
003310  001406                        BEQ     6$                      ;IF TRUE, DON'T DO LOOP
003312  166716  175254          5$:   SUB     UNSZL,(SP)              ;ELSE, SECL = SECL - UNSZL (ADJUST)
003316  103375                        BCC     5$                      ;IF UNSZL > SECL, LOOP
003320  066716  175246                ADD     UNSZL,(SP)              ;ELSE SUBTRACTED ONCE TOO OFTEN
003324  000401                        BR      7$                      ; AND EXIT
003326  005016                6$:     CLR     (SP)                    ;CLEAR LO SECTOR ADDRESS (IF HIGHEST POSSIBLE = 0)
003330  012667  175232          7$:   MOV     (SP)+,SECL              ;SAVE LO SECTOR ADDRESS
```



```
003334  000207                        RTS     PC                      ; RETURN


                                ;GENERATE DISK ADDRESS BY SEQUENTIAL ADDRESSING
003336                          SEQACC:
003336  005267  175224                INC     SECL                    ;INCREMENT THE SECTOR ADDRESS
003342  001405                        BEQ     16$                     ;BR IF ZERO
003344  026767  175216  175220        CMP     SECL,UNSZL              ;OVER LIMIT?
003352  103413                        BLO     18$                     ;BR IF LOWER
003354  000402                        BR      17$                     ;SKIP THE INCREMENT
003356                          16$:
003356  005267  175206                INC     SECH                    ;INCREMENT SECTOR HIGH ADDRESS
003362                          17$:
003362  026767  175202  175204        CMP     SECH,UNSZH              ;OVER LIMIT?
003370  103404                        BLO     18$                     ;BR IF LOWER
003372  005067  175170                CLR     SECL                    ;RESET THE STARTING SECTOR ADDRESS
003376  005067  175166                CLR     SECH                    ;

003402                          18$:
003402  000207                        RTS     PC
```

```
                                        ;*********************************************************************
                                        ;                                                                   ;
                                        ;       DROP A DRIVE                                                ;
                                        ;                                                                   ;
                                        ;       A DRIVE WOULDN'T RESPOND, DROP IT.  SET THIS UP IN DVICE.   ;
                                        ;                                                                   ;
                                        ;       INPUT   UNITNO = UNIT NUMBER OF DRIVE TO DROP               ;
                                        ;               PORTID = BIT SET TO DROP DRIVE                      ;
                                        ;                                                                   ;
                                        ;       OUTPUT  DVICE HAS A BIT CLEARED.  THE BIT POSITION          ;
                                        ;               REPRESENTS THE DRIVE                                ;
                                        ;                                                                   ;
                                        ;*********************************************************************

003404                                  DROP1:
003404  012767  000001  175042                  MOV     #1,NUM
003412  000407                                  BR      DROP4
003414                                  DROP2:
003414  012767  000002  175032                  MOV     #2,NUM
003422  000403                                  BR      DROP4
003424                                  DROP3:
003424  012767  000003  175022                  MOV     #3,NUM
003432  036767  175150  175140          DROP4:  BIT     PORTID,DVICE            ;HAS THE DRIVE BEEN DROPPED, DON'T DROP AGAIN
003440  001445                                  BEQ     10$                     ;
003442  022767  177777  175136                  CMP     #177777,PORTID          ;IF DRIVE HAS BEEN DROPPED, DON'T DROP AGAIN
                                                                                ;        (WILL ZERO DVICE PREMATURE)
003450  001441                                  BEQ     10$                     ;IF =, DRIVE HAS BEEN DROPPED -> EXIT ROUTINE

003452  046767  175130  175120                  BIC     PORTID,DVICE            ;DROP THE DRIVE
                                                ;**************************************************************
                                                                        ;CONVERT UNITNO TO ASCII AND
                                                                        ;STORE AT ADR2
003460  104421  000000' 000602'                 BTODS,BEGIN,UNITNO,ADR2
003466  000475'
                                                ;**************************************************************
003470  105067  175006                          CLRB    ADR2+5
                                                ;**************************************************************
                                                                        ;CONVERT PORTID TO ASCII AND
                                                                        ;STORE AT ADR1
003474  104420  000000' 000606'                 UTOAS,BEGIN,PORTID,ADR1
003502  000466'
                                                ;**************************************************************
003504  012764  177777  000002                  MOV     #177777,2(R4)           ;DESELECT DRIVE SO IT WON'T BE USED AGAIN.
003512  005367  174736                          DEC     NUM                     ;DROPPED FOR WHICH ERROR?
003516  001004                                  BNE     1$                      ;IF NOT FOR ERRORS, CONTINUE
003520  104403  000000' 005244'                 MSGNS,BEGIN,DRP1        ;ASCII MESSAGE CALL WITH COMMON HEADER
003526  000412                                  BR      10$
003530  005367  174720                  1$:     DEC     NUM                     ;WAS UNIT NOT FOUND?(NON EXISTENT UNIT)
003534  001004                                  BNE     2$                      ;IF NOT, CONTINUE
003536  104403  000000' 005262'                 MSGNS,BEGIN,DRP2        ;ASCII MESSAGE CALL WITH COMMON HEADER
003544  000403                                  BR      10$
003546                                  2$:
003546  104403  000000' 005300'                 MSGNS,BEGIN,DRP3        ;ASCII MESSAGE CALL WITH COMMON HEADER
                                                                                ; ACTUAL UNITS FOUND
003554  000207                          10$:    RTS     PC
```

```
                                        ;*****************************************************************************
                                        ;                                                                            *
                                        ;       MAITENENCE READ                                                      *
                                        ;                                                                            *
                                        ;       SET UP A PACKET WITH:                                                *
                                        ;               OPCODE & MODIFIER                                            *
                                        ;               REGION ID & REGION OFFSET                                    *
                                        ;               READ BUFFER DESCRIPTOR                                       *
                                        ;               BYTE COUNT                                                   *
                                        ;       THEN SEND THE PACKET                                                 *
                                        ;                                                                            *
                                        ;*****************************************************************************
003556  004767  001074                  MAITR:  JSR     PC,CLRPAK               ;CLEAR THE PACKETS
003562  012767  000030  174602                  MOV     #OP.MRD,P.OPCD+CMPACK   ;SET THE OPCODE
003570  016767  174654  174606                  MOV     RBFFEA,P.ADEA+CMPACK    ;SET THE BUFFER DESCRIPTOR
003576  016767  174324  174576                  MOV     RBUFPA,P.ADPA+CMPACK    ;
003604  016700  174322                          MOV     RBUFSZ,R0               ;STORE THE BUFFER SIZE IN WORDS
003610  000424                                  BR      MAITP                   ;SET UP THE REST OF THE PACKET

                                        ;*****************************************************************************
                                        ;                                                                            *
                                        ;       MAITENENCE WRITE                                                     *
                                        ;                                                                            *
                                        ;       SET UP A PACKET WITH:                                                *
                                        ;               OPCODE & MODIFER                                             *
                                        ;               REGION ID & REGION OFFSET                                    *
                                        ;               WRITE BUFFER DESCRIPTOR                                      *
                                        ;               BYTE COUNT (EITHER WBUFSZ OR LIMIT IF WBUFSZ > LIMIT)        *
                                        ;       THEN SEND THE PACKET                                                 *
                                        ;                                                                            *
                                        ;*****************************************************************************

003612  004767  001040                  MAITW:  JSR     PC,CLRPAK               ;CLEAR THE PACKETS
003616  012767  000031  174546                  MOV     #OP.MWR,P.OPCD+CMPACK   ;SET THE OPCODE
003624  016767  174622  174552                  MOV     WBFFEA,P.ADEA+CMPACK    ;SET THE BUFFER DESCRIPTOR
003632  016767  174276  174542                  MOV     WBUFPA,P.ADPA+CMPACK    ;
003640  026767  174276  174730                  CMP     WBUFSZ,LIMIT            ;IS THE BUFFER SIZE > LIMIT?
003646  100403                                  BMI     1$                      ;IF NOT, WBUFSZ IS OK
003650  016700  174722                          MOV     LIMIT,R0                ;STORE THE BUFFER SIZE IN WORDS
003654  000402                                  BR      MAITP                   ;AND SKIP
003656  016700  174260                  1$:     MOV     WBUFSZ,R0               ;STORE THE BUFFER SIZE IN WORDS
003662  006300                          MAITP:  ASL     R0                      ;MAKE IT NUMBER OF BYTES
003664  010067  174506                          MOV     R0,P.BCNT+CMPACK        ;SET WRITE BUFFER SIZE
003670  012767  000020  174374                  MOV     #16.,RSPLEN             ;SET RESPONCE PACKET LENGTH
003676  012767  000044  174452                  MOV     #36.,CMPLEN             ;SET CUMMAND PACKET LENGTH
003704  012767  000001  174504                  MOV     #1,P.RGID+CMPACK        ;SET REGION ID = 1
003712  012767  177777  174440                  MOV     #177777,CMPVIR          ;SET COMMAND VIRTUAL CIRCUIT (-1 FOR DM)
003720  012767  177777  174346                  MOV     #177777,RSPVIR          ;SET COMMAND VIRTUAL CIRCUIT
003726  000167  000330                          JMP     SEND                    ;SEND THE PACKET
```



```
                                        ;*****************************************************************************
                                        ;                                                                            *
                                        ;       WRITE                                                                *
                                        ;                                                                            *
                                        ;       SET UP OP CODE, MODIFIERS,BUFFER SIZE (BYTE COUNT),                  *
                                        ;       BUFFER DESCRIPTOR (PYSICAL AND EXTENDED ADDRESS)                     *
                                        ;               GET READ SET SIMULAR DATA IN THE PACKET:                     *
                                        ;       DISK ADDRESS AND CYLINDER ID (LOGICAL BLOCK NUMBER),                 *
                                        ;       THEN SEND THE PACKET.                                                *
                                        ;                                                                            *
                                        ;*****************************************************************************
003732  004767  000720                  WRITE:  JSR     PC,CLRPAK               ;CLEAR PACKETS
003736  012767  000042  174426                  MOV     #OP.WR,P.OPCD+CMPACK    ;SET THE OPCODE
003744  016700  174172                  WRITEA: MOV     WBUFSZ,R0               ;STORE THE BUFFER SIZE IN WORDS
003750  016767  174160  174424                  MOV     WBUFPA,P.ADPA+CMPACK    ;SET THE BUFFER DESCRIPTOR(PA)
003756  016767  174470  174420                  MOV     WBFFEA,P.ADEA+CMPACK    ;SET THE BUFFER DESCRIPTOR(EA)
003764  000415                                  BR      READA                   ;

                                        ;*****************************************************************************
                                        ;                                                                            *
                                        ;       READ                                                                 *
                                        ;                                                                            *
                                        ;       SET UP OP CODE, MODIFIERS,BUFFER SIZE (BYTE COUNT),                  *
                                        ;       BUFFER DESCRIPTOR (PYSICAL AND EXTENDED ADDRESS),                    *
                                        ;       DISK ADDRESS AND CYLINDER ID (LOGICAL BLOCK NUMBER),                 *
                                        ;       THEN SEND THE PACKET.                                                *
                                        ;                                                                            *
                                        ;*****************************************************************************
003766  004767  000664                  READ:   JSR     PC,CLRPAK               ;CLEAR PACKETS
003772  012767  000041  174372                  MOV     #OP.RD,P.OPCD+CMPACK    ;SET THE OPCODE
004000  016700  174126                          MOV     RBUFSZ,R0               ;STORE THE BUFFER SIZE IN WORDS
004004  016767  174440  174372                  MOV     RBFFEA,P.ADEA+CMPACK    ;SET THE BUFFER DESCRIPTOR
004012  016767  174110  174362                  MOV     RBUFPA,P.ADPA+CMPACK    ;
004020  012767  000040  174244          READA:  MOV     #32.,RSPLEN             ;SET RESPONCE PACKET LENGTH
004026  012767  000040  174322                  MOV     #32.,CMPLEN             ;SET COMMAND PACKET LENGTH
004034  006300                                  ASL     R0                      ;MAKE IT NUMBER OF BYTES
004036  010067  174334                          MOV     R0,P.BCNT+CMPACK        ;SET READ BUFFER SIZE
004042  016767  174520  174346                  MOV     SECL,P.LBN+CMPACK       ;SET LOGICAL BLOCK NUMBER
004050  016767  174514  174342                  MOV     SECH,P.LBN+2+CMPACK     ;
004056  000501                                  BR      SEND                    ;SEND THE PACKET
```

```
                                        ;*****************************************************************************
                                        ;                                                                            ;
                                        ;       DETERMINE ACCESS PATHS                                               ;
                                        ;                                                                            ;
                                        ;       SET UP CODE, GO SEND PACKET                                          ;
                                        ;                                                                            ;
                                        ;*****************************************************************************
004060  004767  000572          DAP:    JSR     PC,CLRPAK               ;CLEAR PACKETS
004064  012767  000013  174300          MOV     #OP.DAP,P.OPCD+CMPACK   ;SET OPCODE
004072  012767  000074  174172          MOV     #60.,RSPLEN             ;SET LENGTHS
004100  012767  000074  174250          MOV     #60.,CMPLEN             ;
004106  000465                          BR      SEND                    ;SEND THE PACKET

                                        ;*****************************************************************************
                                        ;                                                                            ;
                                        ;       AVAILABLE PACKET                                                     ;
                                        ;                                                                            ;
                                        ;       SET UP CODE AND MODIFIERS THEN SEND THE PACKET                       ;
                                        ;                                                                            ;
                                        ;*****************************************************************************
004110  004767  000542          AVAILM: JSR     PC,CLRPAK               ;CLEAR PACKETS
004114  012767  000010  174250          MOV     #OP.AVL,P.OPCD+CMPACK   ;SET THE OPCODE
004122  012767  000014  174142          MOV     #12.,RSPLEN             ;SET RESPONCE PACKET LENGTH
004130  000413                          BR      GTSTAA                  ;SEND THE PACKET

                                        ;*****************************************************************************
                                        ;                                                                            ;
                                        ;       GET UNIT STATUS                                                      ;
                                        ;                                                                            ;
                                        ;       SET OPCODE AND MODIFIER (FOR THEN NEXT UNIT                          ;
                                        ;               THEN SEND THE PACKET                                         ;
                                        ;                                                                            ;
                                        ;*****************************************************************************
004132  004767  000520          GTSTAT: JSR     PC,CLRPAK               ;CLEAR PACKETS
004136  012767  000003  174226          MOV     #OP.GUS,P.OPCD+CMPACK   ;SET THE OPCODE
004144  012767  000001  174222          MOV     #MD.NXU,P.MOD+CMPACK    ;CLEAR MODIFIERS
004152  012767  000060  174112          MOV     #48.,RSPLEN             ;SET RESPONCE PACKET LENGTH
004160  012767  000014  174170  GTSTAA: MOV     #12.,CMPLEN             ;SET COMMAND PACKET LENGTH
004166  000435                          BR      SEND                    ;SEND THE PACKET
```



```
                                        ;*****************************************************************************
                                        ;                                                                            ;
                                        ;       SET CONTROLLER CHARACTERISTICS                                       ;
                                        ;                                                                            ;
                                        ;       SET UP CODE AND CONTROLLER FLAG (ENABLE ATTENSION MSGS)              ;
                                        ;       CLEAR MSCP VERSION, HOST TIMEOUT, USE FRACTION,                      ;
                                        ;               AND ALL OF QUAD WORD TIME AND DATE.                          ;
                                        ;       THEN SEND PACKET                                                     ;
                                        ;                                                                            ;
                                        ;*****************************************************************************
004170
004170  004767  000462          SCC:    JSR     PC,CLRPAK               ;GO CLEAR THE COMMAND PACKET
004174  012767  000034  174154          MOV     #28.,CMPLEN             ;SET UP COMMAND PACKET LENGTH
004202  012767  000034  174062          MOV     #28.,RSPLEN             ;SET UP RESPONCE PACKET LENGTH
004210  012767  000004  174154          MOV     #OP.SCC,P.OPCD+CMPACK   ;SET THE OPCODE
004216  012767  000200  174154          MOV     #CF.AVL,P.CNTF+CMPACK   ;SET THE CONTROLLER FLAGS
                                                                        ; TO ENABLE ATTENTION MSGS
004224  000416                          BR      SEND                    ;SEND THE PACKET

                                        ;*****************************************************************************
                                        ;                                                                            ;
                                        ;       ONLINE                                                               ;
                                        ;                                                                            ;
                                        ;       SET OPCODE, MODIFIERS, UNIT ID, HOST ID                              ;
                                        ;       SHADOW UNIT, ERROR FLAGS                                             ;
                                        ;       THEN SEND PACKET                                                     ;
                                        ;                                                                            ;
                                        ;*****************************************************************************
004226  004767  000424          ONLINE: JSR     PC,CLRPAK               ;CLEAR PACKETS
004232  012767  000040  174032          MOV     #32.,RSPLEN             ;SET RESPONCE PACKET LENGTH
004240  012767  000044  174110          MOV     #36.,CMPLEN             ;SET COMMAND PACKET LENGTH
004246  012767  000011  174116          MOV     #OP.ONL,P.OPCD+CMPACK   ;SET THE OPCODE
004254  016767  174322  174140          MOV     UNITNO,P.SHUN+CMPACK    ;SHADOW UNIT = UNITNO.
                                                                        ;SEND THE PACKET
```

```
                                        ;*****************************************************************************
                                        ;                                                                           *
                                        ;       SEND - SEND A PACKET                                                *
                                        ;       INTERP - WAIT FOR AN INTERRUPT                                      *
                                        ;                                                                           *
                                        ;       SET UP THE COMMAND REFERENCE NUMBER AND UNITNO IN THE PACKET        *
                                        ;       SET OWN, CLEAR FLAG IN THE COMMAND RING (FOR UDA)                   *
                                        ;       SET OWN & FLAG IN MESSAGE RING (FOR INTERRUPTS BY UDA)              *
                                        ;       AFTER INTERRUPT, MAKE SURE THE PACKET WAS PROCESSED (NO HARD        *
                                        ;       OR SOFT ERRORS) THEN RETURN TO CYCLED.                              *
                                        ;                                                                           *
                                        ;       INPUT:  CMPACK IS FILLED EXCEPT FOR CMDREF & UNITNO                 *
                                        ;               INTERRUPT VECTOR AND BR LEVEL ARE ESTABLISHED               *
                                        ;                                                                           *
                                        ;       OUTPUT: MSPACK IS FILLED                                            *
                                        ;               CLEAR CARRY IF COMMAND PACKET WAS OK                        *
                                        ;                       ELSE GO DO A HARD/SOFT ERROR                        *
                                        ;                                                                           *
                                        ;*****************************************************************************
20 004262  005267  174002          SEND:   INC     CMDREF                  ;NEW COMMAND REFERENCE NUMBER
21 004266  001775                          BEQ     SEND                    ;COMMAND REF # CANNOT = 0
22 004270  016767  173774  174064          MOV     CMDREF,P.CRF+CMPACK     ;SET COMMAND REF NUMBER
23 004276  016767  174300  174062          MOV     UNITNO,P.UNIT+CMPACK    ;SET UNIT NUMBER
24 004304  042767  040000  173754          BIC     #RG.FLG,COMMND+2        ;CLEAR FLAG
25 004312  052767  100000  173746          BIS     #RG.OWN,COMMND+2        ;SET OWN FOR COMMAND RING
26 004320  052767  140000  173734          BIS     #<RG.OWN+RG.FLG>,RSPUNC+2  ;SET OWN AND FLAG FOR MESSAGE RING
27 004326  005777  173454                  TST     @ADDR                   ;FORCE POLLING TO PACKET
28 004332                          INTERP:
29 004332  104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
30                                      ;
31 004336                          INTRUPT:
32                                      ;-------------------------------------------------------------------
   004336  000004  000000' 004344'      PIRQ$,BEGIN,1$          ; QUEUE UP TO CONTINUE AT 1$ AND RTI
                                        ;-------------------------------------------------------------------
33                                      ;
34 004344                          1$:
35 004344  005067  173706                  CLR     @INTR                   ;CLEAR INTERRUPT FLAG
36 004350  022767  000100  173730          CMP     #UP.AVA,P.OPCD+RSPACK   ;WAS AN AVAILABLE ATTENTION RECIEVED?
37 004356  001524                          BEQ     1$                      ;IF IT WAS, EXIT
38 004360  022767  000102  173720          CMP     #UP.ACP,P.OPCD+RSPACK   ;WAS THE ACCESS PATH ATTENTION RECIEVED?
39 004366  001527                          BEQ     1$                      ;IF IT WAS, GO PROCESS
40                                                                          ; ELSE CHECK SUCCESS
41 004370                          2$:
42 004370  016700  173714                  MOV     P.STS+RSPACK,R0         ; SUCCESS?
43 004374  001513                          BEQ     14$                     ;IF YES, EXIT
44 004376  042700  177740                  BIC     #177740,R0              ;CLEAR UPPER 11 BITS OF SUB-STATUS
45 004402  001510                          BEQ     14$                     ;IF SUCCESS = 0, EXIT OK
46 004404  005067  173476                  CLR     ERRTYP                  ;IF GOT HERE, ERROR
47 004410  122700  000013                  CMPB    #ST.DRV,R0              ; DRIVE ERROR?
48 004414  001015                          BNE     3$                      ;IF NOT NEXT TEST
49 004416  032767  001000  173372          BIT     #SR.DUA,SR1             ;ARE WE DUAL PORTING?
50 004424  001472                          BEQ     12$                     ;IF NOT, GO REPORT ERROR/ELSE EXPECTED
51 004426  022767  000053  173654          CMP     #<ST.DRV+SC.STO>,P.STS+RSPACK   ;IS IT AN SDI RESPONCE TIMEOUT?
52 004434  001464                          BEQ     10$                     ;IF TRUE, DRIVE IS NOT ONLINE, EXIT
53 004436  022767  000113  173644          CMP     #<ST.DRV+SC.INV>,P.STS+RSPACK   ;IS IT THE INVALID SDI RESPONCE?
54 004444  001460                          BEQ     10$                     ;IF TRUE, DRIVE IS NOT ONLINE, EXIT
55 004446  000461                          BR      12$                     ;ELSE HARD ERROR
```



```
56 004450                          3$:
57 004450  122700  000012                  CMPB    #ST.CNT,R0              ; CONTROLLER ERROR?
58 004454  001004                          BNE     4$                      ;IF NOT NEXT TEST
59 004456  012767  000003  173422          MOV     #ERR.3,ERRTYP           ;ELSE, SET ERROR TYPE
60 004464  000531                          BR      ERRORH                  ;AND HARD ERROR
61 004466                          4$:
62 004466  122700  000011                  CMPB    #ST.HST,R0              ; HOST BUFFER ACCESS ERROR?
63 004472  001004                          BNE     5$                      ;IF NOT NEXT TEST
64 004474  012767  000032  173404          MOV     #ERR.32,ERRTYP          ;ELSE, SET ERROR TYPE
65 004502  000522                          BR      ERRORH                  ;AND HARD ERROR
66 004504                          5$:
67 004504  122700  000010                  CMPB    #ST.DAT,R0              ; DATA ERROR?
68 004510  001004                          BNE     6$                      ;IF NOT NEXT TEST
69 004512  012767  000001  173366          MOV     #ERR.1,ERRTYP           ;ELSE, SET ERROR TYPE
70 004520  000533                          BR      ERRORS                  ;AND SOFT ERROR
71 004522                          6$:
72 004522  122700  000006                  CMPB    #ST.WPR,R0              ; WRITE PROTECTED?
73 004526  001431                          BEQ     12$                     ;ELSE HARD ERROR
74 004530                          8$:
75 004530  122700  000004                  CMPB    #ST.AVL,R0              ; STILL AVAILABLE?
76 004534  001005                          BNE     9$                      ;IF NOT NEXT TEST
77 004536  022767  000003  173626          CMP     #UP.GUS,P.OPCD+CMPACK   ;ELSE, IF COMMAND WAS
78                                                                          ;       GET UNIT STATUS
79 004544  001427                          BEQ     14$                     ;THEN EXPECTED & LEAVE ROUTINE
80 004546  000421                          BR      12$                     ;ELSE HARD ERROR
81 004550                          9$:
82 004550  122700  000003                  CMPB    #ST.OFL,R0              ; UNIT OFFLINE?
83 004554  001022                          BNE     13$                     ;IF NOT NEXT TEST
84                                      ; *** OFFLINE WHEN TRIED ONLINE OR GET UNIT STATUS
85 004556  022767  000011  173606          CMP     #UP.ONL,P.OPCD+CMPACK   ;WAS IT AN ONLINE COMMAND?
86 004564  001410                          BEQ     10$                     ;IF SO, SET CARRY/EXIT
87 004566  022767  000003  173576          CMP     #UP.GUS,P.OPCD+CMPACK   ;IS IT GET UNIT STATUS COMMAND?
88 004574  001404                          BEQ     10$                     ;IF SO, SET CARRY/EXIT
89 004576  022767  000042  173566          CMP     #UP.WR,P.OPCD+CMPACK    ;IS IT WRITE COMMAND?
90 004604  001002                          BNE     12$                     ;IF NOT, REPORT HARD ERROR
91 004606  000261                  10$:    SEC                             ;ELSE, SET CARRY TO
92 004610  000207                          RTS     PC                      ;AND RETURN TO DROP DRIVE/AWAIT AVAILABLE DRIVE
93                                      ; *** HARD ERROR EXIT WITH ERROR TYPE = 6
94 004612                          12$:
95 004612  012767  000006  173606          MOV     #ERR.6,ERRTYP           ;ELSE, SET ERROR TYPE
96 004620  000453                          BR      ERRORH                  ;AND HARD ERROR
97                                      ; *** SOFT ERROR EXIT WITH ERROR TYPE = 0
98 004622                          13$:
99 004622  000472                          BR      ERRORS                  ;ERROR WITH ERRTYP = 0 & IS A SOFT ERROR
100                                                                         ;  ST.CMP,ST.MFE,.ST.ABO,ST.CMD
101                                     ; *** SUCESSFUL EXIT
102 004624                         14$:
103 004624  000241                          CLC                             ;CLEAR CARRY 'CAUSE PACKET IS OK
104 004626  000207                          RTS     PC                      ;ELSE, OK, SO FAR.
105 004630                         15$:
106                                     ; *** WAIT FOR ATTENTION INTERRUPT
107                                     ; *** DID WE GET AN AVAILABLE ATTENTION MESSAGE THAT WE EXPECTED?
108 004630  005767  173630                  TST     EXPAV
109 004634  001004                          BNE     16$                     ;IF EXPAV IS NOT 0, WE GOT ONE WE DIDN'T EXPECT
110 004636  012767  177777  173620          MOV     #177777,EXPAV           ;CLEAR EXPECTED AVAILABLE ATTENTION MESSAGE WORD
111 004644  000767                          BR      14$                     ; AND RETURN
112 004646                         16$:
```

```
113 004646  052767  140000  173406          BIS     #<RG.OWN+RG.FLG>,RSPUNC+2       ;
114 004654  000626                          BR      INTERP                  ;WAIT FOR RESPONCE OF LAST PACKET SENT
115
```



```
 1                                      ;*********************************************************************
 2                                      ;                                                                    *
 3                                      ;       CLEAR PACKETS                                                *
 4                                      ;                                                                    *
 5                                      ;       ASSUMPTION:  1)  RESPONCE BUFFER PRECEDES THE COMMAND BUFFER *
 6                                      ;                    2)  TWO WORDS BEFORE EACH BUFFER IS FOR LENGTH  *
 7                                      ;                        OF PACKET AND VIRTUAL CIRCUIT               *
 8                                      ;       OUTPUT: R2 = 0 WHEN DONE                                     *
 9                                      ;               R5 = END OF COMMAND PACKET WHEN DONE                 *
10                                      ;                                                                    *
11                                      ;*********************************************************************
12 004656                              CLRPAK:
13 004656  017767  173370  173570              MOV     @UDASA,NUM              ;IS UDASA NOT ZERO/STORE IN NUM IF TRUE
14 004664  001421                            BEQ     5$                      ;IF UDASA IS ZERO, CLEAR PACKETS
15                                            ;*************************************************************
                                                      ;CONVERT NUM TO ASCII AND
                                                      ;STORE AT ADR1
   004666  104420  000000' 000454'            OTOA$,BEGIN,NUM,ADR1
   004674  000466'
                                              ;*************************************************************
16 004676  104403  000000' 005336'            MSG$S,BEGIN,SANOTO      ;ASCII MESSAGE CALL WITH COMMON HEADER
17 004704  010346                            MOV     R3,-(SP)                ;SAVE R3
18 004706  010446                            MOV     R4,-(SP)                ;SAVE R4
19 004710  004767  174530                    JSR     PC,INITUD               ;RE INIT UDASA
20 004714  012603                            MOV     (SP)+,R3                ;RESTORE R3
21 004716  012604                            MOV     (SP)+,R4                ;RESTORE R4
22 004720  004767  177244                    JSR     PC,SCC                  ;SET CONTROLLER CHARS AGAIN
23 004724  005267  173114                    INC     HRDCNT                  ;INCREMENT HARD ERROR COUNT
24                                                                            ;DOING THIS WILL CAUSE ANOTHER CALL TO CLRPAK
25 004730  012702  000064            5$:     MOV     #52.,R2                 ;R2 = # OF WORDS TO CLEAR
26 004734  012705  000272'                    MOV     #RSPLEN,R5              ;R5 -> RSPLEN, 1ST WORD TO CLEAR
27 004740  005025                    6$:     CLR     (R5)+                   ;CLEAR WORD
28 004742  005302                            DEC     R2                      ;R2 = ZERO? (DONE CONDITION)
29 004744  001375                            BNE     6$                      ;IF NOT ZERO, LOOP
30 004746  000207                            RTS     PC                      ;RETURN
31
32                                      ;*********************************************************************
33                                      ;                                                                    *
34                                      ;       HARD ERROR      CARRY WILL BE SET                            *
35                                      ;                                                                    *
36                                      ;*********************************************************************
37 004750                              ERRORH:
38 004750  032767  000004  173040            BIT     #SR.REP,SR1             ;DO WE REPORT THE ERROR?
39 004756  001403                            BEQ     7$                      ;IF SO, REPORT
40 004760  005267  173060                    INC     HRDCNT                  ;ELSE, INCREMENT THE HARD ERROR
41                                                                            ; COUNT IF NOT REPORTED
42 004764  000407                            BR      8$                      ;SKIP REPORT
43 004766  004767  000056            7$:     JSR     PC,SETTAB               ;SET UP TABLE
44                                            ;*************************************************************
   004772  104405  000000' 000000            HRDER$,BEGIN,NULL       ;
                                              ;*************************************************************
45 005000  004767  000070                    JSR     PC,PRINTE
46 005004  000261                    8$:     SEC
47 005006  000207                            RTS     PC                      ;RETURN TO CYCLED
48
49
50                                      ;                                                                    *
```

```
                                        ;       SOFT ERROR      CARRY WILL BE SET                                       ;
                                        ;                                                                               ;
                                        ;*******************************************************************************
                                        ERRORS:
005010                                  BIT     #SR.REP,SR1             ;DO WE REPORT THE ERROR?
005010  032767  000004  173000          BEQ     9$                      ;IF SO, REPORT
005016  001403                          INC     SOFCNT                  ;ELSE, INCREMENT THE HARD ERROR
005020  005267  173016                                                  ; COUNT IF NOT REPORTED
                                        BR      10$                     ;SKIP REPORT
005024  000407
005026  004767  000016                  9$:     JSR     PC,SETTAB       ;SET UP TABLE
                                        ;*******************************************************************************
005032  104406  000000' 000000          SOFERS,BEGIN,NULL               ;
                                        ;*******************************************************************************
005040  004767  000030                  JSR     PC,PRINTE
005044  000261                          10$:    SEC                     ;SET CARRY
005046  000207                          RTS     PC                      ;RETURN TO CYCLED
                                        ;*******************************************************************************
                                        ;                                                                               ;
                                        ;       SETTAB                                                                  ;
                                        ;                                                                               ;
                                        ;       SET UP A TABLE OF VALUES FOR A SOFT OR HARD ERROR                       ;
                                        ;                                                                               ;
                                        ;*******************************************************************************
005050                                  SETTAB:
005050  016767  172732  173022          MOV     ADDR,CSRA               ;SET UP CONTROL STATUS REG REPORT
005056  016767  173226  173020          MOV     P.STS+RSPACK,ASTAT      ;SET UP STATUS
005064  017767  173162  173010          MOV     @UDASA,ACSR             ;REPORT WHAT IS STATUS REG
005072  000207                          RTS     PC
                                        ;*******************************************************************************
                                        ;
                                        ;       PRINT EXTENDED ERROR MESSAGE
                                        ;
                                        ;       PRINT STATUS, OPCODE, UNIT NUMBER, BYTE COUNT, LBN AND ADDRESS
                                        ;
                                        ;*******************************************************************************
005074                                  PRINTE:
                                        ;*******************************************************************************
                                                                        ;CONVERT P.STS+RSPACK TO ASCII AND
                                                                        ;STORE AT ADR1
005074  104420  000000' 000310'          OTOAS,BEGIN,P.STS+RSPACK,ADR1
005102  000466'
                                        ;*******************************************************************
                                        ;*******************************************************************************
                                                                        ;CONVERT P.OPCD+RSPACK TO ASCII AND
                                                                        ;STORE AT ADR2
005104  104420  000000' 000306'          OTOAS,BEGIN,P.OPCD+RSPACK,ADR2
005112  000475'
                                        ;*******************************************************************
                                        ;*******************************************************************************
                                                                        ;CONVERT P.UNIT+RSPACK TO ASCII AND
                                                                        ;STORE AT ADR3
005114  104420  000000' 000302'          OTOAS,BEGIN,P.UNIT+RSPACK,ADR3
005122  000504'
                                        ;*******************************************************************
                                        ;*******************************************************************************
                                                                        ;CONVERT P.BCNT+RSPACK TO ASCII AND
```



```
                                                                        ;STORE AT ADR4
005124  104420  000000' 000312'          OTOAS,BEGIN,P.BCNT+RSPACK,ADR4
005132  000513'
                                        ;*******************************************************************
                                        ;*******************************************************************************
                                                                        ;CONVERT P.LBN+2+CMPACK TO ASCII AND
                                                                        ;STORE AT ADR5
005134  104420  000000' 000420'          OTOAS,BEGIN,P.LBN+2+CMPACK,ADR5
005142  000522'
                                        ;*******************************************************************
                                        ;*******************************************************************************
                                                                        ;CONVERT P.LBN+CMPACK TO ASCII AND
                                                                        ;STORE AT ADR6
005144  104420  000000' 000416'          OTOAS,BEGIN,P.LBN+CMPACK,ADR6
005152  000531'
                                        ;*******************************************************************
                                        ;*******************************************************************************
                                                                        ;CONVERT P.ADEA+CMPACK TO ASCII AND
                                                                        ;STORE AT ADR7
005154  104420  000000' 000404'          OTOAS,BEGIN,P.ADEA+CMPACK,ADR7
005162  000540'
                                        ;*******************************************************************
                                        ;*******************************************************************************
                                                                        ;CONVERT P.ADPA+CMPACK TO ASCII AND
                                                                        ;STORE AT ADR8
005164  104420  000000' 000402'          OTOAS,BEGIN,P.ADPA+CMPACK,ADR8
005172  000547'
                                        ;*******************************************************************
005174  104403  000000' 005400'          MSGNS,BEGIN,BANNER              ;ASCII MESSAGE CALL WITH COMMON HEADER
005202  000207                          RTS     PC
```

```
                                        .SBTTL  MODULE MESSAGES
005204  005446'                INITE1: MSG2
005206  005502'                        MSG4
005210  177777                        177777

005212  005470'                INITER: MSG3
005214  000466'                        ADR1
005216  005662'                        MSG10
005220  000475'                        ADR2
005222  006010'                        MSG14
005224  000504'                        ADR3
005226  177777                        177777

005230  005446'                INITE2: MSG2
005232  005527'                        MSG5
005234  177777                        177777

005236  005446'                INITE3: MSG2
005240  005545'                        MSG6
005242  177777                        177777

005244  005637'                DRP1:   MSG8
005246  000475'                        ADR2
005250  005647'                        MSG9
005252  006215'                        MSG20
005254  000466'                        ADR1
005256  006702'                        MSGD1
005260  177777                        177777

005262  005637'                DRP2:   MSG8
005264  000475'                        ADR2
005266  005647'                        MSG9
005270  006215'                        MSG20
005272  000466'                        ADR1
005274  006746'                        MSGD2
005276  177777                        177777

005300  005637'                DRP3:   MSG8
005302  000475'                        ADR2
005304  005647'                        MSG9
005306  006215'                        MSG20
005310  000466'                        ADR1
005312  007010'                        MSGD3
005314  177777                        177777

005316  005674'                ERRPAS: MSG11
005320  000475'                        ADR2
005322  005720'                        MSG12
005324  000504'                        ADR3
005326  005756'                        MSG13
005330  000466'                        ADR1
005332  005444'                        MSG1
005334  177777                        177777

005336  006072'                SANOTU: MSG17
005340  000466'                        ADR1
005342  006121'                        MSG18
```



```
005344  177777                        177777

005346  006021'                UNIOFF: MSG16
005350  000466'                        ADR1
005352  177777                        177777

005354  006526'                WARN1:  MSG40
005356  177777                        177777

005360  006403'                WARN2:  MSG37
005362  177777                        177777

005364  006334'                WARN3:  MSG36
005366  177777                        177777

005370  005602'                ABORT:  MSG7
005372  177777                        177777

005374  006165'                ZERO:   MSG19
005376  177777                        177777

005400  006240'                BANNER: MSG21
005402  000466'                        ADR1
005404  006332'                        MSG23
005406  000475'                        ADR2
005410  006332'                        MSG23
005412  000504'                        ADR3
005414  006332'                        MSG23
005416  000513'                        ADR4
005420  006332'                        MSG23
005422  000522'                        ADR5
005424  006332'                        MSG23
005426  000531'                        ADR6
005430  006332'                        MSG23
005432  000540'                        ADR7
005434  006332'                        MSG23
005436  000547'                        ADR8
005440  005444'                        MSG1
005442  177777                        177777
```

```
                                        .SBTTL  MORE MODULE MESSAGES
                                        .NLIST  BEX

005444  045  000          MSG1:   .ASCIZ  '%'
005446  045  125  104     MSG2:   .ASCIZ  '%UDA INIT ERROR, '
005470  045  125  104     MSG3:   .ASCIZ  '%UDASA = '
005502  106  117  125     MSG4:   .ASCIZ  'FOUND BY DIAGNOSTIC '
005527  123  124  105     MSG5:   .ASCIZ  'STEP NOT SET.'
005545  105  130  120     MSG6:   .ASCIZ  'EXPECTED DATA WAS INCORRECT '
005602  045  122  105     MSG7:   .ASCIZ  '%RETRY COUNT EXCEEDED, ABORT'
005637  045  104  122     MSG8:   .ASCIZ  '%DRIVE '
005647  040  104  122     MSG9:   .ASCIZ  ' DROPPED. '
005662  040  111  116     MSG10:  .ASCIZ  ' IN STEP '
005674  045  123  117     MSG11:  .ASCIZ  '%SOFT ERROR COUNT #'
005720  040  040  040     MSG12:  .ASCIZ  '          *** HARD ERROR COUNT #'
005756  045  103  110     MSG13:  .ASCIZ  '%CHECK DATA ERROR COUNT #'
006010  045  101  104     MSG14:  .ASCIZ  '%ADDR = '
006021  045  125  116     MSG15:  .ASCIZ  '%UNIT WAS FOUND OFFLINE.  UNIT NUMBER = '
006072  045  125  104     MSG17:  .ASCIZ  '%UDASA IS NOT ZERO, = '
006121  045  125  104     MSG18:  .ASCIZ  '%UDA IS GOING THROUGH INITIALIZATION'
006165  045  122  111     MSG19:  .ASCIZ  '%RING AREA NOT CLEARED%'
006215  045  104  105     MSG20:  .ASCIZ  '%DEVICE ID BIT = '
006240  045  123  124     MSG21:  .ASCIZ  '%STATUS ENDCOD UNITNO BYTECO HI LBN LO LBN EXTADR PHYADR%'
006332  040  000          MSG23:  .ASCIZ  '
006334  040  041  040     MSG36:  .ASCIZ  ' ! OPERATING WITH NO DISK ACCESSING !%'
006403  007  007  040     MSG37:  .ASCII  <07><07>' ! CUSTOMER DATA WILL BE OVERWRITTEN !%'
006454  040  055  055                     .ASCIZ  ' ----------------------------------%'<07><07>
006526  040  111  106     MSG40:  .ASCII  ' IF YOU WISH TO DESTROY CUSTOMER DATA, SET BIT1 (NOT BIT0)%'
006621  040  111  116                     .ASCIZ  ' IN SWITCH REGISTER 1(SW1) OF UDA?? EQUAL TO 1.%'
006702  045  105  122     MSGD1:  .ASCIZ  '%ERRORS CAUSED DRIVE TO BE DROPPED%'
006746  045  125  116     MSGD2:  .ASCIZ  '%UNIT WAS NOT FOUND BY EXERCISER%'
007010  045  104  126     MSGD3:  .ASCIZ  '%DVID1 BIT SET HIGHER THAN ACTUAL # OF DRIVES FOUND%'

                                        .EVEN
007076                  RBUF:   .BLKW   256.            ;THE READ BUFFER
        000001                  .END
```



```
ABORT   005370R         CF.576= 000001          MC.CMD= 000004          MD.WBV= 000040          UP.ONL= 000011
ACSR    000102R         CINTR   000254R         MC.CPK= 000356R         MUDNAM  000000R         UP.RD = 000041
ADDR    000006R         CLRPAK  004656R         MC.RCT= 000002          MUDSP   000252R         UP.RPL= 000024
ADDR22= 001000          CMDREF  000270R         MC.RES= 000000          MSGD1   006702R         UP.SCC= 000004
ADR1    000466R         CMPACK  000362R         MC.RPK= 000276R         MSGD2   006746R         UP.SHC= 000102
ADR2    000475R         CMPLEN  000356R         MC.SIZ= 000010          MSGD3   007010R         UP.SUC= 000012
ADR3    000504R         CMPVIR  000360R         MRDCNT  000044R         MSGNS = 104403          UP.WK = 000042
ADR4    000513R         COMMND  000264R         MRDERS= 104405          MSGSS = 104402          UTOAS = 104420
ADR5    000522R         CONFIG  000056R         MRDPAS  000050R         MSGS  = 104401          VA      000444R
ADR6    000531R         CSRA    000100R         ICOUNT  000036R         MSG1    005444R         VASCNT  000034R
ADR7    000540R         CYCLEN  002620R         ICOUNT  000040R         MSG10   005662R         VAIR    000556R
ADRM    000547R         CYCLEL  003116R         IDNUM   000122R         MSG11   005674R         VA22    000562R
ASR     000106R         DAP     004060R         IMUDX.= 000000          MSG12   005720R         VICKHK  003150R
ASR04   002030R         DATCKS= 104411          INIT    000030R         MSG13   005756R         VIRWS = 000004
ASTAT   000104R         DATERS= 104404          INITER  005212R         MSG14   006010R         VKTSIZ= 000060
AVAILB  004110R         DDINTR  003100R         INITE1  005204R         MSG16   006021R         VUPSP = 005726
AWAS    000110R         DROP1   003404R         INITE2  005230R         MSG17   006072R         VUPSP2= 022626
BANNER  005400R         DROP2   003414R         INITE3  005236R         MSG18   006121R         VURIID  000606R
BEGIN   000000R         DROP3   003424R         INITUD  001444R         MSG19   006165R         VRINIE  005674R
BIT0  = 000001          DROP4   003432R         INTA    002066R         MSG2    005446R         VRMMSG  000462R
BIT00 = 000001          DRP1    005244R         INTERP  004332R         MSG20   006215R         VRTNUM= 000017
BIT01 = 000002          DRP2    005262R         INTR    000120R         MSG21   006240R         VRTY  = 000000
BIT02 = 000004          DRP3    005300R         LIMIT   000576R         MSG23   006332R         VRTY0 = 000000
BIT03 = 000010          DVICE   000600R         LOOP1   001300R         MSG3    005470R         VRTY1 = 000040
BIT04 = 000020          DVID1   000014R         LOOP2   001330R         MSG36   006334R         VRTY2 = 000100
BIT05 = 000040          EA      000446R         L.CHVR= 000015          MSG37   006403R         VRTY3 = 000140
BIT06 = 000100          EA22    000564R         L.CNTI= 000011          MSG4    005502R         VRTY4 = 000200
BIT07 = 000200          EF.HMR= 000200          L.CYL = 000034          MSG40   006526R         VRTY5 = 000240
BIT08 = 000400          EF.BBU= 000100          L.DATA= 000050          MSG5    005527R         VRTY6 = 000300
BIT09 = 001000          EF.FRS= 000200          L.ERLC= 000030          MSG6    005545R         VRTY7 = 000340
BIT1  = 000002          EF.LOG= 000040          L.EVR1= 000000          MSG7    005602R         VS      177776
BIT10 = 002000          EF.LST= 000100          L.GRP = 000040          MSG8    005637R         VSR     177776
BIT11 = 004000          EF.MLS= 000001          L.SCTR= 000042          MSG9    005647R         VUSH    005746
BIT12 = 010000          EF.SEX= 000020          L.SUUT= 000002          NIKUPT  004336R         VUSHZ = 024646
BIT13 = 020000          ENDITS= 104413          L.THCK= 000043          NULL  = 000000          VWRFLG= 000002
BIT14 = 040000          ENDS  = 104410          L.UHVR= 000027          NUM     000454R         V.ADEA= 000022
BIT15 = 100000          ERRORH  004750R         L.UNIT= 000016          OLDEA   000460R         V.ADPA= 000020
BIT2  = 000004          ERRORS  005010R         L.USVR= 000026          OLDPA   000456R         V.BCNT= 000014
BIT3  = 000010          ERROR1  002144R         L.VSPR= 000044          ONEFIL= 000001          V.BUFF= 000020
BIT4  = 000020          ERROR2  002142R         MAITP   003662R         ONLINE  004226R         V.CMST= 000020
BIT5  = 000040          ERROR3  002140R         MAITR   003556R         OPEN  = 000000          V.CNCL= 000022
BIT6  = 000100          ERROR5  002126R         MAITW   003612R         OP.ABO= 000031          V.CNT = 000006
BIT7  = 000200          ERRPAS  005316R         MAP22S= 104416          OP.ACC= 000020          V.CNIF= 000016
BIT8  = 000400          ERRTYP  000106R         MA18NC  001412R         OP.ACP= 000102          V.CNTI= 000024
BIT9  = 001000          ERR.0 = 000000          MD.CMP= 040000          OP.AVA= 000100          V.CPSP= 000042
BREAKS= 104407          ERR.1 = 000001          MD.ERR= 010000          OP.AVL= 000010          V.CRF = 000000
BRI     000012R         ERR.3 = 000003          MD.EXP= 100000          OP.CCD= 000021          V.CTMO= 000020
BR2     000013R         ERR.32= 000032          MD.FEU= 000001          OP.CMP= 000040          V.CYL = 000050
BTOD$ = 104421          ERR.6 = 000006          MD.HXU= 000001          OP.DAP= 000013          V.ELGF= 000034
CDATA$= 104412          EXIT$ = 104400          MD.SCH= 004000          OP.END= 000200          V.FBBK= 000034
CDERCT  000134R         EXPAV   000464R         MD.SCL= 002000          OP.ERS= 000022          V.FLGS= 000011
CDMJCT  000146R         FREE    000150R         MD.SEC= 001000          OP.FLU= 000023          V.GRP = 000046
CF.AVL= 000200          GEIPA$= 104415          MD.SER= 000400          OP.GCS= 000002          V.HSI1= 000020
CF.MSC= 000100          GFSTAA  004160R         MD.SPD= 000001          OP.GUS= 000003          V.HIMU= 000020
CF.OTH= 000040          GFSTAT  004132R         MD.SSH= 002000          OP.MPD= 000003          V.LBN = 000036
CF.SHD= 000002          GMRUF$= 104414          MD.VUL= 000002          OP.MPR= 000031          V.LGDI= 000014
CF.THS= 000020          HC.CCI= 000006          MD.WBN= 000100          OP.MWR= 000031          V.MEDI= 000034
```

SYMBOL TABLE

```
P.MLUN= 000014          READA   004020R         SA.S3 = 020000          START   000660R         UF.CMW= 000002
P.MUD = 000012          RESTRT  001036R         SA.S4 = 040000          STAT    000026R         UF.INA= 040000
P.UPCB= 000010          RESTR1  001116R         SA.VCE= 000177          ST.ABO= 000002          UF.RAV= 000200
P.UTRF= 000014          RESTR2  001072R         SA.VFC= 000177          ST.AVL= 000004          UF.RPL= 010000
P.RBN = 000014          RES1    000056R         SBADR   000102R         ST.CMD= 000001          UF.SCH= 004000
P.RBNS= 000056          RES2    000060R         SCC     004170R         ST.CMP= 000007          UF.SCL= 002000
P.RCTC= 000057          RG.FLG= 040000          SC.DIS= 000400          SI.CNT= 000012          UF.WBH= 000040
P.RCTS= 000054          RG.OWN= 100000          SC.DUP= 000200          SI.DAT= 000010          UF.WPH= 020000
P.RGID= 000034          RINTR   000256R         SC.INV= 000100          SI.DIA= 000037          UF.WPS= 010000
P.RGOF= 000040          RLIM  = 000004          SC.IOP= 000100          ST.DRV= 000013          UF.576= 000004
P.SFTM= 000040          RSPACK  000276R         SC.NVL= 000040          SI.HST= 000011          UNIOFF  005346R
P.SHST= 000042          RSPLEN  000272R         SC.STU= 000040          ST.MBE= 000005          UNITFL  000610R
P.SHUN= 000040          RSPONC  000260R         SECH    000570R         ST.MSK= 000037          UNITNO  000602R
P.STS = 000012          RSPVIR  000274R         SECL    000566R         SI.OFL= 000063          UNSZH   000574R
P.SZOF= 000012          RSTMT   000112R         SEND    004262R         SI.SUB= 000040          UNSZL   000572R
P.TIME= 000024          R6      =%000006        SEQACC  003336R         SI.SUC= 000000          VA      000442R
P.TRCK= 000044          R7      =%000007        SETTAB  005050R         ST.WPR= 000006          VECTOR  000010R
P.UNFL= 000016          SANOTO  005336R         SETUP   002314R         SVR0    000062R         WARN1   005354R
P.UNIT= 000004          SA.CMD= 170000          SNDSTP  002042R         SVR1    000064R         WARN2   005360R
P.UNSZ= 000044          SA.CME= 000360          SUFCNT  000042R         SVR2    000066R         WARN3   005364R
P.UNTI= 000024          SA.CTP= 003400          SUFFAS= 104406          SVR3    000070R         WASADR  000104R
P.USEF= 000022          SA.DIA= 000400          SUFPAS  000046R         SVR4    000072R         WBFFEA  000452R
P.VRSN= 000014          SA.ERC= 003777          SPOINT  000032R         SVR5    000074R         WBUFEA  000136R
P.VSER= 000050          SA.ERP= 100000          SPSIZ = 000040          SVR6    000076R         WBUFPA  000134R
RANACC  003160R         SA.GO = 000001          SR.CMP= 004000          SYSCNT  000052R         WBUFRQ  000140R
RAND$ = 104417          SA.INE= 000200          SR.DUA= 001000          TABLEW  000614R         WBUFSZ  000142R
RANNUM  000054R         SA.INI= 000200          SR.REP= 000004          TEND    000654R         WDFR    000116R
RBFFEA  000450R         SA.LFC= 040000          SR.SEN= 002000          TIMER = 002260          WDTO    000114R
RBUF    007076R         SA.MCV= 000377          SR.SUM= 000010          TIMOUT= 005670          WDRK    000612R
RBUFEA  000130R         SA.PRG= 000001          SR.XFR= 000002          TRPDFD= 000022          WRITE   003732R
RBUFPA  000126R         SA.RSE= 000017          SR1     000016R         TRY     000604R         WRITEA  003744R
RBUFSZ  000132R         SA.RSP= 007400          SR2     000020R         TSTUFL  002550R         XFLAG   000005R
RBUFVA  000124R         SA.S1 = 004000          SR3     000022R         UDASA   000252R         XMEM    000560R
READ    003766R         SA.S2 = 010000          SR4     000024R         UF.CMR= 000001          ZERO    005374R
```

```
. ABS.  000000      000
        010076      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  12800 WORDS  ( 50 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  71 PAGES
,A:XDUBC0.LST/C=DDXCOM,XDUBC0.DUC,XDUBC0
```


CROSS REFERENCE TABLE (CREF V04.00)

```
ABORT   19-55   30-73#
ACSR     6-3#   29-76*
ADDR     6-3#   16-68   18-26*  19-48   28-27   29-74
ADDR22   6-3#
ADR1     6-60#  17-34   17-35*  19-34   23-33   29-15   29-86   30-7    30-26   30-34   30-42   30-51   30-56   30-61
        30-80
ADR2     6-62#  17-30   17-31*  19-32   23-31   23-32*  29-87   30-9    30-23   30-31   30-39   30-47   30-82
ADR3     6-64#  17-32   17-33*  19-48   29-88   30-11   30-49   30-84
ADR4     6-66#  29-89   30-86
ADR5     6-68#  29-90   30-88
ADR6     6-70#  29-91   30-90
ADR7     6-72#  29-92   30-92
ADR8     6-74#  29-93   30-94
ASB      6-3#
ASR04   17-41   17-59   17-90   18-63   18-81   18-88   18-108#
ASTAT    6-3#   29-75*
AVAILB  21-89   26-21#
AWAS     6-3#
BANNER  29-94   30-79#
BEGIN    6-3#   16-50   16-53   16-56   16-65   17-15   17-30   17-32   17-34   17-36   17-39   17-55   17-57   17-80
        17-88   17-93   18-23   18-31   18-31   18-78   18-85   18-131  18-134  19-25   19-26   19-32   19-34   19-38
        19-42   19-46   19-48   19-49   19-50   19-55   19-56   21-85   21-115  22-15   22-17   23-31   23-33   23-37
        23-41   23-43   28-29   28-32   29-15   29-16   29-44   29-61   29-86   29-87   29-88   29-89   29-90   29-91
        29-92   29-93   29-94
BIT0     6-3#    9-53
BIT00    8-3#
BIT01    6-14    8-4#
BIT02    6-15    8-5#
BIT03    6-16    8-6#
BIT04    8-7#
BIT05    8-8#
BIT06    8-9#
BIT07   8-10#   18-50   18-52
BIT08   8-11#
BIT09    6-17    8-12#
BIT1     6-3#
BIT10    6-3#    6-18     8-13#
BIT11    6-3#    6-19     8-14#
BIT12    6-3#    8-15#
BIT13    6-3#    8-16#
BIT14    6-3#    8-17#   10-4
BIT15    6-3#    8-18#   10-3    18-44   18-49
BIT2     6-3#
BIT3     6-3#
BIT4     6-3#
BIT5     6-3#
BIT6     6-3#
BIT7     6-3#
BIT8     6-3#
BIT9     6-3#
BR1      6-3#   18-128
BR2      6-3#
BREAK$   6-3#   18-31   18-31
BTUD$    6-3#   17-30   17-32   17-34   23-31
CDATA$   6-3#   21-85   21-115
CDERCT   6-3#   16-58*  17-34
```

```
CDWDCT    6-3#
CF.576   13-8#
CF.AVL   13-3#     27-16
CF.MSC   13-4#
CF.OTH   13-5#
CF.SHD   13-7#
CF.THS   13-6#
CINTR     6-24#
CLRPAK   24-13     24-33     25-12     25-30     26-8      26-21     26-34     27-12     27-29     29-12#
CMDREF    6-30#    16-64*    18-69     20-19#    24-20*    28-22
CMPACK    6-38#    18-77     24-14*    24-15*    24-16*    24-34*    24-35*    24-36*    24-43*    24-46*    25-13*    25-15*    25-16*    25-31*
         25-33#    25-34*    25-38#    25-39*    25-40*    26-9*     26-22*    26-35*    26-36*    27-15*    27-16*    27-32*    27-33*    28-22*
         28-23*    28-77     28-85     28-87     28-89     29-90     29-91     29-92     29-93
CMPLEN    6-36#    24-45*    25-36*    26-11*    26-38*    27-13*    27-31*
CMPVIR    6-37#    24-47*
COMMND    6-28#     6-40     18-79*    18-82*    28-24*    28-25*
CONFIG    6-3#
CSRA      6-3#     29-74*
CYCLED   17-69     21-18#    21-48
CYCLEL   17-92     21-110#
DAP      26-8#
DATCK$    6-3#
DATER$    6-3#
DUINTR   21-97#
DRUP1    17-71     23-16#
DROP2    20-34     23-19#
DROP3    20-52     23-22#
DROP4    23-18     23-21     23-24#
DRP1     23-37     30-22#
DRP2     23-41     30-30#
DRP3     23-43     30-38#
DVICE     7-18#    16-61*    17-52     17-64     17-77     20-61     23-24     23-30*
DVID1     6-3#     16-61
EA        6-42#    17-18     17-21     18-62     18-80     18-87
EA22      7-8#
EF.BBR   12-20#
EF.BBU   12-21#
EF.FRS   15-3#
EF.LOG   12-22#
EF.LST   15-4#
EF.MIS   15-5#
EF.SEX   12-23#
END$      6-3#     17-55     19-26     19-56
ENDITS    6-3#     17-80     17-93
ERR.0     8-23#
ERR.1     8-24#    28-69
ERR.3     8-25#    28-59
ERR.32    8-27#    28-64
ERR.6     8-26#    28-95
ERROR1   18-36     18-138    19-30#
ERROR2   18-143    19-29#
ERROR3   18-39     18-53     18-59     19-28#
ERROR5   18-68     19-25#
ERRORH   20-90     21-77     28-60     28-65     28-96     29-37#
ERRORS   28-70     28-99     29-54#
ERRPAS   17-36     30-46#
```



```
ERRTYP    6-3#     28-46*    28-59*    28-64*    28-69*    28-95*
EXIT$     6-3#     18-131    28-29
EXPAV     6-57#    16-59*    21-35*    21-98*    28-108    28-110*
FREE      6-3#
GETPAS    6-3#     17-15     17-39     18-23     18-78     18-85
GTSTAA   26-24     26-38#
GTSTAT   20-30     21-30     21-68     26-34#
GWBUF$    6-3#     17-57     17-88
HC.CCT   10-14#
HC.CMD   10-13#
HC.CPK   10-16#
HC.RCT   10-12#
HC.RES   10-11#
HC.RPK   10-15#    10-16
HC.SIZ   10-8#
HRDCNT    6-3#     17-32     29-23*    29-40*
HRDERS    6-3#     19-49     29-44
HRDPAS    6-3#
ICONT     6-3#
ICOUNT    6-3#
IDNUM     6-3#
IMODX.    6-3#     17-57     17-88
INIT      6-3#
INITE1   19-38     30-2#
INITE2   19-42     30-14#
INITE3   19-46     30-18#
INITER   19-50     30-6#
INITUD   17-22     18-22#    29-19
INTA     18-127    18-133#
INTEPP   21-37     21-100    28-28#    28-114
INTR      6-3#
L.CHVR   15-13#
L.CNTI   15-11#    15-12#
L.CYL    15-18#
L.DATA   15-23#
L.ERLC   15-17#
L.EVNT   15-9#
L.GRP    15-19#
L.SCTR   15-21#
L.SLOT   15-10#
L.TRCK   15-20#
L.UHVR   15-16#
L.UNIT   15-14#
L.USVR   15-15#
L.VSER   15-22#
LIMIT     7-16#    24-37     24-39
LOOP1    17-53     17-57#    17-81
LOOP2    17-63#    17-78
MAIONC   17-45     17-88#    17-94
MAITP    24-18     24-40     24-42#
MAITR    21-112    24-13#
MAITW    21-111    24-33#
MAP22$    6-3#
MD.CMP   12-3#
MD.ERR   12-5#
MD.EXP   12-4#
```

```
MD.FEU  12-14#
MD.NXU  12-16#    26-36
MD.SCH  12-6#
MD.SCL  12-7#
MD.SEC  12-8#
MD.SER  12-9#
MD.SPD  12-13#
MD.SSH  12-10#
MD.VOL  12-15#
MD.WBN  12-11#
MD.WBV  12-12#
MODNAM    6-3#
MODSP     6-3      6-3#
MSG$      6-3#
MSG1     30-52    30-95    31-4#
MSG10    30-8     31-13#
MSG11    30-46    31-14#
MSG12    30-48    31-15#
MSG13    30-50    31-16#
MSG14    30-10    31-17#
MSG16    30-60    31-18#
MSG17    30-55    31-19#
MSG18    30-57    31-20#
MSG19    30-76    31-21#
MSG2     30-2     30-14    30-18    31-5#
MSG20    30-26    30-33    30-41    31-22#
MSG21    30-79    31-23#
MSG23    30-81    30-83    30-85    30-87    30-89    30-91    30-93    31-24#
MSG3     30-6     31-6#
MSG36    30-70    31-25#
MSG37    30-67    31-26#
MSG4     30-3     31-7#
MSG40    30-64    31-28#
MSG5     30-15    31-8#
MSG6     30-19    31-9#
MSG7     30-73    31-10#
MSG8     30-22    30-30    30-38    31-11#
MSG9     30-24    30-32    30-40    31-12#
MSGD1    30-27    31-30#
MSGD2    30-35    31-31#
MSGD3    30-43    31-32#
MSGN$     6-3#    16-50    16-53    16-56    17-36    19-25    19-38    19-42    19-46    19-50    19-55    23-37    23-41    23-43
         29-16    29-94
MSGS$     6-3#
NTRUPT   17-37    18-90    28-31#
NULL      6-3#    19-49    29-44    29-61
NUM       6-47#   19-31*   19-32    19-33*   19-34    23-17*   23-20*   23-23*   23-35*   23-39*   29-13*   29-15
OLDEA     6-49#   16-71#   17-18    17-21*
OLDPA     6-48#   16-70*   17-16    17-20*
ONEFIL    2-4#     2-8      3-570    4-1
ONLINE   21-42    27-29#
OP.ABO   11-3#
OP.ACC   11-4#
OP.ACP   11-25#   28-38
OP.AVA   11-22#   28-36
OP.AVL   11-5#    26-22
```



```
OP.CCD   11-6#
OP.CMP   11-7#
OP.DAP   11-8#    26-9
OP.END   11-21#
OP.ERL   11-23#
OP.ERS   11-9#
OP.FLU   11-10#
OP.GCS   11-11#
OP.GUS   11-12#   26-35    28-77    28-87
OP.MRD   11-19#   24-14
OP.MWR   11-20#   24-34
OP.ONL   11-13#   27-32    28-85
OP.RD    11-14#   25-31
OP.RPL   11-15#
OP.SCC   11-16#   27-15
OP.SHC   11-24#
OP.SUC   11-17#
OP.WR    11-18#   25-13    28-89
OPEN      6-3       6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3
          6-3       6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3      6-3
          6-3#      6-41     6-42
OTUA$     6-3#     19-32    19-34    19-48    23-33    29-15    29-86    29-87    29-88    29-89    29-90    29-91    29-92    29-93
P.ADEA   13-20#    24-15*   24-35*   25-16*   25-33*   29-92
P.ADPA   13-19#    24-16*   24-36*   25-15*   25-34*   29-93
P.BCNT   13-17#    14-9#    24-43*   25-38*   29-89
P.BUFF   13-18#
P.CMST   14-15#
P.CNCL   14-44#
P.CNT    14-53#
P.CNTF   13-40#    14-42#   27-16*
P.CNTI   14-45#
P.CPSP   13-33#
P.CRF    13-13#    14-4#    14-51#   28-22*
P.CTMO   14-43#
P.CYL    14-26#
P.ELGF   13-31#
P.FBHK   14-10#
P.FLGS   14-7#     14-55#
P.GRP    14-25#
P.HSTI   13-29#    14-20#   14-34#
P.HTMO   13-41#
P.LBN    13-21#    25-39*   25-40*   29-90    29-91
P.LGDT   14-57#
P.MEDI   14-46#
P.MLUN   14-18#    14-32#
P.MOD    13-16#    26-36*
P.OPCD   13-15#    14-6#    14-54#   24-14*   24-34*   25-13*   25-31*   26-9*    26-22*   26-35*   27-15*   27-32*   28-36    28-38
         28-77     28-85    28-87    28-89    29-87
P.OTRF   13-25#    14-14#
P.RBN    13-36#
P.RBNS   14-28#
P.RCTC   14-29#
P.RCTS   14-27#
P.RGID   13-46#    24-46*
P.RGOF   13-47#
P.SFTW   13-22#    14-11#
```

```
P.SHST  14-23#  14-47#
P.SHUN  13-32#  14-22#  14-36#  27-33#
P.STS   14-8#   20-80    20-84   28-42   28-51   28-53   29-75   29-86
P.SZOF  14-56#
P.TIME  13-43#
P.TRCK  14-24#
P.UNFL  13-28#  14-19#  14-33#
P.UNIT  13-14#  14-5#   14-52#  20-36   28-23*  29-88
P.UNSZ  14-37#  21-44   21-45
P.UNTI  13-30#  14-21#  14-35#
P.USEF  13-42#
P.VRSN  13-39#  14-41#
P.VSER  14-38#
PA       6-41#  17-16   17-20   18-54   18-79   18-86
PA18     7-5#
PA22     7-7#
PASCNT   6-3#   17-27
PICHAK  21-67   22-11#
PIRQ$    6-3#   18-134  28-32
PKTSIZ  10-9#   10-16
POPSP    6-3#
POPSP2   6-3#
PORTIO   7-21#  17-66*  20-25*  20-51*  23-24   23-26   23-30   23-33
PRINTE  29-45   29-62   29-85#
PRNMSG   6-53#  16-60*  17-27   17-29*
PRTNUM   6-52#   6-53   16-60   17-29
PRTY     6-3#
PRTY0    6-3     6-3#
PRTY1    6-3#
PRTY2    6-3#
PRTY3    6-3#
PRTY4    6-3#
PRTY5    6-3     6-3#
PRTY6    6-3#
PRTY7    6-3#
PS       6-3#
PSW      6-3#
PUSH     6-3#
PUSH2    6-3#
PWRFLG   6-3#
R6       6-3#
R7       6-3#
RANACC  22-14#
RANDS    6-3#   16-65   22-15   22-17
RANNUM   6-3#   16-66   22-16   22-18
RBFFEA   6-44#  17-42*  24-15   25-33
RBUF     6-3     31-34#
RBUFEA   6-3#   17-40
RBUFPA   6-3#   21-85   21-115  24-16   25-34
RBUFSZ   6-3#   24-17   25-32
RBUFVA   6-3#   17-39
READ    21-80   25-30#
READA   25-17   25-35#
RES1     6-3#
RES2     6-3#
RESTR1  17-19   17-24#
```



```
RESTR2  17-17   17-20#
RESTRT   6-3    17-13#
RG.FLG  10-4#   21-36   21-99   28-24   28-26   28-113
RG.OWN  10-3#   21-36   21-99   28-25   28-26   28-113
RINTR    6-25#  28-35*
RLIM     7-26#  19-52
RSPACK   6-34#  10-15   18-84   20-36   20-80   20-84   21-44   21-45   28-36   28-38   28-42   28-51   28-53   29-75
        29-86   29-87   29-88   29-89
RSPLEN   6-32#  24-44*  25-35*  26-10*  26-23*  26-37*  27-14*  27-30*  29-26
RSPONC   6-27#  17-14   18-22   18-65   18-86*  18-89*  21-36*  21-99*  28-26*  28-113*
RSPVIR   6-33#  24-48*
RSTRT    6-3#
SA.CMD   9-25#
SA.CME   9-40#
SA.CTP   9-17#
SA.DIA   9-18#  18-37
SA.ERC   9-13#
SA.ERR   9-9#   18-29   18-137
SA.GO    9-53#  18-75
SA.INE   9-30#
SA.INT   9-23#  18-44
SA.LPC   9-45#
SA.MCV   9-49#
SA.PRG   9-35#
SA.RSE   9-39#
SA.RSP   9-24#
SA.S1    9-5#   18-35   18-37   18-47
SA.S2    9-6#   18-52
SA.S3    9-7#
SA.S4    9-8#
SA.VCE   9-29#
SA.VEC   9-22#
SANOTO  29-16   30-55#
SBADR    6-3#
SC.DIS  15-49#  20-80   20-84
SC.DUP  15-51#  20-80   20-84
SC.INV  15-56#  28-53
SC.IOP  15-48#  20-80   20-84
SC.NVL  15-46#  20-80   20-84
SC.STO  15-55#  28-51
SCC     20-18   27-11#  29-22
SECH     7-11#  16-67#  22-24*  22-40*  22-41   22-44*  22-49*  22-70*  22-72   22-75*  25-40
SECL     7-10#  16-66*  22-57*  22-64*  22-66   22-74*  25-39
SEND    24-49   25-41   26-12   26-39   27-18   28-20#  28-21
SEGACC  22-13   22-63#
SETTAB  29-43   29-60   29-73#
SETUP   17-51   20-16#
SNDSTP  18-48   18-55   18-64   18-126#
SOFCNT   6-3#   17-30   29-57#
SOFERS   6-3#   29-61
SOFPAS   6-3#
SPOINT   6-3#
SPSIZ    1-23#   6-3
SR.CMP   6-19#  21-82   21-113
SR.DUA   6-17#  21-19   21-75   21-86   28-49
SR.REP   6-15#  29-38   29-55
```

```
SR.SEQ    6-18#   22-12
SR.SUM    6-16#   17-25
SR.XFR    6-14#   16-49   16-51   17-44
SR1       6-3#    16-49*  16-51   17-25   17-44   21-19   21-75   21-82   21-86   21-113  22-12   28-49   29-38   29-55
SR2       6-3#    18-71
SR3       6-3#
SR4       6-3#
ST.ABO   15-31#
ST.AVL   15-33#   21-70   28-75
ST.CMD   15-30#
ST.CMP   15-36#
ST.CNT   15-39#   28-57
ST.DAT   15-37#   28-67
ST.DIA   15-41#
ST.DRV   15-40#   20-78   28-47   28-51   28-53
ST.MST   15-38#   28-62
ST.MFE   15-34#
ST.MSK   15-27#   20-84
ST.OFL   15-32#   20-76   28-82
ST.SUB   15-28#
ST.SUC   15-29#
ST.WPR   15-35#   28-72
START     6-3     16-46#  19-54
STAT      6-3#
SVR0      6-3#
SVR1      6-3#
SVR2      6-3#
SVR3      6-3#
SVR4      6-3#
SVR5      6-3#
SVR6      6-3#
SYSCNT    6-3#
TABLE=    7-28#   16-62*  16-63*  17-61   20-22   20-39
TEND      7-31#   17-75   20-41   20-65
TIMER     6-55#   18-27
TIMOUT    7-25#
TRPDFD    6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3
          6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3
          6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3
          6-3#    6-3#    6-3#    6-3#    6-3#    6-3#    6-3#    6-3#    6-3#    6-3#    6-3#    6-3#    6-3#    6-3#
TRY       7-20#   17-23*  18-91*  19-51*  19-52   21-22   21-50*
TSTOFL   20-76#   21-32
UDASA     6-22#   16-68*  16-69*  18-28   18-76*  18-129* 18-136  19-33   29-13   29-76
UF.576   12-38#
UF.CMR   12-28#
UF.CMW   12-29#
UF.INA   12-31#
UF.RMV   12-32#
UF.RPL   12-30#
UF.SCH   12-33#
UF.SCL   12-34#
UF.WBM   12-35#
UF.WPH   12-36#
UF.WPS   12-37#
UNIOFF   30-60#
UNITFL    7-22#
```



```
UNITNO    7-19#   17-43#  17-67*  20-23*  20-24   20-38*  20-45   20-50*  20-56   20-58   20-63*  23-31   27-33   28-23
UNSZH     7-14#   21-44*  21-47   22-25   22-28   22-41   22-72
UNSZL     7-13#   21-45*  21-65*  22-50   22-54   22-59   22-66
VA        6-40#   17-14*  17-15   18-22*  18-23   18-77*  18-78   18-84*  18-85
VECTOR    6-3#    17-37*  18-41   18-90*  18-126
WARN1    16-50    30-64#
WARN2    16-53    30-67#
WARN3    16-56    30-70#
WASADR    6-3#
WBFFEA    6-45#   17-60*  17-91*  24-35   25-16
WBUFEA    6-3#    17-58   17-89
WBUFPA    6-3#    24-36   25-15
WBUFRQ    6-3#
WBUFSZ    6-3#    21-60   24-37   24-41   25-14
WDFR      6-3#
WDIO      6-3#
WORK      7-23#   20-29*  20-32*
WRITE    21-73    25-12#
WRITEA   25-14#
XFLAG     6-3#
XMEM      7-6#
ZERO     19-25    30-76#
```

```
BKMOD     1-111#
BREAK     1-201#  18-31
BTOD      1-220#  17-30   17-32   17-34   23-31
CKDATA    1-256#  21-85   21-115
DATACK    1-265#
DATERR    1-154#
DFSEVN    1-288#  6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3     6-3
DSEVNT    1-298#  6-3     6-3     6-3     6-3     6-3     6-3
END       1-191#  19-26
ENDIT     1-182#  17-80   17-93
ENDMOD    1-187#  17-55   19-56
EQUATS    1-306#  6-3
EXIT      1-136#  18-131  28-29
GETPA     1-247#  17-15   17-39   18-23   18-78   18-85
GWBUFF    1-235#  17-57   17-88
HRDER     1-144#  19-49   29-44
IOMOD     1-107#
IOMODP    1-131#
IOMODR    1-127#
IOMODX    1-123#  6-3
MAP22     1-251#
MODULE    1-24#   6-3
MSG       1-170#
MSGN      1-174#  16-50   16-53   16-56   17-36   19-25   19-38   19-42   19-46   19-50   19-55   23-37   23-41   23-43
         29-16    29-94
MSGS      1-178#
NBKMOD    1-119#
OTOA      1-206#  19-32   19-34   19-48   23-33   29-15   29-86   29-87   29-88   29-89   29-90   29-91   29-92   29-93
PIRQ      1-195#  18-134  28-32
RAND      1-140#  16-65   22-15   22-17
SBKMOD    1-115#
```

```
                        .REM    @

                                        IDENTIFICATION

                        PRODUCT CODE:   AC-S871A-MC

                        PRODUCT NAME:   CXCIAA0 DEC/X11 CISP MODULE

                        DATE:           JANUARY, 1982

                        MAINTAINER:     DIAGNOSTIC GROUP


                        THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT
                        NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL
                        EQUIPMENT CORPORATION.  DIGITAL EQUIPMENT CORPORATION ASSUMES
                        NO RESPONSIBILITY FOR  ANY  ERRORS  THAT MAY  APPEAR IN  THIS
                        DOCUMENT.

                        THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED  UNDER A
                        LICENSE AND MAY ONLY BE USED OR COPIED IN ACCORDANCE WITH THE
                        TERMS OF SUCH LICENSE.

                        DIGITAL EQUIPMENT  CORPORATION ASSUMES NO  RESPONSIBILITY FOR
                        THE USE OR RELIABILITY  OF ITS SOFTWARE  ON EQUIPMENT THAT IS
                        NOT SUPPLIED BY DIGITAL.

                            COPYRIGHT (C) 1982  DIGITAL EQUIPMENT CORPORATION
```



```
                        1.      ABSTRACT

                                THE CISP IS A DEC/X11 BACKGROUND MODULE THAT EXERCISES
                                THE COMMERCIAL INSTRUCTION SET PROCESSOR.  THE CHARACTER
                                STRINGS INSTRUCTIONS ARE TESTED BY MOVING STRINGS IN
                                MEMORY AND COMPARING THE RESULTING DESTINATION WITH THE
                                SOURCE.  THE DECIMAL ARITHMETIC INSTRUCTIONS ARE TESTED
                                BY CALCULATING ALGEBRAIC IDENTITIES INVOLVING ALL
                                OPERATIONS AND COMPARING THE ANSWERS.  NUMBERS FOR
                                CALCULATIONS ARE GENERATED RANDOMLY.

                        2.      REQUIREMENTS

                                HARDWARE:  A CISP COMMERCIAL INSTRUCTION SET PROCESSOR
                                           CONFIGURED ON A PDP-11 PROCESSOR

                                STORAGE:   CISP REQUIRES 1715 OCTAL OR 963 DECIMAL WORDS OF STORAGE.

                        3.      PASS DEFINATION

                                ONE PASS OF THE CISP MODULE CONSISTS OF 200 (OCTAL)
                                CYCLES OF THE TEST SEQUENCE.

                        4.      EXECUTION TIME

                                ONE PASS OF THE CISP MODULE RUNNING ALONE ON PDP-11/44
                                TAKES APPROXIMATELY 20 SECONDS.

                        5.      CONFIGURATION REQUIREMENTS

                                DEFAULT PARAMETERS: NONE

                                VECTOR: NONE

                                REQUIRED PARAMETERS: NONE

                        6.      DEVICE/OPTION SETUP

                                MAKE SURE A COMMERCIAL INSTRUCTION SET PROCESSOR IS INSTALLED



                        7.      MODULE OPERATION

                                THE MODULE XCIAA0 CONSISTS OF 16 TESTS INVOLVING A SEQUENCE OF
                                ONE OR MORE CIS INSTRUCTIONS.  TESTS USUALLY END WITH A COMPARE
                                STRING INSTRUCTION FOR ERROR CHECKING.  ERRORS ARE REPORTED AFTER
                                EACH SEQUENCE AND CONSIST OF STARTING ADDRESS OF SOURCE AND
                                DESTINATION STRINGS PLUS THE FIRST WORD OF EACH STRING.  MODULE
                                OPERATION IS DEPENDENT ON THE SUCCESS OF THE COMPARE INSTRUCTIONS,
                                THEREFORE THEY ARE TESTED INDIVIDUALLY BEFORE EACH SECTION OF THE
                                MODULE.  THE FAILURE OF A COMPARE INSTRUCTION IS FATAL AND RESULTS
                                THE MODULE BEING DROPPED.
                                THE CISP USE THE R6 STACK AS WORK SPACE FOR BOTH THE EXECUTION
                                AND INTERUPTION OF THE CIS INSTRUCIONS. THIS MODULE PROVIDES ITS
```

```
                                OWN R6 STACK TO INSURE ENOUGH WORK SPACE FOR THE CISP.  THE
                                SUSPENSION OF A CIS INSTRUCTION IS INDICATED BY PROCESSOR STATUS
                                BIT 8 BEING SET.  THIS MODULE RELIES ON THE MONITOR STACK TO
                                PROVIDE SPACE FOR AN INTERUPTED CIS INSTRUCTION.

                                TEST SEQUENCE:

                                1)   COMPARE CHARACTER STRING

                                2)   MOVE COMPARE

                                3)   SCAN MOVE COMPARE

                                4)   SPAN MOVE COMPARE

                                5)   MATCH MOVE COMPARE

                                6)   MOVE REVERSE MOVE REVERSE COMPARE

                                7)   MOVE TRANSLATE MOVE TRANSLATE COMPARE

                                8)   LOCATE MOVE COMPARE

                                9)   SKIP MOVE COMPARE

                                10)  COMPARE NUMERIC

                                11)  CALCULATE 10[(A+B=C)]=[10*A-10*C+10*B] COMPARE NUMERIC

                                12)  CONVERT LONG -> NUMERIC -> LONG

                                13)  CONVERT NUMERIC -> PACKED -> NUMERIC COMPARE NUMERIC

                                14)  COMPARE PACKED

                                15)  CALCULATE  10[B+C]=10[(B**2+C**2)/(B-C)] COMPARE PACKED

                                16)  CONVERT LONG -> PACKED -> LONG


                        8.      OPERATION  OPTIONS

                                NONE


                        9.      NON-STANDARD PRINTOUTS

                                NONE@
```



```
                000000          R0      =%0
                000001          R1      =%1
                000002          R2      =%2
                000003          R3      =%3
                000004          R4      =%4
                000005          R5      =%5
                000006          R6      =%6
                000007          R7      =%7
                000007          PC      =R7
                000006          SP      =R6
                076010          MFPT    =076010
                076020          L2D0    =076020
                076021          L2D1    =076021
                076022          L2D2    =076022
                076023          L2D3    =076023
                076024          L2D4    =076024
                076025          L2D5    =076025
                076026          L2D6    =076026
                076027          L2D7    =076027
                076030          MOVC    =076030
                076031          MOVRC   =076031
                076032          MOVTC   =076032
                076040          LOCC    =076040
                076041          SKPC    =076041
                076042          SCANC   =076042
                076043          SPANC   =076043
                076044          CMPC    =076044
                076045          MATC    =076045
                076050          ADDN    =076050
                076051          SUBN    =076051
                076052          CMPN    =076052
                076053          CVTNL   =076053
                076054          CVTPN   =076054
                076055          CVTNP   =076055
                076056          ASHN    =076056
                076057          CVTLN   =076057
                076060          L3D0    =076060
                076061          L3D1    =076061
                076062          L3D2    =076062
                076063          L3D3    =076063
                076064          L3D4    =076064
                076065          L3D5    =076065
                076066          L3D6    =076066
                076067          L3D7    =076067
                076070          ADDP    =076070
                076071          SUBP    =076071
                076072          CMPP    =076072
                076073          CVTPL   =076073
                076074          MULP    =076074
                076075          DIVP    =076075
                076076          ASHP    =076076
                076077          CVTLP   =076077
                076130          MOVCI   =076130
                076131          MOVRCI  =076131
                076132          MOVTCI  =076132
                076140          LOCCI   =076140
                076141          SKPCI   =076141
```

```
193          076142                  SCANCI  =076142
194          076143                  SPANCI  =076143
195          076144                  CMPCI   =076144
196          076145                  MATCI   =076145
197          076150                  ADDNI   =076150
198          076151                  SUBNI   =076151
199          076152                  CMPNI   =076152
200          076153                  CVTNLI  =076153
201          076154                  CVTPNI  =076154
202          076155                  CVTNPI  =076155
203          076156                  ASHNI   =076156
204          076157                  CVTLNI  =076157
205          076170                  ADDPI   =076170
206          076171                  SUBPI   =076171
207          076172                  CMPPI   =076172
208          076173                  CVTPLI  =076173
209          076174                  MULPI   =076174
210          076175                  DIVPI   =076175
211          076176                  ASHPI   =076176
212          076177                  CVTLPI  =076177
213          076600                  MED6X   =076600
214          076601                  MED74C  =076601
215          177776                  PSW     =177776
```



```
240                                  ;*************************************************************************
241 000000                           BEGIN:
242 000000    103     111     101    MODNAM: .ASCII  /CIAA /          ;MODULE NAME.
243 000005    000                    XFLAG:  .BYTE   0PEN             ;USED TO KEEP TRACK OF WBUFF USAGE
244 000006  000000                   ADDR:   0                ;1ST DEVICE ADDR.
245 000010  000000                   VECTOR: 0                        ;1ST DEVICE VECTOR.
246 000012    000                    BR1:    .BYTE   PRTY0            ;1ST BR LEVEL.
247 000013    000                    BR2:    .BYTE   PRTY0            ;2ND BR LEVEL.
248 000014  000002                   DVID1:  2                        ;DEVICE INDICATOR 1.
249 000016  000000                   SR1:    OPEN                     ;SWITCH REGISTER 1
250 000020  000000                   SR2:    OPEN                     ;SWITCH REGISTER 2
251 000022  000000                   SR3:    OPEN                     ;SWITCH REGISTER 3
252 000024  000000                   SR4:    OPEN                     ;SWITCH REGISTER 4
253                                  ;*************************************************************************
254 000026  040020                   STAT:   40020                    ;STATUS WORD.
255 000030  000234'                  INIT:   START                    ;MODULE START ADDR.
256 000032  000234'                  SPOINT: MODSP                    ;MODULE STACK POINTER.
257 000034  000000                   PASCNT: 0                        ;PASS COUNTER.
258 000036  000200                   ICONT:  200                      ;# OF ITERATIONS PER PASS=200
259 000040  000000                   ICOUNT: 0                        ;LOC TO COUNT ITERATIONS
260 000042  000000                   SOFCNT: 0                        ;LOC TO SAVE TOTAL SOFT ERRORS
261 000044  000000                   HRDCNT: 0                        ;LOC TO SAVE TOTAL HARD ERRORS
262 000046  000000                   SOFPAS: 0                        ;LOC TO SAVE SOFT ERRORS PER PASS
263 000050  000000                   HRDPAS: 0                        ;LOC TO SAVE HARD ERRORS PER PASS
264 000052  000000                   SYSCNT: 0                        ;# OF SYS ERRORS ACCUMULATED
265 000054  000000                   RANNUM: 0                        ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
266 000056                           CONFIG:                          ;RESERVED FOR MONITOR USE
267 000056  000000                   RES1:   0                        ;RESERVED FOR MONITOR USE
268 000060  000000                   RES2:   0                        ;RESERVED FOR MONITOR USE
269 000062  000000                   SVR0:   OPEN                     ;LOC TO SAVE R0.
270 000064  000000                   SVR1:   OPEN                     ;LOC TO SAVE R1.
271 000066  000000                   SVR2:   OPEN                     ;LOC TO SAVE R2.
272 000070  000000                   SVR3:   OPEN                     ;LOC TO SAVE R3.
273 000072  000000                   SVR4:   OPEN                     ;LOC TO SAVE R4.
274 000074  000000                   SVR5:   OPEN                     ;LOC TO SAVE R5.
275 000076  000000                   SVR6:   OPEN                     ;LOC TO SAVE R6.
276 000100  000000                   CSRA:   OPEN                     ;ADDR OF CURRENT CSR.
277 000102                           SBADR:                           ;ADDR OF GOOD DATA, OR
278 000102  000000                   ACSR:   OPEN                     ;CONTENTS OF CSR.
279 000104                           WASADR:                          ;ADDR OF BAD DATA, OR
280 000104  000000                   ASTAT:  OPEN                     ;STATUS REG CONTENTS.
281 000106                           ERRTYP:                          ;TYPE OF ERROR
282 000106  000000                   ASB:    OPEN                     ;EXPECTED DATA.
283 000110  000000                   AWAS:   OPEN                     ;ACTUAL DATA.
284 000112  000234'                  RSTRT:  RESTRT                   ;RESTART ADDRESS AFTER END OF PASS
285 000114  000000                   WDTO:   OPEN                     ;WORDS TO MEMORY PER ITERATION
286 000116  000000                   WDFR:   OPEN                     ;WORDS FROM MEMORY PER ITERATION
287 000120  000000                   INTR:   OPEN                     ;# OF INTERRUPTS PER ITERATION
288 000122  000001                   IDNUM:  1                        ;MODULE IDENTIFICATION NUMBER=1
289 000124  000000                   RES3:   OPEN
290 000126  000000                   RES4:   OPEN
291 000130  000000                   RES5:   OPEN
292 000132  000000                   RES6:   OPEN
293          000040                           .REPT   SPSIZ            ;MODULE STACK STARTS HERE.
298 000234                           MODSP:
299                                  ;*************************************************************************
```

```
                                ;*****************************************************************************
                                ;START - GET RANDOM DATA
                                ;*****************************************************************************

000234                          START:
000234  012706  003676'         RESTR1: MOV     #STACK,SP       ;SET STACK POINTER TO MODULE STACK
000240  012700  003700'                 MOV     #BUFF,R0        ;SET SOURCE ADDRESS FOR DATA
000244  012701  004520'                 MOV     #BUF200,R1      ;SET DST ADDRESS FOR DATA
000250  012702  000310                  MOV     #200.,R2        ;GET 200 DECIMAL RANDOM NUMBERS
000254  005037  000102'                 CLR     ACSR            ;CLEAR STATUS REG ADDRESS
000260  104417  000000'         1$:     RAND$   ,BEGIN          ;GO GET RANDOM NUMBER
000264  013710  000054'                 MOV     RANNUM,(R0)     ;GET THE RANDOM NUMBER PRODUCED
000270  042710  100200                  BIC     #100200,(R0)    ;CLEAR HIGH BIT OF EACH BYTE
000274  012021                         MOV     (R0)+,(R1)+     ;MOVE TO DEST FOR COMPARE
000276  077210                         SOB     R2,1$           ;IF NOT DONE, GET NEXT RANDOM NUMBER
000300  042737  000400  177776         BIC     #BIT8,PS#       ;CLEAR BIT 8 IN CASE MONITOR SET IT
```



```
                                .SBTTL  TEST CMPC INSTRUCTION
                                ;*****************************************************************************
                                ;TEST CMPC INSTRUCTION
                                ;COMPARE SRC 1 TO DST 1
                                ;*****************************************************************************

000306  004737  002412'         COMP:   JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
000312  076144  002770' 002774'         CMPCI   ,SRC.1D,DST.1D,'        ;COMPARE STRINGS, FILL WITH SPACES
000322  001410                         BEQ     MOVE                    ;NO ERROR GO TO NEXT TEST
000324  004737  002444'                 JSR     PC,ERROR0               ;GET ERROR DATA
000330  002772' 002776'                 .WORD   SRC.1A,DST.1A           ;PARAMETERS FOR GETTING ERROR DATA
000334  104404  000000'                 DATER$  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
        000340  104410  000000'         END$    ,BEGIN                  ;**FATAL ERROR** CANNOT CONTINUE MODULE
```

```
                                .SBTTL  MOVE CHARACTER STRING
                                ;******************************************************************************
                                ;MOVE CHARACTER STRING
                                ;******************************************************************************
000344  004737  002412'         MOVE:   JSR     PC,SETUP                ;GO SET UP THE DESCRIPTOPS
000350  076067  002770' 002774'         L3D7    ,SRC.1D,DST.1D,CHAR     ;LOAD DESC INTO REG
000360  076030                         MOVC                            ;MOVE STRING
000362  076144  002770' 002774'         CMPCI   ,SRC.1D,DST.1D,'        ;COMPARE SRC AND DST
000372  001411                         BEQ     SCAN                    ;IF EQUAL NEXT TEST
000374  004737  002444'                 JSR     PC,ERRORO               ;GET ERROR DATA
000400  002772' 002776'                 .WORD   SRC.1A,DST.1A           ;PARAMETERS FOR GETTING ERROR DATA
000404  104404  000000'                 DATER$  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
000410  042737  000400  177776         BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```



```
                                .SBTTL  SCAN,MOVC
                                ;******************************************************************************
                                ;SCAN,MOVC
                                ;******************************************************************************

000416  004737  002412'         SCAN:   JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
000422  112737  000001  003034'         MOVB .  #1,SET.1D               ;SET CHAR MASK FOR SPAN AND SCAN
000430  076142  002770' 003034' 1$:      SCANCI  ,SRC.1D,SET.1D          ;SCAN
000436  001003                         BNE     2$                      ;CHAR FOUND MOVE STRING
000440  106337  003034'                 ASLB    SET.1D                  ;NOT FOUND SHIFT MASK
000444  000771                         BR      1$                      ;LOOK AGAIN
000446  010037  002770'         2$:     MOV     R0,SRC.1D               ;MOV NEW ADDRESS TO DESC
000452  010137  002772'                 MOV     R1,SRC.1A               ;MOV NEW LENGTH TO DESC
000456  076130  002770' 002774'         MOVCI   ,SRC.1D,DST.1D,'        ;MOV TEXT STARTING WITH CHAR FOUND
000466  076144  002770' 002774'         CMPCI   ,SRC.1D,DST.1D,'        ;COMPARE SRC AND DST
000476  001411                         BEQ     SPAN                    ;STRINGS EQUAL NEXT TEST
000500  004737  002444'                 JSR     PC,ERRORO               ;NOT EQUAL GET ERROR DATA
000504  002772' 002776'                 .WORD   SRC.1A,DST.1A           ;PARAMETERS FOR GETTING ERROR DATA
000510  104404  000000'                 DATER$  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
000514  042737  000400  177776         BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```

```
                                .SBTTL  SPAN AND MOVC
                                ;******************************************************************************
                                ;SPAN AND MOVC
                                ;******************************************************************************

000522  004737  002412'        SPAN:   JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
000526  112737  000001  003034' 1$:     MOVB    #1,SET.1D               ;SET MASK
000534  076067  002770' 002774'        L3D7    ,SRC.1D,DST.1D,SET.1D   ;LOAD DESC (DUMMY DESC TO GET SET.1D TO R4)
000544  076043                        SPANC                           ;FIND CHAR
000546  001003                        BNE     2$                      ;FOUND MOVE STRING
000550  106337  003034'               ASLB    SET.1D                  ;NOT SHIFT MASK
000554  000767                        BR      1$                      ;AND LOOK AGAIN
000556  010037  002770'        2$:     MOV     R0,SRC.1D               ;GET ADDRESS
000562  010137  002772'                MOV     R1,SRC.1A               ;SET LENGTH
000566  076067  002770' 002774'        L3D7    ,SRC.1D,DST.1D,CHAR     ;LOAD DESC
000576  076030                        MOVC                            ;MOVE STING BEGINNING WITH CHAR
000600  076067  002770' 002774'        L3D7    ,SRC.1D,DST.1D,CHAR     ;LOAD DESC
000610  076044                        CMPC                            ;COMPARE SRC AND DST
000612  001411                        BEQ     MATCH                   ;STRINGS EQUAL NEXT TEST
000614  004737  002444'                JSR     PC,ERRORD               ;NOT EQUAL GET ERROR DATA
000620  002772' 002776'                .WORD   SRC.1A,DST.1A           ;PARAMETERS FOR GETTING ERROR DATA
000624  104404  000000'                DATERS  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
000630  042737  000400  177776        BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```



```
                                .SBTTL  MATCH
                                ;******************************************************************************
                                ;MATCH
                                ;******************************************************************************

000636  004737  002412'        MATCH:  JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
000642  076027  002770' 003000'        L2D7    ,SRC.1D,OBJ.1D          ;LOAD DESC INTO REGISTERS, SOURCE & OBJDECT PNTRS
000650  076045                        MATC                            ;MATCH STRINGS
000652  010137  002772'                MOV     R1,SRC.1A               ;GET NEW SRC ADDRESS
000656  012737  000031  002770'        MOV     #25.,SRC.1D             ;GET NEW SRC LENGTH
000664  076067  002770' 003000'        L3D7    ,SRC.1D,OBJ.1D,CHAR     ;LOAD DESCRIPTORS
000674  076044                        CMPC                            ;COMPARE RESULTS
000676  001411                        BEQ     MOVER                   ;FOUND NEXT TEST
000700  004737  002444'                JSR     PC,ERRORD               ;ERROR DATA
000704  002772' 002776'                .WORD   SRC.1A,DST.1A           ;PARAMETERS FOR GETTING ERROR DATA
000710  104404  000000'                DATERS  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
000714  042737  000400  177776        BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```

```
                                .SBTTL  MOVE REVERSE
                                ;*************************************************************************
                                ;MOVE REVERSE
                                ;*************************************************************************

000722  004737  002412'         MOVER:  JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
000726  076131  002770' 002774'         MOVRCI  ,SRC.1D,DST.1D,'        ;MOVE THE STRING REVERSED
000736  076067  002774' 002774'         L3D7    ,DST.1D,DST.1D,CHAR     ;LOAD DESCRIPTORS
000746  076031                         MOVRC                           ;MOVE REVERSE AGAIN
000750  076067  002770' 002774'         L3D7    ,SRC.1D,DST.1D,CHAR     ;LOAD DESCRIPTORS
000760  076044                         CMPC                            ;COMPARE RESULTS
000762  001411                         BEQ     MOVT                    ;FOUND NEXT TEST
000764  004737  002444'                 JSR     PC,ERRORO               ;ERROR DATA
000770  002772' 002776'                 .WORD   SRC.1A,DST.1A           ;PARAMETERS FOR GETTING ERROR DATA
000774  104404  000000'                 DATER$  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
001000  042737  000400  177776         BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```



```
                                .SBTTL  MOVE TRANSLATE
                                ;*************************************************************************
                                ;MOVE TRANSLATE
                                ;*************************************************************************

001006  004737  002412'         MOVT:   JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
001012  076132  002770' 002774'         MOVTCI  ,SRC.1D,DST.1D,' ,TRANS ;MOVE TRANSLATE, FILL WITH SPACES, TRANS TABLE ADRS
001024  076132  002774' 002774'         MOVTCI  ,DST.1D,DST.1D,' ,TRANS ;MOVE TEXT AGAIN SHOULD BE SAME AS SOURCE
001036  076144  002770' 002774'         CMPCI   ,SRC.1D,DST.1D,'        ;COMPARE SOURCE AND DESTINATIONS
001046  001411                         BEQ     LOCATE                  ;IF OK, GO TO NEXT CHECK
001050  004737  002444'                 JSR     PC,ERRORO               ;GET ERROR DATA
001054  002772' 002776'                 .WORD   SRC.1A,DST.1A           ;PARAMETERS FOR GETTING ERROR DATA
001060  104404  000000'                 DATER$  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
001064  042737  000400  177776         BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```

```
                                        .SBTTL  LOCATE AND MOVE CHARACTER
                                        ;******************************************************************************
                                        ;LOCATE AND MOVE CHARACTER
                                        ;******************************************************************************

001072  004737  002412'          LOCATE: JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
001076  012737  000040  001110'          MOV     #' ,1$+4                ;RESET ASCII SPACE CHARACTER IN LOCATION BELOW
001104  076140  002770' 000040  1$:     LOCCI   ,SRC.1D,'               ;LOCATE CHARACTER
001112  001003                          BNE     2$                      ;FOUND MOVE STRING
001114  105237  001110'                  INCB    1$+4                    ;NOT FOUND INC CHAR FOR SEARCH
001120  000771                          BR      1$                      ;LOOK AGAIN FOR NEW CHAR
001122  010037  002770'          2$:     MOV     R0,SRC.1D               ;ADDRESS OF CHAR FOUND TO SCR.1
001126  010137  002772'                  MOV     R1,SRC.1A               ;LENGTH OF STRING
001132  013703  002776'                  MOV     DST.1A,R3               ;MOVE DST ADDRESS TO R2
001136  013702  002774'                  MOV     DST.1D,R2               ;MOVE STRING LENGTH TO R3
001142  012704  000040                  MOV     #' ,R4                  ;FILL CHAR
001146  076030                          MOVC                            ;MOVE STRING BEGINING WITH CHAR FOUND
001150  076144  002770' 002774'          CMPCI   ,SRC.1D,DST.1D,'        ;COMPARE SOURCE AND DEST
001160  001411                          BEQ     SKIP                    ;STRINGS EQUAL NEXT TEST
001162  004737  002444'                  JSR     PC,ERRORD               ;NOT EQUAL ERROR
001166  002772' 002776'                  .WORD   SRC.1A,DST.1A           ;PARAMETERS FOR GETTING ERROR DATA
001172  104404  000000'                  DATERR  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
001176  042737  000400  177776          BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```



```
                                        .SBTTL  SKIP AND MOVE CHAR STRING
                                        ;******************************************************************************
                                        ;SKIP AND MOVE CHAR STRING
                                        ;******************************************************************************

001204  004737  002412'          SKIP:   JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
001210  012737  000040  001222'          MOV     #' ,1$+4                ;RESET ASCII SPACE CHARACTER IN LOCATION BELOW
001216  076141  002770' 000040  1$:     SKPCI   ,SRC.1D,'               ;SKIP CHAR
001224  001003                          BNE     2$                      ;FOUND A CHAR CHECK IT
001226  105237  001222'                  INCB    1$+4                    ;NOT FOUND INC CHAR
001232  000771                          BR      1$                      ;LOOK AGAIN
001234  010037  002770'          2$:     MOV     R0,SRC.1D               ;GET NEW SRC ADDRESS
001240  010137  002772'                  MOV     R1,SRC.1A               ;NEW SOURCE LENGTH
001244  076130  002770' 002774'          MOVCI   ,SRC.1D,DST.1D,'        ;MOVE STRING
001254  076067  002770' 002774'          L3D7    ,SRC.1D,DST.1D,CHAR     ;LOAD DESCIPTORS FOR COMPARE
001264  076044                          CMPC                            ;COMPARE STRINGS
001266  001411                          BEQ     DECDAT                  ;IF EQUAL, GO TO NEXT TEST
001270  004737  002444'                  JSR     PC,ERRORD               ;NOT EQUAL GET ERROR DATA
001274  002772' 002776'                  .WORD   SRC.1A,DST.1A           ;PARAMETERS FOR GETTING ERROR DATA
001300  104404  000000'                  DATERR  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
001304  042737  000400  177776          BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```

```
                                .SBTTL  DECIMAL ARITHMETIC TESTS
                                ;*************************************************************************
                                ;DECIMAL ARITHMETIC TESTS
                                ;SETUP DECIMAL DATA
                                ;*************************************************************************

001312  004737  002412'        DECDAT: JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
001316  005000                        CLR     R0                      ;CLR R0 FOR COUNTER
001320  005002                1$:     CLR     R2                      ;CLEAR R2, ACCUMALATE NIPPLES
001322  012704  002734'              MOV     #MSKTAB,R4              ;MOV MASK TABLE ADDRESS TO R4
001326  012703  002744'              MOV     #NINTAB,R3              ;MOV NINE TABLE ADDRESS TO R3
001332  012705  002754'              MOV     #SIXTAB,R5              ;MOV SIX TABLE ADDRESS TO R5
001336  016001  003700'        2$:     MOV     BUFF(R0),R1             ;GET DATA TO CONVERT
001342  042401                        BIC     (R4)+,R1                ;CLEAR ALL NIBBLES EXCEPT ONE
001344  022301                        CMP     (R3)+,R1                ;NIBBLE LESS THAN 9 (HEX)
001346  002001                        BGE     3$                      ;YES, CHECK NEXT NIBBLE
001350  161501                        SUB     (R5),R1                 ;NO, SUBTRACT SIX, MAKE SURE R1< 9
001352  005725                3$:     TST     (R5)+                   ;INC R5 BY 2 FOR TABLE ENTRY
001354  050102                        BIS     R1,R2                   ;SAVE NIBBLE IN R2
001356  020327  002754'              CMP     R3,#SIXTAB              ;DONE WITH ALL NIBBLES
001362  002765                        BLT     2$                      ;NO DO NEXT NIBBLE
001364  010260  003700'              MOV     R2,BUFF(R0)             ;STORE NUMBER IN SOURCE FIELD
001370  005720                        TST     (R0)+                   ;INC R0, INDEX FOR BUFF
001372  020027  000070                CMP     R0,#56.                 ;DONE WITH ALL WORDS
001376  003750                        BLE     1$                      ;IF NOT, GET NEXT WORD, OTHERWISE MODIFY DATA TYPE
001400  012701  003004'              MOV     #A,DSC,R1               ;SET DATA TYPE
001404  042711  070000        4$:     BIC     #070000,(R1)            ;CLEAR TYPE BITS
001410  052711  010000                BIS     #010000,(R1)            ;MAKE UNSIGNED ZONED DATA
001414  062701  000004                ADD     #4,R1                   ;GET NEXT DATA TYPE SPECIFIER
001420  020127  003030'              CMP     R1,#D.DSC               ;TEST FOR DONE
001424  101767                        BLOS    4$                      ;NOT DONE DO AGAIN
```



```
                                .SBTTL  TEST COMPARE NUMERIC
                                ;*************************************************************************
                                ;TEST COMPARE NUMERIC
                                ;*************************************************************************

001426  004737  002412'        CMPNUM: JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
001432  076152  003004' 003004'        CMPNI   ,A.DSC,A.DSC            ;COMPARE EQUAL STRINGS
001440  001410                        BEQ     NUMRIC                  ;EQUAL NEXT TEST
001442  004737  002444'              JSR     PC,ERROR0               ;GET ERROR DATA
001446  003006' 003006'              ,WORD   A,A                     ;PARAMETERS FOR EXPECTED AND RECEIVED DATA
001452  104404  000000'              DATE%S  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
001456  104410  000000'              END$    ,BEGIN                  ;**FATAL ERROR** CANNOT CONTINUE MODULE
```

```
                                .SBTTL  ASHN,ADDN AND SUBN TESTS
                                ;******************************************************************************
                                ;ASHN,ADDN AND SUBN TESTS
                                ;CALCULATE {(10*A+10*B)-10*C}=10*{(A-C)+B}
                                ;A=28 DIGITS, B=14 DIGITS AND C=10 DIGITS
                                ;******************************************************************************

                                ;CALCULATE {(10*A+10*B)-10*C}

001462  004737  002412'        NUMRIC: JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
001466  076156  003004' 003030'         ASHN1   ,A.DSC,D.DSC,1          ;SHIFT A ONCE
001476  076156  003010' 003020'         ASHN1   ,B.DSC,E.DSC,1          ;SHIFT B
001506  076150  003020' 003030'         ADDN1   ,E.DSC,D.DSC,E.DSC      ;ADD 10A+10B
001516  076156  003014' 003030'         ASHN1   ,C.DSC,D.DSC,1          ;SHIFT C
001526  076151  003030' 003020'         SUBN1   ,D.DSC,E.DSC,E.DSC      ;SUB 10A+10B-10C

                                ;CALCULATE 10*{(A-C)+B}

001536  076067  003014' 003004'         L3D7    ,C.DSC,A.DSC,F.DSC      ;LOAD DESCRIPTORS
001546  076051                         SUBN                            ;SUB A-C
001550  076067  003010' 003024'         L3D7    ,B.DSC,F.DSC,D.DSC      ;LOAD DESC
001560  076050                         ADDN                            ;ADD F+B
001562  076067  003030' 003024'         L3D7    ,D.DSC,F.DSC,ONE        ;LOAD DESC
001572  076056                         ASHN                            ;MULT BY 10

                                ;COMPARE RESULTS

001574  076027  003020' 003024'         L2D7    ,E.DSC,F.DSC            ;LOAD DESC
001602  076052                         CMPN                            ;COMPARE STRINGS
001604  001411                         BEQ     CONNUM                  ;EQUAL GO TO NEXT TEST
001606  004737  002444'                JSR     PC,ERROR0               ;GET ERROR DATA
001612  003022' 003026'                .WORD   E,F                     ;PARAMETERS FOR EXPECTED AND RECEIVED DATA
001616  104404  000000'                DATER$  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
001622  042737  000400  177776         BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```



```
                                .SBTTL  CONVERT DATA TYPES
                                ;******************************************************************************
                                ;CONVERT DATA TYPES
                                ;LONG -> NUMERIC -> LONG
                                ;NUMERIC -> PACKED -> NUMERIC
                                ;******************************************************************************

001630  004737  002412'        CONNUM: JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
001634  076157  003020' 003040'         CVTLN1  ,E.DSC,LONG.1           ;CONVERT LONG TO NUMERIC
001642  076153  003020' 003044'         CVTN$1  ,E.DSC,LONG.2           ;CONVERT NUMERIC TO LONG
001650  023737  003040' 003044'         CMP     LONG.1,LONG.2           ;CHECK FIRST HALF #LONG WORD
001656  001004                         BNE     1$                      ;NOT EQUAL ERROR
001660  023737  003042' 003046'         CMP     LONG.1+2,LONG.2+2       ;EQUAL CHECK SECOND HALF
001666  001407                         BEQ     NUMPAC                  ;EQUAL NEXT TEST
001670  004737  002502'         1$:     JSR     PC,ERROR1               ;GET ERROR DATA
001674  104404  000000'                DATER$  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
001700  042737  000400  177776         BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```

```
                                        ;CONVERT NUM TO PACK TO NUM
                                        ;
001706  004737  002412'          NUMPAC: JSR     PC,SETUP            ;GO SET UP THE DESCRIPTORS
001712  052737  070000  003024'          BIS     #070000,F.DSC       ;SET B TO PACKED DATA TYPE
001720  076027  003024' 003030'          L2D7    ,F.DSC,D.DSC        ;LOAD DESC FOR SECOND CONVERT
001726  076155  003020' 003024'          CVTNP1  ,E.DSC,F.DSC        ;CONVERT BACK TO NUM
001734  076054                          CVTPN                       ;CONVERT NUM TO PACKED
001736  076152  003030' 003020'          CMPN1   ,D.DSC,E.DSC        ;COMPARE RESULTS
001744  001411                          BEQ     PACDAT              ;EQUAL NEXT TEST
001746  004737  002444'                  JSR     PC,ERRDR0           ;GO GET ERROR DATA
001752  003022' 003026'                  .WORD   E,F                 ;PARAMETERS FOR EXPECTED AND RECEIVED DATA
001756  104404  000000'                  DATERR  ,BEGIN              ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
001762  042737  000400  177776          BIC     #BIT8,PSW           ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```



```
                                        .SBTTL  PACKED DECIMAL ARITHMETIC
                                        ;*****************************************************************************
                                        ;PACKED DECIMAL ARITHMETIC
                                        ;*****************************************************************************
                                        ;
001770  004737  002412'          PACDAT: JSR     PC,SETUP            ;GO SET UP THE DESCRIPTORS
001774  012701  003004'                  MOV     #A.DSC,R1           ;SET DATA TYPE
002000  042711  070000          1$:     BIC     #070000,(R1)        ;CLEAR DATA TYPE BITS
002004  052711  060000                  BIS     #060000,(R1)        ;MAKE SIGNED PACKED DATA
002010  062701  000004                  ADD     #4,R1               ;NEXT DATA TYPE SPEC
002014  020127  003030'                  CMP     R1,#D.DSC           ;DONE YET
002020  101767                          BLOS    1$                  ;NO DO AGAIN
002022  143777  002764' 000756          BICB    #HIMASK,@A          ;CLEAR HI NIBBLE OF A
002030  143777  002764' 000754          BICB    #HIMASK,@B          ;CLEAR HI NIBBLE OF B
002036  143777  002764' 000752          BICB    #HIMASK,@C          ;CLEAR HI NIBBLE OF C
002044  143737  002765' 003716'          BICB    LOMASK,BUFF+14.     ;CLEAR SIGN NIBBLE A
002052  153737  002766' 003716'          BISB    SIGN,BUFF+14.       ;SET SIGN NIBBLE A
002060  143737  002765' 003743'          BICB    LOMASK,BUFF+35.     ;CLEAR SIGN B
002066  153737  002766' 003743'          BISB    SIGN,BUFF+35.       ;SET SIGN B
002074  143737  002765' 003757'          BICB    LOMASK,BUFF+47.     ;CLEAR SIGN NIBBLE C
002102  153737  002766' 003757'          BISB    SIGN,BUFF+47.       ;SET SIGN C
```

```
                                .SBTTL  TEST COMPARE PACKED
                                ;******************************************************************************
                                ;TEST COMPARE PACKED
                                ;******************************************************************************

002110  004737  002412'          CMPPAK: JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
002114  076172  003004' 003004'          CMPPI   ,A.DSC,A.DSC            ;COMPARE EQUAL STRINGS
002122  001410                          BEQ     PACKED                  ;EQUAL GO TEST
002124  004737  002444'                  JSR     PC,ERRORO               ;GET ERROR DATA
002130  003006' 003006'                  .WORD   A,A                     ;PARAMETERS FOR EXPECTED AND RECEIVED DATA
002134  104404  000000'                  DATER$  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
002140  104410  000000'                  END$    ,BEGIN                  ;**FATAL ERROR** CANNOT CONTINUE MODULE
```



```
                                .SBTTL  CALCULATE 10*[(B+C)]=10*[(B**2)-(C**2)/(B-C)]
                                ;******************************************************************************
                                ;CALCULATE 10*[(B+C)]=10*[(B**2)-(C**2)/(B-C)]
                                ;******************************************************************************

002144  004737  002412'          PACKED: JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
002150  076174  003010' 003010' 1$:      MULPI   ,B.DSC,B.DSC,D.DSC      ;MULTIPLY B*B, PUT RESULTS IN D
002160  076174  003014' 003014'          MULPI   ,C.DSC,C.DSC,E.DSC      ;MULTIPLY C*C, PUT RESULTS IN E
002170  076171  003020' 003030'          SUBPI   ,E.DSC,D.DSC,E.DSC      ;SUBTRACT E FROM D, ([B^2]-[C^2]) RESULTS IN E
002200  076171  003014' 003010'          SUBPI   ,C.DSC,B.DSC,F.DSC      ;SUBTRACT B FROM C, RESULTS IN F
002210  001005                          BNE     2$                      ;BRANCH IF DENOMINATOR IS NON-ZERO
002212  076176  003024' 003014'          ASHPI   ,F.DSC,C.DSC,1          ;SHIFT F TO C
002222  000752                          BR      1$                      ;BRANCH BACK TO TRY OVER
002224  076067  003030' 003020' 2$:      L3D7    ,D.DSC,E.DSC,ONE        ;LOAD DESC FOR ASHP
002234  076175  003024' 003020'          DIVPI   ,F.DSC,E.DSC,D.DSC      ;DIVIDE [(B^2)-(C^2)] BY [B-C] (E / F), RESULT IN G
002244  076076                          ASHP                            ;SHIFT  10*D
002246  076067  003010' 003014'          L3D7    ,B.DSC,C.DSC,D.DSC      ;LOAD DESCRIPTORS
002256  076070                          ADDP                            ;ADD B AND C, RESULT IN F
002260  076067  003030' 003024'          L3D7    ,D.DSC,F.DSC,ONE        ;LOAD DESCRIPTORS
002270  076076                          ASHP                            ;SHIFT  10*D

                                ;******THE RESULTS OF THE LEFT HALF OF THE EQUATION IS NOW IN D - COMPARE RESULTS
002272  076172  003020' 003024'          CMPPI   ,E.DSC,F.DSC            ;DOES THE EQUATION BALANCE (LEFT=RIGHT)
002300  001411                          BEQ     COMPAK                  ;BRANCH TO NEXT SECTION IF OK
002302  004737  002444'                  JSR     PC,ERRORO               ;GET ERROR DATA
002306  003022' 003026'                  .WORD   E,F                     ;PARAMETERS FOR EXPECTED AND RECEIVED DATA
002312  104404  000000'                  DATER$  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
002316  042737  000400  177776          BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```

```
                                .SBTTL  CONVERT DATA TYPES
                                ;*************************************************************************
                                ;CONVERT DATA TYPES
                                ;LONG -> PACKED -> LONG
                                ;*************************************************************************

002324  004737  002412'        CONPAK: JSR     PC,SETUP                ;GO SET UP THE DESCRIPTORS
002330  076177  003020' 003040'         CVTLP1  ,E.DSC,LONG.1           ;CONVERT LONG TO PACKED
002336  076173  003020' 003044'         CVTPL1  ,E.DSC,LONG.2           ;CONVERT PACKED TO LONG
002344  023737  003040' 003044'         CMP     LONG.1,LONG.2           ;COMPARE RESULTS
002352  001004                         BNE     1$                      ;BRANCH TO CALL ERROR IF NOT EQUAL
002354  023737  003042' 003046'         CMP     LONG.1+2,LONG.2+2       ;COMPARE RESULTS OF SECOND WORD
002362  001407                         BEQ     DONE                    ;BRANCH AROUND ERROR IF EQUAL
002364  004737  002502'        1$:     JSR     PC,ERROR1               ;GET PC,ERROR DATA
002370  104404  000000'                DATERR  ,BEGIN                  ;*>*>*>*>*>*DATA ERROR*<*<*<*<*<*
002374  042737  000400  177776         BIC     #BIT8,PSW               ;CLEAR BIT 8 IN CASE ERROR ROUTINE SET IT
```



```
002402  104413  000000'        DONE:   ENDITS  ,BEGIN                ;SIGNAL END OF ITERATION, MONITOR TO TEST END OF PASS
002406  000137  000234'                JMP     RESTRT                  ;RETURN TO START
```

```
                                .SBTTL  SUBROUTINES
                                ;*****************************************************************************
                                ;SUBROUTINE=SETUP > SETS UP CHAR STRING DESCRIPTORS
                                ;USAGE: JSR     PC,SETUP        ;
                                ;*****************************************************************************

002412  012737  000310  002770' SETUP:  MOV     #200.,SRC.1D    ;SET SOURCE LENGTH
002420  012737  003700' 002772'         MOV     #BUFF,SRC.1A    ;SET SOURCE ADDRESS
002426  012737  000310  002774'         MOV     #200.,DST.1D    ;DEST LENGTH
002434  012737  004520' 002776'         MOV     #BUF200,DST.1A  ;DEST ADDRESS
002442  000207                         RTS     PC              ;RETURN
```



```
                                .SBTTL  SUBROUTINES-ERROR0 AND ERROR1
                                ;*****************************************************************************
                                ;SUBROUTINES-ERROR0 AND ERROR1
                                ;LOAD THE EXPECTED AND RECEIVED DATA AND ADDRESSES INTO THE PROPER LOCATIONS
                                ;FOR THE ERROR CALL
                                ;USAGE: JSR     PC,ERROR0
                                ;       .WORD   ADRESS,DATA     ;LOCATIONS CONTAINING THE ADDRESS OF EXPECTED AND
                                ;                               ;RECEIVED DATA
                                ;OR>    JSR     PC,ERROR1
                                ;       NO ARGUMENTS
                                ;*****************************************************************************

002444  011600                  ERROR0: MOV     (SP),R0         ;LOAD ADDRESS OF FIRST PARAMETER TO R0
002446  011037  000102'                 MOV     (R0),SBADR      ;LOAD EXPECTED ADDRESS
002452  005037  000106'                 CLR     ASB             ;CLEAR EXPECTED DATA LOCATION
002456  113037  000106'                 MOVB    @(R0)+,ASB      ;LOAD EXPECTED DATA LOCATION BYTE
002462  011037  000104'                 MOV     (R0),WASADR     ;LOAD RECEIVED ADDRESS
002466  005037  000110'                 CLR     AWAS            ;CLEAR RECEIVED DATA LOCATION
002472  113037  000110'                 MOVB    @(R0)+,AWAS     ;LOAD RECEIVED DATA
002476  010016                         MOV     R0,(SP)         ;FUDGE RETURN OVER PARAMETERS
002500  000207                         RTS     PC              ;RETURN

002502  012737  003040' 000102' ERROR1: MOV     #LONG.1,SBADR   ;LOAD EXPECTED ADDRESS
002510  013737  003040' 000106'         MOV     LONG.1,ASB      ;LOAD EXPECTED DATA
002516  012737  003044' 000104'         MOV     #LONG.2,WASADR  ;LOAD RECEIVED ADDRESS
002524  013737  003044' 000110'         MOV     LONG.2,AWAS     ;LOAD RECEIVED DATA
002532  000207                         RTS     PC              ;RETURN
```

```
                                .SBTTL  DATA TABLES
                                ;******************************************************************************
                                ;TRANSLATE AND CHARACTER ATTRIBUTE TABLE
                                ;USED BY MOVE TRANSLATE
                                ;USED BY SPAN AND SCAN INSTRUCTIONS
                                ;128 CHAR ASCII
                                ;******************************************************************************

002534   177   176   175   TRANS:  .BYTE   177,176,175,174,173,172,171,170,167,166,165,164,163,162,161,160
002554   157   156   155           .BYTE   157,156,155,154,153,152,151,150,147,146,145,144,143,142,141,140
002574   137   136   135           .BYTE   137,136,135,134,133,132,131,130,127,126,125,124,123,122,121,120
002614   117   116   115           .BYTE   117,116,115,114,113,112,111,110,107,106,105,104,103,102,101,100
002634   077   076   075           .BYTE   077,076,075,074,073,072,071,070,067,066,065,064,063,062,061,060
002654   057   056   055           .BYTE   057,056,055,054,053,052,051,050,047,046,045,044,043,042,041,040
002674   037   036   035           .BYTE   037,036,035,034,033,032,031,030,027,026,025,024,023,022,021,020
002714   017   016   015           .BYTE   017,016,015,014,013,012,011,010,007,006,005,004,003,002,001,000
```



```
                                ;******************************************************************************
                                ;MASK TABLES
                                ;FOR MAKING VALID DECIMAL DATA
                                ;******************************************************************************

002734  177760  177417  170377  MSKTAB: .WORD   177760,177417,170377,007777
002744  000011  000220  004400  NINTAB: .WORD   000011,000220,004400,070000
002754  000006  000140  003000  SIXTAB: .WORD   000006,000140,003000,100000
002764     360                  HIMASK: .BYTE   360
002765     017                  LOMASK: .BYTE   017
002766     014                  SIGN:   .BYTE   014
                                        .EVEN

                                ;CHARACTER STRING DESCRIPTOR TABLE

002770  000310                  SRC.1D: .WORD   200.
002772  003700'                 SRC.1A: .WORD   BUFF
002774  000310                  DST.1D: .WORD   200.
002776  004520'                 DST.1A: .WORD   BUF200
003000  000031  003762'         OBJ.1D: .WORD   25.,BUFF+50.
003004  000034                  A.DSC:  .WORD   28.
003006  003700'                 A:      .WORD   BUFF
003010  000016                  B.DSC:  .WORD   14.
003012  003734'                 B:      .WORD   BUFF+28.
003014  000012                  C.DSC:  .WORD   10.
003016  003752'                 C:      .WORD   BUFF+42.
003020  000037                  E.DSC:  .WORD   31.
003022  003770'                 E:      .WORD   BUFF+56.
003024  000037                  F.DSC:  .WORD   31.
003026  004030'                 F:      .WORD   BUFF+88.
003030  000037                  D.DSC:  .WORD   31.
003032  004070'                 D:      .WORD   BUFF+120.
003034  000001  002534'         SET.1D: .WORD   1,TRANS
003040  001020  000000         LONG.1: .WORD   000528.,0
003044                          LONG.2: .BLKW   2
003050  000001                  ONE:    .WORD   1
003052  000040  002534'         CHAR:   .WORD   ' ,TRANS
003056                          STKLIM: .BLKW   200.
003676  000000                  STACK:  .WORD   0
                                ;******************************************************************************

                                ;BUFFER SPACE 200 WORDS LONG USED FOR SOURCE AND DESTINATIONS

                                ;******************************************************************************
003700                          BUFF:   .BLKW   200.
004520                          BUF200: .BLKW   200.
005340  000000                  ENDALL: .WORD   0
        000001                          .END
```

SYMBOL TABLE

```
A       003006R     CMPN   = 076052     HRDPAS  000050R     NSKTAB  002734R     R6     =%000006
ACSR    000102R     CMPNI  = 076152     ICON1   000036R     NULP   = 076074     R7     =%000007
ADDN   = 076050     CMPNUM  001426R     ICONT   000040R     NULP1  = 076174     SBADR   000102R
ADDNI  = 076150     CMPP   = 076072     IDNUM   000122R     NCPUOP= 000020     SCAN    000416R
ADDP   = 076070     CMPPAK  002110R     INDPAR= 000040      NINTAB  002744R     SCANC  = 076042
ADDPI  = 076170     CMPPI  = 076172     INIT    000030R     NOAPTY= 000002     SCANCI= 076142
ADDR    000006R     COMP    000306R     INTR    000120R     NULL   = 000000     SETUP   002412R
ADDR22= 001000      CONFIG  000056R     KTPRES= 000400      NUMPAC  001706R     SET.1D  003034R
APTPRE= 000200      CONNUM  001630R     KTXIND= 040000      NUMRIC  001402R     SIGN    002766R
ASB     000106R     CONPAK  002324R     LOCATE  001072R     OBJ.1D  003000R     SIXTAB  002754R
ASHN   = 076056     CSRA    000100R     LOCC   = 076040     ONE     003050R     SKIP    001204R
ASHNI  = 076156     CVTLN  = 076057     LOCCI  = 076140     OPEN   = 000000     SKPC   = 076041
ASHP   = 076076     CVTLNI= 076157      LOMASK  002765R     OTUA$  = 104420     SKPC1  = 076141
ASHPI  = 076176     CVTLP  = 076077     LONG.1  003040R     PACDAT  001770R     SOFCNT  000042R
ASTAT   000104R     CVTLPI= 076177      LONG.2  003044R     PACKED  002144R     SOFER$= 104406
AUTO   = 000010     CVTNL  = 076053     L2D0   = 076020     PARPRE= 002000      SOFPAS  000046R
AWAS    000110R     CVTNLI= 076153      L2D1   = 076021     PASCNT  000034R     SPAN    000522R
A.DSC   003004R     CVTNP  = 076055     L2D2   = 076022     PDPF11= 000002      SPANC  = 076043
B       003012R     CVTNPI= 076155      L2D3   = 076023     PDPLS1= 020000      SPANCI= 076143
BEGIN   000000R     CVTPL  = 076073     L2D4   = 076024     PDP44  = 100000     SPOINT  000032R
BIT0   = 000001     CVTPLI= 076173      L2D5   = 076025     PDP60  = 004000     SPS1Z  = 000040
BIT1   = 000002     CVTPN  = 076054     L2D6   = 076026     PDP70  = 010000     SRC.1A  002772R
BIT10  = 002000     CVTPNI= 076154      L2D7   = 076027     PIRQS  = 000004     SRC.1D  002770R
BIT11  = 004000     C.DSC   003014R     L3D0   = 076060     POPSP  = 005726     SR1     000016R
BIT12  = 010000     D       003032R     L3D1   = 076061     POPSP2= 022626      SR2     000020R
BIT13  = 020000     DATCK$= 104411      L3D2   = 076062     PRHMS$= 000002      SR3     000022R
BIT14  = 040000     DATER$= 104404      L3D3   = 076063     PRTI   = 000000     SR4     000024R
BIT15  = 100000     DECDAT  001312R     L3D4   = 076064     PRTY0  = 000000     STACK   003676R
BIT2   = 000004     DIVP   = 076075     L3D5   = 076065     PRTY1  = 000040     START   000234R
BIT3   = 000010     DIVPI  = 076175     L3D6   = 076066     PRTY2  = 000100     STAT    000026R
BIT4   = 000020     DONE    002402R     L3D7   = 076067     PRTY3  = 000140     STKLIM  003056R
BIT5   = 000040     DST.1A  002776R     MAP22$= 104416      PRTY4  = 000200     SUBN   = 076051
BIT6   = 000100     DST.1D  002774R     MATC   = 076045     PRTY5  = 000240     SUBN1  = 076151
BIT7   = 000200     DVID1   000014R     MATCH   000636R     PRTY6  = 000300     SUBP   = 076071
BIT8   = 000400     D.DSC   003030R     MATC1  = 076145     PRTY7  = 000340     SUBPI  = 076171
BIT9   = 001000     E       003022R     MED6X  = 076600     PS     = 177776     SVR0    000062R
BREAK$= 104407      ECCMEM= 000100      MED74C= 076601      PSW    = 177776     SVR1    000064R
BR1     000012R     ENDALL  005340R     MFPT   = 076010     PUSH   = 005746     SVR2    000066R
BR2     000013R     ENDIT$= 104413      MODNAM  000000R     PUSH2  = 024646     SVR3    000070R
BTOD$  = 104421     END$   = 104410     MODSP   000234R     PWRFLG= 000002      SVR4    000072R
BUFF    003700R     ERROR0  002444R     MOVC   = 076030     QMON22= 000010      SVR5    000074R
BUF200  004520R     ERROR1  002502R     MOVC1  = 076130     RAND$  = 104417     SVR6    000076R
B.DSC   003010R     ERRTYP  000106R     MOVE    000344R     RANNUM  000054R     SYSCNT  000052R
C       003016R     EXIT$  = 104400     MOVER   000722R     RESTRT  002334R     TRANS   002534R
CAPRFS= 000004      E.DSC   003020R     MOVRC  = 076031     RES1    000056R     TRPDFD= 000023
CDATA$= 104412      F       003026R     MOVRC1= 076131      RES2    000060R     USTACK= 000001
CHAR    003052R     F.DSC   003024R     MOVT    001006R     RES3    000124R     VECTOR  000010R
CKHNG$= 000001      GETPAS= 104415      MOVTC  = 076032     RES4    000126R     WASADR  000104R
CLKPRE= 000001      GWBUF$= 104414      MOVTC1= 076132      RES5    000130R     WDFR    000116R
CLKSPS= 104422      HIMASK  002764R     MSG%$  = 104403     RES6    000132R     WDTO    000114R
CMPC   = 076044     HRDCNT  000044R     MSGS$  = 104402     RH70   = 001000     XFLAG   000005R
CMPC1  = 076144     HRDER$= 104405      MSG$   = 104401     RSTRT   000112R
```

```
. ABS.  000000     000
        005342     001
ERRORS DETECTED:  0
```

VIRTUAL MEMORY USED: 6786 WORDS (27 PAGES)

SYMBOL TABLE

```
DYNAMIC MEMORY:  7362 WORDS  ( 28 PAGES)
ELAPSED TIME:  00:00:25
CXCIAA,CXCIAA.SEQ/-SP/CR/NL:TOC=DDXCOM.P11,CXC1AA.P11
```

SYMBOL	VALUE	REFERENCES							
A	003006 R	17-511	17-511	21-586	22-604	22-604	#29-725		
ACSR	000102 R	#5-278	#6-313						
ADDN	= 076050	#4-164	18-534						
ADDNI	= 076150	#4-197	18-525						
ADDP	= 076070	#4-180	23-623						
ADDPI	= 076170	#4-205							
ADDR	000006 R	#5-244							
ADDR22	= 001000	#5-301							
APIPRE	= 000200	#5-301							
ASB	000106 R	#5-282	#27-675	#27-676	#27-684				
ASHN	= 076056	#4-170	18-536						
ASHNI	= 076156	#4-203	18-523	18-524	18-526				
ASHP	= 076076	#4-186	23-621	23-625					
ASHPI	= 076176	#4-211	23-617						
ASTAT	000104 R	#5-280							
AUTO	= 000010	#5-301							
AWAS	000110 R	#5-283	#27-678	#27-679	#27-686				
A.DSC	003004 R	16-496	17-508	17-508	18-523	18-531	21-580	22-601	22-601	#29-724
B	003012 R	21-587	#29-727						
BEGIN	000000 R	#5-241	6-314	7-331	7-331	8-343	9-362	10-384	11-400	12-415
		13-428	14-450	15-470	17-512	17-512	18-545	19-561	20-573	22-605
		22-605	23-632	24-647	25-648					
BIT0	= 000001	#5-301								
BIT1	= 000002	#5-301								
BIT10	= 002000	#5-301								
BIT11	= 004000	#5-301								
BIT12	= 010000	#5-301								
BIT13	= 020000	#5-301								
BIT14	= 040000	#5-301								
BIT15	= 100000	#5-301								
BIT2	= 000004	#5-301								
BIT3	= 000010	#5-301								
BIT4	= 000020	#5-301								
BIT5	= 000040	#5-301								
BIT6	= 000100	#5-301								
BIT7	= 000200	#5-301								
BIT8	= 000400	#5-301	6-319	8-343	9-362	10-384	11-400	12-415	13-428	14-450
		15-470	18-545	19-561	20-573	23-632	24-647			
BIT9	= 001000	#5-301								
BREAKS	= 104407	#5-301								
BR1	000012 R	#5-246								
BR2	000013 R	#5-247								
BTUDS	= 104421	#5-301								
BUFF	003700 R	6-310	16-483	#16-492	#21-589	#21-590	#21-591	#21-592	#21-593	#21-594
		26-657	29-720	29-723	29-725	29-727	29-729	29-731	29-733	29-735
		#29-748								
BUF200	004520 R	6-311	26-659	29-722	#29-749					
B.DSC	003010 R	18-524	18-533	23-612	23-612	23-615	23-622	#29-726		
C	003016 R	21-588	#29-729							
CAPRES	= 000004	#5-301								
CDATAS	= 104412	#5-301								
CHAR	003052 R	8-337	10-377	10-379	11-395	12-408	12-410	15-465	#29-740	

SYMBOL	VALUE	REFERENCES							
CKHNGS	= 000001	#5-301							
CLKPRE	= 000001	#5-301							
CLKSPS	= 104422	#5-301							
CMPC	= 076044	#4-162	10-380	11-396	12-411	15-466			
CMPCI	= 076144	#4-195	7-327	8-339	9-358	13-424	14-446		
CMPN	= 076052	#4-166	18-541						
CMPNI	= 076152	#4-199	17-508	20-569					
CMPNUM	001426 R	#17-507							
CMPP	= 076072	#4-182							
CMPPAK	002110 R	#22-600							
CMPPI	= 076172	#4-207	22-601	23-628					
COMP	000306 R	#7-326							
CONFIG	000056 R	#5-266							
CONNUM	001630 R	18-542	#19-553						
CONPAK	002324 R	23-629	#24-639						
CSRA	000100 R	#5-276							
CVTLN	= 076057	#4-171							
CVTLNI	= 076157	#4-204	19-554						
CVTLP	= 076077	#4-187							
CVTLPI	= 076177	#4-212	24-640						
CVTNL	= 076053	#4-167							
CVTNLI	= 076153	#4-200	19-555						
CVTNP	= 076055	#4-169							
CVTNPI	= 076155	#4-202	20-567						
CVTPL	= 076073	#4-183							
CVTPLI	= 076173	#4-208	24-641						
CVTPN	= 076054	#4-168	20-568						
CVTPNI	= 076154	#4-201							
C.DSC	003014 R	18-526	18-531	23-613	23-613	23-615	23-617	23-622	#29-728	
D	003032 R	#29-735								
DATCKS	= 104411	#5-301								
DATERS	= 104404	#5-301	7-331	8-343	9-362	10-384	11-400	12-415	13-428	14-450
		15-470	17-512	18-545	19-561	20-573	22-605	23-632	24-647	
DECDAT	001312 R	15-467	#16-477							
DIVP	= 076075	#4-185								
DIVPI	= 076175	#4-210	23-620							
DONE	002402 R	24-645	#25-648							
DST.1A	002776 R	7-330	8-342	9-361	10-383	11-399	12-414	13-427	14-442	14-449
		15-469	#26-659	#29-722						
DST.1D	002774 R	7-327	8-337	8-339	9-357	9-358	10-370	10-377	10-379	12-407
		12-408	12-408	12-410	13-422	13-423	13-423	13-424	14-443	14-446
		15-464	15-465	#26-658	#29-721					
DVIOI	000014 R	#5-248								
D.DSC	003030 R	16-500	18-523	18-525	18-526	18-527	18-533	18-535	20-566	20-569
		21-584	23-612	23-614	23-619	23-620	23-622	23-624	#29-734	
E	003022 R	18-544	20-572	23-631	#29-731					
ECCMEM	= 000100	#5-301								
ENDALL	005340 R	#29-750								
ENDITS	= 104413	#5-301	25-648							
ENDS	= 104410	#5-301	7-331	17-512	22-605					
ERRORO	002444 R	7-329	8-341	9-360	10-382	11-398	12-413	13-426	14-448	15-468
		17-510	18-543	20-571	22-603	23-630	#27-673			

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
ERROR1	002502 R	19-560	24-646	#27-683						
ERRTYP	000106 R	#5-281								
EXITS	= 104400	#5-301								
E.DSC	003020 R	18-524	18-525	18-525	18-527	18-527	18-540	19-554	19-555	20-567
		20-569	23-613	23-614	23-614	23-619	23-620	23-628	24-640	24-641
		#29-730								
F	003026 R	18-544	20-572	23-631	#29-733					
F.DSC	003024 R	18-531	18-533	18-535	18-540	#20-565	20-566	20-567	23-615	23-617
		23-620	23-624	23-628	#29-732					
GETPAS	= 104415	#5-301								
GWBUFS	= 104414	#5-301								
HIMASK	002764 R	21-586	21-587	21-588	#29-712					
HRDCNT	000044 R	#5-261								
HRDERS	= 104405	#5-301								
HRDPAS	000050 R	#5-263								
ICONT	000036 R	#5-258								
ICOUNT	000040 R	#5-259								
IDNUM	000122 R	#5-288								
INDPAR	= 000040	#5-301								
INIT	000030 R	#5-255								
INTR	000120 R	#5-287								
KTPRES	= 000400	#5-301								
KTXTND	= 040000	#5-301								
LOCATE	001072 R	13-425	#14-434							
LOCC	= 076040	#4-158								
LOCCI	= 076140	#4-191	14-436							
LOMASK	002765 R	21-589	21-591	21-593	#29-713					
LONG.1	003040 R	19-554	19-556	19-558	24-640	24-642	24-644	27-683	27-684	#29-737
LONG.2	003044 R	19-555	19-556	19-558	24-641	24-642	24-644	27-685	27-686	#29-738
L2D0	= 076020	#4-147								
L2D1	= 076021	#4-148								
L2D2	= 076022	#4-149								
L2D3	= 076023	#4-150								
L2D4	= 076024	#4-151								
L2D5	= 076025	#4-152								
L2D6	= 076026	#4-153								
L2D7	= 076027	#4-154	11-391	18-540	20-566					
L3D0	= 076060	#4-172								
L3D1	= 076061	#4-173								
L3D2	= 076062	#4-174								
L3D3	= 076063	#4-175								
L3D4	= 076064	#4-176								
L3D5	= 076065	#4-177								
L3D6	= 076066	#4-178								
L3D7	= 076067	#4-179	8-337	10-370	10-377	10-379	11-395	12-408	12-410	15-465
		18-531	18-533	18-535	23-619	23-622	23-624			
MAP22S	= 104416	#5-301								
MATC	= 076045	#4-163	11-392							
MATCH	000636 R	10-381	#11-390							
MATCI	= 076145	#4-196								
MED6X	= 076600	#4-213								
MED74C	= 076601	#4-214								

SYMBOL CROSS REFERENCE

SYMBOL	VALUE	REFERENCES								
MFPT	= 076010	#4-146								
MODNAM	000000 R	#5-242								
MODSP	000234 R	5-256	#5-298							
MOVC	= 076030	#4-155	8-338	10-378	14-445					
MOVCI	= 076130	#4-188	9-357	15-464						
MOVE	000344 R	7-328	#8-336							
MOVER	000722 R	11-397	#12-406							
MOVRC	= 076031	#4-156	12-409							
MOVRCI	= 076131	#4-189	12-407							
MOVT	001006 R	12-412	#13-421							
MOVTC	= 076032	#4-157								
MOVTCI	= 076132	#4-190	13-422	13-423						
MSGN$	= 104403	#5-301								
MSGS$	= 104402	#5-301								
MSG$	= 104401	#5-301								
MSKTAB	002734 R	16-480	#29-709							
MULP	= 076074	#4-184								
MULP1	= 076174	#4-209	23-612	23-613						
NCPU0P	= 000020	#5-301								
NINTAB	002744 R	16-481	#29-710							
NOAPTY	= 000002	#5-301								
NULL	= 000000	#5-301								
NUMPAC	001706 R	19-559	#20-564							
NUMRIC	001462 R	17-509	#18-522							
OBJ.ID	003000 R	11-391	11-395	#29-723						
ONE	003050 R	18-535	23-619	23-624	#29-739					
OPEN	= 000000	5-243	5-249	5-250	5-251	5-252	5-269	5-270	5-271	5-272
		5-273	5-274	5-275	5-276	5-278	5-280	5-282	5-283	5-285
		5-286	5-287	5-289	5-290	5-291	5-292	#5-301		
OTOA$	= 104420	#5-301								
PACDAT	001770 R	20-570	#21-579							
PACKED	002144 R	22-602	#23-611							
PARPRE	= 002000	#5-301								
PASCNT	000034 R	#5-257								
PDPF11	= 000002	#5-301								
PDPLSI	= 020000	#5-301								
PDP44	= 100000	#5-301								
PDP60	= 004000	#5-301								
PDP70	= 010000	#5-301								
PIRQ$	= 000004	#5-301								
POPSP	= 005726	#5-301								
POPSP2	= 022626	#5-301								
PRHMS$	= 000002	#5-301								
PRTY	= 000000	#5-301								
PRTY0	= 000000	5-246	5-247	#5-301						
PRTY1	= 000040	#5-301								
PRTY2	= 000100	#5-301								
PRTY3	= 000140	#5-301								
PRTY4	= 000200	#5-301								
PRTY5	= 000240	#5-301								
PRTY6	= 000300	#5-301								
PRTY7	= 000340	#5-301								

```
SYMBOL  VALUE           REFERENCES
PS      = 177776        #5-301
PSW     = 177776        #4-215   #5-301   *6-319   *8-343   *9-362   *10-384  *11-400  *12-415  *13-428
                        *14-450  *15-470  *18-545  *19-561  *20-573  *23-632  *24-647
PUSH    = 005746        #5-301
PUSH2   = 024646        #5-301
PWRFLG  = 000002        #5-301
QMON22  = 000010        #5-301
RANDS   = 104417        #5-301    6-314
RANNUM    000054 R      #5-265    6-315
RESTRT    000234 R       5-284   #6-309    25-649
RES1      000056 R      #5-267
RES2      000060 R      #5-268
RES3      000124 R      #5-289
RES4      000126 R      #5-290
RES5      000130 R      #5-291
RES6      000132 R      #5-292
RH70    = 001000        #5-301
RSTRI     000112 R      #5-284
R6      =%000006        #4-142    4-145   #5-301
R7      =%000007        #4-143    4-144   #5-301
SBADR     000102 R      #5-277   *27-674  *27-683
SCAN      000416 R       8-340   #9-349
SCANC   = 076042        #4-160
SCANCI  = 076142        #4-193    9-351
SETUP     002412 R       7-326    8-336    9-349    10-368   11-390   12-406   13-421   14-434   15-456
                         16-477   17-507   18-522   19-553   20-564   21-579   22-600   23-611   24-639
                        #26-656
SET.10    003034 R      *9-350    9-351   *9-353   *10-369   10-370  *10-373  #29-736
SIGN      002766 R       21-590   21-592   21-594  #29-714
SIXTAB    002754 R       16-482   16-490  #29-711
SKIP      001204 R       14-447  #15-456
SKPC    = 076041        #4-159
SKPCI   = 076141        #4-192    15-458
SOFCNT    000042 R      #5-260
SOFERS  = 104406        #5-301
SOFPAS    000046 R      #5-262
SPAN      000522 R       9-359   #10-368
SPANC   = 076043        #4-161    10-371
SPANCI  = 076143        #4-194
SPOINT    000032 R      #5-256
SPSIZ   = 000040        #1-25     5-293
SRC.1A    002772 R       7-330    8-342   *9-356    9-361   *10-376   10-383  *11-393   11-399   12-414
                         13-427  *14-441   14-449  *15-463   15-469  *26-657  #29-720
SRC.1D    002770 R       7-327    8-337    8-339    9-351   *9-355    9-357    9-358    10-370  *10-375
                         10-377   10-379   11-391  *11-394   11-395   12-407   12-410   13-422   13-424
                         14-436  *14-440   14-446   15-458  *15-462   15-464   15-465  *26-656  #29-719
SR1       000016 R      #5-249
SR2       000020 R      #5-250
SR3       000022 R      #5-251
SR4       000024 R      #5-252
STACK     003676 R       6-309   #29-742
START     000234 R       5-255   #6-308
```



```
SYMBOL  VALUE           REFERENCES
STAT      000026 R      #5-254
STKLIM    003056 R      #29-741
SUBN    = 076051        #4-165    18-532
SUBNI   = 076151        #4-198    18-527
SUBP    = 076071        #4-181
SUBPI   = 076171        #4-206    23-614   23-615
SVR0      000062 R      #5-269
SVR1      000064 R      #5-270
SVR2      000066 R      #5-271
SVR3      000070 R      #5-272
SVR4      000072 R      #5-273
SVR5      000074 R      #5-274
SVR6      000076 R      #5-275
SYSCNT    000052 R      #5-264
TRANS     002534 R       13-422   13-423  #28-696   29-736   29-740
TRPDFD  = 000023        #5-301    5-301    5-301   #5-301    5-301    5-301   #5-301    5-301    5-301
                        #5-301    5-301    5-301   #5-301    5-301    5-301   #5-301    5-301    5-301
                        #5-301    5-301    5-301   #5-301    5-301    5-301   #5-301    5-301    5-301
                        #5-301    5-301    5-301   #5-301    5-301    5-301   #5-301    5-301    5-301
                        #5-301    5-301    5-301   #5-301    5-301    5-301   #5-301    5-301    5-301
                        #5-301    5-301    5-301   #5-301    5-301    5-301   #5-301    5-301    5-301
                        #5-301    5-301    5-301   #5-301
USTACK  = 000001        #5-301
VECTOR    000010 R      #5-245
WASADR    000104 R      #5-279   *27-677  *27-685
WOFR      000116 R      #5-286
WOTO      000114 R      #5-285
XFLAG     000005 R      #5-243
```

MACRO CROSS REFERENCE

MACRO NAME	REFERENCES									
BKMOD	#1-121									
BREAK	#1-219									
BTOD	#1-238									
CKDATA	#1-274									
CLKSP	#1-146									
DATACK	#1-283									
DATAER	#5-220	#8-343	#9-362	#10-384	#11-400	#12-415	#13-428	#14-450	#15-470	#18-545
	#19-561	#20-573	#23-632	#24-647						
DATERR	#1-172									
DFSEVN	#1-306	5-301	5-301	5-301	5-301	5-301	5-301	5-301	5-301	5-301
	5-301	5-301	5-301							
DSEVNT	#1-316	5-301	5-301	5-301	5-301	5-301	5-301	5-301		
END	#1-209									
ENDIT	#1-200									
ENDMOD	#1-205									
EQUATS	#1-322	#5-301								
EXIT	#1-154									
FATLER	#5-225	7-331	17-512	22-605						
GETPA	#1-265									
GWBUFF	#1-253									
HRDER	#1-162									
IOMOD	#1-117									
IOMODP	#1-141									
IOMODR	#1-137									
IOMODX	#1-133									
MAP22	#1-269									
MODULE	#1-26									
MSG	#1-188									
MSGN	#1-192									
MSGS	#1-196									
NBKMOD	#1-129									
OTOA	#1-224									
PIRQ	#1-213									
RAND	#1-158									
SBKMOD	#1-125									
SOFER	#1-178									
STARS	#5-217	#6-304	#6-306	#7-321	#7-324	#8-333	#8-335	#9-345	#9-347	#10-364
	#10-366	#11-386	#11-388	#12-402	#12-404	#13-417	#13-419	#14-430	#14-432	#15-452
	#15-454	#16-472	#16-475	#17-503	#17-505	#18-514	#18-518	#19-547	#19-551	#21-575
	#21-577	#22-596	#22-598	#23-607	#23-609	#24-634	#24-637	#26-651	#26-654	#27-662

IDENTIFICATION

PRODUCT NAME: AC-S866A-MC

PRODUCT NAME: CHQFSA0 XXDP+ FILE STRUCT DOC

PRODUCT DATE: APRIL 1981

MAINTAINER: DIAGNOSTIC SYSTEMS ENGINEERING

DOCUMENT: XXFORM.MEM

DATE: 17 FEB 1981

CHANGE NUMBER: P98-08823

XXDP+ FILE STRUCTURE

SPECIFICATION

REVISION 0.0

MAINTAINED BY: DIAGNOSTIC SYSTEM ENGINEERING

REVISION HISTORY

REVISION 0.0 FEB 17 1981

BY MICHAEL CINNAMON

CONTENTS

1.0 INTRODUCTION

THE STRUCTURE THAT XXDP+ USES FOR STORING FILES ON MEDIA IS UNIQUE TO XXDP+. THE STRUCTURE WAS ORIGINALLY BASED ON DOS-11 BUT IT HAS SINCE BEEN MODIFIED TO ACCOMMODATE THE NEEDS OF XXDP+, ALTHOUGH MANY SIMILARITIES STILL EXIST.

XXDP+ SUPPORTS BOTH RANDOM ACCESS AND SEQUENTIAL ACCESS TYPE DEVICES. A DIRECTORY INDEX STRUCTURE IS USED FOR ACCESSING FILES ON RANDOM ACCESS TYPE DEVICES SUCH AS DISKS. FOR SEQUENTIAL DEVICES LIKE MAGTAPE, A HEADER RECORD CONTAINING FILE INFORMATION PRECEDES EACH FILE.

2.0 DATA STRUCTURES

2.1 DATA BLOCKS

THE BASIC UNIT OF DATA TRANSFERRED IN XXDP+ FILE I/O IS A DATA BLOCK. A DATA BLOCK IS DEFINED AS A GROUP OF 512(10) 8 BIT BYTES.

2.1.1 RANDOM ACCESS DEVICES.

ON RANDOM ACCESS DEVICES, DATA BLOCKS ARE ADDRESSED BY THEIR LOGICAL BLOCK NUMBER. LOGICAL BLOCKS ARE DATA BLOCKS THAT ARE ADDRESSABLE IN A LINEAR ORDERED FASHION. THE ORDER RANGES FROM 0 TO N-1, WHERE N IS THE MAXIMUM NUMBER OF BLOCKS SUPPORTED ON THE DEVICE. THE VARIABLE N VARIES DEPENDING ON THE DEVICE BUT IS NOT GREATER THAN 65535 (10). THIS MEANS THAT NO MATTER HOW MANY DATA BLOCKS PHYSICALLY RESIDE ON A DEVICE ONLY THE FIRST N BLOCKS ARE ACCESSIBLE.

LOGICAL BLOCKS ARE NOT NECESSARILY THE SAME AS PHYSICAL BLOCKS. A PHYSICAL BLOCK IS USUALLY LIMITED TO THE AMOUNT OF DATA A PHYSICAL SECTOR CONTAINS WHICH MAY BE LESS THAN 512 (10) BYTES. ALSO A PHYSICAL BLOCK HOUSES AT MOST ONE LOGICAL BLOCK.

A LINKED LIST OF LOGICAL BLOCKS IS SET UP USING LINKED BLOCKS. A LINKED BLOCK IS A LOGICAL BLOCK THAT DEVOTES THE FIRST WORD OF THE BLOCK TO CONTAIN A LINK WORD. THE LINK WORD CONTAINS THE LOGICAL BLOCK NUMBER OF ANOTHER BLOCK. A ZERO LINK WORD INDICATES THAT THE BLOCK CONTAINING IT IS THE LAST BLOCK OF THE LIST.

2.1.2 SEQUENTIAL ACCESS DEVICES

ON SEQUENTIAL ACCESS DEVICES SUCH AS MAGTAPE AND CASSETTE, DATA BLOCKS ARE STORED AS 512 BYTE RECORDS. BECAUSE XXDP+ USES THE SAME READ/WRITE ROUTINES FOR BOTH RANDOM AND SEQUENTIAL ACCESS DEVICES, LOGICAL LINKED BLOCKS ARE STORED ON SEQUENTIAL TYPE DEVICES.

HOWEVER, THE LINK WORD IS ONLY SOME NON-ZERO VALUE TO INDICATE THAT THERE ARE MORE BLOCKS IN THE LIST. A ZERO LINK WORD INDICATES THE LAST BLOCK OF THE LIST.

3.0 FILE STRUCTURES

3.1 TYPES OF FILES

THERE ARE THREE TYPES OF FILES SUPPORTED BY XXDP+. THEY ARE CONTIGUOUS FILES, TEXT FILES AND BINARY FILES. THIS IS NOT TO MEAN THAT OTHER TYPES OF FILES CANNOT BE STORED ON AN XXDP+ MEDIUM, BUT XXDP+ ONLY HAS THE CAPABILITY TO PRODUCE THESE TYPES.

3.1.1 CONTIGUOUS FILES

A CONTIGUOUS FILE IS A SET OF LOGICAL BLOCKS WHICH PHYSICALLY RESIDE IMMEDIATELY ADJACENT TO ONE ANOTHER ON THE MEDIA. THE TERM 'IMMEDIATELY ADJACENT' IS OBVIOUS FOR SEQUENTIAL DEVICES BUT NOT QUITE AS APPARENT FOR RANDOM ACCESS DEVICES. FOR THESE IT MEANS FOR ANY LOGICAL BLOCK N, THE NEXT CONTIGUOUS BLOCK IS LOCATED AT N+1.

CONTIGUOUS FILES ARE NORMALLY USED TO STORE CORE IMAGE DATA SUCH AS THE XXDP+ MONITOR.

3.1.2 TEXT FILES

TEXT FILES ARE MADE UP OF A SERIES OF LINKED BLOCKS. EACH BLOCK CONTAINS 510(10) 8 BIT ASCII CHARACTERS. AN ASCII NULL CHARACTER (A BYTE WITH A VALUE OF ZERO) IS USED TO INDICATE THE END OF THE FILE.

3.1.3 BINARY FILES

BINARY FILES ARE USED TO STORE EXECUTABLE PROGRAMS. THEY ARE MADE UP OF A SERIES OF LINKED BLOCKS EACH OF WHICH CONTAIN SECTIONS OF THE PROGRAM. THESE SECTIONS ARE IN ABSOLUTE FORMATTED BINARY AND THERE IS AT LEAST ONE PER LOGICAL BLOCK.

THE ABSOLUTE FORMATTED BINARY SPECIFICATION IS AS FOLLOWS:

BYTE 1 - CONTAINS A VALUE OF 1 TO INDICATE STARTING POINT.

BYTE 2 - CONTAINS A VALUE OF 0. THIS MUST FOLLOW BYTE 1.

BYTES 3 AND 4 - CONTAINS NUMBER OF BYTES (N) IN THIS BINARY BLOCK. IT INCLUDES BYTES 1, 2, 3 AND 4 BUT EXCLUDES THE CHECKSUM BYTE.

BYTES 5 AND 6 - CONTAINS THE STARTING MEMORY ADDRESS WHERE THE FOLLOWING DATA BYTES ARE TO BE STORED.

BYTES 7 TO N - DATA BYTES. N <= 509. THE MAXIMUM NUMBER OF DATA BYTES IS 503.

BYTE N+1 - CONTAINS THE CHECKSUM BYTE. THE CHECKSUM IS THE 2'S COMPLEMENT OF THE SUM OF THE DATA IN ALL N BYTES. IT IS GENERATED IGNORING OVERFLOW AND CARRY CONDITIONS.

THE END OF A BINARY FILE IS INDICATED BY A BINARY BLOCK THAT HAS A BYTE COUNT OF 6. BYTES 5 AND 6 CONTAIN THE PROGRAM'S TRANSFER ADDRESS. THIS BLOCK IS KNOWN AS THE 'TRANSFER BLOCK'.

A BINARY BLOCK WITH A BYTE COUNT OF 5 INDICATES A 'BIAS BLOCK'. BITS 0 AND 1 OF BYTE 5 REPRESENT BITS 16 AND 17, RESPECTIVELY, OF THE LOAD ADDRESS FOR THE NEXT BINARY BLOCK.

4.0 MEDIA STRUCTURE

4.1 RANDOM ACCESS STRUCTURE

ALL XXDP+ RANDOM ACCESS DEVICES ARE SET UP TO CONTAIN THE FOLLOWING PIECES OF INFORMATION: BOOTSTRAP, MONITOR CORE IMAGE, MASTER FILE DIRECTORIES (MFD), USER FILE DIRECTORIES (UFD), AND BIT MAPS.

THE BOOTSTRAP IS A PROGRAM THAT ALWAYS OCCUPIES LOGICAL BLOCK 0 ON THE DEVICE. IT IS A CORE IMAGE THAT IS PLACED THERE BY THE UTILITY COMMAND 'SAVM'. THE BOOTSTRAP KNOWS WHERE ON THE DISK THE MONITOR CORE IMAGE RESIDES.

THE MONITOR CORE IMAGE IS A CONTIGUOUS FILE THAT IS 16 BLOCKS LONG. IT IS PLACED ON THE DISK BY THE UTILITY COMMAND 'SAVM'. ITS POSITION DEPENDS ON THE TYPE OF DISK. THE MFD IS A TABLE OF INFORMATION WHICH CONTAINS POINTERS TO THE UFDS AND THE BIT MAPS.

4.1.1 MASTER FILE DIRECTORY

THE MFD IS PLACED ON THE DISK BY THE UTILITY 'ZERO' COMMAND. THE TYPE OF DISK DETERMINES WHICH OF TWO VARIETIES OF MFD IS USED AND WHERE IT IS ON THE DEVICE.

MFD VARIETY #1

MFD1:		WORD OFFSET INTO BLOCK
	LINK TO MFD2	0
	INTERLEAVE FACTOR	1
	BIT MAP START BLOCK #	2
	POINTER TO BIT MAP #1	3
	POINTER TO BIT MAP #2	4
	.	
	.	
	.	
	POINTER TO BIT MAP #N	N+2
	0	N+3
	UNUSED	N+4 TO 255

THE INTERLEAVE FACTOR IS USED AS PART OF THE BLOCK ALLOCATION ALGORITHM. NORMALLY BLOCKS ARE ALLOCATED CONTIGUOUSLY IF POSSIBLE BUT ON CERTAIN DEVICES, THE AMOUNT OF TIME IT TAKES TO ACCESS A LINKED LIST OF BLOCKS IS REDUCED BY PLACING EACH BLOCK OF THE LIST A CONSTANT NUMBER OF BLOCKS APART. THIS CONSTANT NUMBER IS THE INTERLEAVE FACTOR. THE POINTERS IN THE TABLE ARE THE LOGICAL BLOCK NUMBERS OF THE RESPECTIVE BIT MAPS.

MFD2:		WORD OFFSET INTO BLOCK
	0	0
	401	1
	POINTER TO FIRST UFD BLOCK	2
	9 (10)	3
	0	4
	UNUSED	5-255

THE FIRST WORD IS A LINK OF ZERO INDICATING NO MORE MFDS. THE SECOND WORD CORRESPONDS TO THE DOS-11 UIC (1,1). THE THIRD IS THE LOGICAL BLOCK NUMBER OF THE FIRST UFD BLOCK. THE FOURTH IS THE NUMBER OF WORDS IN EACH UFD ENTRY; THIS IS ALWAYS NINE (9).

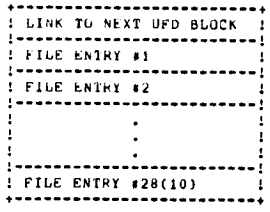
MFD VARIETY #2

MFD 1/2:		WORD OFFSET INTO BLOCK
	0	0
	POINTER TO FIRST UFD BLOCK	1
	# OF UFD BLOCKS	2
	POINTER TO FIRST BIT MAP BLOCK	3
	# OF BIT MAP BLOCKS	4
	POINTER TO MFD 1/2	5
	0	6
	NUMBER OF SUPPORTED BLOCKS	7
	# OF BLOCKS PREALLOCATED	8
	INTERLEAVE FACTOR	9
	0	10
	POINTER TO FIRST BLOCK OF MONITOR CORE IMAGE	11
	0	12
	TRACK + SECTOR ADDRESS FOR BAD SECTOR FILE (SINGLE DENSITY	13
	CYLINDER ADDRESS FOR BAD SECTOR FILE (SINGLE DENSITY)	14
	TRACK + SECTOR ADDRESS FOR BAD SECTOR FILE (DOUBLE DENSITY	15
	CYLINDER ADDRESS FOR BAD SECTOR FILE (DOUBLE DENSITY	16

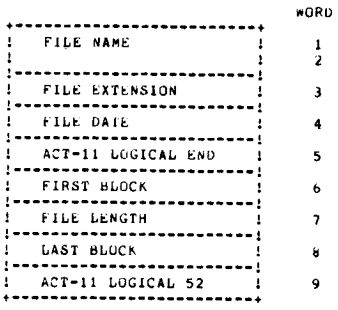
THE ENTRY AT OFFSET 8 MEANS THE NUMBER OF BLOCKS ON THE DEVICE RESERVED FOR DEVICE STRUCTURE INFORMATION. THE LAST 4 ENTRIES IN THE TABLE REFER TO THE LOCATION ON THE DISK OF THE DEC STD 144 BAD SECTOR FILE.

4.1.2 USER FILE DIRECTORY

THE USER FILE DIRECTORY (UFD) IS A LIST OF THE FILES ON THE MEDIA. IT IS CREATED BY THE UTILITY 'ZERO' COMMAND. THE UFD IS ARRANGED AS A LINKED LIST OF LOGICAL BLOCKS AND THE NUMBER OF BLOCKS THAT THE UFD OCCUPIES DEPENDS ON THE DEVICE. EACH BLOCK OF THE UFD CONTAINS SPACE FOR 28(10) FILE ENTRIES.



EACH FILE ENTRY IS A TABLE OF 9(10) WORDS THAT CONTAINS THE FOLLOWING INFORMATION ABOUT THE FILE.



WORDS 1, 2 AND 3 - THE FILE'S SIX CHARACTER FILENAME AND 3 CHARACTER EXTENSION ENCODED IN RAD-50. A DELETED FILE IS INDICATED BY A ZERO IN THESE THREE WORDS.

WORD 4 - THE DOS-11 FORMAT FOR THE DATE GIVEN THE FILE WHEN IT WAS PUT ON THE MEDIA.

WORD 5 AND 9 - ACT-11 USE ONLY. NOT USED IN XXDP+.

WORD 6 - THE BLOCK NUMBER OF THE FIRST LOGICAL BLOCK THAT THE FILE OCCUPIES.

WORD 7 - THE NUMBER OF LOGICAL BLOCKS THAT THE FILE OCCUPIES.

WORD 8 - THE BLOCK NUMBER OF THE LAST LOGICAL BLOCK THAT THE FILE OCCUPIES.

4.1.3 BIT MAP

THE BIT MAP IS A FILE THAT CONTAINS THE CURRENT STATUS OF EVERY SUPPORTED LOGICAL BLOCK ON THE MEDIA.

THE BIT MAP IS ARRANGED AS A LINKED LIST OF LOGICAL BLOCKS. THE NUMBER OF BLOCKS THAT THE MAP OCCUPIES DEPENDS ON THE DEVICE. IT IS CREATED BY THE UTILITY 'ZERO' COMMAND. ONLY THE FIRST 64 WORDS OF EACH MAP BLOCK HAVE MEANING.

	WORD
LINK TO NEXT MAP BLOCK	1
MAP NUMBER	2
60(10)	3
LINK TO FIRST MAP	4
MAP FOR BLOCKS 0-15 (10)	5
MAP FOR BLOCKS 16-31(10)	6
.	.
.	.
.	.
MAP FOR BLOCKS 944-959(10	64

WORD 1 - THE LOGICAL BLOCK NUMBER OF THE NEXT MAP BLOCK CONTAINS ZERO IF IT IS THE LAST MAP.

WORD 2 - WHICH MAP THIS ONE IS.

WORD 3 - NUMBER OF WORDS USED FOR MAP.

WORD 4 - THE LOGICAL BLOCK NUMBER OF FIRST BIT MAP.

WORDS 5-64 - MAP FOR 960 BLOCKS. BIT IS SET WHEN BLOCK IS USED. BIT IS CLEARED WHEN BLOCK IS FREE.

WORDS 65-255 - NOT USED.

4.1.4 RANDOM ACCESS DEVICE INFORMATION

DEVICE	MNEMONIC	1ST UFD BLOCK #	# OF UFD BLOCKS	1ST BIT MAP BLK	# OF MAPS	MFD1	MFD2
TU58	DD	3	4	7	1	1	2
RP04,5,6	DB	3	170.	173.	50.	1	2
RK03,5	DK	3	16.	4795.	5.	1	4794.
RL01/2	DL	24.	146.	2	22.	1	-
RK06,7	DM	31.	96.	2	29.	1	-
RP02,3	DP	3	170.	173.	50.	1	2
RM03	DR	52.	170.	2.	50.	1	-
RS03,4	DS	3	4	7	2	1	2
TU56	DT	102	2	104	1	100	101
RX01	DX	3	4	7	1	1	2
RX02	DY	3	16.	19.	4	1	2

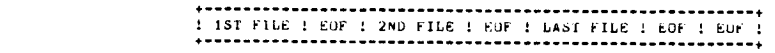
DEVICE	# OF BLKS ON DEV	# OF BLKS TO PREALLOCATE	INTER-LEAVE	BOOT BLK #	MONITOR BLOCK #
TU58	511.	40.	1	0	8.
RP04,5,6	48000.	255.	1	0	223.
RK03,5	4800.	69.	5	0	30
RL01/2	20460. OR 10200.	200.	1	0	170.
RK06,7	27104.	157.	1	0	127.
RP02,3	48000.	255.	1	0	223.
RM03	48000.	255.	1	0	222.
RS03,4	989.	41.	1	0	9.
TU56	576.	69.	5	0	30
RX01	494.	40.	1	0	8.
RX02	988.	55.	1	0	23.

4.2 SEQUENTIAL ACCESS DEVICES

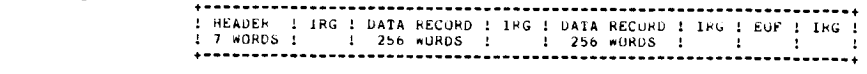
4.2.1 MAGTAPE

ALTHOUGH MAGTAPE IS NOT USUALLY CONSIDERED AS A FILE STRUCTURED DEVICE, CERTAIN STRUCTURE FEATURES ARE IMPLEMENTED TO ENABLE CREATION AND RETRIEVAL OF MULTIPLE FILES.

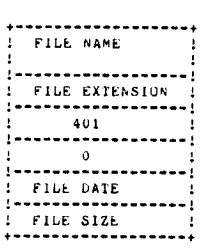
THE FILES ON MAGTAPE ARE TERMINATED BY AN END-OF-FILE MARK (EOF) OR TAPE MARK (TM). THE LAST FILE ON THE TAPE IS TERMINATED BY 2 CONSECUTIVE EOFS TO INDICATE LOGICAL-END-OF-TAPE.



EACH FILE ON MAGTAPE IS MADE UP OF A HEADER AND DATA RECORDS.



THE HEADER RECORD IS STRUCTURED AS FOLLOWS:



WORD 1, 2 AND 3 - THE FILENAME AND EXTENSION ENCODED IN RAD-50.

WORD 4 - DOS-11 UIC [1,1]

WORD 5 - SET TO ZERO.

WORD 6 - DATE GIVEN THE FILE WHEN WRITTEN ON THE TAPE. IT IS IN DOS-11 FORMAT. BIT 15 OF THE DATE, WHEN SET, INDICATES A CONTIGUOUS FILE.

WORD 7 - THE NUMBER OF LOGICAL BLOCKS (RECORDS) IN THE FILE.

THE FIRST FILE ON A MAGTAPE IS NORMALLY THE XXDP+ MONITOR CORE IMAGE. IT IS WRITTEN AS A CONTIGUOUS FILE. EVERY NEW FILE IS WRITTEN AT THE LOGICAL END OF TAPE. THE LAST FILE ON TAPE MAY BE WRITTEN SO PART LIES AFTER THE PHYSICAL EOT MARKER BUT NO FILE WILL BE WRITTEN ENTIRELY AFTER THE PHYSICAL EOT MARK.

THE NUMBER OF FILES THAT A MAGTAPE CAN ACCOMMODATE IS A FUNCTION OF THE FILE SIZE, THE DRIVE DENSITY AND THE TAPE LENGTH.

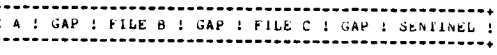
SEQ 0015

SEQ 0016

4.2.2 CASSETTE (TU60)

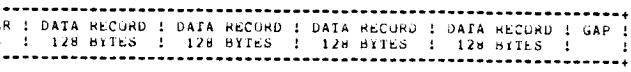
CASSETTE IS STRUCTURED SIMILARILY TO MAGTAPE IN THAT EACH FILE IS PRECEDED BY A HEADER AND A MARKER THAT IDENTIFIES THE LOGICAL END OF TAPE. HOWEVER, THE ACTUAL DATA IN THE HEADER AND END-OF-FILE MARKER ARE DIFFERENT.

THE FILES ON CASSETTE ARE TERMINATED BY A FILE GAP. THE LAST FILE ON THE TAPE IS TERMINATED BY A SENTINEL FILE. A FILE GAP MUST PRECEDE THE FIRST FILE ON THE CASSETTE. THE TAPE IS FORMATTED AS FOLLOWS:



THE SENTINEL FILE IS A 32 BYTE RECORD CONTAINING ALL ZERO'S.

EACH FILE ON CASSETTE IS MADE UP OF A FILE HEADER RECORD AND FILE DATA RECORDS IN MULTIPLES OF 4.



THE FILE HEADER RECORD IS STRUCTURED AS FOLLOWS:

SEQ 0017

	BYTE
FILE NAME	1-6
FILE EXTENSION	7,8,9
FILE TYPE	10
100000	11,12
0	13-16
DATE	17,18
FILE LENGTH	19,20
0	21,32

BYTES 1-9 - FILENAME AND EXTENSION ENCODED IN RAD-50.

BYTE 10 - FILE TYPE INDICATOR. 0=6000, 14=DELETED.

BYTES 11,12 - WORD INDICATING RECORD LENGTH SET TO 100,000 (128 BYTE RECORDS).

BYTE 13 - SEQUENCE NUMBER NOT USED IN XXDP+.

BYTE 14 - HEADER CONTINUATION NOT USED IN XXDP+.

BYTES 15,16 - NOT USED.

BYTES 17,18 - DOS-11 FORMATTED DATE GIVEN THE FILE WHEN PUT ON THE TAPE.

BYTES 19,20 - NUMBER OF LOGICAL BLOCKS IN THE FILE. HAS MEANING FOR CONTIGUOUS FILES ONLY.

BYTES 21-32 - NOT USED.

SEQ 0018

IRG - INTERRECORD GAP. THE GAP THAT IS WRITTEN BETWEEN RECORDS ON MAGTAPE.

MFD - MASTER FILE DIRECTORY

RAD-50 - RADIX-50. A METHOD OF ENCODING 3 ASCII CHARACTERS INTO ONE 16 BIT WORD.

UFD - USER FILE DIRECTORY.

UIC - USER IDENTIFICATION CODE.

```
000000                  .REPT   0
```

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

```
                    ******************
                    *                *
                    * DEC/X11 MODULE *
                    *                *
                    ******************
```

PRODUCT CODE: AC-S330A-MC

PRODUCT NAME: CXCMJA0 CMR11 DEC/X11 MODULE

PRODUCT DATE: 18-NOV-80

MAINTAINER: SPECIAL SYSTEMS, KANATA (002)

```
1                               HISTORY:
2                               --------
3
4
5                               CXCMJA0         18-NOV-80       FIRST RELEASE- NEW OPTION       D.G.W.
```

TABLE OF CONTENTS:

1. ABSTRACT

THE PURPOSE OF THIS MODULE IS TO TEST SYSTEM INTEGRATION OF THE CMR11 HOST MODULE (M8990) ON THE UNIBUS BY TRANSFERRING DATA TO AND FROM THE DEVICE AND CAUSING NON SYNCHRONOUS INTERRUPTS IN A SYSTEM ENVIRONMENT.

THE MODULE MUST EASILY RECOVER FROM A POWER FAILURE EVEN THOUGH THE CMR11 FIRMWARE COMPLETELY RESTARTS UPON A POWER-UP.

IT IS NOT THE PURPOSE OF THIS MODULE TO GENERATE ANY HOST-TO-REMOTE ACTIVITY OTHER THAN TO INITIATE A "BACKGROUND SCAN" SO THAT THE L.E.D.'S ON THE MODEM MODULES "BLINK" AND SO THAT THE HOST MAY DETECT ANY INCOMING REMOTE "ALARMS" ON PORT 0.

IF THE REMOTES HAD BEEN SET UP PRIOR TO RUNNING DEC/X11 TO SCAN AND REPORT ALARMS, THEY WILL CONTINUE TO DO SO BUT THE REPORTING OF THEM TO THE USER BY DEC/X11 WILL NOT BE DETAILED. A SIMPLE ALARM MESSAGE WILL BE PRINTED INDICATING WHICH REMOTE HAD AN ALARM.

IF THIS MODULE IS DROPPED, "BACKGROUND SCAN" WILL BE CLEARED AND THE HOST INITIALIZED SO THAT NO FURTHER COMMUNICATION WILL TAKE PLACE.

2. REQUIREMENTS

HARDWARE:	PDP-11 PROCESSOR WITH A CMR11 HOST MODULE (M8990) ON THE UNIBUS.
OTHER:	THIS MODULE IS MEANT TO BE CONFIGURED AND RUN WITH DEC/X11 ONLY.
	ALL PROCESSOR DIAGNOSTICS AND CMR11 HOST DIAGNOSTICS MUST BE RUN SUCCESSFULLY PRIOR TO RUNNING THIS MODULE.

```
                3.      ASSUMPTIONS
                        -----------


                  IT IS ASSUMED THAT THE CMR11 HOST FIRMWARE IS INSTALLED AND
                RUNNING PROPERLY IN THE TWO 'EPROM' CHIPS E66 AND E61 ON THE M8990.
                  IT IS ALSO ASSUMED THAT IF ANY REMOTE "ALARMS" OCCUR, THEY WILL
                BE CLEARED MANUALLY OR IGNORED. THIS MODULE WILL PRINT AN ALARM
                MESSAGE REPETEDLY AS LONG AS AN ALARM CONDITION EXISTS ON ANY OF
                THE REMOTES ON PORT 0.
                  THE ALARM MAY BE IGNORED BY DISCONNECTING THE COMMUNICATION
                CABLE FROM THE PLUG FOR THE PORT CONNECTED TO THE ALARM-ING
                REMOTE, OR BY SETTING BIT 0 IN SR2 ( CMAA0 20 ).


                4.      OPERATING INSTRUCTIONS
                        ----------------------

                4.1       LOADING AND STARTING:

                  THIS IS A DEC/X11 OBJECT MODULE WHICH MUST BE CONFIGURED AS
                PART OF A DEC/X11 'RUN-TIME' SYSTEM USING THE DEC/X11 RUN TIME
                MONITOR. REFERENCE DXQBA DEC/X11 USERS DOCUMENTATION AND
                REFERENCE GUIDE.

                4.2       ENVIRONMENT:

                  THIS PROGRAM RUNS UNDER DEC/X11 ONLY.

                4.3       OPTIONS:

                SR1 BIT 15 SET (1)      IF ERROR RETRY LIMIT IS EXCEDED, RESET
                                        THE RETRY COUNT AND CONTINUE. (DON'T DROP)

                SR1 BIT 15 CLR (0)      IF ERROR RETRY LIMIT IS EXCEDED, A HARD
                                        ERROR IS ASSUMED AND THE MODULE IS DROPPED.

                SR2 BIT 0 SET (1)       IF A REMOTE ALARM OCCURS, READ IT BUT DON'T
                                        PRINT ANYTHING AS A RESULT. (INHIBIT CMR ALARM
                                        REPORT).

                SR2 BIT 0 CLR (0)       IF A REMOTE ALARM OCCURS, READ IT AND PRINT
                                        ALARM MESSAGE GIVING REMOTE NUMBER.

                4.4       EXECUTION TIMES:

                (STANDALONE) = ABOUT 49 SECONDS PER PASS
```

```
                5.      MODULE OPERATION
                        ----------------

                5.1       CONFIGURATION REQUIREMENTS:

                DEFAULT PARAMETERS:

                        DEVICE ADDRESS: 170000
                        DEVICE VECTOR:  170
                        PRIORITY:       4
                        OPTIONS (SR1):  000000  (DROP MODULE IF RETRY LIMIT EXCEEDED)
                                (SR2):  000000  (PRINT ON REMOTE ALARMS)

                REQUIRED PARAMETERS:

                        (NONE)

                5.2       TEST SEQUENCE:

                A.      SET UP VECTOR, ADDRESSES, MODULE VARIABLES

                B.      CHECK IF MODULE TO BE DROPPED:
                         -WAIT FOR PWR-UP COMPL.
                         -DO "DIE" FUNCTION
                         -READ CSR  SHOULD BE 137600
                         -IF NOT AFTER 777 COUNTS, EXIT TO DROP MODULE
                         -IF YES, INIT DEVICE AND CONTINUE.

                C.      PERFORM CHECK OF POWER UP BIT TO SEE IF AN INTERRUPT
                        IS ON ITS WAY. IF SET, EXIT WITH INTR ENAB SET; IF
                        CLR, CONTINUE TO 'D.'

                D.      SET INTR ENAB & BACKGROUND SCAN.
                        DO A DATA IN-OUT TEST ON REQUEST SCRATCHPAD MEMORY
                        WITH BREAKS TO MONITOR.
                        LOOK AT "ALMINT" SOFTWARE FLAG. IF SET, PRINT CMR11
                        ALARM MESSAGE, THEN CLEAR "ALMINT".

                E.      PERFORM AN INIT OF THE CMR11 TO START ITS POWER-UP
                        SEQUENCE. THEN SET BACKGROUND SCAN.

                F.      GO TO "C".

                UPON INTERRUPT:
                ---------------

                1.      CHECK FOR ALARM INTERRUPT, AND IF ANY, SET "ALMINT"
                        FLAG AND READ REMOTE ADDRESS FROM SCRATCHPAD.;
                        THEN PERFORM "PIRQ" TO QUEUE UP RETURN TO "2" (BELOW).

                2.      CHECK FOR POWER-UP AND INTERRUPT BITS

                3.      IF SET, DO DATA IN-OUT TEST ON THE RESULT SCRATCHPAD
                        MEMORY WITH BREAKS TO MONITOR. THEN DO CLR TOP BYTE
                        AND SIGNAL ***END OF PASS***.

                4.      -IF PWR-UP CLR, FLAG AN ERRONEOUS INTERRUPT.
```

```
58
59                              5.      GO TO "C" ABOVE.
```

```
                        6.      ERROR REPORTS
                                -------------

                                A.      THE STANDARD DEC/X11 ERROR ROUTINES ARE USED IN THIS
                                        MODULE.

                                B.      ERROR REPORTS WHICH DUMP AN EXTENDED PRINTOUT OF THE
                                        DEVICE REGISTERS, DO SO IN THE FOLLOWING ORDER:

                                        CSR     SPAR    SPDR

                                C.      REMOTE "FIELD ALARMS" ARE INDICATED AS FOLLOWS:

                                        --->CMR11 ALARM. REMOTE #NNN<---

                        .ENDR
```

```
                                ;*****************************************************************
                                ;
                                ;CMR11 DEC/X11 SYSTEM EXERCISER MODULE
                                ;
                                        .TITLE  CMAA  DEC/X11 SYSTEM MODULE
                                ;
000000                          IOMOD   <CMAA >,170000,170,4,0,0,8.,0
000000                                  MODULE  140000,CMAA ,170000,170,4,0,0,8.,0
                                        .TITLE  CMAA DEC/X11 SYSTEM EXERCISER MODULE
                                ;       DDXCOM  VERSION  6      23-MAY-78
                                        .LIST   BIN
                                ;*****************************************************************
000000                          BEGIN:
000000  103  115  101           MODNAM: .ASCII  /CMAA / ;MODULE NAME.
000003  101  040
000005  000                     XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF WBUFF USAGE
000006  170000                  ADDR:   170000+0                ;1ST DEVICE ADDR.
000010  000170                  VECTOR: 170+0                   ;1ST DEVICE VECTOR.
000012  200                     BR1:    .BYTE   PRTY4+0         ;1ST BR LEVEL.
000013  000                     BR2:    .BYTE   PRTY0+0         ;2ND BR LEVEL.
000014  000001                  DVID1:  0+1                     ;DEVICE INDICATOR 1.
000016  000000                  SR1:    OPEN                    ;SWITCH REGISTER 1
000020  000000                  SR2:    OPEN                    ;SWITCH REGISTER 2
000022  000000                  SR3:    OPEN                    ;SWITCH REGISTER 3
000024  000000                  SR4:    OPEN                    ;SWITCH REGISTER 4
                                ;*****************************************************************
000026  140000                  STAT:   140000                  ;STATUS WORD.
000030  000224'                 INIT:   START                   ;MODULE START ADDR.
000032  000224'                 SPOINT: MODSP                   ;MODULE STACK POINTER.
000034  000000                  PASCNT: 0                       ;PASS COUNTER.
000036  000010                  ICONT:  8.                      ;# OF ITERATIONS PER PASS=8.
000040  000000                  ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
000042  000000                  SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
000044  000000                  HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
000046  000000                  SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
000050  000000                  HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
000052  000000                  SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
000054  000000                  RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000056                          CONFIG:                         ;RESERVED FOR MONITOR USE
000056  000000                  RES1:   0                       ;RESERVED FOR MONITOR USE
000060  000000                  RES2:   0                       ;RESERVED FOR MONITOR USE
000062  000000                  SVR0:   OPEN                    ;LOC TO SAVE R0.
000064  000000                  SVR1:   OPEN                    ;LOC TO SAVE R1.
000066  000000                  SVR2:   OPEN                    ;LOC TO SAVE R2.
000070  000000                  SVR3:   OPEN                    ;LOC TO SAVE R3.
000072  000000                  SVR4:   OPEN                    ;LOC TO SAVE R4.
000074  000000                  SVR5:   OPEN                    ;LOC TO SAVE R5.
000076  000000                  SVR6:   OPEN                    ;LOC TO SAVE R6.
000100  000000                  CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
000102                          SBADR:                          ;ADDR OF GOOD DATA, OR
000102  000000                  ACSR:   OPEN                    ;CONTENTS OF CSR.
000104                          WASADR:                         ;ADDR OF BAD DATA, OR
000104  000000                  ASTAT:  OPEN                    ;STATUS REG CONTENTS.
000106                          ERRTYP:                         ;TYPE OF ERROR
000106  000000                  ASB:    OPEN                    ;EXPECTED DATA.
000110  000000                  AWAS:   OPEN                    ;ACTUAL DATA.
000112  000224'                 RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
```



```
000114  000000                  WDTO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
000116  000000                  WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
000120  000000                  INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
000122  000000                  IDNUM:  0                       ;MODULE IDENTIFICATION NUMBER=0
        000040                          .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000224                          MODSP:
                                ;*****************************************************************
                                ;DEFINITIONS
                                ;
000000                          REQAD   =       0
001000                          RESAD   =       1000
002000                          PWRUP   =       2000
010000                          ALMDET  =       10000
100000                          INT     =       100000
000200                          MAIN    =       200
000100                          IE      =       100
000020                          CLRTPB  =       20
000004                          ENABKS  =       4
                                ;
                                ;---------------------------------------------------------------
```

```
 1                                      ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 2
 3                                      ;START AND RESTART AT SAME LOCATION:
 4
 5 000224                               START:                                  ;STARTING POINT
 6 000224  005767  177610               RESTRT: TST     ICOUNT                  ;ESPECIALLY FOR DT03,DT07
 7 000230  001024                               BNE     ENT2                    ;-*-*-*-*-*-*-*-*-*-*-*-*-
 8 000232  004767  000620                       JSR     PC,SETUP                ;GEN. DEVICE ADDRESSES
 9 000236  012777  000002  001450               MOV     #2,@CSR                 ;INIT CMR11 ONLY
10 000244  104407  000000'                      BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
   000250  104407  000000'                      BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
11 000254  104407  000000'                      BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
   000260  104407  000000'                      BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
12 000264  004767  000666                       JSR     PC,TSTDRP               ;CHECK IF MODULE TO BE DROPPED
13 000270  105767  001431                       TSTB    DROP
14 000274  001402                               BEQ     ENT2                    ;IF NOT, BEGIN TESTING
15 000276  000167  000464                       JMP     FINI                    ;GO DROP MODULE
16 000302                              ENT2:
   000302  104407  000000'                      BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
   000306  104407  000000'                      BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
17 000312  104407  000000'                      BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
   000316  104407  000000'                      BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
18 000322  105067  001400                       CLRB    ERFLG                   ;CLEAR ERROR INDICATOR
19 000326  032777  002000  001360               BIT     #PWRUP,@CSR             ;IS POWER UP BIT SET?
20 000334  001405                               BEQ     ENT3                    ;NO, DO REQ. MEM TEST
21 000336  052777  000104  001350               BIS     #IE!ENABKS,@CSR         ;SET INTR EN & BKG SCAN
22 000344  104400  000000'                      EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
23
24 000350  052777  000104  001336  ENT3:   BIS     #IE!ENABKS,@CSR         ;SET THEM HERE ALSO
25 000356  012767  000000  001322               MOV     #REQAD,MPOINT           ;SET MEM POINTER TO REQ. MEM
26 000364  004767  001036                       JSR     PC,MEMTST               ;GO DO SCR PAD MEM TEST
27 000370  104407  000000'                      BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
   000374  104407  000000'                      BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
28 000400  105767  001322                       TSTB    ERFLG                   ;ANY ERRORS?
29 000404  001404                               BEQ     ENT4                    ;NO, CONTINUE
30 000406  012746  000350'                      MOV     #ENT3,-(SP)             ;YES, PUSH RETRY ADDRESS
31 000412  000167  000366                       JMP     RETRY                   ;AND RETRY UP TO LIMIT
32
33 000416  105767  001302             ENT4:   TSTB    ALMINT                  ;IS ALARM FLAG SET?
34 000422  001415                               BEQ     ENT5                    ;NO, CONTINUE
35                                              ;*************************************************************
                                                ;CONVERT REMNM TO ASCII AND
                                                ;STORE AT MSG2
   000424  104420  000000' 001710'              OTOA$,BEGIN,REMNM,MSG2
   000432  002032'
                                                ;*************************************************************
36 000434  032767  000001  177356               BIT     #1,SR2          ;CHECK IF ALLOWED TO PRINT ALARM
37 000442  001003                               BNE     1$              ;NOT IF BIT 0 OF SR2 IS SET!!
38 000444  104403  000000' 001734'              MSGN$,BEGIN,REMALM      ;ASCII MESSAGE CALL WITH COMMON HEADER
39 000452  105067  001246             1$:     CLRB    ALMINT                  ;CLR FLAG
40
41 000456  012777  000020  001230  ENT5:   MOV     #CLRTPB,@CSR            ;CLR TOP BYTE
42 000464  012777  000002  001222               MOV     #2,@CSR                 ;INIT CMR11
43 000472  012777  000004  001214               MOV     #ENABKS,@CSR            ;SET BACKGROUND SCAN
44 000500  104407  000000'                      BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
   000504  104407  000000'                      BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
45 000510  104407  000000'                      BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
```



```
   000514  104407  000000'                      BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
46 000520  104407  000000'                      BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
   000524  104407  000000'                      BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
47 000530  000167  177546                       JMP     ENT2
48
49                                      ;-------------------------------------------------------------------
```

```
 1                                      ;*****************************************************************
 2
 3                                      ;INTERRUPT ENTRY POINT.
 4                                      ;THIS ROUTINE CHECKS FOR EXPECTED TYPES OF INTERRUPTS
 5                                      ;AND FLAGS OTHER TYPES AS ERRONEOUS.
 6                                      ;
 7                                      ;EXPECTED TYPES ARE: PWR-UP AND ALM DET.
 8
 9 000534                               CMRINT:                                 ;INTERRUPT ENTRY POINT
10 000534  042777  000100  001152               BIC     #IE,@CSR                ;CLEAR INTR EN ONLY
11 000542  022777  110004  001144               CMP     #INT!ALMDET!ENABKS,@CSR ;ALARM DETECTED FROM REMOTE?
12 000550  001021                              BNE     10$                     ;NO, MUST BE PWRUP
13 000552  112767  177777  001144               MOVB    #-1,ALMINT              ;YES, SET SFTWR FLAG
14 000560  017746  001132                      MOV     @SPAR,-(SP)             ;SAVE CURRENT SPAR ON STACK
15 000564  012777  001003  001124               MOV     #1003,@SPAR             ;LOAD ADDRESS OF REMOTE #
16 000572  017767  001122  001110               MOV     @SPDR,REMNM             ;GET REMOTE #
17 000600  012677  001112                      MOV     (SP)+,@SPAR             ;REPLACE CONTENTS OF SPAR
18 000604  052777  000120  001102               BIS     #CLRTPB!IE,@CSR         ;SET TO LEAVE
19 000612  000002                              RTI                             ;WE'LL DO OUR OWN THIS TIME!
20
21 000614                               10$:
22                                              ;---------------------------------------------------------------
   000614  000004  000000' 000622'              PIRQ$,BEGIN,ENTZ                ; QUEUE UP TO CONTINUE AT ENTZ AND RTI
                                                ;---------------------------------------------------------------
23
24 000622  022777  102004  001064  ENTZ:   CMP     #INT!PWRUP!ENABKS,@CSR  ;PWR-UP & INTER BITS SET?
25 000630  001414                              BEQ     ENTZ1                   ;IF YES, GO CHECK RES. SCPAD MEM
26 000632  004767  001026                      JSR     PC,ERSUB                ;GET ERROR INFO
27 000636  012767  000011  177242               MOV     #11,ERRTYP              ;ILLEGAL INTR OR DONE CLR
28                                              ;*****************************************************************
   000644  104405  000000' 001714'              HRDER$,BEGIN,TABL1      ;UNEXPECTED INTERRUPT
                                                ;*****************************************************************
29 000652  112767  177777  001046               MOVB    #-1,ERFLG               ;SET ERROR FLAG
30 000660  000420                              BR      ENTXX                   ;AND LEAVE
31
32 000662  012767  001000  001016  ENTZ1:  MOV     #RESAD,MPOINT           ;SET MEMM PNTR TO RES. SCPAD
33 000670  004767  000532                      JSR     PC,MEMTST               ;GO DO SCPAD MEM TEST
34 000674  104407  000000'                     BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
   000700  104407  000000'                     BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
35 000704  105767  001016                      TSTB    ERFLG                   ;CHECK ERROR FLAG
36 000710  001413                              BEQ     ENTZX                   ;NO ERRORS, LEAVE
37 000712  012746  000662'                     MOV     #ENTZ1,-(SP)            ;STACK RETRY ADDR
38 000716  000167  000062                      JMP     RETRY
39
40 000722  012777  000002  000764  ENTXX:  MOV     #2,@CSR                 ;INITIALIZE CMR
41 000730  012746  000302'                     MOV     #ENT2,-(SP)             ;STACK RETRY ADDRESS
42 000734  000167  000044                      JMP     RETRY                   ;AND RETRY UP TO LIMIT
43
44 000740  052777  000020  000746  ENTZX:  BIS     #CLRTPB,@CSR            ;CLEAR TOP BYTE
45 000746  105067  000755                      CLRB    TRY
46 000752  105067  000750                      CLRB    ERFLG
47 000756  104413  000000'                     ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
                                                                        ;MONITOR SHALL TEST END OF PASS
48 000762  000167  177314                      JMP     ENT2                    ;ANOTHER LOOP
49                                      ;---------------------------------------------------------------
```



```
 1                                      ;*****************************************************************
 2
 3                                      ;DROP POINT!!!    MODULE IS DROPPED AT THIS POINT.
 4
 5 000766  052777  000020  000720  FINI:   BIS     #CLRTPB,@CSR            ;CLR TOP BYTE OF CSR
 6 000774  005077  000714                      CLR     @CSR                    ;AND CLR THE REST
 7 001000  104410  000000'                     END$,BEGIN              ;
 8
 9                                      ;---------------------------------------------------------------
```

```
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++


                                        ;RETRY ON ERROR ROUTINE

                                        ;RETRY ROUTINE WITH ERROR UP TO RETRY LIMIT (10.)
                                        ;THEN, IF SR1 = 0, DROP THE MODULE

                                        ;ENTER WITH ADDRESS OF FAILING ROUTINE ON THE STACK

001004  105267  000717                  RETRY:  INCB    TRY                     ;UPDATE TRIES
001010  122767  000012  000711                  CMPB    #10.,TRY                        ;DONE 10.?
001016  001016                                  BNE     RTR1                    ;IF NOT, RETRY
001020  005767  176772                          TST     SR1                     ;IF YES, CHECK OPTION BIT 15
001024  100411                                  BMI     RTR0                    ;IF SET, CONTINUE
001026  104403  000000' 001730'                 MSGN$,BEGIN,EXCED       ;ASCII MESSAGE CALL WITH COMMON HEADER
001034  012777  000020  000652                  MOV     #CLRTPB,@CSR            ;CLR THE DEVICE
001042  005726                                  TST     (SP)+                   ;DUMP THE RETRY ADDRESS
001044  000167  177716                          JMP     FINI                    ;AND GO DROP THE MODULE

001050  105067  000653                  RTR0:   CLRB    TRY                     ;CLR RETRY COUNTER
001054  000207                          RTR1:   RTS     PC                      ;TRY AGAIN

                                        ;---------------------------------------------------------------


                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

                                        ;SETUP SUBROUTINE.
                                        ;SET UP THE DEVICE ADDRESSES
                                        ;SET UP THE DEVICE VECTOR & BR1
                                        ;CLEAR FLAGS AND POINTERS


001056  016700  176724                  SETUP:  MOV     ADDR,R0                 ;GET BASE ADDR TO R0
001062  010067  000626                          MOV     R0,CSR                  ;GEN. CSR ADDR
001066  005720                                  TST     (R0)+                   ;R0 = R0+2
001070  010067  000622                          MOV     R0,SPAR                 ;GEN. SPAR ADDR
001074  005720                                  TST     (R0)+
001076  010067  000616                          MOV     R0,SPDR                 ;AND GEN. SPDR ADDR

001102  016700  176702                          MOV     VECTOR,R0               ;GET DEVICE VECTOR
001106  010067  000570                          MOV     R0,CMRVEC               ;LOAD DEVICE VECTOR
001112  005720                                  TST     (R0)+
001114  116710  176672                          MOVB    BR1,(R0)                ;SAVE PRIORITY
001120  012777  000534' 000554                  MOV     #CMRINT,@CMRVEC         ;POINT VECTOR TO INTR ENTRY
001126  105067  000574                          CLRB    ERFLG                   ;CLR FLAGS,
001132  105067  000567                          CLRB    DROP
001136  105067  000565                          CLRB    TRY
001142  105067  000556                          CLRB    ALMINT
001146  012767  000000  000532                  MOV     #REQAD,MPOINT           ;INIT POINTER
001154  000207                                  RTS     PC                      ;RETURN


                                        ;--------------------------------------------------------------
```



```
                                        ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

                                        ;TEST DROP SUBROUTINE
                                        ;TEST THE DEVICE TO SEE IF IT SHOULD BE DROPPED.
                                        ;
                                        ;----------> THIS COULD TAKE UP TO 1 SECOND <------------


001156                                  TSTDRP:                                 ;ENTRY POINT
001156  105067  000543                          CLRB    DROP                    ;CLEAR DROP FLAG
001162  012767  000777  000514                  MOV     #777,CNTR0              ;SET UP TIMER
001170  022777  102000  000516  TDR0:           CMP     #PWRUP!INT,@CSR         ;PWR UP & INTR SET?
001176  001425                                  BEQ     TDR1                    ;YES, CONTINUE
001200  005367  000500                          DEC     CNTR0                   ;NO, COUNT TIMER
001204  001405                                  BEQ     TDRE                    ;ERROR IF TIMEOUT!!
001206  104407  000000'                         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001212  104407  000000'                         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001216  000764                                  BR      TDR0                    ;KEEP LOOKING
001220  112767  177777  000477  TDRE:           MOVB    #-1,DROP                ;NO GOOD; SET DROP FLAG
001226  004767  000432                          JSR     PC,ERSUB                ;GET ERROR INFO
001232  012767  000006  176646                  MOV     #6,ERRTYP               ;DEVICE NOT ON-LINE
                                                ;*************************************************************
001240  104405  000000' 001714'                 HRDER$,BEGIN,TABL1      ;INIT SEQ WON'T COMPLETE
                                                ;*************************************************************
001246  000167  000136                          JMP     TDRX                    ;NOW LEAVE

001252  052777  000020  000434  TDR1:           BIS     #CLRTPB,@CSR            ;CLR TOP BYTE OF CSR
001260  005077  000432                          CLR     @SPAR                   ;ADDR TOP OF REQ SCPAD
001264  012777  000213  000426                  MOV     #213,@SPDR              ;LOAD MAINT FUNC - SUB F #3
001272  005277  000420                          INC     @SPAR                   ;NEXT SP ADDR
001276  012777  000001  000414                  MOV     #1,@SPDR                ;DUMMY # OF WORDS
001304  005277  000406                          INC     @SPAR                   ;NEXT SP ADDR
001310  012777  000001  000402                  MOV     #1,@SPDR                ;DUMMY # OF REQUESTS
001316  052777  000200  000370                  BIS     #MAIN,@CSR              ;SET "MAIN"
001324  012767  000777  000352                  MOV     #777,CNTR0              ;SET UP A COUNTER
001332  022777  137600  000354  TDR2:           CMP     #137600,@CSR            ;CSR SHOULD GO TO 137600
001340  001423                                  BEQ     TDRX                    ;EXIT IF IT DOES
001342  005367  000336                          DEC     CNTR0                   ;IF NOT, WAIT A BIT
001346  001405                                  BEQ     TDRE1                   ;THEN DROP MODULE
001350  104407  000000'                         BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001354  104407  000000'                         BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001360  000764                                  BR      TDR2                    ;KEEP A WATCH

001362  112767  177777  000335  TDRE1:          MOVB    #-1,DROP                ;SET "DROP" FLAG
001370  004767  000270                          JSR     PC,ERSUB                ;GET ERROR INFO
001374  012767  000033  176504                  MOV     #33,ERRTYP
                                                ;*************************************************************
001402  104405  000000' 001714'                 HRDER$,BEGIN,TABL1      ;WRONG RESPONSE TO "DIE" FUNC
                                                ;*************************************************************

001410  012777  000020  000276  TDRX:           MOV     #CLRTPB,@CSR            ;CLR TOP BYTE OF CSR
001416  012777  000002  000270                  MOV     #2,@CSR                 ;INIT CMR11
001424  000207                                  RTS     PC                      ;AND RETURN


                                        ;---------------------------------------------------------------
```

```
                                ;*****************************************************************

                                ;MEMORY TEST SUBROUTINE.
                                ;WRITE INCREMENT PATTERN INTO 256 LOCATIONS
                                ;THEN READ BACK AND CHECK THEM
                                ;THIS IS DONE 128 TIMES (200 OCTAL).

                                ;ENTER WITH BASE ADDR OF SCRATCHPAD IN "MPOINT"
                                ;
                                ;       MOV     #READ,MPOINT
                                ;       JSR     PC,MEMTST
                                ;
                                ;BASE ADDR 0 IS START OF REQUEST SCRATCHPAD
                                ;BASE ADDR 1000 IS START OF RESULT SCRATCHPAD

001426                          MEMTST:                                 ;ENTRY POINT
001426  105067  000274                  CLRB    ERFLG                   ;CLEAR ERROR FLAG
001432  005067  000246                  CLR     CNTR0                   ;CLR ALL PURPOSE COUNTER
001436  016702  000244          MRSTRT: MOV     MPOINT,R2               ;GET SCPAD ADDR POINTER
001442  005067  000244                  CLR     #ORD                    ;CLR COUNT PATTERN

001446  010277  000[illegible]  MLOAD:  MOV     R2,@SPAR                ;LOAD ADDRESS
001452  016777  000234  000240          MOV     #ORD,@SPDR              ;DEPOSIT
001460  120227  000377                  CMPB    R2,#377                 ;DONE ALL LOC.?
001464  001404                          BEQ     MREAD                   ;YES, THEN CHECK THEM
001466  005202                          INC     R2                      ;NO, NEXT LOC.
001470  005267  000216                  INC     #ORD                    ;NEXT DATA
001474  000764                          BR      MLOAD                   ;CONTINUE LOADING

001476                          MREAD:
001476  104407  000000'                  BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
001502  104407  000000'                  BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
001506  016702  000174                  MOV     MPOINT,R2               ;NOW CHECK SCRATCHPAD
001512  005067  176370                  CLR     ASB                     ;SET UP SHD-BE DATA WORD
001516  010277  000174          MR1:    MOV     R2,@SPAR                ;LOAD ADDRESS
001522  017767  000172  176360          MOV     @SPDR,AWAS              ;EXAMINE
001530  026767  176352  176352          CMP     ASB,AWAS                ;IS DATA OK?
001536  001415                          BEQ     MR2                     ;YUP
001540  004767  000120                  JSR     PC,ERSUB                ;GET ERROR INFO
001544  010267  176334                  MOV     R2,WASADR               ;>>>>WASADR IS "SCRATCHPAD" ADDR ONLY<<<<
001550  012767  000106' 176324          MOV     #ASR,SBADR
                                                                        ;******SCRATCHPAD MEMORY DATA ERROR**********
                                        ;*****************************************************************
001556  104404  000000'                  DATER$,BEGIN            ;DATA ERROR!!!
                                        ;*****************************************************************
001562  112767  177777  000136          MOVB    #-1,ERFLG               ;SET ERROR FLAG
001570  000434                          BR      MRX                     ;AND LEAVE

001572  120227  000377          MR2:    CMPB    R2,#377                 ;DONE CHECKING?
001576  001404                          BEQ     MRPT                    ;IF YES, 200 TIMES?
001600  005202                          INC     R2                      ;NEXT ADDRESS
001602  005267  176300                  INC     ASB                     ;NEXT DATA
001606  000743                          BR      MR1                     ;KEEP CHECKING
001610  016702  000072          MRPT:   MOV     MPOINT,R2               ;CLEAR SCRATCHPAD AND TRY AGAIN
001614  005067  000072                  CLR     #ORD
001620  010277  000072          MRPT0:  MOV     R2,@SPAR                ;LOAD ADDRESS
```



```
001624  016777  000062  000066          MOV     #ORD,@SPDR              ;DEPOSIT 0'S
001632  120227  000377                  CMPB    R2,#377                 ;DONE ALL?
001636  001402                          BEQ     MRPT1                   ;YES, SEE IF WE TRY AGAIN
001640  005202                          INC     R2                      ;NO, NEXT ADDRESS
001642  000766                          BR      MRPT0                   ;AND CONTINUE CLEANING
001644  026727  000034  000200  MRPT1:  CMP     CNTR0,#200              ;DONE 200(8) LOOPS?
001652  001403                          BEQ     MRX                     ;YES, EXIT
001654  005267  000024                  INC     CNTR0                   ;KEEP A COUNT OF LOOPS
001660  000666                          BR      MRSTRT

001662  000207          MRX:    RTS     PC

                                ;---------------------------------------------------------------


                                ;*****************************************************************

                                ;ERROR INFO SUBROUTINE:
                                ;GET INFO FOR DEC/X11 MONITOR ERROR ROUTINES


001664  016767  000024  176206  ERSUB:  MOV     CSR,CSRA                ;LOAD ADDR OF STAT REG.
001672  017767  000016  176202          MOV     @CSR,ACSR               ;LOAD CONTENTS OF CSR
001700  000207                          RTS     PC                      ;AND RETURN


                                ;---------------------------------------------------------------
```

```
                                ;*****************************************************************

                                ;VARIABLES AND ASCII STORAGE:

                                ;VARIABLES:

001702  000000                  CMRVEC: .WORD   0               ;CMR VECTOR
001704  000000                  CNTR0:  .WORD   0               ;ALL PURPOSE COUNTER
001706  000000                  MPOINT: .WORD   0               ;POINTER FOR MEM. TEST ROUTINE
001710  000000                  REMNM:  .WORD   0               ;STORAGE FOR REMOTE #
001712  000000                  WORD:   .WORD   0               ;DATA STORAGE FOR MEM TEST

001714                          TABL1:
001714  000000                  CSR:    .WORD   0               ;COMMAND & STATUS REGISTER
001716  000000                  SPAR:   .WORD   0               ;SCRATCHPAD ADDRESS REGISTER
001720  000000                  SPDR:   .WORD   0               ;SCRATCHPAD DATA REGISTER
001722  177777                          .WORD   177777          ;TABLE TERMINATOR

                                ;FLAGS

001724      000                 ALMINT: .BYTE   0               ;INDICATOR FOR REMOTE ALARMS
001725      000                 DROP:   .BYTE   0               ;MODULE DROP FLAG
001726      000                 ERFLG:  .BYTE   0               ;ERROR INDICATOR FLAG
001727      000                 TRY:    .BYTE   0               ;ATTEMPT COUNT FOR ERRORS

                                .EVEN

                                ;ASCII MESSAGES:

001730  001742'                 EXCED:  MSG0
001732  177777                          .WORD   177777
001734  001777'                 REMALM: MSG1
001736  002032'                         MSG2
001740  177777                          .WORD   177777

001742      045     105     122 MSG0:   .ASCIZ  '%ERROR RETRY LIMIT EXCEEDED%'
001745      122     117     122
001750      040     122     105
001753      124     122     131
001756      040     114     111
001761      115     111     124
001764      040     105     130
001767      103     105     105
001772      104     105     104
001775      045     000
001777      045     055     055 MSG1:   .ASCIZ  '%--->CMR11 ALARM, REMOTE #'
002002      055     076     103
002005      115     122     061
002010      061     040     101
002013      114     101     122
002016      115     056     040
002021      122     105     115
002024      117     124     105
002027      040     043     000

                                .EVEN
002032                  MSG2:   .BLKB   6
002040      074     055     055         .ASCIZ  '<---%'
```



```
002043      055     045     000
                                .EVEN
000001                          .END
```

```
ACSR    000102R         CONFIG  000056R         INIT    000030R         PRTY1 = 000040          SPDR    001720R
ADDR    000006R         CSR     001714R         INT   = 100000          PRTY2 = 000100          SPUINT  000032R
ADDR22= 001000          CSRA    000100R         INTR    000120R         PRTY3 = 000140          SPSIZ = 000040
ALMDET= 010000          DATCK$= 104411          MAIN  = 000200          PRTY4 = 000200          SR1     000016R
ALMINT  001724R         DATER$= 104404          MAP22$= 104416          PRTY5 = 000240          SR2     000020R
ASB     000106R         DROP    001725R         MEMTST  001426R         PRTY6 = 000300          SR3     000022R
ASTAT   000104R         DVID1   000014R         MLOAD   001446R         PRTY7 = 000340          SR4     000024R
AWAS    000110R         ENABK$= 000004          MODNAM  000000R         PS    = 177776          START   000224R
BEGIN   000000R         ENDIT$= 104413          MODSP   000224R         PSW   = 177776          STAT    000026R
BIT0  = 000001          END$  = 104410          MPOINT  001706R         PUSH  = 005746          SVR0    000062R
BIT1  = 000002          ENTXX   000722R         MREAD   001476R         PUSH2 = 024646          SVR1    000064R
BIT10 = 002000          ENTZ    000622R         MRPT    001610R         PWRUP = 002000          SVR2    000066R
BIT11 = 004000          ENTZX   000740R         MRPT0   001620R         RAND$ = 104417          SVR3    000070R
BIT12 = 010000          ENTZ1   000662R         MRPT1   001644R         RANNUM  000054R         SVR4    000072R
BIT13 = 020000          ENT2    000302R         MRSTRT  001436R         REMALM  001734R         SVR5    000074R
BIT14 = 040000          ENT3    000350R         MRX     001662R         REMNM   001710R         SVR6    000076R
BIT15 = 100000          ENT4    000416R         MR1     001516R         REQAD = 000000          SYSCNT  000052R
BIT2  = 000004          ENT5    000456R         MR2     001572R         RESAD = 001000          TABL1   001714R
BIT3  = 000010          ERFLG   001726R         MSGN$ = 104403          RESTRT  000224R         TDRE    001220R
BIT4  = 000020          ERRTYP  000106R         MSGS$ = 104402          RES1    000056R         TDRE1   001362R
BIT5  = 000040          ERSUB   001664R         MSG$  = 104401          RES2    000060R         TDRX    001410R
BIT6  = 000100          EXCED   001730R         MSG0    001742R         RETRY   001004R         TDR0    001170R
BIT7  = 000200          EXIT$ = 104400          MSG1    001777R         RSTRT   000112R         TDR1    001252R
BIT8  = 000400          FINI    000766R         MSG2    002032R         RTR0    001050R         TDR2    001332R
BIT9  = 001000          GETPA$= 104415          NULL  = 000000          RTR1    001054R         TRPDFD= 000022
BREAK$= 104407          G%BUF$= 104414          OPEN  = 000000          R6    =%000006          TRY     001727R
BR1     000012R         HRDCNT  000044R         OTOA$ = 104420          R7    =%000007          TSTDRP  001156R
BR2     000013R         HRDER$= 104405          PASCNT  000034R         SBADR   000102R         VECTOR  000010R
BTOD$ = 104421          HRDPAS  000050R         PIRQ$ = 000004          SETUP   001056R         %ASADR  000104R
CDATA$= 104412          ICONT   000036R         POPSP = 005726          SOFCNT  000042R         %OFR    000116R
CLRTPB= 000020          ICOUNT  000040R         POPSP2= 022626          SOFER$= 104406          %DTO    000114R
CMRINT  000534R         IDNUM   000122R         PRTY  = 000000          SOFPAS  000046R         %ORD    001712R
CMRVEC  001702R         IE    = 000100          PRTY0 = 000000          SPAR    001716R         XFLAG   000005R
CNTR0   001704R

. ABS.  000000      000
        002046      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  7986 WORDS  ( 32 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  66 PAGES
DK1:CMAA,CMAA/C/M:TTM=DK1:DDXCOM.P11,CMRDX
```



```
ACSR      9-7#    15-76*
ADDR      9-7#    13-34
ADDR22    9-7#
ALMDET    9-13#   11-11
ALMINT   10-33    10-39*  11-13*  13-49*  16-21#
ASB       9-7#    15-33*  15-36   15-40   15-49*
ASTAT     9-7#
AWAS      9-7#    15-35*  15-36
BEGIN     9-7#    10-10   10-10   10-11   10-11   10-16   10-16   10-17   10-17   10-22   10-27   10-27   10-35   10-38
         10-44    10-44   10-45   10-45   10-46   10-46   11-22   11-28   11-34   11-34   11-47   12-7    13-16   14-16
         14-16    14-21   14-37   14-37   14-43   15-31   15-31   15-42
BIT0      9-7#
BIT1      9-7#
BIT10     9-7#
BIT11     9-7#
BIT12     9-7#
BIT13     9-7#
BIT14     9-7#
BIT15     9-7#
BIT2      9-7#
BIT3      9-7#
BIT4      9-7#
BIT5      9-7#
BIT6      9-7#
BIT7      9-7#
BIT8      9-7#
BIT9      9-7#
BR1       9-7#    13-44
BR2       9-7#
BREAK$    9-7#    10-10   10-10   10-11   10-11   10-16   10-16   10-17   10-17   10-27   10-27   10-44   10-44   10-45
         10-45    10-46   10-46   11-34   11-34   14-16   14-16   14-37   14-37   15-31   15-31
BTOD$     9-7#
CDATA$    9-7#
CLRTPB    9-17#   10-41   11-18   11-44   12-5    13-17   14-24   14-45
CMRINT   11-9#    13-45
CMRVEC   13-42*   13-45*  16-7#
CNTR0    14-11*   14-14*  14-32*  14-35*  15-19*  15-59   15-61*  16-8#
CONFIG    9-7#
CSR      10-9*    10-19   10-21*  10-24*  10-41*  10-42*  10-43*  11-10*  11-11   11-18*  11-24   11-40*  11-44*  12-5*
         12-6*    13-17*  13-35*  14-12   14-24*  14-31*  14-33   14-45*  14-46*  15-75   15-76   16-14#
CSRA      9-7#    15-75*
DATCK$    9-7#
DATER$    9-7#    15-42
DROP     10-13    13-47*  14-10*  14-18*  14-40*  16-22#
DVID1     9-7#
ENABK$    9-18#   10-21   10-24   10-43   11-11   11-24
END$      9-7#    12-7
ENDIT$    9-7#    11-47
ENT2     10-7     10-14   10-16#  10-47   11-41   11-48
ENT3     10-20    10-24#  10-30
ENT4     10-29    10-33#
ENT5     10-34    10-41#
ENTXX    11-30    11-40#
ENTZ     11-22    11-24#
ENTZ1    11-25    11-32#  11-37
ENTZX    11-36    11-44#
```

```
ERFLG   10-18*   10-28    11-29*   11-35    11-46*   13-46*   15-18*   15-43*   16-23#
ERRTYP   9-7#    11-27*   14-20*   14-42*
ERSUB   11-26    14-19    14-41    15-38    15-75#
EXCED   13-16    16-30#
EXIT$    9-7#    10-22
FINI    10-15    12-5#    13-19
GETPA$   9-7#
GWBUF$   9-7#
HRDCNT   9-7#
HRDEN$   9-7#    11-28    14-21    14-43
HRDPAS   9-7#
ICONT    9-7#
ICOUNT   9-7#    10-6
IONUM    9-7#
IE       9-16#   10-21    10-24    11-10    11-18
INIT     9-7#
INT      9-14#   11-11    11-24    14-12
INTR     9-7#
MAIN     9-15#   14-31
MAP22$   9-7#
MEMTST  10-26    11-33    15-17#
MLOAD   15-23#   15-29
MODNAM   9-7#
MODSP    9-7      9-7#
MPOINT  10-25*   11-32*   13-50*   15-20    15-32    15-51    16-9#
MR1     15-34#   15-50
MR2     15-37    15-46#
MREAD   15-26    15-31#
MRPT    15-47    15-51#
MRPT0   15-53#   15-58
MRPT1   15-56    15-59#
MRSTRT  15-20#   15-62
MRX     15-44    15-60    15-64#
MSG$     9-7#
MSG0    16-30    16-36#
MSG1    16-32    16-37#
MSG2    10-35    16-33    16-39#
MSGN$    9-7#    10-38    13-16
MSGS$    9-7#
NULL     9-7#
OPEN     9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7
         9-7      9-7      9-7      9-7      9-7      9-7      9-7#
OTOA$    9-7#    10-35
PASCNT   9-7#
PIRQ$    9-7#    11-22
POPSP    9-7#
POPSP2   9-7#
PRTY     9-7#
PRTY0    9-7      9-7#
PRTY1    9-7#
PRTY2    9-7#
PRTY3    9-7#
PRTY4    9-7      9-7#
PRTY5    9-7#
PRTY6    9-7#
PRTY7    9-7#
```



```
PS       9-7#
PSW      9-7#
PUSH     9-7#
PUSH2    9-7#
PWRUP    9-12#   10-19    11-24    14-12
R6       9-7#
R7       9-7#
RAND$    9-7#
RANNUM   9-7#
REMALM  10-38    16-32#
REMNM   10-35    11-16*   16-10#
REQAD    9-10#   10-25    13-50
RES1     9-7#
RES2     9-7#
RESAD    9-11#   11-32
RESTRT   9-7     10-6#
RETRY   10-31    11-38    11-42    13-11#
RSTRT    9-7#
RTR0    13-15    13-21#
RTR1    13-13    13-22#
SBADR    9-7#    15-40*
SETUP   10-8     13-34#
SOFCNT   9-7#
SOFER$   9-7#
SOFPAS   9-7#
SPAR    11-14    11-15*   11-17*   13-37*   14-25*   14-27*   14-29*   15-23*   15-34*   15-53*   16-15#
SPDR    11-16    13-39*   14-26*   14-28*   14-30*   15-24*   15-35    15-54*   16-16*
SPOINT   9-7#
SPSIZ    1-7#     9-7
SR1      9-7#    13-14
SR2      9-7#    10-36
SR3      9-7#
SR4      9-7#
START    9-7     10-5#
STAT     9-7#
SVR0     9-7#
SVR1     9-7#
SVR2     9-7#
SVR3     9-7#
SVR4     9-7#
SVR5     9-7#
SVR6     9-7#
SYSCNT   9-7#
TABL1   11-28    14-21    14-43    16-13#
TDR0    14-12#   14-17
TDR1    14-13    14-24#
TDR2    14-33#   14-38
TDRE    14-15    14-18#
TDRE1   14-36    14-40#
TDRX    14-22    14-34    14-45#
TRPDFD   9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7
         9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7
         9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7      9-7#     9-7#     9-7#     9-7#     9-7#     9-7#
         9-7#     9-7#     9-7#     9-7#     9-7#     9-7#     9-7#     9-7#     9-7#     9-7#     9-7#     9-7#     9-7#
TRY     11-45*   13-11*   13-12    13-21*   13-48*   16-24#
TSTDRP  10-12    14-9#
```

```
VECTOR    9-7#     13-41
WASADR    9-7#     15-39*
WDFR      9-7#
WDTO      9-7#
WORD     15-21*    15-24    15-28*    15-52*    15-54    16-11#
XFLAG     9-7#
```



```
BKMOD     1-95#
BREAK     1-185#   10-10     10-11     10-16     10-17     10-27     10-44     10-45     10-46     11-34     14-16     14-37     15-31
BTOD      1-204#
CKDATA    1-240#
DATACK    1-249#
DATERR    1-138#   15-42
DFSEVN    1-272#    9-7       9-7       9-7       9-7       9-7       9-7       9-7       9-7       9-7       9-7       9-7       9-7
DSEVNT    1-282#    9-7       9-7       9-7       9-7       9-7       9-7
END       1-175#   12-7
ENDIT     1-166#   11-47
ENDMOD    1-171#
EQUATS    1-288#    9-7
EXIT      1-120#   10-22
GETPA     1-231#
GWBUFF    1-219#
HRDER     1-128#   11-28     14-21     14-43
IOMOD     1-91#     9-7
IOMODP    1-115#
IOMODR    1-111#
IOMODX    1-107#
MAP22     1-235#
MODULE    1-0#      9-7
MSG       1-154#
MSGN      1-158#   10-38     13-16
MSGS      1-162#
NBKMOD    1-103#
OTOA      1-190#   10-35
PIRQ      1-179#   11-22
RAND      1-124#
SBKMOD    1-99#
SOFER     1-144#
```

```
.REM  =
```

IDENTIFICATION

PRODUCT CODE:	AC-S260B-MC
PRODUCT NAME:	CXCMAB0 CSS CMS11K DEC/X MOD
DATE:	19 JUL 1980
UPDATE:	10 FEB 1981
MAINTAINER:	CSS PRODUCT GROUP DIAGNOSTICS
AUTHOR(S):	ROBERT J. COLLINS
UPDATED BY:	JAMES M. DUPRE
	VIJAY ANANDWALA

1. ABSTRACT

CMA IS AN IOMOD THAT EXERCISES THE CMS11-K CARD READER. IT READS A PREPRINTED MARKSENSE DECK, FORMING A CHECKSUM FOR EACH CARD READ. THE CALCULATED CHECKSUM IS COMPARED AGAINST A KNOWN CHECKSUM AND ANY ERRORS ARE REPORTED ON THE TTY. THE MODULE TESTS BOTH THE IMAGE AND ENCODED DATA.

2. REQUIREMENTS

HARDWARE: CMS11-K READER AND INTERFACE
MARKSENSE CARD DECK - MAINDEC-11-D2CMF-C01

3. PASS DEFINITION

ONE PASS OF READING 80 80-COLUMN CARDS (6400 CHARACTERS), AT WHICH TIME THE INPUT HOPPER SHOULD BE EMPTY.

4. EXECUTION TIME

ONE PASS OF BBN RUNNING ALONE ON A PDP-11/40 TAKES APPROXIMATELY 20 SECONDS.

5. CONFIGURATION REQUIREMENTS

DEFAULT PARAMETERS:

DEVADR: 177160, VECTOR: 230, BR1: 6, DEVCNT: 1

REQUIRED PARAMETERS:

NONE

6. DEVICE/OPTION SETUP

A. POWER UP THE READER
B. LOAD THE CARD DECK
C. READY THE READER

7. MODULE OPERATION

TEST SEQUENCE:

A. SETUP DEVICE REGISTER ADDRESSES AND MODULE VARIABLES
B. IF ON-LINE, GO TO C; ELSE WAIT FOR READY
C. READ A CARD - ENABLE INTERRUPT
D. INTERRUPT SERVICE:
    1. CHECK FOR NON-DATA ERRORS (TIMING, ETC.)
    2. COUNT COLUMN
    3. FORM CHECKSUMS (DIRECT AND ENCODED)
    4. IF 80 COLUMNS READ: CHECK DATA - REPORT ANY ERRORS
E. IF HOPPER NOT EMPTY COUNT A CARD AND GO TO B.
F. AT HOPPER EMPTY (OFF-LINE) AND 80 CARDS READ, REPORT END OF PASS AND START AT A. ELSE REPORT ERROR AND GO TO A.

8. OPERATION OPTIONS

NONE

9. NON-STANDARD PRINTOUTS

NONE


```
                                .LIST   SEQ,BIN,LOC

000000'                                 IOMOD   <CMAB >,177160,230,6,0,0,1
(1) 000000'                             MODULE  140000,CMAB ,177160,230,6,0,0,1,
(2)                                     .TITLE  CMAB DEC/X11 SYSTEM EXERCISER MODULE
(2)                             ;       DDXCOM  VERSION  6      23-MAY-78
(2)                                     .LIST   BIN
(2)                             ;*****************************************************************************
(2) 000000'                     BEGIN:
(2) 000000' 046503  041101  040 MODNAM: .ASCII  /CMAB / ;MODULE NAME.
(2) 000005'     000             XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF #BUFF USAGE
(2) 000006' 177160              ADDR:   177160+0                ;1ST DEVICE ADDR.
(2) 000010' 000230              VECTOR: 230+0                   ;1ST DEVICE VECTOR.
(2) 000012'     300             BR1:    .BYTE   PRTY6+0         ;1ST BR LEVEL.
(2) 000013'     000             BR2:    .BYTE   PRTY0+0         ;2ND BR LEVEL.
(2) 000014' 000001              DVID1:  0+1                     ;DEVICE INDICATOR 1.
(2) 000016' 000000              SR1:    OPEN                    ;SWITCH REGISTER 1
(2) 000020' 000000              SR2:    OPEN                    ;SWITCH REGISTER 2
(2) 000022' 000000              SR3:    OPEN                    ;SWITCH REGISTER 3
(2) 000024' 000000              SR4:    OPEN                    ;SWITCH REGISTER 4
(2)                             ;*****************************************************************************
(2) 000026' 140000              STAT:   140000                  ;STATUS #0R0.
(2) 000030' 000222'             INIT:   START                   ;MODULE START ADDR.
(2) 000032' 000222'             SPOINT: MODSP                   ;MODULE STACK POINTER.
(2) 000034' 000000              PASCNT: 0                       ;PASS COUNTER.
(2) 000036' 000001              ICONT:  1                       ;# OF ITERATIONS PER PASS=1
(2) 000040' 000000              ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
(2) 000042' 000000              SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
(2) 000044' 000000              HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
(2) 000046' 000000              SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
(2) 000050' 000000              HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
(2) 000052' 000000              SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
(2) 000054' 000000              RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
(2) 000056'                     CONFIG:
(2) 000056' 000000              RES1:   0                       ;RESERVED FOR MONITOR USE
(2) 000060' 000000              RES2:   0                       ;RESERVED FOR MONITOR USE
(2) 000062' 000000              SVR0:   OPEN                    ;LOC TO SAVE R0.
(2) 000064' 000000              SVR1:   OPEN                    ;LOC TO SAVE R1.
(2) 000066' 000000              SVR2:   OPEN                    ;LOC TO SAVE R2.
(2) 000070' 000000              SVR3:   OPEN                    ;LOC TO SAVE R3.
(2) 000072' 000000              SVR4:   OPEN                    ;LOC TO SAVE R4.
(2) 000074' 000000              SVR5:   OPEN                    ;LOC TO SAVE R5.
(2) 000076' 000000              SVR6:   OPEN                    ;LOC TO SAVE R6.
(2) 000100' 000000              CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
(2) 000102'                     SBADR:                          ;ADDR OF GOOD DATA, OR
(2) 000102' 000000              ACSR:   OPEN                    ;CONTENTS OF CSR.
(2) 000104'                     #ASADR:                         ;ADDR OF BAD DATA, OR
(2) 000104' 000000              ASTAT:  OPEN                    ;STATUS REG CONTENTS.
(2) 000106'                     ERRTYP:                         ;TYPE OF ERROR
(2) 000106' 000000              ASB:    OPEN                    ;EXPECTED DATA.
(2) 000110' 000000              AWAS:   OPEN                    ;ACTUAL DATA.
(2) 000112' 000222'             RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
(2) 000114' 000000              #DTO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
(2) 000116' 000000              #DFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
(2) 000120' 000000              INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
```

```
(2)  000122'                        IDNUM:                          ;MODULE IDENTIFICATION NUMBER=
(2)            000040                       .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
(2)                                         .NLIST
(2)                                         .WORD   0
(2)                                         .LIST
(3)                                         .ENDR
(2)  000222'                        MODSP:
(2)                                 ;*****************************************************************************
```



```
 495  000222'                        START:
 496  000222' 016700  177560         RESTRT: MOV     ADDR,R0         ;SETUP DEVICE
 497  000226' 010067  000766                 MOV     R0,CRS
 498  000232' 005720                 TST     (0)+
 499  000234' 010067  000762                 MOV     R0,CRB1
 500  000240' 005720                 TST     (0)+
 501  000242' 010067  000756                 MOV     R0,CRB2
 502  000246' 016700  177536                 MOV     VECTOR,R0
 503  000252' 012720  000426'                MOV     #INTER,(0)+
 504  000256' 116710  177530                 MOVB    BR1,(0)
 505                                 ;       ----------------------
 506  000262' 005077  000732                 CLR     @CRS    ;CLR ANY JUNK
 507  000266' 005067  000722                 CLR     CRDCNT          ;ZERO CARD COUNTER
 508  000272' 032777  000400  000720         BIT     #BIT8,@CRS      ;READER ONLINE?
 509  000300' 001410                 BEQ     NUCARD          ;YES, BEGIN THE TEST
 510  000302' 005767  177526                 TST     PASCNT          ;FIRST PASS?
 511  000306' 001005                 BNE     NUCARD          ;NO, DO NOT LOG ERROR YET
 512  000310' 004767  000562                 JSR     PC,ERSUB        ;YES, LOAD ERROR INFORMATION
 513                                 ;*****************************************************************
(1)  000314' 104405  000000' 000000         HRDERR$,BEGIN,NULL      ;READER OFF-LINE ... WAITING
(1)                                 ;*****************************************************************
 514  000322' 005067  000670         NUCARD: CLR     COLUMN          ;0 COL. COUNTER
 515  000326' 005067  000656                 CLR     SUM1            ;0 IMAGE SUM
 516  000332' 005067  000654                 CLR     SUM2            ;0 ENCODED SUM
 517  000336' 012767  077777  000662         MOV     #77777,CLK      ;SET CLOCK COUNTER
 518  000344' 032777  000400  000646 1$:      BIT     #BIT8,@CRS      ;READY?
 519  000352' 001416                 BEQ     GO              ;YES, CONTINUE
 520  000354' 104407  000000'                BREAK$,BEGIN            ;TEMPORARY RETURN TO MONITOR....
(1)  000360' 104407  000000'                BREAK$,BEGIN            ;THEN CONTINUE AT NEXT INSTRUCTION.
 521  000364' 005367  000636                 DEC     CLK             ;NO, WAIT SOME MORE
 522  000370' 001365                 BNE     1$              ;LOOP
 523  000372' 004767  000500                 JSR     PC,ERSUB        ;LOAD ERROR INFO
 524                                 ;*****************************************************************
(1)  000376' 104405  000000' 000000         HRDERR$,BEGIN,NULL      ;READER STILL NOT READY ... DROP
(1)                                 ;*****************************************************************
 525  000404' 104410  000000'                END$,BEGIN              ;
 526
 527  000410' 005267  000600         GO:     INC     CRDCNT          ;COUNT A CARD
 528  000414' 012777  000101  000576         MOV     #101,@CRS       ;ENABLE PI AND READ
 529  000422' 104400  000000'                EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
 530                                 ;       ------------------------
 531
 532                                 ;INTERRUPT SERVICE
 533
 534  000426' 105777  000566         INTER:  TSTH    @CRS            ;COLUMN READY TO BE READ?
 535  000432' 100011                 BPL     1$              ;NO, FIND OUT WHY
 536  000434' 067767  000562  000546         ADD     @CRB1,SUM1      ;ADD IMAGE TO SUM
 537  000442' 067767  000556  000542         ADD     @CRB2,SUM2      ;ADD ENCODED TO SUM
 538  000450' 005267  000542                 INC     COLUMN          ;COUNT A COLUMN
 539  000454' 000002                 RTI                             ;EXIT
```

```
   541  000456' 004767  000414          1$:     JSR     PC,ERSUB        ;SAVE ERROR INFO
   542  000462' 042777  000100  000530          BIC     #BIT6,@CRS      ;DISABLE INTERUPT
   543                                          ;---------------------------------------------------------------
   (1)  000470' 000004  000000' 000476'         PIRQ$,BEGIN,WHY         ; QUEUE UP TO CONTINUE AT WHY AND RTI
   (1)                                          ;---------------------------------------------------------------
   544  000476' 005767  177400          WHY:    TST     ACSR            ;ERROR?
   545  000502' 100444                       BMI     2$              ;YES
   546  000504' 032767  040000  177370          BIT     #BIT14,ACSR     ;CARD DONE?
   547  000512' 001027                       BNE     1$              ;YES, CHECK SUMS
   548  000514' 032767  002000  177360          BIT     #BIT10,ACSR     ;TRANS. TO ONLINE?
   549  000522' 001405                       BEQ     5$              ;NO CONTINUE TESTING
   550                                          ;*****************************************************************
   (1)  000524' 104405  000000' 000000         HRDER$,BEGIN,NULL       ;ON-LINE TRANSITION DETECTED!
   (1)                                          ;*****************************************************************
   551  000532' 000167  000356                  JMP     EJECT
   552  000536' 032767  000400  177336  5$:     BIT     #BIT8,ACSR      ;READER READY?
   553  000544' 001405                       BEQ     11$             ;YES GO REPORT UNKNOWN INTERUPT
   554                                          ;*****************************************************************
   (1)  000546' 104405  000000' 000000         HRDER$,BEGIN,NULL       ;READER OFF-LINE !
   (1)                                          ;*****************************************************************
   555  000554' 000167  177542                  JMP     NUCARD
   556  000560'                          11$:
   (1)                                          ;*****************************************************************
   (1)  000560' 104405  000000' 000000         HRDER$,BEGIN,NULL       ;UNKNOWN INTERUPT PROCESSED!
   (1)                                          ;*****************************************************************
   557  000566' 000167  177530                  JMP     NUCARD
   558
   559  000572' 022767  000120  000416  1$:     CMP     #80.,COLUMN     ;ALL COLUMNS CHECKED?
   560  000600' 001451                       BEQ     CHECK           ;YES
   561                                          ;*****************************************************************
   (1)  000602' 104405  000000' 000000         HRDER$,BEGIN,NULL       ;CARD DONE SET BEFOR ALL COL. READ
   (1)                                          ;*****************************************************************
   562  000610' 000167  177506                  JMP     NUCARD          ;TRY ANOTHER CARD
   563  000614' 032767  004000  177260  2$:     BIT     #BIT11,ACSR     ;TIMING ERROR?
   564  000622' 001405                       BEQ     3$              ;NO
   565                                          ;*****************************************************************
   (1)  000624' 104405  000000' 000000         HRDER$,BEGIN,NULL       ;TIMING ERROR
   (1)                                          ;*****************************************************************
   566  000632' 000167  177464                  JMP     NUCARD          ;TRY ANOTHER
   567  000636' 032767  010000  177236  3$:     BIT     #BIT12,ACSR     ;FEED ERROR?
   568  000644' 001405                       BEQ     31$             ;NO
   569                                          ;*****************************************************************
   (1)  000646' 104405  000000' 000000         HRDER$,BEGIN,NULL       ;FEED ERROR
   (1)                                          ;*****************************************************************
   570  000654' 000167  177442                  JMP     NUCARD          ;TRY AGAIN
   571  000660' 032767  020000  177214  31$:    BIT     #BIT13,ACSR     ;HOPPER EMPTY?
   572  000666' 001411                       BEQ     4$              ;0R=NO
   573  000670' 022767  000121  000316          CMP     #81.,CRDCNT     ;DONE 80?
   574  000676' 001473                       BEQ     FINI            ;0R=YES
   575                                          ;*****************************************************************
   (1)  000700' 104405  000000' 000000         HRDER$,BEGIN,NULL       ;CARD COUNT NOT 80 OR STACKER FULL
   (1)                                          ;*****************************************************************
   576  000706' 000167  000154                  JMP     FINI
   577  000712'                          4$:
   (1)                                          ;*****************************************************************
   (1)  000712' 104405  000000' 000000         HRDER$,BEGIN,NULL       ;ERR BIT SET BY ITSELF!
```



```
   (1)                                          ;*****************************************************************
   578  000720' 000167  177376                  JMP     NUCARD
```

```
000724' 026767  000260  000252  CHECK:  CMP     SUM1,CKSUM1     ;IS IMAGE SUM CORRECT?
000732' 001416                          BEQ     1$              ;YES, CHECK ENCODED
000734' 012767  001204' 177140          MOV     #CKSUM1,$BADR
000742' 016767  000236  177136          MOV     CKSUM1,ASB
000750' 012767  001210' 177126          MOV     #SUM1,#ASADR
000756' 016767  000226  177124          MOV     SUM1,AWAS
                                        ;*****************************************************************
(1) 000764' 104404  000000'             DATERR$,BEGIN           ;DATA ERROR!!!
(1)                                     ;*****************************************************************

000770' 026767  000216  000210  1$:     CMP     SUM2,CKSUM2     ;IS ENCODED CORRECT?
000776' 001416                          BEQ     2$              ;YES
001000' 012767  001206' 177074          MOV     #CKSUM2,$BADR
001006' 016767  000174  177072          MOV     CKSUM2,ASB
001014' 012767  001212' 177062          MOV     #SUM2,#ASADR
001022' 016767  000164  177060          MOV     SUM2,AWAS
                                        ;*****************************************************************
(1) 001030' 104404  000000'             DATERR$,BEGIN           ;DATA ERROR!!!
(1)                                     ;*****************************************************************

001034' 032777  000400  000156  2$:     BIT     #BIT8,@CRS      ;ON-LINE?
001042' 001407                          BEQ     3$              ;BR=YES
001044' 022767  000121  000142          CMP     #81.,CRDCNT     ;DONE 80 CARDS?
001052' 001405                          BEQ     FINI            ;BR=YES
                                        ;*****************************************************************
(1) 001054' 104405  000000' 000000      HRDERR$,BEGIN,NULL      ;READER OFF-LINE CHECK READER
(1)                                     ;*****************************************************************
001062' 000167  177234          3$:     JMP     NUCARD          ;GET NEXT
001066'                         FINI:
(1) 001066' 104413  000000'             ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
(1)                                                             ;MONITOR SHALL TEST END OF PASS
001072' 000167  177124                  JMP     RESTRT

001076' 016767  000116  176774  ERSUB:  MOV     CRS,CSRA        ;LOAD ERROR INFO
001104' 017767  000110  176770          MOV     @CRS,ACSR
001112' 000207                          RTS     PC              ;EXIT

001114' 012777  001140' 176666  EJECT:  MOV     #2$,@VECTOR     ;LOAD NEW RETURN
001122' 005077  000072                  CLR     @CRS            ;CLEAR IT
001126' 012777  000103  000064  1$:     MOV     #103,@CRS       ;SET I.E. READ+EJECT
001134' 104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
001140' 017767  000054  176734  2$:     MOV     @CRS,ACSR       ;SAVE CRS
001146' 042777  000100  000044          BIC     #100,@CRS       ;DISABLE INTERRUPTS
                                        ;---------------------------------------------------------------
(1) 001154' 000004  000000' 001162'     PIRQ$,BEGIN,3$          ; QUEUE UP TO CONTINUE AT 3$ AND RTI
(1)                                     ;---------------------------------------------------------------
001162' 032767  020000  176712  3$:     BIT     #BIT13,ACSR     ;HOPPER EMPTY?
001170' 001756                          BEQ     1$              ;BR=NO
001172' 012777  000426' 176610          MOV     #INTER,@VECTOR  ;RESTORE OLD RETURN
001200' 000167  177662                  JMP     FIN1

001204' 107767                  CKSUM1: 107767
001206' 004573                  CKSUM2: 4573
001210' 000000                  SUM1:   0
001212' 000000                  SUM2:   0
```



```
001214' 000000                  CRDCNT: 0
001216' 000000                  COLUMN: 0
001220' 000000                  CRS:    0
001222' 000000                  CRB1:   0
001224' 000000                  CRB2:   0
001226' 000000                  CLK:    0

        000001                          .END                    ;THAT'S ALL FOLKS!
```

```
ACSR    000107R          493#   544    546    548    552    563    567    571    606*   614*   617
ADDR    000006R          493#   496
ADDR22= 001000           493#
ASR     000106R          493#   583*   591*
ASTAT   000104R          493#
AWAS    000110R          493#   585*   593*
BEGIN   000000R          493#   513    520    524    525    529    543    550    554    556    561    565    569
                         575    577    586    594    600    602    613    616
BIT0  = 000001           493#
BIT1  = 000002           493#
BIT10 = 002000           493#   548
BIT11 = 004000           493#   563
BIT12 = 010000           493#   567
BIT13 = 020000           493#   571    617
BIT14 = 040000           493#   546
BIT15 = 100000           493#
BIT2  = 000004           493#
BIT3  = 000010           493#
BIT4  = 000020           493#
BIT5  = 000040           493#
BIT6  = 000100           493#   542
BIT7  = 000200           493#
BIT8  = 000400           493#   508    518    552    596
BIT9  = 001000           493#
BREAKS= 104407           493#   520
BR1     000012R          493#   504
BR2     000013R          493#
BTOD$ = 104421           493#
CDATA$= 104412           493#
CHECK   000724R          560    580#
CKSUM1  001204R          580    582    583    622#
CKSUM2  001206R          588    590    591    623#
CLK     001226R          517*   521*   631#
COLUMN  001216R          514*   538*   559    627#
CONFIG  000056R          493#
CRB1    001222R          499*   536    629#
CRB2    001224R          501*   537    630#
CRDCNT  001214R          507*   527*   573    598    626#
CRS     001220R          497*   506*   508    518    528*   534    542*   596    605    606    611*   612*   614
                         615*   628#
CSRA    000100R          493#   605*
DATCK$= 104411           493#
DATER$= 104404           493#   586    594
DVID1   000014R          493#
EJECT   001114R          551    610#
ENDIT$= 104413           493#   602
END$  = 104410           493#   525
ERRTYP  000106R          493#
ERSUB   001076R          512    523    541    605#
EXIT$ = 104400           493#   529    613
FINI    001066R          574    576    599    602#   620
GETPA$= 104415           493#
GO      000410R          519    527#
GWBUF$= 104414           493#
HRDCNT  000044R          493#
HRDER$= 104405           493#   513    524    550    554    556    561    565    569    575    577    600
```



```
HRDPAS  000050R          493#
ICONT   000036R          493#
ICOUNT  000040R          493#
IDNUM   000122R          493#
INIT    000030R          493#
INTER   000426R          503    534#   619
INTR    000120R          493#
MAP22$= 104416           493#
MODNAM  000000R          493#
MODSP   000222R          493#
MSGN$ = 104403           493#
MSGS$ = 104402           493#
MSG$  = 104401           493#
NUCARD  000322R          509    511    514#   555    557    562    566    570    578    601
NULL  = 000000           493#   513    524    550    554    556    561    565    569    575    577    600
OPEN  = 000000           493#
OTOA$ = 104420           493#
PASCNT  000034R          493#   510
PIRQ$ = 000004           493#   543    616
POPSP = 005726           493#
POPSP2= 022626           493#
PRTY  = 000000           493#
PRTY0 = 000000           493#
PRTY1 = 000040           493#
PRTY2 = 000100           493#
PRTY3 = 000140           493#
PRTY4 = 000200           493#
PRTY5 = 000240           493#
PRTY6 = 000300           493#
PRTY7 = 000340           493#
PS    = 177776           493#
PSW   = 177776           493#
PUSH  = 005746           493#
PUSH2 = 024646           493#
RAND$ = 104417           493#
RANNUM  000054R          493#
RESTRT  000222R          493    496#   603
RES1    000056R          493#
RES2    000060R          493#
RSTRT   000112R          493#
SBADR   000102R          493#   582*   590*
SOFCNT  000042R          493#
SOFER$= 104406           493#
SOFPAS  000046R          493#
SPOINT  000032R          493#
SPSIZ = 000040             7#   493
SR1     000016R          493#
SR2     000020R          493#
SR3     000022R          493#
SR4     000024R          493#
START   000222R          493    495#
STAT    000026R          493#
SUM1    001210R          515*   536*   580    584    585    624#
SUM2    001212R          516*   537*   588    592    593    625#
SVR0    000062R          493#
SVR1    000064R          493#
```

```
SVR2     000066R          493#
SVR3     000070R          493#
SVR4     000072R          493#
SVR5     000074R          493#
SVR6     000076R          493#
SYSCNT   000052R          493#
TRPDFD=  000022           493#
VECTOR   000010R          493#    502     610#    619#
WASADR   000104R          493#    584#    592#
WDFR     000116R          493#
WDTO     000114R          493#
WHY      000476R          543     544#
XFLAG    000005R          493#
```



```
BKMOD      95#
BREAK     185#     520
BTOD      204#
CKDATA    240#
DATACK    249#
DATERR    138#     586     594
DFSEVN    277#     493
DSEVNT    282#     493
END       175#     525
ENDIT     166#     602
ENDMOD    171#
EQUATS    288#     493
EXIT      120#     529     613
GETPA     231#
GWBUFF    219#
HRDER     128#     513     524     550     554     556     561     565     569     575     577     600
IOMOD      91#     493
IOMODP    115#
IOMODR    111#
IOMODX    107#
MAP22     235#
MODULE      8#     493
MSG       154#
MSGN      158#
MSGS      162#
NBKMOD    103#
OTOA      190#
PIRQ      179#     543     616
RAND      124#
SBKMOD     99#
SOFER     144#


. ABS.  000000      000
        001230      001

ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

CXCMAB,CXCMAB.SEQ/CRF=DDXCOM.C6,CXCMAB.SRC
RUN-TIME:  6  8  .9 SECONDS
RUN-TIME RATIO: 97/15=6.2
CORE USED:  7K  (13 PAGES)
```

```
                                .REM_
PRODUCT CODE:  AC-S257A-MC

PRODUCT NAME: CXDRKA0 CSS DR70 DEC/X MOD

DATE:         15 -JUL -1980

AUTHOR(S):    JAMES M. DUPRE

MAINTAINER:   C.S.S. PRODUCT GROUP
```



```
1 .0          ABSTRACT:

                DRK IS AN IOMODX WHICH EXERCISES THE DR70 INTERFACE
              BOTH WRITE AND READ FUNCTIONS ARE EXERCISED.
                THIS MODULE IS PDP11/70 COMPATABLE.

              RUNTIME/PASS:    1MINUTE/PASS


             SWITCH REGISTER VALUES:  NONE
```

```
                        2 .0    REQUIREMENTS:

                                A.SOFTWARE

                                        LOCATION DVID1 MUST BE SET TO THE UNIT UNDER TEST
                                        EXAMPLE: BIT0=UNIT 0,BIT2=UNIT1,ETC.

                                B.HARDWARE

                                        THE OUTPUT AND INPUT OF THIS INTERFACE MUST BE
                                        WRAPPED AROUND I.E. NOT CONNECTED TO AN EXTERNAL
                                        DEVICE.

                                        THIS MODULE RUN THE DR70 IN READ AND WRITE
                                        OPERATIONAL MODE.

                                C. CONFIGURATION REQUIREMENTS

                                        DEFAULT PARAMETERS:

                                        DEVADR: 176700,VECTOR:254,BR1:4

                                        REQUIRED PARAMETERS: NONE
```



```
                        3 .0    STARTING PROCEDURE:

                                A.LOAD AND START THE DEC/X11 PROGRAM ACCORDING TO THE
                                  DEC/X11 USERS HANDBOOK

                                B.INSURE THAT THE DEVICE ADDRESS AND VECTOR ADDRESS ARE
                                  CORRECT FOR THE PARTICULAR DR70 IN TEST

                                C.SET LOCATION DVID1 AS DESCRIBED IN THE REQUIREMENTS
                                  SECTION OF THIS DOCUMENT.
```

```
                                4 .0        PRELIMINARY OPERATIONS:


                                              BE CERTAIN THAT THE INTERFACE IS WRAPPED AROUND AS
                                              SPECIFIED IN THE REQUIREMENTS SECTION
```



```
                                5 .0        ERRORS:


                                              ALL ERRORS ARE IN STANDARD DEC/X11 FORMAT
                                              THE ORDER OF REGISTER PRINTOUT ON ERROR IS
                                              AS FOLLOWS:

                                              RHCS1  RHCS2  RHWC  RHBA  RHIS
```

```
 496
 497  000000'                        -
 (1)  000000'                                IQMODX<DRKA >176700,254,4,,,30000,0,RBUF,100,100
 (1)  000000'                                MODULE  150000,DRKA ,176700,254,4,,,30000,0,RBUF,100,100
 (2)                                         .TITLE  DRKA DEC/X11 SYSTEM EXERCISER MODULE
 (2)                                 ;       DDXCOM  VERSION  6      23-MAY-78
 (2)                                         .LIST   BIN
 (2)                                 ;*****************************************************************************
 (2)  000000'                        BEGIN:
 (2)  000000' 051104  040513  040   MODNAM: .ASCII  /DRKA / ;MODULE NAME.
 (2)  000005'    000                 XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF WBUFF USAGE
 (2)  000006' 176700                 ADDR:   176700+0                ;1ST DEVICE ADDR.
 (2)  000010' 000254                 VECTOR: 254+0                   ;1ST DEVICE VECTOR.
 (2)  000012'    200                 BR1:    .BYTE   PRTY4+0         ;1ST BR LEVEL.
 (2)  000013'    000                 BR2:    .BYTE   PRTY+0          ;2ND BR LEVEL.
 (2)  000014' 000001                 DVID1:  +1                      ;DEVICE INDICATOR 1.
 (2)  000016' 000000                 SR1:    OPEN                    ;SWITCH REGISTER 1
 (2)  000020' 000000                 SR2:    OPEN                    ;SWITCH REGISTER 2
 (2)  000022' 000000                 SR3:    OPEN                    ;SWITCH REGISTER 3
 (2)  000024' 000000                 SR4:    OPEN                    ;SWITCH REGISTER 4
 (2)                                 ;*****************************************************************************
 (2)  000026' 150000                 STAT:   150000                  ;STATUS WORD.
 (2)  000030' 000260'                INIT:   START                   ;MODULE START ADDR.
 (2)  000032' 000252'                SPUINT: MODSP                   ;MODULE STACK POINTER.
 (2)  000034' 000000                 PASCNT: 0                       ;PASS COUNTER.
 (2)  000036' 030000                 ICONT:  30000                   ;# OF ITERATIONS PER PASS=30000
 (2)  000040' 000000                 ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
 (2)  000042' 000000                 SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
 (2)  000044' 000000                 HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
 (2)  000046' 000000                 SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
 (2)  000050' 000000                 HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
 (2)  000052' 000000                 SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
 (2)  000054' 000000                 RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
 (2)  000056'                        CONFIG:
 (2)  000056' 000000                 RES1:   0                       ;RESERVED FOR MONITOR USE
 (2)  000060' 000000                 RES2:   0                       ;RESERVED FOR MONITOR USE
 (2)  000062' 000000                 SVR0:   OPEN                    ;LOC TO SAVE R0.
 (2)  000064' 000000                 SVR1:   OPEN                    ;LOC TO SAVE R1.
 (2)  000066' 000000                 SVR2:   OPEN                    ;LOC TO SAVE R2.
 (2)  000070' 000000                 SVR3:   OPEN                    ;LOC TO SAVE R3.
 (2)  000072' 000000                 SVR4:   OPEN                    ;LOC TO SAVE R4.
 (2)  000074' 000000                 SVR5:   OPEN                    ;LOC TO SAVE R5.
 (2)  000076' 000000                 SVR6:   OPEN                    ;LOC TO SAVE R6.
 (2)  000100' 000000                 CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
 (2)  000102'                        SBADR:                          ;ADDR OF GOOD DATA, OR
 (2)  000102' 000000                 ACSR:   OPEN                    ;CONTENTS OF CSR.
 (2)  000104'                        WASADR:                         ;ADDR OF BAD DATA, OR
 (2)  000104' 000000                 ASTAT:  OPEN                    ;STATUS REG CONTENTS.
 (2)  000106'                        ERRTYP:                         ;TYPE OF ERROR
 (2)  000106' 000000                 ASB:    OPEN                    ;EXPECTED DATA.
 (2)  000110' 000000                 AWAS:   OPEN                    ;ACTUAL DATA.
 (2)  000112' 000252'                RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
 (2)  000114' 000000                 WDTO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
 (2)  000116' 000000                 WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
 (2)  000120' 000000                 INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
 (2)  000122' 000000                 IDNUM:  0                       ;MODULE IDENTIFICATION NUMBER=0
 (2)  000124' 001436'                RBUFVA: RBUF                    ;READ BUFFER VIRTUAL ADDRESS
```



```
 (2)  000126' 000000                 RBUFPA: OPEN                    ;READ BUFFER PHYSICAL ADDRESS
 (2)  000130' 000000                 RBUFEA: OPEN                    ;READ BUFFER EA BITS
 (2)  000132' 000100                 RBUFSZ: 100                     ;SIZE OF THE READ BUFFER
 (2)  000134' 000000                 WBUFPA: OPEN                    ;WRITE BUFFER PHYSICAL ADDRESS
 (2)  000136' 000000                 WBUFEA: OPEN                    ;WRITE BUFFER EA BITS
 (2)  000140' 000100                 WBUFRQ: 100                     ;WRITE BUFFER SIZE REQUESTED
 (2)  000142' 000000                 WBUFSZ: OPEN                    ;WRITE BUFFER SIZE AVAILABLE
 (2)  000144' 000000                 CDERCT: OPEN                    ;CDATA/DATCK ERROR COUNT
 (2)  000146' 000000                 CDWDCT: OPEN                    ;CDATA/DATCK WORD COUNT
 (2)  000150' 000000                 FREE:   OPEN                    ;RESERVED FOR FUTURE USE
 (2)          000040                         .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
 (2)                                         .NLIST
 (2)                                         .WORD   0
 (2)                                         .LIST
 (3)                                         .ENDR
 (2)  000252'                        MODSP:
 (2)                                 ;*****************************************************************************
 498
 499  000252' 005767  177556         RESTRT: TST     PASCNT          ;ANY PASSES YET?
 500  000256' 001065                         BNE     REST1           ;YES GO RESTART
 501  000260' 016702  177522         START:  MOV     ADDR,R2 ;LOAD DEVICE REGISTER STORAGE
 502  000264' 010267  001054                 MOV     R2,RHCS1
 503  000270' 005722                         TST     (R2)+
 504  000272' 010267  001052                 MOV     R2,RHWC
 505  000276' 005722                         TST     (R2)+
 506  000300' 010267  001046                 MOV     R2,RHBA
 507  000304' 022222                         CMP     (R2)+,(R2)+
 508  000306' 010267  001034                 MOV     R2,RHCS2
 509  000312' 005722                         TST     (R2)+
 510  000314' 010267  001036                 MOV     R2,RHIS
 511  000320' 005067  001044                 CLR     FLG70           ;CHECK FOR PDP-11/70
 512  000324' 052777  002000  001012         BIS     #BIT10,@RHCS1   ;BY TRYING TO SET PSEL
 513  000332' 032777  002000  001004         BIT     #BIT10,@RHCS1   ;NOT 11/70 IF SET
 514  000340' 001006                         BNE     1$
 515  000342' 005167  001022                 COM     FLG70           ;SET FLG70,-1=11/70
 516  000346' 062702  000016                 ADD     #16,R2
 517  000352' 010267  000776                 MOV     R2,RHBAE
 518  000356' 016767  177426  000774 1$:     MOV     VECTOR,VCT      ;LOAD VECTOR STORAGE
 519  000364' 016767  000770  000770         MOV     VCT,VCTL
 520  000372' 062767  000002  000762         ADD     #2,VCTL
 521  000400' 116777  177406  000754         MOVB    BR1,@VCTL       ;SET BREAK LEVEL
 522  000406' 012700  001344'                MOV     #RHCS1,R0       ;LOAD ERROR TABLE#1
 523  000412' 012701  001402'                MOV     #TAB1,R1
 524  000416' 012021                 2$:     MOV     (R0)+,(R1)+
 525  000420' 022700  001354'                CMP     #RHBAE,R0
 526  000424' 001374                         BNE     2$
 527  000426' 016711  000724                 MOV     RHIS,@R1
 528  000432' 012777  000740' 000720 REST1:  MOV     #READ,@VCT      ;SET RETURN
 529  000440' 016767  177350  000724         MOV     DVID1,DEVCNT    ;SAVE DRIVE #
 530  000446' 012767  000001  000720         MOV     #1,POINT        ;SET POINT
 531  000454' 005067  000716                 CLR     DRVEN           ;AND DRIVE NUMBER
 532  000460' 036767  000710  000704 3$:     BIT     POINT,DEVCNT    ;IS IT THERE?
 533  000466' 001011                         BNE     33$             ;BR=YES
 534  000470' 000241                         CLC                     ;STEP POINTER
 535  000472' 006167  000676                 ROL     POINT
 536  000476' 005267  000674                 INC     DRVEN
```

```
  537  000502' 022767  000011  000666          CMP     #11,DRVEN       ;MAXED OUT?
  538  000510' 001363                          BNE     3$              ;BR=NO
  539  000512' 016777  000660  000626  33$:     MOV     DRVEN,@RHCS2    ;LOAD DRIVE #
  540  000520' 012777  000011  000616          MOV     #11,@RHCS1      ;CLEAR DEVICE
  541  000526' 052777  000040  000612          BIS     #40,@RHCS2      ;CLEAR CONTROLLER
  542  000534' 104414  000000'                 GWBUF$, BEGIN           ;GET WRITE BUFFER INFORMATION
  543  000540' 005767  000624                  TST     FLG70           ;11/70?
  544  000544' 001425                          BEQ     1$              ;BR=NO
  545  000546' 016767  177362  000642          MOV     WBUFPA,WAD70    ;GET 11/70 22 BIT ADDRESSES
  546  000554' 016767  177356  000636          MOV     WBUFEA,WAD70X
  547  000562' 104416  000000' 001416'         MAP22$, BEGIN,WAD70             ; GET 22-BIT ADDR FROM 18-BIT ADDR
  548  000570' 016777  000626  000554          MOV     WAD22,@RHBA     ;LOAD BUSS ADDRESS REGISTER
  549  000576' 016777  000622  000550          MOV     WAD22X,@RHBAE   ;AND EXTENDED ADDRESS BITS
  550  000604' 005067  000570                  CLR     FUNC            ;CLEAR FUNCTION WORD
  551  000610' 052767  100000  000102          BIS     #100000,4$      ;MAKE IT A MOVB INSTRUCTION
  552  000616' 000425                          BR      2$              ;CONTINUE @2$
  553  000620' 016777  177310  000524  1$:     MOV     WBUFPA,@RHBA    ;LOAD BUSS ADDRESS REGISTER
  554  000626' 016767  177304  000530          MOV     WBUFEA,WBEA     ;POSITION E.A. BITS
  555  000634' 042767  100000  000056          BIC     #100000,4$      ;MAKE IT A MOV INSTRUCTION
  556  000642' 000241                          CLC
  557  000644' 006167  000514                  ROL     WBEA
  558  000650' 006167  000510                  ROL     WBEA
  559  000654' 006167  000504                  ROL     WBEA
  560  000660' 006167  000500                  ROL     WBEA
  561  000664' 016767  000474  000506          MOV     WBEA,FUNC       ;LOAD E.A. BITS
  562  000672' 016767  177244  177232  2$:     MOV     WBUFSZ,RBUFSZ
  563  000700' 016777  177236  000442          MOV     WBUFSZ,@RHWC    ;LOAD WORD COUNT AND
  564  000706' 005477  000436                  NEG     @RHWC           ;NEGATE IT
  565  000712' 062767  000161  000460          ADD     #161,FUNC       ;ADD WRITE AND GO + I.E. BITS
  566  000720' 016777  000454  000416  4$:     MOV     FUNC,@RHCS1     ;GO!!(IF 11/70 THIS WILL BE A MOVB.)
  567  000726' 052777  000020  000422          BIS     #20,@RHIS       ;PRIME CYCLE
  568  000734' 104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
```



```
  570
  571  000740' 042777  000100  000376  READ:   BIC     #100,@RHCS1     ;DROP I.E. BIT
  572  000746' 004767  000324                  JSR     PC,ERROR        ;GO CHECK FOR ERROR
  573                                          ;---------------------------------------------------------------
(1)    000752' 000004  000000' 000760'         PINQS,BEGIN,1$          ; QUEUE UP TO CONTINUE AT 1$ AND RTI
(1)                                            ;---------------------------------------------------------------
  574  000760' 012777  001176' 000372  1$:     MOV     #CMPLT,@VCT     ;SET RETURN
  575  000766' 032777  000200  000362          BIT     #200,@RHIS      ;DEVICE READY?
  576  000774' 001774                          BEQ     .-6             ;LOOP UNTIL READY
  577  000776' 104415  000000' 000124'         GETPA$,BEGIN, RBUFVA            ;GET PHYSICAL ADDRESS FROM 16-BIT RBUFVA
  578  001004' 005767  000360                  TST     FLG70           ;11/70
  579  001010' 001425                          BEQ     11$             ;BR=NO
  580  001012' 016767  177110  000406          MOV     RBUFPA,RAD70    ;GET 22 BIT ADDRESS
  581  001020' 016767  177104  000402          MOV     RBUFEA,RAD70X
  582  001026' 104416  000000' 001426'         MAP22$, BEGIN,RAD70             ; GET 22-BIT ADDR FROM 18-BIT ADDR
  583  001034' 016777  000372  000310          MOV     RAD22,@RHBA     ;LOAD BUSS ADDRESS REGISTER
  584  001042' 016777  000366  000304          MOV     RAD22X,@RHBAE   ;AND EXTENDED ADDRESS BITS
  585  001050' 005067  000324                  CLR     FUNC            ;CLEAR FUNCTION WORD
  586  001054' 052767  100000  000074          BIS     #100000,3$      ;MAKE IT A MOVB INSTRUCTION
  587  001062' 000425                          BR      2$              ;CONTINUE @ 2$
  588  001064' 016777  177036  000260  11$:     MOV     RBUFPA,@RHBA    ;LOAD BUSS ADDRESS
  589  001072' 016767  177032  000266          MOV     RBUFEA,RBEA     ;POSITION E.A. BITS
  590  001100' 042767  100000  000050          BIC     #100000,3$      ;MAKE IT A MOV INSTRUCTION
  591  001106' 000241                          CLC
  592  001110' 006167  000252                  ROL     RBEA
  593  001114' 006167  000246                  ROL     RBEA
  594  001120' 006167  000242                  ROL     RBEA
  595  001124' 006167  000236                  ROL     RBEA
  596  001130' 016767  000232  000242          MOV     RBEA,FUNC       ;LOAD E.A. BITS
  597  001136' 016777  176770  000204  2$:     MOV     RBUFSZ,@RHWC    ;LOAD WORD COUNT AND
  598  001144' 005477  000200                  NEG     @RHWC           ;NEGATE IT
  599  001150' 062767  000171  000222          ADD     #171,FUNC       ;ADD READ AND GO + I.E.
  600  001156' 016777  000216  000160  3$:     MOV     FUNC,@RHCS1     ;GO!!(IF 11/70 THIS WILL BE A MOVB.)
  601  001164' 052777  000020  000164          BIS     #20,@RHIS       ;PRIME CYCLE
  602  001172' 104400  000000'                 EXIT$,BEGIN                     ;EXIT TO MONITOR. MODULE WAIT FOR INTERRUPT.
```

```
  604
  605  001176' 042777  000100  000140  CMPLT:  BIC     #100,@RHCS1      ;DROP I.E. BIT
  606  001204' 004767  000066                  JSR     PC,ERROR         ;GO CHECK FOR ERROR
  607                                           ;-----------------------------------------------------------
  (1)  001210' 000004  000000' 001216'          PIRQ$,BEGIN,1$           ; QUEUE UP TO CONTINUE AT 1$ AND RTI
  (1)                                           ;-----------------------------------------------------------
  608  001216' 032777  000200  000132  1$:     BIT     #200,@RHIS       ;CHECK FOR DEVICE READY
  609  001224' 001774                          BEQ     1$               ;LOOP UNTIL READY
  610  001226' 104412  000000' 000126'         CDATA$,BEGIN,RBUFPA      ; REQUEST FOR MONITOR TO CHECK DATA
  (1)  001234' 001236'                         .+2                      ; IF ERROR, CONTINUE
  611  001236' 104407  000000'                 BREAK$,BEGIN             ;TEMPORARY RETURN TO MONITOR....
  (1)  001242' 104407  000000'                 BREAK$,BEGIN             ;THEN CONTINUE AT NEXT INSTRUCTION.
  612  001246' 104407  000000'                 BREAK$,BEGIN             ;TEMPORARY RETURN TO MONITOR....
  (1)  001252' 104407  000000'                 BREAK$,BEGIN             ;THEN CONTINUE AT NEXT INSTRUCTION.
  613  001256' 104407  000000'                 BREAK$,BEGIN             ;TEMPORARY RETURN TO MONITOR....
  (1)  001262' 104407  000000'                 BREAK$,BEGIN             ;THEN CONTINUE AT NEXT INSTRUCTION.
  614  001266'                         FINI:
  (1)  001266' 104413  000000'                 ENDIT$,BEGIN             ;SIGNAL END OF ITERATION.
  (1)                                                                   ;MONITOR SHALL TEST END OF PASS
  615  001272' 000167  177134                  JMP     REST1
```



```
  617  001276' 005777  000042          ERROR:  TST     @RHCS1           ;ERROR PRESENT?
  618  001302' 100401                          BMI     1$               ;BR=YES
  619  001304' 000207                          RTS     PC               ;RETURN OK
  620  001306' 016767  000032  176564  1$:     MOV     RHCS1,CSRA       ;LOAD ERROR INFO
  621  001314' 017767  000024  176560          MOV     @RHCS1,ACSR
  622  001322' 017767  000020  176554          MOV     @RHCS2,ASTAT
  623                                           ;*************************************************************
  (1)  001330' 104405  000000' 001402'          HRDER$,BEGIN,TAB1        ;
  (1)                                           ;*************************************************************
  624                                           ;-------------------------------------------------------------
  (1)  001336' 000004  000000' 000432'          PIRQ$,BEGIN,REST1                ; QUEUE UP TO CONTINUE AT REST1 AND RTI
  (1)                                           ;-------------------------------------------------------------
```

```
                                        RHCS1:  000
                                        RHCS2:  000
                                        RHWC:   000
                                        RHBA:   000
                                        RHBAE:  000
                                        RHIS:   000
                                        VCT:    000
                                        VCTL:   000
                                        WBEA:   000
                                        RBEA:   000
                                        FLG70:  000
                                        DEVCNT: 000
                                        POINT:  000
                                        DRVEN:  000
                                        FUNC:   000


                                        TAB1:   RHCS1
                                                RHCS2
                                                RHWC
                                                RHBA
                                                RHIS
                                                -1


                                        WAD70:  000
                                        WAD70X: 000
                                        WAD22:  000
                                        WAD22X: 000

                                        RAD70:  000
                                        RAD70X: 000
                                        RAD22:  000
                                        RAD22X: 000

                                        RBUF:   .BLKW   110.

                                                .END
```



```
ACSR     000102R          497#    621*
ADDR     000006R          497#    501
ADDR22=  001000           497#
ASB      000106R          497#
ASTAT    000104R          497#    622*
AWAS     000110R          497#
BEGIN    000000R          497#    542     547     568     573     577     582     602     607     610     611     612     613
                          614     623     624
BIT0   = 000001           497#
BIT1   = 000002           497#
BIT10  = 002000           497#    512     513
BIT11  = 004000           497#
BIT12  = 010000           497#
BIT13  = 020000           497#
BIT14  = 040000           497#
BIT15  = 100000           497#
BIT2   = 000004           497#
BIT3   = 000010           497#
BIT4   = 000020           497#
BIT5   = 000040           497#
BIT6   = 000100           497#
BIT7   = 000200           497#
BIT8   = 000400           497#
BIT9   = 001000           497#
BREAK$=  104407           497#    611     612     613
BR1      000012R          497#    521
BR2      000013R          497#
BTOD$  = 104421           497#
CDATA$=  104412           497#    610
CDERCT   000144R          497#
CDWDCT   000146R          497#
CMPLT    001176R          574     605#
CONFIG   000056R          497#
CSRA     000100R          497#    620*
DATCK$=  104411           497#
DATER$=  104404           497#
DEVCNT   001372R          529*    532     639#
DRVEN    001376R          531*    536*    537     539     641#
DVID1    000014R          497#    529
ENDIF$=  104413           497#    614
END$   = 104410           497#
ERROR    001276R          572     606     617#
ERRTYP   000106R          497#
EXIT$  = 104400           497#    568     602
FINI     001266R          614#
FLG70    001370R          511*    515*    543     578     638#
FREE     000150R          497#
FUNC     001400R          550*    561*    565*    566     585*    596*    599*    600     642#
GETPA$=  104415           497#    577
GWBUF$=  104414           497#    542
HRDCNT   000044R          497#
HRDER$=  104405           497#    623
HRDPAS   000050R          497#
ICONT    000036R          497#
ICOUNT   000040R          497#
IDNUM    000122R          497#
```

```
IMODX.= 000000          497#    542
INIT    000030R         497#
INTR    000120R         497#
MAP22$= 104416          497#    547     582
MODNAM  000000R         497#
MODSP   000252R         497#
MSGN$ = 104403          497#
MSGS$ = 104402          497#
MSG$  = 104401          497#
NULL  = 000000          497#
OPEN  = 000000          497#
OTOA$ = 104420          497#
PASCNT  000034R         497#    499
PIRQ$ = 000004          497#    573     607     624
POINT   001374R         530*    532     535*    640#
POPSP = 005726          497#
POPSP2= 022626          497#
PRTY  = 000000          497#
PRTY0 = 000000          497#
PRTY1 = 000040          497#
PRTY2 = 000100          497#
PRTY3 = 000140          497#
PRTY4 = 000200          497#
PRTY5 = 000240          497#
PRTY6 = 000300          497#
PRTY7 = 000340          497#
PS    = 177776          497#
PSW   = 177776          497#
PUSH  = 005746          497#
PUSH2 = 024646          497#
RAD22   001432R         583     661#
RAD22X  001434R         584     662#
RAD70   001426R         580*    582     659#
RAD70X  001430R         581*    660#
RAND$ = 104417          497#
RANNUM  000054R         497#
RBEA    001366R         589*    592*    593*    594*    595*    596     637#
RBUF    001436R         497     664#
RBUFEA  000130R         497#    581     589
RBUFPA  000126R         497#    580     588     610
RBUFSZ  000132R         497#    562*    597
RBUFVA  000124R         497#    577
READ    000740R         528     571#
RESTRT  000252R         497     499#
REST1   000432R         500     528#    615     624
RES1    000056R         497#
RES2    000060R         497#
RHBA    001352R         506*    548*    553*    583*    588*    631#    648
RHBAE   001354R         517*    525     549*    584*    632#
RHCS1   001344R         502*    512*    513     522     540*    566*    571*    600*    605*    617     620     621     628#
                        645
RHCS2   001346R         508*    539*    541*    622     629#    646
RHIS    001356R         510*    527     567*    575     601*    608     633#    649
RHWC    001350R         504*    563*    564*    597*    598*    630#    647
RSTRT   000112R         497#
SBADR   000102R         497#
```



```
SOFCNT  000042R         497#
SOFER$= 104406          497#
SOFPAS  000046R         497#
SPOINT  000032R         497#
SPSIZ = 000040            7#    497
SR1     000016R         497#
SR2     000020R         497#
SR3     000022R         497#
SR4     000024R         497#
START   000260R         497     501#
STAT    000026R         497#
SVR0    000062R         497#
SVR1    000064R         497#
SVR2    000066R         497#
SVR3    000070R         497#
SVR4    000072R         497#
SVR5    000074R         497#
SVR6    000076R         497#
SYSCNT  000052R         497#
TAB1    001402R         523     623     645#
TRPDFD= 000022          497#
VCT     001360R         518*    519     528*    574*    634#
VCTL    001362R         519*    520*    521*    635#
VECTOR  000010R         497#    518
WAD22   001422R         548     656#
WAD22X  001424R         549     657#
WAD70   001416R         545*    547     654#
WAD70X  001420R         546*    655#
WASADR  000104R         497#
WBEA    001364R         554*    557*    558*    559*    560*    561     636#
WBUFEA  000136R         497#    546     554
WBUFPA  000134R         497#    545     553
WBUFRQ  000140R         497#
WBUFSZ  000142R         497#    562     563
WDFR    000116R         497#
WDTO    000114R         497#
XFLAG   000005R         497#
.     = 001772R         576     610     664#
```

```
BKMOD       95#
BREAK      185#     611      612      613
BTOD       204#
CKDATA     240#     610
DATACK     249#
DATERR     138#
DFSEVN     272#     497
DSEVNT     282#     497
END        175#
ENDIT      166#     614
ENDMOD     171#
EQUATS     288#     497
EXIT       120#     568      602
GETPA      231#     577
GWBUFF     219#     542
HRDER      128#     623
IOMOD       91#
IOMODP     115#
IOMODR     111#
IOMODX     107#     497
MAP22      235#     547      582
MODULE       8#     497
MSG        154#
MSGN       158#
MSGS       162#
NBKMOD     103#
OTOA       190#
PIRQ       179#     573      607      624
RAND       124#
SBKMOD      99#
SOFER      144#


. ABS.  000000      000
        001772      001

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 CXDRKA,CXDRKA.LST/CRF=DDXCOM.C6,CXDRKA.SRC
 RUN-TIME: 3 4 .8 SECONDS
 RUN-TIME RATIO: 38/9=4.1
 CORE USED:  7K  (13 PAGES)
```

```
                        .REM_

PRODUCT CODE:   AC-S254A-MC

PRODUCT NAME: CXRHAA0 CSS RH01 DEC/X MOD

DATE:         15 JUL 1980

AUTHOR(S):    JAMES M. DUPRE

UPDATE AUTHOR(S):JAMES M. DUPRE

MAINTAINER:   C.S.S. PRODUCTS GROUP
```

```
                                1 .0        ABSTRACT:


                                            RHA IS AN SBKMOD THAT EXERCISES THE RH01 MASS BUSS SWITCH.
                                            RHA ALSO CONTROLS THE MODULES THAT ACCESS THE RH01 SWITCH
                                            UP TO NINE (9) MODULES PER RHA MODULE.

                                              THE RHA MODULE CONTROLS THE MODULES FOR ANY DEVICES WHICH
                                              RESIDE ON THE SHARED BUS.UPTO 9 MODULES CAN BE
                                              CONTROLLED BY RHA.
```

```
                                2 .0        REQUIREMENTS:


                                            RH01 AND AT LEAST ONE DEVICE TO ACCESS IT.
                                            TWO SYSTEMS WITH RH01 AND AT LEAST ONE MASSBUS
                                            DEVICE ON THE SHARED BUS.

                                            THE RH01 MUST BE IN DUAL ACCESS MODE.
```

```
                                3 .0        PRELIMINARY OPERATIONS:


                                            1.INSERT MODULE POSITION (OF ACCESSING DEVICES) STARTING
                                            AT LOCATION DVP1
                                            2.IF EXERCISER IS TO CONNECT THE SHARED BUS ON THE
                                              FIRST PASS A ONE MUST BE SET IN LOCATION "SWITCH"(IN
                                              THIS MODULE RELATIVE ADDRESS 1000)


                                                    IT IS RECOMMENDED THAT THE RHA MODULE BE
                                              SINKED IN FRONT OF THE MODULES FOR THOSE DEVICES WHICH
                                              WILL BE RUN ON THE SHARED BUS.FOR EXAMPLE, A SYSTEM
                                              CONTAINS AN RH01 AND A TU45 MAGTAPE DRIVE ON THE SHARED
                                              BUS. THE MAP OF THE RUNTIME SYSTEM WOULD LOOK LIKE
                                              THE ONE GIVEN BELOW.

                                                    THE RHA RH01 MODULE IS IN FRONT OF THE MODULE
                                            FOR TU45.THE PHYSICAL POSITION OF THE TAPE DRIVE MODULE
                                            IS THEN DETERMINED RELATIVE TO THE RH01 MODULE AND THAT
                                            POSITION NUMBER (I.E. 1 IN THIS CASE)IS INSERTED IN TO
                                            MODULE LOCATION DVP1.(MODULE RELATIVE POSITION 754).THIS
                                            IS DONE WITH <MOD> COMMAND.

                                              EX: MOD RHAA0 754 <CR>

                                              THE MODULE RETURNS THE PHYSICAL ADDRESS OF THAT LOCATION
                                              AND THE CONTENTS OF THAT ADDRESS.THEN TYPE IN A 1 <CR>.
                                            EXAMPLE:

                                                    AFTER LINKING, RUN THIS EXERCISER.

                                              .R DECX11



                                              CMD>                    ;;WILL PROMPT AFTER SOME MESSAGES
                                              CMD>MAP                 ;;COMMAND TO PRINT MAP ON CONSOLE

                                              CMD>

                                              RHAA0  AT VA:057266   STAT:040000
                                              TMBJ0  AT VA:063006   STAT:150000
                                              RKAG0  AT VA:072570   STAT:150000
                                              CPAG0  AT VA:061124   STAT:040020


                                                    ALL OF THE ABOVE MODULES ARE LISTED RELATIVE TO THE TOP
                                                    MODULE RHAA0, SO RELATIVE POSITION OF THE TOP
                                                    MODULE RHAA0 IS ZERO AND RELATIVE POSITION OF
                                                    TMBJ0 MODULE IS ONE, RKAG0 MODULE IS NOT ON
                                                    THE SHARED BUS SO IT IS NOT INCLUDED.ONLY MODULE
                                                    ON THE SHARED BUS WILL BE INCLUDED.
                                K                   SO ON.

                                                    IT IS ALWAYS GOOD PRACTICE TO PUT THE RHAA0 MODULE
```

```
499                                     AT THE TOP.
500
501                                             IN THE ABOVE EXAMPLE ONLY ONE MODULE
502                                     IS ASSOCIATED WITH THE RHAA0. INSERT RELATIVE
503                                     POSITION OF THIS MODULE AT THE LOCATION "DVP1"
504                                     AS SHOWN BELOW. USE <MOD> COMMAND TO INSERT
505                                     THIS VALUE.
506
507
508
509                                     DVP1:  1                ;;INCLUDE TMBJ0 MODULE
510
```

```
                                .GLOBL MODQ
                                       SBKMOD <RHAA >0,0,0,0,0,1,0
                                       MODULE  40000,RHAA ,0,0,0,0,0,1,0
                                       .TITLE  RHAA DEC/X11 SYSTEM EXERCISER MODULE
                                ;      DDXCOM  VERSION  6       23-MAY-78
                                       .LIST   BIN
                                ;*****************************************************************************
000000'                         BEGIN:
000000' 044122  040501      040 MODNAM: .ASCII  /RHAA / ;MODULE NAME.
000005'     000                 XFLAG:  .BYTE   OPEN            ;USED TO KEEP TRACK OF WBUFF USAGE
000006' 000000                  ADDR:   0+0             ;1ST DEVICE ADDR.
000010' 000000                  VECTOR: 0+0                     ;1ST DEVICE VECTOR.
000012'     000                 BR1:    .BYTE   PRTY0+0         ;1ST BR LEVEL.
000013'     000                 BR2:    .BYTE   PRTY0+0         ;2ND BR LEVEL.
000014' 000001                  DVID1:  0+1                     ;DEVICE INDICATOR 1.
000016' 000000                  SR1:    OPEN                    ;SWITCH REGISTER 1
000020' 000000                  SR2:    OPEN                    ;SWITCH REGISTER 2
000022' 000000                  SR3:    OPEN                    ;SWITCH REGISTER 3
000024' 000000                  SR4:    OPEN                    ;SWITCH REGISTER 4
                                ;*****************************************************************************
000026' 040000                  STAT:   40000                   ;STATUS WORD.
000030' 000224'                 INIT:   START                   ;MODULE START ADDR.
000032' 000224'                 SPOINT: MODSP                   ;MODULE STACK POINTER.
000034' 000000                  PASCNT: 0                       ;PASS COUNTER.
000036' 000001                  ICONT:  1                       ;# OF ITERATIONS PER PASS=1
000040' 000000                  ICOUNT: 0                       ;LOC TO COUNT ITERATIONS
000042' 000000                  SOFCNT: 0                       ;LOC TO SAVE TOTAL SOFT ERRORS
000044' 000000                  HRDCNT: 0                       ;LOC TO SAVE TOTAL HARD ERRORS
000046' 000000                  SOFPAS: 0                       ;LOC TO SAVE SOFT ERRORS PER PASS
000050' 000000                  HRDPAS: 0                       ;LOC TO SAVE HARD ERRORS PER PASS
000052' 000000                  SYSCNT: 0                       ;# OF SYS ERRORS ACCUMULATED
000054' 000000                  RANNUM: 0                       ;HOLDS RANDOM # WHEN RAND MACRO IS CALLED
000056'                         CONFIG:
000056' 000000                  RES1:   0                       ;RESERVED FOR MONITOR USE
000060' 000000                  RES2:   0                       ;RESERVED FOR MONITOR USE
000062' 000000                  SVR0:   OPEN                    ;LOC TO SAVE R0.
000064' 000000                  SVR1:   OPEN                    ;LOC TO SAVE R1.
000066' 000000                  SVR2:   OPEN                    ;LOC TO SAVE R2.
000070' 000000                  SVR3:   OPEN                    ;LOC TO SAVE R3.
000072' 000000                  SVR4:   OPEN                    ;LOC TO SAVE R4.
000074' 000000                  SVR5:   OPEN                    ;LOC TO SAVE R5.
000076' 000000                  SVR6:   OPEN                    ;LOC TO SAVE R6.
000100' 000000                  CSRA:   OPEN                    ;ADDR OF CURRENT CSR.
000102'                         SBADR:                          ;ADDR OF GOOD DATA, OR
000102' 000000                  ACSR:   OPEN                    ;CONTENTS OF CSR.
000104'                         WASADR:                         ;ADDR OF BAD DATA, OR
000104' 000000                  ASTAT:  OPEN                    ;STATUS REG CONTENTS.
000106'                         ERRTYP:                         ;TYPE OF ERROR
000106' 000000                  ASB:    OPEN                    ;EXPECTED DATA.
000110' 000000                  AWAS:   OPEN                    ;ACTUAL DATA.
000112' 000224'                 RSTRT:  RESTRT                  ;RESTART ADDRESS AFTER END OF PASS
000114' 000000                  WDTO:   OPEN                    ;WORDS TO MEMORY PER ITERATION
000116' 000000                  WDFR:   OPEN                    ;WORDS FROM MEMORY PER ITERATION
000120' 000000                  INTR:   OPEN                    ;# OF INTERRUPTS PER ITERATION
000122' 000000                  IDNUM:  0                       ;MODULE IDENTIFICATION NUMBER=0
```



```
000040                                  .REPT   SPSIZ           ;MODULE STACK STARTS HERE.
                                        .NLIST
                                        .WORD   0
                                        .LIST
                                        .ENDR
000224'                         MODSP:
                                ;*****************************************************************************

000224'                         RESTRT:
000224' 005000                  START:  CLR     R0              ;CLEAR REGISTERS
000226' 005001                          CLR     R1
000230' 005002                          CLR     R2
000232' 005003                          CLR     R3
000234' 012700  000732'                 MOV     #DV1,R0         ;CLEAR MODULE STORAGE
000240' 005020                  1$:     CLR     (R0)+
000242' 022700  000754'                 CMP     #DV11+2,R0
000246' 001374                          BNE     1$
000250' 012700  000754'         2$:     MOV     #DVP1,R0        ;GET POINTER
000254' 012701  000732'                 MOV     #DV1,R1         ;GET STORAGE ADDRESS
000260' 005710                  MODS1:  TST     @R0             ;IS THERE A MODULE HERE?
000262' 100421                          BMI     REJEC           ;YES BUT DON'T SELECT IT
000264' 001410                          BEQ     NODV            ;NO
000266' 012003                  MODS2:  MOV     (R0)+,R3        ;SAVE POSITION INDIACATOR
000270' 006303                          ASL     R3              ;COMPENSATE
000272' 016321  000000G                 MOV     MODQ(R3),(R1)+  ;STORE MODULE ADDRESS
000276' 022700  000776'                 CMP     #DVP11+2,R0     ;FINISHED?
000302' 001366                          BNE     MODS1           ;NO CONTINUE
000304' 000424                  MODS3:  BR      FLPFLP          ;GO CHECK STATUS
000306' 005720                  NODV:   TST     (R0)+           ;STEP R0
000310' 022700  000776'                 CMP     #DVP11+2,R0     ;CHECK FOR LAST MODULE
000314' 001002                          BNE     1$              ;NOT DONE YET
000316' 000167  177762                  JMP     MODS3           ;ALL DONE EXIT SUBROUTINE
000322' 000167  177732          1$:     JMP     MODS1           ;CONTINUE ON MACDUFF!

000326' 111002                  REJEC:  MOVB    @R0,R2          ;CHECK FOR SELECTION
000330' 006302                          ASL     R2
000332' 016203  000000G                 MOV     MODQ(R2),R3
000336' 032763  040000  000026          BIT     #SELBIT,26(R3)
000344' 001760                          BEQ     NODV
000346' 042763  040000  000026          BIC     #SELBIT,26(R3)
000354' 000754                          BR      NODV

000356' 005767  000416          FLPFLP: TST     SWITCH
000362' 001066                          BNE     TRNON
```

```
                                TRNOF:  TST     DV1             ;IS THERE ANY MODULES?
                                        BEQ     2$              ;NO DERE AINT SO LEAVE
                                        MOV #DV1,R0
                                1$:     MOV     (R0),R1
                                        BIT     #SELBIT,26(R1)
                                        BEQ     111$
                                11$:    BIS     #STFLG,26(R1)   ;SET STOP FLAG
                                        BIC     #SELBIT,26(R1)  ;CLEAR SELECT BIT
                                        MOV     6(R1),R4        ;GET DEVICE ADDRESS
                                        MOV     14(R1),DEVCNT   ;GET DRIVE COUNT
                                        MOV     #1,POINT        ;INITAILIZE POINTER
                                        CLR     DRVEN           ;SET DRIVE #
                                3$:     BIT     POINT,DEVCNT    ;DRIVE PRESENT?
                                        BNE     4$              ;BR=YES
                                31$:    CLC
                                        ROL     POINT
                                        INC     DRVEN
                                        CMP     #11,DRVEN       ;LAST ONE?
                                        BNE     3$              ;BR=NO
                                        BR      111$            ;BR=YES
                                4$:     MOV     DRVEN,10(R4)    ;SET DRIVE # TO CONTROLLER
                                        MOV     #13,@R4         ;RELEASE RH01
                                        BR      31$             ;CONTINUE
                                111$:   ADD     #2,R0
                                        TST     (R0)
                                        BNE     1$
                                2$:     INC     SWITCH
                                        ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
(1)                                                             ;MONITOR SHALL TEST END OF PASS
```



```
                                TRNON:  TST     DV1             ;IS DERE ANY MODULES HERE?
                                        BEQ     2$              ;NO BUT TURN ON ANYWAY
                                        MOV     #DV1,R0         ;GET TABLE ADDRESS
                                1$:     MOV     (R0),R1
                                        BIT     #SELBIT,26(R1)
                                        BNE     11$
                                        BIC     #STFLG,26(R1)           ;CLEAR STOP FLAG
                                        BIS     #SELBIT,26(R1)  ;SET SEL
                                        MOV     6(R1),R4        ;GET DEVICE ADDRESS
                                        MOV     14(R1),DEVCNT   ;GET DRIVE COUNT
                                        MOV     #1,POINT        ;INITAILIZE POINTER
                                        CLR     DRVEN           ;SET DRIVE #
                                3$:     BIT     POINT,DEVCNT    ;DRIVE PRESENT?
                                        BNE     4$              ;BR=YES
                                31$:    CLC
                                        ROL     POINT
                                        INC     DRVEN
                                        CMP     #11,DRVEN       ;LAST ONE?
                                        BNE     3$              ;BR=NO
                                        BR      11$             ;BR=YES
                                4$:     MOV     DRVEN,10(R4)    ;SET DRIVE # TO CONTROLLER
                                41$:    MOV     #10,@R4         ;TRY TO SET BIT 2
                                        BIT     #10,@R4         ;SEE IF IT'S THERE
                                        BEQ     41$             ;CONTINUE UNTIL SUCCESSFUL
                                        MOV     #377,16(R4)     ;CLEAR ATTN. BIT
                                        BR      31$
                                11$:    ADD     #2,R0   ;INC ADDRESS TABLE
                                        TST     (R0)
                                        BNE     1$

                                2$:     CLR     SWITCH
                                        ENDIT$,BEGIN            ;SIGNAL END OF ITERATION.
(1)                                                             ;MONITOR SHALL TEST END OF PASS
```

```
                                        ;SPECIAL ADDRESSES, CONSTANTS, & OTHER

000730' 000000          DV0:    0       ;DO NOT USE
000732' 000000          DV1:    0                       ;DEVICE MODULE ADDRESS ABSOLUTE
                                                        ;THERE ARE 9 LOCATION TO INCLUDE
                                                        ;MAXIMUM 9 MODULES.
000734' 000000          DV2:    0
000736' 000000          DV3:    0
000740' 000000          DV4:    0
000742' 000000          DV5:    0
000744' 000000          DV6:    0
000746' 000000          DV7:    0
000750' 000000          DV10:   0
000752' 000000          DV11:   0
000754' 000000          DVP1:   0                       ;;MODULE POINTER, THERE ARE 9 LOCATIONS
                                                        ;;TO INCLUDE MAXIMUM 9 MODULES.
000756' 000000          DVP2:   0
000760' 000000          DVP3:   0
000762' 000000          DVP4:   0
000764' 000000          DVP5:   0
000766' 000000          DVP6:   0
000770' 000000          DVP7:   0
000772' 000000          DVP10:  0
000774' 000000          DVP11:  0
000776' 000000                  0
001000' 000000          SWITCH: 0               ;RESTART ALL FLAG
001002' 000000          YES:    0
001004' 000000          SET:    0
001006' 000000          CNT:    0
001010' 000000          DEVCN1: 000
001012' 000000          DRVEN:  000
001014' 000000          POIN1:  000
        040000          SELBIT=40000
        020000          STFLG=20000
001016' 000310          PATCH:  .BLKW 200.
        000001          .END
```



```
ACSK    000102R         514#
ADDR    000006R         514#
ADDR22= 001000          514#
ASB     000106R         514#
ASTAT   000104R         514#
AWAS    000110R         514#
BEGIN   000000R         514#    582     620
BIT0  = 000001          514#
BIT1  = 000002          514#
BIT10 = 002000          514#
BIT11 = 004000          514#
BIT12 = 010000          514#
BIT13 = 020000          514#
BIT14 = 040000          514#
BIT15 = 100000          514#
BIT2  = 000004          514#
BIT3  = 000010          514#
BIT4  = 000020          514#
BIT5  = 000040          514#
BIT6  = 000100          514#
BIT7  = 000200          514#
BIT8  = 000400          514#
BIT9  = 001000          514#
BREAK$= 104407          514#
BR1     000012R         514#
BR2     000013R         514#
BTOD$ = 104421          514#
CDATA$= 104412          514#
CNT     001006R         656#
CONFIG  000056R         514#
CSRA    000100R         514#
DATCK$= 104411          514#
DATER$= 104404          514#
DEVCNT  001010R         564*    567     598*    601     657#
DRVEN   001012R         566*    571*    572     575     600*    605*    606     609     658#
DVID1   000014R         514#
DVP1    000754R         525     642#
DVP10   000772R         650#
DVP11   000774R         533     537     651#
DVP2    000756R         644#
DVP3    000760R         645#
DVP4    000762R         646#
DVP5    000764R         647#
DVP6    000766R         648#
DVP7    000770R         649#
DV0     000730R         630#
DV1     000732R         521     526     555     557     589     591     631#
DV10    000750R         640#
DV11    000752R         523     641#
DV2     000734R         634#
DV3     000736R         635#
DV4     000740R         636#
DV5     000742R         637#
DV6     000744R         638#
DV7     000746R         639#
ENDIT$= 104413          514#    582     620
```

```
END$  = 104410          514#
ERRTYP  000106R         514#
EXIT$ = 104400          514#
FLPFLP  000356R         535     551#
GETPA$= 104415          514#
GWBUF$= 104414          514#
HRDCNT  000044R         514#
HRDER$= 104405          514#
HRDPAS  000050R         514#
ICONT   000036R         514#
ICOUNT  000040R         514#
IDNUM   000122R         514#
INIT    000030R         514#
INTR    000120R         514#
MAP22$= 104416          514#
MODNAM  000000R         514#
MODQ  = ****** G        513#    532     544
MODSP   000224R         514#
MODS1   000260R         527#    534     540
MODS2   000266R         530#
MODS3   000304R         535#    539
MSGN$ = 104403          514#
MSGS$ = 104402          514#
MSG$  = 104401          514#
NODV    000306R         529     536#    546     548
NULL  = 000000          514#
OPEN  = 000000          514#
OTOA$ = 104420          514#
PASCNT  000034R         514#
PATCH   001016R         662#
PIRQ$ = 000004          514#
POINT   001014R         565#    567     570#    599#    601     604#    659#
POPSP = 005726          514#
POPSP2= 022626          514#
PRTY  = 000000          514#
PRTY0 = 000000          514#
PRTY1 = 000040          514#
PRTY2 = 000100          514#
PRTY3 = 000140          514#
PRTY4 = 000200          514#
PRTY5 = 000240          514#
PRTY6 = 000300          514#
PRTY7 = 000340          514#
PS    = 177776          514#
PSW   = 177776          514#
PUSH  = 005746          514#
PUSH2 = 024646          514#
RAND$ = 104417          514#
RANNUM  000054R         514#
REJEC   000326R         528     542#
RESTRT  000224R         514     516#
RES1    000056R         514#
RES2    000060R         514#
RSTRT   000112R         514#
SBADR   000102R         514#
SELBIT= 040000          545     547     559     562     593     596     660#
```



```
SET     001004R         655#
SOFCNT  000042R         514#
SOFER$= 104406          514#
SOFPAS  000046R         514#
SPOINT  000032R         514#
SPSIZ = 000040            7#    514
SR1     000016R         514#
SR2     000020R         514#
SR3     000022R         514#
SR4     000024R         514#
START   000224R         514     517#
STAT    000026R         514#
STFLG = 020000          561     595     661#
SVR0    000062R         514#
SVR1    000064R         514#
SVR2    000066R         514#
SVR3    000070R         514#
SVR4    000072R         514#
SVR5    000074R         514#
SVR6    000076R         514#
SWITCH  001000R         551     581#    619#    653#
SYSCNT  000052R         514#
TRNOF   000364R         555#
TRNON   000540R         552     589#
TRPDFD= 000022          514#
VECTOR  000010R         514#
WASADR  000104R         514#
WDFR    000116R         514#
WDTO    000114R         514#
XFLAG   000005R         514#
YES     001002R         654#
.     = 001636R         662#
```

```
BKMOD       95#
BREAK      185#
BTOD       204#
CKDATA     240#
DATACK     249#
DATERR     138#
DFSEVN     272#    514
DSEVNT     282#    514
END        175#
ENDIT      166#    582     620
ENDMOD     171#
EQUATS     288#    514
EXIT       120#
GETPA      231#
GWBUFF     219#
HRDER      128#
IOMOD       91#
IOMODP     115#
IOMODR     111#
IOMODX     107#
MAP22      235#
MODULE       8#    514
MSG        154#
MSGN       158#
MSGS       162#
NBKMOD     103#
OTOA       190#
PIRQ       179#
RAND       124#
SBKMOD      99#    514
SOFER      144#


. ABS.  000000      000
        001636      001

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 CXRHAA,CXRHAA.LST/CRF=DDXCOM.C6,CXRHAA.SRC
 RUN-TIME: 3 4 .7 SECONDS
 RUN-TIME RATIO: 25/8=3.0
 CORE USED:  7K  (13 PAGES)
```